Embedding a Mathematical OCR Module into OCRopus

Shinpei Yamazaki, Fumihiro Furukori, Qinzheng Zhao, Keiichiro Shirai* and Masayuki Okamoto*
Faculty of Engineering, Shinshu University
4-17-1 Wakasato Nagano 380-8553 JAPAN
*{*shirai, okamoto*}@*cs.shinshu-u.ac.jp*

***Abstract*—This paper describes embedding a mathematical formula recognition module into the OCR system OCRopus aiming at developing a OCR system for scientific and technical documents which include mathematical formulas. OCRopus is a open source OCR system emphasizing modularity, easy extensibility, and reuse. This system has several basic components such as preprocessing, layout analysis, and text line recognition, so it is a challenging project to embed the mathematical formula recognition module into the OCRopus system. We have developed the math OCR module, then report how to embed our module into the OCRopus system in order to realize a math OCR which can deal with wide variety of documents including mathematical formulas.**

***Keywords*-OCR; OCRopus; Mathematical formula recognition;**

I. Introduction

Since 1990s a lot of studies of document image analysis/recognition have been done and some commercial OCR software are available. However, these software have been specialized for particular applications. There remains still many problems to achieve a math OCR for digitizing large scale documents including mathematical formulas [1], [2].

OCRopus [3], [4] is an open source OCR system and has been developed to overcome the problems related to OCR. It enables us to evaluate or reuse OCR components such as preprocessing, layout analysis, and text line recognition. The OCRopus system can be considered as a commonly used test bench for OCR related researchers or engineers. It is challenging to add a new function into OCRopus.

We have studied and developed the document recognition system which can recognize mathematical formulas [5]–[7]. This system consists of some components such as formula structure analysis and character recognition but is required to improve its performance to deal with wide variety of specific documents. Therefore, we aim to build a math OCR system by using the OCRopus system which is expected for further improvement.

The mathematical formula recognition has been discussed by many researchers, and INFTY [8] is the only practical system to recognize documents including mathematical expressions to our best knowledge. INFTY can recognize many kinds of mathematical expressions with considerable precision but more research is required to digitize existing various documents.

In this paper, we report how to embed our mathematical formula recognition module into the OCRopus system. To realize it, it is required to separate mathematical expressions (embedded or displayed expressions) from ordinary text. For the first step of this project, we plan to develop a math OCR for displayed expressions.

The rest of this paper is organized as follows. Next section describes the architecture of the OCRopus system. Then, Section III describes how to embed the mathematical formula recognition module into the OCRopus system. In Section IV, some experimental results are shown. Finally, Section V presents the conclusions of this work.

II. Architecture of OCRopus

OCRopus has been developed in IUPR Research Group and hosted by Google Code [9]. Its overall architecture consists of three major components, namely layout analysis, text line recognition, and statistical language modeling.

Layout analysis
: Physical layout analysis identifies text columns, text blocks, text lines, and determines the reading order.

Text line recognition
: Text line recognizer separates text line images into the collection of characters, then perform the character recognition based on a hypothesis graph.

Statistical language modeling
: Statistical language modeling integrates alternative recognition hypotheses with prior knowledge about language, vocabulary, grammar, and the domain of the document.

III. Embedding mathematical formula recognition module

In this section, embedding the mathematical formula recognition system which has been developed in our laboratory into the OCRopus system is described. Our recognition system consists of two major modules, symbol recognition module and structural analysis module.

1) Symbol recognition module performs the segmentation of touching symbols, the unification of separated symbols and symbol recognition.
2) Structural analysis module analyzes expressions from left to right by checking types of symbols, their sizes

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.180

IEEE computer society

and relative locations. This module produces structure of mathematical expressions as LaTeX or MathML [10] formats.

In printed documents, expressions appear in two manners, namely, embedded (mixed with text and also referred as inline expression) and displayed (typed on a separate line). However, for the first attempt our system only deals with displayed expressions.

A. System overview

The whole processing steps including the mathematical formula recognition are shown in Figure 1. In this figure, the steps indicated in colored rectangles are implemented in this work, and the other rectangle modules are the components from the OCRopus.

Primary processing steps are as follows. Some of the OCRopus commands perform each processing step.

