UDC [003.63.053:159.937.52:681.32.022.056]:519.683.2(084.21)
Systems ● Computers ● Controls, Vol. 14, No. 3, 1983
Translated from Denshi Tsushin Gakkai Ronbunshi, Vol. 65-D, No. 6, June 1983, pp. 675-682

Online Recognition Algorithm for Hand-Sketched Flowchart by Candidate Lattice Method

Hiroshi Murase, Toru Wakahara and Michio Umeda, Regular Members

Musashino Electrical Communication Laboratory, N.T.T., Musashino, Japan 180

SUMMARY

The recognition of hand-sketched line figures has a wide range of application; for example, automatic fair copying of hand-sketched diagrams and conversations with a computer using hand-sketched figures. This paper investigates online recognition of hand-sketched line figures taking hand-sketched flowcharts as an object to be recognized. In particular, to lighten the load of a writer, the indication of segmentation between symbols (figure elements) is made unnecessary. Thus, to remove restrictions on the number of strokes and the order of strokes, a recognition method consisting of the following three stages is proposed: (1) The structure of each symbol form is represented by a digraph and by a search through paths of a digraph, an arbitrary number of strokes and order of strokes can be processed; (2) To handle handwriting variations DP-matching of strokes is performed; (3) From an input figure a candidate figure is extracted. It is represented by a candidate lattice (table form), and by searching the lattice the segmentation and the recognition of the symbols in the input figure are accomplished. When this method was applied to 120 handwritten flowcharts, the recognition rate of 97.9% was obtained to show the effectiveness of this method.

1. Introduction

Recently, there have been movements of fair copy handwritten documents containing letters and figures and to realize smooth conversations with computers using hand-written figures. To this end, this paper investigates a recognition algorithm of unrestricted hand-sketched line figures taking hand-sketched flow charts as its recognition objects. Together with character recognition techniques, the flowchart recognition techniques can be applied to a number of areas such as automatic fair copying of handwritten documents containing flowcharts, and automatic programming from hand-sketched flowchart input. The recognition algorithms for hand-sketched line figures can be classified roughly into two groups: offline type which accepts already hand-sketched figures from FAX and recognizes them [1 - 4]; and online type which recognizes line figures as they are hand-sketched on a tablet [5, 6]. This paper is concerned with the online type recognition. The existing online recognition algorithm [5] applies a simple matching of strokes to line figure recognition. It has several restrictions on writing such as ① the user must indicate segmentations between symbols (figure elements), and ② each symbol must be drawn in the predetermined number and order of strokes. Thus it is not easy for a layman to use. If these restrictions are removed, the input for the online recognition type becomes very easy to handle. In this paper we investigate a recognition algorithm for unrestricted hand-sketched line figures requiring ① no indication of segmentations, and ② no predetermined number of order of strokes [7].

The recognition algorithm consists of the following three parts.

(1) Extraction of candidate figures: All the subfigures that appear as symbols are extracted from the input figure as candidate figures. The structures of the forms of the symbols are represented by a digraph and the arbitrariness in the number and order of strokes is handled by searching through paths of the digraphs.

(2) Calculation of difference: The distance between an extracted candidate figure and the form of a symbol is calculated.

ISSN0096-8765/83/0003-0037$7.50/0

Table 1. Example of description of standard symbols ("mag. disk")

Stroke label	Type	Start point	End point	Intermediate point
A	Arc	(0, 5)	(4, 5)	(2, 6)
B	Arc	(0, 5)	(4, 5)	(2, 4)
C	Arc	(0, 1)	(4, 1)	(2, 0)
D	Straight-line	(0, 1)	(0, 5)	
E	Straight-line	(4, 1)	(4, 5)	

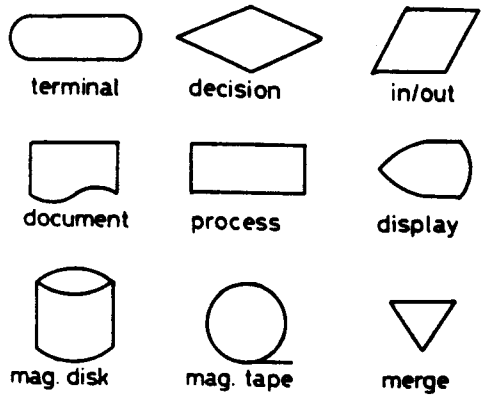

Fig. 1. List of symbols.

