A COMPARATIVE STUDY OF SEVERAL DYNAMIC TIME WARPING ALGORITHMS FOR SPEECH RECOGNITION

by

Cory S. Myers

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREES OF

BACHELOR OF SCIENCE

and

MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 1980

Signature of Author ....
Department of Electrical Engineering and
Computer Science, February 4, 1980.

Certified by ....
Thesis Supervisor (Academic)

Certified by ....
Company Supervisor (VI-A Cooperating Company)

Certified by ....
Company Supervisor (VI-A Cooperating Company)

Accepted by ....
Chairman, Departmental Committee on Graduate Students

A COMPARATIVE STUDY OF SEVERAL DYNAMIC TIME WARPING ALGORITHMS FOR SPEECH RECOGNITION

by

Cory S. Myers

Submitted to the Department of Electrical Engineering and Computer Science on February 4, 1980, in partial fulfillment of the requirements for the Degrees of Bachelor of Science and Master of Science.

ABSTRACT

A comparative study of several dynamic time warping algorithms for speech recognition was conducted. Performance measurements based on memory usage, recognition accuracy and computational speed were made. The first part of the investigation involved dynamic time warping for isolated word recognition. It was assumed that the word endpoints had been reliably obtained. Factors which were considered included local continuity constraints on the dynamic path, global range constraints and the type of normalization. Broad classifications were made to highlight the strengths and weaknesses of the various algorithms. In addition, a new approach to dynamic time warping for isolated word recognition was examined. This approach applied linear normalization to both the test and the reference utterance prior to a non-linear time warping. Results of experiments on this algorithm show comparable performance to the other dynamic time warping algorithms investigated. The practical importance of this new method is presented in the thesis.

In the second part of the investigation two general dynamic time warping algorithms for word spotting and connected speech recognition are described. These algorithms are called the fixed range and the local minimum method. The characteristics and properties of these algorithms are discussed. It is shown that the local minimum method performs considerably better than the fixed range method. Explanations of this behavior are given and an optimized method of applying the local minimum algorithm is discussed. It is shown that, for word spotting problems, successive trials of the local minimum algorithm need not be made at every possible starting point in order to achieve good accuracy. We also demonstrate that one reasonable approach to connected speech recognition is to build reference strings using a single local minimum time warp per word of the test utterance and hypothesizing the beginning of one word based on the end of the previous word.

THESIS SUPERVISOR: Jae S. Lim
TITLE: Assistant Professor of Electrical Engineering and Computer Science

THESIS SUPERVISOR: Aaron E. Rosenberg
TITLE: Member of Technical Staff at Bell Telephone Laboratories

THESIS SUPERVISOR: Lawrence R. Rabiner
TITLE: Member of Technical Staff at Bell Telephone Laboratories

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my two advisors at Bell Laboratories, Dr. Aaron E. Rosenberg and Dr. Lawrence R. Rabiner, for their guidance, encouragement and helpful suggestions. Further special thanks is due to Dr. Rabiner for his continued enthusiasm and his suggestions regarding the writing of this manuscript. I am also particularly indebted to Dr. James L. Flanagan of Bell Laboratories for inviting me to work in his department and for providing all the necessary facilities, and to Dr. Jae S. Lim of MIT for agreeing to be my thesis supervisor.

I wish also to thank Mrs. Kathy L. Shipley for her assistance in the learning of the use of the computer system and for her help in the software development. Special thanks is also due both the Word Processing and the Drafting Departments of Bell Laboratories for their fine work.

Additional thanks is due to Jay Wilpon for his help in providing data files and for his encouragement and to Lori Lamel for her encouragement and helpful comments on the writing of this thesis.

Finally, I am sincerely thankful to my parents, Perry and Eleanor Myers, for their continuing love, support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

FIGURES

TABLES

Table *Page*

Table of Symbols

a,b	initial and final values of t for the general calculus of variations problem
a_j	j^{th} linear prediction coefficient
$\tilde{a}_j^{(n)}$	j^{th} linear prediction coefficient for the n^{th} frame of the reference
b_1,b_2	bounds on the beginning region in a test pattern
$\tilde{b}$	middle of a beginning region
B	size of a beginning region
$c(k)$	local minimum of $D_A(i(k)-1,j)$ along the j axis
$d(i(k),j(k))$	distance between frame $i(k)$ of the reference and $j(k)$ of the test
$d(t,w(t))$	distance between location t of the reference and $w(t)$ of the test
$\tilde{d}(\mathbf{R}(n),\mathbf{T}(m))$	distance between frame n of the reference and frame m of the test
$\bar{d}$	a constant distance value
$\hat{d}((n',m'),(n,m))$	distance from a point (n',m') to a point (n,m) via the best possible path
$D(i(k),j(k))$	normalized distance along a path defined by the functions $i(k)$ and $j(k)$
$D(w(t))$	normalized distance along a curve defined by the function $w(t)$
$\hat{D}$	best normalized distance along any path
$D^{(v)}(i(k),j(k))$	normalized distance along path defined by $i(k)$ and $j(k)$ and using reference pattern v
$\hat{D}^{(v)}$	best normalized distance along path using reference pattern v
$D_A(n,m)$	accumulated distance to the point (n,m) along the best path to it
e_1,e_2	bounds on the ending region in a test pattern
E	size of an ending region
$E^{(i)}$	i^{th} order LPC residual energy
$E_{\max}$	maximum allowable slope of a path
$E_{\min}$	minimum allowable slope of a oath
$F(t,w(t),\dot{w}(t))$	a function of t, $w(t)$ and $\dot{w}(t)$
$g(k)$	Itakura's dynamic time warping function designed to limit compression of a reference pattern
$H(z)$	transfer function of a system
$(i(k),j(k))$	parameterized path functions
K	length of path
K'	length of a subsection of a path
$L(r)$	length of the r^{th} production
M_1,M_2	bounds on the position of speech in a test pattern
M	length of the speech component of a test pattern
$\hat{M}$	normalized length of the test pattern
MS	samples between successive frames

N_1, N_2	bounds on the position of speech in a reference pattern
N	length of the speech component of a reference pattern
$\hat{N}$	normalized length of the reference pattern
$\mathbf{N}(\tilde{W})$	normalization factor associated with weighting function $\tilde{W}$
N_S	samples used per frame
$NTRY$	number of local minimum time warps
p	order of an LPC production
$\mathbf{p}_D(D)$	probability density function of distances for reference and test different
$\mathbf{p}_S(D)$	probability density function of distance for reference and test the same
P_r	r^{th} production rule
P_F	probability of a false alarm
P_M	probability of a miss
$P_D(D)$	cumulative probability distribution of $\mathbf{p}_D(D)$
$P_S(D)$	cumulative probability distribution of $\mathbf{p}_S(D)$
R	range limit, in frames
$\mathbf{R}(n), \mathbf{R}(t)$	a reference pattern, both discrete and continuous time
$R(n,i)$	i^{th} component of $\mathbf{R}(n)$
$\hat{\mathbf{R}}(\hat{n})$	normalized length reference pattern
$\tilde{R}(l)$	l^{th} autocorrelation
$\tilde{R}^{(m)}(l)$	l^{th} autocorrelation of the m^{th} frame of the test
$s(n)$	speech signal
$\tilde{s}(n)$	filtered speech signal
T_s	sampling period for frames
$\mathbf{T}(m), \mathbf{T}(\tau)$	a test pattern, both discrete and continuous time
$T(m,i)$	i^{th} component of $\mathbf{T}(m)$
$\hat{\mathbf{T}}(\hat{m})$	normalized length test pattern
$\hat{v}$	index of the reference pattern which best matches a test pattern
V	vocabulary size
w_a, w_b	initial and final values for the general calculus of variations problem
$w(n), w(t)$	a warping function, both discrete and continuous time
$W(n)$	a windowing function
$\tilde{W}(k)$	discrete time path weighting function
$\tilde{W}(t, w(t), \dot{w}(t))$	continuous time curve weighting function
$x(n)$	samples of $\tilde{s}(n)$ within a frame
$\hat{x}(n)$	windowed samples of $x(n)$
$\alpha_j^{(i)}$	j^{th} order LPC coefficient after i iterations of Durlin's recursion
$(\alpha_l^{(r)}, \beta_l^{(r)})$	l^{th} element of the r^{th} production rule

δ	separation between beginning regions for several local minimum time warps
Δ	total range covered by several local minimum time warps
ϵ	local range in the local minimum time warp
Φ	threshold for overlapping probability distributions
$\phi(t)$	a function of t
$\dot{\phi}(t)$	derivative of $\phi(t)$ with respect to t
ϕ_x	partial derivative of ϕ with respect to x

Chapter 1

Introduction

1.1 Speech Recognition Systems

An important step in the realization of man-machine communications is the development of a practical speech recognition system. Work over the past decades has advanced automatic speech recognition systems to the point where such systems are being used (in some cases on an experimental basis) for such diverse applications as directory assistance [1] and voice control of machinery on an assembly line [2]. However, such applications are still severely limited with regard to vocabulary size, type of speech input, and the environment under which these recognizers can properly function. These constraints arise from the inherent complexity of the speech recognition problem along with the massive amounts of computation generally required to "solve the broad speech recognition problem."

Several factors determine the performance of speech recognition systems. A system may be able to handle a large vocabulary or a small one. The input speech utterances may be as simple as single, isolated letters, digits or words, or as complex as complete sentences. Speech recognition systems are also classified as being either speaker independent or speaker dependent. Speaker dependent systems require a learning period in which the machine is trained to the user's voice. Speaker independent systems, while more versatile in their ability to handle a wide class of talkers without individualized training, are generally more complex and costly.

Many speech recognition systems rely on pattern matching concepts. Such systems have many basic features in common, the major components of which are depicted in the block diagram of Figure 1.1. First the input utterance is filtered, digitized, and analyzed to determine the beginning and ending points of the speech (i.e. to separate the spoken text from the background silence). Next, a set of features is measured for the speech in order to represent the utterance in a form more amenable to recognition (i.e. in a data reduced format). Common parametrizations include some or all of the following measurements: zero crossing rates, linear

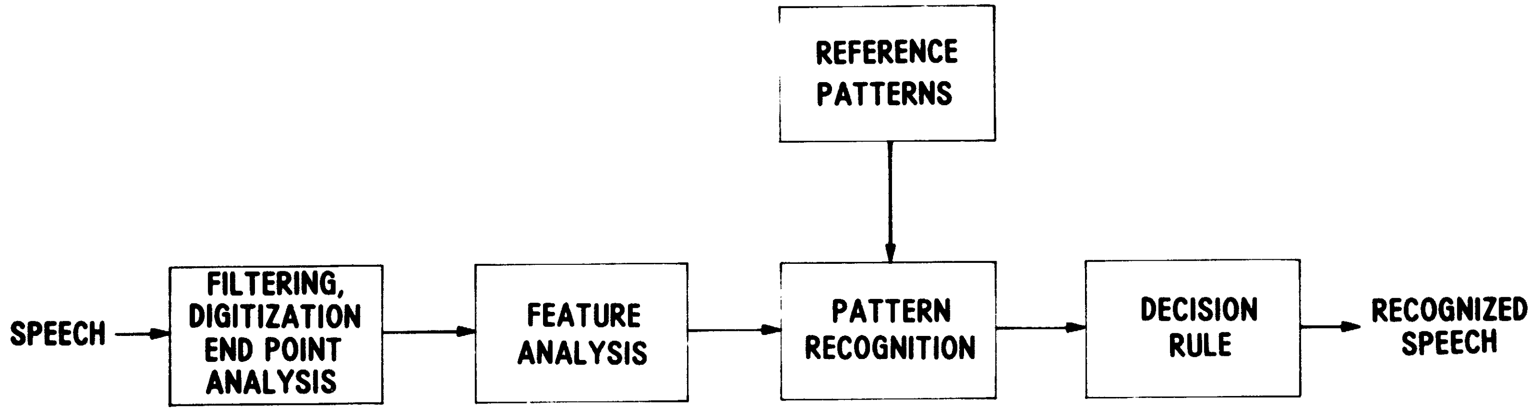


Fig. 1.1 Block diagram of an Automatic Speech Recognition System.

predictive coding (LPC) coefficients, spectral coefficients, cepstral coefficients, etc. In a typical recognition system, these parameters are calculated once every frame, where a frame is typically 20 to 50 milliseconds in duration. Usually, a new frame is calculated every 10 to 30 milliseconds (i.e. frames generally overlap in time). The resulting time sequence of features for the test utterance is defined to be a test pattern. In order to determine what speech is present in the test utterance, the parametrized speech is compared to a set of stored reference patterns consisting of previously parametrized words, syllables or phonemes (obtained from a training set of data), and the "best" fit is selected as the most likely candidate for the speech utterance. For purposes of our investigations into word recognition the unit of recognition will be isolated words. As we shall see in Chapter 2, such an assumption will be fundamental, in that, the use of a smaller recognition unit will require either accurate segmentation of the input utterance or the use of an entirely different time warping procedure than the one which will be used.

In order to obtain such a "best fit", we are faced with the problem of comparing a test pattern with a set of reference patterns. Generally, the time scales of the test and reference patterns are incommensurate. However, even if the time scales are the same, it is highly unlikely that the timing of the test precisely matches the timing of each reference. As such, we must use some method of time warping to optimally register the test pattern with each reference pattern. We discuss the problem of time warping in the next section.

1.2 Time Warping

In order to properly compare the parametrized representation of an input speech signal (the test pattern) with a reference pattern, temporal variations between the two patterns, due to differences in the way in which a speaker may say the same utterance at two different occasions, must be compensated for. Such temporal variations can include absolute differences in the length of a test pattern and its corresponding reference pattern, as well as local variations in which the test utterance may be sped up at one section of time and slowed down at another relative to the reference pattern. Time alignment, or time warping, is a procedure in which the input utterance's temporal feature set is locally stretched and/or compressed in order to achieve the best possible match to the reference. Figure 1.2 shows an example of time warping one function to fit another. Part a shows the input signal and part b shows the reference to which the input is to be matched. The time warping function is shown in part c and the resulting match between the input and the reference is shown in part d. It is clear from this simple example that time warping can provide significant improvements in matching two patterns.

The simplest implementation of time warping is linear compression or expansion of the input utterance to the reference. While this method is often satisfactory for monosyllabic words [3], it is generally unsatisfactory for polysyllabic words or sentences. In 1971, Sakoe and Chiba proposed the use of dynamic time warping to improve the fit between reference and test patterns [4]. In this method, a nonlinear expansion and/or compression of the time scale is used to provide an optimal fit between the patterns. Sakoe and Chiba also suggested the use of dynamic programming, as developed by Bellman [5] in 1962, for the efficient implementation of the time warping algorithm. Such implementations of the time warping algorithm have subsequently led to great improvements in speech recognition systems [4,6,7]. The basic components of a dynamic time warping (DTW) algorithm include a distance metric (for comparing frames of the test and reference), the specification of local and global continuity constraints (to determine the warping contour), and endpoint constraints (to define initial and final registration of the test and reference). The DTW distance measure appropriate for speech recognition is dependent on the feature set used for parametrization. For example, a log spectral difference is suitable for bandpass filter parameters, a Euclidean distance is suitable for cepstral coefficients, and a log likelihood ratio, as originally proposed by Itakura [6], is suitable for LPC coefficients. Global and local continuity constraints are used to insure that time continuity is preserved in the warped pattern, and that excessive warping is not used in the procedure - e.g. locally or globally expanding a very short utterance to match a very long one.

1.3 The Work Undertaken in this Thesis

In the work undertaken in this thesis, several previously proposed and some new dynamic time warping algorithms are compared. For each of the algorithms, measurements of computa-

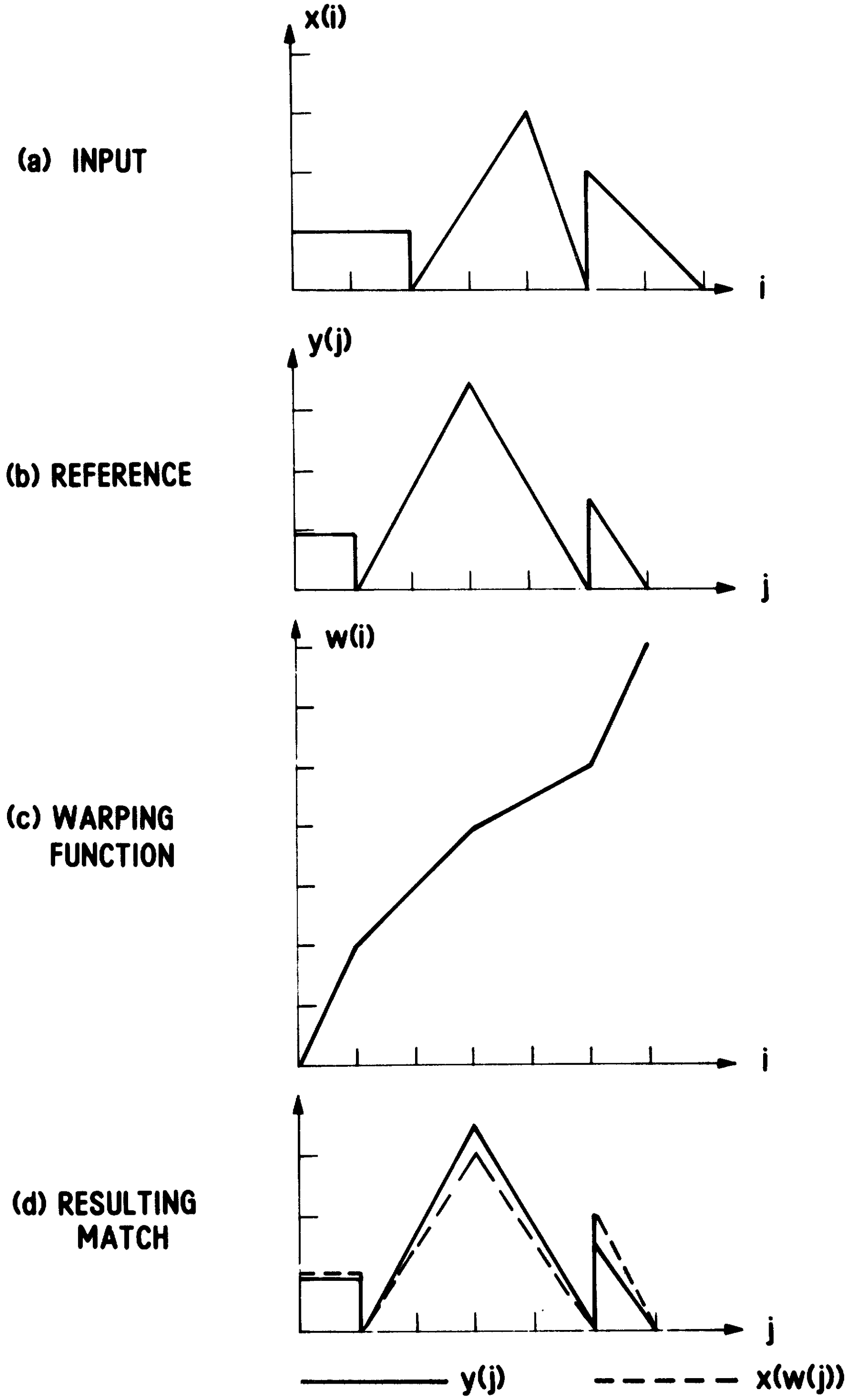


Fig. 1.2 Example of time warping.

tional efficiency, recognition accuracy and memory requirements are made. To perform these measurements, a feature set based on an eighth order LPC analysis for each speech frame was used. As such, the log likelihood ratio distance measure of Itakura was used as the distance metric. A standard set of speakers and reference templates, corresponding to a set of isolated words, was used to test all the DTW algorithms. Two major application areas of the DTW algorithms are examined in this thesis; namely, dynamic time warping for isolated word recognition, and dynamic time warping for continuous speech recognition. Tradeoffs among memory usage, computational efficiency and recognition accuracy are investigated.

For the isolated word recognition system, we assume that a reliable set of endpoints for each word has been found (i.e. we are not concerned here with the problems of endpoint detection). The major variations in the DTW algorithms are related to the global path constraints, the local continuity constraints and the type of normalization in the distance scores. Experimental results are presented on several sets of data for each DTW algorithm, and tradeoffs among the performance variables are discussed. The performance of the dynamic time warping algorithms is found to be highly dependent upon the ratio of the length of the test pattern to the length of the reference pattern. Finally, a DTW algorithm is studied in which the lengths of the test and the reference utterances are normalized to a standard duration prior to the time warping. This algorithm is shown to yield the best performance among all the isolated word dynamic time warping algorithms that were studied. It is also shown that this algorithm has several practical advantages for a hardware implementation of an isolated word recognizer.

1.4 Organization of Subsequent Chapters

The subsequent chapters may be broken into two distinct groups. Chapters 2 through 5 deal with dynamic time warping as it is applied to isolated word recognition. Chapter 2 gives a formal description of DTW algorithms for isolated word recognition and defines the variables of interest. In Chapter 3, performance criteria and methods of evaluation of these criteria are defined for the various dynamic time warping algorithms. Chapter 4 discusses the experiments performed and gives the performance results. Finally, in Chapter 5, tradeoffs are examined and conclusions about the various DTW methods are drawn. In addition, the practical importance of the research is discussed and further work in dynamic time warping for isolated word recognition is proposed.

In the second section of the thesis, Chapters 6, 7 and 8 we discuss the application of DTW algorithms to both word spotting and connected speech recognition. In Chapter 6 we describe the basic principles involved in DTW applications to such areas. We also define two basic DTW algorithms for word spotting and connected speech recognition - the fixed range and the local minimum DTW algorithms. Chapter 7 presents results concerning the comparison of the two algorithms and examination of the parameters of the algorithms. Finally, in Chapter 8, a summary is made and conclusions are drawn. In additions, practical applications of the DTW algorithms are discussed and proposals for future research into the use of DTW algorithms for word spotting and connected speech recognition are made.

Chapter 2

Fundamentals of Dynamic Time Warping

2.1 General Description of the Problem

Time registration of a test utterance and a reference utterance is one of the fundamental problems in the area of automatic isolated word recognition. This problem is important because the time scales of a test pattern and a reference pattern generally are not perfectly aligned. In some cases the time scales can be registered by a simple linear compression or expansion; however, in most cases, a nonlinear time warping is required to compensate for local compressions and expansions of the time scales. In such cases, a general class of procedures, collectively referred to as time warping algorithms, has been developed. These procedures have been shown to be applicable to the "isolated word" speech recognition problem and to greatly improve the accuracy of automatic speech recognition systems [4,6,7].

One possible interpretation of time warping is to consider it as a method for determining an optimal function to map one time axis into another. Optimality is determined by minimizing a distance function (or maximizing similarity) between one pattern and the time warped version of the other. Figure 2.1 illustrates this interpretation of time warping. We will denote a reference pattern as $\mathbf{R}(n)$, $0 \leqslant n \leqslant N$, and a test pattern as $\mathbf{T}(m)$, $0 \leqslant m \leqslant M$. In general $\mathbf{R}(n)$ and $\mathbf{T}(m)$ may be multidimensional feature vectors but for simplicity we show them as simple, one-dimensional functions in Figure 2.1. We denote the range of $\mathbf{R}(n)$ which is of interest by its endpoints N_1 and N_2, which satisfy the trivial relation $0 \leqslant N_1 < N_2 \leqslant N$, and the range of $\mathbf{T}(m)$ of interest by M_1 and M_2 where $0 \leqslant M_1 < M_2 \leqslant M$. The purpose of time warping is to provide an optimal mapping from one time axis (the n scale) to the other (the m scale). Figure 2.1 shows the warping function defined as

$$m = w(n). \tag{2.1}$$

One problem inherent in the interpretation of the time warping problem as finding an optimal mapping from one time axis to another is that such a description assumes that both

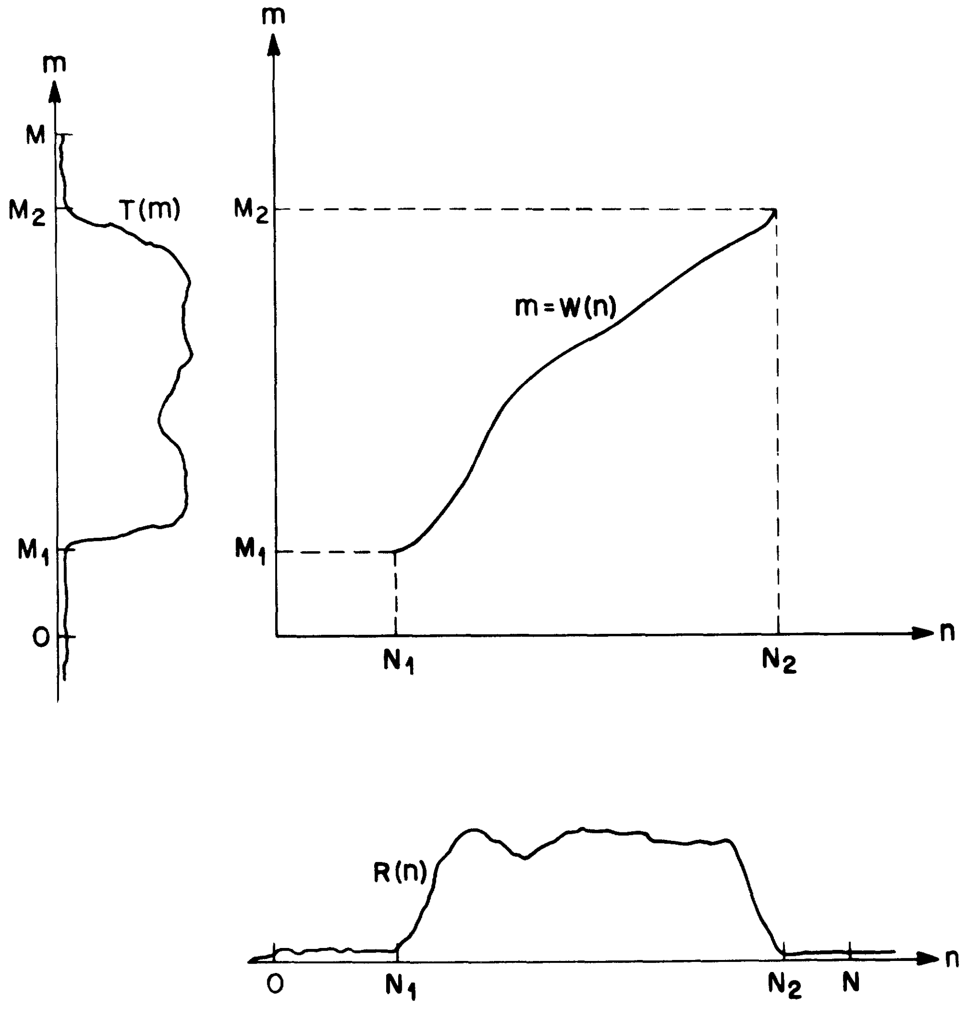


Fig. 2.1 Time warping.

$\mathbf{R}(n)$ and $\mathbf{T}(m)$ are continuous functions of time. This is not the usual case. Typically, $\mathbf{R}(n)$ and $\mathbf{T}(m)$ are sampled signals, with typical sampling intervals of 10 to 30 milliseconds. (Appendix 1 gives one proposed solution to the continuous - time, time warping problem.) Since $\mathbf{R}(n)$ and $\mathbf{T}(m)$ are sampled signals, it is simpler to pose the time warping problem as a path finding problem. We can, without loss of generality, assume that $\mathbf{R}(n)$ is defined for $n = 1,2,...,N$ and that $\mathbf{T}(m)$ is defined for $m = 1,2,...,M$. Once again, the time warping procedure must find a function of the form of Eq. (2.1) to minimize a total distance function, D, of the form

$$D = \sum_{n=1}^{N} \tilde{d}(\mathbf{R}(n),\mathbf{T}(w(n))) \tag{2.2}$$

where $\tilde{d}(\mathbf{R}(n),\mathbf{T}(w(m)))$ is the local distance between frame n of the reference and frame $m = w(n)$ of the test. A typical path, $w(n)$, is shown in Figure 2.2. It is important to notice that $w(n)$ is restricted to begin at the point $n = 1$, $m = 1$, to pass through the grid of points (n,m), where n and m are integers, and to end at the point $n = N$, $m = M$.

Although $w(n)$ has been restricted to integer values, it is still functional in nature, i.e. for any n, the time alignment path passes through at most value of m. It is not unreasonable, however, that the best warping may not be functional. In this situation it is necessary to create a time warping procedure which maps both the reference pattern's time axis and the test pattern's time axis onto a common time axis. Such a procedure requires the use of two functions, $i(k)$ and $j(k)$, where k is the index of the common time axis. These two functions are used to map $\mathbf{R}(n)$ and $\mathbf{T}(m)$ to the common time axis, k, according to the rules

$$n = i(k), \quad k = 1,2,...,K \tag{2.3a}$$

$$m = j(k), \quad k = 1,2,...,K \tag{2.3b}$$

where K is the length of the common time axis. Figure 2.3 shows a typical example in which we plot $i(k)$ and $j(k)$ versus k, and in which we also show the resulting curve in (n,m) space. We see that, in (n,m) space, the resulting curve can be interpreted as a monotonically increasing path from the point (1,1) to the point (N,M) via several intermediate points. It is also

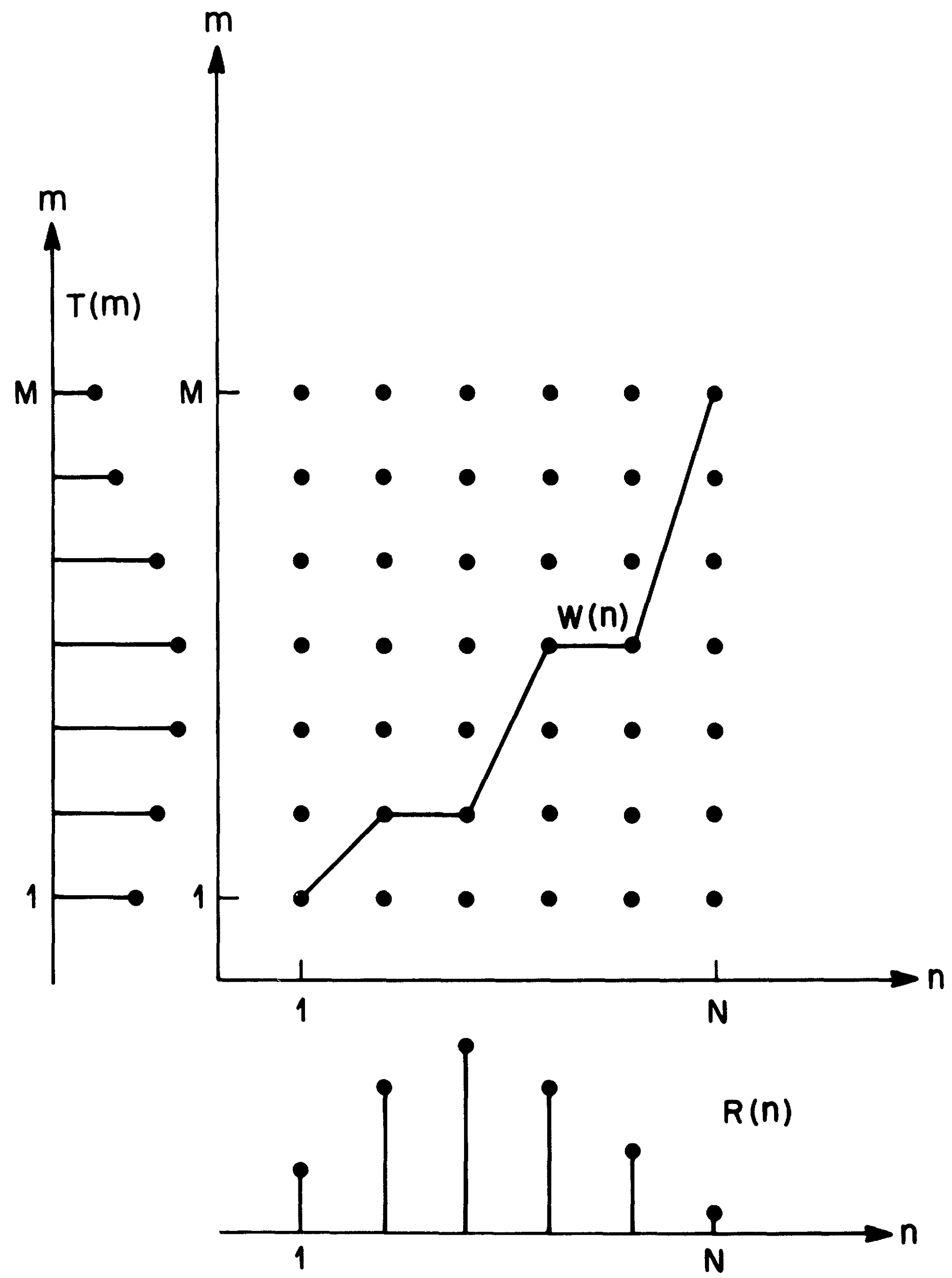


Fig. 2.2 Discrete-time warping.

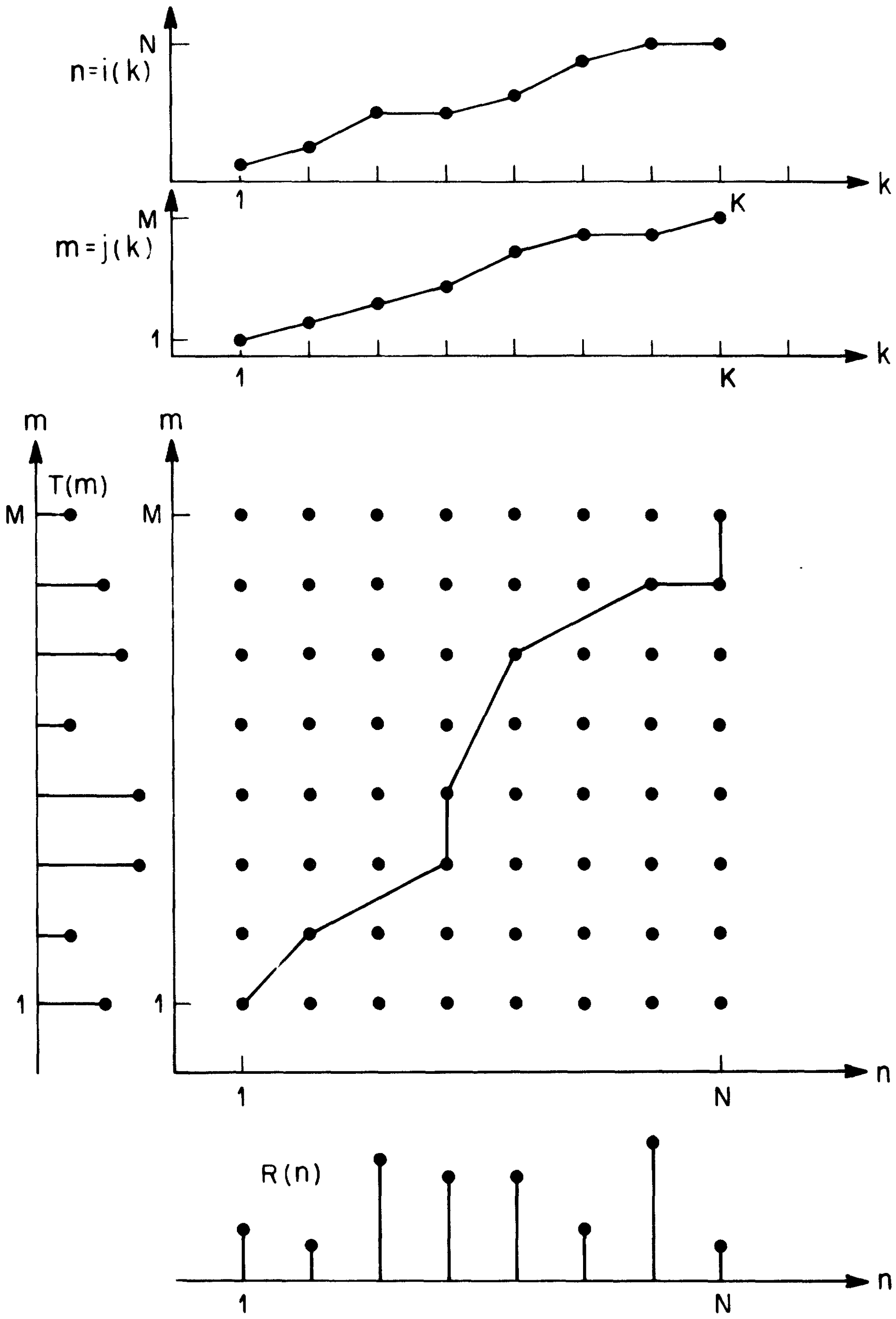


Fig. 2.3 Time warping as path finding.

clear that if the function $i(k)$ is chosen such that $n = i(k) = k$ then $m = j(k) = j(n) = w(n)$, i.e. the problem is equivalent to the discrete time version of the problem given in Eq. (2.1). We will use the interpretation of time warping as finding an optimal path as the general framework for the remainder of this thesis.

2.2 Time Warping as Path Finding

Based on the discussion of the previous section, we see that for the interpretation of time warping as a path finding problem we must specify several features of the problem. The factors which are applicable to the path finding problem are the following:

1. Endpoint constraints - i.e. the way in which the path begins and ends.
2. Local continuity constraints - i.e. the possible types of motion (e.g. directions, slopes, etc.) of the path.
3. Global path constraints - i.e. the limitations on where the path can fall in the (n,m) plane.
4. Axis orientation - i.e. the effects of interchanging the roles of the test and reference patterns.
5. Distance measures - both the local measure of similarity or distance between frames of the reference and test patterns and the overall distance function used to determine the optimal path.

In this section we discuss some possible (and hopefully reasonable) choices for each of the above factors for speech recognition applications.

2.2.1 Endpoint Constraints

Endpoint considerations for speech recognition fall into two broad categories based on whether the application is for connected words or for isolated words. We defer the question of how to handle endpoint constraints of connected words to Chapter 6 of this thesis. For isolated word recognition, endpoint detection is a relatively well-understood problem and several viable solutions have been proposed [8,9]. In the next three chapters we will be solely concerned with speech recognition systems which use simple isolated test utterances and which use reference patterns consisting of isolated words. For time warping algorithms involving isolated utterances it is reasonable to assume that the endpoints of both the test and the reference patterns have been reliably determined. Given such an assumption, a time warping algorithm should be restricted to have all of its paths start at the point (1,1) (the first frame of both the reference and the test) and end at the point (N,M) (the final frame of both the reference and the test). In terms of the path notation we have

$$i(1) = 1, j(1) = 1 \tag{2.4a}$$

$$i(K) = N, j(K) = M. \tag{2.4b}$$

2.2.2 Local Continuity Constraints

Local continuity constraints are another important consideration for time warping in speech recognition systems. Local continuity constraints define what types of paths are allowable. For example, it would not be reasonable to allow a path for which a 10 to 1 expansion or compression of the time axis occurs. Another consideration is the preservation of time order. The functions $i(k)$ and $j(k)$ should both be monotonically increasing, i.e.,

$$i(k+1) \geqslant i(k) \tag{2.5a}$$

$$j(k+1) \geqslant j(k). \tag{2.5b}$$

Local continuity constraints are easily expressed as simple local paths which may be pieced together to form larger paths. For example, to reach a point (n,m) it may be reasonable to have come from the points $(n-1,m-1)$, $(n-1,m-2)$ or $(n-2,m-1)$. Such a set of legal paths may be viewed as shown in Figure 2.4, part a. Further restrictions may be placed on the local paths. For example, the path from $(n-1,m-2)$ to (n,m) may be forced to pass through the point $(n,m-1)$ and the path from $(n-2,m-1)$ to (n,m) may be restricted to pass through the point $(n-1,m)$. Such restricted paths are shown pictorially in part b of Figure 2.4 and are labeled as type I local constraints to distinguish them from other local constraints which will be defined later.