1) Preprocessing including binarization, skew detection and correction, and noise removal. In the OCRopus system, **ocropus-binarize** command performs these processes.
2) Layout analysis identifies text regions and the other regions such as figures or tables. The text regions are segmented into text lines. In this step, the coordinates of each text line are obtained and used for identification of mathematical formulas. **ocropus-pseg** command executes the layout analysis and extracts text lines.

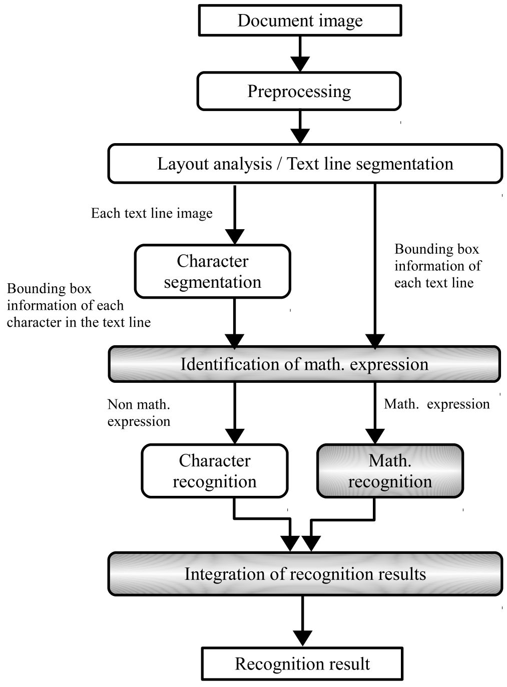
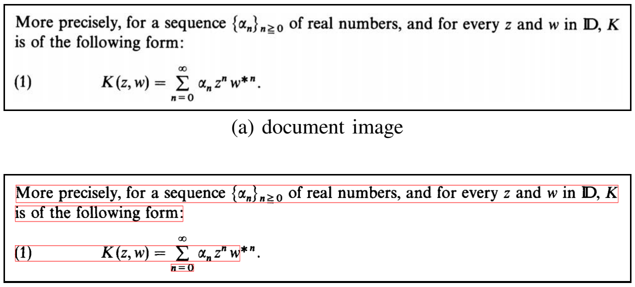

Document image
Preprocessing
Layout analysis / Text line segmentation
Each text line image
Character segmentation
Bounding box information of each text line
Bounding box information of each character in the text line
Identification of math. expression
Non math. expression
Math. expression
Character recognition
Math. recognition
Integration of recognition results
Recognition result

Figure 1. System overview



More precisely, for a sequence $\{\alpha_n\}_{n\geq 0}$ of real numbers, and for every z and w in $\mathbb{D}$, K is of the following form:

(1) $$K(z,w) = \sum_{n=0}^{\infty} \alpha_n z^n w^{*n}.$$

(a) document image

More precisely, for a sequence $\{\alpha_n\}_{n\geq 0}$ of real numbers, and for every z and w in $\mathbb{D}$, K is of the following form:

(1) $$K(z,w) = \sum_{n=0}^{\infty} \alpha_n z^n w^{*n}.$$

(b) RAST algorithm result

Figure 2. An example of text line segmentation error

3) Each text line is segmented into characters and the coordinates of each character are obtained. The classification of displayed expressions from ordinary text lines also uses the features based on these coordinate informations.
4) Text lines identified as displayed expressions are provided to the mathematical formula recognition module. The other text lines are recognized by the OCRopus system. **ocropus-lattices** and **ocropus-align** perform the character recognition based on a hypothesis graph.
5) Each recognition result is integrated into the whole document recognition result. **ocropus-hocr** outputs the recognition result.

Some of the above processing steps are described in the following in detail.

B. Text line segmentation by layout analysis

The features to classify displayed expressions are not computed correctly if the bounding boxes of the text lines are segmented improperly. The OCRopus system provides some algorithms for the text line segmentation, and each algorithm can be evaluated individually. The default RAST algorithm is based on whitespace identification and text line finding [11]. However in our experiments, RAST algorithm sometimes fails in the segmentation of displayed expressions. It often produces multiple lines for a displayed expression with upper or lower subexpressions such as:

$$\sum_{n=0}^{\infty} \alpha_n$$

Figure 2 shows an example of the segmentation error. Therefore, we need to select an algorithm for the text line segmentation. For the time being, our system supposes that input documents are composed of one column. For this type of document, the 1CP (1 Column Projection) algorithm shows the best performance for the text line segmentation. 1CP algorithm extracts text lines by a method based on the horizontal projection.