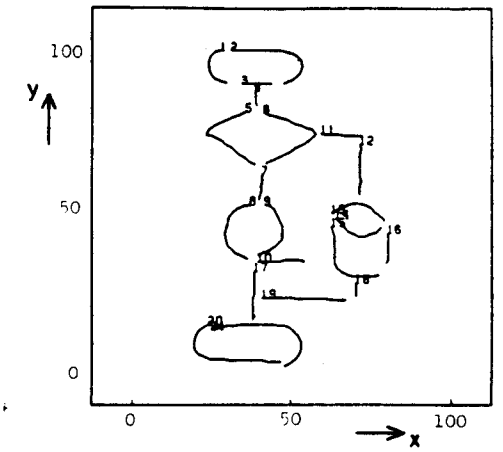

Fig. 2. Example of flowchart.

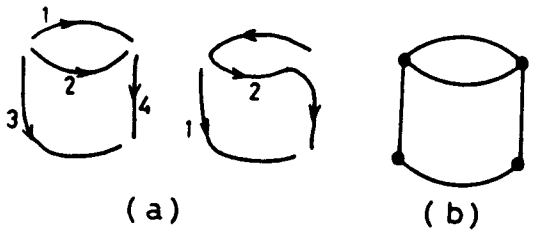

Fig. 3. (a) Examples of stroke order, (b) candidate end point.

To absorb the variations in the form due to hand-sketching, DP-matching is used.

(3) Candidate lattice: For all extracted candidate figures the names of the symbols and the distances are tabulated. (Hereafter, we call this table, candidate lattice. The candidate lattice follows the idea of phonological lattice used in voice recognition [8].) By searching through paths in this candidate lattice an optimum series of candidates is selected for the figure as a whole. Specifically, the recognition of the entire figure is performed simultaneously with the segmentation.

2. Recognition Object and Preprocessing

The figure portion of a flowchart (hereafter simply a flowchart) consists of symbols such as "terminal" and "decision" and line segments connecting the symbols. The flowcharts considered here are restricted to those composed of nine types of symbols of Fig. 1 and straight line segments. Most of the flowcharts fall in this class.

For the input of hand-sketched information we use a data tablet. A tablet can sample the up-down information and the XY-coordinate values of a pen at certain time intervals. The sequence of the coordinate values of a pen from the nth down to up is called the nth stroke. Figure 2 shows an example of a hand-sketched flowchart by tablet input. The numbers in the figure indicate the stroke order.

Sampling and normalization are applied as a preprocessing of a sampled coordinate sequence. The purpose of this is to represent every stroke with a sequence of a fixed number of coordinates and to make all the adjacent coordinates within a stroke equidistance.

3. Description of Standard Symbols

Before deciding on the description of standard symbols a preparatory experiment was conducted. A set of preparatory experiment data was obtained by asking 20 people to draw the nine types of symbol of Fig. 1

four times (altogether 720 symbols) without any restriction. The data analysis shows that there is no regularity in the number and order or strokes but that the possible starting points of a stroke are limited to a finite number of places. We call this finite number of points candidate end point. An example of the number and order of strokes in the data is shown in Fig. 3(a) and the candidate end points of symbol "mag. disk" are shown in Fig. 3(b).

By considering the possible expansion of the set of recognition objects it is desirable to simplify the description of the symbols inside a computer. Based on the results of the preparatory experiment we express each symbol by a set of strokes having candidate end points as their end points and each stroke by a combination of straight line segment and an arc. A straight line segment is represented by the coordinate values of its starting and ending points and an arc by the coordinate values of its starting, ending and middle points. A descriptive example of symbol "mag. disk" is given in Table 1 and Fig. 4.

4. Extraction of Candidate Figure

Input figures are flowcharts composed of symbols and straight line segments. A candidate figure for a symbol is a subfigure of an input figure which possibly is equal to the symbol. To extract candidate figures

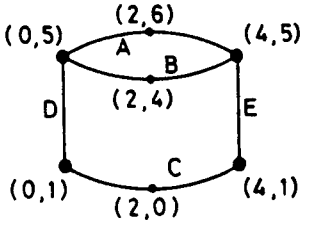

Fig. 4. Description of standard symbol ("mag. disk").

we use a topdown method. That is, assuming the existence of a symbol (called "supposed symbol" hereafter), we investigate the conditions for a given subfigure of an input figure to be a "supposed symbol." If the conditions are satisfied, the subfigure is taken as a candidate figure for the supposed symbol. In the following we give a detailed discussion.

