Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

# Mathematical Figure Recognition for Automating Production of Tactile Graphics

Noboru Takagi
Department of Intelligent Systems Design Engineering
Toyama Prefectural Univeristy
Toyama, Japan
noboru.takagi@ieee.org

***Abstract*—A computer-aided system for transformation of mathematical figures into tactile graphics is useful for the visually impaired when they learn mathematics and science. To develop such a system, a study of mathematical figure recognition techniques is needed. It is natural to assume that (1) a mathematical figure includes characters and mathematical formulas to explain lines or curves that are graphical expressions of functions or relations, and (2) graphs are sometimes drawn using broken lines. Under these assumptions, this paper discusses methods of (1) separating a mathematical figure into the character components and the graph components, and (2) extracting broken line components and grouping these components into each broken line.**

***Index Terms*—Pattern Recognition, Graph Recognition, Image Processing, Tactile Graphs, Tactile Teaching Material**

## I. Introduction

Graphs are frequently used to present functions or relations in mathematics and science. But, most of these graphs are in visual form, they cannot be utilized by visually impaired users. Through tactile graphics, pictures can be understood by the visually impaired. This is because tactile graphics are produced by raised patterns which can be felt with the fingertips. Although tactile graphics are useful for visually impaired students when they learn mathematics and science, in 80% of Japanese schools for the visually impaired, there is no department to produce tactile teaching material[7]. Usually, teachers produce most tactile graphics using less intelligent computer-aided systems. So, a better computer-aided system of making tactile graphics is needed.

Graph recognition techniques are necessary to develop such a system. There is some research on graph recognition. Aso et al.[6] studied a graph recognition method, and in their method, graphs must satisfy many assumptions. For example, a graph has to be drawn inside a rectangular area that is specified by the $x$-axis and the $y$-axis. The graph recognition methods introduced by the authors[1], [4], [8] must also satisfy assumptions about graphs. But, many mathematical figures do not satisfy all of the assumptions. Therefore, to develop a computer-aided system for transformation of mathematical figures into tactile graphics, it is necessary to study mathematical figure recognition techniques.

We are developing such a computer-aided system[9], which will transform mathematical figures into tactile graphics. This paper describes some of the mathematical figure recognition techniques that are introduced for developing our system. First, we focus on a method that separates a mathematical figure into the character image and the graph image. Since mathematical figures often include broken lines, we discuss two methods, one is for extracting broken line components from the graph image, and the other one is for grouping broken line components into each broken line.

This paper is organized as follows. Section describes the separation of a mathematical figure. Then, in Section III, the extraction and the classification of broken line components are discussed. Section IV shows the experimental results of the two methods. Section V is the conclusion.

## II. Separation of a Mathematical Figure into Character and Graph Components

The outline of the method discussed in this section is shown in Figure 1.

### A. Preprocessing

We first apply the 3 processes to an input image $I_{\text{in}}$, binarization, noise reduction, and the labeling process. After the labeling process, we have all of the connected components of $I_{\text{in}}$, that is, $C_1, \ldots, C_t$. Then, for each component $C_i$, a rectangle $R_i$ that circumscribes $C_i$ is determined. Figure 2(a) shows an example of input images, and some components and rectangles that circumscribe the components are given in Figure 2(b).

### B. Separation of A Mathematical Figure into Small and Large Components

In general, a graph includes coordinate axes, and lines or curves that express functions or relations. A single graph component usually occupies a larger area than a character component. In the process (1) shown in Figure 1, we classify large components as graph components, and then separate them from small components.

First, we take the length of the long side of input image $I_{\text{in}}$, and let it be $\ell_{\text{in}}$. Then, for every component $C_i$, if $\ell_i \geq w \times \ell_{\text{in}}$, then it is classified as a graph component, where $\ell_i$ is the length of the long side of $R_i$, and $w$ is a weight of $[0, 1]$.

