Application of Dynamic Time Warping to Connected Digit Recognition

LAWRENCE R. RABINER, FELLOW, IEEE, AND CAROLYN E. SCHMIDT

Abstract—A connected digit recognizer is proposed in which a set of isolated word templates is used as reference patterns and an unconstrained dynamic time warping (DTW) algorithm is used to literally "spot" the digits in the string. Segmentation boundaries between digits are obtained as the termination point of the dynamic path from the previous time warp. A region around the boundary is searched for the optimum starting point for the succeeding digit. At each stage the recognizer keeps track of a set of candidate digit strings for each test string. The string with the smallest accumulated distance is used as the preliminary string estimate. To help improve the recognition accuracy, two "post-correction" techniques were applied to the entire set of hypothesized digit strings. One technique creates a reference string by concatenating reference contours of the digits of the string, and comparing this to the test string using a constrained dynamic time warping algorithm. The second technique performs a similar comparison using voiced-unvoiced-silence contours instead of the measured features. Small but consistent improvements in recognition accuracy have been obtained using these techniques for both speaker-trained and speaker-independent systems with digit strings recorded over dialed-up telephone lines. For variable length digit strings of from 2 to 5 digits (where the recognizer was not told the length of the string), word error rates of about 2-3 percent and string error rates on the order of 8 percent were obtained for both speaker-dependent and speaker-independent systems.

I. INTRODUCTION

RESEARCH in isolated word recognition has advanced to the stage where it is now possible to reliably recognize words from a vocabulary of up to several hundred words and phrases if the system has been trained to the talker [1]-[6]. For speaker-independent recognizers, vocabularies on the order of 50-100 words have been used and reliable recognition has been obtained by using multiple templates per word obtained from a clustering analysis of word tokens by a large set of talkers [7]-[11]. Although isolated word recognizers are suitable for a wide range of applications, for some important vocabularies the requirement for an isolated word format remains a major obstacle. An example of such a vocabulary is the set of digits (i.e., 0 to 9). For applications like all digit dialing of telephone numbers, or data entry for quality testing, a connected digit input format for recognition has the following advantages.

1) It is a more natural input mode for the user. When reading out a string of digits, it is natural to group them into strings of from 2 to 5 digits. For telephone numbers (7 digits), the standard format is a 3 digit string, followed by a 4 digit string.

2) Variable length digit strings are possible. A connected digit recognizer should be able to not only recognize the digits within the string, but also the number of such digits.

Manuscript received November 19, 1979; revised March 3, 1980.

The authors are with the Acoustics Research Department, Bell Laboratories, Murray Hill, NJ 07974.

3) Endpoint problems associated with finding the beginnings and endings of each digit are reduced to finding the beginning and end of each string.

4) The problems associated with artifacts (such as mouth clicks, pops, burbles etc.) at the beginning and end of each digit are essentially eliminated.

Although the connected digit format does have a number of advantages, an equally large number of problems occur in trying to perform the recognition of such strings, including:

1) How to train a connected digit recognizer. Can isolated digits be used as reference templates or must we use digits imbedded in strings? How many such tokens are required for training, and how can such tokens be combined to give a meaningful reference template?

2) How to segment a connected string of digits. Is segmentation performed first, followed by recognition, or are they intimately tied together in a single stage process?

3) How to handle coarticulation between pairs of digits. Do we use some sort of boundary adjustment rule, or modify the recognition procedure to account for such cases? Do we make the recognizer word dependent (i.e., tell it coarticulation rules explicitly), or do we let the recognizer find such cases on its own and handle them in a more algorithmic fashion?

4) How does the recognizer decide on the number of digits in the string?

5) How do we make the recognizer work in a speaker-independent manner and over dialed-up telephone lines?

The advantages of a connected digit format are important enough to justify trying to solve the problems discussed above. The purpose of this paper is to present some partial solutions to some of the problems in designing a recognizer for connected digit strings.

Several previous attempts have been made at designing a connected digit recognizer [12]-[21]. These attempts have been for high-quality recordings and, in general, for speaker trained systems [12], [14], [16], [17]. The work of Nakatsu and Kohda was based on syllable identification for Japanese digits, whereas Sakoe and Chiba, Sakoe, Tsuruta, and Bridle and Brown used word templates for recognition. Sambur and Rabiner had a word template oriented system which explicitly segmented the string based on voiced-unvoiced-silence contours, and then did word recognition on the segments. Davis has recently described a connected digit recognizer for which the digit strings were highly constrained (i.e., only a small percentage of possible digit strings were allowable inputs) to increase the recognition accuracy of the system. Other techniques which have been studied include the use of phonetic templates to segment the string [20], and the use of variable distance metrics to account for the effects of digit coarticulation on word boundaries [15].

0096-3518/80/0800-0377$00.75

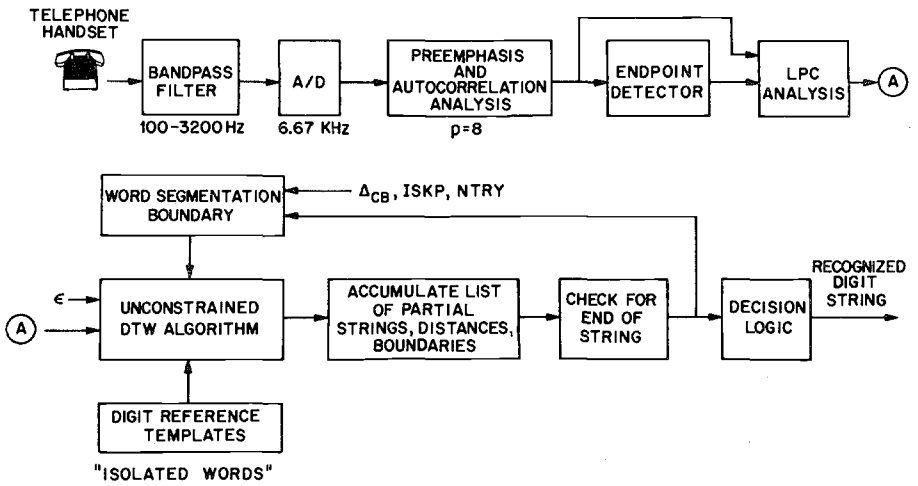


Fig. 1. Block diagram of connected digit recognizer.

In this paper we discuss a connected digit recognizer that has the following features:

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 digits;
5) it uses an unconstrained dynamic time warping algorithm to segment and recognize the digits within the string; and
6) it achieves high digit accuracy (98-99 percent) and high string accuracy (91-95 percent) for both male and female talkers.

In Section II we describe the recognition system. The procedure used to evaluate the recognizer and the recognition accuracy scores are presented in Section III. In Section IV we discuss the use of "post-correction" techniques which provide small, but reliable, improvements in the recognition accuracy.

II. The Connected Digit Recognizer

Fig. 1 shows a block diagram of the connected digit recognition system. The analysis front end is similar to the one originally proposed by Itakura [5], and has been used in a wide variety of recognition experiments [6]-[11]. The input speech, recorded off a conventional dialed-up telephone line, is band-limited from 100 to 3200 Hz, sampled at a 6.67 kHz rate, and a $p = 8$th order autocorrelation analysis is performed on overlapping frames of $N = 300$ samples of speech. The speech is preemphasized (by a simple first-order network) and windowed (by a Hamming window) prior to autocorrelation analysis. The overlap between adjacent frames is $S = 200$ samples, i.e., analysis frames are obtained 67 times per second.

