ANALYSIS OF SPEECH RECOGNITION TECHNIQUES

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

Bachelors of Technology
in
Electrical Engineering

By

BIBEK KUMAR PADHY
10502023

SUDHIR KUMAR SAHU
10502035

Department of Electrical Engineering
National Institute of Technology
Rourkela – 769008.
2009

ANALYSIS OF SPEECH RECOGNITION TECHNIQUES

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

Bachelors of Technology
in
Electrical Engineering

By

BIBEK KUMAR PADHY
10502023

SUDHIR KUMAR SAHU
10502035

Under the Guidance of

PROF. DIPTI PATRA

Department of Electrical Engineering
National Institute of Technology
Rourkela – 769008.
2009

National Institute of Technology
Rourkela

CERTIFICATE

This is to certify that the thesis entitled **"ANALYSIS OF SPEECH RECOGNITION TECHNIQUES"** submitted by Sri **Sudhir Kumar Sahu(10502035)** and Sri **Bibek Kumar Padhy(10502023)** in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Electrical Engineering at the National Institute of Technology, Rourkela (Deemed University) during the session 2008-09 is an authentic work carried out by them under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University/Institute for the award of any Degree or Diploma.

DATED – 10.05.2009

(Prof. DIPTI PATRA)

Dept. of ELECTRICAL ENGINEERING

National Institute of Technology

Rourkela-769008

ACKNOWLEDGEMENT

No thesis is created entirely by an individual, many people have helped to create this thesis and each of their contribution has been valuable. Our deepest gratitude goes to our thesis supervisor, **Dr. Dipti Patra**, Professor, Department of Electrical Engineering, National Institute of Technology for introducing the present topic and for her inspiring guidance, constructive criticism and valuable suggestions throughout the project work. Her readiness for consultation at all times, her educative comments, her concern and assistance even with practical things have been invaluable.

We would also like to thank all professors and lecturers, and members of the department of Electrical Engineering, National Institute of Technology for their generous help in various ways for the completion of this thesis. Lastly, we would like to thank and express our gratitude towards our friends who at various stages had lent a helping hand.

Sudhir Kumar Sahu

Roll. No. 10502035

Dept. Of Electrical Engineering

N.I.T.Rourkela

Bibek Kumar Padhy

Roll. No. 10502023

Dept. Of Electrical Engineering

N.I.T.Rourkela

CONTENTS

LIST OF FIGURES

ABSTRACT

Speech recognition has been an intregral part of human life acting as one of the five senses of human body, because of which application developed on the basis of speech recognition has high degree of acceptance. Here in this project we tried to analyse the different steps involved in artificial speech recognition by man-machine interface. The various steps we followed in speech recognition are feature extraction, distance calculation, dynamic time wrapping. We have tried to find out an approach which is both simple and efficient so that it can be utilised in embedded systems. After analysing the steps above we realised the process using small programs using MATLAB which is able to do small no. of isolated word recognition.

CHAPTER 1

INTRODUCTION

Speech being a natural mode of communication for humans can provide a convenient interface to control devices. Some of the speech recognition applications require speaker-dependent isolated word recognition. Current implementations of speech recognizers have been done for personal computers and digital signal processors. However, some applications, which require a low-cost portable speech interface, cannot use a personal computer or digital signal processor based implementation on account of cost, portability and scalability. For instance, the control of a wheelchair by spoken commands or a speech interface for Simputer.

Spoken language interfaces to computers is a topic that has lured and fascinated engineers and speech scientists alike for over five decades. For many, the ability to converse freely with a machine represents the ultimate challenge to our understanding of the production and perception processes involved in human speech communication. In addition to being a provocative topic, spoken language interfaces are fast becoming a necessity. In the near future, interactive networks will provide easy access to a wealth of information and services that will fundamentally affect how people work, play and conduct their daily affairs. Today, such networks are limited to people who can read and have access to computers---a relatively small part of the population even in the most developed countries. Advances in human language technology are needed for the average citizen to communicate with networks using natural communication skills using everyday devices, such as telephones and televisions. Without fundamental advances in user-centred interfaces, a large portion of society will be prevented from participating in the age of information, resulting in further stratification of society and tragic loss in human potential.

1.1 HISTORICAL BACKGROUND

Speech recognition technology has advanced tremendously over the last four decades, from ad-hoc algorithms to sophisticated solutions using hill-climbing parameter estimation and effective search strategies. We discuss briefly the advances in speech recognition, the computing scenario in portable devices (mostly cell phones), and the applications conundrum that has made these advances technical step children in the consumer driven economy line up with the other, and by accumulating information about the goodness of the match. The time alignment algorithm dynamic time warping (DTW) was an implementation of dynamic programming, promoted by both AT&T on the East Coast, and George White at Fairchild on the West Coast. Once the speech signal could be efficiently characterized, the floodgates opened for speech applications including speech coding using LPC, word spotting, and speech recognition. During the 1970's, the government funded research and testing programs in "word spotting", where known words were to be identified in the speech of talkers unknown to the training algorithm, yielding time warping algorithms for matching acoustic utterances. This technology was introduced to the community by IDA in a conference in 1982, but the IBM research organization had been exploring this space since 1970 in large vocabulary applications. Most modern embedded speech recognition applications use the HMM technology. Hidden Markov Models allow phonetic or word rather than frame-by-frame modelling of speech. They also are supported by very pretty convergence theorems, and an efficient training algorithm. Unlike DTW, the HMM recognition algorithms model the speech signal rather than the acoustic composite, and they tend to be more robust to background noise and distortion. In current applications, the cost associated with a mis-recognition are small enough so that sophisticated noise suppression techniques are not economically viable – it is often enough to train an HMM system with some noisy data.

1.2 PRESENT TREND

At the latest it can be said is a lot of advances has been done in the case of speech recognition. the present version of windows (Windows Vista) is supplied with a well versed speech recognition system which works real fine. Present new technology mobile phones are now being versed with speech recognition also to a large extent. With developing technology speech recognition is also gaining its pace.

With the advent of the ARM-9 processor in phones, it has been possible to support even more sophisticated applications. The Samsung P-207, launched in August 2005, contained a very competent speaker adapted large vocabulary recognition system which allowed users to dictate SMS messages and email. The training sequence for adaptation to the talker was a series of 124 words cued by the phone. The underlying technology is based on a Phonetic speech recognition engine using a Markov model, modified to be very efficient in both computation and footprint. While the details are closely held, this capability demonstrates that the current hardware can support a multitude of speech recognition applications. Among the applications currently being developed are navigation systems, voice enabled mobile search, and continuous dictation for text creation. Because modern cell phones have multiple connections to the network, and because voice channels have increasing fidelity, many speech services are available through the network as well as locally. Many carriers offer voice dialling and messaging services by voice; the technological challenges here are operational, but the underlying speech algorithms and techniques have much in common with embedded systems.

CHAPTER 2

LITERATURE IN SPEECH RECOGNITION

Human speech is the foundation of self-expression and communication with others. In the past, ranges of speech based communication technologies have been developed. Automatic speech recognition systems (ASR) are such an example. before even starting explains the different ASR let us first brush up some of the basics.

2.1 SPEECH PRODUCTION

While you are producing speech sounds, the air flow from your lungs first passes the glottis and then your throat and mouth. Depending on which speech sound you articulate, the speech signal can be excited in three possible ways:

• **Voiced excitation**: - The glottis is closed. The air pressure forces the glottis to open and close periodically thus generating a periodic pulse train (triangle–shaped). This "fundamental frequency" usually lies in the range from 80Hz to 350Hz.

• **Unvoiced excitation**:-The glottis is open and the air passes a narrow passage in the throat or mouth. This results in a turbulence which generates a noise signal. The spectral shape of the noise is determined by the location of the narrowness.

• **Transient excitation**: - A closure in the throat or mouth will raise the air pressure. By suddenly opening the closure the air pressure drops down immediately. ("plosive burst") With some speech sounds these three kinds of excitation occur in combination. The spectral shape of the

speech signal is determined by the shape of the vocal tract (the pipe formed by your throat, tongue, teeth and lips). By changing the shape of the pipe (and in addition opening and closing the air flow through your nose) you change the spectral shape of the speech signal, thus articulating different speech sounds.

2.2 TECHNICAL CHARACTERSTICS OF SPEECH SIGNAL

An engineer looking at (or listening to) a speech signal might characterize it as follows:

• The bandwidth of the signal is 4 kHz

• The signal is periodic with a fundamental frequency between 80 Hz and 350 Hz

• There are peaks in the spectral distribution of energy at

$$(2n - 1) * 500 \text{ Hz} ; n = 1, 2, 3, \ldots$$

• The envelope of the power spectrum of the signal shows a decrease with increasing frequency (-6dB per octave).

2.2.1 Bandwidth

The bandwidth of the speech signal is much higher than the 4 kHz stated above. In fact, for the fricatives, there is still a significant amount of energy in the spectrum for high and even ultrasonic frequencies. However, as we all know from using the (analog) phone, it seems that within a bandwidth of 4 kHz the speech signal contains all the information necessary to understand a human Voice.

2.2.2 Fundamental Frequency

As described earlier, using voiced excitation for the speech sound will result in a pulse train, the so-called fundamental frequency. Voiced excitation is used when articulating vowels and some of the consonants. For fricatives (e.g., /f/ as in fish or /s/, as in mess), unvoiced excitation (noise) is used. In these cases, usually no fundamental frequency can be detected. On the other hand, the

zero crossing rate of the signal is very high. Plosives (like /p/ as in put), which use transient excitation, you can best detect in the speech signal by looking for the short silence necessary to build up the air pressure before the plosive bursts out.

2.2.3 Peaks in the Spectrum

After passing the glottis, the vocal tract gives a characteristic spectral shape to the speech signal. If one simplifies the vocal tract to a straight pipe (the length is about 17cm), one can see that the pipe shows resonance at the frequencies. These frequencies are called formant frequencies. Depending on the shape of the vocal tract (the diameter of the pipe changes along the pipe), the frequency of the formants (especially of the 1st and 2nd formant) change and therefore characterize the vowel being articulated.

2.2.4 The Envelope of the Power Spectrum Decreases with Increasing Frequency

The pulse sequence from the glottis has a power spectrum decreasing towards higher frequencies by -12dB per octave. The emission characteristics of the lips show a high-pass characteristic with +6dB per octave. Thus, this results in an overall decrease of -6dB per octave.

2.3 A VERY SIMPLE MODEL OF SPEECH PRODUCTION

As we have seen, the production of speech can be separated into two parts: Producing the excitation signal and forming the spectral shape. Thus, we can draw a simplified model of speech production:

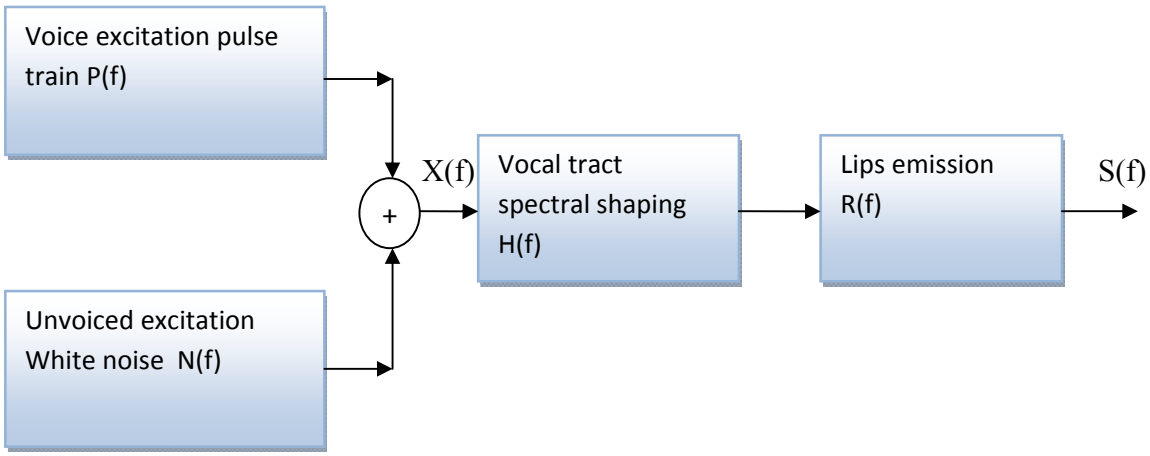


Fig 2.3 : Block diagram for modelling speech production

This model works as follows: Voiced excitation is modelled by a pulse generator which generates a pulse train (of triangle–shaped pulses) with its spectrum given by P(f). The unvoiced excitation is modelled by a white noise generator with spectrum N(f). To mix voiced and unvoiced excitation, one can adjust the signal amplitude of the impulse generator (v) and the noise generator (u). The output of both generators is then added and fed into the box modelling the vocal tract and performing the spectral shaping with the transmission function H(f). The emission characteristics of the lips is modelled by R(f). Hence, the spectrum S(f) of the speech signal is given as:

$$S(f) = (v \cdot P(f) + u \cdot N(f)) \cdot H(f) \cdot R(f) = X(f) \cdot H(f) \cdot R(f) \quad (1.2)$$

To influence the speech sound, we have the following parameters in our speech production model:

• the mixture between voiced and unvoiced excitation (determined by v and u)

• the fundamental frequency (determined by P(f))

• the spectral shaping (determined by H(f))

• the signal amplitude (depending on v and u)

These are the technical parameters describing a speech signal. To perform speech recognition, the parameters given above have to be computed from the time signal (this is called speech signal analysis or ”acoustic pre processing”) and then forwarded to the speech recognizer. For the speech recognizer, the most valuable information is contained in the way the spectral shape of the speech signal changes in time. To reflect these dynamic changes, the spectral shape is determined in short intervals of time, e.g., every 10 ms . By directly computing the spectrum of the speech signal, the fundamental frequency would be implicitly contained in the measured spectrum (resulting in unwanted ”ripples” in the spectrum). Figure 2.3.0 shows the time signal of the vowel /a:/ and fig. 2.3.1 shows the logarithmic power spectrum of the vowel computed via FFT.

