TR－H－032

A Computational Theory for Movement Pattern Recognition Based on Optimal Movement Pattern Generation

Yasuhiro WADA　Yasuharu KOIKE
Eric VATIKIOTIS-BATESON
Mitsuo KAWATO

1993. 9. 29

ATR人間情報通信研究所

〒619-02 京都府相楽郡精華町光台2-2　☎07749-5-1011

ATR Human Information Processing Research Laboratories

2-2, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02 Japan

Telephone: +81-7749-5-1011
Facsimile: +81-7749-5-1008

A Computational Theory for Movement Pattern Recognition Based on Optimal Movement Pattern Generation

Yasuhiro WADA
Yasuharu KOIKE
Eric VATIKIOTIS-BATESON
Mitsuo KAWATO

ATR Human Information Processing Research Laboratories,
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02, Japan

Abstract

We have previously proposed an optimal trajectory planning and control theory for continuous movements, such as reaching or cursive handwriting (Wada and Kawato 1994). According to Marr's three-level description of brain function (Marr 1982), our theory can be summarized as follows; (1) the computational theory is the minimum torque-change model; (2) the intermediate representation of a pattern is given as a set of via-points extracted from an example pattern; and (3) algorithm and hardware are provided by FIRM (Wada and Kawato 1993): a neural network that can generate and control minimum torque-change trajectories. In this paper, we propose a computational theory for movement pattern recognition that is based on our theory for optimal movement pattern generation. The three-levels of description of brain function in the recognition theory are tightly coupled with those for pattern generation. In recognition, the generation process and the recognition process are actually two information flows in opposite directions within a single functional unit. In our theory, if the input movement trajectory data is identical to the optimal movement pattern reconstructed from an intermediate representation of some symbol, the input data is recognized as that symbol. If there exists an error between the movement trajectory data and the generated trajectory, the putative symbol is corrected and the generation is repeated. In particular, we present concrete computational procedures for the recognition of connected cursive handwritten characters, as well as for the estimation of phonemic timing in natural speech. Our most important contribution is to demonstrate the computational realizability for the "motor theory of movement-pattern perception": the movement-pattern recognition process can be realized by actively recruiting the movement-pattern formation process. The way in

which the formation process is utilized in pattern recognition in our theory suggests a duality between movement pattern formation and movement pattern perception.

Acknowledgment:
We would like to thank Drs. E.L.Saltzman, C.G.Atkeson, and S. Schaal for discussing and reading earlier versions of the manuscript. We wish to express our thanks to Dr. Y.Tohkura of ATR Human Information Processing Research Laboratories for his encouragement. This work was supported by a Human Frontier Science Project Grant to M.K.

1. Introduction

Although there are many complex character shapes and speech sounds, humans can nevertheless perceive handwritten characters and spoken language with a facility that far outstrips the computers of today. How do humans recognize handwritten characters and speech? Human pattern perception is really amazing.

One thing that seems clear is that reading and writing and hearing and speaking are related pairs. The real question is then how tightly the movement pattern recognition and pattern formation processes are coupled. We take a rather radical position that formation and recognition are two aspects of a single function. First we review several studies in the psychological literature which suggest that the formation process is actively utilized in the recognition process.

Liberman et al. (1967, 1985) proposed the motor theory of speech perception. Their theory first states that there is a tight coupling between formation and recognition. In particular, they make two claims. The first is that "the objects of speech perception are the intended phonetic gestures of the speaker, represented in the brain as invariant motor commands that call for movements of the articulators through certain linguistically significant configurations." The second is a corollary of the first: " If speech perception and speech production share the same set of invariants, they must be intimately linked. This link is not a learned association, a result of the fact that what people hear when they listen to speech is what they do when they speak."

In a similar manner, Babcock and Freyd (1988) suggested that information relevant to the cursive script formation process is captured in perception of the static trace of handwritten characters. In their experiments a reader can infer dynamic information from

a handwritten character. Freyd (1983) proposed two claims in her theory of handwriting recognition: The first is knowledge. That is, the perceiver has knowledge of the dynamics of the handwriting process represented in the mind in a form that can be used by the recognition processes. The second is sensitivity. The perceiver is sensitive to variations in the static handwritten trace that indicate the order and direction in which the components of the letter were produced.

Both theories emphasize that the recognition process depends on the formation process. From a computational point of view, Kawato (1989) and Haken et al. (1990) have noted that there is a dual relation between pattern formation and pattern recognition. They contend that the pattern recognition process and the pattern formation process are just two aspects of a single function. That is, pattern formation is needed so as to recognize the pattern, and conversely, pattern recognition is needed so as to generate the movement pattern. Kawato (1989), motivated by the revised motor theory of speech perception (Liberman and Mattingly 1985), proposed a neural network model for speech synthesis and speech recognition. It was shown that a cascade network (Kawato et al. 1990), originally proposed as an arm trajectory formation model, can be extended to continuous speech recognition as well to continuous speech trajectory generation. According to Haken et al. (1990), meaningful information for recognizing dynamic visual patterns resides in attractors of the order parameter dynamics. In a neural network, these order parameters represent different macrostates of the net as a whole. That is, the network dynamics for the recognition process is specific to the order parameters that characterize the formation of those patterns.

Generally, movement patterns such as connected cursive handwriting and continuous natural speech are restricted by being generated under body dynamics and physical laws,

and are planned based on optimization principles in the central nervous system. Over the past ten years, computational theories of the brain have been proposed for human arm trajectory formation, that are based on minimization principles. It should be noted that trajectory formation is an ill-posed problem because the hand can move along an infinite number of possible trajectories from the starting point to the target point. However, humans can move an arm between two targets, selecting one trajectory from among an infinite number of trajectories. Therefore, the brain should be able to compute a unique solution by imposing an appropriate cost criterion on the ill-posed problem. The minimum jerk criterion (Flash and Hogan 1986) and the minimum torque-change criterion (Uno, Kawato and Suzuki 1989) have been proposed as the basis of computational theories for reaching movement. Wada and Kawato (1994) proposed a computational theory for handwriting based also on the same minimum-torque-change principle that they proposed for reaching movements. Our basic hypothesis for continuous movement pattern generation, such as connected cursive handwriting, is that the computational theory should be essentially the same as that for reaching movements. Particularly, at the computational level understanding of Marr (1982), our handwriting theory utilizes the minimum-torque-change model. At the representation level, it uses a set of via-points as an intermediate representation between a character symbol and the motor-command stream. This is a natural extension of our reaching movement theory where intermediate representations are start and end points, and in via-point movement cases a single via-point. At the hardware level, our theory utilizes the FIRM (Forward Inverse Relaxation Model) neural network (Wada and Kawato 1993), just as in reaching, to generate and control minimum-torque-change trajectories. In the present paper we argue that when one plans a cursive handwritten character, one solves an optimization problem whose

boundary conditions are a set of via-points, rather than only a few boundary conditions as in reaching movements.

While taking account of psychological studies which suggest active participation of motor control in movement-pattern perception, we discuss the implications that can be derived for movement-pattern perception from Wada and Kawato's optimal theory for cursive handwriting. In the theory, the intermediate representation (a set of via-points) between a character symbol and the motor-command stream is stored in memory and called for every time the character is written. How, then, are these intermediate representations acquired and stored ? We propose that they are acquired through past experiences from a number of examples of producing cursive characters. This is the reason why the via-point estimation algorithm is essential for cursive handwriting. Extraction of via-points is the movement-pattern perception process. Furthermore, our estimation algorithm utilizes movement pattern generation in an essential way. Thus, it is tempting to hypothesize that the via-points extracted from a character are the intermediate representations not only for movement-pattern generation, but also for the movement pattern perception. In order to complete the movement-pattern recognition process, the remaining step is to transform the set of via-points into a corresponding character symbol.

If the pattern formation mechanism can generate a trajectory in a very short time, via-point extraction and pattern recognition can also be done in a short time. Arm trajectories can be generated by the FIRM neural network in a short time (Wada and Kawato 1993, 1994). We believe that understanding the movement pattern formation process is critical in understanding the movement pattern recognition process. Furthermore, we propose that the via-point is one of the key ingredient for movement pattern recognition based on movement-pattern generation.

In this paper, we propose a pattern recognition theory that is firmly based on the computational theory of handwriting, and utilizes the feature extraction algorithm that has already been proposed by Wada and Kawato (1993, 1994). A specific demonstration of an algorithm to recognize cursive connected script is presented within our general theoretical framework. Furthermore, it is shown that our proposed via-point estimation algorithm can extract the via-points from articulator trajectories measured during the natural continuous speech. Surprisingly and interestingly, the temporal locations of the extracted via-points roughly correspond to those of phonemes. However, our most important contribution is to demonstrate the computational realizability for the "motor theory of movement-pattern perception".

