Name:	**Metrius Bin Benedict Sunggoh**
Matric No:	**WEK000429**
Title:	**SYSTEM LOGIN AUTHENTICATION USING VOICE RECOGNITION AND AI METHODS**
Supervised by:	**Mr. Woo Chaw Seng**

Abstract

This thesis will show how the voice recognition technique can be combined with template matching technique (Vector Quantization) to validate a login process. This system will be divided into two modules; Enrollment and Verification. During the Enrollment process, the person offers a 'live sample' of his/her voice. The feature of this voice will be extracted using MFCC technique which involves Frame Blocking, Windowing, Fast Fourier Transform, Mel-frequency Wrapping and Cepstrum. The feature extracted by the MFCC will be processed and stored as codebooks. A codebook is created for each user. To confirm the identity at a future time, the individual will present the 'live sample', which will be matched against the stored template (codebook). Vector Quantization technique will be used to compare the codebook and the feature from the voice provided by the user during login session.

Acknowledgement

I would like to thank everyone who had helped me a lot in the process of writing this thesis, especially my supervisor Mr. Woo Chaw Seng for the guidance given during the weekly meetings.

I also would like to thank my family members and friends, especially Shamsul and Zati Hakim for the idea, support and encouragement.

Without you all, the completion of this thesis is impossible.

Table of Contents

Abstract ii
Acknowledgement iii
Table of Contents iv
List of Tables vi
List of Figures vii
1.1 Introduction 1
1.2 Objective 3
1.3 Project Scope 4
1.4 Project Timeline 5
CHAPTER 2: LITERATURE REVIEW 6
2.1 Principles of Speaker Recognition 6
2.1.1 The Structure of Speaker Identification and Verification Systems 8
2.2 Varieties of Voice Verification System 10
2.3 Programming Language Supported in Existing Product 10
2.4 Summary 10
CHAPTER 3: Methodology 11
3.0 Development method 11
3.1 Explanation of waterfall method 11
3.1.1 System/information engineering and modeling 13
3.1.2 Software requirement analysis 13
3.1.3 Design 13
3.1.4 Code generation 13
3.1.5 Testing 13
3.2 Information Gathering 14
3.2.1 Internet 14
3.2.2 FSKTM Document Room 15
3.2.3 Reference Books 15
CHAPTER 4: SYSTEM ANALYSIS 16
4.1 Non-functional Requirements 16
4.1.1 User Interface 16
4.1.2 Response Time 16
4.2 Functional Requirements 16
4.2.1 Low EER 16
CHAPTER 5: SYSTEM DESIGN 17
4.1 Speaker verification System Design 17
4.2 Feature Extraction 18
4.2.1 Frame Blocking 19
2.3.1 Windowing 19
2.3.2 Fast Fourier Transform (FFT) 20
2.3.3 Mel-frequency Wrapping 20
2.3.1 Cepstrum 21

2.3 Summary 22
CHAPTER 6: SYSTEM IMPLEMENTATION 23
6.1 FUNCTIONAL DETAILS 23
6.1.1 Signal Acquisition 23
6.1.2 Signal Preprocessing 23
6.1.3 MFCC 24
6.1.4 Vector Quantization 24
6.1.5 Threshold Creation 25
6.2 DECISION 25
6.3 SUMMARY 25
CHAPTER 7: SYSTEM TESTING 26
7.1 Units testing 26
7.2 Integration testing 26
7.3 System testing 26
7.4 Summary 29
7.5 Discussion 29
APPENDIX A: GUI & USER MANUAL 30
Appendix B: Hierarchy of the Files called in the Enrollment module 34
APPENDIX C: Hierarchy of the function called in Verification module 35

List of Tables

Table 1.2: Project Timeline.. 5

Table 7.1: Testing Result……………………………………………………………………26

List of Figures

Figure 1.2: Project Timeline..5
Figure 2.1a: Text-independent system..7
Figure 2.1b: Text Dependant System..7
Figure 3.1: Waterfall Model..12
Figure 5.1: Template-based speaker verification system stages..18
Figure 5.2: MFCC Process..19
Figure 5.3: Mel-spaced Filterbank..21
Figure 7.1: 2-D Plot for Two Speakers..28

1.1 Introduction

The amount of cyber crime has been arising in most organization and expected to rise. The significant factor behind this rise is the increasing involvement of technology, in particular computers as an integrated element of the business process. The growth of the Internet has made security as a major concern. In this thesis, I will demonstrate how voice recognition can be used along with AI methods to authenticate the user in a system login process.

Authentication is the process of affirming or rejecting a claimed identity by matching a live template to an enrollment template. This simply means matching information from the person trying to gain admission to a stored collection of allowed. The highest level of security uses 3-factor authentication:

i. Password (including Personal Identification Number - PIN)

ii. Physical devices such as smart cart, key or security token

iii. Biometrics – automated methods of identifying the identification of a person based on the unique physical or behavioral characteristics, such as fingerprints, iris scans, face recognition and voiceprints. All these techniques share a methodology involving enrollment and verification.

    At Enrollment, the person offers a 'live sample' of his/her biometric. This sample will be processed and stored electronically. To confirm the identity at a future time, the individual will present the 'live sample', which will be matched against a stored template.

The biometric technique that will be used in this thesis is voice recognition. This technique will involve speaker verification process, the process of determining which registered speaker provides a given utterance or 'voiceprint'.

1.2 Objective

The main objective of this project is to develop an application that will help in increasing the security level of a system. The conventional technique only requires a user to provide their user name and password for login purpose. This is called a one-level security. With this application, the level of security can be increased to 2-level security by requiring the user to provide their voice as their biometric information.

The goal of this project is to design a real-time speaker verification system which only gives access to a limited number of persons. The user is instructed to type his or her user name and speak it into the microphone. The system analyzes the voice and then either rejects or accepts the user.

1.3 Project Scope

1. This application will perform '1:1' search to identify a user by comparing their voice, as a 'live sample', with the stored template. In '1:1' search, the user will present a user name or PIN along with their live sample, their voice. The system will check the 'live sample only against samples stored under the user name or PIN provided.
2. This project will only investigate how voice recognition technique can be used along with Pattern Recognition (Feature matching), to enhance the security of a system.
3. The interface of this system will be simple and easy to understand.

1.4 Project Timeline

Activity	Duration	Start	End
System study	16 days	15/03/2003	30/03/2003
Requirement analysis	31 days	20/03/2003	30/04/2003
Design	45 days	15/4/2003	30/05/2003
Coding	45 days	01/06/2003	15/07/2003
Testing	45ddays	01/7/2003	15/08/2003
Documentation	61 days	01/08/2003	30/09/2003

Table 1.2: Project Timeline

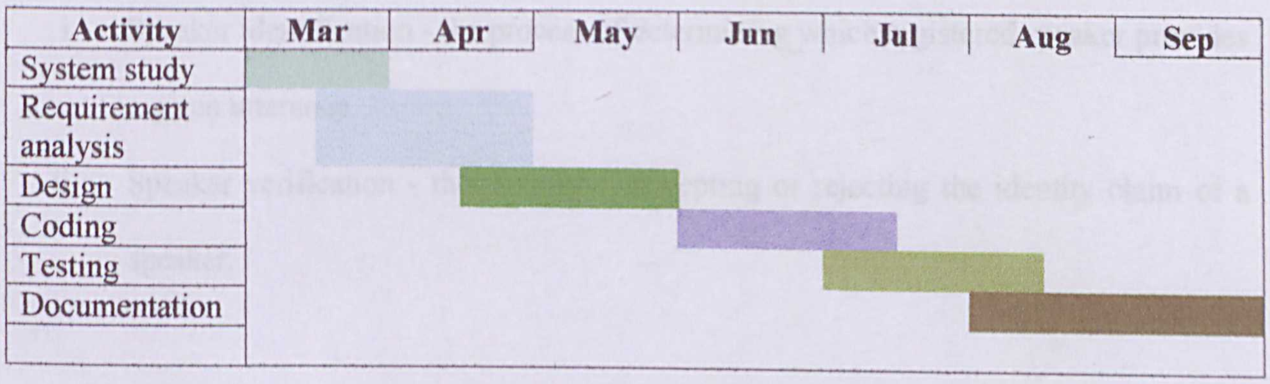


Figure 1.2: Project Timeline

CHAPTER 2: LITERATURE REVIEW

In the process of developing this system, researches have been done to understand various concepts. This process is very important for me to understand the strengths and limitations of several development tools and techniques available.

2.1 Principles of Speaker Recognition

Speaker recognition can be classified into:

i. Speaker identification - the process of determining which registered speaker provides a given utterance.

ii. Speaker verification - the process of accepting or rejecting the identity claim of a speaker.

