Mathematical Formulae Recognition

Daniel Průša and Václav Hlaváč

Czech Technical University, Faculty of Electrical Engineering
Department for Cybernetics, Center for Machine Perception
121 35 Prague 2, Karlovo náměstí 13
Fax: +420 224 357 385, Phone: +420 224 357 465
E-mail: prusa@cmp.felk.cvut.cz, hlavac@cmp.felk.cvut.cz
URL: http://cmp.felk.cvut.cz

Abstract. We present a summary of our work in progress related to mathematical formulae recognition. Our approach is based on the structural construction paradigm and two-dimensional grammars. It is a general framework and can be successfully used in the analysis of images containing objects exhibiting rich structural relations. In contrast to most of all other known approaches, the method does not treat symbols segmentation and structural analysis as two separate processes. This allows the system to solve arising ambiguities more reliably. We have already implemented pilot studies for the off-line as well as on-line mathematical formulae recognition showing that the proposed method can be effectively implemented and practically used.

Key words: mathematical discourse, language processing, formulae recognition, two-dimensional grammars

1 Introduction

We have been studying mathematical formulae recognition for past three years. It is a topic of growing importance. It can be applied to digitize mathematical texts in scanned images (we speak about *off-line recognition*) or to handle mathematical inputs written on tablets (so called *on-line recognition*).

Our motivation comes from the interest in two-dimensional grammars for images. We have chosen mathematical formulae as a test case because of rich structure and because several other people are active here. This gives the possibility to compare the performance with other approaches. The taxonomy of approaches can be found, e.g., in [1]. Most of the cited methods follow a two-phase procedure. First of all, elementary symbols are segmented and recognized, and second, structural analysis is performed. It is hard to recover from errors done during the segmentation.

In our approach, we benefit from the structural construction paradigm. Its general idea has been described in [2]. The novelty is in driving the segmentation by structural analysis. We can distinguish also two phases, however, the difference is that the first phase searches just for elementary symbols candidates and it is up to the parsing algorithm to decide which of the candidates are really a part of the input formula. The parsing process behind is penalty oriented, meaning that each derivation is assigned a penalty value determining a quality of the derivation. The task is to find the derivation with the lowest penalty.

We use two-dimensional grammars to model relations among mathematical symbols. These grammars are a generalization of the two-dimensional context-free grammars presented in [2], the theoretical properties of which were studied in [3].

Our first pilot study aimed the off-line formulae recognition. Related results have been published in [4] and [5]. We have adopted the method for the on-line recognition as well. We are going to present these results in [6].

The idea of structural construction has been also applied by others to the off-line recognition of musical scores [7] and electrical circuits [8].

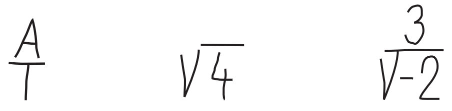
We would like to demonstrate at the workshop current capabilities of our implementation. However, we did not aim at creating a complete system. Both off-line and on-line approach will be demonstrated. We would like to meet people who are interested in mathematical formulae recognition.

2 Method Overview

We consider the following types of inputs:

- *Off-line recognition:* raster images
- *On-line recognition:* sequences of strokes

One of our achievements is suitability of the used method to handle segmentation ambiguities. Figure 1 shows few examples of them.



$\frac{\mathrm{A}}{\mathrm{I}}, \overset{\mathrm{A}}{\mathrm{T}}$ $\qquad$ $\sqrt{4}, \overline{\mathrm{V}4}$ $\qquad$ $\frac{3}{\mathrm{V}-2}, \frac{3}{\sqrt{-2}}$

Fig. 1. Segmentation ambiguities. The pair under each hand-written formula describes an expected interpretation followed by an incorrect interpretation.

As we have already mentioned, our recognition system comprises of two phases, called *terminals detection* and *structural analysis*. They are outlined in the following subsections.

2.1 Terminals Detection

The terminals detection phase is characterized by the following steps:

- Using a suitable strategy, search for rectangular areas (off-line recognition), resp. groups of strokes (on-line recognition), that can contain an elementary symbol. Let $\mathcal{C}_1$ be the set of all found areas, resp. groups.
- Initialize $\mathcal{C}_2 := \emptyset$. Evaluate elements in $\mathcal{C}_1$ by OCR tool. For each element, add matches returned by OCR to $\mathcal{C}_2$.

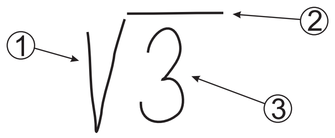
Note that OCR tool can return several matches for one element in $\mathcal{C}_1$, each of the matches assigned by a penalty (a number in $\mathbb{R}^+$) and recognized symbol id. We use implementation based on a simple feature vector extraction and the k-nearest neighbor classifier for off-line recognition, moreover, we have adopted a freely available implementation based on elastic matching technique [9] for on-line recognition.

Let us summarize strategies used to compute $\mathcal{C}_1$.

- (off-line rec.) Move scanning windows of predefined sizes trough the input image. If the sum of pixels of a visited area is greater than some threshold, put it in $\mathcal{C}_1$. Advanced variant: find and add also connectivity components (in this case we consider more general, non-rectangular areas).
- (on-line rec.) Build a graph **G** where nodes correspond to strokes and there is an edge between two nodes iff the corresponding strokes are (informally) close enough. Put in $\mathcal{C}_1$ all connected subgraphs of **G** formed from up to 4 nodes.

