ON THE APPLICATION OF VECTOR QUANTIZATION TO SPEAKER INDEPENDENT ISOLATED WORD RECOGNITION

by

Florina Rogers

Dipl. Ing. Polytechnic Institute of Iasi, 1989

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE
in the School
of
Engineering Science

SIMON FRASER UNIVERSITY
April 1996

APPROVAL

Name: Florina Rogers

Degree: Master of Applied Science

Title of thesis : ON THE APPLICATION OF VECTOR QUANTIZATION TO SPEAKER INDEPENDENT ISOLATED WORD RECOGNITION

Examining Committee: Dr. Shawn Stapleton, Chairman

Dr. Vladimir Cuperman
Professor, Engineering Science, SFU
Senior Supervisor

Dr. Jacques Vaisey
Assistant Professor, Engineering Science, SFU
Supervisor

Dr. Steve Hardy
Professor, Engineering Science, SFU
Examiner

Date Approved: April 2, 1996

PARTIAL COPYRIGHT LICENSE

Title of Thesis/Project/Extended Essay

"On The Application Of Vector Quantization To Speaker Independent Isolated Word Recognition"

Author:

(signature)

Florina Rogers
(name)

April 2, '96
(date)

Abstract

The source coding method referred to as vector quantization (VQ) is used in a speech-recognition system to represent an arbitrary speech spectral vector into one of a fixed number of codebook symbols with the benefit of significantly reduced computation in the recognition process.

In low-complexity speaker-independent isolated-word recognition systems with multiple codebooks, the performance of the VQ has a big impact on the overall performance of the system. This thesis studies different ways of combining temporal and spectral characteristics in the VQ process, with the objective of improving the recognition, while maintaining or decreasing the storage requirement. Two methods of incorporating time information directly into the codebooks are presented and compared to an existent method, based on considering the probability of the time of occurrence of a given spectral vector in the quantization process.

The recognition system implemented to evaluate these methods consists of modules which perform signal pre-processing, feature extraction and vector quantization, with a signal-processing front end based on a bank-of-filters model.

The experimental results show that the methods proposed reduce significantly the recognition error rate and have similar memory requirements to the reference method.

To Dr. Joel G. Rogers, with gratitude for his personal example and his support.

Acknowledgements

I want first to thank Dr. Vladimir Cuperman for his guidance and encouragement during the last year: his generous support made this research possible. This speech recognition project is a result of the collaboration with OMNI Microelectronics, Inc., Santa Clara, CA.

My thanks go also to Dr. Costin Ifrim and Lei Zhang, who worked with me on the same project, and also to Mike McGuire, for the software modules he so generously shared with me.

Finally, I would like to thank everyone in the speech group for making the "Digital Speech Laboratory" such a friendly and productive working environment.

Contents

List of Tables

List of Figures

List of Abbreviations

AGC	Automatic Gain Control
BPF	Band Pass Filter
CB	Codebook
CW	Codeword
DFT	Discrete Fourier Transform
DTW	Dynamic Time Warping
FWR	Full-Wave Rectifier
HMM	Hidden Markov Model
LBG	Linde, Buzo, Gray
LPC	Linear Prediction Coding/ Coefficients
LPF	Low Pass Filter
MTC	Multiple Time Component
OSC	Overlapped Segmented Codebooks
PDF	Probability Density Function
PSD	Power Spectral Density
PT	Probability Tables
RC	Resistance/ Capacitance
SCF	Switched Capacitor Filter
VLSI	Very Large Scale Integration
VQ	Vector Quantization/ Quantizer
VCR	Video Cassette Recorder
WBF	Wide Band Filter

Chapter 1

Introduction

Isolated word recognition represents one of the first efforts in the pursuit of automatic speech recognition. The distinguishing feature of an isolated-word recognition system is that it requires words to be spoken individually, in isolation from other words, and separated by distinct interword pauses. As early as 1952, an isolated digit recognizer based on spectral measurements was built at Bell Laboratories [1]. This research area produced a viable and usable technology only in the 1970's; however, the goal of improving the performance for systems using this technology is still pursued. This is due mainly to the fact that the state-of-the-art speaker independent systems (which do not require speaker specific training) still give error rates of approximately 5% under laboratory conditions, i.e. a high-quality microphone in a low-noise environment [2]. Such a performance would be unacceptable if the system were part of a commercial product to be used in a realistic environment.

Recently, there has been an increased consumer interest in products with voice activated user interfaces. As a result, the development efforts to produce error-free, cost-effective hardware implementations of speech recognizers have been intensified. The advances in VLSI technology and signal processing capabilities, combined with the demand for voice communication, have revived the quest for a reliable isolated-word, speaker-independent recognizer.

A voice recognition chip can be used to implement the user interface of a command-and-control system. In such a system, the user speaks a single command (either an isolated word or phrase), and the machine, upon correctly recognizing the command, acts appropriately. The output of the speech recognizer is the index of the word that

is most likely to have been spoken based on the recognizer's vocabulary. This index is then used to select the corresponding action to be passed on to the physical system under voice control. Several requirements are essential for the recognition system to be used in a hardware implementation of a user interface for a command-and-control system:

- The proposed interface must be "user friendly"; it must make the user feel comfortable with the commands and it must provide an effective means of communication. This imposes restrictions on the size and structure of the vocabulary used, and determines whether the recognizer is speaker independent or speaker dependent.

- The command-and-control system must achieve a specified minimum level of performance on the task associated with the recognition decision. User perception of the recognizer's effectiveness appears to be non-linear [2], in that the absolute level of performance is relatively unimportant as long as the error rate stays under a certain level (5%). This requires a minimum degree of recognition accuracy for the overall system, as well as a high degree of robustness to noise.

- The response time of the recognition system must be minimized.

- The recognition system must be cost effective.

In this context, the system implemented in this thesis is required to be:

- dedicated to a small vocabulary of isolated utterances,

- speaker independent (unrestricted set of speakers),

- highly accurate,

- robust and

- suitable for a low-cost analog VLSI implementation.

The recognition method most often used in practical implementations is the comparison of spectral patterns and consists of spectral feature extraction followed by pattern comparison. The simplest spectral feature extraction analog hardware implementation is a bank of analog filters, used in existent recognition chips such as OKI

MSM6250, or NEC TC8861F [3]. The spectral features are further processed using a vector quantizer (VQ).

Although a VQ is generally used to compress the speech patterns (and thus reduces the computational complexity and storage required by subsequent decision algorithms), in some restricted cases good recognition performance can be obtained with straightforward use of the VQ as a recognizer [4]. The recognition accuracy reported in [4] was of 99% for a speaker-dependent system and of 88% using a speaker-independent system for a highly non-confusable 20-word vocabulary. The use of a VQ-based decision block brings computational savings in comparison to a more sophisticated decision block, like the Dynamic Time Warping (DTW) or the Hidden Markov Model (HMM) processors. VQ-based recognition is also known as "multiple codebook recognition", because it uses one codebook for each vocabulary word, or "recognition without time alignment", to distinguish it from the DTW method, which uses time alignment of input speech patterns.

A performance comparison among different recognition methods is given in Table 1.1, where the spectral analysis methods used for feature extraction are the filter-bank (FB) method and the linear predictive coding (LPC) method, producing either linear prediction coefficients or cepstral coefficients as the speech pattern to be used in recognition. The result of the DTW-FB method corresponds to a VLSI implementation of a recognition system [3], and is outperformed by the simulation results obtained using the VQ-FB design presented in this thesis [5]. When LPC features are used, both the DTW and the HMM methods outperform the VQ method [2], [6]. The test set for the DTW, HMM and VQ methods based on LPC features was recorded in studio conditions, while for the FB methods a more realistic recording environment was used.

The baseline recognition system, designed according to the constraints imposed by the hardware implementation targeted, consists of a filter-bank front-end followed by a VQ on spectral features and gives an accuracy of almost 90%. To insure that an accuracy of at least 95% is obtained for the system working in noisy conditions with a test set of speakers with foreign pronunciation, several improvements to the algorithm needed to be investigated.

The approach used in the baseline system does not preserve the sequential characteristics of the utterance class, or the temporal characterization of the spectral shape.

Recognition Method	Recognition Accuracy [%]	Reference
DTW-FB	86	[3]
DTW-LPC	98	[2]
VQ-FB	89.1	[5]
VQ-LPC	94	[4], [6]
VQ-cepstral	95.5	[6]
HMM-LPC	98	[6]

Table 1.1: Small Vocabulary, Speaker Independent, Isolated Word Recognition Systems - Performance Comparison

In this approach, a single VQ is used for the entire duration of the utterance, while the distortion measure used takes into account only the magnitude of the spectral feature and ignores its time of occurrence. The lack of explicit characterization of the sequential behavior can be remedied either by treating each utterance class as a concatenation of segments, each represented by a VQ codebook, or by combining the spectral distortion with a temporal distortion. The first approach is referred to as *Segmental VQ* [8]. Another technique, using combined spectral and temporal distortions, was proposed and evaluated by Pan et al. [7]. Their approach uses an estimated probability density function (PDF) of the time of occurrence on a normalized time scale, and will be further referred to as the *Probability Tables* method.

Two methods of modifying the VQ structure to take into account temporal information in addition to the spectral features are presented in this thesis:

1. The *Temporal Component Method*: generates an additional temporal component to each vector of spectral features. As a result of training, a temporal codebook is created. The temporal codebook is used in conjunction with the existent spectral codebook during quantization.

2. The *Overlapped Codebooks Method*: time normalizes each utterance and then subdivides it into a number of non-overlapping regions. For each of these regions a sub-codebook is generated, and the sub-codebooks are overlapped to a variable degree. This approach uses the same principle as the segmental VQ described by Burton, but gives a more accurate pattern

representation in codebook space and performs better without significant increases in complexity.

These two methods are compared with the probability tables method, proposed by Pan et al [7]. The decrease in error rate obtained is 40% relative to the baseline system and 30% relative to the Probability Tables method for a 12-word vocabulary and recordings performed both in studio and noisy conditions.

1.1 Contributions of the Thesis

The major contributions of this thesis can be summarized as follows:

1. The development and performance analysis of speaker independent isolated word recognition algorithms which use a VQ as a recognition processor. The error rate was reduced by over 40% by incorporating time information in the quantization process.
2. The study of a new method of adding temporal information to a spectral codebook; the new method gives a 20% improvement in recognition rate over the existing probability tables technique [7], while reducing the required codebook storage space by 70%.
3. The evaluation of the improvement due to using overlapping codebooks in segmental VQ. The results show a 20% improvement in error rate at a 45% reduction in codebook storage space.
4. The design and implementation of a speech recognition algorithm which simulates a real-time analog VLSI recognizer.

1.2 Thesis Outline

Chapter 2 presents an overview of spectral analysis models in the context of the statistical approach to speech recognition. A detailed description of vector quantization, including a discussion of codebook training methods and distortion measures is presented in Chapter 3.

Existent methods of incorporating time information during quantization are presented in Chapter 4, while the two new methods which incorporate time information during quantization are introduced in Chapter 5.

Chapter 6 outlines the speech recognition system used for the evaluation of the spectral-temporal VQ methods proposed in this thesis. The results of this evaluation are presented in Chapter 7.

Chapter 2

Spectral Representation of Speech Signals

Different speech sounds can be characterized by spectral and temporal properties that depend on the acoustic-phonetic features of the sound. Based on this characterization, they can be grouped into sound classes. For each such class, or phonetics, the properties of the acoustic features are relatively invariant across words and speakers. This observation implies that the recognition of speech classes is possible, provided a suitable analysis of the acoustic properties can be implemented.

The ideas of acoustic-phonetic characterization of sounds lead to the implementation of a speech recognition algorithm based on sequential detection of sounds and sound classes, called the *acoustic-phonetic* approach. This approach is based on the theory that postulates that there exist finite and distinctive phonetic units in the spoken language and that the phonetic units can be identified in the evolution in time of the speech signal. The method consists of a *segmentation and labeling* phase, which creates a template of phonemes representing the speech, followed by a *recognition* phase, which attempts to determine a valid entry in the vocabulary. This technique has a number of practical limitations [2], such as the subjective nature of sound classes definitions and difficulty in achieving correct segmentation.

Statistical pattern comparison is the most reliable and widely used method in practical implementations of recognition systems, and is therefore the choice for the system implemented in this thesis. Pattern recognition systems rely on gross estimation of the spectral and temporal properties of speech segments and use pattern classification

methods such as DTW, HMM and VQ. The following sections of this chapter present a description of spectral analysis methods used in the pattern recognition approach to speech recognition.

2.1 The Pattern Recognition Approach

Unlike the acoustic-phonetic approach to speech recognition, the pattern-recognition approach uses a set of attributes characterizing the entire utterance (which can be a sound, a word, or a phrase) as one entity, or *pattern*, and does not require phonetic decomposition of the utterance to be recognized. In pattern recognition methods, *spectral analysis* is used to produce spectral patterns. A large collection of speech patterns is processed during *training* to extract the templates used during the *decision* stage, which produces the recognition result.

The utterance's *pattern* is, by definition, the set of spectral and temporal features generated by the *spectral analysis* of the speech signal. The *training* procedure brings into the system information about the nature of the utterances in the dictionary. If enough versions of a utterance to be recognized are included in a training set provided to the algorithm, the training procedure can adequately characterize (statistically) the acoustic properties of that utterance. The output of this procedure is one reference template (or collection of templates) of spectral features for each dictionary entry. During the *decision* stage, a direct comparison of the unknown utterance pattern with each possible template is performed and the best match represents the recognition result.

This approach is widely used in practical applications due to its following characteristics:

- simplicity of use
- robustness and invariance to different speech vocabularies, users, feature sets, pattern comparison algorithms and decision rules
- proven high performance

The block diagram of a recognizer based on spectral analysis methods is is presented in Figure 2.1.

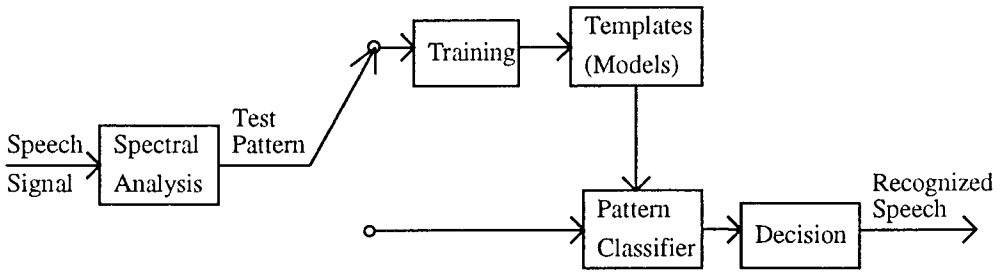

Figure 2.1: Block Diagram of a Pattern-Recognition Speech Recognizer

The *Spectral Analysis* block performs a series of measurements on the input signal to define a *pattern*. For speech signals the feature measurements are usually the output of some type of spectral analysis technique, such as a filter-bank analyzer, a linear predictive coding (LPC) analysis, or a discrete Fourier transform (DFT) analysis.

The *Training* block analyzes one or more patterns, called *training patterns*, corresponding to speech sounds of the same class, to create a representation of the class' features, called the *reference pattern*. The result can be a *template* (or a collection of templates), derived from some type of averaging or selection technique, or it can be a *model* that characterizes the statistics of the features of the reference pattern.

The *Pattern Classifier* compares the unknown pattern presented as input with each of the sound (class) templates and computes a measure of similarity, usually called *distance*, between the test pattern and each template.

The *Decision* block uses these distances to choose the class that best matches the unknown test pattern.

The factors that distinguish different pattern-recognition approaches are:

- the types of feature measurements
- the choice of templates or models

- the training method used to create the reference patterns
- the classification method and decision criteria.

The system implemented for this study uses a filter-bank analyzer for feature measurement and a vector quantization training procedure for creating templates called *codebooks*. Following training, the statistical characteristics of the source for each dictionary word are embedded in the corresponding codebook. The classifier is a vector quantizer (VQ) which uses the set of codebooks to compute distances between the unknown input pattern and each of the codebooks. The decision block simply selects the minimum distance codebook to the input pattern.

The general strengths and weaknesses of the pattern recognition approach include the following:

1. The performance of the system is influenced by the amount of training data available, in that an increase in the training set size decreases the sensitivity to noise and increases the degree of speaker dependency.
2. The reference patterns are sensitive to the speaking environment conditions.
3. The method is applicable to a wide range of speech sounds, including phrases, whole words and subword units, because the algorithm doesn't use vocabulary specific information.
4. The implementation complexity is linearly proportional to the number of vocabulary words, and is a limiting factor for the vocabulary size of the application.

2.2 Spectral Analysis Models

Spectral analysis methods are at the core of the signal processing front end of a speech recognition algorithm, because they characterize in a consistent manner the events in a speech utterance; this is done by providing a set of parameters which quantify perceptually significant characteristics for each speech segment. The resulting set of spectral characteristics extracted for an utterance are used as an input pattern in the recognition process.

The speech signal is a slowly time varying signal in the sense that, when examined over a sufficiently short period of time (between 5 and 100 ms), has approximately stationary characteristics; however, over long periods of time (on the order of 200 ms or more) the signal characteristics change to reflect the different sounds being spoken.

According to the vocal cords status, the events in a speech utterance can be classified in:

- silence (S), when no speech is produced;
- unvoiced (U), when the vocal chords are not vibrating, so that the resulting speech waveform is aperiodic (random);
- voiced (V), when the vocal chords are tensed and vibrate periodically, producing a quasi-periodic speech waveform.

Spectral representations provide information regarding the intensity of the signal in different frequency bands, a fact that can be illustrated by examining the wide-band and narrowband spectrograms of the utterance "It's easy to tell the depth of a well", presented in Figure 2.2 (a) and Figure 2.2 (b), respectively. The spectrograms are created by performing a spectral analysis on overlapping segments of the speech waveform using broad band, and respectively narrow band, filters. (The bandwidth is characterized relatively to the width of the analysis window.)

The wideband spectrogram, presented in Figure 2.2 (a), corresponds to performing a spectral analysis on 15-msec sections of waveform using a broad analysis filter (125 Hz bandwidth) with the analysis advancing in intervals of 1 msec. For the narrowband spectrogram, shown in Figure 2.2 (b), spectral analysis was performed on 50-msec sections of waveform using a narrow filter (45 Hz bandwidth), with an overlap of 1 msec between adjacent analysis windows. The spectral intensity at each point in time is indicated by the intensity (darkness) of the plot at a particular analysis frequency.

In the wideband spectrogram, Figure 2.2 (a), the vertical striations, corresponding to the spectral envelope of individual periods of the speech waveform, are well represented due to good resolution in time domain. In Figure 2.2 (b), because of the good frequency resolution, individual spectral harmonics appear as almost horizontal lines in the spectrogram. During periods of unvoiced speech high-frequency energy can be observed, while during silence there is no spectral activity. This illustrates that a

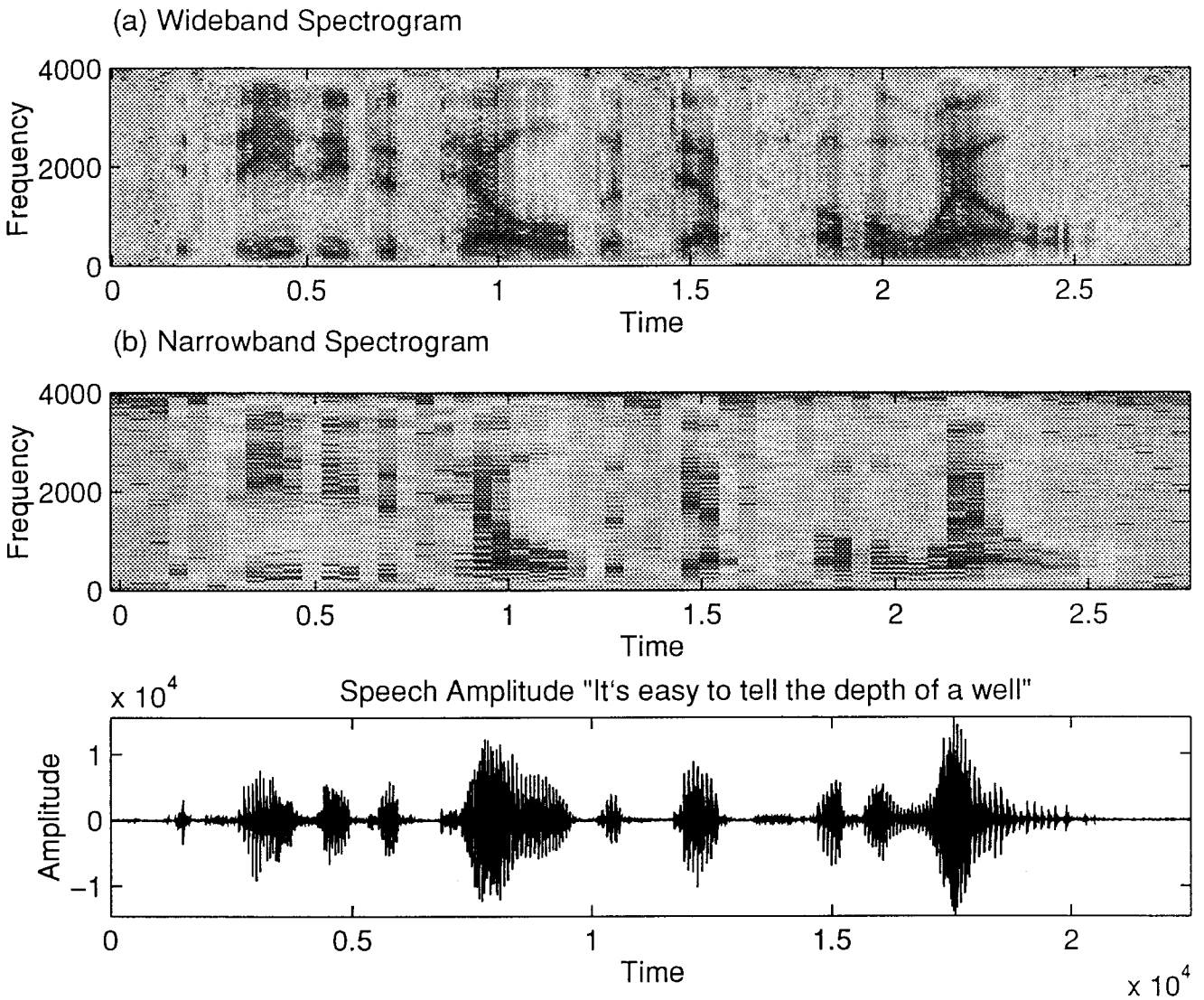

Figure 2.2: Speech Representation - Amplitude and Spectrograms

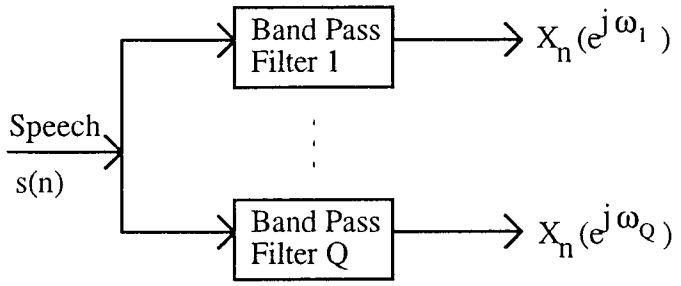

Figure 2.3: Filter-Bank Analysis Model

trade-off between time and frequency resolution must be achieved for a comprehensive analysis of the speech segment's features.

In speech recognition applications, the two most common choices for spectral analysis are:

1. The *filter-bank* model(Figure 2.3).
2. The *linear predictive coding* (LPC) model(Figure 2.4).

In the filter-bank model, the speech signal, $s(n)$, is passed through a bank of Q bandpass filters whose coverage spans the frequency range of interest in the signal (e.g., 100-3400Hz for telephone quality signals, 100-8000Hz for broadband signals). The individual filters can and generally do overlap in frequency. The center frequency of the i-th filter is ω_i,

$$\omega_i = \frac{2\pi f_i}{F_s} \tag{2.1}$$

where F_s is the sampling frequency.

The output of the i-th bandpass filter for the n-th input speech frame $s(n)$, $X_n(e^{j\omega_i})$, is a short-time spectral representation of the signal. In this model, each filter processes the speech signal independently to produce the spectral representation X_n.

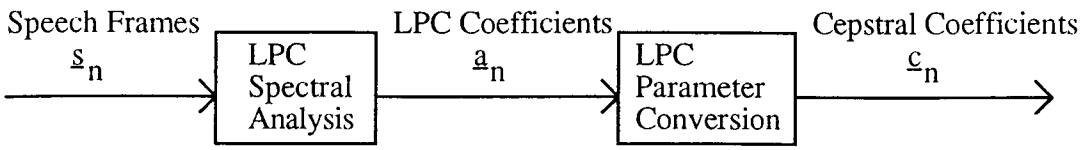

Figure 2.4: LPC Analysis Model

In the LPC approach (Figure 2.4), spectral analysis is performed on blocks (frames) of speech, where $\underline{s}_n$ represents the n-th frame of speech input. The resulting spectral representation $X_n(e^{j\omega})$ is constrained to be of the form $\sigma/A(e^{j\omega})$, where $A(e^{j\omega})$ is a P-th order polynomial with z-transform

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + ... + a_P z^{-P}. \tag{2.2}$$

The order, P, is called the LPC analysis order. The output of the *LPC spectral analysis* block, corresponding to the n-th frame, is a vector of LPC coefficients, $\underline{a}_n$, of dimension equal to the predictor order, P. The LPC coefficients specify the spectrum of an all-pole model that best matches the signal spectrum over the period of time in which the frame of speech samples was accumulated.

Equivalent feature sets, such as cepstral coefficients, have proven to be a more reliable and robust spectral representation in speech recognition [2]. The cepstral vector $\underline{c}_n$ can be derived directly from the LPC coefficients, transformation illustrated in Figure 2.4 by the *LPC parameter conversion* block.

The bank of filters model is used in the implementation presented in this thesis because it is more suitable to the be implemented in analog VLSI technology than the LPC model. The following sections of this chapter present a detailed description of a bank-of-filters analyzer.

2.3 The Filter Bank Spectral Analyzer

A detailed functional diagram of a feature extraction processor based on the filter bank spectral analysis model is presented in Figure 2.5. The purpose of the filter bank analyzer is to obtain a measurement of the speech signal energy in different frequency bands.

The sampled input speech signal, $s(n)$, is passed through a bank of Q bandpass filters, resulting in the signals:

$$s_i(n) = s(n) * h_i(n) = \sum_{m=0}^{M_i-1} h_i(m)s(n-m), \ \forall i = 1, ..., Q \tag{2.3}$$

where $h_i(m)$ is the impulse response of the i-th bandpass filter (BPF) in the finite impulse response (FIR) implementation, with a duration of M_i samples. Each of the bandpass filtered signals, $s_i(n)$, is passed through a non-linearity, such as a full-wave rectifier. The nonlinearity shifts the bandpass signal spectrum to the low-frequency band, as well as creates high frequency images. A low pass filter (LPF) is used to eliminate the high-frequency images, giving as output a set of signals, $t_i(n)$, $\forall i = 1, ..., Q$, which represent an estimate of the speech signal energy in each of the Q frequency bands.

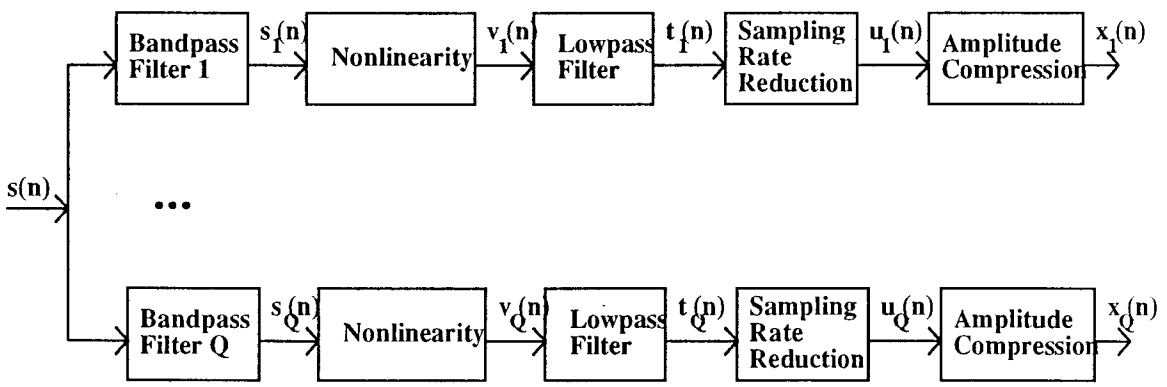

Figure 2.5: Filter-Bank Analyzer

The effects of the nonlinearity and the LPF are illustrated in Figure 2.6, which shows typical waveforms of the following signals:

- input speech, $s(n)$ (a 32 millisecond segment of voiced speech digitized at 8 kHz)
- i-th BPF output $s_i(n)$: the BPF used in this example is centered at 500 Hz, a frequency around which the formant frequencies for a few of the most common vowel sounds are situated
- non-linearity output, $v_i(n)$: the non-linearity in this example is the full-wave rectifier (FWR) discussed in detail below
- LPF output, $t_i(n)$: the cut-off frequency for the LPF is 80 Hz, with an attenuation of 50 dB over a 20 Hz interval, [80 Hz, 100 Hz]. its choice is related to the fastest motion rate of the speech harmonics in a narrow band, which is on the order of 50-100 Hz.

as well as their corresponding Fourier transforms. The output $x_i(n)$ of the filter bank analyzer shown in Figure 2.3 is a downsampled and amplitude compressed version of $t_i(n)$ from Figure 2.6.

