A Hybrid Classifier for Handwritten Mathematical Expression Recognition

Ahmad-Montaser Awal, Harold Mouchère, Christian Viard-Gaudin
IRCCyN/IVC – UMR CNRS 6597
Ecole polytechnique de l'université de Nantes
Rue Christian Pauc – BP 50609 – 44306 Nantes CEDEX 3 – France
{ahmad-montaser.awal, harold.mouchere, Christian.Viard-Gaudin }@univ-nantes.fr

Abstract

In this paper we propose a hybrid symbol classifier within a global framework for online handwritten mathematical expression recognition. The proposed architecture aims at handling mathematical expression recognition as a simultaneous optimization of symbol segmentation, symbol recognition, and 2D structure recognition under the restriction of a mathematical expression grammar. To improve the classifier in this architecture, we consider a two level classifier. A symbol classifier cooperates with a second classifier specialized to accept or reject a segmentation hypothesis. The proposed system is trained with a set of synthetic online handwritten mathematical expressions. When tested on a set of real complex expressions, the system achieves promising results at both symbol and expression interpretation levels.

1. Introduction

Nowadays, devices that rely on digital pen are widely used. PDAs, tablet PCs or electronic white boards are such examples. One of the main purposes is to ease information input. Thus, the emergence of such devices makes it indispensable to develop tools and systems capable of converting handwritten texts into digital formats.

Handwritten text recognition systems have achieved recently significant progress, thanks to developments in segmentation, recognition and language models. Those systems are less powerful when the languages to be recognized have a two dimensional (2D) layout. This is the case for mathematical expressions [1], schemas, diagrams, etc. In this case, it yields to solve the same problems of segmentations, recognition and interpretation but in a 2D context.

Mathematics is used in almost all fields of science, such as physics, engineering, medicine, economics, etc. As a result, an input method for mathematical expressions into scientific documents is a requirement.

Many tools are available to achieve this task. However, most of these tools require some expertise to use them efficiently. Latex and MathML, for example, require knowledge of predefined sets of key words to describe special mathematical symbols and functions in addition to spatial layouts. Other tools, such as Math Type, depends on a visual environment to add symbols using the mouse and thus needs lot of time.

We focus our research on the recognition of online handwritten mathematical expressions. Most researches emerging in this area consider subclasses of mathematical expressions, and have achieved some promising results. Most of those research works regard recognition of mathematical expressions as a set of independent subtasks to perform different steps of the recognition process. Though, a main drawback comes from the fact that any error at any step will be automatically inherited to the next step, and that no joint-optimization is done across the different stages.

We proposed in a previous work [20] an architecture that performs simultaneous segmentation, recognition and interpretation of mathematical expressions. Specifically, the classifier used to recognize the basic symbols is based on a global learning method allowing the system to learn symbols in conjunction with the segmentation process directly from expressions instead of using a pre-trained classifier.

Our contribution in this paper is to use a hybrid classifier more specialized for the problem of mathematical expression recognition. This hybrid classifier, which is composed of two modules is also trained globally within the recognition framework and has as objective to reduce the classifier complexity and to improve recognition accuracy.

In section two, we introduce the problem of mathematical expression recognition. Then, we develop the proposed architecture in section three, and we give some preliminary results that are compared with some of other works [19].

2. Mathematical expression recognition

Usually, tools such -as LATEX or MathML- are used to input mathematical expression into digital documents. However, it is still an annoying and long process, especially with long complex expressions. Thus, using a digital pen presents a more natural way to input such expressions into digital documents.

Mathematical Expression (ME) recognition problem can be divided into three sub problems [2]: segmentation, symbol recognition and expression interpretation. Mathematical expressions have some particularities that should be taken in consideration in any ME recognition system. The required number of symbols used in mathematical expressions to cover correctly most of the scientific applications is huge, typically more than 200 symbols- compared to normal text. Though it is essential to dispose of a good classifier adapted to ME recognition problem.

