ISE-TR-80-~~3~~17

ON-LINE RECOGNITION OF HAND-WRITTEN CHARACTERS UTILIZING POSITIONAL AND STROKE VECTOR SEQUENCES

by

Katsuo Ikeda
Takashi Yamamura
Yasumasa Mitamura
Shiokazu Fujiwara
Yoshiharu Tominaga
and
Takeshi Kiyono

July 29, 1980

INSTITUTE
OF
INFORMATION SCIENCES AND ELECTRONICS

UNIVERSITY OF TSUKUBA

ISE-TR-80-3

On-line Recognition of Hand-written Characters Utilizing Positional and Stroke Vector Sequences

Katsuo Ikeda
Univ. of Tsukuba
Science Information Processing Center
Sakura, Ibaraki, Japan

Takashi Yamamura
Richo Co.

Yasumasa Mitamura
Nichiden Software Co.

Shiokazu Fujiwara
Nippon Telegraph and Telephone Public Co.

Yoshiharu Tominaga
Mitsubishi Electric Co.

Takeshi Kiyono
Osaka Electro-communication Univ.

July 29, 1980

Abstract

An on-line recognition method for hand-written characters utilizing stroke vector sequences and a positional vector sequence has been developed. The number of target characters is about 2000, and fairly good recognition scores have been attained. Our scheme uses the number of strokes as the primary parameter. We employ three types of recognition strategy depending on the number of strokes. The General Stroke Vector Sequence method devised to analyze the shape can represent both skeleton and local characteristics by small amount of information; and the restricted dynamic programming method is effective to determine the shape of a stroke. The similarity of two shapes and the complexity of a stroke have been introduced to reduce the dictionary size and the processing time, respectively.

On-line Recognition of Hand-written Characters Utilizing Positional and Stroke Vector Sequences

Katsuo Ikeda[+], Takashi Yamamura[++], Yasumasa Mitamura[+++], Shiokazu Fujiwara[‡], Yoshiharu Tominaga[§] and Takeshi Kiyono[‖]

[+] Univ. of Tsukuba, Science Information Processing Center, Sakura, Ibaraki, Japan

[++] Richo Co.

[+++] Nichiden Software Co.

[‡] Nippon Telegraph and Telephone Public Co.

[§] Mitsubishi Electric Co.

[‖] Osaka Electro-communication Univ.

Keywords:

Recognition of hand-written characters

Stroke vector sequence

Positional vector sequence

Stroke shape coding and analysis

Restricted dynamic programming

General Stroke Vector Sequence method

Similarity and complexity of a stroke

Skeleton and local characteristics

Summary

An on-line recognition method for hand-written characters utilizing stroke vector sequences and a positional vector sequence has been developed. The number of target characters is about 2000 which include almost all the characters - alphabet, numerals, KATAKANA, HIRAGANA and KANJI (Chinese characters) - usually used in Japan, and fairly good recognition scores of 90 to 98 percent have been attained.

Our scheme uses the number of strokes for the primary selection. We employ three types of recognition strategy depending on the number of strokes; and this is reasonable because the shapes of strokes in characters with small number of strokes are complex, while those in characters with many strokes are fairly simple. The General Stroke Vector Sequence (GSVS) method devised to analyze the shape of a stroke can represent both skeleton and local characteristics by small amount of information; and the restricted dynamic programming method has been found effective to determine the shape of a stroke, removing individual differences which come from the hand-writing process and reducing the size of the shape dictionary. The similarity of two shapes, which is defined as the distance between two points associated with them, also works well to reduce the size of the character dictionary, and the complexity of a stroke, which is defined as the total rotational angle of a stroke, is a useful index for reducing the processing time.

1. Introduction

Processing of Japanese text by computers is strongly required in these days. We are daily using more than 2000 characters in more than four types, such as KATAKANA, HIRAGANA, KANJI (Chinese characters), alphanumeric characters, etc., in Japan. The means of entering Japanese text into a computer is extremely inconvenient and this fact prevents popularization and development of the processing of Japanese text by computers. The main method that has been used so far for entering Japanese text into a computer is to use the full keyboard of a KAN-TELE typewriter. "Rainputto" which utilizes a 48-key typewriter and encodes a KANJI character by two key typings is expected to be useful[10]. However, a professional training will be needed for the operation of devices such as Japanese typewriter. This is the main reason why the development and practical use of on-line recognition of hand-written characters is desired in Japan.

Several research works have been presented on the recognition of hand-written characters; however, many works are dependent on the type of characters[1-7]. Recognition of hand-written characters of daily use in Japan is entirely different from that of alphanumeric characters especially in the following two points. First, the number of characters is more than 30 times as many. Second, the variety of stroke shapes and their combination are so many that the construction of a unified method would be difficult or inefficient. Our method selects different recognition procedures depending on the number of strokes in a character. Each recognition procedure analyzes the shapes of strokes or the positional relations among strokes or both, and then, if necessary, the detailed features. We have devised a general method to analyze and to determine the shape of a stroke[8].

A character is composed of several strokes of specific shapes. Generally speaking, the shapes of strokes in characters with a small number of strokes are more complex and hold more information than the ones in characters of many strokes, in which case they are short and simple. That is, the former characteristics is mainly observed in the shapes of strokes while the later characteristics is observed in the positional relations among strokes. For example, in case of seven-stroke characters, about eighty percent of the whole strokes are classified into only three types of simple shapes; and this trend is more evident in characters which have more strokes (see Figure 1.1).

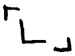
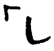
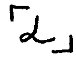
There are two important requirements as to the extraction of characteristics in hand-written characters; the first is how to suppress the individual differences of characters, and the second is how to preserve the detailed features of strokes since they may play important roles in the recognition of characters. These two requirements actually contradict each other. The individual differences mainly appear in the shapes of strokes, while the positional relations among strokes are rather stable. A stroke may be considered as being composed of a stable skeleton pattern superposed by local variations. Thus, a stepwise method which first analyzes the skeleton shape, and then the details, if necessary, will be required.

Characters can be classified into three major classes by the number of strokes in a character as: one-stroke characters which are recognized solely by the shape, two-to-ten-stroke characters recognized by both the positional relation and the shape, and more-than-ten-stroke characters solely recognized by the positional relation.

The summary of our recognition method with emphasis on the important points follows:

A character is entered into a computer by a sequence of X-Y coordinates on a data tablet. This sequence is filtered and smoothed, and then converted into a stroke vector sequence[3]. Positional relations between two successive strokes are calculated and represented by a sequence of positional vectors.

The shape of a stroke is encoded by sampling eleven vectors from the stroke vector sequence, and the type of the shape is determined by consulting the shape dictionary by a restricted dynamic programming method. We call this method General Stroke Vector Sequence (GSVS) method. This method can represent the shape of a stroke efficiently with a small amount of information.

Often the same characters may be written in different patterns. In order to identify this situation, not only the individual shapes of strokes but also the combination of them need be examined. To solve this problem, we could add more entries to the character dictionary; however, it would also increase processing time. Three shapes「」,「」 and「」 in Figure 1.2, for example, are as considered as the same. This sort of situation has motivated us to introduce the concept of similarity between two given shapes.

We have also introduced the complexity of a stroke, which is also useful to reduce the processing time by confining the range of entries in the shape dictionary to be searched.

2. Strategy of character recognition

Since the number of target characters is greater than 2000, it is rather difficult or inefficient to construct a single recognition procedure. We employ five types of recognition procedures depending on the number of strokes. The subsequent sections explain the outline of each type of procedure.

2.1 Recognition of one-stroke characters

Recognition of one-stroke characters is nothing but the determination of the shape of a stroke. The detailed algorithm will be given in Section 3.

2.2 Recognition of two-stroke characters

Recognition of two-stroke characters employs both matching of the shape relations and the positional relations. Analysis of the shape relations and of the positional relations will be discussed in Section 4 and 5, respectively. In the recognition of two-stroke characters, the shape analysis is more important than the analysis of the positional relations; thus, the primary selection is effected by shape comparison. A threshold value for the decision of the stroke similarity is given for each stroke in the character dictionary.

2.3 Recognition of three-to-six-stroke characters

Recognition of three-to-six-stroke characters also employs both matching of the shape relations and the positional relations. However, selection by the positional relations is first performed, then shape matching is executed. The dictionary entries include both shapes and positional vectors. The threshold values for the shape relation matching and the positional relation matching are fixed.

2.4 Recognition of seven-to-ten-stroke characters

The positional relations are first checked. If the fixed threshold value for positional distance, which is explained in Section 5, is employed, the correct determination is rather difficult. Thus, the primary selection is performed by adjusting the threshold value dynamically. If the value of the positional distance is smaller than the given threshold value, the recognition step is completed. Otherwise, the shape analysis explained in Section 3 and matching explained in Section 4 is performed. The value of shape distance is often large, although the individual deviation of the shape similarity is small. This fact shows the usefulness of the concept of "similarity". The number of characters are also large, ranging from 130 to 190.

