# DISTRIBUTED SPEECH RECOGNITION

BY

WIRA GUNAWAN

B.S., The Ohio State University, 1998

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2000

Urbana, Illinois

## ABSTRACT

Speech is a natural mode of communication for human beings and speech recognition is an application that enables the interaction between human and machine via voice. As the cost of software and hardware needed to do recognition decreases, automatic speech recognition (ASR) has entered the consumer product mainstream. A particularly interesting application is wireless speech recognition, which is the integration of ASR technology into wireless applications. Utilizing an ASR system, people can interact via their cell phone using voice, thereby freeing their hands and eyes for other tasks. One aspect of speech recognition useful for wireless applications is digit recognition. In this paper, we perform speaker–independent isolated digit recognition using PLP (perceptual linear predictive) analysis and a DTW (dynamic time warping) algorithm. The effect of quantization of speech recognition features on recognition accuracy is examined to determine the lowest bit rate possible while maintaining high quality performance.

To my family.  Without their love and support, none of this would have happened.

# ACKNOWLEDGMENTS

I would like to thank Professor Mark Hasegawa-Johnson for giving me the opportunity to do research in speech recognition and for his guidance in developing this thesis.

# TABLE OF CONTENTS

Page

# 1. INTRODUCTION

Automatic speech recognition (ASR) has been a subject of research since the 1940s. Many people have envisioned the interaction between human and machine using voice, as can be seen in many science-fiction movies. Speech recognition and speech synthesis are two main areas of voice processing that enable the interaction between human and machine. In the early days, voice processing technologies such as speech recognition could only be found in research laboratories, but now these technologies have found their way into a variety of commercial applications.

Early ASR systems could only do isolated digit recognition for a single speaker [1]. Later generations of ASR were able to do much more complicated tasks such as connected word recognition. The most recent ASR system, which is available commercially, is able to recognize continuous speech with proper training. The popularity of ASR nowadays is fueled by several factors such as the advances in digital signal processing technology, the substantial decrease of computation cost and memory, and the exponential increase of computing power found in both general-purpose and special-purpose processors.

These days we can see that speech recognition has penetrated many areas in everyday life. People can go to computer stores and buy a high-performance dictation system, which allows users to use voice as input instead of a keyboard. Some companies have started to use speech recognition technology to enable customers to obtain weather information, stock quotes, business news, sports news, traffic reports, and local restaurant guides. Telecommunications, a field that enjoys remarkable growth with the booming of the wireless and Internet industries, has also helped propel speech recognition technology into consumer applications.

One application of ASR in the wireless industry is to provide hands-free, eyes-free interaction between user and cellular telephone via voice. This application is particularly useful when people are driving or doing other tasks that require using the hands and eyes.

Some countries even prohibit the use of cellular phones, a prohibition that will no longer be necessary with voice-activated dialing. Cellular phones can also be programmed to store personal phone directories and to perform advanced tasks such as caller ID, call waiting, last number dialed, and call forwarding. Programming a cell phone via speech commands gives a more efficient and smoother interaction between user and cell phone than using a tiny keypad in the cell phone. Hence, as the use of cellular phone becomes more widespread, wireless speech recognition emerges as a research field of interest to industry and academe.

# 2. BACKGROUND

## 2.1 Previous Work

As mentioned previously, the tremendous growth in the wireless communication industry is one of the factors that promotes the realization of speech recognition in digital communication networks. More and more people in industry and academe are now doing research toward the design of a high quality wireless speech recognition system. Kim and Cox [2] state that the research work on a front-end design for wireless speech recognition can be classified into three categories. The first category is shown in Figure 1. This system uses the synthesized speech from a speech decoder as an input to an ASR system.

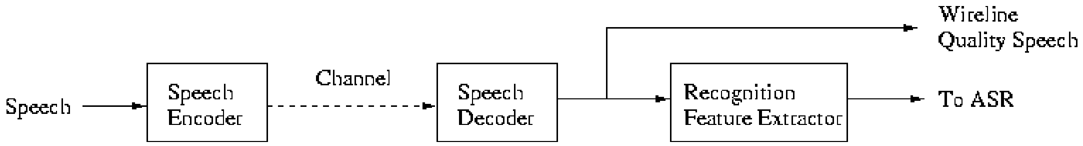


Figure 1: A conventional approach in which the output of the speech decoder serves as the input to the ASR system (after Kim and Cox [2]).

Works on the above system have been reported in [2]-[7] for IS-641, GSM (Global System for Mobile Communications), CELP (code-excited linear prediction), LD-CELP (low delay CELP), and QCELP (Qualcomm CELP) environments. Depending on the vocabulary size to be recognized and the adversity condition of the test utterance, the word error rate (WER) ranges from 0.5% to 7% and the word recognition accuracy (WRA) ranges from 83% to 99%. WER and WRA are two metrics commonly used to measure speech recognition performance. It has been shown that the performance of this system is lower than that for wireline ASR [2]-[6]. Wireline ASR is an ASR that is trained and tested using standard wireline quality speech, which is speech sampled at 8 kHz and coded using μ-law or A-law companding. Kim and Cox show that, given the same database and condition, the word accuracy of wireline ASR is 96.17%, whereas the word accuracy of the above approach is 94.75%. Experiments by Choi et al. [3] also show a similar trend in which recognition rate of the above approach is 3% less than that of wireline ASR. The degradation in performance is due to the spectral distortion introduced by speech coder and decoder.

In the second approach, a front-end processor encodes the recognition feature instead of original speech. Speech recognition parameters, the outputs of the encoder, are quantized and transmitted as illustrated in Figure 2.

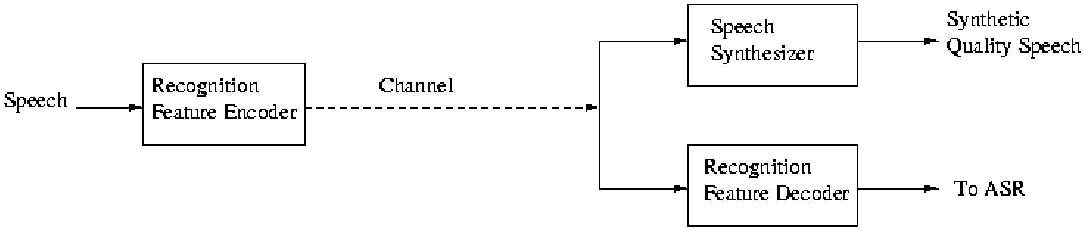


Figure 2: The speech recognition feature is encoded by the front-end processor, and the input to the ASR system is the decoded recognition feature (after Kim and Cox [2]).

This scheme yields recognition accuracy comparable to that of wireline ASR, as reported by Digalakis et al. [8]. Using MFCC (mel frequency-warped cepstral coefficients) as a parametric representation of the speech, they show that the required bit rate to achieve the recognition performance close to that of wireline ASR is 2000 bits per second. They reported a WER of 6.55% and 6.63% for wireline ASR and second category ASR, respectively. However, because only speech recognition parameters are transmitted, much information contained in the original speech is lost. Consequently, the synthesized speech does not have high quality, whereas the first system is able to generate high quality speech in addition to having good speech recognition performance. The advantage of the second approach is that a lower bit rate can be achieved by only encoding the recognition features. Another similar study using this scheme is reported by Tsakalidis et al. [9].

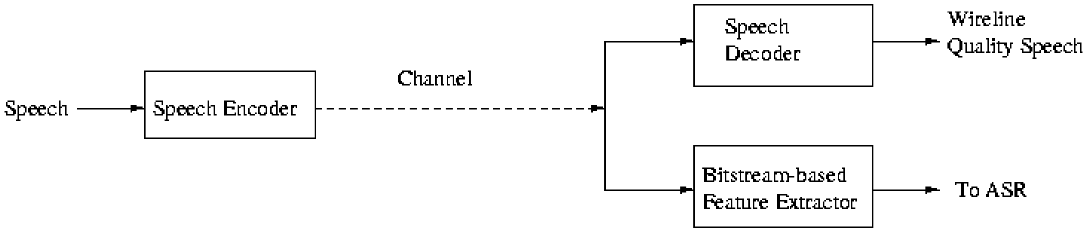


Figure 3: The speech recognition feature is extracted from the bitstream of the speech coder (after Kim and Cox [2]).

The third system, shown in Figure 3, extracts speech recognition parameters from the bitstream of a speech coder. According to Kim and Cox, this approach is able to generate high quality speech and has better recognition accuracy than the first category system. Their experiments yield recognition accuracy of 95.16% and 95.81% for category one and category three systems, respectively. Experiments by Choi et al. [10] confirm this view in which the recognition rate of the first and third systems are 82.5% and 84.1%, respectively. However, the recognition accuracy of this approach is still lower than that of wireline ASR, which is also shown in [2] and [10]. Table 1 on page 6 shows the results of experiments using the three systems.