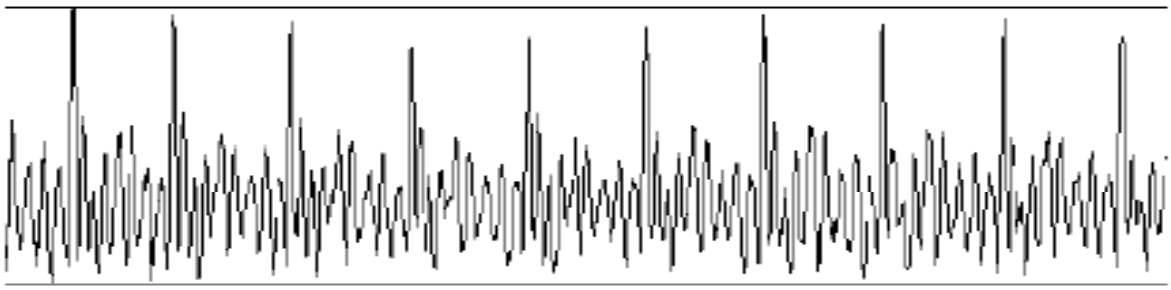


Fig 2.3.0.-Time signal of the vowel /a:/ (fs = 11kHz, length = 100ms).

The high peaks in the time signal are caused by the pulse train P(f) generated by voiced excitation.

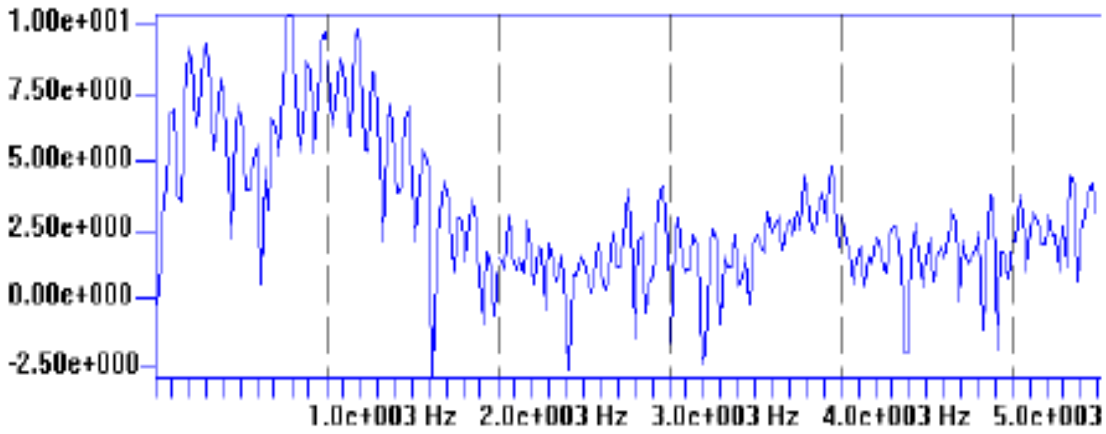


Fig 2.3.1-Log power spectrum of the vowel /a:/ (fs = 11kHz, N = 512).

The ripples in the spectrum are caused by P(f).

2.4 Speech recognition approach

Broadly speaking there are two approaches to speech recognition which can be described by the following block diagram:

Pattern Recognition Approach

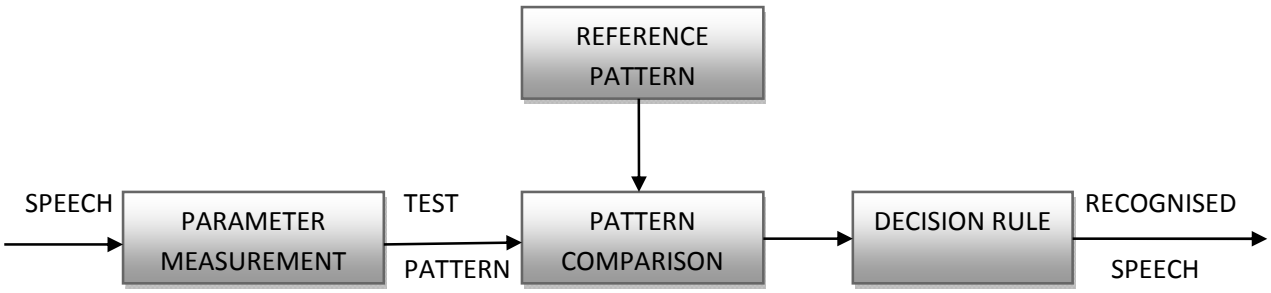


Fig 2.4.0 Pattern Recognition Approach

Acoustic Phonetic Approach

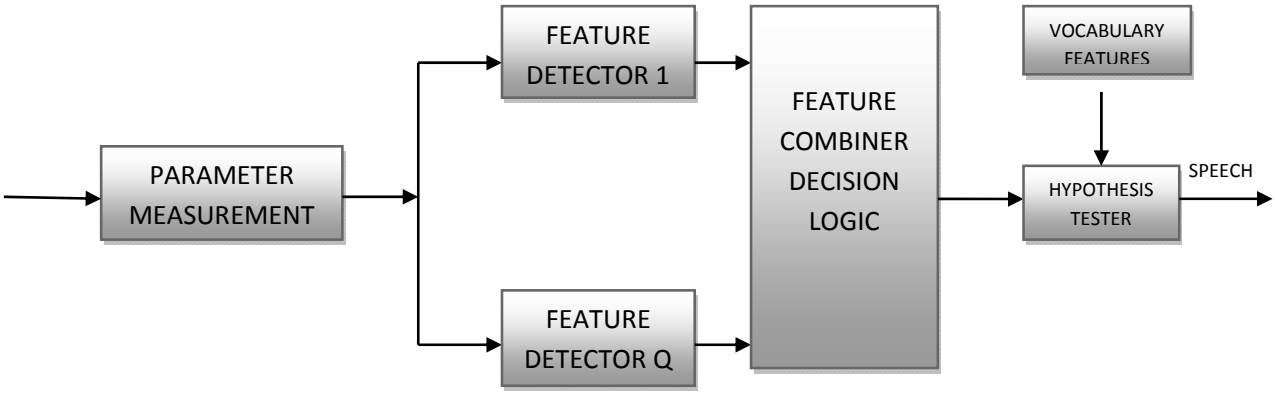


Fig 2.4.1 Acoustic Phonetic Approach

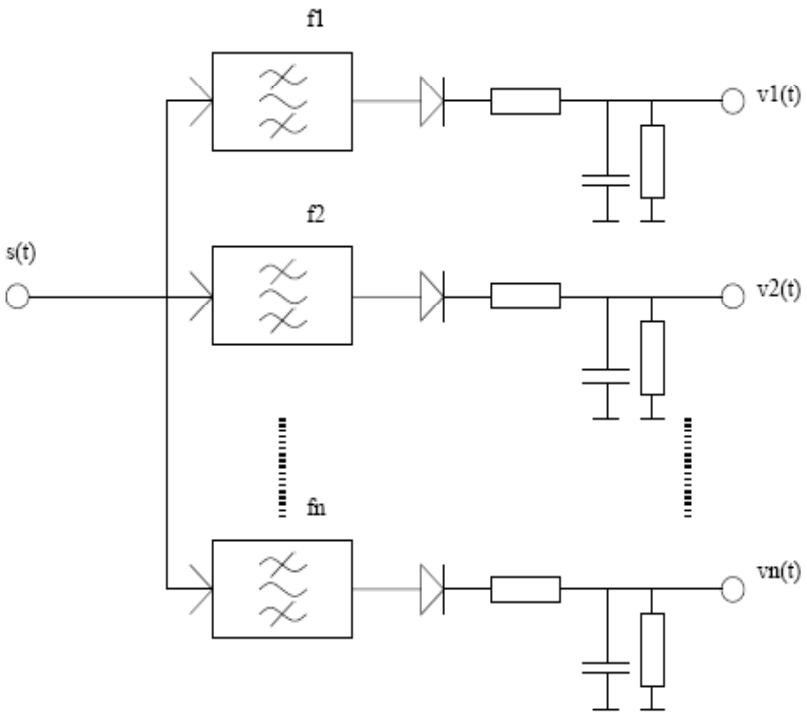
2.5 Speech Parameters Used by Speech Recognition Systems

As shown above, the direct computation of the power spectrum from the speech signal results in a spectrum containing ”ripples” caused by the excitation spectrum X(f). Depending on the implementation of the acoustic pre processing however, special transformations are used to separate the excitation spectrum X(f) from the spectral shaping of the vocal tract H(f). Thus, a smooth spectral shape (without the ripples), which represents H(f) can be estimated from the

speech signal. Most speech recognition systems use the so–called mel frequency cepstral coefficients (MFCC) and its first (and sometimes second) derivative in time to better reflect dynamic changes.

2.6 Band of Filters approach for Computation of the Short Term Spectra

As we recall, it is necessary to compute the speech parameters in short time intervals to reflect the dynamic change of the speech signal. Typically, the spectral parameters of speech are estimated in time intervals of 10ms. First, we have to sample and digitize the speech signal. Depending on the implementation, a sampling frequency fs between 8kHz and 16kHz and usually a 16bit quantization of the signal amplitude is used. After digitizing the analog speech signal, we get a series of speech samples s(k ·t) where t = 1/fs or, for easier notation, simply s(k).



Now a pre emphasis filter is used to eliminate the -6dB per octave decay of the spectral energy:

$$\widehat{S}(k)=S(k)-0.97*S(k-1)$$

Fig 2.6 A filter bank for spectral analysis

Then, a short piece of signal is cut out of the whole speech signal. This is done by multiplying the speech samples ˆs(k) with a windowing function w(k) to cut out a short segment of the speech signal, $V_m(k)$ starting with sample number k = m and ending with sample number k = m + N − 1. The length N of the segment (its duration) is usually chosen to lie between 16ms to 25 ms, while the time window is shifted in time intervals of about 10ms to compute the next set of speech parameters. Thus, overlapping segments are used for speech analysis.

Many window functions can be used, the most common one is the so–called Hamming-Window:

$$w(k) = \begin{cases} 0.54 - 0.46 \quad \cos\left(\frac{2\pi k}{N-1}\right) & : if\ k = 0,1 \dots \dots .. N-1 \\ 0 & : else \end{cases}$$

Where N is the length of the time window in samples. By multiplying our speech signal with the time window, we get a short speech segment $v_m(k)$:

$$v_m(k) = \begin{cases} \hat{S}\ (\mathrm{k}) * w(k-m) & : if\ k = m, m+1, \dots \dots m+N-1 \\ 0 & : else \end{cases}$$

As already mentioned, N denotes the length of the speech segment given in samples (the window length is typically between 16ms and 25ms) while m is the start time of the segment. The start time m is incremented in intervals of (usually) 10ms, so that the speech segments are overlapping each other. All the following operations refer to this speech segment$V_m(k)$, k = m. . .m + N − 1. To simplify the notation, we shift the signal in time by m samples to the left, so that our time index runs from 0..... N −1 again. From the windowed signal, we want to compute its discrete power spectrum $|V(n)|^2$. First of all, the complex spectrum V (n) is computed. The complex spectrum V (n) has the following properties:

• The spectrum V (n) is defined within the range from n = −∞ to n = +∞.

• V (n) is periodic with period N, i.e.,

$$V(n \pm i \cdot N) = V(n);\ i = 1, 2 \ldots$$

• Since $V_m(k)$ is real–valued, the absolute values of the coefficients are also Symmetric:

$$|V(-n)| = |V(n)|$$

To compute the spectrum, we compute the discrete Fourier transform (DFT, which gives us the discrete, complex–valued short term spectrum.

$$\hat{V}(n) = \sum_{k=0}^{N-1} v(k) * e^{-j2\pi kn/N} \ ;\ n=0, 1 \ldots N-1$$

The DFT gives us N discrete complex values for the spectrum $\hat{V}$ (n) at the frequencies n · f where

$$\Delta f = \frac{1}{N\Delta t}$$

Remember that the complex spectrum V (n) is defined for n = −∞ to n = +∞, but is periodic with period N .

