Pattern Analysis and Applications
DOI: 10.1007/s10044-009-0161-2

Symbol Spotting in Vectorized Technical Drawings Through a Lookup Table of Region Strings

Marçal Rusiñol, Josep Lladós, Gemma Sánchez

Computer Vision Center, Dept. Ciències de la Computació, Edifici O, Universitat Autònoma de Barcelona
08193 Bellaterra (Barcelona), Spain
email:{marcal, josep, gemma}@cvc.uab.es
Tel:+34935814090 Fax:+34935811670

Abstract In this paper we address the problem of symbol spotting in technical document images applied to scanned and vectorized line drawings. Like any information spotting architecture, our approach has two components. First, symbols are decomposed in primitives which are compactly represented and second a primitive indexing structure aims to efficiently retrieve similar primitives. Primitives are encoded in terms of attributed strings representing closed regions. Similar strings are clustered in a lookup table so that the set median strings act as indexing keys. A voting scheme formulates hypothesis in certain locations of the line drawing image where there is a high presence of regions similar to the queried ones, and therefore, a high probability to find the queried graphical symbol. The proposed approach is illustrated in a framework consisting in spotting furniture symbols in architectural drawings. It has been proved to work even in the presence of noise and distortion introduced by the scanning and raster-to-vector processes.

Keywords: Document Image Analysis, Graphics Recognition, Symbol Spotting, Graphical Indexing Techniques, Cyclic String Matching.

Originality and Contribution

In this paper we focus on the problem of symbol spotting in graphical document images. Information spotting can be defined as locating queried items in document images contained in large databases for indexation, navigation and retrieval purposes. A spotting process requires a primitive extraction step and an indexing strategy. When it is applied to text documents, both printed or handwritten, it is called word spotting, whereas when it is applied to graphical documents it is called symbol spotting. Word spotting can benefit from the one-dimensional nature of text and underlying language models. It is a more mature problem than symbol spotting. The main contribution of this work is twofold. First, the application itself, *i.e.* symbol spotting in technical drawings. Symbol recognition is a central problem of graphical document analysis. It is a particular case of the shape recognition problem. But an emerging interest in the field of document analysis is to recognize symbols without previous segmentation. This segmentation-free recognition allows to query by shape document images, which is useful to browse, categorize or even index line drawings by sketch. Second, from a methodological point of view, we propose a novel structural approach for indexing vectorized data. In our approach, polygonally approximated closed regions contours are encoded by attributed strings. Thus, strings are used as indexation keys of a lookup table, after clustering them according to a median string formalism.

1 Introduction

The management of a large amount of digital data is of widespread interest. Nowadays many documents are still stored in paper format. A digitization of these technical documents and their organization in digital libraries becomes necessary. Documents are the basis of corporate information workflows. Space saving and digital preservation justify digitizing documents and storing them in image databases. But the raw image format is not rich enough, other challenging interests arise. Document mining, *i.e.* browsing, navigation, querying or classification in terms of document contents are useful activities. It requires efficient indexation methods to be incorporated to digital libraries. Textual documents make use of the nature of alphanumeric information, *i.e.* it is one-dimensional information that may be sorted, and dictionaries and language models underlie the indexation processes. But technical drawings like architectural plans, maps or electronic diagrams have the added difficulty of containing bi-dimensional graphical data. In addition, as line drawings are designed with *Computer Aided Design* (CAD)

software, when digitalizing these documents, a vectorized file format is preferred and a raster-to-vector process has to be applied after scanning. As a result of that, technical document mining applications have to face two different noise sources, namely the inherent noise arising from the scanning process, and then the errors introduced by the raster-to-vector methods. Despite these drawbacks, the vetorized representation is preferred as it offers a compact data representation and makes the documents easy to edit and modify. Usually, the architects or engineers re-use data from previous projects for their new designs. In this context, an efficient application aiming to query graphical symbols in digital libraries of scanned technical line drawings is essential as pointed by Tombre and Lamiroy in [1]. Indexing these technical documents is a requirement when the application has to deal with a big amount of data stored in the library.

The problem of symbol spotting in graphical document images is here solved by means of an indexing *lookup table* (LUT). Usually, spotting has been applied to text documents and we can find in the literature several works focusing on this topic. Rath and Manmatha faced in [2] the problem of spotting handwritten words in historic documents. Kuo and Agazzi presented in [3] a spotting framework for machine printed text. And recently Lu and Tan presented in [4] a spotting method based on word codings. Even if these techniques can not be applied to spot graphical elements since they focus on one-dimensional data, these system architectures are still valid four our purpose. Generally speaking, the spotting problem can be defined as the retrieval of a set of zones of interest from a document image database which are likely to contain an instance of the queried item. Spotting systems are usually queried by example. That is, the user segments an item he wants to retrieve from the document database and this cropped image acts as input of the system. The desired output should be a ranked list of zones of interest likely to contain similar items to the queried one. A spotting process requires a descriptor and an indexing strategy. Because of this architecture, descriptors require to be simple and compact, in order to be able to be organized in an indexation structure with quick access. If more precise recognition rates are required, a more sophisticated symbol recognition approach could afterwards focus on each of these zones of interest. Cordella and Vento reviewed in [5] and Lladós *et al.* in [6] the state of the art on symbol recognition.

The advantage of spotting methods is that they are able to tackle with recognition and segmentation at the same time. Symbol spotting framework differs from usual *Content Based Image Retrieval* (CBIR) systems since the desired output is not a set of images containing the queried shape, but a set of cropped zones of interest from different images containing instances of the query. CBIR methods assume that the images in the collection are already segmented and are always taken as a unit. When trying to locate some zones of interest in large document images a different problem than CBIR has to be faced as it is desired to locate some graphical shapes appearing among a cluttered environment. For example, Califano and Mohan presented in [7] an indexing system to spot shapes in non-segmented images. Indexing strategies are usually formed by a primitive organization and clustering in an indexing table followed by a voting process, as the well-known generalized Hough transform introduced by Ballard in [8], to reinforce the proposed hypothetic locations where the symbol can be found.

The existing literature on primitive indexing methods usually follows the idea of geometric hashing introduced by Lamdan and Wolfson [9]. Geometric hashing takes the coordinates of two point sets used as indexes and it is able to compare two shapes under some transformations such as translations, scaling and rotation; yielding good results for shape discrimination as shown by Cohen and Guibas [10]. Even affine and projective transformations are tolerated. Lamiroy and Gros [11] proposed an enhancement of geometric hashing for 3D polyhedral object recognition. However, for both symbol and object recognition, geometric hashing needs a previous step of reference points extraction, as for instance high curvature points. The major disadvantage of the method is that the same subset has to be chosen for both the model image and the previously acquired images. Most approaches compute a polygonal approximation of the contour shape to recognize, taking the resulting segments as primitives to encode with geometric hashing.

