A Platform to Extract Knowledge from Graphic Documents. Application to an Architectural Sketch Understanding Scenario[⋆]

Gemma Sánchez, Ernest Valveny, Josep Lladós, Joan Mas, and Narcís Lozano

Computer Vision Center, Dept. Informàtica,
Universitat Autònoma de Barcelona,
08193 Bellaterra (Barcelona), Spain
gemma@cvc.uab.es
http://www.cvc.uab.es

Abstract. This paper proposes a general architecture to extract knowledge from graphic documents. The architecture consists of three major components. First, a set of modules able to extract descriptors that, combined with domain-dependent knowledge and recognition strategies, allow to interpret a given graphical document. Second, a representation model based on a graph structure that allows to hierarchically represent the information of the document at different abstraction levels. Finally, the third component implements a calligraphic interface that allows the feedback between the user and the system. The second part of the paper describes an application scenario of the above platform. The scenario is a system for the interpretation of sketches of architectural plans. This is a tool to convert sketches to a CAD representation or to edit a given plan by a sketchy interface. The application scenario combines different symbol recognition algorithms stated in terms of document descriptors to extract building elements such as doors, windows, walls and furniture.

1 Introduction

Graphics Recognition is the subfield of Document Analysis that is concerned on the interpretation of graphical structures present in documents like plans, maps, engineering drawings, musical scores, charts, etc. A lot of research has addressed the problem of converting paper-based graphics documents to electronic formats. Classically, this conversion involves activities organized in three levels: feature and low level primitive extraction, primitive (symbol) recognition, and document understanding using domain knowledge. Techniques from the fields of Image Processing, Pattern Recognition and Artificial Intelligence formulate the above activities.

A number of high performant Graphics Recognition systems exist. Not only in academic domains but also as industrial applications. However, most of such systems are highly domain-dependent. The development of general graphics recognition platforms remains still a challenge. Some interesting approaches exist that are focused on that

[⋆] This work has been partially supported by the spanish project CICYT TIC2003-09291 and catalan project CeRTAP PVPC.

goal [1–3]. In the existing applications the methodological basis is mainly the same. There is no reason to reinvent the wheel for each application, it is better to combine and parametrize basic tools in terms of a domain-dependent strategy. On the other hand, the concept of *document* is becoming more and more extensive. The classical idea of paper document is currently complemented by electronic documents (pdf, html, etc.). Hence, the idea of converting paper-based documents to electronic format is now extended to the idea of understanding poorly structured documents from different sources. This results in new application scenarios as document browsing, indexing, editing by means of on-line sketch-based interfaces, etc. However, these new scenarios continue using the same core of techniques.

The above reasons justify the need for a general platform devoted to graphic document analysis and recognition. This is the goal of the architecture proposed in this work. Inspired by the idea of the french DocMining project [1], we propose an architecture for document mining, i.e. a set of engines to extract different kinds of descriptors from documents. The combination of such descriptor extractors with the domain-dependent knowledge defines an application scenario.

On the other hand, the role of the user in the graphics recognition cycle is also a key issue. The user intervention in a graphics recognition process should not be seen as negative but a natural issue. A recognition and understanding process may need the feedback from the user to set particular parameters, to validate decisions taken from the system when there is uncertainty or just to interact with the system to edit the document or drive the process. The sketchy interfaces paradigm is very relevant to the last issue, i.e. the user interaction by means of pen strokes is a powerful tool to draw new graphic documents, to digitally augment paper documents or to edit documents by sketchy gestures. Our idea in the system architecture proposed in this paper is to include a sketchy interface as a component of a graphics recognition system.