Most of the recent research in speech recognition employs a recognition algorithm called the hidden Markov model (HMM). In this work, an older technique, dynamic time warping (DTW), is used for speaker-independent isolated digit recognition. A closely related work using this algorithm was done by Sakoe and Chiba [12], who described an optimum dynamic programming (DP) based time-normalization algorithm for spoken word recognition. Another important work using dynamic time warping algorithm is that of Rabiner et al. [13].

In this work, perceptual linear predictive (PLP) coefficients are used as parametric representations of acoustic data. PLP analysis was first proposed by Hermansky [14], and he also performed speaker-independent isolated digit ASR experiments using the DTW algorithm. Hermansky showed that PLP is a better representation of the linguistic information in speech than is conventional LP (linear predictive) analysis. In addition, he reported that a PLP-based recognition system consistently performed better than an LP-based system by comparing WRA of ASR systems using 14th-order LP analysis and 5th-order PLP analysis. For example, with two templates per word, a PLP-based system's accuracy is 92%, whereas an LP-based system's accuracy is 90%.

Table 1: The performance of different categories of recognition system.

| Reference | Class[1] | Acoustic features[2] | Mode[3] | Coder[4] | Algorithm[5] | Vocabulary | Metric (%) |
|---|---|---|---|---|---|---|---|
| [2] | WL | LPCC | CW | - | GMHMM | Digits | WRA = 96.17 |
| [2] | 1 | LPCC | CW | IS-641 | GMHMM | Digits | WRA = 95.16 |
| [2] | 3 | LPCC | CW | IS-641 | GMHMM | Digits | WRA = 95.81 |
| [2] | 3 | LPCC, ACG, FCG | CW | IS-641 | GMHMM | Digits | WRA = 95.96 |
| [3] | WL | MFCC | CW | QCELP | DHMM | Digits | WRA = 87.00 |
| [3] | 1 | MFCC | CW | QCELP | DHMM | Digits | WRA = 83.60 |
| [4] | WL | MFCC | IW | - | GMHMM | Digits | WRA = 99.66 |
| [4] | 1 | MFCC | IW | FR-GSM | GMHMM | Digits | WRA = 99.53 |
| [4] | 3 | MFCC | IW | FR-GSM | GMHMM | Digits | WRA = 99.25 |
| [5] | WL | RN-LFCC | IW, CW | - | GMHMM | 43 words, 26 letters, and digits | WRA = 90.00-99.50 |
| [5] | 1 | RN-LFCC | IW, CW | FR-GSM, HR-GSM | GMHMM | 43 words, 26 letters, and digits | WRA = 80.00-99.30 |
| [6] | WL | LPCC | IW | - | GMHMM | 23 words | WRA = 96.61 |
| [6] | 1 | LPCC | IW | CELP, LD-CELP, FR-GSM | GMHMM | 23 words | WRA = 91.95-94.52 |
| [7] | 1 | MFCC | NA | GSM | PHIL90 | Digits and 26 words | WER = 0.5-5.0 |
| [8] | WL | MFCC | CS | - | GMHMM | ATIS[6] | WER = 6.63 |
| [8] | 2 | MFCC | CS | - | GMHMM | ATIS | WER = 6.63 |
| [9] | 2 | MFCC | CS | - | DMHMM | ATIS | WER = 6.25-6.60 |
| [10] | WL | LSP | IW | QCELP | DHMM | 26 words | WRA = 90.20 |
| [10] | 1 | LSP | IW | QCELP | DHMM | 26 words | WRA = 82.50 |
| [10] | 3 | LSP | IW | QCELP | DHMM | 26 words | WRA = 84.10 |
| [10] | WL | MPCEP | IW | QCELP | DHMM | 26 words | WRA = 94.70 |
| [10] | 1 | MPCEP | IW | QCELP | DHMM | 26 words | WRA = 89.80 |
| [10] | 3 | MPCEP | IW | QCELP | DHMM | 26 words | WRA = 90.60 |

[1] Classes 1, 2, 3 correspond to the systems in Figure 1, 2, and 3 respectively. Class WL corresponds to wireline ASR. [2] LPCC = LPC cepstral coefficients, ACG = adaptive codebook gain, FCG = fixed codebook gain, MFCC = mel-frequency cepstrum coefficients, RN-LFCC = root-normalized linear frequency cepstrum coefficients, LSP = line spectral pairs, MPCEP = mel-scale pseudo-cepstrum coefficients. [3] Mode corresponds to the speaking style, which is either IW (isolated words), CW (connected words), CS (continuous speech), or NA (data not available). [4] Speech coder is one of the following: IS-641 is a 7.4 kb/s ACELP (algebraic CELP) coder, QCELP = Qualcomm CELP (13 kb/s), FR-GSM = full rate GSM (13 kb/s), HR-GSM = half rate GSM (5.6 kb/s), LD-CELP = low delay CELP (16 kb/s). [5] Algorithm corresponds to recognition algorithm. Algorithm is one of the following: DHMM = discrete-density HMM, DMHMM = discrete-mixture HMM, GMHMM = Gaussian-mixture HMM. PHIL90 is a speech recognition system developed at France Telecom/CNET (Centre National des Telecommunications). [6] Refer to [11] for more information on the ATIS (air travel information system) domain.

## 2.2 Problem Statement

Speech recognition is computationally expensive, requiring a large amount of memory and processing power. Implementing a complete speech recognition algorithm in a cell phone would put a big burden on the cell phone's hardware and software requirements, which could cause the cost of cell phones to increase dramatically. To overcome this problem, the speech recognition system is distributed over the cell phone and base station. In the cellular phone, the front-end processor calculates, quantizes, and encodes the speech recognition feature. Then the encoded coefficients are transmitted to the base station, and the decoder extracts the coefficients. These decoded coefficients, which are PLP coefficients, are converted into cepstral coefficients, and the speech recognition algorithm (dynamic time warping) is run using cepstral coefficients as input.

The focus of this work is speaker-independent isolated digit recognition, with 11 English digits (zero to nine and the word "oh") serving as valid inputs. First, speech recognition is implemented without any quantization, as shown in Figure 4, which means that the DTW recognizer has a perfect input (no distortion).

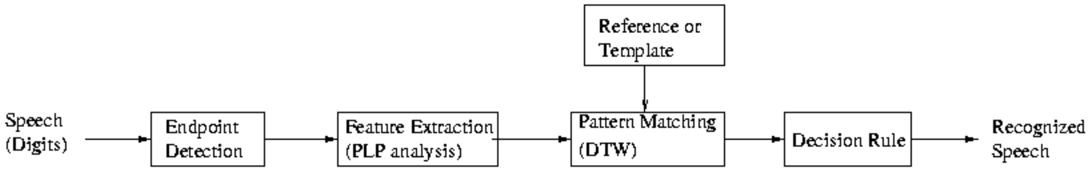


Figure 4: ASR system without quantization

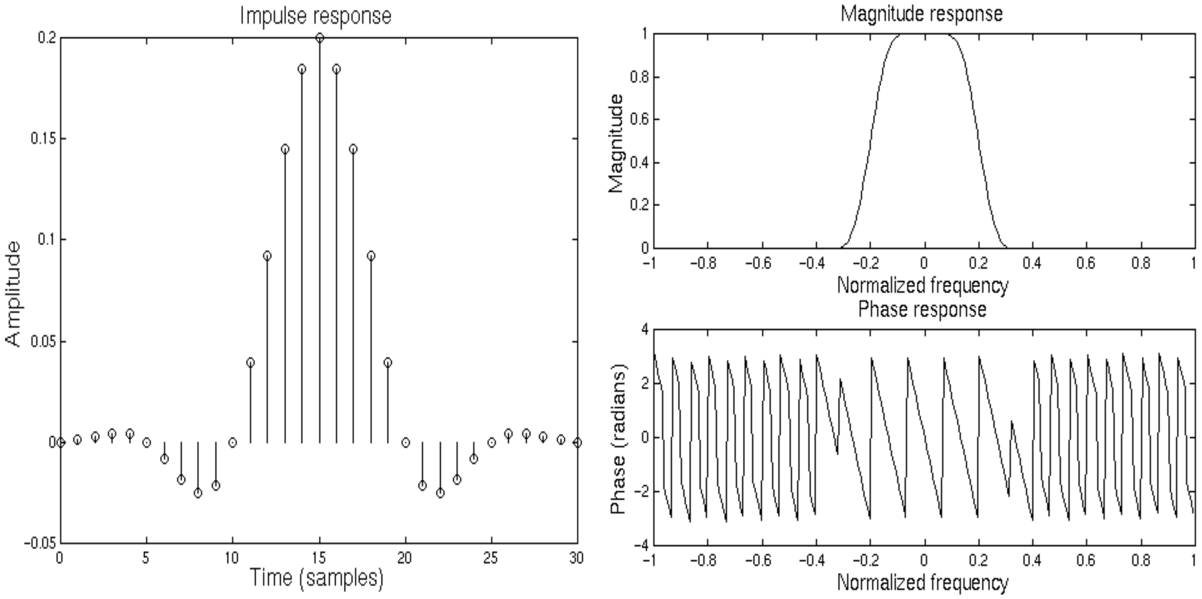
Then we examine the quantization effect on the speech recognition performance. PLP coefficients are converted into LSP (line spectral pairs) before being quantized using vector quantization. Codebooks, designed using the LBG algorithm [15], are varied to explore the trade-off between bit rate and performance, as the performance tends to be lower while the bit rate is decreased. Furthermore, the number of templates per word is also varied.

