Electronic Letters on Computer Vision and Image Analysis 5(2):16-29, 2005

Contextual System of Symbol Structural Recognition based on an Object-Process Methodology

Mathieu Delalandre* and Eric Trupin* and Jean-Marc Ogier+ and Jacques Labiche*

* *PSI Laboratory, Rouen University, 76821 Mont St Aignan, France*
+ *L3I Laboratory, La Rochelle University, 17042 La Rochelle, France*

Received21 July 2004; accepted 10 March 2005

Abstract

We present in this paper a symbol recognition system for the graphic documents. This one is based on a contextual approach for symbol structural recognition exploiting an Object-Process Methodology. It uses a processing library composed of structural recognition processings and contextual evaluation processings. These processings allow our system to deal with the multi-representation of symbols. The different processings are controlled, in an automatic way, by an inference engine during the recognition process. The factual, strategic, and descriptive knowledge are structured according to an Object-Process Methodology of the considered recognition application. The representation of symbols are then adapted during the recognition process according to their classes. This original approach for knowledge structuring allows especially the strategic and descriptive knowledge combination which increases the recognition abilities of system. We present results of this system on the GREC2003 contest of symbol recognition.

Key Words : Symbol Recognition, Graph, Multi-Representation, Automatic Control of Processing, Context, Knowledge, Object-Process Methodology

1 Introduction

Symbol recognition [20] is an important topic of document image analysis [19], and especially of graphic document interpretation [1]. Like any system of document image analysis, we can decompose a symbol recognition system into three main parts [24] : a recognition chain, a control system, and a knowledge base. The recognition chain is composed of a set of processings. These processings allow to extract and to exploit graphic primitives. The control system and the knowledge base are then used in order to manage these processings.

During a recognition process, knowledge are used and produced into these system's parts. It exists several implicit links of structuring between these knowledge. The use of these links of structuring can increase the recognition abilities of a system [25], however they are few exploited. In this paper, we present a system using the knowledge structuring for symbol recognition. Especially, this system deals with three important features : a recognition chain using the multi-representation of symbols, a control system using the recognition context and the structuring of knowledge base. In the section's follow-up we present these features and their importance into recognition systems.

Correspondence to: <mathieu.delalandre@univ-rouen.fr>

Recommended for acceptance by < T. Paquet, G. Sanchez>
ELCVIA ISSN:1577-5097
Published by Computer Vision Center / Universitat Autònoma de Barcelona, Barcelona, Spain

A recognition chain is classically decomposed in two main steps : an image processing step and a recognition step. Two main approaches exist : statistical and structural. This paper deals especially with the structural approach. This approach uses graph representations of objects composing documents. In graphic documents, many objects could be described by graphs and especially symbols [9]. Thus, in a structural recognition system of symbol, the image processing step extracts graphs corresponding to symbols, and the structural recognition step exploits these graphs. The structural recognition step uses two main approaches : graph-matching [5] and graph-grammar [4]. The first one matches extracted graphs with model graphs. The second one applies different rules to transform extracted graphs into model graphs. The image processing step extracts graphic primitives from images (component, stroke, vector, and so on.) and builds graphs with these graphic primitives. This step uses many various approaches [20] like : region graphs, contouring and skeletonisation, line tracking, and so on. In [10] we present how these approaches allow to build different graphs corresponding to different structural representations of symbol. The Figure 1 gives an example of three different structural representations of a symbol. The recognition abilities of a system depends, in an important way, of the adopted structural representation [22]. The exploitation of different structural representations into a recognition system constitutes certainly an important research topic of symbol recognition : we talk about multi-representation of symbol.

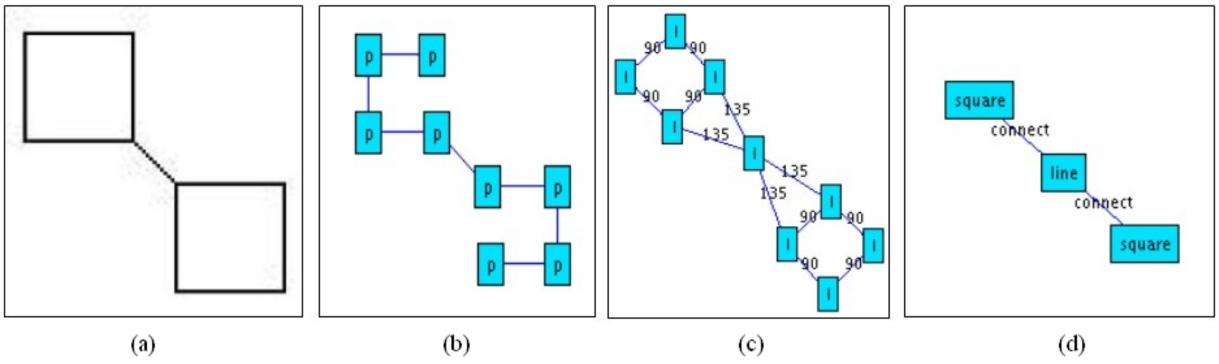


Figure 1: (a) symbol (b) point graph (c) line graph (d) square graph

The control system is the header part of a recognition system. This one manages the recognition chain and the different knowledge used or produced during the recognition process [1]. It exists two main system types : planned and contextual systems. The first ones have fixed recognition strategies. These systems are "historical" and often used for symbol recognition [20]. They are aborted and replaced by contextual systems. These last ones adapt themselves their recognition strategies according to the recognition context. It exists several types of context. The noise on document images is a first type of context. The image processings used by a system may change in an important way according to the noise level and type [23]. The dimension of recognition problem is another example of context. For each set of symbol, a recognition system must change its recognition strategy [21]. The segmentation of symbols is a last example of context. This one needs complex interactions between processings and system [25].

Various approaches exist among contextual systems : the Multi Agent Systems (*MAS*)[*], the blackboard systems, and the systems of automatic control of processings. However none of these approaches is overridden [1]. The MAS are a first approach of contextual systems. They are fluently used in artificial intelligence [27]. In MAS, each processing (or processing chain) is integrated into a agent. They use a decentralized control, indeed the control of complete recognition process is shared between system's agents. However, these systems are few used among systems of graphic document interpretation [15]. The blackboard systems are another approach of contextual system. The blackboard systems [3] are neighbor systems of MAS. In blackboard systems, each processing (or processing chain) is integrated into a process. A control process controls the accesses to a central memory (the blackboard) and runs the processes according to the blackboard state. This approach is fluently

[*]**Please see annexe A for a resume of all acronyms used in this paper.**

used among systems of graphic document interpretation. The systems for automatic control of processings [28] are a last approach of contextual system. They are based on the use of learned strategies combined with production rules. These production rules allow to change strategies according to evaluation of recognition results. Some works use this approach for graphic document interpretation [7].

