Pattern Recognition, Vol. 19, No. 2, pp. 147–160, 1986.
Printed in Great Britain.

0031-3203/86 $3.00+.00
Pergamon Press Ltd.
Pattern Recognition Society

ONLINE HAND-SKETCHED FIGURE RECOGNITION

Hiroshi Murase and Toru Wakahara

Communication Principles Research Section 3, Musashino Electrical Communication Laboratories, Nippon Telegraph and Telephone Corporation, 3-9-11, Midori-Cho Musashino-Shi, Tokyo, 180, Japan

(*Received 26 June 1985; in revised form 20 September 1985*)

Abstract—An online algorithm capable of recognizing hand-sketched symbols such as those used in flowcharts is presented. The algorithm requires no indication of symbol segmentations and no restrictions on the stroke sequence of symbols. The algorithm has three steps: (1) candidate figure extraction for each symbol based on a graph search and distance calculation between candidate figures and reference patterns, (2) selection of the symbol sequence which minimizes the total sum of these distances, (3) connection rules application.

A recognition test performed on 100 hand-sketched flowcharts and block diagrams produced a recognition rate of 96.1%.

Line figures Hand-sketch Symbol recognition Segmentation Graph Data tablet Flowchart Logic circuit diagram

1. INTRODUCTION

Considerable work has been carried out on techniques which realize smooth interaction with computers on the basis of hand-sketched line figures.[1, 2] In combination with character recognition techniques, line figure recognition techniques have 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. The recognition algorithms for hand-sketched line figures can be roughly classified into two groups: offline types, which recognize already hand-sketched figures based on facsimile-input data[3–6]; and online types, which recognizes line figures as they are being hand-sketched on a tablet.[7–9] This paper treats online type recognition. Kato *et al.*[9] proposed an algorithm which can distinguish between stroke types (arcs or straight lines). However, it cannot recognize symbols. The algorithms proposed by Shimizu *et al.*[7] and by Lin *et al.*[8] take stroke-matching approaches to recognizing symbols. However, they imposed several restrictions on figure sketching: (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. These restrictions result in a rather clumsy figure sketching procedure.

This paper proposes an online algorithm based on the stroke sequence independent matching method for recognizing hand-sketched line figures, which requires (1) no indication of symbol segmentations, and (2) no restrictions on the number and order of strokes.

We develop three key ideas; candidate lattice search[10] that achieves symbol recognition and segmentation simultaneously, top-down result verification by symbol connection rules, and graph search technique application to stroke sequence independent matching.

The recognition algorithm consists of the following three parts.

(1) *Extraction of candidate figures*. 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 patterns are calculated through inter-stroke DP (Dynamic-Programming) matching.

(2) *Candidate lattice generation*. For all the extracted candidate figures, a table of matched symbols and calculated distances (hereafter referred to as a candidate lattice) is described. The candidate lattice follows the idea of the phonological lattice used in voice recognition.[11] By searching this candidate lattice for the optimum sequence of candidates which minimizes the total sum of distances along the sequence, we obtain a tentative recognition result. The recognition of the entire figure is performed simultaneously with the segmentation of symbols.

(3) *Application of connection rules*. Connective relations among symbols in the tentative recognition results are established. If any 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. These processes proceed iteratively.

Recognition tests were performed using 100 hand-sketched flowchart and block diagram samples. The obtained recognition rate was 96.1%.

2. INPUT AND REPRESENTATION OF HAND-SKETCHED FIGURE

Objects to be recognized here are figures composed of symbols such as "magnetic disk" and "terminal" and line segments connecting the symbols. The set of symbols used in our experiment is shown in Fig. 1. These are symbols for flowcharts or block diagrams defined as JIS (Japanese Industrial Standard) C6270.

Five persons drew 100 samples of flowcharts and block diagrams on a data tablet. The following rules were in effect.

(1) No indications of segmentations between symbols and line segments are to be given.

(2) Sizes and positions of symbols are arbitrary (free format).

(3) There are no restrictions on the stroke sequence of each symbol.

The tablet is an electro-magnetic coupling type, which samples pen movement in the XY-coordinate system. Pen up-down signals are detected by a micro switch in a penholder. The tablet has a resolution of 0.1 mm and a sampling rate of 100 samples/second.

The nth stroke is defined as a sequence of the coordinate values along the nth continuous pen movement on the data tablet. An example of sampled hand-sketched flowcharts is shown in Fig. 2. The numerals in the figure indicate the stroke order.

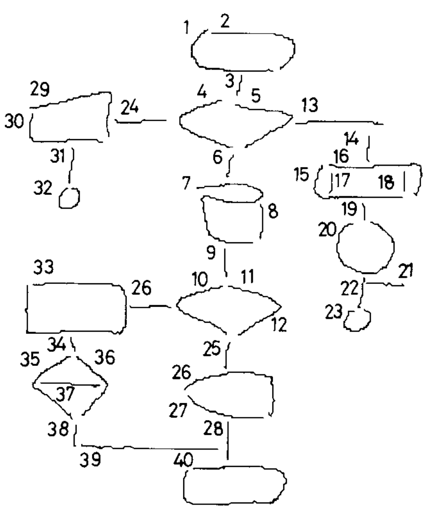

Fig. 2. Example of input pattern.

Coordinate sequences are sampled so that every stroke is represented by a sequence of a fixed number of coordinates.

3. STROKE SEQUENCE INDEPENDENT MATCHING

Input figures are usually composed of many symbols and linkage line segments. However, the following concentrates on the case where an input figure describes only one symbol. The matching method proposed here is independent of the stroke sequence. The

