On-Line Handwriting Recognition

TORU WAKAHARA, MEMBER, IEEE, HIROSHI MURASE, MEMBER, IEEE,
AND KAZUMI ODAKA, MEMBER, IEEE

Invited Paper

On-line handwriting recognition means that a machine recognizes each character as it is being written. For large-alphabet languages, like Japanese, handwriting input using an on-line recognition technique is essential for input accuracy and speed. It offers several advantages over off-line handwriting recognition. However, there are serious problems that prevent high recognition accuracy without imposing handwriting constraints. First, the thousands of ideographic Japanese chsaracters of Chinese origin (called Kanji) can be written with wide variations in the number and order of strokes and significant shape distortions. Also, writing box-free recognition of characters is required to create a better man–machine interface. This paper describes the intense research performed by NTT over the past 15 years to answer the most pressing recognition problems. Prototype systems developed by NTT are also described. Last, the man–machine interfaces made possible with on-line handwriting recognition and anticipated advances in both hardware and software are discussed.

Keywords*—*user friendly input to computers, on-line handwriting recognition, Kanji character recognition, character sequence recognition, tablet digitizers.

I. INTRODUCTION

For preparing a first draft and concentrating on content creation, pencil and paper are often superior to keyboard entry. Handwriting recognition allows the creativity of handwriting to be combined with the advantages of word processors. For large-alphabet languages, like Japanese or Chinese, keyboards are particularly cumbersome and inefficient. If large-alphabet languages could be input via on-line handwritten character recognition, more effective man–machine interfaces could be created. For example, the user can easily detect and correct misrecognized characters on the spot by verifying the recognition results as they appear. Hence, on-line handwriting recognition for transcription is being intensively investigated by mainly Japanese or Chinese scientists. Chinese characters or Kanji in Japan consist of many strokes and many complex compounds need to be distinguished. One character in the block style has an average of 8–10 strokes, the simplest character having one stroke with the most complicated having more than 30. Most people do not strictly follow the correct order of strokes when writing a character. Moreover, a cursive style is preferred when writing quickly. Fewer strokes are used and the radical shapes are often deformed and simplified. To resolve the above problems, NTT proposed a series of robust recognition algorithms based on a stroke by stroke matching strategy composed of two steps. The first step is stroke correspondence determination, which is indispensable for realizing stroke number and stroke order free recognition. The second step is the calculation of intercharacter distances between a large number of templates and a distorted input pattern to realize distortion-tolerant pattern matching. This strategy was proven to be very powerful by extensive recognition tests. Moreover, for writing box-free recognition of character sequences, an enhanced algorithm was proposed that selected the optimal segmentation among all segmentation possibilities advanced by the recognition process. This algorithm was also successfully applied to free-format line figure recognition. Together with research into recognition algorithms, prototype systems were developed to create a user friendly Japanese input device using on-line handwriting recognition. Other important uses of on-line handwriting recognition are editing, annotating, and other applications that are heavily interactive or that use direct pointing and manipulation. An example of this is a gesture interface, an area of substantial recent interest. We discuss the future man–machine interfaces made possible with on-line handwriting recognition as well as anticipated advances in both hardware and software.

Manuscript received Feb. 21, 1991; revised Oct. 30, 1991.

T. Wakahara is with NTT Human Interface Labs, Kanagawa 238-03, Japan.

H. Murase and K. Odaka are with NTT Basic Research Labs, Tokyo 180, Japan.

IEEE Log Number 9202478.

II. ON-LINE VERSUS OFF-LINE RECOGNITION

Off-line handwriting recognition is performed after the writing is completed. An optical scanner converts the image of the writing into a bit pattern. This technique cannot use any dynamic writing information: the number of strokes, the order of strokes, and the direction of stroke creation. This makes it difficult to extract and identify the strokes of each character. The best off-line recognition algorithms

use pattern matching to extract static shape features with coordinates and so match the image as a feature array.

On-line recognition, by contrast, means that the machine recognizes each character as it is being written. The preferred input device is an electronic tablet with a stylus pen. The electronic tablet captures the x-y coordinate data of pen-tip movement, which typically has a resolution of 10 points/mm, a sampling rate of 100 points/s, and an indication of pen-up and pen-down. That is, the trace of the handwriting or line drawing is captured as a sequence of coordinate points with separate stroke indications.

On-line recognition has two major technological advantages over off-line recognition. The first is high-recognition accuracy. This is because on-line recognition captures a character as a set of strokes which are represented a series of coordinate points while off-line recognition only has a bit pattern and shape feature extraction is mostly based on heuristics. This advantage becomes conspicuous when dealing with strongly distorted characters written in the cursive style. The other advantage is interaction. In on-line recognition, it is very natural for the user to detect and correct misrecognized characters on the spot by verifying the recognition results as they appear. The user is encouraged to modify his writing style so as to improve recognition accuracy. Also, a machine can be trained to a particular user's style. Samples of his misrecognized characters are stored to aid subsequent recognition. Thus both writer adaptation and machine adaptation is possible. Moreover, editing, annotating, and other applications that use direct pointing and manipulation are well suited to on-line handwriting recognition.

III. Historical Overview

Tablet digitizers have existed for over three decades. The earliest prototype for on-line handwriting recognition was T. L. Dimond's "stylator" [1] in 1957. The RAND tablet [2] was clearly the most popular of the early digitizers and spurred intense activity in on-line handwriting recognition in the 1960's. Although the recognition objects were limited to English alphanumerics, basic and important ideas for recognition algorithms and system components were proposed in rapid succession. One technique discriminates the shape of strokes by examining the time sequence of direction angles of the pen-tip movement. This is a variant of the pattern matching approach. Another technique used the structure analyzing approach to define a set of shape primitives and represent each character as a primitive-code sequence. Also, fundamental techniques such as preprocessing which includes noise reduction, smoothing, filtering, and size normalization were introduced [3]. These opening studies determined the fundamental direction of subsequent research into on-line handwriting recognition. However, only the early stage of computer simulations was reached and recognition accuracy was not confirmed by prototype testing.

In the 1970's, the intense activity in on-line character recognition ebbed in America, although there existed several attempts to realize automatic signature verification [4]. In Japan, by contrast, research into on-line recognition of handwritten characters started at the end of the 1960's and reached a high level in the 1970's as the most hopeful Japanese text input method. Japanese write with Hiragana, Katakana, Kanji, English alphanumerics, and symbols. Hiragana and Katakana (called Kana) are phonetic alphabets, and each has 46 full-size characters. Eight Kana characters can be written half size and, together with additional markings, indicate subtle phonetic differences. Kanji are ideographic characters of Chinese origin, and the most common Japanese Industry Standard contains 6349 characters. Keyboard input of Kanji is very cumbersome even for professional users. There are two serious problems in on-line Kanji recognition. One is that a large number of characters exist to be recognized. The other is the wide variations in handwritten characters, i.e., stroke number, stroke order, and shape. Researchers began by tackling the first property in that they assume that each Kanji character was accurately written in the block style with the correct stroke number and order. Initially, most research took the structure analyzing approach that recognized primitive strokes or radicals to identify a whole Kanji character. However, at the end of the 1970's recognition accuracy remained quite low because the primitive stroke recognition technique was not sufficiently robust. The hierarchical recognition strategy was perceived to be a failure.