Like this, several approaches of contextual systems and types of context exist. However, the recognition systems deal generally with only one type of context. In order to use an approach based on the multi-representation, a system must deal with several types of context. Indeed, the efficiency of a given representation may change a lot according the context [6].

The system's knowledge can be "dynamic" or "static" [26]. In the last case, the knowledge base groups and structures the used or produced knowledge by a system. A recognition system can deal with three types of knowledge [24] : factual, descriptive and strategic. The factual knowledge groups the processed data (images and/or structured data) and their contextual data (quality, type, and so on.). The processed data [32] can be input data, or produced by other systems. The contextual data can be obtained by machine man interfaces [24], or by contextual evaluation processings [23]. The descriptive knowledge describes the document's objects. It is used by recognition processings. Two representation formalisms are then used in structural recognition systems : rules [4] and model graphs [5]. Other hybrid formalisms can be used in approaches like : statistical/structural, matching/structural, and so on. This descriptive knowledge can be acquired by user edition [11], or by learning [18]. The strategic knowledge describes the graphs corresponding to processing sequences (scenarios). It exists two types of building systems : based on user graphic programming of scenarios [24], or contextual [15]. In the last case, a system produces itself its scenarios according to the recognition context. So, it exists several types of knowledge. Only some knowledge types are generally used in each system [24]. However, it exists several structuring links between these knowledge which can be used to increase the recognition abilities of a system [22]. Moreover, these knowledge are in relation with the multi-representation (strategic and descriptive) and the context (factual).

In this paper we present a contextual system for structural recognition of symbol. The Figure 2 gives an overview of this system. It is based on an automatic control of processings using an inference engine. It uses a processing library for graphic primitive extraction and exploitation, and contextual evaluation. This processing library allows the multi-representation of symbols. Based on this library, our system deals with different types of context : especially the noise and dimension contexts. The different knowledge into system are structured according to an Object-Process Methodology. This structuring allows the strategic and descriptive knowledge combination. It is performed by the use of a set of machine man interface. The representations of symbols are then adapted during the recognition process according to their classes. This original approach for knowledge structuring increases the recognition abilities of system.

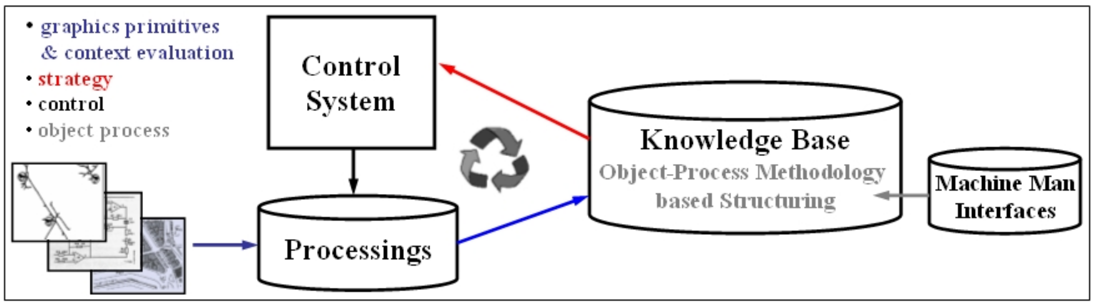


Figure 2: system overview

In the paper's follow-up, we present in section (2) our processing library. Next, we present in section (3) our contextual system based on an Object-Process Methodology, and a set of machine man interface for knowledge acquisition and structuring. In section (4) we present a system's use-case on the GREC2003 contest of symbol recognition. Finally, in section (5) we conclude and give perspectives.

2 Processing Library

In this section we present processings used in our system. These ones are shared according to different functions : pre-processing / extraction / recognition or evaluation. Also, they are divided according to their output data types : images, structural or statistical primitives. The works about these processings have been published in [2] [8] [9] [11] [16]. So, we introduce these ones in the three subsections (2.1) (2.2) (2.3) before the presentation of our system in section (3). In subsection (2.4) we present their combination for the symbol multi-representation.

2.1 Images

We use different image pre-processings and joined processings for contextual evaluation of noise [8] [11]. Our pre-processings are grouped in two types : morphological pre-processings (erosion, dilatation, opening, and closing), and processings based on connected component filtering (using automatic or user thresholds). We use two joined processings for contextual evaluation, one for morphological noise (broken shapes, closed loops, and so on.), and one other for sparse noise (small component adding). The first one estimates thicknesses of shapes according to interiority maps provided by a skeletonisation tool. These thickness estimations allow to detect (under scale constraints) the dilatation and erosion noises on images. The second one is based on analysis of connected component surfaces. This analysis builds and sorts a surface table of image's connected components. From this surface table, a table of surface ratio is computed. This ratio table is processed in order to detect a biggest ratio corresponding to sparse noise.

2.2 Structural Primitives

Concerning the structural primitives, we first use a processing library allowing the multi-representation of region graphs [11]. It extracts the symbol's components and their joined "loops", with their inclusion and neighboring links. The "loops" are background components of symbols (Figure 3 (a)). So, it is possible to build different graph types according to the recognition problem (with the inclusion links, and the neighboring links between loops and/or components). The Figure 3 gives an example of symbol with its loops (a) and its region graph (b). This graph represents symbol's components and loops, with the inclusion links between components and loops, and the neighboring links between loops.

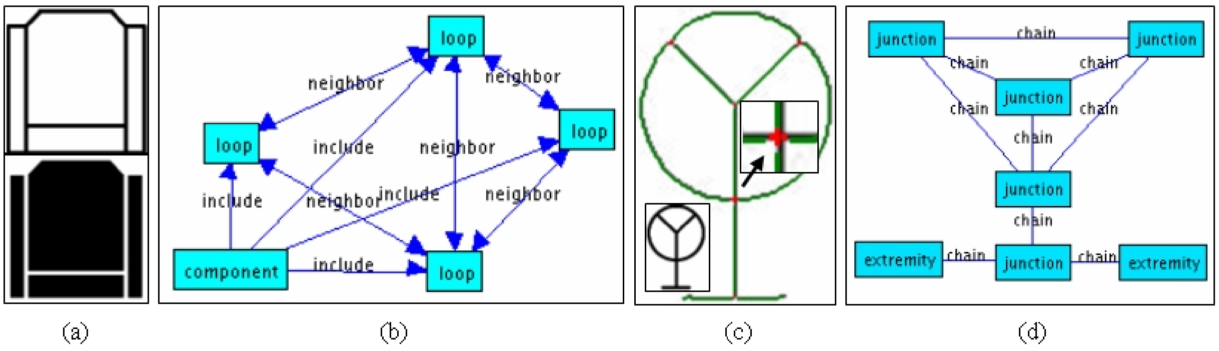


Figure 3: (a) symbol and loops (b) region graph (c) symbol and skeleton chaining (d) skeleton graph

We next use a processing library allowing the multi-representation of line graphs [8]. It combines different approaches like : skeletonisation, contouring, run analysis, direct vectorisation, object simplification, and so on. In order to use this library with our approach based on Object-Process Methodology, we have decomposed this one into granular levels. So, it is possible to build different graph types, or to use different approaches for a same graph type, according to the recognition problem. The Figure 3 gives an example of symbol and graphic representation of skeleton chaining (c) and the corresponding skeleton graph (d). The graph's nodes represent the skeleton's junctions and extremities, and the edges the point chains.