2.5 Recognition of more-than-ten-stroke characters

Recognition algorithm is rather simple and the number of characters with more than ten strokes is small. Shape analysis may produce many cases that the input pattern is to be rejected due to the presence of too many parameters. The shape of strokes is rather simple and holds small information. Thus, only the positional relations are matched. A fixed value of threshold for the positional distance will do.

3. Shape analysis of a stroke

We have devised an effective and powerful method to classify the shape of a stroke which is one of the most important features of this paper. The shape of a stroke is encoded by selecting eleven vectors from a stroke vector sequence which is a sequence of piecewise vectors along the stroke. The shape distance is defined and evaluated to decide the shape of a stroke. In addition, we have introduced the complexity of a stroke to aid efficient shape analysis. The following sections explain shape encoding and analysis.

3.1 On-line input of a stroke

A stroke is represented by a sequence of X-Y coordinates on the writing surface of a data tablet.

It is necessary to remove chattering effects of the up-down sensing contact in the pen and to pick up data which is independent of the pen speed. The chattering of the pen affects greatly the detection of the start and end points of each stroke. We employ a scheme, shown in Figure 3.1, which simulates the state transitions of the virtual pen with "hysteresis loops", by introducing surplus transient states to represent ambiguous movement of the pen at the start and end of each stroke. We employ a spatial filter shown in Figure 3.2 to make data sampling independent of the pen speed as well as to reject noisy data. A stroke just obtained is not necessarily smooth enough for the later processing; thus, a simple smoothing is performed.

3.2 Stroke vector sequence

In order to encode a stroke, the sequence of X-Y coordinates P_0, P_1, P_2, ... , P_n is transformed first to a stroke vector sequence V_i, V_{i+1}, ... , V_{n-i}, where V_k is a short vector starting at point P_{k-i} and ending at P_{k+i}. We have chosen the value i = 2. If the value of i is too small, the effect of smoothing is not enough; on the other hand, if it is too big, there is danger to look over some local characteristics such as a corner or a small loop. In our experiments, the number of points, n, in a stroke ranged 50 to 200, and empirically, five to ten percent of stroke length will be enough to detect all important characteristics of the shape of a stroke. A vector is coded in one of twenty-four directions as shown in Figure 3.3, but the value of the direction may be any integer depending on the rotational angle.

The rule to transform a coordinate sequence to a stroke vector sequence is as follows:

1. The value of the direction of the first vector is selected in the range between -11 to 12.
2. Other values are determined so as to minimize the difference of the directional values of two successive vectors.

The stroke vector sequence closely represents the shape of the original stroke, and includes such information as the direction and angle of rotation, and the curvature and length of a stroke.

For example, Figure 3.4 shows the stroke vector sequence of stroke 「ろ」 in a graphic form. The horizontal axis represents the vector number and corresponds to the stroke length. The vertical axis represents the rotational angle of vectors. A jump in directional value at A' or B' implies that there exists a corner or a cusp on the curve at point A or B. Further, a small loop is found at the end of the stroke from the fact that the directional difference between point B'' and C' exceeds 24 and that the tangent to the curve increases as it comes close to the end of the curve.

3.3 Encoding of the shape

The classification of the stroke shape is accomplished by matching the stroke code against the shape dictionary. The length of the stroke vector sequence obtained by the rule in Section 3.2 differs from stroke to stroke. Thus, it is necessary to normalize the length of the stroke vector sequence in order to compare shapes. Experimentally, two or more major characteristics do not appear within the range of one tenth in stroke length, and no important characteristics is entirely included within that range. Thus, the vector sequence is divided into ten sections at nearly equal spacing and is encoded by selecting eleven representative vector values. If there exists an extreme value in a section, this value is selected as a sample value of this section in order to include the information about a corner or a cusp. We call such a sampled vector sequence a general stroke vector sequence (GSVS). The GSVS for ⌜ろ⌟ of Figure 3.4 is represented as:

G(ろ) = 1, 1, 8, 8, 2, 1, -1, -5, -9, -15, -28.

3.4 Distance of two shapes

Let $G_i(x)$ represent angular direction of stroke G_i at arc length x, and the length of a stroke be normalized to L. The distance (SHSA) of two shapes of strokes G_1 and G_2 is most simply defined as:

$$\mathrm{SHSA} = \int_{1}^{L} \left| G_1(x) - G_2(x) \right| dx. \qquad (3.2)$$

However, there are many cases that Equation (3.2) is too rough to evaluate the shape distance because of the individual differences in shape. In this case Equation (3.3) may be used, where the idea of dynamic programming[4] is used:

$$\mathrm{SHSA} = \min_{v(t)} \int_{1}^{L} \left| G_1(t) - G_2(v(t)) \right| dt, \qquad (3.3)$$

where v(t) is an increasing function of t which adjusts shape G_2 along the axis of arc length so that best matching is obtained. The meaning of Equation (3.3) is explained that shape G_1 is matched shape G_2 that is drawn on a rubber which is elastic in the direction of the arc length axis. The evaluation of Equation (3.3) has two undesirable features:

1. Computing time is big.
2. A close matching may be unintentionally obtained even if G_1 is only partially matched with G_2.

These defects can be avoided by imposing appropriate restrictions on v(t). It is not necessary to get close matching but to determine the shape distance that is enough to classify patterns. Comparing two points which are located in extremely distant locations may leads to rather wrong conclusions.

Hand-written strokes usually have some fluctuations both in direction and length. But if these fluctuations are always taken into account, different patterns might be recognized as the same. Since Equation (3.2) may give big errors when the positions of corresponding corners or cusps of two strokes do not come close, it will be reasonable to match two patterns by matching corresponding corners or cusps. This implies that the function v(t) in Equation (3.3) should be selected so that corresponding corners come close. In the actual implementation, directional differences of ±1 (15°) are ignored, and positional differences of at most +20 percent length are permitted.

3.4.1 Evaluation of shape distance 1

The evaluation of Equation (3.2) is very simple as is shown shortly, and in many cases this equation is sufficient to determine the shape of a stroke. Let an input stroke A and a dictionary entry D be encoded as:

$$\mathrm{GSVS}(A) = a_0 a_1 \ldots a_{10},$$

$$\mathrm{GSVS}(D) = d_0 d_1 \ldots d_{10}. \qquad (3.4)$$

Shape distance of Equation (3.2) (SHSA1) is calculated as follows:

$$\mathrm{SHSA1} = \sum_{i=0}^{10} \min_{c} \left| a_i - d_i + c \right|, \qquad (3.5)$$

where c represents ignorable directional difference as: c = -1, 0, 1.

3.4.2 Evaluation of shape distance 2

When fluctuations of shape are large enough, it is necessary to evaluate the shape distance by Equation (3.3) (SHSA2) where the selection of function v(t) is essential. We have devised an algorithm which gets the local optimum matching by adjusting the indices of vector components. This adjustment is called positional correction below and corresponds to function v(t) in Equation (3.3). The detailed algorithm is explained below (see Figure 3.5):

Step 1: Possibility check

Check to see if two shapes have any possibility to agree. If there exist more than K_2 indices which satisfy Equation (3.6), go to Step 2 and calculate the shape distance; otherwise, two shapes do not agree. (The K_i's are constants in this section.)

$$\left| a_i - d_i \right| < K_1 . \quad (i = 0, 1, \ldots , 10) \qquad (3.6)$$

Step 2: Initialization to evaluate the shape distance

Set initial values;

$i = 0$ (positional index),

$j = 0$ (positional correction),

and go to Step 3.

Step 3 Calculation of the directional difference

Calculate the directional difference between the corresponding positions. If this difference is acceptable, that is, if Equation (3.7) is satisfied, go to Step 4 and get the optimum matching; otherwise, go to Step 6 to search for a better matching.

$$SA = \min_{c} \left| a_{i+j} - d_{i} + c \right| \leqq K_{3} . \qquad (3.7)$$

The meaning of c is same as in Section 3.4.1.

Step 4: Get the optimum matching

If the positional correction has not been applied ($j = 0$), set $c_i = SA$, and go to Step 5. Otherwise, check to see if a better matching is obtained by restoring the positional correction. That is, if the following condition is satisfied:

$$\left| a_{i+j+k} - d_{i} \right| + K_{4} \leqq \left| a_{i+j} - d_{i} \right| , \qquad (3.8)$$

where $k = -\text{sign}(j)$, set

$$c_{i} = \min_{c} \left| a_{i+j+k} - d_{i} + c \right| , \qquad (3.9)$$

else set

$$c_{i} = SA,$$

and go to Step 5 to check termination or iteration.