In the presence of noise of scanned documents, the raster-to-vector process usually results in vectors with considerable deformation and makes pattern description unstable. Besides, vectorization algorithms usually fail on junction positions and tend to fragment segments. Such drawbacks entail a lower performance of geometric hashing since it is very hard to guarantee the same point set selection. In this paper we propose a method aimed to face up these distortions. First, closed regions contours composing a symbol are taken as primitives which act as discriminative features among different shapes. Then, these region contours are represented by a chain of adjacent segments – *polyline*–. We can find in the literature several works which try to encode shapes by strings of boundary segments. For instance, Stein and Medioni presented in [12] a shape descriptor based on a feature vector of length and angles of the chain of segments which approximates the boundary of the shape. This descriptor yields good recognition results if the extracted contours are not very corrupted by noise. However this kind of approaches are still dependent of the number of segments composing two shapes to match. In the method we present, as we take polylines as primitives instead of segments, we avoid the influence of the segment fragmentation of the raster-to-vector algorithms, and similar primitives are compared independently of the number of segments which approximate them. A polyline is

represented by an attributed cyclic string and thus two polylines are compared taking into account the different string edit operations needed to transform a string to another. The use of the string matching algorithm to compute a similarity measure also benefits the computation of median strings. To build an indexing lookup table, similar strings are clustered and a representative of each cluster is computed as a set median string. These set median strings act as indexing keys.

The remainder of this paper is organized as follows: we introduce in the next section how we represent and encode the symbols in terms of regions taken as primitives and polygonally approximated. In section 3, the proposed string edit distance is presented as a similarity measure between polylines; starting with the basics of the string theory and focusing on the proposed attributed string matching for closed polylines recognition. Subsequently, in section 4, we describe the architecture of the spotting process: the used method to build the lookup table clustering the representative regions, the querying process to activate table entries and finally the voting scheme to find the zones of interest where the symbol can be found. Section 5 provides the experimental results and finally section 6 concludes with a summary and a discussion of extensions and future work.

2 Primitive Level Processing

The first step in a symbol spotting framework is to preprocess the scanned images and to decompose in primitives the target documents as well as the queried models. Let us briefly explain in the next subsections the preprocessing step and the primitive extraction.

2.1 Preprocessing Steps

Let us first introduce how the technical line drawings are preprocessed. The preprocessing step is formed by two different phases: a denoising process working at pixel level to remove the distortions introduced by the scanning process and a raster-to-vector process aiming to polygonally approximate the raw image.

First, the technical drawings are scanned and the grayscale images are binarized and denoised using simple binary operations based on morphological operations. When working with documents which have been printed years ago and then scanned, the inherent noise and distortions as warping, paper folds, paper stains, etc. arising from these processes have to be faced. The interested reader is referred to Loce and Dougherty's review [13] of some simple existing techniques for digital acquired document enhancement and restoration.

Moreover, when computing the vectorization step some other problems arise. As stated by Tombre *et al.* in [14], nowadays it does not exist any "perfect" raster-to-vector algorithm and each method has its own lacks and produce its characteristic errors. Rather than representing symbols with a polygonal approximation based on a skeletonisation, our choice focus on computing a closed region labelling and extraction based on a connected component analysis. The contour of these closed regions is then polygonally approximated using the Rosin and West [15] algorithm.

2.2 Shape Representation in Terms of Polygonal Approximation of Primitives

Let us now detail how the extracted regions are represented and encoded in a suitable way to be compared afterwards.

The polygonal approximation of the contour of the closed regions is computed. Afterwards, an association of chains of adjacent segments resulting in a polyline is done. These polylines are encoded as attributed strings used as primitives to describe the symbol to be recognized.

Formally, let R be the contour of a closed region which is polygonally approximated and represented by the chain of adjacent segments $P(R) = \{s_1...s_n\}$ consisting of n segments s_i. Each segment s_i is attributed with the tuple (l_i, ϕ_i), where l_i denotes the length of the segment s_i and ϕ_i denotes the angle between s_i and the previous segment s_{i-1} in the counterclockwise direction. A symbol is then described in terms of its composing p region contours and denoted as $S = \{P(R_1)...P(R_p)\}$. We can see a graphical example in Fig. 1. Let us see in the next section how we can compute a distance between two strings representing a closed region.

3 Polyline Comparison Using Cyclic String Matching

We base the comparison of two polylines representing a closed region contour on the string edit distance algorithm. We can find in the literature a number of methods which use string matching techniques for shape recognition purposes, as for instance the ones presented by Wolfson in [16] or by Kaygin and Bulut in [17]. Of course, the most important point when using this kind of distances is the attributed string representation and the operation cost definition. Let us first introduce some basic definitions and notations from string theory stated by Wagner and Fischer [18].

3.1 String Matching Algorithm

Let Σ be an alphabet of elements, Σ^* the set of all finite strings over Σ and $A = a_1...a_n \in \Sigma^*$, $B = b_1...b_m \in \Sigma^*$ two strings with $n, m \geq 0$. The distance between A and

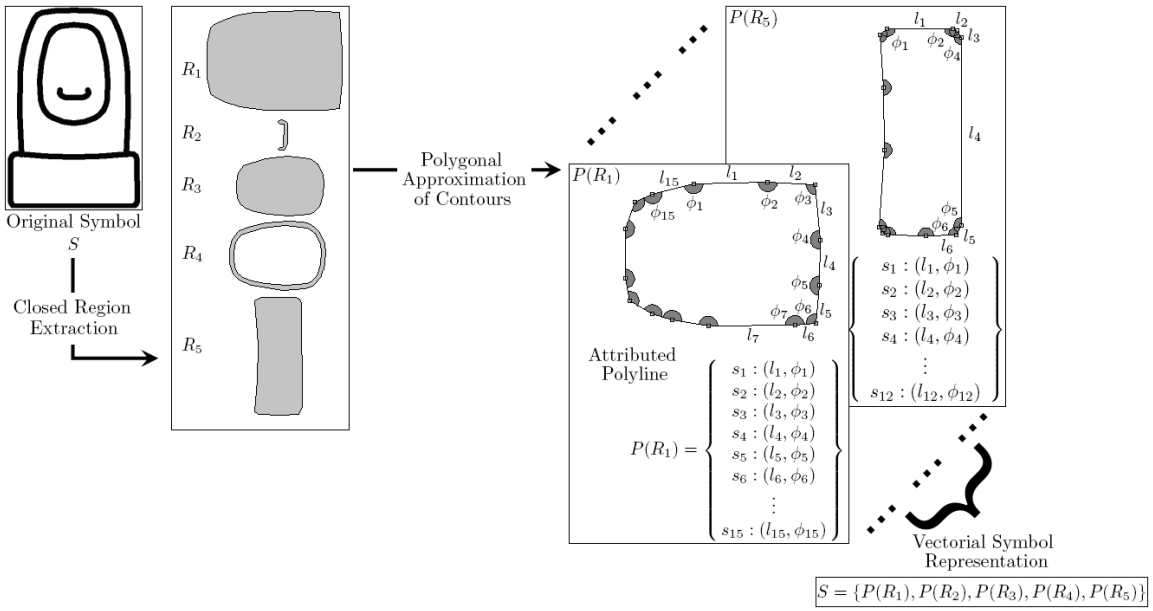


Fig. 1 Symbol representation in terms of polygonal approximation of closed region contours. Each of these region contours are represented by strings attributed by length and angles.