The line and region graphs are next exploited by a graph-matching algorithm [16]. This algorithm allows to compute similarity criterion between graphs (Equation 1), based on the overlap[†] between a candidate graph (g_1) and a model graph (g_2). This overlap corresponds to their common sub-graph (g_c). Two similarity criteria are computed (Equation 2) according to the common elements number $\{nc, ec\}$ on the nodes (δ_n) and on the edges (δ_e). The global similarity criterion is obtained by variance computation of (δ_n) and (δ_e). The classification's result corresponds to model graph's label of minimum global similarity criterion. We also use this graph-matching tool as contextual evaluation processing. The maximum similarity criterion is then used as quality evaluation of extracted graphs.

$$g_1 = (\{n_0, ., n_{n1}\}, \{e_0, ., e_{e1}\}) \quad g_2 = (\{n_0, ., n_{n2}\}, \{e_0, ., e_{e2}\} \quad g_c = (\{n_0, ., n_{nc}\}, \{e_0, ., e_{ec}\}) \tag{1}$$

$$\delta_n(g_1, g_2) = \frac{n1 \times n2}{nc^2} - 1 \qquad \delta_e(g_1, g_2) = \frac{e1 \times e2}{ec^2} - 1 \tag{2}$$

2.3 Statistical Primitives

We use different extractors of statistical primitives (geometric features, Zernike moments, Fourier-Mellin invariants, circular probes) [2] in order to describe the symbol components or/and their loops. These statistical primitives corresponds to feature vectors exploited by a k-nearest neighbor algorithm. This algorithm allows to use different distances between feature vectors. It is also used as contextual evaluation processing, the distance classification gives a quality criterion of extracted primitives.

2.4 Processing Combination for the Multi-Representation of Symbol

All our processings are based on the use of our graphical knowledge formalism [9]. This formalism is object-oriented and based on the inheritance, polymorphism and extensibility properties. Like this, this formalism is "generic" and allows various modellings of a given graphic shape. We represent and operationalize this formalism through a modelling library. This library is then used in our processings, allowing to request graphical knowledge bases according to the processings' requirements. Like this, this approach allows the interoperability between processings.

Based on our processings and our formalism, our system allows to extract different graphic primitives and to build different types of graph. Our processings can be combined in order to specialize graphs. The Figure 4 gives some examples of graph specialization based on the symbol (a). The skeleton graph (b) is obtained following skeletonisation and chaining processings. The vector graph (c) is a specialization of skeleton graph (b) after a polygonalisation processing. The region graph (e) is obtained after a loop extraction (d) of symbol (a), and the neighboring and inclusion links building between the component (a) and its loops (d). A statistical recognition of these loops specializes this region graph (e) into a low-level symbol graph (f). At last, the graph (g) corresponds to a dual specialization of graphs (c) and (f).

3 Contextual System based on an Object-Process Methodology

We use our processing library in our system to control the symbol recognition process. This system is based on an automatic control of processings by an inference engine. The factual, strategic, and descriptive knowledge are managed inside system according to an Object-Process Methodology (*OPM*). In the section's follow-up, we present first in subsection (3.1) an introduction to OPM and object building strategies. Next in subsection (3.2), we present our control system based on an inference engine. In subsection (3.3), we present a set of machine man interface for the management of knowledge bases. Finally in subsection (3.4) we conclude on this system.

[†]We don't develop this algorithm here, it's not the purpose of this paper to do this, we report the reader to [16].

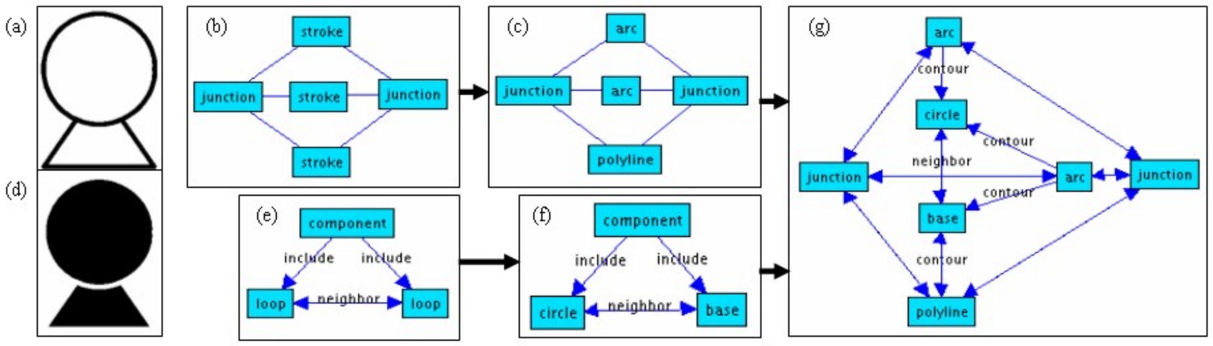
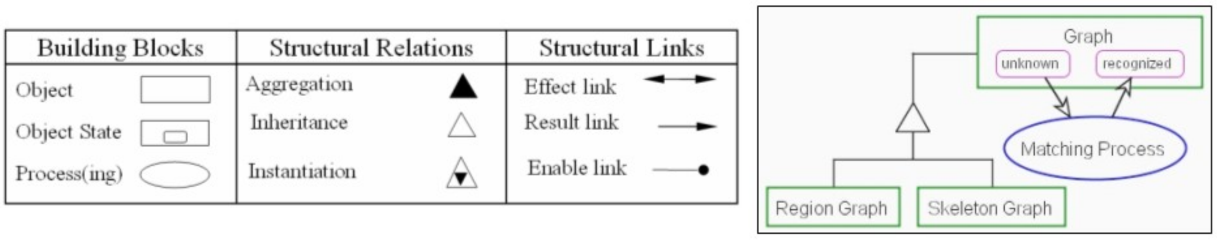