Step 5: Check for termination

If $i = 10$, go to Step 7 for termination; else advance to the next position, $i = i + 1$, and repeat Step 3.

Step 6: Positional correction

Check for a better matching condition by Equation (3.8) and check for a big directional change by Equation (3.10) (any one of them is sufficient):

$$\left.\begin{array}{l} \left| d_{i} - d_{i+1} \right| \geqq K_{5}\text{, or} \\ \left| d_{i-1} - d_{i} \right| \geqq K_{5}\text{, or} \\ \left| a_{i+j} - a_{i+j+1} \right| \geqq K_{5}\text{, or} \\ \left| a_{i+j+1} - a_{i+j} \right| \geqq K_{5}. \end{array}\right\} \qquad (3.10)$$

If both checks fail, no positional correction is taken; set c_i = SA and go to Step 5. If both checks are passed, find the value of the positional correction, s, which satisfies the following:

$$\left| a_{i+s} - d_{i} \right| \leqq K_{3}, \qquad (3.11)$$

where s = 0, 1, -1, 2, -2 is tested in this order.

Then, set j = s and

$$c_{i} = \min_{c} \left| a_{i+j} - d_{i} + c \right|, \qquad (3.12)$$

where if i + j < 0, then i + j = 0;

and if i + j > 10, then i + j = 10.

Go to Step 5. If no such value of the positional correction is found, set c_i = SA, and go to Step 5.

Step 7: Termination

Calculate the shape distance as follows:

$$\mathrm{SHSA2}(A,D) = \sum_{i=0}^{10} c_{i}. \qquad (3.13)$$

If Equation (3.14) is satisfied, the shapes of two strokes are determined to be the same.

$$\mathrm{SHSA2}(A,D) \leqq K_{6}. \qquad (3.14)$$

The following threshold values have been employed in our experiments:

$K_1 = 3$ (45°),

$K_2 = 3$ (30 percent of the stroke length),

$K_3 = 7$ (105°),

$K_4 = 3$ (for better matching),

$K_5 = 6$ (curvature: 90°),

$K_6 = 20$ (30° per one point).

Figure 3.6 gives a flow chart for this method. This method is called GSVS method. Figure 3.7 shows how the individual differences in shapes are suppressed, where patterns with X mark do not satisfy the criterion (3.14). These results indicate that the method is quite satisfactory for the purpose of classifying strokes.

3.5 Complexity of a stroke

It will be wasteful in computing time to compare the input pattern with all the entries in the shape dictionary by the GSVS method. We define <u>complexity</u> of a stroke as the total rotational angle:

$$\mathrm{COMP} = \sum_{i=0}^{9} \left| a_{i+1} - a_i \right| . \qquad (3.15)$$

It turns out, as our experiments indicate, that this quantity is useful in reducing drastically the number of comparisons, say to only 3 to 60 while there are 177 entries in the shape dictionary.

3.6 Determination of shape

A shape dictionary is prepared to classify the shape of a stroke. This dictionary is an array of the following items:

Stroke identifier (i),

GSVS (G_i),

Threshold value of the shape distance ($SHTH_i$),

Complexity ($COMP_i$),

Threshold value of the complexity ($CMTH_i$), and

Identifier of the subroutine analyzing the local features ($SBNO_i$).

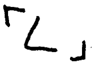
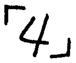
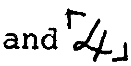
The shape of a stroke is determined as the shape of an entry in the shape dictionary for which values of the shape distance (3.5) or (3.13), and the difference of the complexity (3.16) are minimized.

$$CMSA_i = \left| COMP_i - COMP \right| , \qquad (3.16)$$

As is already stated, only a limited number of entries are selected depending on the number of strokes in a character and the complexity of the stroke. If we use a personal shape dictionary which is prepared for a specific person, it is enough to use SHSA1 rather than SHSA2 to determine the shape. If we use a common shape dictionary rather than a personal shape dictionary, the calculation of SHSA2 is often required. Currently, the number of entries in the common shape dictionary is taken to be 177 in our experiments.

4. Analysis of the shape relation

4.1 Similarity of a shape

Strokes which the GSVS method regards as different patterns may come from the same part of a given character, such as  and in the first stroke of  and , respectively. In order to reduce the size of the dictionary, it is of course desirable to regard such patterns as the same even if the shape distance between these patterns is large. To solve this problem, we define similarity of two shapes. First, a representative point in X-Y plane is assigned heuristically to each stroke in the shape dictionary so that the points which correspond to similar strokes come close. The similarity of two shapes is defined as the distance between two points, measured by the following equation:

$$\mathrm{SIMILARITY(A,B)} = \left| x_a - x_b \right| + \left| y_a - y_b \right| , \qquad (4.1)$$

where X-Y coordinates of two representative points A and B are (x_a , y_a) and (x_b , y_b). Figure 4.1 shows several shapes and their representative points. It turns out, as our experiments indicate, that this quantity is useful in reducing the number of entries of the character dictionary to one half to one forth.

4.2 Matching of the shape relation

Matching of the shapes of strokes in a given character with the characters in the character dictionary is performed by checking the similarity of corresponding strokes. If Equation (4.2) is satisfied for strokes designated in an entry of the character dictionary, the total shape distance is calculated by Equation (4.3), and the entry that has the minimum value of TSD is selected.

$$\mathrm{SIMILARITY}(S_k, D_{ik}) \leqq \mathrm{THSM}_i, \qquad (4.2)$$

$$\mathrm{TSD}(i) = \sum_{k=1}^{n} \mathrm{SIMILARITY}(S_k, D_{ik}), \qquad (4.3)$$

where the suffix k denotes the k-th stroke and i the i-th entry of the shape dictionary. If checking of local characteristics is indicated in the entry of the character dictionary, the specified routine is executed. The details are presented in Section 6.

5. Analysis of the positional relation

In order to recognize multi-stroke characters, it is necessary to analyze the positional relations between succesive strokes. These relations become more and more important when the number of strokes in a character increases.

The positional relation of two strokes is defined by the direction of a vector composed of the center points of each stroke. For example, the positional vector sequence of seven-stroke character 「京」 shown in Figure 5.1 is given as follows:

PV(京) = 19, 16, 2, 16, 18, 12, 0.

If POSITIONAL_DISTANCE (5.1) (see below) is smaller than the given threshold value in an entry of the character dictionary, it is determined that the positional relations match. The positional distance is defined as follows:

Let the positional vector sequence of the input character and a dictionary entry be VA_i and VD_i (i = 1, ... , KAKU - 1), respectively, and KAKU be the number of strokes.

$$\text{POSITIONAL\_DISTANCE} = \sum_{i=1}^{KAKU-1} PSA_i , \qquad (5.1)$$

where

$$PSA_i = \begin{cases} 1, \quad \text{if } VD_i = -1 \\ \text{(This means that } PSA_i \text{ is ignored because of} \\ \text{Reason 1 below.);} \\ PS_i , \text{ if } 0 \leqq VD_i \leqq 23; \\ \min (PS_i , 1). \quad \text{if } VD_i \geqq 100 \\ \text{(This means that } PS_i \text{ can be ignored because of} \\ \text{Reason 1 below.);} \end{cases}$$

$$PS_i = \min_c (\left| VA_i - VD_i \right| \bmod 12 - C)^r ,$$

where c represents ignorable directional difference,

C = 0, 1; and $r > 2$.

The positional relation is not always unique because of the following two reasons:

1. If the distance of two middle points is small or if two strokes cross each other, a large error will occur in the direction of the vector.

2. The positional relations depend on the writing order of strokes.

We have prepared as many entries in the character dictionary as the number of different orders of writing.

6. Other characteristic parameters

The GSVS method mainly takes into account of global shape and does not pay much attention to local shape such as a sweep-up or a corner; thus, the method is unable to distinguish many similar shapes, although it is satisfactory for the classification of strokes. The following cases are examples in which patterns cannot be distinguished by this method:

1. The case where the existence of a corner point need be recognized,
2. The case where the information about length is needed, and
3. The case where another auxiliary information is needed.

In the following sections are presented some useful remarkes on the necessity of additional characteristic parameters.

6.1 Existence of a cusp

Sometimes the detection of a cusp is needed. In this case a cusp will be detected as follows:

Let a stroke vector sequence be V_i $(i = 1, 2, \ldots , n)$.

The point i which satisfies the following equation is considered as a cusp.

$$C = \left| V_{i-3} - V_{i+3} \right| > K. \quad (K: \text{constant}) \qquad (6.1)$$

If such points are detected succesively, the point which exhibits the maximum value of C is selected as a cusp.

6.2 Length of strokes

The difference between「土」and 「士」is the difference of the lengths of the second and third stroke. As strokes are sampled nearly at the equal spacing at the input stage, the number of data points gives the stroke length.

number of data points gives the stroke length.