Local constraints for time warping may be formally expressed as a set of productions in a regular grammar. A production is a rule of the form

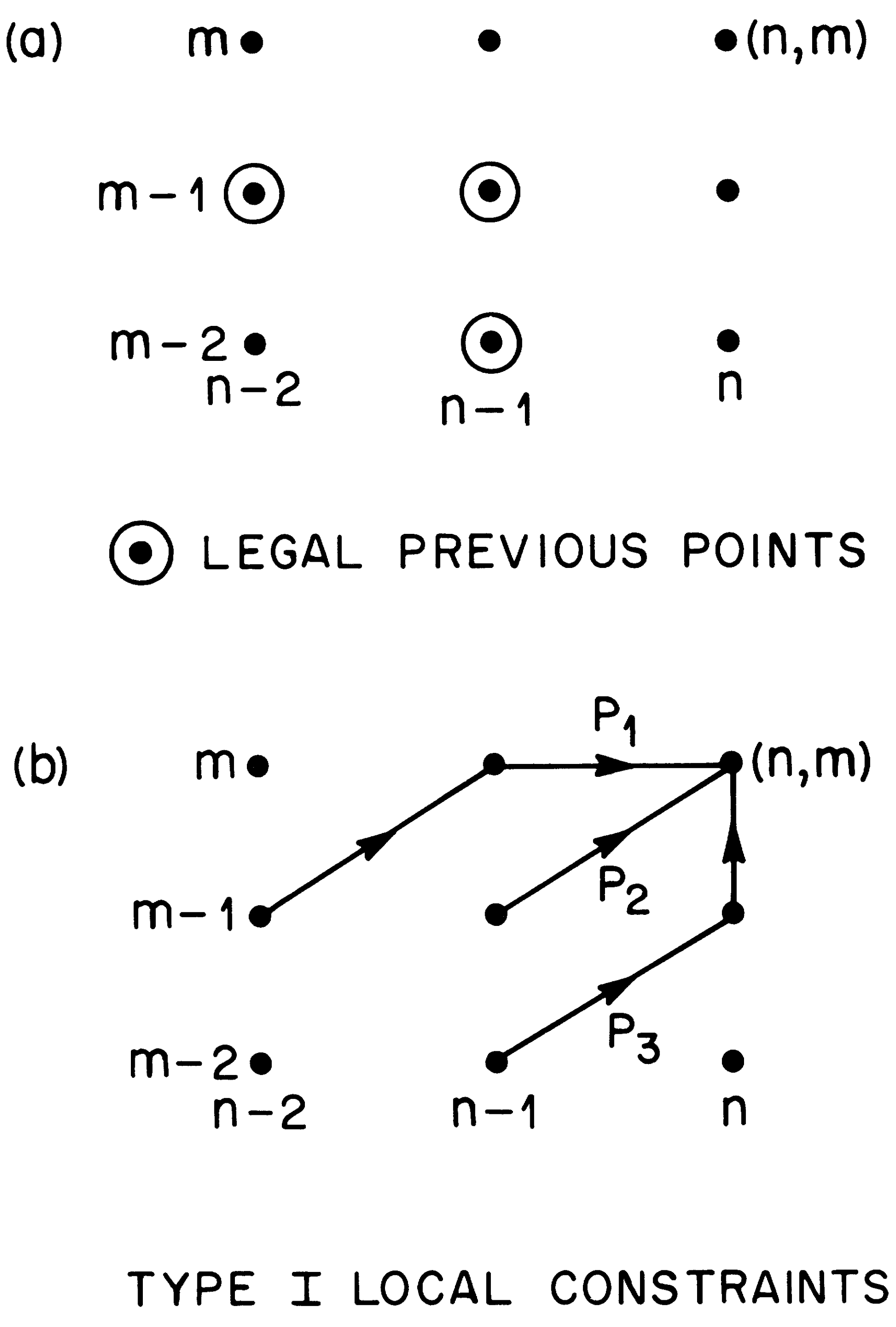


Fig. 2.4 Type I local constraints.

$$P_r \rightarrow (\alpha_1^{(r)},\beta_1^{(r)})(\alpha_2^{(r)},\beta_2^{(r)}) \cdots (\alpha_{L(r)}^{(r)},\beta_{L(r)}^{(r)}) \tag{2.6}$$

where r signifies the r^{th} production and $L(r)$ is the length of the r^{th} production. The interpretation of the $(\alpha_l^{(r)},\beta_l^{(r)})$'s in a production are that of the local changes (i.e. incremental changes) allowable in a path. The interpretation of a production, P_r, used to reach a point (n,m) is as follows (proceeding from (n,m) back along the path[1]):

$$\text{end point: } (n,m) \tag{2.7a}$$

$$1^{st} \text{ point back: } (n-\alpha_1^{(r)},m-\beta_1^{(r)}) \tag{2.7b}$$

$$2^{nd} \text{ point back: } (n-\alpha_1^{(r)}-\alpha_2^{(r)},m-\beta_1^{(r)}-\beta_2^{(r)}) \tag{2.7c}$$

$$\vdots$$

$$s^{th} \text{ point back: } (n-\sum_{l=1}^{s}\alpha_l^{(r)},m-\sum_{l=1}^{s}\beta_l^{(r)}) \tag{2.7d}$$

$$\vdots$$

$$\text{original point: } (n-\sum_{l=1}^{L(r)}\alpha_l^{(r)},m-\sum_{l=1}^{L(r)}\beta_l^{(r)}), \tag{2.7e}$$

or, in terms of the path functions of Eqs. (2.3),

$$k^{th} \text{ point: } i(k) = n,\ j(k) = m \tag{2.8a}$$

$$(k-s)^{th} \text{ point: } \begin{cases} i(k-s) = i(k) - \sum_{l=1}^{s}\alpha_l^{(r)} \\ j(k-s) = j(k) - \sum_{l=1}^{s}\beta_l^{(r)} \end{cases} \tag{2.8b}$$

for $s = 1,2,...,L(r)$.

As an example, the paths of Figure 2.4, part b, may be expressed by the three productions

$$P_1^{I} \rightarrow (1,0)(1,1) \tag{2.9a}$$

$$P_2^{I} \rightarrow (1,1) \tag{2.9b}$$

$$P_3^{I} \rightarrow (0,1)(1,1). \tag{2.9c}$$

[1] In time warping algorithms all paths are retrieved backwards from the end point (N,M) to $(1,1)$. Paths are retrieved backwards because the entire path is not determined until the end is reached.

An entire path from the point (1,1) to the point (N,M) can be expressed as a sequence of productions tracing a path back from (N,M) to (1,1). Figure 2.5 shows an example of the path defined by the sequence of productions, P_1^I P_1^I P_2^I P_1^I P_3^I P_2^I P_2^I (traced backwards from (N,M) to (1,1)). The actual path from (N,M) back to (1,1) is given by substitution from Eqs. (2.9) into the sequence of productions to yield the sequence (1,0) (1,1) (1,0) (1,1) (1,1) (1,0) (1,1) (0,1) (1,1) (1,1) (1,1).

Since $\alpha_l^{(r)}$ and $\beta_l^{(r)}$ are simply the local changes in a path, the time ordering restrictions of Eqs. (2.5) may be formulated as

$$\alpha_l^{(r)}, \beta_l^{(r)} \geqslant 0. \tag{2.10}$$

Also, restrictions on the degree of local compression and/or expansion can be incorporated into the local paths. The maximum and minimum amount of expansion (1/compression), denoted as $E_{\max}$ and $E_{\min}$, can be obtained as

$$E_{\max} = \max_{(r)} \left[\sum_{l=1}^{L(r)} \beta_l^{(r)} / \sum_{l=1}^{L(r)} \alpha_l^{(r)} \right] \tag{2.11a}$$

$$E_{\min} = \min_{(r)} \left[\sum_{l=1}^{L(r)} \beta_l^{(r)} / \sum_{l=1}^{L(r)} \alpha_l^{(r)} \right] \tag{2.11b}$$

For the paths of Figure 2.4b the maximum expansion is 2 and the minimum is 1/2.

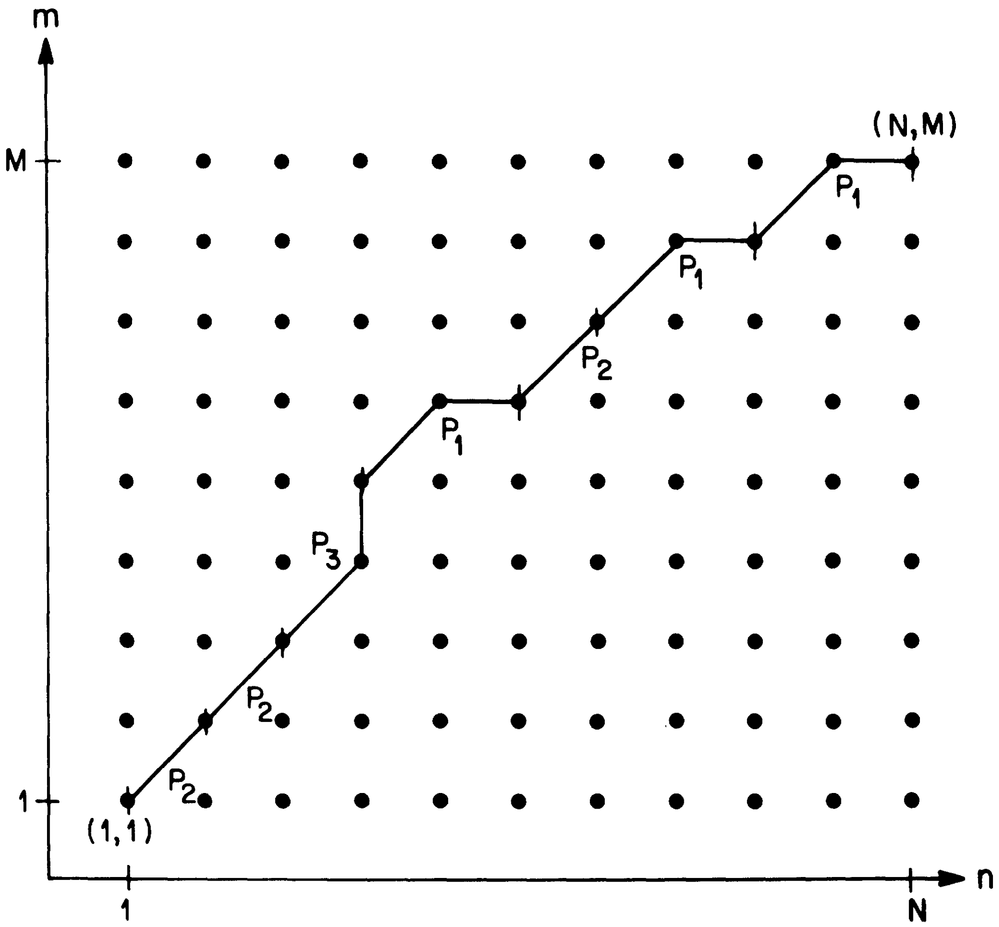
Two other local continuity constraints with the same maximum and minimum slope of 2 and 1/2 respectively are shown in Figure 2.6, parts a and b. The productions associated with these local constraints are as follows:

$$P_1^{II} \rightarrow (2,1) \tag{2.12a}$$

$$P_2^{II} \rightarrow (1,1) \tag{2.12b}$$

$$P_3^{II} \rightarrow (1,2) \tag{2.12c}$$

and

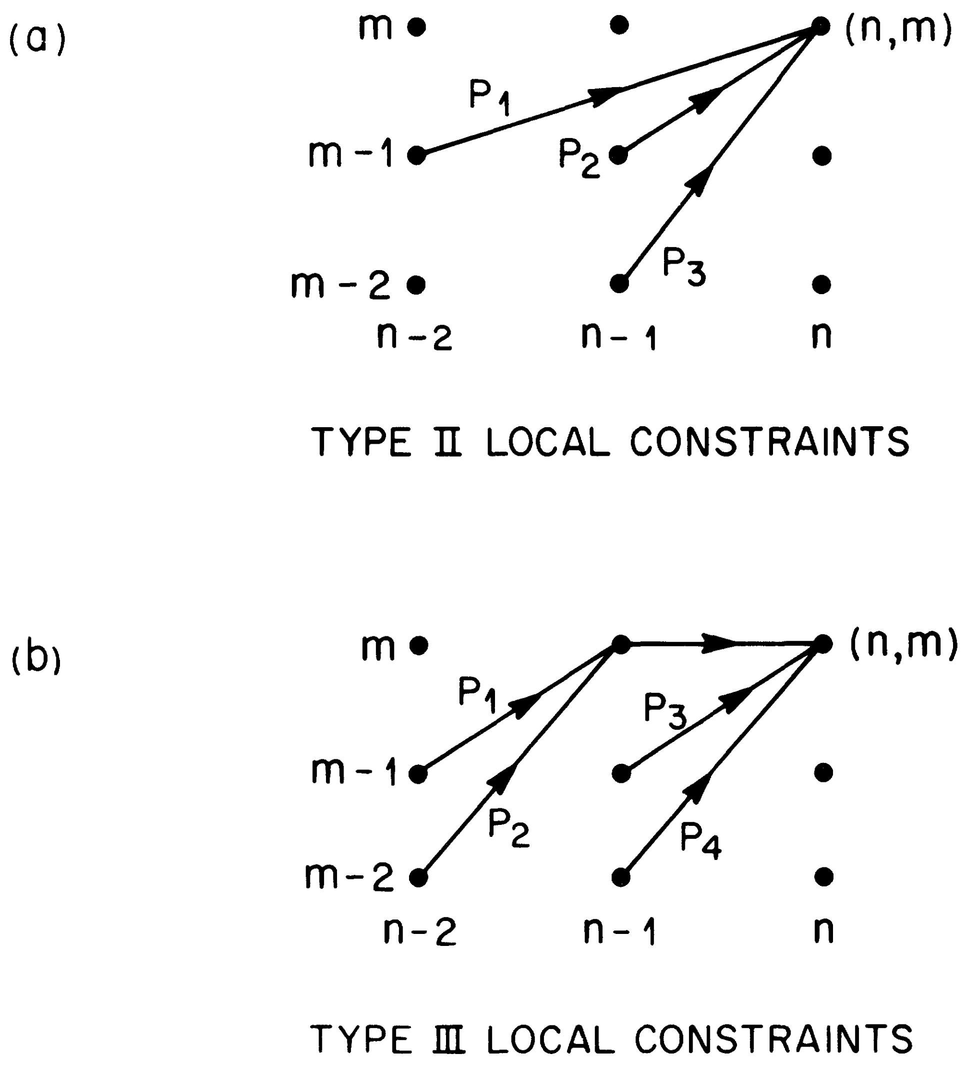


SAMPLE PATH (TYPE I CONSTRAINTS)

PRODUCTIONS: $P_1\, P_1\, P_2\, P_1\, P_3\, P_2\, P_2$

PATH: (1,0) (1,1) (1,0) (1,1) (1,1) (1,0) (1,1) (0,1) (1,1) (1,1) (1,1)

Fig. 2.5 Sample path.



TYPE II LOCAL CONSTRAINTS

TYPE III LOCAL CONSTRAINTS

Fig. 2.6 Local constraints types II and III.

$$P_1^{III} \rightarrow (1,0)(1,1) \qquad (2.13a)$$

$$P_2^{III} \rightarrow (1,0)(1,2) \qquad (2.13b)$$

$$P_3^{III} \rightarrow (1,1) \qquad (2.13c)$$

$$P_4^{III} \rightarrow (1,2). \qquad (2.13d)$$

Type II local constraints are similar to type I, i.e. symmetric and come from the same points as type I, but type II constraints are lacking in the use of intermediate points. Type III constraints are somewhat different in that they are assymetric, using intermediate points in the x direction but not in the y direction. (Type II constraints are a production rule version of the local constraints used by Itakura [6].) It should also be noted that all of the local constraints defined so far have a "memory" of two. That is, to reach a point (n,m) from a point (n',m') in one production it is necessary that $n - n' \leqslant 2$ and $m - m' \leqslant 2$, i.e. the original point is no more than two units away in either axis. As we shall see in Section 2.3, such a limited "memory" (i.e. not infinite) is important in the efficient implementation of a path finding algorithm.

The research undertaken in this thesis involved the use of all three of the local constraints defined thus far. For ease of reference these different local constraints are referred to as types I, II and III, corresponding to the productions of Eqs. (2.9), (2.12) and (2.13) respectively. It should be noted that type I local constraints are exactly those specified by Sakoe and Chiba using a P value of 1, corresponding to a maximum slope of 2 and a minimum slope of 1/2 [10]. Sakoe and Chiba found that this slope constraint was optimal and, for the most part, we will use only a maximum slope of 2 and a minimum slope of 1/2.

2.2.3 Global Path Constraints

Another factor in time warping for speech recognition is that of global range constraints. Global range constraints specify which points $(i(k),j(k))$ are allowed to occur within a legal path. These points constitute the global range. Global range constraints arise naturally as a result of local continuity constraints. Local continuity constraints force certain points of the (n,m) plane to be excluded from legal paths because they would require excessive expansion or compression of the time scales. For example, the point (2,10) would be illegal (under the local constraints described thus far) because it would require a 9 to 1 expansion to be reached from the point (1,1). The global constraints arising from local continuity constraints may be expressed as

$$1 + \frac{(i(k)-1)}{E_{\max}} \leqslant j(k) \leqslant 1 + E_{\max}(i(k)-1) \tag{2.14a}$$

$$M + E_{\max}(i(k)-N) \leqslant j(k) \leqslant M + \frac{(i(k)-N)}{E_{\max}} \tag{2.14b}$$

where $E_{\max}$ is the maximum allowable expansion (and $E_{\min}=1/E_{\max}$). The first constraint of Eq. (2.14) may be interpreted as restricting the range of legal points to those which do not require excessive expansion or compression in order that they be reached from the point (1,1). The second constraint of Eq. (2.14) eliminates those points which would necessitate excessive expansion or compression in order to eventually reach the point (N,M). One useful way to view these constraints is as limiting all legal paths to fall within the bounds of the parallelogram of Figure 2.7. The size of this region depends strongly on the values of N and M. Figure 2.8 shows the affects of the successive nature of N/M of 1, 3/2 and 2 on the global range of paths for a value of maximum slope of $E_{\max} = 2$. The range of legal paths decreases quickly with increasing N/M (or M/N). These effects are often significant in speech recognition systems in which there is a high variability among replications of the same utterance (i.e. M/N approaches or exceeds 2, or falls below 1/2).

Another possible restriction on the global range has been proposed by Sakoe and Chiba [4]. They proposed that the absolute difference, $|i(k)-j(k)|$, be limited to be less than or equal to some integer value, R:

$$|i(k)-j(k)| \leqslant R. \tag{2.15}$$

This type of restriction may be interpreted as imposing a limit on the absolute time difference which can be allowed between frames, i.e. frame $i(k)$ of the reference pattern is restricted to fall within $R \cdot T_s$ seconds of frame $j(k)$ of the test pattern, where T_s is the sampling period for frames (typically 10 to 30 milliseconds). Pictorially, the restriction of Eq. (2.15) cuts

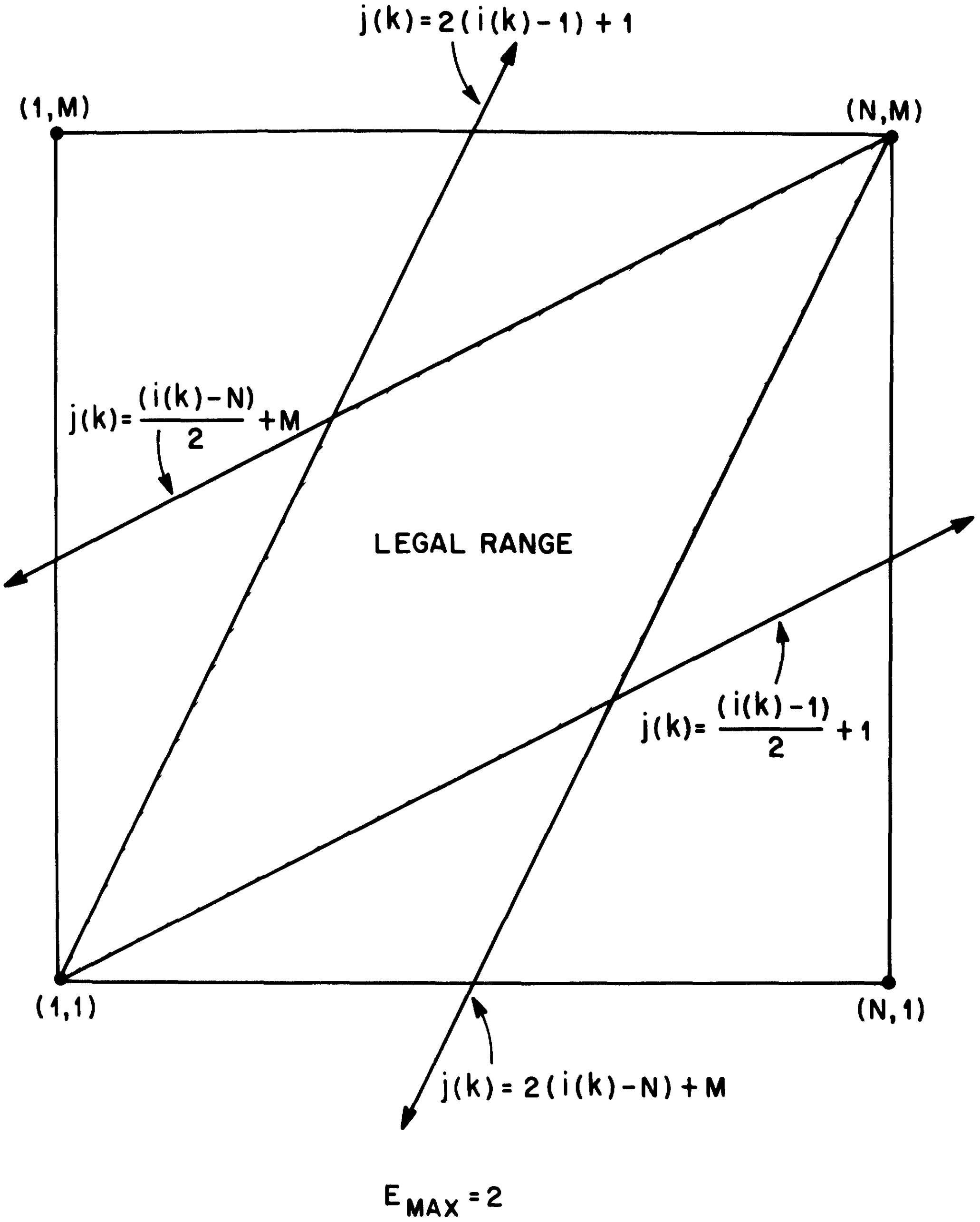


Fig. 2.7 Global range for paths.

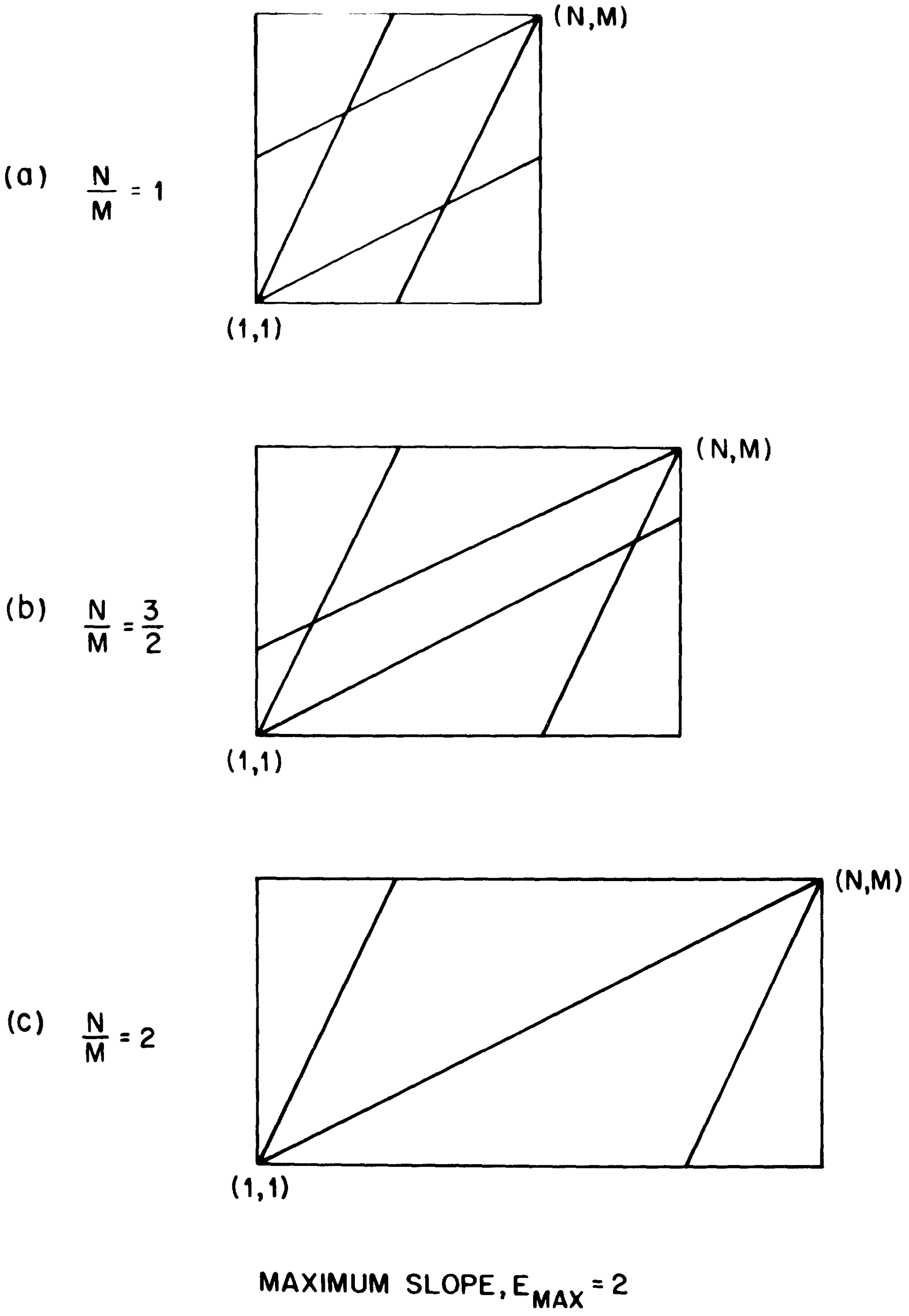


Fig. 2.8 Global range for paths as a function of N/M.

off the corners of the parallelogram of Figure 2.7, as illustrated in Figure 2.9.

For notational purposes we will refer to those time warping algorithms which use an absolute time difference range constraint as range - limited algorithms and those which do not use Eq. (2.15) as a range constraint as range - unlimited algorithms.

2.2.4 Axis Orientation

Axis orientation is another important consideration in a time warping algorithm. Axis orientation determines if the functions of Eqs. (2.3) are used or if the inverse set of equations is used, i.e.

$$n = j(k), \quad k = 1,2,...,K \tag{2.16a}$$

$$m = i(k), \quad k = 1,2,...,K \tag{2.16b}$$

The differences between Eqs. (2.3) and Eqs. (2.16) can be important when the local constraints are not symmetric, as with type III local constraints, or when the distance function for determining an optimal path is not symmetric. In general, we will refer to the paths of Eqs. (2.3) as "reference along the x-axis," as in Figure 2.3, and those paths of Eqs. (2.16) as "test along the x-axis." (For convenience, we will use Eqs. (2.3) in all our discussions, unless otherwise noted.)

2.2.5 Distance Measures

The final consideration in the specification of a time warping algorithm is a distance function which is used to determine the optimal path. A typical distance function has the form

$$D(i(k),j(k)) = \frac{\sum_{k=1}^{K} d(i(k),j(k))\, \tilde{W}(k)}{\mathrm{N}(\tilde{W})} \tag{2.17}$$

where $D(i(k),j(k))$ is the total distance along the path of length K (i.e. $K-1$ arcs or K pairs $(i(k),j(k))$), defined by the functions $i(k)$ and $j(k)$.[2] The overall distance is given as a normalized, weighted sum of local distances where $d(i(k),j(k))$ is the value of the local distance

[2] Technically, $D(i(k),j(k))$ is a functional, that is, a function of a set of functions, $i(k)$ and $j(k)$, but, for sake of simplicity we will refer to it as a function.

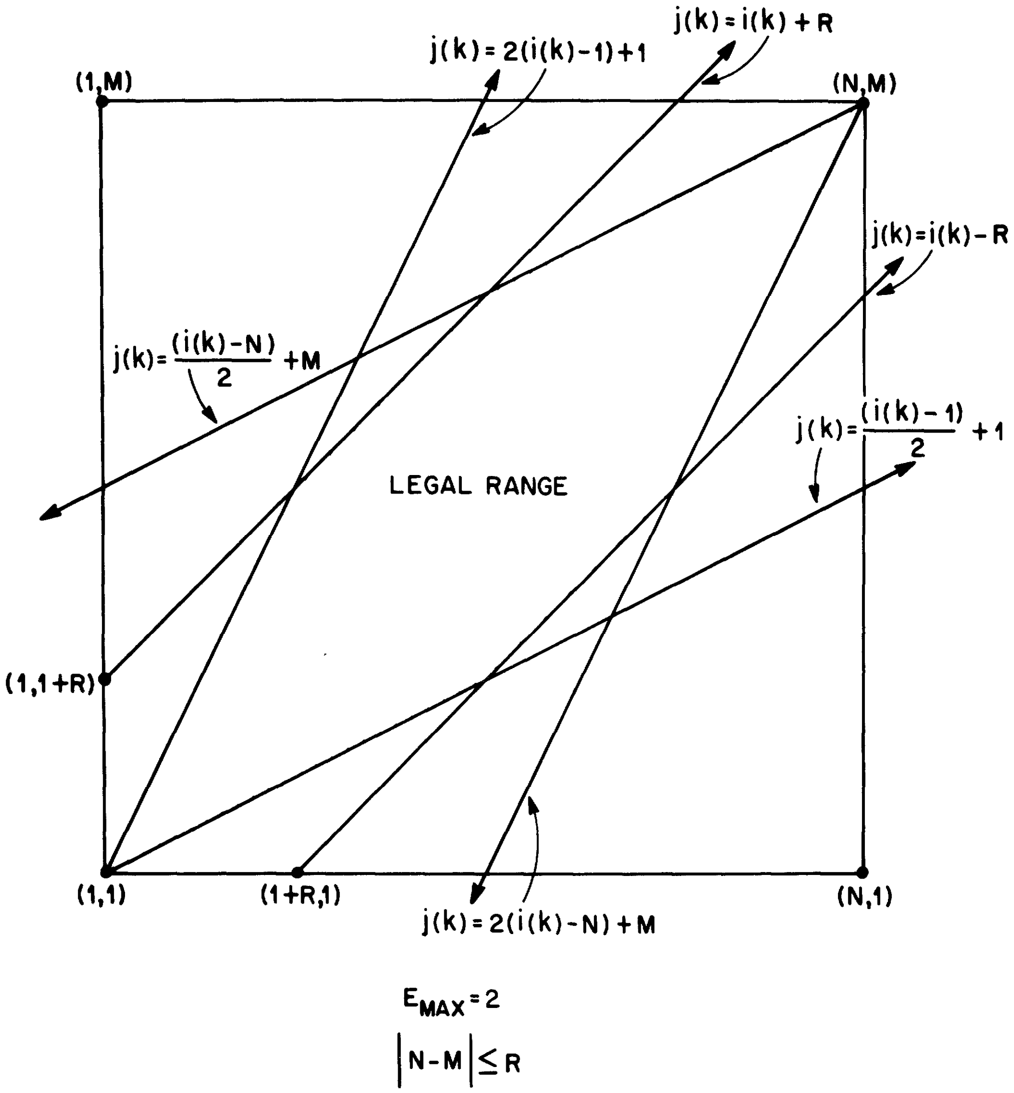


Fig. 2.9 Global range for paths with range limiting.

metric at frames i(k) of the reference and $j(k)$ of the test, $\tilde{W}(k)$ is a set of weights and $\mathrm{N}(\tilde{W})$ is a normalization factor which, in general, depends on the weighting function used. The best path is the set of functions $i(k)$, $j(k)$, $k = 1,2,...,K$ which minimize the distance function. The total distance between a test and a reference, $\hat{D}$, is defined to be the minimum distance achieved by the time warping algorithm, i.e.

$$\hat{D} = \min_{(K,i(k),j(k))} (D(i(k),j(k))). \tag{2.18}$$

To define the total distance function we must define $d(i(k),j(k))$, $\tilde{W}(k)$ and $\mathrm{N}(\tilde{W})$. Choice of the local distance metric, $d(i(k),j(k))$ is dependent on the feature set used to create both the test and the reference patterns. Typical choices include a log spectral difference for energy measurements, a Euclidean distance for cepstral coefficients and a log likelihood ratio for LPC coefficients [6]. Thus, the local distance metric is independent of the particular time warping algorithm. The weighting function and the normalization are not, however, independent of the time warping algorithm.

Typically, a weighting function depends only on the local paths. For example, the weight used on the path from the point $(i(k-1),j(k-1))$ to the point $(i(k),j(k))$ depends only on $i(k) - i(k-1)$ and $j(k) - j(k-1)$. Typical weighting functions which are used include

$$\tilde{W}(k) = \min(i(k)-i(k-1),j(k)-j(k-1)) \quad \text{(type } a\text{)} \tag{2.19a}$$

$$\tilde{W}(k) = \max(i(k)-i(k-1),j(k)-j(k-1)) \quad \text{(type } b\text{)} \tag{2.19b}$$

$$\tilde{W}(k) = i(k) - i(k-1) \quad \text{(type } c\text{)} \tag{2.19c}$$

$$\tilde{W}(k) = i(k) - i(k-1) + j(k) - j(k-1) \quad \text{(type } d\text{)} . \tag{2.19d}$$

Weighting functions c and d have been proposed by Sakoe and Chiba [10] as weighting all the samples of the x-axis pattern equally (type c) or as weighting all samples of both the x and y-axis patterns equally (type d). Weighting function a weights all segments of a path equally, regardless of their length and weighting function b weighs shorter segments less than longer segments. As we shall see, weighting functions c and d have no particular bias in their choice of paths, weighting function a favors longer paths over shorter paths and weighting function b favors shorter paths over longer paths. For initialization purposes, $i(0)$ and $j(0)$ are defined to be 0 and thus $\tilde{W}(1) = 1$ for weighting functions a, b and c and $\tilde{W}(1) = 2$ for weighting function d.

A pictorial representation of these various weighting functions as applied to type II paths is given in Figure 2.10. The number labeling the various arcs are the weighting functions associated with paths that lie along those particular arcs. Figure 2.11 shows two different representations of the various weighting functions as used with type I paths. The left hand column uses the weighting functions exactly as defined. In the right hand column a smoothing process has been applied. Smoothing is a procedure in which multiple segment local paths have their weighting functions averaged. This process was first used by Sakoe and Chiba [10] to prevent certain anomolies such as the left hand side of Figure 2.11, part a, in which some arcs have a zero weight thus allowing a loss of information from local distances which are ignored.

In Figure 2.12 we show an example of a typical type II path with a type d weighting function. In this example the arcs along the path are labeled with the corresponding weighting functions, the values of N, M and K are 11, 9 and 8 respectively and the overall distance is given by

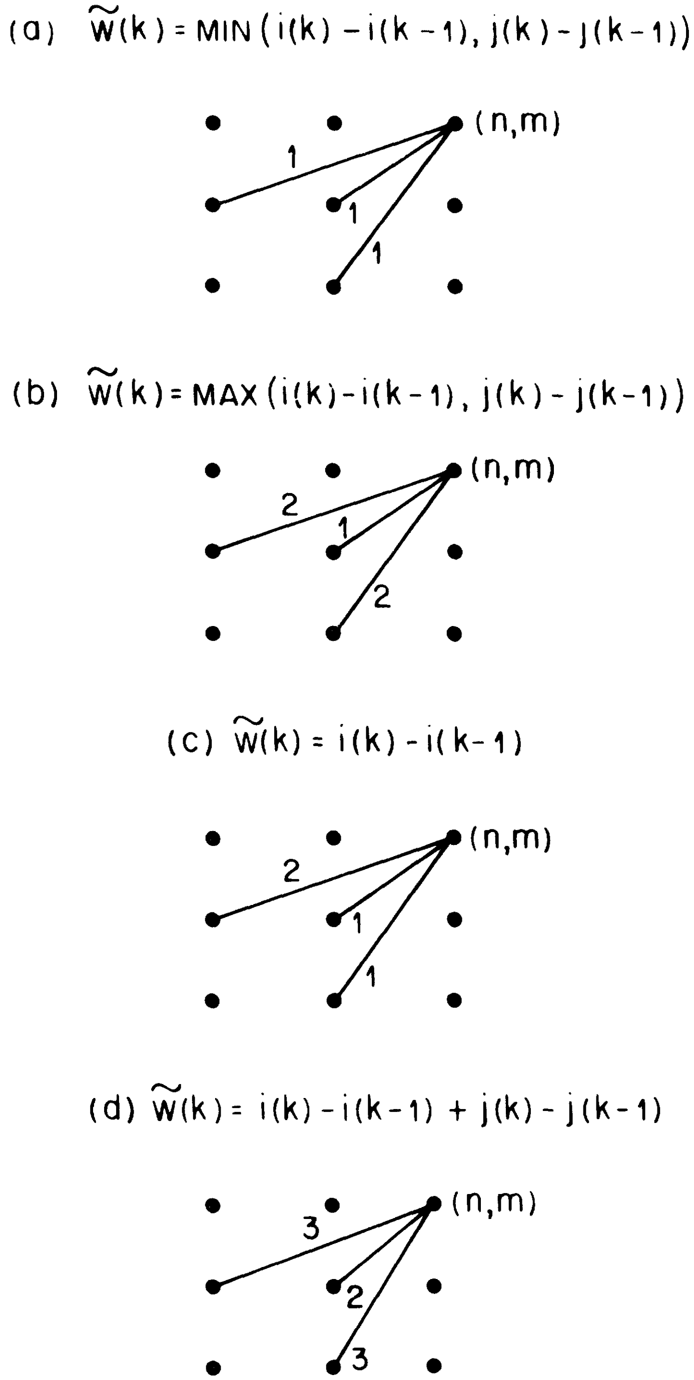
$$D(i(k),j(k)) = [2d(1,1)+3d(3,2)+2d(4,3)+3d(5,5) \qquad (2.20)$$
$$+ 3d(7,6)+3d(9,7)+2d(10,8)+2d(11,9)]/\mathrm{N}(\tilde{W}_d)$$

where $\mathrm{N}(\tilde{W}_d)$ is the normalization associated with weighting function d.