Speaker recognition methods can also be divided into:

i. Text-independent system - a speaker models the capture characteristics of his speech which show up irrespective of what one is saying, i.e. spectral feature. One of the most successful text-independent recognition methods is based on vector quantization (VQ). VQ codebooks consisting of the condensed representative feature vectors are used as an efficient means of characterizing speaker-specific features. A speaker-specific codebook is generated by clustering the training feature vectors of each speaker. In the recognition stage, an input utterance is vector-quantized using the codebook of each reference speaker and the VQ distortion accumulated over the entire input utterance is used to make the recognition decision.

ii. In a text-dependent system, a speaker's identity recognition is based on his pronouncing one or more specific phrases, like a password used in this project.

The figures below show the basic structures of speaker identification and verification systems.

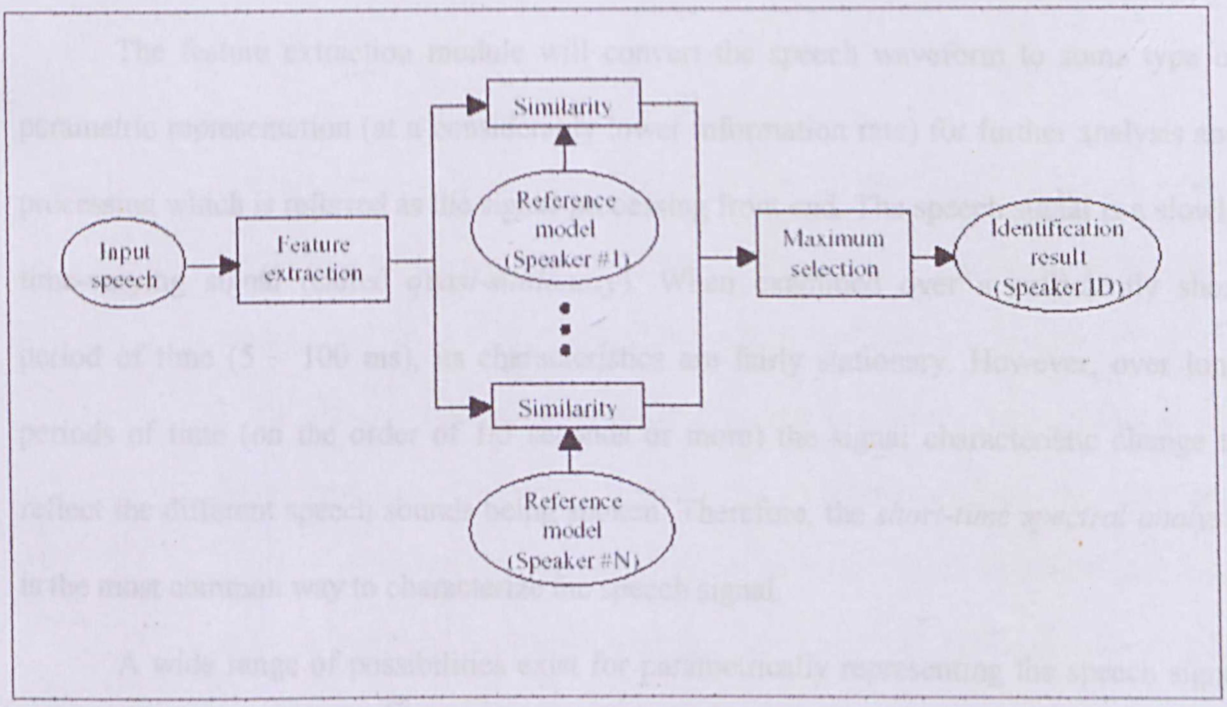

Input speech
Feature extraction
Similarity
Reference model (Speaker #1)
Similarity
Reference model (Speaker #N)
Maximum selection
Identification result (Speaker ID)

Figure 2.1a: Text-independent system

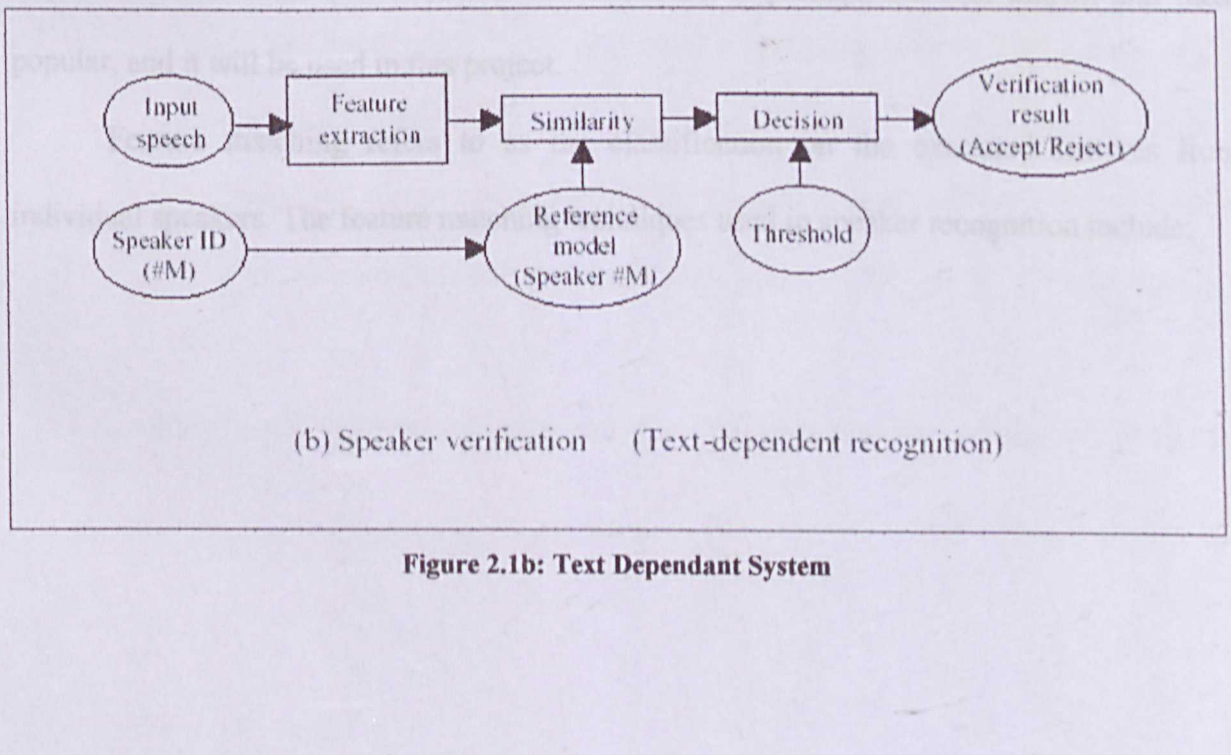

Input speech
Feature extraction
Similarity
Decision
Verification result (Accept/Reject)
Speaker ID (#M)
Reference model (Speaker #M)
Threshold
(b) Speaker verification (Text-dependent recognition)

Figure 2.1b: Text Dependant System

2.1.1 The Structure of Speaker Identification and Verification Systems

In the enrollment phase, each registered speaker has to provide speech samples so that the system can build a reference model for that speaker. During the testing phase, the input speech is matched with stored reference model and recognition decision is made.

The feature extraction module will convert the speech waveform to some type of parametric representation (at a considerably lower information rate) for further analysis and processing which is referred as the signal-processing front end. The speech signal is a slowly time-varying signal (called *quasi-stationary*). When examined over a sufficiently short period of time (5 ~ 100 ms), its characteristics are fairly stationary. However, over long periods of time (on the order of 1/5 seconds or more) the signal characteristic change to reflect the different speech sounds being spoken. Therefore, the *short-time spectral analysis* is the most common way to characterize the speech signal.

A wide range of possibilities exist for parametrically representing the speech signal for the speaker recognition task, such as Linear Prediction Coding (LPC), Mel-Frequency Cepstrum Coefficients (MFCC), and others. MFCC is perhaps the best known and most popular, and it will be used in this project.

Feature matching refers to as the classification on the extracted features from individual speakers. The feature matching techniques used in speaker recognition include:

i. Dynamic Time Warping (DTW) - uses a template matching approach and provides distance measure between the features obtained during training and testing. The DTW method is simple in concept and implementation, however, tends to have a higher error rate than some of the more modern modeling approaches.

i. Hidden Markov Modeling (HMM) - one of the most popular modeling methods used in speaker recognition, the HMM is based on a statistical modeling of the different sounds that a user may have in their password. The HMM models not only the probability that determines how similar a new sound may be to its model, but also the probability of how one sound may make a transition to the next sound in sequence. The HMM has been evaluated extensively for speech recognition and has also more recently been applied to speaker recognition.

ii. Vector Quantization (VQ) - a process of mapping 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 the centroid called codeword. The collection of all codewords consists of the corresponding codebook for a known speaker.

