Math Spotting: Retrieving Math in Technical Documents Using Handwritten Query Images

Richard Zanibbi
Department of Computer Science
Rochester Institute of Technology
Rochester, NY, USA
rlaz@rit.edu

Li Yu
Department of Computer Science
Illinois Institute of Technology
Chicago, IL, USA
lyu17@iit.edu

***Abstract*—A method for locating mathematical expressions in document images without the use of optical character recognition is presented. An index of document regions is produced from recursive X-Y trees produced for each page in the corpus. Queries are provided as images of handwritten expressions, for which an X-Y tree is computed. During retrieval, the query is looked up in the document region index using features of its X-Y tree, producing a set of candidate regions. Candidate regions are ranked by the similarity of vertical pixel projections in their upper and lower halves with those of the query image, as computed using Dynamic Time Warping of the image columns. In an experiment, ten participants each wrote twenty queries from a 200-page corpus. On average, the top-10 retrieval candidates included a candidate covering 43.3% of the test query image ($\sigma = 14.0$), with the correct page being returned between 30.0% and 85.0% of the time across participants ($\mu = 63.2\%$, $\sigma = 14.9\%$). When testing using the original query images, 90.0% of the queries were retrieved correctly.**

***Keywords*-Mathematical Information Retrieval; Math Recognition; Keyword Spotting**

I. Introduction

Most documents do not contain markup for math expressions, not even vector-based electronic document encodings such as Portable Document Format (.pdf) files. One way to address the problem is to insert annotations for math into existing files. Kanahori and Suzuki propose a method for doing this by inserting LaTeX into .pdf files produced by a commercial Optical Character Recognition (OCR) system [1], after passing detected math regions through the Infty math recognition system [2]. A number of methods for recognizing typeset and handwritten mathematics have been devised, and surveys on the subject are available [3]–[5].

Existing Mathematical Information Retrieval systems require queries in the form of a string (e.g. LaTeX and text [6], MathML [7], or Lisp [8]), provided manually or using a graphical template editor (with templates for fractions, summations, arrays, etc.). We believe that many users would prefer to simply draw expressions by hand, and query using the appearance of the expression, as shown in Figure 1. Further, we hypothesize that handwritten queries may be retrieved from technical document images using page segmentation and image-similarity algorithms, without the use of optical character recognition, similar to keyword spotting techniques [9], [10].

The contributions of this paper include: 1) to our knowledge, the first system for querying math in technical documents using images of handwritten queries (image-based retrieval of isolated LaTeX expressions has been explored [11]), and 2) an experimental validation of the proposed technique. We summarize the underlying segmentation and image matching algorithms in Sections II and III, provide the indexing and retrieval model in Section IV, present an experiment testing our model with handwritten and image queries in Section V, and conclude in Section VI.

II. Page and Query Segmentation: X-Y Cutting

X-Y trees represent a hierarchically nested set of rectangular regions (see Figure 2). Nodes represent image regions, with the whole image at the root. Children of a node represent subregions obtained by cutting the image at gaps in a horizontal or vertical histogram of pixel intensities. In standard X-Y trees, cuts are made at all gaps in the pixel projection profile (histogram) (see Figure 2(c)), with cuts alternating in the vertical and horizontal directions (commonly starting with the vertical direction). In recursive X-Y trees, a *single* cut is made either horizontally or vertically, depending on the direction with the largest projection gap (see Figure 2(b)). This produces a binary tree.