In addition, as shown in Figure 1, symbol role might be ambiguous. The same pattern could be interpreted differently according to the context. The cross pattern on the left of Figure 1 can be used for the 'x' variable or the multiplication operator. It is the same for the horizontal bar on the right, which could represent a minus, a fraction bar, a part of plus minus sign, or an equal [6].

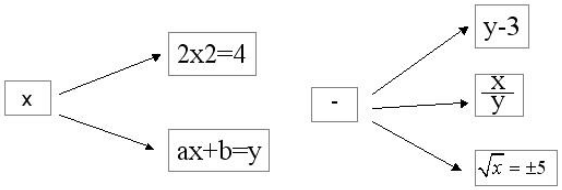

Figure 1 Examples of symbol role ambiguity

Other difficulties arise from the two dimensional symbols layout. Find implicit operations requires resolving spatial relations ambiguity. Figure 2 shows how the transition from the multiplication to superscription is fuzzy in nature.

Figure 2 Example of a fuzzy spatial relation

Relative symbols locations cause also some ambiguities. As shown in Figure 3, in the expression b_c the "c" can be a subscript of b as in $b_c d$, or it can be a variable as in $a^b c$, while b is a superscript of a previous variable [13].

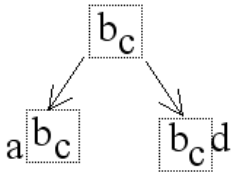

Figure 3 Local ambiguity from [13]

Classic pattern recognition methods can be used in ME recognition systems. Although template matching methods can be slow and time consuming, they have been used by some systems [7, 19] with promising results. Other systems [2, 8] extract structural primitives and compare them with the training data. On the other hand, artificial neural networks (ANN) are known to be better in terms of speed and recognition rate [9, 10]. Statistical models such as hidden markov model [11] can perform a simultaneous segmentation and recognition. Each symbol has its own model and recognition results are obtained based on the likelihood estimations of the different models.

However, correct recognition requires a correct segmentation. Given an online handwritten ME signal, a symbol consists of one or more strokes. A stroke being a trace drawn between a pen down and a pen lift. We consider this stroke as the primitive unit. Thus, segmentation step consists in grouping strokes belonging to the same symbol.

Figure 4 shows possible groupings of a set of strokes. Strokes grouping problem is by itself a complex and challenging problem. This problem is sill more difficult when delayed strokes are present. We will consider those cases, and assume the eventuality of interspersed strokes between symbols.

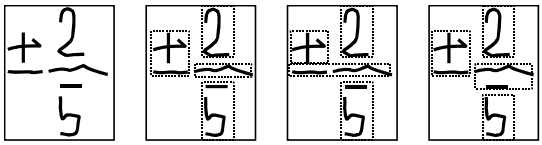

Figure 4 Example of grouping hypothesis

Successive X Y projections [3] of strokes, or strokes bounding boxes [4] can be used in order to build a spatial relation tree between strokes. This tree then helps in grouping strokes of the same symbol together.

Recognition process can also guide the segmentation by grouping strokes that are most likely to represent a symbol [5, 19, 20].

Before applying a syntax analysis, it is required to find the structured description of the expression [12]. This description defines spatial relations among symbols. As mentioned before, those relations are fuzzy in nature and might cause some ambiguities.

Finally, syntax analysis aims at resolving ambiguities and finding the structure of the expression. Two dimensional grammars are usually used in this analysis due to the fact that expressions are considered as 2D languages [14]. Achieving a 2D parsing is a complex process, which requires special techniques and methods to reduce its complexity [15].

In the proposed framework, we try to limit the main drawbacks related to each step of ME recognition. We propose a framework allowing a simultaneous segmentation, recognition, and interpretation.