In this project, VQ will be used due to ease of implementation and high accuracy.

2.2 Varieties of Voice Verification System

1. Nuance Verifier
2. Speaker Verification – Developed by Nortel Networks, can be integrated with Nuance Verifier
3. Voice Security System – Developed by Voice Security Inc, California, USA

2.3 Programming Language Supported in Existing Product

1. Voice XML
2. C++
3. Java

2.4 Summary

The study done during this literature review had helped a lot in building this system. The next chapter will explain the methodology used in this project.

CHAPTER 3: Methodology

3.0 Development method

There are numbers of methodologies available for a software engineering process. They are:

- Waterfall Model
- V-Model
- Prototyping Model
- Rapid Application Development Model
- Incremental Model
- Spiral Model
- WINWIN Spiral Model
- Concurrent Development Model

For this thesis, I have chosen to use the Waterfall Modeling process

3.1 Explanation of waterfall method

The development cycle of this cycle is based on waterfall modeling. Waterfall modeling is the oldest but widely used paradigm in a software engineering. This model suggested a systematic, sequential approach to software development that begins at the system level and progress through analysis, design, coding, testing and support. The steps involved in this modeling are:

1. System/information engineering and modeling

2. Software requirement analysis
3. Design
4. Code generation
5. Testing
6. Support

Figure 3.1 illustrates the waterfall model for software engineering.

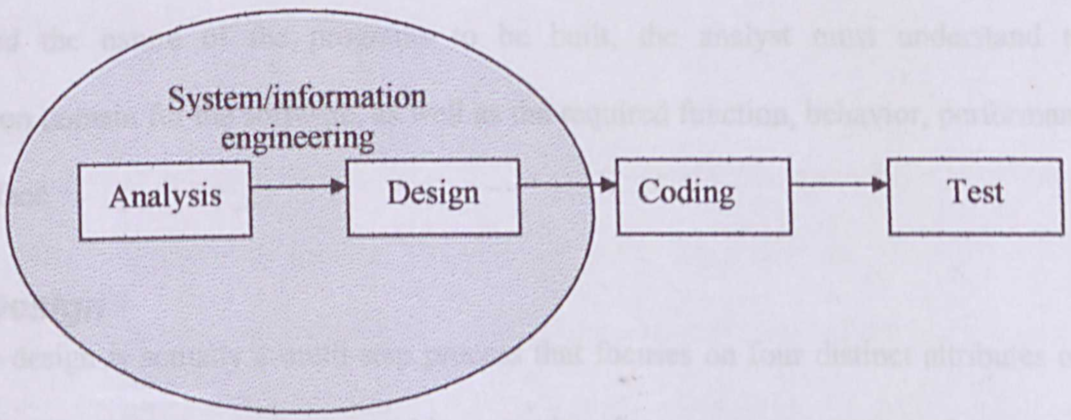


Figure 3.1: Waterfall Model

3.1.1 *System/information engineering and modeling*

Because software is always part of a larger system, work begins by establishing requirements for all systems elements and then allocating some subset of these requirements to software. This system view is essential when software must interact with other elements such as hardware, people and databases. System engineering and analysis encompass requirements gathering at the system level with a small amount of top level design and analysis. Information engineering encompass requirements gathering at the strategic business level and business area.

3.1.2 Software requirement analysis

The requirements gathering process is intensified and focused specifically on software. To understand the nature of the programs to be built, the analyst must understand the information domain for the software, as well as the required function, behavior, performance and interface.

3.1.3 Design

Software design is actually a multi-step process that focuses on four distinct attributes of a program: data structures, software architecture, interface representations, and procedural (algorithmic) detail. The design process translates the requirements into representation of the software that can be assessed for quality before coding begins.

3.1.4 Code generation

The design must be translated into a machine readable form.

3.1.5 Testing

Once code has been generated, program testing has to be done. The testing process focuses on the logical internal of the software, ensuring that all statements have been tested, and on the functional externals; that is, conducting tests to uncover error and ensure that defined input will produce actual results that agree with required results.

3.2 Information Gathering

As been described in the Waterfall Model, the first step in a software engineering is the software requirement analysis so that I can understand the nature of the programs to be built as well as the required functions. There are main sources for my information gathering purpose:

1. Internet
2. FSKTM Document Room
3. Reference Books

3.2.1 Internet

I use the internet to search for the information regarding the voice recognition technology, and how it had been used to develop a system for login authentication. There are a lot of websites set up by companies to promote their existing voice recognition products. There are also websites set up by educational institution, where the research in this field is conducted. This information is very important for me to understand the requirements of the system that is going to be developed.

Internet also plays an important role during the coding stage because examples of codes are available on the Internet. Forums and discussion rooms on the Internet are also a good place to visit when there are problems during the coding stage.

3.2.2 FSKTM Document Room

FSKTM Document room also played a major role during my research. This place provided me with the thesis from the previous years. The information from these theses had helped me to understand the development methodology and the proper way to document my work. Books related to computer science also available here.

3.2.3 Reference Books

When it comes to coding stage, reference books will be used extensively as a reference material.

CHAPTER 4: SYSTEM ANALYSIS

4.1 Non-functional Requirements

4.1.1 *User Interface*

The user interface for this system must be simple because login page is the first page that a user will see before they can use a system. The graphic or command used for the interface must be easy to understand.

4.1.2 *Response Time*

Speaker verification process is quite complex and might take sometime to complete. For this system, I will try to make it to be less than 10 seconds.

4.2 Functional Requirements

4.2.1 *Low EER*

The EER corresponds to the point where the false accept and false reject errors are equal. False reject occurred when the system reject a true user, and false accept occurred when the system accept an impostor. For example, a system that has a false accept rate of 5% and a false reject rate of 5% has an EER of 5%. For this system, my target EER is less than 10%.

CHAPTER 5: SYSTEM DESIGN

This chapter will cover the design procedures of this system. The design is based on feature matching with Vector Quantification (VQ) technique for text-dependent speaker verification.

4.1 Speaker verification System Design

A typical template-based, fixed text speaker verification task consists of:

1. Enrollment: Creation of a set of speech features, as a function of time, for each valid user. Each registered speaker has to provide speech samples so that the system can build a reference model for that speaker.
2. Verification: Comparison of input speech with reference templates at equivalent points in time. Decision will be based on similarity between input and reference.
3. Reference Update: Adaptation of the reference templates to accommodate changes in the valid user's speech after successful verification.

The flow chart for this task is shown in Figure 5.1.

