Robust symbol localization based on junction features and efficient geometry consistency checking

The-Anh Pham, Mathieu Delalandre, Sabine Barrat and Jean-Yves Ramel
Laboratoire d'Informatique 64, Avenue Jean Portalis, 37200 Tours - France.
the-anh.pham@etu.univ-tours.fr,{mathieu.delalandre, sabine.barrat, ramel}@univ-tours.fr

***Abstract*—This paper presents a new approach for symbol localization in line-drawing images using junction features and geometric consistency checking. The proposed system first detects junction points, and then characterizes them by very compact, distinctive, and varying-length descriptors. The detected junctions are used to decompose a document image into a set of *smooth primitives* composing of isolated shapes (e.g., isolated circles and straight lines) and curve segments bounded between either two junctions or a junction and an end-point. These primitives are then associated with a new set of keypoints to form a complete and compact representation of document images. Next, keypoint matching is performed to find the correspondences among the keypoints of the query and those of database documents. The obtained matches are finally refined by a new and efficient algorithm to deal with the problem of geometric consistency checking. Our experiments shown that the proposed system is very time- and memory-efficient, and provides high accuracy rate of symbol localization.**

I. Introduction

The problem of using symbol information is an intensive research activity in the Document Image Analysis (DIA) and the graphic recognition communities. In line drawings, a symbol can be defined as a graphical entity with particular meaning in the context of specific application domain. Symbols can serve in different applications including document re-engineering, understanding, classification and retrieval. Earlier works on symbols were focused on the problem of symbol recognition, that can be considered as a particular application of the general problem of pattern recognition. Several comprehensive surveys [8] review the existing works of symbol recognition on logical diagrams, engineering drawings and maps. Comparative results have been reported throughout a series of symbol recognition contests, which concern with the aspects of performance evaluation [3]. Over the past decade, the interest has moved towards the symbol spotting problem. Symbol spotting can be viewed as a way to efficiently localize possible symbols and limit the computational complexity, without using full recognition methods. In this sense, symbol spotting works like a CBIR system.

A common problem of any symbol processing systems, recognition or spotting, is localization or detection of the symbols. Symbol localization can be defined as the ability of a system to localize the symbol entities in the complete documents. It could be embedded in the recognition/spotting method [10] or works as a separated stage in a two-step system [13]. The approaches used for localization are similar for recognition and spotting. All systems rely first on a primitive extraction step (e.g., connected components, loops, key-points, lines, etc.). These systems differ mainly in the way that the detected primitives are processed, using machine learning or retrieval and indexing techniques. Different approaches have been investigated in the literature to deal with the localization problem.

One of the earliest approach employed in many systems is subgraph matching. Graph is a very effective tool to represent line drawings. Attributed Relational Graphs (ARGs) can be used to describe the primitives, their associated attributes and interconnections. However, subgraph isomorphism is known to be a NP-hard problem, making it difficult to use the graph for large images and document collections, despite the approximate solutions of subgraph isomorphism developed in the literature [2], [9]. In addition, subgraph isomorphism remains very sensitive to the robustness of the feature extraction step, as any wrong detection can result in strong distortions in the ARGs.

An alternative approach to subgraph matching is framing [4], [5], [7]. These techniques involve with the decomposition of the image into frames (i.e., tiles, buckets, windows) in which the frames could be overlapped [7] or disjointed [4], [5]. Local signatures are computed from the primitives contained in the frames and matched to identify the candidate symbols. The size of the frames can be determined based on the symbol models [4], [7] or set at different resolutions [5]. In this way, framing is not scale invariant as the size of the frames cannot be dynamically adapted. The position of frames can be set with a grid [4], [5] or by sliding [7]. Sliding could be performed by steps to reduce the entire processing time [7], as any computations with overlapping incur a polynomial complexity.