A FWR f was used as the nonlinearity:

$$f(s_i(n)) = \begin{cases} s_i(n) & \text{for } s_i(n) \geq 0 \\ -s_i(n) & \text{for } s_i(n) < 0 \end{cases} \tag{2.4}$$

This nonlinearity can be represented as:

$$v_i(n) = f(s_i(n)) = s_i(n)w(n) \tag{2.5}$$

where

$$w(n) = \begin{cases} +1 & \text{if } s_i(n) \geq 0 \\ -1 & \text{if } s_i(n) < 0. \end{cases} \tag{2.6}$$

The nonlinearity output can be viewed as a modulation in time, operation which translates to convolution in frequency domain:

$$V_i(e^{j\omega}) = S_i(e^{j\omega}) * W(e^{j\omega}) \tag{2.7}$$

where $V_i(e^{j\omega})$, $S_i(e^{j\omega})$ and $W(e^{j\omega})$ are the Fourier transforms of the signals $v_i(n)$, $s_i(n)$ and $w(n)$, respectively. It can be seen that $S_i(e^{j\omega})$ has most of its energy in the pass band (the maximum at approximately 500 Hz), while the spectrum of the signal after the full-wave rectification, $V_i(e^{j\omega})$, shows a corresponding low-frequency

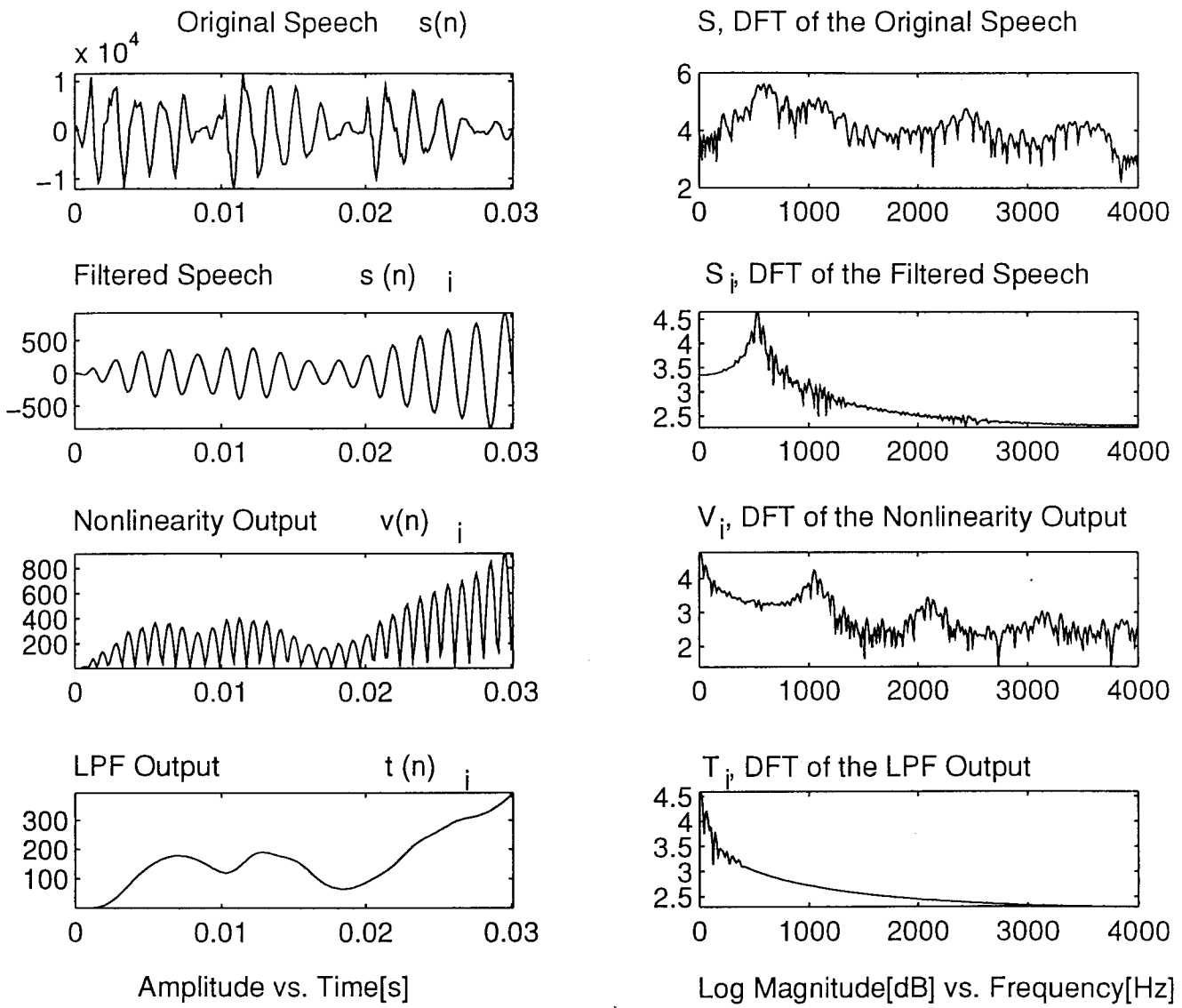

Figure 2.6: Filter Bank Analysis of A Speech Waveform

concentration of energy. The undesired peaks at higher harmonic frequencies, visible in the shape of $V_i(e^{j\omega})$ are eliminated by the LPF, producing the desired spectral estimate, $t_i(n)$.

The bandwidth of the signal $v_i(n)$ is related to the fastest rate of motion of speech harmonics in a narrow band and is on the order of 50-100 Hz. In order to achieve an economy in signal representation, the output of the LPF filter is resampled at a rate of 100-200 Hz. The signal dynamic range is compressed using an amplitude compression scheme, such as logarithmic or μ-law encoding.

2.3.1 Types Of Filter Banks

Among the types of filter banks used for spectral analysis in speech recognition [2], the most common one is the uniform filter bank. In this case the filter frequency responses are equally spaced, the center frequency f_i of the i-th bandpass filter being defined as:

$$f_i = \frac{F_s}{2N} i, \quad \forall i = 1, ..., N, \tag{2.8}$$

where F_s is the sampling rate of the speech signal, and N is the number of uniformly spaced filters required to span the frequency range of the speech. The bandwidth b_i generally satisfies the property:

$$b_i \geq \frac{F_s}{2N} \tag{2.9}$$

with equality meaning that there is no frequency overlap between adjacent filter channels.

The alternative to the uniformly spaced frequency plan is the non-uniform spacing of filter banks, designed according to a given perceptual criterion. The best known criteria for designing non-uniform filter banks are [2]:

- uniform spacing on a logarithmic frequency scale,
- spacing according to the critical band scale, which is based on auditory perceptual studies,
- variants on the critical band scale, such as the mel scale and the Bark scale.

The *critical band* refers to the bandwidth at which subjective responses, such as loudness, become significantly different. The loudness of a band of noise remains

constant as the noise bandwidth increases up to the width of the critical band; after that increased loudness is perceived. Similarly, a complex sound (composed of only several tones) of constant intensity is approximately as loud as an equally intense pure tone of frequency equal to the center of the band. This observation was used to obtain the critical bandwidth as a function of frequency (the center frequency of the band) [10].

The *mel pitch scale* was defined as a result of psychophysical studies which have shown that human perception of the frequency content of sounds does not follow a linear scale [10]. This research has led to the idea of defining a *subjective pitch* of pure tones: for each tone with frequency f, a subjective pitch is measured on a scale called the *mel* scale. As a reference point, the pitch of a 1 kHz tone, 40 dB above the perceptual hearing threshold, is defined as 1000 mels. Other subjective pitch values are obtained by adjusting the frequency of a tone such that it is half or twice the perceived pitch of a reference tone (with known mel frequency). The relationship between the mel frequency M, in mels, and the frequency f, in kHz, of the tone, is given by:

$$M = 1000\, log_2(1 + f). \tag{2.10}$$

The pitch is perceived with more accuracy for sounds of low frequency, fact reflected also by the linear dependency between the subjective pitch and the logarithm of the frequency, towards high frequencies.

The subjective nonlinear perception of frequency has led to an objective computational model that provides a mechanism to convert a physically measured spectrum of a given sound into a psychological, *subjective spectrum.* In the *Bark scale* model, each frequency component of the spectrum f is replaced by a specific loudness level B according to an empirical power law over a range of *tonalness units*:

$$B = 13 \arctan(0.76f) + 3.5 \arctan(\frac{f}{7.5})^2, \tag{2.11}$$

where f is expressed in kHz. A unit of tonalness corresponds in width to a critical band and is called a Bark.

2.4 The Linear Predictive Coding Model

Linear prediction is used in speech recognition as a method of estimating the speech spectrum over short time intervals (10-30 msec), in which the signal can be approximated as stationary. In this method, each input sample is estimated, or *predicted*, from previous input samples.

In the LPC model, the estimate (or predicted value) $\hat{s}(n)$ of a speech sample $s(n)$ is defined as a linear combination of the past P samples, such that:

$$\hat{s}(n) = a_1 s(n-1) + a_2 s(n-2) + ... + a_P s(n-P) \tag{2.12}$$

where the LPC coefficients $a_1, a_2, ...a_P$ are assumed constant over the speech analysis frame.

The prediction error is the difference between the original signal and the signal estimate:

$$e(n) = s(n) - \hat{s}(n) = s(n) - \sum_{i=1}^{P} a_i s(n-i) \tag{2.13}$$

The prediction error filter transfer function, expressed in the z-transform domain, is:

$$A(z) = \frac{E(z)}{S(z)} = 1 - \sum_{i=1}^{P} a_i z^{-i}. \tag{2.14}$$

and represents the transfer function of the linear predictor with input $s(n)$ and output $e(n)$.

For the particular case of the infinite order predictor, when $P \rightarrow \infty$, it can be shown that the stationary input signal is transformed into a white noise process [12], and this is why the filter $A(z)$ is also called the *whitening filter*. The consequence of this property is that the inverse of the infinite order whitening filter, $1/A(z)_\infty$ will reconstruct the original signal $x(n)$ from a white noise signal (where the subscript ∞ indicates the optimal infinite-order prediction).

Another property of the optimal infinite-order predictor is that it contains all the information regarding the signal's power spectral density (PSD) shape. For a system with transfer function $A(z)$ given by 2.14, the PSD of the input, $P_{xx}(\omega)$, and that of the output, $P_{ee}(\omega)$, are related by:

$$P_{ee}(\omega) = |A(e^{j\omega})|^2 P_{xx}(\omega) \tag{2.15}$$

which becomes, in the case of infinite order prediction:

$$P_{xx}(\omega) = \frac{{\sigma_w}^2}{|A_\infty(e^{j\omega})|^2}, \tag{2.16}$$

where ${\sigma_w}^2$ is the variance of the white noise process w at the output of $A_\infty(z)$.

Practically, a good short-time estimate of the speech signal's PSD can be obtained with a finite order predictor, with order P between 10 and 20 [14]. This property is used in spectral estimation by the model-based (parametric) approach.

The basic problem of linear prediction analysis is to determine the set of predictor coefficients, $\{a_i\}$, $i = 1, ..., P$, directly from the speech signal so that the spectral properties of the filter match those of the speech waveform within the given segment. The criteria for finding the optimal predictor coefficients is the minimization of the mean-squared prediction error over a speech segment of short duration:

$${\sigma_e}^2 = E\{e^2(n)\} \tag{2.17}$$

By replacing $e(n)$ with (2.13), and by taking the derivative with respect to each coefficient a_i and setting the result to zero, the following system of equations for the optimal predictor coefficients is found (also known as the Wiener-Hopf or the Yule-Walker equations):

$$\sum_{j=1}^{P} a_j r(|j-i|) = r(i), \quad \forall i = 1, ..., P \tag{2.18}$$

where, if N is the size of the speech segment in samples, then

$$r(k) = E\{s(n)s(n-k)\} \tag{2.19}$$

is the signal's autocorrelation function.

Two approaches are used in practical applications to compute the linear predictor coefficients:

1. The *Autocorrelation Method*: uses a weighted version of the input speech segment, obtained by multiplication with a finite length window (rectangular, Hamming). The autocorrelation function in (2.19) is in this case replaced by an estimate which uses the windowed signal $s_w(n) = s(n)w(n)$:

$$r_w(k) = \sum_{m=0}^{N-1-|k|} s_w(m)s_w(m+|k|) \tag{2.20}$$

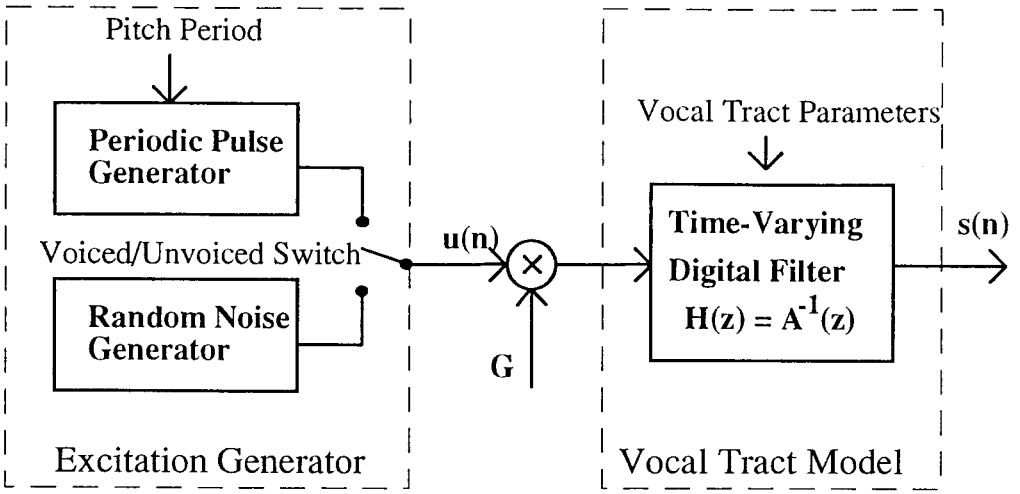

Figure 2.7: Speech Synthesis - LPC Model

2. The *Covariance Method*: minimizes the actual least square error on the given speech segment. The predictor coefficients are the solutions of a system similar to (2.18):

$$\sum_{j=1}^{P} a_j \phi(i,j) = \phi(i,0), \quad \forall i = 1, ..., P \tag{2.21}$$

in which the autocorrelation function is replaced by the short-term "covariance" of the signal, defined as:

$$\phi(i,j) = \sum_{m=0}^{N-1} s(m-i)s(m-j), \quad \forall i = 1, ..., P, \quad \forall j = 0, ..., P. \tag{2.22}$$

The solution to the autocorrelation method can be obtained using a very efficient algorithm (Levinson-Durbin), which reduces significantly the computational complexity in comparison with solutions obtained using the second method. The autocorrelation method also offers the advantage of always having as solution a stable inverse filter, while the covariance method usually requires a stabilization procedure.

The LPC coefficients are related to the vocal tract parameters in the speech production model presented in Figure 2.7. The model consists of two basic elements:

- the *excitation generator*, which models the effect of the air flow through the vocal chords. The excitation function is quasiperiodic for voiced segments of speech and random for unvoiced speech.
- the *vocal tract model*, which accounts also for the effect of radiation at the lips. The vocal tract parameters vary slowly in voiced sounds, but this approximation is not valid for transient sounds [14].

The parameters that completely describe the speech production model are: the voiced/unvoiced classification, the pitch period for voiced sounds, the excitation gain, and the coefficients of the filter modeling the vocal tract, all of which vary with time. For the linear system described in Figure 2.7 the input to the LPC-based synthesis filter, $e(n)$, equals $Gu(n)$, the scaled excitation, where $u(n)$ is the normalized excitation and G is the gain of the excitation. The transfer function $H(z)$ is the inverse of the whitening filter:

$$H(z) = \frac{1}{A(z)} \tag{2.23}$$

2.5 The LPC Processor for Speech Recognition

A typical LPC front-end processor used in speech recognition applications is presented in Figure 2.8.

During *Preemphasis*, the digitized speech signal, $s(n)$, is spectrally flattened, to compensate for the inherent spectral tilt of the signal. The *Preemphasis* block consists usually of a first order FIR filter. The preemphasized speech signal is then blocked in frames of N samples each, with adjacent frames being separated by M samples, with $M << N$ (typically, $N = 3M$ [2]). The overlap insures that the contribution of all the samples in the frame is properly considered in the evaluation of the LPC spectral estimates, and that the transition between values corresponding to adjacent frames is smooth.

Each individual frame is multiplied by a window so that the signal discontinuities at the extremities of the frame are minimized. The LPC coefficients are computed for

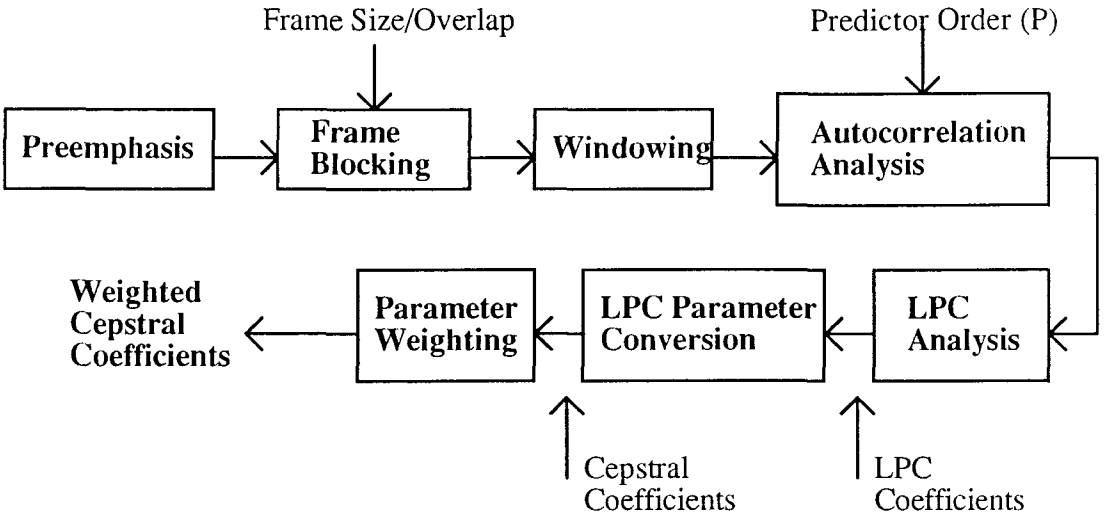

Figure 2.8: LPC Processor for Speech Recognition - Block Diagram

each frame: the windowed autocorrelation is estimated (the first P values, where P is the predictor order) and then the Levinson-Durbin algorithm is used to determine the LPC coefficients. The number of iterations in the algorithm is equal to the predictor order chosen.

Other equivalent coefficients, such as the reflection coefficients, the log area ratio coefficients, or the cepstral coefficients, are determined during the *LPC Parameter Conversion* stage of the block diagram. The cepstral coefficients (which are the coefficients of the Fourier transform representation of the log magnitude spectrum) are considered as the most robust and reliable feature set, used in recognition, among the above alternatives [2]. The cepstral representation requires an increase in the number of iterations in the Levinson-Durbin algorithm, due to the fact that more coefficients than the predictor order are necessary, to obtain a similar accuracy of the spectral representation.

The *Parameter Weighting* block consists of bandpass filtering in cepstral domain, transformation which minimizes the variations due to noise of low-order and high-order coefficients.

Chapter 3

Vector Quantization In Speech Recognition

Quantization is, in the simplest form, the operation which assigns to any scalar value from a continuous range the nearest approximation (uniquely defined) from a finite set of values. An immediate example for scalar quantization is the digitization of an analog signal. The generalization of the above definition to the quantization of a vector (an ordered set of values) is known as *vector quantization.*

Vector quantization is commonly used in data compression, due to the high compression ratios achieved: the input vector is mapped into an index in a finite output set, and the index corresponds to the vector selected as the best approximation for the input vector. Vector quantization can also be viewed as a form of pattern recognition, where an input pattern (described by a vector) is assigned to one of a predetermined set of standard patterns, or templates. In speech recognition, vector quantization is used both as a complexity reduction technique, because it provides an efficient representation of data, and as a classification method for the input.

This chapter presents an introduction to vector quantization, the design procedure of a vector quantizer (VQ), and the application of quantization to speech recognition systems.

3.1 Vector Quantization

Vector quantization is a very efficient compression technique, in which every vector of consecutive input samples is encoded into an integer, or *index*, that is associated with an entry of a collection of reproduction vectors, or *codebook*. The reproduction vector, or *codeword*, chosen is the one that is closest to the input vector in a specified distortion sense. The coding efficiency is achieved in converting the vector into a compact integer representation, which ranges from 1 to N, with N being the size of (number of entries in) the codebook.

More specifically, a VQ performs a mapping Q from a vector $\underline{x}$ (in a k-dimensional vector space X) into a finite set of output vectors $C = \{\underline{y}_j\}_{j=1}^{N}$ with $\underline{y}_j \in X, \forall j$.

$$Q : X \rightarrow C \tag{3.1}$$

The set C represents the *codebook*, and each vector belonging to the set is called a *codevector*. The mapping described above defines also a partition, S_j, of the vector space X with N regions, or *cells*, where S_j consists of all the input vectors $\underline{x}$ which will be quantized into codevector $\underline{y}_j$:

$$S_j = \{\underline{x} \in X : Q(\underline{x}) = \underline{y}_j\} \tag{3.2}$$

The association of an input vector to a given cell is based on a "distance" or *distortion measure* between the two vectors, i. e. the input vector and the codevector representing the partition. A *metric* (distance) function d on a vector space X

$$d : X \times X \rightarrow R \tag{3.3}$$

satisfies the properties of positive definiteness, symmetry and the triangle inequality condition. If a measure of difference satisfies only the positive definiteness property, it is referred to as a *distortion measure*:

$$d(\underline{x}, \underline{y}) = \begin{cases} 0 & \text{if } \underline{x} = \underline{y} \\ > 0 & \text{otherwise} \end{cases} \tag{3.4}$$

The set of codewords, which forms the VQ codebook, must be chosen such that it minimizes the average distortion:

$$D = E\{d(\underline{x}, Q(\underline{x})), \tag{3.5}$$

where Q is the quantization function and $Q(\underline{x})$ the quantized value of the input.

In practical systems, the distortion is estimated by considering the average:

$$\bar{D} = \frac{1}{L}\sum_{i=1}^{L} d(\underline{x}_i, Q(\underline{x}_i)), \tag{3.6}$$

where $\{\underline{x}_i\}$ is a sequence of L input vectors; the larger the number of input vectors over which the estimate is evaluated, the better the estimation accuracy is.

With the optimality criterion for quantizer design defined by 3.5, it can be shown that the necessary conditions for a quantizer to be optimal are [12]:

1. *The Nearest Neighbor Condition*: for a given codebook C, an input vector $\underline{x}$ is assigned to the partition containing the "nearest" codevector:

$$d(\underline{x}, Q(\underline{x})) = \min_{\underline{y}_j \in C} d(\underline{x}, \underline{y}_j) \tag{3.7}$$

where j spans the entire range for codebook indices. The condition can also be expressed in terms of the quantizer's output, as:

$$Q(\underline{x}) = \underline{y}_i \text{ if and only if } d(\underline{x}, \underline{y}_i) \leq d(\underline{x}, \underline{y}_j), \forall j \neq i. \tag{3.8}$$

2. *The Centroid Condition*: for a given partition S_j, $j = 1, ..., N$, of X, the optimal codevectors satisfy

$$\underline{y}_j = \text{centroid}(S_j) \tag{3.9}$$

where the *centroid* is defined as the vector which minimizes the average distortion for the given cluster of input vectors $\underline{x} \in S_j$:

$$\underline{y}_j = \min_{\underline{y} \in X}{}^{-1} E\{d(\underline{x}, \underline{y}) \,|\, \underline{x} \in S_j\}. \tag{3.10}$$

The existence of a unique centroid has been proven for distortion measures of interest [12].

For the squared error distortion measure, denoted by $||\cdot||$:

$$d(\underline{x}, Q(\underline{x})) = ||\,\underline{x} - Q(\underline{x})\,||^2 \tag{3.11}$$

the optimality conditions become:

1. *The Nearest Neighbor Condition*: for a given codebook, the partition S_j, $j = 1, ..., N$ of X, must satisfy the following condition:

$$S_j = \{\underline{x} \in X : ||\underline{x} - \underline{y}_j|| \leq ||\underline{x} - \underline{y}_i||, \ \forall i = 1, ..., N\} \tag{3.12}$$

2. *The Centroid Condition*: the centroid is the minimum mean squared estimate of $\underline{x} \in S_j$:

$$\text{centroid}(S_j) = E\{\underline{x} \,|\, \underline{x} \in S_j\} \tag{3.13}$$

and is uniquely defined. For a finite input set:

$$\tau = \{\underline{x}_1, \underline{x}_2, ..., \underline{x}_L\}, \tag{3.14}$$

the centroid condition 3.13 can be evaluated as:

$$\text{centroid}(S_j) = \frac{\sum_{i=1}^{L} P_j(i)\underline{x}_i}{\sum_{i=1}^{L} P_j(i)} \tag{3.15}$$

where $P_j(i)$ is the probability of each vector $\underline{x}_i$ to be clustered in S_j.

The optimality conditions can be used for the improvement of a VQ codebook, procedure known as *codebook training*, by minimizing the average error over a training data set. Following codebook training, the characteristics of the information source that produced the given training data are embedded in the codebook.

The fact that a VQ is optimally designed for a particular source (i.e. it will achieve a lower average distortion for signals generated by the source than any other VQ not designed for that particular source) suggests that a VQ can be successfully used as a pattern classifier. Details regarding elements of a VQ, such as the training set and the distortion measure, are presented in the next section.

3.2 Structural Properties of a VQ

The basic VQ training and classification structure is shown in Figure 3.1. It is assumed that the input to the VQ consists of the results of the spectral feature extraction block, which are a series of vectors $\underline{v}_l$, $l = 1, ..., L$, characteristic of the time-varying spectral representation of the speech signal.

The following items are required for implementing a VQ in the context of speech recognition:

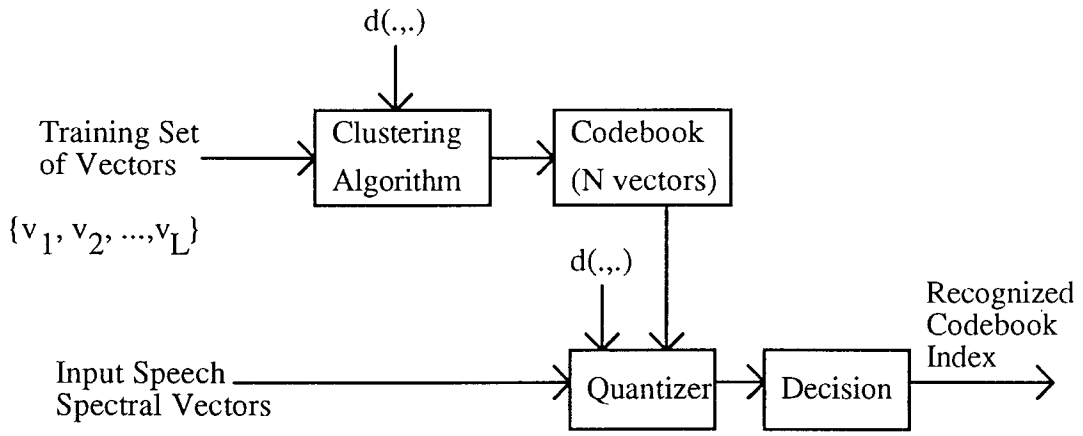

Figure 3.1: Block Diagram of the Basic VQ Training and Classification Structure

1. A large set of spectral analysis vectors, which form a *training set.*
2. A *spectral distortion measure*, which is a measure of similarity, or distance, between a pair of spectral analysis vectors, allowing to cluster the training set vectors as well as to associate or classify arbitrary spectral vectors into unique codebook entries.
3. A *centroid computation procedure*: on the basis of the partitioning that classifies the L training set vectors into N clusters, the N codebook vectors are chosen as the centroid of each of the clusters.
4. A classification procedure for arbitrary speech spectral vectors that chooses the codebook vector closest to the input vector and uses the codebook index as the resulting spectral representation. This is also referred to as the *nearest-neighbor labeling* or *optimal encoding* procedure.

3.2.1 The VQ Training Set

The training set of vectors must be representative of the speech source used in recognition. For a speaker independent recognizer, it should span a wide range of talkers (including ranges in age group, accent, gender, speaking rate, levels) and speaking conditions (such as quiet room, automobile noise, workstation noise, etc.).

The training set is used to create the "optimal" set of codebook vectors for representing the spectral variability of the source. Because the source distribution is generally unknown in practical applications, a large number of training vectors must be used, to provide a good empirical characterization of the source.

Assuming that the size of the VQ codebook is $N = 2^B$ vectors (for a so-called B-bit codebook), then L, the size of the training set, must be a factor of 10 to 100 times larger than N; this ratio is called the *training ratio.*

3.2.2 Spectral Distortion Measures

The distortion measure is a key component of most pattern-comparison algorithms and must be defined according to the nature of the data to be quantized. When the input signal is speech, an important consideration in choosing a distortion measure is its subjective meaningfulness. A detailed review of perceptual considerations in defining spectral distortion measures can be found in [2].

The distance measures commonly used for comparing filter-bank vectors are:

- the mean absolute spectral distortion, L_1
- the root mean square log spectral distortion, L_2
- the covariance weighted or Mahalanobis spectral difference [12].