Following autocorrelation analysis, the endpoint detector finds the beginning and ending frames of the connected digit string, based on the log energy of the signal and the background silence statistics (as obtained during the recording interval) [9]. The pointer for the beginning of the string is set 5 frames before the indicated string beginning frame. (The reason for this modification will be explained when we discuss the segmentation procedure.) A linear predictive coding (LPC) analysis is then performed (using a $p = 8$ pole analysis) on each frame of the detected digit string. At this point we denote the test string as

$$T = T_1 T_2 \cdots T_L \tag{1}$$

where the frames $T_j, j = 1$ to L are pth order (normalized autocorrelation) feature vectors that describe the spectral properties of each frame of the digit string. If we denote the qth reference pattern as

$$R^{(q)} = R_1^{(q)} R_2^{(q)} \cdots R_{M(q)}^{(q)} \tag{2}$$

where $R_i^{(q)}$ is the ith feature vector (autocorrelated linear prediction coefficients) of the qth reference pattern (of total length $M(q)$ frames), then the digit recognition problem is one of finding the optimum concatenation of reference strings to provide an optimum match between T and the concatenated string.

We first define the frame distance between test frame j and reference frame i as

$$d(i,j) = \hat{d}(R_i, T_j) = \log (R_i \cdot T_j) \tag{3}$$

where T_j and R_i are $(p+1)$st order feature vectors, and $R_i \cdot T_j$ is a ratio of prediction residuals as defined by Itakura [5]. Following Sakoe [14], we now define the K-word concatenated reference string as

$$R^K = R^{(q(1))} \oplus R^{(q(2))} \oplus \cdots \oplus R^{(q(K))} \tag{4a}$$

$$= R_1^{q(1)} R_2^{q(1)} \cdots R_{M(q(1))}^{q(1)} R_1^{q(2)} \cdots R_{M(q(K))}^{q(K)} \tag{4b}$$

$$= \hat{R}_1 \hat{R}_2 \cdots \hat{R}_P \tag{4c}$$

where P is the total number of reference frames, i.e.,

$$P = \sum_{k=1}^{K} M(q(k)).$$

We can now state the optimum solution to the digit recognition problem as the sequence $q(k)$, $k = 1, 2, \cdots, K$ that minimizes the quantity

$$D(R^K, T) = D(R^{(q(1))} \oplus R^{(q(2))} \oplus \cdots \oplus R^{(q(K))}, T) \tag{6}$$

over all possible K and $q(k)$. Fig. 2(a) provides a pictorial

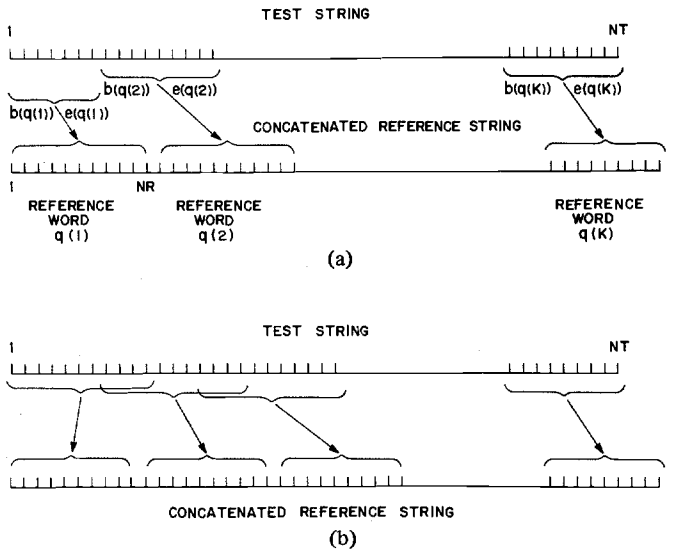


Fig. 2. Illustration of the matching of concatenated reference string to test string. (a) No overlap on test. (b) Overlap on test.

representation of the digit matching process. If we segment T into K regions with beginning and ending points $b(q(k))$, $e(q(k))$, then the distance D of (6) can be decomposed into

$$D(R^K, T) = \sum_{k=1}^{K} \sum_{i=1}^{M(q(k))} \hat{d}(R_i^{q(k)}, T_{w(i)}) \tag{7}$$

where $w(i)$ is the optimum warping (to minimize the distance) between reference frame $R_i^{q(k)}$ and test frame $T_{w(i)}$ as obtained from a dynamic time warping (DTW) algorithm. The beginning and ending functions b and e of the segmented test sequence T trivially obey the formulas

$$b(q(1)) = 1 \tag{8a}$$

$$b(q(k)) = e(q(k-1)) + 1 \tag{8b}$$

$$e(q(K)) = L = NT. \tag{8c}$$

To minimize the quantity of (6) over all k and $q(k)$ requires an inordinate amount of computation, even for modest values of K. For example, for $K = 3$ a total of $10^K = 1000$ digit strings are possible, and for each one a full dynamic time warping must be applied. (Recall that L for a K digit string is on the order of $40 * K$ frames; hence a full dynamic time warping for each string requires about $1600\ K^2/3$ distances.) Furthermore, since K is *a priori* unknown, the computation must be carried out for each value of K that is anticipated (typically $K = 1$ to 5). Clearly an exhaustive solution to the minimization problem is unfeasible.

One alternative has recently been proposed by Sakoe and Chiba [12] and Sakoe [14], and a variation on this procedure has been investigated by Bridle and Brown [21]. This procedure, called two-level dynamic programming matching, first does a word level dynamic time warp matching between digit reference templates and sections of the test string, and then, in a second pass, obtains the best estimate of the string by using dynamic programming to minimize the total distance of a K digit string. This procedure is iterated for each value of K and the minimum over K is chosen as the recognized string.

In order to implement the first phase of this system, namely the word level matching, each reference template had to be matched to the test string once for each possible starting frame in the test string, i.e., a total of about L matches per reference template. Thus, for a total of Q templates per word, and W words in the vocabulary, a total of $W \cdot Q \cdot L$ dynamic time warps had to be computed and stored before the second level of processing could begin. For a speaker-independent system with $Q = 12$, $W = 10$ (digits), and $L = 120$ a total of $120 \cdot 10 \cdot 12 = 14\,400$ dynamic time warps had to be performed for a single string recognition. Although this may be practical for specialized hardware [16], [17], it is still impractical in any reasonable size (and cost) system. Therefore, a more computationally efficient recognition strategy was studied in which the basic philosophy of the two-level algorithm was integrated directly with the word level dynamic time warping procedure.

The algorithm which was chosen to implement the string recognition is illustrated in Fig. 2(b). The recognition of digits in the string is performed sequentially in time; however, the endpoint constraints of (8) are not employed. Instead the kth word is assumed to begin in a region around the end of the $(k-1)$st word, namely

$$e(q(k-1)) - \Delta_{CB} \leqslant b(q(k)) \leqslant e(q(k-1)) + 1 \tag{9}$$

where Δ_{CB} is the number of frames "cut back" from the end of the $(k-1)$st word. The overlap region between words accounts both for digit coarticulation and for differences between isolated versions of a word and its properties when used in a connected format. The determination of K, the number of words in the string, is made from $e(q(k))$, the ending frame of the kth word. K is chosen as the value of k such that

$$e(q(k)) \leqslant L - \epsilon \tag{10}$$

where $\epsilon = 10$ in our simulation, i.e., the last word must end within 10 frames of the end of the input string. Since a region is searched at the beginning of each word, the initial frame of the string was moved back by 5 frames to provide such a region for the first word in the string (and therefore keep the segmentation procedure algorithmic).

