A New System for the Real-time Recognition of Handwritten Mathematical Formulas

Takahiro Suzuki
Graduate School of Eng., Nagoya Univ.
Nagoya 464-8603 Japan
tsuzuki@suenaga.cse.nagoya-u.ac.jp

Shiro Aoshima
NTT Information Sharing Platform Laboratories
Tokyo 180-8585 Japan
aoshima.shirou@lab.ntt.co.jp

Kensaku Mori
Faculty of Eng., Nagoya Univ.
Nagoya 464-8603 Japan
mori@cse.nagoya-u.ac.jp

Yasuhito Suenaga
Faculty of Eng., Nagoya Univ.
Nagoya 464-8603 Japan
suenaga@cse.nagoya-u.ac.jp

Abstract

This paper presents a new system for the on-line and real-time recognition of handwritten mathematical formulas. Mathematical formulas are inputted into the system as hand drawings on a computer screen using a data tablet. The system analyzes the inputted data stroke by stroke in order to recognize the structure of the formula based on the relationship of bounding boxes that include components of the formula. A recognized result is obtained as a source code from the "LaTeX" type setting system. Our system can recognize arbitrary combinations of superscripts, subscripts, square roots, overlines, underlines, and fractions. The input of the math formula is independent of stroke order. In several experiments, the system proved to be very effective for inputting various math formulas in real-time.

1. Introduction

Techniques of real-time handwritten character recognition have been widely used on various electronic equipments, such as PDAs (personal digital assistants) or personal computers. Although most methods can recognize typical handwritten characters including alphanumerics and kanji, it is still impossible to handle the kind of mathematical formulas that are indispensable in technical documents. LaTeX and MathType (Design Science Inc.) are examples of systems that support the creation of documents including mathematical formulas. However, the user often feels difficulty in inserting LaTeX commands to the document, since LaTeX has a lot of tag commands for the typesetting of mathematical formulas. MathType is a representative system that enables the user to insert mathematical formulas by using the WYSIWYG environment. Users can choose desired formulas from the program's menus or buttons, but they must do a lot of mouse operations.

There are few research papers on mathematical formula recognition, and many of them only deal with methods for printed documents with mathematical formulas [1]. Wang and Faure reported a hand-written formula recognition method as an off-line procedure [2]. The on-line character recognition systems reported in [3],[4],[5],[6] cannot handle mathematical formulas.

This paper describes an on-line mathematical formula recognition method that is usable as an input system for technical documents. In the next section, we describe our proposed system. Section 3 explains our recognition methods. Section 4 explains functions of our system. Section 5 shows the evaluation experiments and results. Finally, Section 6 summarizes this paper.

2. Proposed System

Figure 1 shows the processing flow of the proposed system.

The user directly writes a mathematical formula on the computer screen and the system recognizes it. The system also enables the user to delete arbitrary characters, and the input of formulas is independent of stroke orders.

The mathematical formula structure differs from typical text structures where characters in the same line are located on the baseline. Mathematical formulas have special structures, called "*two-dimensional structures*" in this paper, where characters and mathematical symbols are located on different lines, such as overlines, underlines, and fractions. The *two-dimensional structure* is formed by a

1051-4651/00 $10.00 © 2000 IEEE

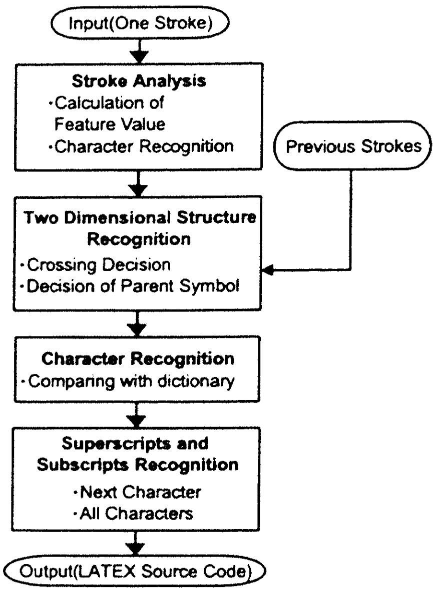


Figure 1. System flow

limited number of mathematical symbols except for superscripts and subscripts. Moreover, this structure depends on the mathematical symbols.

In handwriting, we write characters stroke by stroke, and form each character using a combination of multiple strokes.

The system uses these characteristics for achieving automated and real time recognition of mathematical formulas.

3. Recognition Procedure

3.1. Stroke Analysis

Here, the system calculates the features of an input stroke to select a single stroke character that has the most similar features to the input stroke.

As the user writes a mathematical formula on a data tablet, stroke information is transferred to the host computer as a time sequence of coordinates. The system encodes this stroke as a sequence of Freeman's chain codes and calculates three kinds of features: *chain code histogram, sub-vector sequence*, and *region information*.