Due to the different problems discussed above, a common way to deal with localization is the use of a triggering mechanism. Such a system looks for some specific primitives in line drawing images and triggers a matching process at the symbol level within the Regions of Interest (ROIs) around these primitives. The system in [11] is an typical example. In this work, given a query symbol, the keypoints (i.e., Difference of Gaussian features) and its corresponding vocabularies are computed and used to find the matched keypoints from the database documents. For each pair of matched keypoints, the local scales and orienations extracted at the keypoint in the query symbol are used to generate the ROI in the document that probably contains the instance of the symbol. Because the number of detected keypoints would be very large and the local scale computed at each keypoint could be far from satisfaction, the ROI extraction step is thus fragile and time-consuming. Triggering mechanisms have been also developed from graph-based representations, as in [13], [14]. These proposed systems work from the ARGs, where the structures and attributes of the

graphs are exploited to identify the ROIs. In [14], the ROIs are obtained from the maximum and minimum coordinates of adjacent lines. To deal with the error-prone introduced in the vectorization process, the ARGs are extracted from low resolution images and processed by a contraction step. The authors in [13] apply a scoring process in graphs to look for specific attributes of nodes (e.g., small and perpendicular segments). Scores are propagated through the loops of shortest length in the graph. Triggering-based localization is very sensitive to robustness of the mechanism in that any missed detection at the triggering level will result in the failure of symbol localization.

In this work, we present a new approach for symbol localization using junction features and geometric consistency checking. The junction points are first detected and characterized into different types such as T-, L-, and X-junctions. Using the detected junctions, we decompose a document image into a set of smooth primitives, composing of isolated shapes (e.g., isolated circles and straight lines) and curve segments bounded between either two junctions or a junction and an end-point. These primitives are then associated with a new set of keypoints including Line-, Arc-, and Circle-keypoints. The obtained keypoints, in combination with the junction points and end-points, form a complete and compact representation of document images. Next, keypoint matching is performed to find the correspondences among the keypoints of the query and those of database documents. Finally, geometric consistency checking is applied to the obtained matches using a new and efficient algorithm, which is designed to work on our specific keypoints. For the rest of this paper, we describe the details of the proposed approach in Section 2. Experimental results are investigated in Section 3. Key remarks and future works are given in Section 4.

II. The Proposed Approach

A. Detection of junction points

As discussed in [12], most of the well-known techniques for junction detection are vectorization-based systems. Such methods rely on vectorization, known to be sensitive to setting parameters, and presenting difficulties when heterogeneous primitives (e.g., straight lines, arcs, curves and circles) appear within a same document. Knowledge about the document content must be included, making the systems less adaptable to heterogeneous corpus. In this work, the junction points are detected by using our previous work in [12]. For completeness, we describe the main idea of the junction detector as follows. We directly address the problem of junction detection by finding the optimal meeting points of median lines. At first glance, it seems that our approach would directly encounter the well-known problem of junction distortion. However, it is important to note that, apart from crossing zones or distorted zones (i.e., the areas where several line segments meet), the median lines are known to be very representative for the rest of line segments. This point suggests that if we can successfully remove the distorted zones, the remaining disjointed strokes would be not subjected to the problem of junction distortion. We therefore present a new algorithm to precisely detect and conceptually remove the distorted zones. The remaining line segments are then locally characterized to form structural representations of the crossing zones. Finally, the junction points are reconstructed by two-step process: clustering and optimizing. The clustering step clusters the characterized line segments into different groups based on their topological constraint, and the optimizing step looks for the best position of junction by minimizing the distance errors of the clustered line segments. Figure 1 shows the detected junctions for few noised symbols.

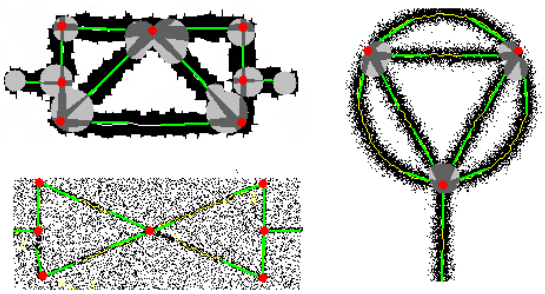

Fig. 1. The detected junctions (small red dots) for few noised symbols.

