Adaptable Vectorisation System Based on Strategic Knowledge and XML Representation Use

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

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

Abstract. This paper presents a vectorisation system based on the use of *strategic knowledge*. This one is composed of two parts: a processing library and a graphic user interface. Our processing library is composed of image pre-processing and vectorisation tools. Our graphic user interface is used for the *strategic knowledge* acquisition and operationalisation. It allows to construct and to execute scenarios, exploiting any processing of our library, according to documents' contexts and users' adopted strategies. A XML data representation is used, allowing an easy data manipulation. A scenario example is presented for graphics recognition on utility maps.

1 Introduction

Vectorisation is used for different purposes like: graphic document recognition, handwriting recognition, and so on. Vectorisation systems use two main types of knowledge: *descriptive knowledge* and *strategic knowledge*. The first one concerns descriptions of documents' objects, and the second one concerns chaining links between image processing tools. The *descriptive knowledge* is usually employed, but not the *strategic knowledge*. This paper presents a vectorisation system based on the use of *strategic knowledge*. We present here our first advancements concerning the experimentation of this system. This one is composed of two parts: a processing library and a graphic user interface. Our processing library is composed of image pre-processing and vectorisation tools. Our pre-processing tools allow to deal with noisy images. Our vectorisation processing tools are of high granularity level in order to use them within a strategic approach. Our graphic user interface is used for *strategic knowledge* acquisition and operationalisation. This graphic user interface allows to construct and to execute scenarios, exploiting any processing of our library. A XML data representation is used in the system, allowing an easy data manipulation. In the paper's follow-up, we present in section (2), an overview on knowledge based vectorisation systems. In section (3) and (4), we present our image processing library and our graphic user interface. In section (5), we present the XML use in the system. In section (6), we present a scenario example for graphics recognition on utility maps. Finally, in section (7), we conclude.

2 Overview on Knowledge Based Vectorisation Systems

Vectorisation is a stage of document interpretation problem that is used for different purposes like: graphic document recognition (technical document [1], map [16] symbol [13], and so on.), handwriting recognition (especially Chinese handwriting [7]), and so on. It is a well-known problem and many commercial applications exist [14]. A vectorisation system can be decomposed into two parts: a processing part (vectorisation) and a system part (the control).

Vectorisation extracts vector data from document images [1]. These vector data correspond to mathematical object graphs composed of: vectors, circles, and curves. Vectorisation is a complex process that may rely on many different methods [5]. Some of them perform vectorisation in two steps [26]. The first one extracts pixel chain graphs. Various approaches may be used [5] like: contouring, skeletonisation, and run decomposition. The second one transforms pixel chains into mathematical object lists [21] (vectors, arcs, curves). Some other methods perform directly vectorisation like: tracking methods, object segmentation methods, and meshes based methods. Several overviews on vectorisation can be found in [1], [5], and [26].

The control system may use various approaches in order to supervise the vectorisation process. These approaches come from pattern recognition and artificial intelligence domains. Among them, let's cite: the knowledge based systems [8], the multi-agent systems [9], blackboard based systems [1], and so on. These systems are generally used for recognition and interpretation, but can have other applications like: learning, indexing, structuring data, and so on. So, systems deal with two main properties: knowledge use [8] and automatic control of processings [9]. The last ones deal currently with the both. Several overviews on vectorisation systems can be found in [1], [16], and [25].

Like document interpretation systems, vectorisation systems use two main types of knowledge: *descriptive knowledge* and *strategic knowledge* [22]. The first one concerns documents' objects and links between them. The second one concerns image processing tools that are used to construct documents' objects and chaining links between these tools. The descriptive knowledge is usually employed in vectorisation systems [1]. At the opposite, the strategic knowledge is less used, whereas many common steps exist among vectorisation process [5]. In the paper's follow-up, we propose a vectorisation system based on strategic knowledge use. It is composed of a processing library and a graphic user interface for scenarios construction and their executions.

3 Image Processing Library

3.1 Introduction

All our image processings are included in the PSI[1] Image Processing Library, which is composed of image pre-processings and vectorisation processings. This library is included in a complete document image recognition library, the PSI Library. A description of some library's processings can be found in [4]. Image pre-processings can be used on grey-level and binary images. We develop its content in section (3.2). Vectorisation processings are based on various approaches (skeletonisation, contouring, region and run decomposition, direct vectorisation). We have decomposed the classical vectorisation chain into granular processings (Table 1) in order to use them within a strategic approach. Our vectorisation processings are decomposed according to three data levels: an image level, a structured data level, and a boundary level between the image data and the structured data levels. We present each level in sections (3.3), (3.4), and (3.5).