4.1 Correspondence to candidate end point

Let us designate as input strokes those composing a given subfigure of an input figure. If an input stroke is a "supposed symbol," an end point of the input stroke corresponds to one of the candidate end points of the "supposed symbol." Thus we normalize the "supposed symbol" (match the maximum vertical values and the maximum horizontal values independently) to make it the same size as the given subfigure. Then we match the end points contained in the input stroke with the respective nearest candidate end points. In the case of the input stroke of Fig. 5(a), for example, if "mag. disk" of Fig. 5(a) is assumed, then the correspondence relation of Table 2 is obtained.

Due to the variations in handwriting the positions of the end points of an input stroke change, the changes are within a certain range. If an end point of an input stroke is not within a certain range from any candidate end point of the "supposed symbol," the given subfigure is judged to be different from the "supposed symbol" and the control moves to the processing of the other subfigures.

4.2 Generation of series of candidate strokes

This section discusses a method which proceeds as follows. A given subfigure is examined first topologically to determine whether or not it is appropriate as a "supposed symbol." If it is appropriate, based on the result of the correspondence of the

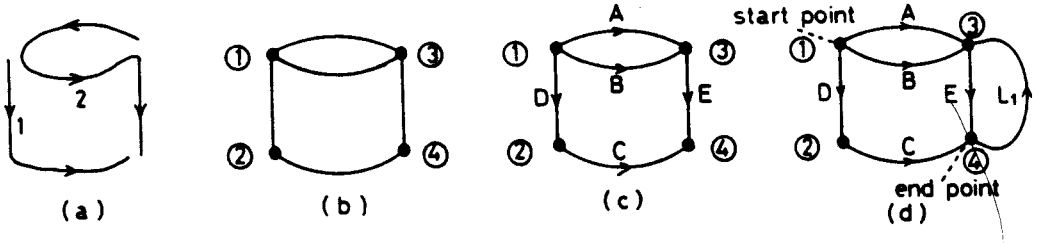

Fig. 5. (a) Input strokes, (b) Supposed symbol, (c) Directed graph, (d) Graph in search.

Table 2. Example of correspondence between input stroke end points and candidate end points

Stroke number	Start point	End point
1	①	④
2	③	④

candidate end points, the expected movements of a pen on the "supposed symbol" are enumerated as a candidate stroke series.

We assume that the given subfigure, that is, the input stroke, is composed of N strokes. Also, based on the description of the standard symbols (see Sect. 3) a "supposed symbol" is expressed as a digraph with candidate end points as nodes, and line and curved line segments as branches. A digraph is used to indicate the direction of the movement of a pen on a branch as positive. If we use v for a node name and a for a branch name, then a digraph can be represented in a computer by a connection matrix $D = (d_{va})$, where

$$d_{va} = \begin{cases} 1 \text{ (if node v is the starting point of branch a)} \\ -1 \text{ (if node v is the end point of branch a)} \\ 0 \text{ (all other cases)} \end{cases}$$

The digraph representation of "mag. disk" is as shown in Fig. 5(c) and its connection matrix is

$$\begin{array}{c} \\ ① \\ ② \\ ③ \\ ④ \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & -1 \end{bmatrix} \end{array}$$

For an input stroke to be appropriate for a "supposed symbol" there must be a path consisting of N strokes which passes all the branches of a digraph once and which has as their starting or ending point a candidate end point corresponding to the input stroke. Thus, the processing problem becomes a graph theory problem of finding an N stroke path with the restriction of the candidate end point correspondence obtained in Sect. 4.1. To simplify the problem further, we add branches corresponding to pen ups between consecutive strokes to change the problem to that of finding an Euler path, i.e., one stroke path. The processing steps are as follows:

(1) Add to the original digraph a new branch L_n (hereafter called fill-in branch) going from the node corresponding to the ending point of the nth stroke ($1 \leqq n \leqq N - 1$) to the node corresponding to the starting point of the (n + 1)st stroke. We give an example in Fig. 5(d). Its connection matrix is

$$\begin{array}{c} \\ ① \\ ② \\ ③ \\ ④ \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & L_1 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ -1 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 0 & -1 & 1 \end{bmatrix} \end{array}$$

(2) Find an Euler path having as its starting point the node corresponding to the starting point of the first stroke and having as its ending point the node corresponding to the ending point of the nth stroke. A condition for a solution to exist is that the degree of each node be even except the start and the end nodes. The degree of node v can be obtained by $\sum_a |d_{va}|$ from the vth row vector of the connection matrix. Hence, if the condition of even degree is not satisfied for a node, the given subfigure can not be the "supposed symbol." Thus another subfigure is selected for examination.

