A Vector-Quantization-Based Preprocessor for Speaker-Independent Isolated Word Recognition

KUK-CHIN PAN, MEMBER, IEEE, FRANK K. SOONG, MEMBER, IEEE, AND LAWRENCE R. RABINER, FELLOW, IEEE

***Abstract*—In this paper, we propose a speaker-independent isolated word recognition system whose performance is comparable to that of a conventional isolated word recognizer, but whose computation is greatly reduced. The structure of the proposed recognizer consists of a word-based vector quantization (VQ) preprocessor, followed by a conventional DTW postprocessor. The purpose of the preprocessor is essentially to eliminate from further consideration all words in the vocabulary which are unlikely recognition candidates. In some cases, the preprocessor will be able to eliminate all word candidates except one; for such cases, there is no further processing required for word recognition. In all other cases (i.e., when more than one word candidate is passed on), a dynamic time warping (DTW) processor is used to resolve finer acoustical distinctions among the remaining word candidates. The performance of this type of recognizer (i.e., using a word-based preprocessor and a standard DTW comparison to make finer distinctions) is affected by a number of factors involved with the details of exactly how the system is implemented—e.g., the distortion measure used in the preprocessor and in the DTW comparison, the size of the VQ codebook for each vocabulary word, the decision thresholds of the preprocessor, etc. Several of these factors were studied experimentally using testing databases consisting of isolated digits and words from a vocabulary of 129 airline terms. The results show that the proposed preprocessor has the capability of reducing computation for recognition by up to an order of magnitude, while maintaining the same performance as that obtained using a DTW comparison without the preprocessor. A somewhat smaller reduction in memory over the straight DTW implementation is also obtained in the proposed approach.**

I. INTRODUCTION

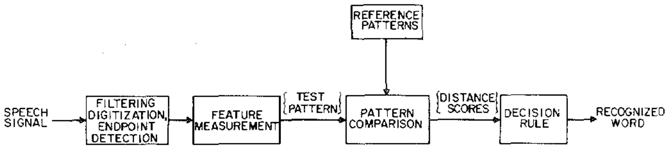

Fig. 1. Block diagram of pattern recognition model for isolated word recognition.

SEVERAL different approaches have been proposed for recognizing isolated words; the most popular and the most successful systems are still the ones based on a conventional statistical pattern recognition model [1]–[5]. Such a model is depicted in Fig. 1. The speech signal is first analyzed and a time series of feature vectors that characterize the spectral content of the speech sounds is extracted. Typically, the feature vectors are the short-time spectra of the speech signal obtained from an analysis via a bank of bandpass filters or through a linear predictive coding (LPC) analysis. The resulting test pattern (i.e., the temporal sequence of feature vectors) is then compared to a set of prestored reference patterns, and the corresponding similarity (distance) score for each reference pattern is determined. The most successful pattern comparison method for speech recognition has been the so-called dynamic programming or dynamic time warping (DTW) algorithm in which reference and test word patterns are dynamically time aligned and the resultant alignment path of maximum similarity is obtained. Associated with the pattern matching algorithm is a distance measure. This distance measure can be as simple as summing the difference of magnitudes between frames of the time-aligned reference and test patterns [1] or as complex as that generated using LPC distances [2]. The final stage of the pattern recognition model is the decision block which makes a recognition decision (or decisions) based on the similarity or distance scores provided by the pattern comparison block. The most commonly used decision rule for recognition is the nearest neighbor rule which chooses the recognized word as the reference word with the smallest distance score. A generalized version of this rule, called the *K*-nearest neighbor (KNN) rule is often used in a speaker-independent word recognition environment to achieve more robust, and usually better, recognition performance [3].

The DTW-based isolated word recognition approach described above has been shown to yield very high recognition accuracies compared to other approaches [1]–[5]. However, it does so at a high computational cost. The high number of computations required by the DTW-based algorithm has been a major obstacle to its widespread use in many applications. Recently, several different approaches have been proposed where the amount of computation is highly reduced over that required for the DTW method. These approaches include using hidden Markov models (HMM's) [6], vector-quantization-based DTW recognition [7], [8], and a recognizer without time alignment [9]. The first two alternatives are still under investigation and show great potential, but at the current time, they yield degraded performance with most reasonable size vector quantizers. The third low computation alternative

Manuscript received July 17, 1984; revised October 10, 1984. This work is based on the M.S. thesis of K. C. Pan, submitted to the Massachusetts Institute of Technology, Cambridge, May 1984.

K.-C. Pan was with the AT&T Bell Laboratories, Murray Hill, NJ 07974. He is now the Hewlett-Packard Corporation,

F. K. Soong and L. R. Rabiner are with AT&T Bell Laboratories, Murray Hill, NJ 07974.

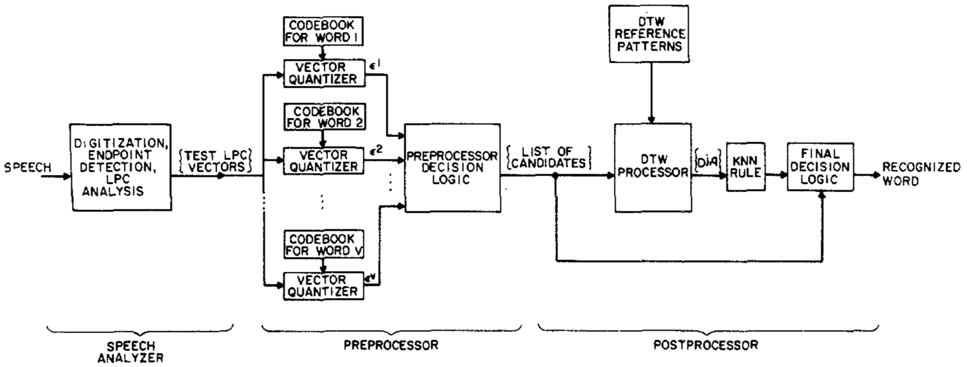

Fig. 2. Block diagram of isolated word recognizer incorporating a word-based VQ preprocessor and a DTW-based postprocessor.

used a word-based vector quantizer without relying on any temporal information. For speaker-trained isolated word recognition and with a highly nonconfusable vocabulary of 20 words, Shore and Burton reported 99 percent word recognition accuracy [9]. The results were significantly worse for speaker-independent word recognition, where an accuracy of 88 percent was achieved with the same vocabulary. This approach required relatively low computation, but its performance in a speaker-independent mode was not adequate for many applications. Recently, this method has been altered to add temporal information directly into the codebook design, with greatly improved recognition performance [10].

In this paper, we propose a complete speaker-independent, isolated word recognition system using a word-based VQ preprocessor to screen out all word candidates which are unlikely to match the unknown word. Based upon the average frame distortions of an unknown utterance with respect to a word-based VQ codebook for each vocabulary word, the preprocessor eliminates as many unlikely word candidates as possible. (In many cases, all word candidates but one are eliminated and no further processing is required.) Consequently, the preprocessor eliminates or greatly alleviates the computational load of the DTW processor. In addition, the distances required for the DTW processor are the same vector quantization distances computed in the preprocessor in our system; hence, the amount of computation of the entire recognizer is reduced even further.

The organization of this paper is as follows: in Section II we describe in detail the overall recognition system which incorporates a word-based VQ as a preprocessor. In this section, we explain the use of a hybrid distortion (distance) measure which combines the LPC likelihood ratio distance with an added energy distance. We also discuss the method used to generate the word-based VQ codebooks. In Section III, we describe the three databases used in our experiments and explain the quantitative performance measures used. In Section IV, we present the results of several pilot experiments in which we chose parameter and threshold values for the overall recognition system. In Section V, we present overall recognition results obtained by testing the recognizer on three different databases. In Section VI, we summarize our findings and discuss some directions for future research.

II. Word-Based VQ as a Preprocessor in an Isolated Word Recognition System

A. Overall Recognition System

A block diagram of the proposed isolated word speech recognizer is given in Fig. 2. The system consists of three major blocks, namely, an LPC spectral analyzer, a word-based VQ preprocessor, and a DTW postprocessor. The input speech signal is first digitized, endpointed into words, and spectrally analyzed. As a result, a time series of LPC vectors is thus formed. The preprocessor includes a set of word-based VQ codebooks. Each word in the vocabulary is characterized by its own codebook. For an unknown test word, the series of LPC vectors extracted from the analyzer is vector quantized by each individual codebook, and the corresponding frame distortions are accumulated over the duration of the word. The average frame distortion of each codebook is calculated, and the preprocessor decision logic chooses a list of possible word candidates for further processing. In the case when only a single word candidate is selected for postprocessing, the preprocessor makes a final recognition decision, and no further processing is required. Otherwise, no final decision will be made by the preprocessor and the set of "good" candidates are passed to the DTW postprocessor (i.e., unlikely candidates which are of large average distortion are eliminated by the preprocessor). By incorporating this preprocessor into the standard recognizer structure of Fig. 1, we can dramatically alleviate the computational load of the DTW postprocessor or even eliminate it completely if a final recognition decision can be made by the preprocessor. In addition, the DTW postprocessor can be even further simplified by vector quantizing the reference word templates with the VQ codebooks stored in the preprocessor. In this way, in the recognition phase, the distances (or the frame distortions) needed in the DTW process are readily available through a simple table-lookup procedure since all distances have been computed in the preprocessor stage. Hence, the computational complexity of the DTW processor is further reduced.

1) The VQ Preprocessor: The recognition procedures of the VQ preprocessor can be formally stated as follows. Assume that there are N frames in a test utterance (word to be recognized) which can be represented spectrally by N LPC vectors $a_1, \cdots, a_N$. Also assume that there are V words in the recognition vocabulary and the LPC codebook for the ith word is $b_1, b_2, \cdots, b_L$ where L is the codebook size. (We assume a fixed codebook size for each word.)