Next, the speech features are down-sampled by a factor of two to lower the transmission bit rate, and the system implements linear interpolation upon down-sampled speech features before running recognition algorithm. Then the recognition accuracy is calculated to see how the performance is affected. The last system implements similar system without linear interpolation. It is expected that a down-sampled system with linear interpolation will outperform its counterpart without interpolation.

# 3. SYSTEM DESCRIPTION

## 3.1 Preprocessing

Input speech waveforms are taken from the TIDIGITS database (Linguistic Data Consortium, 1990). The input speech data has sampling rate of 20 kHz, and telephone speech is normally sampled at 8 kHz. Consequently, the sampling rate has to be changed from 20 kHz to 8 kHz. This is done by up-sampling by a factor of two and down-sampling by a factor of five. To avoid aliasing, a low-pass FIR filter with cutoff $\pi/5$ and order 30 is applied to eliminate frequency contents higher than 4 kHz (Nyquist rate). Figure 5 shows the impulse response and frequency response of the filter, which was designed using a Hamming window.



(a) (b)

Figure 5: A 30th-order low-pass FIR filter: (a) impulse response, (b) magnitude response and phase response.

An important problem in speech recognition is to determine where the speech begins and ends. This is known as the speech endpoint detection problem. Wilpon et al. [16] showed in their multispeaker digit recognition experiment that even a slight error in endpoint detection could result in significant degradation in recognition accuracy. For example, in their experiment, recognition accuracy decreases by 10% because the endpoints are inaccurately identified by approximately 120 ms. Therefore, a good endpoint detection algorithm is necessary in the speech recognition system. The endpoint detection algorithm used in this work was proposed by Rabiner et al. [17]. The algorithm uses energy and zero crossing rate measurements to determine beginning and end points. However, to speed up the running time, only speech energy is used to detect the endpoints in this work. Because the speech data is recorded in a high signal-to-noise ratio acoustic environment, the detection algorithm is expected to work well for most cases. During the experiment, incorrect endpoints are corrected manually to better investigate the recognition accuracy.

The speech energy $E(n)$ is computed by summing the magnitudes of 10 ms of speech centered on the measurement interval. If we denote the speech signal by $x(n)$, then

$$E(n) = \sum_{i=-40}^{40} \left| x(n+i) \right|$$

To detect the endpoint location, first we need to know the statistics of the background silence. The average energy of the background silence is computed by averaging the energy during the first 100 ms of the signal. Computation of the energy of the background silence assumes that there is no speech during the first 100 ms. Second, the peak energy of the entire speech file is computed by finding the maximum of $E(n)$. Let *IMX* be the peak energy and *IMN* be the silence energy; then two thresholds, *ITL* and *ITU*, are calculated based on the following equations:

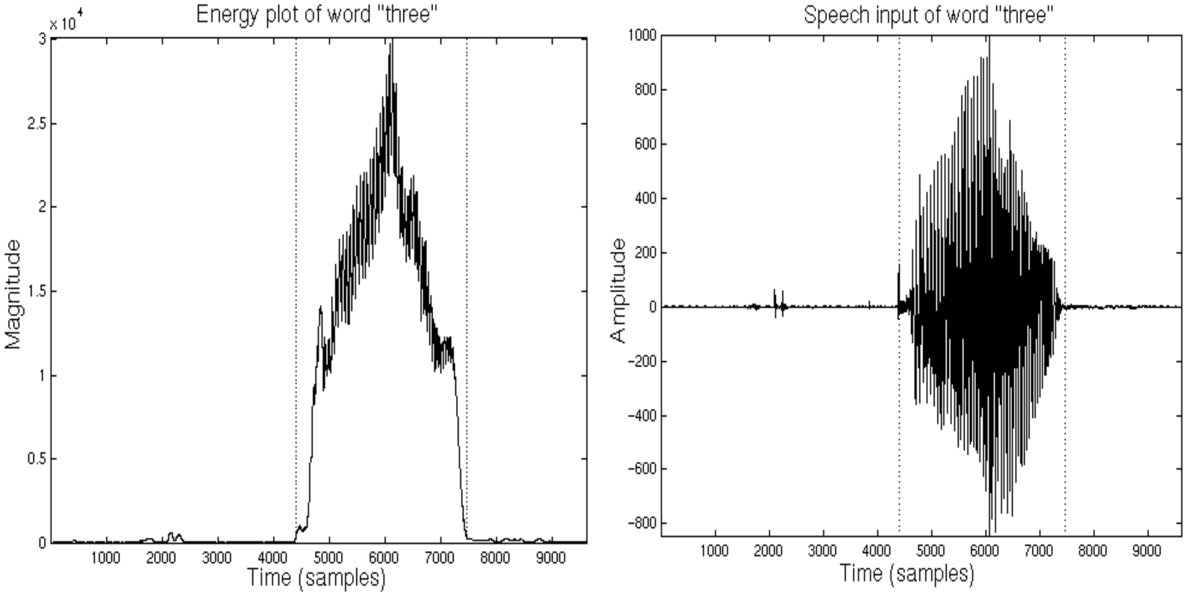
$$I1 = 0.03*(IMX\text{-}IMN)+ IMN$$

$$I2 = 4*IMN$$

$$ITL = \mathrm{MIN}(\mathrm{I}1, I2)$$

$$ITU = 5*ITL$$

To find the beginning point, the algorithm starts by locating the point where the signal energy exceeds the upper threshold *ITU.* From this point, the algorithm searches backwards to find a point at which the energy falls below the lower threshold *ITL* for the first time. The ending point is found in a similar way. Figure 6 shows the energy plot of word “three” and the result of the endpoint detection algorithm, respectively.



(a) (b)

Figure 6: Outputs of endpoint detection algorithm: (a) Energy plot of speech utterance “three” along with the beginning and the ending marks (shown in dotted lines). (b) The original speech signal is plotted with the beginning and the ending marks.

## 3.2 Perceptual Linear Predictive (PLP) Analysis

### 3.2.1 Overview

PLP analysis was proposed by Hynek Hermansky in 1989. PLP analysis is similar to linear predictive coding (LPC), which is a widely known technique in speech coding, except the PLP technique also uses three concepts from the psychophysics of hearing. These three concepts are the critical-band spectral resolution, equal-loudness curve, and intensity-loudness power law. Figure 7 below shows necessary steps to implement the PLP method.

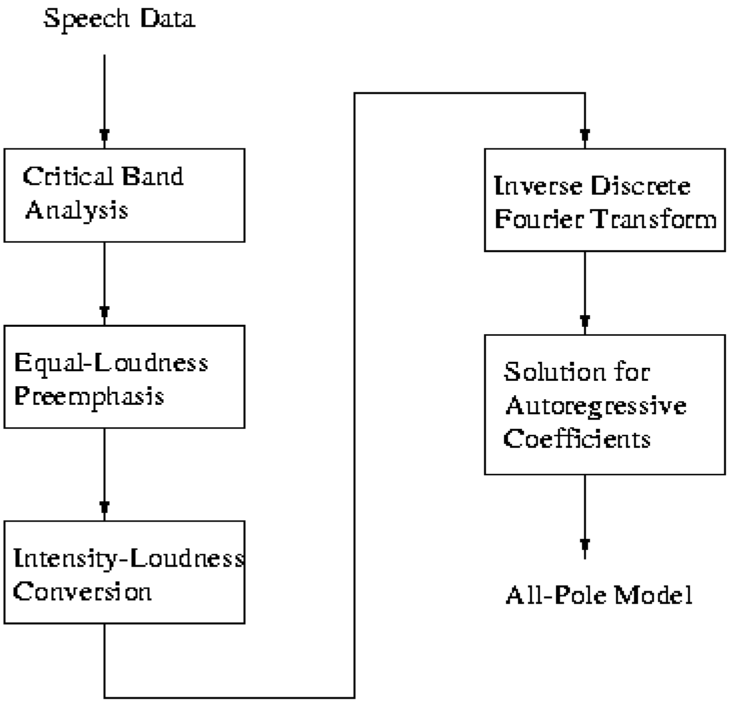

Speech Data
Critical Band Analysis
Equal-Loudness Preemphasis
Intensity-Loudness Conversion
Inverse Discrete Fourier Transform
Solution for Autoregressive Coefficients
All-Pole Model

Figure 7: Block diagram of PLP analysis (after Hermansky [14]).