B. Junction characterization and matching

The detected junctions are characterized and classified into different types such as T-, L-, X-junction. More generally, we wish to characterize any complicated junctions in the same manner based on the arms forming the junction. In our case, as each junction point is constructed from the local line segments of one group, we can consider these line segments as the arms of the junction point. Given a detected junction J associated with a set of m arms $\{U_iV_i\}_{i=0,\ldots,m-1}$, the characterization of this junction is described as $\{p, s_p, \{\theta_i^p\}_{i=0}^{m-1}\}$, where:

- p is the location of J;
- s_p is a local scale computed as the mean length of the arms of J;
- θ_i^p is the difference in degrees between two consecutive arms U_iV_i and $U_{i+1}V_{i+1}$. These parameters $\{\theta_i^p\}_{i=0}^{m-1}$ are tracked in the counterclockwise direction and the θ_{m-1}^p is the difference in degrees between the arms $U_{m-1}V_{m-1}$ and U_0V_0.

It is noticed that the description of each junction point as discussed above is very compact and distinctive. The dimension of this descriptor is limited up to the number of arms of each junction point and in practice, this value is quite small (*e.g.* 3 for a T-junction, 4 for a X-junctions). This point constitutes a great advantage for the detected junctions which supports for the subsequent task of junction matching in very efficient manner. In addition, the junction descriptor is distinctive (i.e., symbolic description) and general that we can describe any junction points appearing in a variety of complex and heterogeneous documents. Given two junction points characterized as $\{p, s_p, \{\theta_i^p\}_{i=0}^{m_p-1}\}$ and $\{q, s_q, \{\theta_j^q\}_{j=0}^{m_q-1}\}$, the information of junction location and junction scale is used to quickly refine the matches to be described later, and the rest is used to computed a similarity score, $C(p, q)$, of matching two junctions p and q as follows:

$$C(p,q) = \max_{i,j}\{\frac{1}{H}\sum_{k=0}^{h-1} D(\theta^p_{(i+k) \bmod m_p}, \theta^q_{(j+k) \bmod m_q})\} \quad (1)$$

where $h = min(m_p, m_q)$, $H = Max(m_p, m_q)$, and

$$D(\theta^p_i, \theta^q_j) = \begin{cases} 1, & \text{if } |\theta^p_i - \theta^q_j| \leq \theta_{thres} \quad (2) \\ 0, & \text{for otherwise.} \quad (3) \end{cases}$$

The similarity score $C(p,q)$ is in the range $[0,1]$ and θ_{thres} is an angle difference tolerance. Two junctions are matched if their similar score is higher than a threshold: $C(p,q) \geq C_{thres}$. Our investigation shown that a good range of these parameters are following: $\theta_{thres} \in [15, 20]$ and $C_{thres} \in [0.65, 0.75]$. In some specific domains taking object localization for example, a query object or symbol is often embedded into complicated documents. It is therefore common case to see that the query object could be touched with other context information appearing in a document. In such cases, using the similarity score $C(p,q)$ could be too restricted to find corresponding junctions. We therefore release the junction matching step by introducing an addition constraint as follows. Two junctions p and q are matched if their similar score is higher than a threshold, or one *inclusion test* is hold for these two junctions. Here, we consider that the junction p is included in the junction q if there are exact $m_p - 1$ angle matches between the angles of p and q. This implies: $C(p,q) * Max(m_p, m_q) = m_p - 1$. Figure 2 shows the corresponding matches of the junctions detected in a query symbol (left) and those of an image cropped from a big document (right).

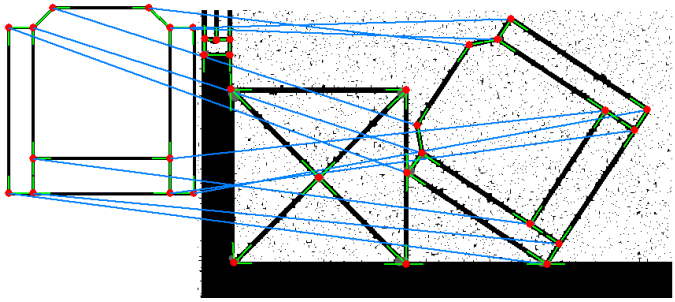

Fig. 2. Corresponding junction matches between a query symbol (left) and a cropped document (right).