The choice of $\mathrm{N}(\tilde{W})$ is typically made such that $D(i(k),j(k))$ is an average local distance along the path defined by the functions $i(k)$ and $j(k)$. As such, the natural choice for $\mathrm{N}(\tilde{W})$ is the sum

$$\mathrm{N}(\tilde{W}) = \sum_{k=1}^{K} \tilde{W}(k). \qquad (2.21)$$

For weighting functions c and d this definition leads to very simple normalization, namely,



(a) $\tilde{w}(k) = \mathrm{MIN}\,(i(k) - i(k-1),\ j(k) - j(k-1))$

(b) $\tilde{w}(k) = \mathrm{MAX}\,(i(k) - i(k-1),\ j(k) - j(k-1))$

(c) $\tilde{w}(k) = i(k) - i(k-1)$

(d) $\tilde{w}(k) = i(k) - i(k-1) + j(k) - j(k-1)$

Fig. 2.10 Weighting functions for type II paths.

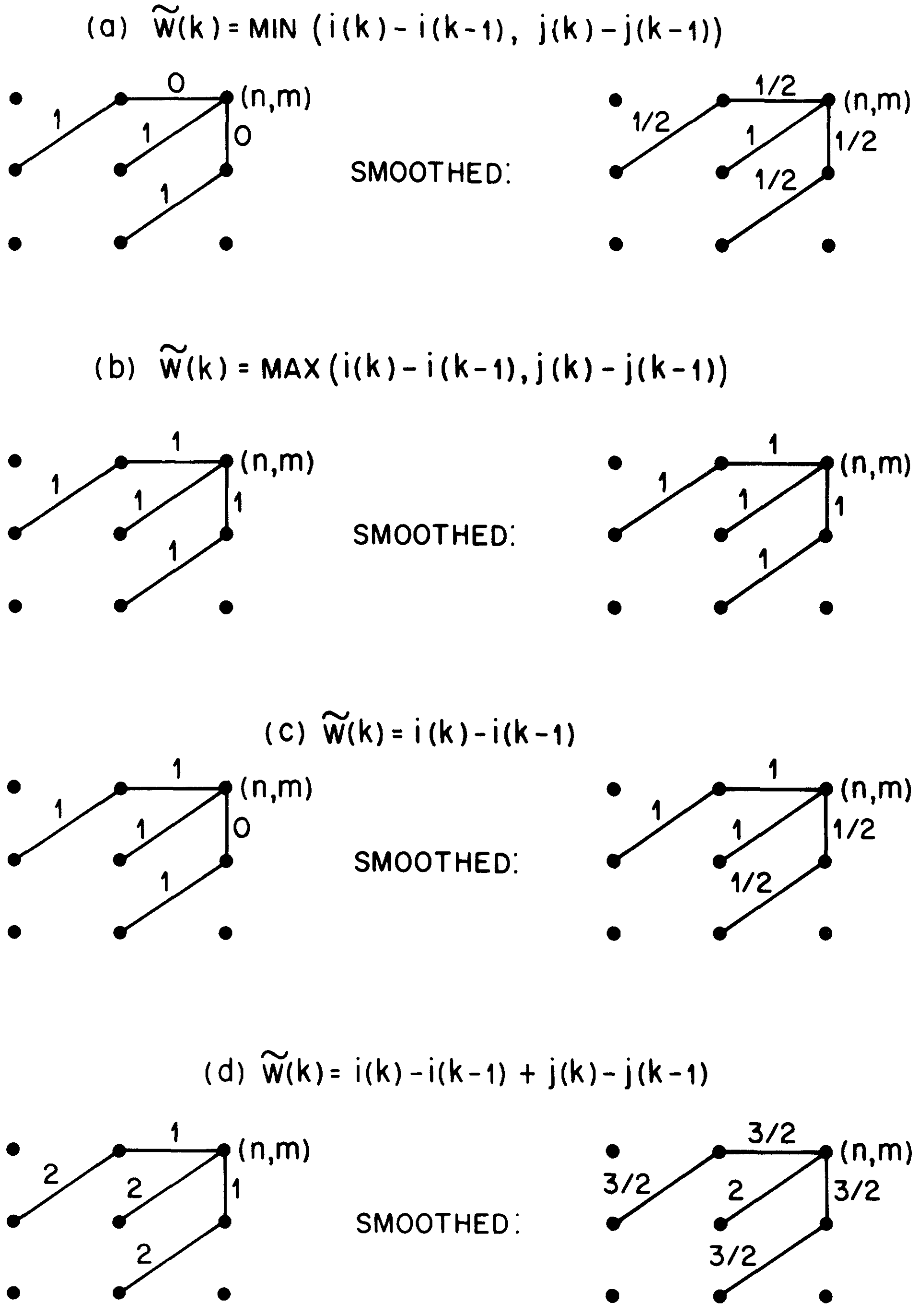
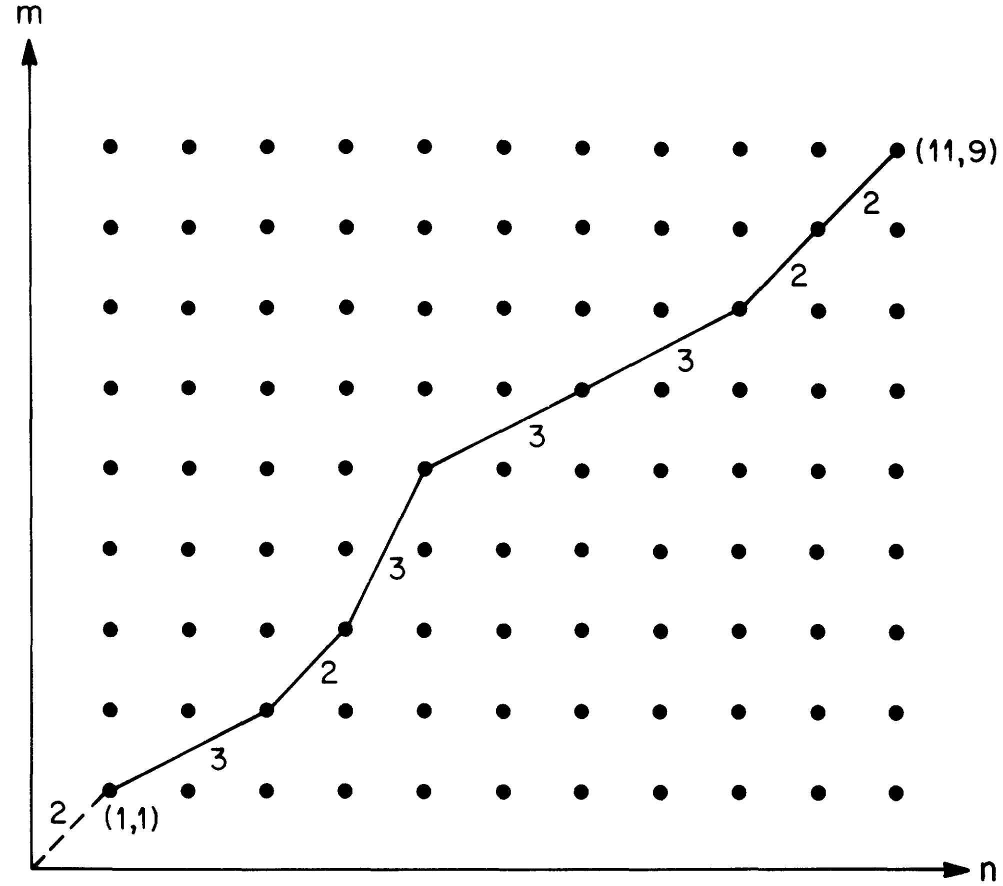

(a) $\tilde{w}(k) = \text{MIN}\,(i(k) - i(k-1),\ j(k) - j(k-1))$
0
(n,m)
1
1
0
1
SMOOTHED:
1/2
(n,m)
1/2
1
1/2
1/2
(b) $\tilde{w}(k) = \text{MAX}\,(i(k) - i(k-1), j(k) - j(k-1))$
1
(n,m)
1
1
1
1
SMOOTHED:
1
(n,m)
1
1
1
1
(c) $\tilde{w}(k) = i(k) - i(k-1)$
1
(n,m)
1
1
0
1
SMOOTHED:
1
(n,m)
1
1
1/2
1/2
(d) $\tilde{w}(k) = i(k) - i(k-1) + j(k) - j(k-1)$
1
(n,m)
2
2
1
2
SMOOTHED:
3/2
(n,m)
3/2
2
3/2
3/2

Fig. 2.11 Weighting functions for type I paths.



N = 11, M = 9 K = 8

TYPE II PATHS

$$\widetilde{W}(k) = i(k) - i(k-1) + j(k) - j(k-1)$$

$$D(i(k), j(k)) = [2d(1,1) + 3d(3,2) + 2d(4,3) + 3d(5,5) + 3d(7,6) + 3d(9,7) + 2d(10,8) + 2d(11,9)] / N(\widetilde{W}_d)$$

Fig. 2.12 Sample path with distance measure.

$$\mathrm{N}(\tilde{W}_c) = \sum_{k=1}^{K}(i(k)-i(k-1)) = i(K) - i(0) = N \tag{2.22a}$$

$$\mathrm{N}(\tilde{W}_d) = \sum_{k=1}^{K}(i(k)-i(k-1)+j(k)-j(k-1)) \tag{2.22b}$$

$$= i(K) - i(0) + j(K) - j(0) = N + M.$$

However, for weighting functions a and b the value of Eq. (2.21) is not a constant, but instead depends upon the path chosen. Figure 2.13 shows two simple paths, Path 1 and Path 2, for the case $N = M$. Path 1 is the straight line path of slope = 1 from (1,1) to (N,M). Path 2 has two sections, the first of which has a slope = 1/2 and the second of which has a slope = 2. If $\mathrm{N}(\tilde{W})$ is defined as in Eq. (2.21) then the values of $\mathrm{N}(\tilde{W})$ for weighting function a are given approximately by

$$\text{Path 1: } \mathrm{N}(\tilde{W}_a) = N \tag{2.23a}$$

$$\text{Path 2: } \mathrm{N}(\tilde{W}_a) = \frac{2N}{3}. \tag{2.23b}$$

The value of Eq. (2.23a) is generated by N segments of weight 1 and the value of Eq. (2.23b) is generated as $2N/3$ segments of slope 1/2 (average value of $\tilde{W}_a(k)=1/2$) and $N/3$ segments of slope 2 (average value of $\tilde{W}_a(k)=1$) for a total of $N(\tilde{W}_a)=1/2 \cdot 2N/3 + 1 \cdot N/3 = 2N/3$. For weighting function b the values of $\mathrm{N}(\tilde{W})$ as defined by Eq. (2.21) are given in an analogous manner by

$$\text{Path 1: } \mathrm{N}(\tilde{W}_b) = N \tag{2.24a}$$

$$\text{Path 2: } \mathrm{N}(\tilde{W}_b) = \frac{4N}{3}. \tag{2.24b}$$

In section 2.3 we will show that, in order to solve for the optimal path efficiently, it is necessary that $\mathrm{N}(\tilde{W})$ be independent of path. Thus, for computational convenience, we define $\mathrm{N}(\tilde{W})$ as follows:

$$\mathrm{N}(\tilde{W}_a) = N \tag{2.25a}$$

$$\mathrm{N}(\tilde{W}_b) = N \tag{2.25b}$$

$$\mathrm{N}(\tilde{W}_c) = N \tag{2.25c}$$

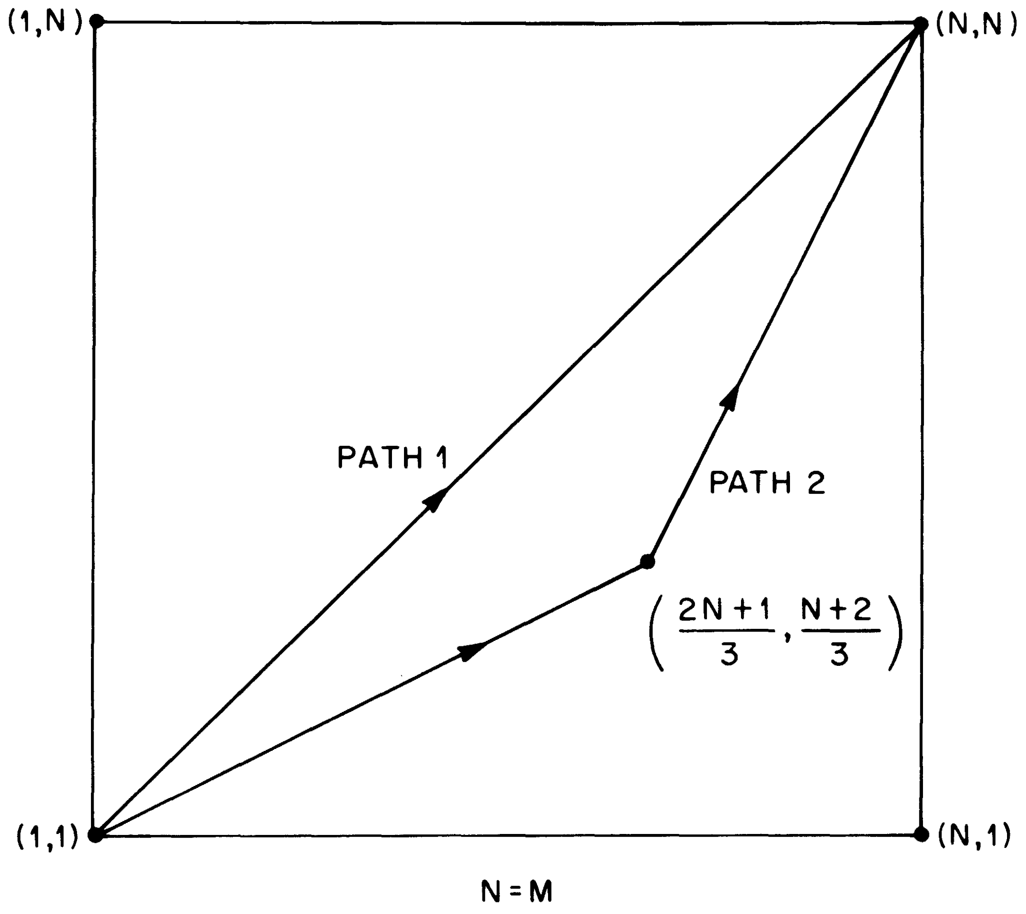


Fig. 2.13 Two different paths.

$$\mathrm{N}(\tilde{W}_d) = N + M. \tag{2.25d}$$

An important consideration of the performance of a time warping algorithm arises from the definitions of Eqs. (2.25). Use of these definitions can create situations in which certain paths are favored over others. An algorithm in which such a situation can arise is said to be biased. Formally, a time warping algorithm is unbiased when the following condition is true:

$$\text{if } d(i(k),j(k)) = \bar{d} \text{ then } D(i(k),j(k)) = \bar{d}, \tag{2.26}$$

i.e. if the local distance is independent of $i(k)$ and $j(k)$ then the global distance function is also independent of $i(k)$ and $j(k)$. Equivalently, if the local distance is independent of the path chosen then there is no preferred path to be chosen.

By direct substitution of $d(i(k),j(k)) = \bar{d}$ into Eq. (2.17) we obtain

$$D(i(k),j(k)) = \frac{\bar{d} \sum_{k=1}^{K} \tilde{W}(k)}{N(\tilde{W})} . \tag{2.27}$$

Thus, condition (2.26) is true if and only Eq. (2.21) is true, and is thus true for weighting functions c and d by Eqs. (2.22). However, condition (2.26) is not true for weighting functions a and b. Using the paths of Figure 2.13 we get the following values for $D(i(k),j(k))$ using weighting function a,

$$\text{Path 1: } D(i(k),j(k)) = \bar{d} \tag{2.28a}$$

$$\text{Path 2: } D(i(k),j(k)) = \frac{2\bar{d}}{3} \tag{2.28b}$$

and using weighting function b,

$$\text{Path 1: } D(i(k),j(k)) = \bar{d} \tag{2.28c}$$

$$\text{Path 2: } D(i(k),j(k)) = \frac{4\bar{d}}{3}. \tag{2.28d}$$

Thus, we see that weighting function a has a preference for the longer path, Path 2 over Path 1 (lower global distance function) and that weighting function b has a preference for the shorter path, Path 1 over Path 2. This form of bias may be expected to be detrimental to the performance of a time warping algorithm because bias may prevent the time warping algorithm from following the truely accurate path.[3]

2.2.6 Solution for the Optimal Path

In Tables 2.1, 2.2, 2.3 and 2.4 we summarize the various features of a time warping algorithm as we have presented them in the previous sections. Once all of the factors for a time warping algorithm have been specified, the optimal path under these conditions may be found. Several methods exist for finding this optimal path. One method is exhaustive search of all possible paths from (1,1) to (N,M). This is, in general, prohibitively expensive and time consuming (on the order of 2^N operations). Another approach would be to apply Dykstra's path finding algorithm to the problem [11]. While faster than exhaustive search methods, this method also can be very time consuming for large values of N and M (order of N^2M^2 operations). A better approach was suggested by Sakoe and Chiba [4]. They proposed the use of dynamic programming to efficiently solve the problem. Dynamic programming is an efficient method to apply because dynamic programming successively builds longer optimal paths from smaller optimal paths [5] (order of NM operations). In the next section we discuss how dynamic programming is applied to our problem.

2.3 Dynamic Time Warping

Two basic principles are involved in dynamic programming as applied to time warping, or dynamic time warping (DTW), as it is referred to. The first principle is that a globally optimal path is also locally optimal. The globally optimal path is the path which minimizes the weighted distance from (1,1) to (N,M) according to Eq. (2.18). A locally optimal path from a point (n',m') to a point (n,m) is the path which minimizes the weighted distance from (n',m') to (n,m). To say that the globally optimal path is also locally optimal is equivalent to the statement that for any two points (n',m') and (n,m) along the globally optimal path, the locally optimal path from (n',m') to (n,m) is exactly the subsection of the globally optimal path from (n',m') to (n,m). This must be true because, if there were a better locally optimal path from

[3] An examination of the literature on speech recognition reveals that White and Neely [3] used weighting function b. Such a situation may account for the lack of improvement in recognition scores using a nonlinear time warping as compared to a linear time warping for the alphabet-digits vocabulary.

LOCAL CONSTRAINTS

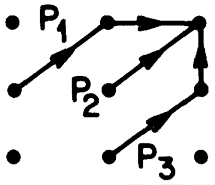
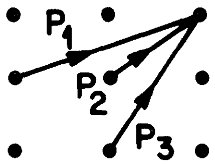
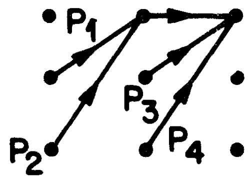
TYPE	PICTORIAL	PRODUCTIONS
I		$P_1 \rightarrow (1,0)(1,1)$ $P_2 \rightarrow (1,1)$ $P_3 \rightarrow (0,1)(1,1)$
II		$P_1 \rightarrow (2,1)$ $P_2 \rightarrow (1,1)$ $P_3 \rightarrow (1,2)$
III		$P_1 \rightarrow (1,0)(1,1)$ $P_2 \rightarrow (1,0)(1,2)$ $P_3 \rightarrow (1,1)$ $P_4 \rightarrow (1,2)$

Table 2.1

Local Constraints

Endpoint Constraints

$$i(1) = 1, \quad i(K) = N$$

$$j(1) = 1, \quad j(K) = M$$

Global Constraints

From Local Constraints:

$$\frac{(i(k)-1)}{E_{\max}} + 1 \leqslant j(k) \leqslant E_{\max}(i(k)-1) + 1$$

$$E_{\max}(i(k)-N) + M \leqslant j(k) \leqslant \frac{(i(k)-N)}{E_{\max}} + M$$

$E_{\max}$ - Maximum Slope

Range Limited: $i(k) - R \leqslant j(k) \leqslant i(k) + R$

Table 2.2

Endpoint and Global Constraints

Axis Orientation

Reference along X-axis: $n = i(k),\ m = j(k)$
Test along X-axis: $n = j(k),\ m = i(k)$

Overall Distance Measure

$$\hat{D} = \min_{(i(k),j(k),K)} \left[\frac{\sum_{k=1}^{K} d(i(k),j(k))\,\tilde{W}(k)}{\mathrm{N}(\tilde{W})} \right]$$

Table 2.3

Axis Orientation and Distance Measure

Weighting Functions

Type	Definition	Normalization $N(\tilde{W})$
a	$\tilde{W}(k) = \min(i(k)-i(k-1),$ $j(k)-j(k-1))$	N
b	$\tilde{W}(k) = \max(i(k)-i(k-1),$ $j(k) - j(k-1)$	N
c	$\tilde{W}(k) = i(k) - i(k-1)$	N
d	$\tilde{W}(k) = i(k) - i(k-1)$ $+ j(k) - j(k-1)$.	$N + M$

Table 2.4

Weighting Functions

(n',m') to (n,m) it could be substituted into the globally optimal path with a corresponding improvement in the globally optimal path.

The other principle involved in a dynamic programming implementation of a time warping algorithm is the dependence of the best path to a point (n,m) only on (n',m') such that

$$n' \leqslant n \tag{2.29a}$$

$$m' \leqslant m. \tag{2.29b}$$

This follows from the monotonicity restriction of Eqs. (2.5).

As a result of these two principles, it is possible to create a partial accumulated distance function $D_A(n,m)$. $D_A(n,m)$ is the accumulated distance from the point (1,1) to the point (n,m) using the best possible path to reach (n,m), i.e.

$$D_A(n,m) = \min_{(i(k),j(k),K')} \left[\sum_{k=1}^{K'} d(i(k),j(k)) \tilde{W}(k) \right] \tag{2.30}$$

where K' is the length of the path from (1,1) to (n,m) and where

$$i(1) = 1, \quad i(K') = n \tag{2.31a}$$

$$j(1) = 1, \quad j(K') = m. \tag{2.31b}$$

Since $D_A(n,m)$ depends only on the paths from (1,1) to (n,m) and since the optimal path to (n,m) depends only on those points (n',m') which satisfy Eqs. (2.29), $D_A(n,m)$ can be defined recursively in terms of (n',m') by

$$D_A(n,m) = \min_{(n',m')} \left[D_A(n',m') + \hat{d}((n',m'),(n,m)) \right] \tag{2.32a}$$

$$D_A(1,1) = d(1,1) \tilde{W}(1) \tag{2.32b}$$

where $\hat{d}((n',m'),(n,m))$ is the weighted distance from (n',m') to (n,m), i.e.

$$\hat{d}((n',m'),(n,m)) = \sum_{l=0}^{L-1} d(i(K'-l),j(K'-l)) \tilde{W}(K'l) \tag{2.33}$$

where L is the number of segments in the path from (n',m') to (n,m) and where

$$i(K') = n, \quad i(K'-L) = n' \qquad (2.34a)$$

$$j(K') = m, \quad j(K'-L) = m'. \qquad (2.34b)$$

For a given set of local constraints it is possible to restrict the range of (n',m'), for a given (n,m) to only those (n',m') which use a single production to reach (n,m) from (n',m'). For example, the type II paths of Figure 2.6b restrict the range as follows:

$$(n',m') \in \{(n-1,m-1),\ (n-1,m-2),\ (n-2,m-1)\}. \qquad (2.35)$$

Thus, Eqs. (2.32) may be interpreted as building up paths to a point (n,m) via application of production rules to that point and minimizing the overall distance to that point. A simple proof that Eqs. (2.32) give the best distance to all points, (n,m) may be given by two dimensional induction on the grid of legal points as follows:

1. For the initial point (1,1) the shortest path to it is just the point itself and the best distance is given by Eq. (2.32b).

2. Assume that, for any point (n,m), $D_A(n',m')$ is the distance of best path to a point (n',m'), $n'+m' < n+m$. Then, since the best path to (n,m) is given by a path from some point (n',m') s.t. $n'+m' < n+m$, (as generated by a production rule), and since the distance of this path is given by some of the distances from (1,1) to (n',m') and from (n',m') to (n,m), then Eq. (2.32a) will give the best distance to the point (n,m).

An example of the $\hat{d}$ function for a type II local constraint with a weighting function $\tilde{W}(k) = i(k) - i(k-1)$ over the range of (n',m') given by Eq. (2.35) is given by

$$\hat{d}((n-1,m-1),(n,m)) = d(n,m) \qquad (2.36a)$$

$$\hat{d}((n-1,m-2),(n,m)) = d(n,m) \qquad (2.36b)$$

$$\hat{d}((n-2,m-1),(n,m)) = 2d(n,m). \qquad (2.36c)$$

Combining this definition of the function *dis* with the definition of $D_A(n,m)$ in Eq. (2.32a) we obtain the following recursive definition for $D_A(n,m)$ when type II paths with weighting function c are used,

$$D_A(n,m) = \min \begin{Bmatrix} D_A(n-1,m-1)+d(n,m), \\ D_A(n-1,m-2)+d(n,m), \\ D_A(n-2,m-1)+2d(n,m) \end{Bmatrix}. \tag{2.37}$$

Further examples are given in Figure 2.14.

With a partial function of the form of Eq. (2.32a) it is possible to solve the minimization problem of Eq. (2.18) when the following condition is true,

$$\hat{D} = \min_{(i(k),j(k),K)} \left[\frac{\sum_{k=1}^{K} d(i(k),j(k))\, \tilde{W}(k)}{\mathrm{N}(\tilde{W})} \right]$$

$$= \frac{\min_{(i(k),j(k),K)} \sum_{k=1}^{K} d(i(k),j(k))\, \tilde{W}(k)}{\mathrm{N}(\tilde{W})} . \tag{2.38}$$

This condition states that the normalization function is independent of the path chosen, or, equivalently, a solution to the unnormalized minimization problem provides a solution to normalized minimization problem. In the previous section we defined $\mathrm{N}(\tilde{W})$ in Eqs. (2.38) so that condition (2.38) would be satisfied. Given $\mathrm{N}(\tilde{W})$ independent of the path, Eq. (2.38) becomes

$$\hat{D} = \frac{D_A(N,M)}{\mathrm{N}(\tilde{W})} . \tag{2.39}$$

Thus, since $D_A(n,m)$ is easy to compute recursively, it is possible to compute $\hat{D}$ as follows:

1. set $D_A(1,1) = d(1,1)\,\tilde{W}(1)$

2. compute $D_A(n,m)$ recursively for

$$1 \leqslant n \leqslant N, \quad 1 \leqslant m \leqslant M$$

3. $\hat{D} = D_A(N,M)/\mathrm{N}(\tilde{W})$.

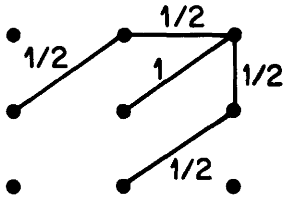
This is a great savings in computation as compared to either exhaustive search or Dykstra's algorithm.

Figure 2.15 shows the results of a typical application of a DTW algorithm. Type II local constraints and weighting function c were used. Thus, Eq. (2.36) is the appropriate dynamic

(a) TYPE I CONSTRAINTS

$\widetilde{W}(k) = \text{MIN}\,(i(k)-i(k-1),\ j(k)-j(k-1))$

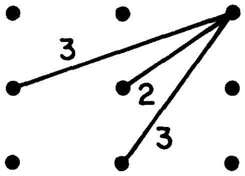
SMOOTHED



$$D_A(n,m) = \text{MIN}\left(D_A(n-1,m-1) + d(n,m), D_A(n-1,m-2) + 1/2\,[d(n,m-1) + d(n,m)], D_A(n-2,m-1) + 1/2\,[d(n-1,m) + d(n,m)]\right)$$

(b) TYPE II CONSTRAINTS

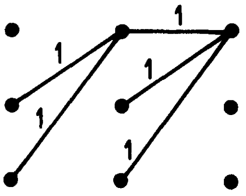
$\widetilde{W}(k) = i(k) - i(k-1) + j(k) - j(k-1)$



$$D_A(n,m) = \text{MIN}\left(D_A(n-1,m-1) + 2d(n,m), D_A(n-1,m-2) + 3d(n,m), D_A(n-2,m-1) + 3d(n,m)\right)$$

(c) TYPE III CONSTRAINTS

$\widetilde{W}(k) = i(k) - i(k-1)$



$$D_A(n,m) = \text{MIN}\left(D_A(n-1,m-1) + d(n,m), D_A(n-1,m-2) + d(n,m), D_A(n-2,m-1) + d(n-1,m) + d(n,m), D_A(n-2,m-2) + d(n-1,m) + d(n,m)\right)$$

Fig. 2.14 Sample accumulated distance functions.

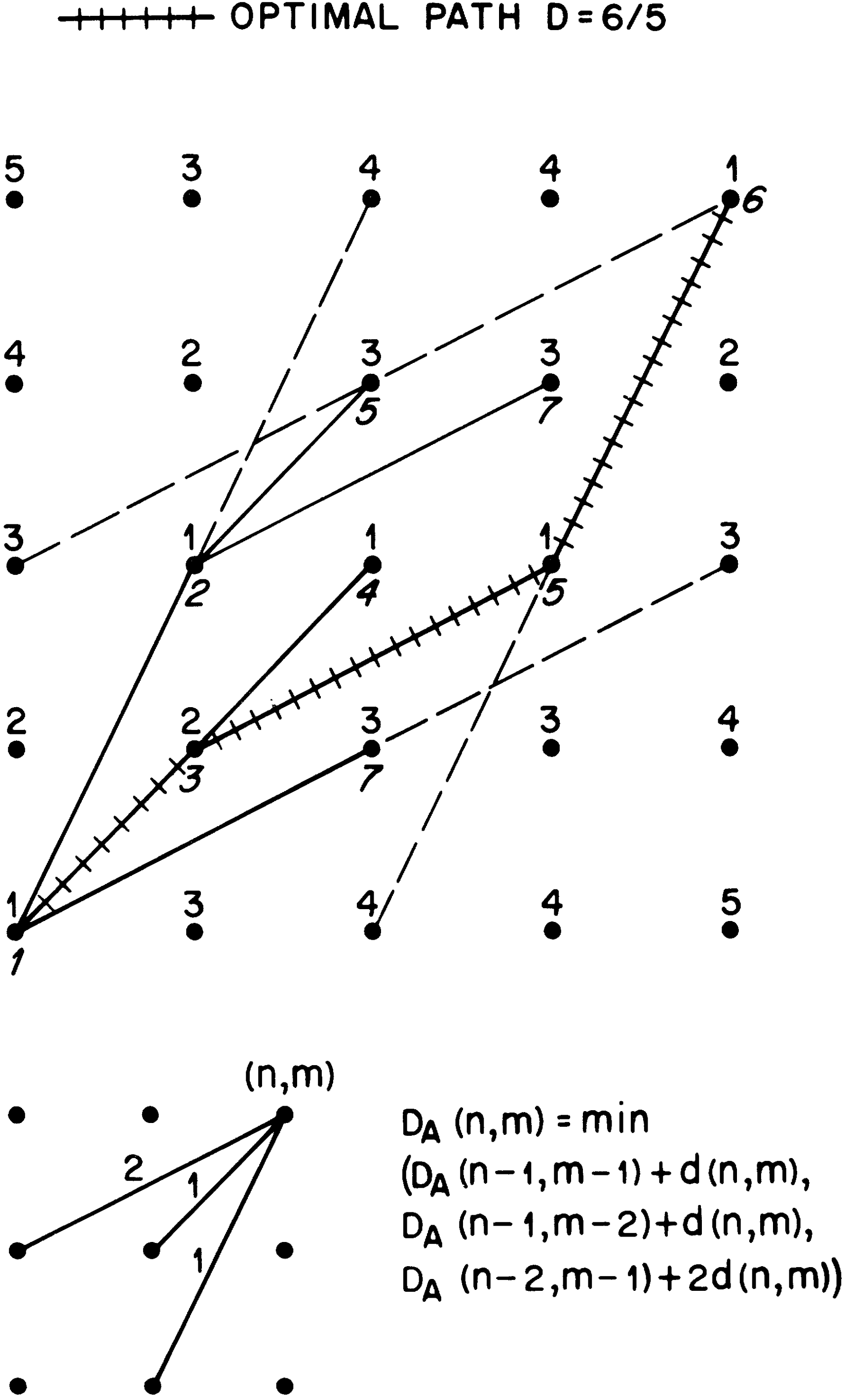


Fig. 2.15 Typical dynamic time warping results.

programming equation. In Figure 2.15 the unslanted numbers represent the local distances at a point and the slanted numbers represent the best accumulated distance to that point. The dashed lines are the global constraints which arise from the local constraints. The solid lines are the local paths used to reach any point (n,m) from $(1,1)$ via the best path to that point. Finally, the globally optimal path is indicated as a cross-hatched line. The value of $D_A(N,M)$ is 6 and $\hat{D} = 6/5$.

Table 2.5 summarizes the partial functions which are of interest in this thesis. In addition to the local constraints previously defined two new entries appear in this table. One entry refers to type IV local constraints, which are shown in Figure 2.16. Type IV constraints have a maximum slope of 3 and a minimum slope of 1/3. Type IV constraints are similar to type III constraints, i.e. assymetric, using intermediate points onlY for the x-axis pattern and Type IV constraints have a "memory" of three. The other unusual entry refers to Itakura's [6] accumulated distance function, namely,

$$D_A(n,m) = \min \begin{Bmatrix} D_A(n-1,m-2)+d(n,m), \\ D_A(n-1,m-1)+d(n,m), \\ D_A(n-1,m)g(k)+d(n,m) \end{Bmatrix} \tag{2.40}$$

where

$$g(k) = \begin{cases} 1 & j(k-1)\neq j(k-2) \\ \infty & j(k-1)=j(k-2) \end{cases} \cdot \tag{2.41}$$

The purpose of the $g(k)$ function is to disallow any paths which go horizontally for more than one arc. A pictoral representation of Itakura's local constraints is given at the bottom of Figure 2.17. The crossed out arc illustrates the restriction that a path may not move horizontally for two consecutive segments. Thus, paths generated by Itakura's algorithm also have a maximum slope of 2 and a minimum slope of 1/2. In facts, paths generated by Itakura's algorithm obey all the restrictions of type III paths. Also, Itakura uses weighting function c, $\tilde{W}(k) = i(k) - i(k-1) = 1$. However, Itakura's algorithm may not find the truly optimal path that the dynamic programming solution of Type III local constraints with weighting function c would produce. Figure 2.17 shows an example of this phoenomenon. Itakura's algorithm was

Accumulated Distance Functions

Local Constraints	Weighting Function	Accumulated Distance Function
I	a	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+d(n,m),$ $D_A(n-1,m-2)+\frac{1}{2}d(n,m-1)+\frac{1}{2}d(n,m),$ $D_A(n-2,m-1)+\frac{1}{2}d(n-1,m)+\frac{1}{2}d(n,m))$
I	b	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+d(n,m),$ $D_A(n-1,m-2)+d(n,m-1)+d(n,m),$ $D_A(n-2,m-1)+d(n-1,m)+d(n,m))$
I	c	$D_A(n,m) =$ $\min(D_A(n-1,\ m-1)+d(n,m),$ $D_A(n-1,m-2)+\frac{1}{2}d(n,m-1)+\frac{1}{2}d(n,m),$ $D_A(n-2,m-1)+d(n-1,m)+d(n,m))$
I	d	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+2d(n,m),$ $D_A(n-1,m-2)+\frac{3}{2}d(n,m-1)+\frac{3}{2}d(n,m),$ $D_A(n-2,m-1)+\frac{3}{2}d(n-1,m)+\frac{3}{2}d(n,m))$

Table 2.5

Accumulated Distance Functions

Accumulated Distance Functions

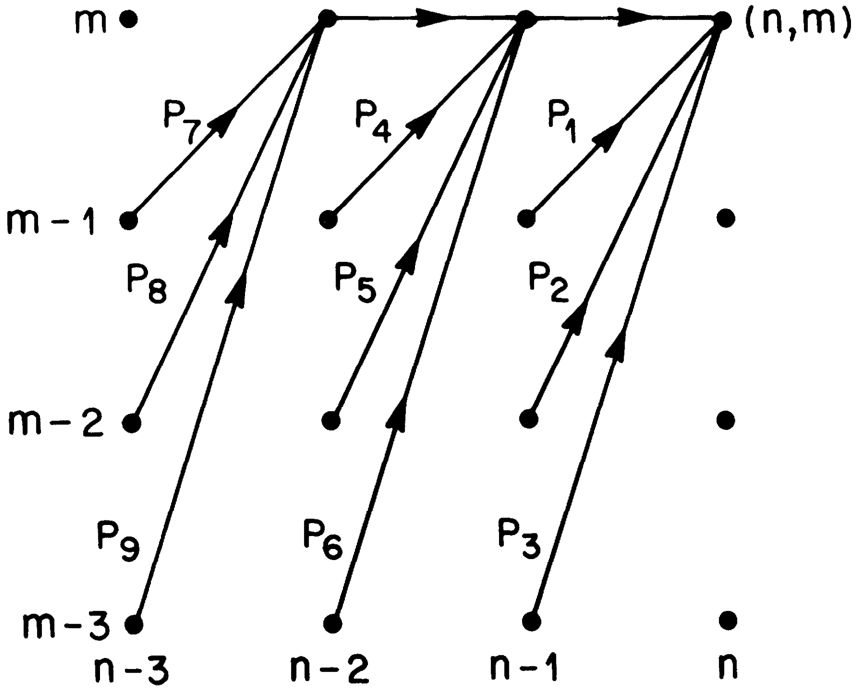
Local Constraints	Weighting Function	Accumulated Distance Function
II	a	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+d(n,m),$ $D_A(n-1,m-2)+d(n,m),$ $D_A(n-2,m-1)+d(n,m))$
II	b	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+d(n,m),$ $D_A(n-1,m-2)+2d(n,m),$ $D_A(n-2,m-1)+2d(n,m))$
II	c	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+d(n,m),$ $D_A(n-1,m-2)+d(n,m),$ $D_A(n-2,m-1)+2d(n,m))$
II	d	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+2d(n,m),$ $D_A(n-1,m-2)+3d(n,m),$ $D_A(n-2,m-1)+3d(n,m))$

Table 2.5 (continued)

Accumulated Distance Functions

Local Constraints	Weighting Function	Accumulated Distance Function
III	c	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+d(n,m),$ $D_A(n-1,m-2)+d(n,m),$ $D_A(n-2,m-1)+d(n-1,m)+d(n,m),$ $D_A(n-2,m-2)+d(n-1,m)+d(n,m))$
IV	c	$D_A(n,m) =$ $\min(D_A(n-1,m-1)+d(n,m),$ $D_A(n-1,m-2)+d(n,m),$ $D_A(n-1,m-3)+d(n,m),$ $D_A(n-2,m-1)+d(n-1,m)+d(n,m),$ $D_A(n-2,m-2)+d(n-1,m)+d(n,m),$ $D_A(n-2,m-3)+d(n-1,m)\,d(n,m),$ $D_A(n-3,m-1)+d(n-2,m)+d(n-1,m)+d(n,m),$ $D_A(n-3,m-2)+d(n-2,m)+d(n-1,m)+d(n,m),$ $D_A(n-3,m-3)+d(n-2,m)+d(n-1,m)+d(n,m))$
Itakura	c	$D_A(n,m) =$ $\min(D_A(n-1,m-2)+d(n,m),$ $D_A(n-1,m-1)+d(n,m),$ $D_A(n-1,m)g(k)+d(n,m))$ $g(k) = \begin{cases} 1 & j(k-1) \neq j(k-2) \\ \infty & j(k-1) = j(k-2) \end{cases}$

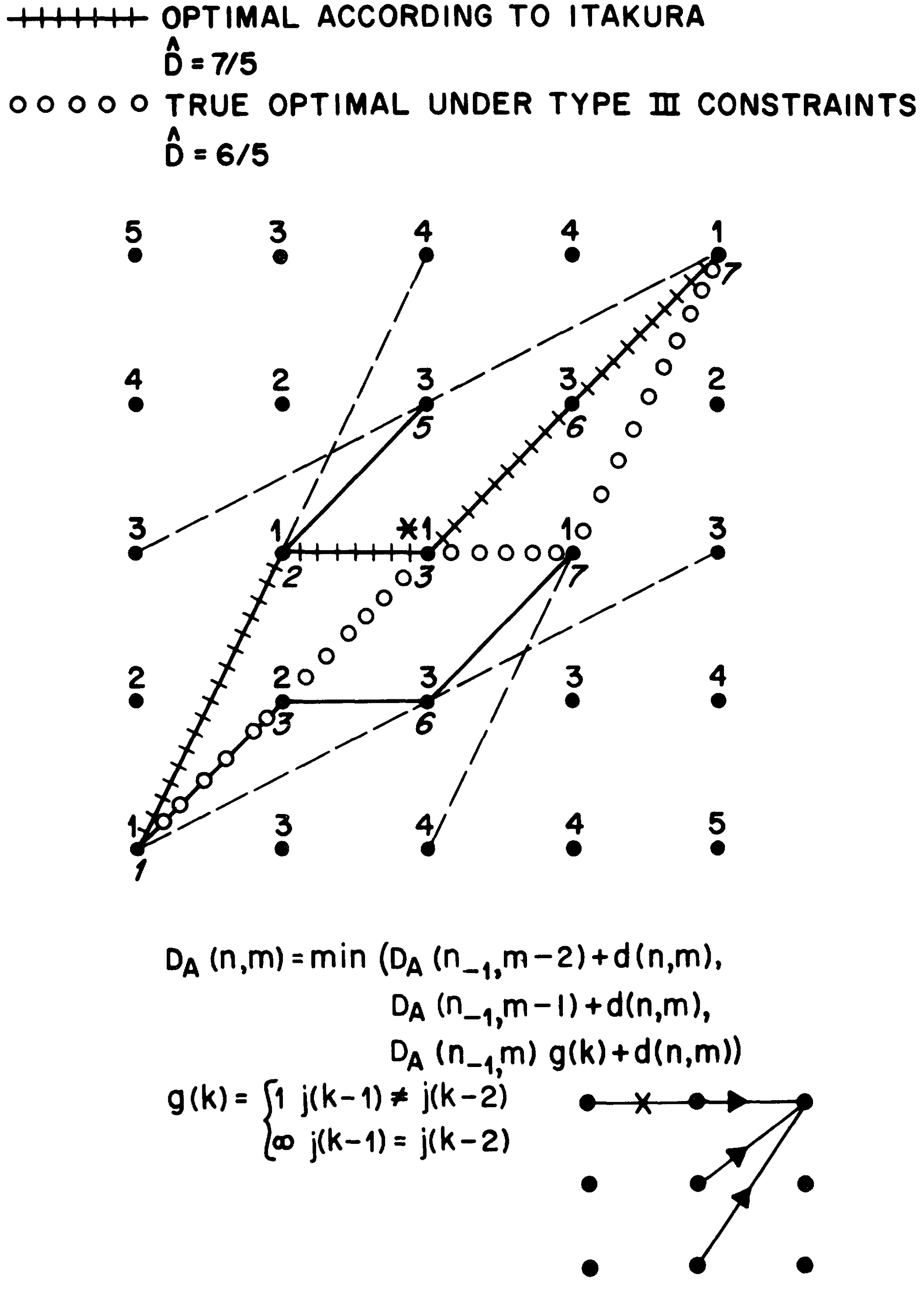
Table 2.5 (Continued)



$P_1 \rightarrow (1,1)$
$P_2 \rightarrow (1,2)$
$P_3 \rightarrow (1,3)$
$P_4 \rightarrow (1,0)\ (1,1)$
$P_5 \rightarrow (1,0)\ (1,2)$
$P_6 \rightarrow (1,0)\ (1,3)$
$P_7 \rightarrow (1,0)\ (1,0)\ (1,1)$
$P_8 \rightarrow (1,0)\ (1,0)\ (1,2)$
$P_9 \rightarrow (1,0)\ (1,0)\ (1,3)$

TYPE IV CONSTRAINTS

Fig. 2.16 Type IV local constraints.


