INFTY — An Integrated OCR System for Mathematical Documents

Masakazu Suzuki[1)], Fumikazu Tamari[2)], Ryoji Fukuda[3)], Seiichi Uchida[4)], Toshihiro Kanahori[5)]

1) Faculty of Mathematics, Kyushu University, Japan
2) Department of Information Education, Fukuoka University of Education, Japan
3) Department of Human Welfare Engineering, Oita University, Japan
4) Faculty of Information Science and Electrical Engineering, Kyushu University, Japan
5) Research Center on Educational Media, Tsukuba College of Technology, Japan

suzuki@math.kyushu-u.ac.jp

ABSTRACT

An integrated OCR system for mathematical documents, called INFTY, is presented. INFTY consists of four procedures, i.e., layout analysis, character recognition, structure analysis of mathematical expressions, and manual error correction. In those procedures, several novel techniques are utilized for better recognition performance. Experimental results on about 500 pages of mathematical documents showed high character recognition rates on both mathematical expressions and ordinary texts, and sufficient performance on the structure analysis of the mathematical expressions.

Categories and Subject Descriptors

I.5.4 [**Pattern Recognition**]: Applications

General Terms

Algorithms

Keywords

Mathematical OCR, character and symbol recognition, structure analysis of mathematical expressions

1. INTRODUCTION

Optical character reader (OCR) systems for mathematical documents which contain not only ordinary texts but also mathematical expressions have been investigated [1]. The development of such OCR provides the following merits.

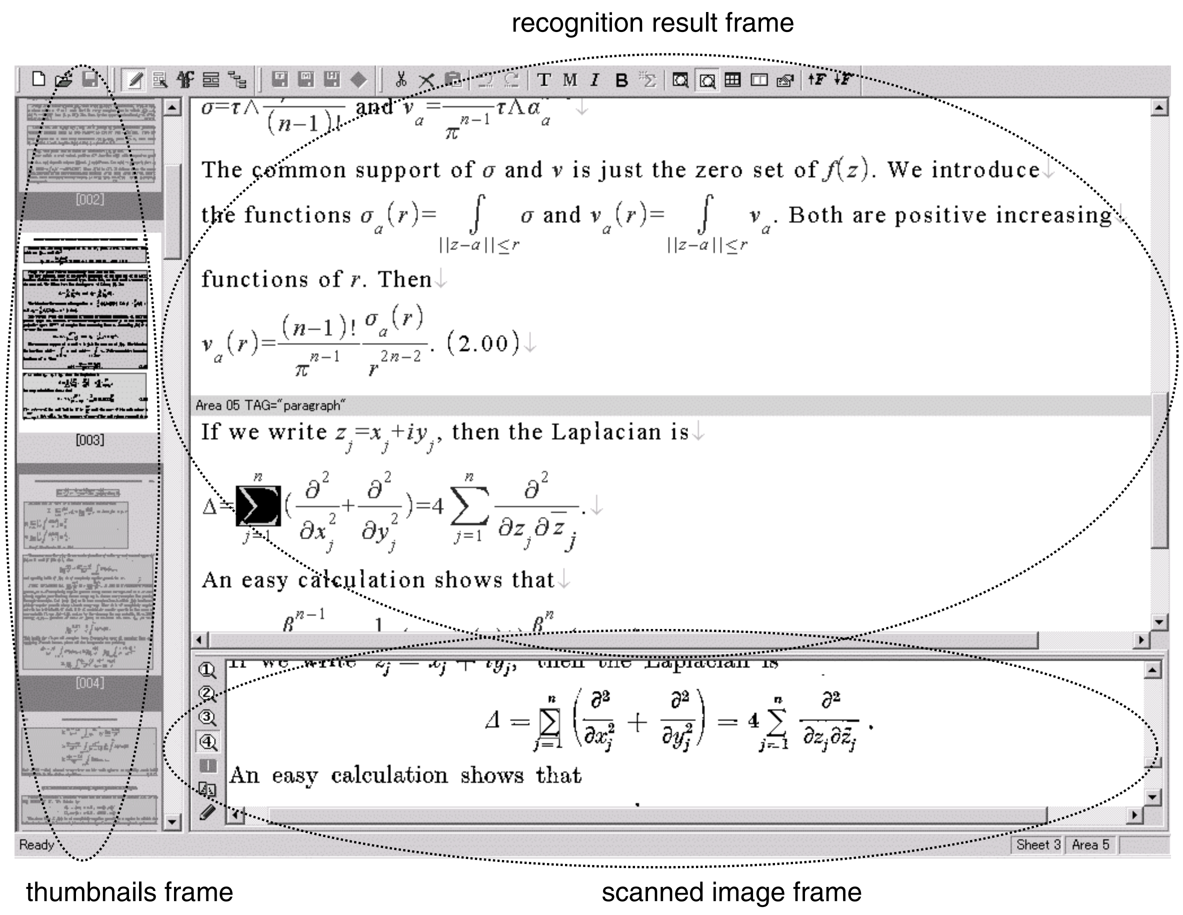
- *Storage size reduction*: The storage size of the mathematical document can be reduced because the mathematical expressions as well as the ordinary texts are stored as ASCII codes instead of scanned images.
- *Search services*: Various search services (e.g., keyword search, definition search, and theorem search) are available not only within each document but also across documents.
- *Format conversion*: The OCR result can be provided in various scientific document formats (e.g., XML, LaTeX, Mathematica notebook, and braille).

Especially, the OCR for mathematical documents is indispensable on digitizing numerous historical mathematical documents for digital library [2, 3].

In this paper, an integrated OCR system for mathematical documents, called *INFTY*, is presented. **Figure 1** shows a snapshot of INFTY on a PC. INFTY reads scanned page images of a mathematical document and provides their character recognition results. INFTY also performs structure analysis of mathematical expressions in the document. Thus, INFTY can produce its recognition result in the LaTeX format (**Fig. 2**) and other math-description formats, such as XML. **Figure 3** shows the diagram of INFTY, which consists of four procedures, i.e., ① layout analysis, ② character recognition, ③ structure analysis of mathematical expressions, and ④ manual error correction.

Novel and distinctive features of INFTY are summarized as follows.

- The character recognition procedure of INFTY consists of two independent and complementary recognition engines; one is a commercial OCR engine not specialized for mathematical documents and the other is a character recognition engine originally developed for mathematical symbols.
- The separation of ordinary text parts and mathematical expression parts is performed in the character recognition procedure while utilizing recognition results.
- The structure analysis procedure is based on an optimization framework and therefore stable against to both character/symbol recognition errors and structural ambiguity in the mathematical expressions.



