A System for Recognition of On-Line Handwritten Mathematical Expressions

Fotini Simistira[1, 2], Vassilis Papavassiliou[1], Vassilis Katsouros[1], George Carayannis[2]
[1]*Institute for Language and Speech Processing, Athena - Research and Innovation Center in Information, Communication and Knowledge Technologies, Athens, Greece*
[2]*School of Electrical and Computer Engineers*
National Technical University of Athens
Athens, Greece
fotini@ mail.ntua.gr
{vpapa, vsk}@ ilsp.athena-innovation.gr
carayan@softlab.ece.ntua.gr

Abstract

We present a system for recognizing online mathematical expressions (ME). Symbol recognition is based on a template elastic matching distance between pen direction features. The structural analysis of the ME is based on extracting the baseline of the ME and then classifying symbols into levels above and below the baseline. The symbols are then sequentially analyzed using six spatial relations and a respective 2d structure is processed to give the resulting MathML representation of the ME. The system was evaluated on the Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) 2011 datasets and demonstrates promising results.

1. Introduction

The problem of mathematical expression (ME) recognition has been a subject of intensive research for over 40 years [1]. Throughout these years a few surveys have been reported [2, 3, 4, 5] with the most recent of Zanibbi et al. in 2011 [4] which also addresses the task of Mathematical Information Retrieval.

ME recognition is still a challenging problem in pattern recognition, not only due to the large variety of mathematical symbols, but also because of the two-dimensional (2d) structure of MEs. In contrast to the way plain text is written, mathematical symbols can be written out of the main baseline. The task of designing a mathematical recognizer becomes difficult as the number of symbols that the system has to process becomes large. The database of such a system varies from symbols including Latin and Greek letters and numerals to more specialized mathematical symbols like summation, integral, gradient etc. In addition, mathematical symbols vary in sizes (e.g. the sum operator is very large) and even the same symbols appear in different sizes (e.g. subscripts). Furthermore, there is a great variance in the writing style of each writer.

Mathematical expression recognition can be divided in two stages: (i) symbol recognition and (ii) structural analysis of the ME. In recent years, the recognition of mathematical symbols has reached an accuracy rate of over 95% [4]. Hence, the research of the ME recognition has been focused on the efficient grouping of strokes that forms the symbols and in the structural analysis of the ME. Research efforts have been made with the use of HMM methods that perform symbol recognition in conjunction with stroke grouping into symbols [6] or by employing features that approximate handwritten strokes via linear combinations of basis vectors and parametric curves [7]. A system that simultaneously segment, recognize and interpret the ME using a contextual language model is described in [8], and a progressive grouping algorithm for symbol recognition is introduced in [9].

In the area of structural analysis of MEs, a number of techniques have been employed based on operator dominance [10], on cutting pixel projection profiles

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.172

[11], on identification of symbols of the dominant baseline [12] and on penalty graph minimization [13].

The organization of the paper is as follows. In Section 2 we present the method of the overall system used for recognizing online handwritten mathematical expressions. In this work we develop a system that simultaneously groups strokes according to the highest recognition rate of the resulted symbol. The method of symbol recognition used in this work is described in detail in [14]. Then, a hierarchical structure, describing the 2d layout of the mathematical symbols in the ME, is build. Section 3 discusses the experimental results and in Section 4 we present our conclusions and discuss future work.

2. Mathematical expression recognition

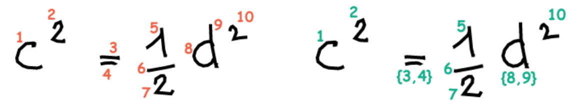
Mathematical expression recognition mainly consists of two stages: i) symbol recognition and ii) structural analysis of the ME. Finally, a mark up language like MathML [15] is used for the interpretation of the content of the ME.

2.1. Symbol recognition

The basic element of an on-line handwritten mathematical expression is called *stroke*. A *stroke* "s" is a finite sequence of coordinates $s = (x_i, y_i)$, between a pen-down and pen-up actions and represents the trace of the digital pen on the writing surface. In Figure 1a, we enumerate the strokes of the mathematical expression following their time order. A mathematical symbol may consist of one or more strokes. In addition, a symbol may result as a combination of intersected strokes (e.g. "+"), or non-intersected strokes (e.g. "="), or a mixture of both intersected and non-intersected strokes (e.g. "$\pm$").