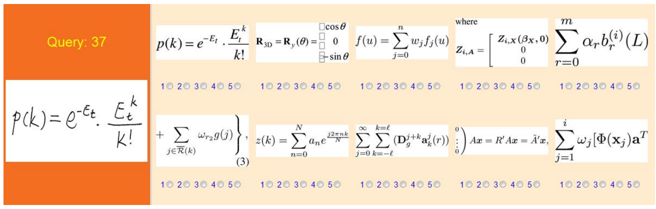
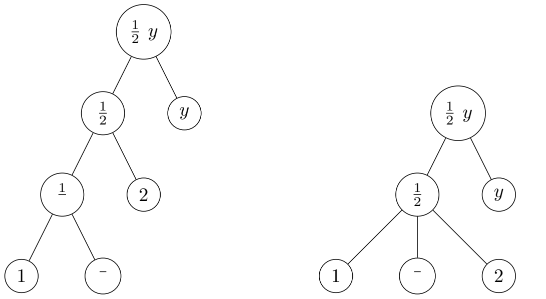


Figure 1. Retrieval Evaluation Interface. The regions shown correspond to the ten best-matching document pages, with the region producing the best match shown. Results are listed in decreasing order of rank (top row: ranks 1-5, bottom row: ranks 6-10)

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.96



(b) Recursive X-Y Tree (c) Standard X-Y Tree

Figure 2. Recursive [15] and Standard [14] X-Y cutting of the expression: $\frac{1}{2}\ y$. Recursive X-Y cutting splits the image at the largest horizontal or vertical projection gap at each node, while standard X-Y cutting alternates in the vertical and horizontal direction, cutting at all projection gaps

Our retrieval method was inspired by the projection-profile-based structural analysis methods of Okamoto et al. [12], [13] for mathematical notation. It occurred to the authors that 1) Okamoto's et al.'s projection-based technique may be understood as a variation of X-Y cutting algorithms used to segment pages for document analysis and recognition [14], [15], and 2) due to the two-dimensional arrangement of symbols in mathematical expressions, X-Y trees for math expressions would often differ significantly from those for text regions, and 3) X-Y trees provide some invariance to the scale and relative sizes of symbols, as they describe only the topology (relative position) of regions in an image. This suggested that one might meaningfully compare X-Y trees for handwritten queries to those for page regions, and return page regions with similar X-Y structure.

In our approach, candidate query match regions are obtained using the (coarse) similarity between recursive and a restricted (depth-two) standard X-Y trees for the query and page regions, along with a simple edge distance feature (see Section IV).

To avoid producing noisy trees, a threshold is often used to filter narrow cuts. Cutting thresholds may be defined using estimates for dominant character heights and widths [16], [17]. We use a minimum cut width of 2 pixels for the two top-level standard X-Y cuts performed on each region; in our experiments handwritten expressions were written on paper and then scanned, with very little noise remaining after binarization.

III. Image Matching: Dynamic Time Warping

Candidate regions are ranked by visual similarity, making our approach a form of content-based image retrieval [18]. Early on we considered using a measure based on tree edit distance to match query and candidate X-Y trees [19], but abandoned this due to the computational cost involved. Marinai et al. came to a similar conclusion in their work on X-Y tree-based document image retrieval [20], where originally they employed tree edit distance.

Using the University of Washington III Database [21], we tried a number of different image distance metrics, of which a form of Dynamic Time Warping (DTW) was most effective [22]. The DTW metric that we use is the minimum-cost alignment between columns of query and candidate region images, after candidates are scaled so that query and candidate image heights match (preserving the aspect ratio of each). For the image columns, we use features similar to those used by Rath and Manmatha [10] for spotting words in historical documents. We first compute binary pixel projection profiles for the top and bottom half of an image, normalizing them by the image height so that each projection value lies in the interval $[0, 0.5]$. Each column is then represented by its values in the upper and lower profiles (u, l). To reduce computational cost, we sub-sample the upper and lower profiles, using the average upper/lower half profile distances for every *five* columns (this value was chosen empirically, again using the UW-III database). For images with widths that are not a multiple of five, the average value of the remaining columns is stored in the final feature vector element.