2.7 The LPC model

The basic idea behind the LPC model is that a given speech sample at time *n* ,*s(n),* can be approximated as a linear combination of the past *p* speech samples, such that

$$s(n) = a_1\, s(n-1) + a_2\, s(n-2) + \ldots + a_p\, s(n-p)$$

where the coefficients a_1 , a_2 ….a_p are assumed constant over the speech analysis frame. We convert the equation to an equality by including an excitation term *G u(n)* giving:

$$s(n) = \sum_{i=1}^{p} a_i\, s(n-i) + G\, u(n)$$

where u(n) is a normalized excitation and G is the gain of the excitation .by expressing in z-domain we get the relation

$$S(z)= \sum_{i=1}^{p} a_i z^{-1} S(z) + G\, u(z)$$

Leading to the transfer function

$$H(z) = \frac{S(z)}{G\, U(z)} = \frac{1}{1-\sum_{i=1}^{p} a_i z^{-1}} = \frac{1}{A(z)}$$

The interpretation of above equation is shows the normalized excitation source u(n) being scaled by the gain *G,* and acting as input to the all-pole system H(z)= $\frac{1}{A(z)}$,to produce the speech signal s(n), based on our knowledge that the actual excitation function for speech is essentially either a quasiperiodic pulse train (for voiced speech sounds) or a random noise source (for unvoiced sounds) ,the appropriate synthesis model for speech, corresponding to the LPC analysis.

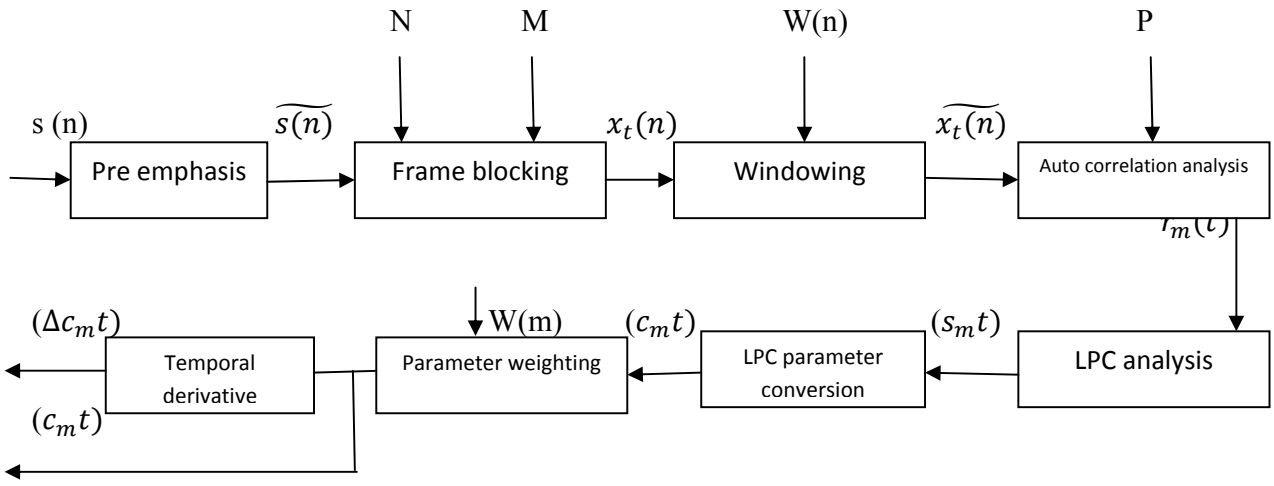


Fig 2.7 block diagram of LPC processor for speech recognition

Here the normalized excitation source is chosen by a switch whose position is controlled by the voiced/unvoiced character of the speech , which chooses either a quasperiodic train of pulses as the excitation for voiced sound or a random noise sequence for unvoiced sounds. The appropriate

gain G of the source is estimated from the speech signal and the scaled source is used as input to a digital filter *(H (z))* which is controlled by the vocal tract parameters characteristic of the speech being produced. Thus the parameters of this model are voiced / unvoiced classification, pitch period for voiced sounds, the gain parameter, and the coefficient of the digital filter (a_k). These parameters all vary slowly with time.

2.8 DYNAMIC PARAMETERS

As stated earlier in, the MFCC are computed for a speech segment v_m at time index m in short time intervals of typically 10ms. In order to better reflect the dynamic changes of the MFCC cm(q) (and also of the energy v_m) in time, usually the first and second derivatives in time are also computed, e.g. by computing the difference of two coefficients lying τ time indices in the past and in the future of the time index under consideration

$$\Delta c_m(q) = c_{m+r}(q) - c_{m-r}(q); q = 0,1, \dots \dots Q-1$$

$$\Delta\Delta c_m(q) = \Delta c_{m+r}(q) - \Delta c_{m-r}(q); q = 0,1, \dots \dots Q-1$$

The time interval usually lies in the range $2 \leq \tau \leq 4$. These results in a total number of up to 63 parameters which are computed every 10ms. Of course, the choice of parameters for acoustic pre processing has a strong impact on the performance of the speech recognition systems For our purposes however, it is sufficient to remember that the information contained in the speech signal can be represented by a set of parameters which has to be measured in short intervals of time to reflect the dynamic change of those parameters.

2.9 FEATURE VECTOR AND VECTOR SPACE

If you have a set of numbers representing certain features of an object you want to describe, it is useful for further processing to construct a vector out of these numbers by assigning each

measured value to one component of the vector If you measure those parameters every second or so and you put the temperature into the first component and the humidity into the second component of a vector, you will get a series of two–dimensional vectors describing how the air in your office changes in time. Since these so–called feature vectors have two components, we

can interpret the vectors as points in a two–dimensional vector space. Thus we can draw a two–dimensional map of our measurements as sketched below. Each point in our map represents the

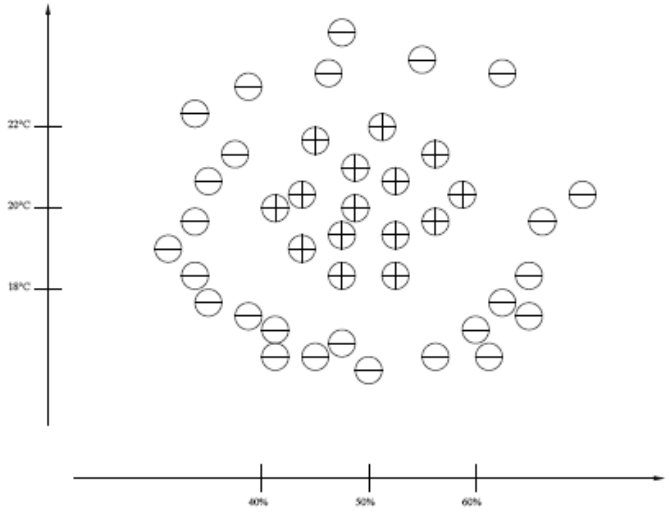


Fig2.9 : A Map of feature Vectors

temperature and humidity in our office at a given time. As we know, there are certain values of temperature and humidity which we find more comfortable than other values. In the map the comfortable value– pairs are shown as points labelled ”+” and the less comfortable ones are shown as ”-”. You can see that they form regions of convenience and inconvenience, respectively.

CHAPTER 3

FEATURE EXTRACTION

One of the first decisions in any pattern recognition system is the choice of what features to use. How exactly to represent the basic signal that is to be classified, in order to make the classification algorithm's job easiest. In this part the details of extracting the features per frame of the speech signal are discussed.

3.1 Pre emphasis

The digitized speech signal is processed by a first order digital network in order to spectrally flatten the signal. This pre emphasis is easily implemented in the time domain by taking difference.

$$\tilde{A}(n) = A(n) - a * A(n-1)$$

a= scaling factor = 0.95

A(n)= Digitized Speech Sample

A(n-1) = Previous digitized Speech Sample

$\tilde{A}(n)$ = Pre emphasised Speech Sample.

n = No. of Samples in the whole frame.

3.2 Blocking into Frames

Section of N (e.g. 300) consecutive speech samples are used as a single frame. Consecutive frames are spaced M (e.g. 100) samples apart.

$$X_{,}(n) = \tilde{A}(M*l+n) \quad , \ 0 <= n <= N-1 \text{ and } 0 <= l <= L-1$$

N = Total No. of samples in a frame.

M = Total No. of sample spacing between the frames. [Measure of overlap]

L = Total number of frames.

3.3 Frame Windowing

Each frame is multiplied by an N sample window W (n). Here we use a hamming window. This hamming window is used to minimize the adverse effects of chopping an N sample section out of the running speech signal. While creating the frames the frames the chopping of N sample from the running signal may have a bad effect on the signal parameter's. To minimize this effect windowing is done.

$$\hat{U}_{,}(n) = X_{,}(n) * W(n) , \qquad 0 <= n <= N-1$$

W(n) = Scale factor i.e. (0.54 - 0.46*Cos(2*pie*n/ N)) , 0 <= n <= N-1

N = Total No. of samples in a frame.

The multiplicative scaling factor ensures appropriate overall signal amplitude.

3.4 Mel frequency Cepstral coefficients

The cepstral coefficients, which are the coefficients of the Fourier transform representation of the log magnitude spectrum, have been shown to be a more robust, reliable feature set for speech recognition than the LPC coefficients. Because of the sensitivity of the low

order cepstral coefficients to overall spectral slope and the sensitivity of the high-order cepstral coeffecients to noise, it had become a standard technique to weight the cepstral coefficients by a tapered window so as to minimize these sensitivities.

3.5 RASTA coefficients

Another popular speech feature representation is known as RASTA-PLP, an acronym for Relative Spectral Transform - Perceptual Linear Prediction. PLP was originally proposed by Hynek Hermansky as a way of warping spectra to minimize the differences between speakers while preserving the important speech information [Herm90]. RASTA is a separate technique that applies a band-pass filter to the energy in each frequency subband in order to smooth over short-term noise variations and to remove any constant offset resulting from static spectral coloration in the speech channel e.g. from a telephone line [HermM94].

CHAPTER 4

DISTANCE MEASURES

So far, we have found a way to classify an unknown vector by calculation of its class–distances to predefined classes, which in turn are defined by the distances to their individual prototype vectors. Now we will briefly look at some commonly used distance measures. Depending on the application at hand, each of the distance measures has its pros and cons, and we will discuss their most important properties.

4.1 Euclidean Distance

The Euclidean distance measure is the" standard" distance measure between two vectors in feature space (with dimension DIM)

$$d^2_{Euclid} \quad (\vec{x}, \vec{p}) \quad = \quad \sum_{i=0}^{DIM-1} (x_i - p_i)^2$$

To calculate the Euclidean distance measure, you have to compute the sum of the squares of the differences between the individual components of ~x and ~p. This can also be written as the following scalar product

$$d^2_{Euclid} \quad (\vec{x}, \vec{p}) \quad = \quad (\vec{x} - \vec{p})' * (\vec{x} - \vec{p})$$

Where" ′ "denotes the vector transpose. Compute the square of the Euclidean distance, d2 instead of d. The Euclidean distance is probably the most commonly used distance measure in pattern recognition.

