Connected Digit Recognition Using a Level-Building DTW Algorithm

CORY S. MYERS, STUDENT MEMBER, IEEE, AND LAWRENCE R. RABINER, FELLOW, IEEE

Abstract—In this paper we present a novel method for recognizing a string of connected digits based upon the use of a recently proposed level-building dynamic time warping (DTW) algorithm. The recognition system attempts to build up the string, level-by-level (i.e., digit-by-digit), by comparing portions of the test string to isolated digit reference patterns. A backtracking procedure is used to find the "best" string (i.e., minimum accumulated distance) as well as a set of reasonable alternative candidates. The system was tested on a number of talkers speaking variable length digit strings (from two to five digits) over dialed up telephone lines. String error rates of 4.8 percent and 4.6 percent were obtained for speaker-trained and speaker-independent systems. Word error rates of 0.7 percent (for speaker-trained tests) and 0.9 percent (for speaker-independent tests) were obtained. The digit reference templates were obtained from autocorrelation averaging of a pair of isolated word templates for each digit of the speaker-trained system, and from a clustering analysis of isolated words for the speaker-independent system.

I. Introduction

The area of speech recognition has progressed to the point where a wide variety of isolated word recognition systems have been implemented and used successfully for many applications [1]-[9]. Typically, these applications have been data entry, sorting, and searching systems. For many of these applications in which streams of numbers (or words) are entered into the machine, a connected word format would be a considerable improvement over the isolated word format now in use. Although several heuristic procedures have been proposed for recognizing connected digits [10]-[12], none of these techniques has been sufficiently general to handle arbitrary strings of connected words.

Recently, Sakoe [13] and Rabiner and Schmidt [14] have proposed sophisticated algorithmic approaches to recognizing strings of words based on modified dynamic time warping optimization procedures. Sakoe's approach, called the 2-level DP warp method, exhaustively tries to match all reference words to all possible subsets of the test string (the first level), and then, based on the distance scores, determines the best match (minimum accumulated distance) to the spoken string (the second level). Rabiner and Schmidt use a sampling approach to the matching procedure by attempting to build up candidate strings (from left to right). The only regions of the test string for which dynamic time warping (DTW) matches are tried are those regions at the end of good matches at the preceding level. Thus, only certain "sampling" points are used as potential beginning regions for matching a reference word to the spoken string, and the recognized string is built up word-by-word using a DTW matching procedure.

Although the 2-level DP warping algorithm achieves very high accuracy [13], the computation needed to do the first level match is excessive, and requires special-purpose hardware to approach or achieve a real-time implementation [15]. The sampling approach, on the other hand, requires significantly less computation than the 2-level approach. However, the drawback of this method is the potential loss of information due to sampling the test string at a small number of points. The loss of information results in a small increase in error rate over the exhaustive approach of Sakoe.

In this paper we present another DTW-based connected word recognizer. This new system is based on a level-building DTW algorithm recently proposed by Myers and Rabiner [16]. It has been shown that the level-building algorithm is significantly more efficient than the 2-level DP warp method, but solves the *exact* same problem [16]. The new method has also been shown to be as efficient as the sampling method, and we will show that the level-building approach achieves slightly higher accuracy.

The characteristics of the level building connected digit recognizer are:[1]

1) It operates over dialed-up telephone lines.
2) It accepts variable length digit strings.
3) It can be used as either a speaker-trained, or a speaker-independent system.
4) It uses isolated word templates for the reference templates.
5) It uses a level-building DTW algorithm to recognize the digits within the string.
6) It gives the best digit string for each possible string length that the system can handle.
7) It provides a list of alternative strings of each possible string length.
8) It achieves high digit accuracy (99.1 percent) and high string accuracy (95.4 percent) for both male and female talkers.

The organization of this paper is as follows. In Section II we describe the level-building word recognition system. We also review the level-building DTW algorithm upon which the recognition system is based. In Section III we describe and give

Manuscript received June 5, 1980; revised October 26, 1980. Work performed during M.I.T. Cooperative assignment at Bell Laboratories.
The authors are with Bell Laboratories, Murray Hill, NJ 07974.

[1] Many of the system characteristics are identical to those of the sampling method of Rabiner and Schmidt [14], since this new system is based upon the framework of the previous one.

0096-3518/81/0600-0351$00.75 © 1981 IEEE

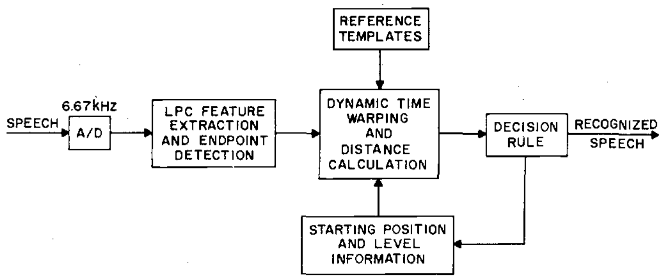

Fig. 1. Block diagram of the connected digit recognition system.

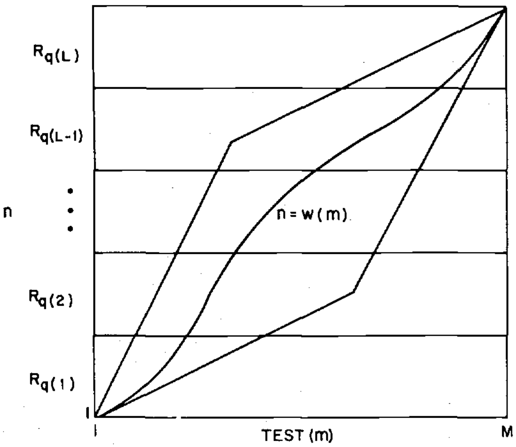

Fig. 2. Illustration of constrained endpoint DTW match of a sequence of reference patterns to a connected word test string.

results of an experimental evaluation of the performance of the recognizer using test strings of connected digits. An important aspect of the performance is the method of creating reference templates, and this problem is also discussed in this section. In Section IV we discuss the results and their implications for connected word recognition systems. Finally, in Section V we summarized our findings.

II. The Level Building Digit Recognition System

Fig. 1 shows a block diagram of the level-building digit recognition system. The front end processing is essentially identical to that used in several isolated word recognition systems—namely, sampling the speech at a 6.67 kHz rate, detecting endpoints of the string, and performing LPC feature extraction [4], [7], [8], [14]. The analysis features are a set of $(p + 1)$ autocorrelation coefficients ($p = 8$) of each $\hat{N} = 300$ sample (45 ms) frame of speech. A shift of $\hat{L} = 100$ samples (15 ms) between frames gives an analysis rate of 67 frames per second. The LPC coefficients are calculated from the autocorrelation coefficients as required in the processing.

Following feature extraction, a level-building DTW algorithm is used to match the test string to a set of stored digit reference templates. At each level (i.e., position in the digit string) information about accumulated distances, best candidate digits, and backtracking pointers are retained, and fed back to the DTW algorithm to begin a new level. A decision rule is used to choose the "best" string—i.e., the string whose accumulated distance at the end of the match is the minimum. The recognition system also provides an alternative list of strings whose distances are close to the minimum distance string. If some form of syntax were available at the output of the system, e.g., the string must form a valid telephone number, etc., then some recognition errors could be detected and corrected using the list of alternative strings. (It is also possible to design a syntax driven level-building algorithm to detect and correct errors [16].)

Although the level-building DTW algorithm has been described in detail elsewhere [16], it is worthwhile reviewing its operation here, since many of the results to be presented here are intimately related to the parameters of the algorithm.

A. The Level-Building DTW Algorithm

Assume we have an L-word connected word sequence which is represented as a test pattern $\boldsymbol{T}(m)$, $m = 1, 2, \cdots, M$ where $\boldsymbol{T}(m)$ is a vector of features for frame m, and M is the number of frames in the test pattern. The goal of the level-building DTW algorithm is to find a sequence of L reference patterns $\boldsymbol{R}_{q(1)}(n), \boldsymbol{R}_{q(2)}(n), \cdots, \boldsymbol{R}_{q(L)}(n)$, where each $q(k)$, $k = 1, 2, \cdots, L$ is one of a set of V reference patterns $\boldsymbol{R}_v$, $v = 1, 2, \cdots, V$, such that the dynamic time warped distance between $\boldsymbol{T}(m)$ and the super reference pattern $\boldsymbol{R}^s = \boldsymbol{R}_{q(1)} \oplus \boldsymbol{R}_{q(2)} \oplus \cdots \oplus \boldsymbol{R}_{q(L)}$ is the minimum over all $q(k)$, where $\oplus$ denotes sequence concatenation. More formally, if we define $D_{q(1)q(2)\cdots q(L)}(M)$ as the accumulated DTW distance between the test pattern $\boldsymbol{T}(m)$ and the super reference pattern $\boldsymbol{R}_{q(1)} \oplus \boldsymbol{R}_{q(2)} \oplus \cdots \oplus \boldsymbol{R}_{q(L)}$, then the goal is to determine the minimum distance D^*, defined as

$$D^* = \min_{q(1)q(2)\cdots q(L)} \left[D_{q(1)q(2)\cdots q(L)}(M)\right]. \tag{1}$$

The indices $q^*(1)\,q^*(2) \cdots q^*(L)$ of the sequence of reference patterns that minimize D^* define the best match to the spoken word string.