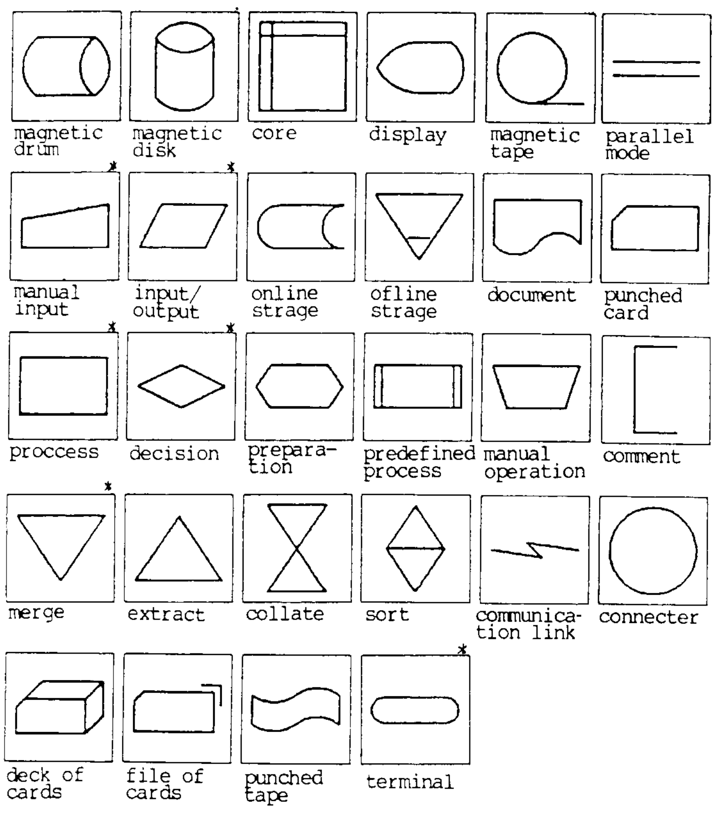

Fig. 1. List of symbols (JIS C6270).

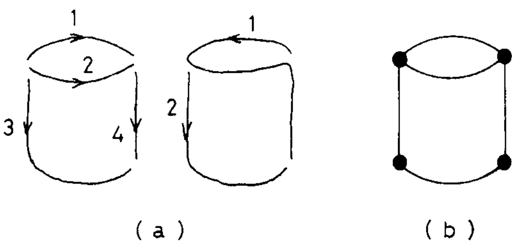

Fig. 3. (a) Example of stroke sequence. (b) feature points.

case of multiple symbols and linkage line segments is treated in the next section.

Let us call a figure drawn on the tablet and represented in the above form a given figure. The method is based on a top-down approach. That is, assuming a symbol (called a "supposed symbol" hereafter), the method investigates the likelihood of the given figure to be the supposed symbol. If the conditions are satisfied, the given figure is regarded as a candidate for the supposed symbol, and distances between corresponding input strokes and standard strokes for the supposed symbol are calculated to give the likelihood.

3.1. *Representation of standard symbols*

A preparatory experiment was conducted in order to decide how to represent standard symbols. We asked 20 people to draw the symbols shown in Fig. 1 without giving them any restrictions. The analysis of the collected data shows that there is no regularity in the stroke sequence (the number and order of strokes). However possible end points of strokes are limited to a finite number of places. We call these finite places feature points. Examples of stroke sequences are shown in Fig. 3(a) and the feature points of the symbol "mag. disk" in Fig. 3(b).

Considering the possible increase in the number of objects to be recognized, it is desirable to simplify the representation of the symbols. The way we adopt to achieve this is to express each symbol in terms of a set of strokes with their end points corresponding to the feature points with each stroke being a straight line segment or an arc segment. 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.

The development of the feature points is done by the following procedure. First, we collected symbol data written by the 20 people under the condition that the stroke sequence of a symbol was free of restrictions. Second, end points (pen up and pen down points) of the written symbols were marked. Third, a symbol was divided to straight line segments and arc segments. Finally, we represented feature points in coordinate values, thus, obtained the representation of the standard symbols. A representation of the symbol "mag. disk" is given in Fig. 4.

3.2. *Correspondence between end points and feature points*

The method assumes that a given figure is a supposed symbol. If this assumption is valid, any end point of the input stroke corresponds to one of the feature points of the supposed symbol. In order to find correspondence between points, we first normalize the supposed symbol in both vertical and horizontal directions so that the given figure and the supposed symbol match in the maximum length in both directions. Then, the end points contained in the input strokes are matched with the respective nearest feature points. For the input strokes in Fig. 5(a), we obtain the correspondence as shown in Table 1 when the supposed symbol is "mag. disk" in Fig. 5(b).

Due to variations in handwriting, the end points' positions of a certain input stroke vary. They stay, however, within a certain range. If either of the end points' positions is not within the range of any of the feature points of the supposed symbol, the given figure is judged to be different from the supposed symbol. In this case, another symbol is assumed and the above process is repeated.

3.3. *Generation of candidate stroke series*

A given figure is examined topologically to determine whether or not it is appropriate as a supposed symbol. If it is appropriate, a candidate stroke series is enumerated in accordance with the expected pen movement along the supposed symbol based on the point-to-point correspondence.

We assume that the given figure is composed of N strokes. In addition, based on the standard symbol representation (see Section 3.1) the supposed symbol is

```
Representation format:

    NAME="symbol name" ("list of labels")

    "label":=("stroke type (arc or line)"
             ,"starting point coordinate"
             ,"ending point coordinate"
             [,"middle point coordinate"])

example

    NAME="magnetic disk" (A,B,C,D,E)

        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:=(line,(0,1),(0,5))
        E:=(line,(4,1),(4,5))
```

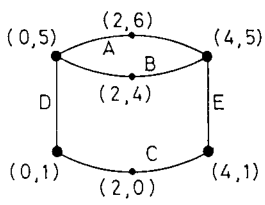

Fig. 4. Representation of standard symbols (e.g. "mag. disk")

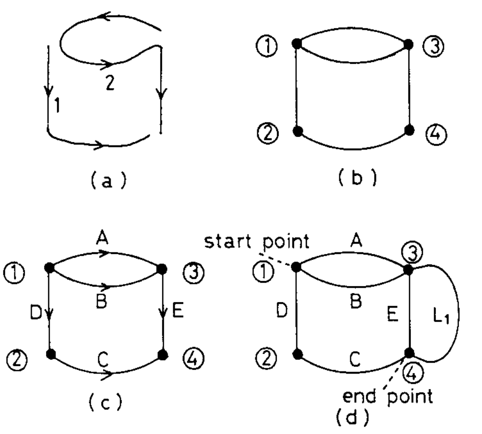

Fig. 5. (a) Input strokes, (b) supposed symbol, (c) directed graph, (d) graph in search.

expressed as a directed graph that has nodes and branches corresponding to feature points and line/arc segments of the supposed symbol, respectively. A directed graph is used to indicate the direction of the pen movement on a branch as positive. If we use "v" for a node name and "a" for a branch name, then a graph can be represented by a connection matrix $D = (d_{va})$, where