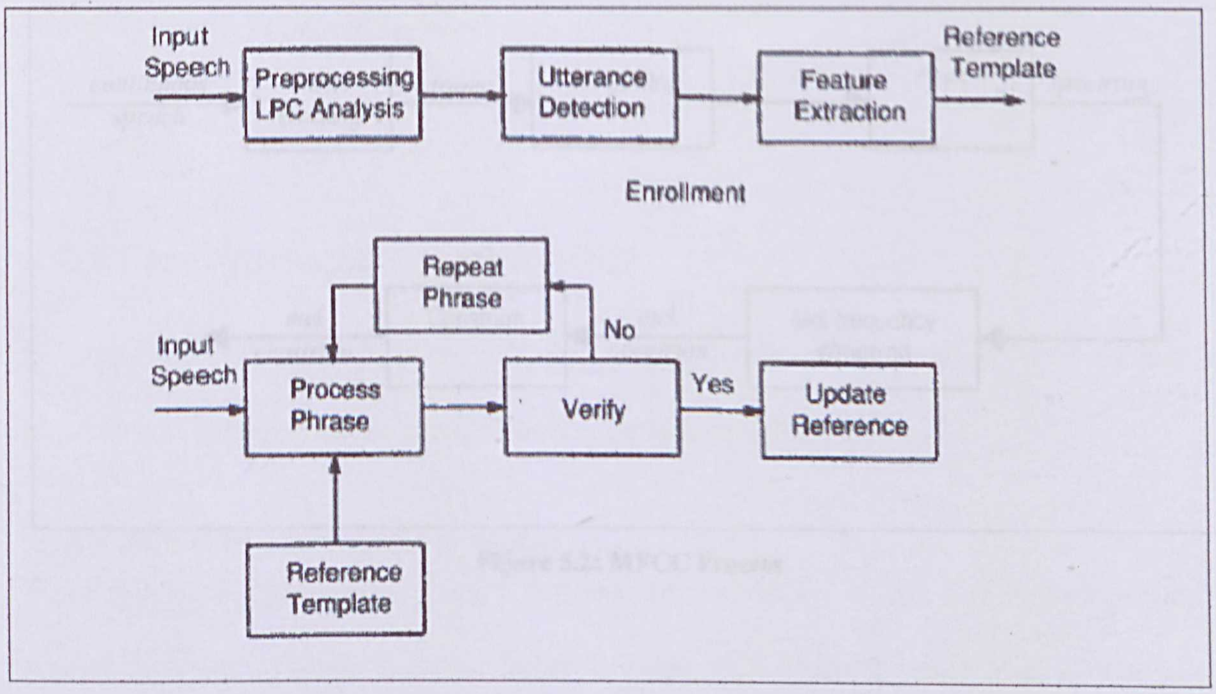


Figure 5.1: Template-based speaker verification system stages

4.2 Feature Extraction

As been discussed in Chapter 2 (literature review), the speech input will be analyzed for feature extraction. This is done at the early stage of the speaker verification process. For this purpose, I will to use Mel-Frequency Cepstrum Coefficients (MFCC) technique. The main purpose of MFCC is to mimic the behavior of human ears.

The stages involved in MFCC feature extraction process is shown in Figure 5.2.

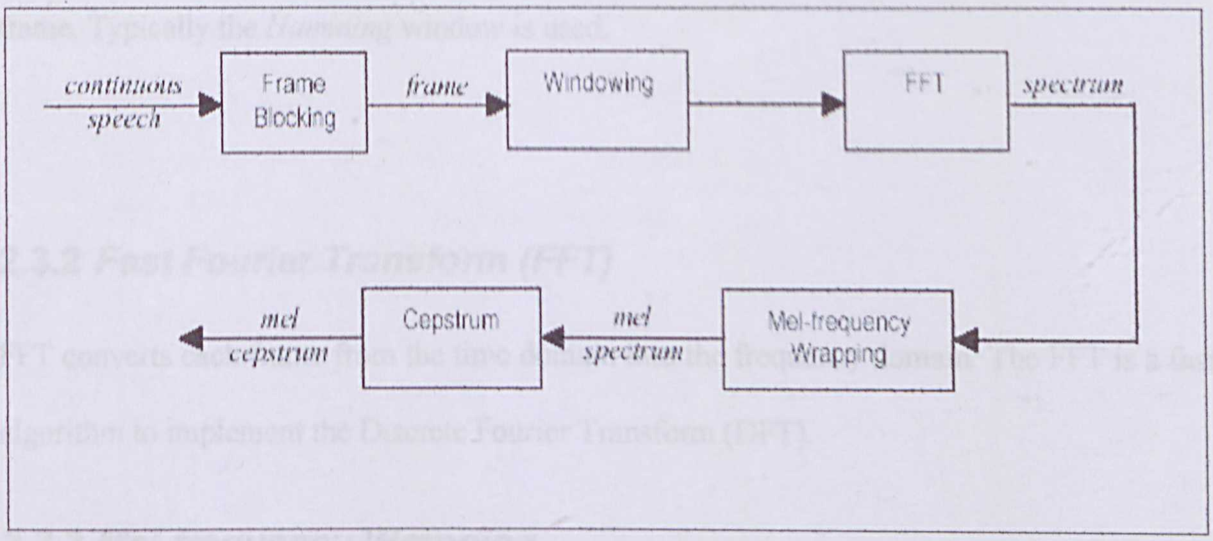


Figure 5.2: MFCC Process

4.2.1 *Frame Blocking*

The continuous speech signal is blocked into frames of N samples, with adjacent frames being separated by M ($M < N$). The 1st frame consists of the first N samples. The 2nd frame begins M samples after the 1st frame, and overlaps it by $N - M$ *samples.* Similarly, the 3rd frame begins 2M samples after the 1st frame (or M samples after the 2nd 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 ~ 30ms windowing and facilitate the fast radix -2 FFT) and M = 100.

2.3.1 *Windowing*

The next step 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. Typically the *Hamming* window is used.

2.3.2 *Fast Fourier Transform (FFT)*

FFT converts each frame from the time domain into the frequency domain. The FFT is a fast algorithm to implement the Discrete Fourier Transform (DFT).

2.3.3 *Mel-frequency Wrapping*

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 1 KHz and a logarithmic spacing above 1 KHz. As a reference point, the pitch of a 1 kHz tone, 40 dB above the perceptual hearing threshold, is defined as 1000 mels.

One approach to simulating the subjective spectrum is to use a filter bank, spaced uniformly on the mel scale (Figure 5.3). 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 20.

Note that this filter bank is applied in the frequency domain; therefore it simply amounts to taking those triangle-shape windows in the Figure 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.

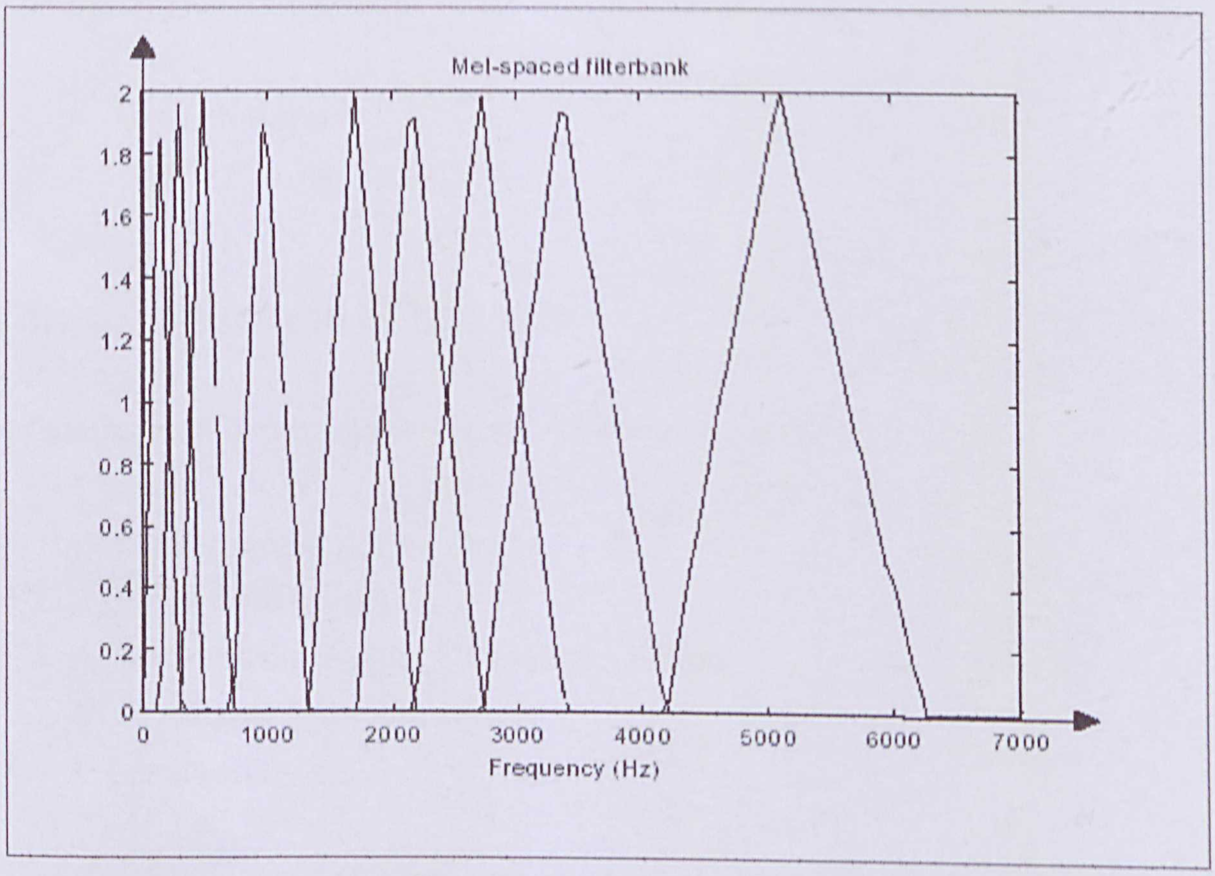


Figure 5.3: Mel-spaced Filterbank

2.3.1 *Cepstrum*

Finally 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, we can convert them to the time domain using the Discrete Cosine Transform (DCT).

2.3 Summary

According to this waterfall modeling, the next stage in building this system is coding.