+++++++ OPTIMAL ACCORDING TO ITAKURA
D̂ = 7/5
○ ○ ○ ○ ○ TRUE OPTIMAL UNDER TYPE III CONSTRAINTS
D̂ = 6/5
5 3 4 4 1
4 2 3 3 2
3 1 *1 1 3
2 2 3 3 4
1 3 4 4 5
1
2
3
3
5
6
6
7
7

$$D_A(n,m) = \min\big(D_A(n_{-1}, m-2) + d(n,m),\ D_A(n_{-1}, m-1) + d(n,m),\ D_A(n_{-1}, m)\, g(k) + d(n,m)\big)$$

$$g(k) = \begin{cases} 1 & j(k-1) \neq j(k-2) \\ \infty & j(k-1) = j(k-2) \end{cases}$$

Fig. 2.17 Itakura's dynamic time warping algorithm.

run on the distances of Figure 2.15 and found a best path with $\hat{D} = 7/5$. However, as shown in Figure 2.15, there is a type III path using weighting function c which has a value for the best path of $\hat{D} = 6/5$. This problem is seen to arise at the point labeled by (*) because Itakura's algorithm correctly finds that the best path to (3,3) from (1,1) comes in horizontally from (2,3). However, since the best path to (4,3) from (1,1) comes in horizontally from (3,3) it is not found because the function $g(k)$ excludes it.

2.4 Summary

We have now defined all the variables which will be of interest for dynamic time warping for isolated word recognition. These variables are endpoint constraints, local continuity constraints, global range constraints, axis orientation and distance measures. We have also presented the basic principles of dynamic programming implementations of time warping. What needs to be considered now are methods of evaluating the performance of the various time warping algorithms. We will discuss this in the next chapter of this thesis.

Chapter 3

Performance Measures for Dynamic Time Warping Algorithms

3.1 Introduction

As discussed in the previous chapter, there exist a large number of factors which must be considered in the specification of a DTW algorithm. Included among these factors are endpoint constraints, local continuity constraints, global range constraints, axis orientation and the choice of an appropriate distance measure. Based on the considerations of the previous chapter it is not possible to specify a theoretically optimal DTW algorithm which would be applicable to any situation. However, because DTW algorithms play such an important role in speech recognition systems which use pattern matching, it is important to understand how the various features of a time warping algorithm interact and how they affect the overall performance of a recognition system. Thus, in this section we first describe the particular speech recognition system which was used to measure the performance of the various time warping algorithms and in the next section we describe the performance measures that were used to evaluate the DTW algorithms.

3.1.1 The Isolated Word Recognizer

The speech recognition system which was used to measure the performance of the various DTW algorithms is shown in the block diagram of Figure 3.1. The system is similar to the isolated word recognition system originally proposed by Itakura [6], and described in detail by Rabiner [12]. Analog speech (in the form of isolated words) is recorded off of a standard telephone line, bandpass filtered from 100 to 3200 Hz (using a 24 db/octave filter) to remove hum and to prevent aliasing, and converted to a 16 bit pulse code modulation digital format at a 6.67 kHz sampling rate. Following digitization the speech signal, $s(n)$, is preemphasized to flatten the speech spectrum using a first order system with the transfer function

$$H(z) = 1 - .95z^{-1}, \tag{3.1}$$

giving

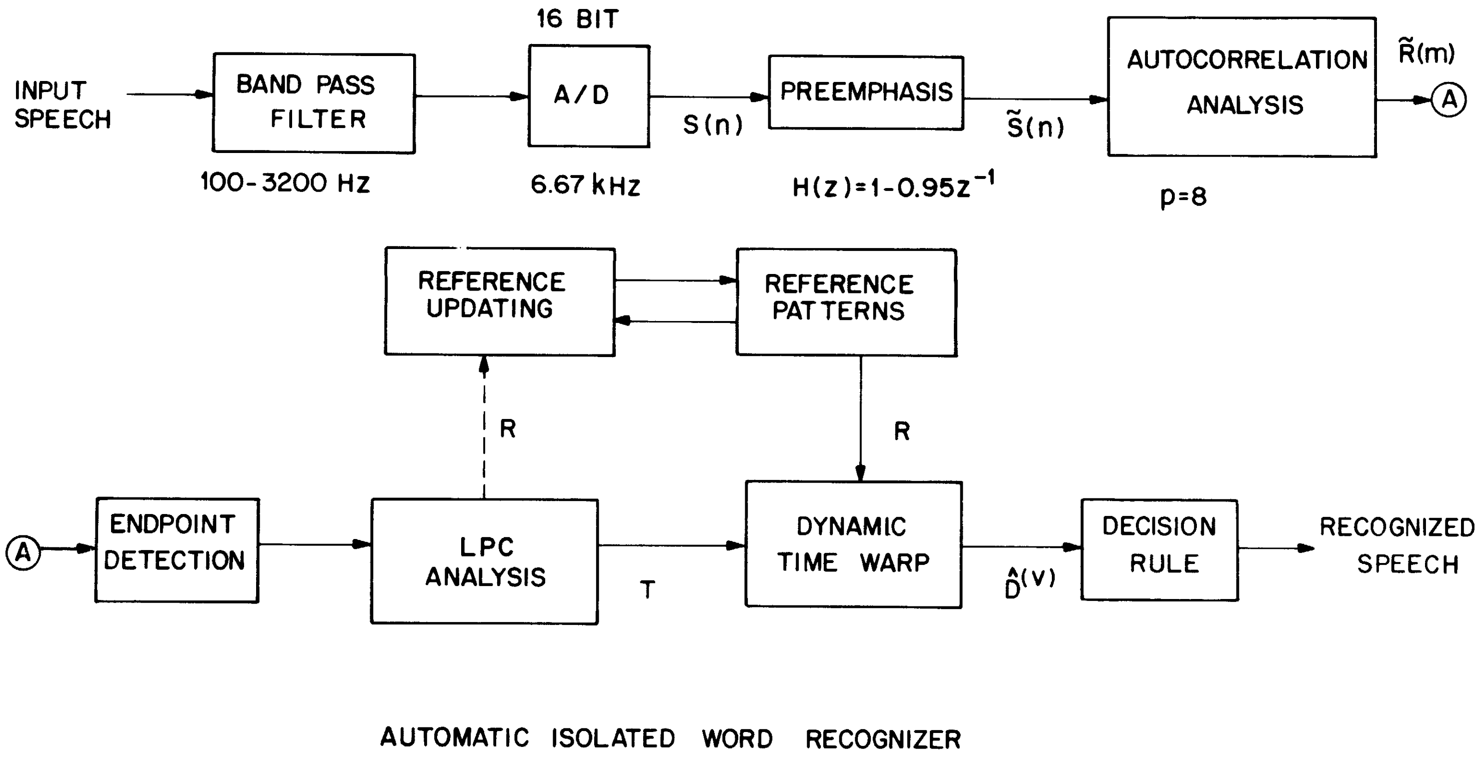


Fig. 3.1 Isolated Word Recognition System.

$$\tilde{s}(n) = s(n) - .95s(n-1)\ . \tag{3.2}$$

Following preemphasis the speech signal is analyzed by an autocorrelation technique. This analysis takes place in NS sample long frames (NS=300 corresponding to 45 msec of data) and occurs every MS samples (MS=100 corresponding to an overlap of 200 samples or 300 msec). We denote the samples of a frame as $x(n)$, $0 \leqslant n \leqslant NS - 1$, irrelevant of the true value of n for the frame. For each frame two separate operations are performed in the analysis, namely:

1. Windowing using a Hamming window. The windowed data, $\hat{x}(n)$, is obtained from $x(n)$ as

$$\hat{x}(n) = x(n)\cdot W(n) \quad , \quad 0 \leqslant n \leqslant NS - 1 \tag{3.3}$$

where

$$W(n) = 0.54 - 0.46 \cos\left[\frac{2\pi n}{NS-1}\right]\ . \tag{3.4}$$

2. Autocorrelation analysis of the frame. The autocorrelation coefficients, $\tilde{R}(l)$, are determined according to the rule:

$$\tilde{R}(l) = \sum_{n=0}^{NS-1-l} \hat{x}(n)\hat{x}(n+l) \quad , \quad l = 0,1,..,p \tag{3.5}$$

where $p = 8$ for this system.

The next step in the isolated word recognition system is endpoint detection. A simple endpoint detection scheme, using only energy measurements, is employed. The log intensity contour of the speech, given by the time pattern of log $[\tilde{R}(0)]$, is measured for silence (the lowest energy values over the entire recording interval) and two threshold levels are set, based on the background energy. When the intensity of the input signal exceeds these thresholds, for a sufficient number of frames, speech is said to be present. A description of the double thresholding technique is given by Lamel [13]. At this point, the endpoint detector is highly susceptible to artifacts such as mouth noises and breathiness. As such, the input speech is both automatically and manually monitored and utterances with such artifacts are repeated. With such monitoring we may be reasonably sure that an accurate determination of the endpoints of the isolated word is made.

Following endpoint detection an LPC analysis is performed. Since autocorrelation coefficients for each frame of the word have been computed it is possible to solve the equations for the LPC coefficients for each frame by Durbin's recursion:

1. initialization

$$E^{(0)} = \tilde{R}(0) \tag{3.6}$$

2. for $i = 1,2,...,p$ do steps 2-5

$$k_i = \frac{\left[\tilde{R}(i) - \sum_{j=1}^{i-1} \alpha_j^{(i-1)} \tilde{R}(i-j)\right]}{E^{(i-1)}} \tag{3.7}$$

(For i=1 the summation from j=1 to 0 is skipped)

3.

$$\alpha_i^{(i)} = k_i \tag{3.8}$$

4. For $j = 1$ to $i - 1$ (skip for i=1)

$$\alpha_j^{(i)} = \alpha_j^{(i-1)} - k_i \alpha_{i-j}^{(i-1)} \tag{3.9}$$

5.

$$E^{(i)} = (1 - k_i^2) E^{(i-1)}$$

6.

$$E = E^{(p)} \tag{3.11}$$

$$a_0 = 1 \quad , \quad a_j = \alpha_j^{(p)} \quad , \quad 1 \leqslant j \leqslant p \tag{3.12}$$

where E is the linear prediction residual of the frame and a_j is the j^{th} linear predictive coefficient of the frame.

After LPC analyses for each frame of the word has been performed, either one of two things may occur. Either the utterance may be used as a test phrase and compared to a set of stored reference patterns, or the utterance may be used to create (or possibly update) the reference patterns. If the utterance is used to create the reference pattern, then the form of creation depends on the characteristics of the recognition system, i.e. whether it is designed as a speaker dependent or speaker independent recognizer. Reference patterns for speaker dependent systems generally consist of one or two replications of each word of the vocabulary of the system, while reference patterns for speaker independent systems generally consist of several patterns per word formed by the use of statistical clustering techniques [13]. However, rather than being stored directly as sequential frames of LPC coefficients, a reference pattern, $\mathbf{R}(n)$, is stored as

$$R(n,1) = \log \left[\sum_{j=0}^{p} a_j^2 \right] \tag{3.13a}$$

$$R(n,i+1) = \frac{2 \sum_{j=0}^{p-i} a_j a_{j+i}}{\sum_{j=0}^{p} a_j^2} \quad , \quad 1 \leqslant i \leqslant p \tag{3.13b}$$

where $R(n,i)$ is the i^{th} component of the n^{th} frame of the reference pattern, $\mathbf{R}(n)$, for $n = 1$ (the first frame of the word) to $n = N$ (the last frame of the word).

If, on the other hand, the utterance is to be used as a test utterance then the set of autocorrelation coefficients and prediction residuals are transformed to a test pattern, $\mathbf{T}(m)$, as follows:

$$T(m,1) = \log \left[E/\tilde{R}(0) \right] \tag{3.14a}$$

$$T(m,i+1) = \tilde{R}(i)/\tilde{R}(0) \quad , \quad 1 \leqslant i \leqslant p \tag{3.14b}$$

where $T(m,i)$ is the i^{th} component of the m^{th} frame of the test pattern, $\mathbf{T}(m)$, for $m = 1$ (the first frame of the word) to $m = M$ (the last frame of the word).

The reason for the transformations is to simplify calculation of the local distance metric, which is the log likelihood ratio as proposed by Itakura [6]. Using the transformed representations Itakura showed that the value of $\tilde{d}(\mathbf{R}(n),\mathbf{T}(m))$, the distance between frame n of the reference and frame m of the test, becomes

$$\tilde{d}(\mathbf{R}(n),\mathbf{T}(m)) =$$

$$R(n,1) - T(m,1) + \log\left[1 + \sum_{j=1}^{p} R(n,j)T(m,j)\right] \tag{3.15}$$

which is simpler to compute than the original form of the log likelihood ratio, namely

$$\tilde{d}(\mathbf{R}(n),\mathbf{T}(m)) =$$

$$\log\left[\frac{\sum_{j=0}^{p}\sum_{k=0}^{p} \tilde{a}_j^{(n)}\tilde{R}^{(m)}(|j-k|)\tilde{a}_k^{(n)}}{E^{(m)}}\right] \tag{3.16}$$

where $\tilde{a}_j^{(n)}$ is the j^{th} LPC coefficient of the n^{th} frame of the reference, $\tilde{R}^{(m)}(|j-k|)$ is the $|j-k|^{th}$ autocorrelation of the m^{th} frame of the test, and $E^{(m)}$ is the linear prediction residual of the m^{th} frame of the test utterance.

The next step in the speech recognition system is application of dynamic time warping. Given a test utterance, it is compared to every possible reference pattern and a distance score is generated

$$\hat{D}^{(v)} = \min_{(K,i(k),j(k))} [D^{(v)}(i(k),j(k))] \tag{3.17}$$

where $D^{(v)}(i(k),j(k))$ is the global distance between the v^{th} reference and the test pattern, along the path defined by $i(k)$ and $j(k)$, with K being the length of the path, and where $\hat{D}^{(v)}$ is the minimum global distance over all paths. The calculation of Eq. (3.17) is made for all reference patterns v, i.e. for $v = 1,2,...,V$ where V is the total number of reference patterns.

The final step in the recognition process is the decision as to which word (or words) is chosen by the recognizer as the best match to the unknown input word. For our purpose the minimum distance decision rule is always used, namely

$$\hat{v} = \underset{(v)}{argmin}\, [\hat{D}^{(v)}] \tag{3.18}$$

where *argmin* $[g(x)]$ returns that value of x which minimizes $g(x)$. The word associated with $\hat{v}^{th}$ reference pattern is chosen as the recognized word.

In the remainder of this chapter we will discuss the specification of performance measures for a DTW algorithm that fit into framework of the recognition system of Figure 3.1. In this thesis we will only be concerned with this particular recognition system, although it is believed that the results to be presented will be applicable to similar type systems (e.g. with different feature sets and distance measures), it should be stressed that some of our assumptions may be central to the system of Figure 3.1 and not be applicable to other systems using dynamic time warping.

3.2 Performance Criteria and Measures

As discussed in the previous section, the DTW algorithm is an essential component of the isolated word recognition system of Figure 3.1. The values of $\hat{D}^{(r)}$ generated by the DTW algorithm are the basis for the decision process, thus it is important that the dynamic time warping process be very accurate in its determination of the optimal path. Also, an examination of the amount of computation involved in the recognition process reveals — that the majority of the computation (from 50 to 90+%, depending on the number of reference patterns) in the system is involved in the DTW algorithm. Thus, we would prefer our DTW algorithm to be as efficient as possible. It is also important for the DTW algorithm to fit into a small amount of memory so that the recognition process can be easily implemented in special purpose hardware. Since it is highly unlikely that the most accurate time warping algorithm will be the most efficient or the most compact, it is important to understand the tradeoffs which can be made among the factors of efficiency, size, and accuracy.

In summary, the factors, and the associated measures which will be used to measure the performance of the various DTW algorithms, are as follows:

1. Memory requirements — the amount of storage required by the time warping algorithm, measured by the amount of memory required for temporary variables.
2. Efficiency (Speed of Computation) — the amount of time required by the time warping algorithm to compute the optimal path, as measured by the average computational time and the average number of calculations of the local distance metric.

3. Recognition accuracy — the percentage of time the reference word with the smallest distance, $\hat{D}^{(\hat{v})}$, matches the spoken test word in a series of isolated word recognition tests.

In this section we discuss the significance of these performance measurements and we try to predict how some of the factors of the time warping algorithms defined in Chapter 2 affect the performance scores.

3.2.1 Memory Requirements

Memory requirements measure the amount of storage required by a particular DTW algorithm. Typically, there are two components of memory usage: a fixed portion, which is independent of the particular DTW algorithm chosen, and a variable portion which depends upon the choice of the DTW algorithm.

Fixed storage is required for the feature vectors, $\mathbf{R}(n)$, $n = 1,2,...,N$ and $\mathbf{T}(m)$, $m = 1,2,...,M$ although it is possible, in some real-time applications, to store only a single frame of $\mathbf{T}(m)$, process it, and then proceed to the next frame. In our discussion, however, we assume that all frames of $\mathbf{R}(n)$ and $\mathbf{T}(m)$ are stored. A small mount of fixed storage is also required for scratch pad work but this amount is negligible (typically, less than ten storage locations).

The storage requirement which varies among the different time warping algorithms is the amount of memory required for storage of the accumulated distance functions. The accumulated distance function is defined over the entire (n,m) plane. Typically, $D_A(n,m)$ is computed for a fixed n over the entire range of legal values for m before n is incremented again. Thus, $D_A(n,m)$ is computed as a series of vectors and may be stored as such. Since computation of $D_A(n,m)$ typically depends only on $D_A(n',m')$ with

$$n - n' \leqslant 2 \qquad (3.19a)$$

$$m - m' \leqslant 2 \qquad (3.19b)$$

only two or three vectors for $D_A(n,m)$ are needed at any time. In addition to the storage for the accumulated distance functions, memory is also used to store local distances when they are used more than once. This occurs in local constraints types I and III, in which $d(n,m)$ is used both to compute $D_A(n,m)$ and $D_A(n+1,m)$. Further storage may also be used for side information, such as Itakura's g function, which must also be stored as a vector.

Since the storage requirements of a DTW algorithm are for a series of vectors, the natural measure for the amount of storage required by a DTW algorithm is the number of such vectors.

It is possible to make at least one general statement about the amount of storage required by a DTW algorithm. That is, that the more information which is required to compute the accumulated distance function in a time warping algorithm, the more storage that will be required by that algorithm. Thus, we would expect that time warping algorithms with type II local constraints would require less storage than those algorithms which use either type I or type III local constraints.

3.2.2 Computational Efficiency (Speed of Exacution)

Computational efficiency is a measure of how fast a time warping algorithm can find an optimal path. Computational efficiency may be broken up into two distinct components — combinatorics and local distance calculations. Combinatorics measures the amount of time required to compute the accumulated distance functions given the values of the local distance metric.

The time for combinatorics will increase with an increasing number of productions in a local constraint and with an increasing number of intermediate points used in a production. Thus, we would expect that type II local constraints would require less computation than local constraints of types I or III.

The other aspect of computational speed which is of interest is the average number of local distance calculations per dynamic time warp. This measure is strictly a function of the global range available for legal paths because a local distance calculation must be performed for every point in the global range. As the global range increases, more and more distance calculations must be performed. Thus we would expect that a larger maximum slope, as in type IV local constraints (maximum slope, $S=3$), would increase the computation time relative to a smaller maximum slope, as in local constraints of types I, II and III (maximum slope, $S=2$), because the global range would be expanded. Also, we would expect that limiting the global range by an absolute time difference would improve the computational speed because the global range would be reduced.

3.2.3 Recognition Accuracy

Recognition accuracy measures how well the time warping algorithm actually works in the given recognition system in a series of isolated word recognition experiments.

The natural measure of recognition accuracy is the recognition error rate,

$$\text{error rate} = \frac{\text{number of improper recognitions}}{\text{number of trials}} \tag{3.20}$$

where an improper recognition occurs when the reference associated with $\hat{v}$ in Eq. (3.18) does not represent the word which is actually the test utterance.

Another measure which can be useful when the error rate does not provide fine enough distinctions in the performance of the various DTW algorithms is a measure of the separation of the values of $\hat{D}^{(v)}$ when the reference and the test are the same word and when they are different. The natural means of expressing such a measure is in terms of a probability distribution. We may define the probability distributions, $P_S(D)$ and $P_D(D)$,

$$P_S(D) = Prob(\hat{D}^{(v)} \leqslant D \,|\, \text{the reference and the test are the same word}) \tag{3.21a}$$

$$P_D(D) = Prob(\hat{D}^{(v)} \leqslant D \,|\, \text{the reference and the test are different words}). \tag{3.21b}$$

The functions $P_S(D)$ and $P_D(D)$ may be measured empirically from experimental data as

$$P_S(D) = \frac{\text{number of trials with } \hat{D}^{(v)} \leqslant D}{\text{number of trials}} \tag{3.22a}$$

(reference and test are the same word)

$$P_D(D) = \frac{\text{number of trials with } \hat{D}^{(v)} \leqslant D}{\text{number of trials}} \tag{3.22b}$$

(reference and test are different words).

With the definitions of $P_S(D)$ and $P_D(D)$ given in Eqs. (3.21) we may define a natural measure of their separation. If we define a threshold ϕ such that when $\hat{D}^{(v)} \leqslant \phi$ we conclude that the reference and the test are the same word then the probability of a false alarm (concluding that they are the same, when, in fact, they are not) is

$$P_F = P_D(\phi) \tag{3.23}$$

and the probability of a miss (concluding that they are different when actually they are the same) is

$$P_M = 1 - P_S(\phi) \ . \tag{3.24}$$

If we equate P_M and P_F we may, given $P_D(D)$ and $P_S(D)$, compute both ϕ and P_M. We shall use P_M as our measure of the separation of the distributions $P_S(D)$ and $P_D(D)$.[1]

In Figure 3.2 we show how the values of ϕ and P_M can be determined. In part a we plot $P_D(D)$ and $1 - P_S(D)$. The value of D for which $P_D(D) = 1 - P_S(D)$ is ϕ and the value of $P_D(D)$ at this point is P_M. In part b we show $\mathbf{p}_S(D)$ and $\mathbf{p}_D(D)$, the probability densities, defined as

$$\mathbf{p}_S(D) = \frac{d}{dD} P_S(D) \tag{3.25a}$$

$$\mathbf{p}_D(D) = \frac{d}{dD} P_D(D) \ . \tag{3.25b}$$

We show ϕ and show that P_M is defined to be the area under $\mathbf{p}_S(D)$ from $D = \phi$ to $D = \infty$ (or equivalently, under $\mathbf{p}_D(D)$ from $D = 0$ to $D = \phi$). The functions $\mathbf{p}_S(D)$ and $\mathbf{p}_D(D)$ are introduced because they can be easily determined from histograms of the distance scores from a time warping algorithm.

Given a measure of accuracy it is difficult, however, to predict exactly how all of the factors of a DTW algorithm will affect its recognition accuracy. For example, increasing the maximum

[1] It should be noted that $P_M \times 100\% \neq$ error rate because P_M is used to measure separation and is based on the assumption of a single comparison of a reference and a test pattern while the error rate is based on comparisons of a test pattern to all reference patterns.

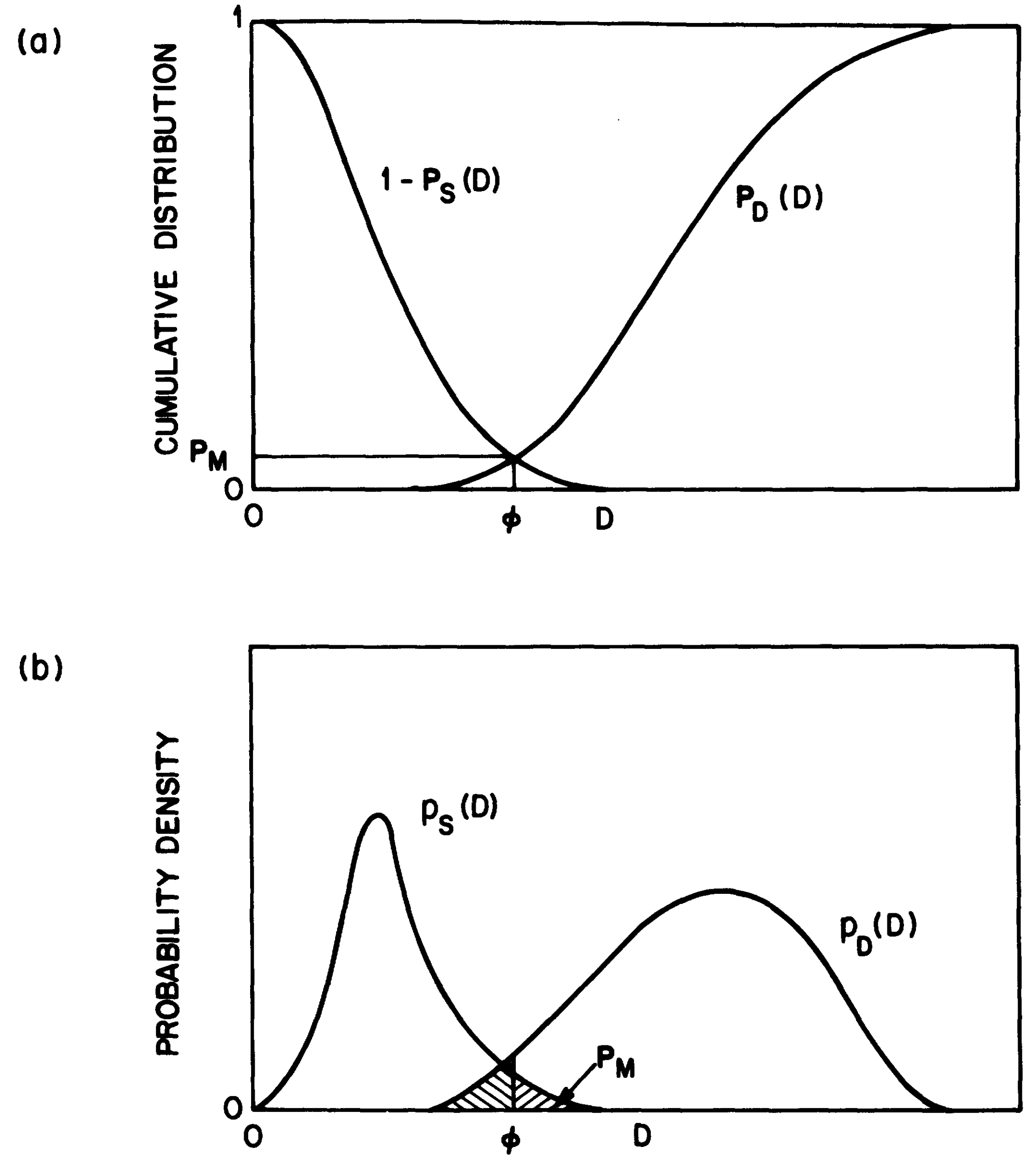


Fig. 3.2 Probability distributions for distance scores.

allowable slope from 2 to 3 may be useful for speakers with a high degree of variability in different replications of the same word, but may also be harmful for speakers who consistently repeat a word at the same rate. One reasonable assumption, however, is that type I local constraints will, in general, provide better accuracy than type II local constraints. Type I paths use intermediate points to compute their accumulated distance functions while type II paths do not. Thus type II paths must lose some information relative to type I paths. Such a loss in information is expected to reduce the accuracy of a time warping algorithm.

3.3 Summary

In this chapter we have described the various performance criteria and their associated measures. We have also made some predictions about the behavior of various DTW algorithms. However, to fully understand the behavior of the different time warping algorithms for isolated word recognition we must have quantitative values for the performance measures. In the next chapter we will describe the experiments performed and their resulting measures of the performance of the different DTW algorithms.

Chapter 4

Results on Isolated Word Recognition

4.1 Initial Experiments

In the previous chapters we have discussed some of the different factors which are important in the implementation of a dynamic time warping algorithm for isolated word recognition. These factors include endpoint constraints, local continuity constraints, global range constraints, axis orientation and choice of distance measure. We have also specified the different performance criteria by which the different DTW algorithms are compared. In this chapter we will give the results of several experiments designed to measure the performance of the various DTW algorithms.

All the recognition experiments were performed using the isolated word recognition system described in the previous chapter. The test involved two different types of word recognition test sets (based on the training), namely:

1. Speaker Dependent Set (TS1)
2. Speaker Independent Set (TS2)

The speaker dependent test set (TS1) used a 39 word vocabulary consisting of the letters A-Z, the digits 0-9 and the command words STOP, ERROR and REPEAT. This vocabulary has been used in applications involving directory assistance and credit card calling [1]. TS1 consisted of five tokens of each word of the 39 word vocabulary for each of two talkers, denoted as SD1 and SD2 (390 words in total). The recordings were made for use in earlier recognition experiments [1,14]. The recognition system for TS1 used two reference templates for each word of the vocabulary. These reference templates were generated from two separate replications of the entire vocabulary for each of the two talkers.

The speaker independent test set (TS2) used the 54 vocabulary of computer terms originally proposed by Gold [15] and listed in Table 4.1. This vocabulary is considered to be less difficult

54 Word Vocabulary

1. INSERT (2)
2. DELETE (2)
3. REPLACE (2)
4. MOVE (1)
5. READ (1)
6. BINARY (3)
7. SAVE (1)
8. CORE (1)
9. DIRECTIVE (3)
10. LIST (1)
11. LOAD (1)
12. STORE (ı)
13. ADD (1)
14. SUBTRACT (2)
15. ZERO (2)
16. ONE (1)
17. TWO (1)
18. THREE (1)
19. FOUR (1)
20. FIVE (1)
21. SIX (1)
22. SEVEN (2)
23. EIGHT (1)
24. NINE (1)
25. MULTIPLY (3)
26. DIVIDE (2)
27. NUMBER (2)
28. NAME (1)
29. END (1)
30. SCALE (1)
31. CYCLE (2)
32. SKIP (1)
33. JUMP (1)
34. ADDRESS (2)
35. OVERFLOW (3)
36. POINT (1)
37. CONTROL (2)
38. REGISTER (3)
39. WORD (1)
40. EXCHANGE (2)
41. INPUT (2)
42. OUTPUT (2)
43. MAKE (1)
44. INTERSECT (3)
45. COMPARE (2)
46. ACCUMULATE (4)
47. MEMORY (2)
48. BITE (1)
49. QUARTER (2)
50. HALF (1)
51. WHOLE (1)
52. UNITE (2)
53. DECIMAL (3)
54. OCTAL (2)

(*n*) - Number of syllables in word

Table 4.1

than the 39 word vocabulary used for TS1 because it has few acoustically similar words. The 54 word vocabulary also has the property that half of its words are polysyllabic as opposed to only five words (W, ZERO, SEVEN, ERROR and REPEAT) in the 39 word vocabulary. Polysyllabic words tend to be recognized more reliably than monosyllabic words, hence one would expect higher recognition accuracy for the 54 word vocabulary, all other factors being equal.

TS2 consisted of one token of each word of the 54 word vocabulary for each of 4 talkers, denoted as SI1, SI2, SI3 and SI4 (216 words in total). (The recordings for TS2 were also obtained from a previous experiment [16].) The recognition system for TS2 used two reference templates for each word of the vocabulary. The reference templates were obtained from a clustering analysis of 100 tokens of each word of the vocabulary. The output of the clustering analysis was a grouping of the 100 tokens into a small number of sets (clusters), in which the tokens within each cluster were similar. The two reference templates were obtained from the two largest clusters (i.e. with the most tokens) and represented an average of the tokens within each cluster [17].

The entire experimental system was implemented in FORTRAN on a Data General Eclipse S230 minicomputer. The results of the recognition experiments (using the two test sets of data) were used to compare the recognition accuracies of the various DTW algorithms. Measurements of memory requirements were made by examination of the FORTRAN code used to implement the different DTW algorithms. Measurements of computational efficiency were made by averaging the timing results of 1500 separate time warps. Timings were made using a computer controlled clock which was accurate to ±10 microseconds. In the next section we give the results of the various experiments.

4.2 Experimental Results

In this section we give the results of the experiments performed on the various DTW algorithms. The performance criteria which we used are those described in Chapter 3, namely:

1. Memory Requirements

2. Computational Efficiency

3. Recognition Accuracy

4.2.1 Memory Requirements

In Table 4.2 we summarize the results of measurements on the amount of storage required by the various DTW algorithms, as a function of the local path constraints. As discussed in Chapter 3, the unit of storage is a vector, which measures the amount of information required to compute an accumulated distance function. The three different types of vectors include storage for accumulated distances, local distances and side information. The results show that Itakura's local constraints and type II local constraints require the least amount of storage, (2 vectors), followed by local constraints of types I and III (3 vectors), and finally by local constraints of type IV (5 vectors).

The other important factor in the measurement of storage requirements is the size of a vector. In general, all vectors must have M entries but the size of a entry may vary. Because the side information required by Itakura's algorithm is simply a "yes" or a "no," each entry of this vector may be encoded as a single bit. However, accumulated distance and local distance vectors must be accurately encoded to several bits (32 in our implementations).

As described in Chapter 3, we found no variation in memory requirements due to any factor aside from local constraints. Thus, Table 4.2 summarizes all the information about storage requirements for the DTW algorithms which we examined.

4.3.2 Computational Efficiency

Computational efficiency (or speed), unlike memory usage, is dependent on several of the factors in a dynamic time warping algorithm. As discussed in Chapter 3, computational time has two distinct components, namely, combinatorics (the time to set up and compute the accumulated distance functions), and local distance computations. We found that, for the log likelihood ratio, that the average time required for distance calculations was .55 milliseconds per point and the average time for combinations was .16 milliseconds per points. Thus, computa-

Memory Requirements

	Local Constraints				
Vector	I	II	III	IV	Itakura
Accumulated Distance	2	2	2	3	1
Local Distance	1	0	1	2	0
Side Information	0	0	0	0	1
Total	3	2	3	5	2

Table 4.2

tional reductions due to fewer distance calculations will have a 350% stronger effect that computational reductions due to less combinations. Table 4.3 summarizes the average time (per dynamic time warp) for combinatorics, as a function of axis orientation. Weighting function c ($\tilde{W}(k)=i(k)-i(k-1)$) was used for all comparisons. We observe that both Itakura's local constraints and type II local constraints are the fastest, followed by local constraints of types I and III, and finally by local constraints of type IV. We also observe that there is no significant difference in combinatorics between those DTW algorithms which use reference along the x-axis and those which use test along the x-axis.