C. Keypoint-based representation of document images

The junctions detected in the previous stage are used to decompose a document image into a set of *smooth primitives*. Here, we define the smooth primitives as those composing of isolated shapes (e.g., isolated circles and straight lines) and curve segments bounded between either two junctions or a junction and an end-point. This definition is derived based on the fact that after the process of junction detection, every median line segment bounded between two junctions are sufficiently smooth. Otherwise, some new junction points are likely to be detected on this segment. In this work, we restrict the smooth primitives to three kinds of segment: straight line segment, arc segment, and circle. These basic-shape primitives could be simply derived using the linear least squares fitting technique. Next, each type of these primitives is characterized as a specific keypoint as follows:

- A straight line primitive is represented by a triple $\{p_L, p_{L_1}, p_{L_2}\}$ corresponding to the middle point and two extremity points, respectively. A triple $\{p_L, p_{L_1}, p_{L_2}\}$ is regarded as a Line-type keypoint or *L-keypoint*.
- An arc primitive is represented by $\{p_A, p_{A_1}, p_{A_2}\}$ with the same meaning as that of a straight line primitive. A triple $\{p_A, p_{A_1}, p_{A_2}\}$ is regarded as an Arc-type keypoint or *A-keypoint*. It is noted that the characterization an A-keypoint is proceeded in the same spirit as that of a junction whose two arms are $p_A p_{A_1}$ and $p_A p_{A_2}$.
- A circle primitive is represented by $\{p_C, r_C\}$ corresponding to its centroid and radius. A couple $\{p_C, r_C\}$ is regarded as a Circle-type keypoint or *C-keypoint*.

For completeness, we name the junction points as J-keypoints and end-points as E-keypoints. As a result, a document image is now completely represented by a set of keypoints, composing of the L-keypoints, A-keypoints, C-keypoints, J-keypoints, and E-keypoints. Figure 3 shows a decomposition of a document image into a set of keypoints.

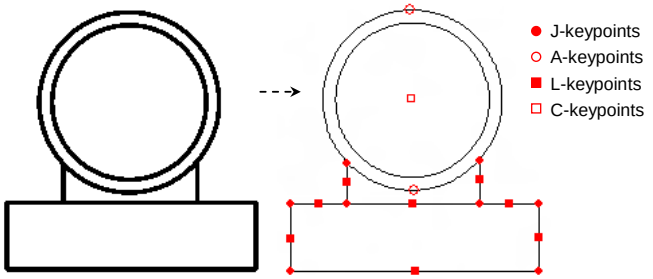

Fig. 3. Keypoint-based representation of a simple document image.

D. Symbol matching and localization

Given a query symbol Q and a database document D, their keypoints are first detected and characterized as described in the previous sections. Next, keypoint matching is performed to find the correspondences among the keypoints of Q and those of D. Keypoint matching is independently processed for each type of keypoints as follows:

- An A-keypoint is matched with another A-keypoint using the same matching procedure as that of the J-keypoints (i.e., junction matching).
- An E-keypoint (*resp.* C-keypoint, L-keypoint) is always matched with any other E-keypoints (*resp.* C-keypoints, L-keypoints).

The obtained matches are finally verified by checking geometric consistency. This step will remove false matches and cluster the remaining matches into different clusters that each of which indicates an instance of the query symbol. Concerning this problem of geometric consistency checking, two main strategies are often exploited in the literature. The

first strategy treats data (i.e., the matches) in a top-down way. One typical technique of this strategy is known as RANSAC (RANdom Sample Consensus) [6]. The key idea of RANSAC is to randomly select k matches for estimating a transformation model (typically an affine transformation and thus $k = 2$ or 3). The model is then assigned with a confidence factor, which is calculated as the number of matches fitting well to this model. Next, these steps are repeated a number of times to find the model with the highest confidence. RANSAC is often used to find *a single transformation model* between two images with a high degree of accuracy, provided that the ratio of inliers and outliers occurring in the data is sufficiently high ($\geq 50\%$). However, when this is not the case, it is difficult to use RANSAC. In addition, RANSAC would be time-consuming because the number of iterations is often large to ensure that an optimal solution could be found. The second strategy treats data in a bottm-up manner by performing a voting process starting from all data points, and then finding the parameters (typically composing of 4 paramters: orienation, scaling, and x, y-translatation) corresponding to the dense density areas of support. One typical technique falling this strategy is known as Generalized Hough Transform (GHT) [1]. GHT is most commonly used for the cases where *multiple transformation models* are presence in the data. It is less accurate than RANSAC but very robust to noise even if a large number of outliers are present. However, because GHT requires a proces of parameter quantization, it is subjected to very high cost of memory space $O(M^4)$, time-consuming $O(N^2)$[1], and is sensivtive to the quantization of parameters.