Taking into account the above considerations, in this paper we propose a general architecture for knowledge extraction from graphic documents. This architecture follows a document mining paradigm and can be summarized in three major issues. First, it consists of a set of engines to extract knowledge sources or descriptors. A descriptor is a feature that afterwards is likely to have a particular meaning in the domain where the document belongs to. Thus, a feature can be a segment after a vectorization process, a shape, a colour or texture based segmented region, a perceptually salient pattern, a text component, etc. The particular interpretation of a given document depends on the combination of descriptors in terms of domain-dependent rules. The second issue of our architecture is the definition of a hierarchical metadata model to represent any graphic document. It consists in an abstract model that, using the set of descriptor extraction engines, allows to convert a graphic document into a *normalized document*. Finally, the third basis of our architecture is a tool for the integration of the user in the document understanding cycle. As stated before, this tool is based on the sketchy interfacing paradigm and, hence allows to either design new diagrams or edit existing ones by means of gestures.

The above architecture makes sense when it can be *instantiated* for a given scenario, i.e. for a particular application. To illustrate it, the second part of this paper presents an architectural plan analysis system. Architectural sketches have been the focus of several works on the recognition of sketched-based interfaces [4, 5]. These systems are specifically designed to analyze and recognize architectural drawings and are based on the application of specific rules about the domain. Our system aims to follow a general framework which could be used in other applications. The scenario presented in this paper is though as a tool to assist an architect in the early design phase of a new project. Hence, in this stage, an architect uses to convert ideas to sketches. Actually, the system presented in this paper is part of a more general platform that aims to get a 3D view of a building from a sketched floor plan. Then the architect can navigate into the building using a Virtual Reality environment. Our proposed system combines the following elements: first, a set of descriptor extraction modules and domain-dependent knowledge to recognize building elements (graphic symbols) as walls, doors, windows, etc. Second, a graph-based structure to represent the documents. And third, a sketch-based interface paradigm to draw architectural floor plans or to interactively edit existing ones by adding new elements. Our goal is not to describe a novel symbol recognition algorithm but to propose a system in which classical graphics recognition techniques and strategies are combined in a particular scenario consisting of interpreting architectural sketches.

The remainder of this paper is organized in two parts. First, in section 2 we describe the general system architecture. The second part of the paper describes the particular use of the architecture elements in the scenario of sketched architectural plans interpretation. Thus, section 3 presents the feature descriptors, and section 4 describes how such descriptors are used in terms of domain-dependent knowledge to recognize building elements. Section 5 discusses the experimental evaluation and finally, section 6 is devoted to conclusions.

2 System Architecture and Application Scenario

The system presented in this paper is a particular scenario of the general platform introduced in Section 1. This general platform can be organized in three layers: the first layer is devoted to acquisition of documents, whatever the acquisition mode: on-line or off-line. The second layer has to do with extraction of relevant features. Finally, the third layer consists of knowledge interpretation, using the extracted features. These three layers are joined using a common data structure that allows to represent, store, search and modify information at several levels of abstraction. The description of that model is presented in Fig. 1.

The data model consists of a generic labelled non-directed graph able to represent the information at different abstraction levels by changing the kind of nodes and edges forming it. Both nodes and edges can be described by Graphic Objects. A Graphic Object is a generic class with some derived classes: Symbol, Point, Line, Arc, Region and Stroke. The Symbol class can be described by another graph, or by a Pattern class. The Pattern class is a generic class describing a shape that can be specialized into a graph, a grammar that describes the pattern by means of grammatical rules or a feature vector. The Stroke class is formed by a set of points.

The second layer extracts the descriptors explained in Section 3, which are divided in four types: Vector-based descriptors: mainly lines, arcs and their relationships. Perceptual grouping descriptors as parallelism, collinearity, closed loops and overlapping.