6.3 Coordinates of start and end points of strokes

In order to discriminate「力」and「刀」, or「土」and「工」, the positional relation of the start points of the first and second strokes is essential.

6.4 Examples of discrimination of similar characters

There are 23 groups of similar characters in seven-stroke characters as shown in Figure 6.1. Several discriminating algorithms are given below:

1. Discrimination of「抜」and「技」
   Compare the X-coordinate of the fifth end point and the seventh start point.
2. Discrimination of「材」and「村」
   Check the directions of the seventh stroke, and then compare the length of the fifth stroke and the seventh stroke.
3. Discrimination of「決」,「沢」and「択」
   Compare the Y-coordinates of the first, second and forth start points.
4. Discrimination of「見」and「貝」
   If the seventh complexity is less than seven, then「貝」; otherwise「見」.

7. Experimental results

Table 7.1 summerizes the recognition scores in our experiments. The recognition scores of 90 to 93 percent (true character only) or 94 to 98 percent (indication of correct set of characters including the true character), depending on the level of processing, have been obtained. The level of processing measures the reliability of the results, and the level is in turn determined by the value of the shape distance and the positional distance, or by whether additional recognition steps are executed. The policy for recognition is to reduce the wrong results even at the expense of high rejection rate. Many characters which were recognized as different ones were those which are rather difficult to be read by average man. Table 7.2 presents some recognition errors and their causes.

Typical errors which occurred in the process of computing the number of strokes are shown in Table 7.1. The cause of errors is considered mainly due to the unstability of the up-down sensing contact of the stylus pen. They are therefore excluded from the statistics. Another problem is that as the number of strokes increases, people tend to make mistakes in the way of writing both in the number of strokes and the order of writing; this problem is resolved by appending additional entries to the character dictionary.

The processing time of our experiment is about one second per character, and the storage requirement is as follows:

Recognition procedure	69 kB,
Character and shape dictionary	80 kB.

8. Conclusion

An on-line recognition method for hand-written characters utilizing stroke vector sequences and a positional vector sequence is presented. The number of target characters is about 2000 which include almost all the characters - alphabet, numerals, KATAKANA, HIRAGANA and KANJI - usually used in Japan. Good recognition scores of 90 to 98 percent, depending on the level of processing, have been attained. In order to improve the recognition rate, it would be necessary to include context analysis; we have, however, separated this problem as a different problem. Since our scheme uses the number of strokes as the parameter for the primary selection, the recognition is sensitive to the detection of strokes. As for the character dictionary, it is still necessary to improve entries for the characters that tend to produce many errors and to make further adjustment to the values of threshold.

We have classified characters into three major types and have employed five types of recognition procedures depending of the number of strokes; and this is reasonable because the shapes of strokes in characters with small number of strokes are complex, while those in characters with many strokes are fairly simple.

The GSVS method, which is devised to analyze the shape of a stroke, can represent both skeleton and local characteristics using only small amount of information. The restricted dynamic programming method is effective to determine the shape of strokes because the method suppresses the individual differences which come from the hand-writing process and because it reduces the sizes of the dictionary as well. Moreover, the similarity of two shapes, which is defined as the distance between two points associated with them, works well to reduce the dictionary size. In addition, the complexity of a stroke, which is defined as the total rotational angle of a stroke, is a useful index for reducing the processing time.

We automatically generated the entries for positional relations in the character dictionary. The whole of the generation algorithm is skipped in this paper.

This research was done during the academic years of 1972 to 1977 when the first author was a faculty member at the department of Information Science of Kyoto University.

The authors wish to acknowledge the people in that department who assisted with the experiments. Finally, this paper has been greatly improved by the insightful comments of Professor Yasuhiko Ikebe of University of Tsukuba and of the anonymous referees. The authors also wish to thank Professor Ikebe and the referees.

Appendix: Automatic generation of the character dictionary

The character dictionary for recognition is constructed by collecting a large amount of sample data and by repeating cut-and-try methods, where every individual difference is taken account. These works take much time and labor. We have succeeded in automatically generating the initial entries of the positional relations in the character dictionary. Further, the dictionary entries have been tuned so that better score may be obtained. The following is the summery of the generation method:

Let the positional vector of the dictionary and the i-th sample be D_j and A_{ij} (j = 1, 2, ... , n-1; n : the number of strokes), respectively.

Step 1: Set the initial value from the first sample.

Set i = 1, and $D_j = A_{ij}$, (j = 1, 2, ... , n-1). Go to Step 2.

Step 2: Begin revising the positional relations.

Set i = i + 1, and j = 1. If i is greater than the number of samples, then stop. Go to Step 3.

Step 3: Branch depending on the difference of the direction of the corresponding vectors.

Let $PD = \left| D_j - A_{ij} \right| \bmod 12$.

If $PD \leq 2$, go to Step 6.

If PD = 3, go to Step 5.

If $PD \geq 4$, go to Step 4.

Step 4: Ignore the j-th term.

Set $D_j = -1$ in order to indicate that this term should be ignored when the positional difference is calculated. Go to Step 6.

Step 5: Revise D_j.

Set D_j so that PD = 2, and go to Step 6.

Step 6: Test end of processing.

Set $j = j + 1$. If $j \leq n - 1$, go to Step 3 to process the next term; otherwise, go to Step 2 to process the next sample.

References

[1] A. Arakawa, I. Masuda: On-line Recognition of Hand-written HIRAGANA Characters, Proc. Annual Conf. IPSJ 33, 1974.

[2] S. Terai, K. Nakata: On-line Recognition of Hand-written KATAKANA Characters, Johoshori 13, 145-153, 1972.

[3] G. M. Miller: On-line Recognition of Hand-generated Symbols, Proc. AFIPS 35, 399-412, 1969.

[4] Y. Isomichi, T. Ogawa: Pattern Matching Using Dynamic Programming Method, Johoshori 16, 15-22, 1975.

[5] T. T. Kim, D. C. Evans: Syntax Directed On-line Recognition of Cursive Writing, TR 4-8, Univ. of Utah, 1969.

[6] M. I. Bernstein: Hand-Printed Input for On-line Systems, SDC Tech. Memo., April 1969.

[7] G. F. Groner: Real-time Recognition of Hand-printed Text, The RAND Corp., May 1968.

[8] S. Fujiwara, K. Ikeda, Y. Tominaga, T. Kiyono: On-line Recognition of Hand-written Characters by General Contour Vector Sequence Method, Johoshori 17, 191-199, 1976.

[9] K. Ikeda et al.: On-line Recognition of Hand-written Characters Utilizing Positional and Stroke Vector Sequences, Proc. 4th IJCPR, 813-815, 1978.

[10] A. Kawakami, T. Kawakami: Human Factors in Rainputto Keyboard for KANJI Input, Proc. 1st UJCC, 292-295, 1972.

Shape	ID number in the shape dictionary	Percentage
→	1, 2	36 %
↙	3	12 %
↓	23, 26, 27	30 %
↘	11, 12	2 %
⤵	15, 16, 17, 18, 19, 20, 21, 22	9 %
Others		11 %

(for seven stroke characters)

Figure 1.1 Shape distribution of seven-stroke characters.