The manner in which the level-building algorithm is implemented for solving the minimization of (1) is illustrated in Figs. 2-6. Fig. 2 shows the simple case of obtaining the DTW distance between the test pattern $\boldsymbol{T}(m)$, and a *given* super reference pattern $\boldsymbol{R}^s = \boldsymbol{R}_{q(1)} \oplus \boldsymbol{R}_{q(2)} \oplus \cdots \oplus \boldsymbol{R}_{q(L)}$, i.e., for fixed indices $q(1)\,q(2) \cdots q(L)$. A constrained endpoint DTW algorithm in which the slope of the warping function $w(m)$ is constrained to lie between $\frac{1}{2}$ and 2 is used to find the best path within the parallelogram matching $\boldsymbol{T}$ and $\boldsymbol{R}^s$. This procedure could, in theory, be used to solve (1) by exhaustively testing every possible $\boldsymbol{R}^s$ and doing the minimization directly. However, it should be clear that the amount of computation (V^L comparisons), even for modest values of L, is untractable.

In order to see how we can efficiently solve (1) we must examine, in more detail, the way in which a DTW algorithm is generally implemented for a fixed $\boldsymbol{R}^s$ and $\boldsymbol{T}$. Fig. 3(a) shows a typical implementation of a constrained endpoint DTW algorithm. Generally, the computation to find the optimum warping path is performed in vertical stripes (i.e., m is indexed sequentially and a range on n is found in which the path is constrained to lie) as illustrated in this figure. An alternative way in which the computation could be performed is illustrated in Fig. 3(b). A set of horizontal lines has been drawn for different ending frames of the references within $\boldsymbol{R}^s$. For

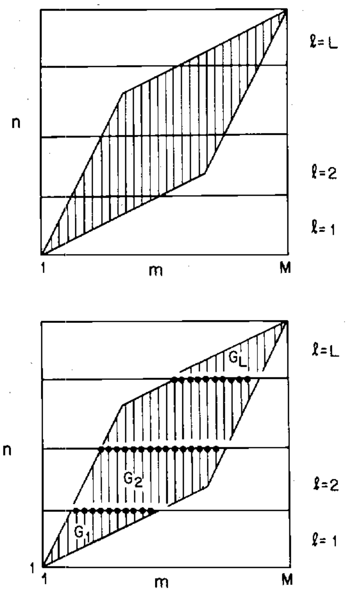

Fig. 3. Two possible implementations of constrained endpoint DTW algorithms.

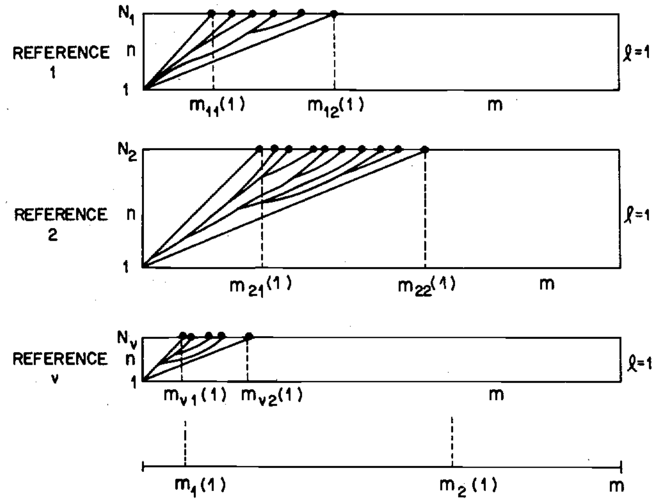

Fig. 4. Sequence illustrating the match regions and the resulting ending regions used in the first level of the level-building DTW algorithm.

this case the computation is done in vertical stripes again; however, the horizontal line formed by the end of each reference forms a constraint on the region in which the computing is done. As such, the computation is again done in vertical stripes until the partial region G_1 of the parallelogram is covered. In order to correctly pick up the computation for the second reference pattern (i.e., in region G_2), the accumulated distance scores for all paths that end at the first horizontal line (denoted by the heavy dots) must be retained and used as initial conditions on distances. In this manner the identical computation, as shown in Fig. 3(a), can be carried out by levels (i.e., words within the sequence of reference patterns) in a series of computations.

The significance of the above results is that the level-building approach to finding the best dynamic path (i.e., finding the best path for each reference pattern in the sequence) can be extended to the case of more than one reference pattern at each level, as illustrated in Figs. 4 and 5. Fig. 4 shows how a set of V reference patterns $\boldsymbol{R}_v$ can be tried at level $l = 1$ to find the best partial matches to a portion of the test pattern. As shown in Fig. 4(a), for reference pattern $\boldsymbol{R}_1$, the algorithm must keep track of the accumulated distance for all paths that end at the grid points (m, N_1), i.e., at the end of reference pattern one. The range of m for which such paths occur is $m_{11}(1) \leqslant m \leqslant m_{12}(1)$, as determined by the intersection of the line $n = N_1$ with the lower and upper warping function constraint lines. Similarly, as shown in Fig. 4(b), for reference pattern $\boldsymbol{R}_2$, the algorithm must keep track of the accumulated distance for all paths that end at the end of reference pattern two, i.e., for $m_{21}(1) \leqslant m \leqslant m_{22}(1)$. This process is repeated for all reference patterns, $\boldsymbol{R}_v$, $v = 1, 2, \cdots, V$, and an overlap range on m is determined as $m_1(l) \leqslant m \leqslant m_2(l)$ where

$$m_1(l) = \min_{1 \leqslant v \leqslant V} [m_{v1}(l)] \tag{2a}$$

$$m_2(l) = \max_{1 \leqslant v \leqslant V} [m_{v2}(l)]. \tag{2b}$$

For each value of m in the ending range $m_1(l) \leqslant m \leqslant m_2(l)$, at level l, we must keep track of three quantities, namely

1) minimum accumulated distance, $\tilde{D}_l^B(m) = \min_{1 \leqslant v \leqslant V} \cdot [\tilde{D}_l^v(m)]$ where $\tilde{D}_l^v(m)$ is the accumulated distance for the vth reference pattern, at level l, ending at frame m of the test pattern;

2) best reference, $W_l(m) = \operatorname{argmin}_{1 \leqslant v \leqslant V} [\tilde{D}_l^v(m)]$ where $\operatorname{argmin}_x [f(x)]$ is the value of x that minimizes the quantity $f(x)$; and

3) backtracking pointer, $\tilde{F}_l^B(m) = \tilde{F}_l^{W_l(m)}(m)$, where $\tilde{F}_l^v(m)$ is the frame of the test pattern at level $l - 1$ at which the best path to test frame m, at level l, for reference pattern $\boldsymbol{R}_v$, ended, i.e., the best path to frame m of the test pattern, at the end of the lth level using reference $\boldsymbol{R}_v$, began at frame $\tilde{F}_l^v(m) + 1$. This pointer basically keeps track of the ending frame of each path at the previous level. For level $l = 1$, it should be clear that $\tilde{F}_1^B(m) = 0$ for all m, since all paths started at frame one of the test pattern.

Fig. 5 illustrates the operation of the level-building algorithm at the second level. The major difference here is that paths can begin at any frame within the starting region $m_1(1) \leqslant m \leqslant m_2(1)$, rather than at a single point, as was the case for the first level. The algorithm keeps track of total accumulated distance for each path and determines a new starting range for each successive level based on the results of the preceding level.

Up to this point we have only shown a set of lower and upper warping function constraints arising from the grid point $m = 1$, $n = 1$. However, a second set of lower and upper warping function constraints come from the upper right-hand corner of the grid, i.e., $m = M$, $n = \phi(L)$ where $\phi(L)$ is the maximum length of the set of L concatenated reference patterns. Hence, when the level approaches the actual length of the sequence, these new constraint lines also determine the region in which the dynamic path can lie. This effect is illustrated in Fig. 6 which shows the ranges and starting regions for each level in a 4-level search. As illustrated previously, the shortest reference template (whose length is shown by the horizontal dashed lines) determines the lower range limit $m_1(l)$, and the

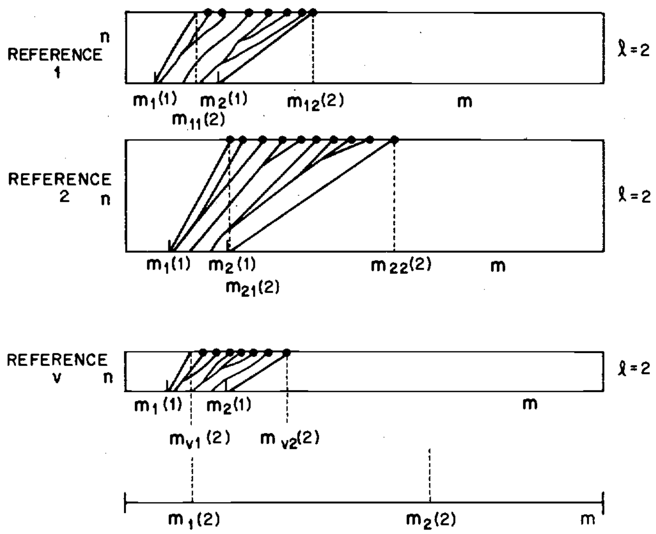

Fig. 5. Sequence illustrating the match regions and the resulting ending regions used in the second level of the level-building DTW algorithm.

