Near Convex Region Adjacency Graph and Approximate Neighborhood String Matching for Symbol Spotting in Graphical Documents

Anjan Dutta and Josep Lladós
Computer Vision Center
Universitat Autònoma de Barcelona
Barcelona, Spain
Email: {adutta,josep}@cvc.uab.es

Horst Bunke
Inst. of Comp. Sc. and Appl. Maths
University of Bern
Bern, Switzerland
Email: bunke@iam.unibe.ch

Umapada Pal
CVPR Unit
Indian Statistical Institute
Kolkata, India
Email: umapada@isical.ac.in

***Abstract*—This paper deals with a subgraph matching problem in Region Adjacency Graph (RAG) applied to symbol spotting in graphical documents. RAG is a very important, efficient and natural way of representing graphical information with a graph but this is limited to cases where the information is well defined with perfectly delineated regions. What if the information we are interested in is not confined within well defined regions? This paper addresses this particular problem and solves it by defining near convex grouping of oriented line segments which results in near convex regions. Pure convexity imposes hard constraints and can not handle all the cases efficiently. Hence to solve this problem we have defined a new type of convexity of regions, which allows convex regions to have concavity to some extend. We call this kind of regions Near Convex Regions (NCRs). These NCRs are then used to create the Near Convex Region Adjacency Graph (NCRAG) and with this representation we have formulated the problem of symbol spotting in graphical documents as a subgraph matching problem. For subgraph matching we have used the Approximate Edit Distance Algorithm (AEDA) on the neighborhood string, which starts working after finding a key node in the input or target graph and iteratively identifies similar nodes of the query graph in the neighborhood of the key node. The experiments are performed on artificial, real and distorted datasets.**

***Index Terms*—Near Convex Region Adjacency Graph, Subgraph Matching, Approximate Edit Distance Algorithm, Symbol Spotting, Graphics Recognition.**

I. Introduction

Symbol spotting in graphical documents is an important problem in the graphics recognition community. There are many methods that have been proposed to solve the problem. The interested readers are referred to [1] for a recent and detailed literature review. Many of the methods have proposed some sort of subgraph matching as the solution, where smaller graphs represent the query symbols and the larger graphs represent the graphical documents. Often this kind of methods use the Region Adjacency Graph (RAG) as the way of representing graphical information [2]–[4], where a region is roughly defined as a white connected component. This is well justified since RAG allows to capture regionwise contextual information. RAG has also been widely used for classification [5] and object detection and recognition [6], [7] in other fields of computer vision. The main advantage of RAG is that it is natural and robust, and allows one to capture regionwise contextual information. But it is not always representative when the region boundaries are not clearly defined or they have some discontinuities (as in the symbol *door1* and *door2* respectively in Fig. 1a and Fig. 1b and in the synthetically distorted example of symbol *table1* in Fig. 1d). So to solve these problems, in this paper we define Near Convex Region Adjacency Graph (NCRAG) where the regions must not be clearly and continuously bounded, but can be nearly convex. This is done by near convex grouping of the oriented line segments and defining the convexity of the regions. Then we use this NCRAG representation to solve the problem of subgraph matching and apply it for symbol spotting in graphical documents. The first step of the method is to create two NCRAGs, one from a graphical document and the other from a symbol and then in the second step apply the Approximate Edit Distance Algorithm (AEDA) for subgraph matching.

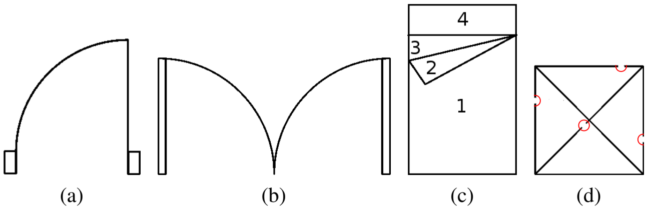


Fig. 1: Limitations of RAG and convex region based representation: (a) the symbol *door1* contains open region, (b) the symbol *door2* also contains open regions, (c) the symbol *bed* contains a region (region 1) which is not convex, (d) the symbol *table1* contains discontinuous boundaries.