In the proposed system, the first step is to parse the strokes of the ME, and identify those stokes that resemble to horizontal lines. A stroke recognized as horizontal line that is not intersected with any other stroke, either represents the fraction symbol, or the minus symbol, or is part of the equal symbol. For example, in Figure 1a we detect strokes with labels {3, 4, 6} as isolated horizontal lines. In particular, strokes 3 and 4 form an equal symbol, while stroke 6 can be either a fraction line or a minus symbol. We shall address this ambiguity later in this section.

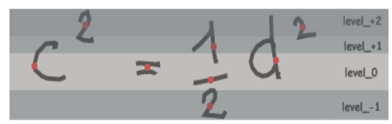
The next step in the proposed pipeline concerns grouping of successive strokes, considering their time order of writing, into symbols. This is done by simultaneously grouping and recognizing the symbol resulting from the candidate group of strokes. The task is performed recursively in a time-window of k successive strokes. The value of k defines the maximum number of successive strokes that belong to the same symbol. Based on the training set of CRHOME 2011 [16], we have set k equal to 4. The most likely symbol is the one with the highest recognition rate among the candidate ones (see Figure 1b and 1c). For instance, by examining strokes 8 and 9 in Figure 1, we conclude that they should be merged to form the symbol "d" (see Figure 1b).



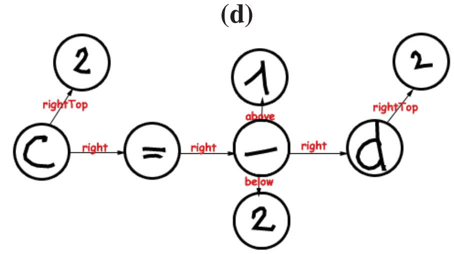
(a) **(b)**

{1} : c_1	{2} : 2_1
{3, 4} : =_1	{5} : 1_1
{6} : -_1	{7} : 2_2
{8, 9} : d_1	{10} : 2_3

(c)



(d)



(e)

Figure 1. Processing steps of ME recognition. (a) ME with strokes labeled in time order, (b) grouping of strokes into symbols, (c) symbol recognition, (d) assignment of symbols to levels, (e) hierarchical structure of ME

For symbol recognition, a template matching distance method is adopted. First, the pen-direction features are quantized using the 8-level Freeman chain coding scheme and the dominant points of the stroke are identified. The distance between two symbols results from the difference of the respective chain codes of the variable speed normalization of dominant points weighted by the respective length proportions of the strokes. The method is explained in detail in [14] and showed a recognition rate of 92 % for the top 1 choice. The output of the symbol recognizer is illustrated in Figure 1c. In this notation, the time-order labels of the strokes that constitute a symbol are included in curly braces (i.e. {3, 4}), they are separated from the recognized symbol by the delimiter ":" and for the recognized symbol we use the symbol itself followed by an underscore and an order number of the symbol in the ME. For example, in Figure 1c, 2_1 denote the first occurrence of numeral 2, while 2_2 and 2_3 denote the second and third occurrences of numeral 2.

2.2. Structural analysis

At this stage all symbols have been recognized and the task now is to identify the relationships among the recognized symbols in order to build a hierarchical structure of the symbols that represents the ME.

The identification of symbol relations is based on layout analysis of the ME. A straightforward solution to this issue is the introduction of constrains that examine the relative spatial relations of the symbols. To this end, we exploit symbol's topological properties such as the *centroid* and the *bounding box* in order to infer the spatial relations among the mathematical symbols, to identify the baseline of the ME and classify the mathematical symbols into levels with respect to the baseline. We connect the symbols of the various levels by defining their spatial relations and, finally, construct the MathML expression of the ME.