The operation of the recognizer is intimately tied to isolated word spotting for each word in the string. We are interested in investigating the region around $b(q(k))$ for the occurrence of word $q(k)$. Since word $q(k)$ can "begin" at any frame in the test, we need to evaluate the function

$$D(R^{q(k)}, T_l) = \frac{1}{M(q(k))} \sum_{i=1}^{M(q(k))} \hat{d}(R_i^{q(k)}, T_{w(i)}) \tag{11}$$

for $l = l_1,\ l_1 + ISKP,\ l_1 + 2ISKP, \cdots,\ l = l_1 + (NTRY - 1) * ISKP$, where $ISKP$ is the frame shift between estimated word beginnings, $NTRY$ is the number of tries at a beginning point, and T_l denotes the starting frame of the test utterance. From (9) the reader can see that $l_1 = e(q(k-1)) - \Delta_{CB}$ and $l_1 + (NTRY - 1) * ISKP = e(q(k-1)) + 1$. Hence the sequence of $NTRY$ dynamic time warps bracket the presumed starting region of the kth word. Therefore, a range of $NTRY * ISKP$ frames is used to find the best starting frame for the kth digit

TABLE I
TYPICAL ACCUMULATION OF PARTIAL DIGIT STRINGS

k	Partial String	Beginning Frame	Ending Frame	Accumulated Average Distance/Digit
1	5	6	35	0.44
	9	6	24	0.48
2	90	20	72	0.74
	50	30	72	0.80
3	907	70	107	1.02
	507	70	107	1.08
	906	68	109	1.25
	506	68	109	1.31

in the string. However, the dynamic time warping procedure itself provides a range of starting frames for the path. (This algorithm will be reviewed in Section II-A.) Hence a fairly wide range of test frames is searched to find the best estimate of the kth digit. It should be noted that for "word-spotting" applications values of $ISKP = 1$ are used, i.e., every single starting frame is searched to see if the word is present in the string. Here we are using a reduced sampling rate to try to spot digits in the string.

The recognition algorithm of Fig. 1 [illustrated in Fig. 2(b)] proceeds from left to right. However, at each stage of the recognition, a list of partial strings, current boundaries, and distances is accumulated, as illustrated in Table I. For this example there were 2 candidates for the first digit ($k = 1$), namely, 5 and 9. Both candidates began at frame 6 of the test, but digit 5 matched until frame 35 whereas digit 9 matched until frame 24. The average distance [from (11)] for digit 5 was 0.44, whereas the average distance for digit 9 was 0.48, i.e., slightly larger. Since both digits were candidates for the first digit in the string, they were both retained as possible candidate strings. For the second digit ($k = 2$), two sets of beginning frames were tried and, in both cases, the digit 0 best matched the input string. However, a significantly better match was obtained in the vicinity of frame 24 than in the vicinity of frame 35; hence the most likely candidate string at the $k = 2$ position is the string 90 rather than 50. At this point the endpoint $e(2)$ for both strings is frame 72; hence there is no longer any possibility that partial string 50 will yield a smaller distance than partial string 90. At the third position ($k = 3$) both partial strings find the digit 7 as the best match, with the digit 6 a somewhat poorer alternative. Hence an ordered list of 4 strings is obtained at this stage. Since all 4 partial strings end within a small number of frames of the digit string length, the recognition is terminated and the string 907 is chosen as the most likely candidate. However, the ordered list of final candidates is retained for final testing using post correction techniques to be described later.

The above example illustrates a number of points about the recognizer. These include the following.

1) Although a true two-level dynamic warp is not used to make the final decision about the digits in the string, the algorithm used is reminiscent of the dynamic warping procedure and has the capability of choosing a partial string at stage k that had a higher distance at stage $(k - 1)$ than other partial strings.

2) The overlap between the end of the $(k - 1)$st digit and the beginning of the (kth) digit is an important feature of the recognizer in that invariably a better digit match is obtained in the overlap region than would be obtained by tip-to-tip matching as required by other recognizers.

3) Although there is a tendency for the number of partial strings to increase with k (hence an increased computational load), it has been found that, in most cases, only a small number (often 1) of partial strings are retained.

In summary, the basic steps of the recognizer are:

1) Use an unconstrained dynamic time warping procedure to provide the best match to the first digit position. At the conclusion of the warping retain the partial string (or strings), beginning and ending frames in the digit string, and the accumulated distance scores.

2) Retain all partial strings (up to some maximum) that are either below a preset threshold, or are within a preset distance of each other.

3) For each partial k digit string, obtain an updated $(k + 1)$ digit string (or strings) whose accumulated distance is minimum by using an unconstrained dynamic time warping procedure to provide the best match in the vicinity of the ending frame of the kth digit.

4) Check if the $(k + 1)$st digit ends at or near the end of the test string. If not, repeat steps 2 and 3; if so, the algorithm is finished.

In the next section we give more detail on the dynamic time warping algorithm, and the procedure used to implement multiple starting points with the warp.

A. The Unconstrained Dynamic Time Warping Procedure

The heart of the recognition procedure is the unconstrained dynamic time warping algorithm which is used to spot the digits in the string. The algorithm that was used here was the unconstrained endpoints, local minimum (UELM) procedure as described by Rabiner, Rosenberg, and Levinson [22]. Because of its importance to the recognition system, we review the UELM algorithm. We assume that the reference frames are along the x axis (the independent axis), and the test string frames are along the y axis. The test is assumed to be much longer than the reference.

For a general dynamic time warping algorithm, the basic recursion rule is

$$D_A(i,j) = d(i,j) + \min_{q \leq j} [D_A(i-1,q)] \tag{12}$$

where $D_A(i,j)$ is the accumulated distance to grid point (i,j), and $d(i,j)$ is the distance from from i of the reference to frame j of the test. If we define the warping path between j and i as

$$j = w(i) \tag{13}$$

and we assume a local continuity constraint on $w(i)$ as

$$\begin{aligned} w(i) - w(i-1) &= 0, 1, 2 \quad \text{if} \quad w(i-1) \neq w(i-2) \\ &= 1, 2 \quad \text{if} \quad w(i-1) = w(i-2), \end{aligned} \tag{14}$$

then (12) becomes

$$\begin{aligned} D_A(i,j) = d(i,j) + \min [&D_A(i-1,j)g(i-1,j), \\ &D_A(i-1,j-1), D_A(i-1,j-2)] \end{aligned} \tag{15}$$

where

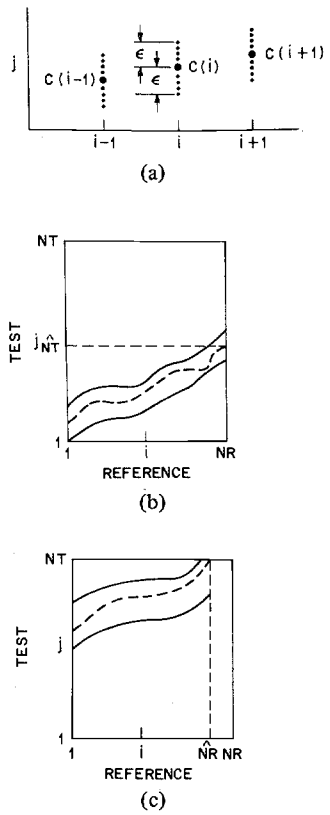


Fig. 3. (a) Definition of the range parameter ϵ for the unconstrained dynamic time warping algorithm. (b) Illustration of the normal stopping procedure for the DTW algorithm. (c) Illustrations of a possible stopping procedure at the end of the test string.

$$g(i,j) = 1 \quad \text{if} \quad w(i) \neq w(i-1)$$
$$= \infty \quad \text{if} \quad w(i) = w(i-1). \tag{16}$$

The ultimate minimization of accumulated distance yields the value

$$D_T = \min_j [D_A(M,j)] \tag{17a}$$

$$= \min_{w(i)} \sum_{i=1}^{M} d(i, w(i)) \ . \tag{17b}$$

Equations (12)-(17) describe the basic recursion and the dynamics of a dynamic time warping procedure. The feature that distinguished the UELM algorithm is the global path constraints. We define a center point $C(i)$ and a range $R(i)$ as shown in Fig. 3(a). The range of values j which are searched for the ith test frame is defined for the UELM algorithm as

$$\max [C(i) - \epsilon, 1] \leqslant R(i) \leqslant \min [C(i) + \epsilon, L], \tag{18}$$

i.e., a range of up to $\pm\epsilon$ frames around the center frame is used at the ith frame in the recursion. The center for the $(i+1)$st frame is determined from the set of accumulated distances at the ith frame as

$$C(i+1) = \underset{j \in R(i)}{\text{argmin}} [D_A(i,j)] \tag{19a}$$

with initial region

$$C(1) = 1 \tag{19b}$$

where argmin means the value of j that minimizes $D_A(i,j)$ over the range $R(i)$. Because of local continuity constraints in the path, the range of (18) is modified slightly to

$$\max [C(i) - \epsilon, 1, R_{\min}(i-1)] \leqslant R(i)$$
$$\leqslant \min [C(i) + \epsilon, L, R_{\max}(i-1) + 2] \tag{20}$$

where $R_{\min}(i)$ and $R_{\max}(i)$ are the values in the equation

$$R_{\min}(i) \leqslant R(i) \leqslant R_{\max}(i). \tag{21}$$

The constraints of (20) guarantee that a path in which no possible link can be made is not searched.

