DIGITAL SPEECH PROCESSING

CMPE 623

TERM PROJECT

A SPEAKER IDENTIFICATION and VERIFICATION SYSTEM

Submitted to : Prof. Dr. Fikret Gürgen

Submitted by : Ömer Ragıp Özkan – 2001700361

Date : 05 / 06 / 2003

1 INTRODUCTION

Speaker recognition is the process of automatically recognizing who is speaking on the basis of individual information included in speech waves. This technique makes it possible to use the speaker's voice to verify their identity and control Access to services such as voice dialing, banking by telephone, telephone shopping, database access services, information services, voice mail, security control for confidential information areas, and remote access to computers.

The goal of this project is to develop a simple *automatic speaker recognition system*. I tested my system on a very small speech database, which based on one speaker training and controlling his/her access to system, this speaker can be authorized by voice and password recognition.

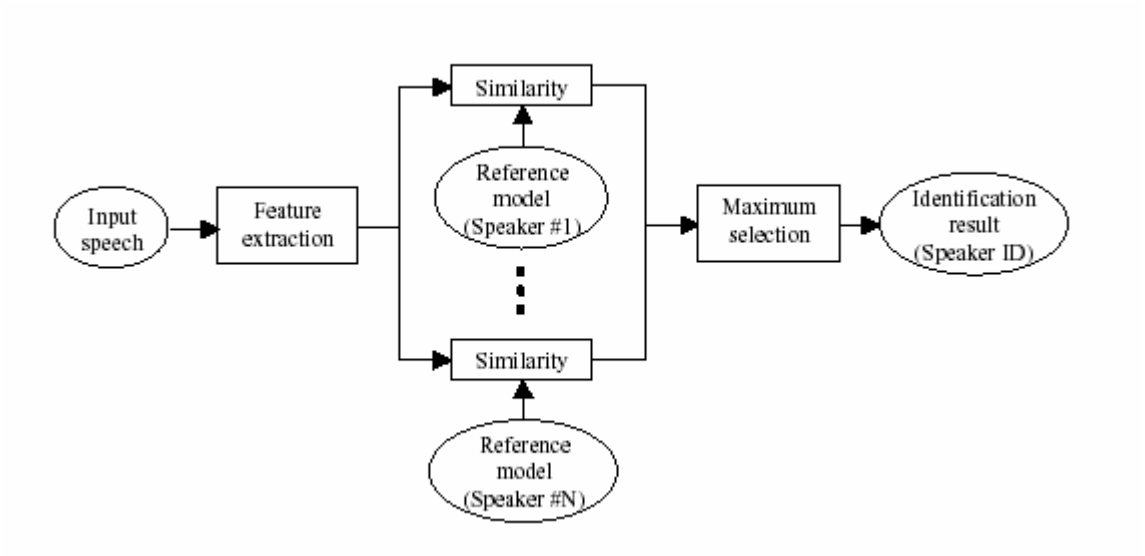
2 PRINCIPLES OF SPEAKER RECOGNITION

Speaker recognition can be classified into identification and verification. *Speaker identification* is the process of determining which registered speaker provides a given utterance. *Speaker verification*, on the other hand, is the process of accepting or rejecting the identity claim of a speaker.

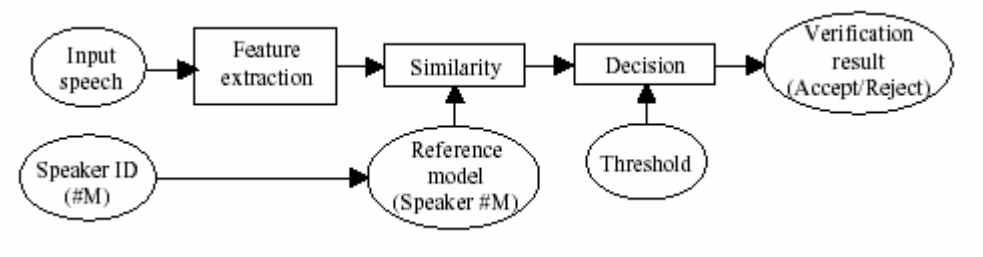
Speaker recognition methods can also be divided into *text-independent* and *textdependent* methods. In a text-independent system, a speaker models the capture characteristics of his speech which show up irrespective of what one is saying, i.e. spectral feature. One of the most successful text-independent 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. In a text-dependent system, on the other hand, a speaker's identity recognition is based on his pronouncing one or more specific phrases, like a password used in this project.