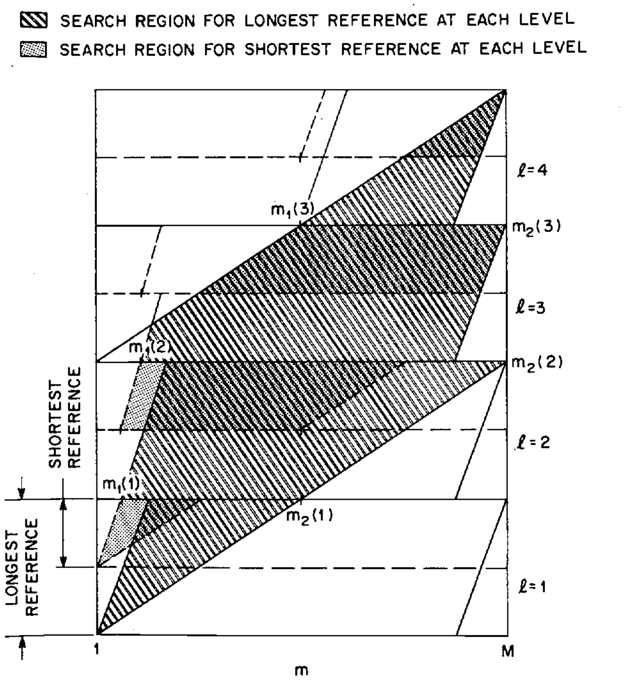

Fig. 6. Illustration of the warping path regions in a 4-level DTW match.

longest reference template (whose length is shown by the solid horizontal lines) determines the upper range limit $m_2(l)$, at level l. When the level l approaches the maximum level L (4 in this example), the path constraint lines (of slope $\frac{1}{2}$ and 2) from the upper right-hand corner of the grid start to control the shape of the region in which the path can lie. This can be seen to occur for $l = 3$ and $l = L = 4$ in Fig. 6. (A general rule which may be used to determine G_l for fixed $\boldsymbol{R}_v$ is to use $m_1(l)$ and $m_2(l)$ to get the constraints from the bottom and to assume levels $l + 1, l + 2, \cdots, L$ will use the longest reference in order to get the constraint from the top).

Fig. 7 illustrates the level-building algorithm for a simple example where it is assumed that there are only two reference patterns, denoted as A and B, each of equal length. It is again

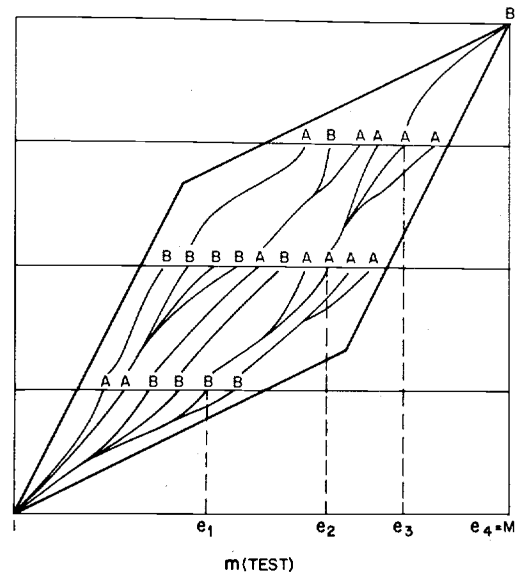

Fig. 7. Illustration of backtracking to recover the best sequence match in a 4-level DTW match.

assumed that a string of length $L = 4$ is known to have been spoken. At the end of the first level, there are six possible ending values of m, and the reference pattern giving the smallest distance is denoted along the horizontal line at the end of the level. Similarly, at levels 2 and 3, best paths to each possible ending frame are noted by the reference, at that level, which gave the minimum accumulated distance. Finally, at level 4, only a single path is retained, as this is the optimum path which minimizes the distance of (1). To determine the best matching string, we must backtrack the path ending at $m = M$ to give the sequence $BAAB$ as the best sequence of four reference patterns matching the test pattern. Also denoted on Fig. 7 are the test frame values e_l corresponding to the end of each reference in the best matching sequence. In principle, these values e_l could be used as best estimates of segmentation points between entries in the test pattern.

It should be noted that the level-building algorithm, as presented above, is capable of determining the best matching string to a test pattern of variable length. As such, the algorithm can generate several "best" matches, each of different lengths as shown in Fig. 8. The overall "best" match is defined as the match giving the smallest distance over all possible sequence lengths. The alternative length strings are useful for applications in which the length is known *a priori*, e.g., telephone number dialing, credit card codes, etc. In Fig. 8 we show the best matches for strings of length $L = 3, 4$, and 5, for the given example.

A second point of note is that, by doubling the storage at each level, we can keep track of both a "best" path and a second-best path, to each frame m of the test pattern. In this manner, alternative estimates of reference strings can be estimated by using second-best paths at any level in the warp. This important point is illustrated in Fig. 9 which shows a "best" path for an $L = 4$ length string, and a series of four alternative paths obtaining by substituting a second-best distance alternative at each level in the warp. These paths are

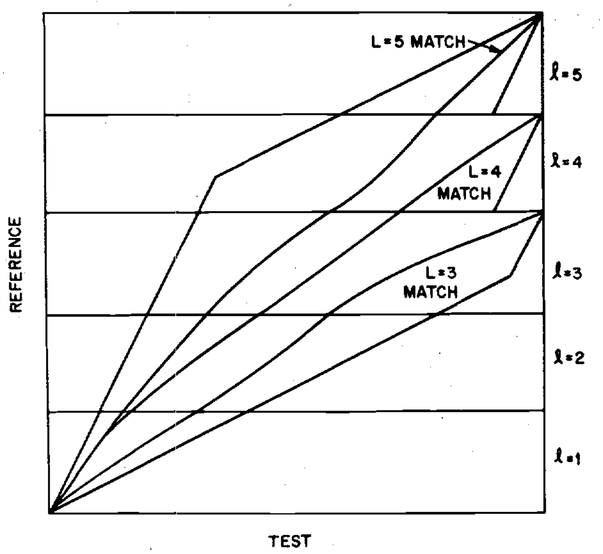

Fig. 8. Example of use of the level-building algorithm with several different length candidates.

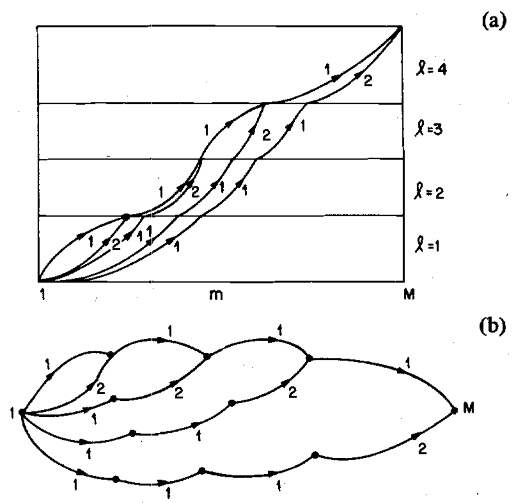

Fig. 9. Example illustrating how additional candidate strings are obtained in the level-building algorithm.

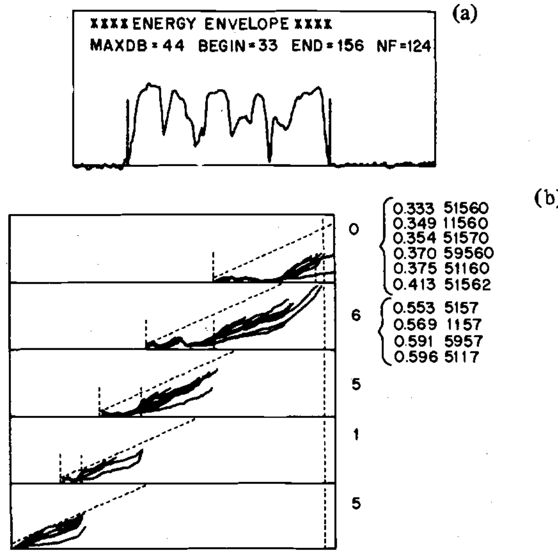

Fig. 10. Log energy contour and accumulated distance at each level of the level building algorithm for the test string 51560.