Let us first define the bounding box and the *centroid* of a symbol as follows:

Bounding Box. The *bounding box* of a symbol A is the rectangular area that firmly encloses the symbol and can be defined by the four coordinates x_1^A = min(x_i^A), x_2^A = max(x_i^A), y_1^A = min(y_i^A) and y_2^A = min(y_i^A), as shown in the following figure :

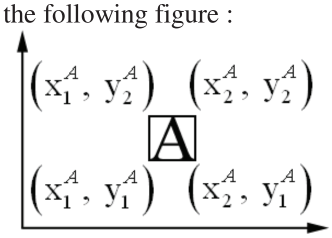


Figure 2. Bounding box of a symbol

Centroid. A very common technique to test whether a symbol lies within a region or not, is to examine the coordinates of its *centroid* [1]. In processing on-line handwritten symbols, the calculation of a symbol's *centroid* cannot be based on the mean value of pixels' coordinates as in the case of off-line symbols. Following the approach suggested in [3], we define the x-coordinate of the *centroid* of a symbol A as $x_c^A = \frac{w^A}{2}$, where $w^A = x_2^A - x_1^A$, and the y-coordinate as $y_c^A = y_1^A + ah^A$, where $h^A = y_2^A - y_1^A$ and $0 \le a \le 1$. The value of *a* depends on the category[1] of the symbol i.e., *ascenders* ("b", "d", etc.), *descenders* ("p", "q", etc.) and *centered* ("+", "=", etc.) as in [3]. For each of the aforementioned categories, we set *a* equal to 0.2, 0.8 and 0.5, respectively.

Now, given a symbol we may define spatial relations with respect to this symbol by examining the other symbols of the ME [3]. In this work, we define the following spatial relations with regard to a given symbol: (i) *rightTop*, (ii) *right*, (iii) *rightBottom*, (iv) *below*, (v) *above* and (vi) *inside*. It is obvious that the spatial relations are dependant on the symbol under consideration. For example the root sign $\sqrt{\ }$ can have an *inside* relation, whereas for the symbol π an *inside* relation cannot be defined. In order to identify the spatial relationship of a symbol A with respect to a symbol B, we introduce the following boolean functions.

rightTop: Symbol A is in the *rightTop* region of symbol B, (e.g. A is superscript of B) if:

$$\tan\frac{\pi}{10} \le \tan(A,B) < \tan(\frac{3\pi}{8})$$

$$y_c^A > y_c^B$$

Where (A,B) is the angle of the line that passes through the *centroids* of symbols A and B with respect to the horizontal line.

right: Symbol A is in the *right* region of symbol B, in other words A is next to B if:

$$\left| y_c^A - y_c^B \right| < a_1 \cdot \max(h^A, h^B)$$

$$x_c^A > x_c^B$$

All the symbols that belong to the same level satisfy this function.

rightBottom: Symbol A is in the *rightBottom* region of the symbol B, (e.g. A is subscript of B) if:

$$\tan\frac{14\pi}{8} \le \tan(A,B) < \tan(\frac{16\pi}{8})$$

$$y_c^A < y_c^B$$

below: Symbol A is in the *below* region of the symbol B (e.g. the denominator of a fraction), if:

$$\left| x_1^A - x_1^B \right| \le a_2 \cdot \min(w^A, w^B)$$

$$\left| x_2^A - x_2^B \right| \le a_2 \cdot \min(w^A, w^B)$$

$$y_2^A \le y_1^B$$

[1] Based on works in the field of typography, each symbol has been pre-assigned to a specific category.

above: Symbol A is *above* symbol B (e.g. the numerator of a fraction), if:

$$\left| x_1^A - x_1^B \right| \leq a_3 \cdot \min(w^A, w^B)$$

$$\left| x_2^A - x_2^B \right| \leq a_3 \cdot \min(w^A, w^B)$$

$$y_1^A \geq y_2^B$$

inside: Symbol A is *inside* symbol B (e.g. a number or variable inside of a root) if

