Symbol Spotting in Technical Drawings Using Vectorial Signatures*

Marçal Rusiñol and Josep Lladós

Centre de Visió per Computador / Computer Science Department
Edifici O, Campus UAB 08193 Bellaterra (Cerdanyola), Barcelona, Spain
{marcal, josep}@cvc.uab.es
http://www.cvc.uab.es

Abstract. In this paper we present a method to determine which symbols are probable to be found in technical drawings using vectorial signatures. These signatures are formulated in terms of geometric and structural constraints between segments, as parallelisms, straight angles, etc. After representing vectorized line drawings with attributed graphs, our approach works with a multi-scale representation of these graphs, retrieving the features that are expressive enough to create the signature. Since the proposed method integrates a distortion model, it can be used either with scanned and then vectorized drawings or with hand-drawn sketches.

1 Introduction

Symbol recognition is one of the major research activities in the field of Graphics Recognition. It has a number of applications to the analysis of technical drawings and maps at large. Examples are the interpretation of scanned drawings for validation or retroconversion, or the iconic indexing in a document image database. In the problem of retrieving images from a database by their content, applications use to formulate queries in terms of textual information as latitude coordinates, street names, etc. or graphical information as the situation of interesting elements as roads, rivers, airports, etc. as explained in [1]. On the other hand, iconic indexing allows to retrieve images from a large database by querying single elements of the drawing. Usually, architects or engineers have a great amount of technical drawings and they re-use data from previous projects for their new designs. Nowadays locating these elements requires visual examination of each document and it is a tedious task. Iconic indexing is suitable to provide solutions to this kind of problems. Often it is more natural and effective, instead of making a textual query to use a sketching interface where the symbol to spot is not stored in a database but drawn in an on-line process by the user.

In this paper we present a method to discriminate symbols in technical drawings for an indexing purpose. Effective symbol recognition methods exist in the

* This work has been partially supported by the Spanish project CICYT TIC 2003-09291 and the Catalan project CeRTAP PVPC.

literature [2]. Structural, usually graph matching, statistical or hybrid methods may be found. However, most of the approaches require a segmentation of the symbol to achieve a classification with a high performance. When dealing with large documents, a paradox appears: to segment for recognizing or to recognize for segmenting. Graph matching or other recognition schemes are too much complex to tackle with segmentation and recognition simultaneously. So, a method to determine which symbols are likely to be found in the drawing is suitable as a pre-processing step to recognition techniques. A compact representation of features —signature— can provide the accuracy and the speed desirable in such cases. A comprehensive review on description techniques for shape representation can be found in [3].

There are two different signature paradigms for symbol description depending on whether it is represented, with pixel based features or with a vectorial representation. Some techniques like those described in [4] use bitmap images to find a signature, named force signature, based on angular information to describe each symbol. The method can handle degradation of the objects to recognize, because force signatures are robust to noise. Also, the method can discriminate very similar objects, but user interaction is needed to correct object segmentation. In [5] constraints as length-ratios or angles between two pixels in reference to a third pixel are accumulated in a histogram used as a symbol descriptor. This method gives high performance under diverse drawbacks as degradation, rotation, scaling. However it also needs a pre-segmentation of the symbols and its complexity is very high ($\mathcal{O}(n^3)$) because every triplet of pixels is considered. Since technical drawings are highly structured and composed by simple geometric sub-shapes, it seems more suitable to work with a vectorial representation rather than with pixel based images. In [6] the raster images are approximated by polygons to obtain boundary segments and to extract constraints from this representation as distance, angle, directions, etc. among segments. An indexing scheme is built and the method gives good results locating and recognizing the objects. However, the method can not handle images with noisy edge data. In [7], the image is completely vectorized and the segments are combined to form a set of tokens, as chains of adjacent lines, or even textual features. The frequency of an indexing feature and the document frequency of the indexing features are taken into account to extract the similarity between a document and the query. We have based our work on the method proposed by Dosch and Lladós [8], where the vectorial signatures are built from the occurrences of constraints between segments, as parallelisms, straight angles, overlap-ratios, etc. But, these signatures are calculated only for small rectangular parts of the image. This fixed bucket partition provokes a lack of flexibility, there is no invariance to scale. It can also cause some problems if the symbol to discriminate is not completely inside of one of these windows.