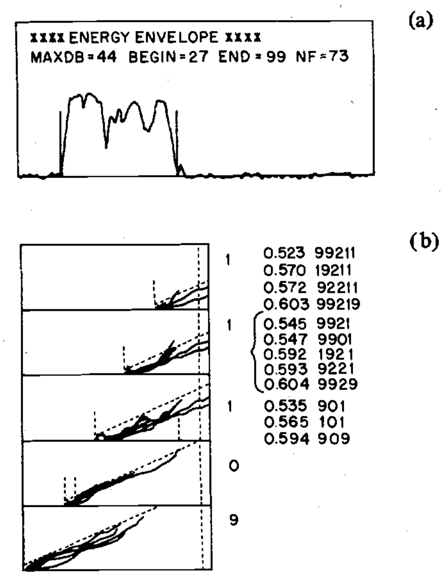

Fig. 11. Log energy contour and accumulated distance at each level of the level-building algorithm for the test string 99211.

shown graphically in Fig. 9(a) and symbolically in Fig. 9(b). If we denote the best path by the sequence of arcs 1111, then the alternative paths are 2111, 1211, 1121, and 1112. However, the arcs labeled 1's occurring before an arc labeled 2 in the alternative paths need not be the same arcs labeled 1 for the best path, since we are now finding a best path to a different ending frame at each level. The set of distances associated with each of these suboptimum paths can be ordered to give an alternative list of strings as estimates for the spoken string.

To illustrate the operation of the level-building algorithm, Figs. 10 and 11 show two examples. In each plot we show the log energy contour of the spoken string [part (a)] and a series of plots of the accumulated distance for each digit at each level. Fig. 10 is for the spoken string 51560. At the end of each level, the program prints out the best local estimate of the digit at each level (shown to the right of each level rectangle). The vertical dashed lines, at each level, denote the initial range of m for which the level allows paths to begin. The sloped dashed line at each level is a distance rejection threshold to eliminate candidates which accumulate large distances. For this example, the best estimate, at each level, of the spoken digit is the actual spoken digit. At the end of the fourth level, the string 5157 matched to the end of the test string with an average distance score of 0.553. Three alternative strings, namely 1157, 5957, and 5117, were also generated at this level by using the second-best distance candidates at each position in the string. At the fifth level the string 51560 (the correct one) was obtained with an average distance score of 0.333. Alternate choices, at this level, included the strings 11560, 51570, 59560, 51160, and 51562 with average distance scores as shown on the figure.

Fig. 11 illustrates a similar set of accumulated distances for the string 99211. In this case, however, the best digit matching at each level is *not* the actual spoken digit. At levels 2 and

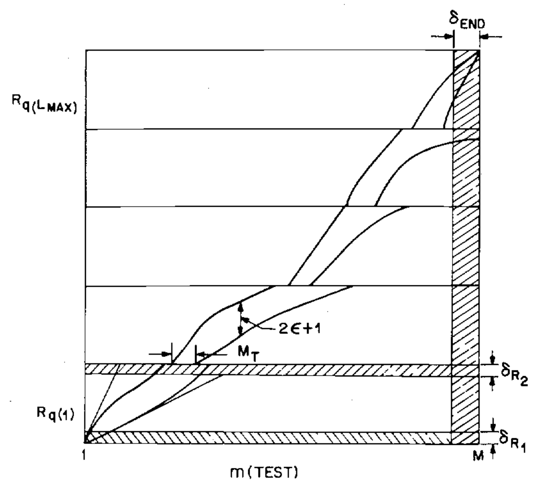

Fig. 12. Illustration of the level-building parameters δ_{R_1}, δ_{R_2}, δ_{END}, M_T, and ϵ.

3, the best digits are 0 and 1 rather than 9 and 2. However, as seen, the correct string is found as the minimum accumulated distance string. Interestingly, for this example, the second-best string is a 3-digit string, namely 901, and the third-best string is a 4-digit string, namely 9921.

B. Increased Flexibility in the Level Building DTW Algorithm

The level-building DTW algorithm, as described in the preceding section, can be easily modified, both to increase its efficiency and to increase its flexibility in handling various types of test input strings. To accomplish these tasks, we have defined a set of variables that can be independently controlled which influence the performance of the level-building algorithm. These variables include:

1) δ_{R_1} = Region of uncertainty at the beginning of the reference pattern

2) δ_{R_2} = Region of uncertainty at the end of the reference pattern

3) δ_{END} = Region of uncertainty at the end of the test pattern

4) M_T = distance multiplier to determine initial m-axis range, at each level, along the test pattern

5) ϵ = Range of DTW local warp along the reference pattern

6) T_{MIN}, T_{MAX} = Threshold parameters on accumulated distance at each level

7) KNN = K-nearest neighbor decision rule criterion.

The effects of some of these variables on the level-building paths are illustrated in Fig. 12. The variables δ_{R_1} and δ_{R_2} define regions, at the beginning and end of each reference pattern, in which the local path can begin or end, i.e., paths need not begin at frame one of each reference and end at the last frame, but instead the best beginning and ending frames, within the specified regions, are found and used for each path.

Similarly, the parameter δ_{END} defines a region at the end of the test pattern in which a total match can end, rather than strictly requiring each path to end at the frame $m = M$. This added flexibility allows for some margin of error in determining the ending frame of the test pattern.

The parameters M_T and ϵ are range reduction parameters which reduce the size of the local regions, G_l, at level l, in which the dynamic path is constrained to lie. The parameter M_T is used to reduce the initial starting range [i.e., from $m_1(l)$ to $m_2(l)$] to the reduced range

$$S_l^1 \leqslant m \leqslant S_l^2 \tag{3}$$

where S_l^1 and S_l^2 are defined as

$$S_l^1 = \text{largest } m \text{ such that } \tilde{D}_{l-1}^B(m) > M_T \cdot \phi_{l-1} \quad \text{for all } m < S_l^1 \tag{4a}$$

$$S_l^2 = \text{smallest } m \text{ such that } \tilde{D}_{l-1}^B(m) > M_T \cdot \phi_{l-1} \quad \text{for all } m > S_l^2 \tag{4b}$$

where

$$\phi_{l-1} = \min_{1 \leqslant m \leqslant M} \left[\frac{\tilde{D}_{l-1}^B(m)}{m}\right] \tag{5}$$

is the minimum average distance at the end of level $l-1$. For practical values of M_T, the range of starting values of m can be reduced by about 50 percent.

Similarly, the parameter ϵ defines a range of width $2\epsilon + 1$ for searching in the n-direction, to find the best path for each reference at each level, as defined by Rabiner *et al.*, in the UELM DTW algorithm [17]. At each frame m, along the test, the range along n is determined by examining a region within $\pm\epsilon$ frames (along n) of the minimum accumulated distance at frame $m - 1$. This parameter is again primarily used to reduce the size of the search region.

The parameters T_{MIN} and T_{MAX} are used to terminate DTW searches on reference patterns which accumulate excessive incremental distance at any level l. Details of the incremental distance test are given in [16].

The last method for increasing flexibility of the level building algorithm, namely the use of the K-nearest neighbor rule (KNN) for speaker independent recognition, is a difficult one to implement. This is because the KNN rule assumes that the distance scores which are being compared (and averaged) all were generated using the same test pattern. For isolated word recognition, this assumption is valid; however, in the level-building algorithm, the best paths at level l and frame m, from two templates representing the same word, can begin at different starting frames. Hence, these two paths use different positions of the test string and cannot be averaged in a KNN rule. Thus, in general, it is not possible to compare distance scores directly with the KNN rule. However, a reasonable heuristic for the KNN rule can be applied; namely, that the distance scores from two reference templates (representing the same vocabulary word) may be averaged in a KNN rule if both templates have warping paths that "come from" the same word. To implement the revised KNN rule, for each token of the vth vocabulary word, we must keep track of the distance accumulated to frame m, and a pointer of the word (at the previous level) from which the best path began. For all frames where KNN (or more) tokens are defined to have come from the same word at the previous level, the KNN rule averages the KNN smallest distances to give the word distance at frame m and level l.

C. Creation of Word Reference Templates for the Level-Building Algorithm

An essential component of the connected word recognition system of Fig. 1 is the set of word reference templates which we compare to portions of the test pattern. There are at least two inherent problems in comparing connected word sequences to concatenated, isolated reference word patterns. First, words spoken in ioslation tend to be substantially longer in duration than the same words embedded in connected strings. Second, words in connected strings coarticulate at the boundaries, thereby producing significantly different spectral behavior than either individual word. To some extent, the level-building DTW algorithm can compensate for durational differences. However, no totally adequate method is known for handling coarticulation effects. The use of variable beginning and ending regions on each reference template (the δ_{R_1}, δ_{R_2} parameters) tends to mitigate these problems somewhat.

In light of the above difficulties, several training methods were investigated for use with the level-building DTW algorithm. For a speaker-trained system, we considered two types of training, namely,

1) casual training in which each digit was spoken in isolation two times, and a reference template was created from *each* replication; and

2) autocorrelation averaging of two casual training replications of each word [18]. In this case the two replications above were averaged to give a single template if their DTW distance was below a threshold; otherwise the shorter (in duration) of the pair of words was used.

For the second training set, we also considered various length normalization techniques including

1) percentage length normalization—all reference templates were linearly expanded or contracted in length by a fixed percentage; and

2) fixed length normalization—all reference templates were linearly expanded or contracted in length to a fixed length.

For each training set and each type of normalization, a set of recognition experiments was performed to measure the performance of the overall connected digit recognition system. Results of these tests are given in Section III.

For the speaker-independent system, the word reference templates were obtained from a set used previously for speaker-independent isolated word recognition [7]. For this set no attempt was made at length normalization.