[Features for stroke analysis]

a. chain code histogram
A histogram of chain codes for each stroke is calculated.

b. sub-vector sequence
Each stroke is divided into n ($n \leq L$) subsections of equal length, where L is the length of a stroke. Each divided stroke is represented by a vector (sub-vector) that connects a starting point and an ending point of each subsection. If $n > L$, the system does not divide a stroke and recognizes the stroke as "dot".

c. region information
The information includes the height and width of the bounding box of a stroke, and the position of the starting point of a stroke in the bounding box.

The system selects the most appropriate character corresponding to the input stroke by consulting the stroke dictionary, which is prepared by inputting stroke and character pairs beforehand.

3.2. Two-dimensional structure Recognition

We should determine the vertical position of an input stroke for recognizing a mathematical *two-dimensional structure*. Characters or mathematical symbols exist under or over some particular symbols (i.e., $\frac{b}{a}$, $\sqrt{c+d}$). We call these characters and mathematical symbols "*child blocks*" (i.e., a, b, $c+d$ in the previous examples), and the particular symbols "*parent symbols*"(i.e., —, $\sqrt{\ }$). Currently, the system can treat only six mathematical symbols as *parent symbols*, ($\sum$, $\prod$, $\_$, $\overline{\ }$, —, and $\sqrt{\ }$).

As described in 3.1, an input stroke is assigned to a single stroke character. When a character or a character with cross-sections from the previous character exists around the six predetermined symbols, the system executes the process described below.

In the case of $\sum$, $\prod$, $\_$, $\overline{\ }$, or —, the system determines the vertical relationship of crossing characters by comparing vertical coordinates. In the case of $\sqrt{\ }$, the system determines it by containing the relationship between the bounding box of the square root and that of crossing character.

This process finds the positional relationship of the *parent symbols* and their *child blocks*. It is iterated for the entire mathematical formula, since there may be multiple and nested *two-dimensional structures* in one mathematical formula.

3.3. Character Recognition

Features for single stroke characters were defined in Section 3.1. We need to obtain features for multiple stroke characters by connecting features of each stroke in the order of writing. They are obtained as shown below.

[Features for character recognition]

a. Connected chain code histogram
Directly connect chain code histograms of all strokes.

b. Connected sub-vector sequence
Connect two sub-vector sequences with the vector between the ending point of the previous sequence and the starting point of the next sequence.

c. Connected region information
Calculate the vertical and horizontal distances between the upper/lower/left/right edges of the two bounding boxes.

According to the number of strokes, the system searches for an appropriate character that has the same number of strokes in the prepared dictionary. This selection is performed by finding a character that maximizes correlation R between an input stroke and a character in the dictionary.

3.4. Superscripts and Subscripts Recognition

Superscripts and subscripts are detected by comparing the vertical positions of two adjacent characters. The right character is determined as a superscript if it is vertically higher than the left character. In the opposite case, a subscript is selected. If neither conditions are satisfied, the two characters are determined to be on the same line.

3.5. Other functions

(a) Deletion For a pleasant input environment, this system enables the user to delete any character that he/she selects at any time. The system then re-recognizes the structure of a mathematical formula and updates the recognition result.

(b) Adding a symbol When we write mathematical symbols (i.e.$\sqrt{\ }$, $\_$, $\overline{\ }$, —) before writing characters, the length of mathematical symbols may often be shorter for covering all characters. In this case, we extend the length of mathematical symbols by adding extra lines, such as an upper line, a lower line, or a line of a fraction. The system will also accept these extensions of the symbol.

When the user inputs an extended part, it is recognized as a subtraction sign in the first stage. If the above stated starting point of the subtraction sign is near the end of four symbols, the system recognizes the subtraction sign as extension of the symbol.

4. Experiment

Table 1 shows a list of characters that the proposed system is currently able to recognize. Figure 2 shows a screen shot of the proposed system. The user uses a data tablet to directly write a mathematical formula on the window. The operation window is divided into two parts; the upper side is a window for writing a mathematical formula, while the lower side is an output area. As the user adds a

Table 1. Recognizable characters

Alphabet	a,b,c,d,e,f,g,h,i,j,k,l,m,n, o,p,q,r,s,t,u,v,w,x,y,z A,B,C,D,E,F,G,H,I,J,K,L, M,N,P,Q,R,S,T,U,V,W,Y,Z
Greek Alphabet	α,β,σ,δ,ε,ω,π
Numeral	0,1,2,3,4,5,6,7,8,9
Mathematical Symbol	-,+,<,>, ≤,≥,≠,√,∫,Σ,Π,∞ (),{} ,sin,cos,tan,lim,log

stroke on the input window, the system executes the recognition procedure stated in Section 3 and promptly updates the LaTeX source code. Two hundred milli-seconds is required for each update.

We have validated recognition rates by writing four kinds of mathematical formulas on the screen. Three kinds of recognition rates were measured: character, superscript and subscript, and *two-dimensional structure* recognition. In this measurement, twenty-four users tried to input the four kinds of mathematical formulas shown in Figure 3. Each user tried to write each mathematical formula five times. Table 2 shows the recognition rates of those formulas. Figure 4 displays an example of input and output of mathematical formula from our system.