Text-dependent methods are usually based on template-matching techniques -dynamic time warping (DTW) algorithm. The hidden Markov model (HMM) can efficiently model statistical variation in spectral features and have achieved significantly better recognition accuracies than DTW. Therefore, HMM-based method is applied into the word recognition.

Figure 1 shows the basic structures of speaker identification and verification systems. *Feature extraction* is the procedure to extract a small amount of data from the voice signal that can later be used to represent each speaker, the well-known MFCC spectral coefficients are adopted here. *Feature matching* involves identifying an unknown speaker by comparing extracted features from his voice with the ones from speaker database.



(a)



(b)

Figure 1: (a) Speaker Identification (Text - Independent) (b) Speaker verification (Text - Dependent)

In the *training phase*, each registered speaker has to provide speech samples so that the system can build a reference model for that speaker. In case of speaker verification, in addition, a speaker-specific threshold is also set from the training samples. During the testing phase, the input speech is matched with stored reference model and recognition decision is made.

3 SPEECH FEATURE EXTRACTION

The purpose of this module is to convert the speech waveform to some type of parametric representation 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 msecs), its characteristics are fairly stationary. However, over long periods of time (on the order of 200 msecs 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. MFCC is based on the known variation of the human ear's critical bandwidths with frequencies, filters spaced linearly at low frequencies and logarithmically at high frequencies have been used to capture the phonetically important characteristics of speech. This is expressed in the *Mel-frequency* scale, linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz.

3.1 Mel-frequency Cepstrum Coefficients Processor

A block diagram of the structure of an MFCC processor is given in Figure 2. The speech input is typically recorded at a sampling rate above 10 KHz. 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. So, 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 is shown to be less susceptible to mentioned variations.

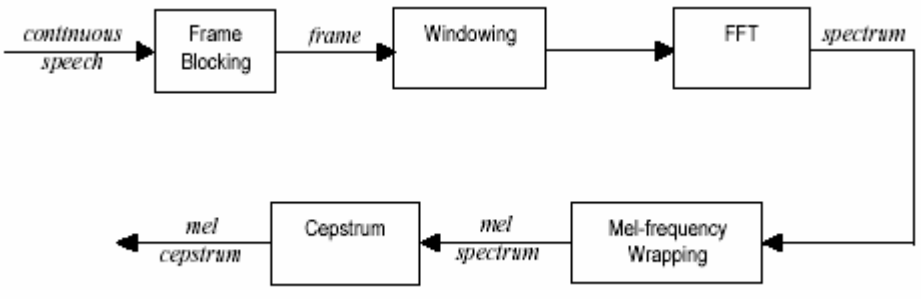


Figure 2: MFCC Processor

3.1.1 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 8 kHz sampled) and $M = 100$.

3.1.2 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. If we define the window as $w(n), \quad 0 \le n \le N-1$ where N is the number of samples in each frame, then the result of windowing is the signal:

$$y(n) = x(n)w(n), \quad 0 \le n \le N-1$$

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

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

3.1.3 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) which is defined on the set of N samples $\{X_n\}$, as follows:

$$X_n = \sum_{k=0}^{N-1} X_k e^{-2pjkn/N}, \qquad \text{n = 0,1,2 …….}$$

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 $1 \le n \le \frac{N}{2} - 1$, while negative frequencies $-f_s/2 < f < 0$ correspond to $\frac{N}{2} + 1 \le n \le N-1$.

3.1.4 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 1KHz and a logarithmic spacing above 1KHz. As a reference point, the pitch of a 1 kHz tone, 40 dB above the perceptual hearing threshold, is defined as 1000 mels. Therefore we can use 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, spaced uniformly on the mel scale (see Figure 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 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 in the frequency domain.