The average distortion by encoding (vector quantizing) the test utterance with the ith codebook is given by

$$\epsilon^i = \frac{1}{N} \sum_{n=1}^{N} \min_{1 \le l \le L} [d(a_n, b_l^i)] \tag{1}$$

where $d(a_n, b_l^i)$ is the distance between the two LPC vectors a_n and b_l^i.

The decision rule logic of the preprocessor consists of two possible decisions based on the average distortions $\{\epsilon^i | 1 \le i \le V\}$. Define the word i^* as the word whose codebook yields the minimum averaged distortion, i.e.,

$$i^* = \operatorname*{argmin}_{1 \le i \le V} \epsilon^i \tag{2}$$

and the word k^* as the word whose codebook yields the second smallest distortion, i.e.,

$$k^* = \operatorname*{argmin}_{\substack{1 \le i \le V \\ i \ne i^*}} \epsilon^i \tag{3}$$

The decision rules of the preprocessor are as follows.

Rule 1 (Final Word Candidate Rule):

$$\text{Choose word } v_{i^*} \text{ iff } \epsilon_{i^*} \le d_1 \text{ and } \epsilon^{k^*} - \epsilon^{i^*} \ge d_2 \tag{4}$$

where d_1 and d_2 are two empirically chosen thresholds for the best absolute average distance and the separation between the best and the second best candidates. Rule 1 basically chooses word v_{i^*} when its average frame distortion is sufficiently small that a good confidence match is achieved, and when no other good matches occur among all other vocabulary words. If Rule 1 fails, then Rule 2 is used to select all likely candidates and pass them to the DTW postprocessor.

Rule 2 (Valid Candidates Rule):

Pass word v_i to DTW postprocessor iff $\epsilon^i - \epsilon^{i^*}$

$$\le d_3 \quad i = 1, 2, \cdots, V. \tag{5}$$

This rule says that any word whose average VQ distortion is within some threshold of the minimum distortion is considered a valid candidate for further DTW processing.

2) DTW Processor: In the DTW processor, the test utterance is compared to reference patterns of the candidates that were not eliminated by the preprocessor. We use $Q = 12$ reference patterns for each word in the recognition vocabulary to account for speaker differences [3]. Hence, Q time warps are performed for each word candidate passed on to the DTW processor, with each time warp resulting in a distance score between the test and reference patterns along the resulting time alignment path.

The KNN rule is used to select the recognized word using the distance scores for all the reference patterns of the DTW candidates. If we assume that there are J word candidates (passed from the preprocessor), and we define $D^{j,q}$ as the DTW distance of the qth reference pattern ($q = 1, 2, \cdots, Q$) of the jth candidate word ($j = 1, 2, \cdots, J$), then by reordering the Q DTW distances for each word, we have

$$D^{j,[1]} \le D^{j,[2]} \le \cdots \le D^{j,[Q]} \tag{6}$$

and the average distance of the best K patterns for each word (i.e., the K nearest neighbor (KNN) rule) is then given by

$$r^j = \frac{1}{K} \sum_{k=1}^{K} D^{j,[k]}. \tag{7}$$

The final word recognition decision is then given by j^* where

$$j^* = \operatorname*{argmin}_{1 \le j \le J} r^j. \tag{8}$$

With $Q = 12$, K is typically chosen as 2.

There are three different ways to prepare the reference and test word patterns for the DTW process, namely, 1) no vector quantization of either reference or test vectors, 2) vector quantizing both the reference and the test word patterns, and 3) vector quantizing the reference patterns only. The first method is expected to achieve the best recognition performance, in general, because no quantization error is introduced. By vector quantizing both the reference and test patterns (in the second method), we can replace the distance computations in the DTW processing by simple table-lookup operations, but at a price of performance degradation. However, it has been shown that the performance degradation due to vector quantizing the test word patterns is absolutely unnecessary [7], [8]. This is because if we want to vector quantize the test word patterns, we have to compute the distances between each VQ codebook vector and each frame of the test utterance. These distances are exactly the same distances needed in the DTW process if we vector quantize only the reference word patterns. Therefore, no computation overhead is required if we choose not to vector quantize the test pattern and the resulting performance degradation is eliminated. The implementation of this procedure is straightforward. The frame distortions, computed in the preprocessor stage, are stored in a table as illustrated in Fig. 3. In the DTW process, the distance between an input frame, e.g., a_i, and the vector quantized reference frame, e.g., b_j^k, can be easily retrieved from the distance table of Fig. 3. There are two possible implementations in vector quantizing the reference patterns. We can vector quantize a reference word pattern either by its own word-based codebook or by all codebooks. Except for the one time effort in quantizing the reference word patterns, these two implementations require exactly the same amount of computation in the recognition process.

In summary, three different ways can be used to prepare word templates, namely,

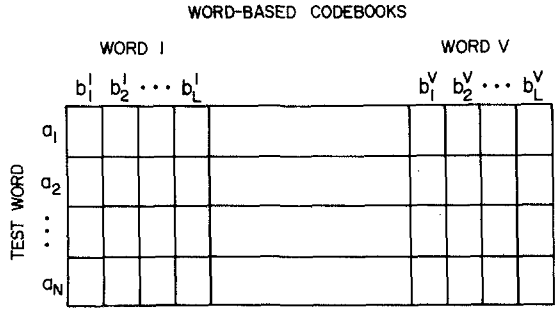

Fig. 3. Distance computation in the word-based VQ preprocessor. By saving each individual distance computation in a table, a reduction in quantization error in the DTW stage is achieved.

1) no VQ, usé unquantized test and reference patterns

2) VQ/REF, SELF VQ, vector quantize each reference pattern with its own codebook only

3) VQ/REF, ALL VQ, vector quantize each reference pattern with all codebooks.

B. Log Likelihood Ratio LPC Distance and Additional Energy Information

The LPC spectral analysis can be interpreted as a spectral matching method which minimizes the difference between the input signal spectrum and all-pole model spectrum. The spectral difference used is commonly known as the Itakura–Saito (I–S) distortion measure, and has the form

$$D_{IS} = \frac{1}{2\pi}\int_0^{2\pi}\left[\log\left[\frac{S(\omega)}{S_{in}(\omega)}\right] + \frac{S_{in}(\omega)}{S(\omega)} - 1\right]d\omega \qquad (9)$$

where $S_{in}(\omega)$ is the input signal spectrum and $S(\omega)$ is the resultant LPC all-pole model spectrum given by

$$S(\omega) = \frac{\sigma^2}{|1 + a_1e^{-j\omega} + a_2e^{-2j\omega} + \cdots + a_pe^{-jp\omega}|^2}\cdot \qquad (10)$$

The same distortion measure can be used to measure the spectral difference between two all-pole spectra $S_R(\omega)$ and $S_T(\omega)$. The corresponding Itakura–Saito distortion measure can be computed in the autocorrelation domain as

$$D_{IS}(S_R,S_T) = \frac{a_R'V_Ta_R}{\sigma_T^2} + \log\left(\frac{\sigma_T^2}{\sigma_R^2}\right) - 1 \qquad (11)$$

where $a_R = [1, a_1, a_2, \cdots, a_p]'$, V_T is the Toeplitz matrix of the autocorrelation of $S_T(\omega)$, and σ_T^2 and σ_R^2 are the LPC gain terms for the two frames. Without modification, the original I–S distortion measure is not appropriate for speech recognition applications because it is sensitive to the absolute gain. For example, the I–S distortion between a frame of speech and an amplified version of itself is

$$D_{IS} = \log\left[\frac{\sigma_T^2}{\sigma_R^2}\right] \neq 0, \qquad (12)$$

which is undesirable. To accommodate this gain sensitivity problem, there exist two modifications of the I–S distortion, namely, a gain-normalized I–S distortion measure and a gain-optimized measure. The first one sets the gains σ_T^2 and σ_R^2 to be equal, and the resultant distortion measure (likelihood ratio) is

$$D_{LR} = D_{IS}\left(\frac{S_R(\omega)}{\sigma_R^2}, \frac{S_T(\omega)}{\sigma_T^2}\right) = \frac{a_R'V_Ta_R}{\sigma_T^2} - 1. \qquad (13)$$

The second one, proposed by Itakura [2], is of the form

$$D_I = \log\left(\frac{a_R'V_Ta_R}{\sigma_T^2}\right). \qquad (14)$$

Both modifications yield roughly the same recognition performance. Since the likelihood ratio measure needs no logarithm operations and it is more mathematically tractable in VQ codebook generation, it has been widely used [11], [12].

Incorporating naively the absolute energy (or gain) information in the distortion measure such as the original I–S distortion is inappropriate for speech recognition since gain variations, between recordings, can lead to large distances between inherently similar spectra. However, since the energy contour of an utterance carries some phonetically relevant prosodic information, by adding energy (or gain) information to the spectral distortion measure *properly*, we should be able to improve the recognizer performance. The method we use is the one proposed by Brown and Rabiner [13]. The combined distortion measure between two frames of speech, i.e., frame n of a test pattern $T(n)$ with an LPC vector a_T and frame m of a reference pattern $R(m)$ with an LPC vector a_R, can be written as

$$D(T(n),R(m)) = D_{CLR}(T(n),R(m)) + \alpha f(D_E(T(n),R(m))). \qquad (15)$$

The likelihood ratio distance has been modified to include a clip threshold d_{CLIP} and the resultant clipped likelihood ratio distance D_{CLR} is given as

$$D_{CLR}(T(n),R(m)) = \min\left\{\frac{a_R'V_Ta_R}{a_T'V_Ta_T} - 1, d_{CLIP}\right\} \qquad (16)$$

where V_T is the Toeplitz autocorrelation matrix of the test frame and d_{CLIP} is a suitably chosen maximum allowable value for D_{LR} (in practice, we use a value of 2.5) which prevents any single distance from greatly altering the average frame distortions in the preprocessor or greatly interfering with the DTW optimal path search. The energy distance $D_E(T(n), R(m))$ is given by

$$D_E(T(n),R(m)) = |(E^R(m) - E_{\max}^R) - (E^T(n) - E_{\max}^T)| \qquad (17)$$

where $E^R(m)$ is the log energy (in decibels) of the frame $R(m)$ and $E_{\max}^R$ is the maximum energy (in decibels) over all frames of the whole utterance, and similarly for $E^T(n)$ and $E_{\max}^T$. In this manner, each frame energy is normalized with respect to the maximum point in the energy contour of the entire utterance and the resultant $E^R(m) - E_{\max}^R$ (or $E^T(n) - E_{\max}^T$) is the relative energy measured in decibels down from $E_{\max}^R$ (or $E_{\max}^T$).