Fig. 3(b) and (c) illustrates the use of the UELM algorithm in two typical cases. The algorithm proceeds from left to right and it terminates when the reference terminates [Fig. 39b)], or when the local minimum of $D_A(i,j)$ occurs at the end of the test string [Fig. 3(c)]. For each application of the UELM algorithm we must retain the accumulated total distance D_T, the frame number of the reference at which the path terminated (usually this is M), and the frame number of the test at which the path terminated. For cases when the warp corresponds to the correct reference, the terminating test frame is the initial test frame for the next digit search.

As mentioned earlier, to account for digit coarticulation, an overlap region between the end of the kth digit (as determined from the UELM algorithm) and the beginning of the $(k+1)$st digit is searched to find the optimum starting frame for the $(k+1)$st digit. Ostensibly, this search is carried out by iterating the UELM at each of several starting frames. We assume that a total of *NTRY* starting frames are used, each separated by *ISKP* frames. For this example the UELM algorithm is run *NTRY* times, with starting frames separated by *ISKP* frames. For the case where *ISKP* is small compared to the UELM range variable (i.e., *ISKP* = 3, $\epsilon = 8$) ϵ, it was found that all *NTRY* runs could be combined into a single dynamic time warp run, with total running time only about 50 percent larger (for *NTRY* = 4) than for a single UELM run. As such, a computationally efficient algorithm was achieved.

III. Experimental Evaluation

To test the digit recognizer a series of recognition tests were run. First a preliminary evaluation was run on 2 talkers, each speaking a randomized list of 10 strings of 3 digits each. For this preliminary run, parameters of the UELM algorithm were systematically varied (both for speaker-trained and for speaker-independent runs) and their effects on the recognition accuracy were investigated. Following this preliminary phase, a new series of recordings was made by 6 talkers (3 male, 3 female) who each spoke 80 strings of from 2 to 5 digits each. In this section we describe the results on both the preliminary and the test recordings.

A. Preliminary Investigation of UELM Parameters

A set of 10 strings of 3 digits each was used in the preliminary tests. The set of strings was recorded off a standard, dialed-up telephone line. For each of the 2 talkers, speaker-dependent reference templates for the digits were obtained by having the talkers say each digit twice in a random, isolated

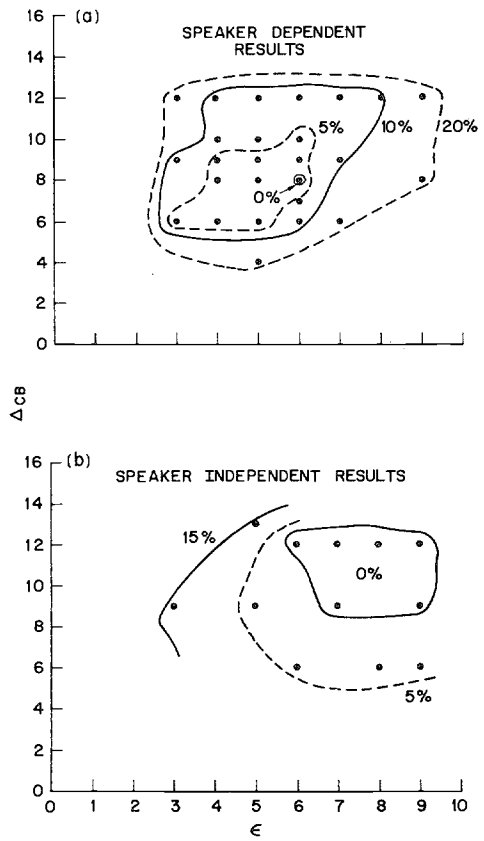


Fig. 4. Contour plots of recognition accuracy as a function of ϵ and Δ_{CB} for training sets of data for (a) speaker-dependent cases, and (b) speaker-independent cases.

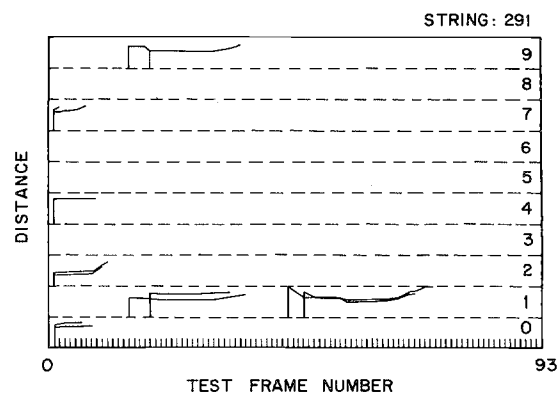


Fig. 5. Plot of DTW distance versus test frame number for the digit string 206 for each possible digit.

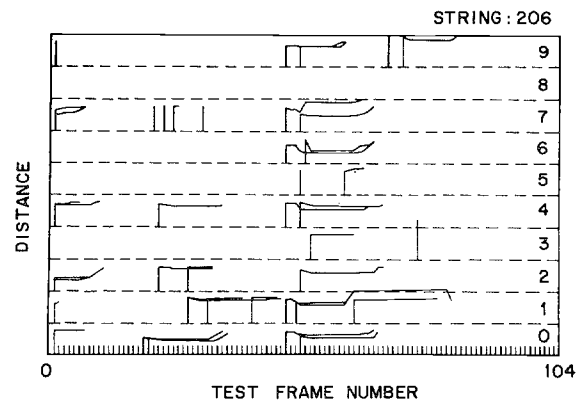


Fig. 6. Plot of DTW distance versus test frame number for the digit string 206 for each possible digit.

sequence, and forming a reference template from each recording. Speaker-independent templates were obtained from a clustering analysis of a 100 talker population [10].

The most significant parameters of the UELM algorithm are the range parameters ϵ, and the cutback parameter Δ_{CB}. Fig. 4 shows plots of contours of equal string error rate as a function of ϵ and Δ_{CB} for the recognition system of Section II for speaker-dependent reference templates [Fig. 4(a)] and speaker-independent reference templates [Fig. 4(b)]. (Values of $ISKP = 3$ and $NTRY = 4$ were used in this and in many subsequent runs.) It can be seen from these figures that for a fairly large portion of the ϵ, Δ_{CB} plane, the string error rate remains 10 percent or less for both speaker-trained and speaker-independent templates. However, it is seen that for this specific run, the regions of the (ϵ, Δ_{CB}) plane in which the minimum error occurs are fairly different. Thus the preliminary run only provided approximate regions of the (ϵ, Δ_{CB}) plane to test the recognizer for speaker-trained and speaker-independent digit templates. For the speaker-trained case, it was anticipated that the optimum region in the plane could vary as more speakers were included in the test. However, for the speaker-independent case since the templates didn't vary, it was anticipated that the chosen region would be stable.