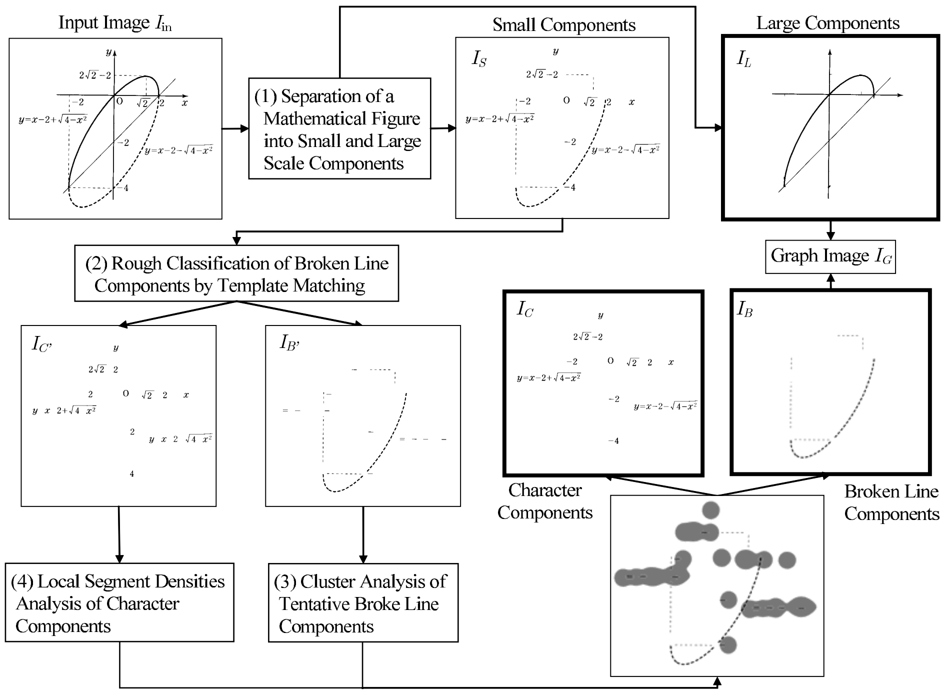
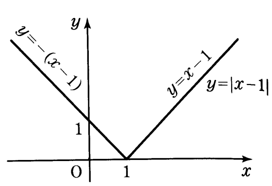
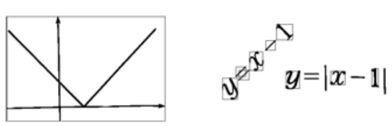



Fig. 1. Outline of Separation of A Mathematical Figure into Character and Graph Components




(a) An Input Image $I_{\text{in}}$




(b) Components and Rectangles That Circumscribe the Components

Fig. 2. An Example of Components and Rectangles That Circumscribe the Components

### *C. Extraction of Broken Line Components*

Graphs are sometimes drawn using broken lines. The input image $I_{\text{in}}$ in Figure 1 is an example of this kind of graph. The method from Section II-B classifies broken line components as small components (see the image $I_S$ in Figure 1). Given an image $I_S$, this section discusses a method of extracting broken line components from $I_S$. The following procedure shows the outline for separation of $I_S$ into broken line and character components.

**Input:** Image $I_S$
**Output:** Images $I_B$ and $I_C$

Step 1: Classify each component of $I_S$ into rectangular or non-rectangular components by template matching. Then, let $I_{B'}$ be the image of rectangular components, and let $I_{C'}$ be the image of non-rectangular components.

Step 2: Let $B'$ be the set of components in $I_{B'}$. Then, apply the agglomerative hierarchical clustering[5] to $B'$ in order for each cluster to consist of components of the same broken line.

Step 3: Calculate local segment densities[3] of $I_{C'}$, and then roughly separate $I_{C'}$ into character areas.

Step 4: For each cluster $G$ given by Step 2, if $\alpha$% or more of $G$ is covered by a character area given by Step 3, then classify all components of $G$ as character components.

Step 5: Remove every component, which was classified as a character component, from $I_{B'}$, and let the remaining image be $I_B$. Let $I_C$ be the image given by combing $I_{C'}$ and the character components. Output $I_B$ and $I_C$.

Let $I_G$ be the image given by combining $I_L$ and $I_B$, then $I_G$ is the graph image of $I_{\text{in}}$. $I_C$ is the image consisting of character components of $I_{\text{in}}$.

*1) Rough Classification of Broken Line Components by Template Matching:* A broken line component usually resembles a rectangle. So, in Step 1, each component of $I_S$

is classified into rectangular or non-rectangular components. A component of $I_S$ is said to be a tentative broken line component, if it was classified as a rectangular component. The following is our template matching procedure.

**Input:** Component $C$ of $I_S$
**Output:** Yes, if $C$ was classified as a tentative broken line component, otherwise No.