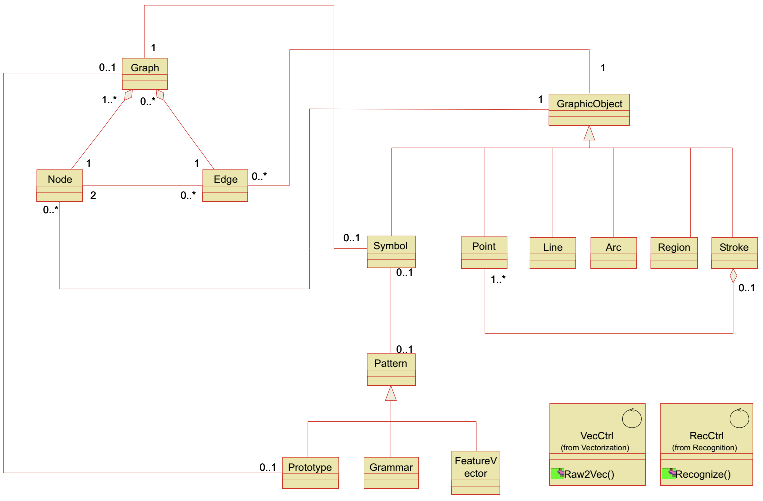

Fig. 1. Graph model to represent the normalized document.

These two descriptors need a vectorization [6] pre-process to obtain the segments and arcs from the input document. Pixel-based descriptors, as Zernike moments and zoning. Dynamic descriptors consisting of the chain of strokes drawn by a user together with its temporal information and speed.

As it has been explained, the particular scenario of application of this general framework is the early design of sketches in floor architectural plans. Sketches can be drawn on-line or off-line. The aim of the platform is to allow the architects to design a plan in a natural way, recognizing its parts and storing it as a normalized document. The sketching interface paradigm allows the architect either to draw a new plan, to edit an existing one or to interact with the system by a sketchy gestures language. From the sketch the system is able to create a structured document consisting of its building elements. Such building elements are described in terms of domain-dependent knowledge that mainly consists of prototype patterns and semantic rules, using the normalized data structure described before. The specialization of the general architecture to this scenario is graphically outlined in Fig. 2.

In the first layer, the document can be obtained off-line by scanning a hand-drawn sketch made on a ”classical way” on a paper, or on-line with a digital tablet, a tablet PC or a digital pen. Both ways cooperate in the creation process of the document. A first version of a sketch can be created in both off-line and on-line way. Then the system can recognize its parts and to store it in a data base to later recover and modify it in the on-line way. When the document is edited with a sketchy interface, the recognition process acts on the new input and not on all the already processed document.

Once the document is obtained its structure is computed in the second layer of the architecture. The structure of this kind of documents is formed by the structural parts of the building: walls, windows and doors, the furniture: tables, sofas, chairs, etc. and the facilities as plugs, TV-connections, pipes, etc. Each kind of component can be identified by means of a set of characteristics that in some cases are common to more than one kind of symbol.

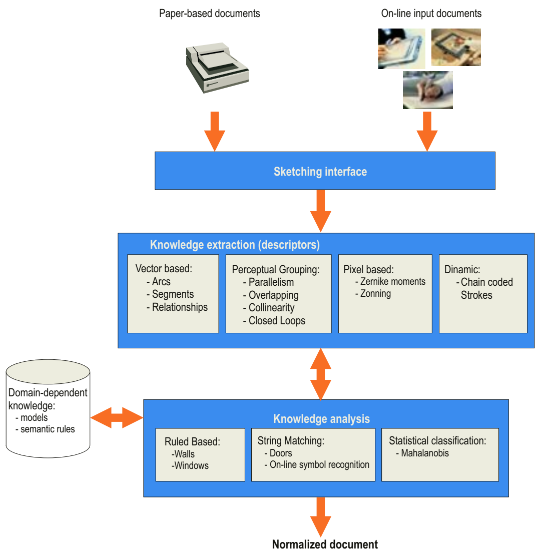
Paper-based documents
On-line input documents
Sketching interface
Knowledge extraction (descriptors)
Vector based: - Arcs - Segments - Relationships
Perceptual Grouping: - Parallelism - Overlapping - Collinearity - Closed Loops
Pixel based: - Zernike moments - Zonning
Dinamic: - Chain coded Strokes
Domain-dependent knowledge: - models - semantic rules
Knowledge analysis
Ruled Based: -Walls -Windows
String Matching: - Doors - On-line symbol recognition
Statistical classification: - Mahalanobis
Normalized document

Fig. 2. The particular architecture for an architectural plan analysis platform.