Formally, the dissimilarity between the query and candidate feature vectors (F_Q, F_C) is given by $D(|F_Q|, |F_C|)$, the minimum cost alignment between the averaged projection profiles.

$$D(i,j) = min \left\{ \begin{array}{c} D(i-1,j) \\ D(i,j-1) \\ D(i-1,j-1) \end{array} \right\} + d(i,j) \qquad (1)$$

$$d(i,j) = (u(F_Q[i]) - u(F_C[j]))^2 + (l(F_Q[i]) - l(F_C[j]))^2 \qquad (2)$$

where $D(0,0) = 0$, $D(x,0) = \infty$ for $1 \leq x \leq |F_Q|$, and $D(0,y) = \infty$ for $1 \leq y \leq |F_C|$. The distance between a pair of feature vector elements $d(i,j)$ is the sum of squared differences between the upper and lower projection values. Unlike Rath and Manmatha [10], we do not constrain the warping path, nor do we normalize the DTW distance by the length of the minimum cost warping path. This is in part because many of the regions to be compared against are nested in the X-Y tree, and we do not mind penalizing longer warping paths as a result. The complexity of the distance computation is $O(|F_Q||F_C|)$.

IV. Document Indexing and Retrieval

We apply recursive X-Y cutting to each document page to be indexed. All nodes in the X-Y tree with fewer than 90 nodes and a depth of at least two are stored in the index. This avoids indexing regions that have many more connected components than common expressions [23] and expressions that are very small (there at most four connected components in a recursive X-Y tree of depth two). Note that we do not make use of the directions of cuts.

Each region in the index is cut again using two standard X-Y cuts: just one vertical and one horizontal cut (see

Section II). This provides a simple high-level description of the structure of the expressions. Finally, using the upper and lower contours for each region, we compute the maximum relative offset for the upper and lower contours, and then record the smaller of these two offsets. More formally, for each column $\{I_1, I_2, ...I_n\}$, the distance from both the top edge and bottom edge are calculated and saved in two sets: $top\{I_1, I_2, ...I_n\}$ and $bottom\{I_1, I_2, ...I_m\}$. We then calculate the smaller maximum offset as R_o :

$$R_o = \frac{min(max(top), max(bottom))}{R_h} \tag{3}$$

where R_h is the region height. The expectation here is that text regions will often have small R_o values, particularly when the bottom of the text rests directly on the writing line (e.g. for the word ‘and’).

Each indexed region is represented by a vector of five features, and then organized into a table. An illustration of the resulting index is shown in Figure 3. The features used are: R_d: Recursive X-Y tree depth, R_s: Recursive X-Y tree size (number of nodes), R_x: Top-level standard vertical X-Y cut elements (entire image), R_y: Top-level standard horizontal X-Y cut elements (entire image), and R_o: Smaller of maximum offsets in upper and lower contours. Entries in the last level in the index are sorted by their vertical offset value (R_o).

For each query image Q, we compute the same five properties for document regions as described above, which we similarly name Q_d, Q_s, Q_x, Q_y, and Q_o. We then recover the set of candidates C_Q from the index table using four tolerances ($\alpha, \beta, \gamma, \delta$: three integers and a floating-point value), plus a filter to remove regions with an aspect ratio differing significantly from the query. Given query Q, an indexed region $R \in C_Q$ iff:

1) Recursive X-Y tree: $R_d \in Q_d \pm \alpha \wedge R_s \in Q_s \pm \beta$
2) Standard X-Y cuts: $R_x \in Q_x \pm \gamma \wedge R_y \in Q_y \pm \gamma$
3) Contour offset: $R_o \in Q_o \pm \delta$
4) Aspect ratio: $Q_a/2 \leq R_a \leq 2Q_a$

A single value (γ) is used to define the tolerance for top-level X-Y cuts in both directions. Once the candidate set C_Q has been defined, *document pages containing candidates* are ranked in decreasing order of visual similarity with the query. Each page containing a candidate region is scored by the smallest DTW distance between the query and a page region, as described in Section III.