3. Recognition framework

Comparing to our work presented in [20], the originality of this paper is the use of a hybrid classifier instead of using a single classifier. The proposed classifier relies on the following points. First, a symbol classifier will be trained from an isolated symbol database. Second, another classifier will be trained from scratch in the context of the whole system including the first classifier, segmentation, 2D parsing and grammar rules to give the best possible interpretation. This second classifier will have a different task from the previous one. It has to discriminate between actual symbols and the false segmentation hypotheses -called further the junk class, generated by the hypothesis generator, see Figure 6. Scores from both classifier will be combined together to calculate the recognition cost of each symbol hypothesis. The main advantage of this hybrid structure of classifier is to allow to use classifiers which can hardly been trained with incremental learning. For instance, a support vector machine (SVM) cannot be trained using incremental learning but might be used as a pre-trained symbol classifier. On the other hand, a neural network (NN) when trained with a local stochastic gradient descent method can be used in the context of the whole system.

We compare our results with those in [19]. They propose a layered search framework for ME recognition performing a simultaneous segmentation and recognition, but using a classifier which is trained on isolated symbols.

3.1. Global architecture

An online mathematical expression is input to the system as a set of strokes. So, recognizing an expression consists in finding the best possible grouping of those strokes to represent expression symbols and spatial relations among those symbols to find out the structure of the expression.

Both expression recognition and system training can be described with the same global architecture of expression recognizer system shown in Figure 5.

It can be briefly described by [20]:

- A symbol hypothesis generator (SHG): lists a number of possible combination of strokes. Each group of strokes is called a symbol hypothesis (SH).
- A symbol recognizer (SR): provides, in addition to the label of each hypothesis, a recognition score that will be used to define the recognition cost. It can be either a hybrid classifier, or a single classifier.
- Structural analyzer (SA): provides structural information about each hypothesis so that contextual evaluation can be performed.
- A language model (LM): defines the grammar that produces acceptable mathematical expressions.
- A decision maker (DM): DM organizes all the SHs, and selects the one that minimizes the cost function and respect the language model in order to represent a valid expression.

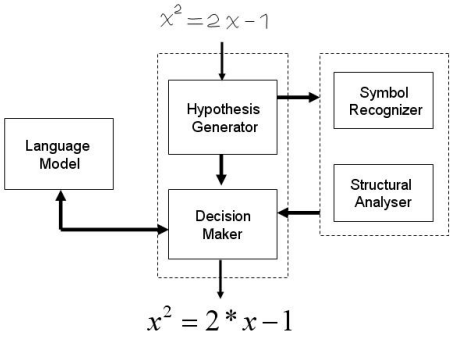

Figure 5 Expression recognizer architecture overview

From a computing perspective, it can be considered as a Dynamic Programming (DP) algorithm, which is well adapted to this kind of decision making problems [16]. However, the key point is that this is not a standard 1D-DP where only consecutive strokes can be considered at a time, but it is an extension to a 2D-DP [20].

Since, not only the actual segmentation will be produced by the SHG but also a lot of spurious hypotheses, the SR should have the ability to deal with such outliers. Hence, the architecture allows learning an additional class we call “junk class”, which presents an incorrect grouping of strokes. This class helps at eliminating non-valid hypothesis.

3.2. Hybrid symbol recognizer

Instead of adding an additional output to the classifier as a junk output as done previously [20]. The classifier is divided into two specialized classifiers, as

shown in Figure 6. First, a target classifier is in charge of recognizing symbols. It can be either trained globally within the whole architecture –provided that it is a NN, or it can also be trained separately from isolated symbols –in case it is a SVM before global learning, and then participates to the training of the junk classifier.

On the other hand, junk classifier is in charge of recognizing only two classes, junk and non-junk. In other words, it accepts or rejects a symbol hypothesis of being a known symbol. Junk Classifier is trained globally, directly from mathematical expressions, within the global structure since the junk class is inherently dependent of the ME segmentation stage.