4.2 Weighted Euclidean Distance

Both the Euclidean distance and the City Block distance are treating the individual dimensions of the feature space equally, i.e., the distances in each dimension contributes in the same way to

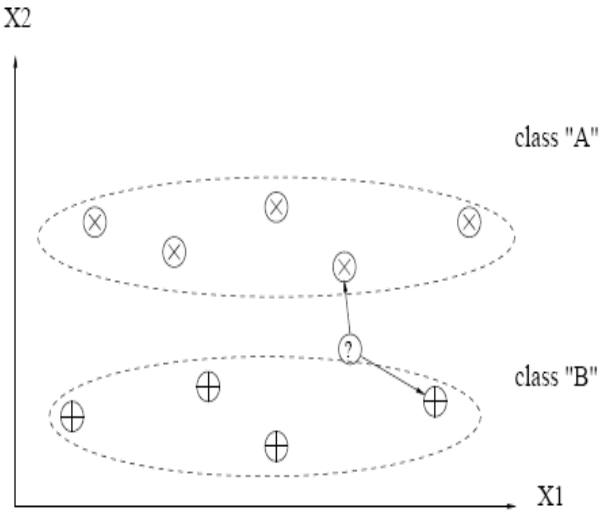


Fig 4.2 two dimensions with different scales

the overall distance. In Figure 4.2 we see a more abstract example involving two classes and two dimensions. The dimension x1 has a wider range of values than dimension x2, so all the measured values (or prototypes) are spread wider along the axis denoted as ”x1” as compared to axis ”x2”. Obviously, Euclidean or City Block distance measure would give the wrong result, classifying the unknown vector as ”class A” instead of ”class B” which would (probably) be the correct To cope with this problem, the different scales of the dimensions of our feature vectors have to be compensated when computing the distance. This can be done by multiplying each contributing term with a scaling factor specific for the respective dimension. This leads us to the so–called”Weighted Euclidean Distance”:

$$d^2_{Weighted\_Euclid}\,(\vec{x},\vec{p}) = \sum_{i=0}^{DIM-1} \lambda_i * (x_i - p_i)^2$$

As before, this can be rewritten as:

$$d^2_{Weighted\_Euclid}\,(\vec{x},\vec{p}) = (\vec{x}-\vec{p})' * \tilde{\Lambda} * (\vec{x}-\vec{p})$$

$$\tilde{\Lambda} = \mathrm{diag}\begin{bmatrix} \lambda_0 & & & \\ & \cdots & & \\ & & \lambda_i & \\ & & & \cdots \\ & & & \lambda_{DIM-1} \end{bmatrix}$$

The scaling factors are usually chosen to compensate the variances of the measured Features:-

$$\lambda_i = \frac{1}{\sigma_i^2}$$

The variance of dimension i is computed from a training set of N vectors {~x0, ~x1. . . ~xN−1}. Let $x_{i,n}$ denote the i-th element of vector ~xn, then the variances σ_i^2 can be estimated from the training set as follows:

$$\sigma_i^2 = \frac{1}{N-1}\sum_{n=0}^{N-1}(x_{i,n} - m_i)^2$$

where m_i is the mean value of the training set for dimension :

$$m_i = \frac{1}{N}\sum_{n=0}^{N-1}(x_{i,n})$$

4.3 Likelihood distortion

The log spectral difference *V (w)* is the basis of many speech distortion measures. The distortion measures originally proposed by Itakura and Saito (called the Itakura-Saito distortion method) in their formulation of linear prediction as an approximate likelihood estimation is

$$d_{1S}(S, \acute{S}) = \int_{-\pi}^{\pi}(e^{V(w)} - V(w) - 1)\frac{dw}{2\pi}$$

$$= \int_{-\pi}^{\pi}\frac{S(w)}{S(\acute{w})}\frac{dw}{2\pi} - log\frac{\sigma_\infty^2}{\acute{\sigma}_\infty^2} - 1$$

where $\sigma_\infty^2, \acute{\sigma}_\infty^2$ are the one –step prediction errors of *S(w)*and $S(\acute{w})$, respectively as defined in the equation. Besides maximum likelihood interpretation the connection of the Itakura-Saito distortion measures with many statistical and information theoretic notions are also well established. These includes the likelihood ratio test and relative entropy. Although we have considered the Itakura-Saito measures a likelihood related quantity, we focus on speech spectrum comparison.

4.4 The Itakura distortion

It is a little variation of the itakura-saito distortion measure. It can be given as

$$d_1\left(S, \acute{S}\right) = \log\left(\int_{-\pi}^{\pi} \frac{|A|^2}{|A_p|^2} \frac{dw}{2\pi} \right)$$

CHAPTER 5

DYNAMIC TIME WARPING

In the last chapter, we were dealing with the task of classifying single vectors to a given set of classes which were represented by prototype vectors computed from a set of training vectors. our speech signal is represented by a series of feature vectors which are computed every 10ms. A whole word will comprise dozens of those vectors, and we know that the number of vectors (the duration) of a word will depend on how fast a person is speaking. Therefore, our classification task is different from what we have learned before. In speech recognition, we have to classify not only single vectors, but sequences of vectors. Let's assume we would want to recognize a few command words or digits. For an utterance of a word w which is TX vectors long, we will get a sequence of vectors $\hat{X}$ = {~x0, ~x1. . . ~xTX−1} from the acoustic pre processing stage. What we need here is a way to compute a "distance" between this unknown sequence of vectors ˜X and known sequences of vectors $\widehat{W}_k$ = nw~k0, w~k1. . .w~kTWko which are prototypes for the words we want to recognize. Let our vocabulary (here: the set of classes) contain V different words w0,w1, . . .$w_{V-1.}$ In analogy to the Nearest Neighbour classification task from chapter 2, we will allow a word w_v to be represented by a set of prototypes $\widehat{W}$k, ω_v, k = 0, 1. . . (Kωv−1) to reflect all the variations possible due to different pronunciation or even different speakers.

5.1 Distance between Two Sequences of Vectors

As we saw before, classification of a spoken utterance would be easy if we had a good distance measure D($\tilde{X}$, $\widetilde{W}$) at hand (in the following, we will skip the additional indices for ease of notation).

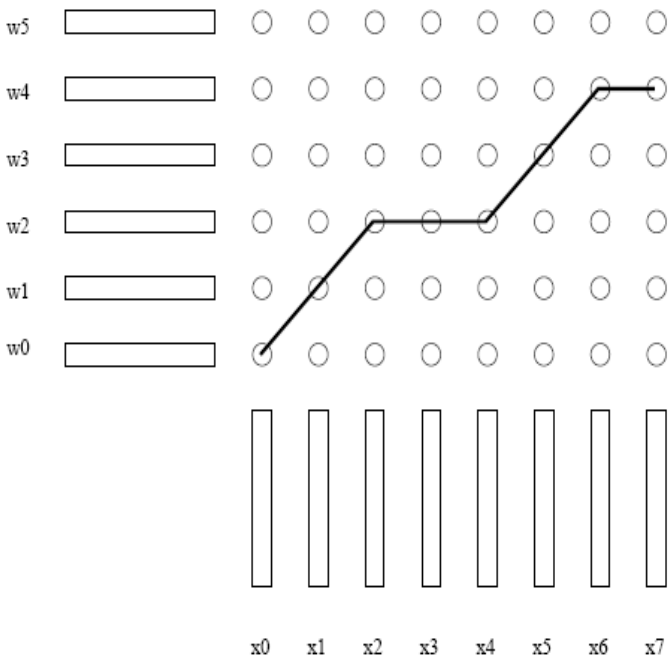


Fig 5.1: Possible assignment between the vector pairs of $\hat{X}$ and $\widehat{W}$

The distance measure we need must:

- Measure the distance between two sequences of vectors of different length (TX and TW)
- While computing the distance, find an optimal assignment between the individual feature vectors of $\hat{X}$ and $\widehat{W}$
- Compute a total distance out of the sum of distances between individual pairs of feature vectors of $\hat{X}$ and $\widehat{W}$

5.2 Comparing Sequences with Different Lengths

he main problem is to find the optimal assignment between the individual vectors of $\tilde{X}$ and W. In Fig. 4.1 we can see two sequences ˜X and ˜W which consist of six and eight vectors, respectively. The sequence ˜W was rotated by 90 degrees, so the time index for this sequence runs from the bottom of the sequence to its top. The two sequences span a grid of possible assignments between the vectors. Each path through this grid (as the path shown in the figure) represents one possible assignment of the vector pairs. For example, the first vector of ˜X is

assigned the first vector of ˜W, the second vector of ˜X is assigned to the second vector of ˜W, and so on.

For the given path P, the distance measure between the vector sequences can now be computed as the sum of the distances between the individual vectors. Let l denote the sequence index of the grid points. Let $d(g_t)$ denote the vector distance d (~x_i, ~w_t) for the time indices i and j defined by the grid point g_t = (i, j). Then the overall distance can be computed as:

$$D(\tilde{X}, \tilde{W}, P) = \sum_{t=1}^{L_p} d(g_t)$$

5.3 Finding the Optimal Path

The criterion of optimality we want to use in searching the optimal path P(opt) should be to minimize D_ ˜X, ˜W , P_:

$$P_{opt} = \arg\min_{P} \left\{ D\left(\tilde{X}, \tilde{W}, P\right) \right\}$$

Fortunately, it is not necessary to compute all possible paths P and corresponding distances D_ ˜X, ˜W , P_ to find the optimum. Out of the huge number of theoretically possible paths, only a fraction is reasonable for our purposes Note that Fig. 5.3 does not show the possible extensions of the path from a given point but the possible predecessor paths for a given grid point. We will soon get more familiar with this way of thinking. As we can see, a grid point (i, j) can have the following predecessors:

- (i − 1, j) : keep the time index j of ˜X while the time index of ˜W is incremented
- (i − 1, j − 1) : both time indices of ˜X and ˜W are incremented
- (i, j − 1) : keep of the time index i of ˜W while the time index of ˜X is incremented

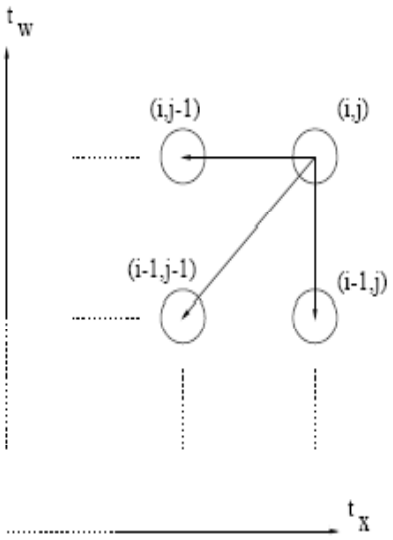


Fig 5.3: Local path alternatives for a grid point

All possible paths P which we will consider as possible candidates for being he optimal path P(opt) can be constructed as a concatenation of the local path alternatives as described above. To reach a given grid point (i, j) from (i−1, j− 1) , the diagonal transition involves only the single vector distance at grid point (i, j) as opposed to using the vertical or horizontal transition, where also the distances for the grid points (i − 1, j) or (i, j − 1) would have to be added. To compensate this effect, the local distance d(~w_i, ~x_i) is added twice when using the diagonal transition.

5.4 Bellman's Principle

Now that we have defined the local path alternatives, we will use Bellman's Principle to search the optimal path P (opt). Applied to our problem, Bellman's Principle states the following: If P (opt) is the optimal path through the matrix of grid points beginning at (0, 0) and ending at (TW −1, TX −1), and the grid point (i, j) is part of path P(opt), then the partial path from (0, 0) to (i, j) is also part of P(opt). From that, we can construct a way of iteratively finding our optimal path P (opt). According to the local path alternatives diagram we chose, there are only three possible

predecessor paths leading to a grid point (i, j): The partial paths from (0, 0) to the grid points (i − 1, j), (i − 1, j − 1) and (i, j − 1). Let's assume we would know the optimal paths (and therefore the accumulated distance δ (.) along that paths) leading from (0, 0) to these grid points. All these path hypotheses are possible predecessor paths for the optimal path leading from (0, 0) to (i, j). Then we can find the (globally) optimal path from (0, 0) to grid point (i, j) by selecting exactly the one path hypothesis among our alternatives which minimizes the accumulated distance δ(i, j) of the resulting path from (0, 0) to (i, j).

The optimization we have to perform is as follows:

$$\delta(i,j) = \min \begin{cases} \delta(i, j-1) & + & d(\overrightarrow{w_i}, \overrightarrow{x_j}) \\ \delta(i-1, j-1) + & 2 * & d(\overrightarrow{w_i}, \overrightarrow{x_j}) \\ \delta(i-1, j) & + & d(\overrightarrow{w_i}, \overrightarrow{x_j}) \end{cases}$$

Termination: D (˜W, ˜X) = δ (TW − 1, TX − 1) is the distance between ˜W and ˜X .The iteration through the matrix beginning with the start point (0, 0). Filled points are already computed, empty points are not. The dotted arrows indicate the possible path hypotheses over which the optimization (4.6) has to be performed. The solid lines show the resulting partial paths after the decision for one of the path hypotheses during the optimization step. Once we reached the top– right corner of our matrix, the accumulated distance δ (TW − 1, TX −1) is the distance D(˜W , ˜X) between the vector sequences. If we are also interested in obtaining not only the distance D(˜W , ˜X), but also the optimal path P, we have — in addition to the accumulated distances — also to keep track of all the decisions we make during the optimization steps . The optimal path is known only after the termination of the algorithm, when we have made the last recombination for the three possible path hypotheses leading to the top– right grid point (TW −1, TX −1). Once this decision is made, the optimal path can be found by reversely following all the local decisions down to the origin (0, 0). This procedure is called backtracking..

CHAPTER 6

RESULTS

As every journey begins with a small step here we are trying to achieve that small step in the field of speech recognition. Here we have presented at first the analysis of different feature extraction procedures. Then we have tried to present an analysis of MFCC as how it is a good approach of feature extraction. Then we have tried to analyse different methods of distance measure used to calculate to the distance between the feature vectors extracted by us. Then we try to do a small analysis of dynamic time warping using dynamic programming approach. At the last but not the least we try to present a small program for small speaker dependent recognition system to recognise isolated words.

Here we want to state that as we were motivated by the application of speech recognition in mobile phones we here are trying to recognise the English numerical digits from 'zero' to nine'.it should be also noted that this applications are not restricted by this and can be used to recognise any isolated words with appropriate changes. All the programming used here is done in matlab due to obvious reasons of it being the most efficient tool for mathematical and signal analysis.