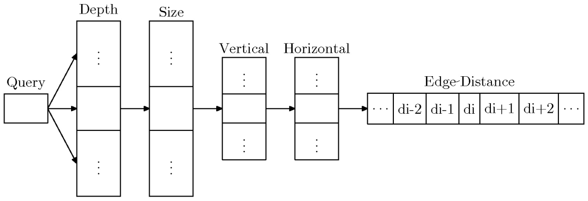


Figure 3. Querying the Document Region Index

V. Experiment

Training and test document pages were taken from the IEEE CVPR 2008 conference, converted from .pdf to .jpg at 300 dpi. 400 pages were then randomly selected, with 200 for training (68 containing math), and 200 for testing (with 61 pages containing math). One expression is selected from each page containing math. If a page contains more than one ‘type’ of expression (with primarily horizontal, vertical, or roughly equal X-Y cutting gaps), we manually extracted one of the type with the fewest representatives. Nearly all selected expressions were offset from the main text (displayed). This expression set was then sampled randomly to produce 20 training and 20 test queries. Test queries used in the experiment are shown in Figure 4. The indexing and retrieval algorithms were implemented in C++ using the OpenCV library [24].

In the training phase, we select values for the search tolerances using twenty expressions handwritten by ten participants. We use a search over parameter values in order to find a set of values that maximizes the average query region recall (defined in Table I). We fixed the tolerance γ for the number of regions in standard X-Y cuts (Q_x, Q_y, R_x, and R_y) at $\gamma = 2$. We then searched over the following parameter ranges: $\alpha \in \{0, 2, 4, 6, 8\}$, $\beta \in \{0, 2, 4, 8, 16\}$ and $\delta \in \{0.05, 0.125, 0.125, 0.2\}$.

In the testing phase, we use the selected tolerances and observe the retrieval performance for a distinct set of twenty expressions handwritten by the same ten participants. For comparison, we also observed retrieval performance for the original query images from the test corpus. The metrics used to assess performance are summarized in Table I. We measure retrieval accuracy using the top n candidates returned ($n = \{1, 5, 10\}$). Note that our page and region recall metrics are conservative: we record matches only where the exact query region and page is returned in the top-n results. This means we do not count matches to similar or identical expressions at different locations on the same page, or on a different page.

Participants evaluated retrieval results for their queries using a web-based interface, a portion of which is of shown in Figure 1. At the top of this web page, a 5-point Likert scale is described, which characterizes the match between a query and a candidate region: 1) No match, 2) Less than half the query is matched, 3) Roughly half the query is matched, 4) More than half the query is matched, and 5) The query is completely matched. Participants were able to click on query and region images and view them in a separate tab of their browser window.

A. Results

In the training phase, the average query region recall was maximized at $\alpha = 4$, $\beta = 16$, and $\delta = \{0.2\}$. The index

$$\frac{\kappa}{4\pi\sinh\kappa}\cosh(\kappa\boldsymbol{\mu}^T\mathbf{x})$$

(a) Query 1

$$f(x,y) \approx \sum_{j=0}^{n} a_j\phi_j(x,y)$$

(b) Query 2

$$\frac{35x^4 - 30x^2 + 3}{8}$$

(c) Query 3

$$\sum_{k=1}^{K} \boldsymbol{\pi}(k)\mathbf{p}_k(x)$$

(d) Query 4

$$\nu = \frac{2\nu_+\nu_-}{\nu_+ + \nu_-}$$

(e) Query 5

$$u_i = \sum_{x\in S_i} p(x) \in [0,1]$$

(f) Query 6

$$\frac{dc_i}{dt} = -\lambda_i^\alpha c_i$$

(g) Query 7

$$\mathbf{f}_{\mathrm{s}} = \frac{1}{2\pi}\int_0^{2\pi} g_\phi \mathbf{f}\, d\phi.$$