Convexity of objects is a very important property and it has been shown that most of the objects, even though they are not fully convex, can be decomposed into multiple convex parts [8]. Also it is important to note that often the object of interest is almost convex. So the property of convexity has already been studied in the field of computer vision and pattern recognition for object detection and recognition [9] and recently it has also been studied in document analysis for symbol spotting [3], [4], [10]. But, as it has been mentioned before, the object of interest might not always be perfectly convex but include some concavity in some parts (as the region 1 of the symbol *bed* in Fig. 1c). Of course, such regions can be split into multiple strictly convex parts as it is studied in [7], [8] but it is inefficient dealing with a large number of smaller purely convex parts rather than few near convex parts. Also small concavity provides discrimination in the representation of objects, so it is an important property to be considered for description. Convexity or near convex decomposition has also been studied in [11] very recently. Hence representing the graphical documents with NCRAG seems worthwhile and useful.

The main contributions of the present paper are: (1) Formulation of NCRs using the near convex grouping of a set of oriented line segments which not necessarily have to be closed and the use of these NCRs to construct the NCRAGs. This NCRAG is able to handle concavity within the convex regions and at the same time it is as expressive as RAG. (2) Application of the Approximate Edit Distance Algorithm (AEDA) [12] to solve the problem of subgraph matching for faster symbol spotting in graphical documents. The method does not need any learning or offline step and can be computed in reasonable time as shown in Table I and Table II.

The rest of the paper is organized into four sections. In Section II we explain the detailed methodology. Section III shows the experimental results of the paper. In Section IV we provide a detailed discussion about limitations of this kind of representation and compare the results with a previously proposed method and note the improvements. At last in Section V we conclude the paper and discuss future directions of work.

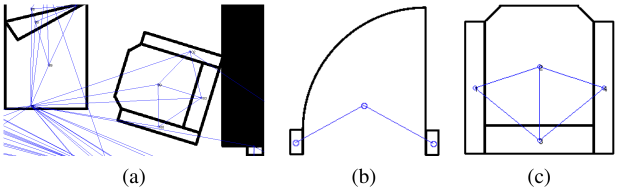
II. Methodology

The first step of the method is to create two NCRAGs, one from the target graphical document and the other from the query symbol. Formally we define an NCRAG as a graph $G = (V, E, \phi, \psi)$, where V is the set of nodes and $E \subseteq V \times V$ is the set of edges of the graph G and are defined as follows:

$$V = \{v_i : v_i \text{ is a convex region (nearly) in the document}\}$$

$$E = \{(v_i, v_j) : v_i, v_j \in V \text{ and } v_i, v_j \text{ are adjacent regions}\}$$

$\phi : V \to \mathbb{R}^n$ is the node labeling function, in this case, the Hu moments invariants concatenated with the Euler number and solidity of each of the regions. Therefore, the node label has the dimension nine and all values are normalized between 0 and 1. $\psi : E \to \mathbb{R}$ is the edge labeling function, in this case, the ratio of the length of the common boundary to the length of the larger boundary between the two regions connecting the edge. Given two NCRAGs, the symbol spotting problem can be formulated as a subgraph matching problem, where the task is to find an instance of the smaller query symbol graph in the larger document graph. Let us denote the NCRAG of the query symbol as the query graph G_Q and that of the document as the input or target graph G_I. As the second step, for matching the subgraph we have used the efficient AEDA proposed by Neuhaus and Bunke in [12]. These two steps are explained in the subsequent subsections.



(a) (b) (c)

Fig. 2: NCRAG representing (a) a part of a floorplan, (b) a symbol with open region (*door1*), (c) a symbol with all closed regions (*armchair*).

A. Near Convex Region Adjacency Graph (NCRAG)

This step starts working on the vectorized images which contain the approximated line segments. Here each line segment is considered as two oriented line segments where an oriented line segment is defined as a line segment where one endpoint is considered as its first endpoint [8]. If l_i is an oriented line segment, then we consider $l_{i,1}$ as it's first endpoint and $l_{i,2}$ is its second. Let us consider their coordinates as (x_{i1}, y_{i1}) and (x_{i2}, y_{i2}) respectively. Just to clarify, if l_i and l_j are two consecutive oriented line segments coinciding end-to-end then the coordinate (x_{i2}, y_{i2}) and (x_{j1}, y_j) denote the coordinates of the same point. Now let $S_n = \{l_1, l_2, \ldots, l_n\}$ be a sequence of oriented line segments and L_i be the length of the segment l_i and γ_i be the gap between $l_{i,2}$ and $l_{i+1,1}$. Then according to the original algorithm [8] we have:

$$L_{i,n} = \sum_{i=1}^{n} L_i, \qquad \gamma_{i,n} = \sum_{i=1}^{n} \gamma_i \tag{1}$$