2. A computational theory for pattern recognition based on pattern generation

The basic ideas underlying our computational theory for pattern recognition are shown in Figure 1. The generation process and the recognition process are the two oppositely directed flows of a single function. That is, the generation process is the flow that generates a movement pattern according to an intended task, and conversely, the recognition process is the flow that estimates the intended symbols or phoneme strings from a given movement data.

When a person intends to write characters or to speak, first, the character symbol sequence or the speech phoneme string is planned. Next, a motor command is calculated by an optimal trajectory planner and motor command generator, according to the

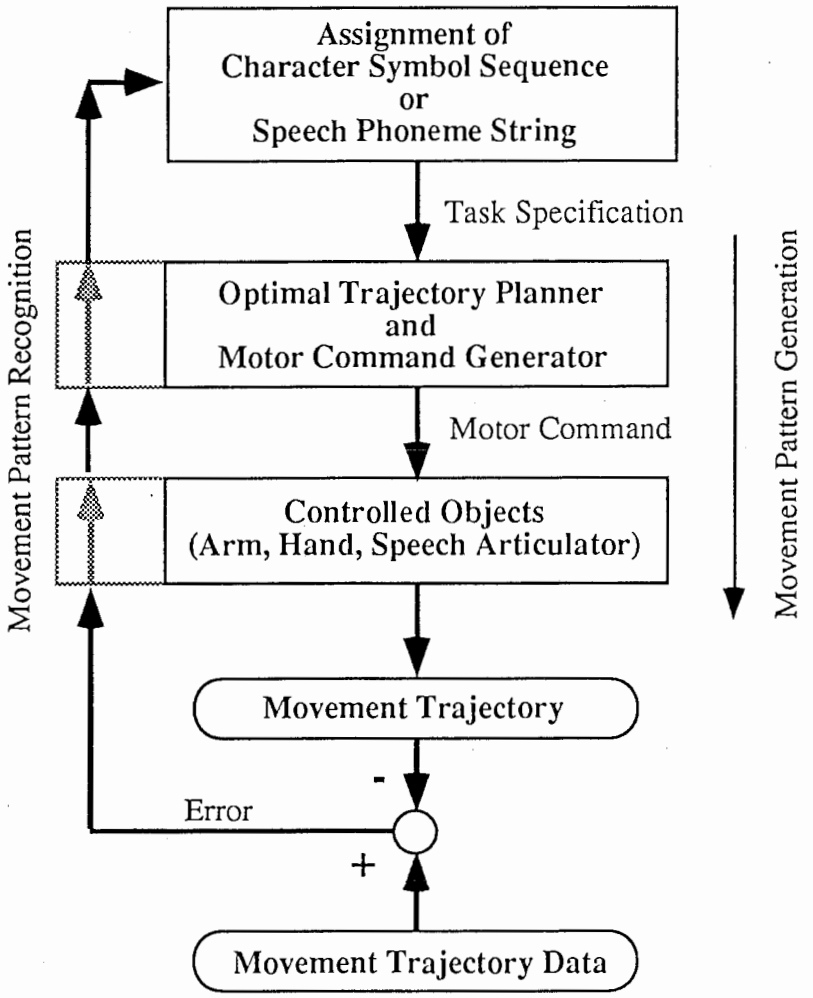


Figure 1
Fundamental scheme of movement pattern recognition based on movement pattern generation.

computational theory of continuous movement-pattern generation. Finally, our arm, hand, or speech articulator is allowed to move according to the generated motor command. Thus, the above steps can be summarized by a transformation from symbols to movement trajectory. In a sense, the symbols can be communicated through the movement trajectories of the arm, hand or articulator, when the perceiver has the capability to recover the intended symbol from the movement trajectory data that is produced by the performer. Accordingly, if we can assign symbols so as to plan the same given data trajectory as the intended movement trajectory when we observe movement trajectory data, it is equivalent to the movement trajectory data being understood. If there exists an error between the movement trajectory data and the putatively generated trajectory, the intended symbol is corrected and the generation is repeated.

There should exist an intermediate representation between the symbol and movement trajectory commonly used for movement perception and generation. We have already proposed the pattern generation theory (Wada and Kawato 1994) based on the optimization principle. There, the via-points on the pattern are understood as an intermediate representation of a symbol, which is also a set of boundary conditions in the optimization process, in the theory. Also, we assumed that the templates of the via-points stored for a given symbol can be acquired through previous experience of handwriting examples. Therefore, in the pattern generation phase, an intended symbol is first translated into a set of via-points by access to the stored template, and then the pattern is generated by using the via-points. In pattern recognition phase, the process is almost perfectly inverted. A movement pattern is given and the via-points are extracted from the pattern. Finally, a symbol is identified with reference to the extracted via-points (shown

in Figure 2). In this more specific version of our movement pattern recognition theory, we propose that perception and generation share the same set of invariant features, i.e. via-points.

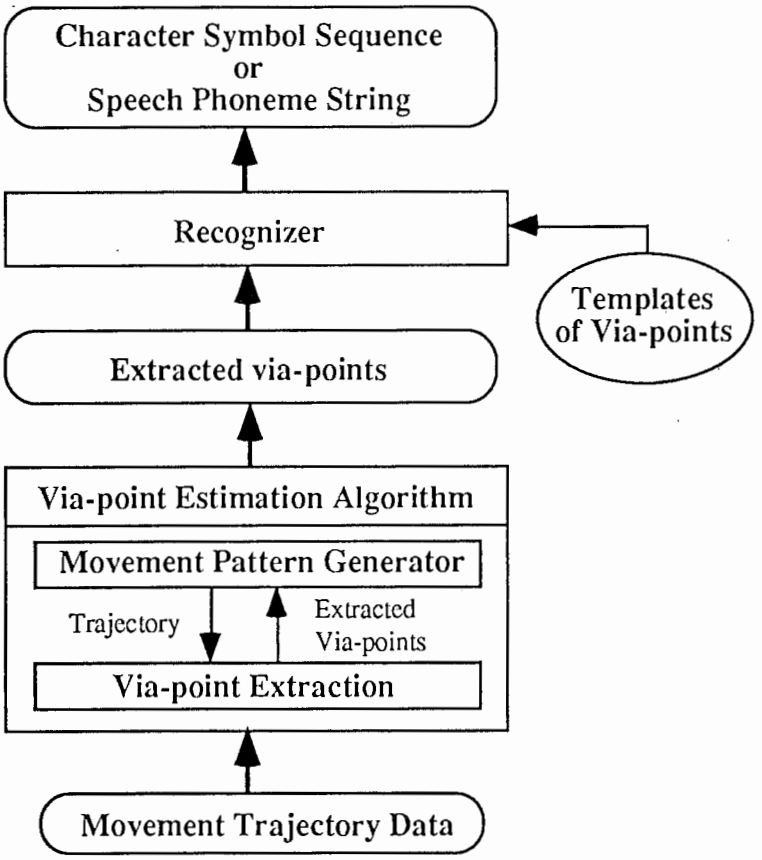


Figure 2
Movement pattern recognition using extracted via-points obtained through movement pattern generator.

Our proposed via-point estimation algorithm (Wada and Kawato 1994) finds the via-points by iteratively activating both the movement pattern generator (FIRM) and the via-point extraction module. The movement pattern generator generates a trajectory based on the minimum torque-change criterion using the via-points which are extracted by the via-point extraction module. The via-point extraction module assigns the via-points so as to minimize the square error between the movement trajectory data and the trajectory generated by the movement pattern generator. The via-point extraction algorithm will stop when the error between the given trajectory and the trajectory generated from the extracted via-points decreases below a threshold. The following three points are important characteristics of the via-point estimation algorithm: (1) the number of the via-points is approximately minimized, (2) there is a good reason for the choice of every via point locus, (3) the trajectory passing through via-points is the optimal trajectory.

This via-point estimation algorithm is an interesting case where the fundamental scheme of movement pattern recognition shown in Figure 1 is computationally realized. We believe that the via-point estimation algorithm should be quite useful in a wide range of continuous movement pattern perception. We will first apply the algorithm to cursive handwritten character recognition, and then to phoneme timing estimation in this paper. By these demonstrations, we show that the computational theory for movement pattern recognition can be actually used for real world data.