Figure 4: (a) symbol (b) skeleton graph (c) vector graph
(d) symbol's loops (e) region graph (f) low-level symbol graph
(g) vector/low-level symbol graph

3.1 Introduction to OPM and Object Building Strategies

The OPM is a modelling tool based on object-oriented concepts. It was introduced by [13]. Several object-oriented modellings (SADT, UML, and so on.) exist to model information systems [17]. The OPM is especially based on a formalism of attributed graph : the Object-Process Diagram(s) (*OPD*). The Figure 5 (a) resumes some main notations used to describe the OPD. The OPD are based on two main building blocks : object and processing. Different structural relations and links are then used between these building blocks like : inheritance, aggregation, result and effect, and so on. These links are usual in the object-oriented modellings. So, the main property of OPM is the dual use of processing and object building blocks. This object/processing use into OPM is especially adapted for the modelling of graphics recognition system [12]. The Figure 5 (b) gives an example of OPD which can be interpreted like this : "region graph and skeleton graph objects are specialized of graph object, matching process changes the state of graph object from unknown to recognized".



Building Blocks		Structural Relations		Structural Links	
Object	▭	Aggregation	▲	Effect link	⟷
Object State	▣	Inheritance	△	Result link	⟶
Process(ing)	⬭	Instantiation	◬	Enable link	—●

Figure 5: OPD (a) notations (b) example

Based on OPM, any graphics recognition system can be modelled through an OPD [12]. Current works based on OPM [31] deal with generic algorithms easily extensible to adapt graphics recognition systems. However, the object-oriented modellings do not involve object building strategies. These ones concern the interactions between objects during the building process in order to simplify the considered problem. From our point of view [10], the object building strategies are an important research perspective of graphics recognition. However, only few works deal with this topic [25] [22]. [25] uses an object building strategy in his Object-Oriented Progressive-Simplification based Vectorization system (*OOPSV*). The Figure 6 gives an result example of OOPSV. The image (b) is obtained after bar vectorisation and simplification on initial image (a). The image (c) is then obtained after arc vectorisation and simplification on image (b). [22] uses a similar strategy but based on a representation of quadrilateral graph.

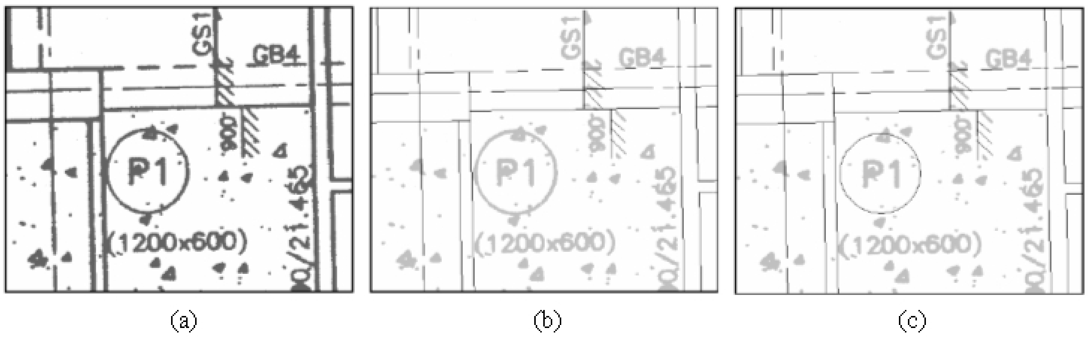


Figure 6: OOPSV (a) image (b) bar simplification (c) arc simplification

In [10] we have proposed a classification of processing combinations into cooperative, hybrid, and comparative. We can consider than [25] [22] use a cooperative combination between processings for their object building strategy. Indeed, the results of a given processing are used to simplify its input data. These simplified input data are then used by the next processing. Our system presented in the next subsection allows hybrid combinations for its object building strategy. In these combinations, the results of different processings are merged to specialize the symbol representation (subsection (2.4)). Like this, our system uses the representation specialization in order to build "step by step" the recognition result, from more simple to more complex. The knowledge used in our system are implicitly formalized according to an OPD linked to a given recognition application. This OPD allows a unified representation of system's knowledge.

3.2 Control System by Inference

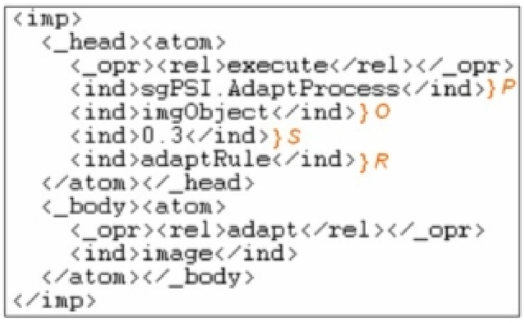
Our control system is named rule based system for Object-Process Methodology (*rsOPM*)[‡]. It allows the automatic control of processings using an inference process. In this way, rsOPM is generic and adaptable. Indeed, only the rule base and rsOPM are used to implement the control part of a recognition application. rsOPM uses two operator types : the control operator and the interface operator.

The control operator is the central part of rsOPM. It allows to infer the rule base and a dynamic load of processings through their corresponding interface operators. It uses the Mandarax engine[§] based on the Rule Markup Language (*RuleMl*) [30]. This rule programming language is based on the use of different rule types [30] like the deduction rule. Therefore, the inference engine depends of a set $\{F, R, Q\}$ where (F) is the fact base, (R) the rule base, and (Q) the query. (F) groups the factual knowledge extracted by the contextual evaluation processings (see section 2). This fact base (F) is used with the rule base (R) to trigger the execution of processings. The deduction rule base (R) corresponds to the recognition strategy and production rules used in the system (see section (1)). The result of the rule evaluation (by the engine), following the query (Q), is used to run a given processing. During the inference process, the OPD of current process is built. The priority between the rules is defined by their order in (R).

The deduction rules used in rsOPM are defined according to the OPM by a set $\{P, O, S, R\}$, with (P) the processing to execute, (O) the objects used or produced by this processing, (S) the settings of this processing, and (R) the rules to update into rule base after processing execution. The Figure 7 (a) gives an example of OPM based rule used into rsOPM. This rule is divided in two parts, the header and body. The header part defines the set $\{P, O, S, R\}$. The body part corresponds to fact (F) triggering the rule execution. So, this rule can be interpreted like this : "execute adaptProcess (P) with imgObject (O), 0.3 (S), adaptRule (R) if adapt image". The deduction rules can be next extended according to the number of fact (F) and their logical relations, and the configuration of $\{O, S, R\}$.