In our case, as each keypoint of the query symbol is often occurred in a database document with a high frequency, the outlier matches are thus significantly greater than the inlier matches. In addition, multiple instances of a query symbol are appeared in the database document, it is therefore not a good idea to use some techniques like RANSAC or GHT because of the aforementioned weaknesses. We therefore present, below, an efficient algorithm to deal with the problem of geometric consistency checking. The proposed algorithm incorporates the advantages of both RANSAC and GHT while avoiding their weaknesses. It requires no parameter quantization and works very efficiently in terms of time complexity and memory space. The basic idea is to directly estimate a geometry model F (i.e., an affine transformation) based on every match formed by a pair of either two L-keypoints or two A-keypoints. This idea is inspired by the fact that a pair of two matched lines (or acrs) provides us with 4 parameters (i.e., orientation, scaling, and x, y-translation) of an affine transformation F. As a result, we need only one match to estimate a model F other than two matches as the cases of GHT and RANSAC. Next, we apply the transformation F to all the keypoints detected on the query Q, resulting in a new set of projected points on the database document D. If there is a sufficiently large overlap among the projected points and the keypoints of D matched with those of Q, the transformation F is accepted and used to localize the position of the corresponding instance of Q on D. Here, we consider two points are overlapped if the distance between them is less than a threshold $\epsilon_{overlap}$. As the number of the L- and A-keypoints of Q are quite small and few real computations are needed, the proposed method is very time-efficient. In particularly, the computation complexity of the proposed method is limited up to a linear order $O(kN_1N_2)$, where N_1 is the number of keypoints of Q, and N_2 is the number of matches corresponding to the L- and A-keypoints of Q, and k is a constant value depending on the threshold $\epsilon_{overlap}$. It is obvious to see that $N_1 << N_2 << N$. Furthermore, with a bit prior knowledge of the dataset, we can quickly prune a large number of matches by setting the lower and upper scales for the query symbol. In this way, the local scales associated to the keypoints are used to prune the matches. It is also noted that there is no need to process all the L- and A-keypoints of Q. In our experiments, we first sort m L- and A-keypoints of Q in a decreasing order of their length and then choose the first $m/2$ keypoints to be processed. As an accepted model F corresponds to an instance of the query, multiple instances of the query are thus successfully detected with respect to the number of accepted models. In order to achieve a high accuracy rate of making an accept/reject decision for F, the decision threshold is empirically determined for each query symbol from a training dataset.

[1] N and M, are the number of matches and sampling bins, respectively.

III. Experimental Results

For performance evaluation of the proposed approach, we have selected the final dataset for symbol spotting in GREC2011[2]. The details of this dataset are described on Table I. We also used the same evaluation metric including Precision (P), Recall (R) and F-score as described in this contest [15].

$$P = \frac{S_{Int}}{S_{Ret}}, R = \frac{S_{Int}}{S_{GT}}, F\text{-}score = 2 \cdot \frac{P \cdot R}{P + R}$$

Where S_{Int} is sum of intersection areas between the bounding boxes returned by the spotting system and that of ground-truth, and S_{Ret} is sum of areas of the bounding boxes returned by the spotting system, and S_{GT} is sum of areas of the bounding boxes in ground-truth. It is worth pointing that this metric does not take into account the overlapping areas of the bounding boxes detected by the system. Consequently, taking one examples as shown in Figure 4 where one system returns three bounding boxes with the same size at the same location, then we obtain a perfect score (1.0) for both Precision and Recall. In order to alleviate this matter, we have set a strict constraint for our system in that the ratio of overlapping area and union area of any two retrieved bounding boxes is alway smaller than a threshold (10% in our implementation).

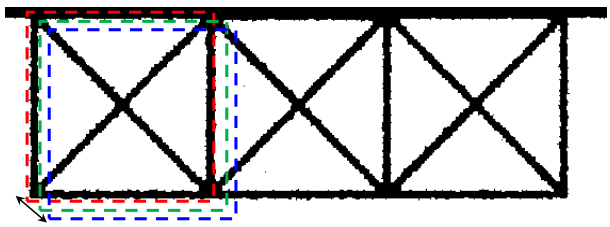

Fig. 4. The case where Precision and Recall obtain a perfect score (1.0).