3. Cursive connected character recognition

3.1 Difficulties in cursive connected character recognition

There are at least two difficulties in cursive handwritten character recognition. The

first is the many variations in the shape of a character. The second is the ambiguity of character segmentation. The first difficulty should be resolved by extracting information that is stable, invariant and which can be reduced to small number of descriptions, that is, the characteristic features of the character. The second difficulty should be resolved by extracting a small number of segmentation candidates, based on the same principle used in the first problem. There is a possibility that we can find the efficient algorithm which can solve both difficulties if we well understand how humans solve the character recognition problem. We take the radical position advocated by the general theory developed in the previous section. Especially, we will use the via-points extracted as the characteristic features of a character. Because our proposed via-point estimation algorithm can extract the information needed to regenerate a character trajectory and the information compression rate is very high, it is expected that the spatial and temporal arrangement of the via-points becomes somewhat an invariant feature of a character. Also, Wada and Kawato (1994) have already confirmed that the via-point estimation algorithm can pick up a segmentation point between characters for connected cursive handwritten characters as one of the extracted via-points. By extending this observation, if one generally assigns a via-point between characters in writing connected cursive characters, the via-point estimation algorithm can always estimate the segmentation point as a via-point. The number of estimated via-points is much smaller than the number of measured data points. Thus, even if all the estimated via-points are treated as segmentation candidates, the ambiguity of character segmentation can be drastically reduced. Furthermore, as shown below, we can devise a more efficient procedure to decrease the number of segmentation-point candidates.

3.2 A recognition schema for cursive connected characters

A schematic illustration of our cursive connected character recognition procedures is shown in Figure 3. We used a simple recognition method: template matching. Generally, in pattern recognition, the most critical problems are what kind of feature space is selected, and what distance function is defined in that space. Particularly, the feature space seems to be the key issue. Thus, especially in this pilot study, we put much less emphasis on the recognition method itself compared with the selection of the feature space. In the following recognition schema, the feature space is constructed using the extracted via-points. It must be emphasized that the extracted via-points are one of the sets of features that can reproduce the original pattern when the appropriate trajectory generator is specified. That is, it can be said that the pattern information resides in the extracted via-points, which are assigned by the specified trajectory formation theory. In the following, we explain (1) the via-point estimation algorithm, (2) character segmentation, (3) normalization of features, (4) a trajectory reformation model, and (5) the distance function:

(1) Via-point estimation algorithm

As mentioned above, Wada and Kawato (1994) have already proposed the via-point estimation algorithm. Our algorithm depends only on the minimization principle and uses no *ad hoc* information to assign the via-points. It has been mathematically shown in Wada and Kawato (1994) that a given trajectory is approximated by this method with infinite accuracy (completeness), and furthermore, that the number of extracted via-points for a given threshold is approximately the minimum (optimality).

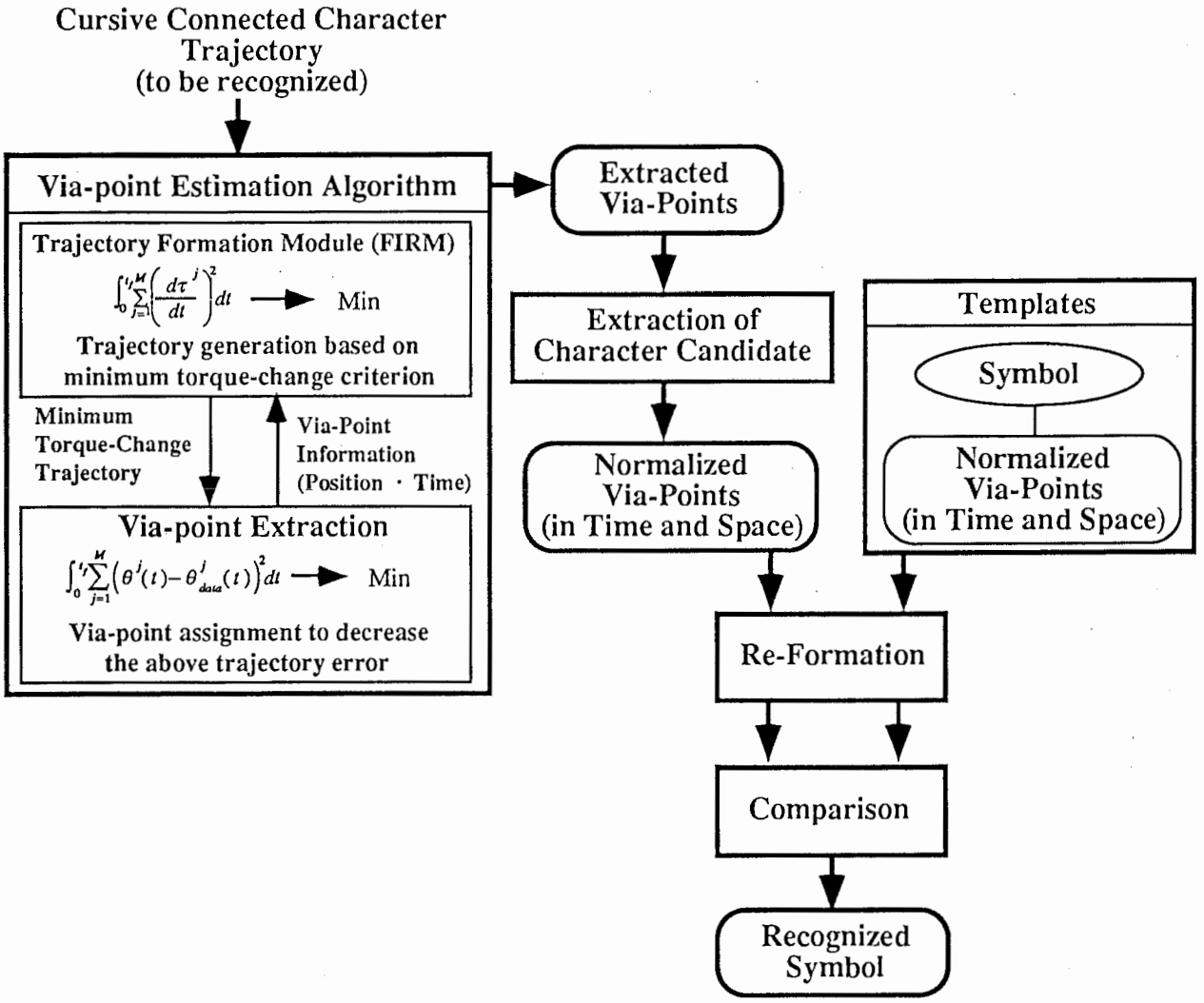


Figure 3
A recognition scheme for cursive connected handwritten characters using the via-point estimation algorithm.

(2) Normalization of features

In pattern recognition using the template matching method, the extracted features must be normalized so that they are directly compared with stored templates. The information about the via-points, i.e. the features, has three components: two Cartesian coordinates (X, Y) in the task plane and the time of passing through the via-point. Normalization of the coordinates (X, Y) means normalization of the character size and position. This is done by normalizing the sum of the distance between via-points to 1 and shifting the start point to the origin of the Cartesian coordinates. The following equation details these steps.

$$\tilde{X}^{i}_{via} = \tilde{X}^{i-1}_{via} + \frac{X^{i}_{via} - X^{i-1}_{via}}{\sum_{i=1}^{N}\sqrt{\left(X^{i}_{via} - X^{i-1}_{via}\right)^2 + \left(Y^{i}_{via} - Y^{i-1}_{via}\right)^2}} \quad (1)$$

$$\tilde{Y}^{i}_{via} = \tilde{Y}^{i-1}_{via} + \frac{Y^{i}_{via} - Y^{i-1}_{via}}{\sum_{i=1}^{N}\sqrt{\left(X^{i}_{via} - X^{i-1}_{via}\right)^2 + \left(Y^{i}_{via} - Y^{i-1}_{via}\right)^2}} \quad (2)$$

where $(N\text{-}1)$ is the number of via-points, $\left(X^{i}_{via}, Y^{i}_{via}\right)$ shows the task space coordinates of the i-th via-point on the given trajectory, and $\left(\tilde{X}^{i}_{via}, \tilde{Y}^{i}_{via}\right)$ expresses the normalized coordinates of the i-th via-point. Here, $i=0$ and $i=N$ are the start point and final point, respectively, and $\left(\tilde{X}^{0}_{via}, \tilde{Y}^{0}_{via}\right) = (0, 0)$ is the origin.

The time when the via-point is passed is normalized using the following equation.

$$\tilde{T}^{i}_{via} = \frac{T^{i}_{via} - T^{0}_{via}}{T^{N}_{via} - T^{0}_{via}} \quad (3)$$

where T^{i}_{via} is the time when the i-th via-point is passed in the original movement, $\tilde{T}^{i}_{via}$ $(0 \le \tilde{T}^{i}_{via} \le 1)$ shows the corresponding i-th normalized via-point time.