Both LPC and PLP use the autoregressive all-pole model to estimate the short-term power spectrum of speech. However, as pointed out by Hermansky, the LPC all-pole model is not consistent with human auditory perception because it does not consider the nonuniform frequency resolution and intensity resolution of hearing. PLP alleviates this problem by applying the all-pole model to the auditory spectrum. The auditory spectrum is designed to be an estimate of the mean rate of firing of auditory nerve fibers. The all-pole model encodes the frequencies and the bandwidths of the two most important peaks in the auditory spectrum, as suggested by the vowel perception model of Chistovich [18].

### 3.2.2 Spectral analysis

After the sampling rate conversion described in Section 3.1, the filtered speech data is blocked into overlapping frames of 240 samples (30 ms). The amount of overlap is 160 samples (20 ms); in other words, adjacent frames are separated by 80 samples (10 ms). Let $s(n)$ be the filtered speech data, $x_k(n)$ be the $k^{\text{th}}$ frame of speech, and $L$ be the total number of frames. Then

$$x_k(n) = s(80k+n), \qquad n=0,1,\ldots,239;\; k = 0,1,\ldots,L\text{-}1$$

A 30-ms asymmetric window is applied to each frame to minimize the signal discontinuities at the beginning and end of each frame by tapering the signal to zero or near zero. The first half of the window is half of a Hamming window, and the second half is a quarter period of a cosine function. If we denote the window by $w(n)$, then

$$w(n) = \begin{cases} 0.54 - 0.46\cos\left(\dfrac{2\pi n}{399}\right), & n=0,\ldots,199, \\ \cos\left(\dfrac{2\pi(n-200)}{159}\right), & n=200,\ldots,239 \end{cases}$$

This asymmetrical window was chosen because it is the window used by the CS-ACELP (conjugate-structure algebraic code-excited linear predictive) speech coder for LP analysis [19].

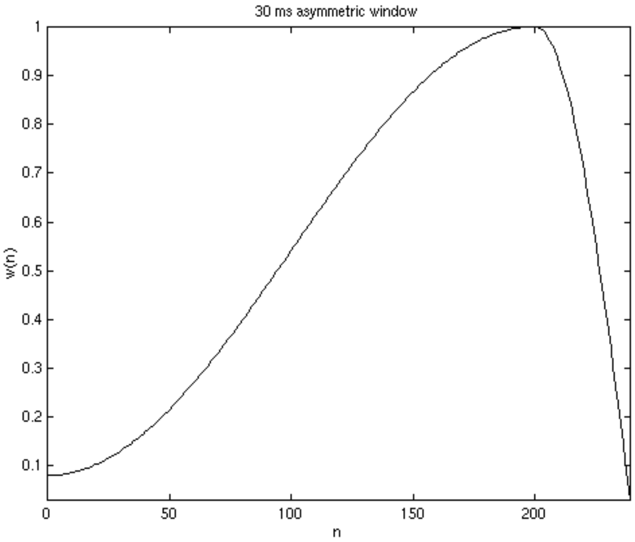


Figure 8: Asymmetric window with 30 ms duration.

Figure 8 shows the plot of this window. The asymmetrical shape is intended to reduce the look-ahead without compromising quality. Now the windowed frame becomes

$$s_k(\mathrm{n}) = x_k(n)w(n), \qquad n = 0,1,\ldots,239;\ k = 0,1,\ldots,L\text{-}1$$

Next, a 256-point fast Fourier transform (FFT) is used to transform 240 speech samples in every frame into the frequency domain by padding 16 zero-valued samples. Let $S_k(\omega)$ be the Fourier transform of $s_k(n)$. Then the short-term speech spectrum is obtained by squaring the real and imaginary components of $S_k(\omega)$ and adding them, i.e.,

$$P(\omega) = \mathrm{Re}[S_k(\omega)]^2 + \mathrm{Im}[S_k(\omega)]^2$$

where $\omega$ is the angular frequency in rad/s. Note that the power spectrum is essentially the squared-magnitude of complex number $S_k(\omega)$.

### 3.2.3 Critical band analysis

A critical band is the smallest bandwidth such that the loudness of a sound is perceived to be different. At a constant sound pressure, several tones lying within a critical bandwidth give the same level of perceived loudness as a single pure tone lying at the center of the band whose intensity equals the sum of the component tone intensities. If tones are separated by more than a critical bandwidth, their combination is perceived to become louder. The frequency scale in which a critical bandwidth is always one unit is called Bark. The power spectrum $P(\omega)$ is warped into Bark frequency $\Omega$ using the equation

$$\Omega(\omega) = 6 \ln\{\omega/1200\pi + [(\omega/1200\pi)^2+1]^{0.5}\}$$

The Bark-scaled power spectrum is convolved with the power spectrum of the critical band masking curve $\Psi(\Omega)$. This step simulates the frequency resolution of the ear in which $\Psi(\Omega)$ integrates all the loudness of the tones lying within a critical bandwidth into one loudness of equivalent single tone. The critical band curve is given by

$$\Psi(\Omega) = \begin{cases} 0 & \text{for } \Omega < -1.3, \\ 10^{2.5(\Omega+0.5)} & \text{for } -1.3 \leq \Omega \leq -0.5, \\ 1 & \text{for } -0.5 < \Omega < 0.5, \\ 10^{-(\Omega - 0.5)} & \text{for } 0.5 \leq \Omega \leq 2.5, \\ 0 & \text{for } \Omega > 2.5 \end{cases}$$

The convolution of $P(\omega)$ with masking curve $\Psi(\Omega)$ in effect reduces the spectral resolution of $\mathrm{P}(\omega)$. Then the output of convolution is down-sampled by sampling it in 1-Bark intervals at integer points (1,…,15) to cover the frequency from 0 to 15.575 Bark (0-4 kHz).

### 3.2.4 Equal-loudness preemphasis

The ear does not hear all frequencies with equal sensitivity, i.e., the perceived loudness is different at different frequencies. The human ear is most sensitive to frequencies between 500 Hz and 4 kHz. To simulate this property, the sampled convolution output is preemphasized by an approximation of the equal-loudness curve, which has the form

$$E(\omega) = \frac{(\omega^2 + 56.8x10^6)\omega^4}{(\omega^2 + 6.3x10^6)^2(\omega^2 + 0.38x10^9)}$$

### 3.2.5 Intensity-loudness power law

This model simulates another concept from the psychophysics of human hearing, that is, the nonlinear relationship between the perceived loudness and sound intensity. Let *I(Ω)* be the output of the equal-loudness preemphasis operation; then the power law approximates the intensity-loudness relationship by the following equation:

$$L(\Omega) = I(\Omega)^{0.33}$$

### 3.2.6 Autoregressive modeling

Autoregressive modeling is basically an LPC analysis that is based on the autocorrelation method. The perceptual autocorrelation coefficients $R_k(n)$ are obtained by applying a 32-point inverse fast Fourier transform (IFFT) to $L(\Omega)$. Hermansky showed in his experiment of isolated digit recognition that the optimal order of the linear predictor is five. To get the predictor coefficients $a_k(m)$, we need to solve the autocorrelation normal equation

$$R_k(n) = \sum_{m=1}^{5} a_k(m) R_k(|m-n|)$$

which can be written in matrix form

$$\begin{bmatrix} R_k(0) & R_k(1) & R_k(2) & R_k(3) & R_k(4) \\ R_k(1) & R_k(0) & R_k(1) & R_k(2) & R_k(3) \\ R_k(2) & R_k(1) & R_k(0) & R_k(1) & R_k(2) \\ R_k(3) & R_k(2) & R_k(1) & R_k(0) & R_k(1) \\ R_k(4) & R_k(3) & R_k(2) & R_k(1) & R_k(0) \end{bmatrix} \begin{bmatrix} a_k(1) \\ a_k(2) \\ a_k(3) \\ a_k(4) \\ a_k(5) \end{bmatrix} = \begin{bmatrix} R_k(1) \\ R_k(2) \\ R_k(3) \\ R_k(4) \\ R_k(5) \end{bmatrix}$$

and can be solved efficiently by using the Levinson-Durbin recursion.