(h) Query 8

$$\int_{X\times X} e_\varphi(x,x')dxdx'$$

(i) Query 9

$$A(\widehat{W}_A - V_A) + N(W_N - V_N)$$

(j) Query 10

$$-\log r(g|\partial g)$$

(k) Query 11

$$\frac{1}{2\mu_0}\exp(-\frac{|\delta(u)|}{\mu_0})$$

(l) Query 12

$$\left(\frac{1}{a^2} + \frac{1}{b^2}\right)$$

(m) Query 13

$$\Delta[n] = \hat{r}_{1,K}^{(1)}[n] - \hat{r}_{1,K}^{(2)}[n],$$

(n) Query 14

$$C^{(+)} = \begin{pmatrix} P & Q \\ Q^T & R \end{pmatrix}^{-1}$$

(o) Query 15

$$\frac{1}{1+\exp\{Af(x)+B\}}$$

(p) Query 16

$$p(k) = e^{-E_t} \cdot \frac{E_t^{\,k}}{k!}$$

(q) Query 17

$$\frac{1}{2}(f_i - f_{\hat{j}})$$

(r) Query 18

$$\frac{1}{M}\cdot\mathbf{B}\cdot\underline{\mathbf{P}}(Y_m = n)$$

(s) Query 19

$$\boldsymbol{g} = \begin{pmatrix} 1 & 0 \\ 0 & \sin^2\theta \end{pmatrix}$$

(t) Query 20

Figure 4. Test Queries Sampled from 200 Pages of the CVPR 2008 Proceedings

Table I
PERFORMANCE METRICS. EXPERIMENTS WERE RUN ON AN IBM THINKPAD (INTEL CORE2 DUO CPU T9600 (2.80GHZ), 4GB RAM) RUNNING THE 32-BIT WINDOWS XP OPERATING SYSTEM

Metric	Definition
P_{recall}	Percentage of queries returning the page a query was taken from
A_{recall}	Maximum percentage area of a query overlapped by a top-n candidate
$Part.$	The highest participant rating (1-5) for a candidate in the top-n matches

Table II
RETRIEVAL ACCURACY FOR TEST SET ($\alpha = 4$, $\beta = 16$, $\gamma = 2$, $\delta = 0.2$). RESULTS ARE SHOWN FOR 20 QUERIES WRITTEN BY TEN PARTICIPANTS, ALONG WITH THE ORIGINAL QUERY IMAGES (SEE FIGURE 4)

	P_{recall} (%)		A_{recall} (%)		$Part.$ (1-5)	
Top	μ	σ	μ	σ	μ	σ
	HANDWRITTEN QUERY IMAGES					
1	38.6	11.7	26.7	13.3	2.06	0.63
5	54.9	14.2	39.8	13.8	2.97	0.77
10	63.2	14.9	43.3	14.0	3.15	0.71
	ORIGINAL QUERY IMAGES					
1			90.0	30.0	4.65	0.08
5			90.0	30.0	4.83	0.05
10			90.0	30.0	4.83	0.05

for the 200 training pages contained 272,465 regions, with 2.77% of the regions were being selected as candidates on average for the training query set. Tolerances for region matching were set as ($\alpha = 4,\ \beta = 16,\ \gamma = 2,\ \delta = 0.2$).

