Symbols Recognition System for Graphic Documents Combining Global Structural Approaches and Using a XML Representation of Data

Mathieu Delalandre[1], Eric Trupin[1], Jean-Marc Ogier[2]

[1] Laboratory PSI, University of Rouen, 76 821 Mont Saint Aignan, France
{mathieu.delalandre, eric.trupin}@univ-rouen.fr
[2] Laboratory L3I, University of La Rochelle, 17042 La Rochelle, France
jmogier@univ-lr.fr

Abstract. In this paper we present a symbols recognition system for graphic documents, based on a combination of global structural approaches. Our system allows to extract components and their loops, with their inclusion and neighboring links. So, it is possible to construct different graph types according to the considered recognition problem. Our system uses an XML representation of data, which allows an easy manipulation of these ones. We present some results on a symbols set of GREC2003's recognition contest.

1 Introduction

This paper deals with the structural recognition of symbols applied to graphic documents. We present here a system based on a combination of global structural approaches. In the paper's follow-up, we present in section (2) the general problem of structural recognition of symbols on graphic documents. The two following sections are dedicated to two main system's parts: global structural analysis and images preprocessing in section (3), and structural classification in section (4). In section (5), we present the use of XML in the system. In section (6), we present some results and analysis on a symbols set of GREC2003's recognition contest. Finally, in section (7), we conclude and give some perspectives.

2 Structural Recognition of Symbols

Classically, a documents recognition system is decomposed into two main steps [7]: an images processing step and a recognition step. Two main approaches exist: statistical & connexionnist, and structural & syntactic[1]. This paper deals especially with the structural approach. This one uses graph representations of documents' objects. Into graphic documents, many objects could be described by graphs, especially

[1] In the paper's follow-up, we talk about "structural" for "structural & syntactic".

symbols [10]. Thus, in a structural recognition system of symbols, the images processing step extracts graphs from images corresponding to symbols, and the structural recognition step exploits these graphs.

The structural recognition step is generally used for symbols recognition, but also for other purposes like: learning, indexing, data structuring, and so on. It uses two main approaches: graph-matching [6], and graph-grammar [2]. The first one matches extracted graphs with model graphs. The second one applies different rules to transform extracted graphs into model graphs. A graph problem depends on two criteria: graph/subgraph, and exact-inexact. If extracted graphs correspond exactly to model graphs, the problem is known as exact. Unfortunately, in image applications, extracted graphs are noisy. So, it is an inexact graph/subgraph problem.

The images processing step extracts (or constructs) graphs from images corresponding to symbols. Into a related paper [5], we talk about "structural analysis" for this images processing step and propose a classification into local and global approaches. The boundary between the two approaches is the connected component. Fig. 1 gives an example. The local approach (b) decomposes the connected component (a) into arc, junction, and vector objects. The global approach (c) groups together three connected components (a) according to distance constraints.

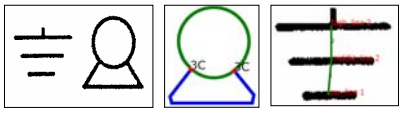

Fig. 1. (a) symbols (b) local structural analysis (c) global structural analysis

This paper especially deals with the global structural analysis. This one extracts spatial links between connected components, in order to construct graph representations of symbols. Therefore, this analysis processes only the segmented symbols (not connected). Moreover, the connected components are graphical primitives of "low semantic". This allows to deal only with the few classes recognition problems. Three main approaches exist. The first one simply extracts connected components' gravity centers in order to use distance constraints between them with graphs based algorithms [11]. The next two extract more complete links from images between connected components: the inclusion [8] and the neighboring [3] links. The first one generally is based on blob coloring methods [8], and the second one on the generalized Voronoi diagrams methods [3]. The Fig. 2 gives an example of symbol with its inclusion graph (a), and the neighboring graph of its central part (b). Also, "hybrid" approaches exist, like global/local [8] or statistical/structural [4] approaches.