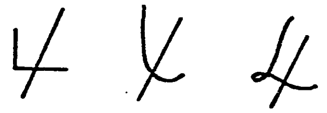
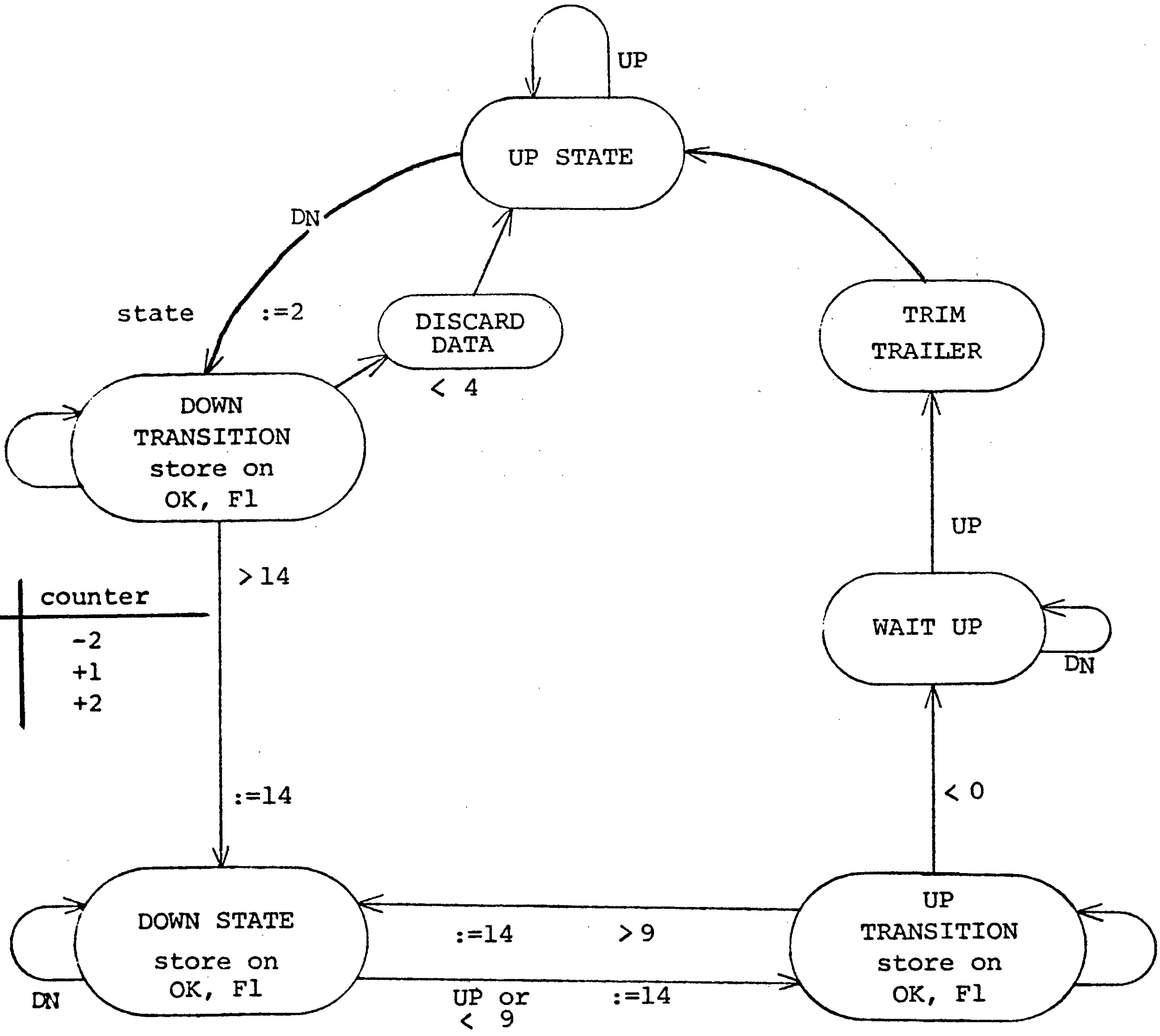


Figure 1.2 Similar figures in a character.



pen state	counter
UP, DNF2	-2
DNF1	+1
DNOK, DNNR	+2

pen state	counter
F2	-2
F1	-1
OK, NR	:=14

pen state	counter
UP, DNF2	-2
DNF1	+1
DNOK, DNNR	+2

[NOTE]
DN : down
UP : up
NR, OK, F1, F2 : refer to Figure 4.2

Numbers represent the state of a counter which controls the state transitions of the virtual pen.

Figure 3.1 Input of hand-written strokes.

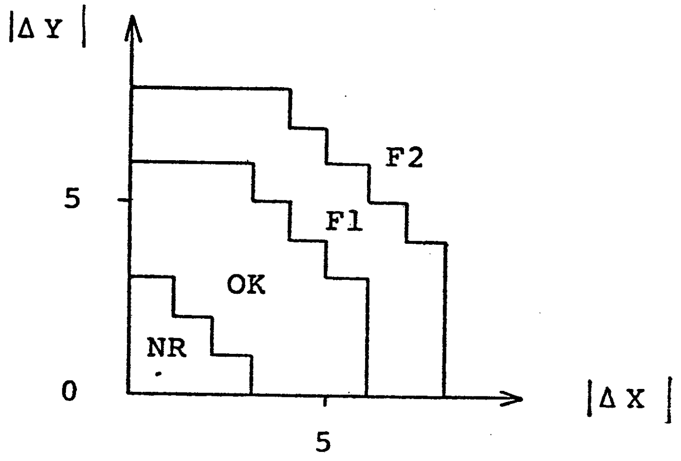


Figure 3.2 Spatial filtering.

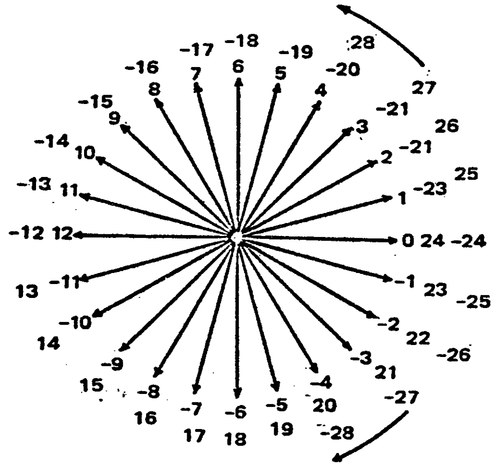


Figure 3.3 Code for direction.

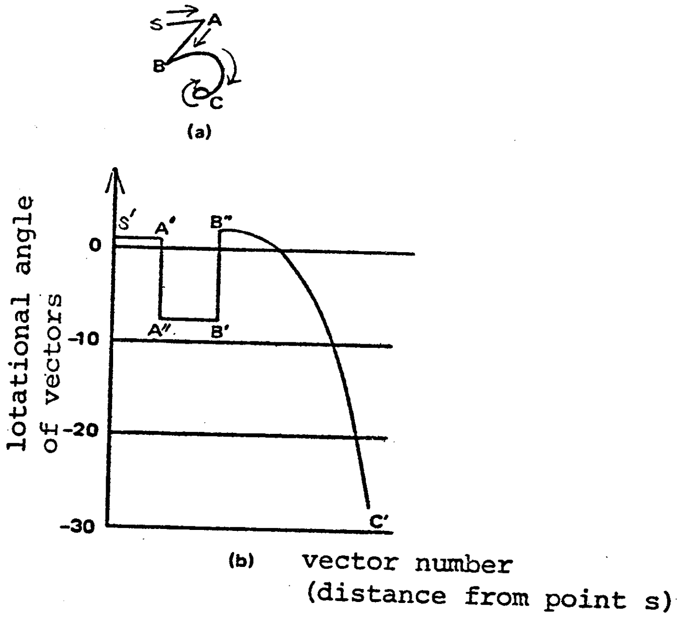


Figure 3.4 An example of a stroke vector sequence.