(3) If the condition on degree is satisfied, we find all the Euler paths that satisfy the conditions given below and take them as candidate stroke series. The following conditions become necessary since the order of input stroke series is considered.

① A path goes through the fill-in branches L_n ($n = 1, 2, \cdots, N - 1$) in the order $L_1, L_2, \cdots, L_{N-1}$.

② A path cannot go through two or more fill-in branches successively.

③ A path cannot traverse a fill-in branch in the reverse direction. But it can traverse other branches in either direction.

In searching for an Euler path the depth first search method is adopted which can be implemented easily by introducing a pushdown automaton. If the search is successful the given subfigure is extracted as a candidate figure for the "supposed symbol," and if it is unsuccessful, the given subfigure cannot be the "supposed symbol."

When we search for a path using the above-mentioned method for the example of Fig. 5(d), we can obtain the following six candidate stroke series:

① $+D, +C, L_1, -B, +A, +E$

② $+D, +C, L_1, -A, +B, +E$

③ $+B, +E, L_1, -A, +D, +C$

④ $+B, -A, +D, +C, L_1, +E$

⑤ $+A, +E, L_1, -B, +D, +C$

⑥ $+A, -B, +D, +C, L_1, +E$

Here "+D" denotes the traverse of branch D in the direction of the arrow and "-B" indicates the traverse of branch B in the reverse direction of the arrow. It can be seen that at the second series of the candidate stroke series given above, the correct correspondence with the input stroke [see Fig. 5(a)] appears.

Candidate stroke series can be obtained as a sequence of branches on a "supposed symbol" as explained above. Each branch is either a straight line segment or an arc according to the description of standard symbols (Sect. 3). Thus for each branch we generate a sequence of coordinate values of a straight line segment or an arc and approximate each stroke with a certain number of points. Using this sequence of coordinate values, in Sect. 5 we calculate the distance between a candidate stroke series and an input stroke.

4.3 Example of extraction of candidate figure

We give an example of extracting candidate figures for symbol "decision" from the hand-sketched flowchart given in Fig. 2. In this case symbol "decision" is the "supposed symbol." First, consider a subfigure of the flowchart (for example, the subfigure consisting of stroke numbers 13 and 14). We relate the end points of the input stroke to the candidate end points (see Sect. 4.1) and based on the resultant correspondence we search (see Sect. 4.2) for a path in the digraph (the "supposed symbol"). If there is no contradiction as a result of these processings, we make the given subfigure a candidate figure. Similar processings are applied successively to each subfigure and all the candidate figures are extracted. In this example candidate figures are extracted at five places as shown in Fig. 6.

5. Calculation of Distance

At the stage of extracting candidate figures only topological structures (the positions of the end points of an input stroke and the interconnections of the end points) are used and detailed forms of the input stroke are not considered. Thus as a second step we match the forms of the candidate stroke series obtained in Sect. 4.2 and the input stroke and calculate the distance between the forms.

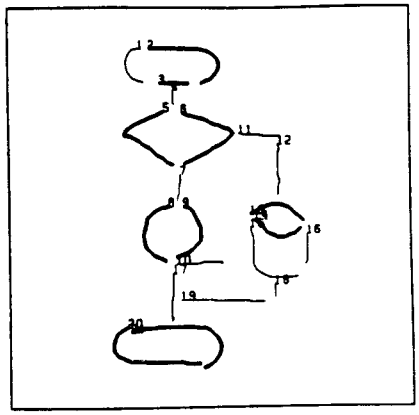

Fig. 6. Candidate figures for symbol "decision."