(3) Trajectory reformation model

As discussed above, the features are normalized in time and space. What is required at the final template matching step, therefore, is that the trajectory reformation model regenerates the expanded or reduced trajectory of the original trajectory without artificial distortion.

Because the minimum torque-change criterion depends on the arm's nonlinear dynamics, the trajectory formation model based on this criterion can not uniformly regenerate an expanded or reduced trajectory of the original trajectory in time and space. The minimum jerk model, however, is size and shift invariant. Additionally, it can be regarded as the first approximation to the minimum torque-change model, and predicts hand trajectories well in front of the body. That are quite similar to those predicted by the minimum torque-change model. Therefore, we use the minimum jerk criterion (Flash and Hogan 1985) as the trajectory reformation model in the recognition scheme for reproducing the trajectory from the normalized via-points.

(4) Distance function

Generally, the basic procedure in pattern recognition is that the distance between an input pattern and a standard pattern is measured, and if the distance is small, it is not unreasonable to assume that the input pattern is the same as the standard pattern.

In the recognition scheme shown in Figure 3, the distance function computes the Euclidean square distance between the template and the regenerated trajectory in the normalized time. Because the number of extracted via-points is controlled only by an error threshold and is not constant, it is difficult to find the distance between the template via-points and the regenerated via-points. The trajectories regenerated from the normalized via-points and the template via-points are therefore compared using the

following distance function:

$$D = \int_0^{t_{normal}} \left\{ \left(\tilde{X}(t) - \tilde{X}_T(t) \right)^2 + \left(\tilde{Y}(t) - \tilde{Y}_T(t) \right)^2 \right\} dt \qquad (4)$$

where t_{normal} expresses the normalized movement time (= *1*); $\left(\tilde{X}_T(t), \tilde{Y}_T(t) \right)$ defines the normalized template trajectory; and $\left(\tilde{X}(t), \tilde{Y}(t) \right)$ defines the trajectory regenerated by using the normalized via-point information.

(5) Character segmentation

Here, a character candidate is extracted from the movement trajectory data in order to compare the input trajectory with the template trajectory. All the via-points of the input trajectory are segmentation candidates of the character at the beginning. The basic procedure in the character segmentation (shown in Figure 4) is as follows:

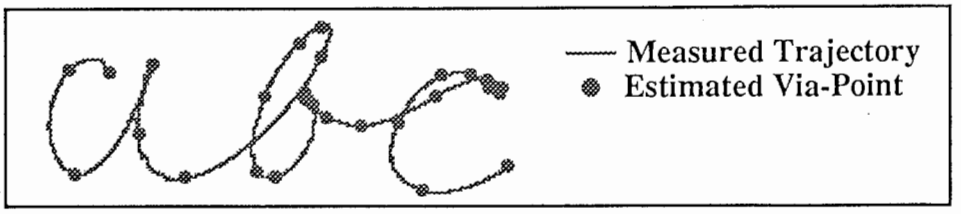
(Step I) At the beginning of the procedure, the window begins with V^0 and ends with $V^{(N_{char} - 1)}$, where $N_{char} = 3$ is the minimum number of via-points in a single window. Thus, at this stage the window ends with V^2. Next a search for the first segmentation point is conducted, by first regenerating a normalized trajectory using the trajectory formation model with the N_{char} via-points included in the window; then comparing this trajectory with the set of template trajectories, assigning a distance function value to each regenerated trajectory. Now, N_{char} is changed to $N_{char} + 1$, creating a new window that include *4* via-points beginning with V^0 and ends with V^3. Another set of comparisons and distance function assignment is made between the current window's set of $N_{char} + 1$ via-points and the stored templates. This iterative procedure continues until the window size include $N_{char} + \Delta N_{max}$ via-points, where $\Delta N_{max} = 22$, and $N_{char} + \Delta N_{max}$ denote the maximum number of via-points in a stored character template.

(Step II) Next, the window size is reset to N_{char}, and the window's first via-point is

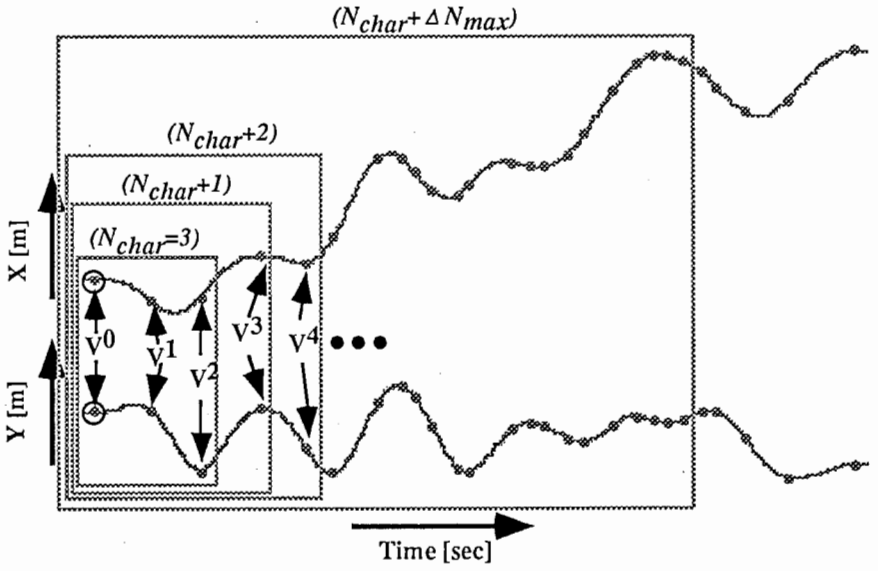
shifted by N_{shift} via-points, where $N_{shift} = 1$. That is, the window's first via-point is shifted from V^0 to V^1, and the window's final via-point is set to equal $V^{(i+1)} + (N_{char}-1) = V^{(i+N_{char})}$. With this new window location, the entire set of window expansions, trajectory comparisons, and score assignments are performed as in Step I. The three top-scoring combination of window size, window location, and stored character template are then used to define a corresponding set of three segmentation point candidates, S_{jk}, that define the final via-point of the first recovered character, where j indexes the recovered character and $k = 1, 2, 3$ indexes the candidate number. Thus, for example, the highest scoring candidate for the first recovered letter word is assigned segmentation point $S11$. A given S_{jk} is defined as the final via-point of the window associated with j-*th* letter's k-*th* highest score.

(Step III) The recovery procedure for the $(j+1)$-*th* letter in the sequence branches at this point into a set of 3 subprocedures, one for each of the S_{jk}'s associated with the preceding j-*th* letter. Each S_{jk} is used to define a window that begins at $V^{(S_{jk}-N_{shift})-1}$ and ends at $V^{(S_{jk}+N_{shift})-1}$. Each of these windows is used as the starting point for the entire set of iterative comparisons described in Steps I and II, and is used to define a set of 3^j segmentation points for the $(j+1)$-*th* letter in the sequence , that is, 3 new segmentation points are generated for each of the preceding letter's set of segmentation points.

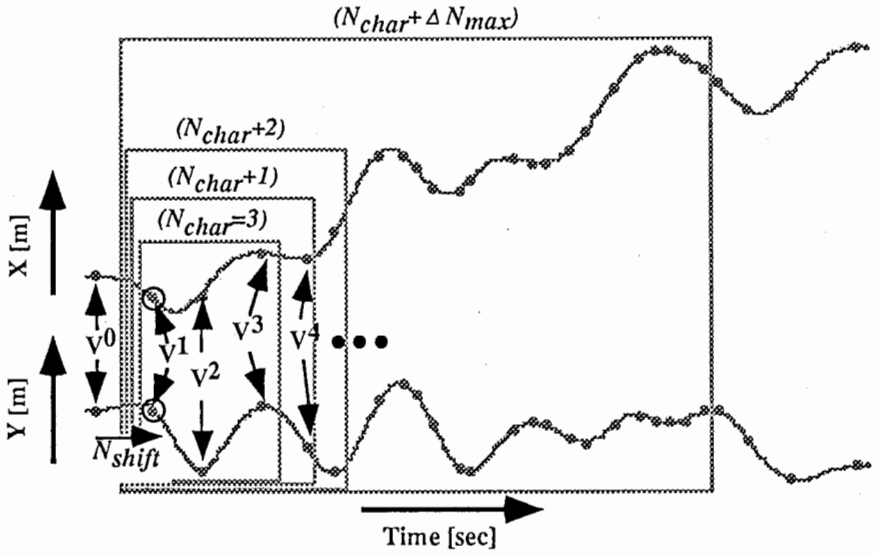
(Step IV) Finally, steps I through III are repeated until the S_{jk}'s on all branches of the computation occur at via-points $\geq V^{final-(N_{char}-1)}$, where V^{final} is the final via-point in the movement trajectory. That is, the process continues until the S_{jk}'s fall inside the minimum window size distance from the movement's final via-point.