B, $\delta(A, B)$, is defined in terms of elementary edit operations required to transform A into B with minimum cost. Conventionally, three edit operations are defined. The *substitution* of an element $a \in \Sigma$ in A by an element $b \in \Sigma$ in B, denoted as $a \rightarrow b$. The *insertion* of an element $b \in \Sigma$ in B, denoted as $\lambda \rightarrow b$. Finally, the *deletion* of an element $a \in \Sigma$ in A, denoted as $a \rightarrow \lambda$, where λ denotes the empty element. Let A be a string $A = a_1...a_n$. The string $B = b_1...b_m$, $m \leq n$, is a substring of A if $A = a_1...a_{i-1}Ba_{j+1}...a_n$. The substring B is denoted as $A_{i,j} = a_i...a_j$, $1 \leq i, j \leq n$. Tsay and Yu proposed in [19] the *merge operation* denoted as $A_{i,j} \rightarrow a$ which approximates the substring $A_{i,j}$ into an element $a \in \Sigma$. This operation becomes essential when facing distorted data or affected by changes in scale since it allows to match two strings with different number of elements.

In order to compute the distance between two strings, each of the four edit operations have an associated cost γ. These costs are defined depending on the attributes which encode the string. In our case, as we want to compare polylines with a string matching algorithm the costs are defined in terms of segment comparisons.

However, the use of the string matching algorithm for closed shape recognition presents a problem. The starting symbol of the corresponding string has to be determined in order to compute the edit path between two strings. Maes proposed in [20] to represent polygonal shapes by cyclic strings thus avoiding the influence of the starting segment. Whereas basic string matching has a complexity of $\mathcal{O}(nm)$, the algorithm proposed by Maes can compute cyclic string matching with a complexity of $\mathcal{O}(nm\ log\ m)$, since all the paths in the edit operation table can be chosen such that two different paths never cross. For the sake of simplicity, from now on with string matching we will refer to the cyclic string matching algorithm proposed by Maes.

3.2 Cost Definition to Match Cyclic Chains of Segments

Visually, two chains of segments are similar if the length attributes and angles between consecutive segments can be aligned. In the literature on polygonal shape recognition, most approaches base the distance definition between two polygonal shapes on length and angle differences. For example, Arkin *et al.* used in [21] the turning function which gives the angle between the counterclockwise tangent and the x-axis as a function of the arc length. Their results are in accordance with the intuitive notion of shape similarity.

Let A and B be two chains of adjacent segments, represented as strings, with total lengths $|A| = n$ and $|B| = m$ and with respectively attributed string representations:

$$\begin{aligned} A &= (l_1^A; \phi_1^A)...(l_n^A; \phi_n^A) \text{ and} \\ B &= (l_1^B; \phi_1^B)...(l_m^B; \phi_m^B) \end{aligned} \tag{1}$$

The costs functions for attributed string matching are as follows:

$$\begin{aligned}
&\gamma((l_i^A;\phi_i^A)\rightarrow(l_j^B;\phi_j^B)) = \tfrac{|\phi_i^A-\phi_j^B|}{360} + \left|\tfrac{l_i^A}{|A|} - \tfrac{l_j^B}{|B|}\right| \\
&\gamma(\lambda\rightarrow(l_j^B;\phi_j^B)) = \tfrac{l_j^B}{|B|} \\
&\gamma((l_i^A;\phi_i^A)\rightarrow\lambda) = \tfrac{l_i^A}{|A|} \\
&\gamma((l_{i,j}^A;\phi_{i,j}^A)\rightarrow(l_u^A;\phi_u^A)) = \tfrac{\left(\sum_{k=i}^{j} l_k^A\right)-l_u^A}{\sum_{k=i}^{j} l_k^A}
\end{aligned} \tag{2}$$

which are the proposed cost functions inspired by the ones proposed by Tsay and Tsai in [22] where they use string matching for shape recognition. Maes proposed in [20] to use a weighting factor in the length costs to compensate undesirable cost bias for angle differences. However, in our experiments we did not observe any improvement in adding such parameter. Finally, for the sake of simplicity, the previous operations are grouped by a block substitution using the merge operation. The total cost of substituting a whole sequence of symbols by another is computed as follows:

$$\begin{aligned}
&\gamma(A_{i,j}\rightarrow B_{k,l}) = \\
&\gamma(A_{i,j}\rightarrow u) + \gamma(B_{k,l}\rightarrow v) + \gamma(u\rightarrow v)
\end{aligned} \tag{3}$$

being u and v the segments starting at the initial point of A_i and B_k and ending at the final point of A_j and B_l respectively.

As all the length comparisons are weighted by the total perimeter of the chain of segments and the angles are computed relatively to the previous segment, the proposed string matching approach is rotation and translation invariant. In addition, the merge operation attributes low edit costs to primitives undergoing noisy transformations as the inherent segment fragmentation from the raster-to-vector process and aims to compare strings with different number of segments making the system tolerant to segment cardinality and to scale changes.

4 Spotting Method

Given a technical document, the idea is to form clusters of similar strings in a lookup table. In addition, each entry of this table has a representative string acting as its indexing key.

The whole spotting method is divided in three different parts. Firstly the off-line step to build the lookup table. Secondly, the symbol querying process by the indexing function. Finally, the voting scheme which spots the zones of interest where there is high probability to find the symbol. Let us further describe the above steps.

4.1 Lookup Table Construction

Each lookup table entry represents a cluster of similar strings appearing in the document database and consists of two different items: a representative polyline of each cluster which acts as indexing key and the stored list of locations –translation vectors– where we can find the polylines belonging to this cluster.

Generally speaking, the representative polyline of each cluster can be computed in two ways, namely the *mean string* or the *set median string*. As proposed in Sánchez *et al.* in [23], the mean string M over a string cluster $C = \{A_1...A_n\}$ is defined as:

$$M = arg\min_{M\in\Sigma^*}\left(\sum_{i=1}^{n}\delta(A_i, M)\right) \tag{4}$$

The mean string is computed as a new string that represents the average shape among all the strings in the set. The main drawback of this approach is its computational cost which increases with a large number of shapes. In our case, we have experimentally verified that a set median string is useful enough to be used as index of a table entry. Besides, it is less expensive since we do not need to compute a new shape being an exact shape average, but to select it between the shapes composing the cluster.