$\sigma = \tau \wedge \frac{}{(n-1)!}$ and $\nu_a = \frac{}{\pi^{n-1}} \tau \wedge \alpha_a$

The common support of σ and ν is just the zero set of $f(z)$. We introduce

the functions $\sigma_a(r) = \int_{||z-a|| \leq r} \sigma$ and $\nu_a(r) = \int_{||z-a|| \leq r} \nu_a$. Both are positive increasing

functions of r. Then

$\nu_a(r) = \frac{(n-1)!}{\pi^{n-1}} \frac{\sigma_a(r)}{r^{2n-2}}$. (2.00)

If we write $z_j = x_j + iy_j$, then the Laplacian is

$\Delta = \sum_{j=1}^{n} (\frac{\partial^2}{\partial x_j^2} + \frac{\partial^2}{\partial y_j^2}) = 4 \sum_{j=1}^{n} \frac{\partial^2}{\partial z_j \partial \bar{z}_j}$.

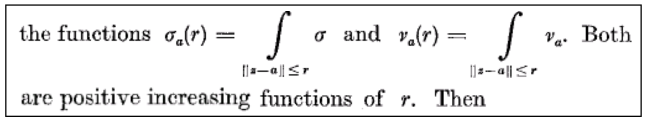
An easy calculation shows that

If we write $z_j = x_j + iy_j$, then the Laplacian is

$$\Delta = \sum_{j=1}^{n} \left(\frac{\partial^2}{\partial x_j^2} + \frac{\partial^2}{\partial y_j^2} \right) = 4 \sum_{j=1}^{n} \frac{\partial^2}{\partial z_j \partial \bar{z}_j} .$$

An easy calculation shows that

Figure 1: Snapshot of INFTY.



the functions $\sigma_a(r) = \displaystyle\int_{||z-a|| \leq r} \sigma$ and $\nu_a(r) = \displaystyle\int_{||z-a|| \leq r} \nu_a$. Both are positive increasing functions of r. Then

```
the functions $\sigma_a(r)={\displaystyle \int}_
{ ||z-a||\leq r} \sigma$ and $\nu_a(r)=
{\displaystyle \int}_{||z-a||\leq r}\nu_a$. Both
are positive increasing functions of $r$. Then
```

Figure 2: (Upper) A part of an input image. The image is scanned in 600 dpi. (Lower) Output of INFTY in LaTeX format.

- A clustering technique is incorporated for higher accuracy and efficiency.

The rest of this paper is organized as follows. In Section 2,3,4, and 5, the details of above four procedures (**Fig. 3**–①∼④) are described, respectively. In those description, the merits of above features are emphasized. Then, in Section 6, the performance of INFTY is evaluated qualitatively and quantitatively through experimental results on about 500 pages of mathematical documents.

2. LAYOUT ANALYSIS

In the layout analysis procedure (**Fig. 3**–①), which is the first procedure of INFTY, several preprocessing operations, such as binarization, noise removal, and deskewing, are performed on the page images (scanned in 600dpi) of a mathematical document.

After all connected components are extracted from the preprocessed page image, they are separated into figure / table areas and non-figure areas. One of the main criteria used in this separation is the size of the connected components. For example, the area with large connected components will be judged as a figure / table area. Note that big symbols, such as root symbols, big parentheses, etc., are ignored in this separation process by some special treatments.

The non-figure area is further decomposed into text lines. On non-mathematical documents, each text line is simply extracted by searching for the periodical local minima on the horizontal projection histogram of the page image. On mathematical documents, however, this strategy is not expedient; the heights of mathematical expressions are very variable and therefore the horizontal projection histogram are often irregular around the mathematical expressions. Our strategy is similar to Kacem et al.[4], where connected components in a certain neighborhood are concatenated to build a text line.