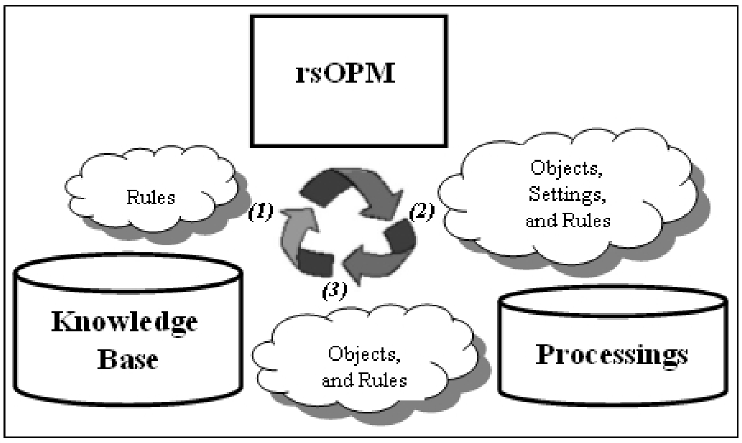
[‡]Available on : http://mdhws.site.voila.fr/

[§]http://sourceforge.net/projects/mandarax



```
<imp>
  <_head><atom>
    <_opr><rel>execute</rel></_opr>
    <ind>sgPSI.AdaptProcess</ind>}P
    <ind>imgObject</ind>}O
    <ind>0.3</ind>}S
    <ind>adaptRule</ind>}R
  </atom></_head>
  <_body><atom>
    <_opr><rel>adapt</rel></_opr>
    <ind>image</ind>
  </atom></_body>
</imp>
```

(a)



(b)

Figure 7: rsOPM (a) OPM rule (b) object building cycle

The interface operators are plug-ins for processings (or processing chains) defined according to the recognition application. They control through the set $\{O, S, R\}$ the used and produced objects by the processings, as well as the update of rule base by the processings. At last, the building rules of OPD are defined inside these interface operators. These ones are then used by the control operator during the process to build the OPD.

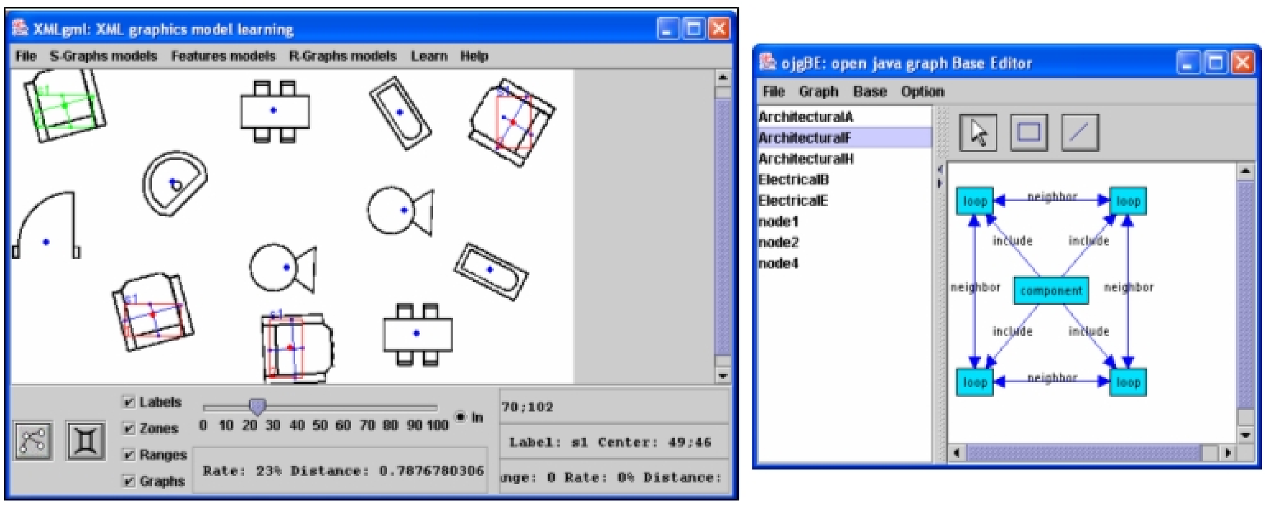
During the object building process, rsOPM interacts with the processings and the knowledge base according to an object building cycle. The Figure 7 (b) presents this object building cycle. First (***1***), the rule base $\{F, R\}$ is exploited by rsOPM to deduct the processing to run. So, this rule base $\{F, R\}$ must be initialized before system starting in order to boot rsOPM. Different knowledge are then extracted (***2***) from knowledge base by rsOPM and given to executed processing like : object to process (image, graph, feature vector, and so on.), settings (thresholds, learning base, and so on.), and fact to update into the knowledge base. Following the processing execution (***3***), different knowledge (object, rule) are then stored into the knowledge base. The object building cycle is following while fact(s) (F) can trigger a processing (***1***).

3.3 Knowledge Management

We use a set of Machine Man Interface (*MMI*) for the management of knowledge bases corresponding to recognition applications. These MMI are used for the learning of graphics primitives and to edit the rule bases.

We use a first MMI in our system for the learning of graphic primitives : the XML graphics model learning (*XMLgml*)‡. XMLgml (Figure 8 (a)) allows the learning of graphic primitives based on example of user. XMLgml uses different representations of graphic primitives of type feature and/or graph. The primitive extraction inside XMLgml uses our processing library (section (2)). Using XMLgml, the user performs similarity search from a selected graphic object (green displayed on Figure 8) according to the used representation. He can next label the similar graphic objects according to their similarity distances (red displayed on Figure 8), analyse the distance map of learned base, and change a labelling action at every time.

XMLgml is based on a mono-representation learning and can't combine different representations in order to specialize graphs (subsection (2.4)). Indeed, a "multi-representation" learning involves to define a learning scenario which needs a more complex use and setting of MMI [24]. In order to solve this problem, we have developed another MMI for graph base edition : open java graph Base Editor (*ojgBE*)‡. ojgBE (Figure 8 (b)) allows to combine our graph representation in a hand-user way. The user can edit the labelled graphs (edges and nodes), directed and/or undirected (right panel on (Figure 8 (b)). The edited graphs are included into a graph base. The ojgBE user can browse into this base through a label list (left panel on Figure 8 (b)). Various actions can be performed on a base like : graph copy, base sorting, and so on.



```
70;102
Label: s1 Center: 49;46
Rate: 23% Distance: 0.7876780306
nge: 0 Rate: 0% Distance:
```

(a) (b)

Figure 8: (a) XMLgml (b) ojgBE

At last, we use TextPad[¶] as programming environment to use rsOPM. Indeed, rsOPM can be use as a TexPad's plug-in to edit, to parse, and to infer the rule bases and to execute the dynamic load of processings. Like this, we use rsOPM in combination with other development tools[‖] in order to develop the operator interfaces and to test them with the rule base.