Remarkable advances in software techniques for Kanji character recognition algorithms were made in the 1980's. The structure analyzing approach was enhanced by providing about 100 primitive strokes to permit constituent shape distortion and improve stroke recognition accuracy. However, the stroke number and stroke order variation permission were not successfully realized for lack of an efficient systematic search technique having no combinatorial exhaustion. With regard to the pattern matching approach, considerable efforts were also made to devise a robust stroke matching measure, and permit wider variations in stroke number and stroke order. NTT proposed a series of robust algorithms based on the pattern matching approach and almost completely resolved the previously mentioned problems. The other was the renewed interest in America and Europe for user-interfaces designed around on-line handwriting recognition. An example of this is the handdrawn gestural interface [5], where "gesture" means erase, copy, move, insert, sum a row of numbers, and so on. A significant hardware advance was the combining of input tablets and flat displays to bring input and feedback onto the same surface.

We can now clarify the present status and future problems of on-line handwriting recognition techniques and develop a clear idea of future user friendly interfaces built around on-line handwriting recognition.

For readers who want to know more about the state of the art in on-line handwriting recognition, the exhaustive survey paper [7] is recommended. There is also a survey [8] solely devoted to the on-line recognition of handwritten Japanese characters.

IV. Kanji Character Recognition

Chinese characters or Kanji in Japan can consist of many strokes. The stroke number of a character in the block style ranges from one for the simplest character to over 30 for the most complicated. Also, the correct stroke order in a character is not strictly observed by most people. Moreover, the cursive style is written faster and with more deformed strokes. Therefore, in on-line recognition of Kanji characters we have to overcome two serious problems: a large character set and wide variations in handwriting properties. To resolve these problems, we proposed a series of robust recognition algorithms based on the stroke by stroke matching strategy taken from the pattern matching approach. We adopted the pattern matching approach instead of the structure analyzing approach. The first reason is that it is much easier to recognize a whole character than to recognize its parts or primitives if it is considerably deformed. The second reason is that it is much more reliable to provide a prototype or prototypes for each Kanji character than to devise ad hoc primitive strokes or radicals when flexible adjustment to an increase or decrease in the number of Kanji characters to be recognized is needed. We provided prototypes for all Kanji characters using feature point representation. Here, "prototype" means a template generated by averaging x-y coordinate values of feature points over learning samples for each character. Then, we divided the stroke by stroke matching strategy into the following two steps. The first step is stroke correspondence determination between a template and an input pattern. This is indispensable for realizing stroke number and stroke order free recognition. The second step is intercharacter distance calculation using all templates and the input pattern to yield distortion-tolerant pattern matching.

In the following, we describe the five kinds of recognition algorithms in the order of their proposal. The order corresponds to the progress made in relaxing handwriting constraints with regard to stroke number, stroke order, and permissible shape distortion.

A. Simple Stroke Matching Method

In this subsection, we deal with the problem of on-line recognition of Kanji written in the block style with the correct stroke number and stroke order. Therefore, only the templates that contain the same number of strokes as the input pattern are candidates for matching. Moreover, as the stroke order of the input pattern is correct, we have only to perform stroke by stroke matching in the order of handwriting between the input pattern and the template being considered. As a result, the problem resolves itself into how to define the interstroke distance.

The proposed method [9] approximates each stroke by a fixed number of feature points and represents the stroke by means of x-y coordinate values of these feature points. Next, the interstroke distance is defined as the sum of interpoint Euclidean distances. Let d_i denote the interstroke distance between the ith stroke of the template and the ith stroke of the input pattern. Then, the intercharacter distance

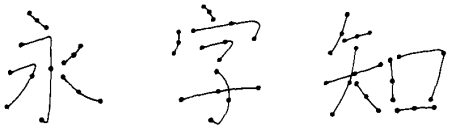


Fig. 1. Feature point representation of Kanji characters.

D is defined as:

$$D = \sum_i d_i. \tag{1}$$

Let N denote the number of strokes common to the template and the input pattern. Then, the category of the template that realizes the minimum intercharacter distance D among all templates with N strokes is the recognition result of the input pattern.

Figure 1 shows examples of Kanji characters in the block style, where the black circles denote the loci of the feature points. From Figure 1, it is easy to see that most Kanji strokes are straight lines. In fact, even if a stroke has bending points, there is usually only one. Only three feature points per stroke (two end points and a middle point) as shown in Figure 1 were found to be necessary to Kanji stroke discrimination. In contrast, six feature points were needed to represent each stroke of Hiragana because Hiragana mainly uses curved strokes. In general, a large number of feature points can represent a curved stroke with high fidelity. However, excessive attention to handwriting distortion on curved portions is counterproductive. Obviously, there exists an optimal number of feature points for each character set considered that maximizes recognition ability.

Recognition tests were made using a mixed set of 881 Kanji characters, 46 Hiragana characters, and English alphanumerics. We obtained the high recognition rates of 99.8% for learning samples and 99.7% for test samples. The character data used in the tests, however, consisted of samples carefully handwritten in the block style with the correct stroke number and stroke order. The strict observance of correct stroke number and stroke order was an excessive burden on users in actual applications. Therefore, the algorithm described in this subsection must be considered as the starting point of subsequent research which attempted to relax the handwriting constraints.

B. Interstroke Distance Matrix Method

In this subsection, we describe a stroke-order free on-line Kanji recognition algorithm, which was the first step in relaxing handwriting constraints. The algorithm ignored information about the stroke order of the input pattern. Namely, a character is considered as not an ordered set of strokes but a nonordered set of strokes. Therefore, the essential problem is to determine appropriate stroke correspondences between the templates and the input pattern without stroke order information. The key idea to resolve the above problem is that the most appropriate stroke correspondence should produce the minimum sum of

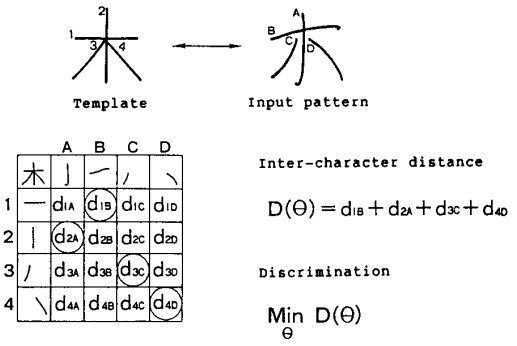


Fig. 2. Interstroke distance matrix (ISDM).

interstroke distances. Based on this key idea, we proposed the interstroke distance matrix method [10].

The proposed method uses the stroke by stroke matching strategy and is a simple but essential extension of the method described in Section IV-A. First, with respect to each stroke in a template, the interstroke distance to all strokes in the input pattern is calculated. Here, let d_{ij} denote the interstroke distance between the ith stroke of the template and the jth stroke of the input pattern, where the stroke numbers of the template and the input pattern are equal to N. Collectively, the $N \times N$ interstroke distance matrix ISDM whose (i, j) element equals d_{ij} can be defined. Next, we search for the minimum value in each row of ISDM. That is, the template stroke to input stroke correspondence that produces the minimum interstroke distance is chosen. This allows complete freedom in stroke order. Last, the intercharacter distance D between the template and the input pattern is calculated as:

$$D = \sum_i \min_j d_{ij} \tag{2}$$

where it is to be noted that the stroke correspondence determined by (2) is not necessarily 1-to-1. However, this causes no serious problem in recognition accuracy when the correct stroke number condition is valid.

Figure 2 shows an example of ISDM between two patterns of

(means "tree") written with different stroke orders.

Tests were made using two kinds of character data sets. The first set was the samples used in Section IV-A, i.e., correct stroke number and correct stroke order. For this character data set the ISDM method achieved much the same recognition rate as that obtained by the simple stroke matching method. The second character data set consisted of new samples of 1851 Kanji characters and 76 Hiragana characters in the block style with the correct stroke number and arbitrary stroke order. The ISDM method achieved the high recognition rate of 99.5% for the second set.