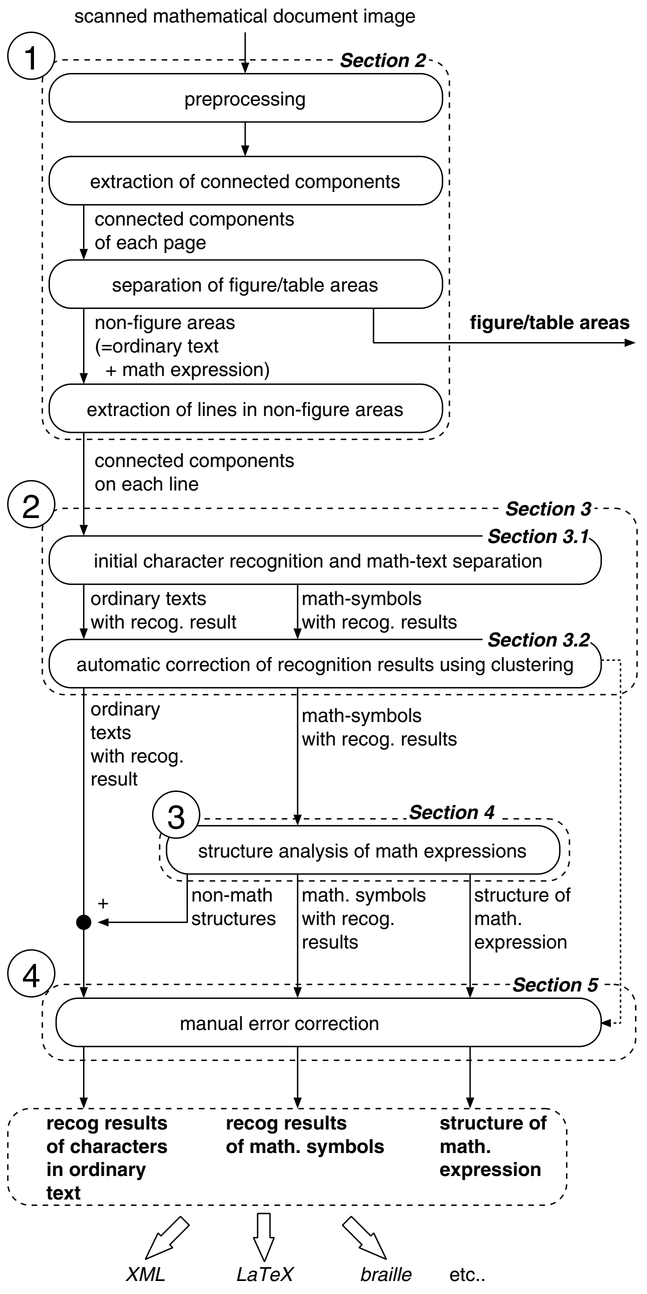

scanned mathematical document image
1
Section 2
preprocessing
extraction of connected components
connected components of each page
separation of figure/table areas
non-figure areas (=ordinary text + math expression)
figure/table areas
extraction of lines in non-figure areas
connected components on each line
2
Section 3
Section 3.1
initial character recognition and math-text separation
ordinary texts with recog. result
math-symbols with recog. results
Section 3.2
automatic correction of recognition results using clustering
ordinary texts with recog. result
math-symbols with recog. results
3
Section 4
structure analysis of math expressions
+
non-math structures
math. symbols with recog. results
structure of math. expression
4
Section 5
manual error correction
recog results of characters in ordinary text
recog results of math. symbols
structure of math. expression
XML
LaTeX
braille
etc..

Figure 3: Diagram of INFTY. INFTY consists of four procedures (surrounded by dashed lines), i.e., ① layout analysis, ② character recognition, ③ structure analysis of mathematical expressions, and ④ manual error correction.

3. CHARACTER RECOGNITION

The character recognition procedure (**Fig. 3**–②), which is the second procedure in INFTY, plays two important roles. The first role is the separation of each text line into mathematical expressions (e.g., "α^2", "$P(a) = \int_{-\infty}^{a} p(x)dx$") and ordinary texts (e.g., "Theorem", "defined"). The second role is the character recognition on both ordinary texts and mathematical expressions.

As shown in **Fig. 3**–②, the character recognition procedure consists of two sub-procedures, i.e., (i) initial character recognition and math-text separation and (ii) automatic correction of recognition results using clustering. The first sub-procedure is incorporated to provide math-text separation results as well as initial character recognition results. The second sub-procedure is incorporated to improve the recognition accuracy by reducing misrecognitions due to slight shape difference. In the following, the details of each sub-procedure are described.

3.1 Initial Character Recognition and Math-Text Separation

Figure 4 illustrates the detail of the sub-procedure for initial character recognition and math-text separation. This sub-procedure has two features. One feature is that this sub-procedure is that the math-text separation is performed while utilizing the result of character recognition. That is, the character recognition and the math-text separation are performed simultaneously and cooperatively. The other feature is that two complementary recognition engines, a commercial OCR engine for ordinary texts and an original recognition engine for mathematical expressions, are used in a two-step manner.

(a) Character Recognition by Commercial OCR Engine

The connected components on a text line is firstly subjected to a commercial OCR engine. When the text line only contains ordinary texts, this OCR engine will produce its good recognition result. However, when the text line contains mathematical expressions, the OCR engine will fail (due to non-roman fonts, sub-/super-scripts, mathematical symbols, etc.) and may produce some meaningless string (e.g., "y?zs" in **Fig. 4**) as the recognition result. In INFTY, this failure is exploited for initial math-text separation. Namely, the connected components recognized as such a meaningless string are selected as the connected components in the mathematical expressions.

(b) Verification Based on Position and Size

Some mathematical expressions might be wrongly recognized as ordinary words (e.g., "x^2"→"at" in **Fig. 4**) by the commercial OCR (sometimes due to the effect of the lexicon) and then they will be classified into the ordinary text part. Thus, after the recognition by the commercial OCR engine, the connected components recognized as some ordinary word should be verified.

In the verification, the consistency of the position and the size of the connected components of each word is checked. For example, in the misrecognition "x^2"→ "at", the size of "a" ("x", actually) is far different from the size of "t" (superscript "2", actually). In addition, the position of "a" and the position of "t" also shows some inconsistency. According to those checks, the connected components of "x^2" are detected as a mathematical expression.

One may be anxious about the fact that this verification is one-way, that is, the ordinary words wrongly classified into the mathematical parts are not targets of this verification. In INFTY, the detection of such ordinary words is performed on the structure analysis procedure as discussed in Section 4.

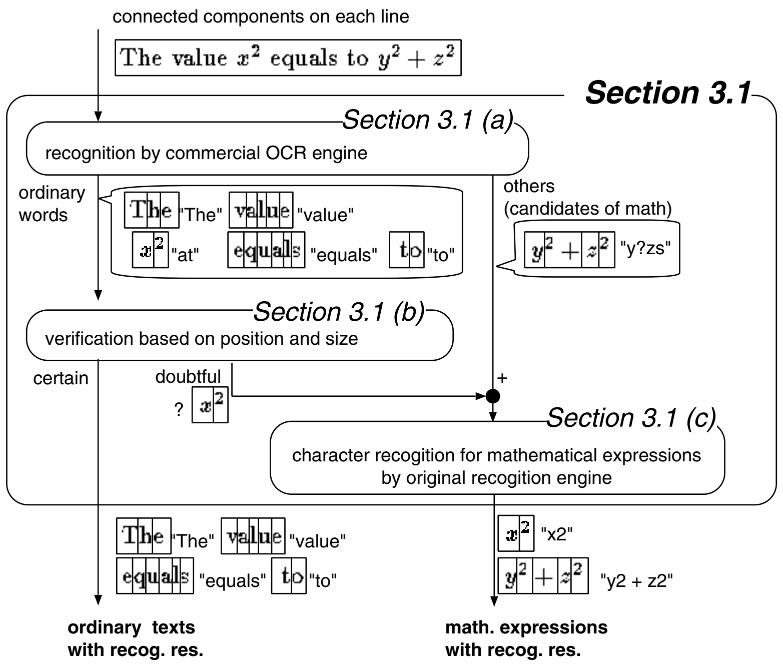

connected components on each line
The value x^2 equals to $y^2 + z^2$
Section 3.1
Section 3.1 (a)
recognition by commercial OCR engine
ordinary words
"The" "value"
"at" "equals" "to"
others (candidates of math)
"y?zs"
Section 3.1 (b)
verification based on position and size
certain
doubtful
?
+
Section 3.1 (c)
character recogition for mathematical expressions by original recogition engine
"The" "value"
"equals" "to"
"x2"
"y2 + z2"
ordinary texts with recog. res.
math. expressions with recog. res.

Figure 4: Initial character recognition and math-text separation described in Section 3.1.