For LPC vectors and related feature sets, measures such as the likelihood and cepstral distances are preferred [2].

The set of norms L_p, $p = 1, 2, ...$ known as log spectral distances, are defined as:

$$L_p = d(\underline{x}, \underline{x}')^p = (d_p)^p = \frac{1}{Q} \sum_{i=1}^{Q} |\log(x_i) - \log(x_i')|^p. \tag{3.16}$$

where $\underline{x}$ and $\underline{x}'$ are two normalized spectral vectors of dimension equal to the number Q of filters in the feature extraction block.

Since the perceived loudness of speech is approximately logarithmic, the log spectral distance appears to be closely related to the subjective evaluation of sound differences and is considered a perceptually relevant distortion measure. It can also be shown that the L_p measures are metrics because they satisfy the conditions of positive definiteness, symmetry and the triangle inequality.

The log spectral distances are much smaller for versions of the same sound, than when different sounds are compared. This property is exploited by accumulating spectral distortions over time when comparing utterances.

The other type of distortion measure used with filter-bank analyzers is the Mahalanobis distortion measure, a particular case of the weighted squared error measure:

$$d(\underline{x}, \underline{y}) = (\underline{x} - \underline{y})^T W (\underline{x} - \underline{y}) \tag{3.17}$$

where W is the inverse of the covariance matrix of the input, and $\underline{x}$ and $\underline{y}$ are column spectral vectors.

The computation of log spectral features is performed very efficiently by analog pre-processors, such as the filter bank, but is very demanding computationally if a digital signal processing front-end is used [2]. The covariance weighted method is also demanding computationally, due to the evaluation of the covariance matrix of the input vector. Among the distortion measures presented above, L_1 has the lowest implementation complexity, and was the method of choice for the baseline implementation of the recognizer.

All the distortion measures presented above are designed to compare two static spectral representations, usually short-time estimates of the speech signal. These distances can be used for sequences of spectra by accumulating their values over time, but this procedure does not reflect the dynamic characteristics of the sequence. Several methods of incorporating the spectral dynamic features into the distortion measure are discussed in the next Chapter.

3.3 VQ Design

The objective of VQ design is to define a codebook and a partition, or encoding rule, that will maximize the VQ performance. The sufficient conditions of optimality, which would generate a closed-form solution for the optimal quantizer, are not known [12].

The optimality conditions (which are only necessary conditions) mentioned in Section 3.1 are at the basis of defining iterative improvements of a given VQ codebook. The Nearest Neighbor condition 3.12 defines a rule for finding the best partition given a codebook C, while the Centroid condition 3.13 defines the optimal codeword $\underline{y}_i$ for each cell S_i in a given partition.

The iteration begins with a VQ consisting of an initial codebook C and a training set which is clustered into a partition, according to the Nearest Neighbor condition. The next step is finding if there are any empty partition cells, that have no vectors assigned after the clustering is finished. The new centroids of non-empty cells (computed according to the centroid condition), together with the codewords assigned as centroids to empty cells (if any), form the improved codebook C'.

This iterative process is known as the Lloyd iteration, and is the basis of the generalized Lloyd algorithm for VQ design, which is a form of the *k-means* algorithm. The application of the necessary conditions for optimality at each step of the algorithm ensures that each iteration reduces or leaves unchanged the average distortion.

Although this algorithm leads to an improved version of the original codebook, it doesn't guarantee the global optimality of the resulting quantizer. However, practical implementations of the algorithm have been found to be very effective [12].

3.3.1 Codebook Initialization

The initialization conditions of the iterative algorithm for generating an improved codebook assume that a training set of vectors and a codebook are available as input data.

There are a variety of techniques for generating a codebook that have been developed in the fields of pattern recognition and vector quantization, such as:

- *random selection* of the codewords according to the source distribution;
- *pruning* the training set until a final set remains as the codebook;
- codebook *splitting*,

to name but a few that were surveyed in [12].

The procedure used for codebook initialization in this implementation is the *splitting algorithm* or the *LBG algorithm*, which produces increasingly larger codebooks, until the desired codebook size is reached [13]. The algorithm is described as follows:

1. **Initialization**: start with a one-vector codebook, which is the centroid of the entire training set.
2. **Splitting**: a codeword is split in two other codewords, thus incrementing the size of the codebook:

$$\underline{y}^{+} = \underline{y} + \epsilon \tag{3.18}$$

$$\underline{y}^{-} = \underline{y} - \epsilon \tag{3.19}$$

   where ϵ is a vector of small euclidean norm compared to the other training vectors.
3. **Training**: iterative improvement of the split codebook using the k-means algorithm. This step does not change the codebook size.
4. **Completion Test**: if the size of the codebook has not been reached yet, steps 2 and 3 are iterated.

3.3.2 Codebook Improvement

The application of the k-means algorithm to a given codebook and consists of the following steps:

1. **Initialization**: start with a codebook, and a large training set.
2. **Nearest Neighbor Search**: each of the vectors in the training data are classified into one of the clusters, according to the minimum distortion measure criterion (3.12).
3. **Average Distortion Computation**: the contribution of each input vector to the distortion measure is the minimum distance, determined during step 2, with respect to the codeword selected as the best approximation for the input vector.
4. **Centroid Computation**: for each non-empty partition, the new codeword will be computed according to 3.15 from all the training vectors

clustered in it. The codeword assigned to an empty cell can be obtained in a number of ways, such as by splitting the centroid of the most populated cell or that of the cell with the highest partial distortion.

5. **Completion Test**: if the change in average distortion falls below some predetermined threshold, the optimization is complete; otherwise another iteration of steps 2 to 4 is performed.

3.4 Applications to Speech Recognition

In the design of VQ-based recognition systems, the following factors which influence the performance must be considered:

- The level of quantization error in representing the analysis vector. Since there is only a finite number of codebook vectors, the process of choosing the "best" representation of a given spectral vector is equivalent to quantizing the vector, and leads, by definition, to a certain level of distortion. As the size of the codebook increases, the size of the quantization error decreases. A compromise must be reached between how the representation accuracy reflects in the recognition performance and the amount of storage available for the codebooks.

- The codebook storage requirement can become too large for recognition systems with large vocabularies. Hence a trade-off among quantization error, processing for choosing the codebook vector, and storage of codebooks must be reached.

The recognition system implemented for this study has the structure presented in Figure 3.2. By comparing Figure 3.1 to the generic block diagram of a pattern recognizer presented in Figure 2.1 it can be seen that the pattern classification structure corresponds to the VQ structure. Indeed, suppose there are V utterance classes (words, phrases) to be recognized, and a separate codebook is built using as training data only utterances corresponding to one class. During quantization, each of the V codebooks is used in turn by the VQ to compute an average distortion score, ϵ_i for $i = 1, ..., V$, and when V scores are computed, the minimum is selected. The recognition decision is the class index corresponding to the minimum distortion score. The V codebooks are analogous to V (sets of) reference patterns (templates) in Figure 2.1.

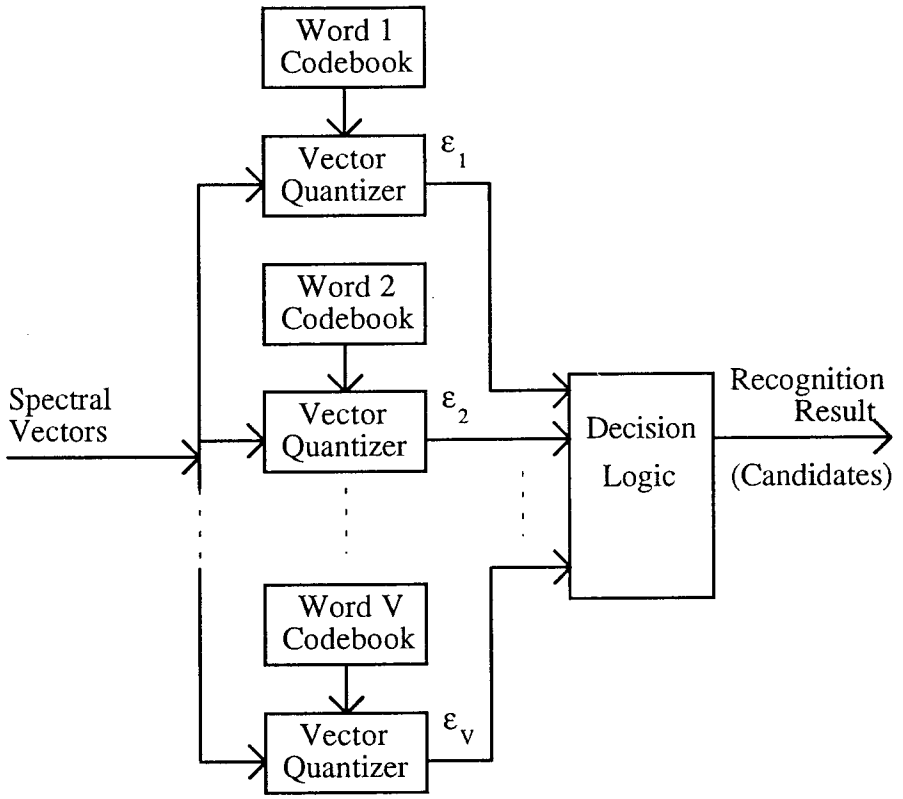

Figure 3.2: VQ-Based Classifier

This method of using a VQ in the recognition process is referred to as "pattern comparison without time alignment", to distinguish it from other procedures that use time alignment, and is used for highly non-confusable vocabularies. From the block diagram of a VQ-based recognition system, presented in Figure 3.2, it can be seen that the decision block can be used not only to find the minimum distortion score candidate, but also to screen out word candidates that are very unlikely to match the unknown utterance. The VQ can then pass the rest of the candidates to a more sophisticated decision block, thus acting as a "pre-processor".

Chapter 4

Temporal Information in Recognition

The performance of VQ-based recognition systems that use the accumulated spectral distortion over the duration of an entire utterance is adequate only if the vocabulary of the application is highly non-confusable, with non-overlapping phonetic content. Improvements in recognition accuracy can be obtained by eliminating variations in utterance duration and spectral features alignment and by using during quantization discriminability criteria such as the spectral dynamic behavior and the order of occurrence of significant features within the spectral profile.

Utterances representing the same class, or word, have different time durations, a fact which can be observed by examining the log energy contours presented in Figure 4.1 for two such utterances. This difference in duration can be easily compensated by normalizing both utterances to have a fixed length in time. The time normalized versions of the two energy profiles are presented in Figure 4.2, which shows that normalization is not sufficient to produce perfectly aligned patterns: indeed, the temporal locations of the vowel peaks are also slightly different.

This analysis shows that it is desirable to normalize the speaking rate fluctuation and to align the speech patterns before a recognition decision can be made. The solution to the problem of time aligning speech patterns is known in speech recognition as *dynamic time warping* (DTW), and will be presented in Section 4.1 of this chapter. DTW increases the recognition performance, but in the same time increases the complexity of the recognizer's implementation, due to the computationally intensive

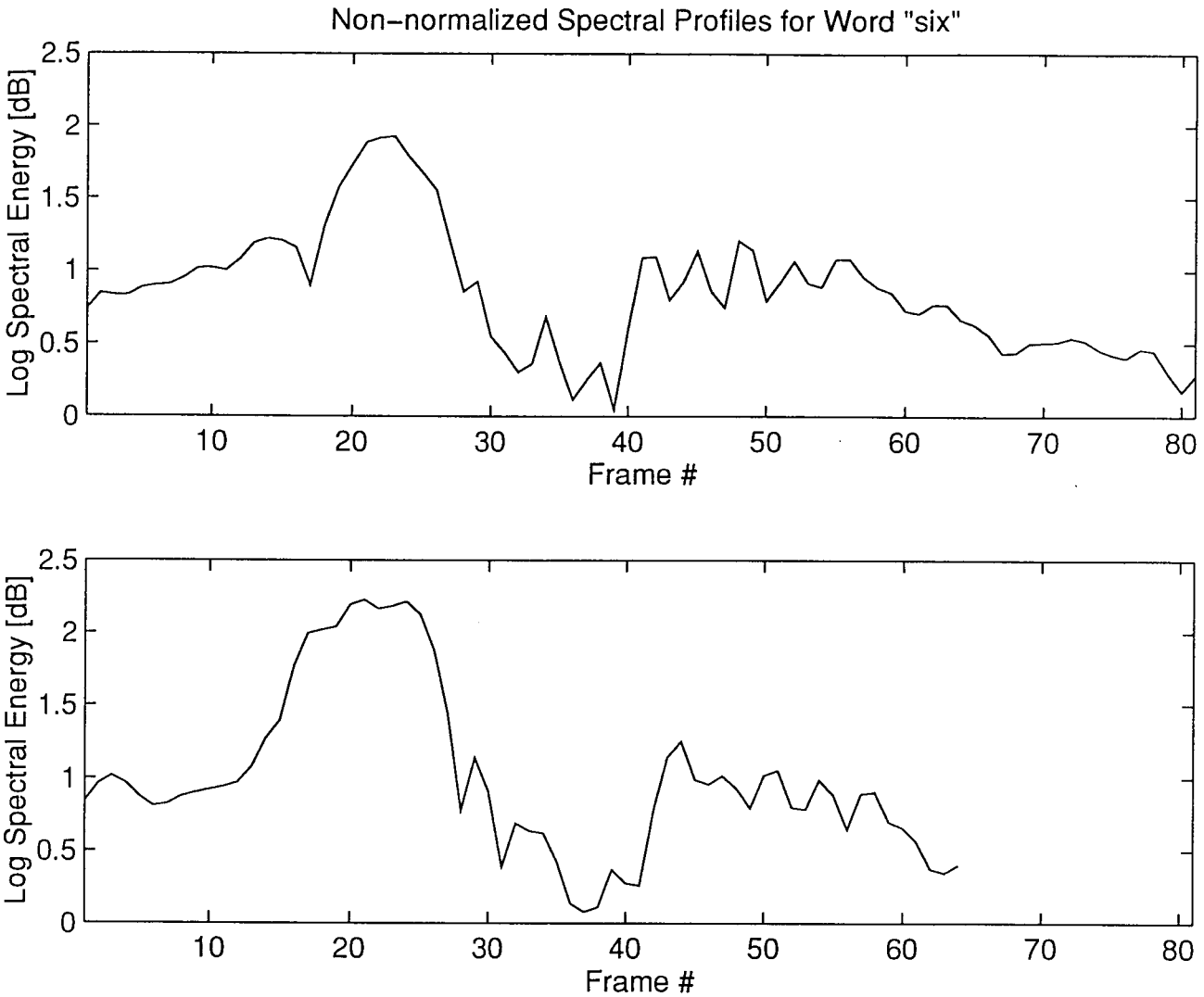

Figure 4.1: Non-normalized Utterances

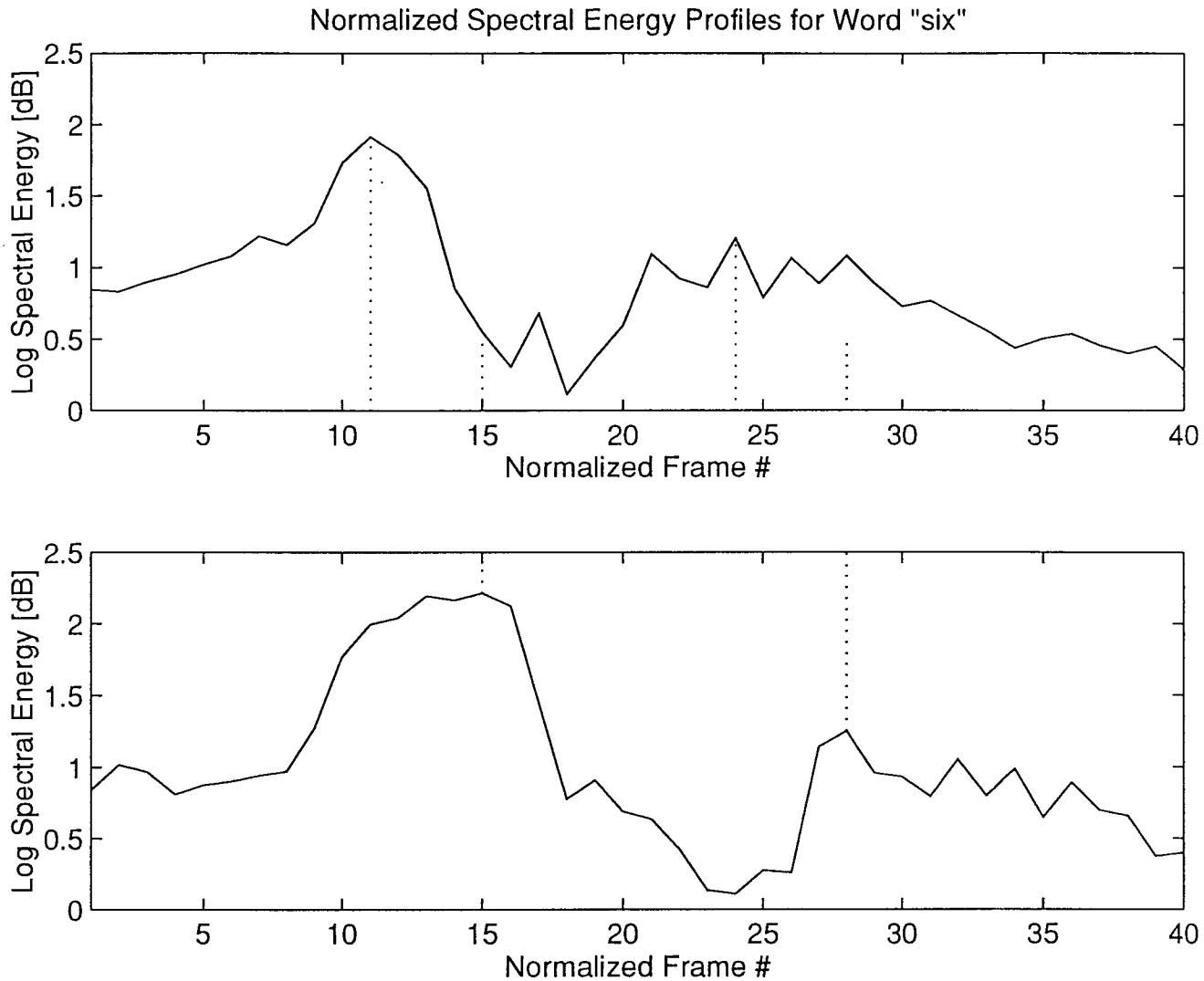

Figure 4.2: Time Normalized Utterances

time alignment procedure.

Less complex VQ-based recognition implementations, with satisfactory performance for small vocabularies, use the methods known as pattern recognition without time alignment. These recognition systems make use of the temporal information during quantization and are presented in Section 4.2. Experimental results have shown that dynamic features of the speech spectrum contribute significantly to the overall recognition [15]. As a result, distortion measures based on the variational spectral features have been used to improve the recognition performance. Another factor which influences the recognition performance is the sequence of occurrence (temporal order) of spectral features, especially for vocabularies containing utterances with overlapping phonetic content, such as "car" or "rack". The following VQ based recognition methods use temporal information during quantization:

- *matrix VQ*, which quantizes an entire block of short-time features;
- *trellis VQ*, which contains information regarding the relationship between adjacent segments in time;
- *segmental VQ*, which uses a separate codebook for each individual segment of an utterance.

A VQ-based recognition method which offers better recognition performance and a high reduction in complexity (10-20 times less than DTW) was developed by Pan et. al. [7]. In a more formal attempt to characterize the temporal behavior of the speech patterns, and to incorporate it in the design of the VQ recognizer, the method evaluates the probability density function (PDF) of the time of occurrence for the spectral vectors in the codebook and, based on it, creates probability tables which are used in conjunction with the spectral codebooks during quantization. This approach is referred to in this thesis as the *temporal probability tables* method and is presented in more detail in Section 4.2.3.

Recognizers using segmental VQ and the probability tables method were implemented as a reference for evaluating the performance of the new spectral-temporal quantization methods proposed in this thesis. The comparison results, presented in Chapter 7, show that the new methods offer an improvement in performance, with lesser or equal computational and storage requirements, than the existent methods.

4.1 Pattern Comparison With Time Alignment

The essential component of pattern recognition methods with time alignment is the normalization and time warping of the patterns in order to eliminate the effects of speaking rate variation and pattern length on the recognition accuracy. These methods emerged as solutions to the problem of comparing speech utterances which, although representing the same vocabulary word (class), can have significantly different durations and profile dynamics. Normalization and alignment, performed in the context of preserving the sequential order of the spectral characteristics, generate consistent spectral pairs which can then be directly compared in the decision process.

4.1.1 Time Normalization

In speech recognition, time normalization of input utterances compensates for the negative effects that differences in utterance duration can have on the system's performance. Normalization is achieved through a transformation, or warping, of the original spectral profiles, so that all utterances have a given fixed length in time (measured in number of short-time spectral samples).

To define the time normalization technique, two speech patterns $\mathbf{X}$ and $\mathbf{Y}$, of different durations T_x and T_y, are considered. Assuming that they are represented by the spectral sequences $(\underline{x}_1, \underline{x}_2, ..., \underline{x}_{T_x})$ and $(\underline{y}_1, \underline{y}_2, ..., \underline{y}_{T_y})$, respectively, where $\underline{x}_i$ and $\underline{y}_i$ are parameter vectors of the short time acoustic features, the dissimilarity between $\mathbf{X}$ and $\mathbf{Y}$ can be defined by considering a distance function d of the short-time spectral distortions (distortions between short-time spectra) given by:

$$d(i_x, i_y) = ||\underline{x}_{i_x} - \underline{y}_{i_y}||, \tag{4.1}$$

where $i_x = 1, 2, ..., T_x$ and $i_y = 1, 2, ..., T_y$ denote time indices of $\mathbf{X}$ and $\mathbf{Y}$, respectively, and i_y can be represented as a function of i_x in 4.1:

$$d(i_x, i_y) = d(i_x, i_y(i_x)). \tag{4.2}$$

The simplest solution to the problem of time alignment and normalization is the *linear time normalization* technique. In linear time normalization, the dissimilarity between $\mathbf{X}$ and $\mathbf{Y}$ is defined as:

$$d(\mathbf{X}, \mathbf{Y}) = \sum_{i_x=1}^{T_x} d(i_x, i_y), \tag{4.3}$$

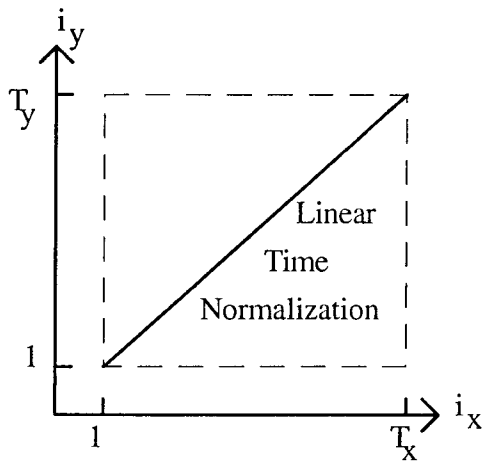

Figure 4.3: Linear Time Normalization

where i_x and i_y satisfy

$$i_y = r(\frac{T_y}{T_x}) \cdot i_x. \tag{4.4}$$

The function r represents a round-off rule which produces an integer result.

Linear time normalization and alignment is based on the assumption that the speaking rate variation is proportional to the duration of the utterance and is independent of the sound being spoken. Evaluation of the distortion measure takes place along the diagonal straight line of the rectangle in the (i_x, i_y) plane, as shown in Figure 4.3.

A more general time normalization scheme involves the use of two warping functions, ϕ_x and ϕ_y, which relate the indices of the two speech patterns, i_x and i_y, respectively, to a common, "normal" time axis k:

$$i_x = \phi_x(k), \quad \forall k = 1, ..., T \tag{4.5}$$

and

$$i_y = \phi_y(k), \quad \forall k = 1, ..., T \tag{4.6}$$

The warping function pair $\phi = (\phi_x, \phi_y)$ represents the time normalization path in the index space.

A global pattern dissimilarity measure $d_\phi(\mathbf{X}, \mathbf{Y})$ can be defined based on the warping function ϕ, as the accumulated distortion over the entire utterance:

$$d_\phi(\mathbf{X}, \mathbf{Y}) = \sum_{k=1}^{T} d(\phi_x(k), \phi_y(k)) \frac{m(k)}{M_\phi} \tag{4.7}$$

where:

- $d(\phi_x(k), \phi_y(k))$ is a short-time spectral distortion defined for $x_{\phi_x(k)}$ and $y_{\phi_y(k)}$,
- $m(k)$ is a non-negative path weighing coefficient and
- M_ϕ is a path normalizing factor.

Figure 4.4 shows an example of the above general time normalization scheme; the solid line in the lower grid, presented Figure 4.4 (c), represents the path along which $d_\phi(\mathbf{X}, \mathbf{Y})$ is evaluated. The grid points on the path are labeled incrementally from $k = 1$ to $k = T$, where T is the normalized duration of the two patterns on the k scale. The indices i_x and i_y, as functions of the normal time scale k, are shown in Figure 4.4 (a) and Figure 4.4 (b). The requirement to maintain temporal order in the spectral representations of $\mathbf{X}$ and $\mathbf{Y}$ means that the warping functions ϕ_x and ϕ_y must be monotonically nondecreasing. If the normalization path ϕ defined above also minimizes the overall distortion between the two patterns, it can be considered as the optimal alignment path.

4.1.2 Time Alignment

The problem of finding the "best" alignment between a pair of patterns is equivalent to finding the "best" path ϕ through a grid mapping the acoustic features of one pattern to the acoustic features of another pattern. The choice of the path must be made such that the overall path dissimilarity d can be measured with consistency. For patterns representing utterances of the same word, the "best" path minimizes the distortion $d_\phi(\mathbf{X}, \mathbf{Y})$:

$$d(\mathbf{X}, \mathbf{Y}) = \min_\phi d_\phi(\mathbf{X}, \mathbf{Y}) \tag{4.8}$$

However, since a "correct" time alignment between utterances of different words does not exist linguistically, the definition for distortion as a minimum over all possible

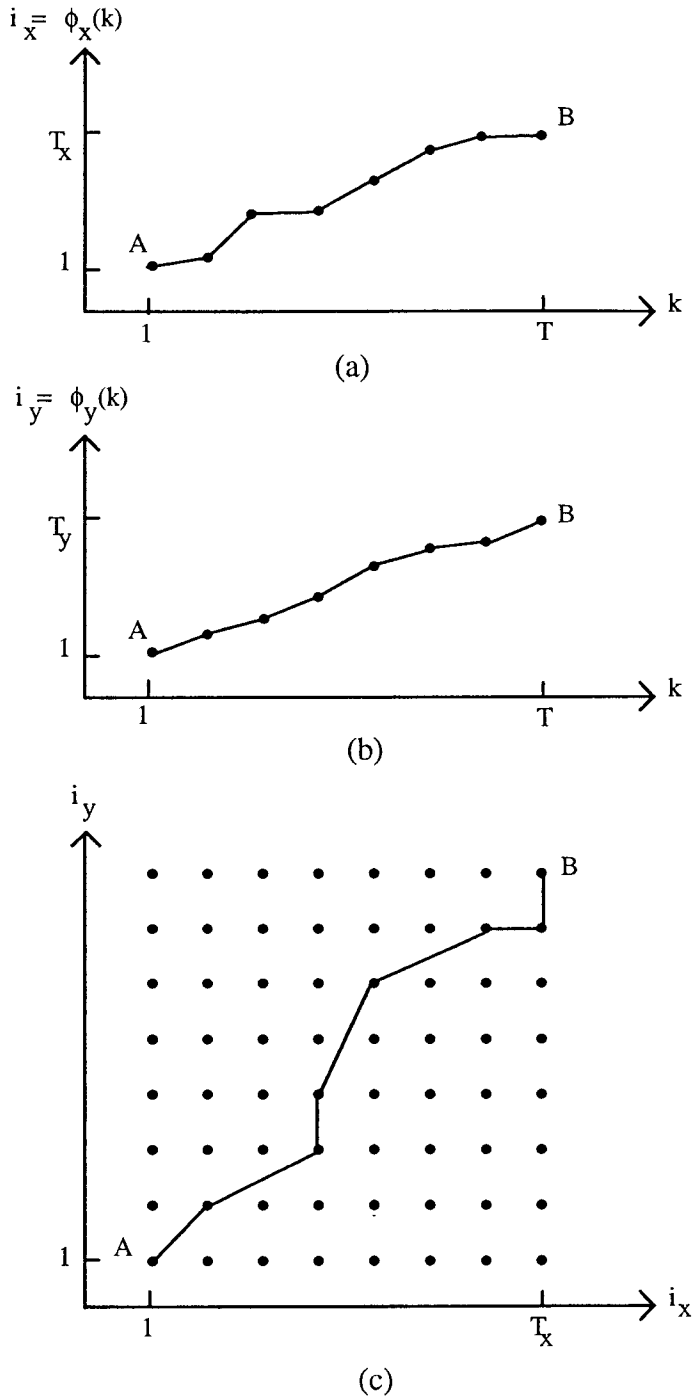

Figure 4.4: Nonlinear Time Normalization to a Common Time Index

paths can be changed so that it will reflect the structure of the vocabulary by improving the discriminability of words differing only on a small, critical portion of the spectral profile. For these cases, a discriminative weighing is introduced in the computation of the distortion measure (equations 4.7 and 4.19).

The solution to the time alignment problem can be found using dynamic programming techniques, in particular the synchronous sequential decision regarding a minimum path solution through a graph [2]. The algorithm finds the optimal sequence in a fixed number, M, of moves, starting from point A and ending at point B in the graph presented in Figure 4.4 (c), and the associated minimum distortion d.