Our work is targeted to vectorized drawings obtained either from scanned printed pages or from hand-drawn sketches. The operation of feature extraction when we are working with a vectorial representation can provide more speed than when we face up to a pixel based approach. The drawback of vectorial data is that a distorted input or even a different set of vectorization parameters may result in a sensitive variation in the obtained segments. Because of that, the spotting technique must be error tolerant and invariant to rotation, scale and translation. These constraints, which are a problem in bitmap images, are easily solved in vectorial representation if the features taken into account are extracted using comparisons between different segments (ratios or angles).

The features are organized in a signature and then matched against a database of signature models in order to index the different regions of interest of the drawing, as explained in [9]. This kind of indexing-based approaches, together with vectorial signatures, do not look for an exact match with the model symbols but to determine which symbols are probable to be found in these regions by a fast technique. Afterwards, the recognition step can be focused in each region and the well-known recognition techniques can take advantage of the extracted knowledge of the spotting process.

Vectorial signatures are suitable for this kind of problem since the drawings are formed by structured sub-shapes, so, the extracted constraints are focused in geometric-invariant relationships between segments, as parallelisms, straight-angles, segment junctions, length-ratios, distance-ratios, etc. The determination of the spatial relationship between a pair of segments is explained in [10]. These simple constraints can be observed and organized in a hierarchical way in order to represent some expressive structural relations between different segments. This means that it is more representative to find a square than two parallelisms and four straight angles. Our method tries to find these sub-shapes to create the vectorial signature.

The remainder of this paper is organized as follows. Section 2 describes how the vectorial signatures are built and learned. In section 3, we present the construction of the regions of interest which allows to apply the signatures in real drawings. In Section 4, the experimental results are explained and finally, the conclusions are discussed in section 5.

2 Building the Vectorial Signatures

Given a line drawing image, it is first vectorized and represented with an attributed graph. Graph nodes represent segments and graph edges represent structural relationships among segments. Formally, a graph G is defined as follows:

Definition 1. *A* graph *is denoted as $G = (V, E)$ where V is the set of nodes representing the segments and E is the set of edges representing the relationship between them. A* subgraph *of G containing the nodes $s_i, ..., s_j$ will be denoted as $G_{\{s_i,...,s_j\}}$. G is a complete graph.*

Definition 2. *An* attributed graph *is denoted as $G = (V, E, \mu, \nu)$ where Σ_V and Σ_E are a set of* symbolic labels, *and the functions $\mu : V \rightarrow \Sigma_V$ and $\nu : E \rightarrow \Sigma_E$ assign a label to each node and each edge. $\Sigma_V = [\theta_{s_i}, \rho_{s_i}]$ will contain the information of each segment s_i according to a polar representation. $\Sigma_E = \{L, T, P, 1, 0\}$ will represent the different kind of relationship between each pair of segments. The possible relationships between segments are:*

1. *L represents a straight angle between a pair of adjacent segments.*
2. *T represents a straight angle between a pair of non-adjacent segments.*
3. *P represents two parallel segments.*
4. 1 *represents two adjacent segments.*
5. 0 *represents a non expressive relation between two segments.*

Definition 3. *M_G is the* adjacency matrix *of a graph G and $M_{G\{s_i,\ldots,s_j\}}$ the adjacency matrix of the subgraph $G_{\{s_i,\ldots,s_j\}}$.*

We define a signature in terms of a hierarchical decomposition of symbols. Thus, it counts the occurrences of primitive shapes that are expressive enough in line drawings as squares, triangles, parallelisms, etc. Since the number of vectors of these shapes are not the same for all of them, we work with a combinatorial approach on the number of graph nodes. In Fig. 1, we have got a graph[1] of a simple symbol, we can see the symbol could be decomposed in a square and in a triangle, the subgraph $G_{\{s_2,s_3,s_4\}}$ represents the triangle and the subgraph $G_{\{s_1,s_4,s_5,s_6\}}$ the square.

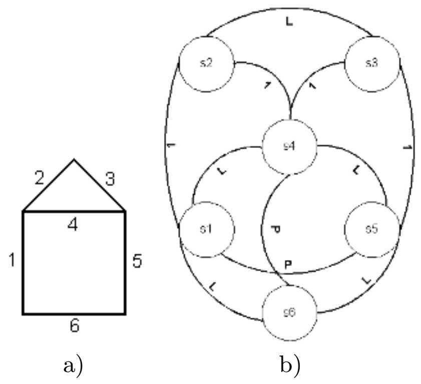


Fig. 1. (a) A simple symbol. (b) Its attributed graph.