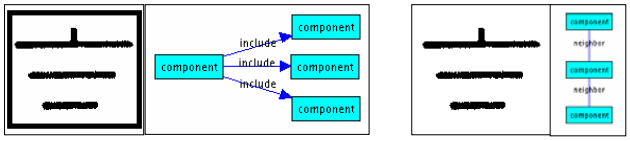

Fig. 2. (a) inclusion links (b) neighboring links

In this paper we present a system for symbols recognition based on global structural analysis. This one exploits an approaches combination of global structural analysis. It allows to extract from images the connected components and their associated loops (see subsection 3.1), with their inclusion and neighboring links. So it is possible to construct different graph types according to the considered recognition problem. These graphs are next processed by a graph-matching algorithm. We present this system in the three next sections.

3 Global Structural Analysis and Images Pre-Processing

We present in this section the first system's part for images processing. Methods presented in this section are grouped in a C++ library for connected components analysis, the CCLib[2]. We present first in subsection (3.1) the general method for connected components labelling and loops extraction. In subsection (3.2), we present the methods to construct connected components graphs. Finally in subsection (3.3), we present a pre-processing method based on connected components' surfaces analysis.

3.1 Connected Components Labelling and Loops Extraction

The central part of system's images processings is a connected components labelling method. Our labelling method is based on regions aggregation with an 8 connectivity analysis of black pixels. The Fig. 3 gives an example with the original image (a), and four successive aggregation steps (b). During these steps, the aggregated pixels are erased (steps 1-3: 3 pixels, step 4: 1 pixel).

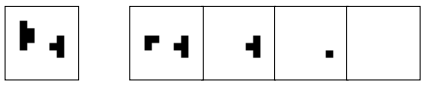

Fig. 3. (a) image (b) four successive aggregation steps

This method is a few more complex than classical blob coloring methods [1], but labels and extracts in one step the connected components and their characteristics: gravity centers, areas "dx; dy", contour pixels, and so on. The Fig. 4 gives an example image (a) with its labelling result into SVG format (b) (see section 5).

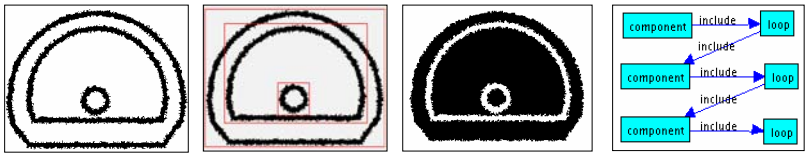

Fig. 4. (a) symbol (b) components labelling (c) loops extraction (d) inclusion graph

[2] Connected Component Library: http://site.voila.fr/mdhws/

We use too this labelling method to extract loops images[3] (Fig. 4 (c)). The image's borders are first initialized into the image's background colour. Next, the image is inverted. This inverted image is next labelled. With the borders initialization, the first labelled connected component corresponds to image's background. This one is erased, and the others are used to create the loops image (Fig. 4 (c)).

3.2 Graphs Construction

Two main links exist between connected components (see section 2), the inclusion and the neighboring links. We present first two methods to extract these links. Next, we present a method to construct hybrid graphs combining these two links.

Firstly, we extract the inclusion links between connected components. The Fig. 4 (d) gives an example of inclusion graph into XGMML format (see section 5) of Fig. 4 (a). These inclusion links are directed, and give a tree description of symbol's components and their associated loops. Our method first extracts symbol's loops image (Fig. 4 (c)). Next, it labels components image and its associated loops image (Fig. 4 (a) and (c)). Finally in these two labelled images, the method analyses the contours' labels of components and loops in order to extract their inclusion links (Fig. 4 (d)).