CHAPTER 6: SYSTEM IMPLEMENTATION

The system implementation phase of this project was done using Matlab. This system is divided into two main modules:

1. User Enrollment
2. User Verification

6.1 FUNCTIONAL DETAILS

There are three major stages involves in this phase:

1. Signal Acquisition
2. Signal Preprocessing
3. MFCC (Mel-Frequency Cepstrum Coefficients)
4. VQ (Vector Quantization)
5. Threshold Creation

6.1.1 Signal Acquisition

In order to use the system, a user needs to provide his/her voice into the system. This task is done by the function `init_sound`. The voice captured by the User Enrollment module will is used to create the user's voice template (***User1st.wav***) and its threshold (***User2nd.wav***). And finally, the Verification Module will compare the voice it captured (***UserVerif.wav***) with the available template for the user; according to the User ID provided by the user. The voice captured for this system is sampled at 16000Hz over 2 seconds.

6.1.2 Signal Preprocessing

Not all parts of the signal captured by the modules will be used by the system. Before the voice can be used in the later MFCC and VQ stages, the Signal Preprocessing stage will make sure that only the voice part of the signal will be use for further processing. The silent part and noises will be separated from the signal. The algorithm for the signal pre-processing can be found in the function ***locatespeech***. The algorithm finds the start and end of speech in a given waveform, allowing the speech to be removed and analyzed. Inside ***locatespeech***, the function ***removed_dc*** will remove 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. Then, the average magnitude and the zero-crossing rate of the input sound will be calculated using the ***avmag*** and ***zerocrossing*** functions respectively.

6.1.3 MFCC

The speech input will be analyzed using MFCC technique. Both modules require the user to provide speech samples. All these samples will go through the MFCC (feature extraction) process. The difference between these two modules is; the feature extracted in the Enrollment Module will be stored as template while the feature extracted in the Verification module will be compared against the stored template. As been shown in Figure 5.2, the steps involved in MFCC are:

1. Frame Blocking
2. Windowing
3. Fast Fourier Transform
4. Mel-frequency Wrapping
5. Cepstrum

6.1.4 Vector Quantization

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

6.1.5 Threshold Creation

A global and static threshold could not be defined. A threshold is needed for each user. In order to fix this, a third pseudo-module was implemented for threshold generation. The function for this task is ***id_threshold***.

6.2 DECISION

The decision in verification is done by calculating the average Euclidean distance between the test vectors and the codeword for that user. This function is done by the ***id_test_verif*** function in the User Verification Module. The average Euclidean distance is then compared to the user's threshold for the pass/fail verdict.

6.3 SUMMARY

The implementation phase involved coding and interface building work. After the interface had been associated with the m-files (callback), the next step is to test the system.

CHAPTER 7: SYSTEM TESTING

Testing phase is carried out to determine whether a program or system performed the desired process. System testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding.

However, testing cannot show the absence of defects. It can only show that the defects are present. Hence, it is very important to keep this in mind while testing is being conducted. There are three steps involved in testing phase:

1. Units testing
2. integration testing
3. system testing

7.1 Units testing

Unit testing is done to test whether the modules are able to carry the desired task. From the test, it was found out that that the enrollment module was successfully executed its task to generate the codebook and the threshold. The

7.2 Integration testing

For this system, the integration testing is done to check whether the sound captured during the Verification Session can be match against the stored template (codebook) generated earlier in the Enrollment Session.

7.3 System testing

The system testing phase is carried out to ensure that the overall system performed to the functionality specifications. A microphone is used for recording the users' voice. For testing purpose, five persons will act as a user and the ability of the system to determine the right user will be tested. These 'users' will be given a unique User ID but the same password: testing. They will have to type their User ID and say their password into the microphone.

During testing, both sessions ran properly. The codebook was generated successfully upon the first recording of the user's voice. The threshold also successfully built upon the second recording of the user's voice.

This system had been tested on 5 users. The recognition result is shown below:

	Norman (M)	Syaril (M)	Patricia (F)	Zati (F)	Shamsul (M)
Norman (M)	/	X	X	X	X
Syaril (M)	/	/	X	X	X
Patricia (F)	X	X	/	X	X
Zati (F)	X	X	X	/	X
Shamsul (M)	X	X	X	X	/

Table 7.1: Testing result

Legend:

(M) – Male

(F) – Female

/ - Verification Success

X - Verification Fail

False Rejection Rate = 0%

False Acceptance rate = 20%

EER = 20%

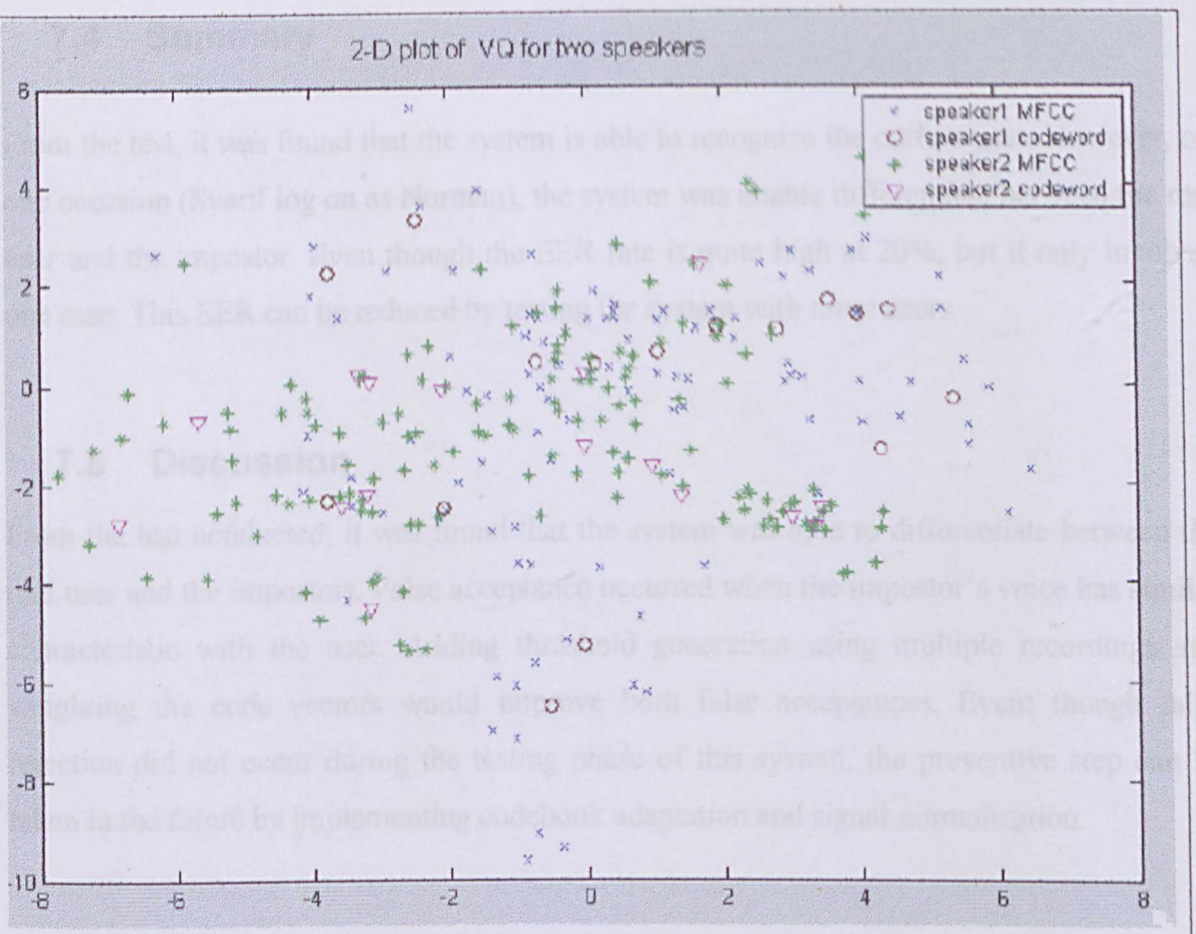


Figure 7.1: 2-D Plot for Two Speakers

Figure 7 shows a 2-D plot of the original trained MFCC vectors by two speakers by selecting any two dimensions (e.g. 5th, 6th) of the 12 coefficient and in comparison a 2-D plot of the codebooks generated by the two speakers using the same two dimensions with a codebook size of 64. Thus we can see that each codeword on the plot exactly represent the corresponding clusters in the original MFCC data points and still accurately represent each speaker's voice characteristics.