The goal of the strategies is to have an acceptable number of elements in $\mathcal{C}_1$ to achieve a good time performance.

Figure 2 shows an example of a terminals detection result for the on-line recognition case.



symbol	variable V	number 3	fraction line	minus sign	square root
strokes	1	3	2	2	1, 2

Fig. 2. An input sequence of three strokes. Terminal symbols found by the first phase are listed in the table.

2.2 Structural Analysis

The structural analysis is similar for both variants (off-line as well as on-line). Here is a pattern the parsing algorithm follows:

- Elements of $\mathcal{C}_2$ computed by the terminals detection are taken as the input.
- New rectangular areas, resp. groups of strokes are incrementally derived by grammar productions from already existing elements. Each such a newly derived element is labeled by a grammar non-terminal and also a penalty of the derivation is computed.
- Let $\mathcal{A}$ be set of derived elements labeled by the grammar initial non-terminal. The penalty of each $R \in \mathcal{A}$ is increased by a special penalization computed for the areas, resp. strokes of the input raster image, resp. sequence of strokes that are not included in R. After that, the result is the derivation with the lowest penalty.

Productions of the grammar express spatial relationships among mathematical symbols. Each production is assigned by a spatial constraint determining which two rectangular areas, resp. groups of strokes, the production can be applied to in order to derive their union.

A production example is shown in Figure 3.

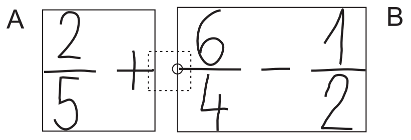


Fig. 3. Applied production example: non-terminal BinaryOperation is derived from ExpressionFollowedByBinaryOperator (A) and Expression (B) non-terminals. The spatial constraint is represented by the dashed rectangle which of size and position are given relative to the bounding box of A. A distinguished point of the bounding box of B is required to be located in the constraining rectangle.

3 Experimental Results

We have implemented the recognition method in Java. The designed two-dimensional grammar supports recognition of usual mathematical symbols and constructs as numbers, variables, brackets, subscripts, superscripts, basic unary and binary operators, power to operations, fractions, sums, integrals and square roots.

We have created a test set consisting of about 400 formulae in both cases. In the off-line case, we considered printed as well as (nicely) hand-written formulae.

The correct rate was influenced by errors done by OCR tools. This is the reason we distinguish two correct rates — the first one computed normally and the second one computed after excluding formulae that have not been recognized primarily due to OCR errors. We have achieved values 88% (correct rate one), resp. 97% (correct rate two) for the on-line recognition and 85%, resp. 94% for the off-line recognition.

The recognition process was responsive for on-line formulae (average time 0.082[s] per formula). On the other hand, it took couple of seconds to recognize an off-line formula. The reason is that the substantially larger number of pixels in a raster image comparing to the number of strokes in its on-line representation.

In the case of raster images, we have faced another limits of the chosen approach. In situations when there are touching elementary symbols which of bounding boxes overlap too much, it is not possible to apply scanning windows strategy and detect the symbols. This problem does not occur for on-line inputs since one stroke is usually a part of exactly one symbol.

Acknowledgement

The authors were supported by the Czech Ministry of Education under project 1M0567.

References

1. Chan, K. F., Yeung, D. Y.: Mathematical expression recognition: a survey. International Journal of Document Analysis and Recognition **3** (2000) 3–15.
2. Schlesinger, M., Hlaváč, V.: Ten lectures on statistical and structural pattern recognition. Volume 24 of Computational Imaging and Vision. Kluwer Academic Publishers, Dordrecht, The Netherlands (2002).
3. Průša, D.: Two-dimensional Languages. Ph.D. thesis, Faculty of Mathematics and Physics, Charles University, Prague (2004).
4. Průša, D., Hlaváč, V.: 2D context-free grammars: Mathematical formulae recognition. In: Holub, J., Žďárek, J., eds.: Proceedings of the Prague Stringology Conference ’06, Prague, Czech Republic, Prague Stringology Club, CTU (2006) 77–89.
5. Průša, D., Hlaváč, V.: Mathematical formulae recognition using 2D grammars. In: Proceedings of the 9th International Conference on Document Analysis and Recognition. Volume II, Curitiba, Brazil (2007) 849–853.
6. Průša, D., Hlaváč, V.: Structural construction for mathematical on-line formulae recognition. In: Proceedings of the 13th Iberoamerican Congress on Pattern Recognition, Havana, Cuba (2008) accepted, to appear.
7. Savchynsky, B., Schlesinger, M., Anochina, M.: Parsing and recognition of printed notes. In: Proceedings of the conference Control Systems and Computers, Kiev, Ukraine (2003) 30–38, in Russian, preprint in English available.
8. Kiyko, V.: Recognition of objects in images of paper based line drawings. In: Third International Conference on Document Analysis and Recognition, Montreal (1995) 970–973.
9. PVMerlin (2003) `http://www.nm.informatik.uni-muenchen.de/~ensel/privat/pvmerlin`.