Step 1: Calculate slop $a$ of the longitudinal orientation of $C$ by Formula (1).

$$a = \frac{\sum_{x=1}^{N}\sum_{y=1}^{M}(x-m_x)(y-m_y)C(x,y)}{\sum_{x=1}^{M}\sum_{y=1}^{N}(x-m_x)^2C(x,y)}, \quad (1)$$

where $(m_x, m_y)$ is the centroid of $C$, $C(x,y) = 1$ when the pixel $(x,y)$ is in $C$, otherwise $C(x,y) = 0$, and $N$ and $M$ are the length of both sides of the rectangle that circumscribes $C$.

Step 2: Determine a rectangle such that (1) it circumscribes $C$ and (2) its longitudinal orientation is equal to the slop $a$. Denote the rectangle by $R^*$.

Step 3: If the following inequality is true (i.e., more than 80% of $C$ is covered by $R^*$), then $C$ is classified as a tentative broken line component.

$$\frac{\text{Number of Pixels of } C}{\text{Number of Pixels of } R^*} > \theta_1$$

The image of tentative broken line components of $I_S$ is $I_{B'}$, and the image of the components of $I_S$ other than the tentative broken line components is $I_{C'}$.

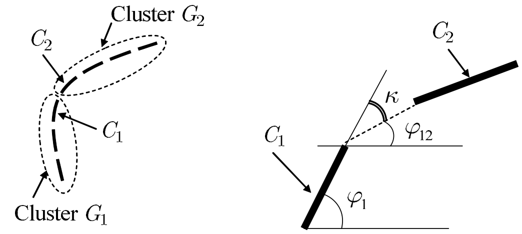
*2) Cluster Analysis of Tentative Broken Line Components:* In this section, we introduce a technique of making up clusters in order for each of them to include the components of the same broken line. Since, generally, the distance between two adjacent components of the same broken line is very short, the agglomerative hierarchical clustering is selected to make clusters. Here, the dissimilarity value $d^*(G_1, G_2)$ between two clusters $G_1$ and $G_2$ is calculated by the following formula.

$$d^*(G_1, G_2) = \min_{C_1 \in G_1, C_2 \in G_2} Dis^*(C_1, C_2), \quad (2)$$

where $Dis^*(C_1, C_2)$ is the distance between $R_1^*$ and $R_2^*$, which are the rectangles determined by the same way in the previous section.

Further, we made an assumption for making up clusters. That is, a broken line is assumed to be either a straight line or a smooth curve. We measure curvature[2] to determine the smoothness of a broken line. Let $G_1$ and $G_2$ be two clusters of broken line components, and let $C_1$ and $C_2$ be components of $G_1$ and $G_2$ such that the distance $Dis^*(C_1, C_2)$ gives the dissimilarity value $d^*(G_1, G_2)$ (see Figure 3(a)). Then, the curvature at the connected point of the clusters $G_1$ and $G_2$ is calculated by

$$\kappa = |\varphi_1 - \varphi_{12}|,$$



(a) Two Clusters $G_1$ and $G_2$ (b) Curvature $\kappa$ at Connected Point

Fig. 3. Curvature of Two Clusters

where $\varphi_1$ is the angle of $C_1$, and $\varphi_{12}$ is the angle of the line segment which connects the ends of $C_1$ and $C_2$ (see Figure 3(b)).

The following is the procedure of our cluster analysis for tentative broken line components.

**Input:** A set of components $\{C_1, \ldots, C_k\}$ of $I_{B'}$
**Output:** Clusters $G_1, \ldots, G_t$

Step 1: Calculate the distance $Dis^*(C_1, C_2)$ between each pair of components $C_i$ and $C_j$ $(i \neq j)$. Let $D$ be the set of all distances. Note that there are ${}_k\mathrm{C}_2 = k(k-1)/2$ distances in the set $D$.

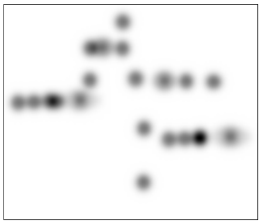
Step 2: Select the minimum distance from $D$, say $\hat{d}$, and set $D \leftarrow D - \{\hat{d}\}$. Continue this process until the two components $C_i$ and $C_j$ (which give the distance $\hat{d}$) are not members of the same cluster. Then, let the two clusters be $G_1$ and $G_2$.

Step 3: If $\hat{d} \geq \theta_2$, then output the clusters and stop the procedure, otherwise go to Step 4.