The optimality principle at the basis of this algorithm states that, at each step in the algorithm, whatever the initial state and decision are, the remaining decisions must be optimal with the respect to the state resulting from the first decision [16]. This optimality principle is at the basis of a class of computational algorithms regarding minimal paths through graphs. For the example of finding the best m-th move in the grid from Figure 4.4 (c), with $m = 1, ..., M$, it is assumed that the minimum path for each of the N points and their associated distortions after step $m - 1$ are known to be $d_{m-1}(A, l)$, $\forall l = 1, ..., N$. For each point k in the column after the m-th move, the associated distortion will be, according to the optimality principle:

$$d_m(A, k) = \min_l [d_{m-1}(A, l) + d(l, k)] \tag{4.9}$$

The algorithm, as a result of the optimality principle, keeps track of only N paths, ending at each of the N points, at the completion of every potential move. The computational complexity of order NM, low relatively to the total number of possible paths N^{M-1}. However, in comparison to considering only linear time normalization, with no weighing, the storage and computational complexity is increased by a factor of M.

When the dynamic programming approach is used to find the time-alignment path for comparing a pair of speech patterns, a set of constraints which result from the nature of the objects being compared must be imposed to restrict the domain of the search, and thus decrease the computational complexity:

- *endpoint constraints* When the endpoints of the speech pattern are well defined prior to DTW, the following set of restrictions for the warping functions result:

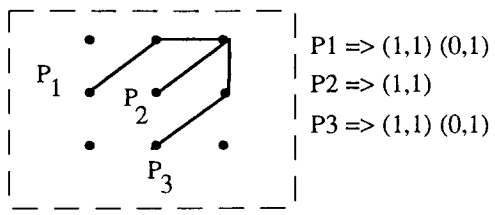

Figure 4.5: Example of Local Continuity Constraints

$$\text{Beginning point:} \quad \phi_x = 1, \phi_y = 1 \tag{4.10}$$

$$\text{Ending point:} \quad \phi_x = T_x, \phi_y = T_y \tag{4.11}$$

In cases where the endpoints cannot be reliably determined (utterances in noisy environments), the constraints are relaxed to compensate for the possible segmentation error, with the penalty of increased computational complexity.

- *monotonicity constraints*, used to maintain the temporal order of the spectral sequence in the speech pattern, thus preserving the linguistic significance of the pattern. According to this criteria, any path which is an acceptable solution must have non-negative slope:

$$\begin{aligned} \phi_x(k+1) &\geq \phi_x(k), \quad \forall k = 1, ..., T_x \\ \phi_y(k+1) &\geq \phi_y(k), \quad \forall k = 1, ..., T_y \end{aligned} \tag{4.12}$$

- *local continuity constraints* expressed as a set of allowable paths to reach a given point. They ensure proper time alignment and reduce the computational complexity by restricting the shape of the path and are based on experimental results (heuristics). An example of local continuity constraints is presented in Figure 4.5.

- *global path constraints*, which are a direct result of applying the local continuity constraints to exclude certain portions of the plane. The allowable regions can

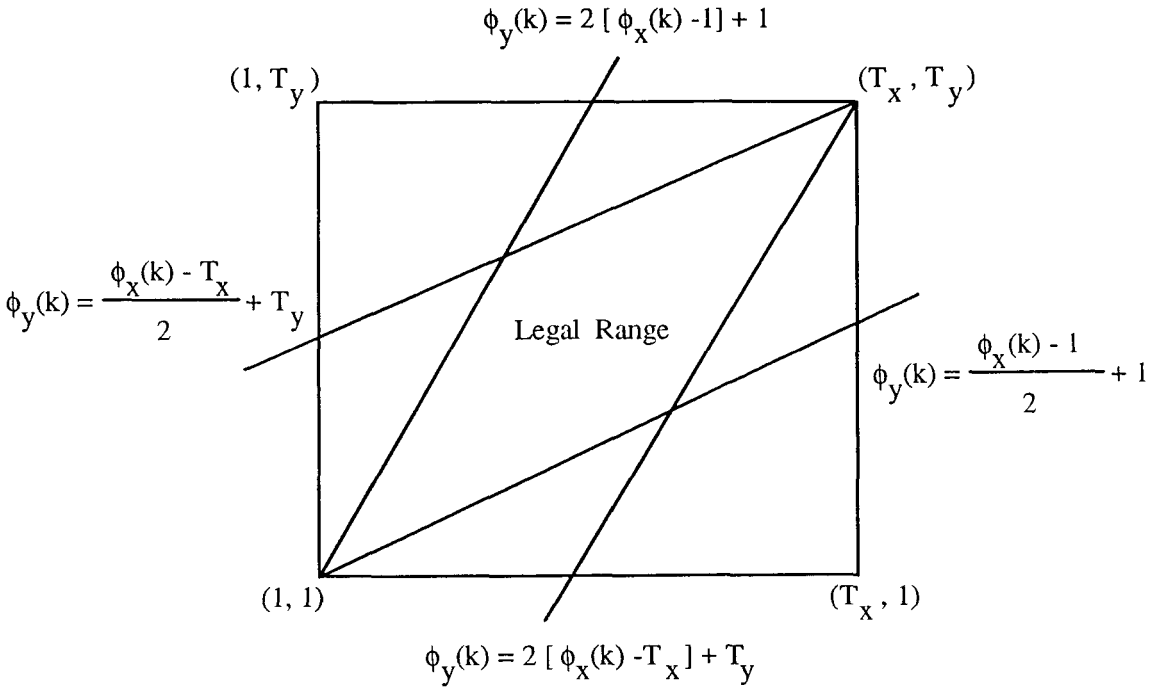

Figure 4.6: Global Continuity Constraints

be defined using the maximum and minimum path expansion, Q_{min} and Q_{max}, as follows:

$$1 + \frac{\phi_x(k) - 1}{Q_{max}} \leq \phi_y \leq 1 + Q_{max}[\phi_x(k) - 1] \tag{4.13}$$

$$T_y + \phi_x(k) - T_x \leq \phi_y \leq T_y + \frac{\phi_x(k) - T_x}{Q_{max}}[\phi_x(k) - 1] \tag{4.14}$$

Figure 4.6 illustrates the effects of the global path constraints when the local continuity constraints presented in Figure 4.5 are used (with $Q_{max} = 1/Q_{min} = 2$). Additional global path constraints exclude any path that involves excessive time stretch or compression [16].

- *slope weighing constraints*, used to define the weighing function $m(k)$, which controls the contribution of each short time distortion to the overall distortion

measure. On a global scale, this function can be used to implement an optimal discriminant analysis for improved recognition accuracy in the case of a confusable vocabulary. On a local scale, it specifies the slope weighing factors for the local path constraints presented in Figure 4.5. In the latter case the weighing factor is higher for less preferable paths, in order to reflect their increased contribution to the distortion measure.

4.1.3 DTW-Based Pattern Comparison

The method of pattern comparison with time alignment uses the normalization and optimal alignment path methods presented above to process the input pattern. The reference patterns used for comparison with unknown patterns are created through template training methods presented below, while recognition is based on the minimum time warped distortion criteria to the set of reference patterns.

Template training methods include:

- casual training,
- robust training using unsupervised averaging
- training using the modified k-means algorithm.

These methods are similar to the vector quantization training procedures presented in Chapter 3. The distortion measure used is given by equation 4.19 and the centroid computation procedure is a warped version of the clustered average.

To describe in more detail the robust training training procedure, which requires a large number of training vectors (as described in Section 3.2.1), only two training patterns $\mathbf{X}_1$ and $\mathbf{X}_2$ of lengths T_1 and T_2, respectively, are considered:

$$\mathbf{X}_1 = (\underline{x}_{11}, \underline{x}_{12}, \underline{x}_{13}, \cdots, \underline{x}_{1T_1}) \tag{4.15}$$

and

$$\mathbf{X}_2 = (\underline{x}_{21}, \underline{x}_{22}, \underline{x}_{23}, \cdots, \underline{x}_{2T_2}) \tag{4.16}$$

are used to generate a reference pattern $\mathbf{Y}$ of normalized size T,

$$\mathbf{Y} = (\underline{y}_1, \underline{y}_2, \underline{y}_3, \cdots, \underline{y}_T), \tag{4.17}$$

where the vectors $\underline{x}$ and $\underline{y}$ are short time spectra.

The training patterns are compared via a DTW procedure, resulting in a distortion score $d(\mathbf{X}_1, \mathbf{X}_2)$, given by:

$$d(\mathbf{X}_1, \mathbf{X}_2) = d_\phi(\mathbf{X}_1, \mathbf{X}_2) = \min_{\phi'} d_{\phi'}(\mathbf{X}_1, \mathbf{X}_2) \tag{4.18}$$

where

$$d_{\phi'}(\mathbf{X}, \mathbf{Y}) = \sum_{k=1}^{T} d(\phi'_1(k), \phi'_2(k)) \frac{m(k)}{M_{\phi'}} \tag{4.19}$$

is based on a spectral distortion d and a set of warping functions ϕ'_1, ϕ'_2 which map the indices of $\mathbf{X}_1$ and $\mathbf{X}_2$, respectively, into the normalized index range $[1, T]$; $m(k)$ is the path weighing coefficient and M_ϕ is the path normalizing factor.

The elements of the reference pattern $\mathbf{Y}$ are then computed based on the optimal path $\phi = (\phi_1, \phi_2)$:

$$y_k = \frac{1}{2}(x_{1\phi_1(k)} + x_{2\phi_2(k)}), \quad \forall k = 1, ..., T \tag{4.20}$$

The training methods which use clustering with unsupervised averaging or the modified k-means algorithm are completely defined by the type of distortion measure used during clustering and by the cluster center selection criteria. For a cluster Ω of L training patterns $\omega = \mathbf{X}_1, ...\mathbf{X}_L$ a dissimilarity or distance matrix $D = (d_{ij})$, $\forall i = 1, ..., L, \forall j = 1, ..., L$ is defined based on the dissimilarity measure given in 4.19 where d_{ij} is calculated as:

$$d_{ij} = \frac{1}{2}[d(\mathbf{X}_i, \mathbf{X}_j) + d(\mathbf{X}_j, \mathbf{X}_i)]. \tag{4.21}$$

With the reference patterns thus defined by the training procedure selected, the recognition decision is based on the minimum distortion criteria.

In addition to the distortion measure defined above, a criteria for selecting the cluster center must be defined. There are several possible criteria for defining a cluster center:

- as the *minimax center*, defined as the pattern in the cluster whose maximum distance to any other pattern in the cluster is the smallest;
- as the *pseudo-average center*, which is the pattern in the cluster with the largest population of neighboring patterns. The subset of neighboring patterns in the cluster is defined as those patterns whose distance to the pattern analyzed falls within a threshold.

- as a *warped average*. For every time index (frame), the new center represents an average of all the cluster patterns warped to the existent cluster center (which is either the minimax center or the pseudoaverage center).

The cluster centers thus defined may not minimize the average intracluster distance and as a result, the modified k-means algorithm is not guaranteed to converge in the sense of the minimum intracluster distance.

4.2 Pattern Comparison Without Time Alignment

Quantization-based recognition systems which incorporate temporal information mainly in the structure of the quantizers and in the decision process are also known as systems without time alignment, to distinguish them from the DTW based systems, which perform time normalization independently of the quantization or decision blocks.

In the VQ design, the following components have been modified to reflect the temporal information:

- the distortion measure used in quantization, as in VQ with spectral variational features;
- the codebook structure, as in segmental VQ and trellis VQ;
- the clustering procedure, as in the trellis VQ.

These examples of spectral temporal VQ design are used as a reference for the VQ methods described in this thesis.

4.2.1 Quantization of Spectral Dynamic Features

The study of modified distortion measures that incorporate spectral dynamic features is justified by the connection between perceptual differentiation of sounds and the variations in their spectral profile.

Spectral transitions play an important role in speech perception. It was demonstrated [15], by using syllables truncated at the initial or final end, that the portion of the utterance where the spectral variation was locally maximum contained the most important phonetic information in the syllable. This result implies that the dynamic,

variational features of the spectrum contribute significantly to the overall recognition performance.

Dynamic features of speech are often represented by a time differential log spectrum. For example, a first order differential (log) spectrum is defined by:

$$\delta(t) = \frac{\partial \log S(\omega, t)}{\partial t} \tag{4.22}$$

where $S(\omega, t)$ is the spectral representation of the utterance obtained by performing short-time spectral analysis (as discussed in Chapter 2. The corresponding first order differential spectral distortion measure is defined as:

$$d_\delta{}^2 = \int_{-\pi}^{\pi} \left| \frac{\partial \log S'(\omega, t)}{\partial t} - \frac{\partial \log S(\omega, t)}{\partial t} \right|^2 \frac{d\omega}{2\pi} \tag{4.23}$$

.

The differential distortion can be combined with the non-differential spectral distance d to generate the overall distortion D:

$$D^2 = \gamma_1 d^2 + \gamma_2 d_\delta{}^2 \tag{4.24}$$

where γ_1 and γ_2 are weighing coefficients.

It was found experimentally that the distances d and d_δ are sufficiently uncorrelated to justify the use of the differential distortion to improve the discriminability of a recognition system [15].

4.2.2 Vector Quantizers With Memory

When the utterances are long enough to cause significant overlap in phonetic content among different utterance classes, or the sequential (temporal) characteristics of the utterance are the only distinguishing factor in recognition, simple, memoryless vector quantizers have inadequate recognition performance. As a possible solution, VQs with memory, such as matrix VQ and trellis VQ, can be used to capture the temporal characteristics of the utterances.

A *matrix quantizer* is a direct extension of the memoryless VQ which encodes several vectors simultaneously. Matrix quantizers can be designed using the same Lloyd algorithm described in Section 3.3.2. If n spectra are encoded at the same

time, the codebook $C = \{\mathbf{Y}_i\}_{i=1}^{N}$ is then designed to minimize

$$D = \frac{1}{T-n+1} \sum_{t=1}^{T-n+1} d'(\mathbf{X}_t, \hat{\mathbf{X}}_t) \tag{4.25}$$

where

$$\mathbf{X}_t = (\underline{x}_t, \underline{x}_{t+1}, \cdots, \underline{x}_{t+n-1}) \tag{4.26}$$

is a sequence of spectral vectors (and thus a matrix) and

$$\hat{\mathbf{X}}_t = \arg \min_{\mathbf{Y}_i \in C} d'(\mathbf{X}_t, \mathbf{Y}_i). \tag{4.27}$$

The distortion d' is often defined for simplicity by:

$$d'(\mathbf{X}_t, \mathbf{Y}_i) = \frac{1}{n} \sum_{j=1}^{n} d(\underline{x}_{t+j-1}, \underline{y}_{ij}). \tag{4.28}$$

Simultaneous encoding of the sequence of spectral vectors, as defined by equation 4.28, implies that the codewords $\mathbf{Y}_i$ have certain embedded block memory constraints. The only differences from Lloyd's algorithm are that minimum distortion criteria applies to a sequence of spectra and that the centroid computation (a matrix of n vectors) involves finding n separate centroids for each codeword.

Another VQ with memory is the *trellis VQ* in which the interdependence in the sequence of input spectra is described by a transition structure represented by a trellis. A trellis VQ is a finite state quantizer, specified by a finite state space Q , an initial state q_0, and three functions:

- an encoder $\alpha : A \times Q \rightarrow N$ where A denotes the space of spectral observations and N is the index set,
- a transition, function $f : Q \times N \rightarrow Q$, also known as the next state function, and
- a decoder $\beta : Q \times N \rightarrow \hat{A}$ where $\hat{A}$ is the space of reproduction spectral vectors (i.e. codewords).

During encoding, the input $\underline{x}_t \in A$ is assigned to a codeword with index $u_t \in N$ based on the current state q_t:

$$u_t = \alpha(\underline{x}_t, q_t). \tag{4.29}$$

The state advances according to the transition function f:

$$q_{t+1} = f(q_t, u_t). \tag{4.30}$$

The decoder β reconstructs $\underline{x}_t$ by $\hat{\underline{x}}_t$:

$$\hat{\underline{x}}_t = \beta(q_t, u_t). \tag{4.31}$$

The encoder α selects u_t based on the minimum distortion criteria:

$$u_t = \alpha(\underline{x}_t, q_t) = \arg\min_{u \in N} d(\underline{x}_t, \beta(q_t, u)). \tag{4.32}$$

The codebook and the next state function are designed to minimize

$$D = \frac{1}{T} \sum_{t=1}^{T} d(\underline{x}_t, \beta(q_t, \alpha(\underline{x}_t, q_t))), \tag{4.33}$$

which describes the centroid computation procedure in the k-means algorithm.

The next state function can be defined by eliminating the non-essential transitions in a trellis which was generated by a regular memoryless VQ. Non-essential transitions are defined as those transitions that can be replaced by an alternate transition with a minimum degradation in distortion performance. A detailed description of the trellis VQ design and clustering algorithm can be found in [12], [17]. In the case of trellis VQ, temporal information is incorporated in the trellis constraints, while in the case of matrix VQ, temporal information is incorporated in the block constraints.

The lack of explicit characterization of the sequential behavior of the utterance can also be remedied by treating each utterance class as a concatenation of a number of N_s segments, each of which is represented by a VQ codebook. This segment-specific VQ approach is known as *segmental VQ* [2], [8].

For an utterance $\{\underline{x}_t\}_{t=1}^{T}$, the simplest way to decompose it into a concatenation of N_s information subsources is to equally divide the utterance into N_s segments. This linear scheme is illustrated in Figure 4.7. More sophisticated segmentation schemes, based for example on phonetic segmentation, are also possible, and can lead to improved performance [9].

The training set of utterances corresponding to a given dictionary word are all segmented using the same scheme and each of the N_s codebooks are trained using the corresponding training segments. These codebooks have an implicit temporal order

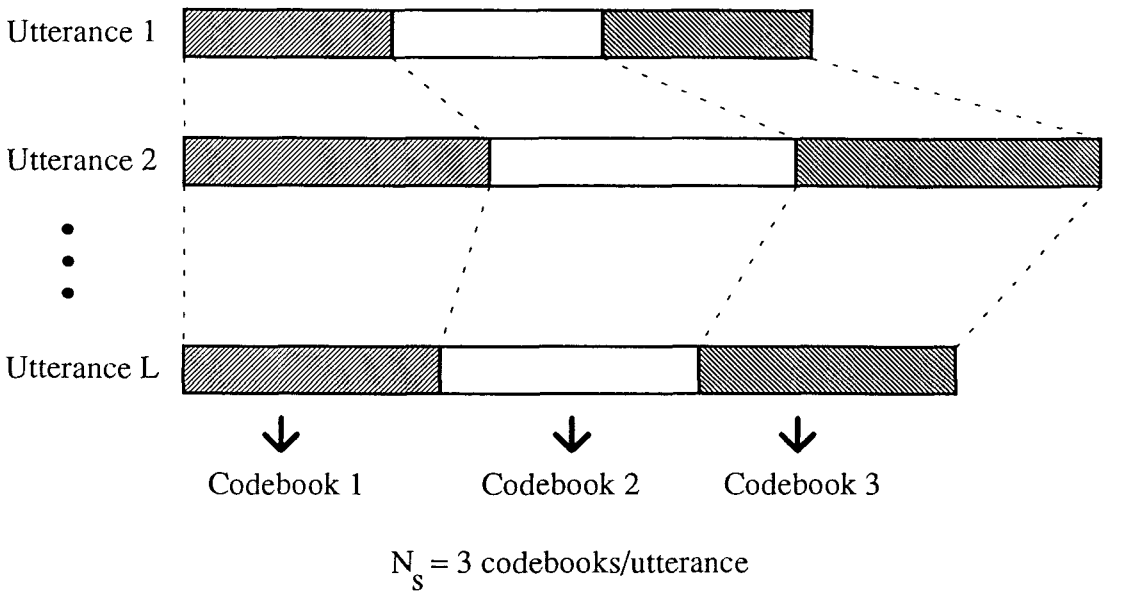

Figure 4.7: Segmental VQ

because they correspond to different portions of the utterances. The average distortion resulting from encoding an unknown utterance with the corresponding successive VQs is the discriminant score for the recognition decision.

Segmental VQ requires the same computational complexity as the previous utterance-based VQ, for the same codebook size. The only complexity increase is in the codebook storage. The preserved sequential relationship of the utterance segments provides an increase in performance in comparison with a VQ without segmentation.

Let V be the number of words in the recognition vocabulary. This implies that V multiple segments codebooks C_k with $k = 1, ..., V$ are used, each comprising of a sequence of segment codebooks C_{kj}, with $j = 1, ..., N_s$. Each segment codebook C_{kj} is designed using n spectral samples (frames) from the normalized input utterance profile, from the $[(j-1)n+1]$-th to the jn-th sample. The ratio n is called the compression factor and is rounded off to the integer nearest to T/N_s .

The average distortion resulting from coding the utterance with the codebook C_k

is

$$D_k = \frac{1}{L} \sum_{j=1}^{N_s} d_{kj} \tag{4.34}$$

where

$$d_{kj} = \sum_{l=(j-1)n+1}^{jn} d(\underline{x}_l, C_{kj}(l)) \tag{4.35}$$

and $C_{kj}(l)$ is the codeword resulting from encoding the sample $\underline{x}_l$ with the segment codebook C_{kj}. The utterance is then classified as the r-th word in the recognition vocabulary, where:

$$D_r = \min_k D_k \tag{4.36}$$

4.2.3 The Temporal Probability Tables Method

An alternative procedure for incorporating temporal information in the structure of a word-based VQ recognizer without time alignment, the *temporal probability tables* method computes the PDF of the codebook vectors' time of occurrence and uses a combined spectral and temporal distortion measure.

In this method, for each codebook vector, the PDF of the time of occurrence (on a normalized time scale), estimated from the same set of training sequences used to derive the codebook vectors, represents the *probability table* associated with the particular class. The collection of probability tables thus created form a temporal codebook. During recognition, for each frame of the unknown input utterance, a spectral distance is computed with respect to the spectral codebooks. The spectral distance is then combined with the temporal probability corresponding to the chosen spectral codebook vector, of minimum spectral distortion, to form the total distortion score.

In the word-based VQ speech recognizer, there is one vector quantizer and one separate codebook for each vocabulary word. Each codebook C consists of a set of n spectral vectors $\underline{y}_j$:

$$C = \{\underline{y}_j\}_{j=1}^{N} \tag{4.37}$$

and can be generated using any of the codebook training procedures described in Chapter 3. The input to the recognizer used in the probability tables method requires a normalization procedure of the input spectral profiles to a fixed length of I frames.

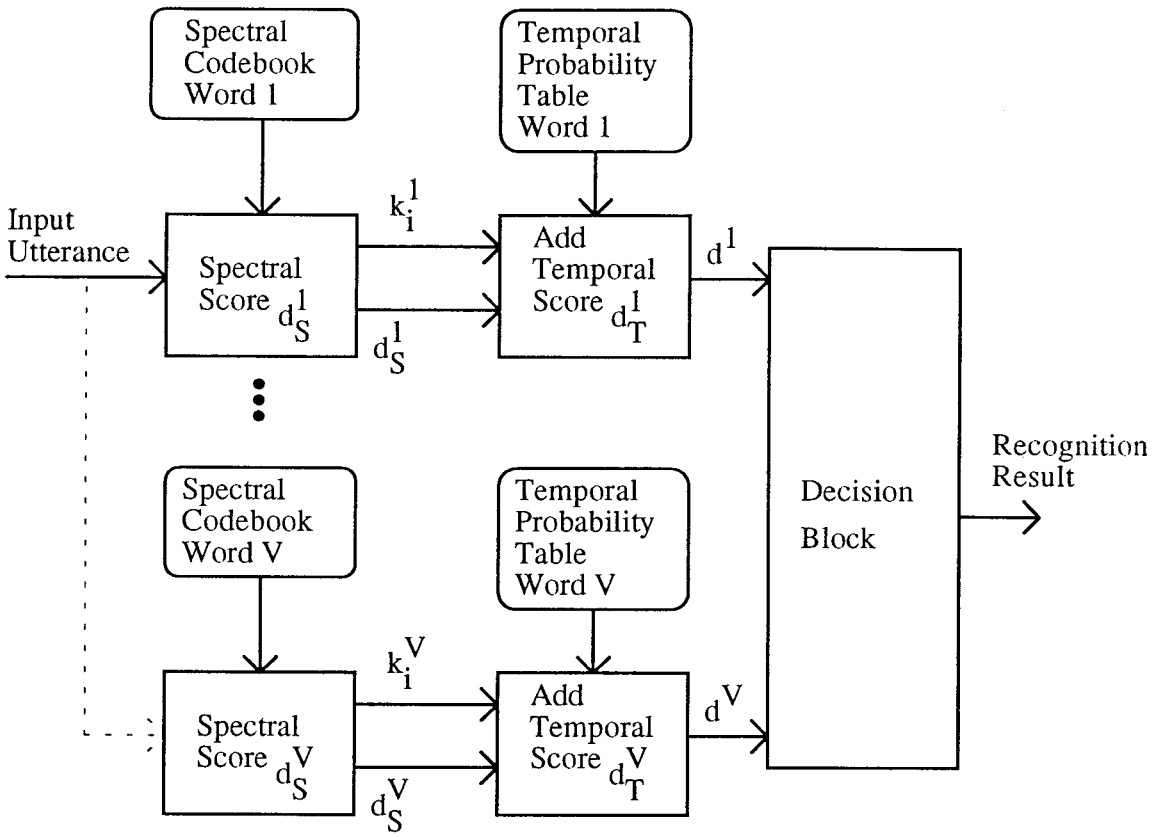

Figure 4.8: Distortion Measure Computation for the Probability Tables Method

Each vector in the vocabulary is also characterized by a temporal probability table P with elements defined as:

$$P(k,t) = \text{the probability that codeword } \underline{y}_k \text{ occurs at the normalized time } t = i/I. \tag{4.38}$$

These probabilities are defined during the training procedure of the spectral codebooks, by recording the number of occurrences of codeword k at time t, $\forall t = 1, ..., I$. For each input training vector, all codebook vectors whose distortion is within a fixed threshold, δ, of the minimum distortion score for the input vector, are considered to have occurred. The value used for $P(k,t)$ is the ratio between the number of times

codebook vector k occurred at time t, and the number of times any codebook vector occurred at time t, over the entire training set for the word. This definition is consistent with the probability definition, because:

$$\sum_{k=1}^{L} P(k,t) = 1, \quad \forall t. \tag{4.39}$$

The temporal probability tables are defined as:

$$\hat{P}(k,t) = \begin{cases} -\tau \ \log P(k,t), & \text{if } P(k,t) > \sigma \\ -\tau \ \log \sigma, & \text{if } P(k,t) \leq \sigma \end{cases} \tag{4.40}$$

The multiplier, τ, was chosen so that, averaged over the entire training set, the average value of $\hat{P}(k,t)$ was the same as the average spectral distortion. The clipping level $\sigma = 10^{-4}$ ensures that the probability tables are consistently defined for null probability scores.

The spectral codebook C and the probability table P for a word in the vocabulary are used to compute a combined spectral-temporal distortion between the input vector $\underline{x}_i$ and the codebook C:

$$d(\underline{x}_i, C, P) = (1-\alpha)\ d_S(\underline{x}_i, C) + \alpha\ d_T(k_i, P) \tag{4.41}$$

where d_S is the spectral distortion and d_T is the temporal probability distortion. The scaling value α determines the mix of spectral and temporal distortions. A value of $\alpha = 0$ represents pure spectral distortion, while a value of $\alpha = 1$ represents pure temporal distortion.

The spectral distance has the form:

$$d_S(\underline{x}_i, C) = \min_k d(\underline{x}_i, \underline{y}_k) \tag{4.42}$$

and the value of the index k for which this minimum is reached represents the value k_i used in equation 4.41. The temporal distance corresponding to k_i has the form:

$$d_T(k_i, P) = \hat{P}(k, i/I), \quad \forall i = 1, ..., I \tag{4.43}$$

The average total distortion is computed using equation 4.41 over all the I input utterance frames. The word corresponding to the codebook with the lowest average distortion is the word selected as the recognition result.

The distortion computation procedure is illustrated in the block diagram of Figure 4.8, for a recognizer with a vocabulary of size V. Each input vector representing the normalized spectral profile an isolated utterance is quantized with respect to each of the spectral codebooks, resulting in a spectral distortion score d_S. The codeword index chosen in codebook V for frame i is denoted by $k_{V,i}$, and is the index of the temporal distortion score, d_T, in the corresponding probability table. The spectral and temporal distortion scores for word V are denoted by ${d_T}^V$ and ${d_S}^V$, respectively, while d^V represents the average combined distortion score used in the recognition decision.

Chapter 5

Spectral-Temporal VQ

Two alternative methods of incorporating the time information in the structure of a VQ-based recognition system are proposed and investigated: *VQ with Time Components*, which uses spectral-temporal codebooks, and *VQ with Overlapped Segmented Codebooks*, which uses multiple spectral codebooks for each input utterance. In both cases, the time information is incorporated in the VQ design directly into the codebook. The proposed approaches are compared respectively with the approach based on probability tables, presented in Section 4.2.3, and with the segmental VQ approach, presented in Section 4.2.2. The results of this comparison are presented in Chapter 7.

In the first approach, VQ with Time Components, each word is represented by a codebook having codevectors with spectral components and temporal components, or *time components*. The codebook is searched using a weighted Euclidean distance applied to the log-spectral components and to the time components. Both components are obtained through a joint spectral-temporal training procedure. The time components approach obtained better recognition results than the probability tables approach, although the former uses significantly less memory than the latter.