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

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

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

In order for the given figure to be appropriate for the supposed symbol there must be a path consisting of N strokes which passes all the branches of the graph of the supposed symbol once and which gives the correspondence between the end points of the strokes and feature points of the supposed symbol. Thus, the problem is reduced to a graph theory problem of finding an N stroke path with the restriction of the end point correspondence obtained in Section 3.2. To simplify the problem further, we add a branch between any two consecutive strokes to represent a pen movement between the strokes, and change the problem to that of finding an Euler path, i.e. one stroke path. The processing steps become as follows:

(1) Add to the original directed graph a new branch L_n (hereafter called a fill-in branch) going from the node corresponding to the ending point of the nth stroke ($1 \leq n \leq N-1$) to the node corresponding to the starting point of the ($n+1$)st stroke. An example is given in Fig. 5(d). The connection matrix is given by

Table 1. An example of correspondence between end points and feature points

Stroke number	Starting point	Ending point
1	1	4
2	3	4

$$\begin{array}{c} \\ (1) \\ (2) \\ (3) \\ (4) \end{array} \begin{array}{c} \begin{array}{cccccc} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} & \mathrm{E} & \mathrm{L1} \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ -1 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 0 & -1 & 1 \end{bmatrix} \end{array} .$$

(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 for the start and 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 is not satisfied for a node, the given figure can not be the supposed symbol. In this case, another symbol is assumed until the condition is satisfied.

(3) Next, we find all the Euler paths that satisfy the conditions given below and take them as a candidate stroke series. The following conditions are necessary to ensure that a path will comply with the order of input strokes.

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

(b) A path cannot go through two or more fill-in branches successively.

(c) A path cannot traverse fill-in branches against their directions, but it can traverse other branches in either direction.

In searching for an Euler path the depth first search method is adopted, which is well known as a graph search technique. If the search is successful, the given figure is extracted as a candidate figure for the supposed symbol.

Searching for a path by this method for the example of Fig. 5(b) yields 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 arrow direction and "−B" indicates the traverse of branch B against the arrow direction. It can be seen that the correct correspondence with the input strokes [see Fig. 5(a)] is the second of the candidate stroke series given above.

A set of candidate stroke series for the supposed symbol can be obtained as sequences of branches as explained above. Each branch is either a straight line segment or an arc segment according to the representation of standard symbols (Section 3.1). Thus, for each branch we generate a sequence of coordinate values of a straight line segment or an arc segment and approximate each candidate stroke with a certain number of points. Using this sequence of coordinate values, the distance between a candidate stroke series and an input stroke is calculated.

3.4. *Distance calculation*

The next step uses more detailed shape information of the input strokes. We match the shapes of the candidate stroke series obtained in the topological process and the input strokes by calculating the distance between the shapes.

The distance is defined as the sum of the distance between the corresponding strokes of a candidate stroke series and the input strokes. We assume that each pair of corresponding strokes consists of M points and that their coordinates are $\{(X_m, Y_m), m = 1 \sim M\}$ and $\{(X'_m, Y'_m), m = 1 \sim M\}$.

We use DP-matching based on the Euclidean distance between coordinate points of strokes and the tangential difference at the points. The distance D is

$$D = \min_u \left[\sum_m \{(X_m - X'_{u(m)})^2 + (Y_m - Y'_{u(m)})^2\}^{1/2} + \alpha \cdot h(m, u(m)) \right].$$

Here $u(m)$ represents the correspondence relation between the coordinate points and follows the restrictions:

$$\begin{cases} u(1) = 1 \\ u(M) = M \\ \text{If } u(i) = j, \text{ then } u(i+l) = \{j \text{ or } j+1 \text{ or } j+2\} \end{cases}$$

$h(i, j)$ represents the difference between the tangential direction at (x_i, y_i) and that at (x'_j, y'_j);

$$h(i,j) = \left| \tan^{-1}\left(\frac{Y_{i+1} - Y_i}{X_{i+1} - X_i}\right) - \tan^{-1}\left(\frac{Y'_{j+1} - Y'_j}{X'_{j+1} - X'_j}\right) \right|.$$

"α" is a coefficient of $h(i, j)$ and its value was determined by a preparatory experiment. In the experiment, several different values of α were used in the recognition procedure to recognize the symbol data (the same data as used in 5.2). From this experiment, it is known that the highest recognition rate was obtained when α is 0.004. This value was chosen for α.

These processes (3.2, 3.3, 3.4) are repeated by assuming the existence of all symbols in the dictionary.

4. LATTICE SEARCH INVOLVING CONNECTION RULES

In this section, we discuss an automatic segmentation and recognition of symbols for the case when input figures are composed of multiple symbols and linkage line segments. A system block diagram of the proposed method is shown in Fig. 6. The diagram consists of control blocks and knowledge blocks. We only prepare the knowledge corresponding to recognition objects.

4.1. *Candidate figure extraction*

A candidate figure for a symbol is a subfigure of an input figure, which is possibly equal to the symbol. Extraction of candidate figures is performed in two steps. First, consider a subfigure of the input figure (for example, the subfigure composed of strokes numbered 10, 11 and 12) which satisfies the two conditions;

(1) the strokes composing the subfigure have succeeding stroke numbers,

(2) the strokes do not exceed the possible maximum number of strokes of candidate stroke series (if the object symbol is "mag. disk", the maximum number is 5,

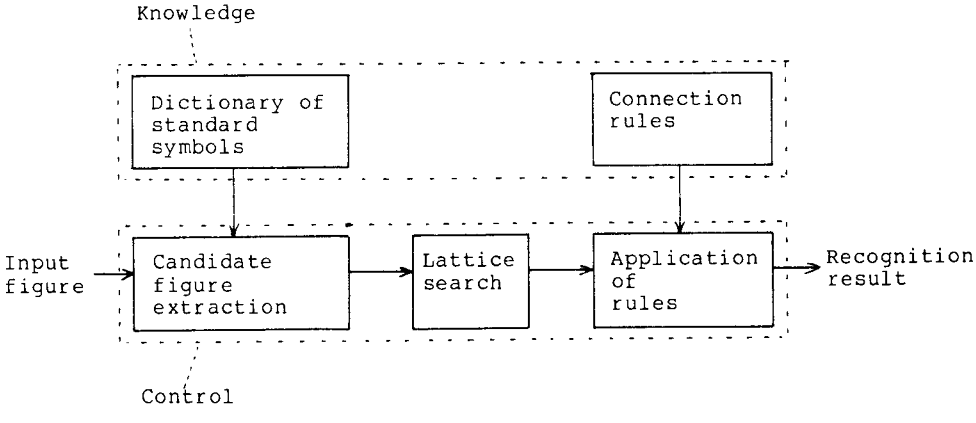

Fig. 6. System block diagram.