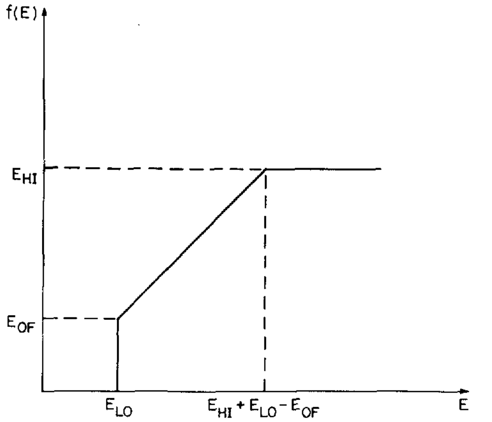

Fig. 4. Nonlinearity applied to log energy difference between frames for the energy distance calculation.

The nonlinear function f on the energy distance is defined as

$$f(E) = \begin{cases} 0 & |E| \le E_{LO} \\ |E| - E_{LO} + E_{OF} & E_{LO} < |E| \le E_{HI} + E_{LO} - E_{OF} \\ E_{HI} & E_{HI} + E_{LO} - E_{OF} < |E| \end{cases} \tag{18}$$

where E_{LO}, E_{OF}, and E_{HI} are three appropriately chosen thresholds. The function $f(E)$ is shown in Fig. 4. It assigns no penalty for any $|E|$ less than E_{LO} (i.e., small energy difference) to account for insignificant variations of energy. For large energy differences, $f(E)$ is clipped at an appropriate level E_{HI} to prevent it from becoming unbounded. Energy differences between these two extremes are linearly weighted by a factor of α.

C. Generation of Word-Based LPC Vector Codebooks

In the preprocessor, each word is represented by the VQ codebook. Each codebook is designed from a training sequence that comprises the LPC vectors from a large number of utterances of the particular word spoken by many different talkers. These same training sequences are used in template creation; hence, no additional recording is required for codebook generation over that required for template generation.

Assume a set of training LPC vectors c_i, $i = 1, \cdots, I$. We design a codebook of an optimal set of LPC vectors b_l, $l = 1, \cdots, L$ such that for this given size L, the average distortion in replacing each vector in the training set c_i by the closest vector b_l is minimized. This average distortion is given as

$$D_L = \frac{1}{I} \sum_{i=1}^{I} \min_{1 \le l \le L} [d(c_i, b_l)]. \tag{19}$$

The iterative algorithm we used to solve (19) is essentially the one proposed by Linde *et al.* [11] or commonly called the binary-split algorithm. There are several variations to the above-mentioned algorithms in terms of different splitting procedures and the way to repopulate any empty cell in the training procedure. However, Rabiner *et al.* [14] have found that several different variations in the training procedure gave rise to insignificant differences in the resulting codebooks. Therefore, the binary-split procedure is chosen because computationally it is the most efficient.

III. Descriptions of Databases and Performance Measure

A. Databases

Three different databases were used to test the proposed overall isolated word recognizer. They consisted of two different vocabularies, a ten-word digits vocabulary (zero through nine) and a 129-word airline vocabulary [15], [16]. Two different databases were used for the digit vocabulary and one for the airline vocabulary.

The first database for the ten-word digits vocabulary [3] was generated by 100 talkers, equally divided between male and female. Each talker gave two repetitions of all the ten digits with a period of four weeks separating the recording of the two repetitions. Hence, for each repetition, there were 100 utterances of each word in the vocabulary, giving a total of 1000 utterances for the entire repetition. We shall call this database the REFERENCE database, denoting the first repetition as the training set and the second repetition as the test set.

The second database for the digits vocabulary, which we shall call the KLS database [17], was generated by a set of ten talkers, also equally divided between male and female. These ten talkers were different from the 100 talkers who generated the REFERENCE database. Each talker spoke each digit 20 times, giving 200 total utterances per talker, and a total of 2000 utterances for the entire database. These ten talkers were chosen from a larger population of 100 talkers on the basis of their observed higher error rates in previous recognition experiments.

The airline vocabulary comprised 129 words that are commonly used in airline information and reservation systems, such as city names, days of the week, digits, words used to combine the terms into sentences, etc. The database for the airline vocabulary consisted of a training set and a test set. The training set was generated by a set of 100 talkers (50 males and 50 females), each of them speaking all the words in the recognition vocabulary once. This gave 100 utterances for each of the 129 words in the recognition vocabulary. The test set for the airlines database was generated by a set of 20 talkers who were different from those who generated the training set. This set of 20 talkers was also evenly divided between male and female; every talker spoke each word in the vocabulary once. All three databases described above have been used in previous recognition experiments [3], [16], [17]. All the isolated utterances were recorded off a standard dialed-up telephone line, and the endpoints of words were carefully monitored for accuracy. The two sets of data for the digits vocabulary were used to give an indication of the performance of the overall recognizer for a small size, standard vocabulary. The airline database was intended to show the feasibility of using the proposed recognizer on a much bigger and harder vocabulary that had many more acoustically similar words than the simple digits vocabulary.

B. Performance Measures

Recognition accuracy, computational complexity, and storage requirements are the three main criteria we use to judge the performance of the overall speech recognizer. To quantitatively characterize the errors (as well as the correct decisions) made by the recognizer at different stages of the overall system, we define the following parameters.

C1: Average fraction of correct final decisions made by the preprocessor, i.e., when a single candidate is chosen by the preprocessor.

E1: Average fraction of errors made by the preprocessor when a final recognition decision is made.

E2: Average fraction of errors made by the preprocessor when it fails to pass the true word candidate to the DTW postprocessor.

C3: Average fraction of correct decisions made by the DTW postprocessor.

E3: Average fraction of errors made by the DTW postprocessor.

Two useful performance parameters can be derived from the above definitions, namely, the following.

γ: Average fraction of all recognition trials in which the preprocessor makes a final decision, correct or wrong. Obviously, $\gamma = C_1 + E_1$.

β: Average fraction of candidates passed to the DTW postprocessor when no final decision is made by the preprocessor.

The breakdown of all the sources of errors and correct decisions and the corresponding stages where the errors and correct decisions are made is depicted in Fig. 5. The total error rate of the overall system is the sum of the preprocessor errors $E1 + E2$ and the DTW postprocessor errors $E3$.

C. Computational Complexity

A major consideration in evaluating the performance of the overall recognizer of Fig. 1 is its computational complexity. To understand this issue, we have to examine the required computation in the preprocessor C_{PRE}, the DTW postprocessor C_{DTW}, and the overall system. To quantify these concepts, we first define C_{PRE} as

$$C_{\text{PRE}} = V \cdot N \cdot L \qquad \text{(distance calculations)} \tag{20}$$

where a distance computation requires approximately $(p+1)$ additions and $(p+1)$ multiplications for a pth-order LPC analysis using the likelihood ratio distance. The energy distance adds negligible computation if the logarithms are computed before the distance calculation.

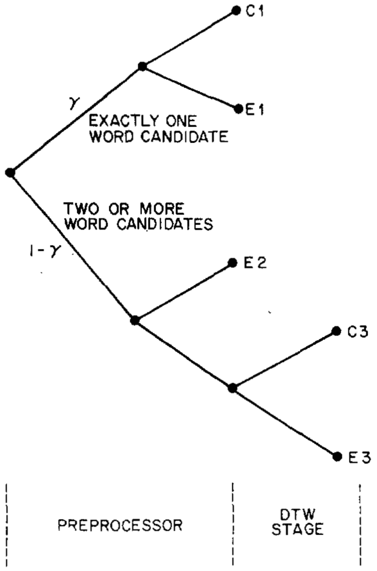

Fig. 5 A breakdown of the potential sources of error in the preprocessor and in the DTW stage of the recognizer.

The computational complexity of the DTW processor C_{DTW} is readily shown to be a function of the vocabulary size V, the number of reference templates for each word Q, and the type of DTW implementation (i.e., whether or not VQ is used). There are two components to C_{DTW}, namely, distance calculations, and combinatorics. When VQ is used, the cost of distance calculations becomes negligible (they become table lookups), and when no VQ is used, the cost of distance calculations dominates the computation. Using the approximation that the cost of a combinatorics calculation is about 1/5 that of a full distance calculation (as is the case on most general-purpose computers), we get the following results:

$$C_{\text{DTW}} = \tfrac{6}{5}\, VQ\, N^2/3 \qquad \text{(distance calculations)} \tag{21}$$

and

$$C_{\text{DTW/VQ}} = \tfrac{1}{5}\, VQ\, N^2/3 \qquad \text{(distance calculations).} \tag{22}$$

The overall computation of the recognizer of Fig. 2, C_{ALL}, can now be expressed in terms of C_{PRE}, C_{DTW}, γ, and β. The resulting expression is of the form

$$C_{\text{ALL}} = C_{\text{PRE}} + (1 - \gamma)\beta C_{\text{DTW}} \tag{23}$$

when no VQ is used in the DTW processor, and

$$C_{\text{ALL/VQ}} = C_{\text{PRE}} + (1 - \gamma)\beta\, C_{\text{DTW/VQ}} \tag{24}$$

when a VQ is used in the DTW processor.