Another factor in the combinatorics time for a DTW algorithm is the particular choice of weighting function. Table 4.4 summarizes the effects of the different weighting functions on the combinatorics times for local constraints I and II. We observe that weighting functions a, b and c perform nearly the same and that weighting function d performs somewhat more slowly.

The final factor which affects the combinatorics time is the presence or absence of range limiting. Figure 4.1 illustrates the effects of an absolute time difference, as defined in Eq. (2.15), on both the combinatorics time and the time for local distance computations. The values for Figure 4.1 were all calculated using local constraints of type III and weighting function c. $R = \infty$ is used to denote that no range limitation was in effect. In parts a and b of Figure 4.1 we observe that relaxing the range limit increases both the combinatorics time and the number of local distance calculations, but in part c we observe that the average combinatorics time per local distance calculation is reduced as the range limitation is relaxed. These results may be explained as follows. First, since a more relaxed range limitation increases the global range more total computation must be performed. However, there is a fixed amount of computation, independent of the global range, and the time required for this computation becomes proportionally smaller as the global range increases.

In Figure 4.2 we show how range limitations affect the size of the global range, i.e. the number of local distance calculations. In part a we observe, for a fixed N and M, that mild

Combinatorics Time by Local Constraints

Local Constraints

Orientation	I	II	III	IV	Itakura
Reference Along x-axis	85.1	63.2	82.8	249.3	63.2
Test Along x-axis	86.5	63.6	83.0	-	63.7
Average	85.8	63.4	82.9	249.3	63.5

Average Time (Milliseconds) Per warp

Table 4.3

Combinatorics Time by Weighting Function

	Weighting Function			
Local Constraints	a	b	c	d
I	90.2	80.6	85.1	90.8
II	57.8	65.9	63.2	69.6
Average	74.0	73.3	74.2	80.2

Average Time Per Warp (Milliseconds)

Table 4.4

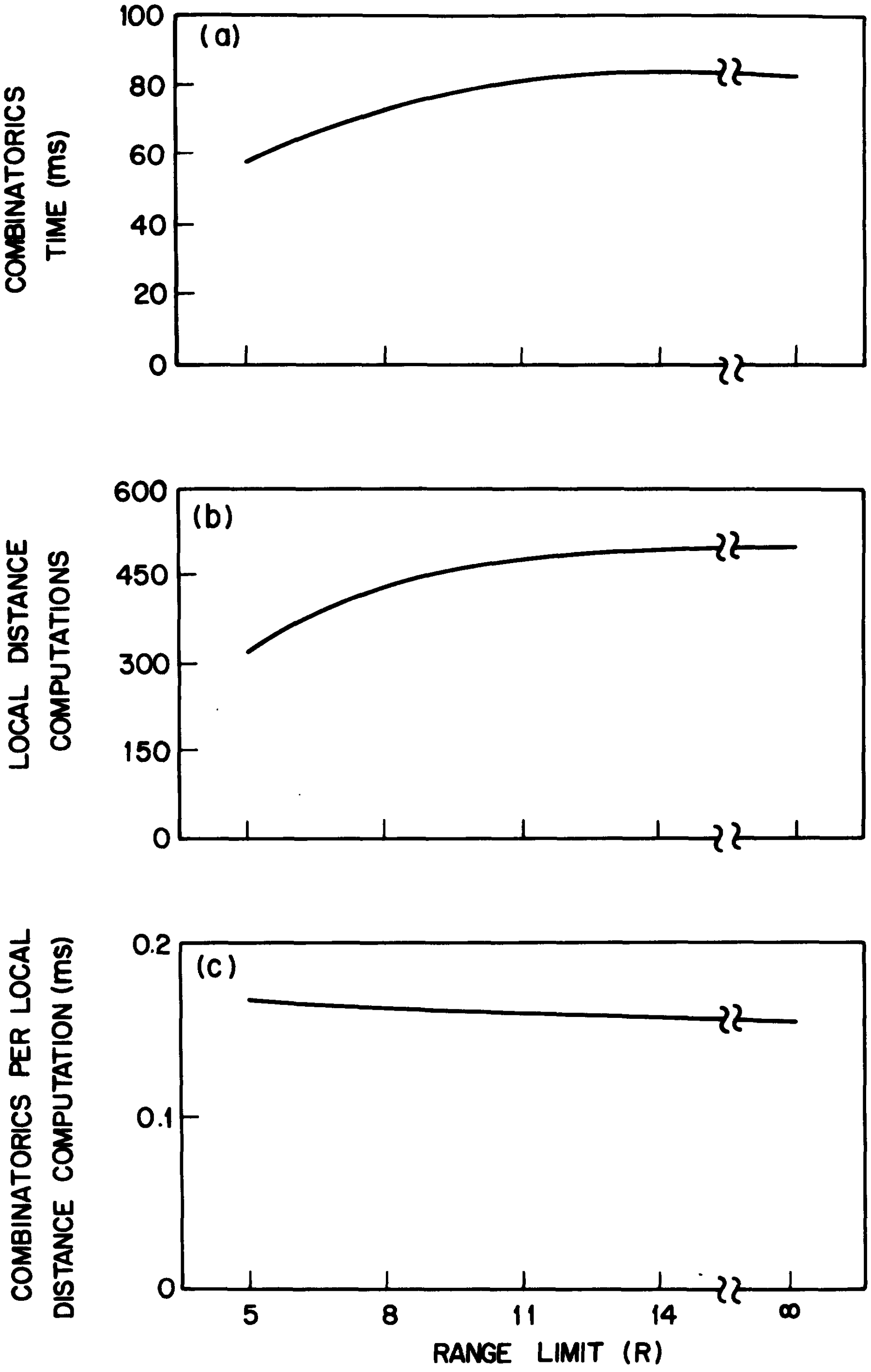


Fig. 4.1 Plot of computational time versus range limit.

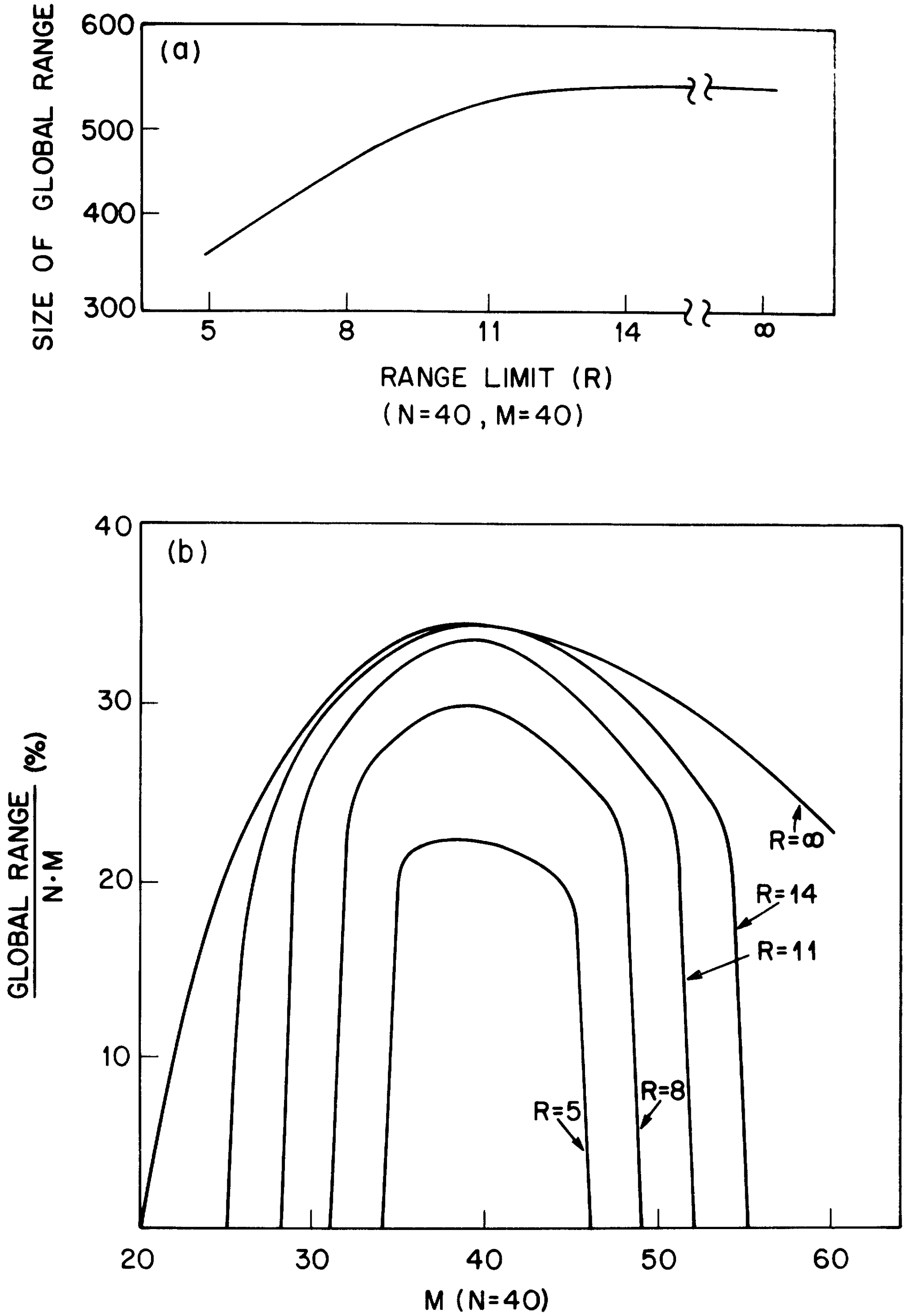


Fig. 4.2 Plot of global range versus range limit and *M*.

range restrictions ($R=11$, $R=14$) do not significantly reduce the global range. In part b we vary both R and M and observe that, as expected, as N/M increases or decreases from 1, the relative space available for time warping paths (i.e. the size of the global range divided by $N \cdot M$) decreases. We also observe that when $|N-M|$ approaches R, a very sudden reduction in the global range occurs. Thus, the largest global range occurs when $N = M$ and R is very large (infinite).

In addition to range limitations, the choice of local constraints also has an effect on the number of local distance calculations. This is summarized in Table 4.5. Because local constraints of type IV have a greater maximum slope, they have a larger global range and thus require more distance calculations. There is also differentiation among the remaining local constraints because the number of intermediate points used in the computation of an accumulated distance function varies with the local constraints. We observe that as the number of intermediate points increases (as in type I local constraints, relative to type II constraints) the number of local distance computations also shows a slight increase.

One final point must be made about the computational efficiency of a DTW algorithm. In the application of a DTW algorithm to a speech recognition problem it is important to know how many time warping calculations are actually required. For example, by severely limiting the global range, many time warps need not be performed because the difference in the lengths of the reference and the test patterns is too large, i.e. $|N - M| > R$. In Figure 4.3 we show how range limitations affect the amount of computation performed (Maximum slope = 2). The entries labeled "same word" and "different word" refer to the percentage of possible time warping computations which are actually performed when the reference and the test were the same word and when they were different words. Figure 4.3 shows a decrease in the number of time warping computations as the range limit is reduced, but that such a decrease occurs more slowly for "same words" than for "different words." The usefulness of range limiting as a method of reducing the number of time warping computations is questionable however, because other methods, such as Itakura's thresholding [6] technique, provide similar, or larger, reductions in the number of time warps for different words (generally, about 50%) at a smaller risk of

Local Distance Calculations

Local Constraint	Average Number of Distance Calculations
I	543.2
II	491.7
III	504.4
IV	781.0
Itakura	504.4

Average Calculations Per Warp

Table 4.5

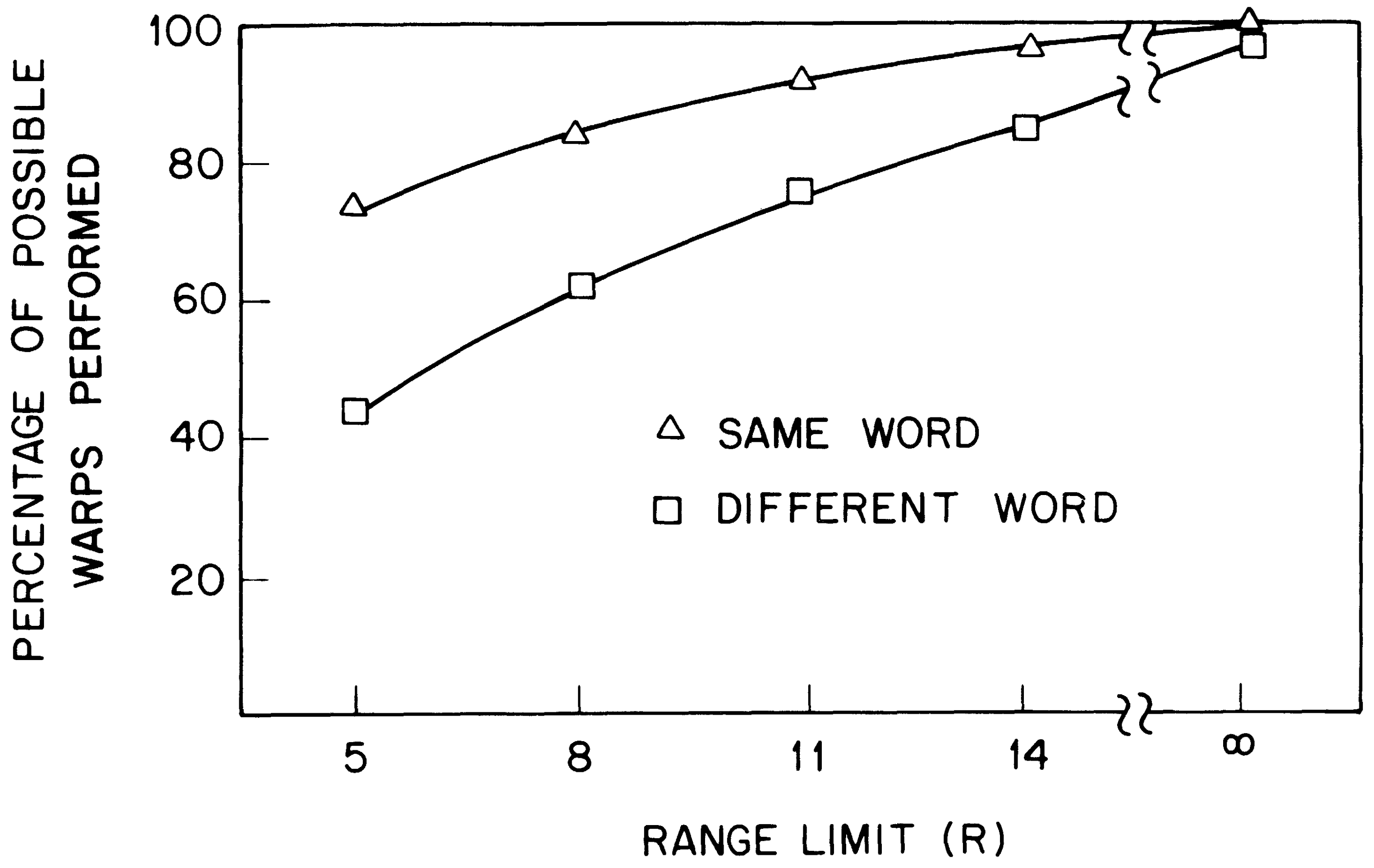


Fig. 4.3 Warps performed plotted as a function of range limit.

eliminating a time warp when the words are the same (generally, less than 5%) [18].

4.2.3 Recognition Accuracy

The third and perhaps the most important criterion in the measurement of the performance of a DTW algorithm is recognition accuracy. In this section we summarize the recognition accuracies of the various DTW algorithms as a function of their different factors. For reference, speakers SD1 and SD2 each had $39 \times 5 = 195$ tokens and speakers SI1, SI2, SI3 and SI4 each had 54 tokens for a total of $195 \cdot 2 + 54 \cdot 4 = 606$ tokens.

The first factor investigated was local continuity constraints. In Figure 4.4 we summarize the error rates for local constraints of types I, II, III and those of Itakura (all have maximum slope S=2). We also show the effect of axis orientation on the recognition error rate. Weighting function c and no range limit were used for all comparisons. From Figure 4.4 we see only slight differences in error rate as a function of the local constraints, with local constraints of type I having the smallest error rate. However, it is seen that a consistent improvement in error rate occurs when the test pattern is placed along the x-axis rather than the reference pattern. This effect has been observed in earlier experiments also [19].

The effect of local constraints of type IV is illustrated in Figure 4.5. In this figure we show measured histograms of the dynamic time warp distance for speaker SD1 using local constraints of type III and weighting function c for two cases. Part a is the case in which the reference and the test pattern represent the same word, and part b is the case in which the reference and the test pattern represent different words. In parts c and d we show the results of using type IV local constraints. We observe that the histogram of part c is essentially the same as the one in part a, but that the histogram of part d is shifted significantly towards lower distances than that of part b. Thus, the distributions of parts c and d overlap more than the distributions of parts a and b and, as discussed in Chapter 3, the probability of a miss, P_M, is higher using type IV local constraints. (In fact, we obtained 14 errors using type IV local constraints for speaker SD1 as opposed to 11 errors using type III local constraints, and a total of 40 errors for type IV local constraints as opposed to 37 errors for type III local constraints, using all test utterances).

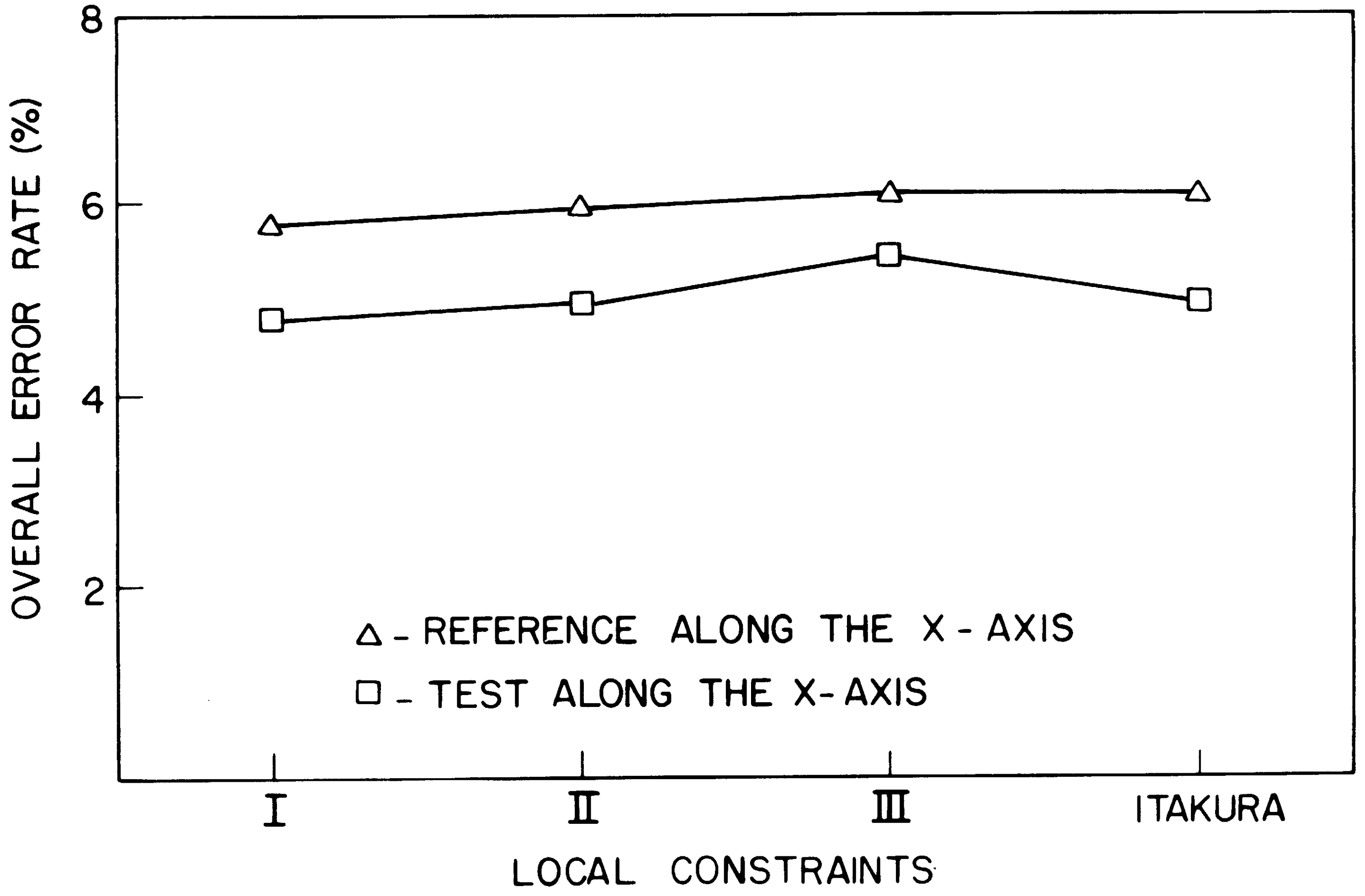


Fig. 4.4 Error rate as a function of local constraints and axis orientation.

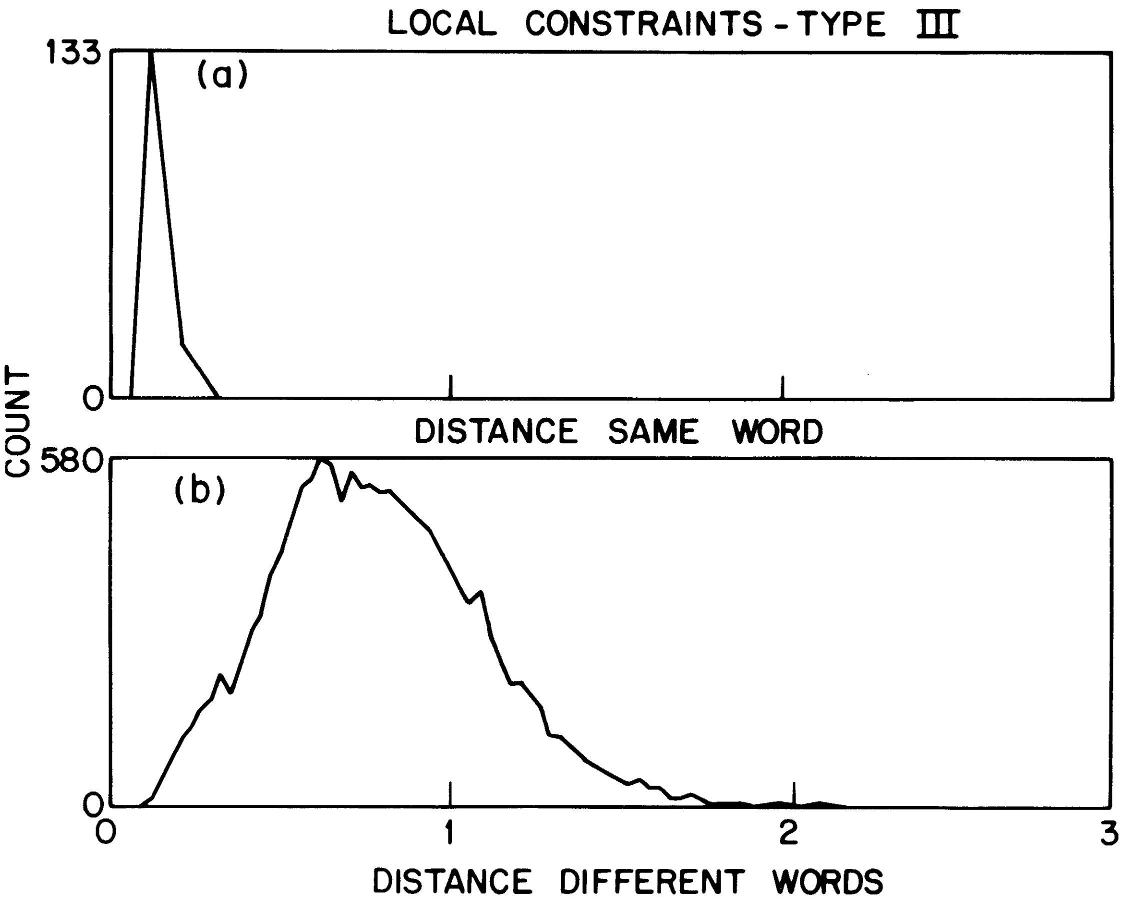


Fig. 4.5 Distance score for local constraints types III and IV.

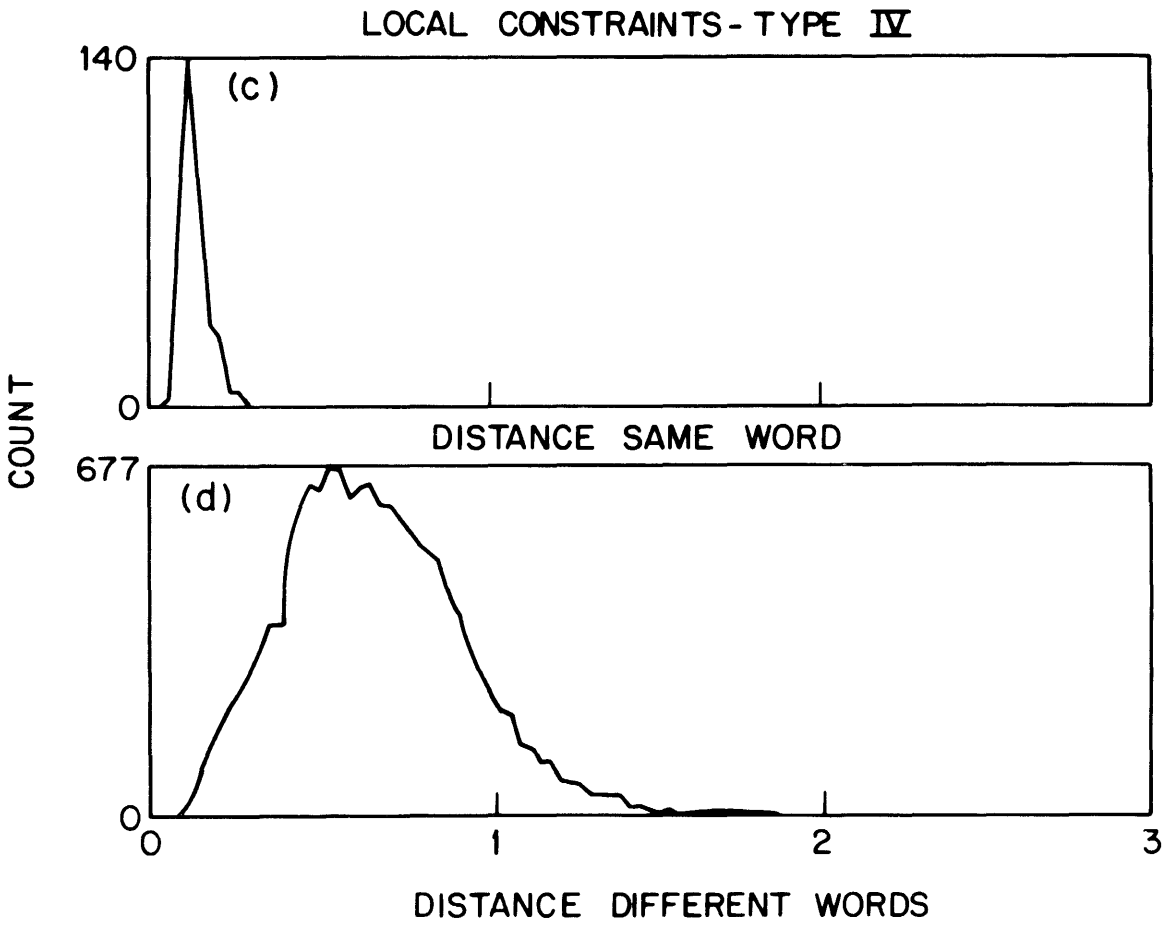


Fig. 4.5 (Continued)

In Figure 4.6 we show how axis orientation affects a particular DTW algorithm. In particular, we use Itakura's local constraints and TS2 data. In parts *a* and *b* we show histograms of the DTW distance scores when the reference and test patterns represent the same word (part *a*) and when they represent different words (part *b*). Both graphs were prepared with the reference pattern along the *x*-axis. In parts *c* and *d* we show the same histograms when the test pattern is along the *x*-axis. It can be seen that the separation of the histograms is somewhat larger when the test is along the *x*-axis than when the reference is along the *x*-axis. This larger separation results in a reduction of the probability of a miss, P_M, from .087 to .079 with a corresponding reduction in the number of errors for TS2 from 20 errors to 11 errors.

In order to understand why axis orientation is important we must examine the effects of the choice of weighting function on the performance of the various DTW algorithms. In Table 4.6 we show the total number of errors for local constraints I and II as a function of the choice of weighting function. The results for weighting functions *a*, *b*, *c* and *d* were computed with reference along the *x*-axis. Weighting function *c'* is weighting function *c* as computed with test along the *x*-axis. Weighting function *c* is used twice because it is the only asymmetric weighting function. We observe that, while weighting function *c* is worst for reference along the *x*-axis, it is the best overall when computed with test along the *x*-axis. Thus, we must conclude that improvements in recognition accuracy that occur when the test pattern is along the *x*-axis are due to some property of weighting function *c*. As Sakoe and Chiba observed, the use of weighting function *c* is equivalent to integration along the *x*-axis [10]. Based on the above reasoning, we conclude that by applying an equal weight to all test frames, a better differentiation between "same" and "different" pairs is achieved. Examination of Table 4.6 reveals another interesting result regarding the performance of a DTW algorithm as a function of the choice of weighting function. We observe that, in agreement with the results previously reported by Sakoe and Chiba [10], a symmetric weighting function (weighting function *d*) performs better than an asymmetric weighting function (weighting function *c*) when the reference pattern is placed along the *x*-axis, but that biased weighting functions (weighting functions *a* and *b*) perform better, not worse, than unbiased weighting functions. However, since the largest

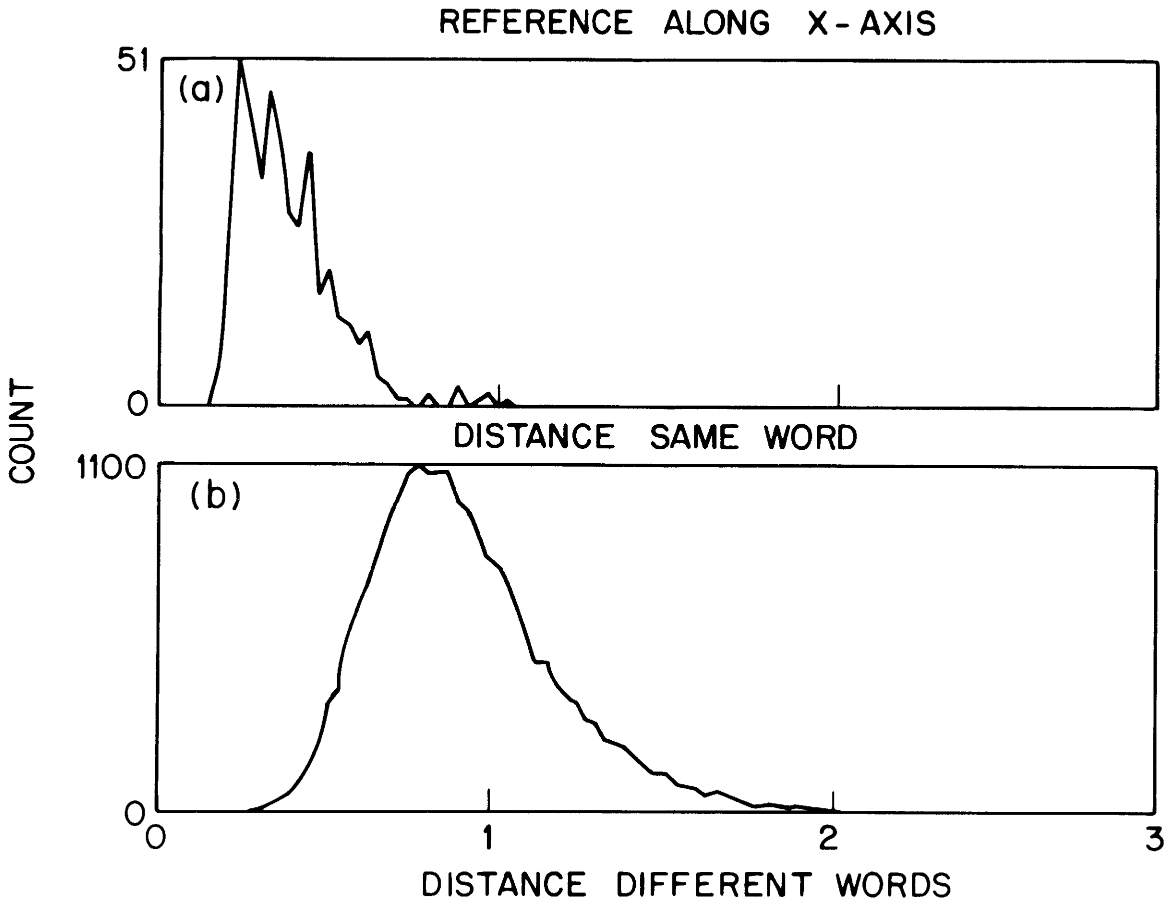


Fig. 4.6 Distance scores for reference along x-axis and test along x-axis.

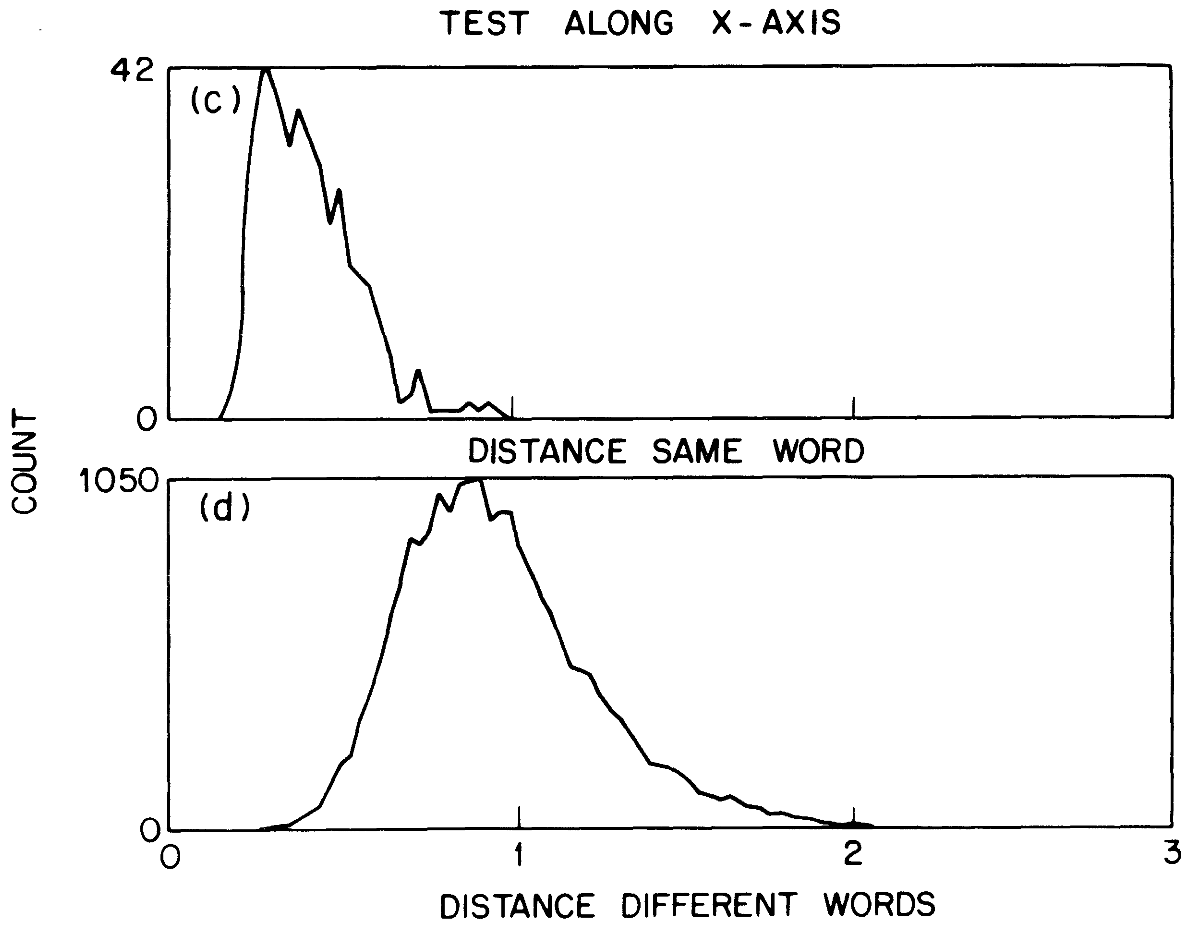


Fig. 4.6 (Continued)

Error Rates by Weighting Function

	Weighting Function				
Speaker	a	b	c	d	c’
SD1	22	21	21	22	23
SD2	8	7	8	9	7
SI1	11	8	10	10	10
SI2	7	12	14	10	7
SI3	12	13	14	12	10
SI4	5	1	4	4	2
Total	65	62	71	67	59

Total Errors

Local Constraints Types I and II

Table 4.6

difference between weighting functions is small, i.e. less than 15% of the error rate, and since the relative error rates are not constant over all speakers we conclude that there is no significant difference in the performance of a DTW algorithm regarding the choice of weighting function but that the combination of test along the x-axis and weighting function c provides significant improvement in recognition accuracy.

The final factor which affects the performance of a dynamic time warping algorithm is the global range. We have already shown that increasing the global range by increasing the slope constraint from $S = 2$ (local constraints of type III) to $S = 3$ (local constraints of type IV) does not improve recognition accuracy. The other factor in the range of a DTW algorithm is the presence or absence of a range limit. In Figure 4.7, part a, we show that as the range limitation is relaxed ($R=\infty$ denotes no limiting), the error rate decreases (using type III local constraints, reference along the x-axis and weighting function c). Thus, we observe that too much restriction on the range of a DTW algorithm is harmful to it's performance. This is also illustrated in part b of Figure 4.7. In this figure we show a plot of the average DTW distance as a function of the ratio of the length of the reference to the length of the test, N/M (when reference and test are the same words). It is seen that the average distance increase as the ratio N/M approaches 1/2 or 2. This result is expected since, as demonstrated in Figure 4.2, the range of a DTW algorithm decreases very rapidly as N/M approaches 1/2 or 2. Thus, as N/M approaches 1/2 or 2, there is very little area (in the (n,m) plane) in which to search for a good time warping path.

4.3 Discussion

Examination of the previous results shows that at least one major tradeoff in DTW algorithm implementation. This tradeoff is illustrated in part a of Figure 4.7 and in parts a and b of Figure 4.1. We observe that, as a more severe range limitation is imposed on a DTW algorithm the amount of computation required decreases but the error rate increases.