At first we are present a small description of the words used:

Word	Sounds	APRABET
Zero	/z I r o/	Z-IH-R-OW
One	/w ΛΛ/	W-AH-N
Two	/t u/	T-UW
Three	/θ r i/	TH-R-IY
Four	/f o r/	F-OW-R
Five	/f a^y v/	F-AY-V
Six	/s I k s/	S-IH-K-S
Seven	/s ɛ v ð n/	S-EH-V-AX-N
Eight	/e^y t/	EY-T
Nine	/n a^y n/	N-AY-N
Oh	/o/	OW

before doing any speech recognition work we have to convert the speech into digital format. The its fft has to be calculated . this is done by our example matlab programme and formulated the following output for word ‘one”:

>> start

say a word immediately after hitting enter:

% spoke one using a microphone connected to the computer

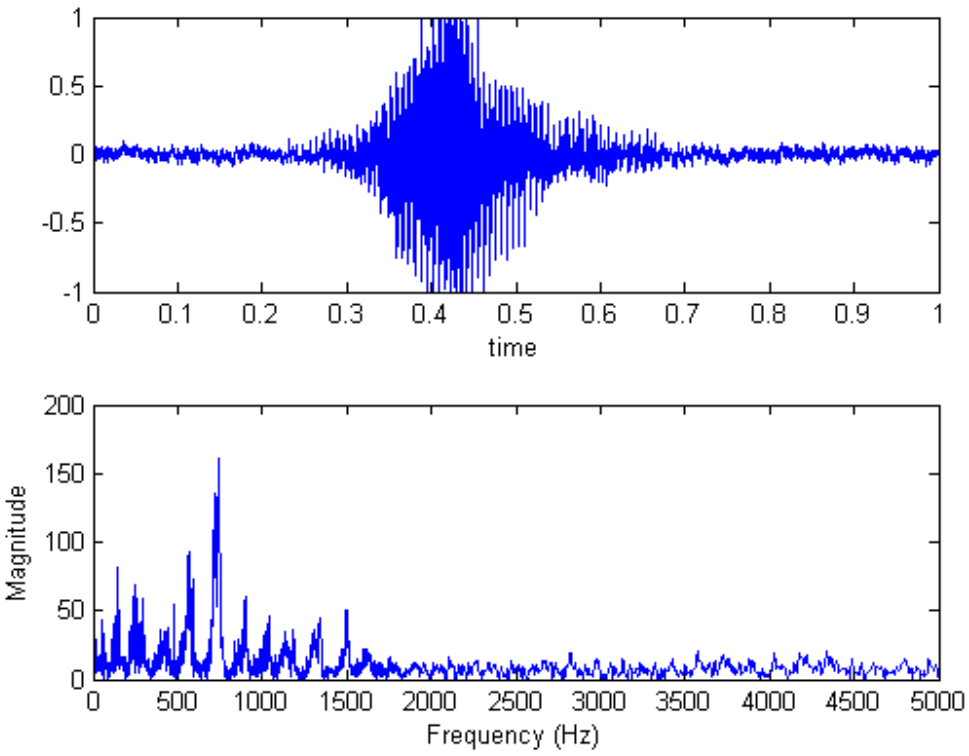


Fig 6.0 plot of time waveform and frequency spectra of spoken ‘one’

6.1 feature extraction of isolated digits

Different feature extraction techniques can be use to extract the features of the given speech sound. Here we try to use compare and find the corresponding features for the digit one to nine.

6.1.1 LPC coefficients

In the LPC based analysis, we use weighted LPC cepstra together with the corresponding time derivatives and energy parameters. To make the feature extraction independent of the absolute energy, which can change quite a lot in analog telephone lines, we only use the derivatives of the log energy and not the energy itself. Tests on real-life data as they arrive on a Speech Recognition board give significantly better results when using no absolute energy.

6.1.2 PLP based analysis

PLP analysis differs from LPC analysis in the sense that we approximate an auditory spectrum by the spectrum of an all-pole model. This auditory spectrum differs from the power spectrum in the sense that we use a nonlinear frequency axis, that we do a critical band analysis with asymmetric weighting cofficients (with low-frequency slopes less steep than high-frequency ones). Also the idea of the non equal sensitivity of hearing at different frequencies and the intensity- loudness power law is included in this more perceptually based LP analysis.

6.1.3 RASTA Analysis

RASTA PLP , which is an extension of the previously described PLP analysis, applies an IIR filtering on the logarithm of the critical band spectrum. The IIR Filter is equivalent to a derivative-reintegration process as to filter out the long-term spectral tilts due to the tele- phone lines. After the two psychoacoustical steps the inverse logarithm is taken, followed by the "traditional" all pole modelling and cepstral recursion.

6.1.4 Mel Frequency Cepstral Coefficients

MFCCs are coefficients that represent audio. They are derived from a type of cepstral representation of the audio clip (a "spectrum-of-a-spectrum"). The difference between the cepstrum and the Mel-frequency cepstrum is that in the MFC, the frequency bands are positioned logarithmically (on the mel scale) which approximates the human auditory system's response

more closely than the linearly spaced frequency bands obtained directly from the FFT (Fast Fourier Transform) or DCT (Discrete Cosine Transform). This can allow for better data processing, for example, in audio compression. However, unlike the sonogram, MFCCs lack an outer ear model and, hence, cannot represent perceived loudness accurately.

MFCCs are commonly derived as follows:

1. Take the Fourier transform of (a windowed excerpt of) a signal

2. Map the log amplitudes of the spectrum obtained above onto the Mel scale, using triangular overlapping windows.

3. Take the Discrete Cosine Transform of the list of Mel log-amplitudes, as if it were a signal.

4. The MFCCs are the amplitudes of the resulting spectrum.

A set of matlab modules were written to find the above mentioned coefficients and the corresponding graps for letters ‘zero’ to ‘nine’ are given below.

For Zero:

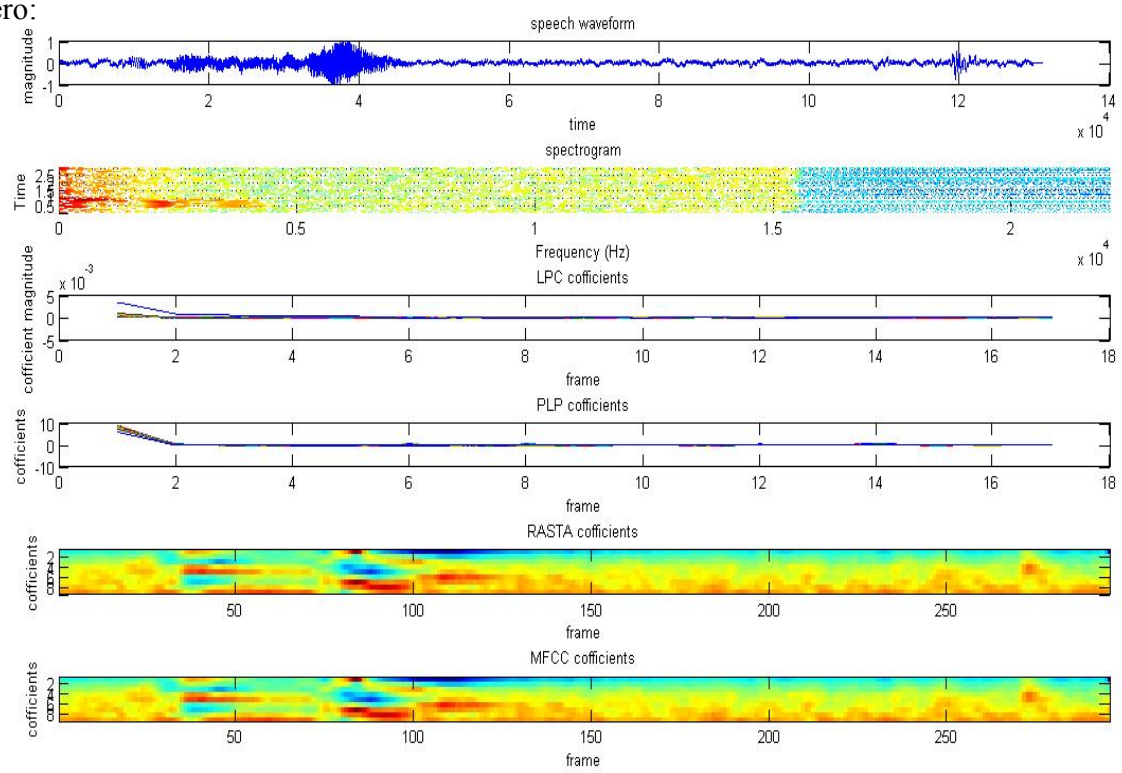


Fig 6.1.4.0 Coefficients for zero

For One:

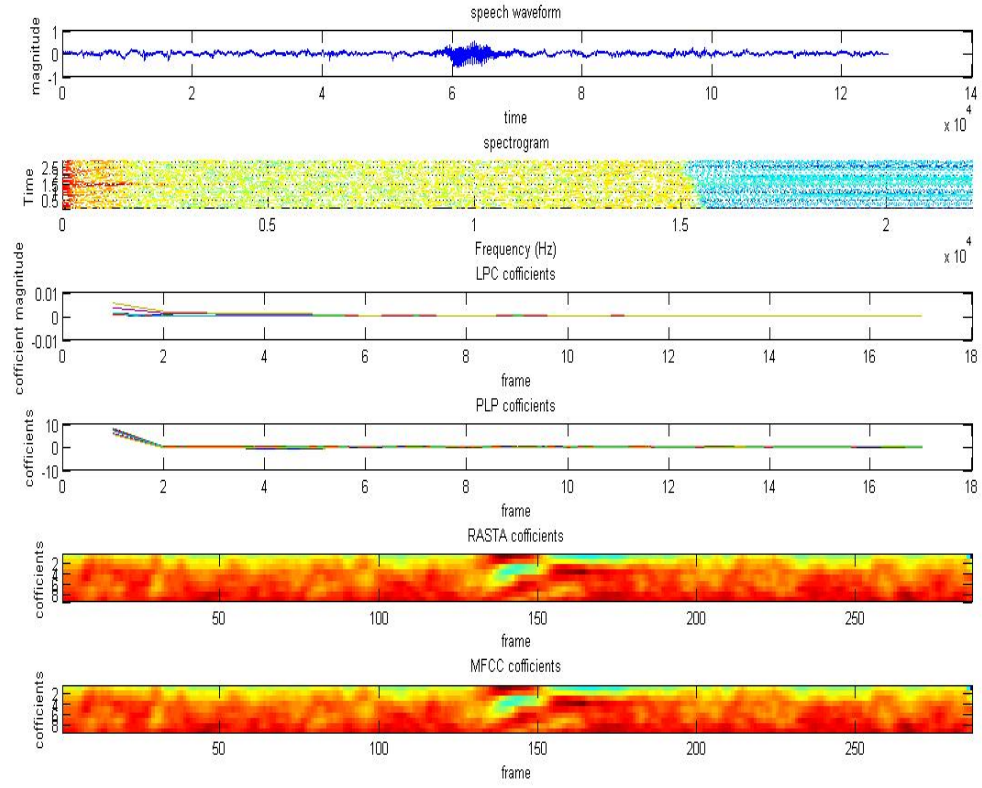


Fig 6.1.4.1 Coefficients for one

For Two:

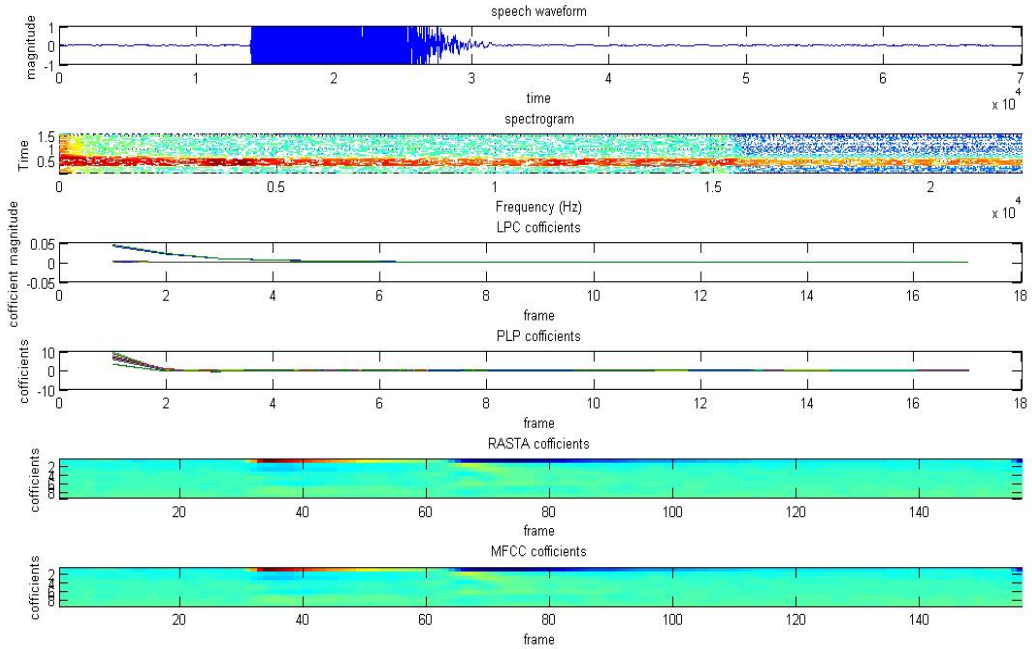


Fig 6.1.4.2 Coefficients for two

For Three:

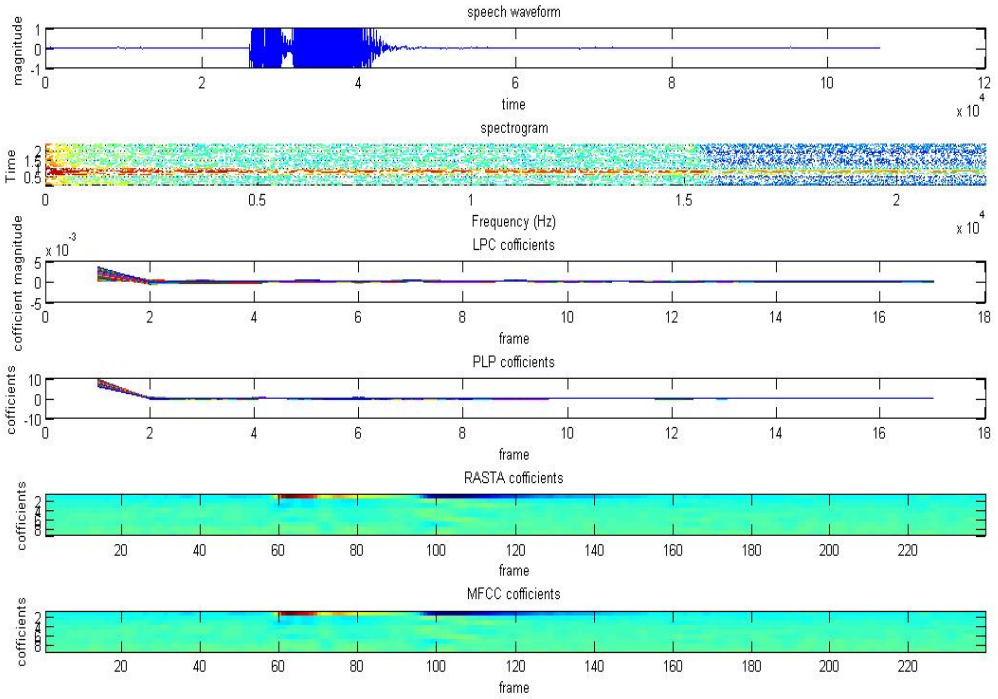


Fig 6.1.4.3 Coefficients for three

For Four:

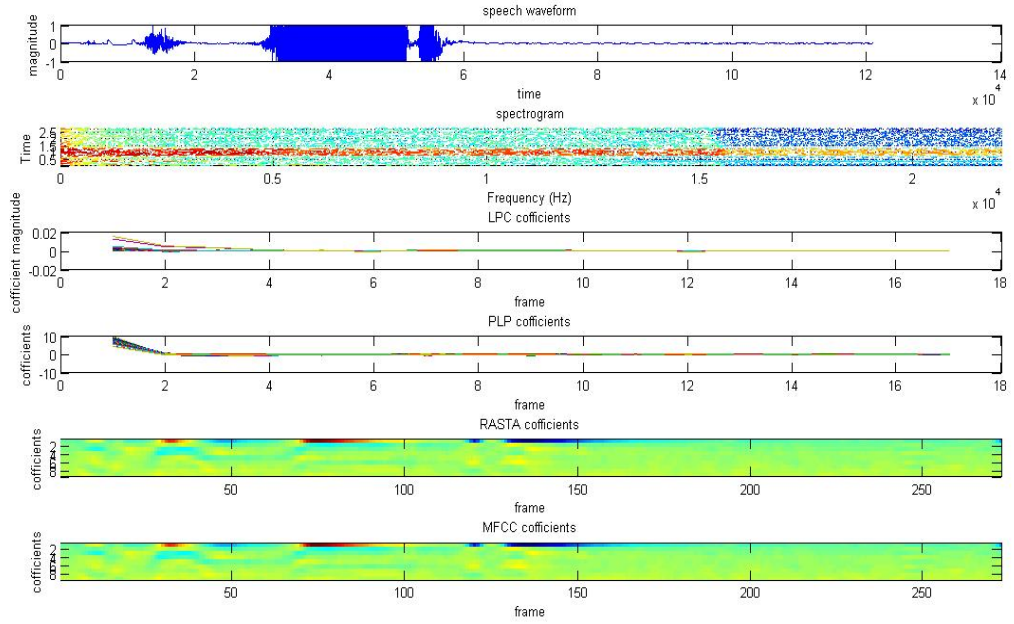


Fig 6.1.4.4 Coefficients for four

For Five:

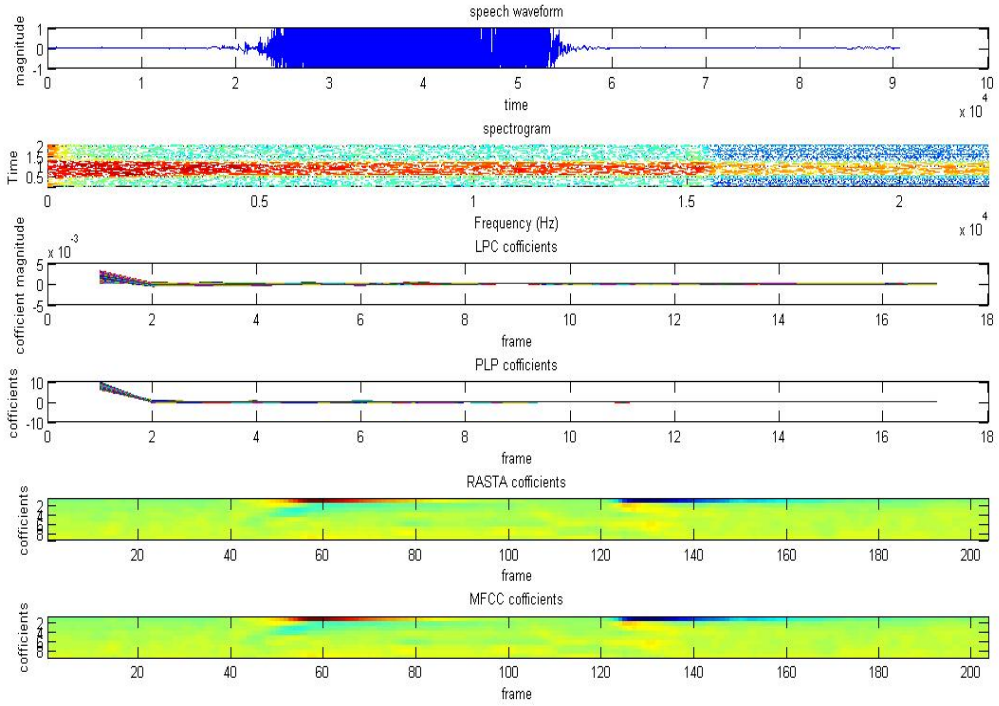


Fig 6.1.4.5 Coefficients for five

For Six:

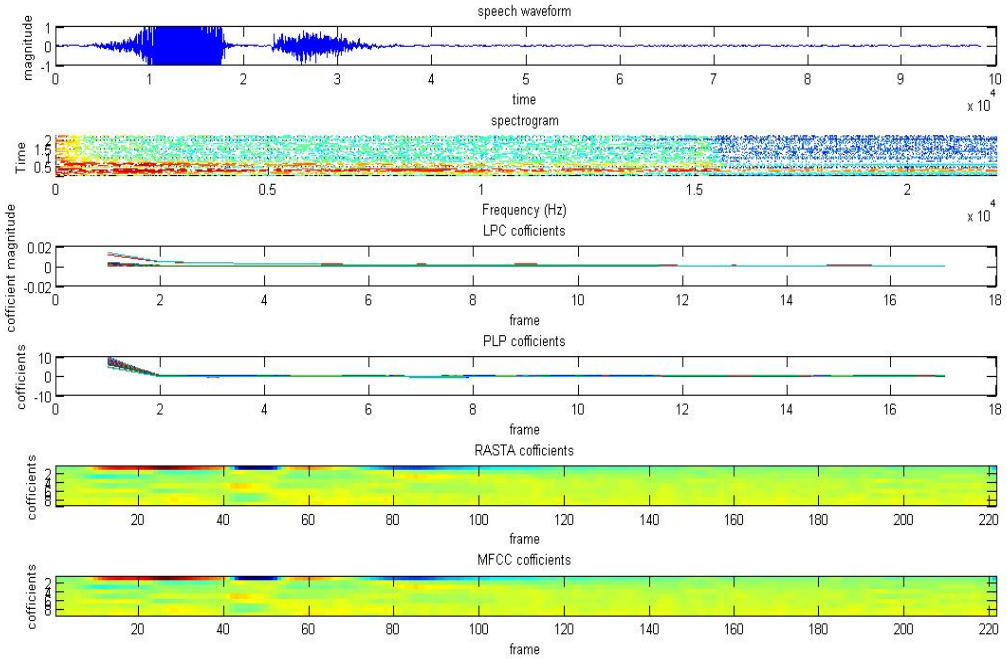


Fig 6.1.4.6 Coefficients for six

For Seven:

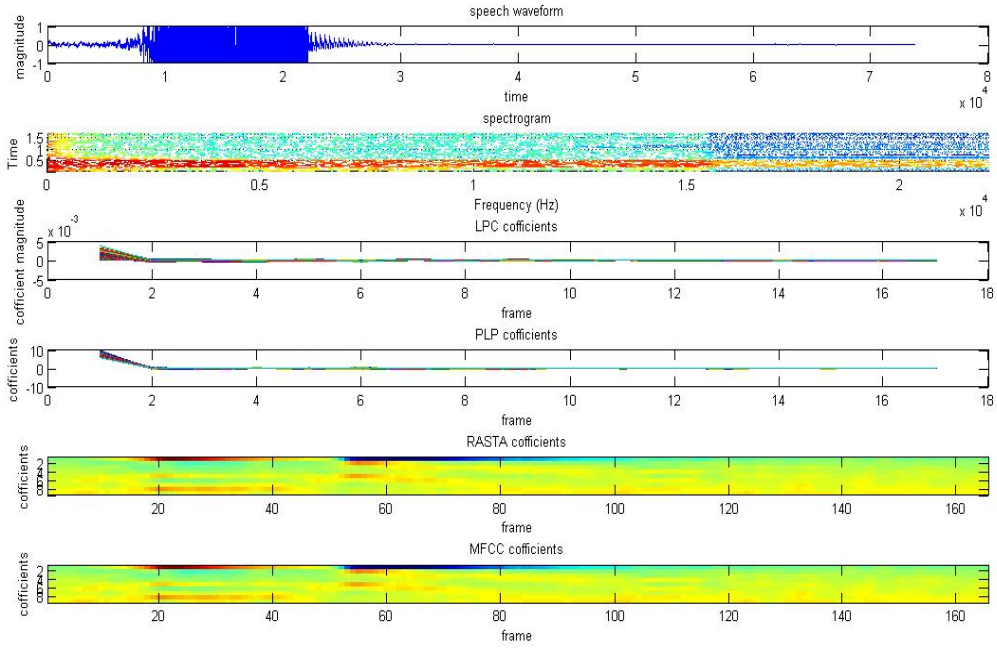


Fig 6.1.4.7 Coefficients for seven

For Eight:

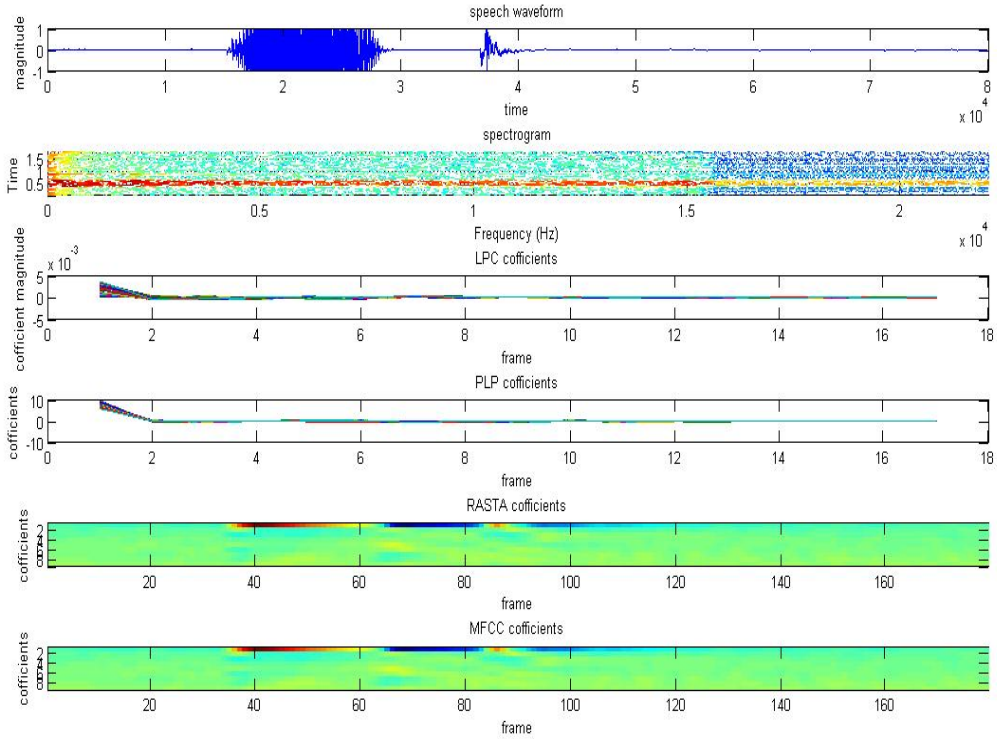


Fig 6.1.4.8 Coefficients for eight

For Nine:

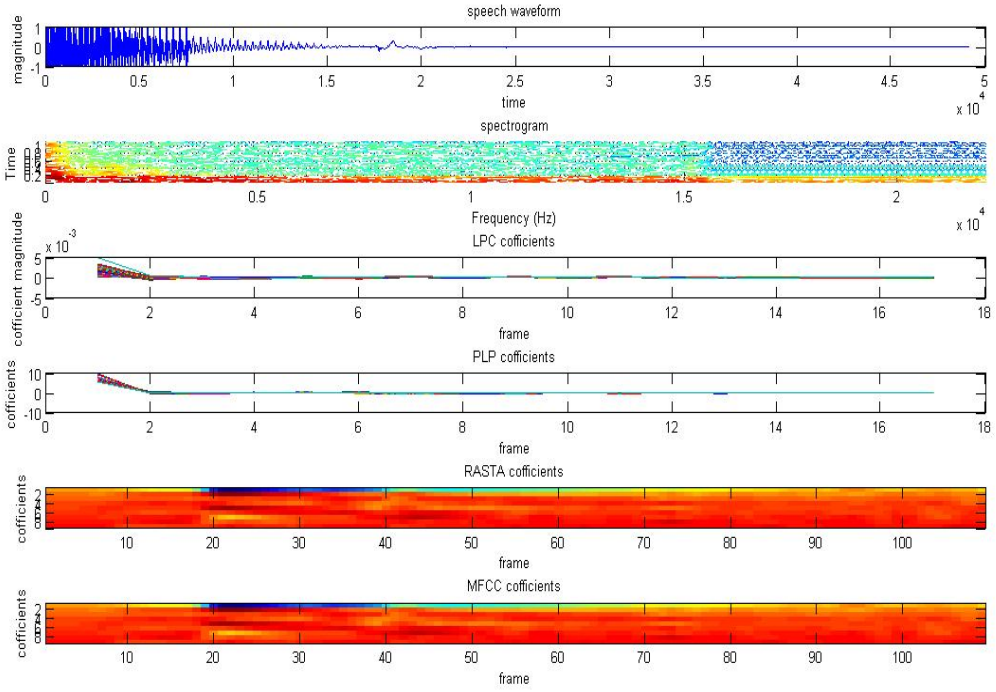


Fig 6.1.4.9 Coefficients for nine

An important conclusion that we can make from the last set of experiments is that one of the main reasons for the need of large training databases for LPC based analysis (without _ltering) , is the large difference between the different telephone lines, which is reected in a difference in spectral distortion.

6.2 MFCC coefficient analysis selection

Out of all the different options available for feature extraction we selected the the MFCC coefficients as in the MFC, the frequency bands are positioned logarithmically (on the mel scale) which approximates the human auditory system's response more closely than the linearly spaced frequency bands obtained directly from the FFT (Fast Fourier Transform) or DCT (Discrete

Cosine Transform). This can allow for better data processing. This feature of MFCC can be analysed by a matlab programme which takes in a speech waveform converts it into the MFCC coffiecients and then reconstructs the waveform from the MFCC and thus compare the power spectra of the original sound and the reconstructed sound.

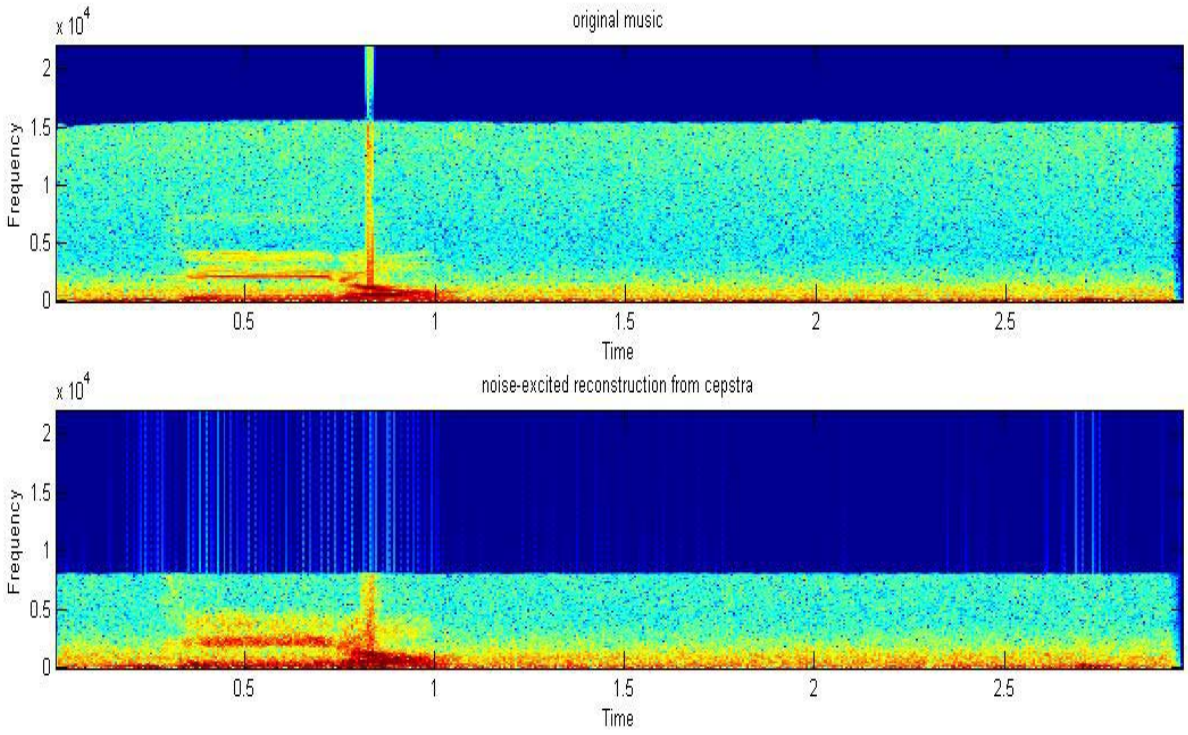


Fig 6.2 MFCC co-efficient analysis

The original sound and the reconstructed sound can also be played and the difference can be marked.

6.3 Distance calculation

After extracting the feature vector the next step which is important in speech recognition is distance calculation between the feature vectors which were calculated by the last step. Here we have analysed the two most prominent method of distance measure which are Euclidean distance

and the itakura-saito distortion measure (likelihood distortion measure). Here we have taken three wave form two similar and one different and shown how method compare the distance.

Take two different versions of one and one five as input.

6.3.1 Euclidean distance

```
>> [d1,sr1] = wavread('one.wav');
>> [d2,sr2] = wavread('one1.wav');
>> [d3,sr3] = wavread('five.wav');
>> y1 = lpcauto(d1,20);
>> y2 = lpcauto(d2,20);
>> y3 = lpcauto(d3,20);
>> y1 = y1';
>> y2 = y2';
>> y3 = y3';
>> b = disteusq(y1,y2,'d');
>> subplot(211)
>> plot(b)
>> title('distance between one and onenew')
>> b = disteusq(y1,y3,'d');
>> subplot(212)
>> plot(b)
>> title('distance between one and five')
```

Output:

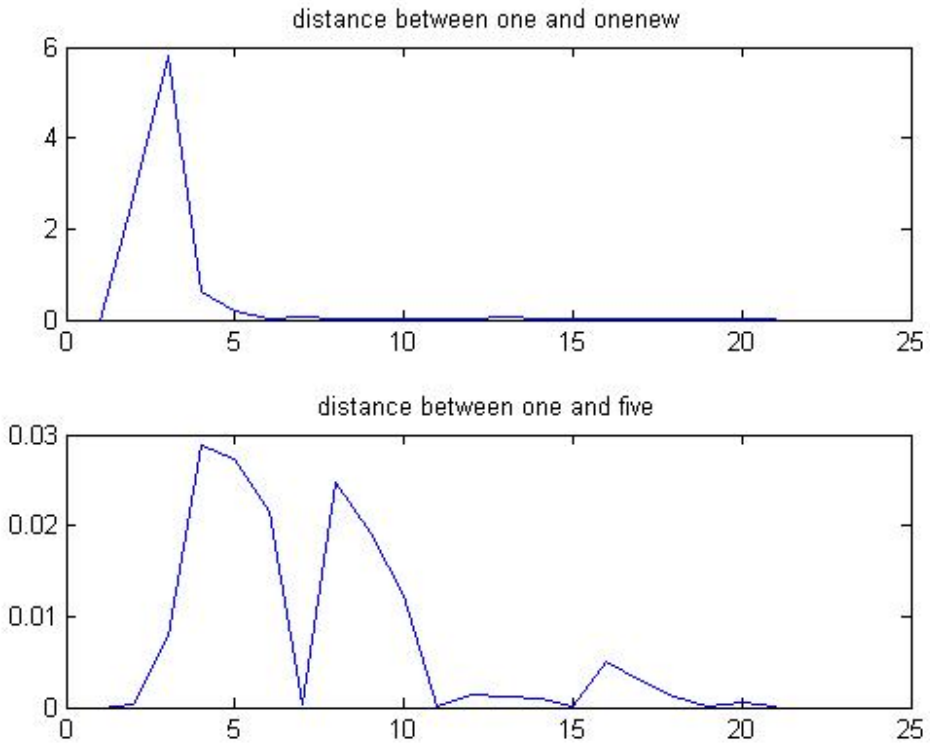


Fig 6.3.1 Euclidian distance

As it can be easily seen that the distance between one and one new is observably very less that one and five.

6.3.2 itakura-saito distance:

```
>> [d1,sr1] = wavread('one.wav');
>> [d2,sr2] = wavread('one1.wav');
>> [d3,sr3] = wavread('five.wav');
>> y1 = lpcauto(d1,20);
>> y2 = lpcauto(d2,20);
>> y3 = lpcauto(d3,20);
>> y1 = y1';
```

```
>> y2 = y2';

>> y3 = y3';

>> b = distitar(y1,y2,'d');

>> subplot(211)

>> plot(b)

>> title('distance between one and onenew')

>> b = distitar(y1,y3,'d');

>> subplot(212)

>> plot(b)

>> title('distance between one and five')
```

Output:

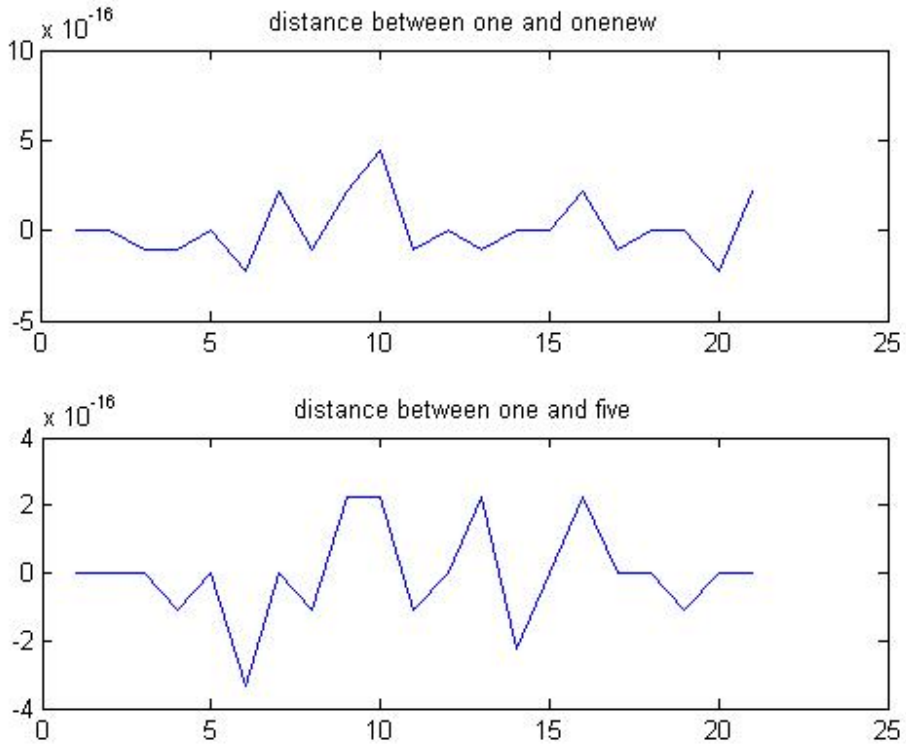


Fig6.3.2 itakura-saito distance

Thus it can be easily seen that even though itakura-saito distance is a very good form of distance measure its performance for the case of isolated word recognition with very little database is very poor. Thus we have decided to use Euclidean distance for our purpose.