In the second approach, VQ with Overlapped Segmented Codebooks, the time information is built implicitly into the codebook by training each codebook with input vectors corresponding to an utterance segment defined to start and end at a given normalized time. Each linearly time-normalized section of the input utterance is represented by a set of codevectors which for a so-called *sub-codebook* and adjacent subcodebooks are overlapped to a variable degree. This technique is a generalization of

the segmental VQ approach, and increases recognition performance without increasing significantly the system's memory requirements.

5.1 Quantization of Spectral Patterns with Time Components

The design of a quantizer for spectral patterns with temporal components consists of defining the spectral-temporal codebook training procedure and the distortion measure used in clustering the spectral-temporal input vectors. The training of a hybrid codebook requires a joint spectral-temporal optimization procedure, while clustering using a hybrid codebook requires the definition of a combined spectral-temporal distance measure, used to compute the accumulated average distortion for the entire input utterance.

Each spectral input vector, combined with the normalized time of occurrence in the spectral profile of the utterance, forms the input vector to the spectral-temporal quantization process. The codewords used during quantization consist also of a spectral part and of a temporal part. The temporal part of a codeword is defined as the most probable time(s) of arrival for the associated spectral codeword. Unlike the input vector considered in this method, which has only one temporal component, the temporal part of the codebook may have more than one time component. As a result, this approach to spectral-temporal quantization is referred to as *VQ with Time Components.*

5.1.1 VQ with Time Components

In VQ with Time Components, temporal information is added to the quantization of short-time spectral features in two ways:

- by explicitly quantizing the normalized time of occurrence (time component) of each spectral vector
- by using a combined distortion measure which reflects the contribution of both spectral and temporal components of the input, to the recognition result.

For a consistent definition of the time components, all input utterances are linearly time normalized to a fixed length L. Assuming that a time-normalized utterance can be represented as a sequence of short-time, Q-dimensional, log spectral vectors

$$\underline{x} = \log \underline{S}(\omega, i), \quad \forall i = 1, ..., L, \tag{5.1}$$

(where i is the normalized time index or time of occurrence), then a spectral-temporal representation of the input is:

$$\underline{x}_i = (\underline{x}, i). \tag{5.2}$$

An example of spectral-temporal representation of an utterance, time normalized to length $L = 40$, is presented in Figure 5.1. A spectral-temporal vector $\underline{x}_i$ consists of 16 spectral components, one for each filter band, plus the time index i, represented on the abscissa by an "x". For each time index, the vertical 17-component collection of points is a graphical representation of the spectral-temporal feature vector used in recognition.

Considering for the beginning the case of one time component per codeword, in the size N codebook, the codevectors are of the form:

$$\underline{y}_{t_k} = (\underline{y}_k, t_k), \quad \forall k = 1, ..., N, \tag{5.3}$$

where t_k is the time component (real number in the range $[1, L]$) and k is the codeword index.

The training of the codebook is done in two steps:

1. the spectral components $\underline{y}_k$ are trained independently of the time component using the Euclidean distortion measure:

$$d_S(\underline{x}, \underline{y}) = ||\underline{x} - \underline{y}||^2 \tag{5.4}$$

2. the time component is trained clustering based on the spectral components only, and represents the average time of arrival for all the spectral-temporal input vectors belonging to the same spectral cluster. Multiple time components describe the distribution of the time of arrival for vectors in the same cluster.

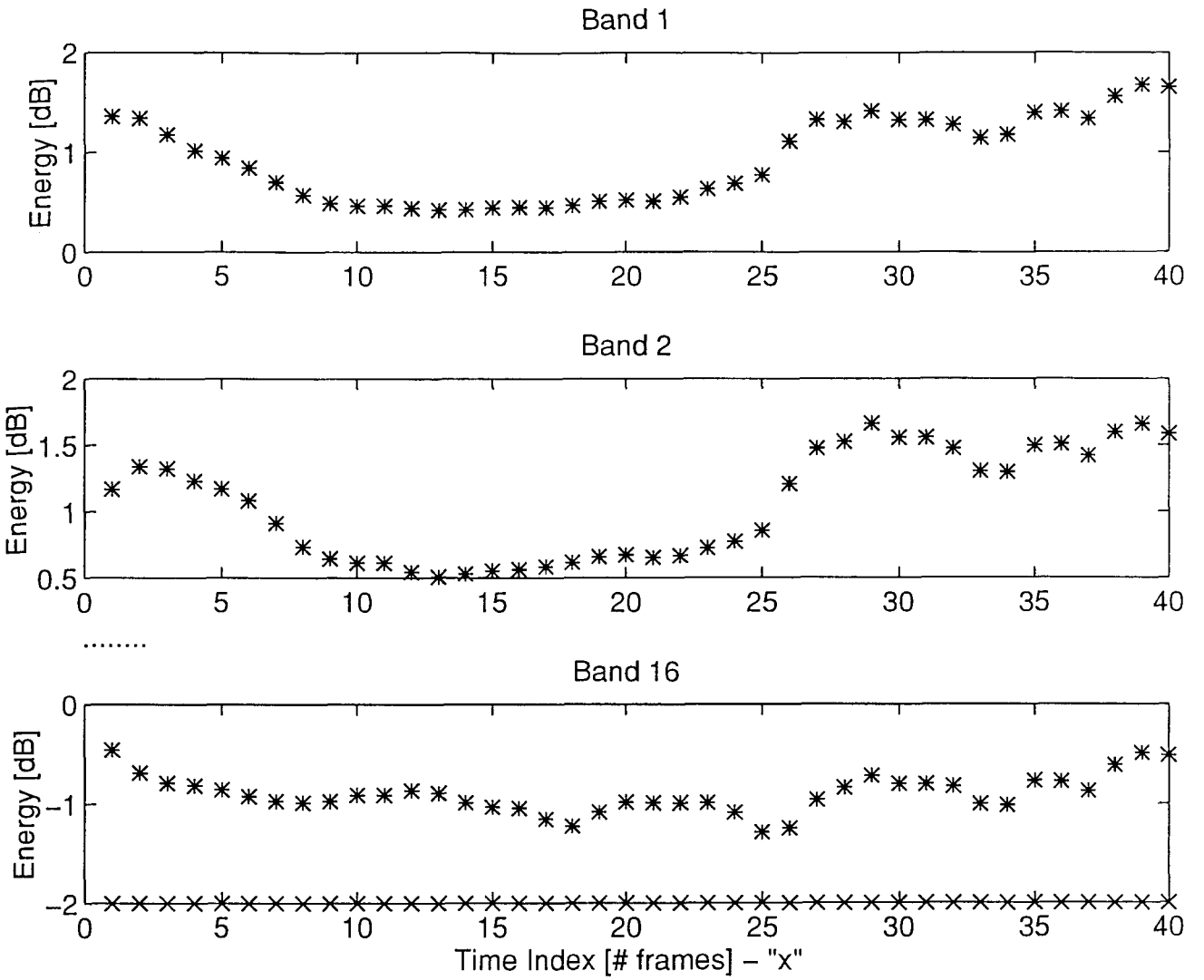

Figure 5.1: Spectral-Temporal Representation of the Time-Normalized Utterance "one"

This definition of the time component, as an approximation of the time of occurrence PDF, is consistent with the time index histograms for each spectral codeword, which have one or more distinctive peaks corresponding to the time component(s) associated with the spectral codeword. Examples of such histograms are presented in Figure 5.2, for codewords 13, 14, 15 and 16 in codebook 1. The histogram function is denoted by $H(k,i)$, and represents the number of selections of codeword k, $\underline{y}_k$, at time index i, $\forall i = 1, ..., L$.

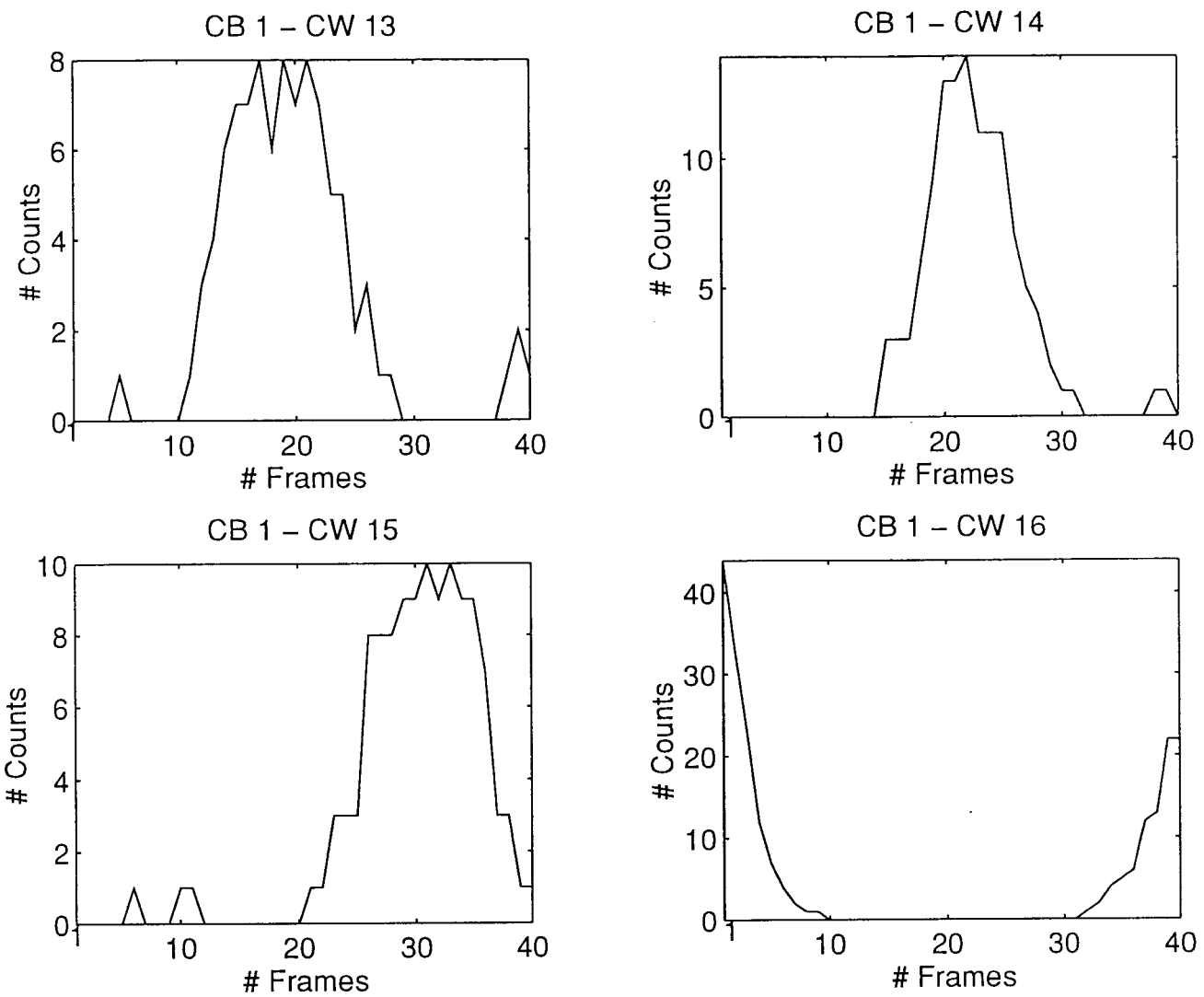

Figure 5.2: Time Index Histograms

When the spectral-temporal codevector has one time component, the value of this component is given by the average value of the histogram H:

$$t_k = \frac{\sum_{i=1}^{L} H(k,i) * i}{\sum_{i=1}^{L} H(k,i)}, \tag{5.5}$$

as can be seen in Figure 5.3 (a), where the approximation of the one peak is represented by the central vertical line in the graph labeled "1 TC". Equation 5.5 is equivalent to the centroid computation procedure given by equation 3.15.

For the case of m time components, histogram averaging is performed by examining intervals around the m highest histogram peaks. The time index interval, $[t_m, t'_m]$,

surrounding each peak, is defined in respect with a count threshold τ, defined as a percentage of the maximum histogram value. The value of the threshold is selected experimentally to maximize the performance of the recognition system. The m-th time component, corresponding to each interval $[t_m, t'_m]$, is given by:

$$t_{mk} = \frac{\sum_{i=t_m}^{t'_m} H(k,i) * i}{\sum_{i=t_m}^{t'_m} H(k,i)}. \tag{5.6}$$

This case is shown in Figure 5.3, where the values of the two time components are represented by the abscissa of lines P1 and P2, respectively, for a threshold $\tau = 30\%$ of the maximum histogram value.

The average spectral and temporal variances, σ_{sk}, is computed for each spectral cluster k, during the temporal training process, as follows:

$${\sigma_{sk}}^2 = \frac{1}{H_k} \sum_{i=1}^{H_k} ||\underline{x}^{(i)} - \underline{y}_k||^2, \tag{5.7}$$

where $||\cdot||$ is defined for a Q-dimensional spectral vector as:

$$||\underline{x}^{(i)} - \underline{y}_k||^2 = \frac{1}{Q} \sum_{j=1}^{Q} (x^{(i)}(j) - y_k(j))^2. \tag{5.8}$$

For the case of one time component, the average temporal variance, σ_{tk}, is defined as:

$${\sigma_{tk}}^2 = \frac{1}{H_k} \sum_{i=1}^{H_k} (i - t_k)^2. \tag{5.9}$$

Above, H_k represents the total number of selections of codeword k:

$$H_k = \sum_{i=1}^{L} H(k,i). \tag{5.10}$$

Similarly, for the case of multiple time components:

$${\sigma_{tmk}}^2 = \frac{1}{H_{mk}} \sum_{i=1}^{H_{mk}} (i - t_{mk})^2. \tag{5.11}$$

with

$$H_{mk} = \sum_{i=t_m}^{t'_m} H(k,i), \tag{5.12}$$

corresponding to the m-th interval above the threshold τ. If the number of desired time components is less than the number of intervals with histogram values above τ,

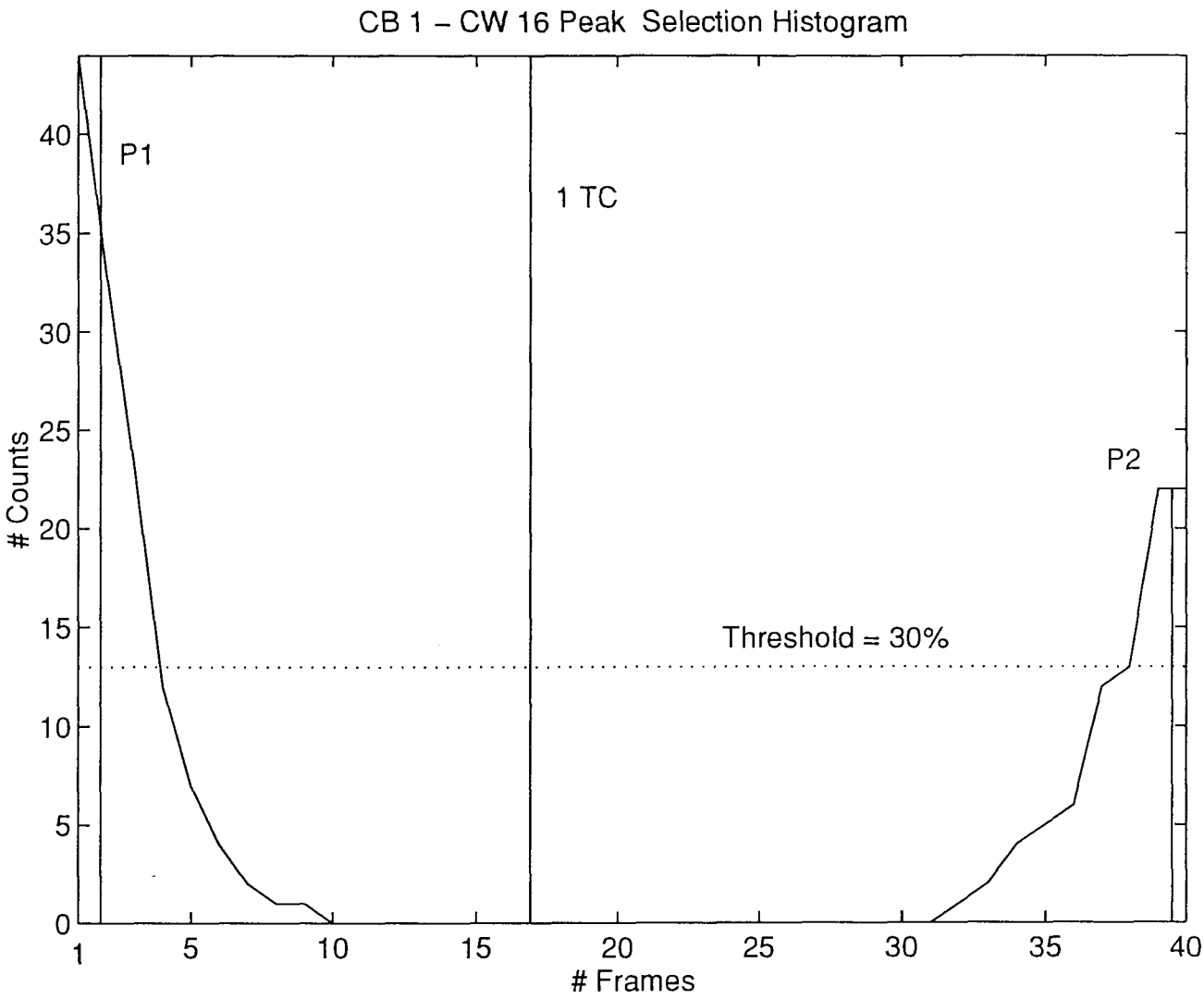

Figure 5.3: Multiple Time Components

the components t_{mk} of the intervals with the highest number of selections H_{mk} are chosen as the temporal part of the codeword. If less intervals than the desired number of time components are found, the remaining time components are initialized to 0.

The distortion measure used during the quantization of an unknown input pattern $\underline{x}_i$ in searching the codebook

$$\underline{y}_{t_k} = (\underline{y}_k, t_k), \quad \forall k = 1, ..., N, \tag{5.13}$$

is given by

$$d(\underline{x}_i, \underline{y}_k) = ||\underline{x}_i - \underline{y}_k||^2 + \frac{{\sigma_{sk}}^2}{{\sigma_{tk}}^2}(i - t_k)^2 \tag{5.14}$$

where σ_{sk}^2 and σ_{tk}^2 are the spectral and temporal variances estimated in the training process for cluster k. For the case of multiple time components, the distortion measure used in searching is based on the time component which is "closest" to the normalized time of occurrence of the input vector.

The block diagram of the training procedure for a size N codebook is presented in Figure 5.4, which summarizes the algorithm described above.

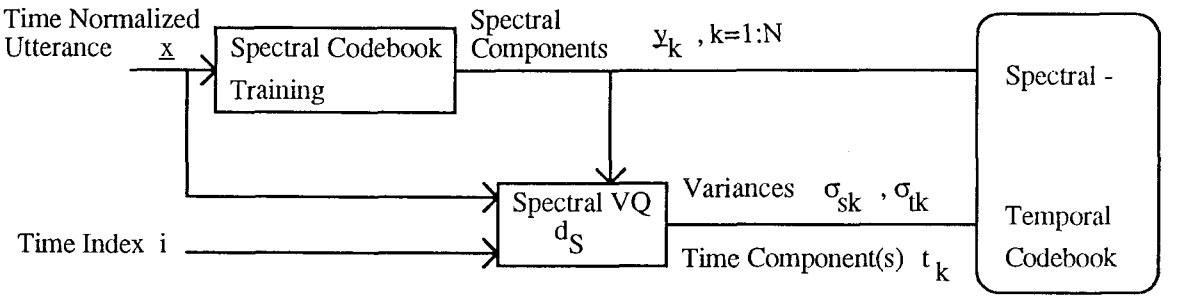

Figure 5.4: Training Procedure for a Codebook with Time Components

5.1.2 Recognition System with Time Components

For a recognition system based on VQ with time components with a size V vocabulary, one spectral-temporal codebook resulting from the training procedure described above is used for each word. The block diagram of the system, presented in Figure 5.5 is similar to the diagram of the recognizer based on probability tables, Figure 4.8 of the previous chapter. The distortion measure computation differs between the two methods, as well as the size of the temporal codebooks used.

The temporal part of the codebook used in the VQ with time components is equivalent to a compact representation of (partial) temporal PDF information integrated within the codebook. Given the relatively small number of temporal parameters used - a time component requires only 2 parameters (one time component and the spectral to temporal variance ratio), while the PDF representation in the case of using probability tables requires L parameters per codevector, a performance degradation is expected with respect to the probability tables approach (L represents the length

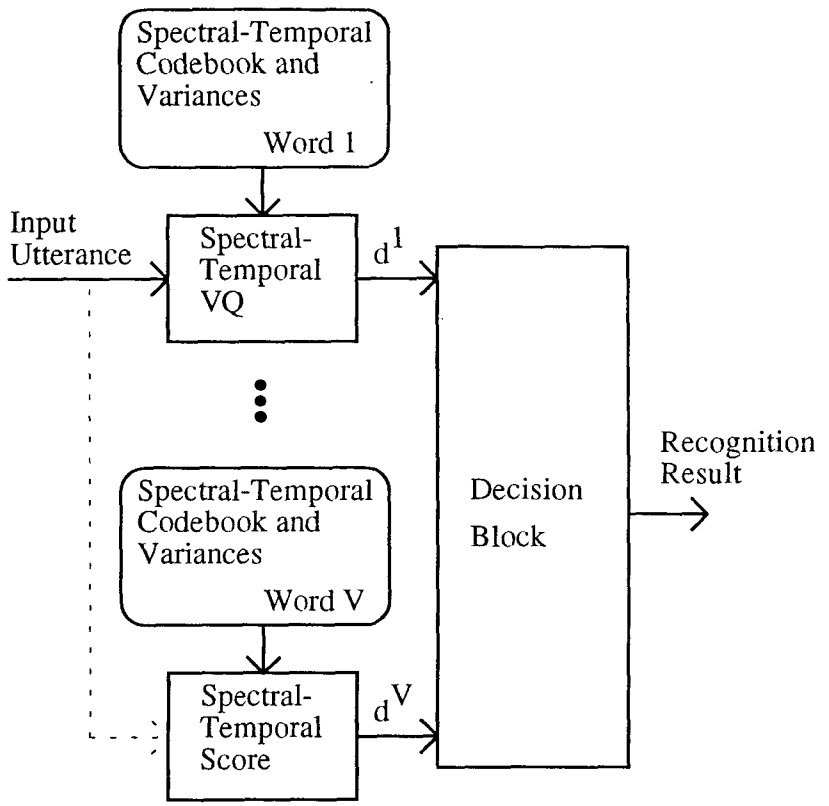

Figure 5.5: Recognition System with Time Components

of the linearly time normalized utterance). The experimental results show instead a performance improvement, which can be explained by the different nature of the distortion measure used in the two cases.

5.2 Overlapped Segmented Codebooks

The representation of the utterance as a sequence of segments is used during the segmentation and labeling phase of recognition systems based on acoustic-phonetic methods, as well as in VQ-based systems, such as the segmental VQ presented in Chapter 4. The results of the former methods show the influence of precise phoneme delimitation on the overall recognition performance, while the results of the latter

suggest that temporal information can be added to spectral VQ by using separate codebooks for each utterance segment.

However, the segmental approach did not address the fluctuations that can be introduced by the normalization procedure required. In this approach, linear time normalization results in an imperfect temporal match, and as a consequence, a given spectral shape may appear at a range of normalized times in different repetitions of the same utterance. The fixed length segments corresponding to phonemes may have similar spectral shapes displaced in reference to one another and thus may be overlapping.

The overlapped codebooks method was developed to account for the effects of normalization and fixed length segmentation and the results show that the method improves indeed the recognition performance at a similar or slightly larger memory requirement than recognition systems based on segmental VQ.

5.2.1 Training Overlapped Segmented Codebooks

The training procedure for overlapped codebooks is based on the k-means algorithm for each of the segments defined. The method uses codebooks of different sizes to represent the overlapping and non-overlapping segments, with different training domains in the normalized input utterance. The overlapped codebooks method is a generalization of the segmental method and includes it as the special case when the overlap between the codebooks is zero.

The definition of the mapping between the utterance space and the codebook space used for training in segmental VQ is presented in Figure 4.7, where the training utterance is linearly time normalized and then segmented in a fixed number of segments N_s, each represented in codebook space by a sub-codebook with N vectors. The generalization to the overlapping codebook method is presented in Figure 5.6 below, where p represents the size of the sub-codebook segment overlapping over an adjacent sub-codebook and $N_s = 4$:

- (a) only one spectral codebook overlapped with $p = N$. This case represents the original spectral VQ presented in Chapter 3, Figure 3.2, in which case no time information is incorporated in the VQ structure. All the vectors in the input utterance are used to train the one resulting codebook;

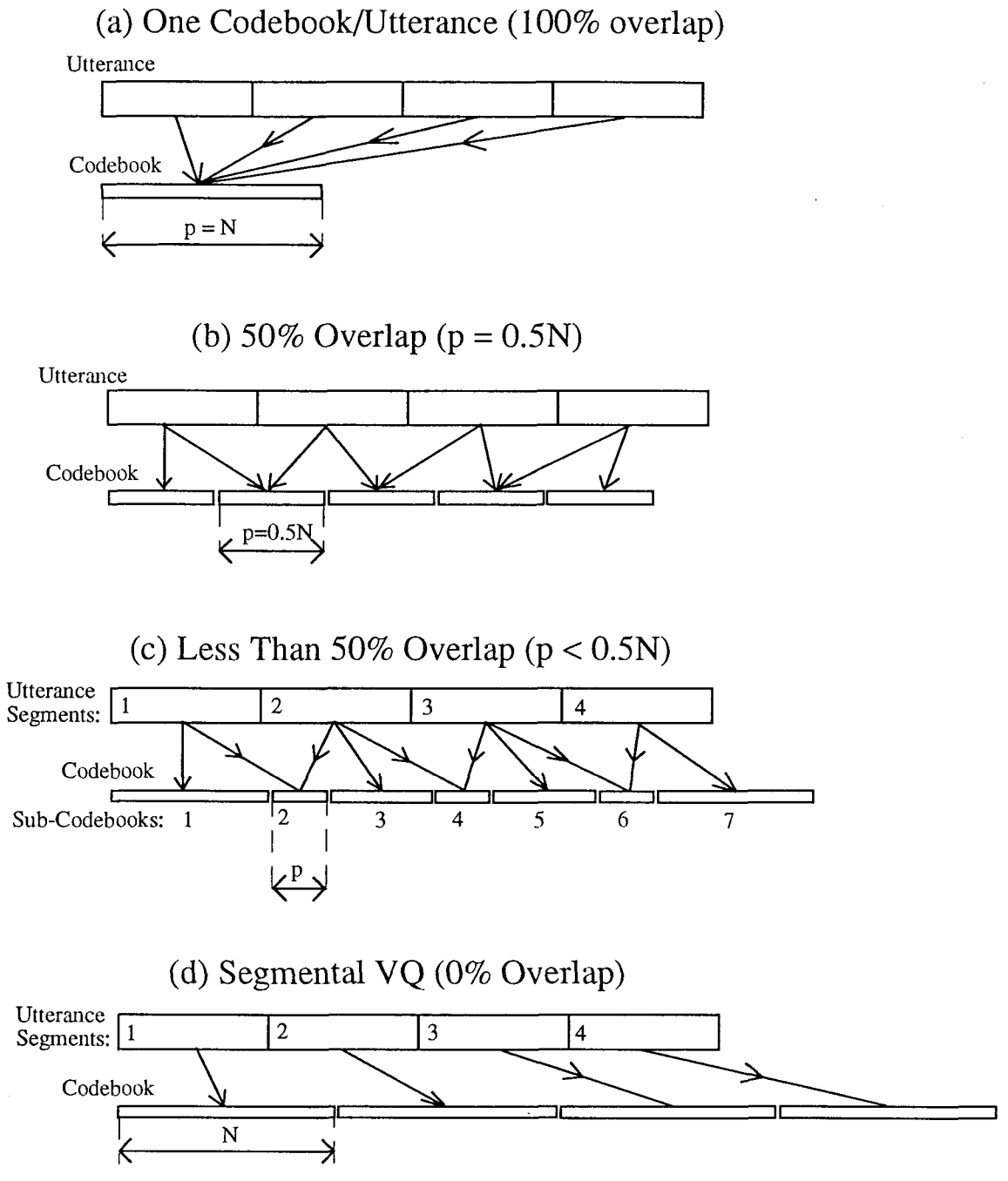

Figure 5.6: Overlapped Codebook Training

- (b) segmental sub-codebooks overlapped with $p = 0.5N$. The first segment of the utterance trains the first sub-codebook and together with the second segment trains the second sub-codebook. The sub-codebooks situated at the extremities of the utterance are trained by one utterance segment only, while all the other are each trained by two adjacent utterance segments;

- (c) segmental sub-codebooks overlapped with $p < 0.5N$ on each side. Each input utterance segment contributes to overlap sub-codebooks, situated at the boundaries with segment-specific codebook.

- (d) segmental VQ, overlap factor $p = 0$. Each sub-codebook corresponds exclusively to one input utterance segment.

The length of the sub-codebooks in Figure 5.6 is proportional with the number of codevectors in each sub-codebook.