Table 1. data levels of vectorisation processings

Image	Skeletonisation Adaptation, Contouring, Skeletonisation
Boundary	Progressive Object Simplification, Pixel List Extraction, Direct Vectorisation, Direct Contouring, Run Decomposition, Region Decomposition
Structured Data	Interiority Degree Segmentation, Junction Reconstruction, Graph Construction, Polygonisation, Curve and circle fitting, Pruning, Merging, Smoothing, List Filtering

3.2 Image Pre-processing

The image pre-processing level is composed of different methods for the processing of noisy images. Firstly, we use grey-level filtering methods on scanned images like median filter and mean filter [18]. Next, we binarise our images. We use two standard algorithms for the automatic computation of binarisation thresholds: the Otsu's method [17] and the Kittler's method [10]. The first one is a histogram based method and the second one a clustering based method. These two methods have been used for map pre-processing [11], in order to segment the network part from the cadastral part, according to parts' grey-levels. Then, we can use two methods families for noise reduction on obtained binary images. The first ones are connected components and loops filtering methods, based on a "classical" blob coloring algorithm. They use automatic or pre-defined user surface threshold, the automatic threshold computation is based on maximum proportional ratio search of connected components' surfaces. The second ones are mathematical morphology operations like dilatation, erosion, opening, and closing. These operations use classical masks 3*3 sized and of "+" "×" type.

[1] Perception Systems Information Laboratory: http://www.univ-rouen.fr/psi/

3.3 Vectorisation's Image Level

The vectorisation's image level uses classical image processings for the contouring and the skeletonisation. For the skeletonisation, we use two standard algorithms: Di Baja [6] and Taconet [24]. These skeletonisation algorithms are based on the medial axis transform [1]. For the contouring, we use a classical neighbouring test method with classical masks 3*3 sized and of "+" "×" type. These contours are then processed in order to obtain 2-connected contour lists. The Fig. 1 (a) illustrates overlapped results of skeletonisation and contouring. The main problem of the skeletonisation is the noise which depends on lines' thickness of images. In order to adapt images for the skeletonisation, we use the 3-4 distance transform of Di Baja [6] in combination with a classical image reducing tool. The Fig. 1 (b) gives an example of the skeletonisation's impact without any reduction (middle) and with a reduction (right).

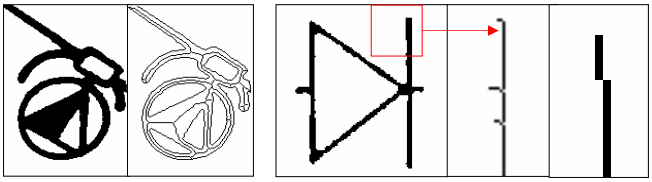


Fig. 1. (a) skeletonisation and contouring (b) skeletonisation adaptation

3.4 Vectorisation's Boundary Level

Our vectorisation's boundary level uses different methods in order to extract structured data from images. We present each method in the six next paragraphs.

We use a contouring method based on a classical blob coloring algorithm. During the connected components labelling, the internal and external shapes' contours are extracted. Then, the contours' pixels are structured into chains. In a following step, the inclusion links between connected components are searched. With this application, we extract inclusion graphs which contain the external and the internal contour chains, and the inclusion links between the connected components. This method gives global/local descriptions of image's shapes [5]. The Fig. 2 (a) gives an example of raster representation of extracted external contours.