III. Experimental Evaluation of the Digit Recognizer

In order to test the level-building digit recognizer, the set of recordings described by Rabiner and Schmidt [14] was used. In this set, each of six talkers (three male, three female) spoke 80 randomly generated strings of from two to five digits. An equal number of strings of each length was used, and the number of occurrences of each digit within the string was balanced within each subset of 20 strings of a given length. All recordings were made over dialed-up telephone lines.

Results are presented in the next four subsections, for the following systems.

TABLE I

Speaker-Dependent Recognizer—Two Templates per Word

A. Full Level-Building Algorithm Results:
$\delta_{R_1} = 3, \delta_{R_2} = 4, \delta_{\text{END}} = 4, \epsilon = \infty, M_T = \infty, T_{\text{MIN}} = \infty, T_{\text{MAX}} = \infty$

B. Reduced Level-Building Algorithm Results:
$\delta_{R_1} = 3, \delta_{R_2} = 4, \delta_{\text{END}} = 4, \epsilon = 15, M_T = 1.4, T_{\text{MIN}} = 3.0, T_{\text{MAX}} = 0.7$

Number of Errors (80 Strings Per Talker)

	Variable Length Strings					*Known Length Strings*							
Talker	2	3	4	5	Sum	2	3	4	5	Sum	I	D	S
LR	0	3	1	0	4	0	0	0	0	0	4	0	0
JG	0	1	2	2	5	0	1	0	2	3	2	1	2
SL	0	0	1	1	2	0	0	1	1	2	0	0	2
KS	1	2	2	2	7	1	2	2	2	7	1	0	6
CS	1	0	1	0	2	0	0	0	0	0	2	0	0
SC	0	2	2	1	5	0	1	2	1	4	0	1	4
Total	2	8	9	6	25	1	4	5	6	16	9	2	14

Number of Errors (80 Strings Per Talker)

	Variable Length Strings					*Known Length Strings*							
Talker	2	3	4	5	Sum	2	3	4	5	Sum	I	D	S
LR	0	3	0	0	3	0	1	0	0	1	3	0	0
JG	0	2	1	2	5	0	1	0	2	3	2	1	2
SL	0	0	2	1	3	0	0	1	1	2	0	1	2
KS	1	2	2	2	7	1	2	2	2	7	2	0	5
CS	1	0	1	0	2	0	0	1	0	1	2	0	0
SC	0	3	3	1	7	0	3	3	1	7	0	3	4
Total	2	10	9	6	27	1	7	7	6	21	9	5	13

1) Speaker-trained recognizer—two templates per word obtained from casual recordings.

2) Speaker-trained recognizer—one template per word obtained from a pair of templates.

3) Speaker-independent recognizer—12 templates per word obtained from clustering analysis of isolated digits.

4) Speaker trained recognizer—length normalized test and reference patterns.

A. Speaker-Trained Recognizer—Two Templates per Word

The performance of the level building digit recognizer using 2 speaker-trained templates per word is given in Table I. For the "full" level-building algorithm (i.e., with no computational reductions in parameters), a total of 25 string errors (5.2 percent) were made for variable length strings, and a total of 16 string errors (3.3 percent) were made for known length strings. These results are broken down by talker and string length in Table I(a). The columns labelled I, D, S are the number of strings (in the variable length tests) with insertions (i.e., a string too long was chosen), deletions (i.e., a string too short was chosen), and substitutions (i.e., an incorrect digit was chosen in place of a correct digit). In some cases, more than one substitution occurred in a string; hence, the number of substitutions plus insertions plus deletions must be equal to or greater than the number of string errors. The values of $\delta_{R_1} = 3$, $\delta_{R_2} = 4$ were chosen by searching the $(\delta_{R_1}, \delta_{R_2})$ plane and minimizing the number of string errors. Similarly, $\delta_{\text{END}} = 4$ was chosen to minimize string errors. Values of ϵ,

TABLE II
SPEAKER-DEPENDENT RECOGNIZER—ONE TEMPLATE PER WORD
REDUCED LEVEL-BUILDING ALGORITHM RESULTS:
$\delta_{R_1} = 4$, $\delta_{R_2} = 6$, $\epsilon = 15$, δ_{END} 4, $M_T = 1.4$, $T_{\text{MIN}} = 3.0$, $T_{\text{MAX}} = 0.6$

Number of Errors (80 Strings Per Talker)

	Variable Length Strings					*Known Length Strings*							
Talker	2	3	4	5	Sum	2	3	4	5	Sum	I	D	S
LR	0	3	0	0	3	0	2	0	0	2	3	0	0
JG	0	1	1	1	3	0	0	1	1	2	1	0	2
SL	0	1	0	1	2	0	0	0	1	1	1	0	1
KS	0	2	2	2	6	0	2	2	2	6	2	0	4
CS	2	0	1	0	3	0	0	1	0	1	3	0	0
SC	0	4	2	0	6	0	4	2	0	6	0	2	4
Total	2	11	6	4	23	0	8	6	4	18	10	2	11

M_T, T_{MIN}, and T_{MAX} were chosen to be large enough to represent the level-building algorithm with no range reduction.

The results of Table IA show significantly better performance for strings of length 2, than for strings of length 3, 4, or 5 digits. Also, it is seen that the number of insertions, 9, was quite a bit larger than the number of deletions, 2, indicating a bias in the method for inserting short digits, notably 2 and 8, into longer strings.

The results shown in Table IB are for the "reduced" level-building algorithm with a reasonable operating point, i.e., finite values of ϵ, M_T, and T_{MIN} and T_{MAX}. Based on extensive experimentation, values chosen were $\epsilon = 15$, $M_T = 1.4$, $T_{\text{MIN}} = 3.0$, and $T_{\text{MAX}} = 0.7$. It is seen that a slight increase occurs in the number of string errors (from 25 to 27) for variable length strings. This increase in error rate is compensated by a decrease in computation of about 2 to 1. (We will discuss computational issues later.)

B. Speaker-Trained Recognizer–One Template per Word

As mentioned previously, a second speaker-trained template set was created from the two casual replications of each digit by dynamically time warping the pair of replications to each other, and averaging frames of autocorrelation coefficients (that have been time aligned) if the overall DTW distance was below a preset threshold. Otherwise, the shorter of the two reference tokens was used as the template. This reduced number of templates has been shown to perform as well as the non-reduced template set for isolated word recognition by Rabiner and Wilpon [18].

The results of using the reduced template set in the level-building connected digit recognizer are given in Table II and Figs. 13–15. Table II shows that, for variable length strings, a total of 23 string errors were made (i.e., two *fewer* than for the two template per word set of Table I, and for known-length strings a total of 18 string errors were made (i.e., two *more* than for the two template per word set). The distribution of errors among talkers, and string lengths, and the number of insertions, deletions, and substitutions remained more or less the same as for the earlier results.

Figs. 13–15 show the effects of varying the parameters of the level building DTW algorithm around the "operating

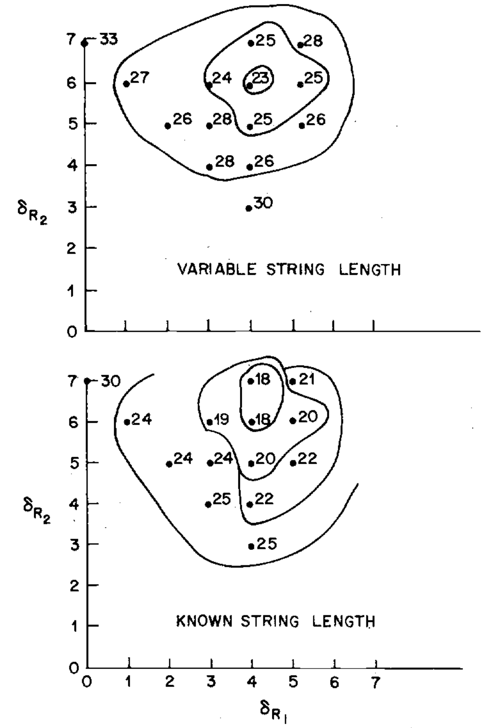

Fig. 13. Contour plots of the number of string errors as a function of δ_{R_1} and δ_{R_2} for the speaker-trained recognizer for both variable- and known-length strings.

point" used to give the results of Table II. Fig. 13 shows the results of varying δ_{R_1} and δ_{R_2} (keeping all other parameter values fixed). A distinct minimum string error rate is obtained for the parameters $\delta_{R_1} = 4$, $\delta_{R_2} = 6$, with small increases in error rate obtained for different values of these parameters. The importance of using the (δ_{R_1}, δ_{R_2}) parameters is seen in this figure, as fairly significant values are used to give the low string error rate. Hence, the ability to eliminate up to ten frames of each reference pattern is an essential factor in the level-building algorithm.

Fig. 14 shows the effects of varying δ_{R_1}, and δ_{R_2} on the number of string insertions, deletions, and substitutions. It can be seen that the number of insertions and deletions is relatively insensitive to values of (δ_{R_1}, δ_{R_2}); however, the number of digit substitutions is greatly dependent on values of (δ_{R_1}, δ_{R_2}), thereby accounting for the values used at the operating point.