C. Character segmentation

Character segmentation is performed by the OCRopus system and the information about character sizes and locations is used for the following step.

D. Identification of mathematical formulas

1) Features of mathematical formulas: Some studies dealing with identification of mathematical formulas have been done [12]–[16]. Our identification method of displayed expressions is based on Garain's method [17], but additional features are proposed and the classification method is changed. We use the following three features (f_{ws}, f_{ms}, f_{mh}) proposed by Garain, and these features are slightly modified from original ones:

$$f_{ws} = \frac{r}{r_\mu},\ f_{ms} = \sigma_y,\ f_{mh} = \frac{h}{h_\mu}, \tag{1}$$

where r denotes the average of the white space (measured in number of pixel rows) above and below a text line and r_μ denotes the mean of the white space between two consecutive text lines. σ_y denotes the standard deviation among the y-coordinates of the lower-most pixels of the characters of a text line. h is the height of a text line in terms of pixel rows and h_μ is the mean of all h-values.

Additional new features are defined as follows:

$$f_{ma} = \sigma_a,\ f_{mi} = ind, \tag{2}$$

where σ_a denotes the standard deviation among the aspect ratios of the bounding boxes of the characters in a text line, and ind denotes the left indent (measured in number of pixel rows).

2) Proposal method: In Garain's method, the features get combined and mapped into a scalar value by using an averaging method. Then, the value is compared with a threshold determined empirically to identify a text line as an ordinary text or a displayed expression.

Our method uses the support vector machine (SVM) for classification. We adopt SVM^{light} [18] for the reason for license.

3) Training: Some documents are scanned and segmented into text lines by the OCRopus system, then each text line is labeled as ordinary text line or displayed expression manually. This dataset is used for training for the SVM.

E. Character recognition

Text lines classified as non displayed expressions are recognized by the OCRopus system.

F. Mathematical formula recognition

Text lines classified as displayed expressions are recognized by our mathematical formula recognition module. Figure 3 shows an example of the mathematical formula recognition result in LaTeX format.

$$\boxed{J(\boldsymbol{x}) := \det \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} = \prod_{1 \leqq i < j \leqq d} (x_i - x_j).}$$

(a) An example of displayed expression in a document

```
\begin{math}
J \left( x \right) : = \det
\frac{\displaystyle{\delta u}}{\displaystyle {\delta x}}
= \displaystyle{\prod _{{1 \leqq i < j \leqq d }}^{}}
\left( {x }_{i }^{}- {x }_{j }^{}\right) .
\end{math}
```

(b) The recognition result in LaTeX format

Figure 3. A mathematical formula recognition result for a displayed expression

G. Coding

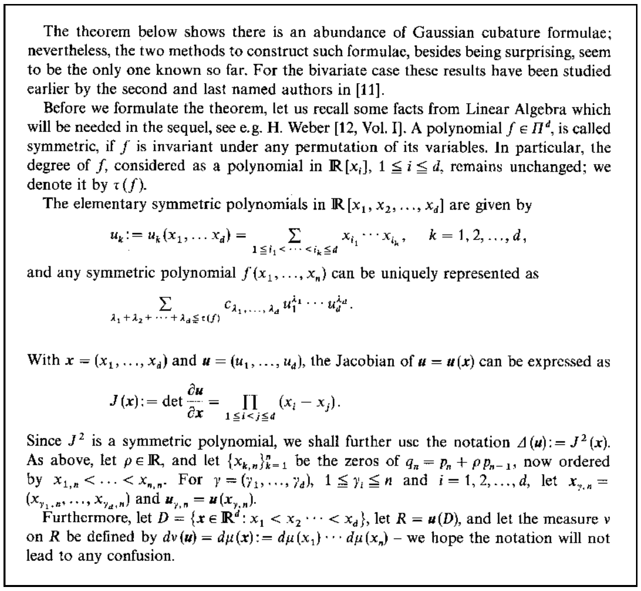
The overall document recognition steps are executed by the pseudo code in Algorithm 1.

```
Algorithm 1: Math OCR for displayed expressions
  Input: document image
  Output: overall recognition result
  image $\leftarrow$ preprocess( document image );
  pseg $\leftarrow$ segmenter$\langle$1CP$\rangle$.segment( image );
  textlines $\leftarrow$ extract( pseg );
  foreach line in textlines do
      height $\leftarrow$ line.height;
      white_space $\leftarrow$ mean( line.above_white_space,
      line.below_white_space );
      bboxes $\leftarrow$ character_segmentaton( line );
      ws $\leftarrow$ white_space / mean( textlines.white_space );
      ms $\leftarrow$ std_coord_y( bboxes.bottom_coord_y );
      mh $\leftarrow$ height / mean( textlines.height );
      ma $\leftarrow$ std_aspect( bboxes.width / bboxes.height );
      mi $\leftarrow$ line.left_coord_x;
      identification( ws, ms, mh, ma, mi );
      if line is displayed expression then
          math_OCR( line );
          output_result( line );
      else
          output_recognition_result( line );
      end
  end
```