Secondly, we extract the neighboring links between connected components. These links are undirected, and give a graph description of symbol's components or loops. Our method is based on the extension of connected components' contours. During this extension, the extended contours are labelled. The process is stopped when the extended contours meet other extended contours, connected components, or null zones "out of image". The Fig. 5 (b left) gives an example of four successive steps of contours extension of Fig. 5 (a). Then, the method extracts the boundary points from the obtained labelled map of extended contours. The Fig. 5 (b right) gives an example of image representation of extracted boundary points of Fig. 5 (a). Finally, these boundary points are next analyzed in order to find the neighboring links between connected components. The Fig. 5 (c) gives the neighboring graph into XGMML format (see section 5) of Fig. 5 (a). This method is more complex (θ^2 complexity)[4] than generalized Voronoi diagrams based methods (θ complexity) [3]. However, this method deals with the "high precision" problems. It allows to extract the exact boundary points (with a pixel precision), and gives in some cases more complete results than generalized Voronoi based methods.

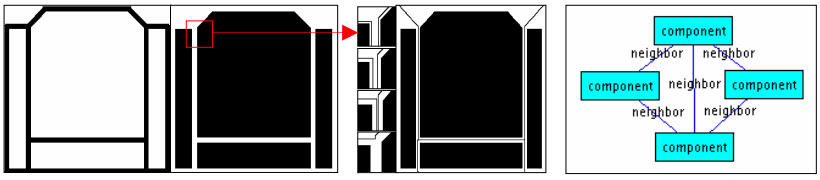

Fig. 5. (a) symbol (b) contours extension (c) neighboring graph

[3] In the paper's follow-up, we talk about symbol's "components" and associated "loops".
[4] 25.1 seconds for a no compressed plan of 364 Kbytes (CPU 2 GHz, Windows System)

Finally, we use a last method in order to construct hybrid graphs, with the inclusion and neighboring links. The Fig. 6 (a) gives an example of hybrid graph into XGMML format (see section 5) of Fig. 5 (a), with the include links between connected components and loops, and the neighboring links between loops (Fig. 5 (c)). Our method uses the neighboring and inclusion graphs, with their associated connected components' characteristics provided by the labelling method (label, gravity centers, areas, and so on.). Next, the method analyses the two graphs and connected components' characteristics in order to construct the hybrid graphs. So, it is possible to construct three hybrid graph types, components based, loops based, loops and components based "both". The Fig. 6 (b), (c), and (d) give examples of three hybrid graph types into XGMML format (see section 5) of Fig. 4 (a).

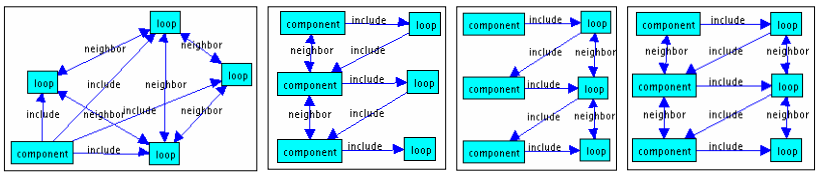

Fig. 6. (a) hybrid graph (b) components based (c) loops based (d) both based

3.3 Images Pre-Processing

The graph construction methods presented in the last subsection (3.2) are based on connected components extraction. Therefore, these methods are very sensitive to sparse noise (small components and loops adding). The Fig. 7 (b) gives an example of sparse noised image of Fig. 7 (a).

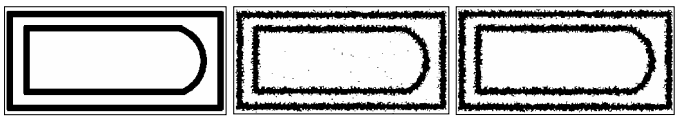

Fig. 7. (a) model image (b) sparse noised image (c) filtered image

To solve this problem, we use a filtering method based on connected components' surfaces analysis (**1**). First, a table (*S)* of no null connected components' surfaces is created. This one can be created from components image, loops image, or to merge the both. From this surfaces table, a surface ratios table *(R)* is computed. The maximum ratio *(r)* is searched in order to find the surface threshold of image *(th)*. The Fig. 7 (c) gives an example of filtered image of Fig. 7 (b).