Consistent improvements in recognition over segmental VQ, (d), and over the baseline system (a) are obtained for all cases when the segmental codebooks do not overlap more than 50% (cases (b) and (c)). The overall size N_{os} of the codebook varies with the overlap factor p:

$$N_{os} = N_s \cdot N - p \cdot (N_s - 1) \tag{5.15}$$

where N_s is the total number of equal size partitions in the input utterance space and N is the size of the partition in codebook space.

The codebook training procedure is based on the fact that a codevector $\underline{y}_k$ is accessed during the search by input vectors with the normalized time indices in the range $i_{k,min} \leq i \leq i_{k,max}$, and hence should be trained only by these input vectors. The interval limits for training can be determined easily based on the interval limits used for search. For example, in Figure 5.6 (c), sub-codebooks 1, 3, 5, 7 are trained respectively by segments 1 to 4 of the input utterance, while codebooks 2, 4, 6 are trained each by two adjacent input segments: (1, 2), (2, 3) and (3, 4), respectively.

5.2.2 VQ with Overlapped Segmented Codebooks

The temporal information is built implicitly into codebooks by defining a search pace for each input vector $\underline{x}_i$ consisting of codevectors $\underline{y}_k$ with indices in the interval

$k_{i,min} \leq k \leq k_{i,max}$. These codevectors form a sub-codebook and the sub-codebooks for different neighboring indices are overlapped.

During the clustering procedure, each input spectral vector is quantized using the image in the codebook space of the normalized index attached to the spectral vector. To determine the index range $[k_{i,min}, k_{i,max}]$, the mirror image of Figure 5.6 (b) is created in Figure 5.7, where the projection direction indicates the sub-codebooks used in quantizing spectral vectors with time indices in a given normalized segment. For example, a spectral index with normalized time index in the interval corresponding to utterance segment 2 will be quantized using sub-codebooks 2, 3 and 4. The minimum distortion codevector in the sub-codebooks selected contributes to the distortion score of the entire utterance. For simplicity, the boundaries in the codebook space corresponding to the other utterance segments are not represented. Instead, the sub-codebook indices used during quantization are indicated in parentheses above each input utterance segment.

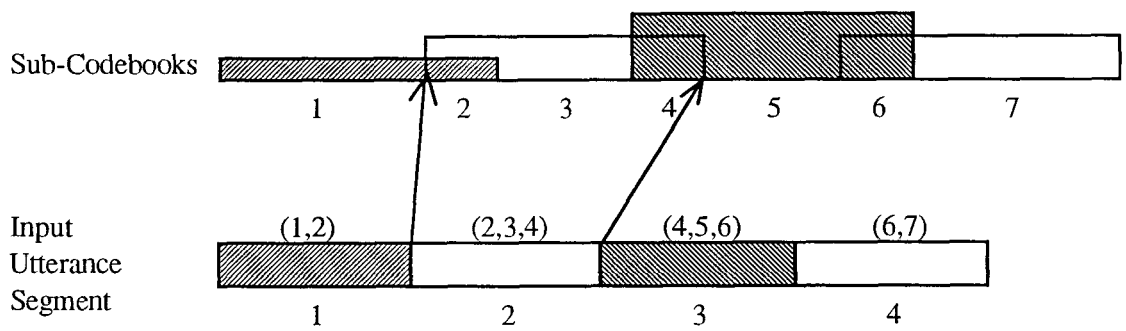

Figure 5.7: VQ with Overlapped Codebooks

A recognition system based on a VQ with overlapped segmented codebooks has the same structure as presented in Section 4.2.2, Figure 4.7. The computational requirements are increased only by the computation of the sub-codebook index range, while the storage requirements are defined by the overall codebook size N_{os}.

Chapter 6

Recognition System Overview

The joint spectral-temporal VQ methods presented in the previous chapter were designed to improve the performance of a baseline recognizer based only on spectral VQ. The objective was to design a recognizer that can be successfully used in a hardware implementation for commercial applications, such as a voice controlled command systems for automobiles, or remote control for consumer electronics products (TV, VCR). Requirements imposed by the specific hardware implementation intended for the recognizer and cost-performance criteria are the premises for the solution presented below, and are presented in Section 6.1.

Figure 6.1 presents on overview of the recognition system implemented and used to evaluate the performance of the spectral-temporal VQ methods described in the previous Chapter. The *Pre-Processing* block, Figure 6.1 (a), performs signal level control on the analog speech signal collected from a microphone, to cover the available dynamic range of the sampling device which records the speech in digital form on permanent storage media. An isolated utterance recording is segmented by the *Endpoint Detection* block, Figure 6.1 (b), which extracts the relevant speech segment from the background noise, by examining the wide band energy profile estimated, as described in Section 6.2.

The spectral profile of the speech segment, which is a collection of energy estimates in 16 adjacent frequency bands, is generated by the *Feature Extraction* block, Figure 6.1 (c), based on a filter-bank analyzer for feature measurement, and described in Section 6.3. A more detailed structure of *Energy Estimation* block is shown in Figure 6.2, which indicates that rectification of each band pass filtered signal, followed by

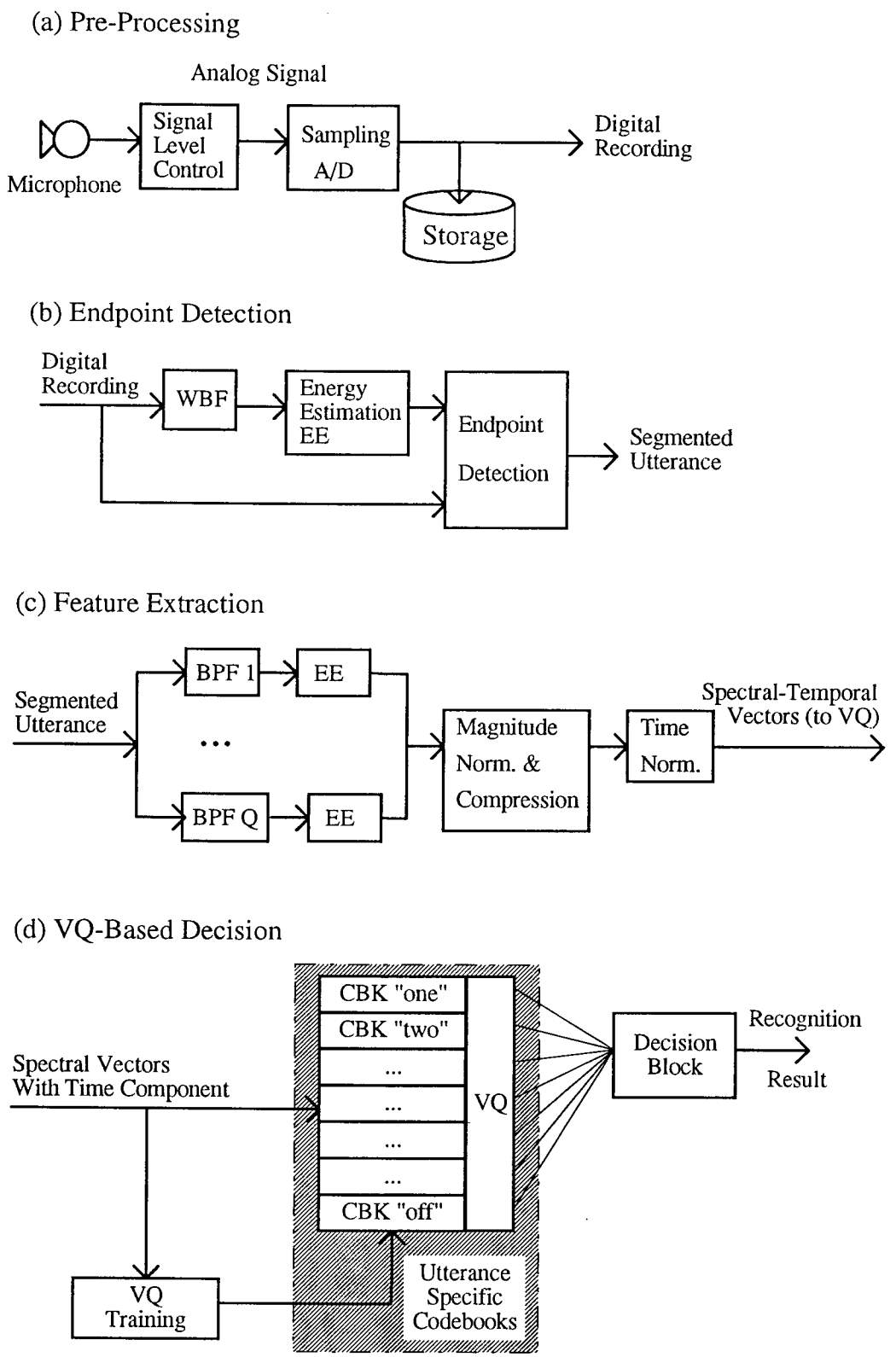

Figure 6.1: Speech Recognition System Overview

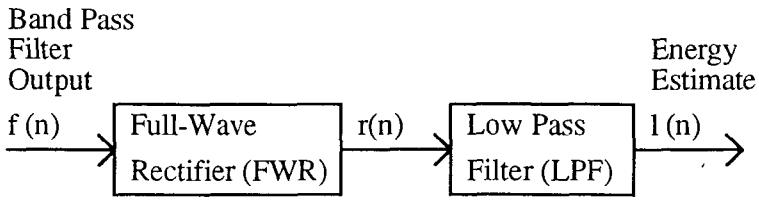

Figure 6.2: Spectral Energy Estimation

low pass filtering, is performed in order to remove the high frequency components introduced by the non-linearity. The magnitude of each energy profile is log-compressed and normalized. The spectral profiles are then time normalized, using re-sampling, to a fixed duration. During time normalization, temporal information can be added to the feature vectors.

The resulting profile of features are used by the *VQ-Based Decision* block either in training, to generate the reference patterns (utterance specific codebooks), or in testing, to determine the recognition result (Figure 6.1 (d)). The VQ-based methods presented in the previous Chapter are used to implement this block, while the parameters of the other blocks are maintained constant, to provide a consistent testing and evaluation environment.

6.1 Design Requirements

In addition to the functional requirements, presented in Chapter 2 for a generic recognizer that can be used in command-and-control systems, the recognition system presented in this thesis has a number of technological requirements imposed by the structure and functionality of the low-cost analog VLSI hardware implementation intended for the system. The analog nature of the signal processing implementation has the advantage of a fast response time for the recognizer, but restricts the available range, complexity and precision of the operations that can be performed on the signal. The cost effectiveness criteria imposes a minimum area requirement on the VLSI implementation, which translates into filter bank design compromises,

due to the fact that the chip area is directly proportional to the number of filters implemented. The design solutions were achieved as a trade-off between the overall recognition performance and the development and production cost requirements.

Design specifications for the implementation of each of the functional blocks presented in Figure 7.1 can be summarized as follows:

- for the *Pre-Processing* block:
  - input is provided through a low-cost microphone followed by an amplification and automatic gain control (AGC) block from an off-the-shelf chip, ISD-2560,
  - at the output the wide-band filter (WBF), the desired frequency range for the signal is [100 Hz, 8 kHz];
- for the *Sampling* block:
  - 16 kHz sampling frequency,
  - 2 seconds of digitized speech samples maximum storage capacity,
  - 16 bit/sample digital representation;
- due to restrictions on the on-chip nonvolatile memory, the *Endpoint Detection* algorithm has access only to the current speech frame, and cannot use past samples to determine the boundaries of the utterance;
- for the *Feature Extraction* block:
  - the design of the filter bank must take into account that the hardware implementation will consist of a single band pass Switched Capacitor Filter (SCF), which can be tuned at different center frequencies and bandwidths by changing its clock frequency,
  - the logarithmic compression is approximated by a transfer function implemented also in SCF technology, and introduces a DC bias,
  - at each step in the algorithm that requires storage of an intermediate result, the quantization effect (equivalent to a 8-bit linear quantization) of the analog memory used must be simulated in the design of the codebooks,

  - the non-volatile memory used for intermediate storage supports only a limited number of WRITE accesses, which determine the usage time of the chip. As a result, the number of storage steps to the same intermediate storage area must be minimized;

- for the *VQ-Based Decision* block:

  - the size of the codebook space is limited by the size of the analog storage array, and must be lower than 200k cells,
  - for the speaker independent recognizer, the codebooks used by the VQ are produced by the simulation software and stored during the manufacturing process of the chip;
  - no training logic is provided in the VLSI implementation of the recognizer.

To minimize the development time and the implementation cost, the simplest solution was chosen for the VLSI implementation of each given block, although alternative solutions were investigated for comparison and future performance enhancements. Also, the chip does not contain a micro-controller, and as a result the logic must be kept very simple for all the functional blocks described above. The VQ-based decision block and the design of spectral-temporal VQ algorithms presented in Chapter 5 were chosen to satisfy the simplicity criteria imposed by the hardware implementation.

With the above requirements, the hardware block diagram of the recognizer chip can be represented as in Figure 6.3. The amplified (AGC) and wide band filtered (WBF) input speech utterance is stored in the *Nonvolatile Memory* under the control of the *Endpoint Detector Logic*. The utterance boundaries determined during *Segmentation* are also used to determine the down sampling frequency necessary for the control of the *Time Normalization* block, which is implemented as a tunable down sampler.

During *Feature Extraction*, the segmented utterance stored in the non-volatile memory is passed through the SCF 16 times. The spectral-temporal profile resulting at the output of the *Time Normalization* block is stored in the *VQ Memory*. The VQ codebooks, pre-computed and stored in the *Codebook Memory*, are then used to compute the distortion measures with respect to each word. Finally, the minimum

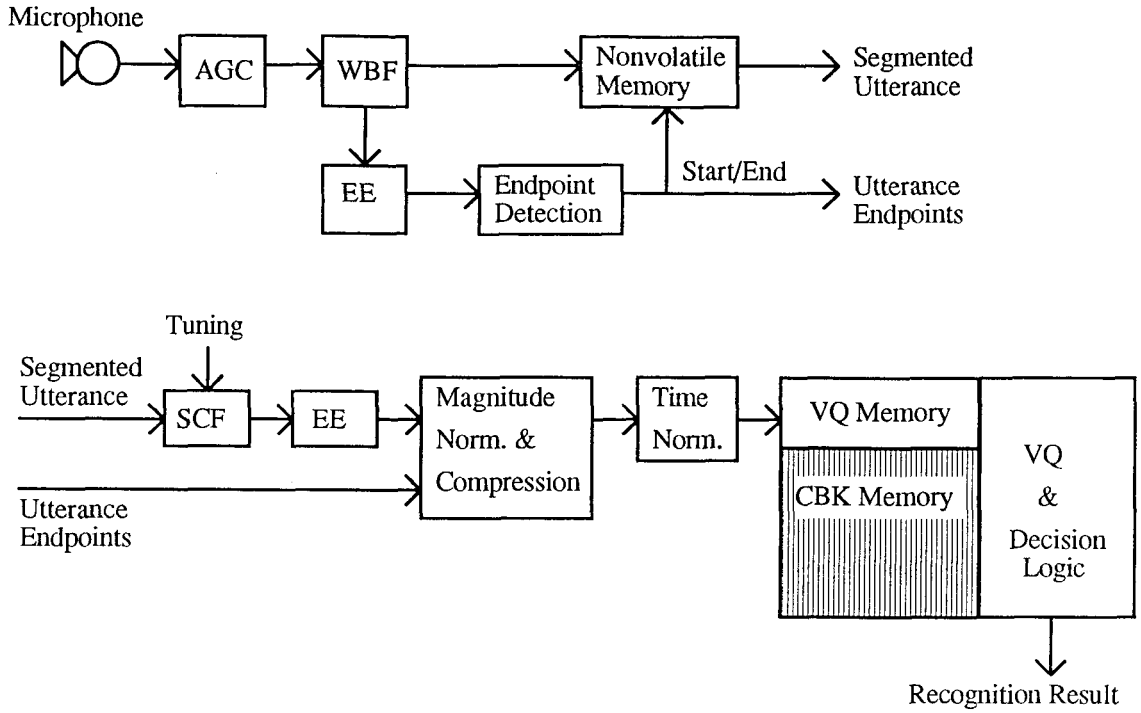

Figure 6.3: Hardware Block Diagram of the Recognizer

distortion score is selected by the *Decision Logic* block, to represent the recognition result.

6.2 Endpoint Detection

A recognition system developed for isolated words must determine the beginning and end of the utterance of interest, with higher energy profile than the background noise energy (silence). The database recordings used for tests consist each of an isolated word, preceded and followed by silence or other background noise. The process of separating the speech segments of an utterance from the background is called *endpoint detection*, or *segmentation*.

Accurate detection of the endpoints of a spoken word is important because it is directly related to the recognition performance of the system [18]. Problems in endpoint detection arise from transient noise (often the beginning or end of an isolated word is accompanied, and thus concealed, by mouth noises such as clicks, pops, lip

smackings and heavy breathing) and nonstationary backgrounds where there may be concurrent conversations and noises due to movements of chairs, door slams, etc. A noise cancellation microphone may be used to eliminate the unwanted effects of a nonstationary background, and could be added as a further improvement to the recognition system; however, the version of the evaluation setup used for the results presented below did not include a noise cancellation system. Segmentation methods can be classified, according to the manner in which the endpoints are specified during the recognition algorithm, as [18]:

- *explicit* segmentation, performed prior to and independently of the recognition and decision stages of the recognizer.
- *implicit* detection, when the endpoints are determined only during recognition and available only after the decision stage is completed. In this case there is no separate processing stage for endpoint detection.
- *hybrid* method, which incorporates ideas from both the explicit and implicit methods.

Although the hybrid technique produces the best results, due to the real-time nature of the endpoint detection algorithm and the hardware requirements of Section 6.1, an explicit segmentation method was chosen for this implementation. The algorithm proposed is a simpler version of the explicit endpoint detection algorithm proposed in [18].

Endpoint segmentation is performed on the energy profile of the signal segment at the output of the WBF. The Wide Band Filter has the frequency characteristic shown in Figure 6.4 and has a 150 Hz to 7200 Hz passband (3 dB ripple) with 30 dB attenuation at 60 Hz and 7800 Hz, respectively. The filter coefficients are provided in Section A.1. The band limited signal is passed through a rectifier, a low pass filter (LPF) and then down sampled to generate an estimate of the signal's energy, as presented in Figure 6.2, discussed in Section 6.3.2.

The segmentation algorithm investigated for this design uses heuristically defined energy thresholds to separate the relevant speech segment from silence. The procedure for defining the thresholds is described in [18], and consists of interactive user

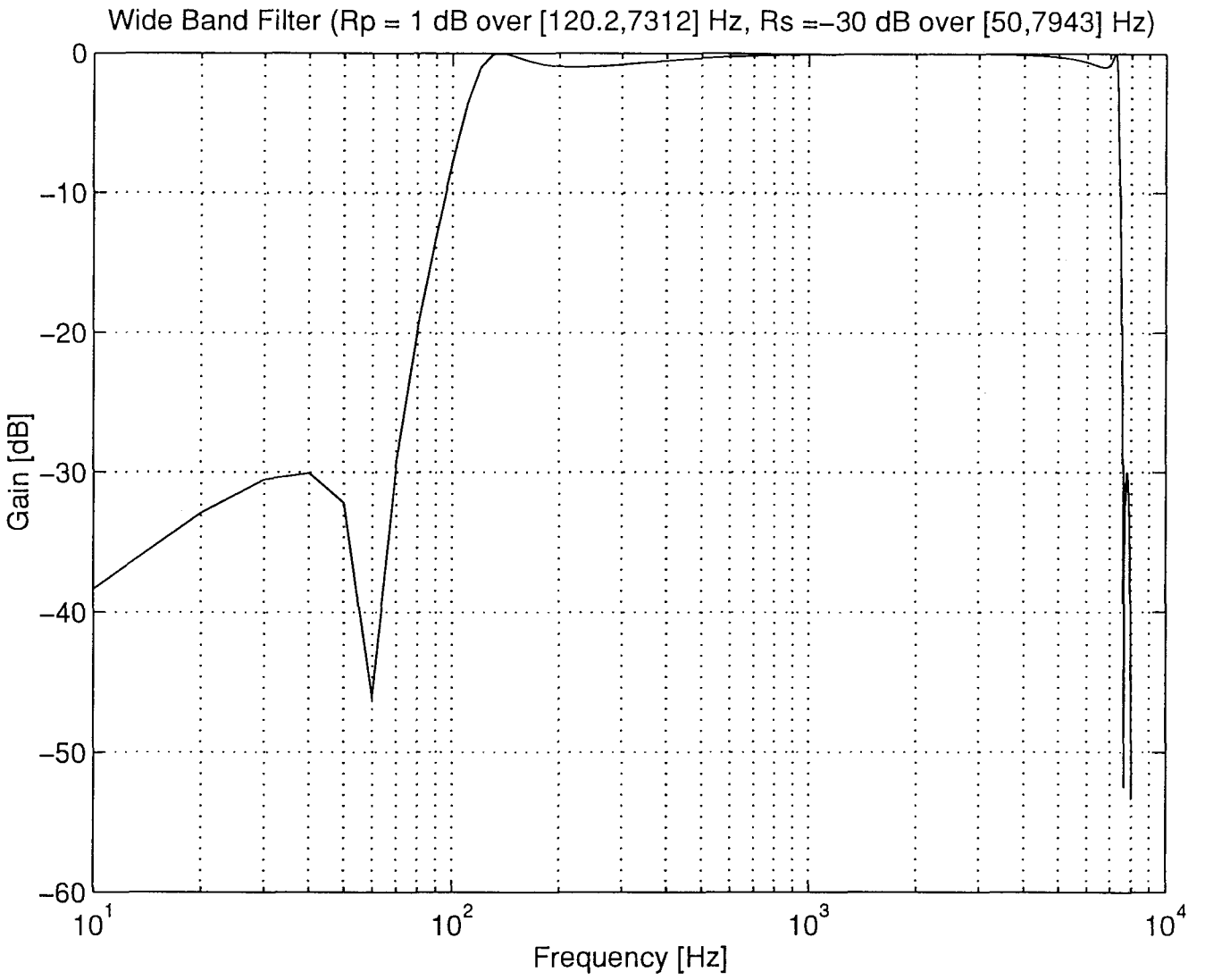

Figure 6.4: Wide Band Filter Frequency Characteristic

segmentation, to measure the distribution of energy thresholds. The endpoint detection algorithm in its general form refers to the signal segment profile presented in Figure 6.5 and consists of the following steps:

- Step 1. Finds the index j_0 of the first frame with energy above a given energy threshold, k_2, referred to as *minimum pulse energy threshold*, and represents the minimum energy which indicates the presence of a speech-like burst of energy in the segment.

- Step 2. Backs up $\tau_0 = 200$ ms, where τ_0 is referred to as the *back-up time*. Back-up is necessary because otherwise the real start of the utterance, at a lower energy level on the rising slope of the profile, may be ignored.

- Step 3. Advancing towards the end of the utterance, finds the index j_1 of the first frame with energy above the *utterance start threshold* k_1, with $k_1 < k_2$. This step results from the observation that frames within τ_0 of the minimum pulse energy threshold and having energy at least equal to k_1 belong also to the utterance and could be ignored if the search stops at Step 1. The index j_1 represents the start of an energy pulse which could be a word.

- Step 4. Finds the index j_2 of the first frame with energy below the *utterance end threshold* k_3.

- Step 5. (optional) Checks if j_2 represents the end of the utterance, by comparing the utterance length, $j_2 - j_1$, with the *minimum pulse duration*, denoted by τ_1. If $j_2 - j_1 > \tau_1$ then the start index is j_1 and the end index is j_2, otherwise restarts the search from index $j_2 + 1$.

- Step 6. (optional) Checks for the presence of consecutive energy pulses in the utterance, by searching for a frame with energy above the utterance start threshold k_2, at a distance equal at most with the *maximum inter-pulse pause* τ_2. If such a frame is found, then Step 4. is performed, otherwise the end of the utterance is given by j_2.

The following versions of the above algorithm were tested using the studio database and the mixed database:

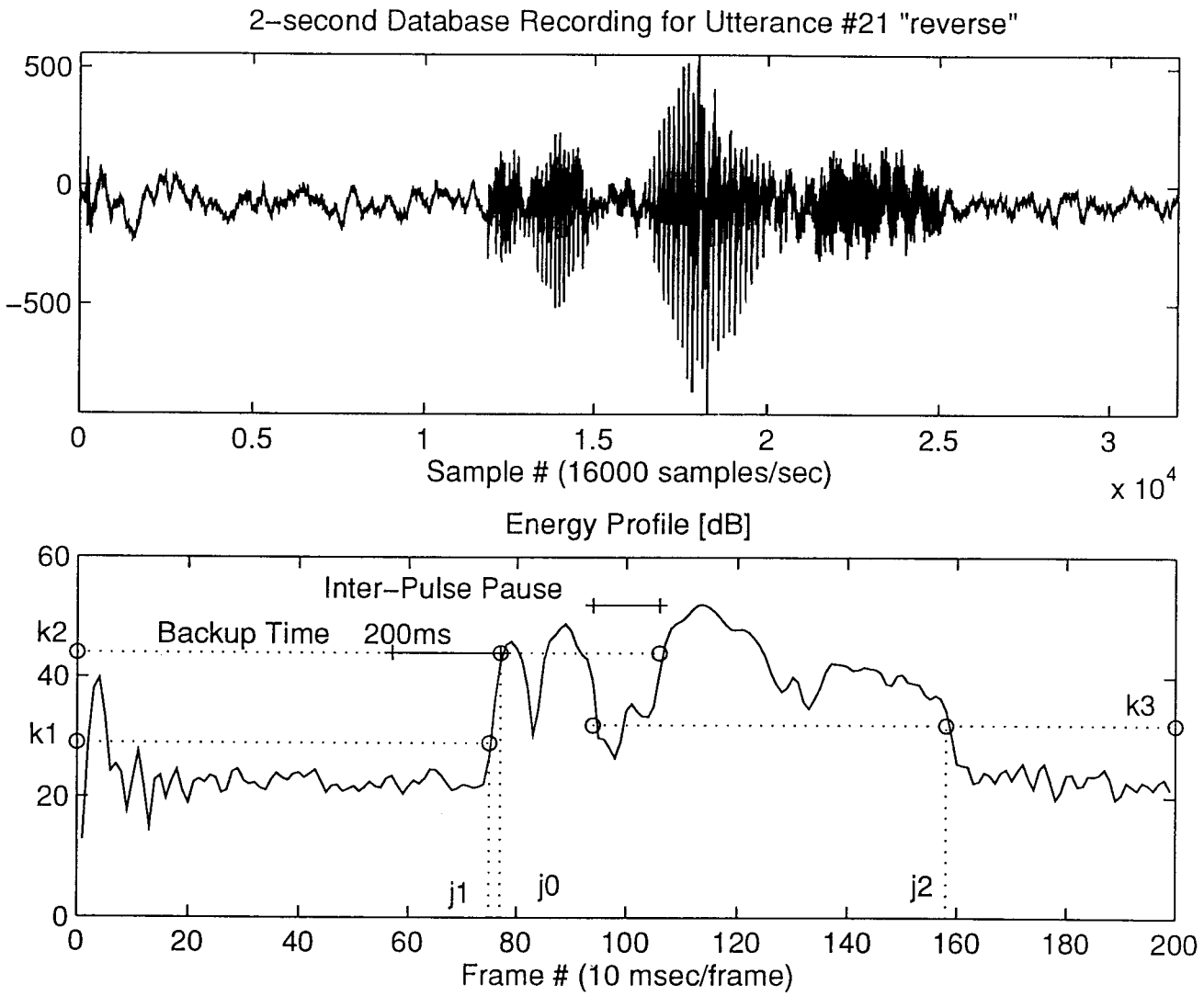

Figure 6.5: Energy Thresholds and Time Intervals Used in the Endpoint Detection Algorithm

Segmentation Version Database	Studio Database	Mixed Database
V1	98.4 %	85.87 %
V2	98.08 %	86.05 %
V3	98.24 %	89.4 %

Table 6.1: Recognition Results for Endpoint Algorithm Comparison

- Version 1 (V1): Uses only one energy threshold for both start and end, and has no back-up time. Does not check the minimum pulse duration or the maximum inter-pulse pause (steps 5 and 6).

$$k_1 = k_2 = k_3 \quad \tau_0 = 0 \quad \tau_1,\ \tau_2 \text{ not used.} \tag{6.1}$$

- Version 2 (V2): Uses only one energy threshold for both start and end, and has fixed back-up time of 200 ms. Does not check the minimum pulse duration or the maximum inter-pulse pause (steps 5 and 6).

$$k_1 = k_2 = k_3 \quad \tau_0 = 200ms \quad \tau_1,\ \tau_2 \text{ not used.} \tag{6.2}$$

- Version 3: Implements the generalized algorithm described above. Uses different energy thresholds for both start and end and has fixed back-up time of 200 ms. Checks the maximum inter-pulse pause (step 6), but does not check the minimum pulse duration (step 5).

$$k_1 < k_2 \neq k_3 \quad \tau_0 = 200ms \quad \tau_1 \text{ not used},\ \tau_2 = 100ms. \tag{6.3}$$

The recognition accuracy results for comparing the three versions of the segmentation algorithm are presented in Table 6.1. The feature extraction algorithm and the VQ training procedure are the same for all tests. Both the training and the test utterances are segmented using the same algorithm version; the codebooks are re-trained for each experiment. Only studio database utterances were used for codebook training. The recognition method used was based on spectral VQ only (no time information was used during quantization).