Step I



Step II



Step III

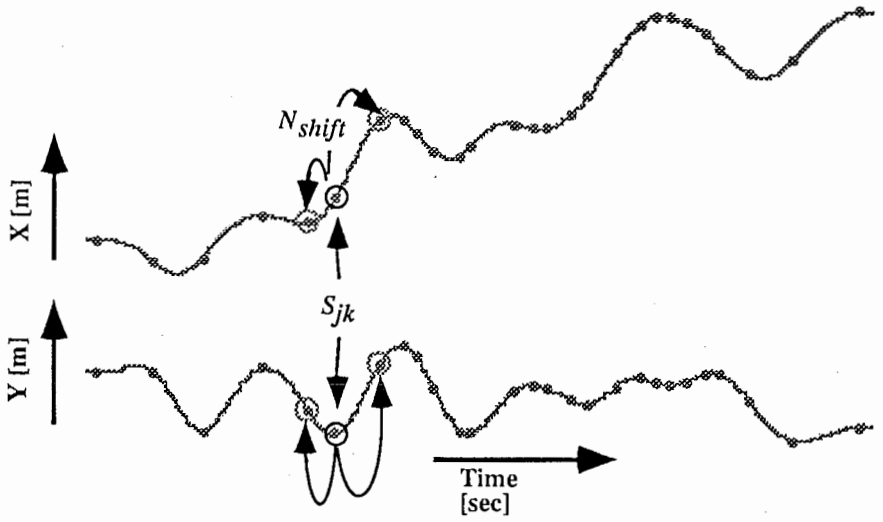


Figure 4
The procedure adopted to segment a character candidate from cursive connected characters. Dotted squares are windows whose inside trajectory is assumed to consist a single segregated character. The window width and the origin of the window are systematically changed according to the segmentation procedure explained in the text.

3.3 Performance of the character recognition model

The results of character recognition experiments are shown in this section. Figure 5 shows some measured trajectories and extracted via-points. The black circles represent the extracted via-points. Some regenerated trajectories produced by the minimum jerk model are shown in Figure 6. The regenerated trajectories produced by the minimum jerk model are almost the same as the measured trajectories confirming that the minimum-jerk model is a good first approximation to the minimum torque-change model for these trajectories. In these experiments, we do not deal with movements such as those used to form '*i*', '*j*', '*t*' and '*x*' because detection of pen-up and pen-down is necessary for these characters.

The performance of the character recognition model is shown in Figure 7. The 36 template trajectories shown in Figure 6 were used in this experiment. The trajectories in Figures 6 and 7 were produced by two different persons. The reason why several templates for a single character are stored is that even if the character is the same, the shape of the character placed first in a word is different from its shape when placed second or the third in the word. For example, '*d*' in Figure 7 is a typical case. The right-hand side in Figure 7 shows the recognition results for the left-hand side. The three candidates for recognition are listed. Numerals after the recognized character express values of the distance function (4), and the numbers in parentheses show the numbers of starting via-points and the final via-point for the recognized character. All via-points are numbered sequentially from the start to the end. Starting via-points are assigned the number 0. It is clear that the characters can be recognized. Although we know that it is hard to recognize a character that includes a part similar to other characters, for example, the third candidate for '*good*' was mistakenly identified as '*gooa*'. However, our

recognition model does not use a word dictionary or any other information such as context, only trajectory information. A mistake such as '*gooa*' can be corrected by using other information.

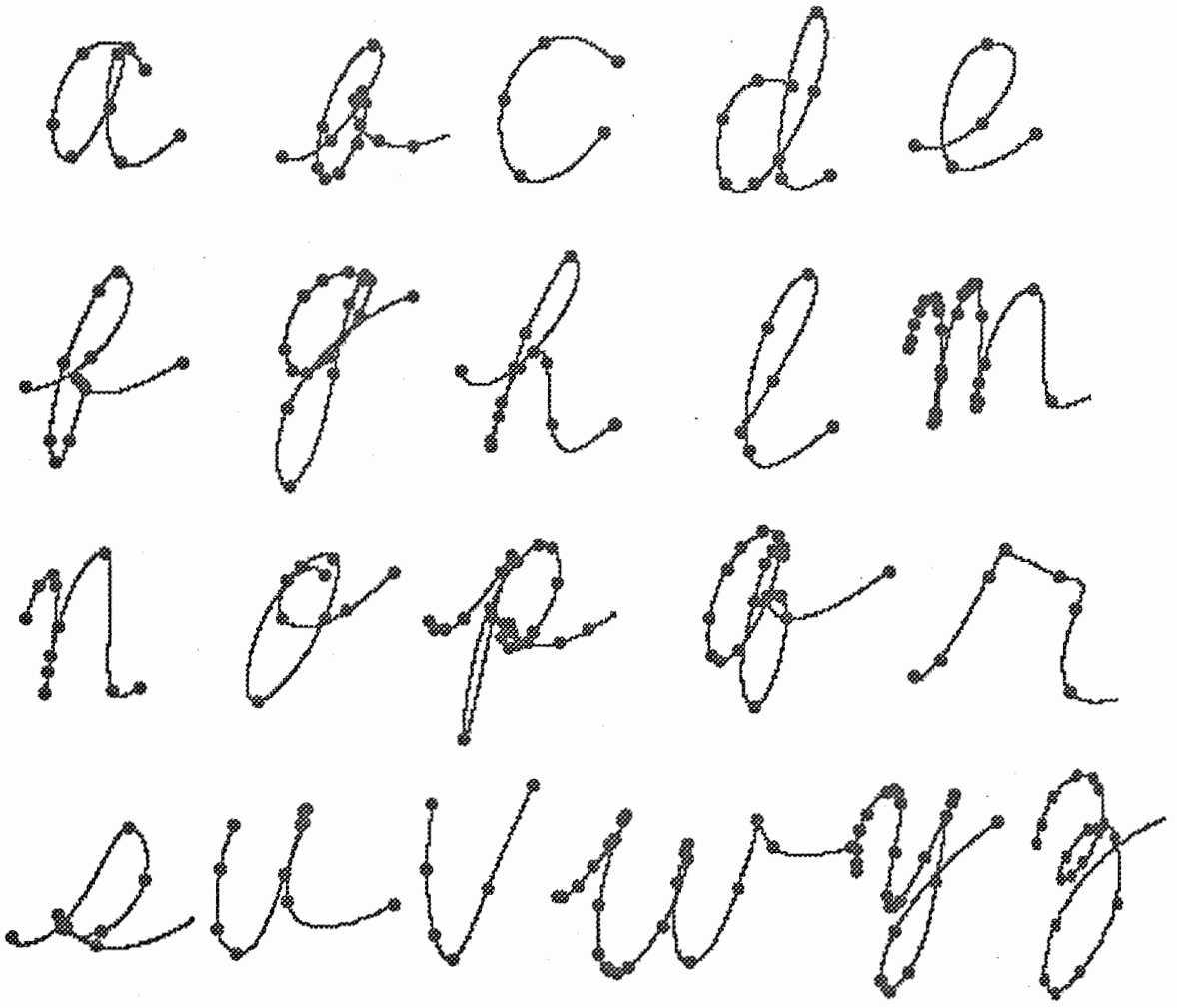


Figure 5
Measured characters and estimated via-points. Heavy dots denote via-points.