Finally the third layer extracts the knowledge from the descriptors and save it in the normalized format. Three kind of recognition approaches are used. Rule-based recognition uses the information of parallelism to recognize windows and walls. String matching recognizes doors and some furniture symbols from the vectorial and dynamic information. Statistical classification recognizes some furniture symbols from pixel-based descriptors. All these approaches are explained in more detail in Section 4.

The general data model described previously is used to get a description of the plan at several levels of abstraction in the following way. The first level of abstraction is represented by a graph describing the layout of the sketch. The edges represent segments and arcs appearing on the document and the nodes represent the points connecting them. In a second level of abstraction the nodes in the graph can represent the closed regions in the document, and the edges their relationships. At a higher abstraction level the nodes represent rooms with its associated class of room: a kitchen, a bathroom, etc, and the edges are the relations among them.

3 Feature Extraction Engines

An architectural drawing is composed of different types of elements: walls, doors, windows, stairs, symbols describing furniture elements, etc. There is not a single representation scheme nor recognition method able to describe and identify all of them. Then, according to the general system architecture described in section 2, several descriptors are extracted from the drawing, in order to get an optimal representation of every kind of element. These descriptors are combined to get a global representation of the drawing, using the common data structure explained in section 2. We have grouped those descriptors into four categories: vector-based descriptors, perceptual grouping descriptors, pixel-based descriptors and dynamic descriptors.

3.1 Vector-Based Descriptors

Vectors always play an important role in the description of technical drawings. There are several approaches to vectorization [6]. In this case, we apply a vectorization process based on thinning and polygonal approximation of the image skeleton. The result of vectorization is a set of vectors (lines or arcs) and their relationships. They are organized using the common graph structure described in section 2.

Vectorial representation is used for the description and recognition of those elements describing the structure of the drawing, such as walls, doors and windows. It is also the basis for obtaining perceptual grouping descriptors.

3.2 Perceptual Grouping Descriptors

Segmentation is a very important task in any graphics recognition system. The elements of the drawing must be located before being recognized. Usually, segmentation follows specific rules for each kind of drawing as the type of elements and their relationship are different from one kind of drawing to another. Sometimes, segmentation is carried out along with recognition, locating and recognizing the elements of the drawing at the same time.

Another approach is to detect possible locations of elements in the drawing, and then, verify the existence of the element in that position with some recognition method. The detection of such possible locations are usually based on perceptual grouping techniques where salient features are detected. These features are necessary, but not sufficient, to determine the existence of a given element. Therefore, they can be used as seeds to search for one specific element.

We have used this last approach and therefore, we have defined a set of descriptors which will be employed later in segmentation tasks, such as the detection of walls, windows and symbols. These descriptors are based on the vector-based descriptors obtained as the result of vectorization:

- **Parallelism:** Detected when the difference in orientation between two nearby lines is near 0 or 180, given some preselected threshold.
- **Overlapping:** The projection, either in the horizontal or vertical axis, of two lines must overlap, and the between the length of overlapping and the shortest line must be larger than a given threshold.
- **Collinearity:** this condition is detected when a parallelism between two non overlapping lines is found, and the difference in orientation between the line joining the farther end points and both lines is near 0 or 180.
- **Closed loops:** they are detected trough the analysis of the vector graph. The loops are represented as a graph of adjacent regions, using the common data structure.

3.3 Pixel-Based Descriptors

Some of the symbols which appear in architectural drawings are quite small. As we are working with hand-drawn symbols, they cannot be represented by vectors, as the variability of hand-drawing introduces too much distortion in the vectorization results.

Therefore, these symbols must be described using several descriptors computed from the binary image and not their vector representation. We have used two different kinds of pixel-based descriptors: circular zoning and *Zernike* moments. Both methods rely on a previous segmentation of the symbol, which is carried out by analysis of connected components in the drawing.