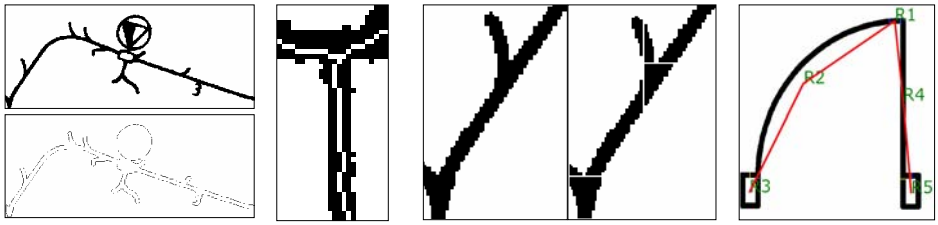


Fig. 2. (a) contouring (b) direct vectorisation (c) run (d) region

We use a direct vectorisation method. This method has been used for roads extraction on cadastral maps [15]. Firstly, image is analyzed to find an entry point. The following tracking process is two types: line tracking and junction tracking [5]. The line tracking uses a point element, which advances into line's middle according to contours follow-up. The displacement's length is proportional to line's thickness. The junction tracking analyses connection breaks of contours follow-up. In this case, a circle including the junction is used to find the other junction's starting lines. The crossings between the lines and the circle are searched. From these data, a image's graph is updated and used to start the tracking of new lines. The Fig. 2 (b) gives an example of raster representation of line and junction extraction.

We use a run decomposition method. This method is under construction. In a first step the image is encoded into runs. Then, runs are structured into run graphs [3]. In these graphs, the successive runs are represented by run chains and the 3-connected runs (or more) by junctions. The encoded runs can be horizontal and/or vertical. From these run graphs, contours and skeleton can be extracted. The Fig. 2 (c) gives an example of mixed raster representation of a horizontal and vertical run graph (the junction nodes are represented in white).

We use region decomposition method based on a wave aggregation. Firstly, the image is analyzed to find an entry point. Then, the method searches the neighbouring points. These points are labelled, aggregated, and stored into a wave object. Successively, the previous waves are used for new aggregation processes. The wave breaking and stopping cases define the regions' boundaries. The boundaries are next used in order to create entry waves for the new region search. From the labelled region map, a primary region graph is created. In a following step, this primary region graph is analysed in order to construct the line and junction regions. From these region graphs, contours and skeletons can be extracted. Also, graphs' regions can be processed by statistical recognition methods [2]. The Fig. 2 (d) gives an example of constructed primary region graph from a labelled region map.

We use a pixel list extraction method to convert skeleton and contour images into structured data. This method is based on the connected pixel destruction. Firstly, all 3-connected pixels are destroyed. After, we chain connected pixels into pixel list. Each list is composed of 1-connected pixel (extremity) and 2-connected pixel. The Fig. 3 (a) gives an example of extracted pixel lists.

We use a last object simplification method. This one allows to exploit structured data on images. We use this approach to simplify processed images during vectorisation scenarios [23]. Currently, this tool can process only circle objects. The Fig. 3 (b) gives an example of circle erasing result.

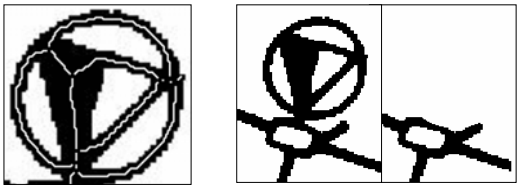


Fig. 3. (a) pixel list extraction (b) object simplification

3.5 Vectorisation's Structured Data Level

The structured data level is the central part of vectorisation scenario. Indeed, all the processings use the same input/output data format. Our data format represents a graph base of geometrical objects like: circles, curves, polylines, pixel lists, and junctions. In practice, we can call any processing in any order, but in theory, some processings depend on a minimum structuring level of data (for example the circle fitting depends on polygonisation results). All boundary level's processings (see section 3.4) export their data in this format. The goal of structured data level's processings is to add semantic information to basic lists obtained by the boundary level's processings. For that, we use different granular processings (see Table 1). We can decompose these processings in two types: lists processing and graphs processing. We present these two types in the two following sections.

3.5.1 Lists Processing

The lists processings are used for different purposes like the interiority degree segmentation and the mathematical approximation (vectors, circles, and curves).

For the interiority degree segmentation, we apply a thickness segmentation threshold based on a simple test of thickness' variation. Information about pixels' interiority degrees is obtained by the successive calls of skeletonisation/pixel list extraction tools. The Fig. 4 (a) gives an example of interiority degree segmentation, with the original image (high), the graphic representation of pixel lists before segmentation (middle) and after segmentation (low).