$$x_1^B < x_c^A < x_2^B$$

$$y_1^B < y_c^A < y_2^B$$

Firstly, we detect the main baseline (*level_0*) of the expression. In order to achieve this, we assume that the starting symbol of the expression is the first symbol written. For example in the ME shown in Figure 1, the starting symbol is "c". Given the start symbol, the spatial relation *right* is checked between the *centroid* of the start symbol and the *centroids* of the other symbols of the mathematical expression. When we find a new symbol that belongs to the baseline, we update the list of symbols of *level_0* and we check the spatial relation *right* of all other symbols of the ME with respect to the elements of this list. In this way we identify the symbols that define the main baseline (*level _0*) of the ME. We then follow the same procedure iteratively starting from the time-ordered symbol that is not previously assigned to the baseline. In this fashion we define the various levels of the ME, which we order them with reference to the main baseline. We remunerate the levels using the plus sign for the levels above the main baseline (*level_0*) and we use the minus sign for the levels below the main baseline (*level_0*) (see Figure 1d).

Let us introduce the concept of *connectors* between the symbols of a ME. Given a symbol that belongs to *level_i* a connector is defined with respect to the nearest/closest symbol belonging to *level_J*, where $i \neq J$, and J>0 if i>0 or J<0 if i<0. Depending on the spatial relationship between the two symbols we also define the type of the connector as defined above, i.e. {*rightTop*, *right*, *rightBottom, above, below, inside*}. Note that, the symbols of the main baseline (*level_0*) of the ME, starting from the start symbol, are connected with connector type *right* as shown in Figure 1e. We proceed with defining the *connectors* and their type for all symbols in each level. In this way all possible relations between symbols are defined, and all the symbols are assigned to a sub-expression of a defined level in the ME. Note that the proposed method deals with only up to one level of superscripting or subscripting.

For example, let us consider the ME shown in Figure 1. As described above, first we identify *level_0* and then all the other levels. To do so we split into levels the symbols of the expression (see Figure 1d). For the ME of Figure 1c, the output of the symbol recognizer is {c_1, 2_1, =_1, 1_1, -_1, 2_2, d_1, 2_3}. Since the start symbol is the letter c_1, we search for all the symbols in the expression that have *right* spatial relationship with respect to c_1. The main baseline is denoted by *level_0* and the following symbols are assigned to this level, i.e. *level_0*={c_1, =_1, -_1, d_1}. Then examining the remaining non-assigned symbols iteratively by starting from the top in the list of the time-order we define the other levels as follows: *level_+1*={1_1}, *level_+2*={2_1, 2_3} and level_-1={2_2}. The next step is to search for connectors in the defined levels (level_+1, level_+2, level_-1) so as to define possible sub-expressions based on symbols from the baseline (see Figure 1e). Hence in the example of Figure 1d, we find *level_+1* symbol {*1_1*} is connected to *level_0* symbol {-_1} with the *above* relationship, while *level_-1* symbol {*2_2*} is connected to the same symbol with relationship *below, level_+2* symbol {2_1} is connected with relation type *rightTop* with symbol {c_1} of *level_0,* while *level_+2* symbol {2_3} is connected with relation type *rightTop* with the symbol {d_1}. The resulting scheme of the procedure is depicted in Figure 1e.

```
<annotationXML type="truth" encoding="Content-MathML">
    <math xmlns='http://www.w3.org/1998/Math/MathML'>
        <mrow>
            <msup>
                <mi xml:id="c_1">c</mi>
                <mn xml:id="2_1">2</mn>
            </msup>
            <mrow>
                <mo xml:id="=_1">=</mo>
                <mrow>
                    <mfrac xml:id="_1">
                        <mn xml:id="1_1">1</mn>
                        <mn xml:id="2_2">2</mn>
                    </mfrac>
                    <msup>
                        <mi xml:id="d_1">d</mi>
                        <mn xml:id="2_3">2</mn>
                    </msup>
                </mrow>
            </mrow>
        </mrow>
    </math>
</annotationXML>
```

Figure 3. MathML structure of the example of Figure 1

We have to point out that the symbol of fraction/minus, such as the {_=1} symbol in the example ME of Figure 1, is clarified at the structural analysis stage and not at the symbol recognition stage.