(c) *Character and Symbol Recognition for Mathematical Expressions by Original Recognition Engine*

The connected components classified into the mathematical expression part are then subjected to a recognition engine originally developed for the mathematical characters and symbols (i.e., English and Greek alphabets, numerals, operators, parentheses, etc.,). In the mathematical expressions, font-types (e.g., roman, italic, calligraphic, and German) are often very important. For example, "i" and "i" are used in different meanings. Thus, this engine should be designed to distinguish the font-types and therefore "i" and "i" (as well as "B", "B", "$\mathcal{B}$", "$\mathbb{B}$", "$\mathfrak{B}$", and "$\mathscr{B}$") are considered as characters belonging to different categories. Consequently, the number of categories distinguishable by this recognition engine exceeds 500. (Precisely speaking, 564 categories are defined in INFTY, as discussed in Section 6.1.) Thus, the recognition engine should be computationally efficient enough to handle these many categories.

In this original recognition engine, a three-step coarse-to-fine classification strategy is employed for higher computational efficiency and recognition accuracy. At the first step, a coarse classification is performed using low-dimensional features (aspect ratio and crossing features). The features are extracted from every connected component (i.e., every character) and then simply compared to those of reference patterns prepared in the engine. As the result of this step, 10 $\sim$ 50 category candidates are selected. At the next step, 36-dimensional directional features are used to select 5 (or a bit more) category candidates from the above candidates. At the final step, those 5 candidates are ranked again according to three different measurements (i.e., distances based on the above 36-dimensional directional features, 64-dimensional peripheral features, and 64-dimensional mesh features) and then those three ranking results are unified by a voting method.

The character/symbol recognition results of mathematical expressions are not fixed to one category in this subprocedure. Namely, each connected component still has several category candidates. The recognition result will be finally fixed in the following structure analysis procedure (as discussed in Section 4).

3.2 Automatic Correction of Recognition Results by Clustering

In INFTY, a clustering technique is utilized in order to correct recognition results. In the clustering technique, all connected components (i.e., all characters and symbols in both the ordinary text part and the mathematical expression parts) are divided into several independent sets, called clusters, according to their shapes. Simultaneously, one representative, called *centroid*, is elected for each cluster.

The main idea of the correction by the clustering is simple; the connected components belonging to the same cluster are forced to have the same recognition results by majority voting within the cluster. As this result, the connected components initially misrecognized due to slight shape differences will be recognized into their correct category. Of course, if a cluster wrongly contains connected components of different categories, new misrecognitions are induced by the majority voting. In order to avoid this side-effect, the clustering should be "mild" so that each cluster only contains the connected components of the same category.

The clustering technique employed in INFTY is based on a sequential appending and splitting strategy. Basically, each connected component is appended to its nearest cluster one after another. During this appending procedure, the cluster whose variance exceeds some threshold is split into two independent clusters. This simple clustering technique is far faster than conventional iterative clustering techniques, such as k-means method. Since each cluster should contain

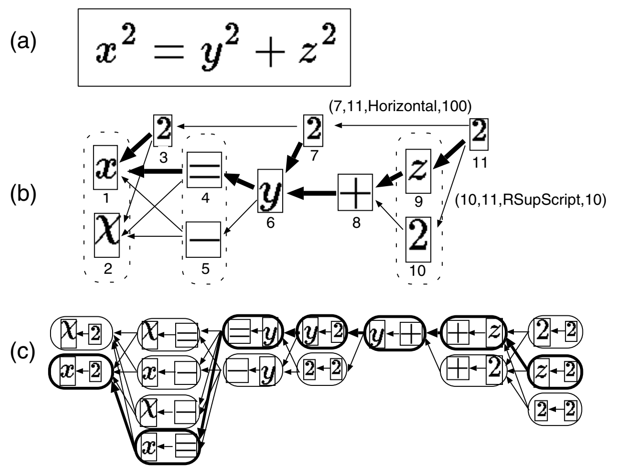


Figure 5: (a) A mathematical expression, (b) its digraph representation, and (c) the search graph for finding the minimum-cost spanning tree of the digraph (b).

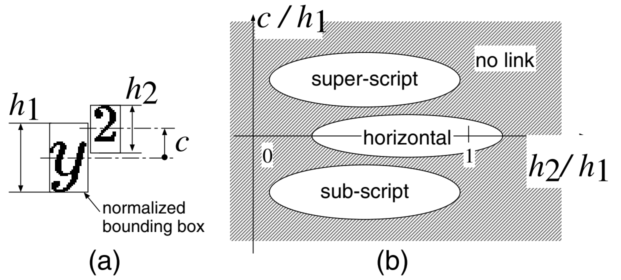


Figure 6: (a) Three values h_1, h_2, and c and (b) labelling conditions based on those values.

the connected components belonging to the same category as noted above, the threshold should be set to a small value.

Note that the clustering algorithm plays another important role in INFTY, i.e., the reduction of manual error correction operations. This role will be discussed in Section 5.

4. STRUCTURE ANALYSIS OF MATHEMATICAL EXPRESSIONS

In this section, the procedure for the structure analysis of mathematical expressions (**Fig. 3**–③), the third procedure in INFTY, is described. The connected components classified into the mathematical expression part by the preceding character recognition procedure (**Fig. 3**–②) are subjected to this procedure.

This procedure has three roles in INFTY. The first role is to represent the structure of each mathematical expression by a tree for converting the mathematical document into XML, LaTeX, and other math-description formats. The second role is to fix the character recognition result of the mathematical expressions. (As noted in Section 3.1(c), the character/symbol recognition results of mathematical expressions are not fixed to one category yet.) The third role

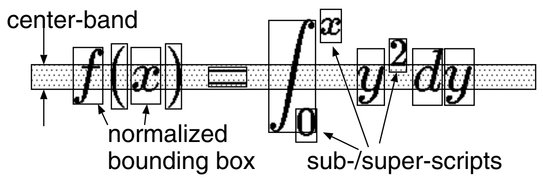


Figure 7: Center-band of line.