Table 3. Recognition rate of symbols

	Distance 1	Distance 2	Distance 3
Recognition rate	95.0%	96.9%	97.3%

For a measure to represent the distance between the form of a candidate stroke series and that of an input stroke we investigated the following three difference measures. The distance is defined as the sum of the distances between the corresponding strokes of a candidate stroke series and an input stroke. Thus we discuss the distance between a pair of strokes. We assume that each pair of corresponding strokes between an input stroke and a candidate stroke series consists of M points and that their coordinate value series is $\{(x_m, y_m)_{m=1,M}\}, \{(x'_m, y'_m)_{m=1,M}\}$. Here M is an approximation for the number of points in a stroke.

(1) Distance 1: d_1

Pair the coordinate points of an input stroke with those of a candidate stroke. Take the sum of the Euclidean distances between the corresponding pairs of points as the distance. That is, distance d_1^2 is

$$d_1^2 = \sum_{m=1}^{M} \{(x_m - x'_m)^2 + (y_m - y'_m)^2\} \qquad (1)$$

(2) Distance 2: d_2

To obtain Distance 2 proceed as for Distance 1 but use DP-matching to pair the coordinate points of strokes. That is, distance d_2^2 is

$$d_2^2 = \min_u \left[\sum_{m=1}^{M} \{ (x_m - x'_{u(m)})^2 + (y_m - y'_{u(m)})^2 \} \right] \quad (2)$$

Here u(m) represents the correspondence relation between the coordinate points and it has the following restrictions:

$$u(1) = 1 \quad (3)$$

$$u(M) = M \quad (4)$$

If u(i) = j, then $u(i+1) = \{ j \text{ or } j+1 \text{ or } j+2 \}$

$$j+1 \text{ or } j+2 \} \quad (5)$$

(3) Distance 3: d_3

In addition to the Euclidean distance between points we use DP-matching considering the tangential difference at each point. That is, distance d_3^2 is

$$d_3^2 = \min_u \left[\sum_{m=1}^{M} \{ (x_m - x'_{u(m)})^2 + (y_m - y'_{u(m)})^2 + \alpha \cdot h(m, u(m)) \} \right] \quad (6)$$

Here h(i, j) represents the difference between the tangential direction at (x_i, y_i) and that at (x_i', y_i'). It is given by

$$h(i,j) = \left| \tan^{-1}\left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i} \right) - \tan^{-1}\left(\frac{y'_{j+1} - y'_j}{x'_{j+1} - x'_j} \right) \right| \quad (7)$$

α is a coefficient of h(i, j) and it is a constant.

Table 3 shows the recognition rates when the data from the preparatory experiment (those used in Sect. 3) were recognized using the three types of distance. The results show that Distance 3, which uses DP-matching based on the Euclidean distance between points and the tangential differences at the points, gives the highest recognition rate. Based on these observations we decided to use Distance 3 for our recognition algorithm.

6. Candidate Lattice

6.1 Generation of candidate lattice

A candidate lattice is a table which shows as what candidate figure or what part of it each stroke in an input figure has been extracted. Below, we give a generation procedure of a candidate lattice. First, for a symbol we extract a candidate figure from an input figure (see Sect. 4). Since a candidate figure is obtained as a sequence of stroke numbers, we register the name of the candidate figure (symbol name) at the position of the corresponding stroke number sequence in a candidate lattice. Furthermore, we calculate the distance for the candidate figure (see Sect. 5) and register the distance. We repeat this processing for other symbols. Since a straight line segment is described by one stroke, for each stroke of an input figure a "straight line segment" is registered as a candidate figure. As an example part of the candidate lattice generated for the input figure shown in Fig. 2 is given in Fig. 7.

Table 4. Flowchart data

No. of writers	No. of writings	No. of data
6	Four each of five types	120

From a candidate lattice we can find the name of the candidate figure and its distance for a stroke number of an input figure.

6.2 Search for optimum figure sequence

By using the candidate lattice, we determine an optimum figure sequence covering the entire input figure and accomplish the segmentation and recognition of the symbols in the input figure. We show the procedure for that in the following. First, we choose candidate figures successively from the candidate lattice and extract a figure sequence going from the first stroke to the last stroke of the input figure. However, in general, there is more than one such figure sequence and it cannot be determined uniquely. Thus we introduce the following objective function and choose a sequence which minimizes it as an optimum figure sequence. We proposed two types of function for the objective function and compared them.