Circular zoning is based on a method developed by Adam [7], and it allows to get a rotation and scale invariant feature vector. Once the symbol is segmented, the image is divided into concentric zones as illustrated in figure 3. For each zone, the number of black pixels is counted and normalized according to the area of the zone. Thus, the feature vector contains one value for each zone, representing the ratio of black pixels in it.

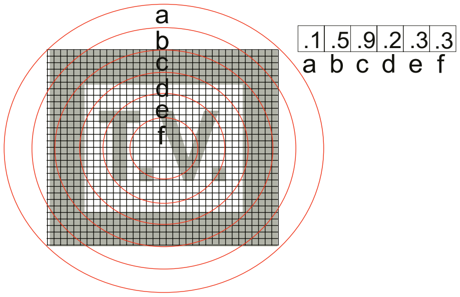

Fig. 3. Representation of a symbol using circular zoning.

Circular zoning is not able to distinguish between all symbols, taking into account the large amount of variability in hand-drawn symbols. Therefore, we have introduced another set of invariant features based on Zernike moments in order to improve recognition accuracy. Zernike moments [8] have been successfully used in the context of OCR. They can be related to usual geometric moments, but they allow to reduce certain degree of redundancy in the information conveyed by geometric moments. Moreover, they can be easily used to define a rotation invariant feature vector. They are based on the projection over the Zernike polynomials of the image mapped on the range of the unit circle. We have taken Zernike moments from order 2 up to 6 (fourteen values), to build the feature vector.

3.4 Dynamic Descriptors

As we have explained, the system can take both off-line and on-line input. For on-line input, we can take advantage of the time information to improve recognition accuracy. With that aim, we have also defined a set of dynamic descriptors, obtained from the on-line input, organized at two levels of abstraction.

The first level of descriptors is simply the sequence of points at each moment of time. This sequence of points is described with the usual *chaincode* representation. The second level of descriptors aims to extract the structure of the on-line sequence. This way, each stroke (sequence of points) is divided into segments, each segment corresponding to a straight line or an arc in the drawing. This segmentation is achieved by locating the breaking points where a corner can be found, looking for changes in local and global stroke curvature - see figure 4 -.

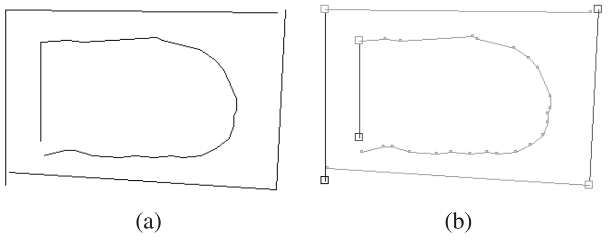

Fig. 4. Division of strokes into segments. (a) Original image. (b) Segmentation.

4 Knowledge Extraction

Once descriptors have been extracted from the drawing, different processes must be activated to identify and recognize all the elements in it. As we have used different set of descriptors for each kind of element, we must also use different recognition methods to locate and identify them. We can group these methods into three different kinds of pattern recognition approaches: rule-based recognition, string matching and statistical classification.

4.1 Rule-Based Recognition

This approach is used for the detection of the structure of the building: walls and windows. It relies basically on the perceptual grouping descriptors. Starting from the parallelism and overlapping descriptors, it consists in applying a set of rules to determine if two or three parallel and overlapping vectors can be considered as a wall or a window, respectively (see figure 5). Collinearity is used to join different lines belonging to the same wall.

The set of rules applied are based on the distance between parallel and collinear lines, the degree of overlapping between them, the aspect ratio of the rectangle enclosing the wall or the window and the length of the segments composing them.

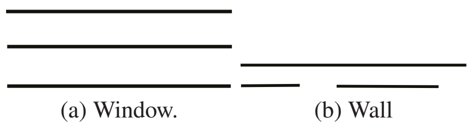

Fig. 5. Detection of walls and windows with three or more lines.