We first define the polyline $\widetilde{P_C}$ which is the set median string acting as the representative of a certain cluster $C = \{P(R_1)...P(R_n)\}$. It is computed as the string in C with minimum accumulated distance from each other $P(R_i)$ in C as follows:

$$\widetilde{P_C} = arg\min_{\widetilde{P_C}\in C}\left(\sum_{i=1}^{n}\delta(P(R_i), \widetilde{P_C})\right) \tag{5}$$

Then, when a new polyline $P(R_i)$ has to be added to the lookup table, the selection of the cluster to which it belongs is done by applying the string matching algorithm proposed above. We select the cluster where the cost of editing the string $P(R_i)$ to match the set median string $\widetilde{P_C}$ is lower than a threshold thr. We add $P(R_i)$ to this cluster. The set median string $\widetilde{P_C}$ of the corresponding cluster is recomputed in order to keep offering a good cluster representative. If no cluster has a set median string similar to $P(R_i)$ we define a new cluster.

The set median strings act as indexing keys of the lookup table where at each entry a list of translation vectors $\overrightarrow{v_i} = (x_i, y_i)$ where (x_i, y_i) are the coordinates of the middle point of the polyline $P(R_i)$ are stored. We can see the details in the Algorithm 1:

Algorithm 1 Build a LUT from a list of primitives.

```
for i = 0 to length(R) do
  if LUT[P(R_i)] is not NULL then
    LUT[P(R_i)].AddValue(v_i)
    LUT[P(R_i)].UpdateKey()
  else
    LUT[P(R_i)].CreateNewPos(v_i)
  end if
end for
```

When applying the algorithm described above, the order followed to add polylines in the LUT is important and the primitive clustering can be influenced by this fact. However as in spotting applications the user can add more and more documents to the database at any time, we preferred to use an incremental primitive clustering than a classical classification method which would need a learning stage, making it difficult to increase the data collection. In addition, the coarse primitive clustering offered by the LUT is compensated by the use of a voting scheme.

4.2 Querying Symbols: Activating Table Entries

Given a symbol $S = \{P(R_1)...P(R_p)\}$ to query and the lookup table containing q entries, a maximum of p table entries are activated resulting on the one hand in a list of locations where cast votes and on the other hand in a list of vote values. A table entry represented by a certain median string $\widetilde{P_j}$ is activated depending on the following condition:

$$\begin{array}{l} \delta(P(R_i), \widetilde{P_j}) < thr \\ \text{where} \;\; 1 \leq i \leq p \;\; \text{and} \;\; 1 \leq j \leq q \end{array} \quad (6)$$

The values of the votes at each activated entry are proportional to every $\delta(P(R_i), \widetilde{P_j})$. Symbol detection is then performed as a voting procedure in terms of the indexation over the lookup table. The locations where there is a presence of most of the polylines composing the symbol S form clusters of coherent votes. The presence of similar polylines in other locations of the line drawing provokes false positive votes which are scattered into the voting space.

4.3 Voting Scheme

The voting space is a three dimensional space (x, y, s) consisting of $2D$ position coordinates and a scale ratio. Given a query string P_q we accumulate votes in the translation coordinates $\overrightarrow{v_i} = (x_i, y_i)$ of the line drawing image. The third dimension of this space represents the scale factor between the query polyline and the polylines stored in the LUT. This voting scheme formulates hypothesis of position and scale of the queried symbol. The zones of the scanned line drawing where we find similar region contours at a similar scale that the ones that form the query symbol tend to accumulate more votes and thus to form clusters in the voting space. The problem of finding zones where a symbol is likely to be found is then reduced to a local maxima localization problem in the voting space.

For the sake of simplicity the voting space is split into several bins, $I_{(1,1,1)}...I_{(m,n,s)}$ named buckets. The bin size has to be related to the scale of the symbol so the different votes fall in nearby buckets. In our experiments the grid size has been empirically set and is determined in terms of the size of the original image. In our case, m and n are determined such as $\max(m, n) = 128$, preserving the aspect ratio of the original image. The scale dimension is sampled to $s = 8$ possible buckets. Following a similar idea than the proposed by Lorenz and Monagan in [24] we use a voting method known in signal processing as anti aliasing to relate $(\overrightarrow{v_i}, s)$ to a set of I_j neighboring buckets, based on the Euclidean distance between the voting location and the discrete bin partition buckets. Each $(\overrightarrow{v_i}, s)$ has the edit cost vote to distribute among its eight neighboring buckets, depending on their proximity.

Given a symbol S, the activation of the lookup table entries results on a list $L = \{(\overrightarrow{v_1}, s_1)...(\overrightarrow{v_n}, s_n)\}$ of translation vectors. Being $d((\overrightarrow{v_i}, s_i), I_j)$ the Euclidean distance between $(\overrightarrow{v_i}, s_i)$ and one of the eight neighboring buckets I_j we define the value of the vote $V(I_j)$ received in the bucket I_j is accumulated as:

$$V(I_j) = V(I_j) + \frac{w_1}{d((\overrightarrow{v_i}, s), I_j)} + \frac{w_2}{\delta(P(R_i), \widetilde{P_j})} \quad (7)$$

Where w_1 and w_2 weight the distance factor and the edit cost. We can see an example of the voting distribution scheme in Fig. 2.

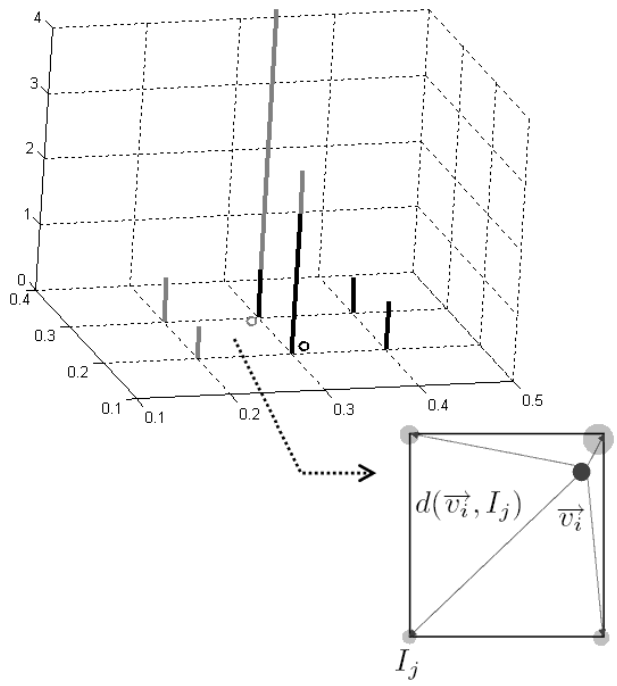


Fig. 2 Anti-aliasing method to cast votes. Even if two votes fall in different buckets due to the discretization, they still contribute to form coherent peaks in the desired values of the voting space.