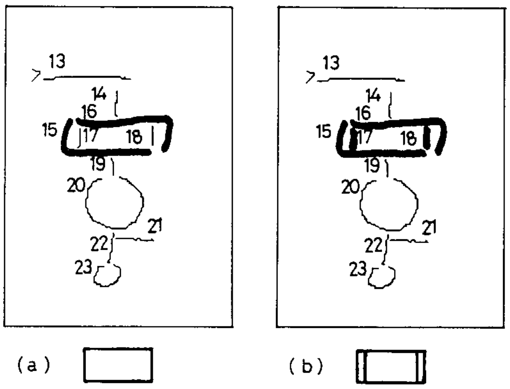

Fig. 7. Candidate figures for symbols.

see Section 4.3). Second, the considered subfigure is recognized as a symbol using the stroke sequence independent matching method discussed in Section 3. The method seeks the correspondence between the stroke end points of the subfigure and the feature points of a supposed symbol, and based on the resultant correspondence a path in the graph (the supposed symbol) is searched (see Section 3.2). If there is no contradiction in a result of these processes, the given subfigure becomes a candidate figure. The processes are repeated for all other subfigures until all the candidate figures are extracted.

As examples, four candidate figures for the symbol "process" are extracted in Fig. 7(a), and one candidate for a "predefined process" is extracted in Fig. 7(b).

4.2. *Candidate lattice generation*

A candidate lattice is a table which shows what candidate figures or what parts of strokes in an input figure have been extracted. The following gives a generation procedure of a candidate lattice. First, we extract candidate figures for symbols from the input figure (see Section 4.1). Since a candidate figure for a symbol is obtained as a sequence of stroke numbers, we assign the names of the symbols to the corresponding stroke number sequences in a candidate lattice. Then, we calculate the distances for the candidate figures (see Section 3.4) and register the distances in the lattice. Since a linkage line segment is drawn in one stroke, a "linkage line segment" is also registered as a candidate figure for every stroke of an input figure. As an example, a part of the candidate lattice generated for the input figure shown in Fig. 2 is given in Fig. 8.

4.3. *Tentative search through lattice*

Based on the candidate lattice, we determine the optimum figure sequence spanning the entire input figure and accomplish the segmentation and recognition of the symbols in the input figure.

We extract from the lattice all the possible sequences of candidate figures which connect between the first stroke and the last stroke of the input figure. From among these sequences, the optimum figure sequence is selected. The optimum sequence is that which minimizes the following objective function S:

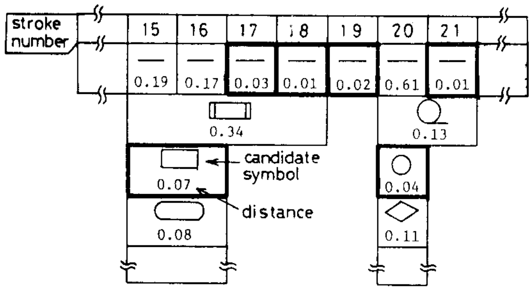

Fig. 8. Candidate lattice.

$$S = \sum_{\substack{\text{candidate figure} \\ \in \text{ sequence}}} (\text{distance of a candidate figure}).$$

where S is the sum of the distances along a candidate figure path. The optimum path is searched effectively by a DP (Dynamic Programming) procedure which is often applied to serial problems. The figure sequence corresponding to the optimum pass is called the "tentative recognition result". In the case of the candidate lattice of Fig. 8, thick line squares show the optimum figure sequence, and the tentative recognition result is shown in Fig. 9.

4.4. *Description of connective relations among symbols*

An input figure is composed of symbols and linkage line segments connecting the symbols. The connective relations among symbols in the tentative recognition result can be represented as follows.

(1) Each symbol or linkage line segment is assigned an identification sign (see Table 2), and has a pointer which shows the identification signs of symbols and linkage line segments connected to it.

(2) Each symbol or linkage line segment has stroke numbers which indicate the position, size and shape of the subfigure in the input figure.

(1) is necessary in applying connection rules and (2) is necessary for making a fair copy of the recognition result. Table 2 shows the connection description of the tentative recognition result shown in Fig. 9. Identification signs in Table 2 correspond to the signs in Fig. 9. From this description, we know, for example, that symbol C ("decision") is composed of the strokes numbered 4 and 5, and is connected to the linkage line segments B, D, H and R.

4.5. *Application of connection rules*

There are several illegal symbol connection re-

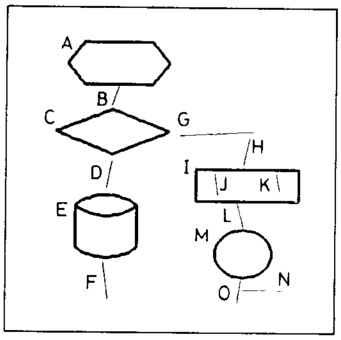

Fig. 9. Tentative recognition result.

Table 2. Description of connective relation among symbols

Stroke number	id sign	Symbol name	Connection pointer
1, 2	A *	preparation	B
3	B	line	A, C
4, 5	C	decision	B, D, H, R
6	D	line	C, E
7, 8	E	magnetic disk	D, F
9	F	line	E, G
10, 11, 12	G	decision	F, S, T
13	H	line	C, I
14	I	line	H, J
15, 16	J *	process	J, K, L, M
17	K	line	J, J
18	L	line	J, J
19	M	line	J, N
20	N *	connector	M, O, P
21	O	line	N
22	P	line	N, Q

lations in figures such as flowcharts. For example, the following is obviously inconsistent with flowchart structures: (1) "terminal" located at a branch of a process flow, (2) a linkage line segment not connecting between two symbols. The connection rules are registered in the dictionary as "if-then rules". In the event that illegal symbol connection relations are found in the tentative recognition result by referring to the dictionary, the responsible symbols are removed from the lattice. Then, the figure sequence optimization process is applied to the modified lattice to give the optimum figure sequence. These processes are repeated until there are no illegal connection relations.

For an example of flowcharts, we used the following 52 connection rules.

R1: If [the number of linkage line segments connected to a "terminal" is greater than one]
then [the "terminal" is removed from the lattice].