$$S = \bigcup_{i=1}^{n} s_i \;;\; R = \bigcup_{i=1}^{n-1} \left(r_i = {}^{s_{i+1}}\!/_{s_i} \right) \;;\; r_j = \max(R) \;;\; th = s_j \;. \qquad (1)$$

4 Structural Classification

The extracted graphs (see section 3) are next exploited by an inexact graph-matching algorithm [9]. This algorithm allows to compute[5] similarity criterion between graphs (**2**), based on the overlap between a candidate graph *(g1)* and a model graph *(g2)*. This overlap corresponds to their common sub-graph *(gc)*. Two similarity criteria are computed (**3**) according to the common elements number *{cnn, cen}* on the nodes (δn), and on the edges (δe). The global similarity criterion is obtained by variance computation of (δn) and (δe). The classification's result corresponds to model graph's label of minimum global similarity criterion.

$$g1 = \{n_0,.,n_{nn1};e_0,.,e_{en1}\}; g2 = \{n_0,.,n_{nn2};e_0,.,e_{en2}\}; gc = \{n_0,.,n_{cnn};e_0,.,e_{cen}\}\ . \quad (2)$$

$$\delta e(g1,g2) = \frac{en1 \times en2}{cen^2} - 1\ ;\ \delta n(g1,g2) = \frac{nn1 \times nn2}{cnn^2} - 1\ . \quad (3)$$

The model graphs are learned through a graphics user interface: ojgBE[6]. The Fig. 6 gives examples of graphs' graphics visualizations. ojgBE allows to edit labelled graphs, directed and/or undirected. The edited graphs are included into a graphs base. The ojgBE user can browse into this graphs base and perform various actions like: graph labelling, graph copy, base sorting, graph orientation change, similarity analysis, and so on. The graphs base is stored into XGMML format (see section 5).

5 XML Use in System

The different system's parts (sections 3 and 4) generate output data into XML and its sub-languages. The use of this data representation language offers several enhancements in comparison with classical formats. XML is a meta-language because it is defined as a root language, which enables to define specialized sub-languages. We use SVG[7] for graphic data representations (Fig. 4 (b)). We also use XGMML[8] for graphs descriptions (see Fig. 6) in the global structural analysis tools (see section 3), the structural classifier (see section 4), and the graphics user interface ojgBE (see section 4). XML uses parsers and transforming processors. The parsers easily access to XML data, and the processors easily transform the XML data according to XSLT[9] scripts. This enables an easy data manipulation in the system, for data communication between system' parts (sections 3 and 4), and results evaluation (see section 6).

---

[5] We don't develop this algorithm here and report the reader to [9].
[6] open java graph Base Editor: http://site.voila.fr/mdhws/
[7] Scalable Vector Graphics
[8] eXtensible Graph Markup and Modeling Language
[9] eXtensible Stylesheet Language Transform

6 Experiment, Results, and Analysis

We present here some results obtained on a symbols set of GREC2003[10]'s recognition contest. This symbols recognition contest deals with the recognition of segmented architectural and electrical symbols (Fig. 8). Different sample tests are available[10] according the classes number (symbols set), binary or vectorial degradations[11], scale or orientation changes, and so on. These sample tests are available with their models files, in order to evaluate recognition results.

We have tested our system with a symbols set of 9 symbols (Fig. 8). We have used a hybrid graph representation based on loops' neighboring links (see subsection 3.2). Indeed, the inclusion and neighboring graphs aren't adapted for this recognition application. The inclusion graphs are similar for symbols s8 and s9, and the loops based neighboring graphs for symbols s1 and s7. Also, the components based hybrid graphs are similar for symbols s1 and s4. We have tested our system on 600 images, with 6 sample tests (of 100 images sized) of binary degradations (degrad-level2-m1 to degrad-level2-m6[12]). The Fig. 7 (a) gives an example of binary degradation of Fig. 7 (b).