(1) Objective function 1: S_1

$$S_1 = \sum_{\substack{\text{candidate figure} \\ \in \text{sequence}}} (\text{distance of a candidate figure}) \quad (8)$$

(2) Objective function 2: S_2

$$S_2 = \sum_{\substack{\text{candidate figure} \\ \in \text{sequence}}} \frac{(\text{distance of a candidate figure})}{(\text{the number of strokes of the candidate figure})} \quad (9)$$

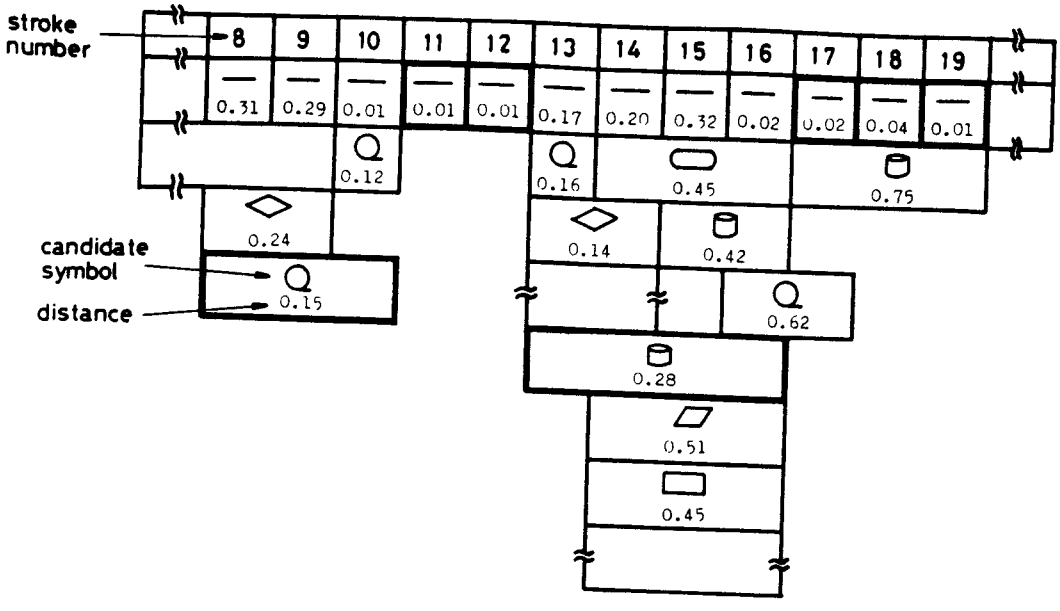

Fig. 7. Candidate lattice.

When objective function 1 is used, a path which minimizes the arithmetic sum of the distances becomes a solution.

When, on the other hand, objective function 2 is used, since the arithmetic sum is obtained after the distances are divided by the number of strokes, figures with a larger number of strokes are selected over those with a smaller number of strokes. We call this method the largest-subfigure-first method.

For example, if we select objective function 2 and search through the candidate lattice of Fig. 7, the figure sequence indicated by thick lines (Fig. 7) is chosen and the recognition result of Fig. 8 is obtained. In Fig. 8 together with symbol names the center coordinates (LOCATION) of each symbol extracted from the input figures and the vertical and horizontal sizes (SIZE) of each symbol are shown according to the coordinate system of Fig. 2.

7. Recognition Experiment

By using the recognition algorithm presented in Sects. 4, 5 and 6 we conducted a recognition experiment of hand-sketched flowcharts. The recognition objects are flowcharts composed of 5 to 9 symbols similar to those shown in Fig. 2 and straight line segments. In Table 4 we show the number of collected flowchart data used in the experiment.

The recognition rates are shown in Table 5. The recognition rate was 88.6% for objective function 1 and 97.9% for objective function 2; that is, when the largest-subfigure-first method was adopted. In the former case the recognition error rate 11.4% is divided into 1.8% for erroneous symbol name recognition and 9.6% for segmentation error. A segmentation error is (for example, as shown in Fig. 9) a mistaking of "mag. disk" as one "decision" and three "straight line segments." The recognition error rate 2.1% for objective function 2 is divided into 1.8% of erroneous symbol name recognition and 0.3% of segmentation error. Thus the segmentation accuracy was increased by adopting the largest-subfigure-first method and the effectiveness of objective function 2 has been recognized.