Fig. 15 shows the effects of varying ϵ, M_T, and δ_{END} on the number of string errors. (Each parameter is varied independently of all other parameters.) All three parts of this figure show an interesting phenomenon, namely, that a finite optimum value of the parameter exists which minimizes string error rates. For example, $\epsilon = 15$ gives a lower error rate than either $\epsilon = 12$ or $\epsilon = 20$. A value of M_T of 1.4 gives the lowest error rate, while $\delta_{\text{END}} = 4$ gives the best performance.

The results of this section show that a single template per digit in the level-building system is adequate for achieving recognition performance comparable to that obtained from two templates per word for a speaker-trained system. Furthermore, no degradation in recognition accuracy was obtained by using finite values of all parameters of the level-building algorithm; in fact, a small improvement was obtained.

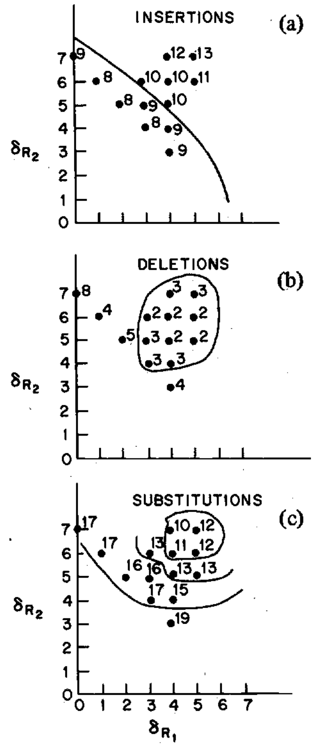

Fig. 14. Contour plots of the number of insertions, deletions, and substitutions, as a function of δ_{R_1} and δ_{R_2} for the speaker-trained recognizer.

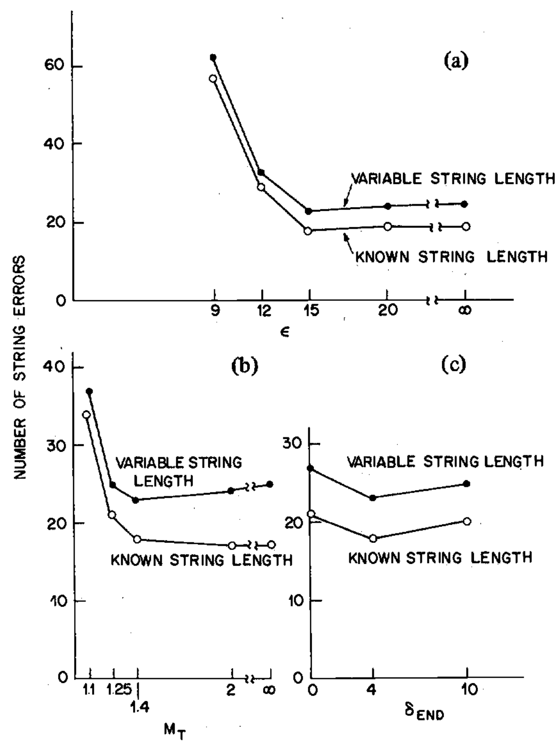

Fig. 15. Plots of the number of string errors versus ϵ, M_T, and δ_{END} for the speaker-trained recognizer.

C. *Speaker-Independent Recognizer—12 Templates Per Word*

The results of the recognition tests using speaker-independent templates with a KNN = 2 rule are given in Table III and Figs. 16-19. As seen in Table III, for variable length strings

TABLE III

SPEAKER-INDEPENDENT RECOGNIZER—12 TEMPLATES PER WORD

REDUCED LEVEL-BUILDING ALGORITHM RESULTS:

$\delta_{R_1} = 0$, $\delta_{R_2} = 3$, $\epsilon = 20$, $\delta_{END} = 4$, $M_T = 1.4$, $T_{MIN} = 5.0$, $T_{MAX} = 0.7$

Number of Errors (80 Strings Per Talker)

	Variable Length Strings					*Known Length Strings*							
Talker	2	3	4	5	Sum	2	3	4	5	Sum	I	D	S
LR	0	1	0	0	1	0	1	0	0	1	1	0	0
JG	2	2	2	1	7	2	1	2	1	6	1	0	7
SL	0	0	2	1	3	0	0	2	1	3	0	0	3
KS	0	1	2	0	3	0	1	2	0	3	1	0	2
CS	1	1	1	2	5	1	0	0	2	3	3	0	2
SC	0	1	1	1	3	0	0	0	1	1	1	1	1
Total	3	6	8	5	22	3	3	6	5	17	7	1	15

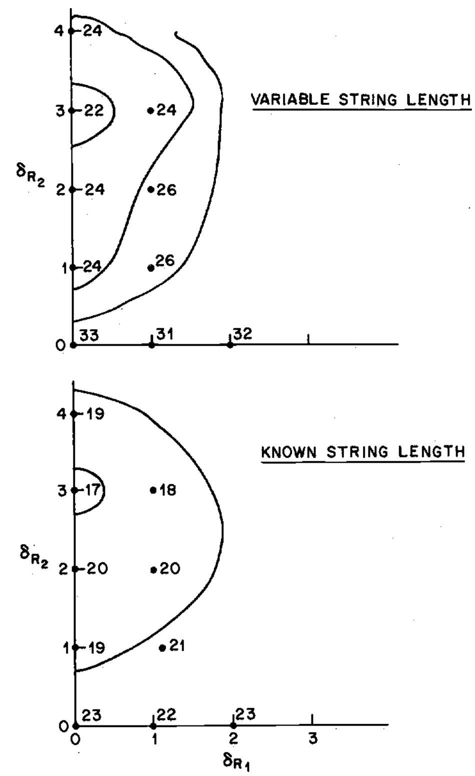

Fig. 16. Contour plots of the number of string errors as a function of δ_{R_1} and δ_{R_2} for the speaker-independent recognizer for variable and known length strings.

the total number of string errors was 22, and for known length strings the number of string errors dropped to 17. Again, the performance was better on shorter strings than on longer strings; however, the differences in performance were smaller than for the speaker trained results of Tables I and II. The operating point, from which the results of Table III were measured, was

$$\delta_{R_1} = 0,\quad \delta_{R_2} = 3,\quad \epsilon = 20,\quad \delta_{END} = 4,\quad M_T = 1.4,$$
$$T_{MIN} = 5.0,\quad T_{MAX} = 0.7.$$

This optimum operating point differed from that of the speaker dependent system primarily in one set of parameters—namely (δ_{R_1}, δ_{R_2}). To illustrate this point further, Figs. 16

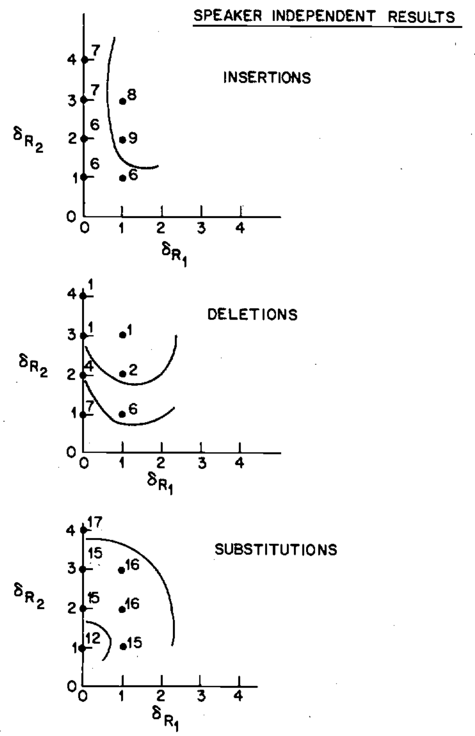

Fig. 17. Contour plots of the number of insertions, deletions, and substitutions, as a function of δ_{R_1} and δ_{R_2} for the speaker-independent recognizer.

and 17 show plots of string error rate (Fig. 16) and the number of insertions, deletions, and substitutions (Fig. 17) as a function of δ_{R_1} and δ_{R_2}. A distinct optimum performance is obtained for $\delta_{R_1} = 0$, $\delta_{R_2} = 3$, with smaller increases in the number of string errors as δ_{R_1} and/or δ_{R_2} were varied away from this point. As seen in Fig. 17, a tradeoff exists between the number of deletions (which favors larger values of δ_{R_2}) and the number of substitutions (which favors smaller values of δ_{R_2}). Again, the number of insertions is relatively insensitive to values of δ_{R_1} and δ_{R_2}.

An important question posed by these results is why we have the discrepancy between the optimum ($\delta_{R_1} = 4$, $\delta_{R_2} = 6$) for the speaker-dependent case (one template per word), and the optimum ($\delta_{R_1} = 0$, $\delta_{R_2} = 3$) for the speaker-independent case. One simple explanation is that the use of the (δ_{R_1}, δ_{R_2}) parameters allows shortening of reference templates to better match the reduced length of words spoken in isolation. For the speaker-dependent case, a fair amount of shortening is required, and can only be achieved in this manner. For the speaker-independent runs, however, there is a large degree of duration variability in the templates themselves (due to the different talkers in the training set); hence, no extra variability is necessary (or desirable) for matching isolated digits to connected digit sequences.