We would like to find a method of time warping which is not so severely affected in its recognition accuracy by range limitations. Figure 4.7, part b, suggests that a DTW algorithm

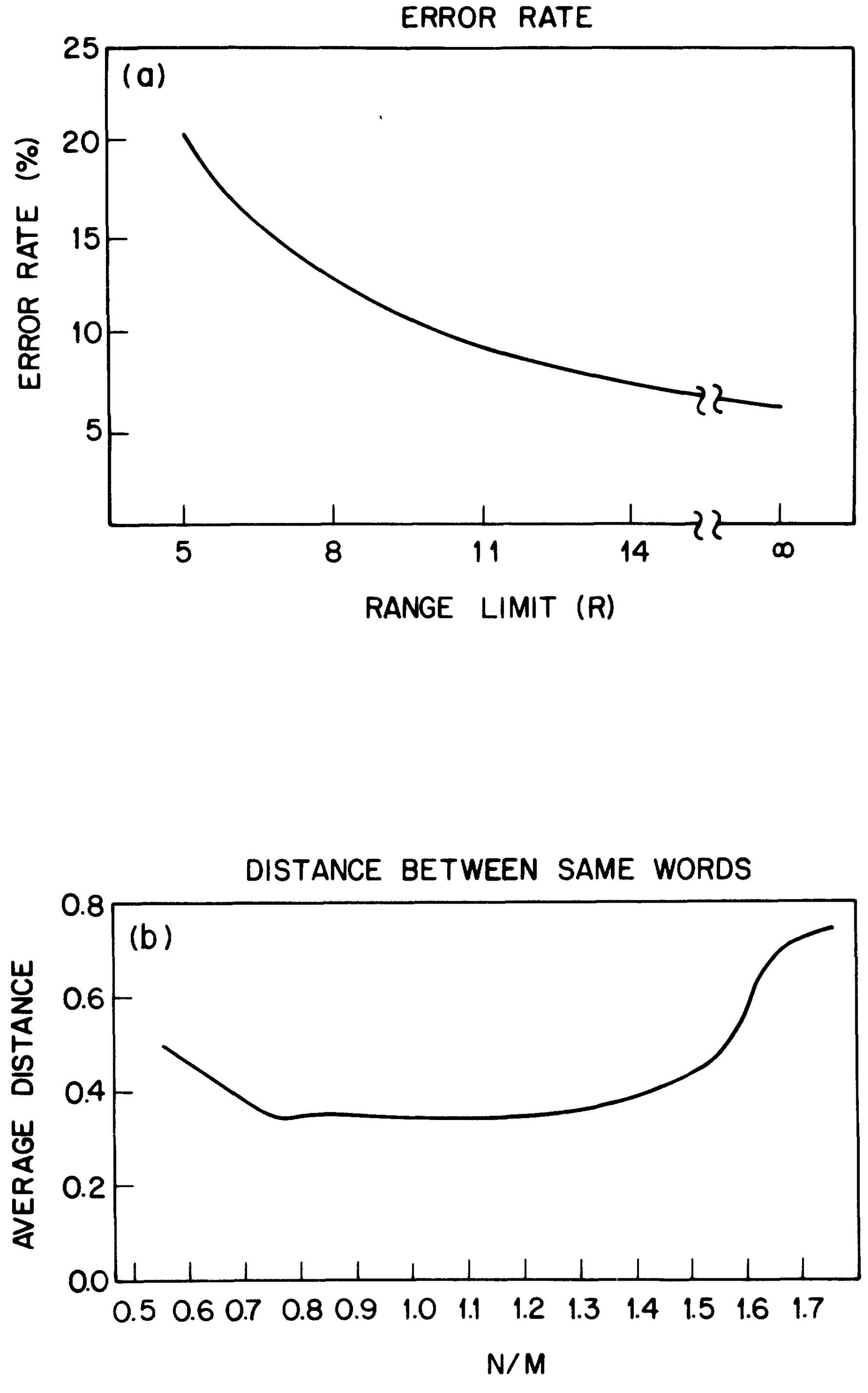


Fig. 4.7 Error rate plotted as a function of range limit and distance as a function of N/M.

which operates with nearly equal size reference and test patterns should perform very well. In the next section we describe such a method and give measurements of its performance.

4.4 Normalize/Warp Algorithm

The first step in the new algorithm is to linearly interpolate (or decimate) the length of all reference and test patterns to a fixed length so that the resulting length ratio, $(\hat{N}/\hat{M})$, is 1. If we denote the fixed length as $\hat{N}$, then the normalized reference pattern, $\hat{\mathbf{R}}(n)$, is given by

$$\hat{\mathbf{R}}(\hat{n}) = (1-s)\cdot\mathbf{R}(i) + s\cdot\mathbf{R}(i+1), \hat{n} = 1,...,\hat{N} \tag{4.1}$$

where

$$i = \lfloor(\hat{n}-1)\cdot\frac{(N-1)}{(\hat{N}-1)}+1\rfloor \tag{4.2a}$$

$$s = (\hat{n}-1)\cdot\frac{(N-1)}{(\hat{N}-1)} + 1 - i \tag{4.2b}$$

and $\lfloor x \rfloor$ is the greatest integer less than or equal to x. A similar transformation is applied to $\mathbf{T}(m)$ to yield $\hat{\mathbf{T}}(\hat{m}), \hat{m} = 1,2,...,\hat{M}$ where $\hat{M} = \hat{N}$.

It can be shown that simple linear interpolation (or decimation) as described in Eq. (4.1) is adequate by examining the log spectrum of any component of the reference template $\mathbf{R}(i)$. Such a spectrum is shown in Figure 4.8. The log spectrum is obtained as the log magnitude of the Fourier transform of the time signal $R(n,i)$ the i^{th} feature of the vector $\mathbf{R}(n)$. Figure 4.8 shows that the log spectrum is strongly bandlimited. It is seen that the log spectrum, for this example, is down 30 dB for frequencies above .18 of the sampling frequency (6.67 kHz). Hence, for such a time signal, simple linear interpolation (or decimation) is entirely adequate so long as the sampling rate does not change by a large factor. For this implementation $\hat{N}$ was chosen to be 40 frames (the average length of all words) so that the ratio $N/\hat{N}$ (i.e. the interpolation or decimation ratio) always fell within a range from 1/2 to 2 for all words in both TS1 and TS2.

After the reference and the test patterns were all normalized to the fixed length, $\hat{N}$, a DTW algorithm was applied to the normalized patterns. We refer to this entire process as a

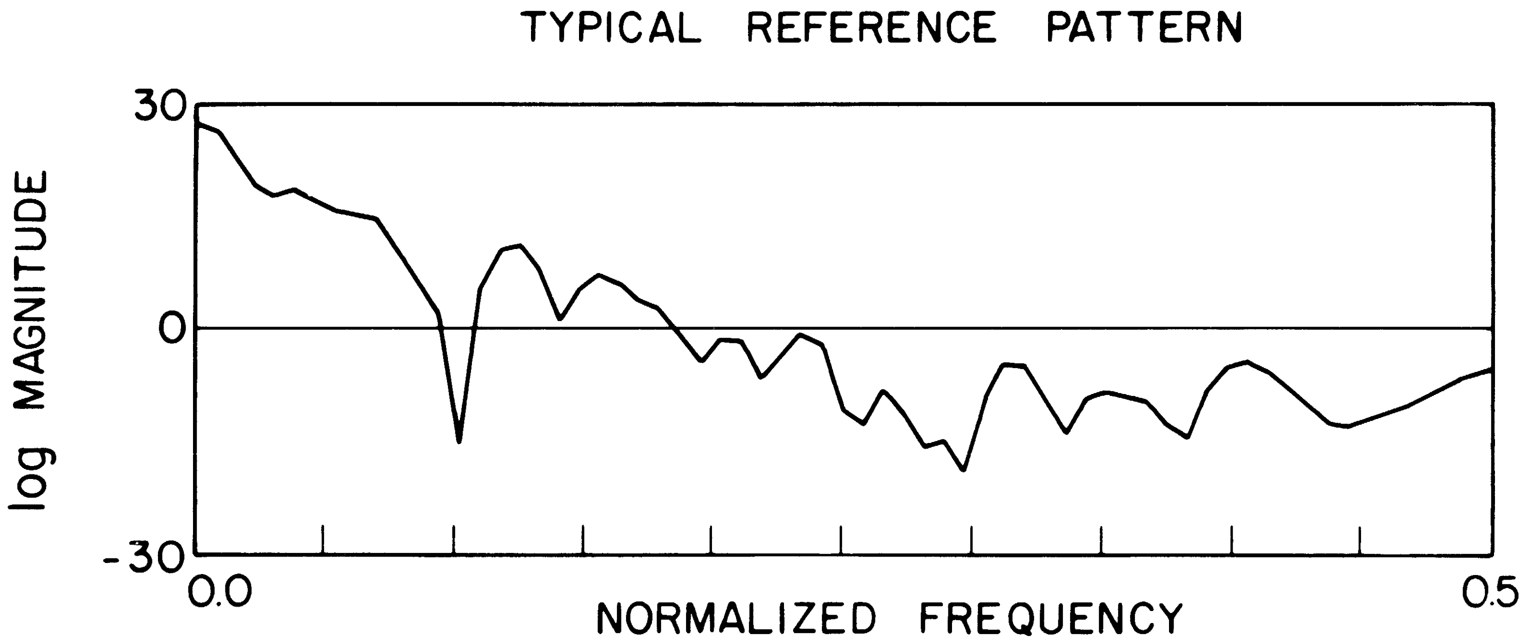


Fig. 4.8 Log magnitude of the transform of a typical reference pattern component.

normalize/warp algorithm. Examination of the performance of the normalize/warp algorithm reveals some very interesting results. The memory requirements for the normalize/warp algorithm are, on average, the same as those for the unnormalized version. However, since all reference and test patterns are of the same length, and since all accumulated distance function vectors and local distance vectors are of the same length the implementation of the algorithm is greatly simplified. This feature is very important for hardware applications.

The computational efficiency of the normalize/warp algorithm is, on average, the same as the efficiency of the unnormalized algorithm. Reference patterns may be normalized during training of the system and the normalization of a test pattern (for testing/requires a negligible amount of time (less than 5 milliseconds in our applications). Local distance calculations and computations of the accumulated distance function are the same as in the average unnormalized case and, in fact, range limit calculations may be reduced because the global range is fixed by the fixed lengths of the reference and test patterns.

Two new considerations to the computational efficiency must be made with the normalize/warp algorithm. First, because the lengths of the test and the reference patterns are all fixed, there is no need to normalize the accumulated distance score in order to compare distance scores. Second, because we choose to normalize the lengths of the reference and the test patterns to the same length no computation may be eliminated because the lengths of all the reference and all the test patterns are compatible. However, as we stated previously there are methods which are more effective in reducing the number of time warps computed (e.g. Itakura's thresholding technique [6]) than simple comparison of the lengths of the reference and the test patterns.

The effect on recognition accuracy for the normalize/warp algorithm is shown in Figure 4.9. We have shown both the unnormalized warping algorithm results (normal line) and the normalized warping algorithm results (dashed line). We observe that, in general, the normalize/warp algorithm performs *slightly better* in *almost all* cases.

In Figure 4.10 we show the most important property of the normalize/warp algorithm. In

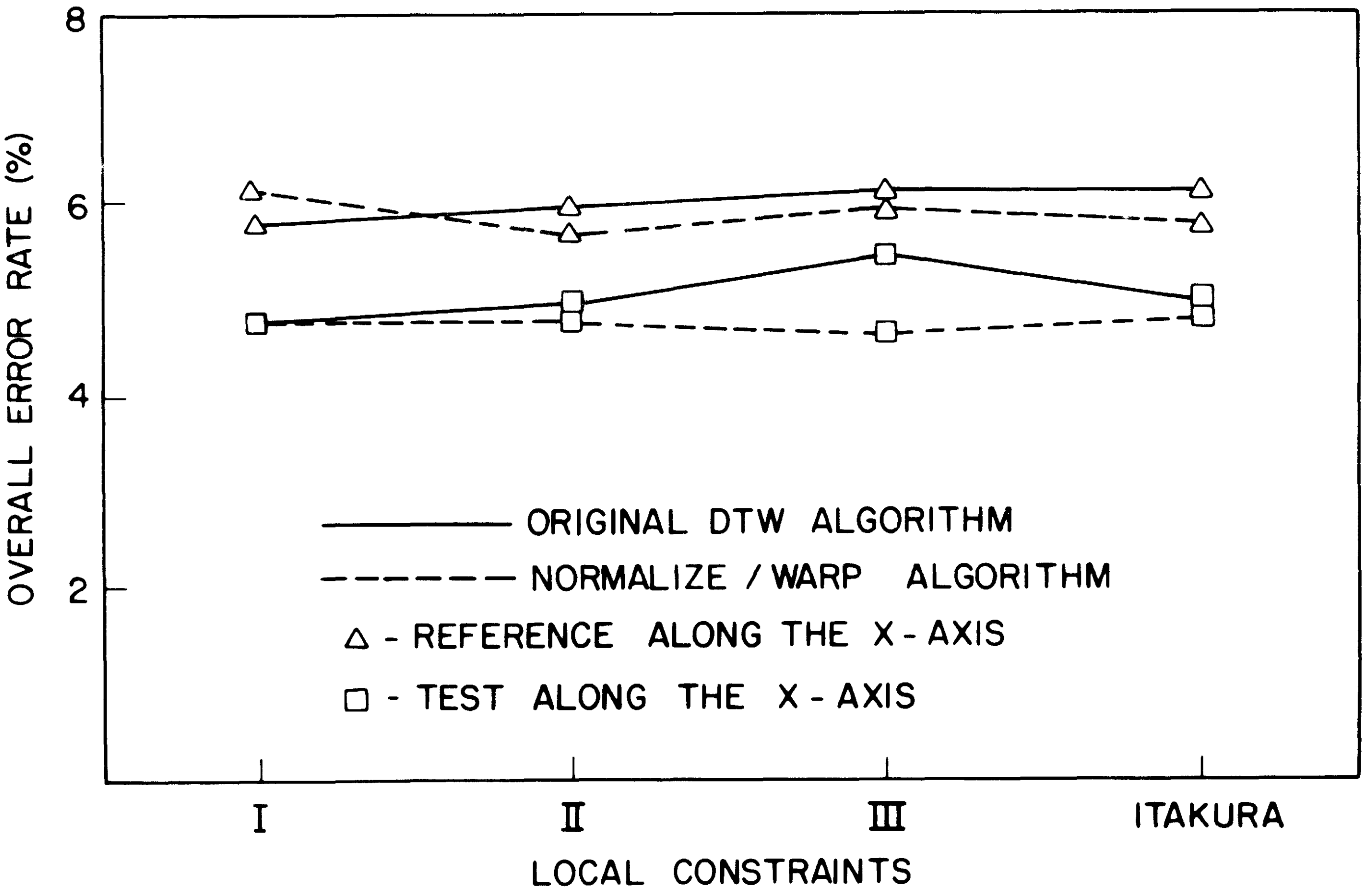


Fig. 4.9 Error rate of the normalize/warp algorithm.

this figure we show the effect of range limitations on the recognition accuracy of both the unnormalized warping algorithm (solid line) and on the normalized warping algorithm (dashed line). We observe that a range limitation, applied to the normalized axis, is not harmful to the normalize/warp algorithm, but is, in fact, somewhat helpful to its performance.[1] Such a result has strong practical significance because, as we have already shown, range limiting is a useful technique for reducing the amount of computation (combinatorics and local distance calculations) required by a DTW algorithm.

In the next chapter we will summarize the results of this chapter, give implications of these results, and suggest further areas for investigation. In Appendix 2 we give complete tables for all of the results reported in this chapter.

[1] Obviously, if improvement were to continue until $R = 0$ we would have shown that linear normalization performs better than dynamic time warping. This is *not* the case, however. For values of R less than 5 we found that the error rate increased, but not extremely quickly, rising from 5.3% when $R = 5$ to only 8.9% when $R = 1$.

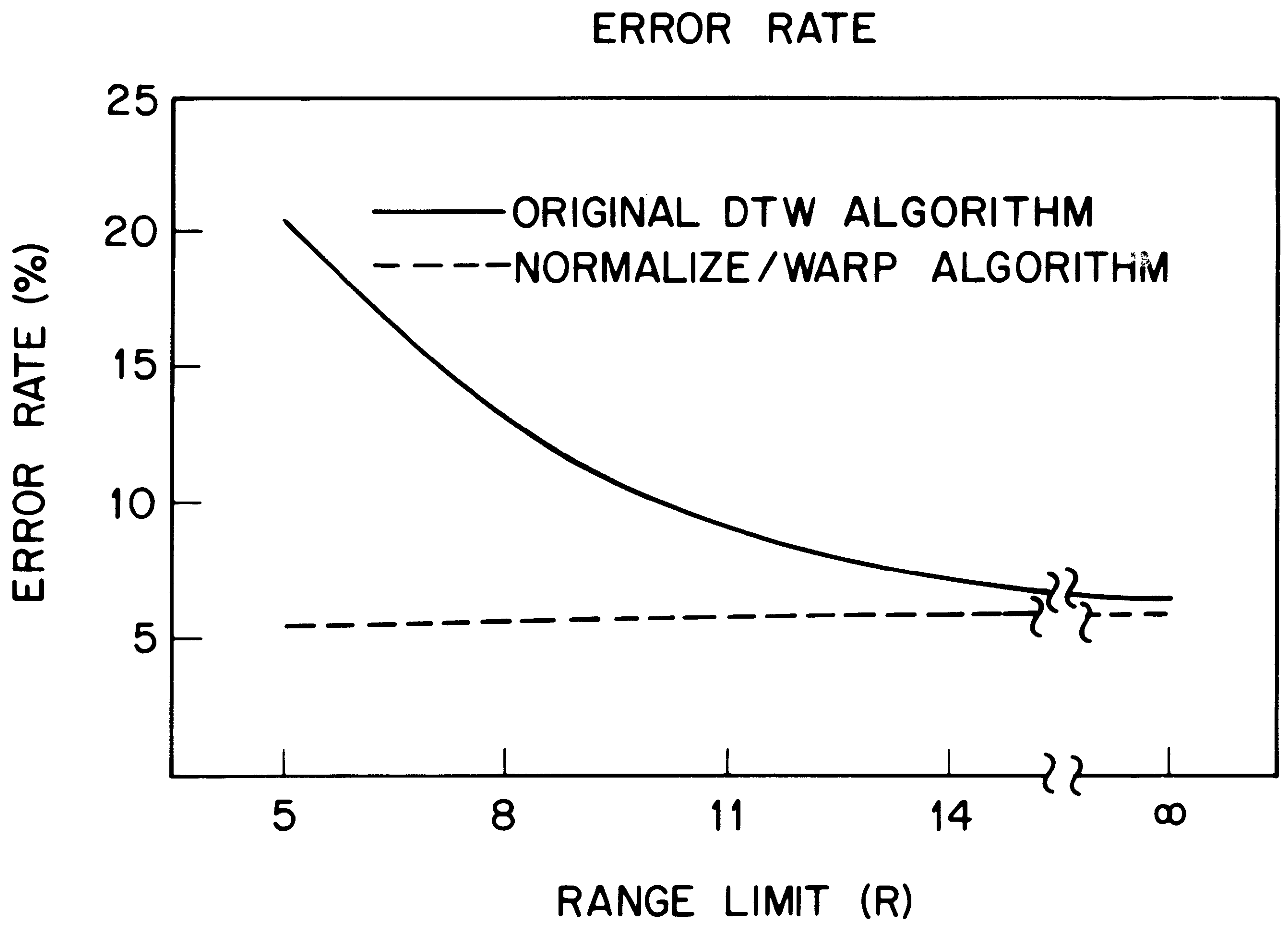


Fig. 4.10 Normalize/warp results as a function of range limit.

Chapter 5

Summary of Results on Dynamic Time Warping for Isolated Word Recognition

5.1 Introduction

In Chapters 2-4 we presented numerical results comparing the performance of several time warping algorithms for isolated word recognition. We examined the effects of several implementation factors, including, local continuity constraints, global range constraints, axis orientation and choice of distance measure, on the performance of DTW algorithm. In this chapter we summarize and make some comments on results given in Chapter 4.

5.1.1 Memory Requirements

As far as the memory requirements of a DTW algorithm are concerned, it was shown that the "simpler" (fewer and smaller productions) the local constraints used in the specification of a DTW algorithm, the smaller the memory requirements of that algorithm. No other factor in the implementation of a DTW algorithm which contributed significantly to differences in memory requirements was found in the investigations.

5.1.2 Computational Efficiency

The computational efficiency of a DTW algorithm was shown to be dependent upon the local constraints of the DTW algorithm in the same manner that memory requirements were dependent upon the local constraints, i.e. the "simpler" the local continuity constraints that are used, the more computationally efficient (less combinatorics, fewer local distance calculations) the DTW algorithm. We also found that computational efficiency was relatively insensitive to axis orientation or to the choice of weighting function. In addition, we found that the imposition of a global range limit is effective in reducing the total amount of computation. This effect is achieved by reducing both the amount of combinatorics and the number of local distance calculations.

5.1.3 Recognition Accuracy

The results for recognition accuracy are best summarized according to the various factors of a DTW algorithm which were defined in Chapter 1. The results are as follows:

1. Local Continuity Constraints — We found that more complicated local constraints (Type I) provide only slightly better accuracy than more simply defined local constraints. Also, Type IV local constraints (maximum slope, $S=3$) performed worse than the other local constraints (maximum slope, $S=2$). This result agrees with earlier work by Sakoe and Chiba [10].

2. Global Range Constraints — We found that the presence of an absolute time difference range limit was detrimental to the performance of a DTW algorithm. We also found that a DTW algorithm is able to provide a better match (lower distance) between words when the ratio of the length of the test pattern to the length of the reference pattern close to 1, than when it is close to 2 or 1/2.

3. Axis Orientation — We have found that placing the test pattern along the x-axis (when combined with weighting function c) performed significantly better than placing the reference pattern along the x-axis (with any choice of weighting function).

4. Distance Measure — We found that the performance of a DTW algorithm was relatively insensitive to the choice of weighting function, including those weighting functions which were biased. However, because weighting function c is asymmetric it was the only one affected by axis orientation. For this weighting function better recognition accuracy was obtained when it was combined with having the test along the x-axis than for any other weighting function.

5.1.4 Tradeoffs

Examination of the previous results reveal two areas in which tradeoffs are possible in the specification of a DTW algorithm, and two areas in which clear out choices are obvious. Since recognition accuracy is improved by the combined choice of weighting function c and test along the x-axis, (with no apparent cost in either memory requirements or computational efficiency), these two options should always be used. It is also worth noting that the combined choice of weighting function c and test along the x-axis eliminates the need for normalization because the normalization factor in Eq. (2.17) ($\mathrm{N}(\tilde{W}_c) = M$ for test along the x-axis) is constant for any given test utterance.

Tradeoffs exist, however, in the choice of local continuity constraints. It was shown that it is possible to improve the computational efficiency and reduce memory usage by the choice of a "simple" set of local continuity constraints (e.g. Itakura's or Type II local constraints) but these cases lead to a slight decrease in recognition accuracy.

Range limiting also presents a similar tradeoff. Inclusion of a range constraint helps to limit the amount of computation required by a DTW algorithm but at a cost of reduced recognition accuracy. However, we have suggested a method, the normalize/warp algorithm, in which range limiting can be used without the loss of recognition accuracy. We summarize our results on the normalize/warp algorithm in the next section.

5.2 Results Concerning the Normalize/Warp Algorithm

Our investigations have shown that a normalize/warp DTW algorithm has about the same memory requirements as a normal DTW algorithm, but that this storage may be organized more simply in the normalize/warp algorithm. We have also shown that a normalize/warp algorithm is at least as computationally efficient as a regular DTW algorithm and may be made more so by the careful use of the knowledge that the lengths of the reference and the test patterns are fixed. More importantly, we found that the normalize/warp algorithm consistently improved recognition accuracy and that this performance was not degraded by the addition of a range limit.

5.3 Practical Significance

By utilizing the results which we have presented in this thesis we are able to define the type of DTW algorithm which would perform well in many isolated word recognition systems. This DTW algorithm would use normalized length reference and test patterns, utilize weighting function c with the test pattern along the x-axis, impose a moderate size range constraint and would use a "simple" local continuity constraint (Type II or Itakura's). Such a DTW algorithm would provide recognition accuracy comparable to, or better than, most other DTW algorithms, be fairly simple to organize and implement in hardware, and would operate as, or more, efficiently than any other DTW algorithm for isolated word recognition.

5.4 Future Research Areas

We have presented important results in the study of the tradeoffs involved in the implementation of a DTW algorithm for isolated word recognition. Some further issues still should be investigated. Perhaps the major question to be answered is why DTW algorithms which use the test pattern along the x-axis perform better than those DTW algorithms which use the reference pattern along the x-axis. Hopefully an answer to this question will give more insight into the DTW algorithms which we have studied and might suggest new approaches to the problem. Another question directly related to the results of this thesis is in the choice of an optimal interpolation or decimation technique for the normalize/warp algorithm. Finally, it would be important to know if, as assumed, our results for dynamic time warping algorithms are applicable to other sets of feature vectors and distance metrics.

Chapter 6

Issues in Dynamic Time Warping for Connected Speech Recognition

6.1 Introduction

As we have demonstrated in Chapters 2-5, dynamic time warping algorithms can be applied to the problem of isolated word recognition yielding highly reliable and robust recognition systems. We would like, however, to extend the use of DTW algorithms to connected speech recognition and word spotting problems. Bridle [7], and Christiansen and Rushforth [20] have demonstrated effective DTW algorithms for word spotting, and recently Sakoe [21] and Rabiner and Schmidt [22] have successfully applied time warping techniques to connected digit recognition. In this chapter we will describe the basic principles involved in using DTW algorithms for word spotting and connected speech recognition, giving particular emphasis to those factors of the problem which make it different from the isolated word recognition problem. It should be emphasized once again that our basic unit of recognition is a word and, as such, the results which we present may or may not be applicable to other types of recognition systems.

For the remainder of this thesis we shall assume that the test pattern consists of a sequence of connected words, spoken in a normal manner, and that the first frame of the test pattern represents the first frame of the spoken utterance, and that the last frame of the test pattern represents the last frame of the spoken utterance, i.e. the global beginning and ending points of the word sequence have been located properly. Given such a framework, the word spotting problem is to find all subsections of the test pattern which match with a specified reference pattern, called the keyword. Thus for word spotting, a multiplicity of regions in the test pattern must be compared with the keyword pattern. The computation must be carried out in an efficient manner or the problem quickly becomes computationally unfeasible. Furthermore, the possibility of multiple occurrences of the keyword within the test pattern must be taken into account.

The connected speech recognition problem, on the other hand, is to piece together reference patterns (obtained from isolated occurrences of the words) in order to match the test pattern. The general approach to this problem will be the one proposed by Levinson and Rosenberg [23], namely, finding the word that best fits a given section of the test pattern at which the word is postulated to begin, using the ending frame of that word as an estimate of the beginning frame of the next word, and continuing to concatenate reference patterns in this manner until the test pattern is exhausted.

In the next sections we will explain why the DTW algorithms which we have defined for isolated word recognition cannot be directly applied to these problems, and we will propose various methods in which dynamic time warping algorithms are applicable to the word spotting and connected speech recognition problems.

6.1.1 Basic Difficulties in Connected Speech Recognition Using Word Size Templates

Dynamic time warping algorithms, as we have described them thus far, are not directly applicable to the connected speech recognition or word spotting problems. There are two reasons why this is so. Figure 6.1 illustrates some of the problems which are encountered. In this figure the time patterns of the log intensity for two speech utterances, "3", "8", in part a, and "38" in part b are shown. The utterance in part a is spoken as a sequence of isolated words (i.e. there is a discernible pause between the "3", and the "8,"), while the utterance of part b is spoken as a connected word sequence. We observe that, in part b, it would be extremely difficult to obtain a reliable set of beginning and ending points for either the "3" or the "8." Thus, one of our fundamental assumptions in the use of DTW algorithms for isolated word recognition, namely that accurate word boundaries can be obtained, is not valid for connected speech.

Another difficulty in using DTW algorithms, based on isolated word reference templates, for connected speech applications, is the problem of coarticulation between words. For example, the final /*i*/of the word "3" and the initial /e^i/ of the word "8" coarticulate strongly with each other. Thus, another fundamental assumption that has been relied on, namely that the characteristics of the isolated reference words which we are trying to match to our test utterance can

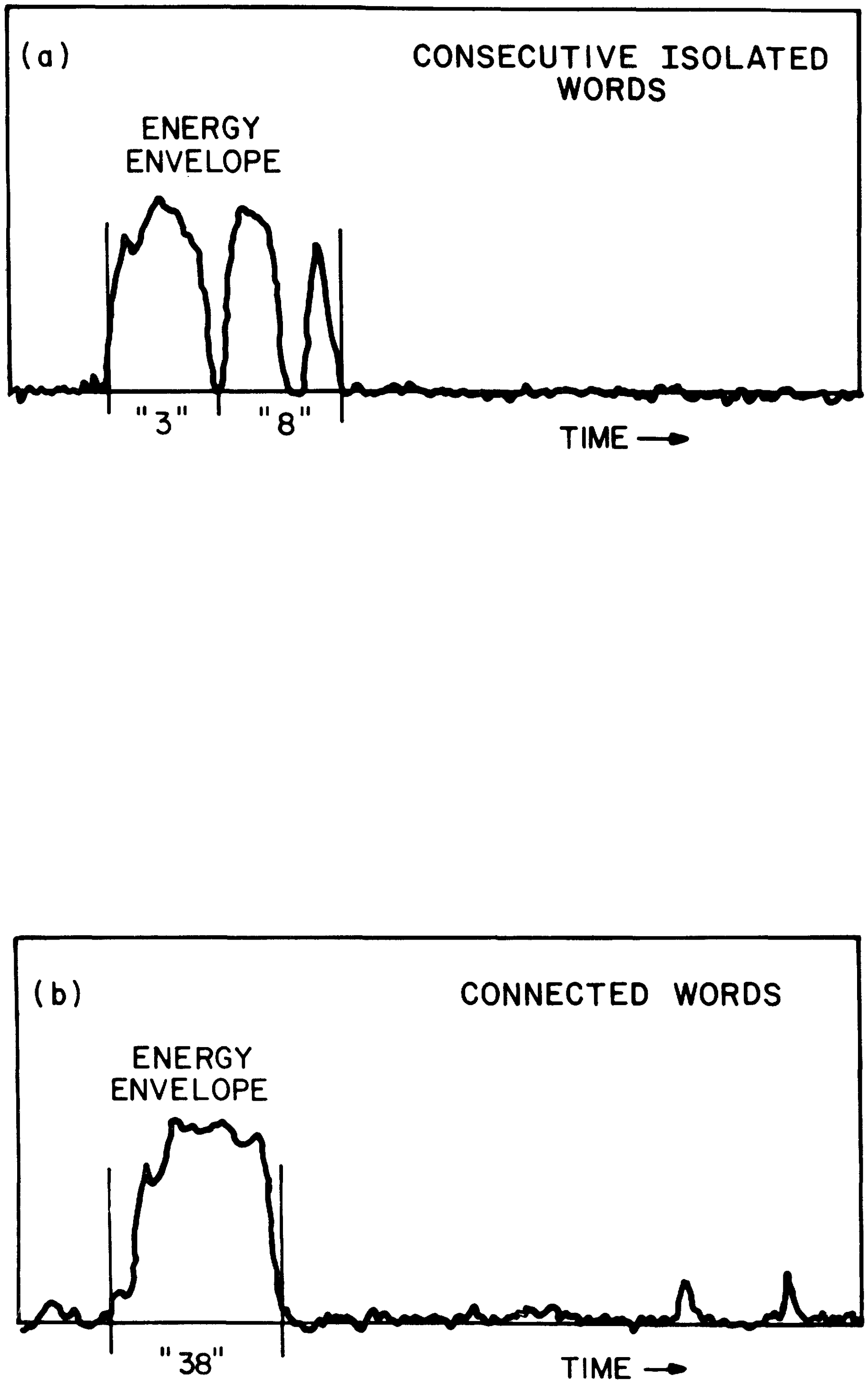


Fig. 6.1 Log energy for two speech utterances.

be truly found in the test pattern, is not valid. In the next section we will describe the basic techniques which will be used to overcome these difficulties.

6.1.2 Basic Approaches to the Problems of Connected Speech Recognition

In our approach to connected word recognition and word spotting using dynamic time warping, we will make two changes from the structure of the isolated word recognizer used in Chapters 2-5. One change is to no longer attempt to find entire isolated reference pattern in the test utterance. We will still use isolated words as our reference patterns but we will expect a good match in the middle of the word only, and not near the ends. (For monosyllabic words, in which the coarticulation effects the entire word, it is possible that no good match will be found.) Thus, we will not require that, by matching a reference pattern to a portion of the test pattern, we will be able to accurately match the beginning and ending points of the reference pattern to points within the test pattern. As a result, we would like to consider the possibility of overlapping reference patterns in order to recognize connected speech. In this manner we can account for both errors in the endpoint locations and for some of the gross features of coarticulation.

Another fundamental difference in the DTW algorithms for word spotting and connected speech recognition is the use of beginning and ending *regions*, rather than beginning and ending *frames*. In this manner, we attempt to avoid problems inherent in requiring an accurate segmentation of the test utterance. The idea of using beginning and ending regions is illustrated in Figure 6.2. A beginning region of size B (frames), with potential starting frames between b_1 and b_2 ($B=b_2-b_1+1$), is specified, and an ending region of size E, with potential ending frames between e_1, and e_2 ($E=e_2-e_1+1$) is also specified. The best match of the reference pattern to the test pattern may begin at *any* frame in the beginning region and end at *any* frame in the ending region. Three such possible paths are shown in Fig. 6.2. It is, of course, possible to expand either the beginning or the ending region to incorporate the entire test pattern. Thus, the framework of Figure 6.2 may be used for either word spotting, in which neither the beginning nor the ending frame is known, or for connected word recognition, in which the ending

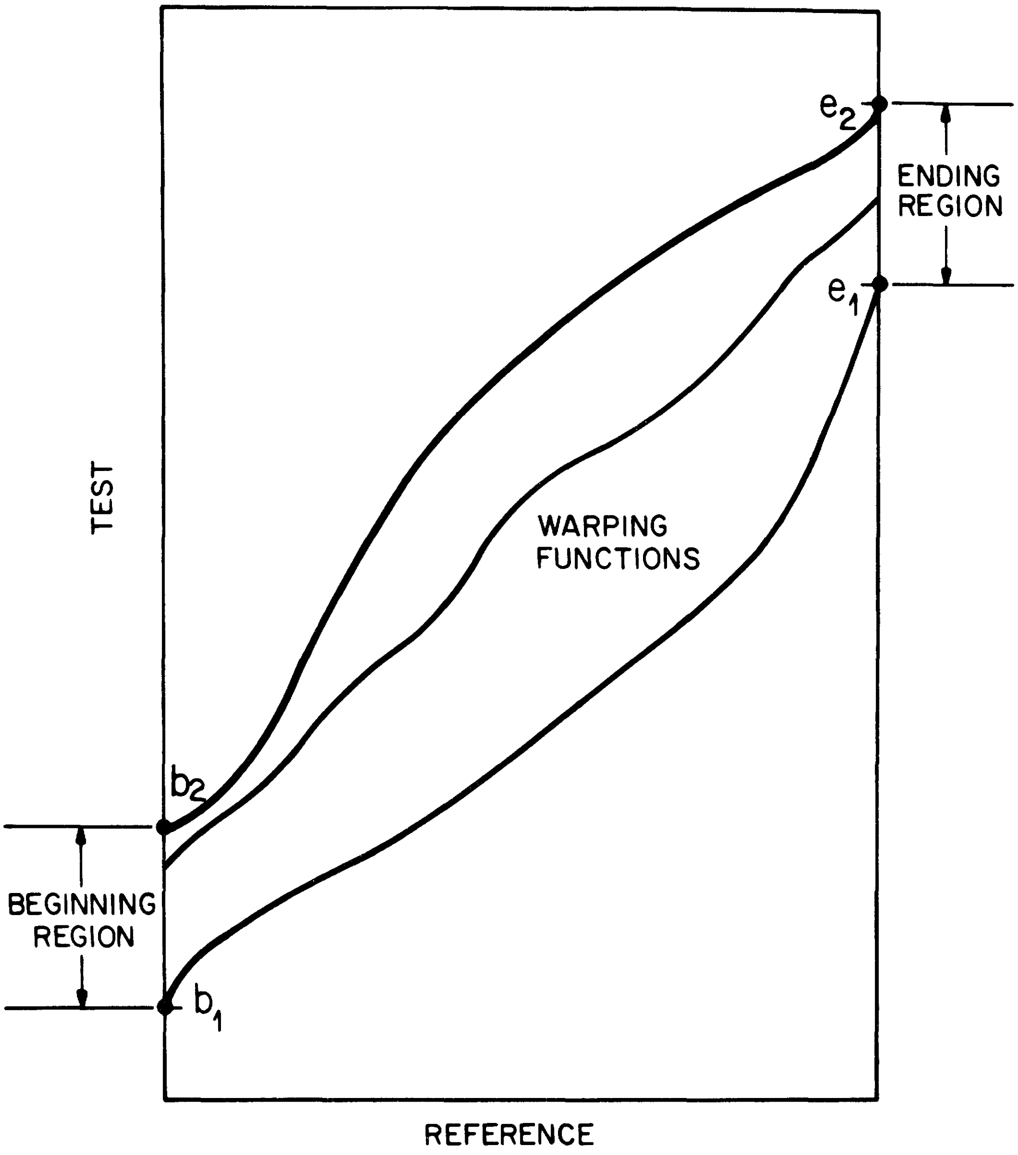


Fig. 6.2 Illustration of the general time warping problem.

frame of one word is used to postulate the beginning frame of the next. In the next section of this chapter we will define various DTW algorithms designed to be applicable to both word spotting and connected word recognition.

6.2 Dynamic Time Warping Algorithms

As we have discussed, the purpose of a DTW algorithm for word spotting or connected speech recognition is to provide the optimal time alignment between a reference pattern, $\mathbf{R}(n)$, and some portion of the test pattern, $\mathbf{T}(m)$. We shall assume, as in Chapter 2, that:

1. The best path is parameterized by the functions $n = i(k)$, $m = j(k)$.
2. The path is restricted to obey some local constraints.
3. The optimal path is chosen to minimize a global distance metric.

$$D(i(k),j(k)) = \frac{\sum_{k=1}^{K} d(i(k),j(k))\,\tilde{W}(k)}{\mathrm{N}(\tilde{W})} \tag{6.1}$$

where, as in Chapter 2, the length of the time alignment path is given by K, $d(i(k),j(k))$ is the local distance metric used to measure dissimilarity between frames $i(k)$ of the reference pattern and $j(k)$ of the test pattern, $\tilde{W}(k)$ is one of the weighting functions defined in Chapter 2, and $\mathrm{N}(\tilde{W})$ is the normalization factor associated with the particular weighting function chosen.

In our study of DTW algorithms for connected speech recognition and word spotting, we shall be using only those local constraints defined in Chapter 2 which limit the slope of the warping function to lie between 1/2 and 2 (types I, II and III and Itakura's). However, based on our results for isolated word recognition, it is felt that the particular choice of local constraints will not be an important factor in the performance of the DTW algorithm.