4.2 String Matching

String matching [9] is used for the recognition of two different kinds of elements in the drawings: the detection of doors and the recognition of furniture symbols in on-line drawings. The recognition of doors is based on the previously detected loops, represented as a string of vectors. Then, string matching relies on computing an edit distance between the loop found in the image and the loop representing the model of a door. The final distance value allows to determine if both loops are similar.

For symbols in on-line drawings, the string matching is applied after splitting each stroke into a set of segments, and representing each segment with its *chaincode*. String matching allows to define a distance measure between each segment in the symbol image and each segment in the model of the symbol. This distance is computed weighting the final edit cost resulting from the string matching with other measures, such as the distance between end points of both segments and the difference in length between both segments. Once we have computed the distance among all segments, we can find the association between image segments and model segments yielding to the minimal global distance. This global distance is used as a measure of classification.

4.3 Statistical Classification

The recognition of furniture symbols in off-line drawings relies on pixel-based descriptors, namely circular zoning and *Zernike* invariant moments. As images will be represented as feature vectors, we must use some statistical method to classify them. We have used the Mahalanobis distance as a criterion for classification, as it allows to take into account the variability in each class, learned from a set of symbol samples.

5 Experimental Evaluation

This section is devoted to illustrate the performance of the application scenario of architectural sketch understanding. As described above, the system is a tool for early design stages of architectural projects. The system converts sketches of floor plans to a CAD representation consisting of building elements. We can distinguish three major categories of elements: structure (walls, doors and windows), furniture (chairs, beds, tables, etc.), and facilities (electrical, illumination, etc.). Notice that since any of the above elements has its own diagrammatic representation, the sketch understanding problem actually consists of a symbol recognition problem. As it was noticed above, a key component of our architecture is a sketchy interface. Actually, the system can work either on-line or off-line and hence, the symbol recognition algorithms take into account such a twofold input procedure. Conceptually, we use a digital pen and paper paradigm, i.e. the system allows the user to input a scanned sketch, to interactively draw a new one or edit an existing one by the use of a TabletPC, or even to use paper as input medium but with paper augmented functionallity [10]. Figure 6 illustrates the sketch understanding process. First, the initial paper sketch of Fig. 6(a) has been scanned. The different feature extraction procedures and symbol recognition strategies described in sections 3 and 4 have been applied. Figure 6(b) shows the reprinted document once different symbols have been recognized. Afterwards, the user by means of a sketchy interface in a TabletPC has edit the plan adding some elements, see Fig. 6(c). In that case, dynamic descriptors have been extracted and on-line symbol recognition methods have been applied to get the final result of Fig. 6(d).

Let us now further analyze the performance of the system in a quantitative way. This analysis is formulated in terms of the performance of the different symbol recognition methods that are applied for the recognition of the graphical elements. To do so we have used the following groud-truth. We have used a set of 25 classes of architectural symbols. Ten different people have been asked to sketch at least ten instances of each symbol. We have then collected a database with 4200 symbol images with different distortion levels. Symbols have been drawn using a Logitech io pen device [10]. It has allowed us to have for each sample an on-line and an off-line version. The symbol recognition algorithms described in sections 3 and 4 have been applied to the samples in the database. The off-line process formulated in terms of statistical classifiers had an overall recognition rate of 69.3%. This rate was very sensitive to the symbol instances used to learn pattern models. Thus, the learning set was constructed by taking one symbol instance for each people and symbol class. This results in a very high intra-class variability and hence a high inter-class confusion rate. It can be noticed if we reduce the test set to the set of symbols drawn by only five people. In that case, the recognition rate ranges from 76% to 81%, depending on the five people selected. In addition, when a symbol is missclassified, we have considered the second option given by the classifier. If it is taken into account, the recognition rate achieves a 84%. This suggests a system in which the models are personalized to a given user. Concerning to the on-line recognition symbol recognition method, we have gotten an overall recognition rate of 99.08%, ranging from 95% in the symbol showing the worst performance to 100% in the symbols showing the best performance. For some of the symbols, we have defined two models in order to be able to adjust the method to different drawing styles. The on-line system is able to recognize 200 images per second. These results are still very preliminary and they can be improved introducing some variations in the classifiers.