For all the segments, all the subgraphs formed by at least two nodes, and a maximum of four nodes are analyzed to search some representative shapes. The equation 1, being n the number of segments, gives the number of subgraphs to analyze.

$$\#G_{\{\ldots\}} = \sum_{k=2}^{4} C_n^k = \sum_{k=2}^{4} \frac{n!}{(n-k)! \times k!} \tag{1}$$

For each subgraph, we work with its adjacency matrix. The matrix M_G is in fact only computed once for all the segments, and then, when we want to focus to a subgraph, a group of rows and columns of this matrix will be selected. Notice that in most cases the relations between segments could be extracted in the

[1] The edges labelled with a 0 are not shown.

vectorization process. In the extraction of these constraints, each comparison has associated a threshold value in order to be more tolerant. For the simple shape of Fig. 1, we can see below (2) its corresponding adjacency matrix M_G.

$$M_G = \begin{pmatrix} s_1 & 1 & 0 & L & P & L \\ 1 & s_2 & L & 1 & 0 & 0 \\ 0 & L & s_3 & 1 & 1 & 0 \\ L & 1 & 1 & s_4 & L & P \\ P & 0 & 1 & L & s_5 & L \\ L & 0 & 0 & P & L & s_6 \end{pmatrix} \tag{2}$$

From this matrix, we examine all the possible combinations of matrices taking four, three and two of the six possible nodes. Hence three different levels are considered. Below, in (3) we can see the resulting sub-matrices considering only the triangle and the square of the shape of Fig. 1.

$$M_{G_{\{s_2,s_3,s_4\}}} = \begin{pmatrix} s_2 & L & 1 \\ L & s_3 & 1 \\ 1 & 1 & s_4 \end{pmatrix} \, and \, M_{G_{\{s_1,s_4,s_5,s_6\}}} = \begin{pmatrix} s_1 & L & P & L \\ L & s_4 & L & P \\ P & L & s_5 & L \\ L & P & L & s_6 \end{pmatrix} \tag{3}$$

For all the sub-matrices representing the subgraphs, normally the analysis of one single row can determine the shape that it encodes.

Definition 4. *Let us denote $d(M_{G_{\{s_i,\ldots,s_j\}}})$ a single row of an adjacency matrix. It will be used as a* descriptor *of the expressive sub-shapes. Every descriptor d has associated an equivalence class $L : \{d_1, ..., d_n\}$ of its synonyms.*

Definition 5. *Let S be a vector where in each position we have the number of occurrences of each descriptor $d(M_{G_{\{s_i,\ldots,s_j\}}})$ representing an expressive sub-shape. S is defined as the* vectorial signature *of the analyzed shape.*

A descriptor of a reference segment having two straight angles and a parallelism represent a square $d = [s_iLLP]$. As the information inside the matrix depends on the order of definition of the segments when the vectorization step is done, we must have a dictionary of its synonyms $L : \{d_1, ..., d_n\}$. For instance, a segment having a descriptor $d = [s_iLLL]$ is the same that another segment having a descriptor $d = [s_iPLP]$ because if a segment is perpendicular to three other segments, one of them is parallel to two of them and perpendicular to the last one. We can see some example of synonyms list and some of the expressive shapes taken into account to perform the voting scheme to build the signature in Table 1. It seems redundant to store the information for multiple levels of the subgraph, if in the level of four nodes we find a square, it is obvious we will find two parallelisms and four straight angles in the level of two nodes. But this redundancy helps to detach completely all the multiple shapes in the drawing. For instance, a square with a cross inside can be seen as a square and a straight angle, or it can be seen as a set of triangles (see example in Fig. 2). This redundancy helps to be more error tolerant and to store all the structural information

Table 1. Some possible rows of the sub-matrix, with its synonyms and its representations

Level	Synonyms list	Image representation of the sub-shapes
4 nodes	$[[s_iPPP]]$	
4 nodes	$[[s_iPPL],[s_iPLP],[s_iLPP],[s_iLLL]]$	
4 nodes	$[[s_iPLL],[s_iLPL],[s_iLLP]]$	
3 nodes	$[[s_iPP]]$	
3 nodes	$[[s_iPL],[s_iLP],[s_iLL]]$	
3 nodes	$[[s_i11]]$	
2 nodes	$[[s_iP]]$	
2 nodes	$[[s_iL]]$	
2 nodes	$[[s_i1]]$	