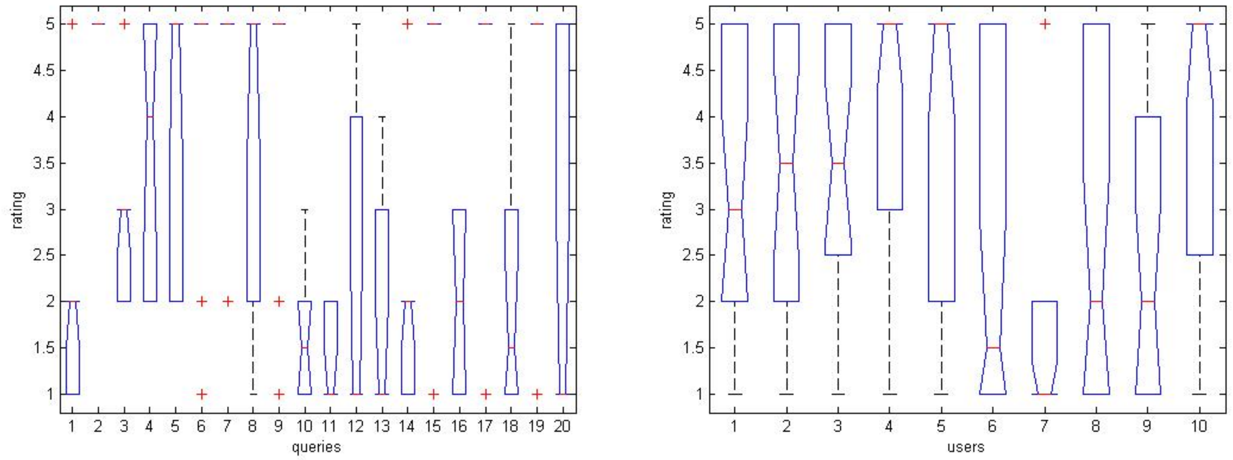
Table II shows the mean and standard deviation for the highest participant ratings for the top-1, top-5, and top-10 query results, along with the page and region match metrics computed offline (these metrics are defined in Table I). The average highest participant match rating for a top-10 candidate is 3.15, where 3 represents a candidate matching roughly half the query; this corresponds well to the actual average query region match (43.3%). All three accuracy metrics increase as we move from top-1 to top-10.

Retrieval of original queries was highly effective; for all but two queries (queries 13 and 16), the original query region was completely recovered. For queries 13 and 16, they produced no match at all in the top-10 regions, producing a mean A_{recall} of 90%, and leading to a high standard deviation in recall for region matching (30%). The average user ratings for the original queries are also very high. We inspected the results, and found that query 13 did have a rank 3 candidate that was very similar, '$(\frac{1}{a^2} - \frac{1}{b^2})xy$'. Though there were a number of fractions returned for query 16 (7 of the 10 candidates returned), none of the candidates closely matched.

Figure 5 illustrates box plots for the distributions of participant ratings of query results. Median values are shown as red horizontal lines, with the middle half of the values in the boxed regions. Outliers are marked using '+'. For a number of queries (numbers 2, 6, 7, 9, 15, 17, and 19), we can see that almost all match ratings were 5's, with the exception of one or two outliers. Queries 11, 12, 13 and 20 had median ratings of 1 (no match), though there is more variation in responses here than for the best recovered queries. Eight queries have a median ranking of 1 or 2. Only one query had a median ranking of 3 (query 3).

Across participants there is significant variation in the ratings assigned to the best of the top-10 query results. Participant 7 in particular provided a median ranking of 1. Half of the participants have a median rating greater than 3 (i.e. indicating matches of more than 1/2 the query), one has a median value of 3 (half the query), and four have ratings less than 3, but with large variations in ratings other than for participant 7.

For page-level retrieval rates, participant 7 had the lowest top-10 rate (30.0%), while participant 3 had the highest (85.0%), with an average retrieval rate of 63.2% across the ten participants ($\sigma = 14.9\%$), as shown in Table II.



(a) Ratings by query (see Figure 4)

(b) Ratings by participant

Figure 5. Participant Ratings for Handwritten Test Query Results

B. Discussion

The results confirm our hypothesis that it is possible to recover math from typeset technical documents using handwritten queries. However, the recursive X-Y cutting method we are using can be brittle; in particular, for handwritten expressions, the gaps between symbols may vary significantly from a typeset expression, producing recursive X-Y trees that greatly differ. The X-Y cutting performed particularly poorly at producing cuts for one of our participants (participant 7), who used a compact writing style in which symbols overlapped a great deal (i.e. there was a lot of kerning of symbols). We might try cutting at angles other than 0 and 90°, and/or use different cutting termination conditions, allowing connected components to be removed when a cut cannot be made [25].