Test results indicate that there are two reasons for the high recognition ability of the ISDM method. One is that the stroke by stroke matching strategy is so powerful that the stroke order information can be ignored. The other reason is that the correct stroke number condition plays a major role in excluding most similarly shaped, but different characters, from the recognition candidates. On the other hand, misrecognition mostly results from shape distortion for an input pattern composed of a small number of strokes. A complicated Kanji character composed of a large number of strokes is rather easy to recognize because of the considerable redundancy in shape. At any rate, if the correct stroke number condition is violated, the stroke by stroke matching strategy is in danger of losing its high recognition ability. Relaxing the stroke number restriction allows significantly more handwriting distortion. In subsequent subsections, we propose and improve stroke number free recognition algorithms based on the pattern matching approach by incorporating structural information.

C. Selective Stroke Linkage Method

In this subsection, we describe an on-line character recognition algorithm that handles the stroke number and stroke order variations common in natural Japanese handwriting formed in the block style.

In order to enhance the stroke by stroke matching strategy by permitting both splitting of a single stroke and concatenation of successive strokes, it is necessary to resolve the following problems:

1) whole-part and distortion-tolerant stroke matching;
2) robust stroke correspondence determination.

To resolve the first problem, we changed the feature point representation of a stroke so that a stroke was approximated by feature points at regular intervals rather than by a fixed number of points. Thus the number of feature points is nearly proportional to stroke length. This allows us to introduce two kinds of interstroke distances. One for whole-part matching calculates the sum of interpoint Euclidean distances in order for the whole of the shorter stroke to be matched with the front part of the longer stroke. The other for distortion-tolerant matching calculates the sum of interpoint Euclidean distances by using dynamic programming (DP) matching.

Next, to resolve the second problem, we proposed the selective stroke linkage method [11]. Let M be the larger stroke number of the input pattern and the template and let N be the smaller stroke number, i.e., $M \geq N$. Also, let PM be a name for the pattern with M strokes and let PN be a name for the pattern with N strokes. If PM is the input pattern, PN is the template, and vice versa. The procedure for determining stroke correspondences is divided into two steps. The first step determines N stroke pairs, i.e., 1-to-1 correspondence between N strokes in PM and N strokes in PN, that minimize the sum of interstroke distances using whole-part stroke matching. This step allows for stroke order variations. Figure 3 shows an example of stroke correspondence determination for two patterns of

(means "character"), that have different stroke numbers and

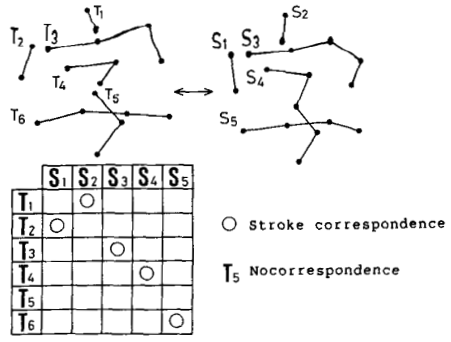


Fig. 3. Stroke correspondence determination.

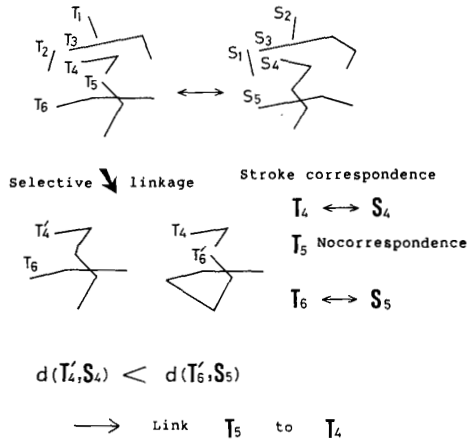


Fig. 4. Selective stroke linkage.

stroke orders. In the second step, unmatched $(M - N)$ strokes in PM are selectively linked to the adjacent stroke in PM that yields the best interstroke matching result. As a result, the stroke numbers of the template and the input pattern are equalized to N. This step allows for stroke number variations. Figure 4 shows an example of selective stroke linkage.

Last, the interstroke distance is recalculated between pairs of corresponding strokes by DP matching. The total sum of N interstroke distances is taken as the final intercharacter distance D. The template that minimizes D is the input pattern recognition result.

A template set with the correct stroke number and order was prepared for the 1851 Kanji characters used daily in Japan. Tests were made using two character data sets. The first set used the same samples used in Section IV-B, i.e., correct stroke number and arbitrary stroke order. The second character data set consisted of new samples of 126 Kanji characters written in the block style and containing stroke number variations, single stroke splitting and successive strokes concatenation, and arbitrary stroke order variations. The second character data set achieved a recognition rate of 98.9%. This means that the selective stroke linkage method can satisfactorily deal with a reasonable range of stroke number variations which often happens in natural Kanji handwriting in the block style. On the other hand, a high recognition rate of 99.8% was obtained for the first character data set. That is, handwritten Kanji characters are never misrecognized if the user uses the correct stroke number. Therefore, misrecognition can be corrected by simply using correct stroke number.

However, this method failed to determine the correct stroke correspondences when pattern balance or stroke shape distortions were significant. This is the limitation of the selective stroke linkage method that assumes the block style handwriting. The next step was to reinforce the recognition algorithm so that even cursive handwriting was acceptable.

D. Stroke Linkage Rule Method

This subsection describes an on-line cursive script recognition algorithm that incorporates structural information about how strokes are most likely to be connected as stroke linkage rules [12] into the pattern matching approach. In cursive handwriting, each subset of strokes is one continuous stroke. The problem is that the subsets vary from time to time, and from person to person.

We introduced a new concept regarding stroke rank, to absorb the above mentioned variety. The essence of stroke rank information is to flexibly represent the role of each stroke, when written in the cursive style, while using templates formed in the block style. We first define that a stable stroke is one that always occurs by itself even in the cursive style. Then each stroke is assigned a stroke rank as follows:

Rank 1: A stable stroke that is always written first within a group of strokes or a character.
Rank 2: All other stable strokes.
Rank 3: All nonstable strokes.

Actually, all strokes in the templates of 1945 Kanji characters were classified into the above three ranks according to where and how they appear in an extensive sample of cursive handwriting. Next, we introduced the following stroke linkage rules to generate only reasonable stroke combinations by utilizing the stroke rank information:

Rule 1: Stroke concatenation occurs according to the correct stroke order.
Rule 2: Strokes of stroke rank 1 or 2 link only to strokes of stroke rank 3.
Rule 3: Strokes of stroke rank 1 always precede successive strokes concatenation.

The stroke linkage rule method is composed of the following three steps. The first step is candidate preselection using the selective stroke linkage method wherein only the ends of a stroke are taken as its feature points. This is based on the property that each stroke endpoint in the cursive style always corresponds to some stroke endpoint in the block style. That is, cursive handwriting has fewer stroke endpoints. The second step is to generate concatenated stroke patterns from each candidate template. Strokes are concatenated to create the same number of strokes as the

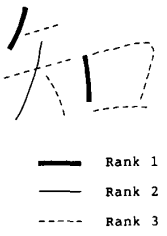


Fig. 5. Cursive pattern generation by stroke linkage rules.

input pattern while conforming to the above mentioned stroke linkage rules. This step allows for variations in the stroke number. Figure 5 shows an example of a concatenated stroke pattern. The correct stroke number is eight for the character

知

(means "know") while the cursive set has only five. The strokes of the template can be ranked and new templates formed according to the stroke linkage rules. The last step calculates intercharacter distances between the input pattern and all the patterns generated from the template. Here, the previous ISDM method based on DP matching is used to permit stroke order variation. Then, the least value is taken as the definitive intercharacter distance D between the input pattern and the template. The candidate template that achieves the minimum D value is determined as the input pattern recognition result.