The above source code and math OCR module can be found in public at http://syorserv.cs.shinshu-u.ac.jp/src/ocr.

IV. Experimental result

The scanned mathematical journal, *Archiv der Mathematik* from the project "*Retro-digitalization of mathematical journals, and their integration searchable digital libraries*" [19] are used for both of training and testing. The documents are



The theorem below shows there is an abundance of Gaussian cubature formulae; nevertheless, the two methods to construct such formulac, besides being surprising, seem to be the only one known so far. For the bivariate case thesc results have been studied earlier by the second and last named authors in [11].

Before we formulate the theorem, let us recall some facts from Linear Algebra which will be needed in the sequel, see e.g. H. Weber [12, Vol. I]. A polynomial $f \in \Pi^d$, is called symmetric, if f is invariant under any permutation of its variables. In particular, the degree of f, considered as a polynomial in $\mathbb{R}[x_i]$, $1 \leqq i \leqq d$, remains unchanged; we denote it by $\tau(f)$.

The elementary symmetric polynomials in $\mathbb{R}[x_1, x_2, \ldots, x_d]$ are given by

$$u_k := u_k(x_1, \ldots x_d) = \sum_{1 \leqq i_1 < \cdots < i_k \leqq d} x_{i_1} \cdots x_{i_k}, \qquad k = 1, 2, \ldots, d,$$

and any symmetric polynomial $f(x_1, \ldots, x_n)$ can be uniquely represented as

$$\sum_{\lambda_1 + \lambda_2 + \cdots + \lambda_d \leqq \tau(f)} c_{\lambda_1, \ldots, \lambda_d} u_1^{\lambda_1} \cdots u_d^{\lambda_d}.$$

With $\boldsymbol{x} = (x_1, \ldots, x_d)$ and $\boldsymbol{u} = (u_1, \ldots, u_d)$, the Jacobian of $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$ can be expressed as

$$J(\boldsymbol{x}) := \det \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} = \prod_{1 \leqq i < j \leqq d} (x_i - x_j).$$

Since J^2 is a symmetric polynomial, we shall further usc the notation $\Delta(\boldsymbol{u}) := J^2(\boldsymbol{x})$. As above, let $\rho \in \mathbb{R}$, and let $\{x_{k,n}\}_{k=1}^n$ be the zeros of $q_n = p_n + \rho p_{n-1}$, now ordered by $x_{1,n} < \cdots < x_{n,n}$. For $\gamma = (\gamma_1, \ldots, \gamma_d)$, $1 \leqq \gamma_i \leqq n$ and $i = 1, 2, \ldots, d$, let $\boldsymbol{x}_{\gamma,n} = (x_{\gamma_1,n}, \ldots, x_{\gamma_d,n})$ and $\boldsymbol{u}_{\gamma,n} = \boldsymbol{u}(\boldsymbol{x}_{\gamma,n})$.

Furthermore, let $D = \{\boldsymbol{x} \in \mathbb{R}^d : x_1 < x_2 \cdots < x_d\}$, let $R = \boldsymbol{u}(D)$, and let the measure ν on R be defined by $d\nu(\boldsymbol{u}) = d\mu(\boldsymbol{x}) := d\mu(x_1) \cdots d\mu(x_n)$ – we hope the notation will not lead to any confusion.