The saliency measurement of the convex group S_n can be defined as:

$$Saliency(S_n) = \frac{L_{i,n}}{L_{i,n} + \gamma_{i,n}} \tag{2}$$

The saliency parameter helps to incorporate the erroneous gaps that might be generated during binarization or vectorization as we have shown in one of our experiments in Section III. Before adding any oriented line segment to a sequence, the saliency measurement of the sequence is checked. In case the saliency of the sequence is less than t_{sal} the current line segment is added to the sequence.

The convexity of the group S_n is defined as:

$$Convexity(S_n) = \frac{area(S_n)}{area(CHS_n)} \tag{3}$$

where CHS_n is the convex hull of S_n. Since any group S_n is not guaranteed to be closed, its area is computed as:

$$area(S_n) = \frac{\sum_{i=1}^{n}(x_{i1}y_{i2} - x_{i2}y_{i1}) + (x_{i2}y_{((i+1)\%n)2} - x_{((i+1)\%n)1}y_{i2})}{2}. \tag{4}$$

Like the saliency measurement, before adding any oriented line segment to a sequence S_n, its convexity together with S_n is checked and if it is less than t_{conv}, it is added to the sequence.

To make the idea clear it is to be mentioned that for efficient computation, for each oriented line segment l_i, the original algorithm precomputes the list of all other oriented line segments $List(l_i)$ with which it is mutually convex and sorts them according to the distance. Secondly, it also precomputes the angle that is turned when going from one oriented line segment to another. Since we take into account the convexity of a sequence S_n we only sort the list according to the distance and check the saliency and convexity of the current sequence together with the line segment to be added before adding it to S_n. These NCRs are then used to create NCRAG. Fig. 2 shows some results of the NCRAG construction. Construction of the NCRAG can be done in time complexity of $\mathcal{O}(m^2 logm + m^2)$, where m is the number of oriented line segments.

B. Approximate Edit Distance Algorithm (AEDA)

The AEDA starts by finding a similar node in G_I to a node in G_Q. These nodes are called the key nodes [12]. The similarity of the nodes is inversely proportional to the Euclidean distance of the node labels, and the edge labels of the graph are not taken into account here. Then the algorithm looks at the neighborhood nodes considering the key nodes as the center nodes. The neighborhood nodes are then arranged in clockwise order to create a string. Here the connectivity information between the neighborhood nodes is taken into account. If any two nodes are connected the corresponding edge label is concatenated with the incident node label and form the attributed string. After having constructed the attributed string, cyclic string edit distance is applied to get the node-to-node correspondences. Then each of the nodes in each correspondence is considered as a key node and the previous steps are repeated. This algorithm continues working until it gets new correspondences. In the cyclic string the edge label is augmented with the originating node and the cost function is defined as:

$$\lambda\,|\phi_1 - \phi_2| + (1-\lambda)\,|\psi_1 - \psi_2|\,, \text{ where } 0 \le \lambda \le 1.$$

where ϕ and ψ are, respectively, the node and edge labels. For the original algorithm the readers are referred to [12].

For each node in G_Q we consider n key nodes in G_I and perform the AEDA. Therefore for a single query graph G_Q, we perform $n \times m$ iterations of the AEDA in G_I, where m is the number of nodes in G_Q. In this case, n should be greater than the actual number of instances of the query symbol in a graphical document to get all the relevant instances. Here it is to be mentioned that greater values of n might produce more false positives but the system produces a ranked list of the relevant retrievals. So it does not hamper the performance, since the true positives suppose to appear at the beginning of the ranked list of retrieved symbols. The edge label is only used when we perform the cyclic string matching on the strings obtained from the neighborhood subgraphs by considering the nodes in clockwise order. At the end, we obtain a distance measure between a retrieved subgraph and the query graph by calculating the distance of the node labels. Later we use this distance to rank the retrieved subgraphs.