This method was applied to a cursive character data set of 300 Kanji characters gathered without any instruction or direction and the recognition rate of 95.2% was obtained. Figure 6 shows correctly recognized character samples. There are two misrecognition causes. The first one is the limit of pattern matching based on block-style templates even if stroke concatenation and DP matching are fully utilized. The second one is the violation of stroke linkage rules. However, the ability of stroke correspondence determination is greatly improved. As a matter of fact, when this method was applied to the character data set tested the previous selective stroke linkage method, the recognition rate was improved from 98.9% to 99.4%. From these results, it is concluded that a user friendly Japanese input device, using on-line character recognition, can be achieved that handles even cursive handwriting.

All recognition methods described up to this subsection were devised to improve the following two abilities. The first ability is concerned with stroke correspondence determination, which is indispensable for realizing stroke number and stroke order free recognition. From the viewpoint of combinatorial searching, the stroke linkage rule method

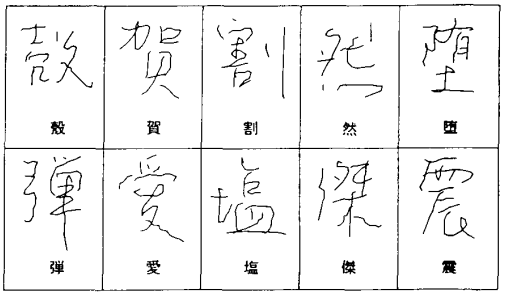


Fig. 6. Examples of correctly recognized patterns.

is a fair working solution. On the other hand, the second ability is how to estimate character shape deformation as an intercharacter distance. This is an inherent problem in the pattern matching approach. However, we cannot tell what measure is best to calculate the intercharacter distance, because shape deformation properties or rules have not yet been developed. In fact, in the present algorithm, the main cause of misrecognition was handwriting distortion that weakens the ability to discriminate between similarly shaped but different characters. In the next subsection, we discuss the problem of shape deformation estimation and present a promising solution.

E. Local Affine Transformation Method

In general, the pattern matching approach addresses three essential problems: how to extract deformation between two patterns, how to describe the extracted deformation, and how to estimate the described deformation.

Extraction of deformation is equal to determining the correspondence of subpatterns within two patterns. This problem is being very actively investigated because it is the most fundamental procedure in recognizing patterns. For on-line character recognition, we divided this problem into two steps as described in the previous subsections. The first step was stroke correspondence determination in order to create stroke number and stroke order flexibility. The second step was based on the feature point correspondence determination of paired strokes by means of a DP matching technique.

On the other hand, research into description and estimation of deformation has not been very successful. Regarding deformation description, the "feature point displacement vector" concept does exist. However, this concept only gives the lowest level of deformation description. For deformation estimation, most methods apply only simple measures of distance to the overall amount of deformation. In fact, we defined the intercharacter distance simply as the overall sum of interpoint Euclidean distances between two patterns. Therefore, one question still remains: how much deformation can be considered small?

In this subsection, we propose for on-line character recognition one robust algorithm for the description and estimation of handwriting deformation. First, as the result of DP matching in the previous method, a set of vectors is

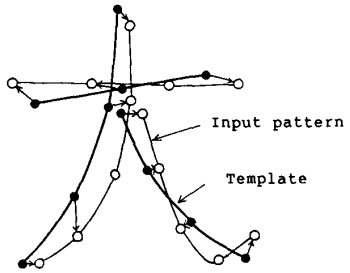


Fig. 7. Deformation vector field (DVF).

derived which links feature points in a template to matched feature points in the input pattern. This set of vectors is termed the deformation vector field (DVF). Figure 7 shows an example of DVF. DVF contains information about global and local correlations between deformation vectors. When the template and the input pattern belong to the same category, DVF might be expected to vary continuously or linearly in each local area. If the above expectation is valid, DVF might be considered as a kind of "local" affine transformation. On the other hand, when the template and the input pattern do not belong to the same category, DVF may vary randomly in each local area and may not be well approximated as a "local" affine transformation. These considerations lead to a new idea: deformation components that are equivalent to a "local" affine transformation should be ignored in deformation estimation. This is the key idea in introducing local affine transformation [13]. Concretely, around each feature point in a template, a local affine transformation (LAT) is defined which approximates neighboring deformation vectors by the least squares data fitting technique. The residue vectors are further approximated with even more localized affine transformations. That is, each deformation vector in a DVF can be expanded into a series of LAT's. Figure 8 shows an example of iterative LAT application for a template and an input pattern of

(means "eternal"). From Figure 8, it can be seen that the overall rotation component of DVF was extracted in the first LAT iteration. Subsequently, higher-order LAT components were extracted. As a result, LAT modified templates monotonically approached the input pattern. An enhanced intercharacter distance is calculated between the input pattern and the deformed template created with low-order LAT components. Last, the template that minimizes the enhanced intercharacter distance value is determined as the input pattern recognition result. With this enhanced intercharacter distance concept, it might be possible to improve recognition ability of the pattern matching approach.

Tests were made on the same samples of the cursive style 300 Kanji characters used in Section IV-D. The recognition rate of the LAT method was 97.2% while the stroke linkage rule method achieved 95.2%.

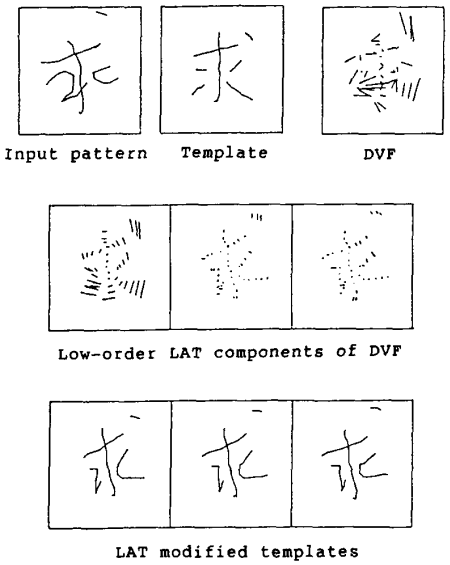


Fig. 8. Example of iterative LAT application.

These results suggest that the concept of local affine transformation can provide a powerful tool for analyzing handwriting and improving the recognition ability of the pattern matching approach. The results also point to a possible definition of "small deformation." Further research is needed to clarify what type of deformation is easy or difficult to describe by means of local affine transformation. By the way, advances in handwriting recognition and our understanding of handwriting go hand in hand. A fundamental step toward this understanding is the gathering and analyzing of statistics related to variation in character shape, slant, stroke number and order, and direction. A careful and intelligent analysis of these statistics would enable us to construct a robust model of handwriting deformation. In this sense, it is critical to establish databases of on-line handwriting. The databases are also useful to assess the value of the many recognition methods.

Finally important is a brief description of the processing time and our computing environment. We previously described the five kinds of recognition algorithms. As the recognition algorithm is sophisticated, it requires further processing time. Moreover, the number of matching templates increases to permit the stroke number and order variations. As a matter of fact, the maximum processing time needed to recognize one character by the local affine transformation method linked with the stroke linkage rule method was 3 min using a 1-MIPS minicomputer. However, the special hardware developed to implement the selective stroke linkage method realized real-time recognition that took less than a half second per single character recognition.

V. Character Sequence Recognition

Recognition methods for already segmented characters were discussed in the previous section. These on-line char-

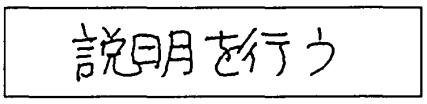


Fig. 9. Example of free-format Japanese writing.

acter recognition methods are useful for the input of characters. However, all these methods impose some restrictions such as writing each character in a box, or indicating the end of each character. Such restrictions limit the application of the recognition systems as well as troubling writers because writing characters in boxes is not common. We usually write characters on blank or ruled paper. The problem of character sequence recognition has been studied in both on-line and off-line situations where the methods employed were based on mostly similar ideas. However, on-line recognition has the great advantage over off-line recognition in that temporal information and stroke information can be fully utilized.