R2: If [both end points of linkage line segment are connected to one "process"]
then [the "process" is removed from the lattice].

...

Ri: If [...]
then [...].

...

R52: If [the size of a "terminal" is very small compared with the average size of other symbols]
then [the "process" is removed from the lattice].

In Table 2, there are three violations of these rules, which are indicated by "*". These symbols are removed and another symbol sequence is selected from the candidate lattice in Fig. 8. In this example, the lattice modification and the sequence optimization are repeated twice. The respective tentative recognition results are shown in Table 3. Figure 10 shows a fair output of the final recognition result.

Table 3. Tentative recognition results

Stroke number	1st tentative recognition result	2nd tentative recognition result	3rd tentative recognition result (the final)
.			
.			
10			
11	decision	decision	decision
12			
13	line	line	line
14	line	line	line
15	process (*)	terminal (*)	
16			predefined process
17	line	line	
18	line	line	
19	line	line	line
20	connector (*)	magnetic tape	magnetic tape
21	line		
22	line	line	line
.			
.			

(*) illegal symbol

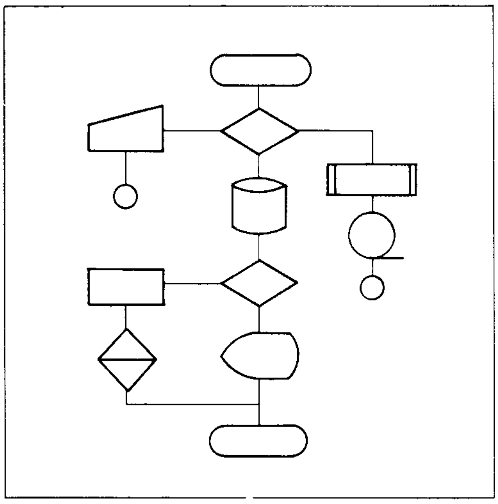

Fig. 10. Output of recognition result.

5. EXPERIMENTS

5.1. *Recognition rate*

An experiment of recognizing hand-sketched line figures is conducted based on the proposed recognition algorithm. The recognition objects are one hundred flowcharts and block diagrams composed of 5 to 9 symbols and linkage line segments (see Section 2). The recognition rate was 96.1%. Examples of errors are described in Section 5.3.

The recognition rate is defined as

$$\text{Recognition rate} = \frac{\text{the number of correctly recognized symbols}}{\text{the number of symbols in the input figure}}.$$

5.2. *Evaluation of stroke sequence independent matching*

The stroke sequence independent matching method described in Section 3 allows arbitrary stroke order and number of symbols owing to the dynamic generation of candidate stroke sequences corresponding to an input stroke sequence. This section discusses the effects of the method applied to unrestrictedly drawn symbol data. We asked 20 people to draw symbols of nine different types four times without any restrictions to obtain 720 symbols altogether. The symbols used here most often appear in flowcharts and are marked by "*" in Fig. 1.

As a result of the analysis of the symbol data, we obtained the most frequently occurring sequences for "process" and "mag. disk". These sequences are shown in Fig. 11. Their occurrence rates were 67% and 22%, respectively. Here, we present a simple matching method for the purpose of comparing it with the stroke sequence independent matching method. The method proceeds as follows:

(1) As the standard stroke sequence for each symbol, the most frequently appearing sequence in drawing the symbol is selected.

(2) The distances between an input stroke sequence and the standard stroke sequences for all the symbols are calculated by the inter-stroke DP matching described in Section 3.4.

(3) The symbol yielding the minimum distance is selected as the recognition result.

Since this method uses only one standard stroke sequence as a reference pattern for each symbol, it is difficult for the recognition rate for unrestrictedly drawn figures of the symbol to be higher than the percentage of the appearance of the standard stroke sequence, e.g. 67% for "process" and 22% for "mag. disk". This is because the percentage is equal to the rate of the coincidence between the input and the standard stroke sequences (see Fig. 11). In the experiment based on the above data, a recognition rate of 46.4% was obtained, with the average computation time for one symbol being 0.6 s at a computation speed of 1 MIPS.

On the other hand, standard stroke sequences are dynamically generated using the end point positions of input strokes in the stroke sequence independent matching method. The standard stroke sequences can be generated for all combinations of the positions of the end points; 632 sequences for "process" and 7996 for "mag. disk". These are shown in Fig. 11. The ability of the method is equivalent to that of the simple matching method having all of these sequences as the standard stroke sequences.

The stroke sequence independent matching method proposed here only generates a very limited number of sequence variations for input strokes, because of the restriction imposed by the end point positions of input strokes. The average number of generations for the above data is 1.2 for "process", 2.8 for "mag. disk". Hence, the number of distance calculations is two digits below as compared with the simple matching method having all the standard sequence variations. In our experiment a recognition rate of 97.2% was attained, with the computation time being 1.2 s as shown in Table 4.

5.3. *The effect of connection rules in lattice search*

The lattice search method removes the procedure for the indication of segmentations between symbols. In the event that the lattice search does not use connection rules but only uses distance information based on symbol shapes, it only produces a first tentative recognition result and no corrections are made regarding the result. Thus, the method is likely to cause segmentation errors such as that shown in Fig. 12(a), where "core" is mistaken for one "process" and two "linkage line segments". In addition, it cannot distinguish precisely between similarly shaped symbols such as shown in Fig. 12(b), where "terminal" is mistaken for "preparation". The confusion matrix without using rules is shown in Fig. 14(a). To solve the

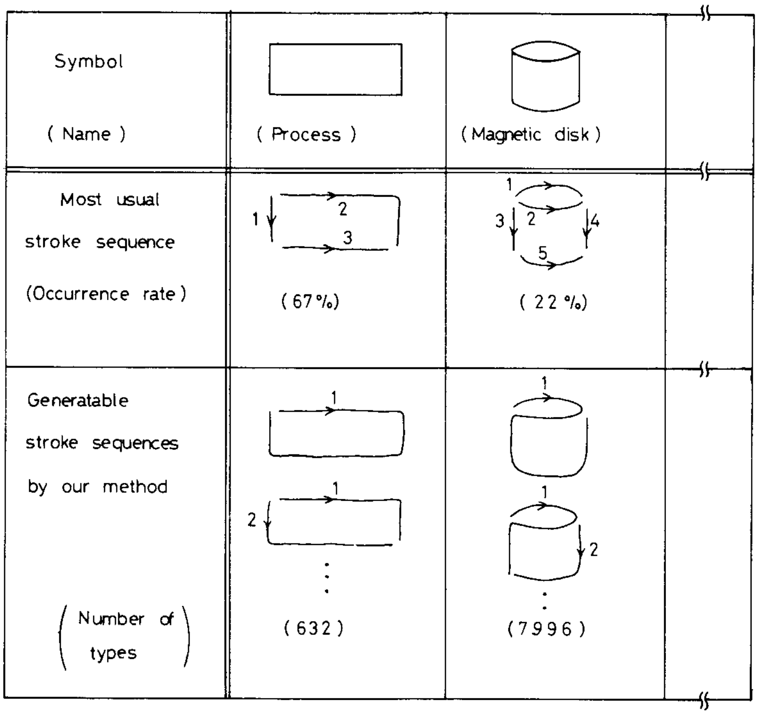

Fig. 11. Stroke sequence variations.