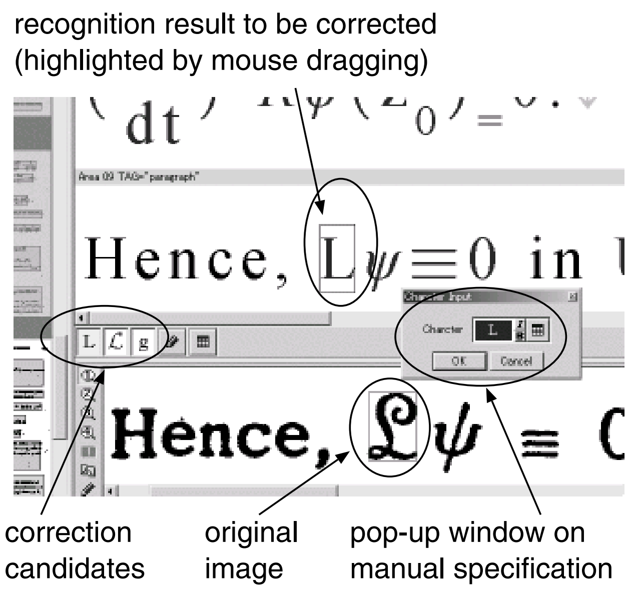

recognition result to be corrected (highlighted by mouse dragging)
Hence, $L\psi \equiv 0$ in
Character Input
Character
OK
Cancel
Hence, $\mathfrak{L}\psi \equiv$
correction candidates
original image
pop-up window on manual specification

Figure 8: Snapshot of manual error correction on INFTY. The misrecognition "$\mathcal{L}$" → "L" is to be corrected.

is to detect the ordinary texts wrongly classified into the mathematical expression part.

In order to simplify the structure analysis, each fraction is decomposed into its numerator and denominator in advance. This decomposition can be done rather easily since fraction lines in the mathematical expressions can be detected from its location, shape, and size. After analyzing the structures of the numerator and the denominator independently, their results are integrated to express the entire structure of the fraction. The mathematical expression in a root sign ("$\sqrt{\ }$") is also extracted from the sign before applying the structure analysis.

4.1 Digraph Representation of Mathematical Expression

The problem of the structure analysis is represented as a minimum-cost spanning-tree problem on a weighted digraph, or network. In **Fig. 5**(a) and (b), a mathematical expression and its digraph representation are shown, respectively.

Each node of the digraph corresponds to a category candidates of a character in the subjected mathematical expression. For example, node 1 and node 2 of the digraph of **Fig. 5**(b) correspond to the two category candidates of the character "x". A link, or a directed path, is prepared

Table 1: Detail of database for experimental evaluation. (1)

type	#categories	examples	#components in database (26 articles, 476 pages)				
			normal	touching	broken	touching & broken	total
accent	14	$\hat{}\ \tilde{}\ \bar{}\ \ddot{}\ \grave{}\ \breve{}$	2,483	44	2	25	2,554
arrow	16	$\leftarrow\leftrightarrow\longleftarrow\nwarrow$	926	4	0	0	930
bigsymbol	14	$\sum\int\prod$	1,230	2	0	0	1,232
blackboard bold	52	$\mathbb{ABCDEF}$	414	8	0	0	422
calligraphic	52	$\mathcal{ABCDEF}$	547	0	0	0	547
german	52	$\mathfrak{ABCabc}$	955	0	89	0	1,044
greek	40	$\Gamma\Delta\Theta\alpha\beta\gamma$	12,400	134	66	34	12,634
italic	52	$ABCabc$	108,314	2,172	1,044	433	111,963
numeral	10	012345	26,856	108	71	18	27,053
operator	83	$+ - \times / <$	19,877	137	73	27	20,114
others	44	#%∞∀∃†	9,287	279	191	27	9,784
parenthesis	18	( ){ }[ ]	36,932	777	62	43	37,814
point	13	, . ‘’	26,991	14	547	35	27,587
roman	52	ABCabc	435,597	8,599	1,138	268	445,602
script	52	$\mathscr{ABCDEF}$	3	0	0	0	3
total	564		682,812 (97.64%)	12,278 (1.76%)	3,283 (0.47%)	910 (0.13%)	699,283 (100%)

Table 2: Detail of database for experimental evaluation. (2)

part	#components in database (26 articles, 476 pages)				
	normal	touching	broken	touching & broken	total
ordinary text	534,750	10,018	1,717	655	547,140
math. exp.	148,062	2,260	1,566	255	152,143
total	682,812	12,278	3,283	910	699,283

between two nodes if their corresponding characters are adjacent and satisfy two conditions, whose details will be discussed later. Each link l is represented by a four-tuple, (*parent*, *child*, *label*, *cost*), where *parent* and *child* are the parent and the child nodes of the link l, respectively. Generally, the character corresponding to *parent* lies in the left side of *child*. For example, at the link between "x" and "2" of "x^2", the node of "x" will be the parent node [1]. The element *label* represents the positional relation between *parent* and *child*. There are 9 types of the label. For example, "`Horizontal`" is the label indicating that *child* and *parent* are adjacent horizontally, and "`RSubScript`" ("`RSupScript`") is the label indicating that *child* is a right subscript (superscript) of *parent*. The element *cost* is a value proportional to an uncertainty level of the link l.

As noted above, a link is prepared between two nodes only if the characters corresponding to those nodes satisfy two conditions. The first condition restricts the relative position of the two characters corresponding to *parent* and *child* nodes. This relative position is evaluated using three values, h_1, h_2, and c (**Fig. 6** (a)). The values h_1 and h_2 are the heights of normalized bounding boxes of those two characters respectively, where the normalized bounding box is a rectangle covering a character expanded with imaginary ascender and descender. The value c is the vertical difference between the two normalized bounding boxes. For preparing a link, the point $(h_2/h_1, c/h_1)$ should fall on one of three ellipsoidal regions of **Fig. 6** (b), which are predetermined through the statistical analysis of actual mathematical expressions. The label of the link is also decided at this time. For more details of this condition, see [5].

The second condition restricts the label of each link using the absolute vertical position of the corresponding characters. In this condition, a *center-band* (**Fig. 7**) is utilized. The center-band is a horizontally elongated region containing neither ascender nor descender. If the center-band is covered by the normalized bounding box of a character (e.g., "f", "(", "x" in **Fig. 7**), the character is considered as neither a subscript nor a superscript. Thus the link whose element *child* is such character is not allowed to have the label "`RSubScript`".

4.2 Optimization Algorithm

The structure analysis problem is now can be considered as the problem of searching for the minimum-cost spanning-tree on the weighted digraph prepared in the previous section. Our spanning-tree is somewhat peculiar and different from common spanning-trees which will consist of all nodes of the digraph. Our spanning-tree consists of only nodes which do not correspond to the same connected component. For example, any spanning-tree of the digraph of **Fig. 5** (b)

[1]Left subscripts, such as "m" of "${}_mC_n$" are flipped as right subscripts by some special treatment in advance to building the digraph.