The detailed results of our system are reported on Table II including Precision, Recall, F-score, maximum ratio between overlapping area and union area of any two bounding boxes retrieved by our system, and mean processing time for a query done an input image. General speaking, the proposed system

[2] http://mathieu.delalandre.free.fr/projects/sesyd/symbols/isrc2011.html

TABLE I. DATASET USED FOR SYMBOL SPOTTING IN GREC2011.

Test Set	Models	Images	Queries	Symbols	Noises	Image Size
Elec1.	21	20	118	246	Ideal	Min:1700x1600
Elec2.	21	20	127	274	Level 1	
Elec3.	21	20	114	237	Level 2	Max:4400x2100
Elec4.	21	20	156	322	Level 3	
Archi1.	16	20	247	633	Ideal	Min:2300x2500
Archi2.	16	20	245	597	Level 1	
Archi3.	16	20	245	561	Level 2	Max:5400x2900
Archi4.	16	20	249	593	Level 3	

TABLE II. EXPERIMENTAL RESULTS OF OUR SYSTEM (%).

Test Set	Precision	Recall	F-Score	Max $\frac{Overlap}{Union}$	Mean Time (ms)
Elec1.	0.85	0.84	0.84	0.00 %	723.38
Elec2.	0.86	0.79	0.82	0.00 %	677.83
Elec3.	0.92	0.81	0.86	5.76 %	655.39
Elec4.	0.80	0.81	0.80	5.76 %	1097.05
Archi1.	0.91	0.96	0.93	6.69 %	1521.47
Archi2.	0.91	0.92	0.92	9.92 %	1341.90
Archi3.	0.94	0.89	0.92	9.92 %	1605.80
Archi4.	0.91	0.90	0.90	10.63 %	1409.88

achieves quite good results on both detection rate and accuracy. On average, the proposed system obtains the F-score(s) of 0.83 and 0.92 on electrical and architectural datasets, respectively. It is worth pointing that these scores are obtained under a very small overlap of the detected bounding boxes. The processing time consists of all stages included in the proposed system and is calculated as mean time of a complete query relying on our specific computer configuration: Intel(R) Core(TM) i5 CPU 2.4GHz, RAM 2.4 GB, Windows 8. As discussed by the authors of this contest [15], the performance of a spotting system is not only dependent on the level of noises but also other factors including the number of model symbols, query symbols, and the number of symbol instances in document database, *etc*. It is noted that the results obtained on the architectural dataset are much higher than those on electrical dataset. The reason lies in the fact that in the electrical dataset, more query symbols are used and many query symbols looks very similar making it difficult to be correctly distinguished.

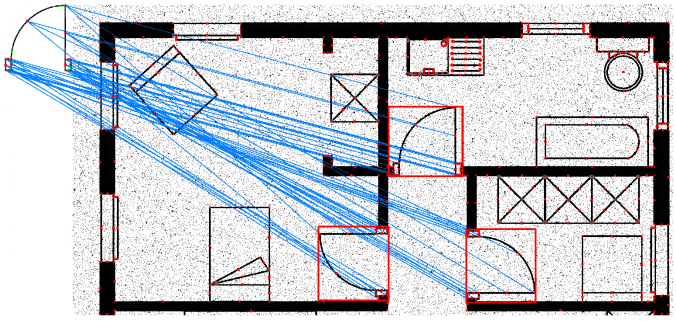

Fig. 5. An example of symbol localization: a query symbol (left) and the detected instances of the query (red bounding boxes).

Figure 5 shows an example of our symbol localization system where the query is perfectly localized even if it is embeded into a complicated database document. There are, however, some queries as shown in Figure 6 that they are very difficult to be correctly distinguished.