The recognition rate of a mathematical structure (superscript, subscript, and *two-dimensional structure*) was about 90%. The system could accurately recognize combinations of structures within typical mathematical formulas. The system could process a stroke within a quarter of a second. The user can input mathematical formulas fairly freely, without any significant stress.

The character recognition rate was about 80%. A low recognition rate was caused by the loss of important feature points in the stroke analysis. This process divides a

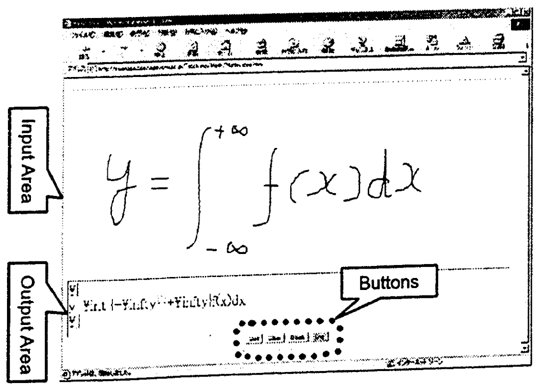


Figure 2. Window screen from system

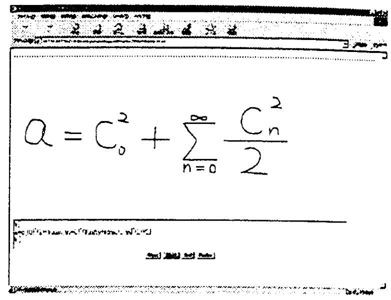
(1) $y=\int_{-\infty}^{+\infty} f(x)dx$ (2) $g=\frac{2q}{1+\sqrt{1+p^2+q^2}}$

(3) $f(x)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-m)^2}{\sigma^2}}$ (4) $a=c_0^2+\sum_{n=0}^{\infty}\frac{c_n^2}{2}$

Figure 3. Four mathematical formulas used the experiment

(a)

(b)



(c) $a = c_0^2 + \sum_{n=0}^{\infty} \frac{c_n^2}{2}$

Figure 4. An example result. ; (a) input, (b) recognition result (LaTeX source code), (c) output

stroke into n-divisions in order to obtain features for the stroke analysis. Important feature points such as bending points may disappear in this dividing process. Since many excellent methods of character recognition except for mathematical formulas have been reported by researchers, the employment of these methods should improve the character recognition rate and lead to better performance for our system.

5. Conclusions

In this paper, we have proposed an on-line and real-time recognition system of handwritten mathematical formulas that is implemented on a personal computer. The experimental recognition rates of this system were about 80% in character recognition and about 90% in structure recognition. The average processing time required for the processing was 200 milli-seconds per stroke. In the future work, we plan to develop new features that can express stroke forms more accurately, improve the dictionary and character recognition process, and develop a new method that can recognize other mathematical formula structures such as matrices.

Table 2. Experimental results

	Character Recognition	Superscripts and Subscripts Recognition	Two-Dimensional Structure Recognition
Formula(1)	74% (1158/1560)	94% (1352/1440)	
Formula(2)	86% (1551/1800)	93% (1230/1320)	75% (186/240)
Formula(3)	81% (2130/2640)	94% (1810/1920)	89% (527/600)
Formula(4)	76% (1468/1920)	93% (1221/1320)	88% (422/480)
Sum	80% (6307/7920)	94% (5613/6000)	86% (1135/1320)

References

[1] Masayuki Okamoto and Hiroyuki Higashi, "Mathematical Expression Recognition by the Layout of Symbols," Transactions of Institute of Electronics, Information and Communication Engineers (D-II), **J78-D-II**, 3, pp. 474-482 (March, 1995).

[2] Zi-Xiong Wang and Claudie Faure, "Structural Analysis of Handwritten Mathematical Expressions," Proc. ICPR, pp. 32-34 (1988).

[3] Yasushi Yamazaki and Naohisa Komatsu, "An Extraction Indicidual Characteristics Based on Categorized Handwritting Information," Transactions of Institute of Electronics, Information and Communication Engineers (D-II), **J79-D-II**, 8, pp. 1335-1346 (August, 1993).

[4] Kimiyasu Kiyota, Toshihiko Ssakurai and Shinji Yamamoto, "On-line Character Recognition for the Visually Disabled Person Based on the Relative Position of Stroke Representative Points," Transactions of Institute of Electronics, Information and Communication Engineers (D-II), **J80-D-II**, 3, pp. 715-723 (March, 1997).

[5] Yannis A. Dimitriadis and Juan Lopez Coronado, "Towards an Art Based Mathematical Editor That Uses On-line Handwritten Symbol Recognition," Pattern Recognition, **28**, 6, pp. 807-822 (June, 1995).

[6] Andreas Kosmala and Gerhard Rigoll, "On-Line Handwritten Formula Recognition Using Statistical Methods," In Proc. Int. Conference on Pattern Recognition (ICPR), pp. 1306-1308 (August, 1998).