Figure 4. A part of an example test image

scanned in 600 dpi. Dataset for the training consists of 50 pages from the journal. The pages include 1074 ordinary text lines and 174 displayed expressions. The proposed system have tested on other 64 pages. Figure 4 shows a part of an example test document.

A. Identification results of mathematical formulas

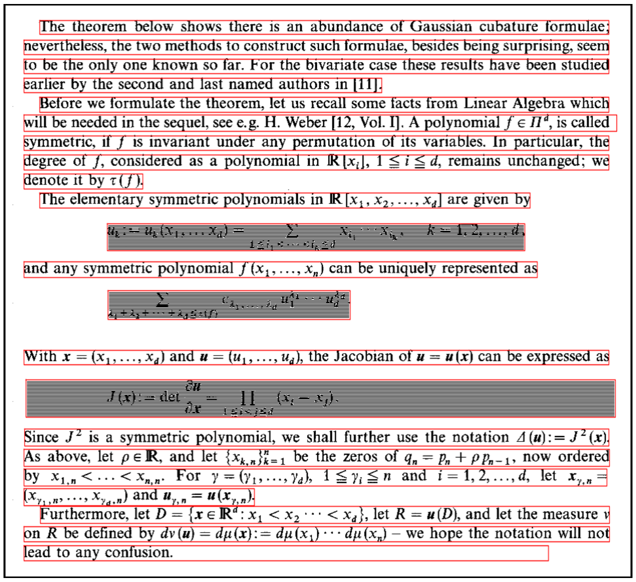
The test dataset includes 542 displayed expressions and 531 expressions are identified correctly. Figure 5 shows results of text line segmentation and identification of displayed expressions for the above test document. In Figure 5, black regions show text lines classified as displayed expressions. In this example, all displayed expressions are identified correctly.

B. Final recognition result

Figure 6 shows a part of the recognition result of the test document. In this trial, OCR results for ordinary text lines are represented by HTML format and displayed expressions are represented by MathML. The HTML source created by the OCRopus system is based on hOCR [20]. MathML is a markup language for describing mathematics aiming at inclusion of mathematical formulas in Web pages, so the HTML and MathML output is suitable to represent the whole document including mathematical formulas.

V. Conclusion and Future Work

The OCRopus system is adequate to embed newly developed modules. In this paper, embedding our mathematical formula recognition module into the OCRopus system is discussed. As the first step of this research, developing a math OCR for displayed expressions is intended. In this development, how to embed our mathematical formula recognition module into the OCRopus system is investigated. Then, identification of displayed expressions are studied. Some experiments show the possibility of developing a math OCR using the OCRopus system to convert scientific or technical documents into digital forms.



The theorem below shows there is an abundance of Gaussian cubature formulae; nevertheless, the two methods to construct such formulae, besides being surprising, seem to be the only one known so far. For the bivariate case these results have been studied earlier by the second and last named authors in [11].

Before we formulate the theorem, let us recall some facts from Linear Algebra which will be needed in the sequel, see e.g. H. Weber [12, Vol. I]. A polynomial $f \in \Pi^d$, is called symmetric, if f is invariant under any permutation of its variables. In particular, the degree of f, considered as a polynomial in $\mathbb{R}[x_i]$, $1 \leqq i \leqq d$, remains unchanged; we denote it by $\tau(f)$.

The elementary symmetric polynomials in $\mathbb{R}[x_1, x_2, \ldots, x_d]$ are given by

$$u_k := u_k(x_1, \ldots x_d) = \sum_{1 \leqq i_1 < \cdots < i_k \leqq d} x_{i_1} \cdots x_{i_k}, \qquad k = 1, 2, \ldots, d,$$

and any symmetric polynomial $f(x_1, \ldots, x_n)$ can be uniquely represented as

$$\sum_{\lambda_1 + \lambda_2 + \cdots + \lambda_d \leqq \tau(f)} c_{\lambda_1, \ldots, \lambda_d} u_1^{\lambda_1} \cdots u_d^{\lambda_d}.$$