From (20)–(24), we get

$$C_{\text{ALL}} = NV[L + (1 - \gamma)\beta\, \tfrac{6}{15}\, QN] \tag{25a}$$

$$C_{\text{ALL/VQ}} = NV[L + (1 - \gamma)\beta\, \tfrac{1}{15}\, QN]. \tag{25b}$$

We can readily define computational gain ratios as

$$R_{\text{ALL}} = \frac{C_{\text{DTW}}}{C_{\text{ALL}}} \tag{26a}$$

and

$$R_{\text{ALL/VQ}} = \frac{C_{\text{DTW}}}{C_{\text{ALL/VQ}}} \tag{26b}$$

which express the reduction in computation of the overall recognizer as compared to a conventional DTW system without VQ. Consider "typical" values of the system parameters for a digits vocabulary. For this case, we have $\gamma = 0.9$, $\beta = 0.2$, $V = 10$, $Q = 12$, $N = 40$, $L = 16$, giving

$$R_{\text{ALL}} = 10$$

$$R_{\text{ALL/VQ}} = 11.5.$$

Hence, a reduction in computation of more than 10 to 1 is entirely feasible with the proposed approach.

D. Storage Considerations

In our storage analysis, we will only be concerned with the amount of memory or storage required by the VQ codebooks, the test utterance, the reference patterns for the DTW processor, and the local distance storage required when DTW processing is done using vector-quantized reference patterns. We will neglect the differences in program storage requirements between the overall recognizer and that of the DTW processor for the reasons that the program storage requirements are negligible compared to the factors mentioned above. Recall that there are V words in the recognition vocabulary, L LPC vectors in each VQ codebook, Q DTW reference templates per word, and we assume that there is an average of N frames of speech per reference pattern, and each reference or test template is of $(p+1)$ dimensions. Under these assumptions, the amount of memory required by the preprocessor and the DTW processor is given by

$$M_{\text{PRE}} = V \cdot L \cdot (p+1) \quad \text{(floating-point numbers)} \tag{27a}$$

$$M_{\text{DTW}} = V \cdot Q \cdot N \cdot (p+1) \quad \text{(floating-point numbers).} \tag{27b}$$

The memory required to store the test utterance is

$$M_{\text{TEST}} = N \cdot (p+1) \qquad \text{(floating-point numbers).} \tag{28}$$

The memory required to store the local distances computed in the preprocessor state is given by

$$M_{\text{DIST}} = N \cdot V \cdot L \qquad \text{(floating-point numbers).} \tag{29}$$

When using vector-quantized reference patterns for the DTW processor, the LPC vectors of the text utterance do not have to be retained since the DTW algorithm will only require the table of local distances computed by the preprocessor. The memory required for storing the distance table then becomes $M_{\text{DIST}} = N \cdot L$. In addition, each vector of the reference patterns need only be stored as an integer index of the corresponding codebook vector, as opposed to the $(p+1)$ floating-point numbers required for each LPC vector. The resultant memory requirement is then

$$\begin{aligned} M_{\text{DTW/VQ}} &= V \cdot Q \cdot N \qquad \text{(fixed-point numbers)} \\ &= \tfrac{1}{2} V \cdot Q \cdot N \qquad \text{(floating-point numbers)} \end{aligned} \tag{30}$$

where we have assumed the storage requirement for a fixed-point number to be half the storage requirements for a floating-point number.

With the above assumptions, the total amount of memory M_{ALL} is

$$\begin{aligned} M_{\text{ALL}} &= M_{\text{PRE}} + M_{\text{TEST}} + M_{\text{DTW}} \qquad \text{(DTW/no VQ)} \\ M_{\text{ALL/VQ}} &= M_{\text{PRE}} + M_{\text{DIST}} + M_{\text{DTW/VQ}} \qquad \text{(DTW/VQ).} \end{aligned} \tag{31a}$$

The storage requirement of a straight DTW recognizer (without a VQ preprocessor) is

$$M_{\text{ALL}} = M_{\text{DTW}} + M_{\text{TEST}}. \tag{32}$$

IV. Initial Pilot Experiments

Before evaluating the proposed recognizer, we first carried out some pilot experiments to study the effects of various system parameters on the recognizer performance. In particular, we studied the following.

1) Performance of the preprocessor when energy information is incorporated into the distortion measure.

2) Performance of the preprocessor when the decision thresholds d_1, d_2, d_3 are varied.

3) Effects of adding energy information to the distortion measure on the performance of the DTW processor.

4) Effects of using word-based VQ codebooks for quantizing the reference templates in the DTW processor.

All pilot experiments were conducted using the REFERENCE database unless specified otherwise.

A. The Effects of Adding Energy Information to the Distortion Measure in the Preprocessor

Three different sets of parameters were used to find the best way to incorporate energy information in the distortion measure. We used several different sets of parameters, α, E_{LO}, E_{HI}, and E_{OF}, including the following.

1) $\alpha = 0.0$ (LPC likelihood ratio only, no energy); this condition is labeled LR in the text.

2) $\alpha = 0.1$ $E_{LO} = 6.0$, $E_{HI} = 20.0$, $E_{OF} = 0.0$ (LR with energy information, smooth offset); this condition is labeled LRSO in the text.

3) $\alpha = 0.1$ $E_{LO} = 6.0$, $E_{HI} = 20.0$, $E_{OF} = 6.0$ (LR with energy information, discontinuous offset); this condition is labeled LRDO in the text.

The first case with $\alpha = 0.0$ is the original likelihood

TABLE I(a)
AVERAGE DISTORTIONS FOR LPC AND ENERGY (E) DISTANCES AS A FUNCTION OF CODEBOOK SIZE AND TYPE OF DISTORTION MEASURE NOTE THE ENERGY DISTANCE SHOWN HAS NOT BEEN SCALED DOWN BY THE WEIGHTING FACTOR α

Distortion Measure	Codebook-Size							
	4		8		16		32	
	LPC	E	LPC	E	LPC	E	LPC	E
LR	.55	—	.41	—	.31	—	.24	—
LRSO	.65	1.3	.50	.53	.37	.30	.28	.20
LRDO	.70	2.47	.55	.79	.42	.27	.32	.09

TABLE I(b)
AVERAGE DIGIT ERROR RATES OF THE PREPROCESSOR AS A FUNCTION OF CODEBOOK SIZE AND DISTORTION MEASURE

Distortion Measure	Codebook-Size			
	4	8	16	32
LR	17.3	11.8	9.9	7.2
LRSO	8.4	6.7	4.7	5.0
LRDO	9.5	6.8	4.8	4.8

ratio distortion measure with no additional energy information. The second and third distortion measures are a linear combination of the likelihood ratio LPC spectral distortion measure and a linearly weighted (α=0.1) energy distance. The LRSO distortion adds a gradually increasing energy distance to the LPC distance, whereas the LRDO distortion has a discontinuous energy distance step at E_{LO} = 6 dB; hence, a heavy penalty is paid when the energy difference between frames exceeds 6 dB.

Using the training set of the REFERENCE database (which comprised 100 utterances for each of the ten digits), different size word-based VQ codebooks consisting of 4, 8, 16, and 32 LPC vectors were designed for each digit. Each of the three distortion measures LR, LRSO, and LRDO were used. A breakdown of the resultant average distortions for different codebook sizes and distortion measures is given in Table I(a). As expected, the average distortions decrease with increasing codebook sizes. It is also interesting to see that, for a given codebook size, the LPC spectral distortions are higher for LRSO and LRDO than for LR. This is a natural consequence of incorporating energy into the distortion measure since, implicitly, some of the codebook vectors are used to encode the energy information. Comparing the breakdown of the LPC spectral distortions and the energy distortions, we also note that by using LRDO as the overall distortion measure, a greater emphasis is placed on energy distortion. Therefore, for larger codebook sizes, LRDO achieves lower energy distortions at the expense of higher spectral distortions than the LRSO. With the codebooks generated through these three different distortion measures, we used the preprocessor as a complete recognizer (i.e., no DTW processing was used) and measured its performance. The test set of the REFERENCE database was used in these tests. The recognition error rates (average fraction times 100) of the preprocessors are shown as a function of codebook size and distortion measures in Table I(b).

The results of Table I(b) show that codebooks designed on the basis of the VQ distortion and energy distortion (LRSO and LRDO) performed significantly better than codebooks designed with LPC spectral distance alone (LR). The performance difference between LRSO and LRDO is not significant for codebook sizes larger than 4. These results strongly indicate that by using energy information, we can improve the performance of the preprocessor. We also see that the performance saturates at codebook size 16 for both the LRSO and LRDO distortion measures. For this digits database, a codebook of size 16 seems to be adequate to characterize the vector space spanned by most of the utterances. Thus, although having more than 16 vectors might allow more outlying frames in the training sequence to be represented in the word-based codebook, in the recognition phase, this tends to reduce both the average distortion of incorrect words and the average distortion of the correct word; hence, no recognition performance improvement is obtained.

B. *Effects of Preprocessor Decision Thresholds on Performance*

We recall that the word-based preprocessor had three decision thresholds, namely,

d_1 = threshold on average word distance for best candidate to be considered as a valid recognition candidate

d_2 = threshold on difference in average word distance between best and second best word candidates for only a single candidate to be chosen by the preprocessor

d_3 = threshold on difference in average word distance between best and any word candidate for passing on the word candidate to the DTW postprocessor.