IV. CONCLUSION

In this work, a new approach for symbol localization in line-drawing images has been presented. The main contribution

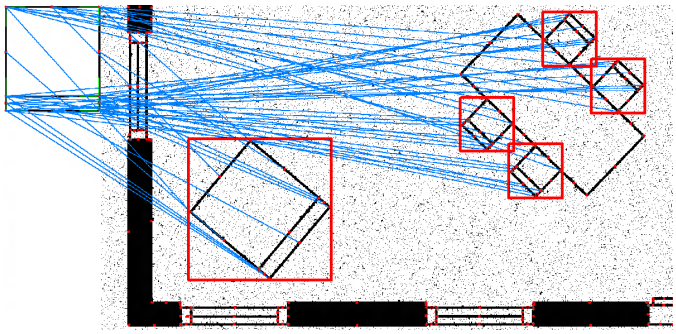

Fig. 6. False alarms (small bounding boxes) due to the difficulty of a document content.

of our work is three-fold. First, a new set of keypoints is presented to work on the line-drawing documents. Second, a new decomposition method is described to construct a complete and compact representation of the graphic document images. Finally, an efficient method is proposed to deal with the problem of geometric consistency checking. All of these advantages are attributed to the proposed symbol localization system, making it very time- and memory-efficient. We have also demonstrated that the proposed system achieves quite interesting results for a large and standard dataset. Further works on keypoint indexing would be very useful to deal with the time-critical applications such as symbol retrieval or symbol spotting.

REFERENCES

[1] D.H. Ballard. Generalizing the hough transform to detect arbitrary patterns. *Communications of the ACM*, 13(2):111–122, 1981.

[2] P. Le Bodic, H. Locteau, S. Adam, P. Heroux, Y. Lecourtier, and A. Knippel. Symbol detection using region adjacency graphs and integer linear programming. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 1320–1324, 2009.

[3] M. Delalandre, E. Valveny, and J. Llads. Performance evaluation of symbol recognition and spotting systems: An overview. In *Workshop on Document Analysis Systems (DAS)*, pages 497–505, 2008.

[4] P. Dosch and J. Llads. Vectorial signatures for symbol discrimination. In *Workshop on Graphics Recognition (GREC), LNCS(3088)*, pages 154–165, 2004.

[5] A. Dutta, J. Llados, and U. Pal. Symbol spotting in line drawings through graph paths hashing. In *International Conference on Document Analysis and Recognition (ICDAR)*, 2011.

[6] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381395, 1981.

[7] X. Kong, E. Valveny, G. Sanchez, and L. Wenyin. Symbol spotting using deformed blurred shape modeling with component indexing and voting scheme. In *Workshop on Graphic Recognition (GREC)*, 2011.

[8] J. Llads, E. Valveny, G. Snchez, and E. Mart. Symbol recognition : Current advances and perspectives. In *Workshop on Graphics Recognition (GREC), LNCS(2390)*, pages 104–127, 2002.

[9] B. Messmer and H. Bunke. Automatic learning and recognition of graphical symbols in engineering drawings. In *Workshop on Graphics recognition (GREC), LNCS(1072)*, pages 123–134, 1996.

[10] N. Nayef and T.M. Breuel. On the use of geometric matching for both: Isolated symbol recognition and symbol spotting. In *Workshop on Graphics Recognition (GREC)*, 2011.

[11] T.O. Nguyen, S. Tabbone, and A. Boucher. A symbol spotting approach based on the vector model and a visual vocabulary. In *International Conference on Document Analysis and Recognition (ICDAR'09)*, pages 708–712, 2009.

[12] The-Anh Pham, Mathieu Delalandre, Sabine Barrat, and Jean-Yves Ramel. Accurate junction detection and reconstruction in line-drawing images. In *Proceedings of the 21st Conference of the International Association for Pattern Recognition (ICPR2012)*, pages 693–696, 2012.

[13] R.J. Qureshi, J.Y. Ramel, D. Barret, and H. Cardot. Spotting symbols in line drawing images using graph representations. In *Worksop on Graphics Recognition (GREC), LNCS(5406)*, pages 91–103, 2008.

[14] M. Rusiol and J. Llads. Symbol spotting in technical drawings using vectorial signatures. In *Workshop on Graphics Recognition (GREC), LNCS(3926)*, pages 35–46, 2006.

[15] Ernest Valveny, Mathieu Delalandre, Romain Raveaux, and Bart Lamiroy. Report on the symbol recognition and spotting contest. In *Proceedings of the 9th International Workshop on Graphics Recognition (GREC'11) - LNCS(7423)*, 2011.