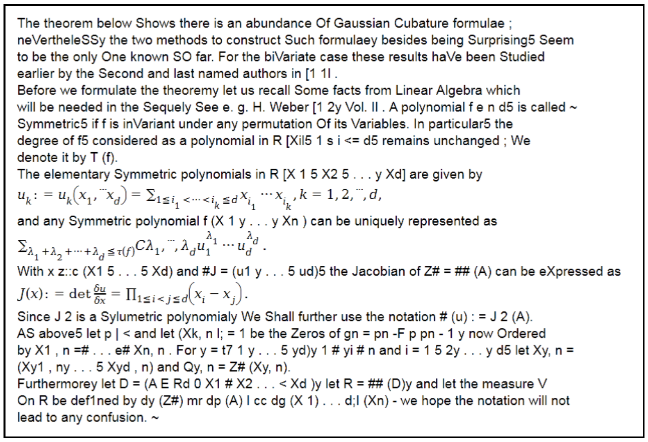
With $\boldsymbol{x} = (x_1, \ldots, x_d)$ and $\boldsymbol{u} = (u_1, \ldots, u_d)$, the Jacobian of $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$ can be expressed as

$$J(\boldsymbol{x}) := \det \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} = \prod_{1 \leqq i < j \leqq d} (x_i - x_j).$$

Since J^2 is a symmetric polynomial, we shall further use the notation $\Delta(\boldsymbol{u}) := J^2(\boldsymbol{x})$. As above, let $\rho \in \mathbb{R}$, and let $\{x_{k,n}\}_{k=1}^n$ be the zeros of $q_n = p_n + \rho p_{n-1}$, now ordered by $x_{1,n} < \cdots < x_{n,n}$. For $\gamma = (\gamma_1, \ldots, \gamma_d)$, $1 \leqq \gamma_i \leqq n$ and $i = 1, 2, \ldots, d$, let $\boldsymbol{x}_{\gamma,n} = (x_{\gamma_1,n}, \ldots, x_{\gamma_d,n})$ and $\boldsymbol{u}_{\gamma,n} = \boldsymbol{u}(\boldsymbol{x}_{\gamma,n})$.

Furthermore, let $D = \{\boldsymbol{x} \in \mathbb{R}^d : x_1 < x_2 \cdots < x_d\}$, let $R = \boldsymbol{u}(D)$, and let the measure ν on R be defined by $d\nu(\boldsymbol{u}) = d\mu(\boldsymbol{x}) := d\mu(x_1) \cdots d\mu(x_n)$ – we hope the notation will not lead to any confusion.

Figure 5. Text line segmentation and identification of displayed expressions



The theorem below Shows there is an abundance Of Gaussian Cubature formulae ;
neVertheleSSy the two methods to construct Such formulaey besides being Surprising5 Seem
to be the only One known SO far. For the biVariate case these results haVe been Studied
earlier by the Second and last named authors in [1 1l .
Before we formulate the theoremy let us recall Some facts from Linear Algebra which
will be needed in the Sequely See e. g. H. Weber [1 2y Vol. Il . A polynomial f e n d5 is called ~
Symmetric5 if f is inVariant under any permutation Of its Variables. In particular5 the
degree of f5 considered as a polynomial in R [Xil5 1 s i <= d5 remains unchanged ; We
denote it by T (f).
The elementary Symmetric polynomials in R [X 1 5 X2 5 . . . y Xd] are given by
$u_k: = u_k(x_1, ^{\cdots} x_d) = \Sigma_{1 \leqq i_1 < \cdots < i_k \leqq d} x_{i_1} \cdots x_{i_k}, k = 1, 2, ^{\cdots}, d,$
and any Symmetric polynomial f (X 1 y . . . y Xn) can be uniquely represented as
$\Sigma_{\lambda_1 + \lambda_2 + \cdots + \lambda_d \leqq \tau(f)} C\lambda_1, ^{\cdots}, \lambda_d u_1^{\lambda_1} \cdots u_d^{\lambda_d}.$
With x z::c (X1 5 . . . 5 Xd) and #J = (u1 y . . . 5 ud)5 the Jacobian of Z# = ## (A) can be eXpressed as
$J(x): = \det \frac{\delta u}{\delta x} = \Pi_{1 \leqq i < j \leqq d}(x_i - x_j).$
Since J 2 is a Sylumetric polynomialy We Shall further use the notation # (u) : = J 2 (A).
AS above5 let p | < and let (Xk, n l; = 1 be the Zeros of gn = pn -F p pn - 1 y now Ordered
by X1 , n =# . . . e# Xn, n . For y = t7 1 y . . . 5 yd)y 1 # yi # n and i = 1 5 2y . . . y d5 let Xy, n =
(Xy1 , ny . . . 5 Xyd , n) and Qy, n = Z# (Xy, n).
Furthermorey let D = (A E Rd 0 X1 # X2 . . . < Xd)y let R = ## (D)y and let the measure V
On R be def1ned by dy (Z#) mr dp (A) l cc dg (X 1) . . . d;l (Xn) - we hope the notation will not
lead to any confusion. ~