Figs. 5 and 6 illustrate, for 2 strings in the preliminary evaluation, why the recognition system can be successful with left to right, tip-to-tail, digit recognition. These plots show the results of evaluating the distance between the test string and each reference template (for the speaker-trained case) for the case of reinitiating the warp every single frame of the test. Thus we are evaluating (11) for the special case where $l_1 = 1$ (begin at the first test frame), $ISKP = 1$ (do the warp every single frame), and $NTRY = L$ (include every possible starting frame). Fig. 5 shows distance scores plotted as a function of digit reference and frame number for the string 291; similarly, Fig. 6 shows results for the string 206. (All distances greater than a threshold are not plotted.) It is readily seen in Fig. 5 that for a wide range of starting frames (i.e., from frame 1 to 9) the distance for the reference digit 2 is small and no other reference digit is even close in distance (only 0, 4, and 7 even had distances below the threshold). For the second digit in the string, we again see that the reference digit 9 had the smallest distance over a broad range of starting frames with the reference digit 1 a close second candidate. Finally for the third digit, the only candidate was the digit 1. Thus the recognizer would provide as output the candidate strings 291 (correct answer) with the smallest distance, and 211 as an alternative, slightly high distance, string.

For the example of Fig. 6, the behavior of the distance scores is similar to that of Fig. 5; however, it is more complicated in its details. For the first digit in the string, the only valid reference is the digit 2, which has a small distance score over a fairly wide range of starting points. For the second digit in the string, only the digit 0 has a reasonably low distance score (although 1, 2, and 4 all have distances below the thresh-

TABLE II
DIGIT STRINGS USED IN THE TEST EVALUATION

No.	*String*	*No.*	*String*	*No.*	*String*	*No.*	*String*
1	3673	21	23	41	837	61	19481
2	34	22	77	42	3534	62	8155
3	83278	23	20	43	8858	63	222
4	712	24	1051	44	75	64	4900
5	910	25	2181	45	19095	65	70
6	452	26	386	46	77555	66	4972
7	589	27	744	47	206	67	71722
8	51655	28	539	48	55	68	2771
9	545	29	401	49	30	69	7927
10	18	30	96	50	76228	70	88
11	62004	31	510	51	768	71	7070
12	19	32	52084	52	98	72	67476
13	42	33	63	53	71459	73	6216
14	6603	34	5842	54	3548	74	2299
15	91	35	84007	55	899	75	9099
16	4630	36	63630	56	686	76	11230
17	41	37	24254	57	976	77	93439
18	4453	38	343	58	100	78	83966
19	25	39	6618	59	81193	79	9986
20	40	40	713	60	66	80	43830

old). For the third digit in the string 8 of the 10 digits have scores below threshold; however, the digit 6 (the correct choice) has the lowest score by a small amount. Thus, if we knew that there were 3 digits in the string we would most likely have estimated the correct string. However, if there might have been 4 digits in the string, the possibility exists of identifying a fourth digit since both reference digits 1 and 9 have distance scores below threshold for frames well beyond the beginning of the third digit. In this case, since the distance scores of the candidates for the fourth digit were large, the recognizer would choose only the sequences with 3 digits (based on average distance scores). However, in many cases we will see that a major problem with digit string recognition in which the number of digits in the string is *a priori* unknown is that digit insertions (i.e., addition of an extra digit into the string) and digit deletions (i.e., skipping over a valid digit in the string) occur and cannot readily be detected or corrected.

B. Recognition Tests on Digit Strings

In order to test the recognition system, a randomized set of 80 strings of digits was used. The number of digits in the strings varied from 2 to 5 (20 strings each), and the number of times each digit appeared in each length string was uniform, but randomly generated. Table II gives the randomized list of strings that was used. Each of 6 talkers (3 male, 3 female) spoke the strings over a dialed-up telephone line (a new line for each talker) and recorded the strings directly into the computer using a high-speed array processor (the CSP MAP-200) for real-time analysis and endpoint detection. As in the preliminary experiment, speaker-dependent reference templates were obtained for each talker by having them recite each of the digits 2 times in an isolated word format. The speaker-independent templates were again the clustered digits set as discussed previously.

The results of the recognition tests are given in Tables III and IV, and in Fig. 7. Table III gives results for the speaker-dependent case for 2 sets of recognition parameters, and Table IV gives results for the speaker-independent case for 2 sets of recognition parameters. The data that are tabulated include the following:

TABLE III

Talker	Number of String Errors	Number of Unfinished Strings	Number of Digit Errors	Number of Insertions	Number of Deletions
CS	2	1	2	2	0
LR	11	0	5	1	6
KS	9	0	7	1	3
SC	3	0	1	0	3
SL	9	0	4	0	7
JG	13	1	6	1	7
Totals	47	2	25	5	26
Percent Error	9.8	0.4	1.5		

(a) Speaker Dependent Recognition Results for $\epsilon = 6$, $\Delta_{CB} = 8$, $ISKP = 3$, $NTRY = 4$

Talker	Number of String Errors	Number of Unfinished Strings	Number of Digit Errors	Number of Insertions	Number of Deletions
CS	4	2	3	2	0
LR	5	0	3	1	2
KS	5	0	5	0	1
SC	3	0	2	0	2
SL	10	1	3	3	8
JG	11	0	6	2	5
Totals	38	3	22	8	18
Percent Error	7.9	0.6	1.3		

(b) Speaker Dependent Recognition Results for $\epsilon = 8$, $\Delta_{CB} = 12$, $ISKP = 3$, $NTRY = 4$

1) The number of string errors. A string error occurs when the best recognition candidate (with the lowest average distance) does not exactly match the input string. String errors can occur because of digit errors, digit insertions, or digit deletions. For each talker a total of 80 strings were used; hence for each subject the number of string errors is relative to 80, and for the total count it is relative to 480.

2) The number of unfinished strings. A string was unfinished whenever one of two conditions was met. Either a digit insertion occurred in a 5 digit string causing the number of digits to exceed 5, or no reference digit was able to match the string features in the neighborhood of the ending point of the previous digit in the string. In either case the recognition was terminated with the partial match and no attempt was made to clear up the problem. (Unfinished strings were not counted as string errors; in fact in most cases the correct string was found.)

3) The number of digit errors. A digit error occurred whenever the wrong digit was recognized in place of the correct digit, at any place in the string where a digit truly occurred.

4) The number of insertions. A digit insertion was defined as the occurrence of an extraneous digit between two well-defined digits in the string. This situation occurs in several well-defined sequences, e.g., the introduction of a spurious 2 in the string 81 after the initial 8.

5) The number of deletions. A digit deletion was defined as the total absence of a digit which was actually spoken in the string with no subsequent replacement of that digit by another digit.

For the data of Table III (for the speaker-trained system), it is seen that the results for the best set of recognition param-

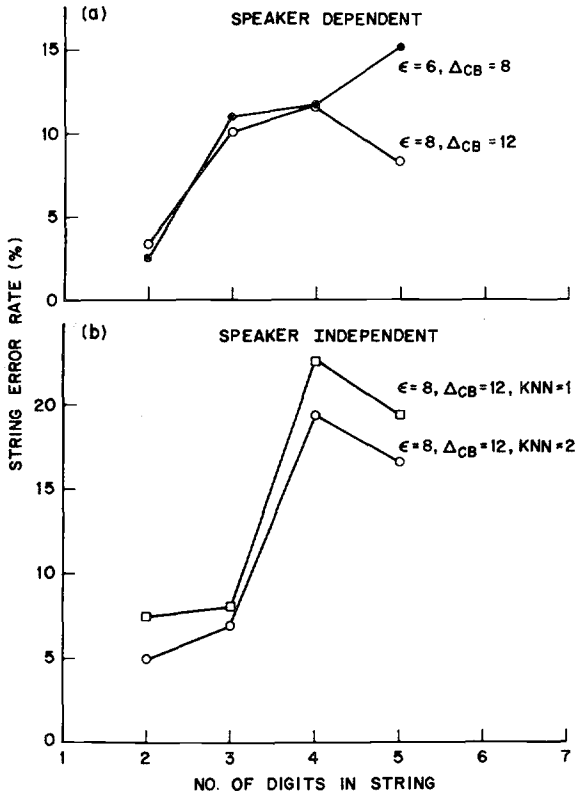


Fig. 7. Plots of average string error rate for (a) speaker-dependent results, and (b) speaker-independent results.