This section first describes an on-line recognition algorithm for free-format handwritten Japanese character sequences, which may contain characters with separated constituents or overlapping characters. Next we show that the same algorithm can be applied to recognize free-format line figure recognition.

A. Free-Format Japanese Sequence Recognition

It is a difficult problem to segment characters if they overlap or have irregular pitch. Of course humans can easily recognize such characters. Here recognition implies character segmentation and character recognition. Humans do this task by combining character segmentation, character recognition, and linguistic information.

Several methods on the automatic segmentation of English characters have been reported [14]. Some of these methods utilize time out information and spatial information such as gaps between characters or words, or character shape information provided by the character recognition process. However, Japanese characters are essentially different from English characters. For example, Japanese characters in general contain many strokes, and quite often consist of more than two separate stroke subgroups, i.e., radicals. Therefore, the segmentation methods of English characters are not directly applicable to Japanese character sequence recognition. Figure 9 shows an example of Japanese free-format handwriting.

The three serious recognition problems due to the nature of Japanese handwriting are:

1. Handwritten characters may contact each other, and their sizes vary considerably. Therefore, spatial information such as character pitch and gaps between characters does not help much in determining character segmentation.
2. About 30% of Japanese characters consist of more than two parts (e.g., left-hand radical and right-hand body). This characteristic makes segmentation difficult.

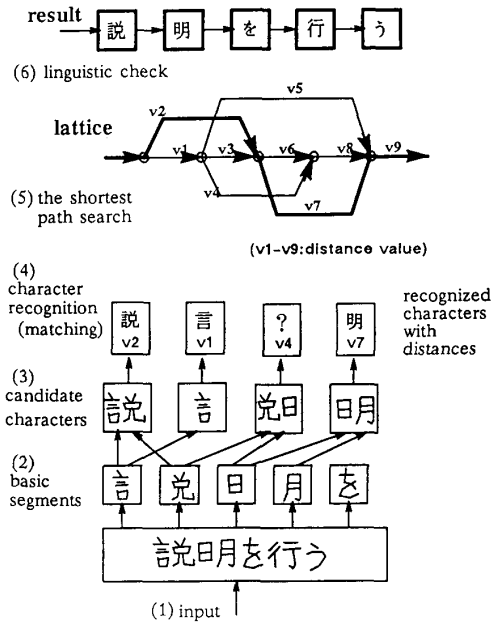


Fig. 10. Block diagram of candidate lattice method.

3. Any part can be an individual character by itself. Such characters account for 13% of the total. They might be mistaken for a sequence of two or more characters, if only character shape scores are used in segmentation and recognition.

This subsection describes the candidate lattice method [15] that automatically segments and recognizes Japanese character sequences. This method realizes on-line recognition of free-format character sequences which are written on blank paper; no indication of segmentation is needed.

The essential idea of the method is to segment and match all writing units that can be characters. The method describes a character sequence as a directional graph representing possible segmentations and character shape scores. Then, by means of DP (dynamic programming) the candidate character sequence that achieves the shortest path with the minimum of cumulative shape recognition scores under linguistic constraints is selected as the segmentation and recognition result.

The candidate lattice method consists of six steps as shown in Figure 10: 1) handwriting data input, 2) separation into basic segments, 3) generation of candidate characters, 4) matching of the candidate characters, 5) candidate lattice formation and shortest path search, and 6) linguistic processing.

The first step captures data handwritten on an electronic tablet with a stylus pen. The character set is composed of the 1945 basic Kanji characters and the 46 Hiragana characters. A preprocessing operation detects the direction of character sequences by determining if the average direction of pen movement is top-to-bottom or left-to-right. Thus the method can deal with both vertical and horizontal

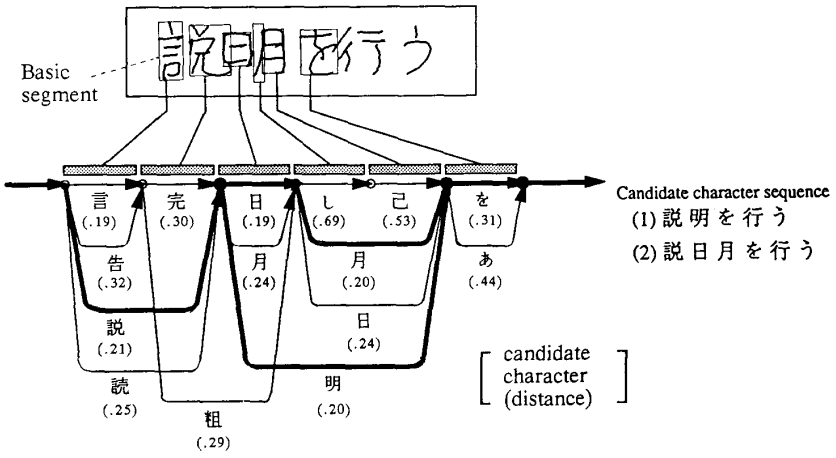


Fig. 11. Candidate lattice for character sequence.

writing, although only horizontal writing is treated in this paper.

The second step is to divide the input stroke sequences into basic segments. A basic segment is introduced that corresponds to a complete character or a character component. The strokes do not usually continue from one character to another in Japanese writing. Using the projection of all strokes in a sequence or line on the x axis, parts that exceed a profile overlapping threshold (Th) are extracted as segments. Th is a parameter denoting the ratio of profile overlapping to the average height of the character sequence. The smaller Th is, the more strokes a segment can contain. This leads to possible segmentation errors, since a character segment might extend to other characters. On the other hand, an excessive Th leads to oversegmentation, which greatly increases subsequent calculation operations. Incidentally, we decided the optimal value for Th to be 0.15 based on the investigation of the actual character sequences written by seven people. The value Th specifies the tolerable limit of intercharacter overlapping. Namely, $Th = 0.15$ means that up to 15% overlapping of neighboring characters can be permitted.

The third step generates candidate characters by combining one or more basic segments according to the following conditions.

1. The number of strokes forming a candidate should be less than the maximum stroke number.
2. The ratio of candidate character width to the average height of the character sequence should be less than a threshold.

These two conditions are sufficient to extract probable candidate characters (see Figure 10(3)).

The fourth step is character matching. The distances between each candidate character and all templates in the dictionary are calculated, as follows.

1. Approximate each candidate character stroke with equidistant sampling points on the stroke.
2. Calculate the distances (d_{ij}) between the ith stroke of the candidate character and the jth stroke of a template for all possible combinations of i and j. We use DP matching based on interpoint Euclidean distances.
3. Calculate the inter-character distance D by

$$D = \sum_i \min_j (d_{ij})$$

as described in Section IV.

4. Choose the first K templates that have the shortest intercharacter distances. Then, the names and distances of these characters are stored for use in the next step. Here, the value of K was set to 3, since over 99% of the correct choices were included in the top three candidates according to a preliminary investigation.

The fifth step describes the result of segmentation and candidate character recognition in terms of a two-terminal weighted directional graph (candidate lattice). Each node of the graph is defined by a boundary between basic segments, a branch by a candidate character, and the weight by calculated distance, as shown in Figure 11. Here, the shortest path in the candidate lattice is located by DP. The shortest path is the series of branches, which minimizes the following objective function, S.

$$S = \sum (\text{distance}) \times (\text{no. of segments}). \qquad (3)$$

The sequence of candidate characters that corresponds to the determined path becomes the tentative recognition result.