In our VQ part, we calculate the Euclidean distance between the unknown word and codebooks, and then the lowest value of the distances is identified as the correct person. For people already trained in database trying to access the system, VQ helps to pinpoint the password associated with the speaker in the database. From the table below, we can see how the Euclidean Distance varies between the 'real' user and the impostor.

7.4 Summary

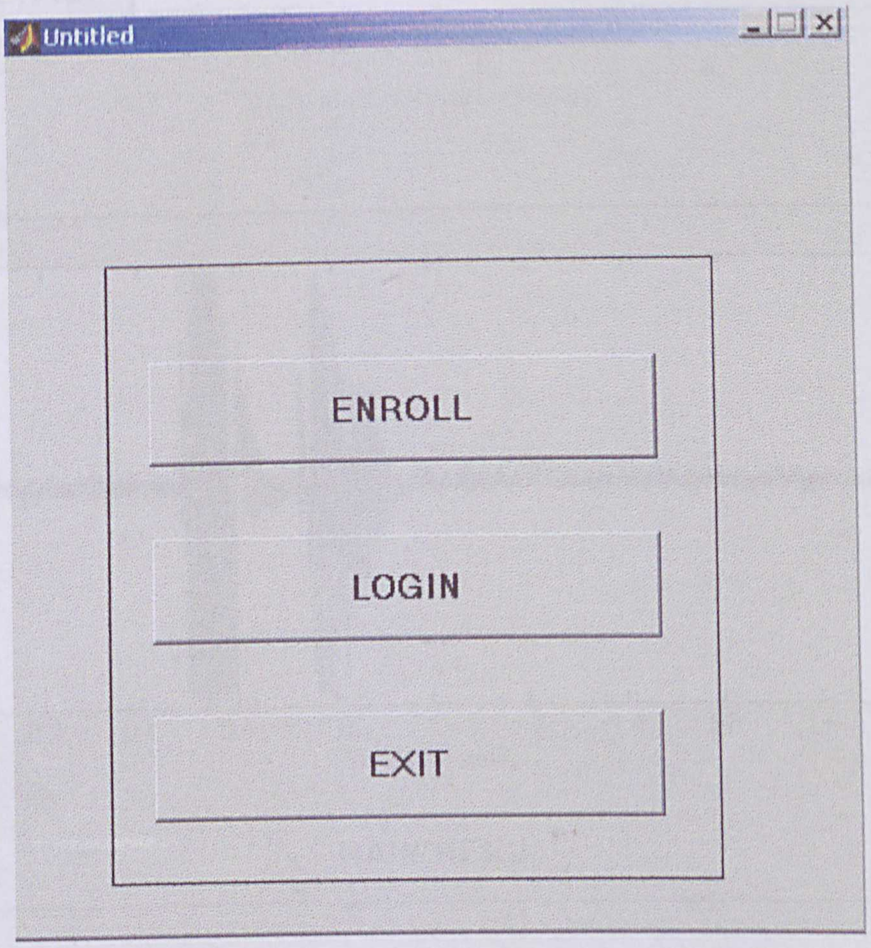
From the test, it was found that the system is able to recognize the correct user. However, on one occasion (Syaril log on as Norman), the system was unable differentiate between the real user and the impostor. Even though the EER rate is quite high at 20%, but it only involves one case. This EER can be reduced by testing the system with more users.

7.5 Discussion

From the test conducted, it was found that the system was able to differentiate between the real user and the impostors. False acceptance occurred when the impostor's voice has similar characteristic with the user. Adding threshold generation using multiple recordings and weighting the code vectors would improve both false acceptances. Event though false rejection did not occur during the testing phase of this system, the preventive step can be taken in the future by implementing codebook adaptation and signal normalization.

APPENDIX A: GUI & USER MANUAL

From Matlab, the main menu for the system can be opened whether by activating the file main.***m*** or activating the figure ***main.fig*** from Matlab's GUIDE (GUI Builder).



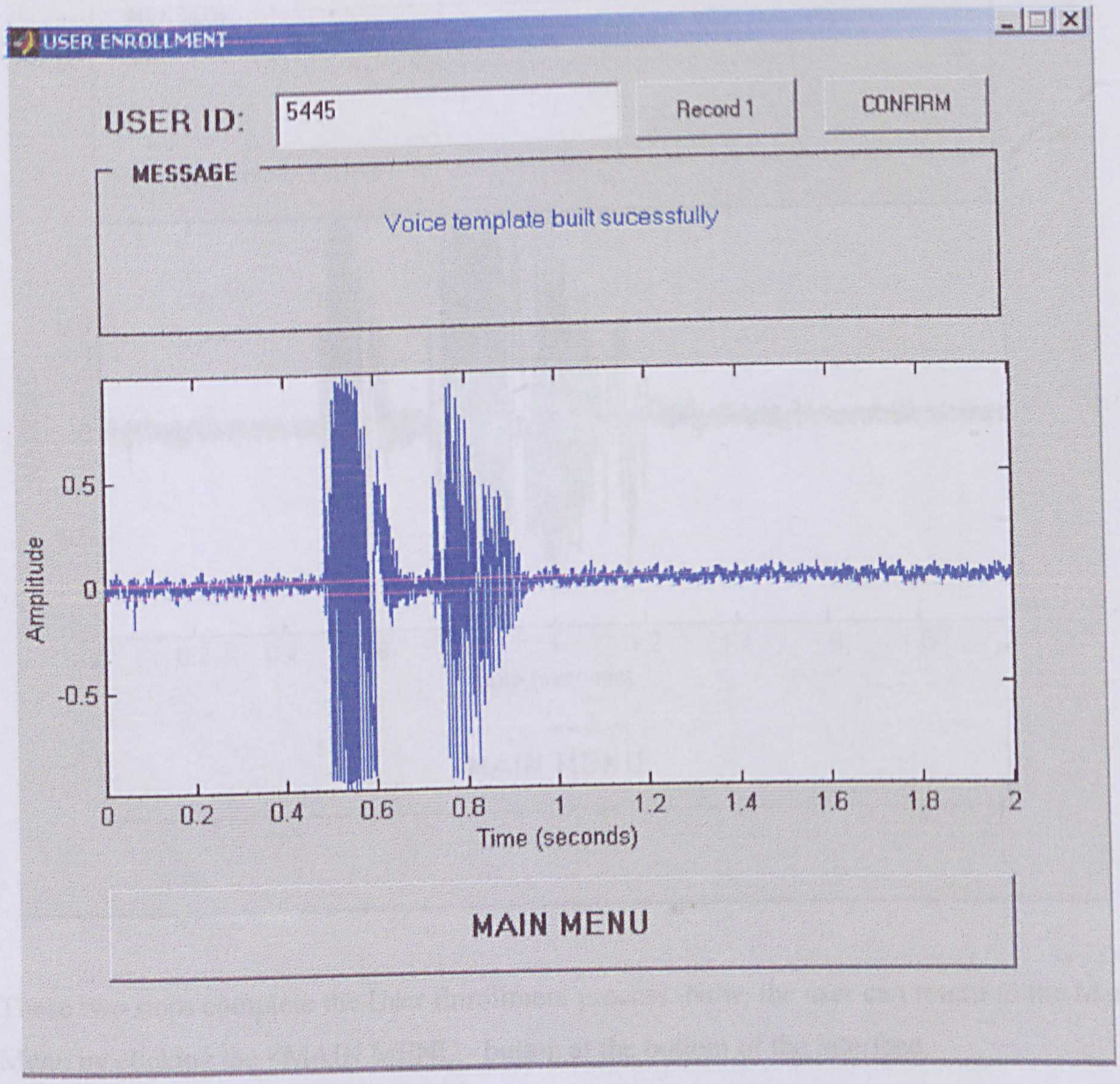
The main menu will show the user the two main modules:

1. Enrollment Module
2. User Verification Module (Login)

Clicking the ***Enroll Button*** will open the Enrollment Module which is divided into two parts:

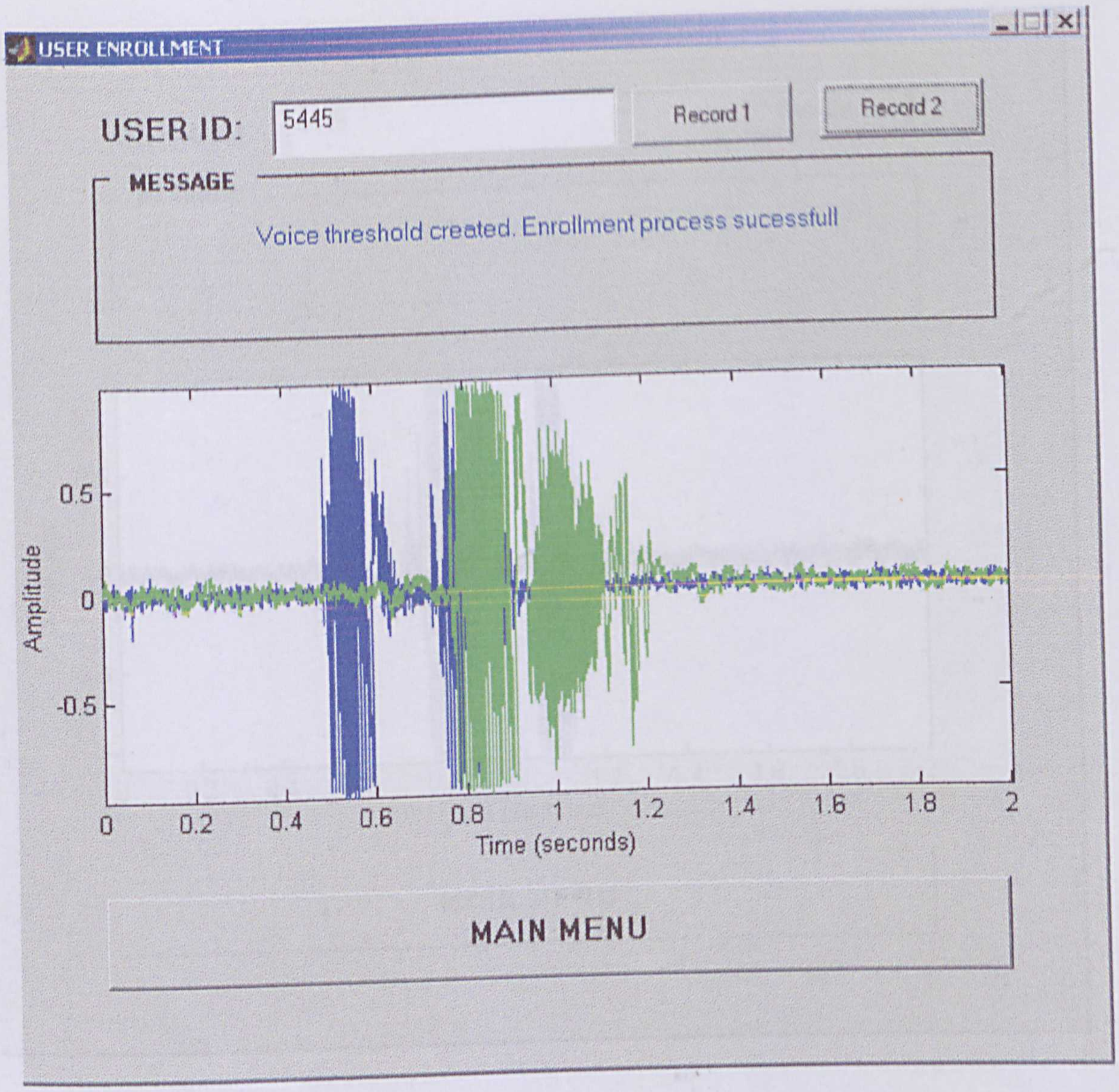
1. Codebook Generation
2. Threshold Generation

A user has to record his/her voice to store his/her voice template, by clicking the <RECORD> button.



Voice template built sucessfully

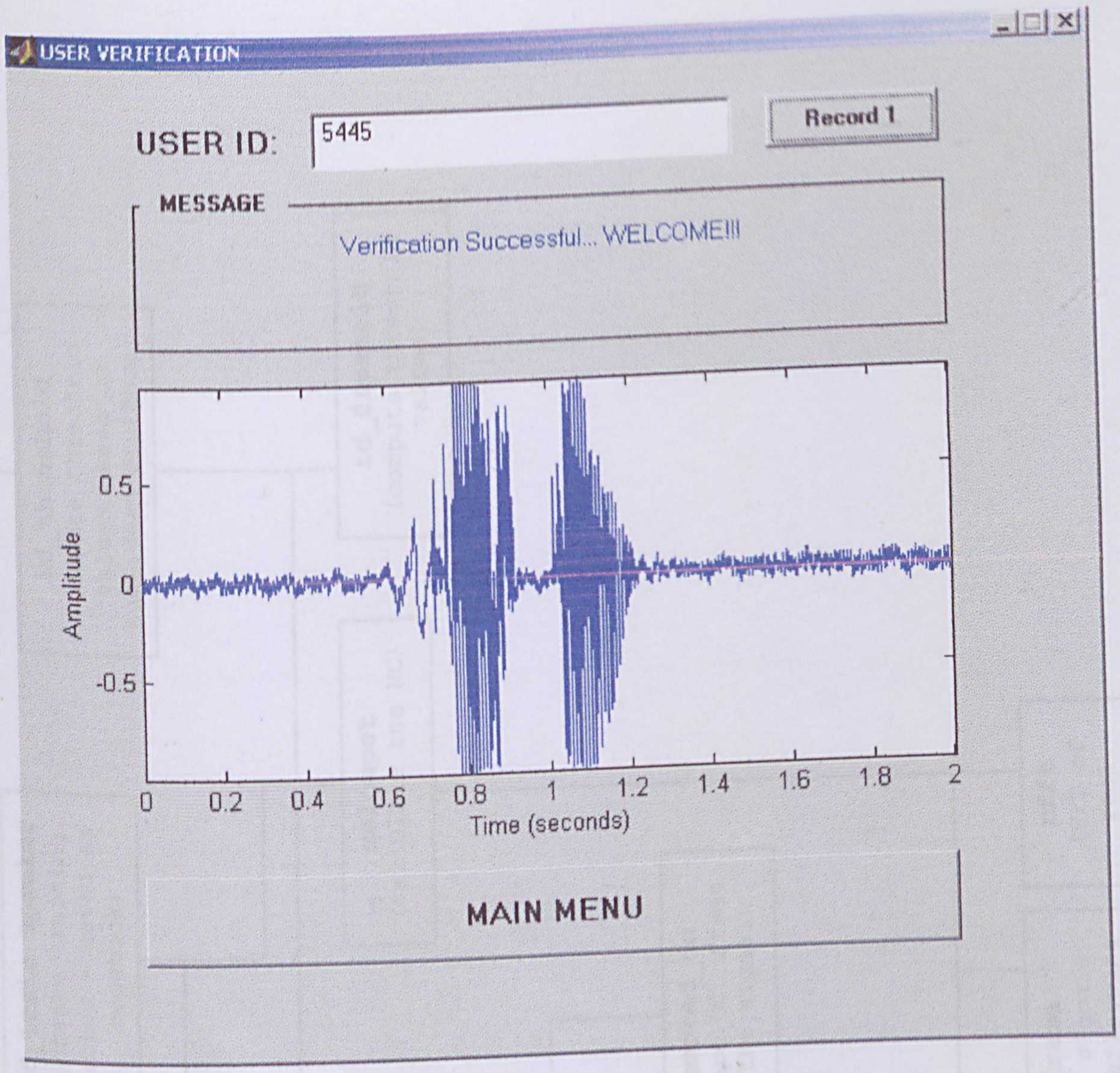
After the voice template has been successfully created, he/she must click the <CONFIRM> button to create the threshold for the stored voice (password).



Voice threshold created. Enrollment process sucessfull

These two steps complete the User Enrollment process. Now, the user can return to the Main Menu by clicking the <MAIN MENU> button at the bottom of the interface.

At the Main Menu, the user who had gone through the Enrollment process can choose whether to proceed to the login process, or exit. If he/she choose to Login, he/she have to click the <LOGIN> button to bring up this interface.