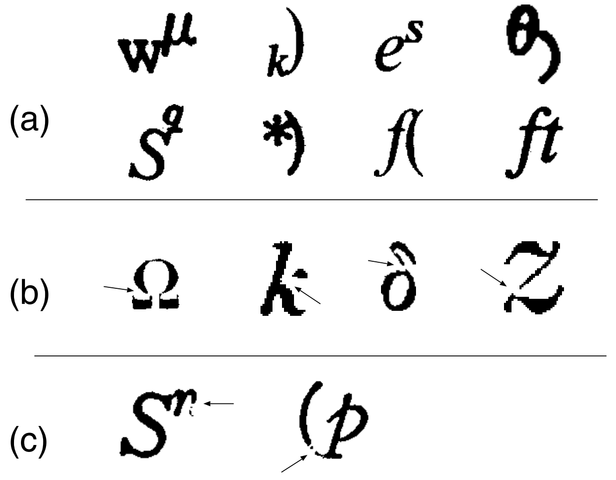


Figure 9: Three types of abnormal characters. (a) Touching characters, (b) broken characters, and (c) touching and broken characters in mathematical expressions. Broken points are indicated by arrows.

will not include node 9 ("z") and node 10 ("2") simultaneously because those nodes correspond to the same character "z" in **Fig. 5** (a). The tree by the thickened links in **Fig. 6** (b) is an example of the spanning-tree.

Again, the structure analysis problem can be considered as the problem to search all possible spanning-trees for minimum-cost one. This strategy is based on the fact that the most reliable structure is provided by the minimum-cost spanning-tree. Although the common minimum spanning-tree problem where the number of links is to be minimized has well-known algorithms (such as Kruskal's algorithm), our problem has no such algorithm due to its peculiarity. Thus, we have developed an algorithm based on breadth-first search with a pruning technique. In the algorithm, the search graph of **Fig. 5** (c) is built from the network of **Fig. 5** (b) and then its right-to-left path corresponding to the minimum-cost spanning-tree is searched for. For the details of the algorithm, see [5].

Once the minimum-cost spanning-tree of the weighted digraph is obtained, the structure of the subjected mathematical expression is represented as a (spanning-) tree and the character recognition result is fixed. Thus, the first and the second role noted at the beginning of this section are fulfilled by this procedure. On the other hand, if the spanning-tree is not obtained, the mathematical expression is rejected as an ordinary text. Thus, the third role can be fulfilled along with this procedure.

5. MANUAL ERROR CORRECTION

The manual error correction procedure (**Fig. 3**–④) is the last and optional procedure of INFTY. The errors of the character recognition and the mathematical structure analysis can be manually corrected using a graphical user interface. **Figure 8** is a snapshot where a character recognition error ("$\mathcal{L}$"$\rightarrow$ "L") is likely to be corrected. This correction is done by simply selecting the correct category from correction candidates (e.g., three candidates "L", "R", and "$\mathcal{L}$" shown in **Fig. 8**) while watching the scanned document image. If the correct category is not be found in those candidates, the correct result can be manually specified by keyboard input (or table pick-up) on a pop-up window.

The distinctive feature of this error correction procedure is that the result of the clustering performed in the character recognition procedure is exploited to reduce correction operations. Specifically, if the recognition result of a connected component belonging to a cluster is manually altered to its correct category, the recognition results of the other connected components belonging to the same cluster are also altered automatically to the same category. Thus, similar misrecognitions can be corrected by one operation, the correction operations can be reduced significantly in total.

Table 3: Character recognition results.

part	recognition rate (%)		
	best article	worst article	average over 26 articles
ordinary text	99.81	98.51	99.44
math. exp.	98.19	79.31	95.18
total(text+math)	99.68	93.79	98.51

Table 4: Relation between the abnormal character ratios and the recognition rates (of all normal and abnormal characters) in mathematical expression part.

abn. char. ratio in math part(%)	<1	1∼3	3∼10	>10	total
# articles	6	9	9	2	26
ave. recog. rate in math part(%)	97.70	96.60	93.00	82.42	95.18

6. EXPERIMENTAL EVALUATION

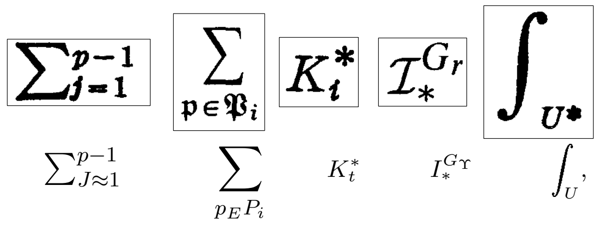
6.1 Database

A recognition experiment for the quantitative and the qualitative evaluations of INFTY was performed on 26 scientific articles; 25 English papers from 15 pure mathematical journals (e.g., *Bulletin of American Math. Soc.*, *Bulletin de la Soc. Math. France*, *Mathematische Annalen*, and *Kyushu J. Math.*) issued in 1960–1990 and one book on physics. The number of total pages was 476 (443 pages from the mathematical journals + 33 pages from the book on physics). The number of the subjected connected components was about 700,000 (547,140 from ordinary texts and about 152,143 from mathematical expressions). The articles were scanned at 600dpi. For each character and math symbol, its ground truth (correct category) was attached manually.

The details of the all connected components in the database are shown in **Table 1** and **Table 2**. The database contains various kind of characters peculiar to mathematical documents, such as Greek alphabets ($\alpha, \beta, \dots$), calligraphic typeface ($\mathcal{A}, \mathcal{B}, \dots$), and German typeface ($\mathfrak{A}, \mathfrak{B}, \dots$), in addition to mathematical symbols, such as big symbols ($\sum, \int$) and mathematical operators ($+, \times$). As shown in **Table 1**, the number of categories are 564.