Table 4. Recognition rate and computation time

	Simple matching method	Stroke sequence independent matching method
Recognition rate	46.4%	97.2%
Computation time	0.6 s	1.2 s

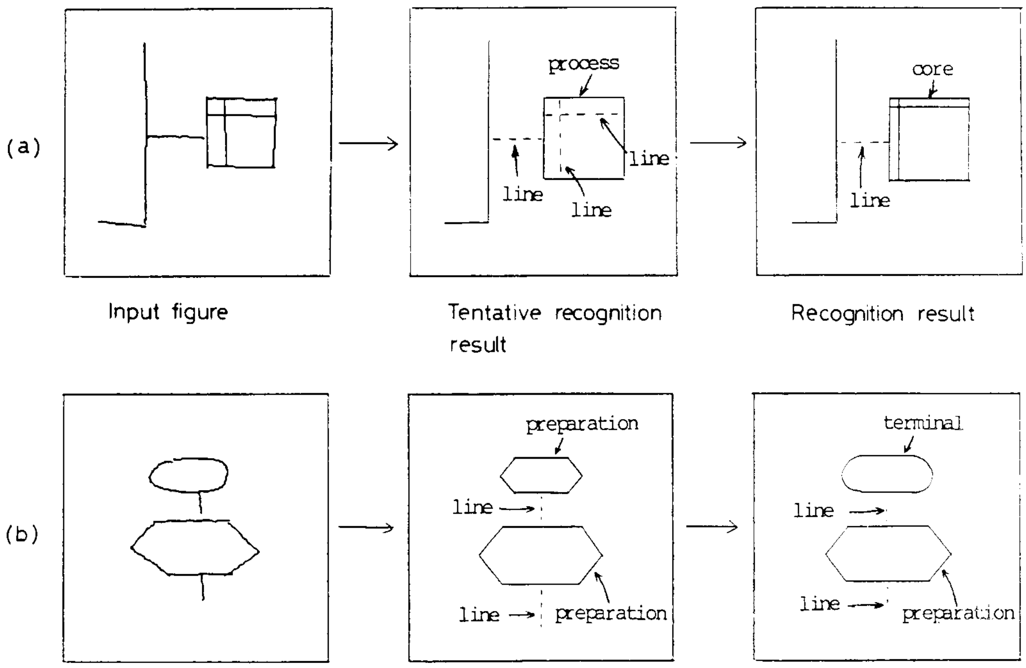

Fig. 12. Examples of error correction by rules.

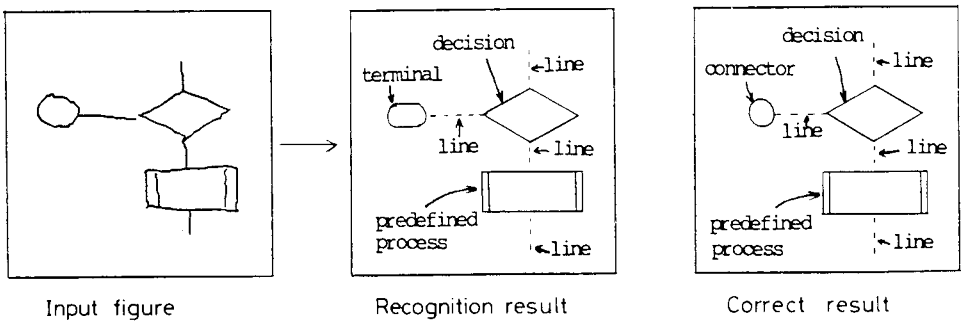

Fig. 13. Examples of error occurring in spite of applying the rules.

problem, 52 connection rules are introduced. The rules can be classified into the following four types.

(1) Type 1: If both end points of a linkage line segment are connected to the same symbol, the symbol is removed from the candidate lattice. (There are 28 rules of this type.)

(2) Type 2: If the number of linkage line segments connected to specific symbols (such as "preparation", "decision") is equal to one, the symbol is removed. (There are six rules of this type.)

(3) Type 3: If the number of linkage line segments connected to specific symbols (such as "terminal") is greater than one, the symbol is removed. (There are two rules of this type.)

(4) Type 4: If the sizes of specific symbols (such as "terminal") are very small compared with the average size of other symbols, the symbol is removed. (There are 16 rules of this type.)

The effect of each of the four types of rules was investigated in recognition experiments. The relationship between recognition rate and rule applied is shown in Fig. 15. It was shown from the experiments that the type 1 rules corrected segmentation errors such as shown in Fig. 12(a), and improved the recognition rate from 89.8% to 91.6%; the type 2 rules corrected symbol recognition errors such as shown in Fig. 12(b), and improved the recognition rate from 91.6% to 94.4%. When all four rule types were applied,

Input symbols \ Recognition results	PROCESS	DECISION	PREPARATION	PREDEFINED PROCESS	MANUAL OPERATION	MERGE	COLLATE	SORT	MANUAL INPUT	INPUT/OUTPUT	ONLINE STRAGE	OFFLINE STRAGE	DOCUMENT	PUNCHED CARD	DECK OF CARDS	PUNCHED TAPE	MAGNETIC TAPE	MAGNETIC DRUM	MAGNETIC DISK	CORE	DISPLAY	COMMUNICATION LINK	CONNECTER	TERMINAL	segmentation error
PROCESS	76				2				1															1	
DECISION		60																							
PREPARATION			20																						
PREDEFINED PROCESS				19																					1
MANUAL OPERATION					20																				
MERGE						19																			1
COLLATE							20																		
SORT								18																	2
MANUAL INPUT									18							1									1
INPUT/OUTPUT										18				1										1	
ONLINE STRAGE			3								14													3	
OFFLINE STRAGE												20													
DOCUMENT													20												
PUNCHED CARD			1											18											1
DECK OF CARDS															19										1
PUNCHED TAPE													1			19									
MAGNETIC TAPE																	20								
MAGNETIC DRUM																		58							2
MAGNETIC DISK																			60						
CORE																				11					9
DISPLAY			4																		56				
COMMUNICATION LINK																						60			
CONNECTER			18																		2		40		
TERMINAL			18										3			1					1			37	

Fig. 14(a). A confusion matrix without using rules.