The system uses a complete XML based representations for its not-image data. This XML representation is used in control system, processing library and thus MMI (theses ones are based on our processing library)[**]. XML is a meta-language because it is defined as a root language, which enables to defined specialized sub-languages. We use two sub-languages in addition with the XML root language : Rule Markup Language (*RuleML*) in rsOPM, and eXtensible Graph Markup and Modelling Language (*XGMML*) for our graph representations.

3.4 Conclusion

In our system, the knowledge used are implicitly structured according to an OPD linked to a given recognition application. This OPD allows a unified representation of system's knowledge. The objects and joined processings represent the descriptive and strategic knowledge. The factual knowledge is represented through the different building states of objects. This unified representation offers a main advantage : the strategic organization of descriptive knowledge. Indeed, the different learning bases of a recognition application can be shared according to the different building states of objects. These learning bases are then structured through a rule base and thus the recognition application's OPD. This allows a "step-by-step" recognition, through the different state of object building, from more simple to more complex. The factual knowledge is then used to control the building of objects. During the object building process, rsOPM builds the OPD corresponding to the current recognition. We present in the next section a use-case of recognition application with a linked OPD.

[¶]http://www.textpad.com/
[‖]gcc and g++, J2SDK, Xalan, and so on.
[**]We report the reader to [8] [9] [11] for a detailed presentation of the XML use into processing library.

4 Use-Case

We present here a system's use-case and results on GREC2003[††] contest of symbol recognition [29]. The symbol recognition contest deals with the recognition of segmented (A) architectural and (E) electrical symbols (Figure 9) on binary images. Different symbol sets are online available according the class number of symbol (15, 20, and 40). Two main models of noise have been applied (and combined) on the ideal symbols, the binary degradation and the vectorial distortion (Figure 10). These noise models use settings to apply different noise levels. In the two following subsections we present our recognition applications and their results for the symbol sets with 20 and 15 classes.

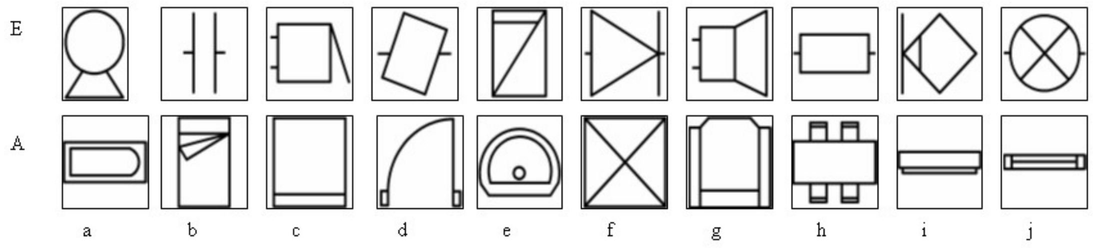


Figure 9: Symbol set with 20 classes

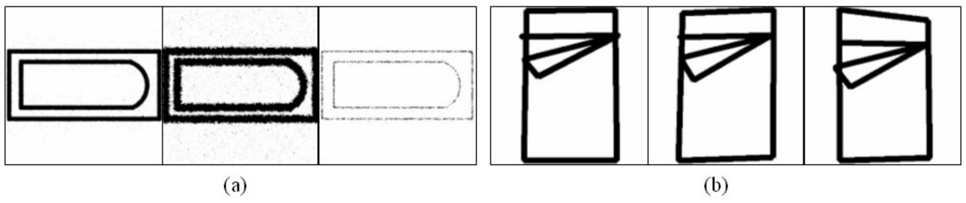


Figure 10: (a) 3 levels of binary degradation (b) 3 levels of vectorial distortion

4.1 Symbol Set with 20 Classes

This symbol set is given on (Figure 9). 9 tests are available (100 images each, for complete tests of 900 images) according to different noise levels of binary degradation (Figure 10 (a)). We have used rsOPM through TextPad (subsection (3.3)) in order to edit the rule base of this application.

First of all, we use our system in order to perform a contextual pre-processing on degraded binary images. This contextual pre-processing is based on the evaluation of sparse and morphological noises. Two evaluation processings analyse the image, and update facts into the rule base. According these facts, the system executes different pre-processings. The Figure 11 gives execution examples of contextual pre-processing. The Figure 11 (a) corresponds to one evaluation (image valid and sparse), one execution (component filtering). The Figure 11 (b) corresponds to two evaluations and executions : eroded and sparse thus morphological filtering, eroded and no sparse thus thickness restoring.

Next, we process our pre-processed images for the structural recognition of symbol. This recognition uses a statistical and structural recognition of symbols (see subsection (2.4)). The representation of symbol is based on region graph of Figure 4 (e). The loops of symbols are recognized during a statistical recognition step, and the graphs are specialized into low-symbol graphs (Figure 4 (f)). The statistical recognition of loops is based on geometric features (surface, perimeter, compactness). We have used XMLgml znd ojgBE for the learning of statistical and structural bases (section (3.3)). This learning deals with only ideal images. The Figure 12 (a) compares recognition results of OPM and direct approaches based on pre-processed images.