The last step checks the obtained sequence of characters grammatically using morphological analysis, a dictionary containing about 8000 words and 130 rules of basic Japanese grammar (e.g., "adjective + noun phrase can form another noun phrase"). When a contradiction to the rules is found, the tentative character sequence is rejected and the next shortest path in the lattice is selected. The above process is performed iteratively until

a grammatically correct sequence is selected as the final recognition result. Figure 11 shows the case where two similar candidates were extracted. Both candidates are valid from the standpoint of character recognition. However, the lower one is grammatically incorrect. By this method, the upper one, which is the correct answer, is selected.

Recognition tests were made using 105 sentences written by seven people. The sentences were excerpts from technical reports. Instructions given to the writers were: 1) to write each character in the block style with the correct stroke number and the free stroke order, and 2) to write characters discretely along a nearly horizontal line. The method achieved a correct segmentation rate of 98.8% and an overall recognition rate of 98.7%. It is to be noted that a low segmentation rate of 68.5% was obtained if only spatial gap information was used. The main causes of misrecognition were excessive deformation and the limited grammar rules. The second cause can be removed by reinforcing the linguistic dictionary.

Japanese character sequence recognition by this method requires a vast amount of calculation. Therefore, we developed the special hardware with 16 processors linked in parallel. Through experiments using the hardware, the processing time was found to be approximately proportional to the length of a character sequence while the time needed for a single character recognition was about one second.

We can claim several advantages for the newly introduced candidate lattice method. One advantage is that the candidate lattice method can be successfully applied to various writings that contain overlapping characters with different sizes or pitches, because the method does not use pitch or size information. Another advantage is that the use of basic segments instead of individual strokes realizes real-time segmentation and recognition because only reasonable candidate characters are generated. For example, in Figure 9, only 51 candidates were generated using basic segments, whereas the much larger number of 419 candidates was generated with individual strokes. This means that the efficiency of pattern matching and shortest path search is greatly improved.

B. Line Figure Recognition

Line figure recognition has many applications in areas such as automatic fair copying of hand-written documents containing flowcharts or block diagrams, and automatic programming from hand-sketched flowchart data. By the way, line figure recognition methods have similar problems to those of character sequence recognition. The conventional methods impose several restrictions on figure sketching: 1) the user must indicate segmentations between symbols [16], and 2) each symbol must be drawn using the predetermined number and order of strokes. Clearly, these restrictions result in a rather clumsy figure sketching procedure.

This subsection applies the candidate lattice method to the on-line recognition of hand-sketched line figures in flowcharts of block diagrams [17]. The method requires 1) no indication of segmentations, and 2) no restrictions on the stroke sequence (number and order of strokes). Figure 12

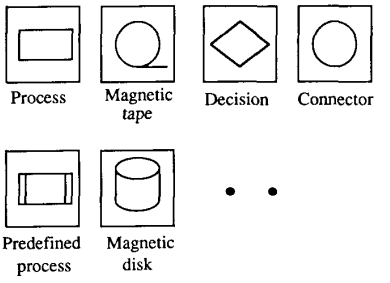


Fig. 12. Flowchart symbols (JIS C6270).

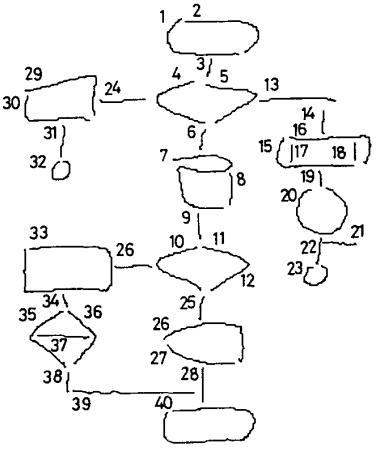


Fig. 13. Example of flowchart.

shows examples of symbols used in our experiment. These are symbols of flowcharts or block diagrams defined as the Japanese Industrial Standard (JIS) C6270. An example of a handwritten flowchart is shown in Figure 13.

The concrete procedure of segmentation and recognition is as follows. First, all the subfigures that may possibly be symbols are extracted from an input sketch as candidate figures based on the stroke sequence independent matching method. This method represents the structures of symbols in terms of a directed graph. The arbitrariness of the number and order of strokes is handled by searching through paths of the graph. Next, the distances between extracted candidate figures and symbol templates are calculated through interstroke DP matching. For all the extracted candidate figures, names of matched symbols and calculated distances are stored in a candidate lattice. Figure 14 shows an example of a candidate lattice. Then, by searching this candidate lattice for the optimum candidate sequence, i.e., that which minimizes the total sum of distances along the sequence, we obtain a tentative segmentation and recognition result. Last, illegal connective relations among symbols in the tentative segmentation and recognition result are checked. For example, the following connections are obviously inconsistent with flowchart structures: 1) a "terminal" is located at the branch of a process flow, 2) a "linkage line segment" does not connect two symbols, and so on. If any

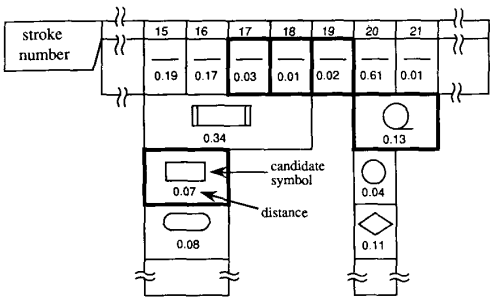


Fig. 14. Candidate lattice for flowchart.

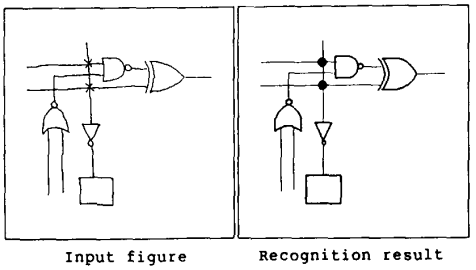


Fig. 15. Recognition of logic circuit diagram.

of the relations violate the connection rules, the responsible symbols are removed from the lattice and replaced with other symbols according to the connection rules. Then, the searching-for-the-optimum sequence step is repeated for the renewed lattice until the final consistent segmentation and recognition result is obtained.

Tests were made using 100 figures of hand-drawn flowcharts. The writers were instructed to write symbols one by one in a series of strokes. There were no restrictions on stroke shapes or stroke order within a symbol. With regard to the stroke number variation, only concatenation of strokes was permitted. The obtained recognition rate of 96.1% was satisfactorily high. On the other hand, a rather low recognition rate of 89.5% was obtained when no symbol connection rules were used. We have shown that: 1) stroke sequence independent matching is effective in handling input figures that have an arbitrary number and order of strokes, 2) higher accurate recognition can be achieved by adding a correction process that uses connection rules as a postprocess.

As is easily seen, the candidate lattice method can deal with the problem of segmentation and recognition of any line figures composed of symbols and linkage line segments by preparing the symbol shape dictionary and appropriate connection rules. As another example, we applied the method to logic circuit diagram recognition. A symbol shape dictionary was prepared for a set of 26 symbols ("AND," "OR," "NAND," "NOR," "EXCLUSIVE-OR," "NOT," and their equivalent rotated (90, 180, 270) symbols and, "CONNECT" and "BLOCK"). A total number of seven connection rules were registered in the system, e.g., "if the number of "linkage line segments" connected to a "NOT" is greater than two, the symbol is removed from the lattice." Tests were made on 40 logic circuit diagrams. We obtained a high recognition rate of 93.2%. Here, misrecognition causes were significant distortion of figure shapes and unexpected splitting into strokes which should be written in one stroke. Figure 15 shows an example of an input figure and its recognition result.