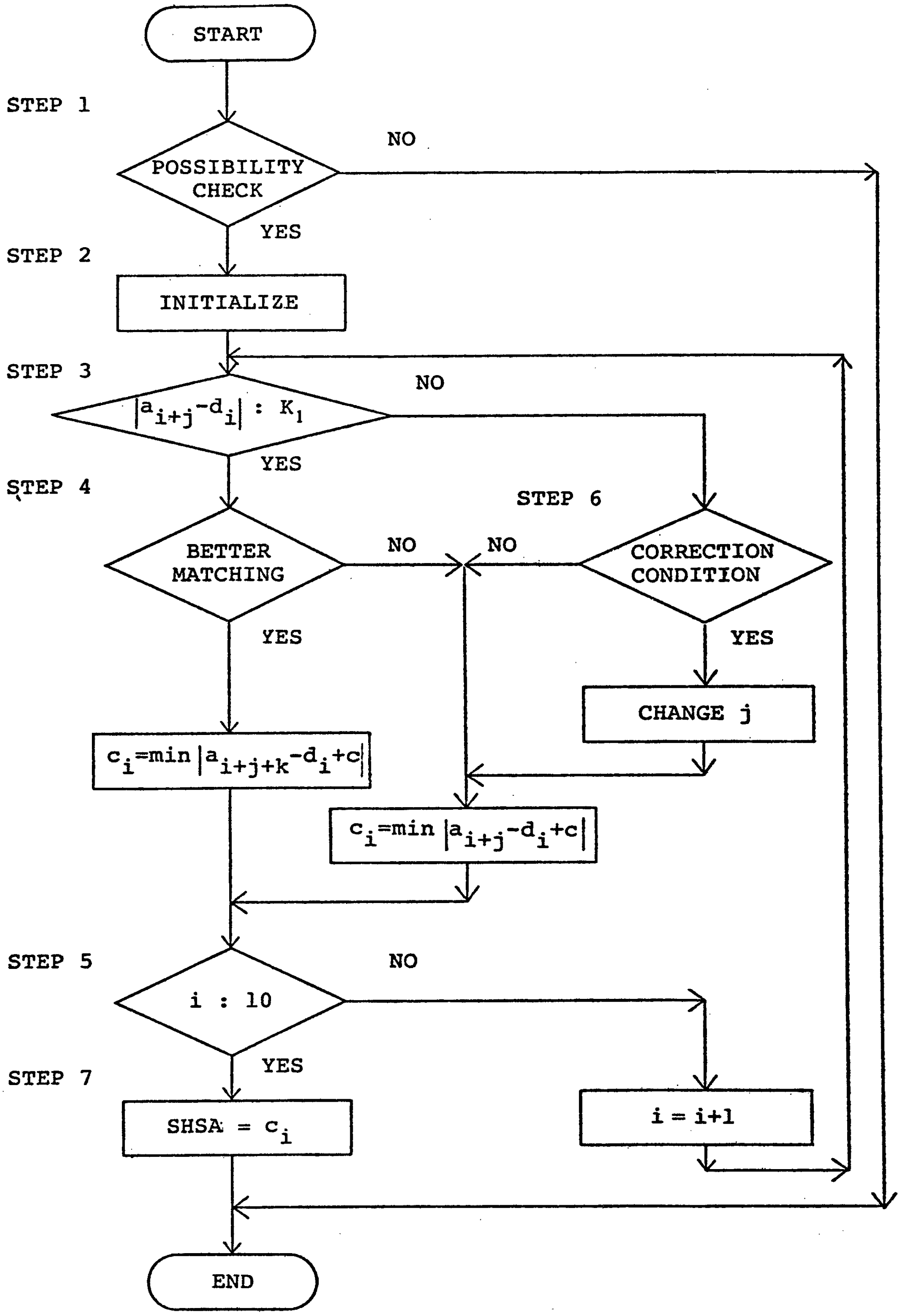
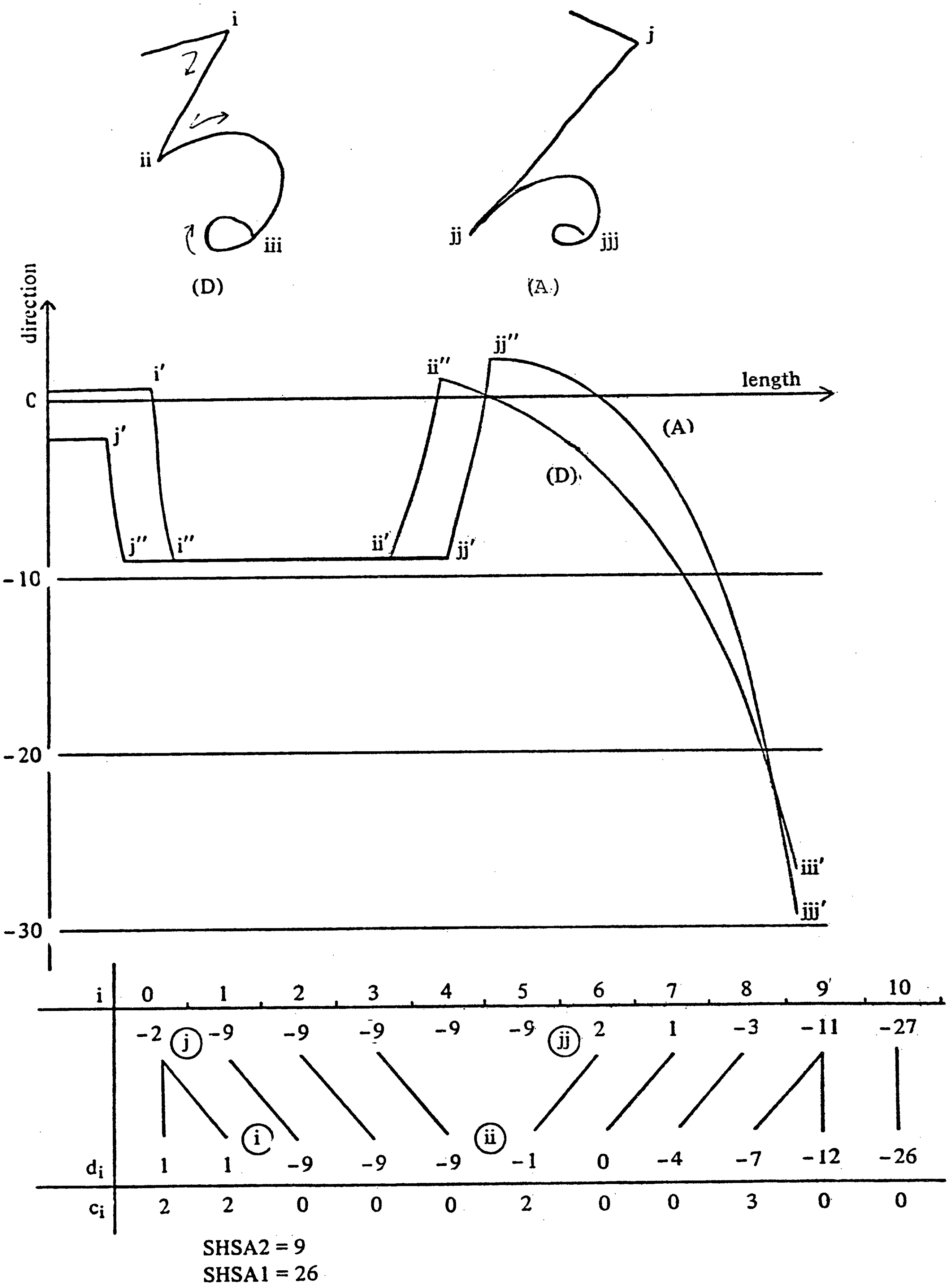


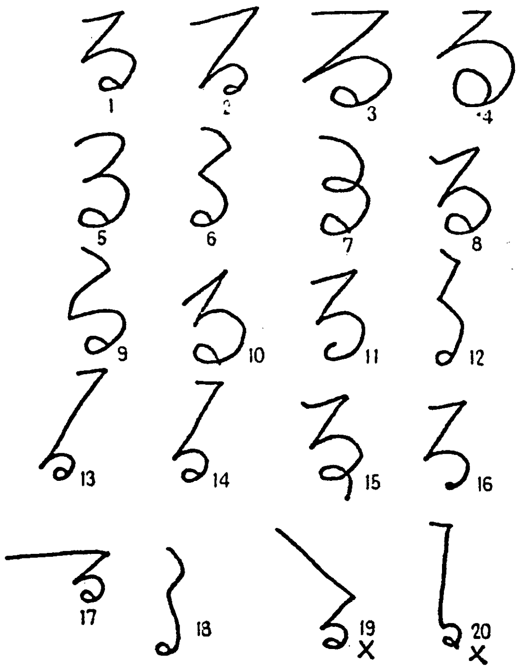
Figure 3.5 Calculation of the shape distance.



i	0	1	2	3	4	5	6	7	8	9	10
	-2	-9	-9	-9	-9	-9	2	1	-3	-11	-27
d_i	1	1	-9	-9	-9	-1	0	-4	-7	-12	-26
c_i	2	2	0	0	0	2	0	0	3	0	0

SHSA2 = 9
SHSA1 = 26

Figure 3.6 An example of calculation of the shape distance.



#2-#20 are compare with #1.
X denotes that the shape is not different.

Figure 3.7 Results of shape analysis.

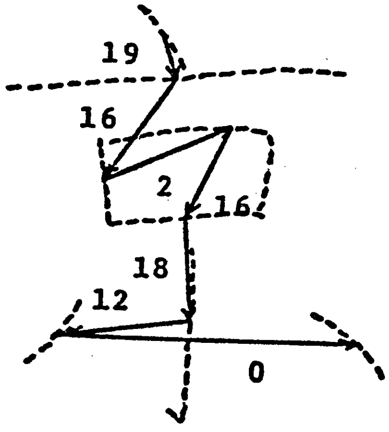


Figure 5.1 Analysis of the positional relation.