Here the recognition rate is defined as

$$\text{Recognition rate} = \frac{\text{the number of correctly recognized symbols}}{\text{the number of symbols in the input figure}} \times 100. \quad (10)$$

8. Discussion

According to simulation experiments on a minicomputer (1 MIPS) the computation times

STROKE	NAME	LOCATION	SIZE
1,2,3	TERMINAL	(39,94)	(29,10)
4	LINE		
5,6	DECISION	(40,71)	(34,17)
7	LINE		
8,9,10	MAG. TAPE	(42,41)	(24,18)
11	LINE		
12	LINE		
13,14,15,16	MAG. DISK	(71,39)	(17,23)
17	LINE		
18	LINE		
19	LINE		
20,21	TERMINAL	(36, 6)	(33,13)

Fig. 8. Recognition result.

Table 5. Recognition rate

	Objective function 1	Objective function 2
Recognition rate	88.6%	97.9%

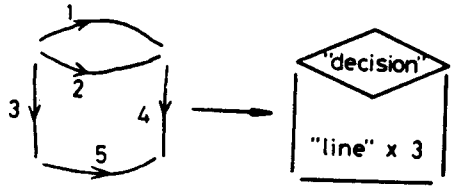

Fig. 9. Example of segmentation error.

(language is FORTRAN) for the respective stages in the case of the example of Fig. 2 were 25 sec for the first stage (extraction of candidate figures), 70 sec for the second stage (calculation of distance) and 1 sec for the third stage (search through a candidate lattice). Here we consider methods of discarding unnecessary candidate figures and reducing the amount of processing in the first and second stages which take up a majority of the time. However, if we discard too many candidate figures, correct candidate figures are discarded, also, thereby reducing the recognition rate. To select an optimum discarding parameter we investigate the following two points.

① Discarding at the extraction of candidate figures (stage 1) [see Sect. 4.1]

As mentioned in Sect. 4.1, if the distance between an end point of an input stroke and the nearest candidate end point is not within a certain threshold value, that candidate figure is discarded. To optimize this threshold value we obtained the recognition rate and the mean number of candidate figures [Fig. 10(a)] using 120 flowchart data given in Table 4 and taking the threshold for the distance between end points as a parameter. The mean number of candidate figures is the mean of the number of the candidate figures extracted for one symbol in an input figure. The distance between end points is the value when the size of a given figure (the length of the longer side of a rectangle containing the figure) is made equal to 1.0. From Fig. 10(a) we select 0.35 from within the range of the value of the threshold which does not reduce the recognition rate. As a result the mean number of candidate figures reduces from 8.3 to 6. For the candidate figures which are discarded by this processing the extraction process can be halted before the completion. Since the number of candidate figures decreases, the amount of processing in the second and third stages also decreases.

② Discarding at distance calculation (second stage) [see Sect. 5]

It is not necessary to register candidate figures with a sufficiently large distance on a candidate lattice. Therefore we discard candidate figures having a distance larger than a certain threshold value. Figure 10(b) gives the results of obtaining the recognition rate and the mean number of candidate figures using the threshold value for the distance as a parameter as in ①. We selected 0.1 from within the range of the value of the threshold which does not reduce the recognition rate. The mean number of candidate figures decreases from 6 to 2.8. Since we can stop the calculation of the distance by this process when the distance exceeds a threshold value, the number of candidate figures decreases and the amount of processing in the third stage decreases.