Input symbols \ Recognition results	PROCESS	DECISION	PREPARATION	PREDEFINED PROCESS	MANUAL OPERATION	MERGE	COLLATE	SORT	MANUAL INPUT	INPUT/OUTPUT	ONLINE STRAGE	OFFLINE STRAGE	DOCUMENT	PUNCHED CARD	DECK OF CARDS	PUNCHED TAPE	MAGNETIC TAPE	MAGNETIC DRUM	MAGNETIC DISK	CORE	DISPLAY	COMMUNICATION LINK	CONNECTER	TERMINAL	segmentation error
PROCESS	77				2				1																
DECISION		60																							
PREPARATION			20																						
PREDEFINED PROCESS				20																					
MANUAL OPERATION					20																				
MERGE						20																			
COLLATE							20																		
SORT								20																	
MANUAL INPUT									18							1									1
INPUT/OUTPUT										19				1											
ONLINE STRAGE			3								15			2											
OFFLINE STRAGE												20													
DOCUMENT													20												
PUNCHED CARD														19											1
DECK OF CARDS															19										1
PUNCHED TAPE													1			19									
MAGNETIC TAPE																	20								
MAGNETIC DRUM																		60							
MAGNETIC DISK																			60						
CORE																				20					
DISPLAY			1																		58			1	
COMMUNICATION LINK																						60			
CONNECTER									1												4		52	3	
TERMINAL													3			1					2			54	

Fig. 14(b). A confusion matrix using rules.

a recognition rate of 96.1% was achieved. The confusion matrix using 52 rules is shown in Fig. 14(b). Segmentation errors are responsible for 0.4% and symbol recognition errors (such as shown in Fig. 13) for 3.5% of the total errors recognized. In case that higher level rules cannot help, a recognition procedure based on more detailed feature analysis must be included in the whole process. Some of the detailed features will be curvatures, positions of corners, etc.

6. DISCUSSION

6.1. *Computation time*

According to simulation experiments on a minicomputer (DG-MV8000, 1 MIPS) the computation time for the example of Fig. 2 was 100 s by FORTRAN. The relationship between the computation time and the number of strokes contained in an input figure is shown in Fig. 16. It is known that the amount of

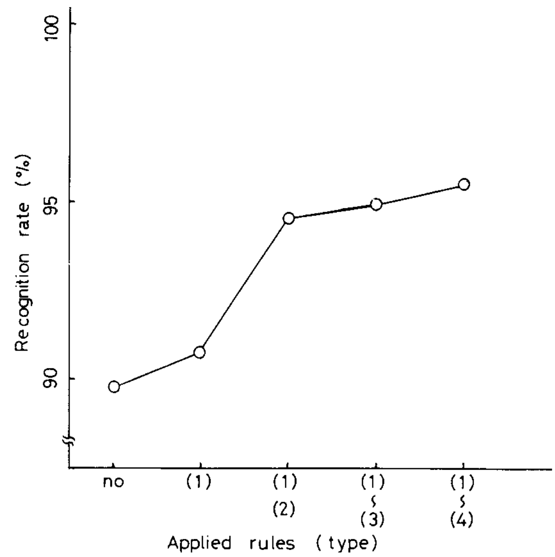

Fig. 15. Recognition rate vs applied rules.

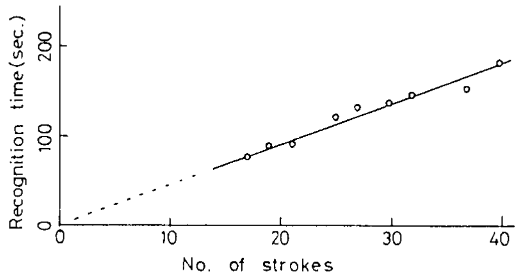

Fig. 16. Recognition time vs No. of strokes.

processing is proportional to the number of strokes in this method. Here, since 80% of the computation time is consumed by calculations of distances (DP matching), a reduction in computation time can be attained easily by DP matching hardware often used in speech recognition. The candidate lattice search with the connection rules can be made by a simple symbol operation. Consequently, the amount of the processing is minimized.

6.2. *Application to logic circuit diagram*

The method described in this paper is separated into knowledge and control as shown in Fig. 6. The method can recognize any line figures composed of symbols and linkage line segments by preparing knowledge composed of a symbol dictionary (such as that in Fig. 4) and connection rules. We applied the method to logic circuit diagrams. Twenty-six symbols ("AND", "OR", "NAND", "NOR", "EXCLUSIVE-OR", "NOT", and their rotations by 90, 180, 270 degrees, "CONNECT" and "BLOCK") are registered in the symbol dictionary. There are seven connection rules 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'. In an experiment on 40 logic circuit diagrams, a recognition rate of 93.2% was obtained. An input figure and the recognition result are shown in Fig. 17.

7. CONCLUSIONS

An online algorithm for recognizing unrestrictedly hand-sketched line figures such as flowcharts, block diagrams and logic circuit diagrams has been proposed and its performance has been verified by experiments. It has been shown that:

(1) The stroke sequence independent matching is effective in handling an arbitrary number and order of input figure strokes. This is achieved by representing the shape structure of each symbol in terms of a directed graph and by searching the graph for paths representing shape structures.

(2) The segmentation and recognition of individual symbols in input figures can be simultaneously performed by searching for an optimum symbol sequence, i.e. a sequence which minimizes the total sum of distances in the lattice of candidate figures.

(3) Highly accurate recognition can be achieved by adding a correction process through the application of connection rules imposed on input figures to the tentative recognition result. In an experiment on 100 figures, the recognition rate was 96.1%. The rate was 89.5% without using the rules.