Figs. 18 and 19 show the effects of variation in the parameters δ_{END}, M_T, ϵ, and T_{MIN} and T_{MAX}, on the overall string error rate for the speaker-independent case. As in the speaker-dependent case, finite optimum values of $\delta_{END} = 4$, $M_T = 1.4$, $\epsilon = 20$, $T_{MIN} = 5.0$, $T_{MAX} = 0.7$ are found to yield the lowest error scores. The values of these parameters are essentially the same as those found for the speaker-dependent case, with the exception of T_{MIN}, which had to be raised slightly to a value of 5.0 to account for several strings which matched poorly over the first couple of frames.

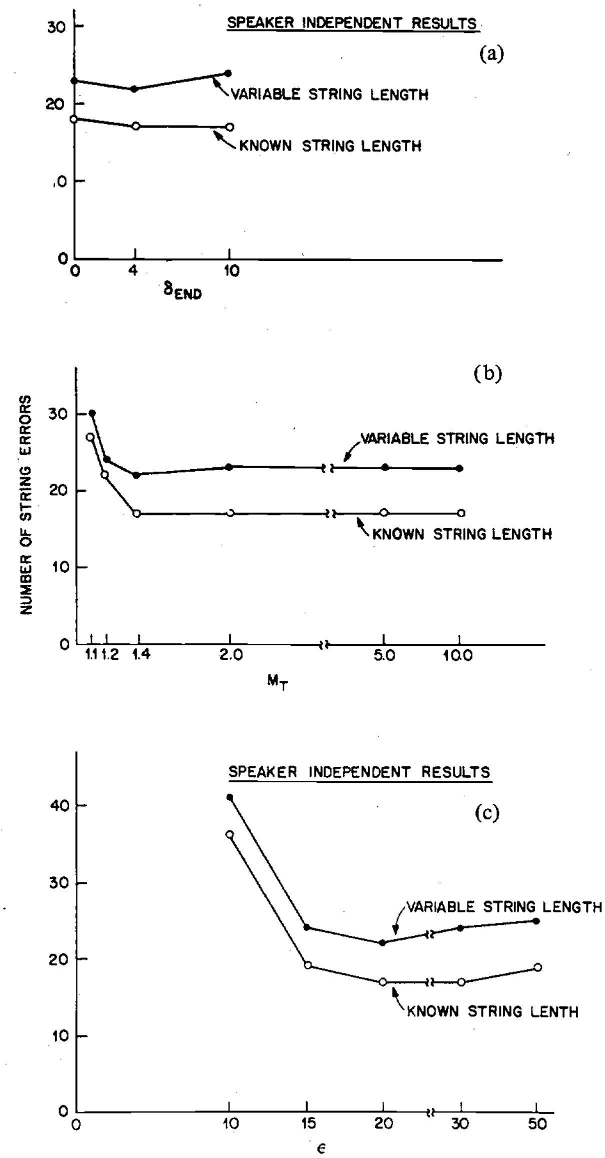

Fig. 18. Plots of the number of string errors versus ϵ, M_T, and δ_{END} for the speaker-independent recognizer.

The results given here show that the performance of the level-building algorithm on a connected digit recognition task is extremely good for both a single-template per word speaker trained system, and a 12 template per word speaker-independent system. The overall performance results are summarized in Table IV which shows string and word error rates (in percent), where a word error is a substitution error, and a breakdown by talker of the string errors.

D. Effects of Normalized Length Templates and Test Utterances

In earlier work on isolated word recognition, it was shown that the best performance was obtained by normalizing the length of the reference and/or test patterns prior to the DTW time alignment [18], [19]. To see whether such ideas were

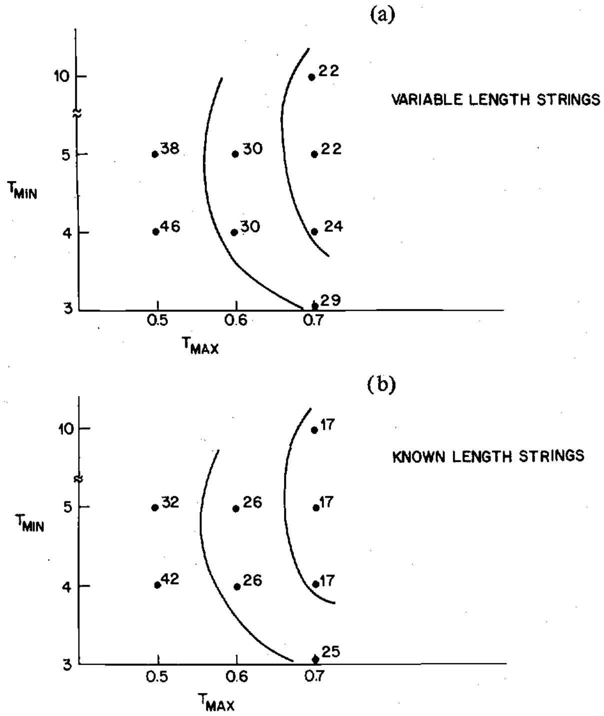

Fig. 19. Contour plots of the number of string errors as a function of T_{MIN} and T_{MAX} for the speaker-independent recognizer for variable and known length strings.

TABLE IV
CONNECTED DIGIT RECOGNITION SCORES
A. SIX TALKERS—THREE MALE, THREE FEMALE; 80 STRINGS PER TALKER; 20 EACH OF LENGTH 2, 3, 4, 5 DIGITS; BALANCED DIGITS WITHIN STRINGS
B. NUMBER OF STRING ERRORS

	STRING ERROR RATE	WORD ERROR RATE	STRING ERROR RATE-KNOWN LENGTH
SPEAKER TRAINED (Single Template Per Word)	4.8%	0.7%	3.8%
SPEAKER INDEPENDENT (12 Templates Per Word)	4.6%	0.9%	3.5%

TALKER	SPEAKER TRAINED	SPEAKER INDEPENDENT
LR (M)	3	1
JG (M)	3	7
SL (M)	2	3
KS (F)	6	3
CS (F)	3	5
SC (F)	6	3

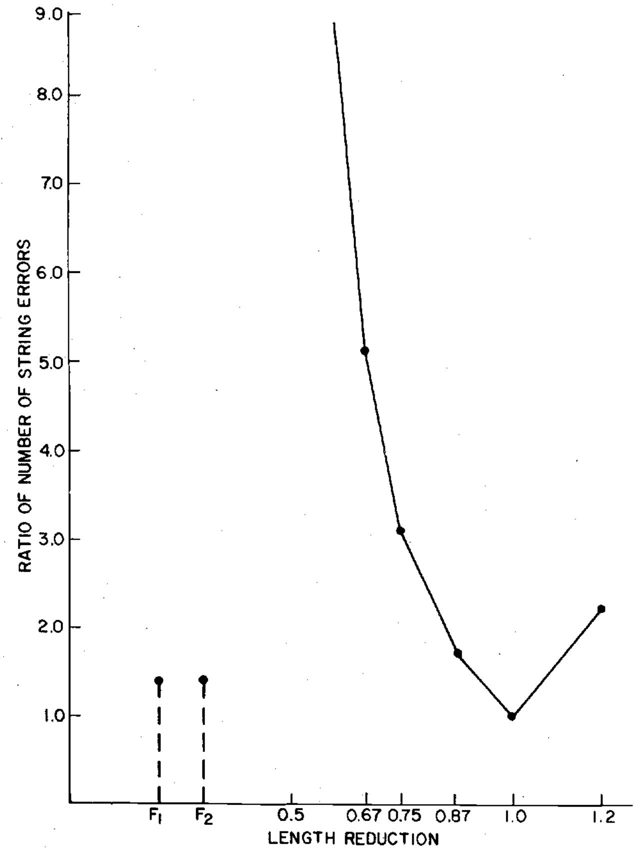

Fig. 20. Plot of the effect of template length reduction on relative string error rate for the speaker-trained recognizer.

of practical utility in a connected digit recognizer, we performed a series of tests whereby the length of the reference patterns (for the 1 template per word speaker-trained system) were linearly warped to either a percentage of their actual length or to a fixed average length. For the normalization to a percentage of their normal length, we considered using the following percentages: 120 percent, 100 percent, 87 percent, 75 percent, 67 percent, and 50 percent. For fixed length normalizations we used both a per talker average length and an overall average length.

Fig. 20 shows the results of the recognition tests with length-adjusted templates. Plotted in this figure is a curve of the ratio of the number of string errors at the given length reduction to the number of string errors with no length reduction. Also shown are two points labeled F_1 and F_2) for fixed duration templates, where F_1 is templates whose lengths are normalized on a per-speaker basis, and F_2 is templates whose lengths are normalized on an overall basis. The results in Fig. 20 show that any form of length normalization of the *templates alone* (i.e., with no normalization of the test pattern) increases the error rate of the system. In fact, the only real improvement in performance was obtained for five-digit sequences with a length reduction of 0.87. All other test sequences showed degraded performance.

Based on the above result, a second length normalization test was made. However, this time both the test and reference patterns were length normalized. (For the test pattern, *a priori* knowledge of the number of digits was required for proper normalization. Hence, the results given here are essentially an overbound on performance). For this case five fewer string errors occurred. This improvement is substantial, and its implication is that length normalization of both reference and test patterns can have a major impact on the performance of the connected digit recognizer.