By the processing ① and ② explained above the total amount of processing

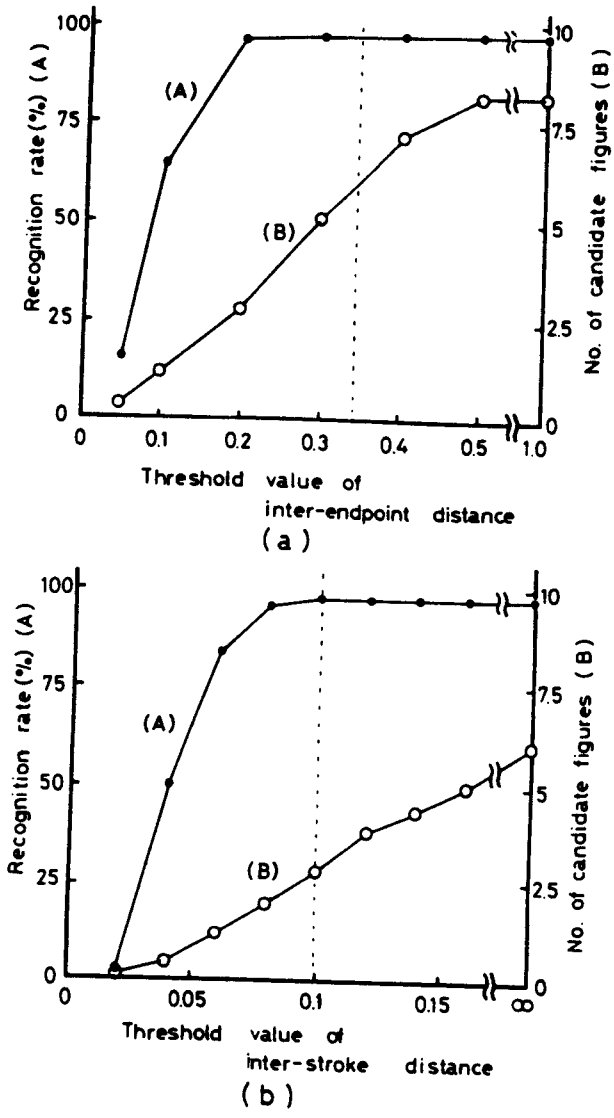

Fig. 10. Improvement of candidate extraction.

(computation time) of this method has been reduced by about 50% without decreasing the recognition rate.

In this method the amount of processing is proportional to the number of symbols contained in an input figure. But since the writing time is also proportional to the number of symbols for online inputs, by performing the recognition as soon as an input comes in, the increase in the amount of processing can be prevented from becoming a serious problem.

9. Conclusions

To achieve an online recognition of unrestricted hand-sketched line figures, especially flowchart figures, we proposed a recognition algorithm and examined its performance by experiments. We list the results below.

(1) By representing the form structure of each symbol with a digraph and by searching through the paths of a digraph, an arbitrary number and order of strokes can be handled.

(2) For the calculation of the distance between the forms of strokes, a DP-matching method is proposed, which incorporates the difference of tangential directions as well as the distance between the coordinate points. This method has been shown by experiments to be effective.

(3) For a method of assigning optimum symbols and sequence of straight line segments to an input figure we considered a candidate lattice method and achieved a high accuracy segmentation and recognition of individual symbols. As a result of a recognition experiment on 120 flowchart figures a recognition rate of 97.9% was obtained.

The candidate lattice method we proposed in this paper is highly general and it can be applied generally to the online recognition of hand-sketched line figures. Therefore, later we shall expand the set of recognition objects to a set of more general line figures not restricted to flowchart figures. As the number of symbols to be recognized increases with that expansion, it is expected that the number of candidate figures increases proportionately, the amount of processing increases, and the recognition deteriorates. Also, the increase in the number of symbols in an input figure, which is not the problem at this time, could cause difficulty as the set of recognition objects is expanded. These are some of the future research problems.

Acknowledgement. The authors would like to thank Mr. Azeyanagi, Mr. Hashimoto and Mr. Masuda for their constant guidance. They would also like to express their appreciation for valuable discussions with the researchers of their research group.

REFERENCES

1. Sato and Muneage. A method of processing hand-sketched drawings, 1981 National Conference of Information Processing Society (22nd).
2. Yoshida, Masui, Nagata and Oda. Automatic input/processing device for hand-sketched drawings, Information Processing, 22, 4, pp. 300-306 (Apr. 1981).
3. Nakura and Suenaga. A method of transforming hand-sketched drawings to figure data structures using FAX and special marks, 10th Picture Image Engineering Conference, 8-6 (1979).
4. C.Y. Suen and T. Radhakrishnan. Recognition of Hand-drawn Flowchart, IJCPR, pp. 424-428 (1976).
5. W.C. Lin and J.H. Pun. Machine Recognition and Plotting of Hand-sketched Line Figures, I.E.E.E. Trans., SMC-8, pp. 52-57 (1978).
6. Kato, Iwase, Yoshida and Tanahashi. Interactive Hand-drawn Diagram Input System, Conf. on PRIP, pp. 544-549 (1982).
7. Murase, Wakahara and Umeda. Investigations of a recognition algorithm for hand-sketched line figures by tablet input, Technical Report of I.E.C.E., PRL81-69 (1981).
8. Shikano and Yoshida. Language processing in the machine recognition of conversation voices, Trans. I.E.C.E. (D), J61-D, 4, pp. 253-260 (Apr. 1978).