As we can see, the anti aliasing method reduces the problem to work with a discrete grid to distribute votes.

Voting schemes are only efficient if a high number of votes fall in the right bin, so that the bin is easily detected among the background noise. If some votes fall in the neighboring bins, the significance of the correct bin decreases. Since the votes are now distributed among nearby buckets, even if the locations of a symbol do not fit a unique bin, the votes of close buckets collaborate between them.

Since our framework is a retrieval by query process, given a query symbol, the top k zones of interest in terms of accumulated votes are retrieved and returned to the user. The more primitives a symbol has, the more votes can be accumulated in a given zone. However, since only one query is done at the same time, there is no need to normalize the votes to retrieve the zones of interest.

5 Experimental Results

In order to evaluate the proposed spotting methodology we present three different experiments. The first one only focuses on the string matching algorithm as a distance measure between polygonal shapes. It aims to empirically determine a well suited *thr* value. The second experiment is designed to test if the primitives proposed to represent a graphical symbol are sufficiently discriminative. Finally, the third experiment tests the symbol spotting method by querying a document image database of real architectural floor-plans.

5.1 Silhouette Shape Matching

The first experiment is designed to test the efficiency of the string matching algorithm as a shape descriptor. The algorithm is used as a distance between two shapes represented by a polygonally approximated contour. This experiment also aims to empirically determine a well suited value of the threshold *thr* which determines whether two polylines are similar or not. We used a subset of isolated silhouette shapes from the MPEG-7 core experiment described by Latecki *et al.* in [25].

For each of the 15 shape models shown in Fig. 3, the noise model presented by Kanungo *et al.* in [26] is applied to generate 300 degraded images per class, which are then polygonally approximated. We can also see in Fig. 3 the variation of the number of segments per class. With all this dataset we run a classification experiment. The distance between each model and the 4500 vectorized shapes is computed by using the cyclic string matching. These results are sorted by increasing distance to extract a Receiver Operating Characteristic (ROC) curve (the interested reader is referred to paper by Fawcett [27] on ROC analysis), which plots the true positive rates against the false positive rates.

We can appreciate in Fig. 4 the tradeoff between the correctly classified items and the appearance of false positives. In our framework, as we use a voting scheme to accumulate evidences, we are more interested in achieving high true positive rates values rather than having low false positive rates. We can find in Table 1 the obtained false positives rates and thresholds for different true positives rates.

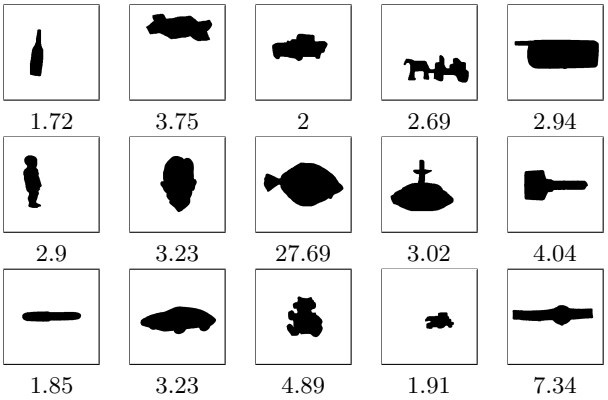


Fig. 3 Selection of the MPEG database and the number of segments variation per class.

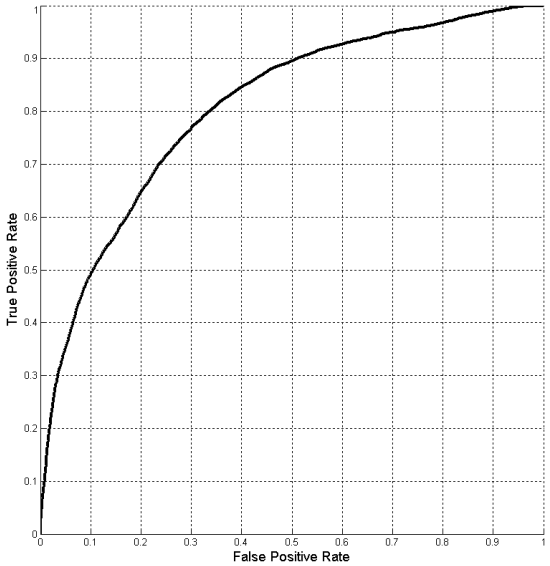


Fig. 4 Receiver operating characteristic curve for the silhouette matching experiment.

Table 1 Obtained false positive rates and decision threshold *thr* for several true positive rates.

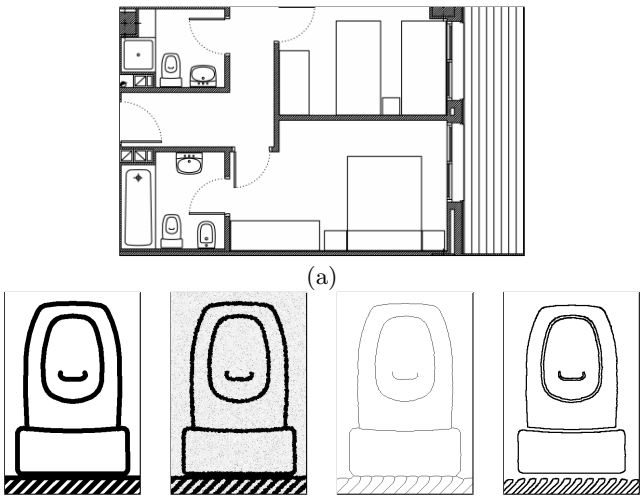
True Positive rates	False Positive rates	*thr*
0.25	0.025	0.008
0.5	0.105	0.014
0.75	0.281	0.024
0.9	0.511	0.035

In the presented spotting method the LUT offers a coarse clustering that is then refined by the use of a

voting scheme. The presence of false positives in a LUT entry is not a problem but we want to minimize the missed primitives. In our experiments, we used a *thr* value of 0.03 which guarantees about a 75% of correctly clustered shapes in the LUT. False positives appear but the voting strategy hopefully discards them.

5.2 Contours vs. Skeletons

The second test aims to see if the region contours are suitable primitives to represent a graphical symbol. We can find in the literature mainly two strategies to vectorize graphical documents. One which is based on a skeleton extraction followed by a polygonal approximation and another one which approximates the extracted contours. We compare the performance of the presented method by using both vectorization strategies. To carry this experiment, a real floor-plan has been degraded to build a collection of 500 synthetically distorted plans again by using the noise method of Kanungo *et al.*. These distorted images are then polygonally approximated with both representations: contours and skeleton primitives. We can appreciate the differences between both primitives in Fig. 5(d) and 5(e).