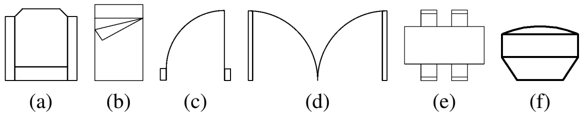


Fig. 3: Model symbols: (a)-(e) SESYD, (f) FPLAN-POLY: (a) *armchair*, (b) *bed*, (c) *door1*, (d) *door2*, (e) *table2*, (f) *television*.

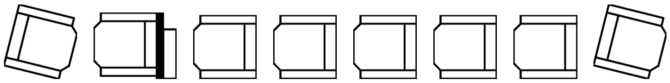


Fig. 4: First eight retrievals of *armchair*.

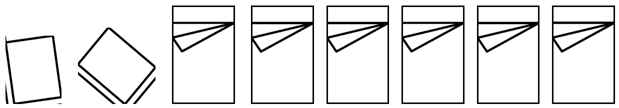


Fig. 5: First eight retrievals of *bed*.

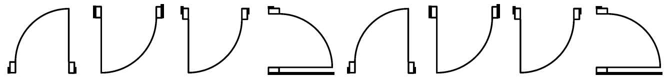


Fig. 6: First eight retrievals of *door1*.

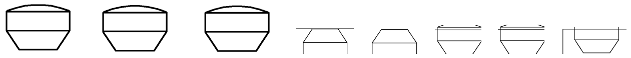


Fig. 7: First eight retrievals of *television*.

III. Experimental results

Experiments are carried out to show (1) the robustness of the algorithm for constructing NCRAG and (2) the efficiency of the AEDA for subgraph matching in NCRAG. We have considered two different datasets: (1) SESYD (floorplans)[1] and (2) FPLAN-POLY[2] for experiments. SESYD (floorplans) contains a set of synthetic floorplans and FPLAN-POLY contains a set of real floorplans. Both datasets contain different kind of query symbols, for example, symbols with closed and open regions, and symbols having slightly concave regions. The set of available query symbols for each dataset are used as query to evaluate with the ground truth. The graphical

[1] http://mathieu.delalandre.free.fr/projects/sesyd/

[2] http://www.cvc.uab.es/ marcal/FPLAN-POLY/index.html

documents as well as the query symbols are available as binary images. To get the line segments, the vectorization algorithm of Qgar[3] is applied. For each of the symbols the performance of the algorithm is evaluated in terms of precision (**P**), recall (**R**) and average precision (**AveP**). To have an idea about the computation time we calculate the per document retrieval time (**T**) required for each of the symbols with each document. For each of the datasets the mean of the above mentioned metrics is shown (Table I) to judge the overall performance of the algorithm. Throughout our experiments we have chosen $t_{sal} = 0.95$, $t_{conv} = 0.8$ and $\lambda = 0.6$; we run a set experiments varying these parameters, then the best values for these parameters are chosen to give the best performance. The detailed experiments for choosing these parameters are beyond the size of the paper.

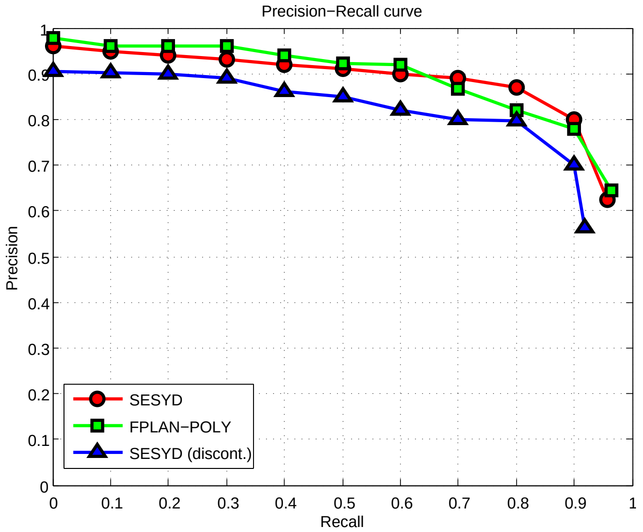


Fig. 9: Precision recall curve for different dataset.

A. Experiments on SESYD

This dataset contains 10 different subsets, each of which contains 100 synthetically generated floorplans. All the floorplans in a subdatasets are created upon a same floorplan template by putting different model symbols in different places in random orientation and scale. In this experiment we have only considered a subset of 100 images (floorplans16-01). Each of the NCRAGs of each of the floorplans approximately contains 150 NCRs, whereas each of the query symbols contains 3-6 NCRs. The quantitative results are shown in the first row of Table I. The high recall values for this dataset show that the algorithm works pretty well for most of the symbols. There are some cases of failure or partial detection, the reason of which will be discussed in Section IV. Qualitative results are shown in Figs. 4 to 6, which, respectively, include symbols with closed, near convex and open regions.