Since the output during the recognition process can be either the fraction sign or the minus sign, we check to see whether the symbol has relations of type *above* and *below*. If it does, then the line corresponds to a fraction symbol, otherwise it is recognized as the minus symbol.

Finally, after all levels and relations between the symbols of the ME are clarified, a parser is employed that takes as input the structure of the example of Figure 1e and produces the corresponding *mathML* structure of the expression (see Figure 3).

3. Experimental Results

In order to evaluate the overall system, we used the dataset of the Competition on Recognition of Online Handwritten Mathematical Expressions Contest CROHME 2011[16]. A training dataset consisting of 921 mathematical expressions and associated ground truth, were given to participants by the organizers. The test datasets of the contest was partitioned into two parts. Part-I contains 181 expressions of 36 distinct symbols whereas Part-II includes 348 expressions and 57 distinct symbols. In particular, the mathematical symbols of Part-II, which includes symbols of Part I,. are: 10 digits (0-9), 16 Latin letters (a, b, c, d, e, i, k, n, x, y, z, A, B, C, F, j), 6 Greek letters (alpha, beta, gamma, phi, theta, pi), 4 function words (*sin*, *cos*, *tan*, *log*), 2 structure symbols (root and fraction), 5 operator symbols (+, -, div, times, ±), 5 relational operator symbols (=, ≠, ≤, <, ≥), 2 parenthesis symbols ('(' and ')'), 2 elastic operators (∑, ∫), 3 structural operators (lim, → and !) and 2 special symbols (∞ and dots).

The ink trace corresponding to each expression is stored in an *InkML* file that contains mainly three groups of information: (i) the ink: a sequence of individual strokes each one represented by their trace co-ordinates; (ii) the symbol level ground truth: the segmentation and label information of each symbol of the expression; and (iii) the expression level ground truth: the *MathML* structure of the expression.

The performance evaluation was based on four aspects: (i) *STrec*: stroke-level classification rate, (ii) *SYMseg*: symbol segmentation rate, (iii) *SYMrec*: symbol recognition rate (considering only correctly segmented ones) and (iv) *EXPrec*: expression-level recognition rate. The final rating of the systems was based on their correct expression recognition accuracies. Note that at the expression-level a ME can be considered either correct (score equal to one), if the respective *MathML* structure is the same as the ground truth, or not (score equal to zero). This means that if there is at least one mistake at the lower levels, then this mistake is propagated to the expression-level and the score for the ME will be zero. In other words, there is no score for partly correct MEs.

We present the evaluation results of the proposed method in Tables 1 and 2 for the Part I and Part II datasets of CROHME 2011. Four systems, in total, participated in the competition (see [16] for description of the systems). It must be noted that the proposed method is identical with the submitted method IV and the difference in the reported scores is due to bugs fixing in the implementation of the algorithms.

The symbol recognition rate for Part-I dataset is 81.76% and for Part-II dataset is 88.90%, both well below the reported performance in [14] of 92% for the top-1 choice. By analyzing the mistakes we have identified mainly three reasons that explain this discrepancy, i.e., the symbol recognition algorithm (i) does not handle compounds of letters that represent function names, e.g. *log*, (ii) does not take into account the relative location of individual strokes for multiple-stroke symbols, for example, the *not equal* sign consists of three strokes and is confused with a three-stroke capital *F*, and (iii) there are symbol classes that are very much alike, e.g. the capital *C* and the small *c* classes.

Table 1. Part-I test results

Systems	ST_{rec}	SYM_{seg}	SYM_{rec}	EXP_{rec}
I	53.23	59.06	88.78	4.42
II	22.39	27.98	82.11	0.55
III	78.73	88.07	92.22	29.28
IV	37.41	55.15	81.71	0.00
proposed	**75.90**	**86.67**	**81.76**	**9.94**

Table 2. Part-II test results

Systems	ST_{rec}	SYM_{seg}	SYM_{rec}	EXP_{rec}
I	51.58	56.50	91.29	2.59
II	22.11	28.25	83.76	0.29
III	78.38	87.82	92.56	19.83
IV	52.28	78.77	78.67	0.00
proposed	**63.80**	**79.86**	**88.90**	**5.45**

It is also clear from results regarding the recognition rates at the level of the ME that the structural analysis algorithm is rather simplistic and process properly only simple mathematical expressions. As mentioned above, the proposed method deals with only up to three levels in a ME and this is the reason that the performance for Part-II is worse since it includes more complicated structures of MEs. In addition, in certain cases, e.g. MEs with fractions, the structural analysis fails because of the erroneously choice of the starting symbol and in some other cases there is failure in the detection of the baseline due to the slope in the writing of the ME.