For convenience, and since it is reasonable, we set d_3 = d_2 in all simulations. We also used the LRSO distortion measure in all experiments to be described in this section.

The threshold values d_1 and $d_2 = d_3$ were systematically varied, and we measured their effects on the system error rate parameters $C1$, $E1$, $(1 - \gamma)$, $E2$, and β. Figs. 6 and 7 show typical plots of the variations of these parameters as a function of d_1 (for the fixed value $d_2 = d_3 = 0.05$) and for codebook sizes varying from 4 to 32. From Fig. 6, we see that for small values of d_1, values of $(1 - \gamma)$ are very close to 1, indicating that very few recognition decisions were made by the preprocessor. As d_1 increases, $(1 - \gamma)$ quickly approaches a steady-state value of about 0.1, indicating that about 90 percent of the words were uniquely recognized by the preprocessor. The resulting steady-state values of $C1$ and $E1$ are about 0.9 and 0.0, indicating that at these thresholds, essentially the preprocessor made final decisions about 90 percent of the time, and all the preprocessor decisions were correct. Similar behavior of $E2$ and β are noted in Fig. 7 for this set of $E2$ and β are noted in Fig. 7 for this set of decision thresh-

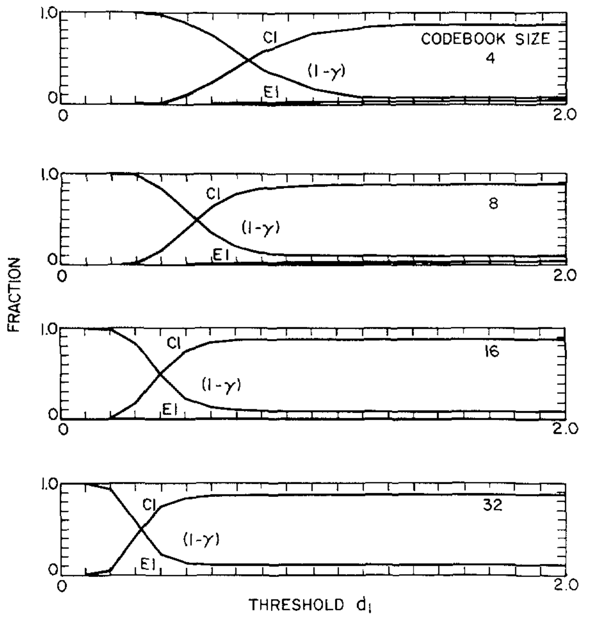

Fig. 6. Plots of the variation of C_1, E_1, and $(1 - \gamma)$ as a function of the preprocessor threshold d_1 for several codebook sizes for the digits vocabulary.

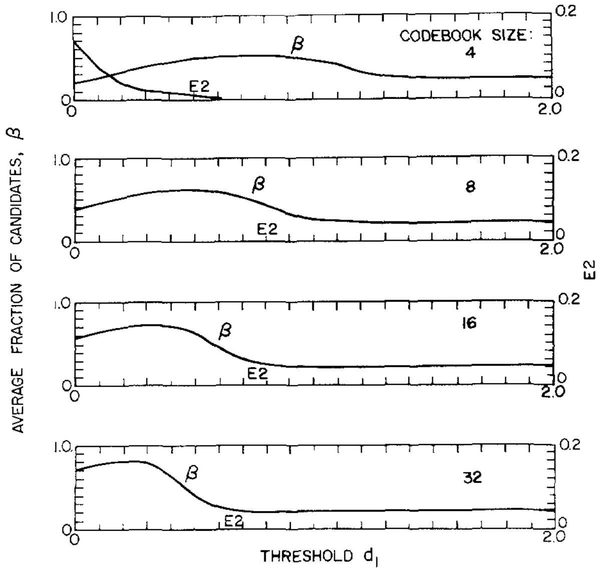

Fig. 7. Plots of the variation of β and E_2 as a function of the preprocessor threshold d_1 for several codebook sizes for the digits vocabulary.

TABLE II
AVERAGE DIGIT RECOGNITION ERROR RATES FOR SEVERAL DISTANCE METRICS AND METHODS OF QUANTIZATION

Distance Variant	Data Base	
	REFERENCE	KLS
LR	2.7	3.6
LRSO	2.1	2.8
LRDO	2.8	3.3

(a)

Quantization in DTW	Data Base		
	REFERENCE	KLS	AIRLINE
NO VQ	2.1	2.8	10.2
VQ/REF	2.8	4.2	14.0
VQ/REF VQ/TEST	3.8	4.0	—

(b)

Codebook Size	Quantization Method	
	VQ/REF, SELF VQ	VQ/REF, ALL VQ
4	4.7	3.2
8	2.3	2.8
16	2.1	1.8
32	1.9	2.1

(c)

olds. The steady-state values of β and $E2$ for large values of d_1 are about 0.22 and 0, indicating that an average of 2.2 words get passed to the DTW postprocessor and the correct word is essentially always one of the words passed on.

From Figs. 6 and 7, we also see that for smaller size codebooks, in order for the preprocessor to make the same percentage of recognition decisions [i.e., fixed values of $(1-\gamma)$], d_1 had to be set to a higher value since, intrinsically, a smaller size codebook yielded larger average distortions.

Based on curves of the type shown in Figs. 6 and 7, it is clear that a good strategy is to choose the smallest value of d_1 for which the parameters $C1$, $E1$, $E2$, β, and $(1 - \gamma)$ are all at steady-state values. We will defer a choice of $d_2 = d_3$ until later in this section since this threshold parameter has the most influence on the performance tradeoffs of interest.

C. Effects of VQ and Energy Metrics on DTW Performance

To provide benchmarks on the performance of the overall recognizer structure of Fig. 2, we ran several experiments using only the DTW processor. The first such experiment used each of the three distance measures, namely, LR (LPC likelihood distance, no energy), LRSO (LPC likelihood distance combined with energy with no offset energy distance), and LRDO (LPC likelihood distance combined with energy with a 6 dB offset). Each of these distance metrics was used in recognition tests using the REFERENCE and the KLS digit sets. Results of these experiments are given in Table II(a), which shows average digit error rates for the three types of distance measures. The results show that the LRSO measure achieved the best performance for both databases. This shows that energy information, when used properly, can be used to improve recognition performance, even for a simple vocabulary such as the digits. The results also show that both the energy discontinuity measure (LRDO) and the measure without energy (LR) performed essentially the same, namely, about 0.5 percent worse than the LRSO measure. On the basis of these results, the LRSO measure was used exclusively in all the remaining DTW processing.

The second experiment considered the effect of using a word-independent VQ on the performance of the DTW word recognizer (i.e., again without the preprocessor). In this case, a VQ with 128 codebook entries was used, and the testing sets included all three databases, namely, the digit sets of the REFERENCE and KLS databases, and the AIRLINE words. The results of this experiment are given in Table II(b), which shows average digit error rates as a function of the type of quantization of the reference patterns in DTW. The results show a substantial deterioration in performance, for all three sets, when the VQ was used on just the reference templates alone. An even greater performance degradation occurred when the VQ was used on *both* the reference and test patterns. Thus, as mentioned earlier, word-independent VQ leads to performance degradation in return for reduced computation. For the AIRLINE vocabulary, it can be seen that a larger performance degradation occurs than for the digits. Sufficient evidence exists that for large vocabularies (e.g., the AIRLINE set), the required size of the VQ codebook must be on the order of 512–1024 entries [7], [8].

The last experiment considered the performance of the DTW processor when the reference patterns were quantized using either the individual word codebooks (SELF VQ) or using all codebooks for the vocabulary (ALL VQ). For this test, only the REFERENCE digits were used. Results of this test are given in Table II(c), which shows average digit error rates (average fraction times 100) as a function of the individual word codebook size. It can be seen that with SELF VQ, codebook size 4 gave degraded performance (there was too much quantization error); however, for codebook sizes 8, 16, and 32, the performance was essentially identical to that of the DTW system *without* VQ! For the ALL VQ case, a small degradation in performance occurred for the small codebooks (sizes 4 and 8); however, for codebook sizes 16 and 32, we again achieved the same performance as the DTW processor without VQ.

D. Summary of Pilot Experiment Results

The results of the pilot experiments show the following.

1) Incorporating energy information into the distance metric improves the recognizer performance.

2) The preprocessor decision thresholds can be set to allow the preprocessor to make a final decision on close to 90 percent ($\gamma = 0.9$) of all trials reliably for the REFERENCE database. When the preprocessor was unable to make a final decision, it eliminated close to 80 percent ($\beta = 0.2$) of all candidates.

3) Vector quantization can be incorporated into the DTW process to significantly reduce computation at a cost of a small increase in recognition error rate.

4) Word-based VQ codebooks perform better than word-independent VQ codebooks, and are comparable to the case when no VQ is used at all in the DTW stage.

V. Performance of the Overall Recognition System

To evaluate the performance of the complete speaker-independent, isolated word recognizer of Fig. 2, three evaluations were performed. The first evaluation used the REFERENCE digits testing database and studied the behavior of the overall system as a function of the codebook size, the preprocessor decision threshold d_2, and as a function of the method for quantization of the reference patterns used in the DTW processing. The second evaluation used the KLS digits database and made similar measurements. The third evaluation used the AIRLINE vocabulary database and, using the best values of d_2 and DTW implementation from the digit runs, measured the resulting error rate and the reduction in computation.