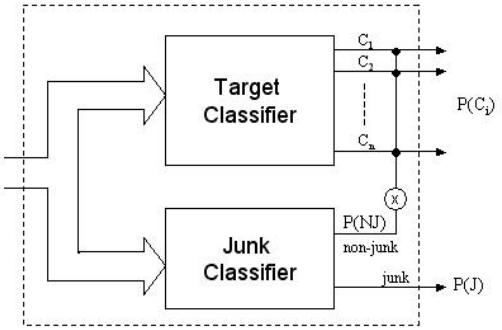

Figure 6 Hybrid classifier overview

We chose a multi-layer perceptron neural network (MLP) as a junk classifier. In this case, a gradient-based backpropagation algorithm is used. It takes into account the ground truth of the given ME (ideal segmentation and corresponding labels of symbols) and the best current interpretation resulting from a specific segmentation, and the corresponding recognized symbols.

3.3. Hybrid recognition cost

The difficulty of using a hybrid classifier lays in finding a good way of combining scores of the two different classifiers. Output scores of those two classifiers might be of different natures and thus can not be combined directly. However, to overcome this problem we apply softmax function on the outputs of both classifiers. Hence, outputs of each classifier are in form of independent probabilities.

The probability of a target output being the target class C_i , knowing that it is not a junk (NJ) is:

$$p(C_i \mid \mathrm{NJ}) \quad ; \sum_{i=1}^{N} p(C_i \mid \mathrm{NJ}) = 1 \tag{1}$$

On the other hand, the output of the junk classifier gives the probability of being an accepted class (NJ) or a junk (J):

$$p(NJ) + p(J) = 1 \tag{2}$$

However, the probability of an output of the hybrid classifier being a class C_i is:

$$\begin{aligned} p(C_i) &= p(C_i, NJ) + p(C_i, J) \\ &= p(C_i \mid \mathrm{NJ}).\mathrm{p(NJ)} + p(C_i \mid \mathrm{J}).\mathrm{P(J)} \end{aligned} \tag{3}$$

But by definition: $p(C_i \mid \mathrm{J}) = 0$, hence:

$$p(C_i) = p(C_i \mid \mathrm{NJ}).\mathrm{p(NJ)} \tag{4}$$

This new probability is comparable with the junk probability allowing then to combine the junk with the target class, where:

$$\sum_{i=1}^{N} p(C_i) + p(J) = 1 \tag{4}$$

The recognition cost is then measured on a negative log scale depending on the recognition score.

In addition, the structural cost represents how good the hypothesis fit to both the layout of the whole expression and the context. Calculating this cost requires a structural analysis of the expression using spatial information such as height of the considered symbol and position with respect to the baseline. A language model is also used to respect mathematical expression producing rules. Thus invalid expressions are not accepted. In the next section, we present some results of the preliminary tests.

4. Expressions database

A database of online handwritten mathematical symbols written by 280 writers was previously collected. This database serves as an isolated database to train target classifiers. Moreover, the same database is used to generate stochastic pseudo-synthetic handwritten expressions database by using "Latex2Ink" [18] –a tool developed in our research group. However, generated expressions are not intended to replace real ones; it aims to provide a large quantity of examples to be able to train and tune the system.

Training complexity is related to expressions complexity and number of classes. However, optimizing global learning parameters requires many experiments. Thus, we use two corpora of mathematical expressions in our experiments. The "Calculette" one, which presents a simple expression database allowing good parameter optimizations, and the "Aster [17]" one, which presents a

larger and more complicated expression database allowing a good test of the system with very realistic expressions.

4.1. Calculette database

Experiments on this corpus are conducted on a simple calculator application. The symbol class number is 15. They are digits [0-9] and simple math operators [+, -, *, ÷] and the equal sign [=]. Table 1 shows the constitution of both train and test databases for isolated symbols and expressions.

Table 1 Calculette database constitution

	# Writers	# Isolated Symbols	# Expressions	# Symbols
Train	180	180 × 15 = 2700	180 × 5 = 900	5448
Test	100	100 × 15 = 1500	100 × 5 = 500	3051

A total number of 1,400 expressions were produced where 900 expressions are used to train the recognizer and 500 expressions used for the test. Produced expressions are referred as Synthetic expressions. Figure 7 displays a sample of such an expression that has been generated randomly from the string "??? + ??? = ???".