### 3.2.7 LPC cepstrum coefficients

LPC cepstrum coefficients, which are also called cepstral coefficients, are more robust and reliable for speech recognition than LPC coefficients because cepstral coefficients have flat spectral sensitivity and low correlation. Therefore, PLP coefficients need to be converted into cepstral coefficients. As PLP is analogous to LPC, PLP coefficients can be converted into cepstral coefficients in the same way that LPC coefficients are converted into cepstral coefficients. Cepstral coefficients can be derived from PLP coefficients as follows:

$$c_k(n) = a_k(n) + \sum_{i=1}^{n-1} \left( \frac{i}{n} \right) c_k(i) a_k(n-i), \qquad 1 \le n \le 5$$

$$c_k(n) = \sum_{i=1}^{n-1} \left( \frac{i}{n} \right) c_k(i) a_k(n-i), \qquad n > 5$$

In this work, seven cepstral coefficients are calculated for each frame.

## 3.3 Dynamic Time Warping (DTW)

Dynamic time warping (DTW) is a nonlinear time-normalization algorithm for speech recognition based on dynamic programming. DTW works by comparing a parametric representation of the input speech to stored templates. The stored templates contain the parametric representation of the vocabulary words. As mentioned in the previous section, the parametric representation of speech used in this work is cepstral coefficients, which are derived from PLP coefficients. Pattern comparison is done by searching for the item in the templates that minimizes the distance between the reference pattern and cepstral coefficients of the input. Discussion of DTW in this section follows closely the explanation in [1].

3.3.1 Time alignment and normalization

A general problem in comparing two spectral sequences associated with speech is the variability in speaking rate and duration of the same speech utterance, even for the utterances of the same speaker. This problem is known as time alignment and normalization, and a general solution is to use a time warping function to eliminate the timing differences between two speech patterns. Let $X$ and $Y$ be the parametric representation of two speech inputs, i.e.,

$$X = \left(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{T_x}\right)$$
$$Y = \left(\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_{T_y}\right)$$

where $T_x$ and $T_y$ are the durations of $X$ and $Y$ (the number of frames associated with $X$ and $Y$), respectively. Figure 9 illustrates a typical warping function that maps $i_x$, the frame indices of $X$, into $i_y$, the frame indices of $Y$.

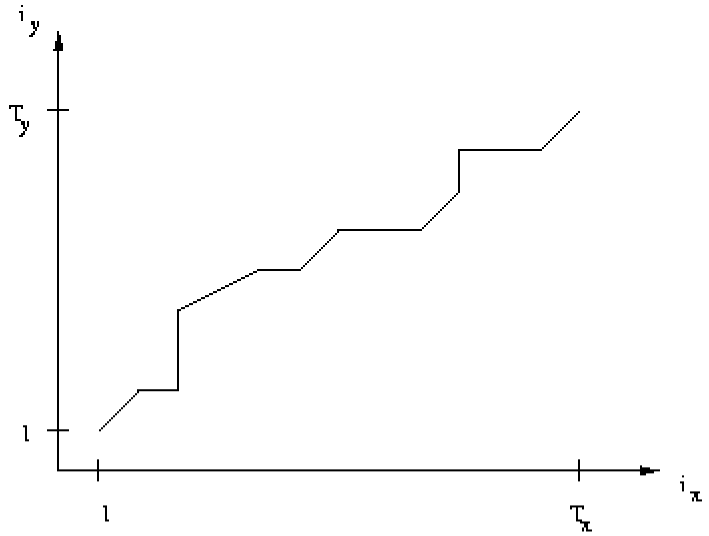


Figure 9: Time warping function between $i_x$ and $i_y$.

Note that if the two speech patterns have no time differences, the path of the warping function would be a diagonal line starting at point (1,1) and ending at point $(T_x,T_y)$. A general time alignment and normalization method maps the two indices $i_x$ and $i_y$ into a common time index *k* which is described by warping functions $\phi_x$ and $\phi_y$, i.e.,

$$i_x = \phi_x(k), \qquad k = 1,2,\ldots,T$$
$$i_y = \phi_y(k), \qquad k = 1,2,\ldots,T$$

The purpose of DTW is to find a warping function $\phi = (\phi_x, \phi_y)$ that minimizes the accumulated distortion over the entire utterance, i.e.,

$$d(X,Y) = \frac{1}{N} \min_{\phi} \sum_{k=1}^{T} d(\phi_x(k), \phi_y(k)) w(k)$$

where *d(X,Y)* is the dissimilarity measure, *w*(*k*) is a nonnegative path weighting coefficient, $d(\phi_x(k), \phi_y(k))$ is the short-time spectral distortion or distance between **x** and **y** at frame indices $\phi_x$ and $\phi_y$, respectively, and *N* is a path normalizing factor. To measure the dissimilarity between two spectral vectors, Euclidean distance is used as a distance metric. If **x** and **y** are two five-dimensional vectors, then Euclidean distance *d(***x**,**y**) is

$$d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{\mathrm{k}=1}^{5} |x_k - y_k|^2}$$

To minimize the accumulated distortion over the entire utterance is essentially to find the best path among all possible paths, and the solution can be found using dynamic programming, which is a recursive procedure based on the principle of optimality.

### 3.3.2 Dynamic programming principle

Dynamic programming has been widely used to solve the optimal path problems and synchronous sequential decision problems. To illustrate the optimal path problem, which is also called the asynchronous sequential decision problem, suppose we have a set of points labeled from 1 to N, and every pair of points *(i,j)* has a nonnegative cost $\zeta(i,j)$. This cost indicates the cost of moving from the $i^{\text{th}}$ point to the $j^{\text{th}}$ point in one step. Using as many steps as necessary, we need to find the minimum cost of moving from point *i* to point *j*. The solution of the problem, based on the optimality principal from Bellman [20], states that to obtain the optimal consecutive sequence of moves from *i* to *j*, all partial intermediate moves must also be optimal. The steps to determine the minimum cost path between points *i* and *j* are as follows:

$$
\begin{array}{lll}
\varphi_1(i,j) = \zeta(i,l), & l = 1,2,\ldots,N & \\
\varphi_2(i,l) = \min\limits_k(\varphi_1(i,k) + \zeta(k,l)), & k = 1,2,...,N & l = 1,2,...,N \\
\varphi_3(i,l) = \min\limits_k(\varphi_2(i,k) + \zeta(k,l)), & k = 1,2,...,N & l = 1,2,...,N \\
\varphi_S(i,l) = \min\limits_k(\varphi_{S-1}(i,k) + \zeta(k,l)), & k = 1,2,...,N & l = 1,2,...,N \\
\varphi(i,j) = \min\limits_{1\le s\le S}(\varphi_s(i,j)) & &
\end{array}
$$

where *S* is the maximum number of steps allowed and $\varphi_s(i,l)$ is the *s*-step best path from point *i* to point *l*.

Now let us consider the synchronous sequential decision problem. The problem requires us to find the minimum cost $\varphi_M(i,j)$ of an optimal sequence of moves from point *i* to point *j* in a fixed number of moves *M*. Again, the principle of optimality is used to solve this problem. The steps necessary to implement the algorithm are as follows:

1. Initialization:

$$\varphi_1(i,n) = \zeta(i,n)$$

$$\xi_1(n) = i$$

$$\text{for } n = 1,2,...,N$$

2. Recursion:

$$\varphi_{m+1}(i,n) = \min_{1\le l\le N}[\varphi_m(i,l) + \zeta(l,n)]$$

$$\xi_{m+1}(n) = \arg\min_{1\le l\le N}[\varphi_m(i,l) + \zeta(l,n)]$$

for $n = 1,2,...,N$ and $m = 1,2,...,M$-2

3. Termination:

$$\varphi_M(i, j) = \min_{1\le l\le N}[\varphi_{M-1}(i,l) + \zeta(l, j)]$$

$$\xi_M(j) = \arg\min_{1\le l\le N}[\varphi_{M-1}(i,l) + \zeta(l, j)]$$

4. Path backtracking:

optimal path = $(i,i_1,i_2,...,i_{M-1},j)$,

where

$$i_m = \xi_{m+1}(i_{m+1}), \qquad m = M\text{-}1, M\text{-}2,...,1$$

$$i_M = j$$

Notice that the complexity of this algorithm is on the order of $NM$ computations since the algorithm only needs to trace $N$ paths at the end of every move (for a total of $M$ moves).