TABLE IV

Talker	Number of String Errors	Number of Unfinished Strings	Number of Digit Errors	Number of Insertions	Number of Deletions
CS	8	2	4	8	1
LR	12	0	9	1	3
KS	2	0	2	0	1
SC	11	0	13	0	0
SL	12	0	7	0	7
JG	9	0	10	2	0
Totals	54	2	45	11	12
Percent Error	11.3	0.4	2.7		

(a) Speaker Independent Recognition Results for $\epsilon = 8$, $\Delta_{CB} = 12$, $ISKP = 3$, $NTRY = 4$, $KNN = 2$

Talker	Number of String Errors	Number of Unfinished Strings	Number of Digit Errors	Number of Insertions	Number of Deletions
CS	10	2	6	9	0
LR	10	1	12	2	0
KS	8	2	3	4	4
SC	12	1	9	5	1
SL	17	0	10	0	8
JG	6	0	6	1	2
Totals	63	6	46	21	15
Percent Error	13.1	1.3	2.7		

(b) Speaker Independent Recognition Results for $\epsilon = 8$, $\Delta_{CB} = 12$, $ISKP = 3$, $NTRY = 4$, $KNN = 1$

eters from the preliminary run [Table III(a)] were somewhat worse than the results for the recognition parameters set to values obtained for speaker-independent recognition [Table III(b)]. A string error rate of 7.9 percent with a digit error rate of 1.3 percent was obtained for the speaker-dependent system. A total of 8 digit insertions and 18 digit deletions occurred for these tests.

For the speaker-independent recognizer the error rates were somewhat higher than for the speaker-trained recognizer, as seen in Table IV. For these cases the K-nearest neighbor recognition rule [9] was exploited to improve recognition accuracy by using $KNN = 2$ (average of 2 best templates) instead of $KNN = 1$ (minimum distance rule). For this case the average string error rate was 11.3 percent, with a digit error rate of 2.7 percent, and with 11 insertions and 12 deletions occurring in the tests.

Fig. 7 shows plots of the string error rate as a function of the number of digits in the input string for the test cases of Tables III and IV. Fig. 7(a) is for the speaker-dependent recognizer, whereas Fig. 7(b) is for the speaker-independent case. It can be seen that for the 2 digit sequences, the string error rate is 2.5 percent for the speaker-trained system and 5 percent for the speaker-independent system. For 3 digit sequences there is a sharp increase in error rate to 10 percent for the speaker-trained system; however, for the speaker-independent recognizer the error rate rises only to about 6.5 percent. For 4 and 5 digit strings, there are only small changes in error rate for the speaker-trained system; however, a very sharp increase in error rate occurs for the speaker-independent system. As the number of digits in the string increases, the average time per digit tends to decrease, i.e., the talkers speak more rapidly. The results of Fig. 7 indicate that the speaker-independent recognizer can handle speaking rates corresponding to 3 digit strings or less with reasonably low error rates; beyond this point a sharp breakdown in accuracy occurs. On the other hand, the speaker-dependent recognizer tends to degrade more gracefully as the number of digits in the string increases. In fact for 5 digit strings the average error rate was smaller than for 3 digit strings.

One final point worth noting about the errors in recognition concerns the digit errors, the digit insertions, and digit deletions that occur. For the speaker-dependent system, the digit errors occurred uniformly across all digits; however, the digit insertions occurred primarily for the digit 8, and the digit deletions occurred primarily for the digits 2 and 8. For the speaker-independent case about half the digit errors occurred for the digit 2. The vast majority of the digit insertions and deletions were for the digits 2 and 8. It was anticipated that the digits 2 and 8 would experience the worst recognition problems because they are short digits which are heavily coarticulated and which can readily be deleted or inserted in connected strings. It was also expected that the problems with the digits 2 and 8 would be more severe for the speaker-independent system since the variability across the 12 templates was much larger than the variability across 2 speaker specific templates.

IV. Discussion

As discussed earlier, the philosophy of the digit recognizer is a left-to-right, tip-to-tail, digit recognition from isolated

reference word templates. The tip boundary for the kth word was assumed to be in the neighborhood of the tail boundary of the $(k-1)$st word. The neighborhood here was defined as the region from Δ_{CB} frames *before* the $(k-1)$st word ending up to the ending frame of the $(k-1)$st word. The reasons for searching the neighborhood that overlapped only to the left[1] were as follows.

1) Digit coarticulation predicts that the shared boundary region between digits would be assigned entirely by the DTW algorithm to the first digit in the pair; hence backtracking is required in all such cases.

2) Since the reference templates for the digits were isolated occurrences of each digit, the tendency of the DTW matching was to go beyond the end of the region of the digit to match the last few frames of the usually longer and more emphasized reference.

Although the backtracking was generally successful, it did lead to some problems, especially with the shorter digits, i.e., 2 and 8. There was a tendency to insert spurious occurrences of these digits since the backtracking interval of $\Delta_{CB} = 12$ frames (180 ms) was often a significant percentage of the duration of the reference pattern.

The digit deletion problem was another result of the tip-to-tail recognition algorithm used here. For some cases the DTW algorithm was able to "look ahead" over shortened, coarticulated digits (especially 2 and 8) and match the succeeding digit in the string with a lower distance error than the heavily coarticulated correct digit.

Although compensation could be made for such digit deletions and insertions, the nature of the recognition algorithm would be changed considerably since one would have to literally keep track of "short" words (digits) and perform a different backtracking than for "long" words. As such the algorighmic properties of the procedure would be sacrificed for heuristic rules. An alternative would be to specify the number of digits in the string so that all single digit insertions and deletions (almost all cases) could be detected automatically and, in most cases, corrected. This approach is more desirable for the reasons discussed above.

The recognition accuracy of the system, although somewhat poorer than that of Sakoe [13], is still quite good, especially in the speaker-independent mode and for telephone quality inputs. The computation for this recognizer is an order of magnitude less than the computation of the Sakoe system, with somewhat worse accuracy scores. The results given here indicate that the information required to recognize digits in a string can be obtained by "sampling" the time scale nonuniformly, and at each of several time regions trying to match the test pattern to an isolated digit reference. The choice of sampling regions is dictated by the segmentation-recognition procedure. By concentrating the computational load to a few selected regions of the test pattern, a tremendous reduction in the computational load of the system is achieved. (This procedure is, for recognition, reminiscent of and similar to the UELM algorithm for time warping in continuous speech.)

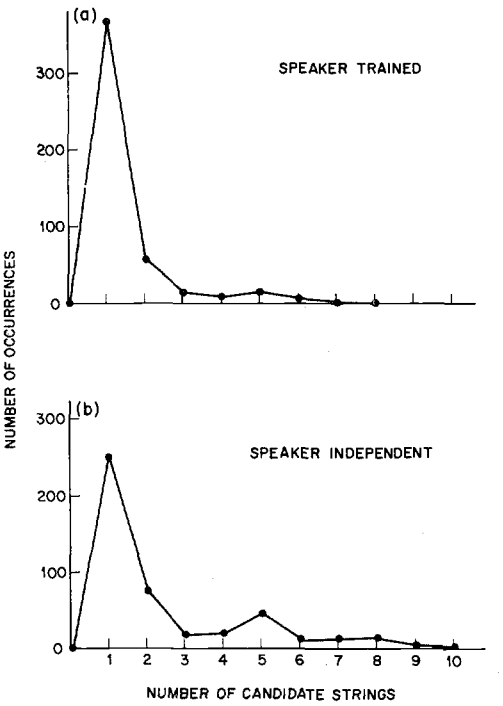


Fig. 8. Histograms of the number of candidate strings generated by the string recognizer for (a) speaker-trained results, and (b) speaker-independent results.