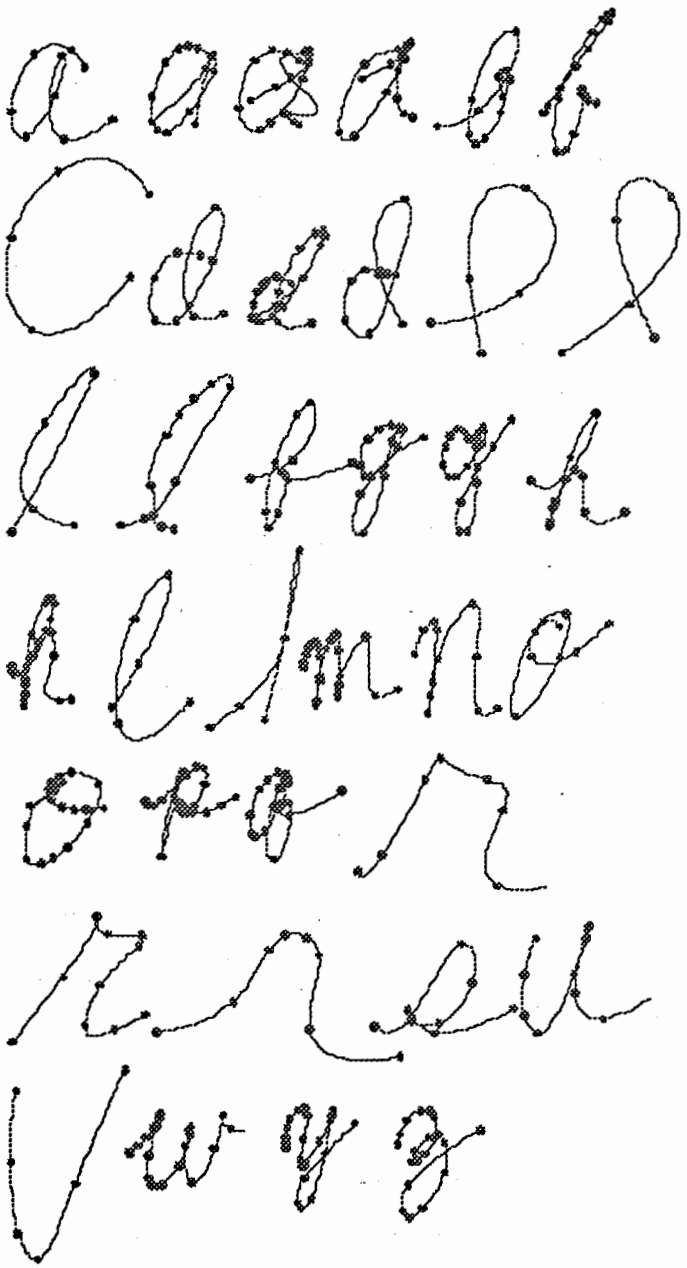
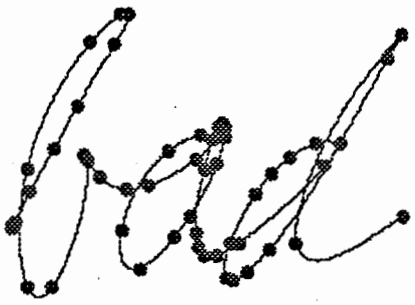
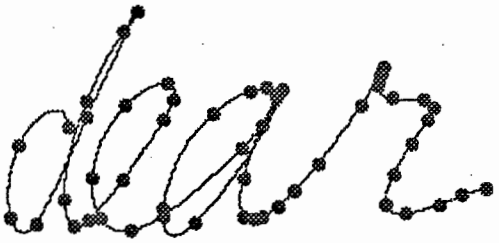


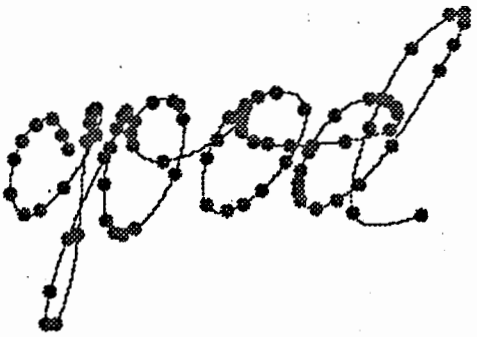
Figure 6
Template trajectories for several characters regenerated by the minimum jerk criterion. Heavy dots are the extracted via-points from the measured trajectory.



1 :BAD : 0.004072 (start via-point No. for /b/ = 0, final via-point No. for /b/ = 17)
(start via-point No. for /a/ = 18, final via-point No. for /a/ = 35)
(start via-point No. for /d/ = 36, final via-point No. for /d/ = 52)
2 :BAD : 0.004155 (start via-point No. for /b/ = 0, final via-point No. for /b/ = 18)
(start via-point No. for /a/ = 18, final via-point No. for /a/ = 35)
(start via-point No. for /d/ = 36, final via-point No. for /d/ = 52)
3 :BAD : 0.004179 (start via-point No. for /b/ = 0, final via-point No. for /b/ = 17)
(start via-point No. for /a/ = 18, final via-point No. for /a/ = 35)
(start via-point No. for /d/ = 35, final via-point No. for /d/ = 52)



1 :DEAR : 0.018376 (start via-point No. for /d/ = 0, final via-point No. for /d/ = 8)
(start via-point No. for /e/ = 9, final via-point No. for /e/ = 18)
(start via-point No. for /a/ = 19, final via-point No. for /a/ = 31)
(start via-point No. for /r/ = 30, final via-point No. for /r/ = 51)
2 :DEAR : 0.018791 (start via-point No. for /d/ = 0, final via-point No. for /d/ = 8)
(start via-point No. for /e/ = 9, final via-point No. for /e/ = 18)
(start via-point No. for /a/ = 19, final via-point No. for /a/ = 31)
(start via-point No. for /r/ = 30, final via-point No. for /r/ = 50)
3 :DEAR : 0.019001 (start via-point No. for /d/ = 0, final via-point No. for /d/ = 8)
(start via-point No. for /e/ = 9, final via-point No. for /e/ = 18)
(start via-point No. for /a/ = 19, final via-point No. for /a/ = 30)
(start via-point No. for /r/ = 30, final via-point No. for /r/ = 51)



1 :GOOD : 0.007163 (start via-point No. for /g/ = 0, final via-point No. for /g/ = 21)
(start via-point No. for /o/ = 20, final via-point No. for /o/ = 38)
(start via-point No. for /o/ = 39, final via-point No. for /o/ = 58)
(start via-point No. for /d/ = 59, final via-point No. for /d/ = 84)
2 :GOOD : 0.007692 (start via-point No. for /g/ = 0, final via-point No. for /g/ = 21)
(start via-point No. for /o/ = 20, final via-point No. for /o/ = 38)
(start via-point No. for /o/ = 39, final via-point No. for /o/ = 59)
(start via-point No. for /d/ = 60, final via-point No. for /d/ = 84)
3 :GOOA : 0.007916 (start via-point No. for /g/ = 0, final via-point No. for /g/ = 21)
(start via-point No. for /o/ = 20, final via-point No. for /o/ = 38)
(start via-point No. for /o/ = 39, final via-point No. for /o/ = 58)
(start via-point No. for /d/ = 58, final via-point No. for /d/ = 82)

Figure 7
Results of word recognition without dictionary.

4. Estimating the temporal locations of phonemes from speech articulator motion using the via-point estimation algorithm

In the this section, the via-point estimation algorithm is modified slightly and applied to speech articulator motion. It is shown that the model can assign via-points that correlate well with phonemes.

4.1 An algorithm for identifying the temporal locations of phonemes

The algorithm proposed here for phoneme localization is based on the via-point estimation algorithm proposed for handwriting (Wada and Kawato 1994). A trajectory generation model for speech articulators has been proposed which is based on the forward dynamics model of the musculoskeletal system (Kawato 1989; Hirayama, Vatikiotis-Bateson, Honda, Koike and Kawato 1993). Generally, the articulator biomechanics is complex and difficult to model. Hirayama et al. (1993) succeeded in learning the dynamical system of the articulators by using neural networks that process physiological data from muscles, kinematic data from the lip and jaw, and speech acoustics. Furthermore, Hirayama, Vatikiotis-Bateson and Kawato (1993) succeeded in speech synthesis of natural sentences using the acquired forward acoustic model. Most recently, Hirayama, Vatikiotis-Bateson and Kawato (1994) generated an articulator trajectory utilizing the learned forward and inverse dynamics models of the speech articulators in the FIRM configuration. Thus, it is possible in principle to use the dynamical system model of the speech-articulator musculoskeletal system, and then develop the dynamic optimization principle for trajectory formation such as the minimum-

muscle tension change model. However, in practice, it is much more difficult to construct a dynamical model of the speech articulators than one for the arm partly because its inherent biomechanics is more complicated than arms', and it is technically much more difficult to collect reliable physiological data such as EMG.

Thus, in the present paper, we model each speech articulator (tongue body vertical position, tongue tip vertical position, jaw vertical position and lower-lip vertical position in the extrinsic Cartesian coordinates: see TBY, TTY, JY and LLY respectively in the bottom of Figure 8) as a simple point mass. Then, a dynamic optimization principle such as minimum EMG-change reduces to a kinematics optimization principle such as the minimum jerk model. In another interpretation, the minimum-jerk, the minimum motor-torque-change, and the minimum muscle-tension change (Uno, Suzuki and Kawato 1989) are the first, second and third approximation to the minimum motor-command change model (Kawato 1994). In the absence of reliable quantitative model of speech articulator dynamics, we are forced to select the first simplest approximation.

Consequently, the via-point estimation algorithm which we actually used for speech was even simpler than that for handwriting, in that we used the minimum-jerk model in extrinsic Cartesian coordinates as the optimal trajectory generation mechanism. Note that the trajectory is represented and planned in an 8-dimensional space defined by the vertical and horizontal positions of the 4 speech articulators. The minimum-jerk trajectory is generated by using a spline function because the minimum-jerk criterion is equivalent to the definition of the spline function. In this method, the required spline coefficients are computed using matrix inversion in the via-point estimation algorithm (Wada and Kawato 1994).