Table 5: Character recognition rate of each character/symbol type. In this evaluation, abnormal characters (i.e., touching/broken characters) were excluded.

type	#(normal) characters	recog rate(%)
accent	2,483	96.46
arrow	926	98.60
bigsymbol	1,230	75.69
blackboard bold	414	78.99
calligraphic	547	78.98
german	955	66.18
greek	12,400	97.05
italic	108,314	97.81
numeral	26,856	97.80
operator	19,877	97.10
others	9,287	92.38
parenthesis	36,932	99.51
point	26,991	98.92
roman	435,597	99.84
script	3	0.00
total	682,812	99.01

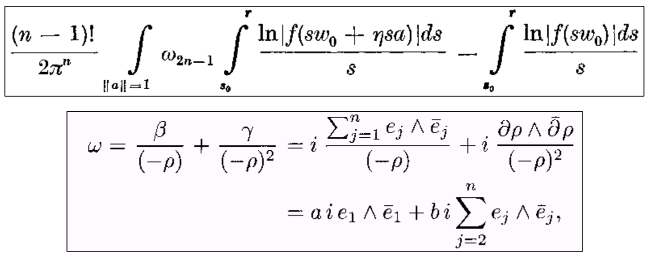


$\sum_{J\approx 1}^{p-1}$ $\qquad \sum_{p_E P_i}$ $\qquad K_t^*$ $\qquad I_*^{G\Upsilon}$ $\qquad \int_U ,$

Figure 10: Misrecognitions in mathematical expressions.

As shown in **Table 1**, there are many *abnormal characters*, i.e., (a)touching characters, (b)broken characters, and (c) touching and broken characters, in the database (**Fig. 9**). Most of the abnormal characters in the ordinary text part are correctly recognized by the commercial OCR engine, because the lexicon and the horizontal segmentation strategy employed in the engine will help to correct the misrecognitions of such abnormal characters. On the other hand, most of the abnormal characters in the mathematical expression part are misrecognized by the present INFTY. Thus, the detection and the separation/concatenation of the touching/broken characters are left as urgent future tasks. Among the tasks, the separation of the touching characters in the mathematical expressions is troublesome. This is because those touching characters might be touching diagonally or vertically (**Fig. 9**(a)) and therefore can not be separated into single characters by the conventional segmentation technique (such as projection based techniques [6, 7]) where the characters are assumed to be touching horizontally. One idea for this task can be found in [8].

In **Table 1** and the following experiments, bold fonts (e.g., bold-roman "**ABCabc**..." and bold-italic "***ABCabc***...") are treated as non-bold fonts. Although the discrimination between bold and non-bold is important for mathematical documents, their difference is very subtle and journal-dependent. Future work will include this discrimination problem. Note also that only matrices were manually detected and excluded as figure/table areas. The character recognition and structure analysis of the matrices are very challenging problems themselves and their implementation is also included in our future work. Recently, Kanahori and Suzuki citekanahori have proposed a structure analysis technique of various matrices.



$$\frac{(n-1)!}{2\pi^n}\int_{\|a\|=1}\omega_{2n-1}\int_{s_0}^{r}\frac{\ln|f(sw_0+\eta sa)|ds}{s}-\int_{s_0}^{r}\frac{\ln|f(sw_0)|ds}{s}$$

$$\begin{aligned}\omega = \frac{\beta}{(-\rho)}+\frac{\gamma}{(-\rho)^2} &= i\,\frac{\sum_{j=1}^{n} e_j\wedge\bar{e}_j}{(-\rho)}+i\,\frac{\partial\rho\wedge\bar{\partial}\rho}{(-\rho)^2}\\ &= a\,i\,e_1\wedge\bar{e}_1+b\,i\sum_{j=2}^{n}e_j\wedge\bar{e}_j,\end{aligned}$$

Figure 11: Scanned image of mathematical expressions analyzed perfectly.

6.2 Reference Patterns

Reference patterns used in the character/symbol recognition of the mathematical expression part (Section 3.1(c)) were prepared by the following steps. First, about 180,000 patterns were manually collected (and ground-truthed) from scanned page images of mathematical journals and hardcopies of LaTeX documents. The LaTeX documents were used supplementally to collect characters and symbols rarely found in the journals (e.g., calligraphic characters). Note that those patterns were completely independent of the above database for experimental evaluation.

Second, for each category, several representative patterns were elected as reference patterns of the category. A clustering technique was employed for the election. The number of the reference patterns of each category was often 2 or more in order to manage the variations in character sizes (there are sub-/super-scripts in the mathematical expressions) and shapes.

Reference patterns used in the character recognition of the ordinary text part (Section 3.1(a)) are unknown because they are concealed in the commercial OCR engine.

6.3 Accuracy of Character Recognition

As shown in **Table 3**, the character recognition rate averaged over the 26 articles was 98.51% (without manual error correction) and the recognition rates on the ordinary text part and the mathematical expression part were 99.44% and 95.18%, respectively. Those evaluations were very strict; the characters recognized in wrong font-types (e.g., "L"→"*L*" and "*L*"→"$\mathcal{L}$") were counted for misrecognitions as well as the characters recognized to their hardly-distinguishable characters (e.g., "o"→"O" and "l"→"1"). In spite of the strict evaluations, the recognition rate on the ordinary text part was high enough for practical uses. On the other hand, the recognition rate on the mathematical expression part should be improved in our future work, although the rate of 95.18% seems to be considerably higher than or comparable to recent mathematical OCR systems (such as [10, 11]).

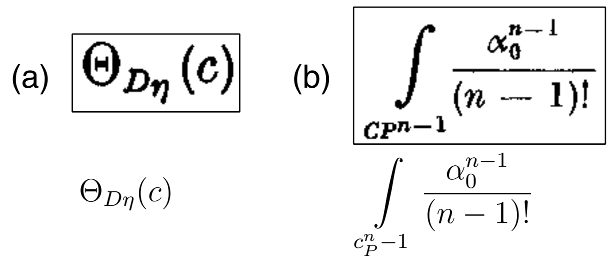


Figure 12: Failure results of math-structure analysis.

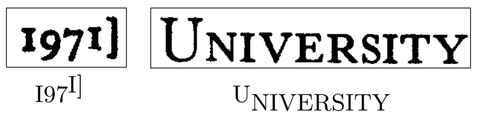


Figure 13: Ordinary texts wrongly classified into mathematical expression part.

The degradation of the recognition rate on the mathematical expressions was mainly due to abnormal characters. As noted in **6.1**, there are many abnormal characters in our database (over 2% of all characters) and most of the abnormal characters in the mathematical expression part are to be misrecognized. **Table 4** shows the relation between the ratio of the abnormal characters over all the characters in the mathematical expression part and the character recognition rate of the part. Clearly, the recognition rates of the articles with many abnormal characters are low. Thus, the separation/ concatenation of the touching/broken characters will be very effective to attain higher recognition rates especially in the mathematical expression part. It is worth to note that if all abnormal characters are excluded from the evaluation, the overall character recognition rate increases from 98.51% to 99.00%.