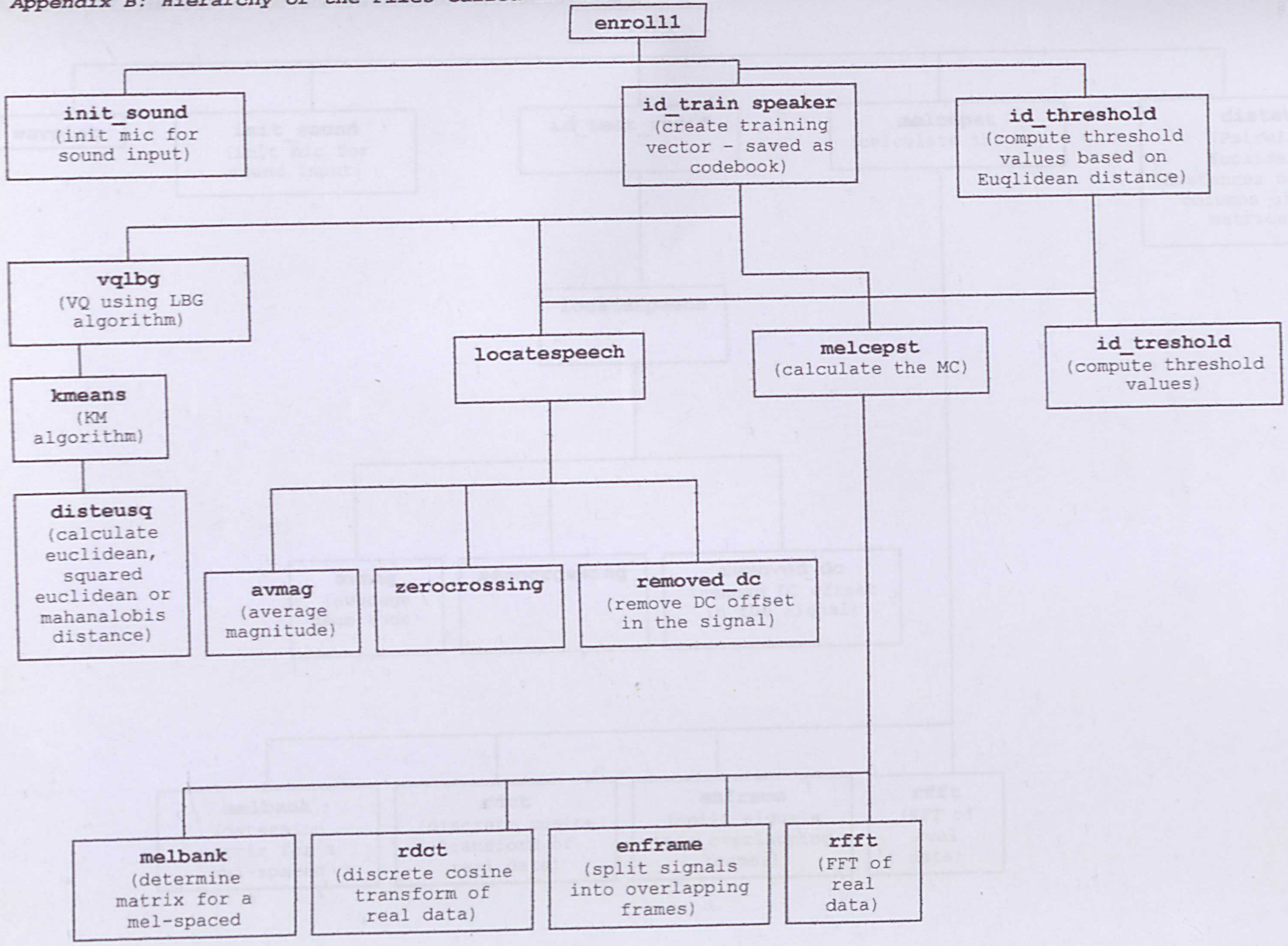


5445

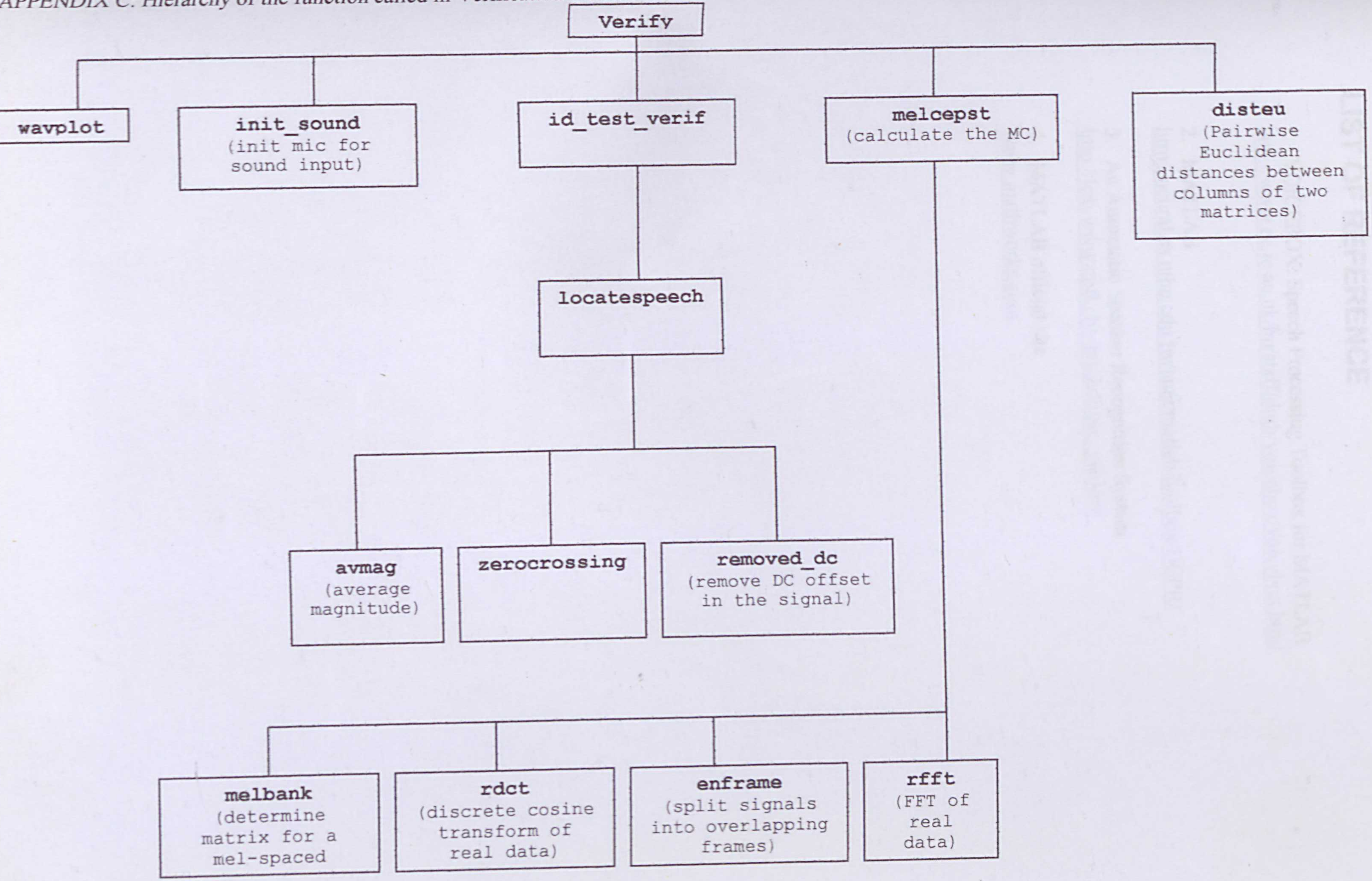
Verification Successful... WELCOME!!!

In order to login, the user is required to type his/her User ID and then click the <LOGIN> button. When the message asking for the password comes out, speak the password into the microphone. Then, the user will be presented with the message whether the login process is successful or fail.

Appendix B: Hierarchy of the Files called in the Enrollment module


enroll1
init_sound
(init mic for sound input)
id_train speaker
(create training vector - saved as codebook)
id_threshold
(compute threshold values based on Euqlidean distance)
vqlbg
(VQ using LBG algorithm)
locatespeech
melcepst
(calculate the MC)
id_treshold
(compute threshold values)
kmeans
(KM algorithm)
disteusq
(calculate euclidean, squared euclidean or mahanalobis distance)
avmag
(average magnitude)
zerocrossing
removed_dc
(remove DC offset in the signal)
melbank
(determine matrix for a mel-spaced
rdct
(discrete cosine transform of real data)
enframe
(split signals into overlapping frames)
rfft
(FFT of real data)

APPENDIX C: Hierarchy of the function called in Verification module


Verify
wavplot
init_sound (init mic for sound input)
id_test_verif
melcepst (calculate the MC)
disteu (Pairwise Euclidean distances between columns of two matrices)
locatespeech
avmag (average magnitude)
zerocrossing
removed_dc (remove DC offset in the signal)
melbank (determine matrix for a mel-spaced
rdct (discrete cosine transform of real data)
enframe (split signals into overlapping frames)
rfft (FFT of real data)

LIST OF REFERENCE

1. VOICEBOX: Speech Processing Toolbox for MATLAB
http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html

2. MATLAB
http://neural.cs.nthu.edu.tw/jang/matlab/toolbox/DCPR/

3. An Automatic Speaker Recognition System
http://lcavwww.epfl.ch/~minhdo/asr_project/

4. MATLAB official site
www.mathworks.com