B. Experiments on FPLAN-POLY

This dataset contains 42 real floorplans, each being part of a real floorplan. The images contain distortions, text-graphic interference, etc. Here we have used all the floorplans and 10 randomly chosen model symbols. In this dataset each floorplan image contains approximately 110 NCRs, whereas a query symbol contains 4-8 NCRs. The recall value obtained in this dataset is also very good which is shown in Table I. Qualitative results of querying *television* are shown in Fig. 7 (note the disappearance of some boundaries). The results obtained in this dataset is slightly better than SESYD. The reason is mentioned in the discussions part (Section IV). The parameter t_{sal} has less influence on SESYD and FPLAN-POLY dataset, since the line segments are end-to-end coinciding.

C. Experiments on SESYD with discontinuous edges

This experiment is performed to prove the robustness of the algorithm constructing the NCRAG. To do that we have taken the same subset of SESYD and randomly drawn white horizontal lines of 2-3 pixels width. This generates random discontinuity of black pixels. After vectorizing the image we apply the algorithm to spot the symbol on it. The quantitative and qualitative results are respectively shown in the Table I (3^{rd} row) and Fig. 8 (note the white gaps on the black edges). Here the parameter t_{sal} poses an important role and to be tuned according to the existing gap in edges. The method fails when the drawn white lines remove substantial portion of a symbol. The precision recall curve (Fig. 9) shows the performance of the method for these three datasets, from that it is clear that the method performs worse in case of the discontinuous edges than the other two.

TABLE I: Dataset wise mean results with NCRAG representation.

Symbol	P	R	F	AveP	T (secs)
SESYD (floorplans16-01)	62.33	95.67	74.76	70.66	0.57
FPLAN-POLY	64.56	96.32	76.21	73.68	0.65
SESYD (discont.)	56.33	91.75	70.81	64.65	0.59

IV. Discussion

Although NCRAG is capable of capturing contextual information well in terms of regions, there are some serious limitations of NCRAG or, more generally, of the RAG based representation. One problem is shown in Fig. 10, where there are two symbols called *sink3* and *sink4* and the difference between them when they appear in a model symbol (Fig. 10(a),(c)) and in a document (Fig. 10(b),(d)). This is due to the difference in stroke width in images. Particularly, in the example of *sink3* when it appears in the document it looses the thin peripheral portion in the left of the region and also small circular part detaches the upper right corner part of the square. These create some difference in the regions but apparently they appear the same with our high level vision. The dissimilarity in regions also changes the NCRAG representation. As a result it partially finds some nodes of a graph and results in partial detection or complete loss. Hence it lowers the similarity score and precision. Since in FPLAN-POLY the query symbol is generated by cropping the floorplan image, there is no

[3] www.qgar.com

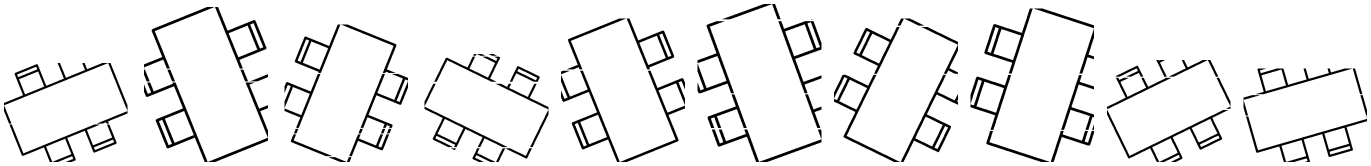


Fig. 8: First 10 retrievals of *table2* on the database of floorplans having discontinuous line noise.

discrepancy like that. This explains the slight better results in FPLAN-POLY.

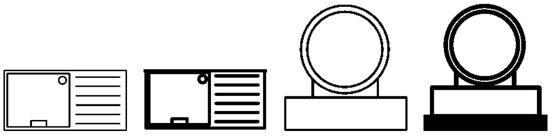


Fig. 10: Limitations of region based representation: (a) model symbol of *sink3*, (b) *sink3* as it appears in the document, (c) model symbol of *sink4*, (d) *sink4* as it appears in the document.