The method proposed in this paper is highly general because it is easy to expand a range of standard symbols with appropriate connection rules. Therefore, it has many applications in the area of online recognition of a wide range of hand-sketched line figures. Also, its ability to permit stroke sequence independent drawing and no indication of symbol segmentations is very promising in a future user-friendly figure input system. Further research is needed to incorporate online handwritten character recognition technique in this line figure recognition method. Especially, application of syntactic and semantic knowledge is necessary to deal with roughly handwritten documents.

SUMMARY

Line figure recognition techniques have many applications in areas such as automatic fair copying of

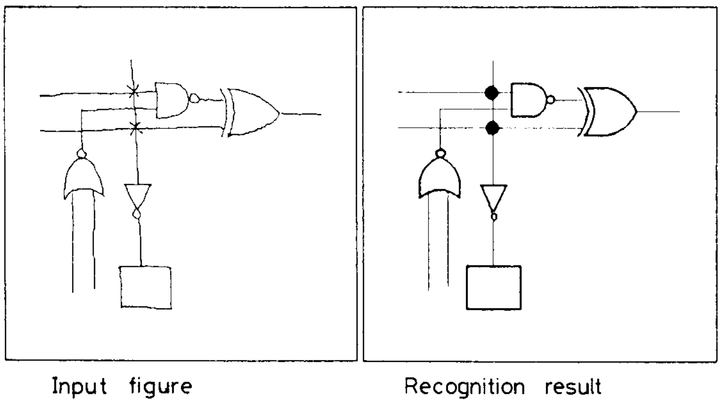

Fig. 17. Recognition of logic circuit diagram.

hand-written documents containing flowcharts or block diagrams, and automatic programming from hand-sketched flowchart data. This paper treats the online type recognition method which recognizes line figures as they are being hand-sketched on a tablet. The conventional methods impose several restrictions on figure sketching: (1) the user must indicate segmentations between symbols, and (2) each symbol must be drawn in the predetermined number and order of strokes. These restrictions result in a rather clumsy figure sketching procedure.

This paper proposes an online algorithm for recognizing hand-sketched line figures, which requires (1) no indication of segmentations, and (2) no restrictions on the stroke sequence (number and order of strokes). The algorithm is applied to the recognition of symbols used in flowcharts or block diagrams.

The recognition algorithm consists of the following three parts.

(1) *Extraction of candidate figures.* 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 patterns are calculated through inter-stroke DP (Dynamic-Programming) matching.

(2) *Candidate lattice generation.* For all the extracted candidate figures, a table of matched symbols and calculated distances are described. This table is called a candidate lattice. 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 recognition result. The recognition of the entire figure is performed simultaneously with the segmentation of symbols.

(3) *Application of connection rules.* Connective relations among symbols in the tentative recognition results are established. If any 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. These processes proceed iteratively.

The performance of our method has been verified by experiments. It has been shown that:

(1) The stroke sequence independent matching is effective in handling an arbitrary number and order of strokes of input figures.

(2) The segmentation and recognition of individual symbols in input figures can be simultaneously performed by searching for the optimum symbol sequence in the lattice of candidate figures.

(3) Highly accurate recognition can be achieved by adding a correction process through the application of connection rules. In an experiment involving 100 figures, the recognition rate was 96.1%. The rate was 89.5% without using the rules.

The method proposed in this paper is highly general and has many applications in the area of online recognition of hand-sketched line figures. It improves the load of operators by permitting free stroke sequence drawing and no indication of segmentations between symbols.

Acknowledgements—The authors would like to thank Dr. Noriyoshi Kuroyanagi and Dr. Isao Masuda for their useful suggestions and encouragement. They would also like to express their appreciation to Dr. Michio Umeda for their valuable discussions. They are also indebted to Dr. Noriyoshi Osumi for his advice in preparing this paper.

REFERENCES

1. I. E. Sutherland, Sketchpad: a man machine graphical communication system, *Spring Joint Computer Conf.*, pp. 329–346 (1963).
2. J. F. Jarvis, The line drawing editor: schematic diagram editing using pattern recognition techniques, *Comput. Graph. Image Process.* **6**, 452–484 (1977).
3. S. Ito, Automatic input of flow chart in document image, *Int. Conf. on Software Engineering*, pp. 319–328 (1982).
4. T. Sato and A. Tojo, Recognition and understanding of hand-drawn diagram, *Int. Conf. on Pattern Recognition*, pp. 647–677 (1982).
5. C. Y. Suen and T. Radhakrishnan, Recognition of hand-drawn flowchart, *Int. Joint Conf. on Pattern Recognition*, pp. 424–428 (1976).
6. Y. Fukada, A primary algorithm for the understanding of logic circuit diagrams, *Pattern Recognition* **17**, 125–134 (1984).
7. S. Shimizu, S. Nagata, A. Inoue and M. Yoshida, Logic circuit diagram processing system, *Int. Conf. on Pattern Recognition*, pp. 717–719 (1982).
8. W. C. Lin and J. H. Pun, Machine recognition and plotting of hand-sketched line figures, *IEEE Trans.* **SMC-8**, 52–57 (1978).
9. O. Kato, H. Iwase, M. Yoshida and J. Tanahashi, Interactive hand-drawn diagram input system, *Conf. on Pattern Recognition and Image Processing*, pp. 544–549 (1982).
10. H. Murase, T. Wakahara and M. Umeda, Online hand-sketched line figure recognition by candidate lattice method with connection rules, *Trans. IECE*, **J67-D3**, 273–280 (1984) (in Japanese).
11. P. Rovner, B. Nashwebber and W. A. Woods, Control concepts in a speech understanding system, *IEEE Trans.*, **ASSP-23** (1), 136–140 (1975).

About the Author—HIROSHI MURASE was born in Aichi, Japan on 28 September 1955. He received his B.E. and M.E. degrees in Electrical Engineering from Nagoya University, Aichi, Japan in 1978 and 1980, respectively. He has been engaged in pattern recognition research at NTT Musashino Electrical Communication Laboratories since 1980. He is a member of the Institute of Electronics and Communication Engineers of Japan and the Audio-Visual Information Research Group of Japan.

About the Author—TORU WAKAHARA was born in Gifu, Japan on 30 January 1952. He received his B.E. and M.E. degrees in Applied Physics from Tokyo University, Tokyo, Japan in 1975 and 1977, respectively. He has been engaged in image analysis and pattern recognition research at NTT Musashino Electrical Communication Laboratories since 1977. He is a member of the Institute of Electronics and Communication Engineers of Japan and the Audio-Visual Information Research Group of Japan.