For the polygonisation (transformation of pixel lists into vector lists "polylines"), we use the "standard" Ramer's method [20], with the "standard" split & merge Pavlidis' method [19]. For each polyline, we compute some attributes, like the vectors' thickness (from original pixel lists), vectors' lengths, and angular links between two consecutive vectors according to the polyline tracing. Our circle fitting algorithm is a basic tool only based on the test of angular and length links inside a polyline. We use here a standard geometrical property: a circle can be approximated by a regular polygon (in length and angle). The Fig. 4 (b left) gives an example of polygonisation and circles fittings applied on a contours image. Recently, we have extended the geometrical object extraction with the standard Bernstein's curve approximation [12] of pixel lists. The Fig. 4 (b right) gives an example of Bernstein's curve approximation.

Furthermore, we use some post-correction processings on lists. In one hand we use a list filtering processing. The Fig. 4 (c left-middle) gives an example of list filtering. The used threshold can be automatically computed, or pre-defined by the user. The automatic computation is based on maximum proportional ratio search of lists' lengths. This processing decreases the complexity of the junction detection method (section 3.5.2). On the other hand we use a list smoothing algorithm. The Fig. 4 (c middle-right) gives an example of list smoothing. It is based on variations analysis of Freeman directions into pixel lists. This processing increases data qualities for the polygonisation and curve fitting.

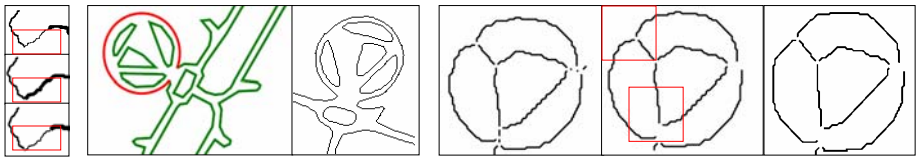
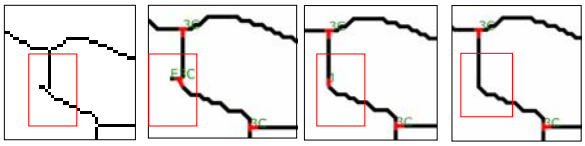


Fig. 4. (a) interiority segmentation (b) mathematical approximation (c) list correction

3.5.2 Graphs Processing

An important processing of structured data level is the junction detection. We base our approach on a junction reconstruction algorithm. This algorithm constructs all the connections between the objects' extremities according to a distance threshold. During this construction, the algorithm forbids the connections between extremities of same objects. The distance threshold can be defined by the user or automatically computed. This automatic computation is based on maximum proportional ratio search of connections' lengths. Next, the algorithm analyzes all the connections to find the inter-connections (group of joined extremities). Each inter-connection constitutes a junction. With information about junctions, we use a graph construction algorithm in order to obtain the geometrical object graphs composed of pixels, circles, curves, and polylines. We exploit information about junctions for two standard processings in the vectorisation process: pruning and merging [1]. The Fig. 5 gives a reconstruction example (a, b) with the successive pruning (b, c) and merging (c, d).



(a) (b) (c) (d)

Fig. 5. (a-b) junction reconstruction (b-c) pruning (c-d) merging

4 Vectorisation Scenarios Construction

We use a graphic user interface for strategic knowledge acquisition and operationalisation: ACTI_VA[2] [22]. It enables to construct scenarios for document image recognition according to documents' contexts (recognition goal, document type, and so on.), and users' adopted strategies. In a first step, the user defines the context of the analyzed image (quality, type, and so on.) (Fig. 6 (a)). The PSI library's processings (see section 3) are then proposed to the user. Each processing is realized from its dashboard "or processing panel" representing a scenario stage. The user

[2] in French: "Acquisition de Connaissances Traiteur d'Images pour leur VAlorisation"

oversees the scenario's construction, following permanently the results' evolutions. So, a state viewer shows the data's intermediate graphic representations (image and/or structured data results) between each scenario's stage (Fig. 6 (b)). For an adaptable and evolving assistance, the user can at any time return to any previous stage in order to, modify the parameters setting, change the processing stage, seek a textual help, or display use examples. When the user reaches his purpose, he saves its scenario in a scenario base. During the scenario's construction, the user can search similar scenario examples in the scenario base with two research tools: a query language allowing to request the scenario base, and a graph-matching tool allowing to compare the scenarios' structures.