Figure 7 Example of Calculette synthetic expression

4.2. Aster database

Aster base [17] consists of 62 different expressions covering many domains with an average length of 13 symbols per ME. The total number of symbols is 839 symbols within 48 distinct classes including digits, Roman letters, Greek letters, binary operators, elastic symbols and functions.

With respect to the current grammar rules that we have implemented, we considered a sub-group of the Aster base consisting of 36 expressions covering all common math disciplines of the base, see Table 2. For this subset, number of classes is restricted to 34 symbols.

Table 2 Corpus extracted from Aster base

Domains	# Expressions
Simple fractions and expressions	8
Examples of Knuth	6
Continuous fractions	1
Algebraic expressions	3
Square roots	2
Trigonometric identities	6
Logarithms	3
Series	3
Integrals	1
Summations	2
Hyperbolic functions	1
Total	36

Table 3 shows the constitution of both train and test databases for Aster isolated symbols and expressions. bases. Figure 8 shows some examples of Aster generated expressions.

Table 3 Aster database constitution

	# Writers	# Isolated Symbols	# Expressions	# Symbols
Train	180	180 × 34 = 6120	180 × 36 = 6480	180 × 412 = 74160
Test	100	100 × 34 =3400	100 × 36 = 3600	100 × 412 = 41200
Test (Real)	10	-	2 × 36 = 72	2 × 412 = 824

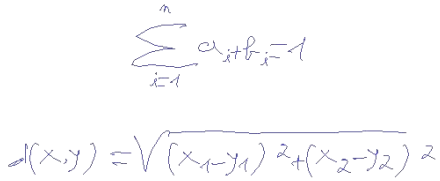

Figure 8 Examples of synthetic expressions

Since the system needs also to be tested with a real set of online handwritten mathematical expressions, each of the 36 expressions has been written by two different writers forming a new database of 72 expressions to test the system. Ten writers have been involved in this dataset, referred as Test (Real) in Table 3.

5. Experiments and Results

The evaluation of the system at the ME level is too global to be significant. It is necessary to give performances at some intermediate levels to have a better insight of the actual behavior of the system. Thus, we chose three measurements similar to those used recently in [6, 19] in order to be able to compare our results, keeping in mind that expressions test bases are different. They are presented in the sub-section 5.2.

Inputs are re-sampled to (50) points along the trajectory of each hypothesis with seven local features

for each, resulting in a 350 dimension input vector. Most meta parameters of all classifiers have been chosen experimentally.

Target classifiers are trained from isolated symbols databases. Three different target classifiers are tested. First, we chose a multi layer perceptron (MLP) architecture with one hidden layer, using 100 neurons. Next, we used a time delayed neural network (TDNN) [22] for its interesting properties of being insensitive to position shifts. The last classifier used is a support vector machine [21] (SVM) using a Gaussian kernel. On the other hand, the architecture of the junk classifier is always a multi layer perceptron but with only 50 neurons in the hidden layer. Any other classifier could be used for the target classifier since it is trained independently of the system. This choice reduces as well the time needed for global learning.

5.1. Isolated symbol recognizer performance

For both corpora, the target classifier is trained with the isolated symbols train dataset. Then it is tested with the test data set, see Table 4 .

Table 4 Calculette and Aster isolated symbol recognition performance

Recognizer	Calculette Test% (15 classes)	Aster Test% (34 classes)
MLP	96.6	95.4
TDNN	96.8	95.7
SVM	96.7	96.2

In order to train junk classifiers globally, those trained target classifiers are integrated within the global architecture. Though, their objective is limited to drive the global learning process of the Junk/No-Junk MLP classifier. We notice that the MLP and the TDNN have similar performance on isolated symbols. Furthermore, they tend to have same behavior on expression databases. Though, in the next section we present only results using a TDNN and a SVM as target classifiers.

5.2. Expression recognizer performance

The recognizer performance was evaluated by these three different measurements:

- Segmentation rate (SegRate),
- Recognition rate (RecRate),
- Expression recognition rate (ExpRate),

representing respectively the percentage of correctly segmented symbols, correctly segmented and recognized symbols and expressions totally correctly interpreted.