### 3.3.3 Warping function constraints

There are several warping constraints associated with DTW to preserve linguistic properties in both speech patterns being compared. For example, if the beginning point and the ending point of one speech pattern are reversed, then the comparison performed does not make sense linguistically. The necessary constraints for time alignment in DTW are endpoint constraints (boundary conditions), monotonicity conditions, local continuity constraints, and slope weighting.

Endpoint constraints are the boundary conditions where the warping function starts and ends. If we have a highly precise speech endpoint detector, then the time-warped starting point would correspond to the first frame of test and reference speech patterns, and the ending point would correspond to the last frame of test and reference speech patterns. However, due to possible inaccuracy in the speech endpoint detection, the endpoint constraints are relaxed, i.e.,

| | | |
|---|---|---|
| beginning point | $1\le\phi_x(1)\le\varepsilon$ , | $1\le\phi_y(1)\le\varepsilon$ |
| ending point | $T_x\text{-}\varepsilon\le \phi_x(T)\le T_x$ | $T_y\text{-}\varepsilon\le \phi_y(T)\le T_y$ |

where ε is the tolerance parameter and it is set to 5. The second constraint imposed on the warping function is monotonicity, which means that the temporal order of the speech spectral sequence cannot be changed while doing time normalization. Clearly, if the temporal order is changed, then the speech will have a totally different linguistic meaning, or it may not have any meaning anymore. Furthermore, this constraint implies that the evaluated path will not have a negative slope. The mathematical forms of the monotonicity conditions are

$$\phi_x(k+1) \geq \phi_x(k)$$
$$\phi_y(k+1) \geq \phi_y(k)$$

Other important restrictions in time warping are local continuity constraints, which are necessary so that the time normalization process does not throw away any important information about the speech patterns. There are several local continuity conditions commonly used for time warping. Figure 10 shows a set of local continuity constraints, called Type I constraints by Rabiner and Juang, used in this work.

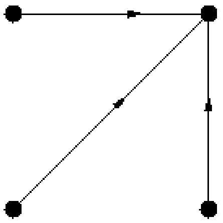


Figure 10: Type I local continuity constraints.

Type I constraints were proposed by Sakoe and Chiba [12], and they can be written mathematically as

$$\phi_x(k+1)-\phi_x(\mathrm{k})\leq 1$$
$$\phi_y(k+1)-\phi_y(\mathrm{k})\leq 1$$

By employing Type I constraints, no restriction is imposed on the warping function's slope. Some other local continuity constraints restrict the warping function's slope and as a result, certain regions of the ($i_x$, $i_y$) plane are excluded from the time warping function. Slope weighting is another constraint to give a better measure of the warping function. Many slope weighting functions associated with time warping exist, and here we use a slope weighting function proposed by Sakoe and Chiba:

$$w(k)=\phi_x(k)-\phi_x(k-1)+\phi_y(k)-\phi_y(k-1)$$

Sakoe and Chiba called this weighting coefficient the "symmetric form". The combination of local continuity constraints and slope weighting is illustrated in Figure 11.

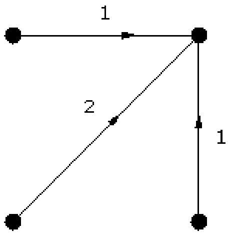


Figure 11: Type I local continuity constraints with slope weighting.

The symmetric form of weighting coefficient yields a normalizing factor $N=T_x+T_y$.

### 3.3.4 DTW algorithm summary

After discussing all the constraints used in this work, we can summarize the necessary steps to implement DTW algorithm as follows:

1. Initial condition:

$$D(\phi_x(1), \phi_y(1)) = 2d(\phi_x(1), \phi_y(1))$$

Here we have to find a pair of values ($\phi_x(1)$, $\phi_y(1)$) within the boundary conditions to get minimum accumulated distance. To find the optimal starting point, the complete DTW algorithm needs to be run for every candidate of ($\phi_x(1)$, $\phi_y(1)$). To reduce computation time, we implement a suboptimal procedure to find the values of $\phi_x(1)$ and $\phi_y(1)$ by selecting the values that give a minimum initial condition, i.e.,

$$(\phi_x(1), \phi_y(1)) = \arg \min_{\phi_x(1), \phi_y(1)} d(\phi_x(1), \phi_y(1))$$

for $1 \leq \phi_x(1) \leq \varepsilon$ and $1 \leq \phi_x(1) \leq \varepsilon$.

2. Implement DP recursion based on local continuity constraints and slope weighting:

$$D(i_x, i_y) = \min \begin{bmatrix} D(i_x - 1, i_y) + d(i_x, i_y) \\ D(i_x, i_y - 1) + d(i_x, i_y) \\ D(i_x - 1, i_y - 1) + 2d(i_x, i_y) \end{bmatrix}$$

for $\phi_x(1) \leq i_x \leq T_x$ and $\phi_y(1) \leq i_y \leq T_y$.

3. Calculate time-normalized distance or accumulated distortion:

$$d(X, Y) = \frac{1}{N} \min_{\phi_x(T), \phi_y(T)} D(\phi_x(T), \phi_y(T))$$

for $T_x - \varepsilon \leq \phi_x(T) \leq T_x$ and $T_y - \varepsilon \leq \phi_y(T) \leq T_y$.

### 3.4 Quantization

After performing speech recognition using DTW with perfect (nonquantized) coefficients, we investigate the effect of quantization on the speech recognition performance. PLP coefficients, which are analogous to LPC coefficients, can lead to unstable filters when they are quantized and an unstable filter would cause large errors in the PLP cepstrum, and probably speech recognition errors. For this reason, PLP coefficients are converted into LSP coefficients before applying vector quantization.

### 3.4.1 Line spectral pairs (LSP)

LSP is widely used in speech coding as a representation of the LPC parameters. LSP encodes speech spectral information in the frequency domain, and it has better characteristics than other LPC representations such as LAR (log area ratio) or PARCOR (partial correlation). LSP and LAR are less sensitive to small quantization errors than PARCOR. Most recent speech coders use LSP instead of LAR because interframe and intraframe predictability of LSP can be used to reduce the bit rate or increase quality. Suppose we have direct form LPC coefficients $a_k$, then the transformation from LPC parameters to LSP coefficients [21] is as follows:

$$A(z) = \sum_{k=1}^{5} a_k z^{-k}$$
$$P(z) = A(z) + z^{-(p+1)} A(z^{-1})$$
$$Q(z) = A(z) - z^{-(p+1)} A(z^{-1})$$
$$p_n = \arg(\mathrm{roots}(P(z))), \quad 0 < p_n < \pi$$
$$q_n = \arg(\mathrm{roots}(Q(z))), \quad 0 < q_n < \pi$$

Some nice characteristics of LSPs are the following:

1. Terms $p_n$ and $q_n$ alternate with each other, i.e.,

$$0 < p_1 < q_1 < p_2 < q_2 < p_3 < \pi$$

2. Terms $p_n$ and $q_n$ are correlated with each other.
3. LSP coefficients have high correlation or change slowly from frame to frame.

The nice ordering and high degree of interframe and intraframe correlation can be effectively exploited by a speech coder via predictive quantization and vector quantization, respectively.

### 3.4.2 Vector quantization (VQ)

Vector quantization is commonly used as a data compression method in speech and image coding. According to Gersho and Gray [22], VQ maps a $k$-dimensional vector in vector space $R^k$ into a finite set of $k$-dimensional vectors $Y$=($\mathbf{y}_i$; $i$=1,2,…,$N$). Each vector

$\mathbf{y}_i$ is called a codeword or centroid vector, and the set *Y* is called the codebook with size *N*. For example, consider the two-dimensional VQ system illustrated in Figure 12. Here the two-dimensional space is divided into several regions or cells, which are called Voronoi regions. Every Voronoi region has one codeword, and every vector in the region is assigned to the corresponding centroid.