A. Results on the REFERENCE Digits Database

For all experiments to be described in this section, a value of $d_1 = 2.4$ was used. This value guaranteed the minimum preprocessor sensitivity to codebook size, vocabulary, etc. The first tests performed used a DTW implementation with no VQ (Method 1), one with VQ from individual word codebooks (SELF VQ or Method 2), and one with VQ from all word codebooks (ALL VQ or Method 3), and measured the various recognizer error rates ($C1$, $E1$, $E2$, $C2$, $E2$) as a function of L, codebook size, and d_2, the preprocessor decision threshold.

The results of these recognition tests are given in Table III (in which the average fractions are converted to error rates by multiplying by 100) and Fig. 8-11, which show plots of overall error rate (Fig. 8), average fraction of candidates passed to the DTW processor (Fig. 9), average fraction of decisions made by the preprocessor (Fig. 10), and computational complexity (Fig. 11) as a function of d_2 and codebook size. (The results in Figs. 8–11 are for Method 1; similar results were obtained for Methods 2 and 3.)

The results of Table III and Figs. 8–11 show the following.

1) As d_2 increases (especially for codebook sizes 4 and 8), the overall system error rate decreases dramatically, the average number of DTW candidates increases slightly, the percentage of decisions made by the preprocessor falls, and the computational complexity rises. This is because by increasing d_2, the preprocessor is deferring decisions (and hence potential errors) to the DTW processor at an increasing rate. For larger codebook sizes (16 and 32 vectors per word), the error rate is much less sensitive to d_2; hence, for smaller values of d_2, we can achieve very high performance with a very modest amount of computation.

2) At small values of d_2, for each codebook size, the preprocessor made most of the errors; at larger values of d_2, the preprocessor error rate was almost negligible and the DTW processor made most of the errors. For larger codebook sizes (e.g., 16 and 32 vectors), optimum performance was attained with reasonably small values of d_2.

3) The introduction of VQ (via either Methods 2 or 3) led to essentially no increased error rate for all codebook sizes and for all values of d_2. Hence, the proposed quantization schemes are very good ones for use with this system.

To see how the performance of the resulting recognizer

TABLE III
AVERAGE RATES (FRACTION * 100) OF CORRECT AND INCORRECT DECISIONS FOR BOTH THE VQ-BASED PREPROCESSOR AND FOR THE THREE METHODS OF QUANTIZATIONN IN THE DTW PROCESSOR AS A FUNCTION OF CODEBOOK SIZE AND VALUE OF THE DECISION THRESHOLD d_2

					DTW IMPLEMENTATION								
					METHOD 1			METHOD 2			METHOD 3		
Codebook Size	d_2	$C1$ *100	$E1$ *100	$E2$ *100	$C3$ *100	$E3$ *100	Overall Error Rate	$C3$ *100	$E3$ *100	Overall Error Rate	$C3$ *100	$E3$ *100	Overall Error Rate
4	0.05	87.7	4.7	0.3	7.1	0.2	5.2	6.6	0.7	5.7	7.2	0.1	5.1
4	0.075	84.5	4.0	0.5	10.7	0.3	4.8	10.0	1.0	5.5	10.7	0.3	4.8
4	0.1	81.0	2.5	0.8	15.0	0.7	4.0	13.9	1.8	5.1	14.8	0.9	4.2
4	0.125	77.0	1.9	0.6	19.6	0.9	3.4	18.0	2.5	5.0	18.9	1.6	4.1
4	0.15	73.1	1.1	0.8	24.1	0.9	2.8	22.2	2.8	4.7	23.4	1.6	3.5
8	0.05	87.9	3.0	0.9	7.9	0.3	4.2	7.7	0.5	4.4	7.8	0.4	4.3
8	0.075	83.8	2.4	0.7	12.6	0.5	3.6	12.3	0.8	3.9	12.4	0.7	3.8
8	0.1	79.4	1.3	0.8	17.9	0.6	2.7	17.3	1.2	3.3	17.5	1.0	3.1
8	0.125	73.7	0.5	0.9	24.1	0.8	2.2	23.4	1.5	2.9	23.6	1.3	2.6
8	0.15	67.8	0.4	0.3	30.7	0.8	1.5	30.0	1.5	2.2	30.1	1.4	2.1
16	0.05	88.8	1.6	0.7	8.4	0.5	2.8	8.2	0.7	3.0	8.2	0.7	3.0
16	0.075	83.9	1.1	0.3	14.0	0.7	2.1	13.9	0.8	2.2	13.9	0.8	2.2
16	0.1	78.1	0.6	0.3	20.1	0.9	1.8	20.0	1.0	1.9	25.6	1.3	1.8
16	0.125	72.6	0.3	0.2	25.5	1.4	1.9	25.5	1.4	1.9	25.6	1.3	1.8
16	0.15	65.2	0.2	0.2	32.8	1.6	2.0	32.9	1.5	1.9	33.0	1.4	1.8
32	0.05	87.6	1.0	0.1	10.4	0.9	2.0	10.3	1.0	2.1	10.4	0.9	2.0
32	0.075	79.4	0.7	0.1	18.6	1.2	2.0	18.8	1.0	1.8	18.7	1.1	1.9
32	0.1	71.7	0.2	0.2	26.5	1.4	1.8	26.7	1.2	1.6	26.5	1.4	1.8
32	0.125	62.5	0.2	0.0	35.8	1.5	1.7	36.0	1.3	1.5	35.8	1.5	1.7
32	0.15	54.7	0.0	0.0	43.7	1.6	1.6	43.9	1.4	1.4	43.7	1.6	1.6

compares to that of the conventional DTW recognizer (with no preprocessor), and the preprocessor alone, we compared error rate, memory, and computation for these systems. Using a value of $d_2 = 0.075$ and $L = 16$ for the codebook size, we get the following comparisons.

	Overall Error Rate (%)	Memory (k)	Computation (k)
DTW Alone	2.1	44	76.8
Method 1	2.1	45.3	8.5
Method 2	2.2	45.3	6.4
Preprocessor Alone	5.2	—	6.4

The results show a 9-to-1 reduction in computation with *no loss* in accuracy or gain in memory for Method 1 over DTW alone; similarly, we get a 12 -to -1 reduction in computation with no loss in accuracy or gain in memory for Method 2 over DTW alone. When we use the preprocessor alone as a complete recognizer (i.e., by setting $d_1 = \infty$, $d_2 = 0$ so a single candidate is selected each time), the error rate increases by 3.1 percent over DTW alone. Thus, without the temporal information provided by the DTW processor, the performance of the word recognizer is greatly degraded.

B. Results on the KLS Digits Database

The KLS digits database was selected as a somewhat more difficult test of the recognizer since the error rate using the DTW recognizer alone was somewhat higher than for the REFERENCE digits set. Based on varying codebook sizes and values of d_2, the following results were obtained with this database.

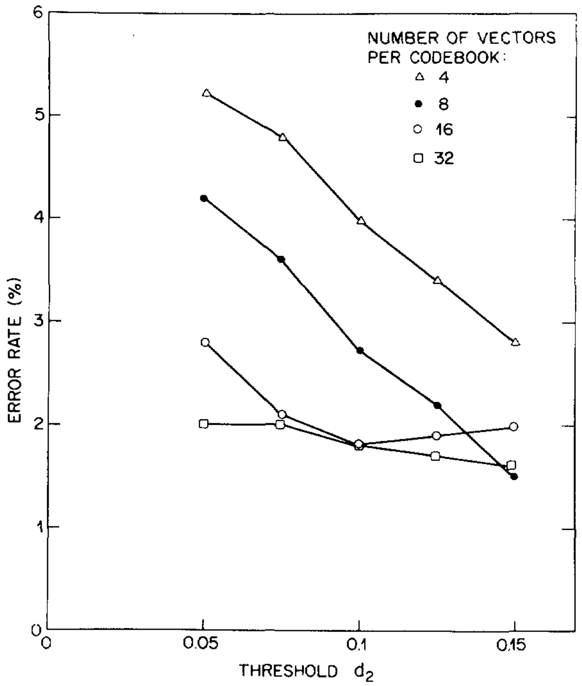

Fig. 8. Plots of total error rate versus preprocessor decision threshold d_2 for several codebook sizes for the digits vocabulary.

Recognition Method	Overall Error Rate (%)	Computation (k)
DTW Alone	2.8	76.8
Method 1 (L = 16)	3.1	18
Method 2 (L = 16)	3.5	8
Preprocessor Alone (L = 32)	10.9	12.8

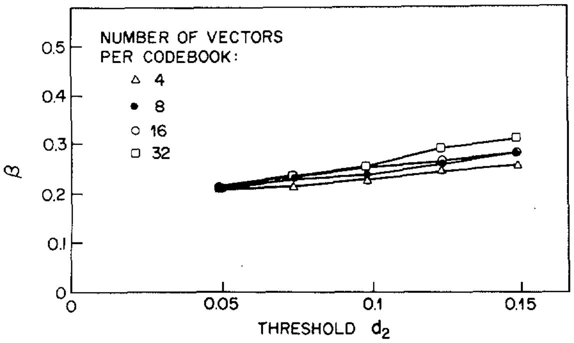

Fig. 9. Plots of average fraction of candidates sent to the postprocessor β versus preprocessor decision threshold d_2 for several codebook sizes for the digits vocabulary.