Table 5 show the results obtained when testing with the Calculette database. While results of testing with the Aster database are shown in Table 6.

Table 5 Calculette Expression recognition rates on test dataset

		SecRate%	RecoRate%	ExpRate%
TDNN	a) Isolated	96.6	94.3	81.1
	b) Globally	99.2	96.2	84.2
	c) Hybrid	99.2	97	87.3
SVM	a) Isolated	88.6	87.5	82.2
	c) Hybrid	99.6	97.6	89.2

Results of using different hybrid classifiers (c) are compared with either a classifier pre-trained on isolated data (a) with no explicit junk capability, or a single classifier trained globally, which in that case contains explicitly an additional junk class (b) [20]. Table 5 shows a very important improvement when using hybrid classifier, especially on expression recognition rates. This assures that a hybrid classifier is more adapted globally to the problem of Calculette expression recognition. However, this observation cannot be generalized on the Aster database.

Table 6 Aster Expression recognition rates on test dataset

Synthetic expressions		SecRate%	RecoRate%	ExpRate%
TDNN	a) Isolated	64.2	62.9	25.6
	b) Globally	86.9	84.6	59.6
	c) Hybrid	84.1	81.8	40.6
SVM	a) Isolated	74.6	73.5	45.5
	c) Hybrid	73.6	72.9	50.6
Real expressions		SecRate%	RecoRate%	ExpRate%
Average rates [19]		94.8	84.8	29.2
TDNN	a) Isolated	50	46.6	11.4
	b) Globally	82.6	75.5	31.4
	c) Hybrid	76	71.1	25.7
SVM	a) Isolated	67	63.7	28.6
	c) Hybrid	67.6	64	27.1

Table 6 shows that TDNN and SVM behave differently when moving from isolated to hybrid training (a to c). Clearly, the TDNN classifier takes advantage of the hybrid architecture. The expression recognition rates increase from 25.6% to 40.6% on the synthetic expressions and from 11.4% to 25.7% on the real expressions. This is not always the case when using the SVM classifier; if a noticeable improvement is obtained with the synthetic expressions (45.5% to 50.6%), a slight loss is observed with real expressions (28.6% to 27.1%).

We hypothesize that the way the softmax function is applied to convert the outputs of the SVM to probabilities has to be investigated more in details. Another explanation is the better capability of the SVM

compared to the TDNN with respect to outlier situations. It is well known that due to its discriminant nature [23] a neural network has poor modeling capability, whereas using Gaussian kernels with the SVM allows to model more explicitly the feature space, and hence such classifier has some intrinsic reject properties.

It is also worth to note that the best results are however obtained when the system is trained globally ((b) lines).While it is not reported in Table 6, it is also the case when a MLP is used instead of a TDNN within a global learning scheme. An expression recognition rate of 35.71% on real expressions is obtained, which is even better than the rate of 31.4% obtained with the TDNN.

6. Conclusion and perspective

In this paper, we proposed a new hybrid classifier for mathematical expression recognition. This classifier, which is integrated within a global architecture, aims at handling mathematical expression recognition as a simultaneous optimization of segmentation, recognition, and interpretation. Being trained with synthetic expressions, the system is not only tested with synthetic expressions but also on a real complex expressions database. On the latter, we obtain encouraging results in the domain of mathematical expression recognition.

However, experiments on complicated expressions shows that a hybrid classifier with a pre-trained classifier on isolated symbols might not be the best solution. Using a multi output junk classifier, and multi target classifiers might presenting a good track for future experiments and tests. How to incorporate a SVM classifier in such a global learning scheme is still an open problem.

7. References

[1] Blostein D., A.G., Recognition of mathematical notation, in *Handbook on Optical Character Recognition and Document Image Analysis*, Q.s.U., 1997, World Scientific Publishing Company: Kingston, Ontario, Canada. p. 557-582.

[2] Chan K-, D.-Y.Y., An efficient syntactic approach to structural analysis of on-line handwritten mathematical expressions, *Pattern Recognition*, 2000. 33: p. 375 - 384.