One interesting and important question remains about the results presented here—that is, how would the accuracy change if one could solve the recognition problem exhaustively—namely, by comparing the test string to every possible K digit string (for all anticipated values of K). As discussed earlier, such a set of comparisons is unfeasible. However, a modified set of such comparisons can be made based on the output of the recognizer. The reader will recall that the recognizer kept track of *all* digit strings whose distances were either sufficiently small, or whose distances were within a fixed amount of the string with the next smallest distance. As such up to 25 candidate strings were retained at the output of the recognizer for a single input string. Fig. 8 shows a set of histograms of the number of candidate strings (out of a possible 480 strings) for the speaker-dependent recognitions [Fig. 8(a)] and for the speaker-independent recognitions [Fig. 8(b)]. For the speaker-dependent case, 368 out of 480 strings (i.e., almost 70 percent) had only a single candidate string. For the speaker-independent case 251 out of 480 (about 52 percent) had only a single candidate string, and about 95 percent of the time 8 or fewer candidate strings were generated by the recognizer. For this small set of candidate strings it was possible to perform some "global" comparisons between the test and a suitably created reference string to either confirm the original recognition decision, or to provide a new possible recognition candidate string. We call the class of techniques of obtaining a second recognition score "post error correction" techniques.

We have considered two distinct types of post error correction techniques. The first is based on a full DTW comparison

[1] Technically, because of the properties of the time warping algorithm, the beginning point could in some cases, go past the end of the preceding digit by a small amount. Experimental evidence indicated that, in almost all cases, the beginning point of the kth word was to the left of the ending point of the $(k-1)$st word.

between the test string and a reference string obtained by concatenating the reference patterns that gave the test matches in the first recognition matching process. The second technique does a simpler DTW comparison based on the contours of voiced-unvoiced-silence frames of the test and suitably created reference strings. We describe these techniques in this section and present results on their performance for the test set of strings.

A. Full DTW Post-Error Correction

The output of the recognizer is an ordered list of strings of digits, ordered in terms of average distance per digit. For some strings the second (and often even lower position candidates) string has an average distance which is only slightly larger than the first string. It is these cases for which a post-error correction method might be able to correct errors.

The full DTW correction performed a constrained endpoint dynamic time warping distance calculation by comparing the entire input string T to each concatenated set of reference templates corresponding to each candidate string. Since more than one template per word was used in the recognizer, the actual template that provided the best individual digit match for the string was used in the concatenated string.

The results of the post error correction processing are given in Table V, which shows string error rates for each talker for the following 3 cases:

1) single stage recognition, i.e., the system described in Sections II and III; the distance obtained here is called D_I;

2) single stage recognition based entirely on the whole string recognizer distance D_{II}; and

3) combined two-stage recognition in which the combined distance

$$D_{\text{SUM}} = D_I + D_{II}$$

is used as the basis for recognition in which D_I is the total distance generated by the digit-by-digit recognizer, and D_{II} is the total distance generated by the whole string recognizer.

Table V has 4 parts corresponding to the 2 sets of parameters studied for speaker-dependent recognition [parts (a) and (b)], and the 2 sets of parameters studied for speaker-independent recognition.

For the speaker-dependent case the string error rates using either D_I or D_{II} alone were essentially comparable; however, for the combined distance a decrease in string errors of 4 (out of 47) and 6 (out of 38) occurred, corresponding to decrease in string error rates of 1 percent and 1.5 percent, respectively. For the combined distance measure, an analysis of the changes from the D_I distance measure is given in the columns labeled "strings corrected," which represents the number of strings that were recognized incorrectly from D_I but correctly from D_{SUM}, and "strings inverted" which represents the number of strings correct using D_I but in error using D_{SUM}. It is seen that the improvements in string error rate here represent fairly large changes in the individual strings [i.e., 18 changes occurred for part (a) and 22 changes for part (b)]. However, the net effect of the post-error correction is a small but significant decrease in error rate.

TABLE V

Talker	D_I Number of String Errors	D_{II} Number of String Errors	$D_I + D_{II}$ Number of String Errors	Number of Corrections	Number of Inversions
CS	2	6	4	0	2
LR	11	4	5	6	0
KS	9	9	7	3	1
SC	3	7	5	0	2
SL	9	9	9	0	0
JG	13	13	13	2	2
Totals	47	48	43	11	7
Percent Error	9.8	10.0	9.0	2.3	1.5

(a) Speaker Dependent Results for $\epsilon = 6$, $\Delta_{CB} = 8$

CS	4	4	4	1	1
LR	5	1	2	4	1
KS	5	6	4	2	1
SC	3	4	3	2	2
SL	10	8	8	2	0
JG	11	12	11	3	3
Totals	38	35	32	14	8
Percent Error	7.9	7.3	6.7	2.9	1.7

(b) Speaker Dependent Results for $\epsilon = 8$, $\Delta_{CB} = 12$

Talker	D_I Number of String Errors	D_{II} Number of String Errors	$D_I + D_{II}$ Number of String Errors	Number of Corrections	Number of Inversions
CS	8	6	6	3	1
LR	12	10	7	7	2
KS	2	3	3	1	2
SC	11	7	7	8	4
SL	12	14	11	3	2
JG	9	12	9	2	2
Totals	54	52	43	24	13
Percent Error	11.3	10.8	9.0	5.0	2.7

(c) Speaker Independent Results for $KNN = 2$

CS	10	8	7	4	1
LR	10	10	7	5	2
KS	8	6	5	3	0
SC	12	12	10	5	3
SL	17	12	10	7	0
JG	6	9	7	1	2
Totals	63	57	46	25	8
Percent Error	13.1	11.9	9.6	5.2	1.7

(d) Speaker Independents Results for $KNN = 1$

For the speaker-independent runs [parts (c) and (d)], significiantly larger improvements in accuracy are obtained from the D_{SUM} measure than from D_I or D_{II} alone. A net reduction of 11 string errors was obtained for $KNN = 2$ data, and a net of 17 string errors were eliminated for the $KNN = 1$ data. Although there were some string inversions for these cases, the reductions in string error rate using post-error correction are impressive. The overall string error rate for $KNN = 2$ was 9 percent with this post-error correction scheme.

A concomitant result of the reduction in string error rate was a similar reduction in digit error rate for both the speaker-dependent and the speaker-independent data.

The results presented in Table V indicate that a full DTW comparison based on the set of possible strings as generated automatically by the recognizer is justified since only a modest amount of extra computation is required, and a modest reduction in string and digit error rates is obtained.

B. Voiced-Unvoiced-Silence Post-Error Correction

A second method of post processing was investigated based on the voiced-unvoiced-silence (VUS) characteristics of the digits both in isolated and in connected strings. The basic idea was that some fairly confusable digit strings could be reliably recognized based on the voiced-unvoiced-silence contour of the test string. For example, confusions between the sequences 99 and 95 could theoretically be resolved based *entirely* on the VUS contour of the string. Hence, if the candidate list had such confusions (with comparable distance scores), the VUS contour could discriminate these strings.

The VUS contour of the test string was estimated using the pattern classification procedure originally described by Atal and Rabiner [23], [24]. For convenience the features used for VUS analysis were the set of $(p + 1)$ unnormalized autocorrelation coefficients of each frame of the signal. As such, no additional computation was required for feature analysis, and only a modest amount of computation was required for the VUS estimate.