It is well-known that segmenting displayed/offset expressions is much easier than embedded expressions, and additional work is needed to address this difficult problem. Strategies used for detecting embedded expressions in document images include coarse classification of connected components followed by region growing around detected operators [26], and exploiting symbol n-grams in OCR output for textlines with and without mathematical expressions [27]; this n-gram information is paired with geometric features and the number of occurrences for a set of common mathematical operators (e.g. ‘=’, ‘+’) in the context of another region-growing algorithm. One might combine these strategies with X-Y cutting to produce a hybrid detection algorithm that can handle embedded expressions. Textline n-grams could be augmented with n-grams for linearized trees describing symbol layout in expressions vs. text (see Watt [28]), and/or similar statistics characterizing spatial arrangements of visual features in X-Y trees.

VI. Conclusion

We have adapted word spotting techniques to the problem of retrieving mathematical expressions in technical documents, using handwritten queries. Document regions of potential interest are stored in an index organized around X-Y tree properties. Candidates are retrieved by comparing X-Y tree properties of a *handwritten* query to entries in the index, after which candidates are ranked using Dynamic Time Warping (DTW) of image columns, based on pixel projection profiles for the upper and lower halves of each query and page region image. An experiment was presented in Section V, demonstrating that one might produce useful top-10 results using this simple method.

We are curious whether our technique is language dependent; for example, can this method work for technical documents in Mandarin? It would also be worth investigating whether our method can be easily adapted to detect other objects such as chemical diagrams, tables, figures, and text in technical documents.

Acknowledgements: Our thanks to David Snyder for proofreading this paper. This research was supported by the Xerox Foundation, and we wish to thank William Stumbo for his assistance. This material is based upon work supported by the National Science Foundation under Grant No. IIS-1016815.

References

[1] T. Kanahori and M. Suzuki, “Refinement of digitized documents through recognition of mathematical formulae,” in *Proc. 2nd Int’l W. Document Image Analysis for Libraries*, Lyon, France, 2006, pp. 27–28.

[2] M. Suzuki, T.Kanahori, N.Ohtake, and K.Yamaguchi, "An integrated ocr software for mathematical documents and its output with accessibility," in *Computers Helping people with Special Needs, 9th Int'l Conf. ICCHP2004*, ser. LNCS, vol. 3119. Paris: Springer, 2004, pp. 648–655.

[3] D. Blostein and A. Grbavec, "Recognition of mathematical notation," in *Handbook of Character Recognition and Document Image Analysis*. World Scientific Publishing Company, 1997, pp. 557–582.

[4] K.-F. Chan and D.-Y. Yeung, "Mathematical expression recognition: a survey," *International Journal on Document Analysis and Recognition*, vol. 3, pp. 3–15, Aug 2000.

[5] U. Garain and B. Chaudhuri, "OCR of printed mathematical expressions," in *Digital Document Processing*. Springer, 2007, pp. 235–259.

[6] B. R. Miller and A. Youssef, "Technical aspects of the digital library of mathematical functions," *Annals of Mathematics and Artificial Intelligence*, vol. 38, pp. 121–136, May 2003.

[7] M. Kohlhase and I. Sucan, "A search engine for mathematical formulae," in *Proceedings of Artificial Intelligence and Symbolic Computation, AISC 2006*, ser. LNAI, T. Ida, J. Calmet, and D. Wang, Eds., no. 4120. Springer Verlag, 2006, pp. 241–253.

[8] T. H. Einwohner and R. J. Fateman, "Searching techniques for integral tables," in *Proc. 1995 Int'l Symp. Symbolic and Algebraic Computation - ISSAC 95*. ACM Press, 1995, pp. 133–139.

[9] D. Doermann, "The indexing and retrieval of document images: a survey," *J. Comput. Vis. Image Underst.*, vol. 70, pp. 287–298, June 1998.