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

Fig. 5 Symbol Primitive Representations. (a) Model floor-plan. (b) Zoom of the toilet symbol. (c) Degraded image. (d) Symbol with skeleton primitives. (e) Symbol with contour primitives.

In Fig. 6 we can see a precision and recall graph (the interested reader is referred to van Rijsbergen book [28] on information retrieval) showing that the selected primitives are more expressive since the spotting method using this representation outperforms the skeleton in all cases. In average there is a gain near a 17.5% of precision for the same recall values. Details are shown in Table 3, where we can see the number of false positives we have when requesting a certain number of the 500 possible solutions. In addition, using contours as primitives, in only 5 of the 500 images the queried symbol has been missed and using the skeleton we miss the symbol in 73 of the 500 images. This yields to a significant gain in the recall value when using contours instead of approximating the symbol skeleton.

Table 2 Number of false positives when requesting a certain number of retrieved zones.

Primitives	**Retrieved Items**			
	200	300	400	475
False positives with Contours	6	7	29	159
False positives with Skeletons	73	76	89	273

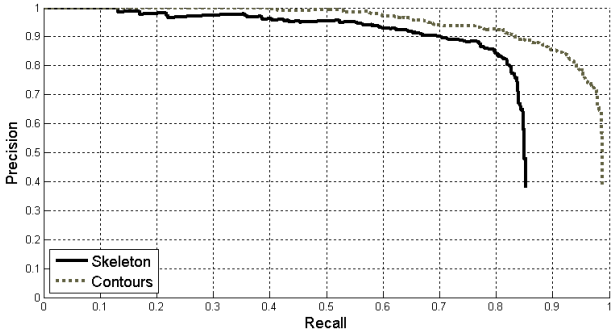


Fig. 6 Precision and recall plot when spotting the toilet symbol shown in Fig. 5 using two different symbol primitives. The contours outperforms in both precision and recall the skeleton primitives.

5.3 Symbol Spotting in a Document Database

Finally, we tested our method with a collection of ten real floor-plans and ten different symbols as queries. The query symbols appear in the floor-plans several times and are segmented by cropping a zone in the floor-plan image and vectorizing it. Each floor-plan has been polygonally approximated and ground-truthed. The database consist on approximately 14200 polylines which after the LUT construction result in near 320 table entries. We can see in Fig. 7 the precision and recall plot resulting from spotting these symbols in the whole floor-plan database.

In Table 3 we present a detailed set of measures to evaluate the performance of retrieval systems which aim to evaluate the spotting architecture. As we can see, the recall ratio is quite good. However there is an important number of false positives in the results which harm

Table 3 Detailed retrieval measures for each model symbol for the symbol spotting in a document database experiment.

Symbol		**Retrieval Measures**				
Class	p	Precision (%)	Recall (%)	F-score (%)	AveP (%)	Time (secs./plan)
Bidet	4	30.8	100	47.1	87.5	0.76
Chair	5	36.8	100	53.8	83.3	0.64
Burners	9	5.1	100	9.6	59.1	1.09
Toilet	5	50	37.5	42.9	27.1	0.98
Toilet sink	5	30	100	46.2	68.7	1.89
Kitchen sink	5	11.8	50	19	33.3	1.16
Single sofa	4	37.5	100	54.6	100	0.43
Double sofa	6	15	75	25	65	0.22
Table	7	16.7	100	28.6	100	0.24
Tv set	4	20	100	33.3	95	0.12
AVERAGE	5.4	25.4	86.2	36	71.9	0.75

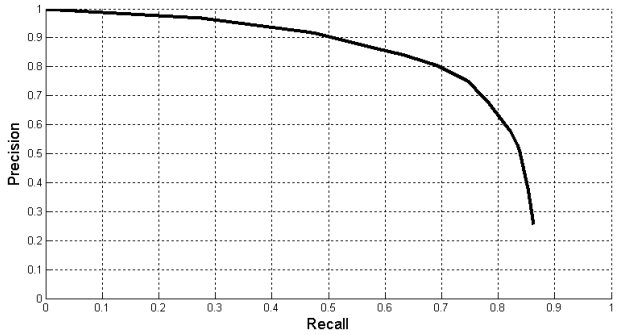


Fig. 7 Precision and recall plot for symbol spotting in the document database.

the precision value. The F-score is a composite measure which aims to rank the results. However, the most interesting point here is to notice the difference between the precision and the average precision AveP values. The average precision is a measure of quality which rewards the earliest return of relevant items. As we can see, even if in our experiments the precision values are quite low, the average precisions are significantly higher. That means that usually the false positives are ranked worst than the correct results, as we can also see in the qualitative results shown in Fig. 8. Finally, we also show the average time taken by our software prototype to spot a symbol per plan. It is remarkable that usually the symbols which are composed by common simple primitive shapes (circles, squares, etc.) are the ones which are more time consuming since the entries of the LUT are more populated and more hypothesis have to be considered. No significant differences due to the number of polylines composing a symbol can be appreciated.

6 Conclusions

In this paper we have presented a method to spot graphical symbols in scanned technical documents. First a suitable symbol representation as a set of closed region contours and its codification with attributed strings has been presented. The distance definition using a cyclic string matching algorithm allows to tolerate the segment fragmentation problem. Then, a clustering of salient zones of interest and a voting method have been presented and tested to spot symbols in real technical line drawings.

The experiments show that the representation and distance approaches are able to tackle with the inherent noise arising from the scanning process and the distortions introduced by the raster-to-vector algorithms. The presence of false positives is not a critical problem since the purpose of spotting methods is to find by a fast technique a coarse identification of zones where a given symbol appears. Finally, we can see that the use of voting strategies are of vital importance for spotting problems. To reach higher precision one can use better shape descriptors, however this also entails a complexity increment. The accumulation of evidences allows to work with a coarsely recognition in the indexing step.