Since we will be using beginning and ending regions rather than beginning and ending frames, the initial and final values for $i(k)$ and $j(k)$ are defined over the range as:

$$i(1) = 1 \ , \ j(1) = b \ , \ b_1 \leqslant b \leqslant b_2 \tag{6.2a}$$

$$i(K) = N \ , \ j(K) = e \ , \ e_1 \leqslant e \leqslant e_2 \, . \tag{6.2b}$$

Hence, the warping contour begins at the first frame of the reference pattern, and within the beginning region of the test pattern, and ends at the last frame of the reference, and within the ending region of the test.

In the most general case, it is necessary to determine the optimal path by trying every possible beginning and ending point, i.e.

$$\hat{D} = \min_{b_1 \leqslant b \leqslant b_2} [\min_{e_1 \leqslant e \leqslant e_2} [D(i(k),j(k)) s \, . \, t \, . \, j(1)=b \ , \ j(K)=e]] \, . \tag{6.3}$$

$\hat{D}$ of Eq. (6.3) is the distance score of the best possible path using any possible beginning and ending frame. The amount of computation to solve for this best path, however, can be excessive, i.e. theoretically we require $B \cdot E$ separate time warps in the most general case. It is possible, however, to reduce the amount of computation required to solve Eq. (6.3) to a *single* time warp by judicious selection of the weighting function and the axis orientation. If the reference pattern is placed along the x-axis, and $\tilde{W}(k)$ is chosen to be weighting function type c ($\tilde{W}_c(k)=i(k)-i(k-1)$) with $\mathrm{N}(\tilde{W})$ chosen accordingly ($\mathrm{N}(\tilde{W}_c)=N$), then $\hat{D}$ may be computed efficiently by a modified DTW algorithm, as follows:

1. Set $D_A(1,m) = d(1,m)$ for $b_1 \leqslant m \leqslant b_2$.

2. Compute $D_A(n,m)$ recursively for $1 \leqslant n \leqslant N$, $b_1 \leqslant m \leqslant e_2$.

3. $\hat{D} = \frac{1}{N} \cdot \min_{e_1 \leqslant m \leqslant e_2} D_A(N,m)$.

This algorithm works because step 1 initializes all possible beginning points, step 2 computes the best possible path to a point (n,m) from any of the beginning points initialized in step 1, and step 3 finds the best possible ending point along a path from any possible beginning point. The particular choice of weighting function type c and reference along the x-axis is important because only this combination is unbiased and has its normalization unaffected by the choice of beginning and ending points, i.e. the normalization factor is N, regardless of the beginning or ending point. A dependence on the length of the test, on the other hand, would necessitate the use of several separate time warps because the effective length of the test (for normalization purposes) is different for different sets of beginning and ending points. (The effective length is $e-b+1$, where e is the ending point and b is the beginning point.)

We shall assume for the remainder of this thesis that $\tilde{W}(k)$, $\mathrm{N}(\tilde{W})$ and the axis orientation have been chosen as follows:

$$\tilde{W}(k) = i(k) - i(k-1) \quad \text{(Type } c\text{)} \tag{6.4a}$$

$$\tilde{W}(1) = 1 \tag{6.4b}$$

$$\mathrm{N}(\tilde{W}) = N \tag{6.4c}$$

$$\text{Axis Orientation: Reference Along the } x\text{–Axis} \tag{6.4d}$$

As such, a single time warp encompasses the extended parallelogram of Figure 6.3 (assuming local constraints which restrict the slope of the warping function to lie between 1/2 and 2). An important factor to consider in the application of the DTW algorithm is the size of this global range. If both the beginning and ending regions are known, then the size of the global range is somewhat larger than $N^2/3 + B\cdot N$ points in the global range (a typical N by N isolated word DTW algorithm of size $N^2/3$ i.e. single beginning and ending frame, plus the extra region generated by the beginning region), but, if only the beginning region is known, then the size of the global range can be greater than $3N^2/4 + B\cdot N$ points (a single beginning point, which generates a triangle of base $3N/2$ and height N, plus the extra region generated by the beginning region). For many applications, the ending region will not be known, and, as such, a very large amount of computation may be required to do even a single time warp.

Two modifications to the DTW algorithm have been suggested in order to reduce this amount of computation. In particular, Sakoe and Chiba [4] have proposed that a time warping path not be allowed to create excessive time differences, i.e. for any $i(k)$, $j(k)$ is restricted such that

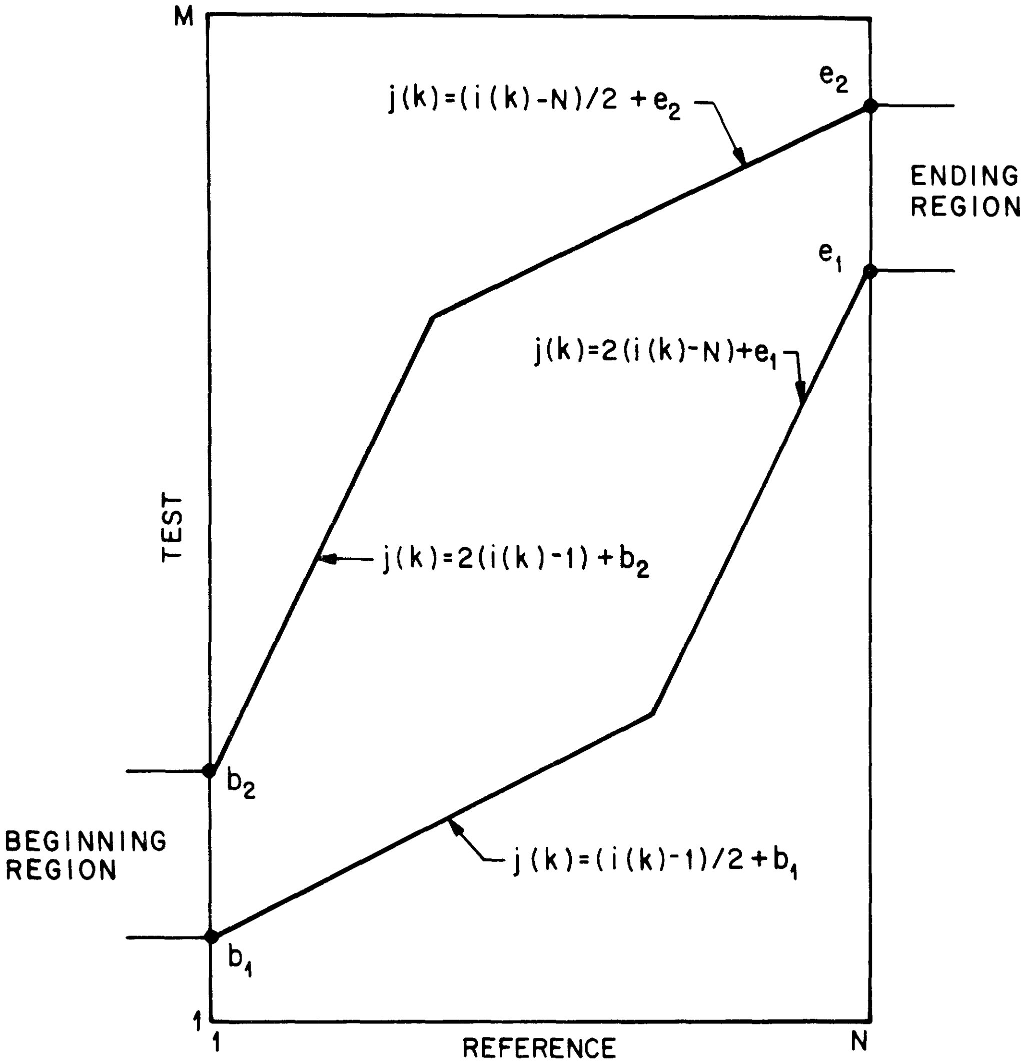


Fig. 6.3 Expanded range for a single time warp.

$$|j(k)-i(k)-\tilde{b}+1| \leqslant R \tag{6.5}$$

where $\tilde{b}$ is the center of the beginning region $(\tilde{b}=(b_1+b_2)/2)$ and R is the maximum time difference which is allowed. R must be chosen (at least) to cover the entire beginning region, i.e. $2R+1 \geqslant B$. This algorithm will be referred to as the *fixed range* DTW algorithm and is illustrated in Figure 6.4, part a.

Another range reduction technique, proposed by Rabiner, Rosenberg and Levinson [19], and described in detail by Rabiner and Schmidt [22], is illustrated in part b of Figure 6.4. Here $j(k)$ is restricted to be within a fixed range about the best path so far, that is, the local minimum. Formally, we have

$$|j(k)-c(k)| \leqslant \epsilon \tag{6.6a}$$

$$c(k) = \underset{m}{argmin}\, [D_A(i(k)-1,m)] \tag{6.6b}$$

$$c(1) = \tilde{b} \;, \tag{6.6c}$$

where $c(k)$ is the position, in the vertical direction, of the local minimum of $D_A(i(k)-1,m)$, and ϵ is the allowable range around this local minimum. Thus, if $D_A(n,m)$ is computed in consecutive vertical strips (i.e. n is fixed and m is varied), then the range of one vertical strip is $\pm\,\epsilon$ about the local minimum of the previous vertical strip. This algorithm is referred to as the *local minimum* DTW algorithm.

Two fundamental differences exist between these two algorithms. The fixed range DTW algorithm, a priori, specifies the ending region by specifying the beginning region, i.e.

$$E = 2R+1 \tag{6.7a}$$

$$e_1 = \tilde{b} + N - R \tag{6.7b}$$

$$e_2 = \tilde{b} + N + R \;, \tag{6.7c}$$

while the local minimum DTW algorithm defines the ending region implicitly from the local minimum of the last vertical strip, i.e.

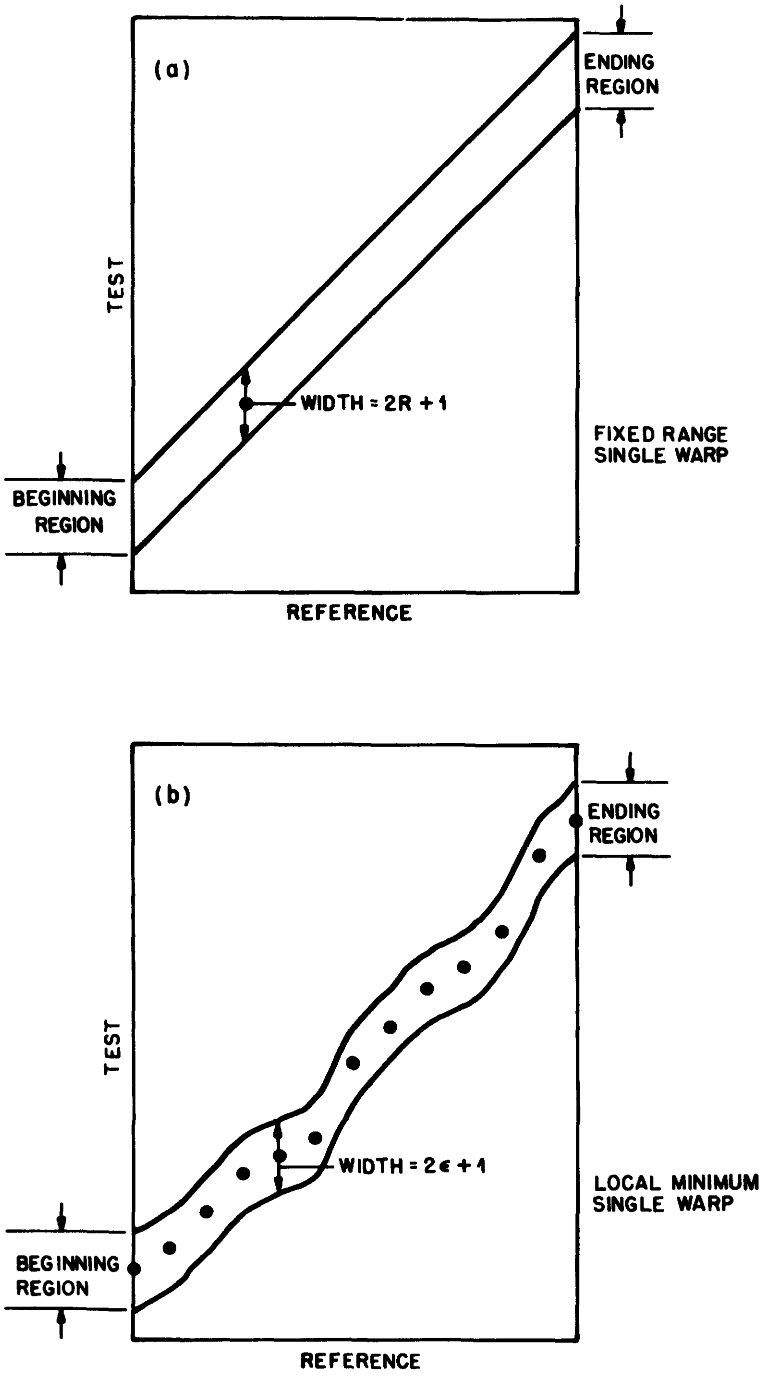


Fig. 6.4 Illustration of the fixed range and the local minimum DTW algorithms.

$$E = 2\epsilon + 1 \tag{6.8a}$$

$$e_1 = c(K) - \epsilon \tag{6.8b}$$

$$e_2 = c(K) + \epsilon \ . \tag{6.8c}$$

The other fundamental difference between the two time warping algorithms involves the number of time warps required to cover a beginning region. For the fixed range DTW algorithm, the entire beginning region is most efficiently covered in a single time warp with $2R + 1 = B$ since adjacent time warps may be merged together without loss of accuracy.

However, an analogous specification of the local minimum time warping algorithm ($2\epsilon+1=B$) may not be truly optimal. Since one application of the local minimum DTW algorithm may follow only one local minimum path, erroneous decisions may be made because the true path may be "lost," i.e. the globally best path need not be the best to any given vertical strip nor even within $\pm\epsilon$ of the local best. As such, it may be better to try several smaller local minimum time warps, thus allowing several different local minimum paths to be tried, and to compare the results of the different paths in order to determine the proper path. Such a procedure is illustrated in Figure 6.5. It is assumed that $NTRY$ local minimum time warps are to be computed. Each time warp has a local range of $\pm\epsilon$ about their respective local minima, and the centers of two adjacent time warps are initially separated by δ. The entire beginning region covered by the $NTRY$ time warps is given by

$$\Delta = 2\epsilon + 1 + (NTRY-1)\cdot\delta \ . \tag{6.9}$$

To cover the original beginning region, $NTRY$, ϵ and δ are chosen such that $\Delta = B$.

In the next section of this chapter we discuss the major issues raised by the use of the fixed range and the local minimum DTW algorithms for word spotting and connected speech recognition.

6.3 Issues in the Dynamic Time Warping Algorithms

The main areas in which we wish to apply the DTW algorithms of Section 6.2 are word spot-

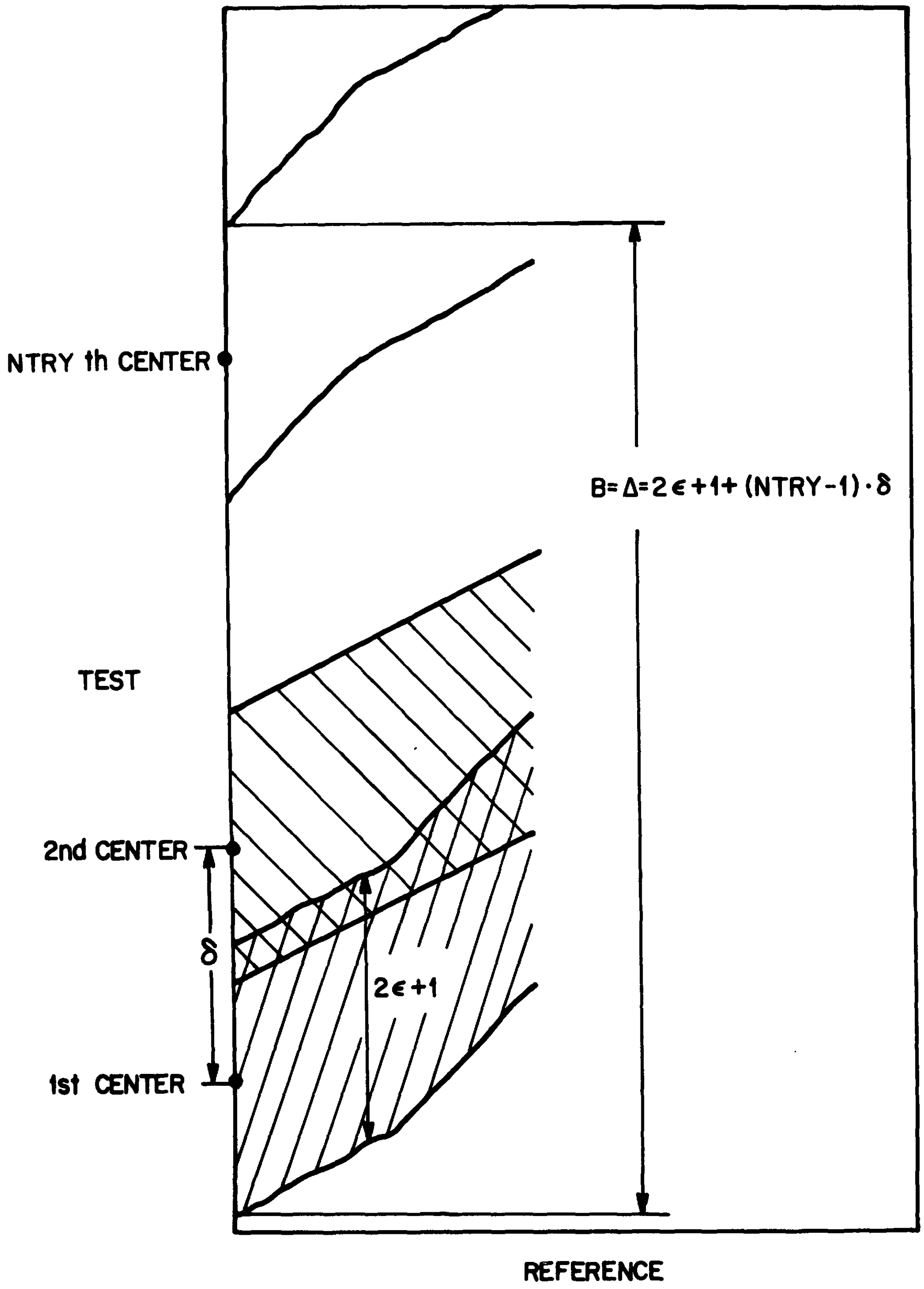


Fig. 6.5 Illustration of the parameters in the local minimum DTW algorithm.

ting and connected speech recognition. For such problem areas, it is important to understand the relative performance and the range of applicability of the DTW algorithms described in the previous section. Among the issues which must be investigated are:

1. Which of the two algorithms, (i.e. the fixed range DTW algorithm or the local minimum DTW algorithm) gives better performance results when applied to a series of recognition and word spotting experiments?

2. In the local minimum DTW algorithm, for a given Δ, what are the optimal choices of ϵ, *NTRY* and δ. In particular, the main question is whether more than one time warp is required, and, if so, how should the parameters ϵ, δ, and NTRY be chosen?

In order to answer these questions we have performed several recognition and word spotting experiments. The results of these experiments will be described in the next chapter.

Chapter 7

Experiments in Dynamic Time Warping for Connected Speech Recognition

7.1 Introduction

As discussed in Chapter 6, it is possible, in theory, to adapt dynamic time warping algorithms for applications such as word spotting and connected speech recognition. In this chapter we will present the results of several simple experiments designed to determine the relative performance of the fixed range and the local minimum DTW algorithms described in Chapter 6.

For purposes of the experiments, the basic recognition system was the one described in Chapter 3. Our experiments fall into two broad classes - namely, experiments designed to compare the relative performance of the fixed range and the local minimum DTW algorithms, and experiments designed to study the parameters of the local minimum DTW algorithm. In the next section of this chapter we describe the results of the experiments designed to measure the relative performance of the two DTW algorithms.

7.2 Comparison of the Two Time Warping Algorithms

In our initial experiment we compared the recognition accuracies achieved by both the fixed range and the local minimum DTW algorithms for a modified *isolated* word recognition problem. The test utterances were those of test set 2 (TS2) of Chapter 4, namely one replication by each of 4 talkers of a 54 word vocabulary, using speaker independent (2 templates per word) reference patterns. In order to evaluate the relative performance of the two DTW algorithms, the test utterances were modified so that a beginning region could be specified as some range about the true beginning point. No ending region was specified. For sake of comparison, R and ϵ were both set equal to 8 frames, and $NTRY$ was set to 1. Figure 7.1 shows the recognition error rates for both algorithms, as a function of the four local constraints that were studied. We observe that the local minimum algorithm performed consistently better than the fixed range DTW algorithms for *all* local constraints.

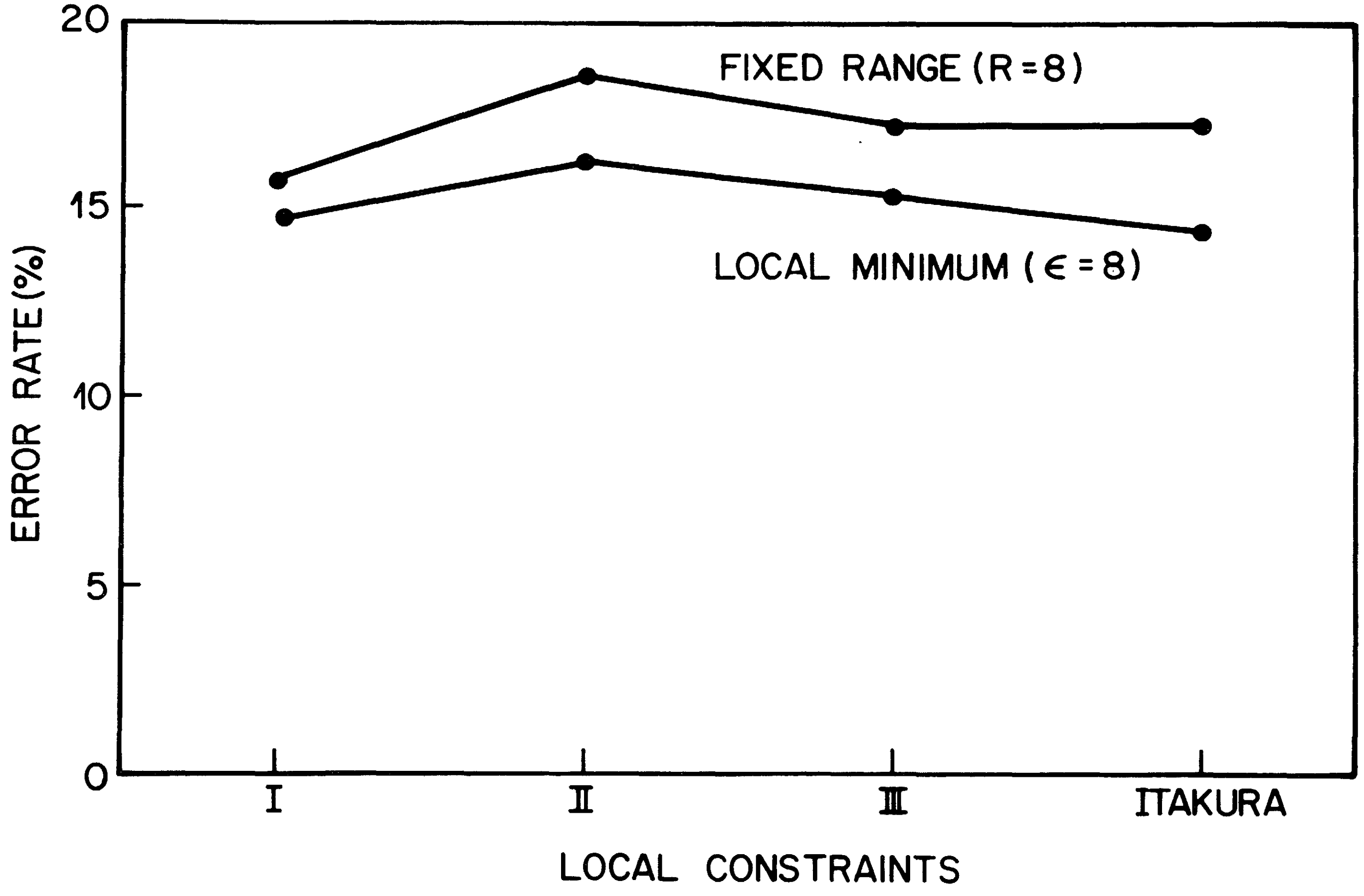


Fig. 7.1 Results for word recognition using both the fixed range and the local minimum DTW algorithms.

In another comparison we artificially imbedded (at an arbitrary frame) an isolated digit into a connected digit sequence, both uttered by the same speaker. We then used both DTW algorithms to "spot" the imbedded digit using two speaker dependent templates per digit. The parameters of the two DTW algorithms that were used were the same ones as in our initial experiment (ϵ=8,R=8). In order to spot the imbedded digit every possible beginning region of size $2\epsilon + 1$ ($=2R+1$) was tried. The number of times that the DTW algorithm found the (correct) best path (as determined by the lowest overall distance achieved by any beginning region) was recorded. We also recorded the ending point of the imbedded word, as estimated by the word spotting procedure. Both the local minimum and the fixed range DTW algorithms were able to locate the endpoint of the imbedded word with a high degree of accuracy. (The average error was 1.2 frames.)

In Figure 7.2 we show the relative performance for this simple word spotting experiment for the two DTW algorithms. The count refers to the number of times that the particular DTW algorithm found the proper path (as determined by the lowest distance score achieved) for each of the imbedded digits. We observe from Fig. 7.2 that the local minimum DTW algorithm almost always found the best path more often than the fixed range DTW algorithm. We also observe that the local minimum algorithm was able to find the best path 17 times (the maximum number possible, $2\epsilon+1$) for 8 of the 10 digits, while the fixed range algorithm never achieved this accuracy.

The results of these two experiments show that the local minimum DTW algorithm performed consistently better than the fixed range DTW algorithm. In the next section of this chapter we examine more fully some of the parameters of the local minimum time warping algorithm.

7.3 Examination of the Parameters of the Local Minimum Dynamic Time Warping Algorithm

In order to understand the effects of the various combinations of the parameters Δ, δ, $NTRY$ and ϵ, on the performance of the local minimum DTW algorithm, a series of connected digit recognition experiments were performed. A total of 80 strings of from 2 to 5 connected

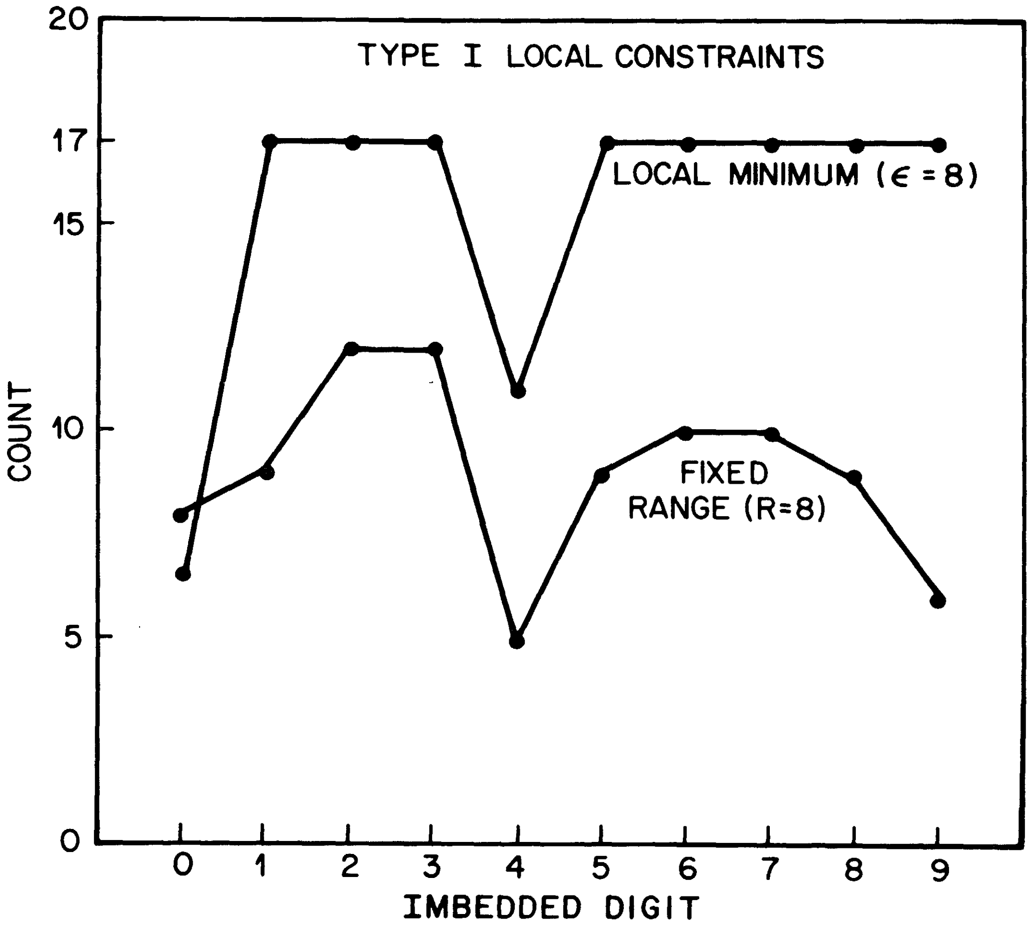


Fig. 7.2 Results for word spotting using both the fixed range and the local minimum DTW algorithms.

digits each (20 of each length) were recorded by each of two talkers. These utterances were taken from the data base of an earlier experiment [22]. In the recognition work we used two speaker dependent templates per digit. The first step in the experiment was to "spot" the ending point of the first digit in each string via a local minimum algorithm (ϵ=11,*NTRY*=1) using the known beginning point of the first digit. Then an attempt was made to recognize the second digit in the string. Because of inaccuracies in "spotting" the ending point of the first digit, and because of coarticulation effects, it was not possible to accurately determine the beginning point of the second digit, and, as such, the beginning region of the second digit was centered around the ending frame of the first digit, as determined by the "spotting" procedure. The best candidate for the second digit was chosen as that template which achieved the lowest overall distance, regardless of where it ended. Several values of ϵ δ, Δ and *NTRY* were used and the accuracies and distance scores for the recognition of the second digit were recorded.

In Figure 7.3 we plot, for a large value of Δ (27 in this case) the average best distance score for all *NTRY* warps as a function of δ, for several values of ϵ. There are 2 curves shown in each figure. The solid curve is the case when the reference word was the same as the second word in the test string. the dashed curve represents cases when the reference was different from the second work in the test string. Examination of Figure 7.3 shows that the average best distance for both "same words" and "different words" increases as δ increases. However, we observe that when the reference is different from the second digit of the test utterance (i.e. the dashed curves), the average distance is generally increasing as δ increases, but, that when the reference and the test word are the same, the average best distance is constant for small values of δ and increases only beyond the critical value $\delta = 2\epsilon + 1$. The critical value $\delta = 2\epsilon + 1$ (shown by a carret in the scales of Fig. 7.3) is a particularly important value of δ because for $\delta < 2\epsilon + 1$, consecutive time warps overlap in their beginning regions, and for $\delta > 2\epsilon + 1$ there are frames between two consecutive time warps which are not covered by either beginning region. From these results we conclude that, on average, there is no loss in performance for the local minimum DTW algorithm when consecutive starting points are separated by $\delta \leqslant 2\epsilon + 1$. When $\delta = 2\epsilon + 1$ we have the case where there is no overlap in adjacent

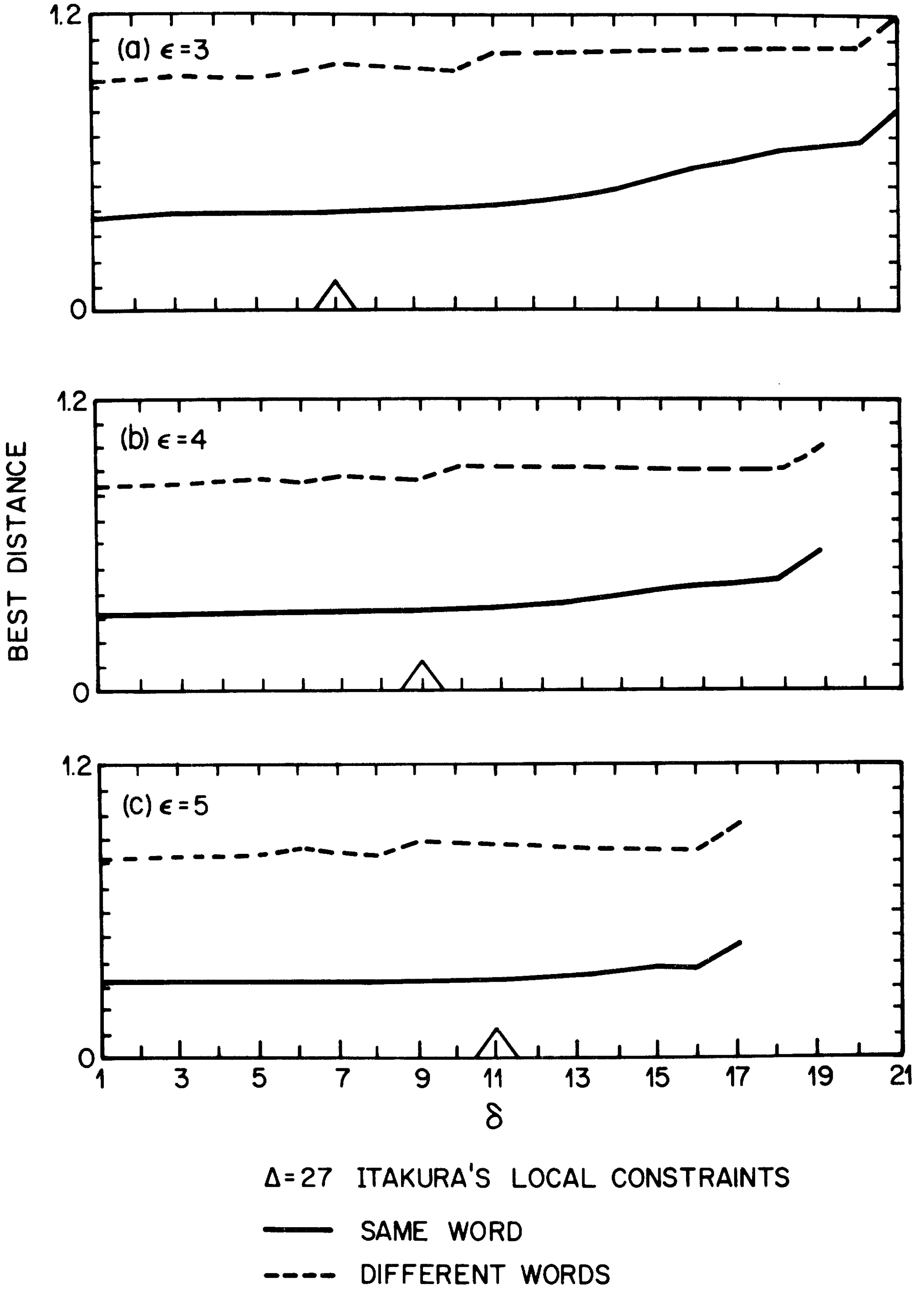


Fig. 7.3 Distance scores for the local minimum DTW algorithm as applied to connected word recognition.

beginning regions, and no skipped frames between these regions.

One explanation of why overlapping of beginning regions is unnecessary is given in Figure 7.4. Here we show the progress of a set of typical paths in which the starting regions overlap. By the nature of the local minimum DTW algorithm, best paths from overlapping time warps tend to merge if there is a good path common to both of their beginning regions. The effects of path merging (of the local minimum DTW algorithm) on the digit recognition accuracies is shown in Figure 7.5. Here we plot recognition error rate for the second digit in the test sequences as a function of δ, for various values of ϵ. We see that, for a fixed ϵ, it is possible to increase δ with essentially no loss in accuracy until $\delta > 2\epsilon + 1$.[1]

The results of the previous experiment have shown that, given a value for ϵ, the most appropriate choice of δ is $\delta = 2\epsilon + 1$. However, the question of the most appropriate choice of ϵ, Δ and *NTRY* remains. For the word spotting problem the obvious choice for Δ is $\Delta = M$, the length of the test. For this case optimal values for ϵ and *NTRY* must still be determined. In general, the selection of ϵ and *NTRY* depend on several factors. As ϵ is increased, the chance of a missed keyword decreases because more paths are examined, but the chance of a false alarm increases. Also, as ϵ increases, the value of *NTRY* decreases ($NTRY=\Delta/(2\epsilon+1)$ for $\delta=2\epsilon+1$), thereby reducing the amount of computation required. Thus, misses, false alarms and amount of computation must be traded-off in the selection of ϵ and *NTRY* for a word spotting application.

For connected word recognition applications, the choice of Δ, ϵ and *NTRY* is somewhat different from the word spotting problem in that the beginning region of a word is not the entire test utterance but can be, in general, reduced to a considerably smaller range. For such a situation we would like to have a simple rule for determining the optimal choice of δ, ϵ and *NTRY* for a given Δ. In Figure 7.6 we plot recognition error rate for the second digit of our test utterances for two cases, namely $\epsilon = (\Delta-1)/2$ ($NTRY=1$), and for the best combination of ϵ, δ and *NTRY*. We see that, for smaller values of Δ, a single warp performs as well as any

[1] Note that for Δ fixed, the largest possible δ is $\Delta - 2\epsilon - 1$ ($NTRY=2$) so that the curves for the various values of ϵ are defined only for values of δ such that $\delta \leqslant \Delta - 2\epsilon - 1$.