In addition to the abnormal characters, the following factors also induce the misrecognitions in the mathematical expressions, i.e., heavy size variations (sub-/sup-scripts, bigsymbols), large categories (English and Greek alphabets, operators, parentheses, etc.), font variations (italic, calligraphic, etc.), and the existence of similar characters (e.g., r, γ, and Υ). **Figure 10** shows five misrecognitions due to these reasons. Among those misrecognitions, the last two are due to double sub-/super-scripts (i.e., a sub-script of a sub-script and a super-script of a super-script), which are one of the most serious reasons of the misrecognitions.

Table 5 shows the character recognition rate of each character/symbol type. In this evaluation, all the abnormal characters were excluded. The result shows that characters of some special font-types (often complex and infrequent as shown in **Table 1**) are hard to be recognized.

The clustering technique of Section 3.2 was performed on every article for the automatic correction of the initial recognition results given by the procedure of Section 3.1. From our rough observation, it was shown that misrecognitions were reduced to 2/3 by the use of the clustering technique.

6.4 Accuracy of Structure Analysis of Mathematical Expressions

Among 12,493 mathematical expressions [2] in 476 pages, 11,194 expressions were perfectly analyzed, where the term "perfect" means that both the character recognition result and the structure analysis result are correct. Thus, the perfect analysis rate was 89.6%. **Figure 11** shows two perfectly analyzed results.

The failure results were mainly due to the misrecognitions of their component characters. In fact, there were 9602 mathematical expressions without misrecognitions and their perfect analysis rate was 97.9% (i.e., far higher than 89.6%). The double sub-/super-scripts also induce failure results. **Figure 12** (a) shows a failure result due to the double sub-/super-scripts. The double sub-script "η" is wrongly analyzed as one of sub-scripts of "Θ". Namely, the slight difference between the baselines of "D" and "η" was not distinguished during the structure analysis. Another typical failure result is shown in **Figure 12** (b). In this result, the misrecognition "C"$\rightarrow$"c" badly affects the analysis of the structure of "CP^{n-1}". Thus, the reduction of the misrecognition is an essential task for higher accuracy.

There were several ordinary texts wrongly classified into the mathematical expression part. **Figure 13** shows two examples. In the left example, a fluctuation of its baseline causes the miss-detections. In the right example, "small capitals" ("NIVERSITY") were detected as sub-scripts of "U" in the verification process in the character recognition procedure. The number of such wrong classifications were 1,249 in all the ordinary text parts (i.e., 0.2% of all 547,140 characters of the part) and often found in the running headers and the footers.

6.5 Computation Time

INFTY requires about 10 s/page to perform all procedures (except for the manual error correction procedure) on a desk top PC (Pentium III, 1GHz) and about 20 s/page on a notebook PC (Pentium III, 700MHz). This computation time is very comparable to the computation times of (commercial) OCR softwares which can not handle mathematical expressions.

6.6 Manual Error Correction

Manual error correction was performed on several articles for evaluating the effect of the clustering to reduce the number of error correcting operations by users. Its result showed that the operations could be reduced to about 1/3 by the automatic error correction using the result of the clustering.

7. CONCLUSION

An integrated OCR for mathematical documents, called INFTY, was presented. In INFTY, several novel techniques, such as simultaneous character recognition and math-text separation based on two complementary recognition engines, are utilized for better recognition performance. From experimental results on 476 pages of mathematical documents it was shown that sufficient character recognition rates (99.44% on ordinary texts, 95.18% on mathematical expressions, and

[2]Mathematical expressions with a single character, such as "x", were disregarded in this evaluation. In addition, one-dimensional mathematical expressions, such as "$a = b$" and "$x \times y + z$", were also disregarded here.

98.51% in total). It was also shown that 89.6% mathematical expressions are perfectly analyzed, i.e., no error in both character recognition and structure analysis.

8. ACKNOWLEDGMENTS

The authors greatly thank alumnae and alumni of Suzuki Laboratory of Kyushu University for their generous efforts on the development of INFTY. The authors also thank Toshiba Cooperation for their continuous support to our research project.

A beta version of INFTY can be downloaded freely. For further information, please visit [12].

9. REFERENCES

[1] K.-F. Chan and D.-Y. Yeung, "Mathematical expression recognition : a survey," *Int. J. Doc. Anal. Recog.*, 3(1):3–15, 2000.

[2] G. O. Michler, "Report on the retrodigitization project "Archiv der Mathematik"," *Archiv der Mathematik*, 77:116–128, 2001.

[3] K. Dennis, G. O. Michler, G. Schneider, and M. Suzuki, "Automatic reference linking in distributed digital libraries," *Proc. Workshop on Document Image Analysis and Retrieval (DIAR-03)*, 2003.

[4] A. Kacem, A. Belaïd, and M. B. Ahmed, "EXTRAFOR: automatic EXTRAction of mathematical FORmulas," *Proc. ICDAR*, 527–530, 1999.

[5] Y. Eto and M. Suzuki, "Mathematical formula recognition using virtual link network," *Proc. ICDAR*, 762–767, 2001.

[6] R. G. Casey and E. Lecolinet, "A survey of methods and strategies in character segmentation," *IEEE Trans. Pat. Anal. Mach. Intell.*, 18(7):690–706, 1996.

[7] Y. Lu, "Machine printed character segmentation – an overview," *Pattern Recognition*, 28(1):67–80, 1995.

[8] A. Nomura, K. Michishita, S. Uchida, and M. Suzuki, "Detection and segmentation of touching characters in mathematical expressions," *Proc. ICDAR*, 1:126-130, 2003.

[9] T. Kanahori and M. Suzuki, "A recognition method of matrices by using variable block pattern elements generating rectangular area," Graphics Recognition. Algorithms and Applications (Lecture Notes in Computer Science, 2390), Springer-Verlag, 2001.

[10] J. -Y. Toumit, S. Gracia-Salicetti, and H. Emptoz, "A hierarchical and recursive model of mathematical expressions for automatic reading of mathematical documents," *Proc. ICDAR*, 119–112, 1999.

[11] H.-J. Lee and J.-S. Wang, "Design of a mathematical expression understanding system," *Pattern Recognition Letters*, 18(3):289–298, 1997.

[12] `http://infty.math.kyushu-u.ac.jp`