[3] C. Faure, Z.X.W, Automatic perception of the structure of handwritten mathematical expressions, in *Computer Processing of Handwriting*, 1990, World scientific, Singapore.

[4] J. Ha, R.M.H., and I. T. Phillips, Understanding mathematical expressions from document images, third *International Conference on Document Analysis and Recognition*, 1995, p. 956-959.

[5] Steve Smithies, K.N., James Arvo. A Handwriting-Based Equation Editor, the Graphics Interface, 1999, Kingston, Ontario, Canada.

[6] Yamamoto R., S.S., Nishimoto T., Sagayama S., On-Line Recognition of Handwritten Mathematical Expressions Based on Stroke-Based Stochastic Context-Free Grammar, *tenth International Workshop on Frontiers in Handwriting Recognition* 2006, La Baule, France. p. 249 - 254

[7] Nakayama, Y. A prototype pen-input mathematical formula editor, in *EDMEDIA*, 1993.

[8] A. Belaid, J.-P.H, A syntactic approach for handwritten mathematical formula recognition, in *Transactions on Pattern Analysis and Machine Intelligence*, 1984.

[9] Marzinkewitsch R., Operating computer algebra systems by handprinted document, *International Symposium on Symbolic and Algebraic Computation*, 1991.

[10] Yannis A. Dimitriadis, J.L.C., Towards an ART based mathematical editor that uses online handwritten symbol recognition, *Pattern Recognition*, 1995. 28(6): p. 807 822.

[11] Lehmberg S., H.-J.W., Lang M., A Soft-decision approach for symbol segmentaiton within handwritten mathematical expressions, *Int. Conference on Acoustics, Speech, and Signal Processing*, 1996, Atlanta, USA.

[12] Fukuda R., S.I., Tamari F., Xie M., and Suzuki M., A technique of mathematical expression structure analysis for the handwriting input system, *fifth International Conference on Document Analysis and Recognition* , 1999. p. 131 - 134.

[13] Martin, W. Computer input/output of mathematical expressions, in *Symbolic and Algebraic Manipulations*, 1971, New York.

[14] Prusa D., V.H. 2D Context-Free Grammars: Mathematical Formulae Recognition. in *The Prague Stringology Conference*, 2006, Prague.

[15] Liang P., M.N., Shilman M., and Paul Viola. Efficient Geometric Algorithms for Parsing in Two Dimensions, *eightth International Conference on Document Analysis and Recognition*, 2005, Seoul, South Korea. p. 1172 - 1177.

[16] Held M., R.M.K., The construction of discrete dynamic programming algorithms, *IBM Syst. J.* 4 (2), 1965: p. 136-147.

[17] Raman, T.V., Audio system for technical readings, 1994, Cornell University.

[18] Awal A.M., Cousseau R., Viard-Gaudin C. Convertisseur d'équations LATEX2Ink,. *in Colloque International Francophone sur l'Ecrit et le Document* 2008, Rouen, France, . p. 193 – 194.

[19] Rhee T-K., K.-E.K., Kim J., Robust Recognition of Handwritten Mathematical Expressions Using Search-based Structure Analysis, 11th *International Conference on Frontiers in Handwriting Recognition*, 2008, Montreal. p. 19 - 24.

[20] Awal A.M, Mouchère H., Viard-Gaudin C., Twoards handwritten mathematical expression recognition, *tenth International Conference on Document Analysis and Recognition*, 2009, Barcelona, Spain. To be published

[21] Cortes C. and Vapnik V., Support-vector networks. *Machine Learning* , 20(3) :273-297, 1995.

[22] Schenkel M., Guyon I. and Henderson D., Online cursive script recognition using time delay neural networks and hidden Markov models, Mach. Vis. Appl., *Special Issue on Cursive Script Recognition* 8 (1995) 215–223.

[23] C.M. Bishop, "Neural Networks for Pattern Recognition", *Oxford University Press*. ISBN 0-19-853849-9, pages, 1995