E. Computational Considerations in the Reduced Level-Building Recognizer

As discussed earlier, the variables of the "reduced" level-building algorithm account for both increased flexibility (i.e., accuracy in recognizing strings) and for decreased computation. The way in which the reduced computation is obtained

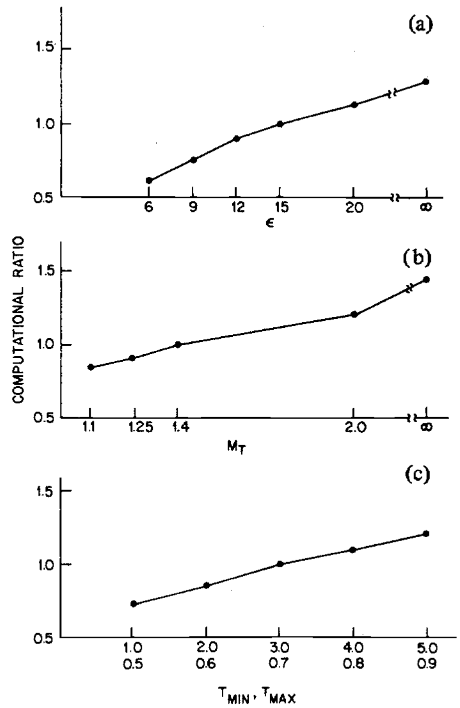

Fig. 21. Plots of relative computation time versus ϵ, M_T, and T_{MIN}, T_{MAX}.

is in the reduction of the size of the region searched to find the best dynamic warping path for each reference template. Fig. 21 illustrates the amounts of computation reduction achieved by using finite values of the parameters ϵ, M_T, and T_{MIN}, T_{MAX}. For each of these plots, the computation at the operating point $\epsilon = 15$, $M_T = 1.4$, and $(T_{MIN}, T_{MAX}) = (3.0, 0.7)$ is normalized to 1.0, and the reduction (or increase) in computation is shown relative to this value. For the ϵ parameter, a reduction of about 30 percent in computation is achieved as compared to $\epsilon = \infty$; for the M_T parameter, a reduction of about 45 percent in computation is achieved; and for the (T_{MIN}, T_{MAX}) set, a reduction of about 22 percent in computation is achieved. By combining all three computational reduction methods, the overall computation at the operating point is reduced by a factor of about 2.1 (i.e., slightly more than a 2-to-1 reduction in computation). As shown earlier, this computational reduction is achieved without increasing the string error rate at all.

IV. Discussion

The results given in the previous section demonstrate the accuracy and flexibility of a connected word recognizer based on a level-building DTW algorithm. Although the results presented are encouraging, they also point out the inherent limitations of recognizing connected words based on isolated word templates. One such limitation is the effect of coarticulation on the results. Clearly, the techniques used to minimize this effect, namely, skipping over frames at the beginning and end of reference patterns, is at best a partial solution to this pervasive problem. More realistically, some form of parameter smoothing at the boundary is necessary to model the physical manifestations of word coarticulation. Another problem with the method concerns the insertion (and deletion) problems associated with the shorter words in the vocabulary. For example, the digit 8 is often inserted in a sequence like 32 by matching the end of the 3 and the beginning of the 2. Perhaps the only reliable solution to these problems is to impose syntactical constraints on the strings so as to be able to detect (and possibly correct) incorrect strings.

Several important observations can be made about the overall string recognizer. First, we have shown the importance of optimizing the parameters of the level-building algorithm in terms of both recognition accuracy and speed of implementation. It was shown that the effects of several of the parameters were identical for both the speaker-trained and the speaker-independent system. However, some of the level building parameters, notably the δ_R's for the reference templates, were optimized differently for the speaker-trained and speaker-independent systems. This result illustrated the inherent variability built into the speaker-independent templates, and the lack of it for the speaker-trained system.

Another important observation concerned the effects of length normalization on the performance of the system. It was shown that, when the length of the test and reference patterns were appropriately normalized, the performance of the system improved a fair amount. Such length normalization can only be applied to the test pattern when the number of words in the string is known. Hence, for the most general systems, this technique is of little value. However, it does point out the importance of matching test and reference lengths in connected word recognition.

Lastly it is worthwhile comparing the performance of the level-building digit recognizer with previous digit recognizers, notably those of Sakoe [13] and Rabiner and Schmidt [14]. For a direct comparison, the work of Rabiner and Schmidt is most relevant, since the identical set of training and testing data was used in both systems. On a string accuracy basis, the level-building algorithm yields an accuracy improvement of about 2 percent for the speaker-trained system, and about 5 percent for the speaker-independent system. These improvements are significant and indicate the gains that are achieved by the level-building approach, since the level-building approach has been shown to be as efficient as the sampling approach [16]. In comparison to Sakoe, the string accuracy is somewhat lower. However, Sakoe used Japanese digits, over a high quality microphone, with a speaker-trained system [13]. The differences in test conditions account for the differences in accuracy scores, since earlier work has shown that the level-building algorithm in its full form is simply a more efficient implementation of the 2-level DP matching algorithm [16], and the increased flexibility of the reduced level-building approach only aids its performance. Thus, the results presented represent the state of the art based on DTW algorithm matching techniques using isolated word reference templates.

V. Summary

We have presented a new approach to connected word recognition based on a level-building DTW algorithm. Experiments in connected digit recognition have shown the resulting recognition system to be accurate and efficient. The inherent flexibility of the method allows the user to optimize values of several parameters of the level-building DTW algorithm to maximize the performance of the overall system. In digit recognition experiments over dialed-up telephone lines, string recognition accuracies of about 95-96 percent were obtained for variable length strings for both speaker-trained and speaker-independent systems.

References

[1] T. B. Martin, "Practical applications of voice input to Machines," *Proc. IEEE*, vol. 64, pp. 487-501, Apr. 1976.

[2] M. B. Herscher and R. B. Cox, "Source data entry using voice input," in *Rec. 1976 IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 190-193, Apr. 1976.

[3] S. L. Moshier, "Talker-independent speech recognition in commercial environments," in *Speech Comm. Papers*, J. J. Wolf and D. H. Klatt, Eds., Paper YY12, June 1979, pp. 551-553.

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

[5] A. E. Rosenberg and F. Itakura, "Evaluation of an automatic word recognition system over dialed-up telephone lines," *J. Acoust. Soc. Amer.* (abstract), vol. 60, suppl. 1, p. S12, Nov. 1976.

[6] S. E. Levinson, L. R. Rabiner, A. E. Rosenberg, and J. G. Wilpon, "Interactive clustering techniques for selecting speaker-independent reference templates for isolated word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 134-141, Apr. 1979.

[7] 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. 236-349, Aug. 1979.

[8] L. R. Rabiner, J. G. Wilpon, and A. E. Rosenberg, "A voice-controlled repertory dialed system," *Bell Syst. Tech. J.*, vol. 59, 1980.

[9] B. Aldefeld, L. R. Rabiner, A. E. Rosenberg, and J. G. Wilpon, "Automated directory listing retrieval system based on isolated word recognition," *Proc. IEEE*, vol. 68, pp. 1364-1379, Nov. 1980.

[10] R. Nakatsu and M. Kohda, "Computer recognition of spoken connected words based on VCV syllable unit" (in Japanese), in *Rep. 1974 Autumn Meeting, Acoust. Soc. Japan*, Oct. 1974.

[11] M. R. Sambur and L. R. Rabiner, "A statistical decision approach to the recognition of connected digits," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 550-558, Dec. 1976.

[12] R. L. Davis, "Application of clustering to the generation of reference patterns for speaker-independent connected digit recognition," Ph.D. dissertation, Univ. Pennsylvania, Philadelphia, 1979.

[13] H. Sakoe, "Two-level DP-matching–A dynamic programming-based pattern-matching algorithm for connected word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 588-595, Dec. 1979.

[14] L. R. Rabiner and C. E. Schmidt, "Application of dynamic time warping to connected digit recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 377-388, Aug. 1980.

[15] S. Tsuruta, "DP-100 voice recognition system achieves high efficiency," *JEE*, pp. 50-54, July 1978.

[16] C. S. Myers and L. R. Rabiner, "A level building dynamic time warping algorithm for connected word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 284-297, Apr. 1981.

[17] L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson, "Considerations in dynamic time warping for discrete word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26, pp. 575-582, Dec. 1978.

[18] L. R. Rabiner and J. G. Wilpon, "A simplified, robust training procedure for speaker-trained, isolated word recognition systems," *J. Acoust. Soc. Amer.*, vol. 68, pp. 1271-1276, Nov. 1980.

[19] C. S. Myers, "A comparative study of several dynamic time warping algorithms for speech recognition," M.S. thesis, Massachusetts Inst. of Technology, Cambridge, May 1980.

Cory S. Myers (S'78), for a photograph and biography, see p. 297 of the April 1981 issue of this TRANSACTIONS.

Lawrence R. Rabiner (S'62-M'67-SM'75-F'76), for a photograph and biography, see p. 297 of the April 1981 issue of this TRANSACTIONS.