There are several ways to define error thresholds between the reconstructed and data

trajectories for applying the via-point estimation algorithm to speech articulator motion. For example, (1) if the four articulators are completely independent, the extracted via-point times for each articulator can be likewise independent, or (2) if the four articulators move completely cooperatively, then the extracted via-point times will be identical across articulators. The former possibly is based on the assumption that there exists no interaction between the four articulators; the second assumes tight couplings similar to those found between the elbow and the shoulder. The third, which we adopted here, is a method intermediate between (1) and (2), and is based on the assumption that patterns of interaction vary cooperatively over time within a given utterance. Thus, two error thresholds are introduced. One is a threshold ($S1$) for the sum, E_A, of errors at a given point on time between the measured trajectories and the regenerated trajectories across all of the articulators: $E_A(t) = E_J(t) + E_{TT}(t) + E_{TB}(t) + E_{LL}(t)$, where J, TT, TB, and LL denote, respectively, the jaw, tongue tip, tongue blade, and lower lip. The other is a threshold, $S2_i$, for each separate articulator, where $i = J, TT, TB$, and LL. Thresholds $S1$ and $S2_i$ are used to find a via-point candidate in the following manner. When $E_A(t) > S1$, the error $E_i(t)$ for each articulator-i is checked and, if it is above $S2_i$, the corresponding point on articulator-i's measured trajectory is selected as a via-point.

4.2 Performance in identifying the temporal location of phonemes

The acoustic waveform, spectrogram, and vertical positions for the four articulators are shown as functions of time in normal (Figures 8) and faster (Figures 9) speaking rates. The locations of the phonemes, which were identified by eye from acoustic and spectrogram data, are represented by solid vertical lines. The open circles mark the via-

points estimated by the algorithm. Rather good agreement was found between the locations of the phonemes and estimated via-points. The dotted curves show the articulator trajectories reconstructed by the trajectory formation model based on the minimum jerk criterion. The reconstructed (dotted curves data) and the measured trajectories (solid curves) almost overlap and cannot be seen separately in Figure 8. As can be seen in Figure 8, the number of via-points required for adequate trajectory estimation is only slightly greater than the number of acoustically and spectrally derived segment labels (phonemes). Notice that via-points are not always assigned to all articulators in time. This situation arises when the error threshold for a specific articulator is not reached. Error thresholds for $S1$ and $S2_i$ for the via-point assignments are the same in both normal rate (Figure 8) and faster rate (Figure 9) utterances. If we compare the trajectories in Figures 8 and 9, we see that they are quite similar, but fewer points tend to be assigned at the faster rate.

On the other hand, when a regular sampling method is used instead of the via-point estimation algorithm, the agreement between the locations of the phonemes and the sampled points was almost the same as our proposed algorithm. That is, when the sampling pitch is 73 msec, which was calculated as the movement duration divided by the number of extracted via-points by own method, for the normal speaking rate (Figure 8), the good agreement was found. For the faster speaking rate (Figures 9), the sampling pitch (73 msec) should be changed to 57 msec, which was calculated as the movement duration divided by the number of extracted via-points by own method. However, parameters for the via-point estimation algorithm are completely the same in both normal rate and faster rate utterances.

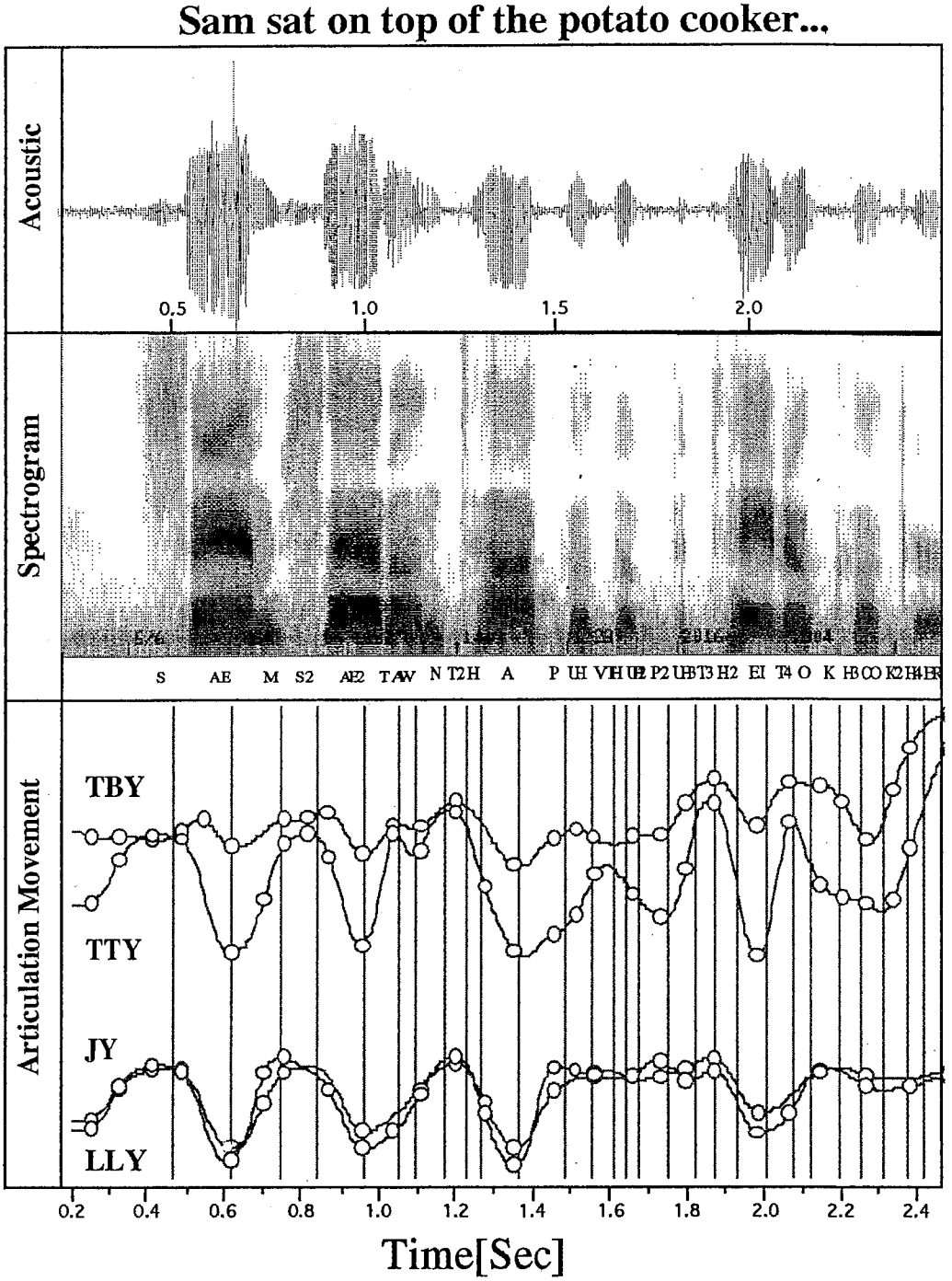


Figure 8 Estimation result of phoneme time. Temporal acoustics and vertical positions of the tongue blade (TBY), tongue tip (TTY), jaw (JY), and lower lip (LLY) are shown with overlaid via-point trajectories. Vertical lines correspond to centers of phonemes which are acoustically segmented; ○ denote via-points. Speaking rate was normal.