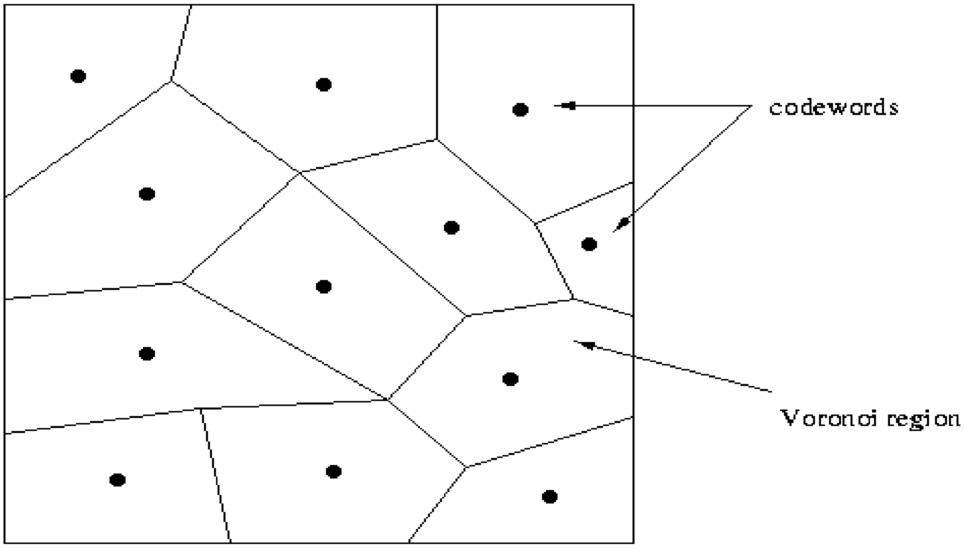


Figure 12: Two-dimensional space is divided into Voronoi regions and every region is represented by a codeword.

The first task in running the VQ algorithm is to design the codebook. Before discussing how to design a codebook, two things need to be addressed first, namely, the similarity or distance measure and a centroid computation procedure. The distance measure used is Euclidean distance of LSP coefficients, which is also the distance measure used for the DTW algorithm. The centroid of each cell is computed by summing all the vectors in the corresponding cell and normalizing it by the number of vectors in the cell, i.e.,

$$\mathbf{y} = \frac{\mathbf{1}}{K}\sum_{i=1}^{K}\mathbf{v}_i$$

where *K* is the number of vectors in the cell.

The codebook design algorithm employed in this work is a binary splitting version of the LBG algorithm, named after Linde et al. [15]. The algorithm is implemented as follows:

1. Design a 1-vector codebook ($N$=1) with centroid $\mathbf{y}_1$, i.e.,

$$\mathbf{y}_1 = \frac{\mathbf{1}}{K}\sum_{i=1}^{K}\mathbf{v}_i$$

   where $\mathbf{v}_i$ is the training vector. Initialize the average distortion value $D^*$ to an arbitrarily large value.

2. For $m$=1,2,…,$N$, split the centroid to double the size of the codebook, i.e.,

$$\mathbf{y}^*_m = \mathbf{y}_m(1+\varepsilon)$$
$$\mathbf{y}^*_{m+N} = \mathbf{y}_m(1-\varepsilon)$$

   where ε is 0.02 in this case. Also, the value of $N$ needs to be updated to 2$N$.

3. Classify vectors using the nearest neighbor condition by assigning each training vector into its associated centroid based on the Euclidean distance measure.

4. Update the centroid in each cell, i.e.,

$$\mathbf{y}_m = \frac{\mathbf{1}}{K_m}\sum_{i=1}^{K_m}\mathbf{v}_i$$

   where $K_m$ is the number of vectors in the cell.

5. Compute the average distortion of the updated codebook $D$ using squared error distortion measure, i.e.,

$$D = \sum_{m=1}^{N}\sum_{i=1}^{K_m}(\mathbf{v}_i^{(m)} - \mathbf{y}_m)^2$$

Then compute $\delta=(D^*-D)/D$ and set $D^*=D$. If $\delta$ is below a predetermined threshold, then stop. If not, then repeat steps 2 to 4 until the value of $\delta$ is below threshold. The value of $\delta$ is set to 0.001 in this case.

# 4. EXPERIMENTAL RESULTS

The first experiment is to do speaker-independent isolated digit recognition without any quantization process. The test set consists of 220 utterances from 10 men and 10 women. The test set and the template set are completely different; in other words, no overlap between test files and reference files. The speech boundaries are manually corrected for all experiments. At first, two templates per word are used to do recognition, i.e., the reference template consists of 22 utterances for 11 digits (two for each digit). Half of those are men's utterances, and the other half are women's. The accuracy of the speech recognition system in the first experiment is 93.18%.

The recognition accuracy is somewhat low for real application and it is expected to decrease further as quantization is inserted into the system. However, as Hermansky shows in his experiment of speaker-independent isolated digit recognition, the accuracy can be increased dramatically as the number of templates per word is increased. Therefore, we increase the number of templates per word to 9 and 12. The recognition accuracy reaches 95.91% and 97.73% for 9 and 12 templates per word, respectively. Hermansky did experiments by varying the number of templates per word from 2 to 23, and the recognition accuracy's range is from 92% to 98%. Furthermore, the results obtained in this work are comparable to those of Hermansky's experiments.

Next, LSP coefficients are quantized and the codebook size is varied from 64 (six bits) to 256 (eight bits). The codebook is designed using the LBG algorithm from a set of 792 utterances (36 men and 36 women). For all subsequent experiments, nine and twelve templates per word are used because the accuracy of the system with two templates per word is already too low and the quantization process will decrease the performance further. To further reduce the bit rate, PLP analysis is done every 20 ms instead of 10 ms, which is essentially down-sampling the speech pattern by a factor of two. Then the speech patterns are linearly interpolated to get the original sampling rate before recognition is performed. As LSP coefficients vary slowly from frame to frame, the distortion introduced by down-sampling and interpolation operations is expected to be small so the degradation in recognition accuracy is not significant. Experiments using this scheme are also performed without quantization and with quantization using six to eight bits.

Figure 13 shows the results of experiments discussed so far (excluding experiments using two templates per word). An infinite number of bits corresponds to no quantization process.

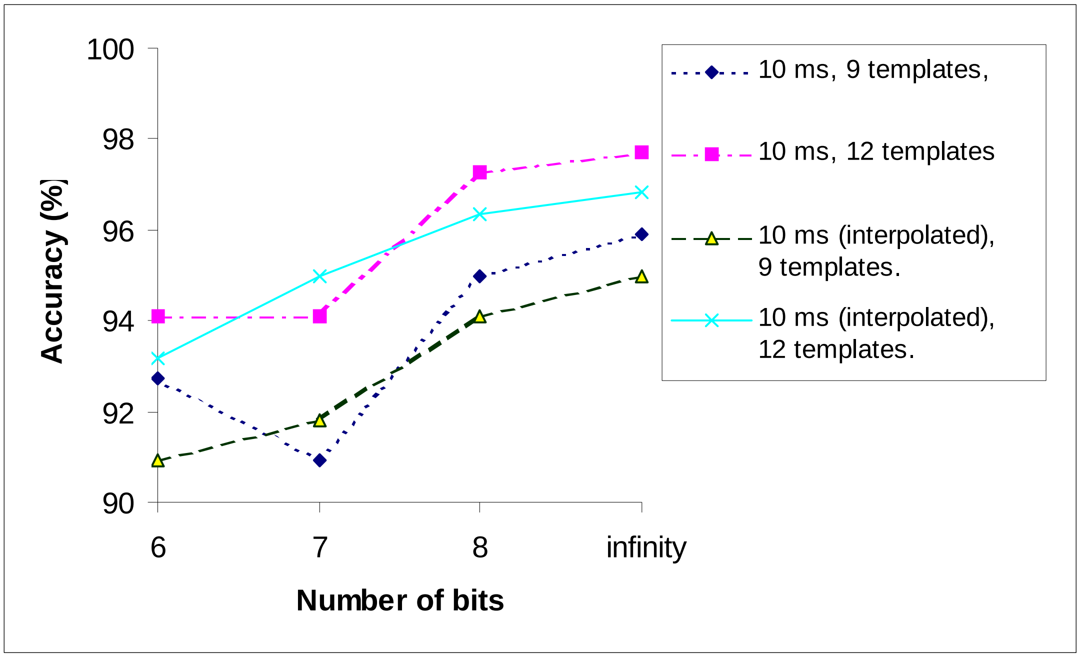


Figure 13: Recognition accuracy as a function of number of bits

As expected, we can see from Figure 13 that the recognition accuracy declines as the number of bits used in quantization decreases, except for one case. The system with the 10-ms analysis step and 9 templates per word shows a different behavior because 7-bit quantization performs worse than 6-bit quantization. For all cases, 8-bit quantization yields only slight degradation compared to the no quantization scheme. The difference between the results of 8-bit quantization and no quantization is statistically insignificant because the accuracies of 8-bit quantization systems are still within the range of standard deviation of the accuracies of systems without quantization. For example, the accuracy of a system with an interpolated 10-ms analysis step and 12 templates per word is 96.82% without quantization. Let $P$ be the recognition accuracy and $N$ be the total number of test files; then the standard deviation $\sigma$ can be calculated as follows:

$$\sigma = \sqrt{\frac{P(1-P)}{N}} = \sqrt{\frac{0.9682(1-0.9682)}{220}} = 0.9840$$

A similar system quantized with eight bits yields accuracy of 96.36%, which is still within a range of 96.82 ± 0.9840 = [95.836, 98.804].