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

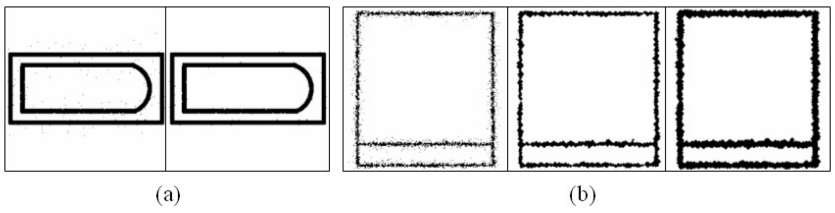


Figure 11: contextual pre-processing (a) one processing (b) two processings

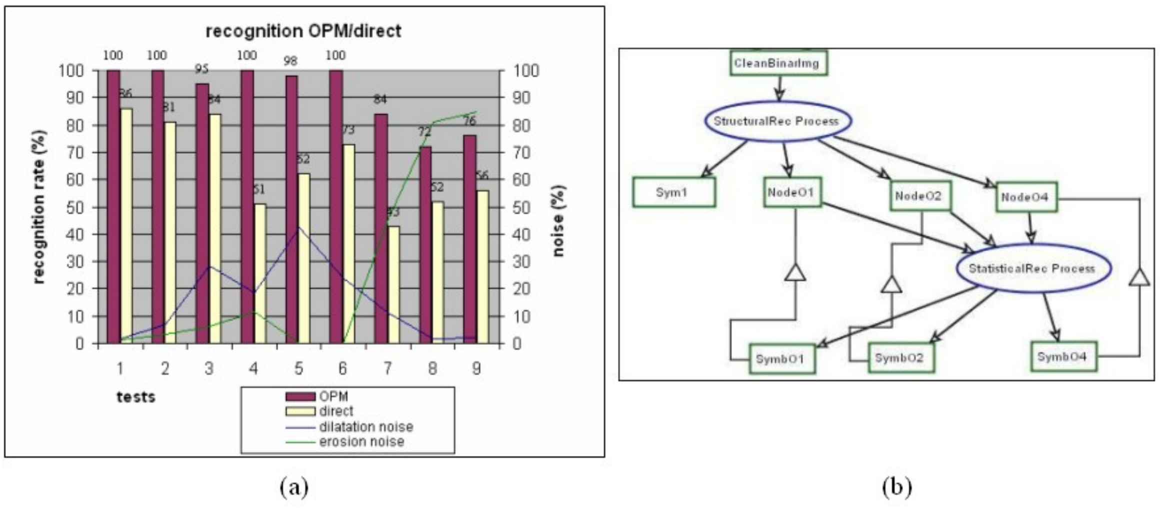


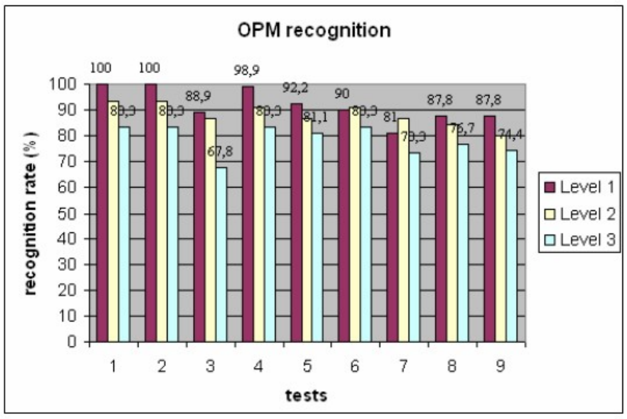
Figure 12: (a) recognition OPM/direct (b) OPD of OPM based recognition

The direct recognition performs a sequential recognition, chaining successively the different recognition steps (extraction of a region graph, statistical recognition then structural). The contextual recognition is based on OPD presented on Figure 12 (b). This contextual recognition first performs the structural recognition of symbols. Following this recognition step, the recognized graphs are shared into four objects : $Sym1$ $\{A_a$, A_e, A_h, E_b, E_e), $Node01$ $\{E_c$, E_d, E_f, $E_h\}$, $Node02$ $\{A_c$, A_d, A_i, E_a, E_g, $E_i\}$, $Node04$ $\{A_b$, A_f, A_g, A_j, $E_j\}$. The $sym1$ object groups the symbol graphs directly recognized during the first structural recognition step. The other objects group the symbol graphs (composed of 1, 2, and 4 loops) which need a specialization with a statistical recognition (Figure 4 (e) (f)). The results of structural recognition are then used to control the OPD building of processed symbol, and thus to decide of path into OPD given on Figure 12 (b). Through this OPD, the descriptive knowledge has been strategically organized in several structural and statistical bases. This strategic organization shares the descriptive knowledge during the recognition process, reducing the recognition complexity, and increasing then the system's recognition abilities (Figure 12 (a)). Indeed, for tests 1 to 6, the recognition results are of 98.8% whereas the learning is based on ideal images.

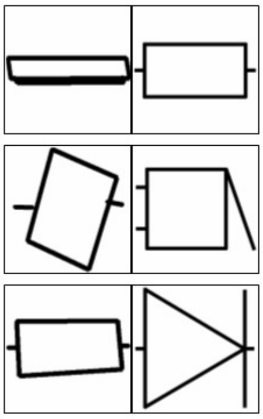
However, this approach is bounded by recognition abilities of the first building step [11]. For tests 1 to 6, this recognition is of 99.7%. Indeed, the connected components and their joined loops, as well as their inclusion and neighboring links, are simple to extract and to recognize. But, these graphic primitives are very sensitive to morphological noises (erosion and dilatation). The Figure 12 (a) gives evaluation measures of morphological noises [11]. This estimation is computed between the test and model images. It is based on the search of the common black pixel number between the test and model images, and the black pixel number of test image. We can see on Figure 12 (a) correlations between the evaluations of two noise types and recognition results. For tests 1 to 6, the no perfect results correspond to high level of dilatation noise. Next, the low recognition rates for tests 7 to 9 correspond to high level of erosion noise.

4.2 Symbol Set with 15 Classes

This symbol set is a subset of the symbol set with 20 classes (see subsection (4.1)). It corresponds to Figure 9 without symbols A_a, A_d, A_e, E_a, and E_j. 27 tests are available (90 images each, for complete tests of 2430 images) according to different noise levels combining binary degradations (Figure 10 (a)) and 3 vectorial distortions (Figure 10 (b)). The recognition application used for this symbol set is based on the same approach than subsection (4.1). Obviously, a new OPM have been defined for this symbol set. The Figure 13 (a) gives recognition results on this symbol set according to the three levels of vectorial distortions. These results are lower than results on symbol set with 20 classes (Figure 12 (a)). So, our approach is sensitive to vectorial distortions. However, the recognition errors result of symbols of "low" structural representation (few loops and components). Indeed, the recognition of these symbols is then based in main part on the statistical recognition. This recognition uses basic geometric features and is sensitive in some cases to vectorial distortions. The Figure 13 (b) gives frequent recognition errors with distorted symbols (left) and false recognized symbols (right).



(a)



(b)

Figure 13: (a) OPM recognition of symbol set with 15 classes (b) 3 frequent cases of recognition error

5 Conclusion and Perspectives

In this paper we have presented a contextual system of symbol recognition based on an Object-Process Methodology. This one exploits a processing library composed of graphic primitive extraction processings and contextual evaluation processings. These processings allow to our system to deal with the multi-representation of symbols. An inference engine controls, in an automatic way, the processings during the recognition process. The factual, strategic, and descriptive knowledge of system are structured according to an Object-Process Diagram of considered recognition application. This Object-Process Diagram follows a building strategy to adapt the representation of symbols during the recognition process. This original approach for the knowledge structuring increases the recognition abilities of system. We present a system's use-case and results on GREC2003 contest of symbol recognition.

For the perspectives we first want to apply our system to other applications, like document interpretation or indexing. In this way, we wish to improve our processing library for the graphic primitive extraction on several approaches : contour matching, direct vectorisation, and run analysis. The use of these different processings will allow us to deal with more complex OPD, and thus more complex building strategies. Next, we would like to learn, in an automatic way, the knowledge's system (and especially OPD) from training symbol sets. This automatic learning of structuring knowledge dealing with the symbol multi-representation seems a complex problem. We think that the use of the Minimum Description Length (MDL) [14] for the representation selection can constitute an interesting way for this problem.