[10] T. Rath and R. Manmatha, "Word image matching using dynamic time warping," in *CVPR (2)*, 2003, pp. 521–527.

[11] R. Zanibbi and B. Yuan, "Keyword and image-based retrieval for mathematical expressions," in *Proc. Document Recognition and Retrieval XVIII*. San Francisco, CA: SPIE, Jan. 2011.

[12] N. Okamoto and B. Miao, "Recognition of mathematical expressions by using the layout structures of symbols," *Proc. First Int'l Conf. Document Analysis and Recognition*, pp. 242–250, 1991.

[13] H. M. Twaakyondo and M. Okamoto, "Structure analysis and recognition of mathematical expressions," in *Proc. Int'l Conf. Document Analysis and Recognition*, vol. 1, Montréal, Canada, 1995.

[14] G. Nagy and S. Seth, "Hierarchical representation of optically scanned documents," *Proc. of ICPR*, pp. 347–349, 1984.

[15] J. Ha, R. M. Haralick, and I. T. Phillips, "Recursive X-Y cut using bounding boxes of connected components," *Proceedings of the Third International Conference on Document Analysis and Recognition*, vol. 2, pp. 952–955, 1995.

[16] Y. Zheng, S. Member, H. Li, and D. Doermann, "Machine printed text and handwriting identification in noisy document images," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 26, no. 3, 2004.

[17] F. Shafait, D. Keysers, and T. M. Breuel, "Performance evaluation and benchmarking of six-page segmentation algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 6, pp. 941–954, 2008.

[18] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image retrieval: Ideas, influences, and trends of the new age," *ACM Comput. Surv.*, vol. 40, no. 2, pp. 1–60, 2008.

[19] L. Yu and R. Zanibbi, "Math spotting in technical documents using handwritten queries," in *Proc. Work. Pen-Based Mathematical Computation*, Grand Bend, Canada, July 2009, (abstract).

[20] S. Marinai, E. Marino, and G. Soda, "Layout based document image retrieval by means of XY tree reduction," vol. 1, Aug.-1 Sept. 2005, pp. 432–436.

[21] I. Phillips, "Methodologies for using UW databases for OCR and image understanding systems," *Proc. Document Recognition V*, vol. 3305, pp. 112–127, 1998.

[22] L. Yu, "Image-based math retrieval using handwritten queries," Master's thesis, Rochester Institute of Technology, NY, USA, April 2010.

[23] S. Kamali and F. Tompa, "Improving mathematics retrieval," in *Proc. DML 2009: Towards a Digital Mathematics Library*, Grand Bend, Canada, July 2009, pp. 37–48.

[24] G. Bradski and A. Kaehler, *Learning OpenCV*. Cambridge, MA: O'Reilly, 2008.

[25] A. Raja, M. Rayner, A. P. Sexton, and V. Sorge, "Towards a parser for mathematical formula recognition," in *Mathematical Knowledge Management (MKM)*, ser. LNAI, J. M. Borwein and W. M. Farmer, Eds., vol. 4108. Springer Verlag, Berlin, Germany, 2006, pp. 139–151.

[26] A. Kacem, A. Belaid, and M. Ben Ahmed, "Automatic extraction of printed mathematical formulas using fuzzy logic and propagation of context," *International Journal on Document Analysis and Recognition*, vol. 4, pp. 97–108, Dec 2001.

[27] U. Garain, "Identification of mathematical expressions in document images," in *Prof. Int'l Conf. Document Analysis and Recognition*, Barcelona, Spain, July 2009, pp. 1340–1344.

[28] S. M. Watt, "An empirical measure on the set of symbols occurring in engineering mathematics texts," in *Proc. 8th IAPR International Workshop on Document Analysis Systems, (DAS 2008)*. Nara, Japan: IEEE Computer Society, 2008, pp. 557–564.