However the presented method still suffers from some drawbacks that must be improved in future work. The order followed to add polylines in the LUT is determinant and in some cases could lead to some misclassifications. However for spotting applications where the user can add more and more documents at any time, the primitive clustering must be incremental. The use of incremental classifiers such as iPCA [29] or iLDA [30] applied to primitive clustering should be studied. Another concern is about the primitives describing a symbol. The symbols of some other kind of line-drawings such as electronic diagrams may have no loops and thus they could not be indexed by the presented method. In addition, the use of a voting scheme only has sense if the symbols are formed by several primitives and then several hypothesis can be casted in a given location. For symbols consisting of a low number of regions, other describing features must be taken into account. The use

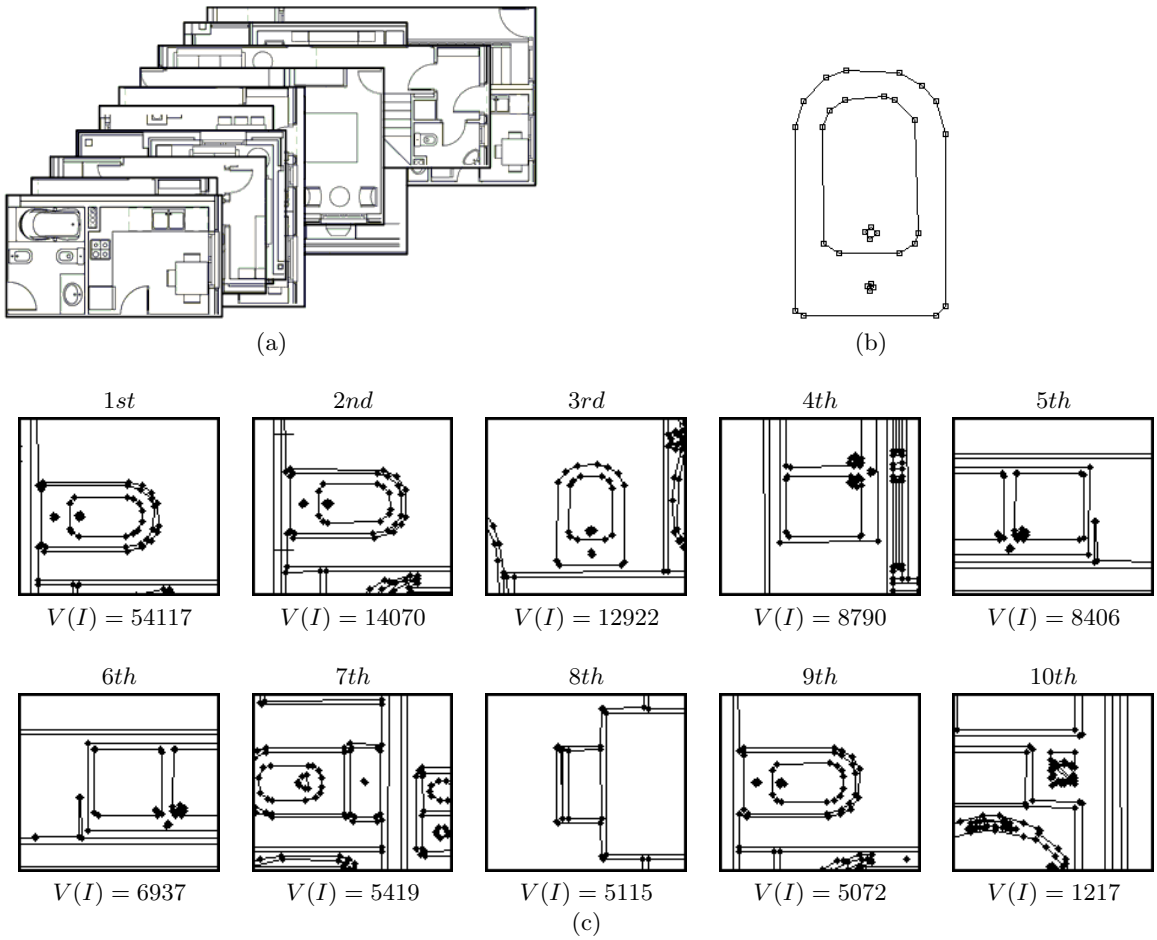


Fig. 8 Qualitative results for symbol spotting by cyclic string matching. (a) Vectorized floor-plan database. (b) Query example. (d) Ranked top ten results.

of other primitives as for instance the chain points used by Zuwala and Tabbone in [31] should be considered in order to use our spotting methodology to technical documents from domains other than the architectural. Finally, the presented matching approach can not cope with occlusions which will provoke the polylines to be broken. A partial matching algorithm as presented in [32] by Tănase *et al.* could be helpful in such situations.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful and constructive comments as well as the architect Enric Farrerons for providing the floor-plan images and Silvia Sánchez for proofreading the manuscript. This work has been partially supported by the spanish projects TIN 2006-15694-C02-02 and CONSOLIDER - INGENIO 2010 (CSD 2007-00018).

References

1. Tombre K, Lamiroy B (2008) Pattern recognition methods for querying and browsing technical documentation. In: Progress in Pattern Recognition, Image Analysis and Applications, LNCS 5197. pp. 504–518. DOI 10.1007/978-3-540-85920-8_62
2. Rath T, Manmatha R (2003) Word image matching using dynamic time warping. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 521–527. DOI 10.1109/CVPR.2003.1211511
3. Kuo S, Agazzi O (1994) Keyword spotting in poorly printed documents using pseudo 2-D Hidden Markov Models. IEEE Trans Pattern Anal Mach Intell 16(8):842-848. DOI 10.1109/34.308482
4. Lu S, Tan CL (2008) Retrieval of machine-printed Latin documents through word shape coding. Pattern Recogn 41:1799–1809. DOI 10.1016/j.patcog.2007.10.017
5. Cordella L, Vento M (2000) Symbol recognition in documents: a collection of techniques? International Journal on Dcument Analysis and Recognition 3(2):73–88. DOI 10.1007/s100320000036
6. Lladós J, Valveny E, Sánchez G, Martì E (2002) Symbol Recognition: Current Advances and Perspectives. In: Graphics Recognition Algorithms and Applications, LNCS 2390. pp. 104–128. DOI 10.1007/3-540-45868-9_9
7. Califano A, Mohan R (1994) Multidimensional indexing for recognizing visual shapes. IEEE Trans Pattern Anal Mach Intell 16(4):373-392. DOI 10.1109/34.277591