The VUS contour of the reference digits was measured once and stored as a new reference set. A dynamic time warp comparison was then made between the test string VUS contour $(\tilde{T}_j)$ and the set of concatenated reference contours $(\tilde{R}_i)$ corresponding to each candidate string generated by the recognizer. The frame distance measure was the distance

$$\tilde{d}(j, i) = (\tilde{T}_j - \tilde{R}_i)^2$$

where

$\tilde{T}_j = 1$ if frame j silence
$= 2$ if frame j unvoiced
$= 3$ if frame j voiced

and similarly for $\tilde{R}_i$. Thus distances between silence and voiced frames were higher (value 4) than distances between any other pair of comparisons since such cases represented extreme differences in VUS contours.

The accumulated distance of the time warp $\tilde{D}$, defined as

$$\tilde{D} = \sum_{j=1}^{L} \tilde{d}(j, \tilde{w}(j))$$

where $\tilde{w}(j)$ was the optimum alignment path between $\tilde{T}$ and $\tilde{R}$ that minimized $\tilde{D}$, was then used as a measure of distance between $\tilde{T}$ and $\tilde{R}$. A threshold $\hat{W}$ was defined such that if $\tilde{D} < \hat{W}$, then the VUS of the test and reference contours were considered essentially identical. If $\tilde{D} > \hat{W}$, then the candidate string was considered potentially in error and the alternate with the lowest value of $\tilde{D}$ was chosen as the best candidate string.

Fig. 9 illustrates the behavior of the post-correction method for the speaker-trained recognizer obtained with $\epsilon = 8$, $\Delta_{CB} = 12$. Shown in this figure are the number of string corrections and the number of string inversions as a function of $\hat{W}$. It is seen that for $\hat{W} = 6$ there were 7 string corrections made for this data set. Although the net result here is essentially the same as for the combined distance of Table V, the effect is somewhat different since the net of 6 fewer string errors for the first correct method involved 14 string corrections and 8

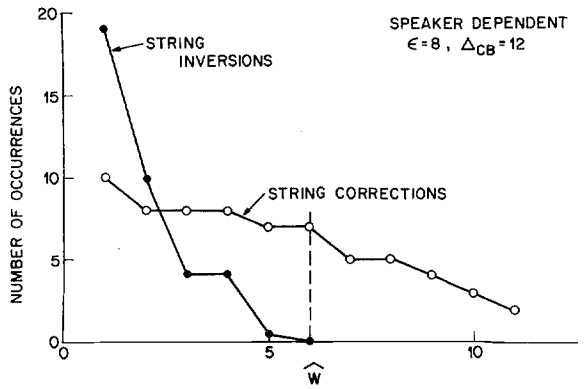


Fig. 9. Plots of the number of string corrections and string inversions as a function of threshold $\hat{W}$ for the VUS post-correction method for speaker-dependent results.

string inversions. A similar set of results were obtained for the alternate set of parameters ($\epsilon = 6$, $\Delta_{CB} = 8$) for the speaker-trained recognizer in which a net of 4 fewer string errors occurred using the post-error VUS correction.

For the speaker-independent recognizer, the VUS post-correction did not provide *any* string correction capability. For these data the VUS contours of the reference digits were sufficiently variable among the 12 template set, that almost any concatenated reference VUS contour provided a match (to any string contour) that was about the same as any other concatenated reference. Also, on average, a large number of essentially identical candidate strings (to within the VUS contours) were generated in the speaker-independent case, thereby adding to the confusion among candidate strings. Hence the VUS contours provided essentially no error correction capability.

C. Summary of Post-Correction Methods

The overall conclusion on the applicability of post-error correction methods is that they provide a small but useful reduction in the string error rate. The full DTW error correction appeared to be more robust across all data sets than the VUS method. However, for the speaker-trained cases, the simplicity and the minimal amount of computation could potentially make the VUS method an attractive one.

V. Summary

In this paper we have discussed a system for the recognition of strings of connected digits over dialed-up telephone lines. The string length is unspecified and is determined by the recognition algorithm. The system uses isolated word templates as the basis of a pattern matching algorithm and can be used as either a speaker-trained, or a speaker-independent recognizer, depending on the set of templates.

The recognition process can be viewed as a modified form of the two-stage dynamic programming procedure proposed by Sakoe and used in the NEC hardware recognizer. The digits in the string are recognized sequentially using an unconstrained dynamic time warping algorithm and then a region around the endpoint of the kth digit is used as the beginning region of the $(k + 1)$st digit. The output of the recognizer is an ordered set of candidate strings of digits, ordered by average distance.

A post-correction process was then used to provide a small improvement in the recognition accuracy based on string comparisons using a constrained dynamic time warping algorithm. The recognition system obtained digit accuracies of about 97-99 percent, and overall string accuracies (after post-correction) of from 91-93 percent.

REFERENCES

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

[2] P. Vicens, "Aspects of speech recognition by computer," Ph.D. dissertation, Stanford Univ., Stanford, CA, Apr. 1969.

[3] D. R. Reddy, "Speech understanding systems-Summary of results of the five-year research effort at Carnegie-Mellon University," Carnegie-Mellon Univ., Pittsburgh, PA, Tech. Rep., Aug. 1977.

[4] V. M. Velinchko and N. G. Zagoruyko, "Automatic recognition of 200 words," *Int. J. Man-Machine Studies*, vol. 2, pp. 223-234, 1970.

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

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

[7] L. R. Rabiner, "On creating reference templates for speaker-independent recognition of isolated words," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26, pp. 34-42, Feb. 1978.

[8] 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.

[9] 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.

[10] L. R. Rabiner and J. G. Wilpon, "Considerations in applying clustering techniques to speaker independent word recognition," *J. Acoust. Soc. Amer.*, vol. 66, pp. 663-673, Sept. 1979.

[11] —, "Speaker-independent, isolated word recognition for a moderate size (54 word) vocabulary," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 583-587, Dec. 1979.

[12] H. Sakoe and S. Chiba, "A dynamic programming approach to continuous speech recognition," in *Proc. 7th ICA*, paper 20 C13, Aug. 1971.

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

[14] 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.

[15] 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.

[16] S. Tsuruta, H. Sakoe, and S. Chiba, "DP-100 connected speech recognition system," in *Proc. INTELCOM 79*, Feb. 1979.

[17] S. Tsuruta, "DP-100 voice recognition system achieves high efficiency," *J. Eng. Educ.*, pp. 50-54, July 1978.

[18] G. Doddington, unpublished results.

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

[20] B. T. Lowerre, "The HARPY speech recognition system," Ph.D. dissertation, Carnegie-Mellon Univ., Pittsburgh, PA, 1976.

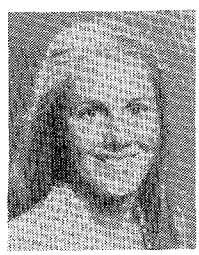
[21] J. S. Bridle and M. D. Brown, "Connected word recognition using whole word templates," in *Proc. Autumn Conf. Institute of Acoustics*, 1979.

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

[23] B. S. Atal and L. R. Rabiner, "A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 201-212, June 1976.

[24] L. R. Rabiner, C. E. Schmidt, and B. S. Atal, "Evaluation of a statistical approach to voiced-unvoiced-silence analysis for telephone-quality speech," *Bell System Tech. J.*, vol. 56, pp. 455-482, Mar. 1977.

Lawrence R. Rabiner (S'62-M'67-SM'75-F'76), for a photograph and biography, see p. 78 of the February 1980 issue of this TRANSACTIONS.



Carolyn E. Schmidt was born in Plainfield, NJ, on July 5, 1952. She received the B.S. degree in mathematics from Lafayette College, Easton, PA, in 1974.

Upon graduation, she joined the Acoustics Research Department, Bell Laboratories, Murray Hill, NJ, where she has been involved in speech communication work including recognition, digital simulation of telephone quality speech, and voiced-unvoiced-silence detection.

Ms. Schmidt is a member of Phi Beta Kappa.