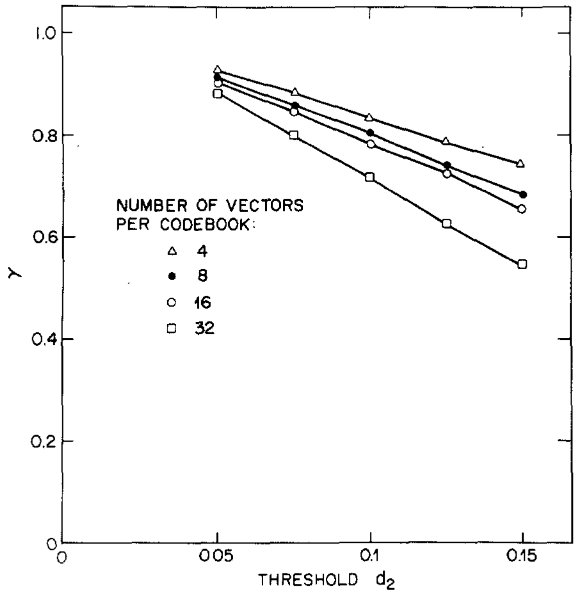

Fig. 10. Plots of average fraction of decisions made by the preprocessor γ versus preprocessor decision threshold d_2 for several codebook sizes for the digits vocabulary.

These results again show that the preprocessor alone gave very bad performance, whereas the overall system of Fig. 2 gave performance that was only slightly degraded from that of the DTW processor alone, with reductions in computation of 4.3 to 1 for Method 1 and 9.6 to 1 for Method 2, as compared to the conventional DTW processor.

C. Results on the AIRLINE Database

The tests using the AIRLINE database were intended to check the extensibility of the overall recognizer to larger, more complex vocabularies than the digits. This vocabulary, with 129 words, is rich in similarly sounding words (e.g., MAY, MAKE, MANY, etc.) and would have a tendency to make the preprocessor less useful as a total recognizer. For this test, there was no experimentation with system variables; a codebook size of 16 vectors was used for each word and only two values of d_2 were considered. A comparison on the best results on the various recognizers for this vocabulary is as follows.

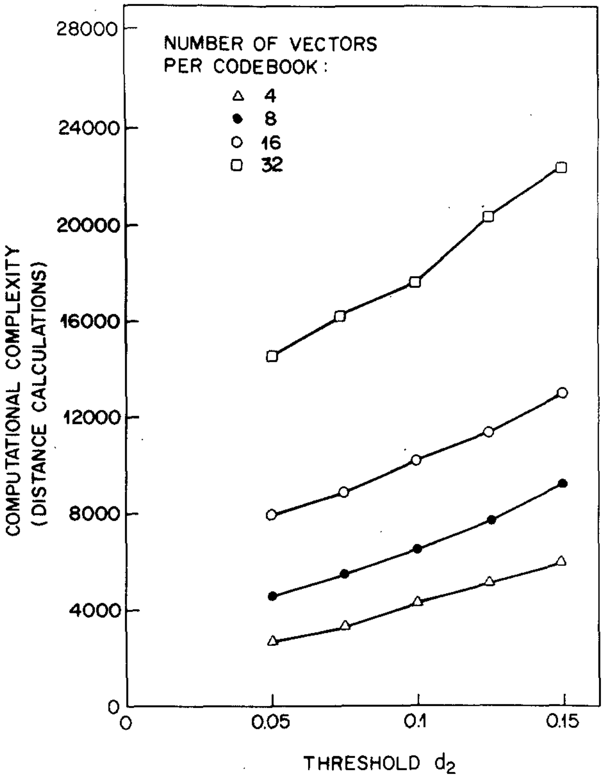

Fig. 11. Plots of computational complexity of the overall recognizer versus preprocessor decision threshold d_2 for several codebook sizes for the digits vocabulary.

Recognition Model	Overall Error Rate (%)	Computation (k)
DTW Alone	10.2	990
Method 2	12.8	107.3
Preprocessor Alone	35.8	82.5

The results given above again show that, when forced to make a unique decision, the preprocessor alone performs very poorly; however, when used in conjunction with the DTW processor, the error rate increases only by 2.6 percent over that of the DTW processor alone with a reduction in computation of close to 9.2 to 1. This tradeoff in accuracy for computation is one which is probably quite acceptable in many proposed applications of this vocabulary, especially ones with tasks which can be used to correct recognition errors via syntax and/or semantics of the proposed application [15].

For the AIRLINE vocabulary, one other comparison was made, namely, we measured the performance of the DTW system alone with a word-independent VQ used on all reference patterns. The VQ codebook that we used had 128 vectors. The resulting error rate of the system was 14.5 percent and the computation was about $\frac{1}{6}$ that required for the DTW system without VQ. Thus, the combined recognizer (Method 2 above) outperformed the VQ based DTW implementation by almost 2 percent in error rate and required about half the computational effort.

VI. Discussion

The results presented in the previous two sections have shown the following.

1) The proposed speaker-independent, isolated word recognition structure consisting of a word-based VQ preprocessor and a DTW postprocessor is indeed a viable structure for recognition. Its performance, measured in terms of word error rate, was comparable to or better than that of a conventional DTW recognizer. However, it required significantly less computation than the DTW recognizer.

2) VQ methods can be incorporated into the DTW processor with little or no loss in performance (word error rate) and with greatly reduced computation.

3) Temporal information is indeed necessary for a high-performance word recognizer, even for a vocabulary as simple as the ten digits.

The philosophy of the proposed recognizer is simple, namely, that since the DTW processor performs so well, it is not acceptable to use any preprocessor which seriously degrades the overall performance of the system. Hence, the job of the preprocessor is to reliably eliminate word candidates which are highly unlikely matches to the spoken word. It is not required, or desirable, for the preprocessor to make fine recognition distinctions; the DTW does this job quite well. The remarkable result is that for simple vocabularies, such as the digits, the preprocessor can make a final decision (i.e., it eliminates all word candidates but one) about 80 percent of the time, and even when it cannot make a unique choice, it usually only passes about two word candidates to the DTW processor. For the more complex vocabulary of airline words, a final decision was made much less often; however, again only about 20 percent of the words were passed on to the DTW processor. Thus, the gain in performance of the proposed system is mainly in computation effort, which is reduced somewhere between 2–20 times, at the expense of little or no increase in word error rate.

Before giving some general comments, it is worthwhile reviewing the major results presented in the previous section on each of the three datasets that were studied.

A. Summary of Results on REFERENCE Digits Set

The results on the REFERENCE digits set showed the following.

1) The best error rate achieved was 1.4 percent (versus 2.1 percent for the DTW processing alone), using Method 2 for VQ of the reference patterns, with an $L = 32$ codebook for each word and with $d_2 = 0.15$. For this system, the values of $(1 - \gamma)$ and β were 0.453 and 0.28, respectively, and a 5.3-to-1 reduction in computation and a 1.5-to-1 reduction in storage (over the conventional DTW processor) was obtained.

2) For maintaining the same error rate as the DTW processor alone (i.e., 2.1 percent), the system variables were $L = 8$ codebooks, Method 2 reference patterns, $d_2 = 0.15$. This implementation gave values of $(1 - \gamma)$ and β of 0.32 and 0.25, respectively, and led to an 18-to-1 reduction in computation and a 4.8-to-1 reduction in memory.

It is interesting to note that for system implementations with error rates lower than that of the DTW processor, the preprocessor actually eliminated some DTW errors by not passing these candidates to the postprocessor.

B. Summary of Results on KLS Digits Set

The results on the KLS digit set showed the following.

1) The DTW stage with either form of VQ (e.g., Methods 2 or 3) had a somewhat higher error rate than the DTW stage without VQ (Method 1).

2) For maintaining the same error rate as the conventional DTW recognizer, the system implementation required $L = 32$ codebooks, $d_2 = 0.2$, with Method 1 DTW processing. This system yielded a 2.8 percent error rate, and required 2.4 times less computation than the DTW implementation. Values of $(1 - \gamma)$ and β of 0.76 and 0.34 resulted from this choice of system variables.

3) If a slight degradation in performance were allowed (i.e., an error rate of 3.4 percent), then Method 2 VQ representations of the reference templates could be used and a 4.5-to-1 reduction in computation and a 1.7-to-1 reduction in memory could be achieved.

When the performance of the overall system is compared to that of a conventional DTW system with a word-independent VQ (with 128 codebook vectors), the performance of the proposed system was always (for all codebook sizes, methods of implementation of the reference quantization, etc.) comparable to or better than the DTW/VQ system, and the computation was less by a factor of two.

C. Summary of Results on AIRLINE Test Set

The results on the AIRLINE test set showed the following.

1) Compared to the standard DTW implementation, the proposed system has a 2.4 percent increase in error rate (from 10.2 percent for DTW to 12.6 percent for the proposed system), but achieved about a 9-to-1 reduction in computation and almost a 3-to-1 reduction in memory.

2) Compared to the DTW/VQ, using a 128-word codebook for the reference patterns, the error rate of the proposed system was 2 percent lower than that of the DTW/VQ system, and the computation was about half the amount.

D. General Comments

The experimental results have indicated that the overall recognizer proposed in this paper is certainly a very attractive method for speaker-independent isolated word recognition. We have seen that the strength of the overall recognizer lies not only in its reduced computational requirement, but also in its ability to achieve a high recognition accuracy. For the digits vocabulary, the overall recognizer was always able to achieve a reduction in computational complexity over the conventional DTW processor and still maintain a comparable (or even lower) error rate. For a recognition system with a large vocabulary, it is clear that some form of preprocessing is essential to eliminate unlikely candidates so that the DTW phase of the recognition system would be used only on reasonably close word candidates. In this manner, with a given amount of computational power, a larger vocabulary can be handled than without such a preprocessor.