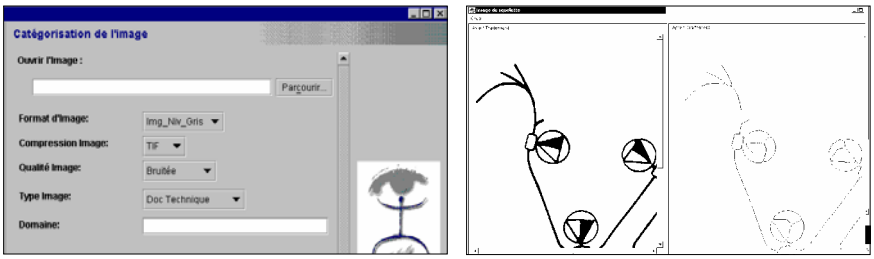


Fig. 6. (a) document's context acquisition (b) state viewer

5 XML use in System

Our two system's parts (processing library and graphic user interface) use the XML language for the data representation. The use of this data representation language offers several possibilities and as much as enhancement for our system. XML is a meta-language because it is defined as a root language, what enables to define specialized sub-languages. We use SVG[3] in the system for our data's graphic representations. We also use XGMML[4] for the graphs' description provided by some PSI Library's processings. XML allows to use transforming processors. These processors easily transform the XML data streams with the help of XSLT[5] scripts. This enables easy data communications between the processings, and between processings and our graphic user interface. XML enables to request XML file bases with XML-QL[6]. We use the candidate languages for XML-QL: Quilt[7]. We use Quilt for management of scenarios bases and reconstruction of final XML documents (see section 6.5). The use of Quilt with XSLT enables to transform scenarios into graphs base for the structural classifiers of the PSI Library [4] (see section 4).

---

[3] Scalable Vector Graphics
[4] eXtensible Graph Markup and Modelling Language
[5] eXtensible Stylesheet Transform Language
[6] XML Query Language
[7] http://www.almaden.ibm.com/cs/people/chamberlin/quilt.html

6 Scenario Example

6.1 Introduction

We present here our first advancements concerning the experimentation of this system, through a scenario example on FT's utility maps[8]. We have two main components on these utility maps (Fig. 7 (a)): the characters, and the graphic parts. We present here the graphic parts recognition. We suggest the reader to consult [2] for the description of the character recognition. Also, we have two main components on these graphic parts (Fig. 7 (a)): the symbols and the network. The symbols represent technical equipments allowing connections on the network. We have three symbols' classes (Fig. 7 (a)): room, CP1[9] and CP2. Our graphics recognition scenario may be decomposed into four main steps: noise pre-processing and characters segmentation, network's contours vectorisation, characters detection, and XML object reconstruction. We present each scenario's step in the four next sections.

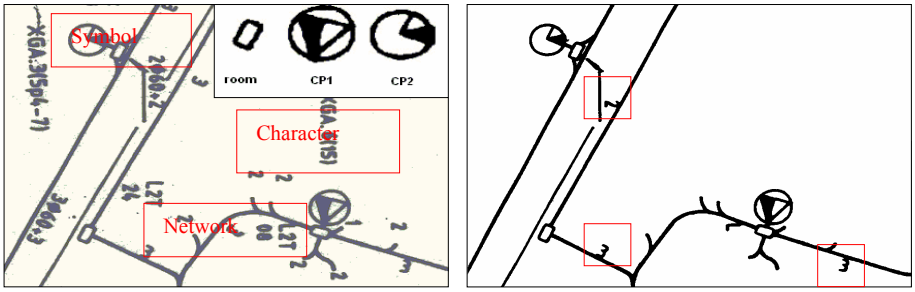


Fig. 7. (a) FT's utility map (b) clean map

6.2 Noise Pre-Processing and Characters Segmentation

In a first step, we use a pre-processing scenario (see section (3.2)) on scanned images in order to reduce the result noise of the acquisition stage, and to segment the characters and the graphic parts. We have applied successively a median filtering, an Otsu's binarisation, an automatic connected component filtering, and an opening. Thus, we have obtained clean network images (Fig. 7 (b)). However, this pre-processing step cannot deal with the connected characters to the network (Fig. 7 (b)'s rectangles).