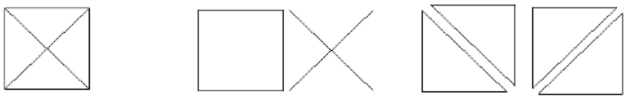


Fig. 2. (a) Original symbol. (b) Symbol detached at level four and at level two. (c) Symbol detached at level three.

of all the multiples sub-shapes of the drawing. The occurrences of each sub-shape will vote to build the vectorial signature. To have more information, we can add some additional information as the length-ratio and the distance-ratio at the end of the signature. These measure features can take values from 0 to 1; this space is the split into five bins where the votes are accumulated. We can see an example of a signature of a symbol in Fig. 3.

The learning process of the signatures has been made using a selection of the symbol database used in the *GREC 2003* symbol recognition contest [11], containing symbols of both architectural and electronic fields. The symbols with arcs are not taken into account, except the doors which are approximated by a polyline. The signature of the twenty-seven symbols has been calculated, and a mean of signatures has been made with a set of at least ten distorted representations of each symbol. With this symbol database, there is no need to use higher order relationships to discriminate the symbols between them.

Once the database of signatures is constructed, we can compare the obtained signature with this database and associate a probability to each correspondence between the original symbol and all the symbols of the database following a distance function. This comparison process could be done in a hierarchical way,

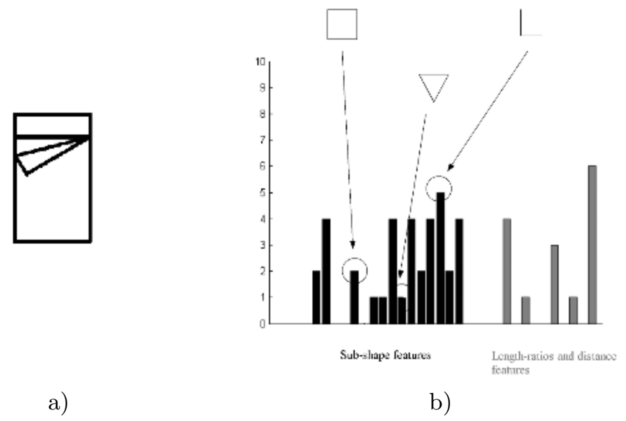


Fig. 3. (a) A symbol of a bed. (b) A graphical representation of its signature divided in the sub-shapes features and the measure features.

in order to discard some non possible solutions, or simply associating weights to each feature of the signature to classify the most relevant features. The used distance function is a simple Euclidean distance which gives acceptable results.

3 Regions of Interest

When we work with complete drawings we need to divide the drawing into separate windows framing every symbol. In each zone of interest a voting scheme of structural relations is used. The idea is that every structural relation found in this zone will contribute to form the signature to be compared with the models. Every framing window has its own voting scheme and the regions where the votes reach their maximum will be selected as candidates to contain the queried symbol using a voting approach inspired by the idea of the GHT [12]. These regions are dynamic since they are built depending on the original line drawing, this approach works much better than other segmentation techniques used in technical drawings which only divide the drawing in a fixed bucket partition which loses flexibility.

These regions of interest are computed from the maximum and minimum coordinates of several adjacent segments. So, the size of the regions of interest is variable. Also, a first filter of area and aspect-ratio can be easily implemented in order to delete some non relevant symbols as for example the walls in the architectural field or the wiring connections in electronic diagrams.

For each node n_{s_i} of G (segment in the drawing) we build a list of all the nodes connected to n_{s_i} by an edge. We have a list of all the endpoints of the adjacent segments to reference segment. In this list, we get the maximum and minimum coordinates of the endpoints that will construct a framing window of these segments. As in most cases of technical drawings the symbols have a low eccentricity, its bounding-box are square-shaped and this kind of windows frame them. But, as the windows are based on the connection of the segments, the efficiency decrease if the symbols are disconnected or overlapped.