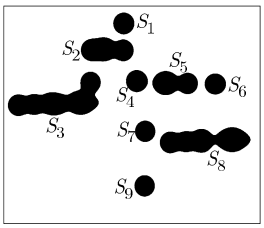
Step 4: Compute the curvature $\kappa$ between the two clusters $G_1$ and $G_2$. If $\kappa \leq \theta_3$, then merge $G_1$ and $G_2$. Go to Step 2.

*3) Local Segment Density Analysis of Character Components:* Usually in the template matching process of Section II-C1, the equal signs (=) and the minus signs (−) etc. are misclassified as tentative broken line components (see $I_{B'}$ in Figure 1). Furthermore, it is possible for a character component to be misclassified as a tentative broken line component, if it resembles a rectangle.

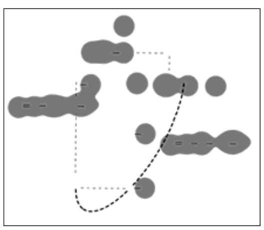
Given an image $I_{C'}$, we first calculate the local segment density for every pixel. The characteristic feature of local segment density is that the more characters that exist around a pixel, the higher the density at that pixel. Figure 4(a) shows local segment densities of the image $I_{C'}$ in Figure 1. Then, the binary image of local segment densities roughly gives us areas in which characters exist. For example, Figure 4(b) is the binary image given from Figure 4(a), and then the character components are roughly separated into the black areas. Let $S_i$ be a character area of $I_{C'}$ (see Figure 4(b)). Then, lastly, for every cluster $G$ (which is given by the procedure of the previous section), if $\alpha$% or more of the components in $G$ was included in $S_i$, then components in $S_i$ are distinguished



(a) Local Segment Densities

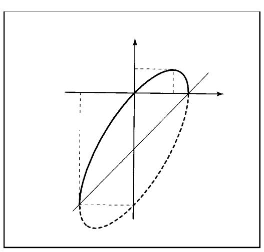


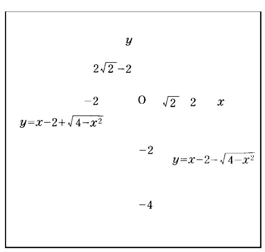

(b) Binary Image



(c) Overlapping Image

Fig. 4. Local Segment Densities of Character Components $I_{C'}$






(a) Graph Image $I_G$ (b) Character Image $I_C$

Fig. 5. Separation of A Mathematical Figure into Character and Graph Images

as character components. If not, all of the components in $G$ are distinguished as components that compose the same broken line. Figure 5(a) shows the image $I_G$ that includes graph components classified as large scale components and broken line components, while Figure 5(b) is the image $I_C$ that includes character components.

## III. Dotted/Chain Line Classification

The clustering method of Section II-C2 can not grope components of the same broke line into one cluster. So, it is necessary to discuss a method that classifies clusters, which were obtained by the method from Section II-C2, into chain lines and dotted lines.

Consider the chain line shown in Figure 6. The number of pixels for each component was counted, and Table I is a list of these numbers. The number of the left-most of the list is the number of pixels of the component that is on the left-most of the chain line. We then applied $k$-means clustering[5] to the set of these numbers by setting $k = 2$. The clustering result is shown in Table I. This clustering result was acceptable, i.e., the short segments were correctly grouped into one cluster, and the large segments were also correctly grouped. Then, the problem is how to evaluate the correctness of a clustering result. To

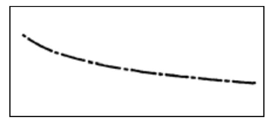


Fig. 6. An Example of Chain Lines

TABLE I
Clustering Result of Chain Line

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| # of Pixels | 88 | 632 | 95 | 504 | 104 | 553 | 93 |
| Cluster ID | A | B | A | B | A | B | A |
| # of Pixels | 593 | 113 | 558 | 115 | 550 | 126 | 526 |
| Cluster ID | B | A | B | A | B | A | B |

evaluate the clustering result, we measure two cluster validities, Davies-Bouldin Index $V_{DB}$ and Dunn's Index $V_D$[10], which is defined in the following. Let $X$ be a set of samples, and let $U = \{G_1, \ldots, G_c\}$ be a $c$-partition of $X$, that is, $\cup_{i=1}^{c} G_i = X$, $G_i \cap G_j = \emptyset$ if $i \neq j$, and $G_i \neq \emptyset$ for any $i = 1, \ldots, c$. The distance between two samples $\boldsymbol{x}$ and $\boldsymbol{y}$ is denoted as $\delta(\boldsymbol{x}, \boldsymbol{y})$. Then, Davies-Bouldin Index $V_{DB}$ is defined by the following formula.