The last experiment is using 20-ms PLP analysis also; however, no linear interpolation is involved in the speech recognition system. Both test and reference speech patterns have 20-ms analysis steps, so the number of computations is much less than that of previous experiments. The experiment is also carried out using 9 and 12 templates per word and by varying the degree of quantization. Since much information is lost due to down-sampling, the recognition accuracy of this system is inferior to systems in previous experiments using the same number of templates per word. Without quantization, the accuracy only reaches 93.18% and 95% for 9 and 12 templates per word, respectively. For convenience, all results of the experiments obtained in this work are tabulated in Table 2.

Table 2: The recognition accuracy of experiments in this work.

| Templates per word | Analysis step | Quantization | | | |
|---|---|---|---|---|---|
| | | 6 | 7 | 8 | Infinity |
| 2 | 10 ms | - | - | - | 93.18 |
| 9 | 10 ms | 93.18 | 90.91 | 95.00 | 95.91 |
| 9 | 10 ms (interpolated) | 91.82 | 91.82 | 94.09 | 95.00 |
| 9 | 20 ms | 89.55 | 91.82 | 93.64 | 93.18 |
| 12 | 10 ms | 94.55 | 94.55 | 97.27 | 97.73 |
| 12 | 10 ms (interpolated) | 93.18 | 96.36 | 96.36 | 96.82 |
| 12 | 20 ms | 91.36 | 93.64 | 94.09 | 95.00 |

The results of the experiments suggest that a good speaker-independent digit recognition system can be developed using PLP analysis and the DTW algorithm with a fairly low bit rate. Of all experiments carried out in this work, the system employing down-sample and interpolation operations seems most promising. If 8-bit quantization is used, the achieved bit rate is only 400 bit/s. This number is much lower than the bit rate of the speech coder used in wireless communication, which typically has a bit rate of 4.8 kb/s to 8 kb/s.

Transmitting both CS-ACELP speech coder parameters and quantized PLP parameters would require 8.4 kb/s, an increase of only 5% over the normal CS-ACELP bit rate.

## 5. CONCLUSION AND FUTURE WORK

A speaker-independent isolated digit recognition system using PLP analysis and the DTW algorithm has been examined in this work. The system is examined with and without vector quantization. Vector quantization is used instead of scalar quantization because vector quantization gives lower distortion than scalar quantization for the same number of bits. Based on the results of experiments, quantization using eight bits or higher is recommended since the degradation introduced in recognition performance is not significant compared to the system without quantization.

To minimize the bit rate further without degrading recognition performance, the analysis step at the front-end processor is changed to 20 ms. The experiment shows that the performance does not decrease substantially as long as linear interpolation is carried out at the decoder. A big advantage of this system is the very low bit rate required to transmit LSP coefficients; the bit rate is half that of a system with a 10-ms analysis step.

Although the performance of the recognition system in this work is fairly good, further work is needed to determine the effect of a speech coder on the recognition system as a speech coder is known to reduce the recognition accuracy slightly. According to [2] and [3], extracting recognition features from synthesized speech at the decoder rather than from unquantized speech results in a WRA decrease from 96.17% to 95.16% and from 83.6% to 78.1%, respectively. The degradation is worse than that introduced by quantizing LSP coefficients. Therefore, it is probably better to do recognition using the quantized LSP coefficients rather than using the decoded speech.

Furthermore, a good endpoint detection algorithm needs to be incorporated into the recognition system in this work. Using automatic endpoint detection, the performance of the system needs to be examined to determine whether the accuracy achieved in this work changes significantly or not. In this work, no template clustering was used to design the reference templates. According to [1], template training by clustering is necessary to achieve high performance for practical tasks. Hence, the effect of template clustering on the WRA in this work needs to be studied further. Another important problem that needs to be examined is the effect of the presence of background noise on the recognition performance. Finally, the accuracy of the system in this work using the HMM recognition algorithm is worth investigating since HMM is commonly used as a recognizer.

## REFERENCES

[1] L.R. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. New Jersey: PTR Prentice-Hall, 1993.

[2] H.K. Kim and R.V. Cox, “Bitstream-based feature extraction for wireless speech recognition,” in *Proceedings ICASSP 2000* (to be published).

[3] S.H. Choi, H.K. Kim, and H.S. Lee, “LSP weighting functions based on spectral sensitivity and mel-frequency warping for speech recognition in digital communication,” in *Proceedings ICASSP 1999*, vol. 1, March 1999, pp. 401-404.

[4] A. Gallardo-Antolin et al., “Avoiding distortions due to speech coding and transmission errors in GSM ASR tasks,” in *Proceedings ICASSP 1999*, vol. 1, March 1999, pp. 277-280.

[5] S. Dufour, C. Glorion, and P. Lockwood, “Evaluation of the root-normalised front-end (RN_LFCC) for speech recognition in wireless GSM network environments,” in *Proceedings ICASSP 1996*, vol. 1, May 1996, pp. 77-80.

[6] S. Euler and J. Zinke, “The influence of speech coding algorithms on automatic speech recognition,” in *Proceedings ICASSP 1994*, vol. 1, April 1994, pp. 621-624.

[7] C. Mokbel, L. Mauuary, L. Karray, D. Jouvt, J. Monne, J. Simonin, and K. Bartkova, “Towards improving ASR robustness for PSN and GSM telephone applications,” in *Proceedings, Third IEEE Workshop on Interactive Voice Technology for Telecommunications Applications, 1996,* pp. 73-76.

[8] V. Digalakis, L. Neumeyer, and M. Perakakis, “Product-code vector quantization of cepstral parameters for speech recognition over the www,” in *Proceedings ICSLP*, 1998, pp. 2641-2644.

[9] S. Tsakalidis, V. Digalakis, and L. Neumeyer, “Efficient speech recognition using subvector quantization and discrete-mixture HMMs,” in *Proceedings ICASSP 1994,* vol. 2, March 1999, pp. 569-572.

[10] S.H. Choi, H.K. Kim, H.S. Lee, and R.M. Gray, “Speech recognition method using quantised LSP parameters in CELP-type coders,” *Electronic Letters*, vol. 34 no. 2, pp. 156-157, January 1998.

[11] P. Price, “Evaluation of spoken language systems: The ATIS domain,” in *Proceedings of the Third DARPA Speech and Natural Language Workshop,* Hidden Valley, PA, June 1990, Morgan Kaufmann.

[12] H. Sakoe and S. Chiba, “Dynamic programming optimization for spoken word recognition,” *IEEE Trans. Acoustics, Speech, Signal Processing,* vol. ASSP-26, pp.43-49, February 1978.

[13] L.R. Rabiner, A.E. Rosenberg, S.E. Levinson, “Considerations in dynamic time warping algorithms for discrete word recognition,” *IEEE Trans. Acoustics, Speech, Signal Processing,* vol. ASSP-26, pp. 575-582, December 1978.

[14] H. Hermansky, “Perceptual linear predictive (PLP) analysis of speech, ” *Journal of the Acoustical Society of America*, vol. 87 no. 4, pp. 1738-1752, 1990.

[15] Y. Linde, A. Buzo, and R.M. Gray, “An algorithm for vector quantizer design,” *IEEE Transactions on Communications*, vol. COM-28, pp. 84-95, January 1980.

[16] J.G. Wilpon, L.R. Rabiner, and T.B. Martin, “An improved word-detection algorithm for telephone-quality speech incorporating both syntactic and semantic constraints,” *AT&T Technical Journal*, vol. 63 no. 3, pp. 479-498, March 1984.

[17] L.R. Rabiner and M.R. Sambur, “An algorithm for determining the endpoints of isolated utterances,” *The Bell System Technical Journal*, vol. 54 no. 2, pp. 297-315, February 1975.

[18] L.A. Chistovich, “Central auditory processing of peripheral vowel spectra,” *Journal of the Acoustical Society of America*, vol. 77 no. 3, pp. 789-805, 1985.

[19] R. Salami et al., “Design and description of CS-ACELP: A toll quality 8 kb/s speech coder,” *IEEE Transactions on Speech and Audio Processing*, vol. 6 no. 2, pp. 116-130, March 1998.

[20] R.E. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton University Press, 1957.

[21] M.A. Hasegawa-Johnson, *Course Notes in Speech Production, Speech Coding, and Speech Recognition,* University of Illinois at Urbana-Champaign, 1998.

[22] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*. Boston, MA: Kluwer Academic Publishers, 1992.