4. Conclusions

We have presented a system for the recognition of handwritten mathematical expressions. The system has been evaluated on the CROHME 2011 dataset and showed promising results. We believe that the performance of the system can be further improved with respect to all measures (*STrec*, *SYMseg*, *SYMrec* and *EXPrec*). Currently the proposed approached is being extended to support more complicated structures of MEs and add more levels in the structural analysis method. Furthermore the detection of the baseline of the ME may be improved by adding a slope correction technique before the structural analysis step. We also intend to employ grammatical rules for the last step of building the mathML structure. We plan to submit an updated version of the algorithm in the forthcoming CROHME 2012.

References

[1] R.H. Anderson. *Syntax-Directed Recognition of Hand-Printed Two-Dimensional Equations*. PhD thesis, Harvard Univ., Jan. 1968.

[2] D. Blostein and A. Grbavec. *Recognition of mathematical notation*. Handbook of Character Recognition and Document Image Analysis, 557–582, 1997.

[3] K. Chan and D. Yeung. *Mathematics expression recognition: A Survey*. Journal on Document Analysis and Recognition, 3(1):3-15, August 2000.

[4] R. Zanibbi and D. Blostein. *Recognition and Retrieval of Mathematical Expressions*. International Journal of Document Analysis and Recognition, 1-27, October 2011.

[5] E. Tapia and R. Rojas. *A survey on recognition of on-line handwritten mathematical notation*. Technical Report B-07-01, Free University of Berlin, 2007.

[6] A. Kosmala and G. Rigoll. *On-line handwritten formula recognition using statistical methods*. In Proc. Int'l Conf. Pattern Recognition, 1306-1308, 1998.

[7] N. Matsakis. *Recognition of handwritten mathematical expressions*. Master's thesis, MIT, Cambridge, MA, 1999.

[8] A.M. Awal, H. Mouchere, and C. Viard-Gaudin. *Towards handwritten mathematical expression recognition*. In Proc. Int'l Conf. Document Analysis and Recognition, 1046–1050, 2009.

[9] S. Smithies, K. Novins, and J. Arvo. *A handwriting-based equation editor*. Graphics Interface, 84–91, June 1999.

[10] S. K. Chang. *A method for the structural analysis of two-dimensional mathematical expressions*. Information Sciences, 2:253-272, 1970.

[11] M. Okamoto and B. Miao. *Recognition of mathematical expressions by using the layout structures of symbols*. Int'l Conf. Document Analysis and Recognition, 1:242-250, 1991.

[12] E. Tapia and R. Rojas. *Recognition of on-line handwritten mathematical expressions using a minimum spanning tree construction and symbol dominance*. Graphics Recognition: Recent Advances and Perspectives, 3088:329–340, 2004.

[13] Y. Eto and M. Suzuki. *Mathematical formula recognition using virtual link network*. Int'l Conf. Document Analysis and Recognition, 430–437, 2001.

[14] F. Simistira, V. Katsouros, and G. Carayannis. *A Template Matching Distance for Recognition of On-Line Mathematical Symbols*. 11th International Conference on Frontiers in Handwriting Recognition (ICFHR), August 2008.

[15] http://www.w3.org/Math/

[16] H. Mouchère, C. Viard-Gaudin, U. Garain, D. H. Kim, and J. H. Kim. *CROHME 2011: Competition on Recognition of Online Handwritten Mathematical Expression.* 11th International Conference on Document Analysis and Recognition (ICDAR), September 2011.