But, in the vectorization step, more problems may happen: small vectors can appear due to noise, straight lines can be split into several collinear vectors, the arcs are approximated by polylines, some neighboring lines in the drawings are not adjacent in the vectorial representation because of gaps, dashed lines appear as a set of small segments instead of one unique instance, etc. To solve this kind of problems, the best results are reached when we work with a lower resolution of the drawing to calculate the windows. This sub-sampling step reduces local distortions in the vectorial representation but preserving the most salient geometrical properties.

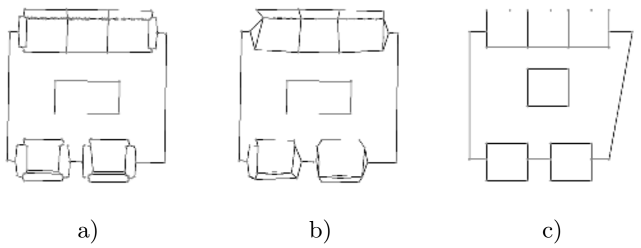


Fig. 4. (a) Original drawing. (b) Graph contraction by distance. (c) Low resolution representation.

First, a contraction of the normalized graph is done, merging the adjacent nodes having a lower distance than a threshold thr. Then, applying the equation 4 to each node coordinate we get a lower resolution graph. With this representation with decreased resolution, the problems of the gaps, or the split segments are solved. Every endpoint is sampled for each step of m, so the minor errors are corrected.

$$\begin{aligned} x &= m \times round(x/m) \\ y &= m \times round(y/m) \end{aligned} \tag{4}$$

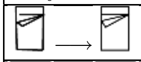
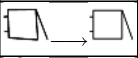
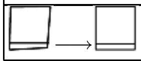
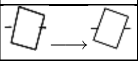
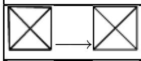
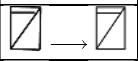
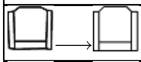
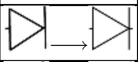
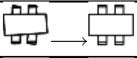
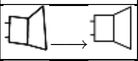
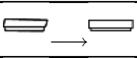
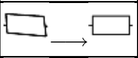
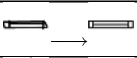
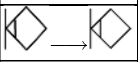
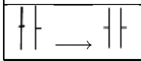
Experimentally, in Fig. 4(a) the graph has 154 nodes because an horizontal line has been split in the vectorization process. When the graph contraction by distance is done (with a threshold value $thr = 0.06$) Fig. 4(b), we get a graph with 52 nodes which lines are crooked due to the node contraction, and with the decreased resolution graph (with $m = 35$) Fig. 4(c) we have to face up to only 33 nodes.

This change of resolution can cause some other errors, for example some lines which are almost horizontal or vertical can be represented with a very different slope. But these errors do not interfere with the obtained windows, since they continue to frame the symbols. Notice that these lowest resolution images will only be used to calculate the regions of interest, not for the spotting process. Since each segment proposes a region of interest, there is no problem if one of the segments of a symbol gives a mistaken window.

4 Experimental Results

Our experimental framework consisted of two scenarios. First we have tested the performance of our approach to classify isolated symbols. Secondly, we have used the method for symbol spotting in real architectural drawings.

Table 2. Results of the recognition

Symbol	Recognition rate	Symbol	Recognition rate
	36/38		19/19
	39/39		19/20
	15/15		15/15
	27/27		16/16
	15/15		20/20
	7/8		20/20
	36/38		14/14
	11/11	Total	309/315

The first tests were done using the *GREC 2003* database. This database contains, in addition of the models, some synthetical distorted symbols, rotated symbols and symbols at different scales. Working with fifteen different classes of symbols, a set with 315 examples of different levels of distortion has been tested and we achieved a 98% of recognition. We can see the detailed results in Table 2. With 230 examples of rotated and scaled symbols we achieved the 100% of recognition.

In the second test, we tried out the vectorial signatures with real architectural drawings. Allowing a higher error than when we are working with the database, the symbols can be spotted, and they are usually well discriminated. As we can see in Fig. 5, some false positives appear (dashed zones) and one sofa is not spotted (grey zone). False positives appear when a window does not correctly frame a symbol, or when a symbol (like the shower) is not in our signature database. The stairs which consist of a lot of segments give a lot of regions of interest where false positives appear, and the wrong segmentation of the tables makes that the part where the chairs are drawn a sofa is spotted, because their representation is very close. When we use vectorial signatures in real drawings there are two factors that cause the spotting results not to be so good. First of all, the symbols can be adjacent between them or to a wall, or the region of interest could not frame perfectly the symbol, in this case we face up to occlusions and additions of segments. On the other hand, in real drawings, the symbol design may be different of the learned model, so the learned features of a symbol could not appear in real drawings, in this case it is obvious the symbol can not be spotted, a semantic organization of different design instances for any symbol is necessary.