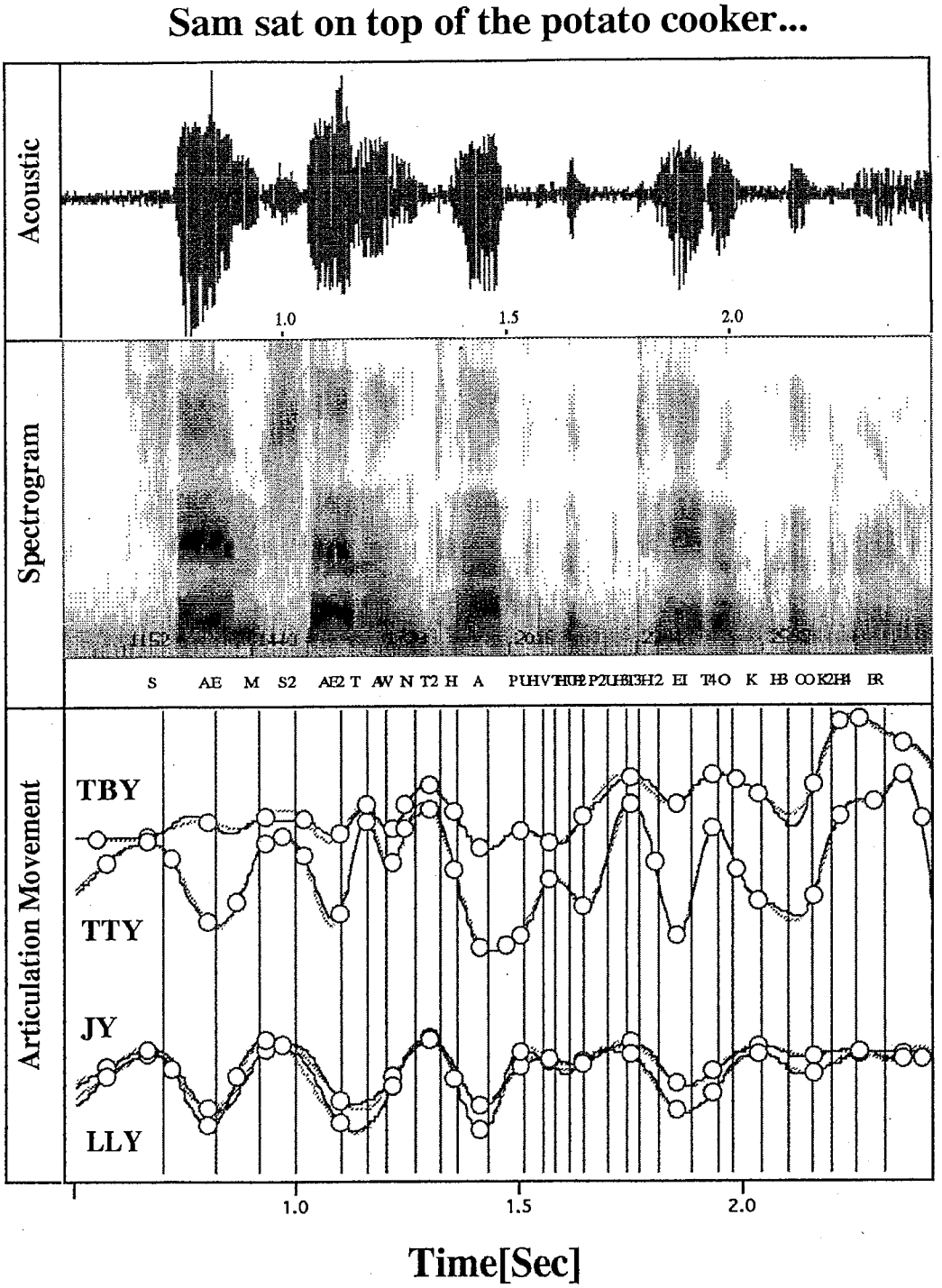


Figure 9 Estimation result of phoneme time for fast movement. Articulator trajectories and via-point assignments were produced at the faster speaking rate for the phrase shown in Figure 8.

From the above experiments, we can point out two possible engineering applications of the via-point estimation algorithm for phoneme location. It should be possible to extend the via-point estimation algorithm to both speech data compression and recognition if an inverse acoustic mapping from the acoustic waveform to the articulator motion trajectory is identified (Shirai and Kobayashi 1991; Papcun et al. 1992).

Since the regenerated articulator trajectories in Figures 8 and 9 are almost the same as the original trajectories, this means that the information encoded in the original trajectories was preserved in compressed form by the via-points extracted by our proposed algorithm. Thus, if the forward acoustics mapping from the articulator motion to the acoustic waveform (Hirayama, Vatikiotis-Bateson and Kawato 1993) is utilized in combination with the trajectory generator which reconstructs the articulator motion from the via-points, it should be possible to use the via-point estimation model for speech data compression. Finally, it should also be possible to perform speech recognition based on the via-point patterns extracted from the articulator trajectories recovered from the inverse accoustic mapping. We note, however, that before a recognition model using a template matching method on the extracted via-point patterns is attempted, it will be necessary to more fully understand the mapping between the via-points and the corresponding phonemes.

5. Conclusion

We have demonstrated the computational potential of our theory of movement pattern recognition based on movement-pattern generation in two areas of movement pattern perception: connected cursive handwritten character recognition and the estimation of

phoneme timing. The via-point estimation algorithm already proposed was applied to both of the above two areas. In these demonstrations, there is a crucial relationship between pattern recognition and pattern generation. That is, the pattern generator also critically works in pattern recognition. We incorporated the formation model into the recognition model and computationally realized the perception model suggested by Freyd (1983), Liberman and Mattingly (1985), and Kawato (1989). As a conceptual base of these demonstrations, we proposed the computational theory for movement pattern recognition shown in Figure 1.

The specific computational procedures actually adopted in the demonstrations such as the distance measure in character recognition are not so important. As engineering applications, we need to refine most of the procedures, so that the proposed method can be competitive with other state-of the-art techniques for character recognition. On the other hand, as actual computation executed in the brain, many of the current procedures are not biologically plausible. The objective of this study is to demonstrate the computational realizability of the "motor theory of speech perception". With the recent development of our optimal control theory for continuous movement and the FIRM neural network, we showed that it is computationally possible and actually quite powerful to perceive the continuous movement pattern based on the optimal movement pattern generator.

REFERENCES

Babcock MK, Freyd JJ (1988). Perception of dynamic information in static handwritten forms. *American Journal of Psychology*, **101**(1):111-130.

Freyd JJ (1983). Representing the dynamics of a static form. *Memory & Cognition*, **11**:342-346.

Flash T, Hogan N (1985). The coordination of arm movements; An experimentally confirmed mathematical model. *Journal of Neuroscience*, **5**: 1688-1703

Haken H, Kelso JAS, Fuchs A, Pandya AS (1990). Dynamic pattern recognition of coordinated biological motion. *Neural Networks*, **3**: 395-401.

Hirayama M, Vatikiotis-Bateson E, Kawato M (1993). Physiologically-based speech synthesis using neural networks. *IEICE Trans. Fundamentals*, E76-A, 1898-1910.

Hirayama M, Vatikiotis-Bateson E, Honda K, Koike Y, Kawato M (1993). Physiologically based speech synthesis. In Giles CL, Hanson SJ, Cowan JD (eds) *Advances in Neural Information Processing Systems 5*. 658-665, San Mateo, CA: Morgan Kaufmann Publishers.

Hirayama M, Vatikiotis-Bateson E, Kawato M (1994). Inverse dynamics of speech motor control, *Advances in Neural Information Processing Systems 6*, San Mateo, CA: Morgan Kaufmann Publishers (in press).

Kawato M (1989). Motor theory of speech perception revisited from minimum torque-change neural network model. In *8th Symposium on Future Electron Devices*, 141-150, Tokyo, Japan.

Kawato M (1994). Trajectory formation in arm movements: minimization principles and procedures. In Zelaznik HN(Ed.) *Advances in Motor Learning and Control*, Human Kinetics Publishers, Chanpaign Illinois (in press).

Kawato M, Maeda Y, Uno Y, Suzuki R (1990). Trajectory formation of arm movement by cascade neural network model based on minimum torque-change criterion. *Biol. Cybern.* **62**: 275-288

Liberman AM, Cooper FS, Shankweiler DP, Studdert-Kennedy M (1967) .Perception of the speech code. *Psychological Review*, **74**: 431-461.

Liberman AM, Mattingly IG (1985). The motor theory of speech perception revised. *Cognition*, **21**: 1-36.

Marr D. (1982). Vision. New York: Freeman

Papcun J, Hochberg J, Thomasm TR, Laroche T, Zacks J, Levy S (1992). Inferring articulation and recognition gestures from acoustics with a neural network trained on x-ray microbeam data. *Journal of Acoustical Society of America*, **92 (2) Pt. 1.**

Shirai K, Kobayashi T (1991). Estimation of articulatory motion using neural networks. *Journal of Phonetics*, **19**, 379-385.

Uno Y, Kawato M, Suzuki R (1989) Formation and control of optimal trajectory in human arm movement - minimum torque-change model. *Biol. Cybern.*, **61**: 89-101

Uno Y, Suzuki R, Kawato M (1989). Minimum muscle-tension-change model which reproduces human arm movement. *Proceedings of the 4th Symposium on Biological and Physiological Engineering*, 229-302 (in Japanese)

Wada Y, Kawato M (1993). A neural network model for arm trajectory formation using forward and inverse dynamics models. *Neural Networks* , **6**: 919-932

Wada Y, Kawato M (1994). A theory for cursive handwriting based on the minimization principle. Submitted to *Biol. Cybern.*