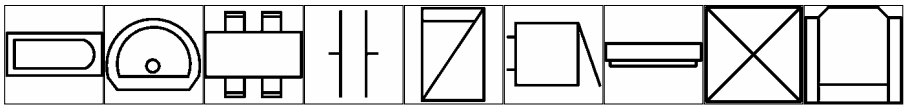

Fig. 8. symbols set

The Fig. 9 (a) gives the samples tests' results. We obtain perfect results on all tests, except on the test5 (89%). In order to complete these results, we have measured two noise types. The first one is an estimation of dilatation noise (**4**). This estimation *(Êd)* is computed between test images *(ti)*, and model images *(mi)*. It is based on the search of common black pixels number *(cbp)* between test and model images, and the black pixel number *(bp)* of test image. The second one is a structural noise. This structural noise gives the rate of noised graphs into the extracted graphs. These noised graphs are detected in regard to their model graphs. These two measures give noise rates before and after the global analysis step. We can see on Fig. 9 a partial correlation between the two noise curves. Indeed, loops and components are "bullet-proof" graphical primitives (the perfect results prove it). The main problem is the dilatation noise which created loops closures and components merges on symbols' noised images. This dilatation noise is the main problem area of structural noise.

$$\hat{E}_d(ti, mi) = 1 - cbp(ti, mi)/bp(ti) \ . \qquad (4)$$

[10] International Conference on Graphics Recognition 2003: http://www.cvc.uab.es/grec2003/

[11] We report the reader to contest's web page[10] for information on the noise models used.

[12] In the section's follow-up, we talk about test1, test2, and so on.

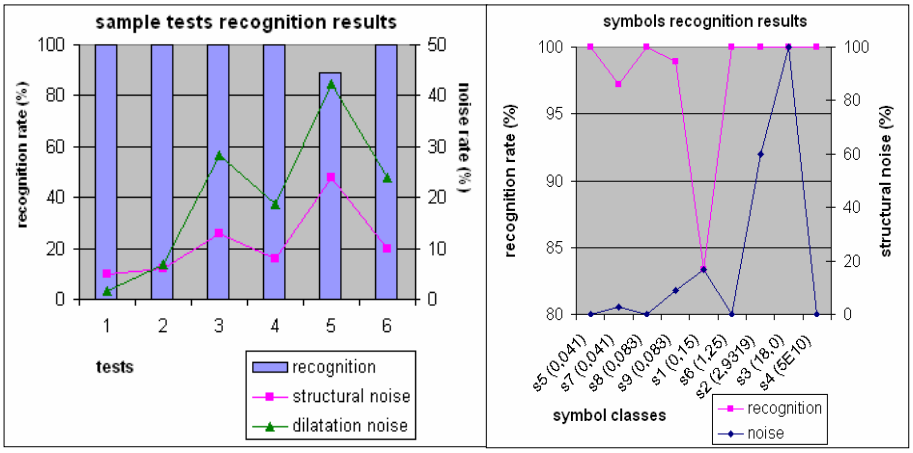

Fig. 9. (a) sample tests recognition results (b) symbols recognition results

In order to analyze the structural noise distribution, we have computed the recognition rate and structural noise rate for each symbol class with all tests' results (Fig. 9 (b)). We can see a high level of noise (>15%) for symbols s1, s2 and s3. The Fig. 10 gives examples of the three main cases of structural noise met for these symbols. The first one is the components merge case (a). Indeed, the dilatation noise can merge the nearest components. It is a typical structural noise of symbol s1 and s2. The second one is the loops closure case (b). The dilatation noise can close the small size loops. This structural noise appears on symbols s2, s3, and s7. The last one is the neighboring links distortion case (c). The loops' distortions create and delete neighboring links in some difficult cases (like s3). The Fig. 9 (c) gives the model neighboring image (left), and an example of test neighboring image (right). The loops' distortions add diagonal neighboring links (with small boundaries) between symbol's loops. Still, the structural noise's effects depend of symbol classes. In the case of symbol s7 and s1, a low level of noise has important effects on the symbols recognition. In the case of symbols s2 and s3, a high level of noise has no effect on the symbols recognition. Indeed, the graph models used haven't the same "bullet-proof" criterion. In Fig. 9 (b) we have computed (and sorted) for each model graph the minimum similarity criterion with the other base's model graphs. This minimum criterion can be view as the "bullet-proof" criterion of considered model graph. Like this, we can see two main classes in the model graphs base: symbols of low minimum similarity criteria (≤0.15) (s1, s5, s7, s8, and s9), and symbols of high minimum similarity criteria (≥ 1.25) (s2, s3, s4, and s6).