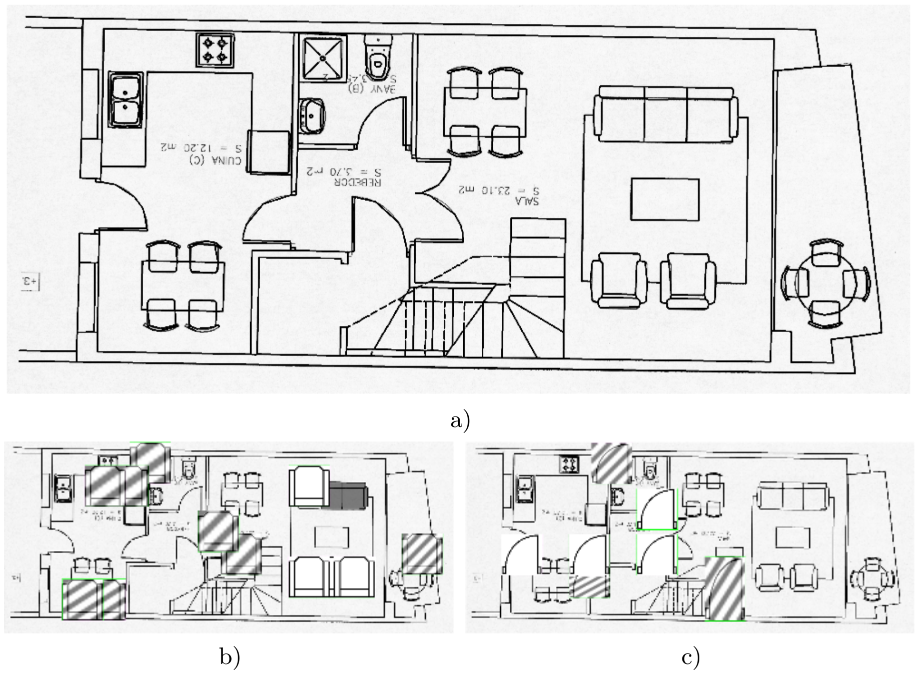


Fig. 5. (a) Original image. (b) Locations with high probability to find a sofa. (c) Locations with high probability to find a single door.

Finally we tried the method with an application of image database retrieval by sketches. A sketch of a symbol is drawn and then vectorized. Its signature is computed and the locations of a drawing with high probability to contain the queried symbol are spotted. We can see the results in Fig. 6. With this kind of applications we have to face up to two important drawbacks, first of all, the signature of the sketch is not the same of the models, because in the vectorization process a lot of small single segments appear. Secondly, as we said before, the symbol design may change from a drawing to another. That is why in this test we must allow a higher error in the spotting process and the constraints to obtain the regions of interest must be much more restrictive, otherwise a lot of false positives appear. This kind of applications, give much better results when they are used in an on-line process in order to have the users feedback when the sketch is drawn to correct the representation errors.