There are several areas in which the overall recognizer can still be improved further. In the implementation of the overall recognizer, the computational cost of the preprocessor, although significantly lower than the computational cost of the DTW stage, increases linearly with the number of words in the vocabulary. With V words in the recognition vocabulary and L vectors in each word-based VQ codebook, there are a total of LV codebook vectors in the system. One would expect a certain number of these LV codebook vectors to overlap each other since every codebook was designed separately. In research on vector quantization for speech coding, it has been found that 1024 or 512 LPC vectors are sufficient to encode all speech [11], [12]. Hence, one method to reduce the computational requirement of the overall recognizer, when a large vocabulary is used, is to start with a general word-independent codebook with 512 or 1024 vectors. From this word-independent codebook, 16 or 32 vectors can then be selected to form the word-based codebook for each word. In this way, regardless of the number of words in the vocabulary, the maximum number of codebook vectors in the recognition system is 512 or 1024, as opposed to LV. Although the word-based codebooks generated in this way are expected to be suboptimal, it is not clear if the accuracy of the overall recognizer will degrade.

Alternatively, a slightly different approach to reduce the computational cost of the preprocessor is to generate the optimal word-based VQ codebooks individually and then remove some of the vectors that are very close to each other in the LPC space. In other words, all vectors from the optimal word-based codebooks are reclustered and the centroid of each cluster is used to replace the vectors within that cluster. The total number of codebook vectors can thus be reduced. One problem that might be caused by the above approaches is that the reduced number of codebook vectors in the system may not provide a fine enough spectral resolution for dynamic time warping with vector quantization. A possible solution to this problem is to use a slightly different structure for the preprocessor in which the optimal word-based codebooks for the likely candidates selected by the preprocessor are used to requantize the test word. In this way, the DTW/VQ processor will still have the same fine spectral resolution provided by the optimal word-based codebooks.

Another method to reduce the computational cost in the vector quantizer is to use a different search algorithm. In this research, we have used a full search algorithm in which each test vector is compared to every codebook vector in order to find the best matching codebook vector. A binary search algorithm can be used instead to reduce the computational cost of the preprocessor. The binary search algorithm is suboptimal and has been found to give slightly degraded performance in speech recognition [7]. Shikano [7] has proposed an alternative search method that can lead to almost optimal (full search) performance, but that only requires, on the average, the computational cost of the binary search algorithm. It should be noted that both the binary search algorithm and the Shikano method are more effective for large VQ codebooks and can only be used in the overall recognizer using Method 1 (unquantized test and reference patterns for the DTW algorithm) or using vector quantized patterns for both test and reference.

It should also be pointed out that the savings in computational complexity which we have found in the overall recognizer is an average overall recognition trials for each of the databases used. The amount of computation required to recognize each utterance is not fixed. However, if this is a problem for any application, it can easily be solved by requiring the preprocessor to pass a fixed percentage of candidates to the DTW stage for all test utterances. In this way, the response time of the overall recognizer can be set to a fixed value.

One of the main weaknesses in the preprocessor is the loss of temporal information in the recognition process. It appears that some form of time alignment of the test and reference patterns is essential to eliminate this class of errors. If some temporal information can be brought back, even in very crude forms as suggested by Buzo [18] and investigated by Burton *et al.* [10], the performance of the word-based VQ approach can be significantly improved when it is used either as a complete word recognizer or as a preprocessor of an overall recognizer.

VII. Summary

We have proposed a structure for isolated word recognition in which a word-based VQ preprocessor is used to screen out bad matches to an unknown test word and eliminate them from further consideration. In many cases, especially with a small, noncomplex vocabulary such as the digits, the screening process eliminates all words except one. In such cases, no further processing is needed for recognition. Otherwise, a conventional DTW processor is used to resolve any fine acoustical distinctions between words which were passed by the preprocessor. An evaluation of this new recognition system showed that implementations could be achieved for which the performance degradation (i.e., word error rate) was small or negligible, and for which the reduction in computation was as large as 20 to 1 in some cases. The major weakness in the preprocessor was the lack of temporal information for doing the screening. We are currently considering several alternatives for including such temporal information in the preprocessor.

References

[1] G. M. White and R. B. Neely, "Speech recognition experiments with linear prediction, bandpass filtering, and dynamic programming,"

IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-24, pp. 183-188, Apr. 1976.

[2] F. I. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 67-72, Feb. 1975.

[3] L. R. Rabiner, S. E. Levinson, A. E. Rosenberg, and J. G. Wilpon, "Speaker-independent recognition of isolated words using clustering techniques," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 336-349, Aug. 1979.

[4] G. R. Doddington and T. B. Schalk, "Speech recognition: Turning theory to practice," *IEEE Spectrum*, vol. 18, pp. 26-32, Sept. 1981.

[5] L. R. Rabiner and S. E. Levinson, "Isolated word connected word recognition—Theory and selected applications," *IEEE Trans. Commun.*, vol. COM-29, pp. 621-659, May 1981.

[6] L. R. Rabiner, S. E. Levinson, and M. M. Sondhi, "On the application of vector quantization and hidden Markov models to speaker-independent, isolated word recognition," *Bell Syst. Tech. J.*, vol. 62, pp. 1075-1105, Apr. 1983.

[7] K. Shikano, "Spoken word recognition based upon vector quantization of input speech," *Trans. Committee Speech Res.*, pp. 473-480, Dec. 1982.

[8] L. R. Rabiner, K. C. Pan, and F. K. Soong, "On the performance of isolated word speech recognizers using vector quantization and temporal energy contours," *AT&T Bell Lab. Tech. J.*, vol. 63, pp. 1245-1260, Sept. 1984.

[9] J. E. Shore and D. K. Burton, "Discrete utterance speech recognition without time alignment," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 473-491, July 1983.

[10] D. K. Burton, J. T. Buck, and J. E. Shore, "Parameter selection for isolated word recognition using vector quantization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Mar. 1984, pp. 9.4.1-9.4.4.

[11] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantization," *IEEE Trans. Commun.*, vol. COM-28, pp. 84-95, Jan. 1980.

[12] B. Juang, D. Wong, and A. H. Gray, Jr., "Distortion performance of vector quantization for LPC voice coding," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, pp. 294-303, Apr. 1982.

[13] M. K. Brown and L. R. Rabiner, "On the use of energy in LPC-based recognition of isolated words," *Bell Syst. Tech. J.*, vol. 61, pp. 2971-2987, Dec. 1982.

[14] L. R. Rabiner, M. M. Sondhi, and S. E. Levinson, "Note on the properties of a vector quantizer for LPC coefficients," *Bell Syst. Tech. J.*, vol. 62, pp. 2603-2616, Oct. 1983.

[15] S. E. Levinson, A. E. Rosenberg, and J. L. Flanagan, "Evaluation of a word recognition system using syntax analysis," *Bell Syst. Tech. J.*, vol. 57, pp. 1619-1626, May 1978.

[16] J. G. Wilpon, L. R. Rabiner, and A. Bergh, "Speaker-independent isolated word recognition using a 129-word airline vocabulary," *J. Acoust. Soc. Amer.*, vol. 72, pp. 390-396, Aug. 1982.

[17] A. E. Rosenberg, K. L. Shipley, and D. E. Bock, "A speech data base facility using a computer controlled cassette tape deck," *J. Acoust. Soc. Amer.*, suppl. 1, vol. 72, p. 580, Fall 1982.

[18] A. Buzo, C. Riviera, and H. G. Martinez, "Discrete utterance recognition based upon source coding techniques," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, May 1982, pp. 539-542.

Kuk-Chin Pan (M) was born in Malaysia on September 16, 1961. He received the S.B. and S.M. degrees in electrical engineering and computer science, both from the Massachusetts Institute of Technology, Cambridge, in 1984.

From 1981 to 1984 he participated in a Cooperative Program in Electrical Engineering and Computer Science at AT&T Bell Laboratories, Murray Hill, NJ, where he worked on digital circuit design and vector quantization applied to speech recognition. He is currently with Hewlett-Packard Corporation, working in the area of VLSI design. His interests include speech processing and recognition and VLSI systems design.

Mr. Pan is a member of Tau Beta Pi.

Frank K. Soong (S'76–M'82) was born in Kaohsiung, Taiwan, on August 3, 1951. He received the B.S.E.E. degree from the National Taiwan University in 1973, the M.S.E.E. degree from the University of Rhode Island, Kingston, in 1977 and the Ph.D. degree from Stanford University, Stanford, CA, in 1983.

From 1973 to 1975 he was a Lecturer at the Chinese Naval Engineering School, Tsoying, Taiwan, from 1975 to 1977 he was a Research Assistant at the University of Rhode Island, and from 1977 to 1982 he was a Research and Teaching Assistant at Stanford University. Since 1982 he has been a member of the Technical Staff at AT&T Bell Laboratories, Murray Hill, NJ, in the Acoustics Research Department. His current research interests are in speech coding and speech recognition.

Lawrence R. Rabiner (S'62–M'67–SM'75–F'75) was born in Brooklyn, NY, on September 28, 1943. He received the S.B. and S.M. degrees simultaneously in 1964, and the Ph.D. degree in electrical engineering in 1967, all from the Massachusetts Institute of Technology, Cambridge.

From 1962 through 1964 he participated in the Cooperative Plan in Electrical Engineering at Bell Laboratories, Whippany and Murray Hill, NJ. He worked on digital circuitry, military communications problems, and problems in binaural hearing. Presently he is engaged in research on speech recognition and digital signal processing techniques at AT&T Bell Laboratories, Murray Hill. He is coauthor of the books *Theory and Application of Digital Signal Processing* (Englewood Cliffs, NJ: Prentice-Hall, 1975), *Digital Processing of Speech Signals* (Englewood Cliffs, NJ: Prentice-Hall, 1978), and *Multirate Digital Signal Processing* (Englewood Cliffs, NJ: Prentice-Hall, 1983).

Dr. Rabiner is a member of Eta Kappa Nu, Sigma Xi, Tau Beta Pi, the National Academy of Engineering, and is a Fellow of the Acoustical Society of America.