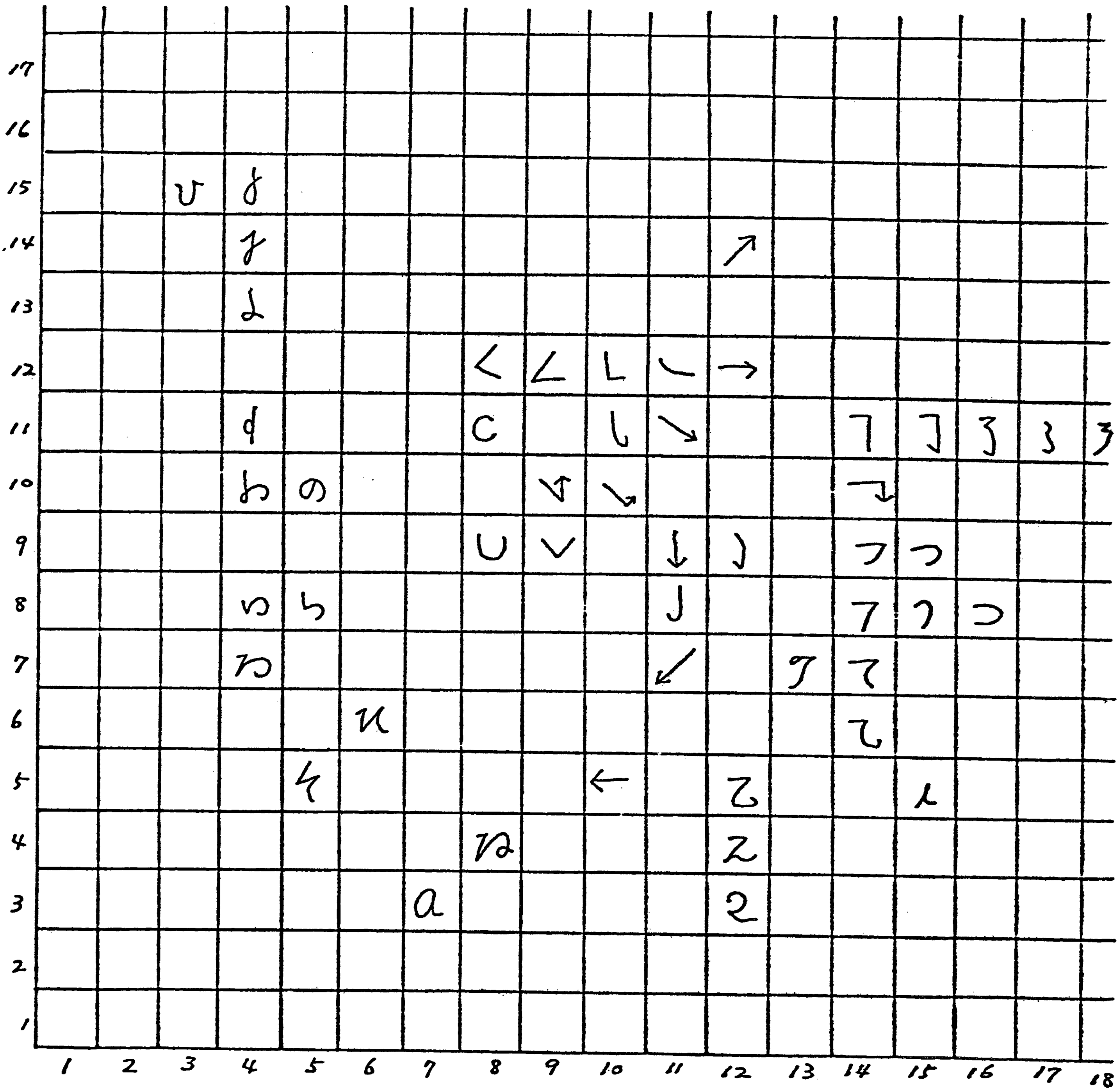


Figure 4.1 Assignment of the similarity.

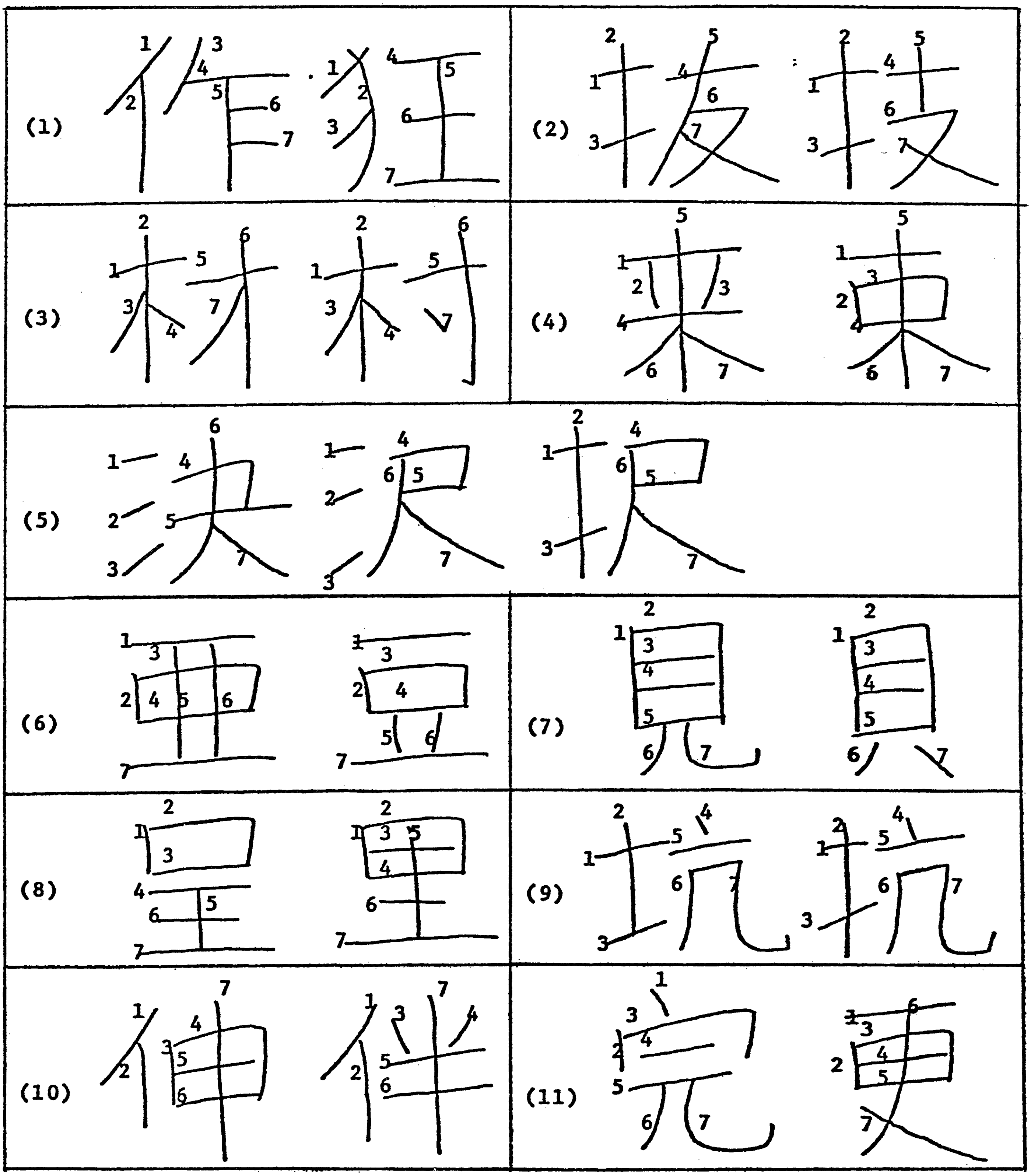


Figure 6.1 Similar seven-stroke characters.

KAKU	KINDS	CNTP	NOTF	LEVEL 1				LEVEL 2				LEVEL 3				TOTAL
				OK11	OK21	OK31	NOC1	OK12	OK22	OK32	NOC2	OK13	OK23	OK33	NOC3	
1	61	36	2	94	0	0	11	138	0	0	20	241	0	0	23	565
2	83	0	20	189	2	0	18	247	16	0	21	15	0	0	4	532
3	60	0	0	12	2	0	0	317	5	5	12	3	1	0	4	361
4	66	18	4	11	0	0	0	448	7	2	2	19	3	2	14	530
5	90	1	35	20	0	0	0	555	10	0	39	92	9	0	217	978
6	108	14	15	135	0	0	2	628	13	0	19	140	10	1	45	1022
7	132	26	3	171	0	0	5	797	26	5	23	210	28	7	72	1373
8	177	2	0	259	0	0	3	1379	44	2	38	116	29	0	76	1948
9	173	13	44	135	0	0	1	1022	43	1	52	320	23	2	80	1736
10	189	20	80	154	0	0	2	775	35	0	19	512	17	0	89	1703
11	180	13	22	314	9	148	2	0	5	0	4	1	0	0	7	525
12	171	7	44	513	23	1	2	217	6	1	3	32	2	0	10	861
13	142	9	43	581	10	0	0	671	31	0	3	64	3	0	7	1422
14	109	57	48	515	8	2	0	473	15	0	6	19	0	0	2	1145
15	99	16	16	387	8	0	6	527	15	1	12	8	0	0	0	996
16	71	33	24	126	0	0	0	519	19	2	10	0	0	0	0	733
17	34	9	24	58	0	0	0	263	0	0	9	0	0	0	0	363
18	34	0	54	50	0	0	0	215	0	0	0	0	0	0	0	319
19	21	0	36	31	0	0	0	106	0	0	0	0	0	0	0	173
20	13	11	18	12	0	0	0	67	0	0	0	0	0	0	0	108
21	5	43	0	0	0	0	0	0	0	0	0	0	0	0	0	43
22	3	23	0	0	0	0	0	0	0	0	0	0	0	0	0	23
23	X	4	0	0	0	0	0	0	0	0	0	0	0	0	0	4
24	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	2021	855	532	3767	62	151	52	9364	290	19	292	1792	125	13	650	17463

Table 7.1 Experimental results.