$$V_{DB} = \left(\frac{1}{c}\right) \sum_{i=1}^{c} \left[ \max_{j, j \neq i} \left\{ \frac{\alpha_i + \alpha_j}{||\overline{\boldsymbol{v}}_i - \overline{\boldsymbol{v}}_j||} \right\} \right],$$

where $\alpha_i = \sum_{\boldsymbol{x} \in G_i} \frac{||\boldsymbol{x} - \overline{\boldsymbol{v}}_i||}{|G_i|}$ and $\overline{\boldsymbol{v}}_i = \sum_{\boldsymbol{x} \in G_i} \frac{\boldsymbol{x}}{|G_i|}$ for $i = 1, \ldots, c$. Furthermore, Dunn's Index $V_D$ is defined as follows.

$$V_D = \min_{1 \leq i \leq c} \left\{ \min_{1 \leq j \leq c, j \neq i} \left\{ \frac{\hat{\delta}(G_i, G_j)}{\max_{1 \leq k \leq c} \{\Delta(G_k)\}} \right\} \right\},$$

where for any clusters $S$ and $T$, $\Delta(S) = \max_{\boldsymbol{x}, \boldsymbol{y} \in S} \{\delta(\boldsymbol{x}, \boldsymbol{y})\}$ and $\hat{\delta}(S, T) = \min_{\boldsymbol{x} \in S, \boldsymbol{y} \in T} \{\delta(\boldsymbol{x}, \boldsymbol{y})\}$.

Generally, for Davies-Bouldin Index, the larger the value $V_{DB}$, the better the $c$-partition $U$. On the other hand, for Dunn's Index, the smaller the value $V_D$, the better the $c$-partition $U$.

The parameter $k$ of $k$-mean clustering is set at from 2 to 5. We then apply fuzzy inference to classify a cluster into a dotted line and a chain line. We determined the following 6 fuzzy rules.

Let $V_{DB}^i$ and $V_D^i$ be the Davies-Bouldin Index value and the Dunn's Index value when the number of cluster is $k = i$, respectively. Further, let $P_i$ be the number of pixels for a component $C_i \in G$ (where $G$ is a cluster of broken line components), and denote the maximum and the minimum among all $P_i$'s as $P^+$ and $P^-$, respectively.

Rule 1: If $V_{DB}^2$ is small, then $G$ is a chain line.
Rule 2: If the difference between $V_D^2$ and the minimum of $V_D^3, \ldots, V_D^5$ is small, then $G$ is a chain line.
Rule 3: If the ratio $(P^-)/(P^+)$ is small, then $G$ is a chain line.
Rule 4: If $D_{DB}^2$ is large, then $G$ is a dotted line.
Rule 5: If the difference between $V_D^2$ and the minimum of $V_D^3, \ldots, V_D^5$ is large, then $G$ is a dotted line.
Rule 6: If the ratio $(P^-)/(P^+)$ is large, then $G$ is a dotted line.

Fuzzy inference is performed by the product-max-centroid method. The membership functions are shown in Figure 7.

## IV. Experimental Results

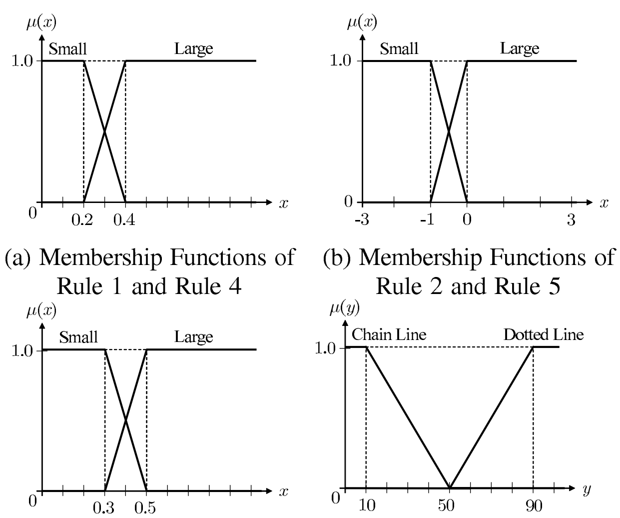
This section shows experimental results of the two methods, the separation of mathematical figures and the extraction of broken lines.