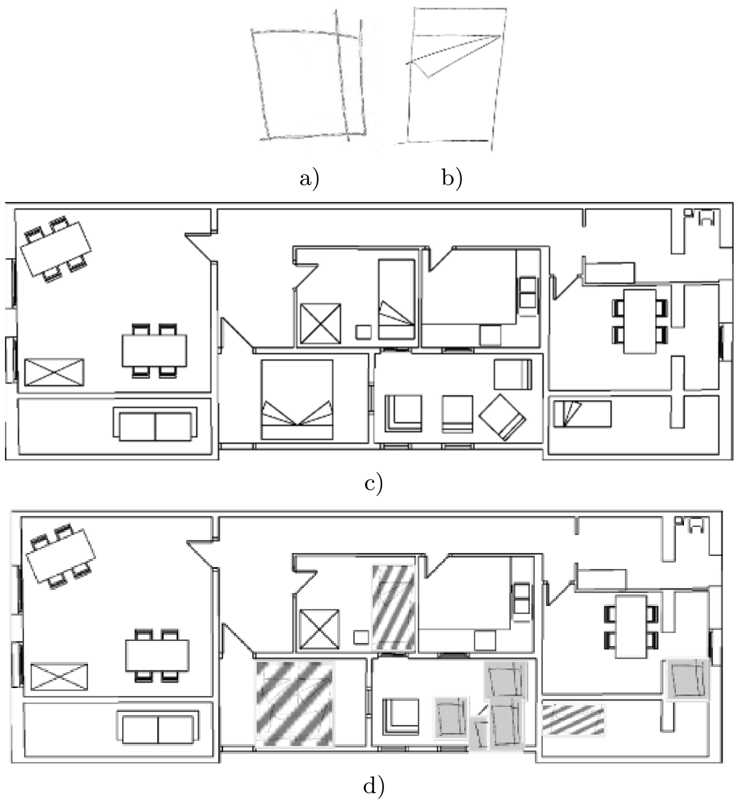


Fig. 6. (a) Sketch of a sofa. (b) Sketch of a bed. (c) Original drawing. (d) Locations with high probability to find a sofa (grey zones) and a bed (dashed zones).

5 Conclusions and Discussion

In this paper we have presented a method to detect the regions of a technical drawing where a symbol is probable to be found. This method is suitable for applications of iconic indexing and retrieval. Our method, starting from a vectorial representation of the line drawing, builds a vectorial signature in each region of interest using a voting scheme. These signatures are formed by the occurrences of some sub-shapes which can appear in line drawings that are expressive enough, and by some additional information as length-ratios, distance-ratios, etc. These signatures are then matched with the database of learned signatures of the models to determine which symbol is probable to be present.

We can see that the symbol discrimination using vectorial signatures gives good results when we are working with the database and with symbols with synthetical distortion, which is a controlled framework. In real scanned architectural drawings, the segmentation and discrimination of symbols are done simultaneously, even if the symbols are usually well spotted, a lot of false positives appear.

In the image database retrieval by sketches test we get acceptable results. As the objective of this technique is not to give a recognition of the symbol but in some way to index the drawing, the false positives problem is not so significant.

When working with vectorial data, one of the main drawbacks is the arc representation. In this paper we have approximated arcs with polylines, but it is necessary to add an arc segmentation algorithm in the vectorization process to consider the arcs and circles as expressive sub-shapes. Moreover the hierarchical organization of the signatures is essential to achieve a fast and accurate indexation of line drawings. Some features are more discriminative than others, and the presence or absence of a feature can cluster the candidate symbols database.

References

1. R.C. Thompson and R. Brooks. Exploiting perceptual grouping for map analysis, understanding and generalization: The case of road and river networks. In *Graphics Recognition, Algorithms and Applications*, pages 148–157. 2002.
2. J. Lladós, E. Valveny, G. Sánchez, and E. Martí. Symbol recognition: Current advances and perspectives. In *Graphics Recognition, Algorithms and Applications*, pages 104–127. 2002.
3. D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37(1):1–19, 2004.
4. S. Tabbone, L. Wendling, and K. Tombre. Matching of graphical symbols in line-drawing images using angular signature information. *International Journal on Document Analysis and Recognition*, 6(2):115–125, 2003.
5. S. Yang. Symbol recognition via statistical integration of pixel-level constraint histograms: A new descriptor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2):278–281, 2005.
6. F. Stein and G Medioni. Structural indexing: Efficient 2-d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(12):1198–1204, 1992.
7. O. Lorenz and G. Monagan. A retrieval system for graphical documents. In *Symposium on Document Analysis and Information Retrieval*, pages 291–300, 1995.
8. P. Dosch and J. Lladós. Vectorial signatures for symbol discrimination. In *Graphics Recognition: Recent Advances and Perspectives*, pages 154–165. 2004.
9. H. Wolfson and I. Rigoutsos. Geometric hashing: an overview. *IEEE Computational Science and Engineering*, 4(4):10–21, 1997.
10. A. Etemadi, J.P. Schmidt, G. Matas, J. Illingworth, and J. Kittler. Low-level grouping of straight line segments. In *Proceedings of the British Machine Vision Conference*, pages 118–126, 1991.
11. E. Valveny and P. Dosch. Symbol recognition contest: A synthesis. In *Graphics Recognition: Recent Advances and Perspectives*, pages 368–386. 2004.
12. D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.