	LEVEL 1						LEVEL 2						LEVEL 3					
KAKU	ACPT1	EROR1	RJCT1	TRUE1	INC21	INC31	ACPT2	EROR2	RJCT2	TRUE2	INC22	INC32	ACPT3	EROR3	RJCT3	TRUE3	INC23	INC33
1	17.8	2.1	80.2	89.5	89.5	89.5	43.9	5.9	50.3	88.2	88.2	88.2	89.4	10.2	0.4	89.8	89.8	89.8
2	35.5	3.8	60.7	90.4	91.4	91.4	82.0	10.7	7.3	88.4	92.1	92.1	84.8	11.5	3.8	88.1	91.6	91.6
3	3.3	0.6	96.1	85.7	100.0	100.0	91.1	6.6	2.2	93.2	95.2	96.6	92.0	8.0	0.0	92.0	94.2	95.6
4	2.1	0.0	97.9	100.0	100.0	100.0	89.6	2.1	8.2	97.7	99.1	99.6	93.4	5.9	0.8	94.1	96.1	96.9
5	2.0	0.0	98.0	100.0	100.0	100.0	58.9	5.0	36.1	92.1	93.8	93.8	68.3	28.1	3.6	70.8	72.8	72.8
6	13.4	0.2	86.4	98.5	98.5	98.5	75.7	3.4	20.9	95.7	97.4	97.4	89.6	8.9	1.5	90.9	93.3	93.4
7	12.7	0.4	86.9	97.2	97.2	97.2	71.9	4.4	23.8	94.3	96.8	97.3	87.5	12.3	0.2	87.6	91.7	92.6
8	13.3	0.2	86.5	98.9	98.9	98.9	84.2	4.5	11.4	95.0	97.5	97.6	90.1	9.9	0.0	90.1	93.9	94.0
9	7.8	0.1	92.1	99.3	99.3	99.3	67.2	5.6	27.2	92.3	95.7	95.8	85.7	11.7	2.6	88.0	91.9	92.1
10	9.2	0.1	90.7	98.7	98.7	98.7	55.2	3.3	41.5	94.3	97.9	97.9	85.6	9.6	4.8	89.9	93.1	93.1
11	61.3	31.1	7.6	66.4	68.3	99.6	61.3	32.8	5.9	65.1	68.0	98.8	61.5	34.2	4.3	64.3	67.1	97.3
12	60.1	3.0	36.9	95.2	99.4	99.6	85.5	4.2	10.3	95.3	99.1	99.3	89.2	5.6	5.2	94.1	97.9	98.1
13.	41.1	0.7	58.2	98.3	100.0	100.0	88.6	3.1	8.3	96.6	99.8	99.8	93.1	3.8	3.0	96.1	99.3	99.3
14	47.3	0.9	51.7	98.1	99.6	100.0	90.8	2.8	6.3	97.0	99.2	99.4	92.6	3.0	4.4	96.8	99.0	99.2
15	39.5	1.4	59.1	96.5	98.5	98.5	93.3	4.3	2.4	95.6	98.0	98.1	94.1	4.3	1.6	95.6	98.0	98.1
16	18.0	0.0	82.0	100.0	100.0	100.0	92.1	4.4	3.4	95.4	98.2	98.5	92.1	4.4	3.4	95.4	98.2	98.5
17	16.4	0.0	83.6	100.0	100.0	100.0	90.7	2.5	6.8	97.3	97.8	97.3	90.7	2.5	6.8	97.3	97.8	97.3
18	15.7	0.0	84.3	100.0	100.0	100.0	83.1	0.0	16.9	100.0	100.0	100.0	83.1	0.0	16.9	100.0	100.0	100.0
19	17.9	0.0	82.1	100.0	100.0	100.0	79.2	0.0	20.8	100.0	100.0	100.0	79.2	0.0	20.8	100.0	100.0	100.0
20	12.4	0.0	87.6	100.0	100.0	100.0	81.4	0.0	18.6	100.0	100.0	100.0	81.4	0.0	18.6	100.0	100.0	100.0
21	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
22	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
23	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
24	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
25	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
TOTL	22.0	1.5	76.4	93.4	95.0	98.7	76.8	5.1	18.2	93.8	96.3	97.5	87.2	9.7	3.1	90.0	92.9	94.0

<abbreviations>

CNTP: Cannot process because of data error etc.,
NOTF: Not found,
OKij: Found as the i-the result in the j-the level,
NOGj: Error in the j-th level,
KAKU: Number of strokes.

ACPTi = (GOOD1i / TOTAL) * 100%
ERORi = ((LEVELi - GOOD1i) / TOTAL) * 100%
RJCTi = (1 - LEVELi /TOTAL) * 100%
TRUEi = (GOOD1i / LEVELi) * 100%
INCji = (GOODji / LEVELi) * 100%

where

$$\mathrm{GOOD}ji = \sum_{h=1}^{j} \sum_{k=1}^{i} \mathrm{OK}hk$$

$$\mathrm{LEVEL}i = \sum_{k=1}^{i} \mathrm{SBTTL}k$$

$$\mathrm{SBTTL}k = \sum_{j=1}^{3} \mathrm{OK}jk + \mathrm{NOGD}k$$

TOTAL = LEVEL3 + NOTF

$$\mathrm{NOGD}i = \sum_{k=1}^{i} \mathrm{NOG}k$$

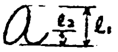
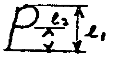
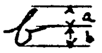
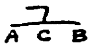
Input pattern	Result	Cause of error
/	1	Small curvature in 「)」 .
O	a	Deffects of discriminating routine of 「a」 and 「o」. If $\ell_1/\ell_2 < 0.4$, then a . 
Q	2	Deffect of input pattern.
p	n	Deffects of discriminating routine of 「p」 and 「n」. If $\ell_1/\ell_2 > 0.5$, then p . 
b	f	Deffects of discriminating routine of 「b」 and 「f」. If a/b < 0.8, then b . 
2	` Z	Deffects of discriminating routine of 「2」 and 「Z」.
l	1	Small curvature in 「l」 .
q	a	The last part of 「q」 is neglected.
ク	7	Deffects of discriminating routine of 「ク」 and 「7」 .
う	ラ	
ヲ	子	「フ」 is recognized as 「ろ」 .
丁	T	「亅」 is recognized as 「l」 .
ユ	コ	Deffects of discriminating routine of 「ユ」 and 「コ」 . If CB/AC > 0.33, then 「ユ」. 
れ	わ	The last part of 「れ」 is neglected.
チ	千	The first stroke is written as 「ヽ」 .
ツ	川	The first and the second stroke are recognized as 「l」 .
す	さ	「亅」 is recognized as 「l」 and deffects of discriminating routine.
チ	千	Small curvature in 「J」 .

Table 7.2 Examples of recognition errors.

INSTITUTE OF INFORMATION SCIENCE AND ELECTRONICS
UNIVERSITY OF TSUKUBA
SAKURA-MURA, NIIHARI-GUN, IBARAKI JAPAN

REPORT DOCUMENTATION PAGE	REPORT NUMBER ISE-TR-80-3

TITLE

On-line Recognition of Hand-Written Characters Utilizing Positional and Stroke Vector Sequences

AUTHOR(s)

Katsuo Ikeda (Univ. of Tsukuba, Science Information Processing Center, Sakura, Ibaraki, Japan)

Takashi Yamamura (Richo Co.)

Yasumasa Mitamura (Nichiden Software Co.)

Shiokazu Fujiwara (Nippon Telegraph and Telephone Public Co.)

Yoshiharu Tominaga (Mitsubishi Electric Co.)

Takeshi Kiyono (Osaka Electro-communication Univ.)

REPORT DATE July 29, 1980	NUMBER OF PAGES 39
MAIN CATEGORY Pattern Recognition	CR CATEGORIES 3.63

KEY WORDS Recognition of hand-written characters, Stroke vector sequence, Positional vector sequence, Stroke shape coding and analysis, Restricted dynamic programming, General Stroke Vector Sequence method, Similarity and complexity of a stroke, Skeleton and local characteristics

ABSTRACT

An on-line recognition method for hand-written characters utilizing stroke vector sequences and a positional vector sequence has been developed. The number of target characters is about 2000, and fairly good recognition scores have been attained. Our scheme uses the number of strokes as the primary parameter. We employ three types of recognition strategy depending on the number of strokes. The General Stroke Vector Sequence method devised to analyze the shape can represent both skeleton and local characteristics by small amount of information; and the restricted dynamic programming method is effective to determine the shape of a stroke. The similarity of two shapes and the complexity of a stroke have been introduced to reduce the dictionary size and the processing time, respectively.

SUPPLEMENTARY NOTES