8. Ballard DH (1981) Generalizing the Hough transform to detect arbitrary shapes. Pattern Recogn 13(2):111-122.
9. Lamdan Y, Wolfson HJ (1988) Geometric hashing: A general and efficient model-based recognition scheme. In: Proc of the Second International Conference on Computer Vision. pp. 238–249.
10. Cohen S, Guibas LJ (1997) Shape-based image retrieval using geometric hashing. In: Proc. of the ARPA Image Understanding Workshop. pp. 669–674.
11. Lamiroy B, Gros P (1996) Rapid object indexing and recognition using enhanced geometric hashing. In: Proc. of the fourth European Conference on Computer Vision. pp. 59–70. DOI 10.1007/BFb0015523
12. Stein F, Medioni G (1992) Structural indexing: Efficient 2D object recognition. IEEE Trans Pattern Anal Mach Intell 14(12):1198-1204. DOI 10.1109/34.177385
13. Loce RP, Dougherty ER (1997) Enhancement and restoration of digital documents: Statistical design of nonlinear algorithms. Society of Photo-Optical Instrumentation Engineers (SPIE), Bellingham USA.
14. Tombre K, Ah-Soon C, Dosch P, Masini G, Tabbone S (2000) Stable and robust vectorization: How to make the right choices. In: GREC '99: Selected Papers from the Third International Workshop on Graphics Recognition, Recent Advances, LNCS 1941. pp. 3–18. DOI 10.1007/3-540-40953-X_1
15. Rosin PL, West GA (1989) Segmentation of edges into lines and arcs. Image Vision Comput 7(2):109–114. DOI 10.1016/0262-8856(89)90004-8
16. Wolfson HJ (1990) On curve matching. IEEE Trans Pattern Anal Mach Intell 12(5):483–489. DOI 10.1109/34.55108
17. Kaygin S, Bulut MM (2002) Shape recognition using attributed string matching with polygon vertices as primitives. Pattern Recogn Lett 23:287–294. DOI 10.1016/S0167-8655(01)00111-8
18. Wagner R, Fischer M (1974) The string-to-string correction problem. J Assoc Comput Mach 21(1):168–173. DOI 10.1145/321796.321811
19. Tsay W, Yu S (1985) Attributed string matching with merging for shape recognition. IEEE Trans Pattern Anal Mach Intell 7(4):453-462.
20. Maes M (1990) On a cyclic string-to-string correction problem. Inform Process Lett 35:73–78. DOI 10.1016/0020-0190(90)90109-B
21. Arkin EM, Chew LP, Huttenlocher DP, Kedem K, Mitchell JSB (1991) An efficiently computable metric for comparing polygonal shapes. IEEE Trans Pattern Anal Mach Intell 13(3):209–216. DOI 10.1109/34.75509
22. Tsay YT, Tsai WH (1989) Model-guided attributed string matching by split-and-merge for shape recognition. Int J Pattern Recogn Artif Intell 3(2):159–179. DOI: 10.1142/S0218001489000140
23. Sánchez G, Lladós J, Tombre K (2002) A mean string algorithm to compute the average among a set of 2D shapes. Pattern Recogn Lett 23:203–213. DOI 10.1016/S0167-8655(01)00122-2
24. Lorenz O, Monagan G (1995) A retrieval system for graphical documents. In: Symposium on Document Analysis and Information Retrieval. pp. 291-300.
25. Latecki L, Lakämper R, Eckhardt U (2000) Shape descriptors for non-rigid shapes with a single closed contour. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 424–429. DOI 10.1109/CVPR.2000.855850
26. Kanungo T, Haralick RM, Baird HS, Stuezle W, Madigan D (2000) A statistical, nonparametric methodology for document degradation model validation. IEEE Trans Pattern Anal Mach Intell 22(11):1209–1223. DOI 10.1109/34.888707
27. Fawcett T (2006) An Introduction to ROC Analysis. Pattern Recogn Lett 27(8):861–874. DOI 10.1016/j.patrec.2005.10.010
28. van Rijsbergen CJ (1979) Information Retrieval. Butterworths, London.
29. Artač M, Jogan M, Leonardis A (2002) Incremental PCA for on-line visual learning and recognition. In: Proc. of the International Conference on Pattern Recognition. pp. 781–784. DOI 10.1109/ICPR.2002.1048133
30. Pang S, Ozawa S, Kasabov N (2005) Incremental linear discriminant analysis for classification of data streams. IEEE Trans Systems, Man and Cybernetics 35(5):905–914. DOI 10.1109/TSMCB.2005.847744
31. Zuwala D, Tabbone S (2006) A Method for Symbol Spotting in Graphical Documents. In: Proc. of the 7th Workshop on Document Analysis Systems, LNCS 3872. pp. 518–528. DOI 10.1007/11669487_46
32. Tănase M, Veltkamp R, Haverkort H (2005) Multiple Polyline to Polygon Matching. In: Proc. of the 16th International Symposium ISAAC05, LNCS 3872. pp. 60-70. DOI 10.1007/11602613_8

Marçal Rusiñol received his B.Sc. and his M.Sc. degrees in Computer Sciences from the Universitat Autònoma de Barcelona (UAB), Barcelona, Spain, in 2004 and 2006, respectively. In 2004 he joined the Computer Vision Center where he is currently pursuing the Ph.D. degree under the supervision of Dr. Josep Lladós. He is also a Teaching Assistant at the Computer Sciences Department of the Universitat Autònoma de Barcelona. His main research interests include Graphics Recognition, Structural Pattern Recognition and Performance Evaluation.

Josep Lladós received the degree in Computer Sciences in 1991 from the Universitat Politècnica de Catalunya and the PhD degree in Computer Sciences in 1997 from the Universitat Autònoma de Barcelona (Spain) and the Université Paris 8 (France). Currently he is an Associate Professor at the Computer Sciences Department of the Universitat Autònoma de Barcelona and a staff researcher of the Computer Vision Center, where he is also the director. He is the head of the Pattern Recognition and Document Analysis Group (2005SGR-00472). His

current research fields are document analysis, graphics recognition and structural and syntactic pattern recognition. He has been the head of a number of Computer Vision R+D projects and published several papers in national and international conferences and journals. J. Lladós is an active member of the Image Analysis and Pattern Recognition Spanish Association (AERFAI), a member society of the IAPR. He is currently the chairman of the IAPR-ILC (Industrial Liaison Committee). Formerly he served as chairman of the IAPR TC-10, the Technical Committee on Graphics Recognition, and also he is a member of the IAPR TC-11 (reading Systems) and IAPR TC-15 (Graph based Representations). He serves on the Editorial Board of the ELCVIA (Electronic Letters on Computer Vision and Image Analysis) and the IJDAR (International Journal in Document Analysis and Recognition), and also a PC member of a number of international conferences. He was the recipient of the IAPR-ICDAR Young Investigator Award in 2007. Josep Lladós has also experience in technological transfer and in 2002 he created the company ICAR Vision Systems, a spin-off of the Computer Vision Center working on Document Image Analysis, after win the entrepreneurs award from the Catalonia Government on business projects on Information Society Technologies in 2000.

Gemma Sánchez received the degree in Computer Sciences in 1994 from the Universitat Politècnica de Catalunya and the PhD degree in Computer Sciences in 2001 from the Universitat Autònoma de Barcelona (Spain) and the Université Henry Poincar (France). Currently she is a lecturer at the Computer Sciences Department of the Universitat Autònoma de Barcelona and a staff researcher of the Computer Vision Center where she has participated in a number of industrial and research projects. Her current research interests are document analysis, graphics recognition and structural and syntactic pattern recognition. She is now working in the domain of visual languages for graphics representation and recognition and document image analysis. She has published several papers in national and international conferences and journals. She is a member of the IAPR TC-10 (Technical Committee on Graphics Recognition).