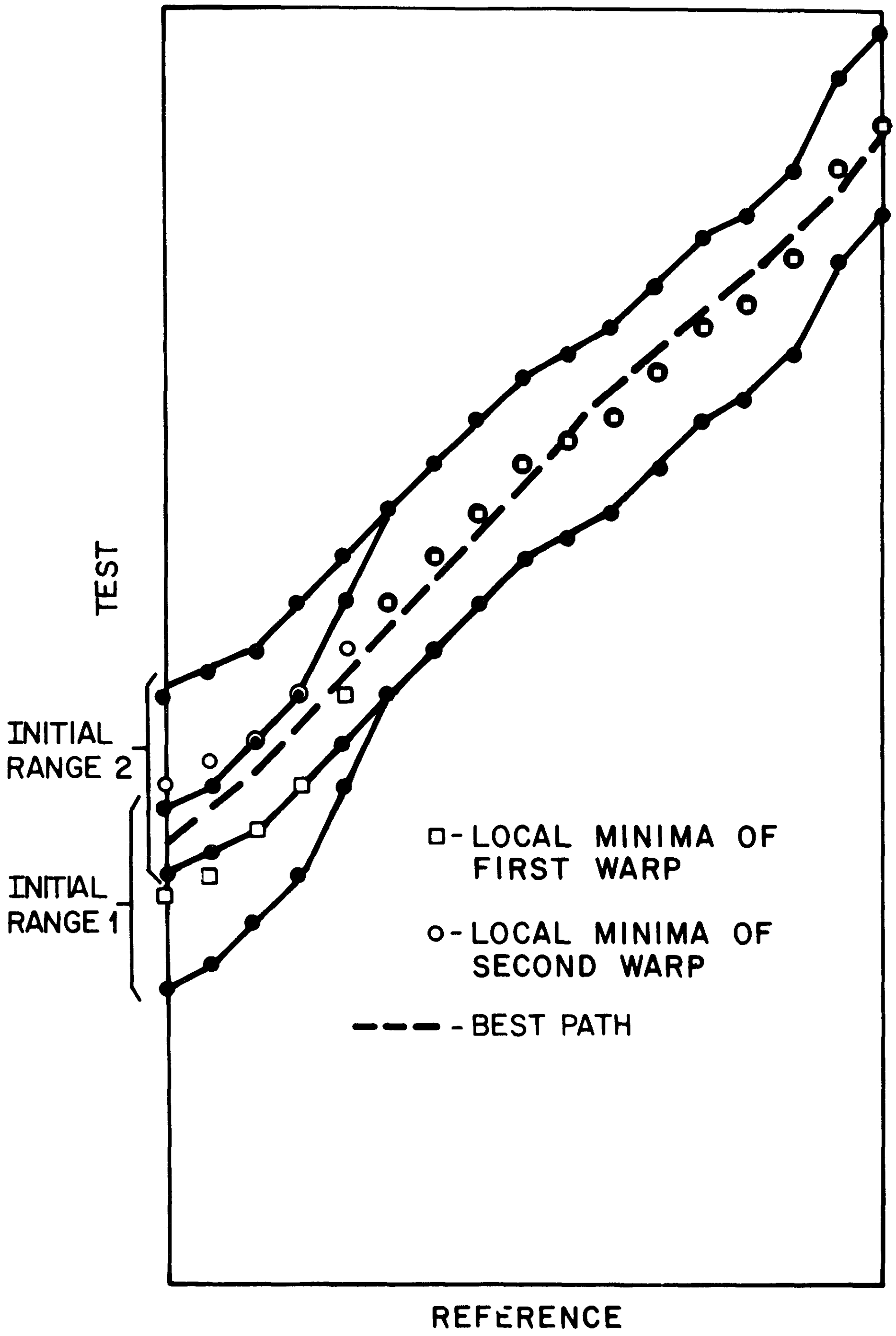


Fig. 7.4 Illustration of path merging for two adjacent local minimum time warps.

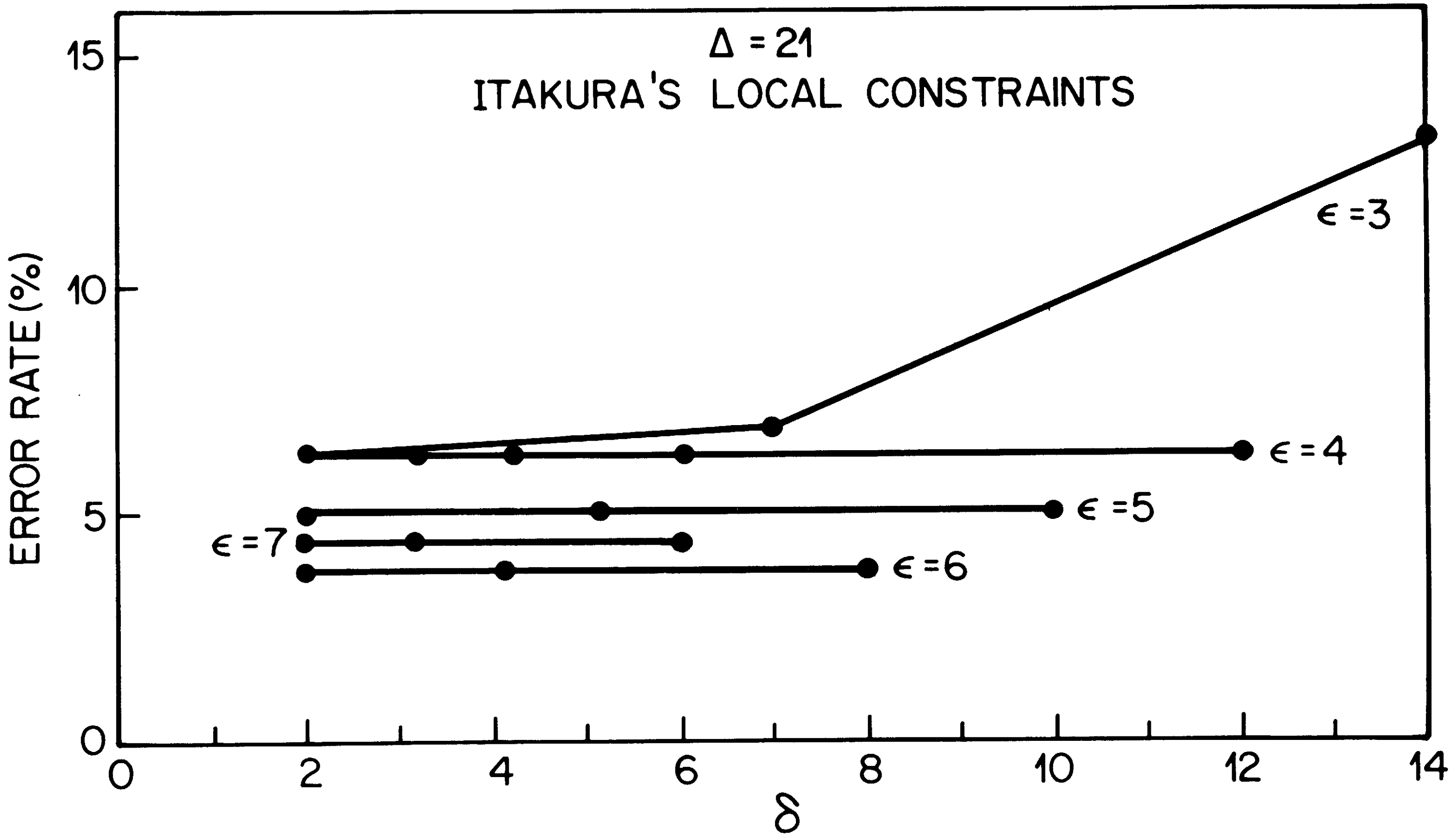


Fig. 7.5 Digit error rate for connected digit recognition using local minimum DTW algorithm and several values of ϵ and δ.

combination of ϵ, δ and *NTRY*, and that as Δ increases the differences in error rates between the best possible ϵ, δ and *NTRY* combination and a single warp remains less than 2.5%. Since a single warp is computationally more efficient than several time warps, and since it should be possible to defined an accurate beginning region to within 255 milliseconds (Δ=17 frames at 15 milliseconds between frames), a single local minimum time warp is a reasonable approach to the problem of connected word recognition using word size reference templates. In fact, though we show that the minimum error rate occurred at $\Delta = 21$ (in Fig. 7.6), work by Rabiner and Schmidt [22] on connected digit recognition has shown that it is possible to reduce Δ, without loss in accuracy, by more judicious positioning of the beginning region than simply centering it around the end of the previous word. Since Δ could be made smaller than 17 in this case, the results of Fig. 7.6 indicate that a single time warp would be adequate for many connected speech recognition applications.

In the final chapter of this thesis we summarize the results, of this chapter, give implications of these results and suggest further areas for research into the use of dynamic time warping algorithms for word spotting and connected speech recognition.

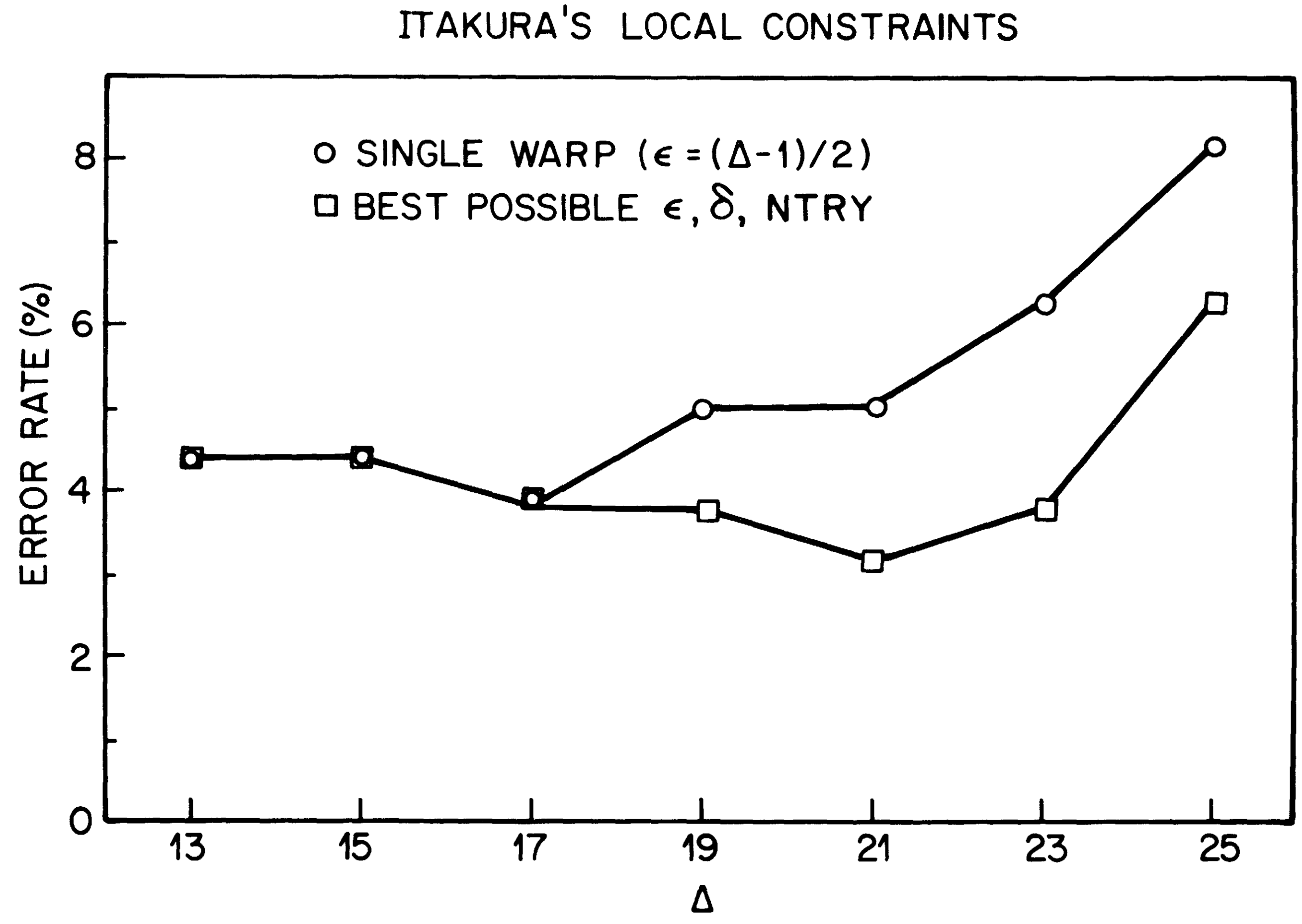


Fig. 7.6 Digit error rate for connected digit recognition using the local minimum DTW algorithm.

Chapter 8

Summary of Results on Dynamic Time Warping For Connected Speech Recognition

8.1 Introduction

In Chapters 6 and 7 we have presented two different dynamic time warping algorithms which are applicable for both connected speech recognition and word spotting applications. The algorithms are the fixed range and the local minimum time warping algorithms. We have compared their relative performance and have examined the effects of various combinations of the parameters of the local minimum algorithm on its performance. In this chapter we summarize and make some comments on the results given in Chapter 7.

8.1.1 Relative Performance of the Fixed Range and the Local Minimum Time Warping Algorithms

In the imbedded digit experiment, it was shown that both the fixed range and the local minimum DTW algorithm were able to accurately determine word boundaries when they were very clearly defined (as in our artificially imbedded digits). We also demonstrated that, for both word spotting and connected speech recognition, the local minimum DTW algorithm performed significantly better than the fixed range algorithm. The local minimum algorithm achieved both lower recognition error rates, and higher word spotting accuracy than the fixed range DTW algorithm.

8.1.2 Optimal Choice of the Parameters of the Local Minimum Time Warping Algorithm

In our examination of the parameters Δ, δ, ϵ and $NTRY$, of the local minimum DTW algorithm we obtained two important results. First of all, it was shown that it is not necessary to overlap successive beginning regions of the test pattern in order to achieve good time alignment, i.e. the best time alignment path is, in general, found from a single application of the local minimum DTW algorithm. We also found that, for small size beginning regions (small Δ), a single local minimum time warp, (with $\epsilon=(\Delta-1)/2, NTRY=1$), was as accurate as (and more computationally efficient than) any combination of the parameters ϵ, δ, and $NTRY$.

8.2 Practical Significance of the Results

Our results suggest some general approaches to both the word spotting and the connected speech recognition problems. Word spotting using the local minimum DTW algorithm can be accomplished by choosing a local range, ϵ, and then sampling the test utterance for the keyword using a spacing of $\delta = 2\epsilon + 1$. This method should provide accuracy comparable to most other word spotting algorithms using time warping.

Connected speech recognition implemented in a manner similar to word spotting, i.e. sample the test utterance for all words of the vocabulary using a spacing of $\delta = 2\epsilon + 1$ and then piece together the resulting time alignment paths to form the candidate string. Connected speech recognition may also be accomplished by building up test strings one word at a time. In such a method the ending point of one word is used to hypothesize the beginning region of the next and then, using a *single* local minimum time warp per word of the vocabulary, an attempt is made to match the next word of the test string. Either of these methods should, according to our results, provide recognition accuracy comparable to other connected speech recognition algorithms using time warping.

8.3 Future Research Areas

We have presented important results on the use of DTW algorithms for both word spotting and connected speech recognition. However, many questions still remain to be answered. One important question is whether the local minimum and the fixed range DTW algorithms are truly accurate enough for most word spotting and connected speech recognition applications, or whether a full range DTW algorithm is required for some applications. Another important question is whether the results presented here, using the ten digits as a vocabulary, may be extended to other vocabularies, particularly polysyllabic vocabularies. Within the context of the local minimum time warping algorithm, we must examine the effect on performance for various values of the local range, ϵ. Such a question is particularly important in the word spotting problem, in which a large value of ϵ should help to eliminate misses but will, most likely, increase false alarms. In the connected speech recognition problem, the question of the optimal size and positioning of a beginning region is important. Also, questions of how to piece together reference patterns in the presence of several good candidates, multiple beginning regions and syntax constraints are very important in connected speech recognition. Finally, on a more fundamental level, the question of whether or not the DTW algorithms which we have described can be applied to other units of recognition, such as syllables, phones, or demi-syllables, is an important unanswered question.

REFERENCES

1. A. E. Rosenberg and C. E. Schmidt, "Directory Assistance by Means of Automatic Recognition of Spoken Spelled Names," *Bell System Technical Journal*, Vol. 58, pp. 1797-1823, October 1979.

2. T. B. Martin, "Practical Applications of Voice Input to Machines," *Proceedings of the IEEE*, Vol. 64, pp. 487-501, April 1976.

3. G. M. White and R. B. Neely, "Speech Recognition Experiments with Linear Prediction, Bandpass Filtering and Dynamic Programming," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP-24, pp. 183-188, April 1976.

4. H. Sakoe and S. Chiba, "A Dynamic Programming Approach to Continuous Speech Recognition," *Proceedings of International Congress on Acoustics*, Budapest, Hungary, Paper 20C-13, 1971.

5. R. Bellman and S. Dreyfus, *Applied Dynamic Programming*, New Jersey: Princeton University Press, 1962.

6. F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP-23, pp. 57-72, February 1975.

7. J. S. Bridle, "An Efficient Elastic Template Method for Detecting Given Words in Running Speech," *Proceedings of British Acoustical Society Meeting*, London, England, Paper 73SHC3, April 1973.

8. L. R. Rabiner and M. R. Sambur, "An Algorithm for Determining the Endpoints of Isolated Utterances," *Bell System Technical Journal*, Vol. 54, pp. 297-315, February 1975.

9. M. R. Sambur and L. R. Rabiner, "A Speaker-Independent Digit-Recognition System," *Bell System Technical Journal*, Vol. 54, pp. 151-172, January 1975.

10. H. Sakoe and S. Chiba, "Dynamic Programming Optimization for Spoken Word Recognition," *IEEE Trans. Acoust. Speech, Signal Processing*, Vol. ASSP-26, pp. 43-49, February 1978.

11. A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms,* Massachusetts: Addison-Wesley Publishing Company, 1974.

12. L. R. Rabiner, "The Overall Structure of an Isolated Word Recognizer," unpublished paper.

13. L. Lamel, "Methods of Endpoint Detection for Isolated Word Recognition," Masters Thesis, MIT, February, 1980.

14. L. R. Rabiner, S. C. Levinson, A. E. Rosenberg and J. G. Wilpon, "Speaker-Independent Recognition of Isolated Words using Clustering Techniques," *IEEE Trans. Acoust., Speech, Signal Processing,* Vol. ASSP-27, pp 336-349, August 1979.

15. B. Gold, "Word Recognition Computer Program," RLE Technical Report 452, MIJ, June 1966.

16. L. R. Rabiner and J. G. Wilpon, "Speaker Independent, Isolated Word Recognition for a Moderate Size (54 Word) Vocabulary," to be published.

17. S. E. Levinson, L. R. Rabiner, A. E. Rosenberg and J. G. Wilpon, "Application of Clustering Techniques to Speaker Independent Word Recognition," to be published.

18. L. R. Rabiner, private correspondence.

19. L. R. Rabiner, A. E. Rosenberg and S. E. Levinson, "Considerations in Dynamic Time Warping for Discrete Word Recognition," *IEEE Trans. Acoust., Speech and Signal Processing,* Vol. ASSP-26, pp. 575-582, December 1978.

20. R. W. Christiansen and C. K. Rushforth, "Detecting and Locating Key Words in Continuous Speech Using Linear Predictive Coding," *IEEE Trans. Acoust., Speech, Signal Processing,* Vol. ASSP-25, pp. 361-367, October 1977.

21. 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, December 1979.

22. L. R. Rabiner and C. E. Schmidt, "Application of Dynamic Time Warping to Connected

Digit Recognition," to be published.

23. S. E. Levinson and A. E. Rosenberg, "A New System for Continuous Speech Recognition - Preliminary Results," Proceedings of the International Converence on Acoustics, Speech and Signal Processing, April 1979, pp. 239-244.

24. G. F. Simmons, *Differential Equations, with Applications and Historical Notes*, New York: McGraw-Hill Book Company, 1972.

25. P. A. Fox, ed., "The PORT Mathematical Subroutine Library," Murray Hill, New Jersey: Bell Laboratories, 1976.

Appendix 1

Time Warping for Continuous Functions

In this section we give one possible solution to the problem of time warping for continuous functions. The problem here is to find a continuous function $w(t)$ where

$$\tau = w(t), \tag{A1.1}$$

which maps a continuous reference pattern $\mathbf{R}(t)$ into a continuous test pattern $\mathbf{T}(\tau)$. Without loss of generality we may assume that $\mathbf{R}(t)$ has endpoints $t = 1$ and $t = N$ and that $\mathbf{T}(\tau)$ has endpoints $\tau = 1$ and $\tau = M$. An example of a typical time warping function, $w(t)$ is shown in Figure A1.1. $\mathbf{R}(t)$ and $\mathbf{T}(\tau)$ are shown as one dimensional functions of time although they are often multidimensional. The function $w(t)$ is shown as a curve in (t,τ) space from the point $(1,1)$ to the point (N,M). If we assume that the endpoints are properly identified we must find a continuous function, $w(t)$, which preserves the endpoint locations, i.e.

$$w(1) = 1 \tag{A1.2a}$$

$$w(N) = M. \tag{A1.2b}$$

The choice of $w(t)$ is made so as to provide the best possible match (i.e. minimum distance) between the reference and the time warped test pattern. The measure of similarity which we shall use is a weighted integral of local distances, i.e.

$$D(w(t)) = \int_1^N d(t,w(t))\, \tilde{W}(t,w(t),\dot{w}(t))\, dt \tag{A1.3}$$

where $D(w(t))$ is the value of the global distance function for a particular warping function $w(t)$, $d(t,w(t))$ is the local distance between location t of the reference pattern and location $w(t)$ of the test pattern, $\dot{w}(t)$ is the derivative of $w(t)$ with respect to t, and $\tilde{W}(t,w(t),\dot{w}(t))$ is a weighting function which depends upon the shape of $w(t)$ along the time warping curve.

The best choice for $w(t)$ is defined to be that function $w(t)$ which minimizes the value of $D(w(t))$. Thus, the best possible match between $\mathbf{R}(t)$ and $\mathbf{T}(\tau)$ is given by

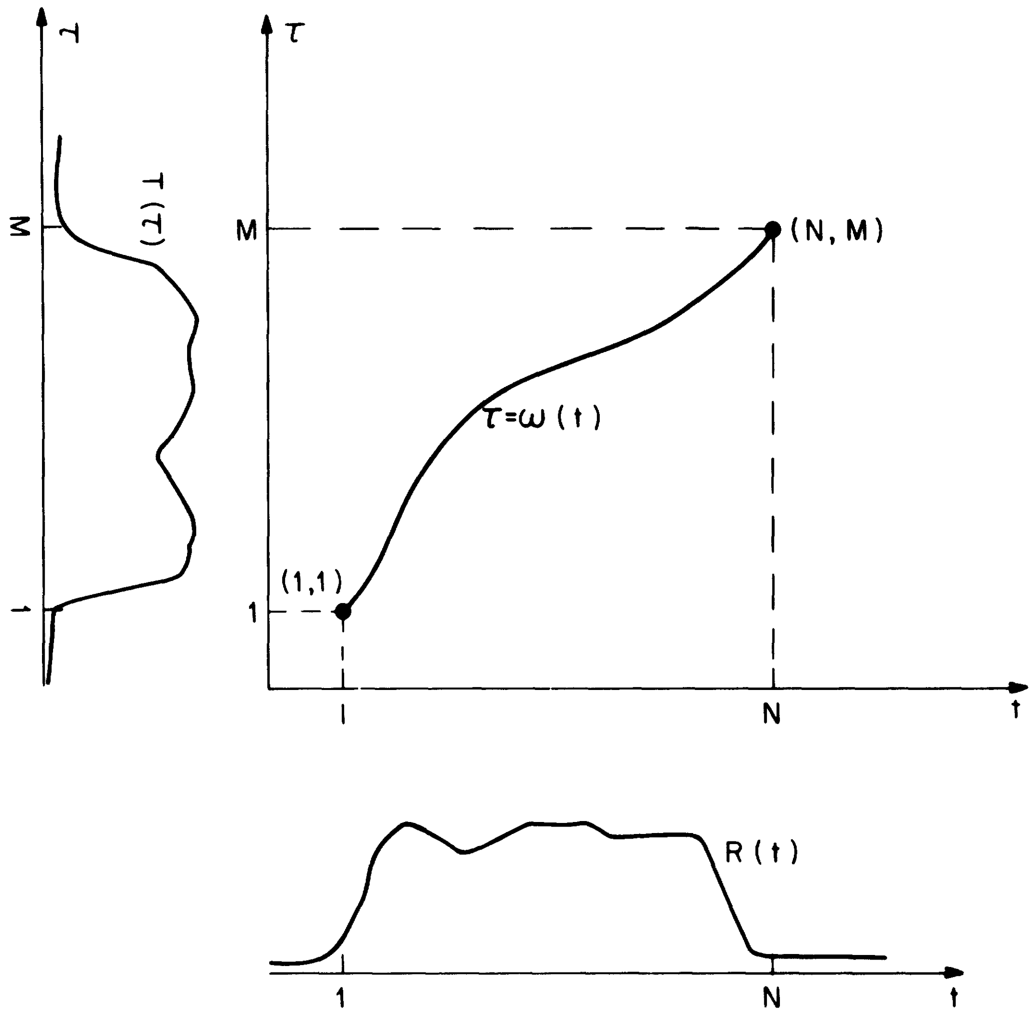


Fig. A1.1 Typical continuous-time warping function.

$$\hat{D} = \min_{(w(t))} D(w(t)) = \min_{(w(t))} \int_1^N d(t, w(t))\, \tilde{W}(t, w(t), \dot{w}(t))\, dt \qquad \text{(A1.4)}$$

where $\hat{D}$ is the value of $D(w(t))$ when the best match is achieved.

Given the problem of Eq. (A1.4) we observe that this problem is identical to the classical calculus of variations problem, i.e.

$$\min_{(w(t))} \int_a^b F(t, w(t), \dot{w}(t))\, dt \qquad \text{(A1.5a)}$$

with

$$w(a) = w_a \qquad \text{(A1.5b)}$$

$$w(b) = w_b \qquad \text{(A1.5c)}$$

where the particular problem of Eq. (A1.4) uses

$$a = 1, \quad b = N \qquad \text{(A1.6a)}$$

$$w_a = 1, \quad w_b = M \qquad \text{(A1.6b)}$$

and

$$F(t, w(t), \dot{w}(t)) = d(t, w(t))\, \tilde{W}(t, w(t), \dot{w}(t)). \qquad \text{(A1.6c)}$$

It is known that the solution to the calculus of variations problem of Eqs. (A1.5) is given by the solution to Euler's differential equation [24], i.e.

$$F_w - \frac{d}{dt} F_{\dot{w}} = 0 \qquad \text{(A1.7)}$$

where

$$F_w = \frac{\partial}{\partial w} F(t, w, \dot{w}) \qquad \text{(A1.8a)}$$

$$F_{\dot{w}} = \frac{\partial}{\partial \dot{w}} F(t, w, \dot{w}). \qquad \text{(A1.8b)}$$

Eq. (A1.7) may be further expanded to yield the following second order differential equation

$$F_w - F_{t\dot{w}} - F_{w\dot{w}}\, \dot{w} - F_{\dot{w}\dot{w}}\, \ddot{w} = 0 \qquad \text{(A1.9)}$$

where

$$F_{tw} = \frac{\partial^2}{\partial t \partial w} F(t,w,\dot{w}) \qquad \text{(A1.10a)}$$

$$F_{w\dot{w}} = \frac{\partial^2}{\partial w \partial \dot{w}} F(t,w,\dot{w}) \qquad \text{(A1.10b)}$$

$$F_{\dot{w}\dot{w}} = \frac{\partial^2}{\partial \dot{w} \partial \dot{w}} F(t,w,\dot{w}) \qquad \text{(A1.10c)}$$

$$\ddot{w} = \frac{d^2}{dt^2} w(t). \qquad \text{(A1.10d)}$$

For the particular problem of Eq. (A1.4) we may derive the differential equation,

$$(d\tilde{W})_{w} - (d\tilde{W})_{t\dot{w}} - (d\tilde{W})_{w\dot{w}}\dot{w} - (d\tilde{W})_{\dot{w}\dot{w}}\ddot{w} = 0 \qquad \text{(A1.11)}$$

where $d\tilde{W}$ is used to denote the term $d(t,w(t))\,\tilde{W}(t,w(t),\dot{w}(t))$.

At this point we must define $\tilde{W}(t,w(t),\dot{w}(t))$ in order to proceed. Logically, $\tilde{W}(t,w(t),\dot{w}(t))$ should be independent of t and $w(t)$ since all points in the (t,τ) plane should be weighted equally. Also $\tilde{W}(t,w(t),\dot{w}(t))$ should be twice differentiable in $\dot{w}(t)$ in order to preserve the second order nature of Eq. (A1.11). (If Eq. (A1.11) were first order it would, in general, be impossible to both solve it and satisfy Eqs. (A1.2) simultaneously.) One logical choice for $\tilde{W}(t,w(t),\dot{w}(t))$ is

$$\tilde{W}(t,w(t),\dot{w}(t)) = \sqrt{1+\dot{w}(t)^2}, \qquad \text{(A1.12)}$$

i.e. the arc length. Using this definition of $\tilde{W}(t,w(t),\dot{w}(t))$, Eq. (A1.3) becomes

$$D(w(t)) = \int_1^N d(t,w(t))\sqrt{1+\dot{w}(t)^2}\,dt \qquad \text{(A1.13)}$$

or, simply, the line integral of $d(t,w(t))$ over the curve $w(t)$ from the point (1,1) to the point (N,M).

More sophisticated choices for $\tilde{W}(t,w(t),\dot{w}(t))$ may be made. For example, $\tilde{W}(t,w(t),\dot{w}(t))$ may be chosen to increase very rapidly for $\dot{w}(t)$ outside some range, thus effectively limiting the amount of expansion or compression in the time warping. However, for simplicity we will proceed with the definition of $\tilde{W}(t,w(t),\dot{w}(t))$ as given in Eq. (A1.12).

Substitution of the definition of $\tilde{W}(t,w(t),\dot{w}(t))$ (Eq. (A1.12)) into Eq. (A1.11) yields the following differential equation

$$d_{w'}(1+\dot{w}^2)^{1/2} - \frac{d_t\dot{w}}{(1+\dot{w}^2)^{1/2}} - \frac{d_{w'}\dot{w}^2}{(1+\dot{w}^2)^{1/2}}$$

$$- \frac{d\ddot{w}}{(1+\dot{w}^2)^{3/2}} = 0 \tag{A1.14}$$

where

$$d_{w'} = \frac{\partial}{\partial w} d(t,w(t)) \tag{A1.15a}$$

$$d_t = \frac{\partial}{\partial n} d(t,w(t)) \tag{A1.15b}$$

which may be simplified to

$$d\ddot{w} = d_{w'}(1+\dot{w}^2) - d_t\dot{w}(1+\dot{w}^2). \tag{A1.16}$$

In order to evaluate the applicability of Eq. (A1.16) to actually solve for an optimal path, it was implemented as a system of the two equations

$$\dot{u} = (d_{w'}(1+u^2) - d_t u(1+u^2))/d \tag{A1.17a}$$

$$\dot{w} = u. \tag{A1.17b}$$

The solution of this system was determined by a modified midpoint rule using extrapolation [25]. Values for $d_{w'}$ and d_t were determined by a first order two dimensional Lagrangian interpolation on a sampled version of $d(t,w(t))$ []. For the trial case $\mathbf{R}(t)$ and $\mathbf{T}(\tau)$ were taken to be p = **8th** order LPC coefficients and, as such, Itakura's log likelihood ratio distance was used for $d(t,w(t))$ [6]. The samples of $d(t,w(t))$ were specified at $t = 1,2,..,N$; $\tau = 1,2,...,M$. In the trial case, however, the differential equation solver failed to converge. Detailed examination showed that $d(t,w(t))$ was not smooth over the (t,τ) plane. This occurs because, under the assumptions of Itakura's log likelihood ratio, $d(t,w(t))$ is a chi-squared random variable and thus, a smooth minimum distance contour is difficult to find.

Although the algorithm, as proposed, failed to converge, further study may be useful. If an appropriate method may be found for the solution of Eq. (A1.16) then the solution to this equation may be used as a standard against which the performance of other time warping algorithms is measured. Possible areas for improvement include use of a smoothed version of Itakura's distance metric, use of some other feature set and distance metric, or use of a more exact numerical method for interpolation and integration.

Appendix 2 *Performance Results*

Timing Results - Average Per Warp

Local Constraints	Weighting Function	Axis Orientation	Range Limit	Combinatorics (Milliseconds)	Local Distance Calculations
I	a	R	∞	90.2	543.2
I	b	R	∞	80.6	543.2
I	c	R	∞	85.1	543.2
I	d	R	∞	90.8	543.2
II	a	R	∞	57.8	491.7
II	b	R	∞	65.9	491.7
II	c	R	∞	63.2	491.7
II	d	R	∞	69.6	491.7
III	c	R	∞	82.8	504.4
IV	c	R	∞	249.3	781.0
Itakura	c	R	∞	63.2	504.4
I	c	T	∞	86.5	544.9
II	c	T	∞	63.6	491.7
III	c	T	∞	83.0	505.3
Itakura	c	T	∞	63.7	505.3
I	c	R	11	81.5	520.8
II	c	R	11	60.6	472.4
III	c	R	5	58.7	328.0
III	c	R	8	74.1	429.9
III	c	R	11	82.0	482.5
III	c	R	14	84.9	501.1

Table A2.1

Local Constraints Type III
Range Size ($N = 40$)

M	R	Range Size	$\frac{\text{Range Size} \cdot 100\%}{N \cdot M}$
21	∞	59	7.0%
24	∞	168	17.5%
27	∞	265	24.5%
	14	250	23.1%
30	∞	350	29.2%
	14	343	28.6%
	11	310	25.8%
33	∞	423	32.0%
	14	421	31.9%
	11	397	30.1%
	8	346	26.2%
36	∞	484	33.6%
	14	484	33.6%
	11	469	32.6%
	8	418	29.0%
	5	313	21.7%
39	∞	533	34.2%
	14	533	34.2%
	11	523	33.5%
	8	463	29.7%
	5	349	22.4%
	2	181	11.6%
40	∞	547	34.2%
	14	547	34.2%
	11	535	33.4%
	8	472	29.5%
	5	355	22.2%
	2	184	11.5%
42	∞	570	33.9%
	14	570	33.9%
	11	550	32.7%
	8	481	28.6%
	5	358	21.3%
	2	181	10.8%
45	∞	595	33.1%
	14	586	32.6%
	11	550	30.6%
	8	472	26.2%
	5	340	18.9%

Table A2.2

M	R	Range Size	Range Size · 100%
			N · M
48	∞	608	31.7%
	14	578	30.1%
	11	524	27.3%
	8	436	22.7%
51	∞	609	29.9%
	14	546	26.8%
	11	474	23.2%
54	∞	598	27.7%
	14	490	22.7%
57	∞	575	25.2%
60	∞	540	22.5%

Table A2.2 (continued)

Time Warps Performed

Maximum Slope	Range Limit	Speaker	Possible Warps		Actual Warps		Percentage	
			Same Word	Different Words	Same Word	Different Words	Same Word	Different Words
2	5	SD1	390	14820	314	6906	80.5	46.6
		SD2	390	14820	378	8627	96.9	58.2
		SI	432	22896	195	7683	45.1	33.6
		Total	1212	52536	887	23216	73.2	44.2
	8	SD1	390	14820	365	9699	93.6	65.4
		SD2	390	14820	389	11727	99.7	79.1
		SI	432	22896	280	11326	64.8	49.5
		Total	1212	52536	1034	32752	85.3	62.3
	11	SD1	390	14820	379	11732	97.2	79.2
		SD2	390	14820	390	13554	100.0	91.5
		SI	432	22896	350	14470	81.0	63.2
		Total	1212	52536	1119	39756	92.3	75.7
	14	SD1	390	14820	387	13134	99.2	88.6
		SD2	390	14820	390	14427	100.0	97.3
		SI	432	22896	391	16980	90.5	74.2
		Total	1212	52536	1165	44541	96.1	84.8
	∞	SD1	390	14820	390	14751	100.0	99.5
		SD2	390	14820	390	14804	100.0	99.5
		SI	432	22896	431	22300	99.8	97.4
		Total	1212	52536	1211	51855	99.9	98.7
3	∞	SD1	390	14820	390	14820	100.0	100.0
		SD2	390	14820	390	14820	100.0	100.0
		SI	432	22896	432	22850	100.0	99.8
		Total	1212	52536	1212	52490	100.0	99.9

Table A2.3

Recognition Accuracy

				Errors (possible)			Probability of Error		
Local Constraints	Weighting Function	Axis Orientation	Range Limit	SD1 (195)	SD2 (195)	SI (216)	SD1	SD2	SI
I	a	R	∞	12	4	18	.030	.023	.089
I	b	R	∞	11	3	16	.027	.023	.095
I	c	R	∞	11	4	20	.033	.020	.086
I	d	R	∞	12	4	18	.031	.022	.076
II	a	R	∞	10	4	17	.032	.024	.090
II	b	R	∞	10	4	18	.029	.022	.091
II	c	R	∞	10	4	22	.034	.021	.090
II	d	R	∞	10	5	18	.034	.021	.081
III	c	R	∞	11	5	21	.033	.021	.089
IV	c	R	∞	14	2	24	.034	.023	.100
Itakura	c	R	∞	12	5	20	.033	.022	.087
I	c	T	∞	12	4	13	.031	.025	.080
II	c	T	∞	11	3	16	.032	.026	.082
III	c	T	∞	15	4	14	.032	.026	.079
Itakura	c	T	∞	15	4	11	.031	.025	.079
I	c	R	11	11	4	39	.035	.017	.076
II	c	R	11	10	4	41	.036	.019	.077
III	c	R	5	11	24	88	.039	.018	.068
III	c	R	8	11	9	57	.037	.018	.072
III	c	R	11	11	5	39	.035	.018	.076
III	c	R	14	11	5	28	.034	.019	.073

Table A2.4

Average Distance Between Same Words
Weighting Function c, Reference along x-axis

	Local Constraints				
N/M	I	II	III	Itakura	Average
.5 - .6	.482	.485	.483	.483	.483
.6 - .7	.426	.418	.417	.418	.420
.7 - .8	.353	.337	.341	.344	.344
.8 - .9	.365	.348	.350	.354	.354
.9 - 1.0	.355	.338	.342	.347	.346
1.0 - 1.1	.344	.330	.332	.339	.339
1.1 - 1.2	.361	.344	.349	.358	.353
1.2 - 1.3	.366	.352	.356	.364	.360
1.3 - 1.4	.371	.368	.364	.373	.369
1.4 - 1.5	.413	.413	.405	.410	.410
1.5 - 1.6	.476	.476	.470	.478	.475
1.6 - 1.7	.696	.697	.694	.709	.699
1.7 - 1.8	.745	.754	.741	.745	.746

Table A2.5

Recognition Accuracy of Normalize/Warp Algorithm

				Errors (Possible)			Change From Previous Errors		
Local Constraints	Weighting Function	Axis Orientation	Range Limit	SD1 (195)	SD2 (195)	SI (216)	SD1	SD2	SI
I	a	R	∞	--	--	17	--	--	-1
I	b	R	∞	--	--	13	--	--	-3
I	c	R	∞	13	4	20	+2	0	0
I	d	R	∞	--	--	16	--	--	-2
II	a	R	∞	--	--	15	--	--	-2
II	b	R	∞	--	--	15	--	--	-3
II	c	R	∞	11	3	20	+1	-1	-2
II	d	R	∞	--	--	17	--	--	-1
III	c	R	∞	12	4	20	+1	-1	-1
IV	c	R	∞	--	--	25	--	--	+1
Itakura	c	R	∞	12	3	20	0	-2	0
I	c	T	∞	12	5	12	0	+1	-1
II	c	T	∞	11	5	13	0	+2	-3
III	c	T	∞	11	5	12	-4	+1	-2
Itakura	c	T	∞	12	5	12	-3	+1	+1
I	c	R	11	--	--	20	--	--	-19
II	c	R	11	--	--	20	--	--	-21
III	c	R	1	16	5	33	--	--	--
III	c	R	3	10	3	20	--	--	--
III	c	R	5	12	4	17	+1	-20	-71
III	c	R	8	12	4	19	+1	-5	-38
III	c	R	11	12	4	20	+1	-1	-19
III	c	R	14	11	4	20	0	-1	-8

Table A2.6