Figure 6. A part of recognition results for the test document by Firefox [21]

Future work will involve investigating a method to identify inline expressions in ordinary text lines, and development of a math OCR for whole scientific or technical documents.

Acknowledgment

This work is funded by Grants-in-Aid for Scientific Research (C) (21500163) from Japan Society for the Promotion of Science (JSPS).

References

[1] K.-F. Chan and D.-Y. Yeung, “Mathematical expression recognition: a survey,” *IJDAR*, vol. 3, pp. 3–15, 2000.

[2] U. Garain and B. Chaudhuri, “A corpus for ocr research on mathematical expressions,” *IJDAR*, vol. 7, pp. 241–259, 2005.

[3] T. M. Breuel, “The ocropus open source ocr system,” in *Proc. IS&T/SPIE 20th Annual Symposium*, 2008.

[4] T. Breuel, “Recent progress on the ocropus ocr system,” in *Proc. of the International Workshop on Multilingual OCR*, 2009, pp. 1–10.

[5] O. Masayuki and H. Hiroyuki, “Mathematical expression recognition by the layout of symbols,” *The transactions of the Institute of Electronics, Information and Communication Engineers*, vol. 78, no. 3, pp. 474–482, 1995.

[6] H. M. Twaakyondo and M. Okamoto, “Structure analysis and recognition of mathematical expressions,” in *ICDAR ’95*, vol. 1, 1995, pp. 430–437.

[7] M. Okamoto, H. Imai, and K. Takagi, “Performance evaluation of a robust method for mathematical expression recognition,” in *ICDAR ’01*, 2001, pp. 121–128.

[8] M. Suzuki, F. Tamari, R. Fukuda, S. Uchida, and T. Kanahori, “Infty: an integrated ocr system for mathematical documents,” in *DocEng ’03: Proc. of the 2003 ACM symposium on Document engineering*, 2003, pp. 95–104.

[9] “OCRopus,” http://code.google.com/p/ocropus/.

[10] “MathML,” http://www.w3.org/Math/.

[11] T. Breuel, “High performance document layout analysis,” in *Proc. of the Symposium on Document*, 2003.

[12] R. J. Fateman, “How to find mathematics on a scanned page,” in *Proc. SPIE*, 1999, pp. 98–109.

[13] J. Toumit and H. Emptoz, “From the segmentation to the reading of a mathematical document,” *GKPO ’98, Machine Graphics and Vision, Borki, Poland*, pp. 483–504, 1998.

[14] A. Kacem, A. Belaid, and M. Ben Ahmed, “Automatic extraction of printed mathematical formulas using fuzzy logic and propagation of context,” *IJDAR*, vol. 4, no. 2, pp. 97–108, 2001.

[15] S. P. Chowdhury, S. Mandal, A. K. Das, and B. Chanda, “Automated segmentation of math-zones from document images,” in *Proc. of ICDAR ’03*, vol. 2, 2003, p. 755.

[16] J. Jin, X. Han, and Q. Wang, “Mathematical formulas extraction,” in *Proc. of ICDAR ’03*, vol. 2, 2003, p. 1138.

[17] U. Garain, “Identification of mathematical expressions in document images,” in *Proc. of ICDAR ’09*, 2009, pp. 1340–1344.

[18] T. Joachims, *Making Large-Scale SVM Learning Practical*, B. Schlkopf, C. Burges, and A. Smola, Eds. MIT – Press, 1999.

[19] G. O. Michler, “How to build a prototype for a ditributed digital mathematics archive library,” in *Annals of Mathematics and Artificial Intelligence*, 2003, pp. 137–164.

[20] T. Breuel, “The hocr microformat for ocr workflow and results,” in *Proc. of ICDAR ’07*, vol. 2, 2007, pp. 1063–1067.

[21] “Firefox,” http://www.firefox.com/.