6 Conclusions

A number of high performant graphics recognition systems exist. Most systems often use *ad-hoc* methods and are very domain-dependent. However, if we further analyze the existing systems we can notice that the methodological basis is almost the same, and from one system to the other the differences are the tuning parameters and the domain-dependent knowledge. In this paper we have proposed a general graphics recognition architecture. The architecture combines a set of feature extraction modules that, combined in terms of domain-dependent knowledge, allow to recognize document entities in a given application scenario. A second important component of our architecture is the definition of a relational metamodel that allows to hierarchically represent a document at different abstraction levels (from features to entities). Finally, in our architecture we have also been concerned in the HCI by defining a sketch-based interface that not only allows to create technical drawings but also to edit existing ones by adding new elements or interpreting gestures as edit commands. The second part of the paper has been

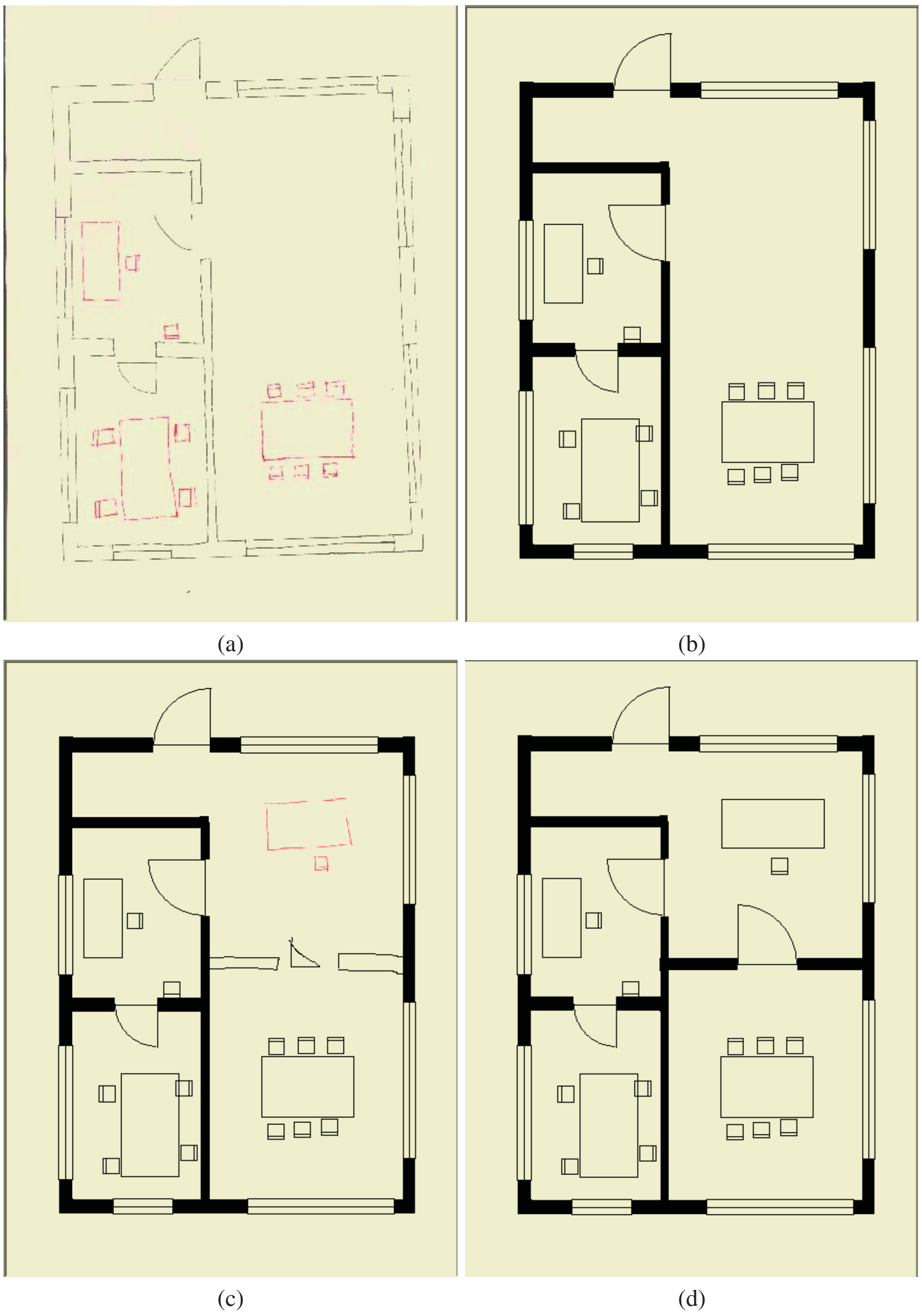

Fig. 6. Sketch recognition process: (a) Initial sketch, (b) recognized entities, (c) added entities using a sketchy interface, (d) final result.

devoted to describe an application scenario, in particular, a system to convert architectural sketches, either off-line or on-line, to a CAD representation. The set of descriptors used by the symbol recognition methods have been briefly described and finally the performance evaluation of these methods has been analyzed. For this last part, a groudtruth database of more than 4000 hand-drawn symbols have been used. The proposed architecture is still in an early stage. In addition to introduce improvements in the symbol recognition methods to increase the recognition rates, the improvements that we are now working on are the inclusion of other descriptor extraction methods and the design and development of a prototyping framework, i.e. a way to combine descriptor extraction modules with domain-dependent knowledge to generate application scenarios.