### A. Experimental Results of Separation of Mathematical Figures

We selected 6 mathematical figures from 4 different mathematics textbooks. Two mathematical figures were correctly separated into the character and the graph images. Figure 8 shows an example of the successful results, while Figure 9 gives an example of the failed results.

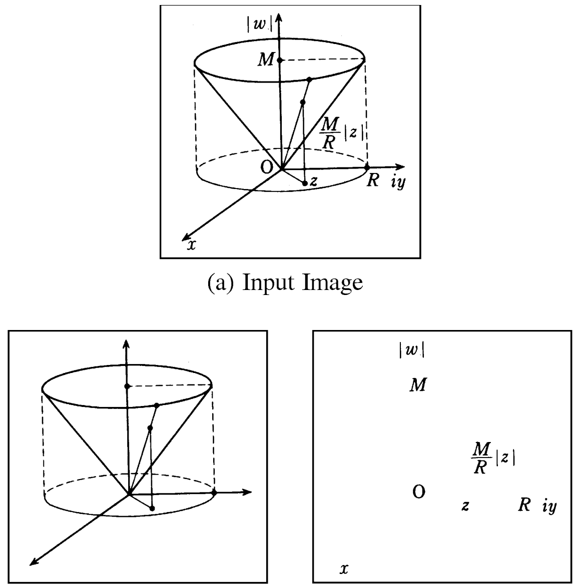
The cases where misclassifications occurred are summarized as follows.

(1) Misclassifications by Template Matching



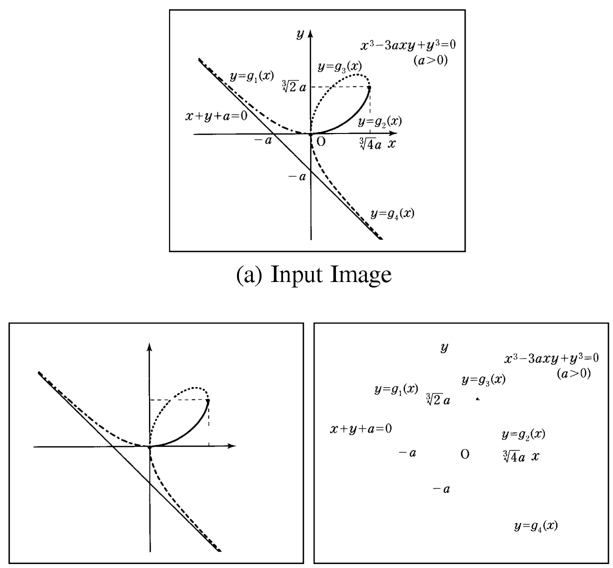

(a) Membership Functions of Rule 1 and Rule 4
(b) Membership Functions of Rule 2 and Rule 5
(c) Membership Functions of Rule 3 and Rule 6
(d) Membership Functions of Consequence

Fig. 7. Membership Functions




(a) Input Image
(b) Graph Image
(c) Character Image

Fig. 8. A Successful Result




(a) Input Image
(b) Graph Image
(c) Character Image

Fig. 9. A Failed Result

The template matching often misclassifies a graph component that does not resemble a rectangle as a character component. Arrows and broken line components that are not straight segments etc. are examples of graph components which are often misclassified as character components. The character image (c) of Figure 9 includes 1 broken line component that was misclassified, since the component was drawn by connecting two dotted line components.

(2) Misclassifications by Hierarchical Clustering
For example, if a character component that resembles a rectangle was merged with a cluster of the broken line, the character component will be misclassified as a graph component.

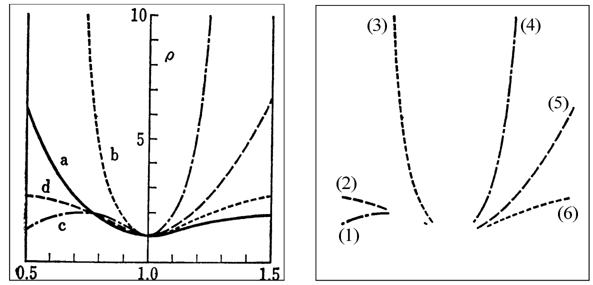
(3) Misclassifications by Local Segment Density
For example, if a negative integer, say $-1$, was drawn in a figure, and no character existed around this integer, then the area of the character '1' determined by local segment density sometimes does not cover the minus sign. Such components are misclassified as a graph component.