From these results, we can conclude that the candidate lattice method is highly general and robust. This was due to the following three key ideas: 1) the shortest path search in a candidate lattice achieves symbol segmentation and recognition simultaneously, 2) symbol connection rules effectively realize top-down verification of the symbol segmentation and recognition result, and 3) the graph search technique achieves stroke order free matching between an input figure and symbol templates. As a result, the system greatly lessened the user's load by requiring no indication of symbol segmentation and no restriction on stroke order. It is very promising that the candidate lattice method can provide a powerful tool to realize a user friendly interface using on-line recognition of free-format hand-sketched line figures.

VI. Discussion

Among Japanese input methods, Kana-Kanji, i.e., phoneme-ideogram, automatic translation is the current favorite. However, nonprofessional users cannot input more than 30 Kanji characters per minute. Also, the user cannot devote his attention solely to content creation. Thus the stress of this method is being socially recognized.

On-line recognition of handwritten characters is by far more reasonable than keyboard entry and provides the most natural way to input Japanese text. It is easy to learn, easy to use, and efficient as well. In this section, we describe prototypes developed by NTT using on-line recognition of handwritten characters and gestures with the aim of realizing a user friendly interface. Also, anticipated advances in both hardware and software are discussed.

A. User Friendly Interface and Prototypes

Dozens of on-line handwriting recognizers are commercially available in America, Europe, and Japan [7]. Especially in Japan, there have been over twenty commercial systems developed to recognize Kanji, Kana, and English alphanumerics. Most systems use opaque tablets, while the newer systems use an integrated tablet display. Of interest are prototype systems that recognize characters, hand-drawn symbols, and also "gestures." The term gesture here refers to handmarkings, such as circles, brackets, and arrows, that function to indicate scope and commands. Typical is the set of editing gestures. These systems clearly intend that on-line handwriting recognition be used for heavily interactive applications, such as CAI, CAD, and user friendly communication tools via telecommunications networks. Also, down-sizing of recognizers has been intensively pursued. As described in the following, NTT has

Table 1 Specifications of the Japanese Text Editor Developed by NTT

Alphabet size	2400 (Kanji, Kana, alphanumerics, special symbols) 300 (any characters or symbols for personal use)
Segmentation	Boxes 8 mm ~ 12 mm
Writing restrictions	Stroke order free, stroke number partially free
Editing functions	Insertion, movement, deletion, copy
Recognition time	Less than 1.5 s/character
Output	RS232C/IEEE488 300-9600BPS (RS232C)
Size/weight	410 (W) mm×370 (D) mm×60 (H) mm/5 kg
Power	10 W

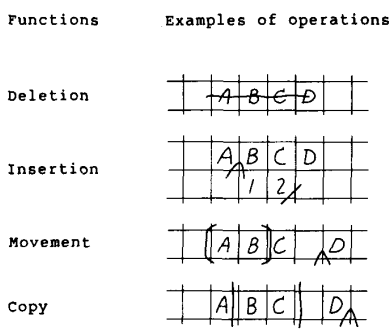


Fig. 16. Examples of editing operations by handwriting gestures.

continually taken the lead in the above mentioned trend by developing prototype systems based on leading-edge recognition algorithms.

NTT developed and improved a series of prototype systems aiming at a user friendly Japanese text editor. The first system [10], developed in 1982, was an experimental Japanese word processor with both sight-typing and handwriting input methods. Sight-typing is where the user must search for each character among thousands of keys. With respect to handwriting input, this system recognized a set of Kanji and Hiragana characters by using special hardware. Thus input of Katakana, English alphanumerics, and symbols together with editing commands was made by cumbersome sight-typing.

The second system [18] was developed in 1984 and refined in 1986 to realize a pure handwriting input Japanese text editor. Specifications of the system are listed in Table 1. In particular, a set of handdrawn editing gestures, e.g., delete, insert, move, copy, were included to perform required actions as shown in Figure 16. Recognition hardware was constructed on one printed circuit board using 16-bit microprocessors connected in parallel. Also, with regard to recognition software, we devised calculation time reduction methods that optimized the matching order of strokes of each template and interrupted the matching process when

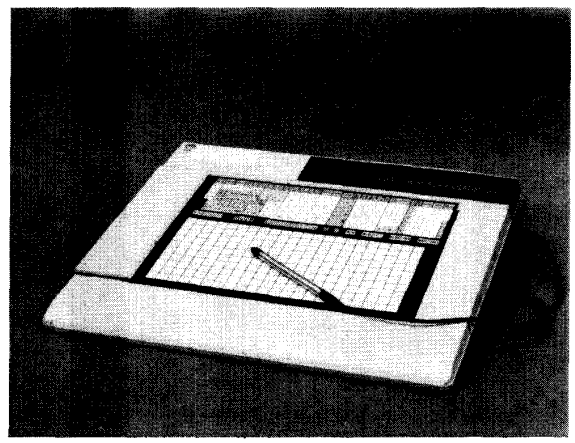


Fig. 17. Top view of the Japanese Text Editor.

the accumulated sum of interstroke distances exceeded a threshold value. Moreover, a user could easily register additional templates in his own handwriting style. Figure 17 shows the top view of the system which was very compact in size and shape. However, the editing result of each text page had to be checked on the display of a personal computer.

Then, a third system that used an integrated tablet display was developed in 1984 and refined in 1988. The first version [19] used a transparent electromagnetic coupling tablet over an electro-luminescent display. Other specifications of the system were the same as those of the first system. Comparative studies on user friendliness of this system were made for three input methods: sight-typing, handwriting input using the opaque tablet, and handwriting input using the integrated tablet display. Handwriting input using the integrated tablet display was found to achieve the maximum input speed and impose the least mental and physical stress on the user. The second version [20] used a transparent electromagnetic coupling tablet over an LCD screen. The size of which was 192 × 120 mm with a resolution of 3.3 dots/mm. This system could recognize handdrawn line figures in a free-format. Other specifications of this system matched those of the second system. The immediate feedback to the user was highly appreciated. However, from the user interface point of view, there still remain several problems. With regard to the display, parallax and insufficient resolution interfered with natural handwriting. Also, these prototype systems adopted simplified recognition methods that suppressed the computational complexity to realize real-time recognition. Therefore, because of the available computer power, the use of more powerful algorithms that could recognize cursive handwriting was abandoned.

B. Anticipated Advances

Future developments in hardware will provide unified tablet/display systems jointly manufactured by interspersing tablet and display elements. This should essentially

eliminate the parallax present with the current technology. Large-sized units with high resolution should also become available. Advances in computers, such as parallel processors and VLSI architectures, will enable the use of more sophisticated algorithms.

In the area of software development, enhancing the flexibility of recognition algorithms is very important. Writing on a tablet without constraints, similar to writing with pen on paper, is the final goal. Future research topics are as follows: cursive handwriting recognition, segmentation of characters, line figures, and gestures without constraints, personalization of feature extraction/selection and decision rules, and robust linguistic checks in postprocessing. Gathering statistics about handwriting properties such as character shape, slant, and stroke variation is indispensable [21].

A clear vision of the user interfaces possible with on-line handwriting recognition is necessary. Using a single input device, the stylus, for entering data and commands on an integrated tablet display, has tremendous potential to realize productivity gains over the conventional alternatives for a number of applications. However, on-line handwriting input systems are not suited to the input of large amounts of data in short periods of time. Continuous and free input of a small amount of data in a long period of time is preferable with this system. Therefore, a dynamic medium the size of a notebook is very promising [22]. User friendly devices that mimic ordinary pens, erasers, and paper may become a personal necessity in the future.