For the mixed database, the generalized algorithm V3 has a significant performance advantage in comparison with both V1 and V2. The back-up procedure could not

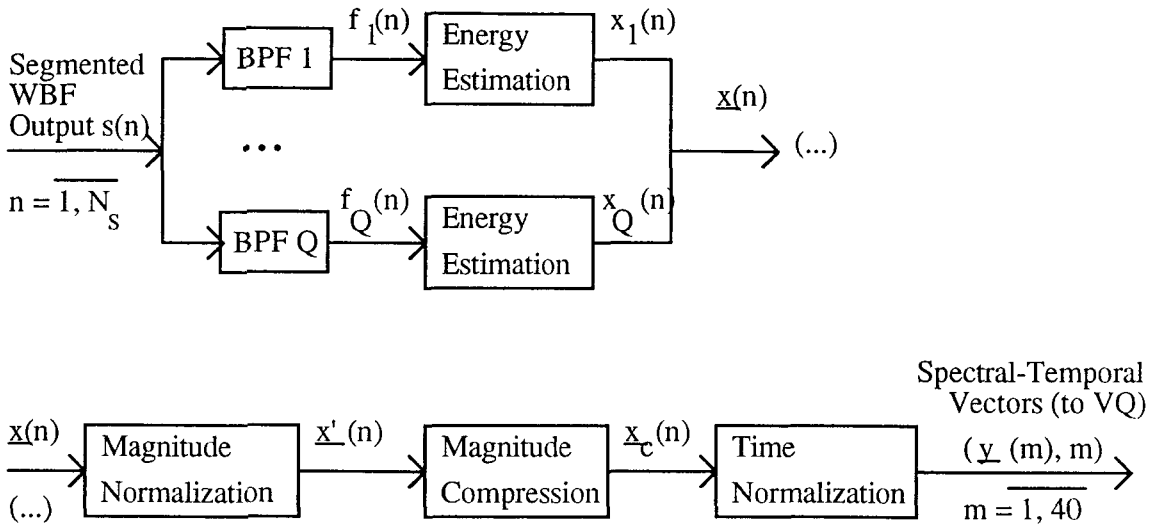

Figure 6.6: Feature Extraction Block Diagram

be implemented in hardware, due to restrictions imposed by the limited number of WRITE accesses to the nonvolatile memory, where the input utterance is stored. As a result, the first version of the segmentation algorithm, V1 with $k_1 = 25dB$, was used in the recognizer's simulation.

6.3 Spectral Feature Extraction

The *Feature Extraction* block for the simulation of the recognizer is based on a bank of bandpass filters, as introduced in Section 2.3, and is presented in detail in Figure 6.6. The block consists of a bank of $Q = 16$ band pass filters *BPF*, followed by the *Energy Estimation* block, which evaluates the filtered signal energy in the corresponding band and the *Time Normalization* block, which generates the fixed size sequence of spectral, or spectral-temporal, vectors used in quantization. A detailed description of the Energy Estimation block is given in Figure 6.2.

In Figure 6.6, the input signal vector $s(n), \forall n = 1, ..., N_s$ represents of the segmented utterance produced by the Endpoint Detection block. For each band $i = 1, ..., Q$ and each frame $n = 1, ..., N_s$, an energy estimate $x_i(n)$ is produced. As a

result, the bank of Q filters output for frame n is a Q - dimensional vector $\underline{x}(n) = (x_1(n), ..., x_Q(n)), \forall n = 1, ..., N_s$. Each spectral vector $\underline{x}(n)$ is then normalized in magnitude to $\underline{x'}(n)$ and then the signal's dynamic range is compressed using a log transformation, to produce the compressed normalized version $\underline{x_c}(n)$, which represents the input to the Time Normalization block. The time normalized version of $\underline{x_c}(n)$ is denoted by $\underline{y}(m)$.

In addition to performing a re-sampling of the energy profile in each band, from variable length N_s to the normalized utterance length M ($M = 40$ used in this simulation), the Time Normalization block also computes the time component, $t(m)$, associated with each Q-dimensional spectral vector $\underline{y}(m) = (y_1(m), ..., y_Q(m))$ as :

$$t(m) = m, \quad \forall m = 1, ..., M. \tag{6.4}$$

The frequency characteristic of the bank-of-filters pre-processor consists of adjacent bands and is illustrated in Figure 6.7.

6.3.1 Filter Design

In designing the bank of filters, the first consideration was the type of filters used. The analog implementation using tunable switched capacitor filters (SCF) was chosen due to its cost efficiency. The digital filters used in the simulation of the recognizers are FIR filters with a frequency response close to the corresponding analog filters.

The number of bandpass filters cannot be smaller than 8 or else the ability of the filter bank to resolve the speech spectrum is greatly impaired [2]. If more than 32 filters are used, the filter bandwidths would be too narrow for some talkers and there would be very likely that certain bands would have extremely low speech energy, and could not be adequately quantized during storage in the analog memory. In this implementation, the number of filters was chosen to be 16, with non-uniform spacing, to reflect the human perception of speech [2].

It has been theoretically shown that filter banks with frequency plans based on perceptual scaling, such as the Bark scale, improve significantly the performance of a recognition system. The improvement is due to the fact that each band on such a scale has equal contribution to the intelligibility (perception) of speech [2]. The cut-off frequencies for a bank of 16 filters linearly distributed on the Bark scale for

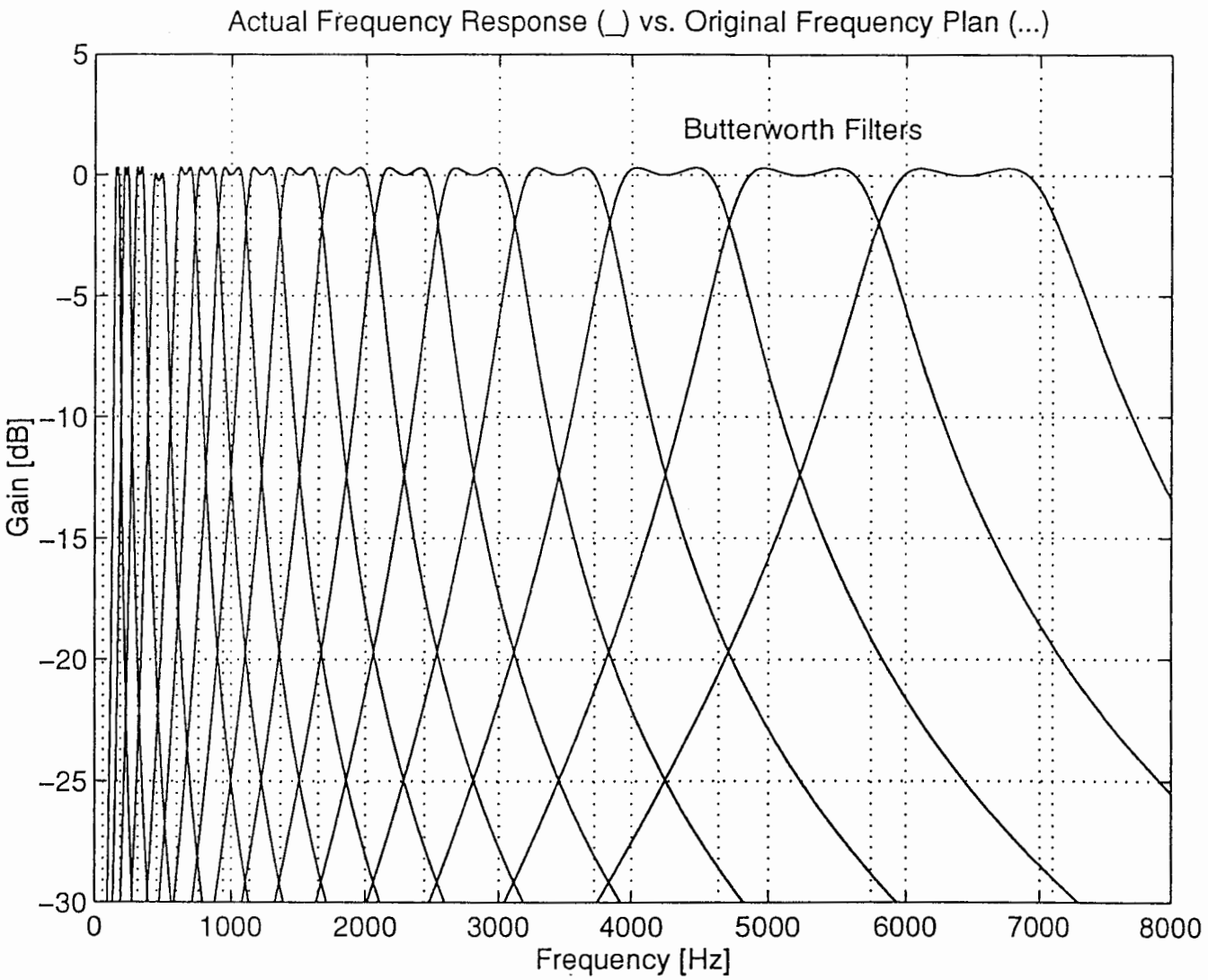

Figure 6.7: Filter Bank Frequency Characteristics

Band	Center Frequency [Hz]		Bandwidth [Hz]	
	Bark Scale (a)	SCF Scale (b)	Bark Scale (a)	SCF Scale (b)
1	107	161	127	34
2	244.8	229	131	53
3	381.4	326	137	75
4	523	463	144	106
5	675.28	659	160	152
6	845.22	811	180	187
7	1034.27	997	198	229
8	1248.62	1227	232	282
9	1504.86	1509	283	347
10	1818.24	1856	347	427
11	2207.26	2283	436	525
12	2703.33	2808	564	646
13	3342	3453	723	794
14	4156.29	4247	917	977
15	5163.02	5224	1105	1202
16	6386.67	6426	1355	1478

Table 6.2: Bark Scale vs. SCF Scale - Frequency Plans Comparison

a bank of 16 filters is presented in Table 6.2, columns (a). An approximation of this frequency plan which can be implemented using a tunable filter (SCF) is given in Table 6.2, columns (b). The penalty in recognition performance for using the SCF frequency plan, as opposed to the original plan based on the Bark scale, is less than 1%. The filter coefficients used in the recognition system are listed in Section A.2.

6.3.2 Energy Estimation

Energy estimation is required in the recognition system for:

- the WBF output. The resulting energy profile is further used by the Endpoint Detection block to isolate the relevant utterance from silence;
- each BPF output (in the bank of Q filters). The Q dimensional energy profile is used as input to the VQ-based decision block.

In both cases, the signal at the filter output, $f(n)$, is rectified using a Full-Wave Rectifier such that:

$$r(n) = \begin{cases} f(n) & \text{for } f(n) \geq 0 \\ -f(n) & \text{for } f(n) < 0. \end{cases} \tag{6.5}$$

The FWR output is then low pass filtered (LPF), producing the energy estimate $l(n)$, as shown in Figure 6.2. The low pass filter was designed to have the characteristic presented in Figure 6.8, in order to remove the high frequency images introduced by the non-linearity in the energy spectrum, as discussed in Section 2.3. The filter coefficients are given in Section A.3.

6.3.3 Time Normalization

Two time normalization methods were implemented and tested during the simulation development:

- *Resampling*, from the variable input length N_s to the fixed length M, which computes the output frame index m as the "closest" input frame index n:

$$n = \text{round}(m \cdot \frac{N_s - 1}{M - 1}) \quad \forall m = 1, ..., M, \tag{6.6}$$

as illustrated in Figure 6.9, (a) and (b) for $N_s > M$ and $N_s < M$, respectively.

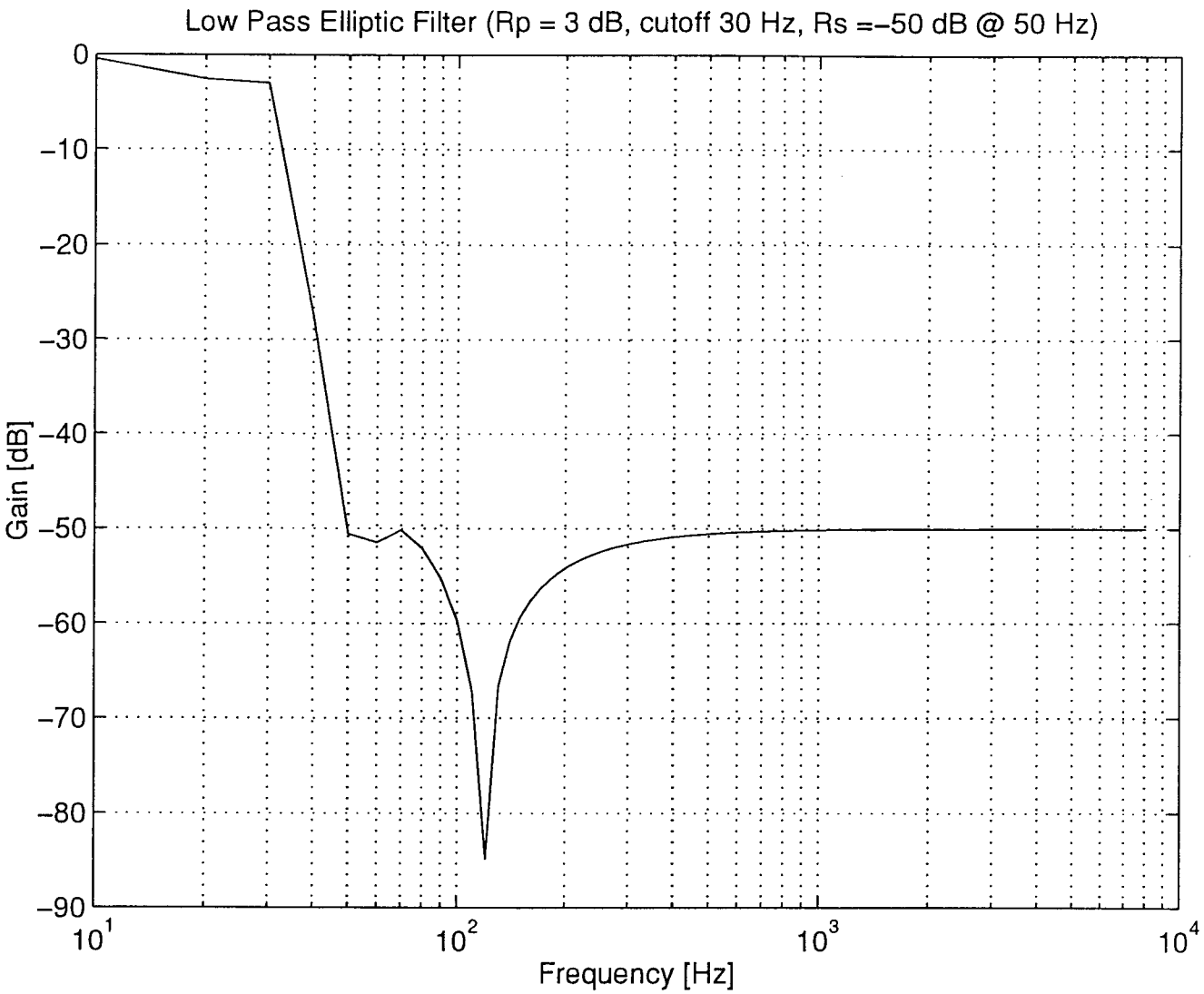

Figure 6.8: Low Pass Filter Frequency Response

- *Linear Interpolation* between the values of consecutive input vectors $\underline{x}(n)$, $\underline{x}(n+1)$, where n is given by:

$$n = \text{round}(m \cdot \frac{N_s - 1}{M - 1}), \tag{6.7}$$

and m is the output frame index. The value of the output vector, $\underline{y}(m)$ is given by:

$$\underline{y}(m) = \underline{x}(n) \cdot (1 - f) + \underline{x}(n) \cdot f \quad \forall m = 1, ..., M, \tag{6.8}$$

where

$$f = m \cdot \frac{N_s - 1}{M - 1} - n. \tag{6.9}$$

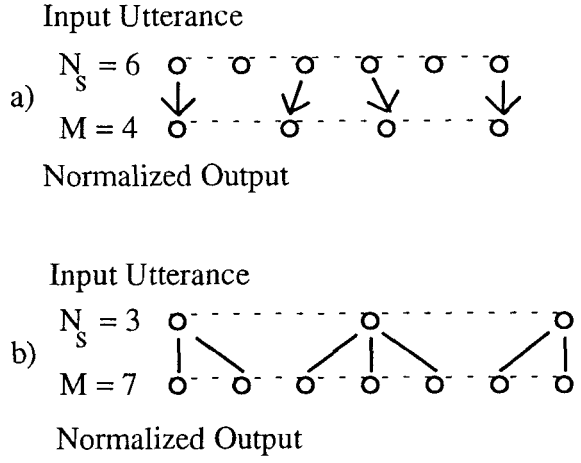

Figure 6.9: Time Normalization Using Resampling

The improvements in recognition performance obtained using the linear interpolation method were too small (under 1%) to justify the increase in design complexity. As a result, the time normalization block is implemented as a tunable down sampler, with sampling frequency f adjustable according to the variable input utterance size N_s. The term "down sampler" is used because in most cases the input utterance size $N_s > M = 40$, and as such appears to be sampled at a higher frequency than 1/40. The value $M = 40$ was chosen to provide a valid comparison with the Probability Tables method, which uses the same time normalization length of 40, and to reduce the size of the on-chip memory. Doubling M varies only slightly the recognition accuracy performance (less than 0.5%), variation which does not justify the increase in storage space.

Chapter 7

Results

The isolated words database used for testing was collected to reflect the vocabulary, the recording conditions and the audience of the intended application for the product. The worst case test conditions for the recognizer's evaluation are obtained when the noisy database recordings are combined with a simulation which complies to all the algorithmic restrictions imposed by the technical requirements.

A generic block diagram of the recognizer's evaluation system is presented in Figure 7.1, and consists of an *Analog Signal Control* block, which amplifies and band limits the signal collected from a microphone and a *Sampling* block, which digitizes the analog signal. Isolated utterances are then stored in individual files on the file system, which can be accessed by the software simulation of the recognizer (represented by the *Recognition* blocks in the diagram). A detailed description of the recording conditions and speech database structure is given in Section 7.1.

An isolated utterance recording stored in the database is processed by the *Endpoint Detection* block, which extracts the relevant speech segment from the background of silence. The spectral profile of the speech segment is generated by the *Feature Extraction* block, based on a filter-bank analyzer for feature measurement, and described in Section 6.3. The spectral profile can be used by the *VQ-Based Decision* block for training, to generate the reference patterns (codebooks), or for testing, to determine the recognition result.

The performance analysis of various quantization methods, with speech databases recorded in a variety of conditions, shows that the recognizer presented can be successfully used in practical applications with similar hardware requirements, due to its

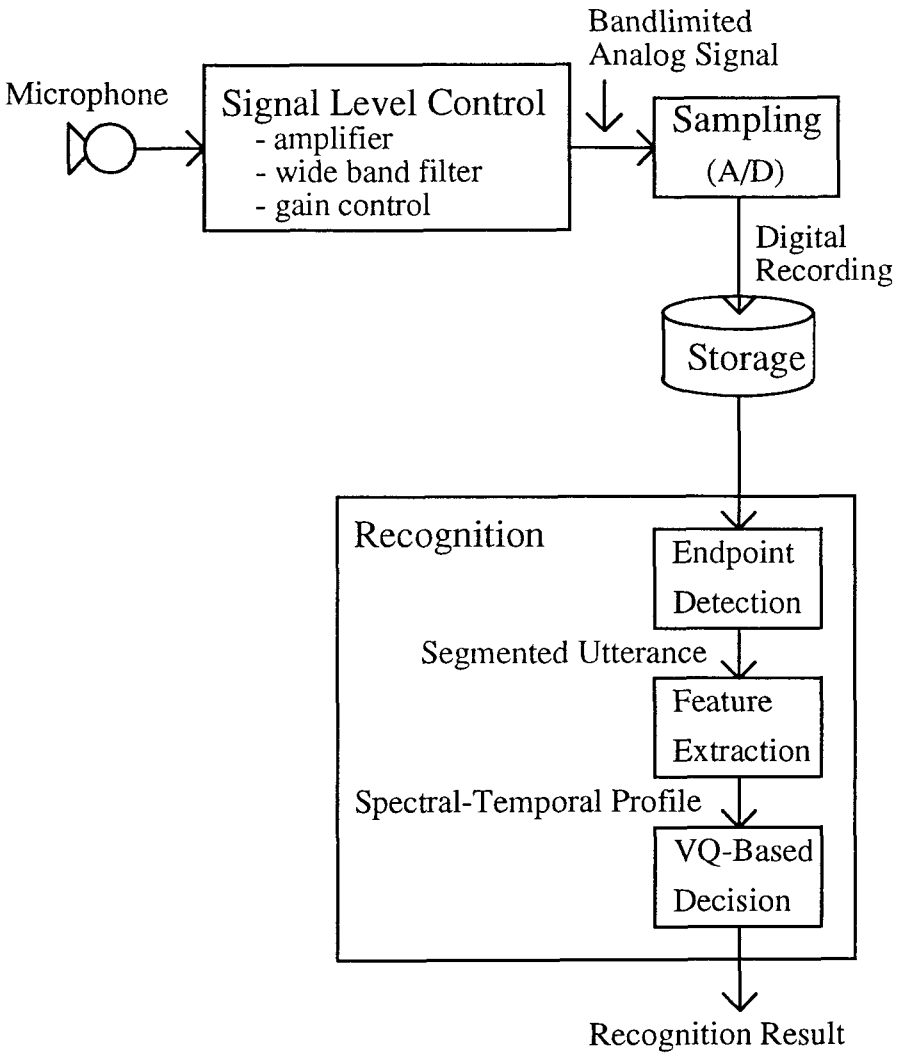

Figure 7.1: Speech Recognition Evaluation System

robustness, simplicity and high performance.

7.1 Speech Databases

The structure and recording conditions of the database used to train and test the recognition system were chosen so that the influence of the following factors on the recognition performance can be estimated: recording equipment, recording conditions (level and type of background noise) and speaker variability.

The word-based vocabulary was defined based on the application intended for the product, and contains the digits *zero* to *nine* plus *oh* and the words: *speed, error, dial, hang-up, repeat, stop, play, eject, slow, reverse, search, record, pause, rewind, forward, on, off, up, down*, collected from two sets of talkers:

- set A of 10, with English as their mother tongue;
- set B of 4, with English as a second language.

Each set was composed of 50% male and 50% female talkers. In all cases the words were spoken with pauses between them so that the tokens were not influenced by any context dependency due to neighboring words, and 10 repetitions were collected for each word.

The two different recording environments:

- (a) soundproof room conditions, and
- (b) noisy conditions,

used to collect the speech database are presented in detail in Appendix B.

The Signal Level Control block structure required by the hardware design specifications of the recognizer, was implemented according to the block diagram of Figure 7.2, to provide further noise reductions and a test setup as close as possible to the real environment of the recognizer. The microphone, the amplifier, the AGC and the WBF are as specified in Section 6.1. The AGC dynamically adjusts the gain of the internal amplifier A1 to compensate for a wide range of microphone input levels and use the dynamic range to the fullest. The gain adjustment varies between -15 dB and +24 dB. The "attack" and "release" times of the AGC can be set using an

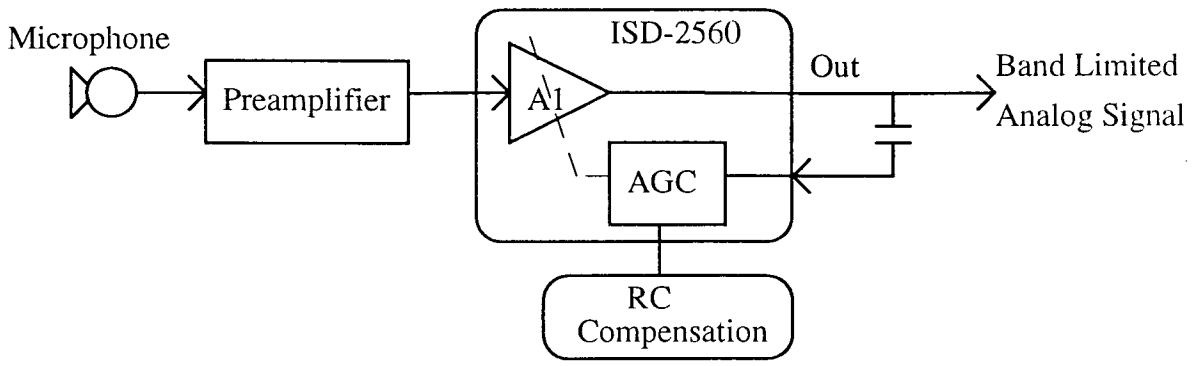

Figure 7.2: Signal Level Control Block Diagram

RC compensation network; during testing, 0.3 seconds was used for the "attack" time and 1 second was used for "release" time.

The results presented below, obtained from recordings of the digits *zero* to *nine* and the words *stop* and *reverse* repeated 10 times each, performed in the conditions of setup (a) and (b), for the speaker groups A and B, will further be referred to as:

- *studio database*: contains accent-free utterances in soundproof room conditions.
- *mixed database*: contains accent-free utterances, A, in soundproof room conditions, (a), plus utterances with foreign accent, B, in noisy conditions, (b).
- *noisy database*: contains utterances with foreign accent, B, in noisy conditions, (b).

7.2 Testing Environment Configuration

The recognition systems used to evaluate the new VQ methods proposed in this thesis, in comparison with the existent methods, were designed using the same training set for generating the dictionary of codebooks.

For a consistent evaluation, the functional structure of the testing environment, presented in Figure 7.1, must be the same for all the experiments; the functional block that changes, to reflect the particular VQ method evaluated, is the Decision block. The basic structure of a VQ-based decision block was presented in Figure 3.2. The structure of the codebooks in the dictionary and the recognition method are completely characterized by the codebook training procedure and the type of distortion method used by the algorithm. According to these criteria, the recognition systems implemented can be classified in:

- Baseline Recognizer: the training and the distortion measure are based on spectral components only, and presented in detail in Section 3.3. Uses 16 codewords, of 16 spectral components each, for every codebook.

- Probability Tables (PT) Recognizer: with training based on the temporal probability tables method, presented in Section 4.2.3, using a combined spectral-temporal distortion measure during quantization. The functional block diagram is presented in Figure 4.8. Uses 16 codewords (each with 16 spectral components and 40 components for the associated probability table) per codebook.

- 1(2) Time Component(s) (TC) Recognizer: implements the spectral-temporal VQ design with time components described in Section 5.1, and has the block diagram presented in Figure 5.5. Uses 16 codewords per codebook. Each codeword consists of 16 spectral components and 1(2) time components.

- Overlapped Segmented Codebooks (OSC) Recognizer: uses a number of adjacent overlapping sub-codebooks for each word in the dictionary and spectral distortion measure only. The training and quantization procedure are as described in Section 5.2. Each sub-codebook has 16 spectral codewords with 16 spectral components each. The overall size of the codebook, in codewords, is given by equation 5.15, for various overlap factors and number of segments per utterance.

- Segmental VQ Recognizer: as described in Section 4.2.2, with zero overlap between adjacent sub-codebooks, having 16 spectral codewords with 16 components each. The overall size of the codebook is the product between the number of segments and the sub-codebook size.

The distortion measure used in the PT recognizer requires the selection of a value for the parameter α, describing the mix of temporal and spectral distortions, so that it optimizes the recognition performance of the resulting system. The value selected was $\alpha = 0.8$, corresponding to the minimum error rate of 9.24% in Figure 7.3, which represents the recognition error of the PT recognizer for different values of α.

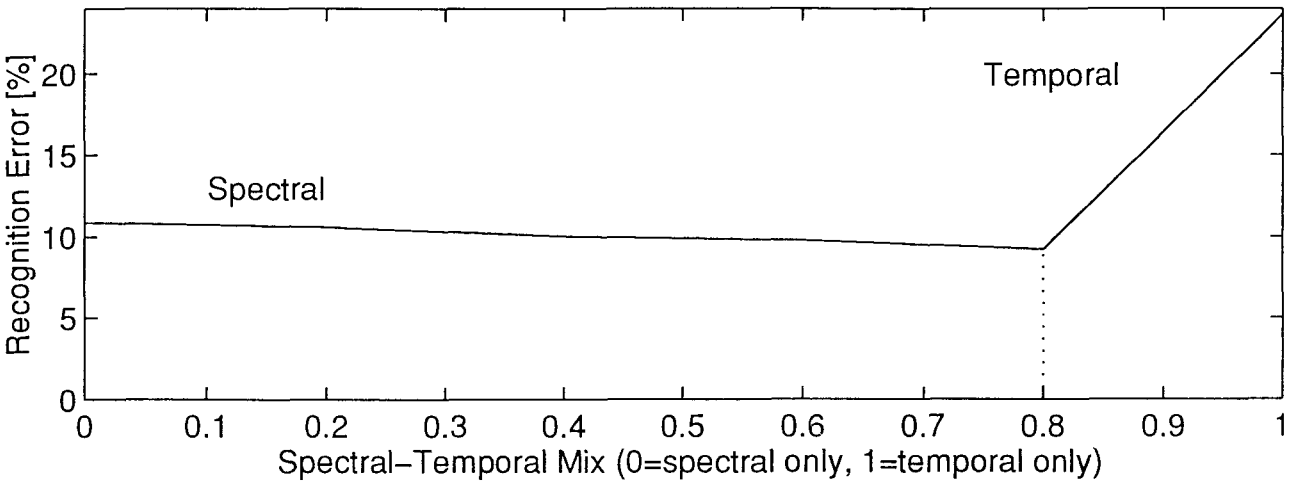

Figure 7.3: Spectral-Temporal Mix Selection

Threshold selection in the multiple time component method was performed on a similar criteria, of minimizing the recognition error of the system. The graph presented in Figure 7.4 shows the error rate for the 2 TC recognizer for the noisy database and the mixed database test set. The optimal performance is obtained for a threshold value of $\tau = 30\%$; however, the difference between the two curves suggests that the threshold selection depends on the recording conditions, and that a similar calibration be performed if these conditions change.