References

1. Clavier, E., Masini, G., Delalandre, M., Rigamonti, M., Tombre, K., Gardes, J.: Docmining: A cooperative platform for heterogeneous document interpretation according to user-defined scenarios. In: Proceedings of Fifth IAPR Workshop on Graphics Recognition. (2003) 21–32 Barcelona, Spain.
2. Lamiroy, B., Najman, L.: Scan-to-XML: Using software component algebra. In Blostein, D., Kwon, Y.B., eds.: Graphics Recognition: Algorithms and Applications. Springer, Berlin (2002) 211–221 Vol. 2390 of LNCS.
3. Pridmore, T., Darwish, A., Elliman, D.: Interpreting line drawing images: A knowledge level perspective. In Blostein, D., Kwon, Y., eds.: Graphics Recognition: Algorithms and Applications. Springer, Berlin (2002) 245–255 Vol. 2390 of LNCS.
4. Gross, M.: The electronic cocktail napkin - working with diagrams. Design Studies **17** (1996) 53–69
5. Leclercq, P.: Absent sketch interface in architectual engineering. In Lladós, J., Kwon, Y.B., eds.: Selected Papers from Fith International Workshop on Graphics Recognition, GREC'03. Springer-Verlag (2004) 351–362 Vol. 3088 of LNCS.
6. Tombre, K., Ah-Soon, C., Dosch, P., Masini, G., Tabonne, S.: Stable and robust vectorization: How to make the right choices. In: Proceedings of Third IAPR Work. on Graphics Recognition. (1999) 3–16 Jaipur, India.
7. Adam, S.: Interprétation de Documents Techniques: des Outils à leur Intégration dans un Système à Base de Connaissances. PhD thesis, Univeersite de Rouen-Mont Saint Aignan UFR de sciences et techniques (2001)
8. Belkasim, S., Shridar, M., Ahmadi, A.: Pattern recognition with moment invariants: A comparative study. Pattern Recognition **24** (1991) 1117–1138
9. Chen, D., Cheng, X.: Pattern Recognition and String Matching. Kluwer Academic Publishers (2002)
10. Logitech: IO digital pen (2004) www.logitech.com.