VII. Conclusion

One major advantage of on-line handwriting recognition is its user friendliness. This is the same advantage that pencil and paper enjoys over keyboard entry. Especially, for text input of large-alphabet languages, like Japanese or Chinese, a user has only to write naturally with a stylus pen on a tablet. Another major advantage of on-line handwriting recognition is immediate feedback. The recent interest has been in the use of gestural interfaces to highly interactive, visually oriented applications. Tablets are a powerful tool for the input of both writing and graphics.

This paper described the intense research at NTT aimed at on-line cursive script recognition of Kanji characters. The enhanced stroke by stroke matching strategy proved very successful in overcoming wide variations in stroke number, stroke order, and shape distortions. Also, writing box free recognition of character sequences was realized by the powerful algorithm that incorporated segmentation, recognition, and linguistic information. Moreover, we described Japanese text editors developed by NTT using on-line handwriting recognition and discussed their user friendliness.

Future hardware developments in tablet and display technology will extend the fascinating vision of interfaces constructed with on-line handwriting recognition. Also, advances in computers will enable the use of more sophisticated recognition algorithms for natural free-format writing, including cursive script. Finally, future man-machine interfaces might integrate a variety of input/output technologies. For example, we might interact with machines by using combinations of voice, handwriting, body movements, and facial expressions. On-line recognition technology will certainly play a major role in handwriting interaction. Here, handwriting means characters, shorthand, line figures, and many kinds of gestures.

References

[1] T. L. Dimond, "Devices for reading handwritten characters," in *Proc. Eastern Joint Comput. Conf.*, Dec. 1957, pp. 232–237.

[2] M. R. Davis and T. O. Ellis, "The Rand tablet: A man-machine graphical communication device," in *Proc. FJCC,* 1964, pp. 325–331.

[3] G. F. Groner, "Real-time recognition of handprinted test," in *Proc. FJCC,* 1966, pp. 591–601.

[4] R. Plamondon and G. Lorette, "Automatic signature verification and writer identification—The state of the art," *Pattern Recognition,* vol. 22, pp. 107–131, 1989.

[5] C. G. Wolf and P. Morrel-Samuels, "The use of hand-drawn gestures for text-editing," *Int. J. Man-Machine Studies,* 1987.

[6] J. R. Ward and B. Blesser, "Interactive recognition of handprinted characters for computer input," *IEEE Comput. Graphics Appl.,* vol. 5, pp. 24–37, Sept. 1985.

[7] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The state of the art in on-line handwriting recognition," *IEEE Trans. Patt. Anal. Mach. Intell.,* vol. 12, pp. 787–808, 1990.

[8] M. Nakagawa, "Non-keyboard input of Japanese text—on-line recognition of handwritten characters as the most hopeful approach," *Inform. Proc.,* vol. 13, pp. 15–34, 1990.

[9] K. Odaka, H. Arakawa, and I. Masuda, "On-line recognition of handwritten characters by approximating each stroke with several points," *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-12, pp. 898–903, 1982.

[10] K. Odaka, T. Wakahara, and I. Masuda,"Stroke-order-independent on-line character recognition algorithm and its application," *Rev. Elec. Commun. Lab.,* vol. 34, pp. 79–85, 1986.

[11] T. Wakahara and M. Umeda, "Stroke-number and stroke-order free on-line character recognition by selective stroke linkage method," in *Proc. 4th Int. Conf. Text Processing,* Oct. 1983, pp. 157–162.

[12] T. Wakahara, "On-line cursive script recognition using stroke linkage rules," in *Proc. 7th Int. Conf. Pattern Recognition,* July 1984, pp. 1065–1068.

[13] T. Wakahara, "On-line cursive script recognition using local affine transformation," in *Proc. 9th Int. Conf. Pattern Recognition,* Nov. 1988, pp. 1133–1137.

[14] T. Fujisaki, T. E. Chefalas, J. Kim, and C. C. Tappert, "Online recognizer for runon hand-printed characters," in *Proc. 10th Int. Conf. Pattern Recognition,* Jun. 1990, pp. 450–454.

[15] H. Murase, "Online recognition of free-format Japanese handwritings," in *Proc. 9th Int. Conf. Pattern Recognition,* Nov. 1988, pp. 1143–1147.

[16] W. C. Lin and J. H. Pun, "Machine recognition and plotting of hand-sketched line figures," *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-8, pp. 52–57, 1978.

[17] H. Murase and T. Wakahara, "Online hand-sketched figure recognition," *Pattern Recognition,* vol. 19, pp. 147–160, 1986.

[18] T. Sakai, K. Odaka, and T. Toida, "Several approaches to development of on-line handwritten character input equipment," in *Proc. 7th Int. Conf. Pattern Recognition,* July 1984, pp. 1052–1054.

[19] T. Wakahara, K. Odaka, and M. Umeda, "Electro-luminescent display superimposed by transparent electromagnetic coupling tablet and its application to script input Japanese word processor," (in Japanese), *Trans. Inst. Electron. Commun. Eng. Japan,* vol. J67-D, pp. 981–988. Sept. 1984.

[20] H. Kojima and T. Toida, "On-line hand-drawn line-figure recognition and its application," in *Proc. 9th Int. Conf. Pattern Recognition,* Nov. 1988, pp. 1138–1142.

[21] Y. Kimura and S. Miyahara, "The properties of characters on different writing restrictions," (in Japanese), Tech. Rep. IECEJ, vol. PRU87-39, pp. 19–26, July 1987.

[22] B. W. Mel, S. M. Omohundro, A. D. Robinson, S. S. Skiena, K. H. Thearling, L. T. Young, and S. Wolfram, "Tablet: Personal computer in the year 2000," *Commun. Ass. Comput. Mach.*, pp. 639–646, June 1988.

Toru Wakahara (Member, IEEE) was born in Gifu, Japan, on Jan. 30, 1952. He received the B.E. and M.E. degrees in applied physics and the Ph.D. degree in electrical engineering from the University of Tokyo, Tokyo, Japan, in 1975, 1977, and 1986, respectively.

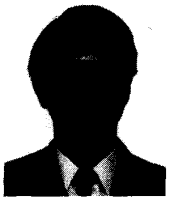
Since 1977 he has been with Nippon Telegragh and Telephone Corporation (NTT) where he is engaged in research on character recognition and text processing. He is presently a Senior Research Engineer, Supervisor, in NTT Human Interface Laboratories.

Dr. Wakahara is a member of the Institute of Electronics, Information, and Communication Engineers of Japan and the Audio-Visual Information Research Group of Japan.

Hiroshi Murase (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in electrical engineering from the University of Nagoya, Japan, in 1978, 1980, and 1987, respectively.

He has been engaged in pattern recognition and character recognition research at Nippon Telegraph and Telephone Corporation (NTT) since 1980. He is presently a Senior Research Scientist in NTT Basic Research Laboratories in Tokyo. His current research interests lie in the areas of computer vision, pattern recognition and human visual perception.

Dr. Murase is a Member of the Institute of Electronics, Information, and Communication Engineers of Japan and the Audio-Visual Information Research Group of Japan.



Kazumi Odaka (Member, IEEE) received the B.E. degree in electronics engineering from Kogakuin University, Japan in 1972, the M.E. degree from Chiba University, Japan in 1975, and the Ph.D degree from Keio University, Japan, in 1985.

He was engaged in research and development in an on-line handwritten character recognition in the Electrical Communication Laboratories of Nippon Telegraph and Telephone Corporation (NTT) from 1975 to 1986. He has been teaching at Chiba Institute of Technology. Also, he is currently working at NTT Basic Research Laboratories and is a Senior Research Scientist. His current research interests are visual information processing mechanisms in man. He is engaged in research on a brain and is studying visual evoked potentials and magnetic fields from the human brain.