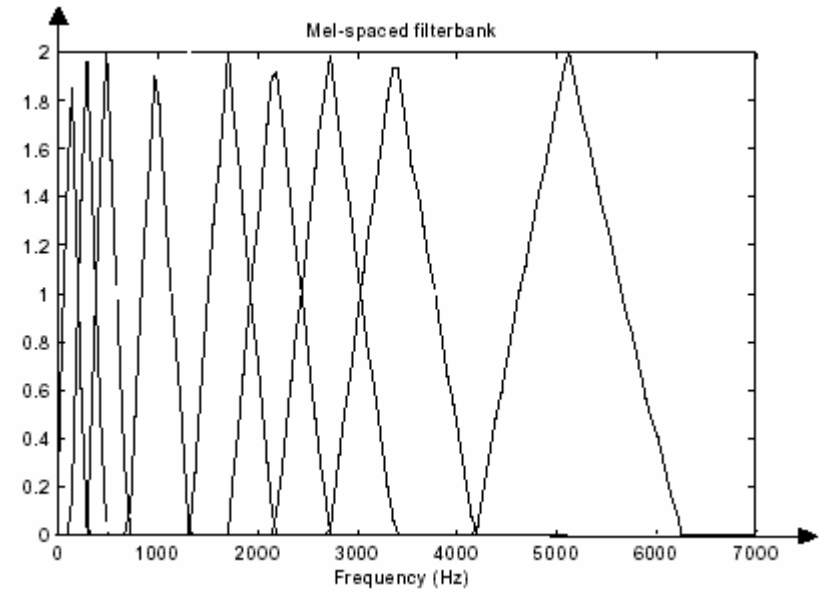


Figure 3: Mel – spaced filterbank

3.1.5 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). Therefore if we denote those mel power spectrum coefficients that are the result of the last step are $\tilde{\tilde{S}}_k, k = 1,2,3,.....K$ we can calculate the MFCC's:

$$\tilde{C}_n = \sum_{k=1}^{K} \left(\log \tilde{S}_k \right) \cos\left[n\left(k - \frac{1}{2} \right) \frac{p}{k} \right], \qquad \text{n = 1, 2, 3…… K}$$

Note that we exclude the first component, $\tilde{C}_0$, from the DCT since it represents the mean value of the input signal which carried little speaker specific information.

4 FEATURE MATCHING

Feature matching refers to as the classification on the extracted features from individual speakers. The feature matching techniques used in speaker recognition include Dynamic Time Warping (DTW), Hidden Markov Modeling (HMM), Gaussian Mixture Models (GMM) and Vector Quantization (VQ).

4.1 Vector Quantization

In this project, since little data is available I did not implement the system with GMM (Gaussian Mixture Models), VQ will be suitable for voice recognition, due to ease of implementation and high accuracy. VQ is 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. Figure 4 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 training acoustic vectors. The result codewords are shown in Fig 4 by black circles and black triangles for speaker 1 and 2, respectively.

The distance from a vector to the closest codeword is called *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.

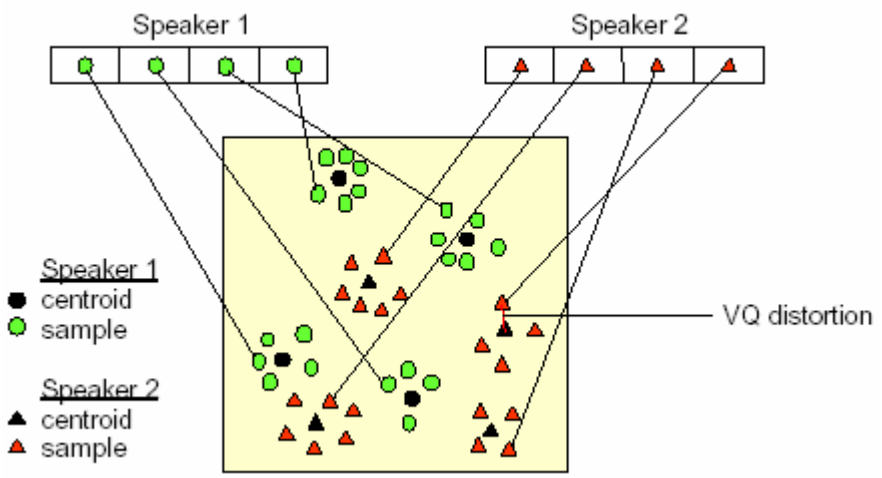


Figure 4: Conceptual diagram illustrating vector quantization codebook formation.