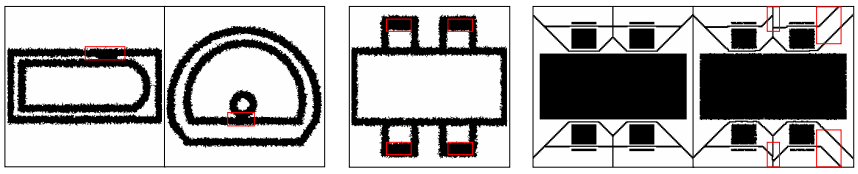

Fig. 10. (a) components merge (b) loops closure (c) neighboring links distortion

7 Conclusion and Perspectives

In this paper we have presented a symbols recognition system applied to graphic documents. This system is based on a combination of global structural approaches. It allows to extract components and their loops, with their inclusion and neighboring links. It is possible to construct different graph types according to the considered recognition problems. These graphs are next processed by an inexact graph-matching algorithm. We present some recognition results and analysis on a noisy symbols set of GREC2003's contest. This system gives very good results. Still, it is sensitive to dilatation noise. Moreover, the global approaches allows to deal only with segmented symbols, and the few classes recognition problem.

For the perspectives, we want first realized contextual pre-processing step, in order to erode the dilated images. Next, we want to combine our global approach with statistical approach [4] in order to deal with the large classes recognition problems. Also, we want to extend our methods library with a generalized Voronoi method, in order to reduce the images' process times in the case of "low precision" problems. Finally, we want to combine our graph-matching algorithm with a graph-grammar tool, in order to correct some structural noise cases.

References

1. A.K. Aggarwal, A.V. Kulkarni. A Sequential Approach to the Extraction of Shape Features. Computer Graphics and Image Processing (CGIP), 6(6) : 538-557, 1977.
2. D. Blostein, H. Fahmy, A. Grbavec. Issues in the Practical Use of Graph Rewriting. Lecture Notes in Computer Sciences (LNCS), 1073 : 38-55, 1996.
3. M. Burge, G. Monagan. Using the Voronoi Tessellation for Grouping Words and Multi Part Symbols in Document. Vision Geometry IV, 1995.
4. M. Delalandre, P. Héroux, S. Adam, E. Trupin, J.M. Ogier. A Statistical and Structural Approach for Symbol Recognition Using XML Modelling. Structural and Syntactical Pattern Recognition (SSPR), 2002.
5. M. Delalandre, E. Trupin, J.M. Ogier. Local Structural Analysis: A Primer. Graphics Recognition (GREC), 2003.
6. E. Hancock, R. Wilson. Graph-Based Methods for Vision: A Yorkist Manifesto. Structural and Syntactical Pattern Recognition (SSPR), 2002.
7. R. Kasturi, L. O'Gorman, V. Govindaraju. Document Image Analysis: A Primer. Sadhana, 27(1) : 3-22, 2002.
8. O. El Badawy, M. Kamel. Shape Representation using Concavity Graphs. International Conference on Pattern Recognition (ICPR), 2002.
9. P. Héroux, S Diana, E. Trupin, Y. Lecourtier. A Structural Classification for Retrospective Conversion of Document. Structural and Syntactical Pattern Recognition (SSPR), 2000.
10. J. Lladós, E. Valveny, G. Sánchez, E. Martí. Symbol Recognition : Current Advances an Perspectives. Graphics Recognition (GREC), 2001.
11. P.K. Loo, C.L. Tan. Detection of Word Group Based on Irregular Pyramid. International Conference on Document Analysis And Recognition (ICDAR), 2001.