6.4 Dynamic Time Warping

One of the difficulties in speech recognition is that although different recordings of the same words may include more or less the same sounds in the same order, the precise timing - the durations of each sub word within the word - will not match. As a result, efforts to recognize words by matching them to templates will give inaccurate results if there is no temporal alignment.

Although it has been largely superseded by hidden Markov models, early speech recognizers used a dynamic-programming technique called Dynamic Time Warping (DTW) to accommodate differences in timing between sample words and templates. The basic principle is to allow a range of 'steps' in the space of (time frames in sample, time frames in template) and to find the path through that space that maximizes the local match between the aligned time frames, subject to the constraints implicit in the allowable steps. As the duration of speaking for different persons are different DTW is highly unavoidable. The most common algorithm used for this purpose is dynamic programming. Here we bring an matlab program to calculate the DTW for two given signal

the input signal is two different versions of word ‘one’

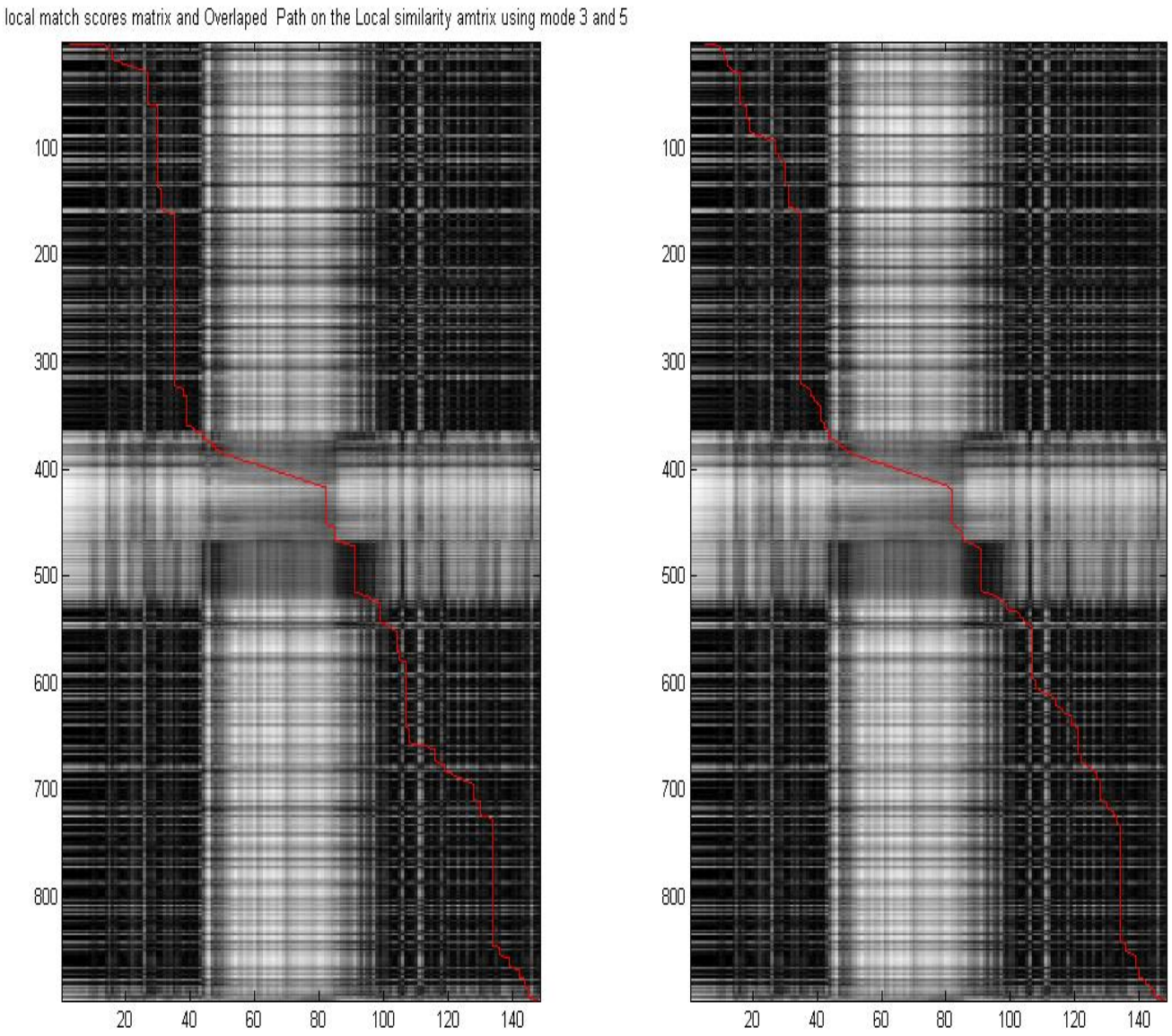


Fig 6.4 dynamic time warpping

6.5 An overall program

After analysing the different parts of the speech recognition analysis here we try to present a small program which does two tasks. The first task is to produce a data base of templates for once spoken wards for example ‘zero’ to ‘nine’. This is known as the tanning of the recogniser. The next task is to recognise. The MFCC feature coefficient is used here for reasons stated earlier Euclidean distance is used to measure the distance between the feature vectors. Here we first give the process of training.

You will get one second to say each word.

Hit enter and say immediately ' zero':

Hit enter and say immediately ' one':

Hit enter and say immediately ' two':

Hit enter and say immediately 'three':

Hit enter and say immediately ' four':

Hit enter and say immediately ' five':

Hit enter and say immediately ' six':

Hit enter and say immediately 'seven':

Hit enter and say immediately 'eight':

Hit enter and say immediately ' nine':

After this we have to run the recognition programme which works as shown:

You will get one second to say your word.

Hit enter and say your word immediately:

You said: eight

The accuracy of the above code is around 60% which is acceptable for a simple code given.

CHAPTER 7

APPLICATION

Automatic Speech Recognition Systems are being used in many applications such as.

• speech to text conversion using large vocabulary speech recognition systems.

• Message typing in mobile phones.

• Time and Attendance Systems

• Access Control Systems

• Telephone-Banking/Booking

• Security control for confidential information Forensic purposes

• Voice command and control

• Voice dialing in hands-free devices

• Credit card validation or personal identification number (PIN) entry in security systems

CHAPTER 8

CONCLUSION

In this project we at first calculated the different type of feature vectors such as LPC,PLP,RASTA,and MFCC .after performing such operations we analysed MFCC in particular, and selected it it as the preferred mode of feature vector coding because they follow the human ear's response to the sound signals. we also found different methods of distance measurement and compared them and concluded that equilidean distance measyre is a preferd one when the template database of sound is very low.we also performed a quick analysis of dynamic time wrapping algorithm and found the least path between two sounds. Then we designed a small model by writing a simple code which was able to recognise small set of isolated words.

The performance of this model is limited by a single template generated by the training programme, as it does not incorporate training algorithm of any short. The performance factor can be optimized busing high quality audio devices in a noise free environment. There is a possibility that thespeech can be recorded and can be used in place of the original speaker .This would not be a problem in our case because the MFCCs of the original speech signal and the recorded signal are different. Finally I conclude that although the project has certain limitations, its performance and efficiency have outshined these limitations at large.

CHAPTER 9

FUTURE LINE OF ACTION

In our Future work we are planning to further our work in many fields of speech recognition. some of those are mentioned below:-

- Implementation of end point detection.
- Implementation of K – Mean training algorithm for template production
- Create Vector Quantizer Code book for pattern recognition approach
- Implimentation of Hidden Markof Model (HMM) for recognition process which is a statistical approach of speech recognition.
- Formulate speaker independent isolated word recognition
- Implementation issues in connected word recognition.
- Implementation of continuous speech recognition

LIST OF MATLAB PROGRAM USED

- **melfcc.m** - main function for calculating PLP and MFCCs from sound waveforms, supports many options - including Bark scaling (i.e. not just Mel! but cannot do rasta).
- **invmelfcc.m** - main function for inverting back from cepstral coefficients to spectrograms and (noise-excited) waveforms, options exactly match melfcc (to invert that processing).
- **rastaplp.m** - the original main routine to convert waveform data into a sequence of feature frames. Outputs are both cepstra and spectra features, and options allow for selection of RASTA, PLP, both, or neither. (Cannot do Mel-scaling).
- **powspec.m** - calculate the short-time power spectrum, basically a wrapper around Matlab's specgram.
- **audspec.m** - map the power spectrum to an auditory frequency axis, by combining FFT bins into equally-spaced intervals on the Bark axis (or one approximation of it).
- **fft2barkmx.m** - function to create the weight matrix that maps FFT bin magnitudes to the Bark frequency axis, used by audspec.m.
- **fft2melmx.m** - generates a matrix of weights to convert FFT magnitudes into Mel bands, just like fft2barkmx above.
- **rastafilt.m** - filter each frequency band (now in terms of log energy) with the RASTA filter.
- **postaud.m** - fix-up the auditory spectrum with equal-loudness weighting and cube-root compression.
- **dolpc.m** - convert the auditory spectra directly to LPC coefficients via Levinson-Durbin.
- **lpc2cep.m** - convert LPC coefficients directly to cepstral values.

- **lpc2spec.m** - convert LPC coefficients back into spectra by sampling the z-plane.
- **spec2cep.m** - calculate cepstra by taking the DCT/DFT of the log of a set of spectra.
- **hz2bark.m** - convert frequency in Hz to the auditory Bark scale.
- **bark2hz.m** - convert back from Bark units to frequency in Hz.
- **hz2mel.m** - convert frequency in Hz to the auditory Mel scale (either Slaney's or HTK mapping).
- **mel2hz.m** - convert back from Mel units to frequency in Hz.
- **lifter.m** - apply (or remove) weighting from cepstral dimensions.
- **deltas.m** - calculate delta features over a limited window, just like feacalc/calc_deltas etc.
- **process_options.m** - Mark Paskin's utility to parse long 'name', value pair lists (which I found out about through Kevin Murphy's KPMtools), used by melfcc.m.
- **cep2spec.m** - inverse of spec2cep, undoes the DCT.
- **invpowspec.m** - invert powspec.m i.e. go back from an STFT magnitude to a (noise-excited) time waveform.
- **ispecgram.m** - precisely invert the short-time Fourier transform performed by specgram, taking the same argument (but fudges inverting the window at the moment).
- **invaudspec.m** - invert audspec i.e. expand the condensed, nonlinear frequency axis to the full FFT detail. Intrinsically lossy, but does its best.
- **invpostaud.m** - undo the weighting and compression of postaud, mostly lossless except the very edge bands are lost.

- **disteusq -** calculates the squared euclidean distance between all pairs of rows of two matrices.

- **distitar -** calculates the Itakura spectral distances between sets of AR coefficients.
- **distitpf -** calculates the Itakura spectral distances between power spectra.
- **distisar -** calculates the Itakura-Saito spectral distances between sets of AR coefficients.
- **distispf -** calculates the Itakura-Saito spectral distances between power spectra.

- **simmx.m** - a utility to calculate the full local-match matrix i.e. calculating the distance between every pair of frames from the sample and template signals.
- **dp.m** - implementation of the simple dynamic programming algorithm that allows three steps - (1,1), (0,1) and (1,0) - with equal weights.
- **dp2.m** - experimental alternative version that allows 5 steps - (1,1), (0,1), (1,0), (1,2), and (2,1) - with different weights to prefer sloping paths but without a hard limit on regions in which matches are found. Seems to work better, but not much tested. Syntax etc. the same as dp.m.
- **training** – creates a database of templates for recognition.
- **recognition** – recognises digits from ‘zero’ to ‘nine’.

REFERENCES

[1] Claudio Becchetti and Lucio Prina Ricotti, “Speech Recognition”, Chichester: John Wiley & Sons, 2004.

[2] John G. Proakis and Dimitris G. Manolakis, “Digital Signal Processing”, New Delhi: Prentice Hall of India. 2002.

[3] Rudra Pratap. Getting Started with MATLAB 7. New Delhi: Oxford University Press, 2006

[4] S. Furui, “Speaker independent isolated word recognition using dynamic features of speech spectrum”, IEEE Transactions on Acoustic, Speech, Signal Processing, Vol.34, issue 1, Feb 1986, pp. 52-59.

[5] Lawrence Rabiner and Biing - Hwang Juang , “Fundamentals of speech Recognition”, New Delhi : pearson Education, 2003.