### B. Experimental Results of Broken Line Component Classification

We collected 20 mathematical figures, and then applied the two processes from Section III. We then extracted 112 clusters of broken line components. Note that a cluster that includes only one component was not applied to these processes. This is because $k$-means clustering does not work if the data set contains only one sample. Table II shows the results of dotted/chain line classification. Figure 10 is an successful example, i.e., all of the 6 broken lines were correctly classified into dotted lines and chain lines. Figure 11 shows a failed example. Two dotted lines were grouped into the cluster (1), because the distance between two dotted lines are short and the curvature at the connected point of the two dotted lines is small. Then, since the cluster (1) contains two different dotted lines, it was misclassified as a chain line.

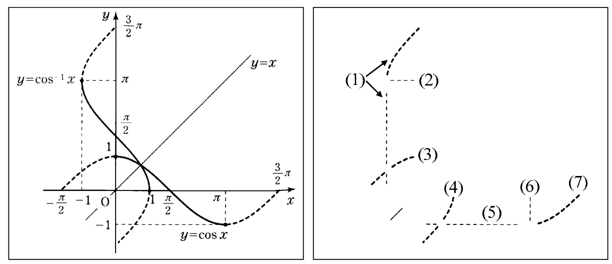
TABLE II
RESULTS OF DOTTED/CHAIN LINE CLASSIFICATION

| | # of Images | # of Clusters |
|---|---|---|
| Success | 15/20 | 104/112 |
| Failure | 5/20 | 8/112 |




(a) Input Image (b) Processing Result

Fig. 10. A Successful Example




(a) Input Image (b) Processing Result

Fig. 11. A Failed Example

## V. CONCLUSIONS

In this paper, we discussed pattern recognition for transformation of mathematical figures into tactile graphics. This paper introduced two methods, the separation of a mathematical figure into the character components and the graph components, and the extraction and the classification of broken line components. Local segment density, hierarchical clustering, and fuzzy inference play an important role in our methods. As discussed in the section of experimental results, our methods do work well for many mathematical figures, but we have some mathematical figures that our method did not return correct results. So, the improvement of the accuracy of our methods is our future problem.


## ACKNOWLEDGEMENT

This work is supported by Grant-in-Aid for Scientific Research of Japan Society for the Promotion of Science (19500812).


## REFERENCES


[1] S. Shimada, S. Kakumoto and M. Ejiri, "A Recognition Algorithm of Dashed and Chained Lines for Automatic Inputting of Drawings", *IEICE Transactions on Information and Systems*, Vol.J69-D, No.5, pp.759-770, 1986.

[2] N. Ono and R. Takiyama, "On Calculations of Curvature of Sampled Curves", *IEICE Technical Report*, Vol.IE93-74, pp.7-14, 1993.

[3] O. Shiku, A. Nakamura and H. Kuroda, "A Method for Character String Extraction Using Local and Global Segment Densities", *Journal of Information Processing*, Vol.40, No.8, pp.3230-3238, 1999.

[4] N. Yokokura and T. Watanabe, "Recognition of Business Bar-graphs Using Layout Structure Knowledge", *Journal of Information Processing*, Vol.40, No.7, pp.2954-2966, 1999.

[5] R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification*, 2nd Edition, A Wiley-Interscience Publication, 2001.

[6] T. Fuda, S. Omachi and H. Aso, "Recognition of Line Graph Images in Documents by Tracing Connected Components", *IEICE Transactions on Information and Systems*, Vol.J86-D-II, No.6, pp.825-835, 2003.

[7] S. Oouchi, M. Sawada, T. Kaneko and K. Chida, "A Survey on Making and Using Tactile Educational Materials in Schools for the Blind", *Bulletin of National Institute of Special Needs Education*, Vol. 31, pp.113-125, 2004.

[8] M. Yo, T. Kawahara and R. Fukuda, "Handwriting Graphics Input System for High School Mathematics", *IEICE Technical Report*, PRMU2003-291, pp.43-48, 2004.

[9] N. Takagi, "Pattern Recognition in Computer-Aided Systems for Transformation of Mathematical Figures into Tactile Graphics", *International Journal of Intelligent Computing in Medical Sciences and Image Processing* (submitted)

[10] J. C. Bezdek *et. al*, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Kluwer Academic Publishers, 1999.