The LBG VQ algorithm we used 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

$$y_n^+ = y_n(1+e)$$
$$y_n^- = y_n(1-e)$$

where n varies from 1 to the current size of the codebook, and e is a splitting parameter (we choose e = 0.01).

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. Figure 5 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.

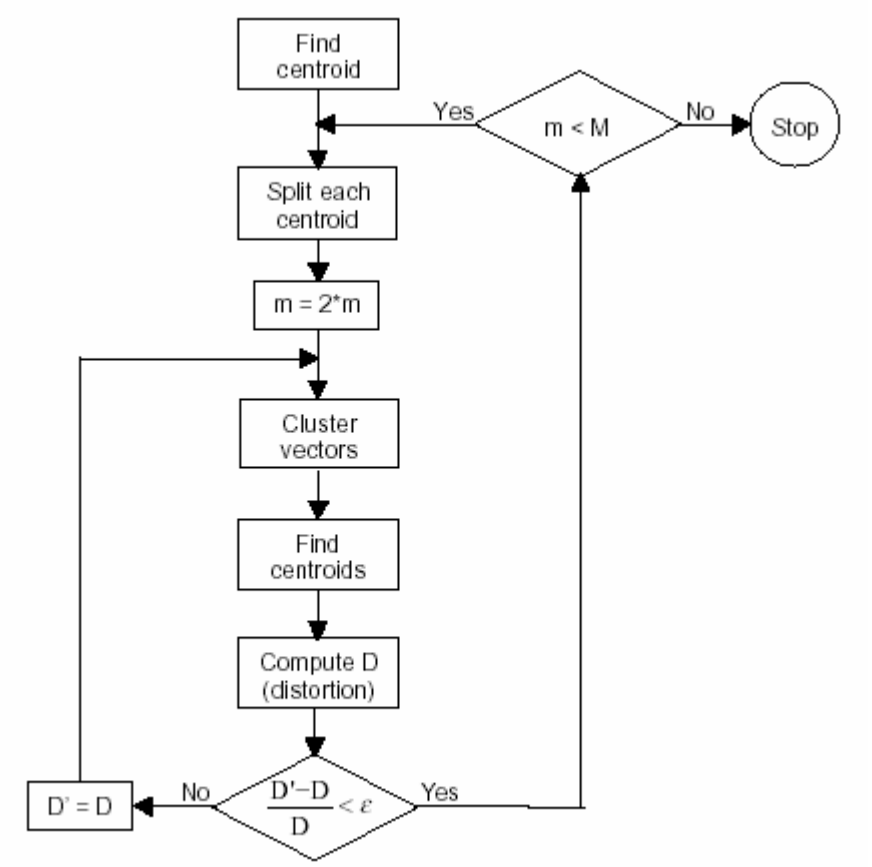


Figure 5: Flow diagram of the LBG algorithm

4.2 Hidden Markov Model (HMM)

For speech signals, a left-right HMM 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 word recognition. 6-state left-right HMM is used for modelling 2-phoneme password.