7.3 Recognition Results

The baseline recognizer using only the spectral components obtains a performance of about 97% for speakers who contributed to the training data or speakers with similar pronunciations recorded in conditions identical to those used for training.

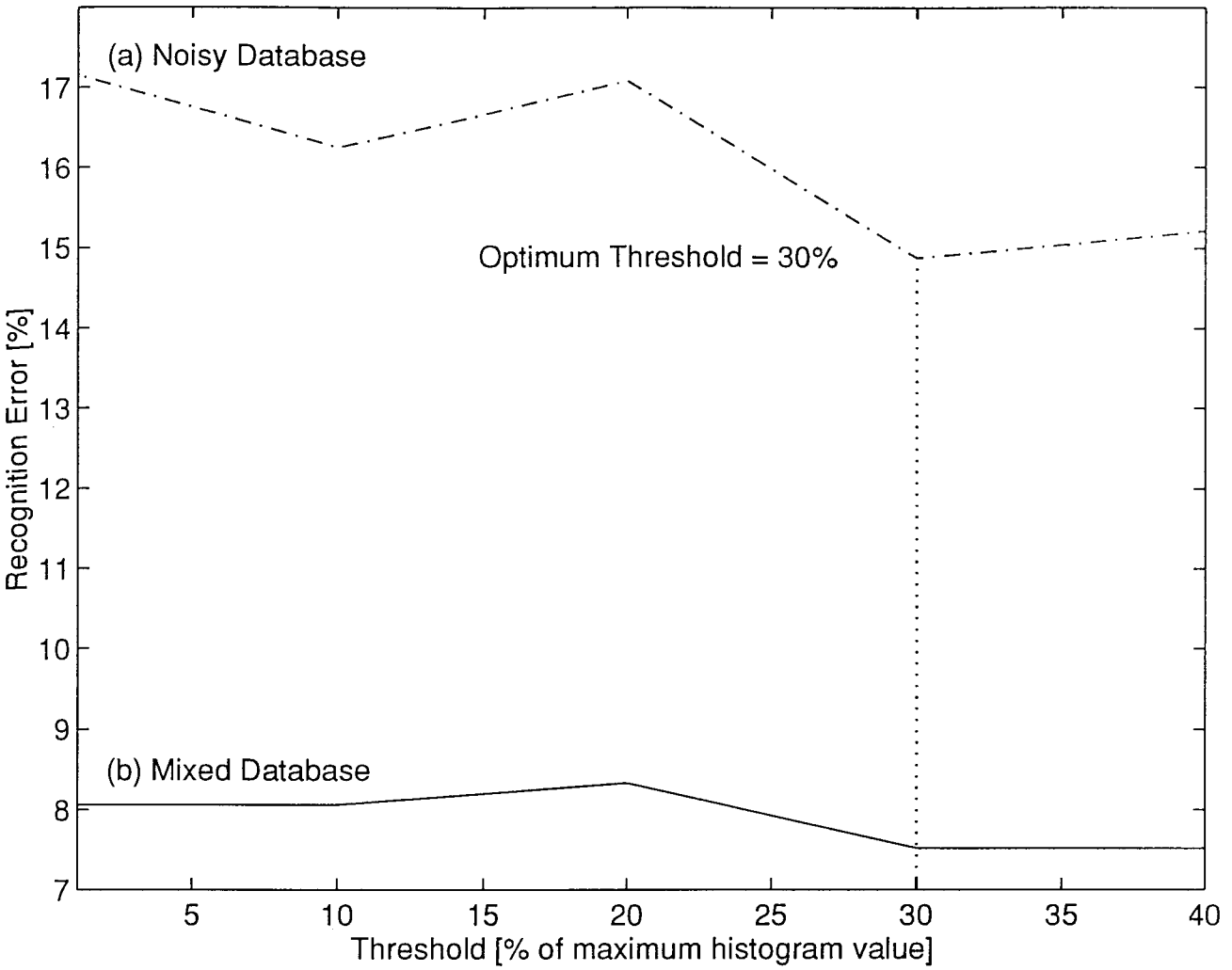

Figure 7.4: Threshold Definition for VQ with 2 Time Components

The performance degrades significantly for foreign pronunciations or different recording conditions. To enhance the performance differentiation the mixed studio database was used, with a total of 92 tokens per dictionary word (for 12 dictionary words) out of which 52 tokens were similar (pronunciation, recording conditions), to the training set and 40 tokens were recorded in different conditions using talkers with English as a second language (set B). On this test set, the baseline recognizer's performance degrades to about 89%.

By combining spectral and temporal information in the VQ structure, the performance was improved, and the error rate was reduced by about 40%.

VQ Method	Error Rate [%]	Memory/Codeword [KB]
Spectral Information Only	10.87	1
Probability Tables	9.24	3.5
1 Time Component (TC)	8.06	1.06
2 TC	7.52	1.09
Overlapped Codebooks	6.34	3.62

Table 7.1: Error Rates For Recognition Using Spectral Temporal VQ

The temporal probability tables (PT) method described in Section 4.2.3 was implemented to provide a reference for the comparison of the systems described above. For the optimal value of the parameter α, representing the mix of spectral and temporal distortions, the introduction of probability tables reduced the recognition error rate on the test set by about 1.6% (compared to about 2% for the digit set in [7]).

The recognition error rates for the systems presented in this thesis along with the required memory in words per dictionary entry is shown in Table 7.1. The results indicate that using only one time component (TC) results in better performance than the PT approach at a lower memory requirement.

The multiple time component (MTC) approach uses a variable number of time components per dictionary word with an average of about 1.5 components. The result is a further (relatively small), performance improvement.

The overlapped segmented codebooks (OSC) approach shows the best recognition accuracy: an improvement of about 2.9% with respect to the PT method, at the expense of a slightly larger memory requirement.

To obtain the result given in Table 7.1 for the OSC method, a number of tests were performed to measure the recognition rate for a variable number of input intervals (partitions) and for different overlap ratios. The outcome of these tests is shown in Figure 7.5. The best performance is achieved for an overlap ratio of 12.5 % and for 4 input partitions.

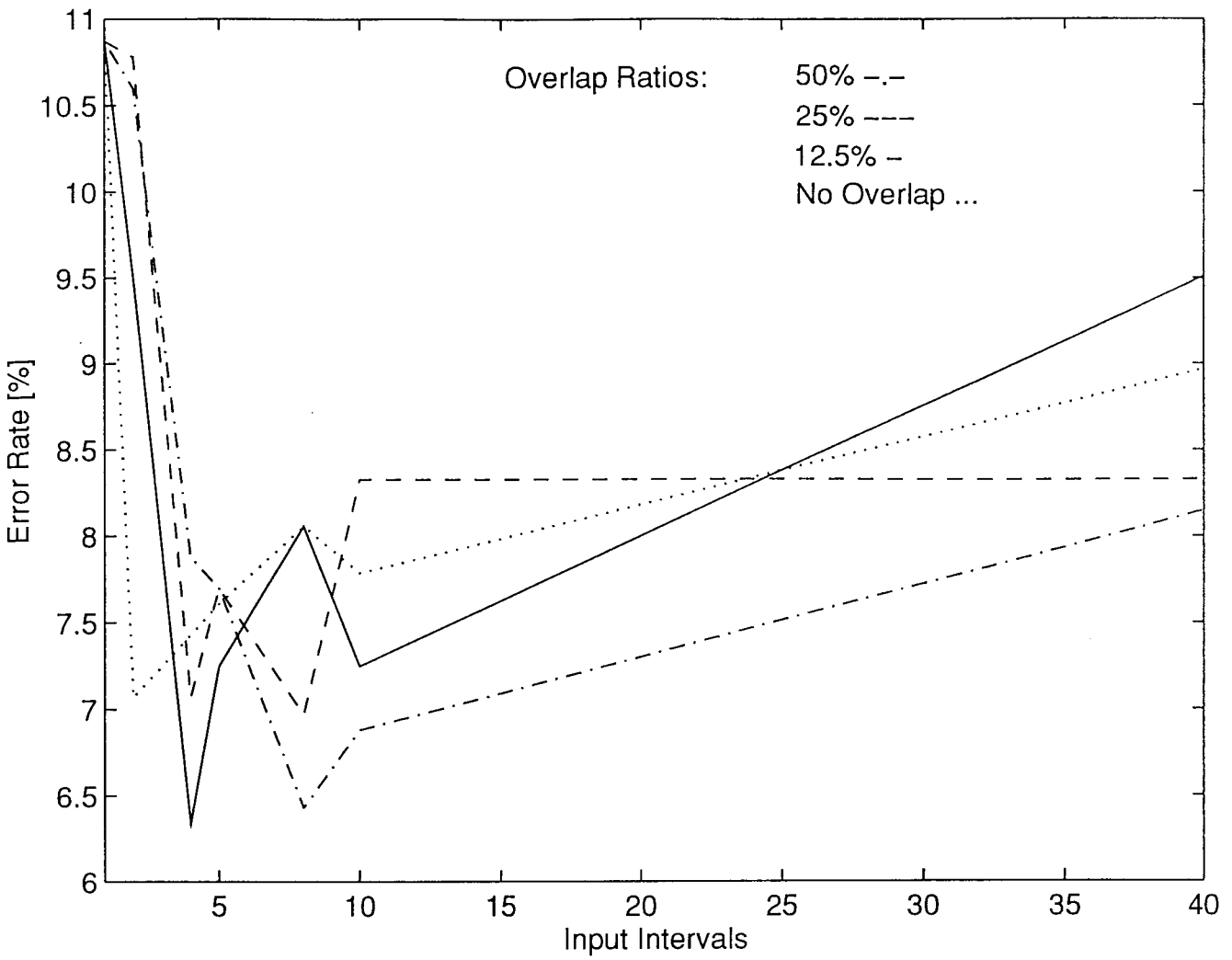

Figure 7.5: Error Rates for the Overlapped Segmented Codebooks Method

7.4 Conclusions

The performance analysis of speaker independent isolated word recognition algorithms that use a VQ as a recognition processor was presented. The error rate of a baseline recognition system based only on spectral VQ was reduced by over 40% by incorporating time information in the quantization process.

A new method, VQ with time components, of adding temporal information to a spectral codebook, was proposed and investigated. The new method gives a 20% improvement in recognition rate over the existing probability tables technique [7],

while reducing the required codebook storage space by 70%. The evaluation of the improvement due to using overlapping codebooks in segmental VQ, shows a 20% improvement in error rate at a 45% reduction in codebook storage space.

The design and implementation of a speech recognition algorithm which simulates a real-time analog VLSI recognizer (which incorporates the new VQ methods) was performed.

7.5 Future Research

The scope of the two VQ methods presented can be extended to patterns with similar temporal evolution to speech, with an expectation of increased VQ-based recognition performance.

For the temporal component method, a generalization of the spectral-temporal pattern, as a combination of any two random variables, may be explored. The random variables, denoted by A and B, must have a similar dependency, $A = f'(B)$, to the spectral-temporal function, $X = f(T)$ (where X is the spectral component, and T the temporal component). The analysis of a recognition system based on simulated patterns (A, B) may be used to determine more thoroughly the optimality of the VQ design (choice of distortion measure and joint training procedure) for recognition purposes.

The influence of the choice of spectral analysis method (LPC vs. filter banks) can also be evaluated for both methods in order to confirm that similar improvements in recognition accuracy are obtained. This result would mean that either method could be used to increase the performance of VQ preprocessors for DTW or HMM based recognizers.

References

[1] K. H. Davis, R. Biddulph, and S. Balashek, "Automatic Recognition of Spoken Digits", *J. Acoust. Soc. Am.*, 24(6), 1952, pp. 637-642.

[2] L. Rabiner, B. H. Juang, *Fundamentals of Speech Recognition*, PTR Prentice Hall, New Jersey, 1993.

[3] M. Morito, K. Yamada, A. Fujisawa, M. Takeuchi, "A Single-Chip Speaker Independent Voice Recognition System", *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 1986, pp. 377-380.

[4] J. E. Shore and D. K. Burton, "Discrete Utterance Speech Recognition Without Time Alignment", *IEEE Transactions on Information Theory*, IT-29 (4), July 1983, pp. 473-491.

[5] F. Rogers, P. Van Aken, and V. Cuperman, "Time-Frequency Vector Quantization with Application to Isolated Word Recognition", *in Proc. of IEEE International Symposium on Time-Frequency and Time-Scale Analysis*, Jun. 1996.

[6] M. McGuire, D. Ingraham, and V. Cuperman, "Speaker-Independent Isolated-Digit Recognition Using Neural Networks and Hidden Markov Models", in *Proc. Canadian Conf. on Electrical and Computer Engineering*, pp46.4.1-4, Sep. 1991.

[7] K. C. Pan, F. K. Soong, L. R. Rabiner, A. F. Bergh, "An Efficient Vector-Quantization Preprocessor for Speaker Independent Isolated Word Recognition", *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, April 1985, pp. 874-877.

[8] D. K. Burton, J. T. Buck, J. E. Shore, "Parameter selection for isolated word recognition using vector quantization," *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, March 1984, pp. 9.4.1 - 9.4.4.

[9] M. Hoshimi, M. Miyata, S. Hiraoka, K. Niyada, "Speaker Independent Speech Recognition Method Using Training Speech from a Small Number of Speakers," *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 1992, pp. 469-472.

[10] S. S. Stevens and J. Volkmann, "The relation of pitch of frequency: A revised scale," *Am. J. Psychol.*, 53, 1940, pp. 329-353.

[11] S. Furui, *Digital Signal Processing, Synthesis and Recognition*, Marcel Dekker Inc., New York, 1989.

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

[13] Y. Linde, A. Buzo and R. M. Gray, "An algorithm for vector quantizer design," *Proc. IEEE Trans. on Communications*, vol. COM-28, Jan. 1980, pp. 84-95.

[14] V. Cuperman, "Speech Coding," *Advances in Electronics and Electron Physics*, vol.82, 1991, pp. 97-196 (100 pages).

[15] S. Furui, "On the role of dynamic characteristics of speech spectra for syllable perception," *Fall Meeting of Accoustic Soc. Japan*, 1-1-2, October 1984.

[16] R. E. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, New Jersey, 1957.

[17] B. H. Juang, "Design and performance of trellis vector quantizers for speech signals," *IEEE Trans. on Acoustic, Speech, Signal Proc.*, ASSP-36,9:1423-1431, September 1988.

[18] L. F. Lamel, L. R. Rabiner, A. E. Rosenberg, J. G. Wilpon, "An Improved End-point Detector for Isolated Word Recognition", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-29, No. 4, Aug. 1981.

[19] W. C. Scratchley, "A word-based vector quantization speech recognizer with segmentation", B.A.Sc. thesis, Simon Fraser University, 1989.

[20] M. T. J. McGuire, "Neural Networks for Pre and Postprocessing in a Hidden Markov Model Based Speech Recognizer", M.A.Sc. thesis, Simon Fraser University, 1991.

Appendix A

Filter Coefficients

This section contains the coefficients for all the digital filters used in the recognition system:

- Wide Band Filter, used for input signal control
- 16 Band Pass Filters, used for feature extraction
- Low Pass Filter used to eliminate undesired high frequency images of a spectral energy profile.

All the filters are FIR filters and have the transfer function given by:

$$h(z) = \frac{b(z)}{a(z)} \tag{A.1}$$

where:

$$b(z) = B(1) + B(2)z^{-1} + ... + B(N+1)z^{-N} \tag{A.2}$$

$$a(z) = A(1) + A(2)z^{-1} + ... + A(N+1)z^{-N} \tag{A.3}$$

and N represents here the filter order.

A.1 Wide Band Filter

Wide Band Filter coefficients for band 1.202000e+02 Hz to 7.312310e+03 Hz (Rp 1 dB, Rs 30 dB)

WBF Filter Order = 6

WBF Filter Coefficients:

B(1)	0.71837450467066	A(1)	1.00000000000000
B(2)	-0.01347488386971	A(2)	-0.46155382185554
B(3)	-2.12649578320810	A(3)	-2.25202059997925
B(4)	0.0000000000000	A(4)	0.64653464063571
B(5)	2.12649578320811	A(5)	1.84370109359342
B(6)	0.01347488386971	A(6)	-0.28610725682104
B(7)	-0.71837450467066	A(7)	-0.48948744939138

A.2 Band Pass Filters

Bandpass Filter coefficients for band 1 (143 - 180 Hz), Rp = 3dB

Order 4

B(1)	0.00005224160306	A(1)	1.00000000000000
B(2)	0.00000000000000	A(2)	-3.97155676986944
B(3)	-0.00010448320613	A(3)	5.92287117772536
B(4)	0.00000000000001	A(4)	-3.93096016709417
B(5)	0.00005224160306	A(5)	0.97966134480384

Filter coefficients for band 2 (205 - 257 Hz)

Order 4

B(1)	0.00010276038200	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	-3.95500151331497
B(3)	-0.00020552076399	A(3)	5.88183340065618
B(4)	-0.00000000000000	A(4)	-3.89830117317275
B(5)	0.00010276038200	A(5)	0.97153426656749

Filter coefficients for band 3 (291 - 366 Hz)

Order 4

B(1)	0.00021242247515	A(1)	1.00000000000000
B(2)	0.00000000000000	A(2)	-3.92588881470170
B(3)	-0.00042484495031	A(3)	5.81191384748465
B(4)	0.00000000000000	A(4)	-3.84496501047260
B(5)	0.00021242247515	A(5)	0.95920349965459

Filter coefficients for band 4 (367 - 520 Hz)
Order 6

B(1)	0.00002555396912	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	-5.79351631502136
B(3)	-0.00007666190737	A(3)	14.07035995752901
B(4)	-0.00000000000000	A(4)	-18.33393286930893
B(5)	0.00007666190737	A(5)	13.51775368951552
B(6)	0.00000000000000	A(6)	-5.34739786336269
B(7)	-0.00002555396912	A(7)	0.88675724940010

Filter coefficients for band 5 (5.878600e+02 - 7.398600e+02 Hz)
Order 4

B(1)	0.00085442569533	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	-3.78501064494205
B(3)	-0.00170885139066	A(3)	5.49873944416437
B(4)	-0.00000000000000	A(4)	-3.62851147398423
B(5)	0.00085442569533	A(5)	0.91905000195763

Filter coefficients for band 6 (7.230300e+02 - 9.100300e+02 Hz)
Order 4

B(1)	0.00128109640076	A(1)	1.00000000000000
B(2)	0.00000000000000	A(2)	-3.70013909705517
B(3)	-0.00256219280153	A(3)	5.32118520153999
B(4)	0.00000000000000	A(4)	-3.51278504440851
B(5)	0.00128109640076	A(5)	0.90135840643878

Filter coefficients for band 7 (8.898200e+02 - 1.118800e+03 Hz)
Order 4

B(1)	0.00189944068486	A(1)	1.00000000000000
B(2)	0.00000000000000	A(2)	-3.57927907608794
B(3)	-0.00379888136973	A(3)	5.07884387767968
B(4)	0.00000000000000	A(4)	-3.35856514058870
B(5)	0.00189944068486	A(5)	0.88058729920618

Filter coefficients for band 8 (1.094400e+03 - 1.376400e+03 Hz)
Order 4

B(1)	0.00284077343489	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	-3.40536886301004
B(3)	-0.00568154686978	A(3)	4.74693150383291
B(4)	-0.00000000000000	A(4)	-3.14850316635546
B(5)	0.00284077343489	A(5)	0.85503657515053

Filter coefficients for band 9 (1346 - 1693 Hz)
Order 4

B(1)	0.00422876376876	A(1)	1.00000000000000
B(2)	0.00000000000000	A(2)	-3.15710875488731
B(3)	-0.00845752753753	A(3)	4.30492116508163
B(4)	0.00000000000000	A(4)	-2.86645839095400
B(5)	0.00422876376876	A(5)	0.82472244548174

Filter coefficients for band 10 (1.655500e+03 - 2.082500e+03 Hz)
Order 4

B(1)	0.00627263797547	A(1)	1.00000000000000
B(2)	0.00000000000000	A(2)	-2.80458730459993
B(3)	-0.01254527595093	A(3)	3.73640912814043
B(4)	0.00000000000000	A(4)	-2.48996378405634
B(5)	0.00627263797547	A(5)	0.78888544528158

Filter coefficients for band 11 (2.036300e+03 - 2.561300e+03 Hz)
Order 4

B(1)	0.00924971391535	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	-2.31054167525971
B(3)	-0.01849942783070	A(3)	3.05095709329683
B(4)	-0.00000000000000	A(4)	-1.99554964992515
B(5)	0.00924971391535	A(5)	0.74710399217871

Filter coefficients for band 12 (2.504500e+03 - 3.150500e+03 Hz)
Order 4

B(1)	0.01359086172003	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	-1.63238655147888
B(3)	-0.02718172344005	A(3)	2.31493228579207
B(4)	-0.00000000000000	A(4)	-1.36233781571738
B(5)	0.01359086172003	A(5)	0.69856637432951

Filter coefficients for band 13 (3.080800e+03 - 3.874800e+03 Hz)
Order 4

B(1)	0.01981145288645	A(1)	1.00000000000000
B(2)	0.00000000000000	A(2)	-0.73470896791235
B(3)	-0.03962290577290	A(3)	1.70054318427316
B(4)	0.00000000000000	A(4)	-0.58773168065435
B(5)	0.01981145288645	A(5)	0.64348953375065

Filter coefficients for band 14 (3.789300e+03 - 4.766300e+03 Hz)
Order 4

B(1)	0.02874299903119	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	0.38446385765339
B(3)	-0.05748599806238	A(3)	1.50393938979007
B(4)	-0.00000000000000	A(4)	0.29161428040783
B(5)	0.02874299903119	A(5)	0.58141945620473

Filter coefficients for band 15 (4.660700e+03 - 5.862700e+03 Hz)
Order 4

B(1)	0.04137310433702	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	1.63713343294644
B(3)	-0.08274620867405	A(3)	2.03212801121938
B(4)	-0.00000000000000	A(4)	1.16125363280203
B(5)	0.04137310433702	A(5)	0.51341511023813

Filter coefficients for band 16 (5.732800e+03 - 7.210800e+03 Hz)
Order 6

B(1)	0.01477763071382	A(1)	1.00000000000000
B(2)	-0.00000000000000	A(2)	4.17950567813215
B(3)	-0.04433289214146	A(3)	7.77512675828897
B(4)	-0.00000000000002	A(4)	8.22966185291065
B(5)	0.04433289214144	A(5)	5.23432766780934
B(6)	-0.00000000000001	A(6)	1.89608409864762
B(7)	-0.01477763071382	A(7)	0.30764623533518

A.3 Low Pass Filter

Low Pass Filter coefficients for 30 Hz cutoff (Rp 3 dB, Rs 50 dB at 50 Hz)

LPF Filter Order = 4

LPF Filter Coefficients:

B(1)	0.00315251123003	A(1)	1.00000000000000
B(2)	-0.01260181988650	A(2)	-3.99305764535535
B(3)	0.01889862029658	A(3)	5.97934313480186
B(4)	-0.01260181988650	A(4)	-3.97951261319635
B(5)	0.00315251123003	A(5)	0.99322712796432

Appendix B

Speech Database Recording Environments

Two different recording environments were used to collect the database:

- (a) for recordings performed in soundproof room conditions, having the block diagram presented in Figure B.1. The microphone and Signal Level Control blocks were supplied by the recording studio, while the digitization was performed by a DAT tape recorder, at a sampling frequency of 48 KHz. The conversion to 16 bit speech digitized at 16 KHz was performed using the Ariel S32-C system, which has no automatic gain control (AGC). To use most of the dynamic range, the amplification gain was selected manually and maintained constant for each speaker. This setup was used to collect data only from talker set A.

- (b) for live recordings (noisier than (a)), having the front end electronics specified for the hardware implementation of the recognizer, and the block diagram of Figure B.2. The microphone was connected directly to the Ariel S32-C system. The gain control was the same as for (a), no AGC was used. This setup was used to collect data only from talker set B.

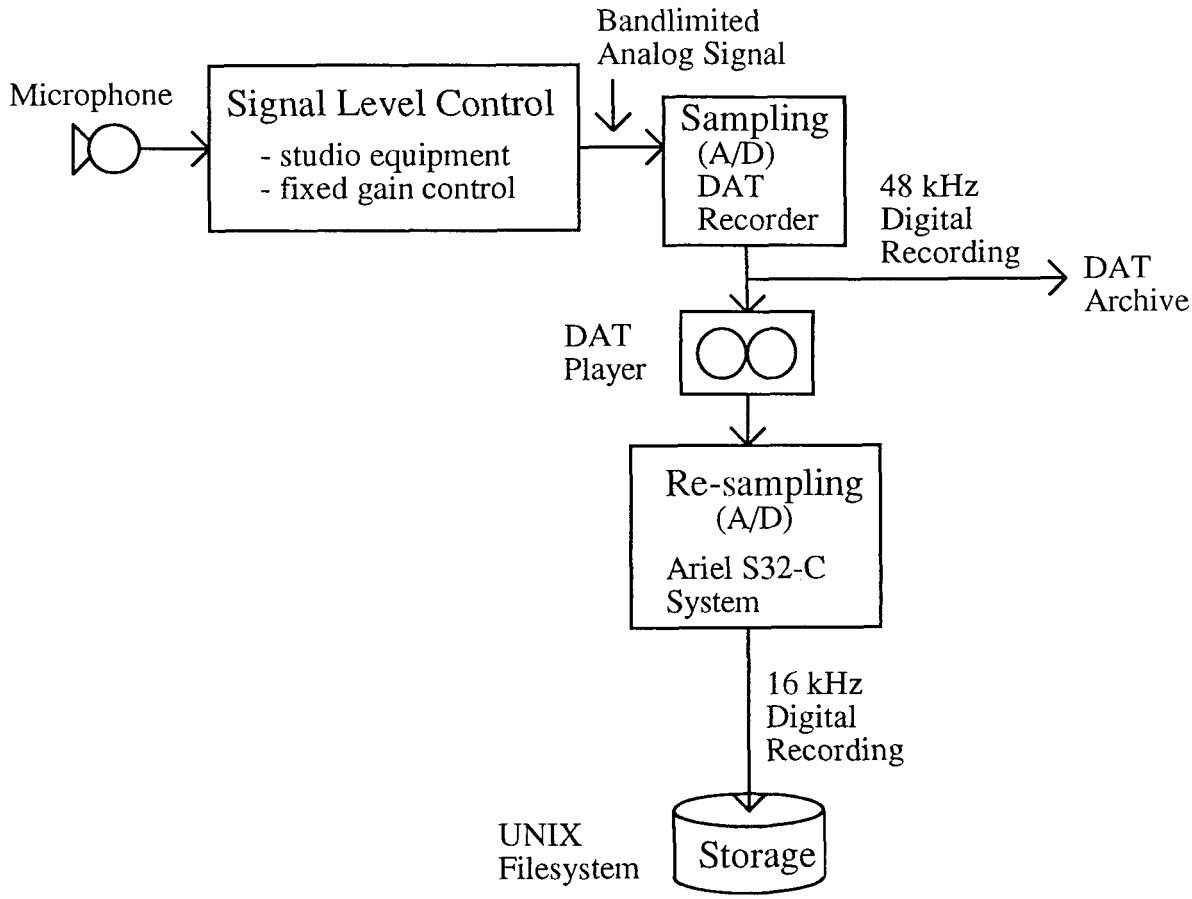

Figure B.1: Soundproof Room Recording System

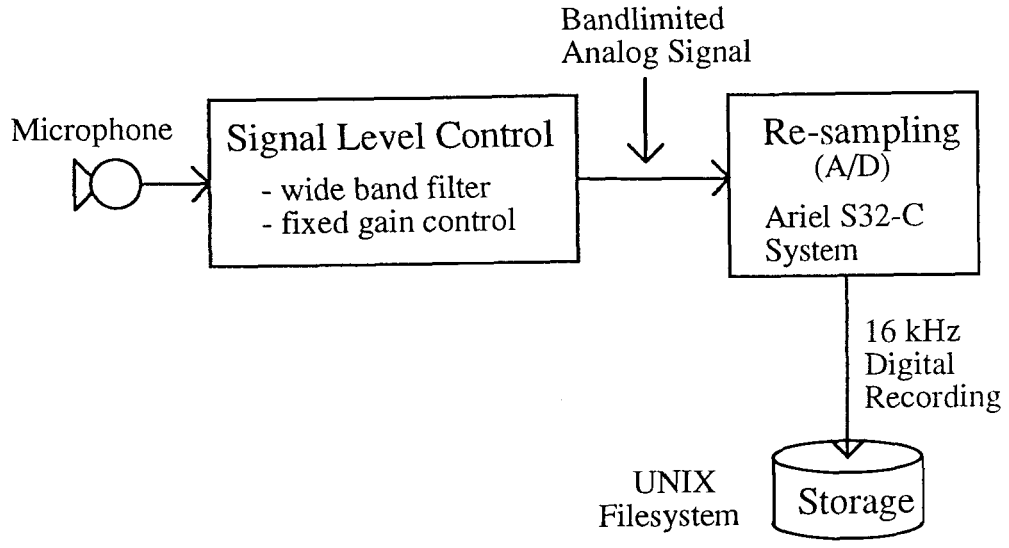

Figure B.2: Live Recording System