TABLE II: Comparison between a state-of-the-art method and the current method.

Symbol	P	R	F	AveP	T (secs)
Dutta et al. [1]	41.33	82.66	51.24	52.46	0.07
Current method	62.33	95.67	74.76	70.66	0.57

As we have used the same dataset (floorplans16-01 of SESYD) as the method proposed in [1], we can do a direct comparison between the results. Table II shows the results obtained by these two methods. Clearly the NCRAG based representation improves the performance remarkably. This was expected since this kind of representation takes into account contextual information. But at the same time it has some limitations as explained above. Also the time complexity of the proposed method is quite high compared to the other one. Since [1] uses an indexation technique of the serialized subgraphical structure, the online part of the method is quite fast. But it is to be noted that the method needs an offline steps to create the indexation or hash table. The interested readers are referred to [1] for detailed comparisons of different symbol spotting methods.

V. Conclusion and future works

In this paper we have proposed a near convex grouping approach to create NCRAG on graphical documents. Then this representation has been used for spotting symbols on graphical documents with the application of the efficient AEDA. We have shown the results of the proposed method on three different sets of images and the results obtained are quite satisfactory. We have also compared results with a previously proposed method and noticed the methodological differences and performance improvement. At the end we have shown some limitations of this kind of region based representation.

From the experimental results it is clear that this kind of region based representation is very robust despite having some limitations and higher time complexity. In the past we have seen that hashing or indexation of the graphical structure can make the online part really fast. So, in future, it would be interesting to investigate region based indexation techniques. Apart from that it would be worth to investigate an improvement of NCRAG that could handle the above mentioned limitations. The detailed performance comparison of the improved version of the current method with the other existing methods will also be done in future.

Acknowledgement

This work has been partially supported by the Spanish projects TIN2009-14633-C03-03, TIN2011-24631, TIN2012-37475-C02-02 and the PhD scholarship 2013FI_B2 00074.

References

[1] A. Dutta, J. Lladós, and U. Pal, “A symbol spotting approach in graphical documents by hashing serialized graphs,” *Pattern Recognition*, vol. 46, no. 3, pp. 752–768, 2013.

[2] J. Lladós, E. Martí, and J. J. Villanueva, “Symbol recognition by error-tolerant subgraph matching between region adjacency graphs,” *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 23, pp. 1137–1143, 2001.

[3] P. L. Bodic, P. Hèroux, S. Adam, and Y. Lecourtier, “An integer linear program for substitution-tolerant subgraph isomorphism and its use for symbol spotting in technical drawings,” *Pattern Recognition*, vol. 45, no. 12, pp. 4214–4224, 2012.

[4] A. Barducci and S. Marinai, “Object recognition in floor plans by graphs of white connected components,” in *21st International Conference on Pattern Recognition (ICPR)*, 2012, pp. 298–301.

[5] Z. Harchaoui and F. Bach, “Image classification with segmentation graph kernels,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, june 2007, pp. 1–8.

[6] M. Suk and T.-H. Cho, “An object-detection algorithm based on the region-adjacency graph,” *Proceedings of the IEEE*, vol. 72, no. 7, pp. 985–986, july 1984.

[7] D. W. Jacobs, “Grouping for recognition,” 1989.

[8] ——, “Robust and efficient detection of salient convex groups,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 1, pp. 23–37, Jan. 1996.

[9] D. G. Lowe, “Three-dimensional object recognition from single two-dimensional images,” *Artif. Intell.*, vol. 31, no. 3, pp. 355–395, Mar. 1987.

[10] N. Nayef and T. Breuel, “Statistical grouping for segmenting symbols parts from line drawings, with application to symbol spotting,” in *Proceedings of 11th International Conference on Document Analysis and Recognition (ICDAR)*, 2011, pp. 364–368.

[11] Z. Ren, J. Yuan, C. Li, and W. Liu, “Minimum near-convex decomposition for robust shape representation,” in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, nov. 2011, pp. 303–310.

[12] M. Neuhaus and H. Bunke, “An error-tolerant approximate matching algorithm for attributed planar graphs and its application to fingerprint classification,” in *Proceedings of 10th International Workshop on Structural and Syntactic Pattern Recognition (S+SSPR)*, 2004, pp. 180–189.