[8] French Telecommunication operator: http://www.rd.francetelecom.fr/
[9] Concentration Point 1

6.3 Network's Contours Vectorisation

Next, we use a vectorisation scenario in order to extract vector data corresponding to graphic parts (see sections (3.3), (3.4), and (3.5)). Successively, we have applied a contouring, a pixel list extraction, a smoothing, a polygonisation, a split and merge method, and a circle fitting. The Fig. 8 (a, and b) gives an example of graphic parts' vectorisation. Following the vectorisation, a vector data reconstruction tool is used. Firstly, this one links internal contours with circles, according to their inclusion links. Indeed, the symbols' internal contour numbers define their classes: 4 for CP1, 2 for CP2, and 1 for room. These numbers are next used during the XML object reconstruction step to give the symbols' labels (see section 6.5). Secondly, the vector data reconstruction tool matches contours in order to reconstruct network parts' vectors. For example, on the Fig. 8 (a) we have three network's parts. Thus in the data's graphic representation, the internal contours and the circle correspond to symbols (Fig. 8 (b)), and the other vectors correspond to the different network's parts (Fig. 8 (a)).

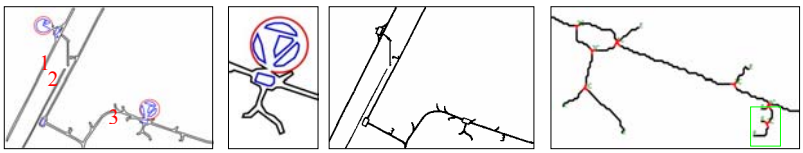


Fig. 8. (a) network's vectorisation (b) symbol's vectorisation
(c) circle simplification (d) interest zone research

6.4 Connected Characters Detection

In a third step, we use another vectorisation scenario in order to deal with the connected characters detection. Firstly, we use vector data of the last step in order to reduce complexity of processed images (Fig. 7 (b)). We apply a circle object simplification (see section 3.4) with an automatic connected component filtering (see section (3.2)). The automatic connected component filtering erases the possible small connected components obtained after the circle simplification. The Fig. 8 (c) gives an example of result after circle simplification. Next, we use a skeleton graph in order to research the zones of interest for connected characters. We have applied successively a skeletonisation adaptation, a skeletonisation, a pixel list extraction, a junction detection, a graph construction, and a pruning and a merging. Finally, we use a graph tool [4] in order to search the graphs' parts corresponding to zones of interest. We construct graphs with length information of pixel lists (two node labels: *short* and *long*), and search into the image's graphs groups of connected nodes, which are labelled *short*. The Fig. 8 (d) gives an example of result, with the detection of "3" connected character. Thereafter, these zones of interest can be exploited by statistical filtering methods [2].

6.5 XML Object Reconstruction

The two last steps give recognition results into XML format, stored into different XML files. These XML data are weakly structured: there are no links between them. In order to solve this problem, we use an XML object reconstruction step. This step organizes the different XML streams of the two last steps, in order to reconstruct the document's objects. This reconstruction allows to structure the symbols and connected characters with their network's parts. During this step, the internal contours numbers are used to give the symbols' labels. We use for that a reconstruction scenario based on XSLT and Quilt (see section 5).

7 Conclusion

In this paper we have presented a vectorisation system based on strategic knowledge use. This system is under experimentation, but allows first uses. Our processing library is composed of image pre-processing and vectorisation tools. Our pre-processing tools allow to deal with noisy images. Our vectorisation tools are of high granularity level in order to use them within a strategic approach. In both case, several different processings can be used for a same recognition goal. The graphic user interface is used to construct and execute vectorisation scenarios. Scenarios are stored in a database and then can be replayed partially or totally. In both system's parts, a XML data representation is used allowing an easy data manipulation. So, this system allows to construct several vectorisation scenarios, to test different strategies according to the recognition goals, and can be easily adapted to new applications.

For the perspectives, currently the system does not analyze the consistency of scenario's results. For that, we plan to use RuleML[10], which will allow to use domain knowledge [22], stored in rules base, and depending on processed document's type. Next we will extend our graphic user interface for the scenario construction of symbol recognition [4]. Finally, we will plan to use RDF[11] in order to structure the knowledge provided scenarios' results with the strategic knowledge provided by the graphic user interface.

References

1. S. Ablameyko, T.P. Pridmore. Machine Interpretation of Line Drawing Images. Springer-Verlag, 2000.
2. S. Adam, J.M. Ogier, C. Cariou, J. Gardes, Y. Lecourtier. Combination of Invariant Pattern Recognition Primitive on Technical Documents. Graphics Recognition (GREC), 1999.
3. M. Burge, W.G. Kropatsh. A Minimal Line Property Preserving Representation of Line Images. Structural and Syntactical Pattern Recognition (SSPR), 1998.