5 IMPLEMENTATION

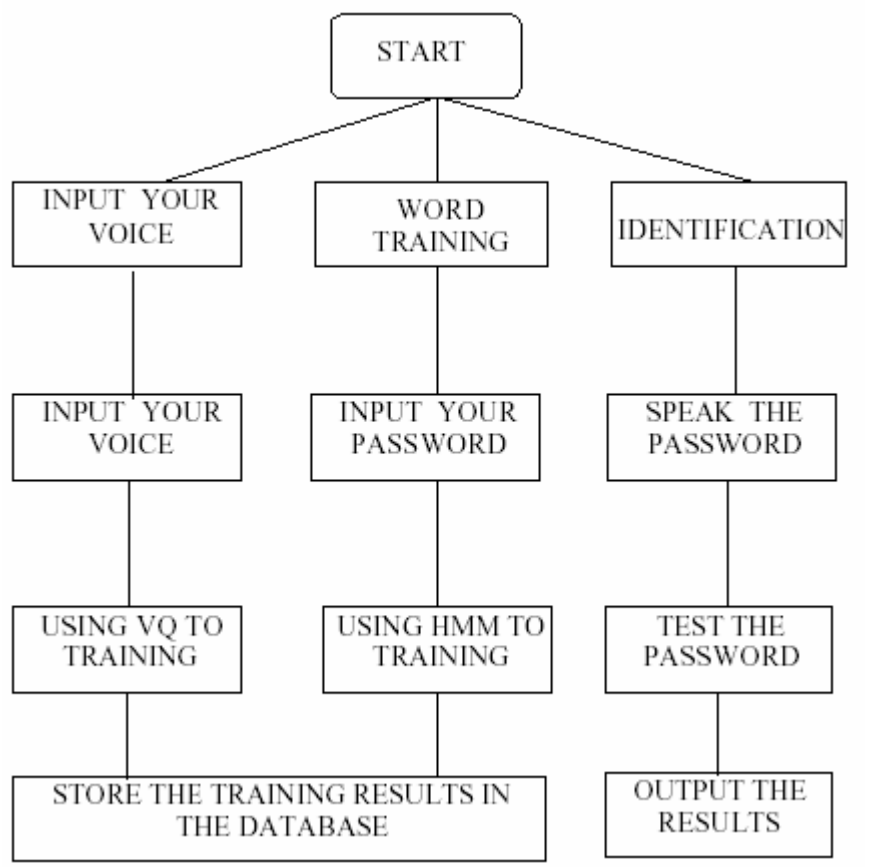


Figure 6: System Implementation

6 RESULTS and CONCLUSION

Based on the simple equipment, I used to record my voice and password for training and testing, my correct rate for the whole identification system is 85% by carefully recording and properly adjusting relevant parameters, such as codebook size, iteration number, etc.

In VQ part, I calculate the Euclidean distance between the unknown word and codebooks, then the lowest value of the distances is identified as the correct person. But on the other hand, it is very hard to set up a threshold value for this distance to distinguish people who try to gain access not in the database. Since this threshold value varies for each word, so if a threshold is set up, either most of the values would be too high or too low. So the limitation of my system is that people who have not been trained in the system can sometimes pass voice recognition realized through VQ algorithm. This limitation has not been expected.

Anyway, that person will not be able to pass the password verification, thus ensuring the security of system. For people already trained in database trying to access the system, VQ helps to pinpoint the password associated with the speaker in the database, and thus improves the security of the whole system and serves an integral part of our security system.

The password verification system is established by use of Malcolm Slaney's **mfcc.m** code. In this part, I assume all the passwords have the similar number of phonemes so that I can make some constant number of states, which is N in my matlab files.

In addition, for word recognition, I need to discard the error password not to find a most likely word. This means I have to set a threshold to discard the wrong password. After a series of tests I determined that the threshold would be 700.

In future works, I am going to implement a system using GMM with large sample databasis. Unfortunately, I did not complete my database for GMM and also I encountered some problems while I was forming the codes for GMM.

About the code that I developed, please, notice that 39 line at the Project.m because referred directory at this line will have to be change according to your own directory.

7 REFERENCES

[1] S. Furui, Digital Speech Processing, Synthesis, and Recognition, 2nd Edition, 2001, New York, USA

[2] S. Furui, "An overview of speaker recognition technology", *ESCA Workshop on Automatic Speaker Recognition, Identification and Verification* , pp. 1-9, 1994.

[3] F.K. Song, A.E. Rosenberg and B.H. Juang, "A vector quantization approach to speaker recognition", *AT&T Technical Journal,* Vol. 66-2, pp. 14-26, March 1987.

[4] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In Proceedings of the IEEE, 77(2), pages 257-285, February 1989