References

[1] S. Ablameyko and T. Pridmore. *Machine Interpretation of Line Drawing Images*. Springer Verlag Publisher, 2000. ISBN 3-540-76207-8.

[2] S. Adam, J. Ogier, C. Cariou, J. Gardes, and Y. Lecourtier. Combination of invariant pattern recognition primitives on technical documents. In *Workshop on Graphics Recognition (GREC)*, pages 238–245, 1999.

[3] A. Barr, P. Cohen, and E. Feigenbaum. *The Handbook of Artificial Intelligence*, volume 1-4. Addison Wesley Publisher, 1989. ISBN 0201118106.

[4] D. Blostein, H. Fahmy, and A. Grbavec. Issues in the practical use of graph rewriting. *Lecture Notes in Computer Science (LNCS)*, 1073:38–55, 1996.

[5] H. Bunke. Recent developments in graph matching. In *International Conference on Pattern Recognition (ICPR)*, pages 117–124, 2000.

[6] Y. Chen and Y. Yu. Thinning approaches for noisy digital patterns. *Pattern Recognition (PR)*, 29(11): 1847–1862, 1996.

[7] E. Clavier, G. Masini, M. Delalandre, M. Rigamonti, K. Tombre, and J. Gardes. Docmining: A cooperative platform for heterogeneous document interpretation according to user-defined scenarios. *Lecture Notes in Computer Science (LNCS)*, 3088:13–24, 2004.

[8] M. Delalandre, Y. Saidali, J. Ogier, and E. Trupin. Adaptable vectorisation system based on strategic knowledge and xml representation use. *Lecture Notes in Computer Science (LNCS)*, 3088:196–207, 2004.

[9] M. Delalandre, E. Trupin, J. Labiche, and J. Ogier. Graphical knowledge management in graphics recognition systems. In *Workshop on Graph-based Representations (GbR)*, pages 35–44, 2005.

[10] M. Delalandre, E. Trupin, and J. Ogier. Local structural analysis: A primer. *Lecture Notes in Computer Science (LNCS)*, 3088:220–231, 2004.

[11] M. Delalandre, E. Trupin, and J. Ogier. Symbols recognition system for graphic documents combining global structural approaches and using a xml representation of data. In *Conference on Structural and Syntactical Pattern Recognition (SSPR)*, pages 425–433, 2004.

[12] D. Dori. Syntactic and semantic graphics recognition: The role of the object-process methodology. *Lecture Notes in Computer Science (LNCS)*, 1941:277–287, 2000.

[13] D. Dori. *Object-Process Methodology, a Holistic Systems Paradigm*. Springer Verlag Publisher, 2002. ISBN 3540654712.

[14] P. Grunwald. A tutorial introduction to the minimum description length principle. In *Advances in Minimum Description Length: Theory and Applications*. MIT Press, 2004. ISBN 0-262-07262-9.

[15] T. Henderson and L. Swaminathan. Agent-based engineering drawing analysis. In *Symposium on Document Image Understanding Technology (SDIUT)*, 2003.

[16] P. Héroux, S. Diana, E. Trupin, and Y. Lecourtier. A structural classification for retrospective conversion of documents. In *Conference on Structural and Syntactical Pattern Recognition (SSPR)*, pages 154–162, 2000.

[17] I. Jacobson, M. Ericsson, and A. Jacobson. *The Object Advantage : Business Process Reengineering With Object Technology*. Addison Wesley Publishing, 1994. ISBN 0201422891.

[18] X. Jiang, L. Schiffmann, and H. Bunke. Computation of median shapes. In *Asian Conference on Computer Vision (ACCV)*, pages 300–305, 2000.

[19] R. Kasturi, L. O'Gorman, and V. Govindaraju. Document image analysis : A primer. *Sadhana*, 27(1): 3–22, 2002.

[20] J. Lladós, E. Valveny, G. Sánchez, and E. Martí. Symbol recognition : Current advances and perspectives. In *Workshop on Graphics Recognition (GREC)*, pages 104–127, 2001.

[21] B. Messmer. *Efficient Graph Matching Algorithms for Preprocessed Model Graphs*. PhD thesis, Bern University, Switzerland, 1995.

[22] J. Ramel and N. Vincent. Strategies for line drawing understanding. In *Workshop on Graphics Recognition (GREC)*, pages 13–20, 2003.

[23] C. Rosenberg, J. Ogier, C. Cariou, and K. Chehdi. A statistical solution to evaluate image processing techniques. In *Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, pages 1645–1648, 1998.

[24] Y. Saidali, S. Adam, J. Ogier, E. Trupin, and J. Labiche. Knowledge representation and acquisition for engineering document analysis. *Lecture Notes in Computer Science (LNCS)*, 3088:25–36, 2004.

[25] J. Song, F. Su, C. Tai, and S. Cai. An object-oriented progressive-simplification based vectorisation system for engineering drawings: Model, algorithm and performance. *Pattern Analysis and Machine Intelligence (PAMI)*, 24(8):1048–1060, 2002.

[26] J. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Cole Publishing Co, 1999. ISBN 0-534-94965-7.

[27] P. Stone and M. Veloso. Multi-agent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383, 2000.

[28] M. Thonnat and S. Moisan. Knowledge based system for program supervision. In *Workshop on Knowledge-Based systems for the (re)Use of Program Libraries*, 1995.

[29] E. Valveny and P. Dosch. Symbol recognition contest: A synthesis. *Lecture Notes in Computer Science (LNCS)*, 3088:364–383, 2004.

[30] G. Wagner. How to design a general rule markup language. In *Workshop on XML technologien für das Semantic Web (XSW)*, pages 19–37, 2002.

[31] L. Wenyin and D. Dori. Object-process based graphics recognition class library: Principles and applications. *Software: Practice and Experience (SPE)*, 15(29):1–24, 1999.

[32] M. Worring and al. Automatic indexing of text and graphics in technical manuals. In *Conference on Multimedia and Expo (ICME)*, pages 949– 952, 2001.

6 Annexe A

(*MMI*) Machine Man Interface, (*MSA*) Multi Agent System(s), (*ojgBE*) open java graph Base Editor, (*OOPSV*) Object-Oriented Progressive-Simplification based Vectorization, (*OPD*) Object-Process Diagram, (*OPM*) Object-Process Methodology, (*rsOPM*) rule based system for Object-Process Methodology, (*RuleML*) Rule Markup Language, (*XGMML*) eXtensible Graph Markup Language, (*XMLgml*) XML graphics model learning