[10] Rule Markup Language
[11] Resource Description Framework

4. M. Delalandre, S. Nicolas, E. Trupin, J.M. Ogier. Symbols Recognition by Global-Local Structural Approaches, Based on the Scenarios Use, and with a XML Representation of Data. International Conference on Document Analysis And Recognition (ICDAR), 2003.
5. M. Delalandre, E. Trupin, J.M, Ogier. Local Structural Analysis: A Primer. Graphics Recognition (GREC), 2003.
6. G.B Di Baja. Well shaped, Stable, and Reversible Skeletons from the 3-4 Distance Transform. Journal of Visual Communication and Image Representation, 5(1) : 107-115, 1992.
7. J. Fan. Off-line Optical Character Recognition for Printed Chinese Character-A Survey. Technical Report, University of Colombia, USA, 2002.
8. J.E. Den Hartog. Knowledge Based Interpretation of Utility Maps. Computer Vision and Image Understanding (CVIU), 63(1) : 105-117, 1996.
9. T.C. Henderson, L. Swaminathan. Agent Based Engineering Drawing Analysis. Symposium on Document Image Understanding Technology (SDIUT), 2003.
10. J. Kittler, J. Illingworth. Minimum Error Thresholding. Pattern Recognition (PR), 19(1) : 41-47, 1986.
11. A. Lassaulzais, R. Mullot, J. Gardes, Y. Lecourtier. Segmentation d'Infrastructures de Réseau Téléphonique. Colloque International Francophone sur l'Ecrit et le Document (CIFED), 1998.
12. C. W Liao, J. S. Huang. Stroke Segmentation by Bernstein-Bezier Curve Fitting. Pattern Recognition (PR), 23(5) : 475-484, 1990.
13. J. Lladós, E. Valveny, G. Sánchez, E. Martí. Symbol Recognition : Current Advances an Perspectives. Graphics Recognition (GREC), 2001.
14. E.F El-Mejbri, H. Grabowski, H. Kunze, R.S. Lossack, A. Michelis. A Contribution to the Reconstruction Process of Article Based Assembly Drawings. Graphics Recognition (GREC), 2001.
15. J.M. Ogier, C. Olivier, Y. Lecourtier. Extraction of Roads from Digitized Maps. European Signal Processing Conference (EUSIPCO), 1992.
16. J.M. Ogier, S. Adam, A. Bessaid, H. Bechar. Automatic Topographic Color Map Analysis. System.Graphics Recognition (GREC), 2001.
17. N.Otsu. A Threshold Selection Method from Gray-Level Histograms. Transactions on Systems, Man and Cybernetics (TSMC), 9(1) : 62-66, 1979.
18. J.R. Parker. Algorithms for Image Processing and Computer Vision. Paperback editions, 1996.
19. T. Pavlidis, S. L. Horowitz. Segmentation of Plane Curves. Transactions on Computers (TC), 23 : 860-870, 1974.
20. V. Ramer. An Iterative Procedure for the Polygonal Approximation of Plane Curves. Computer Vision Graphics and Image Processing, 1(3) : 244-246, 1972.
21. P.L. Rosin, A.W. West. Nonparametric Segmentation of Curves Into Various Representations. Pattern Analysis and Machine Intelligence (PAMI), 17(12) : 1140-1153, 1995.
22. Y. Saidali, S. Adam, J.M. Ogier, E. Trupin, J. Labiche. Knowledge Representation and Acquisition for Engineering Document Analysis. Graphics Recognition (GREC), 2003.
23. J. Song, F. Su, C. Tai, 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.
24. B. Taconet, A. Zahour, S. Zhang, A. Faure. Deux Algorithmes de Squelettisation. Reconnaissance Automatique de l'Ecriture (RAE), 1990.
25. K. Tombre. Ten Years of Research in the Analysis of Graphics Documents, Achievements and Open Problems. Image Processing and Image Understanding, 1998.
26. K. Tombre, C. Ah-Soon, P. Dosch, G. Masini, S.Tabbone. Stable and Robust Vectorisation : How to Make the Right Choices. Graphics Recognition (GREC), 1999.