Noname manuscript No.
(will be inserted by the editor)

Flowchart Recognition for Non-Textual Information Retrieval in Patent Search

Marçal Rusiñol · Lluís-Pere de las Heras · Oriol Ramos Terrades

Received: date / Accepted: date

Abstract Relatively little research has been done on the topic of patent image retrieval and in general in most of the approaches the retrieval is performed in terms of a similarity measure between the query image and the images in the corpus. However, systems aimed at overcoming the semantic gap between the visual description of patent images and their conveyed concepts would be very helpful for patent professionals. In this paper we present a flowchart recognition method aimed at achieving a structured representation of flowchart images that can be further queried semantically. The proposed method was submitted to the CLEF-IP 2012 flowchart recognition task. We report the obtained results on this dataset.

Keywords Flowchart Recognition · Patent Documents · Text/Graphics Separation · Raster-to-Vector Conversion · Symbol Recognition.

1 Introduction

Information retrieval in the intellectual property domain has been a major research area within the Information Retrieval field for many years. Although being an already mature research topic, it is far from being a solved problem. Patent professionals need advanced search tools in order to assess the viability of a given invention with respect to the state of the art. Information retrieval tools such as cross-lingual information retrieval, document categorization or query expansion are often used to provide a thorough analysis of patent applications. However, not all the information in a patent document is conveyed

M. Rusiñol · L.P. de las Heras · O.R. Terrades
Computer Vision Center, Dept. Ciències de la Computació
Edifici O, Universitat Autònoma de Barcelona
08193 Bellaterra (Barcelona), Spain
E-mail: {marcal,lpheras, oriolrt}@cvc.uab.es

by textual elements. This is usually the case in the chemical, pharmaceutical or electronics fields in which the core information of the patent invention is depicted by means of a line-drawing instead of being explicitly written in textual format. Discussion papers such as (Adams, 2005) and (List, 2007), already pointed out the interest of image search in patent documents.

However, as surveyed in (Hanbury et al, 2011) and (Bhatti and Hanbury, 2012), relatively little research has been done on the topic of patent image retrieval. In general most of the published approaches dealing with patent image retrieval follow the Content-Based Image Retrieval (CBIR) paradigm (Lew et al, 2006), in which the retrieval is performed in terms of a similarity measure between the query image and the images in the corpus.

For example in (Huet et al, 2001), patent drawings are represented using attributed graphs and the retrieval task is then casted as a graph similarity computation. Methods like (Codina et al, 2008; Vrochidis et al, 2010, 2012) and (Sidiropoulos et al, 2011) base the image description on histograms encoding the centroid positions at different levels. Such descriptors can be understood as a special case of a quad-tree (Samet and Webber, 1985) encoding of the image under analysis. The retrieval part just relies on the Euclidean distance between query and corpus descriptors. In (Tiwari and Bansal, 2004), the PATSEEK framework is presented in which patent drawings are described by means of edge orientation autocorrelograms (Mahmoudi et al, 2003).

In order to promote comparability among methods and easily track the progress in the field of information retrieval in the intellectual property domain, within the CLEF initiative a track specialized on Intellectual Property (IP) retrieval was first organized in 2009. Until 2011, the IP track served as a benchmarking activity on prior art retrieval focusing only on textual patent documents. However, in 2011 two image-based tasks were added (Piroi et al, 2011). One devoted to find patent documents relevant to a given patent document which contained images and another aimed at categorizing patent images into predefined categories of images (such as graphs, flowcharts, drawings, etc.). In order to tackle the classification task the participant methods (Mörzinger et al, 2011; Csurka et al, 2011) holistically described the individual images and such descriptors were then fed to a supervised classifier.

However, the CBIR paradigm might not be the most suitable tool to provide an image search in the intellectual property domain. In order to assess whether an invention is new or has already been submitted, the patent professional should look for images that depict the *same concept* (Vrochidis et al, 2012) instead of images that *look visually similar* to the query. That is, image retrieval methods should be able to bridge the *semantic gap* between the visual appearance of the images and the semantic meaning they convey (Lupu et al, 2012). Although a large variety of different images can be found within patent documents (chemical structures, math formulas, device designs, trademarks, etc.), we will focus on line-drawings of flowcharts, since they carry an important semantic meaning and it would be beneficial to “translate” such graphical information into a structured format that will allow to browse the contained information.

Conversely, within the pattern recognition community, and specifically in the graphics recognition field, the task of "understanding" line drawings in order to obtain a structured representation of graphical images has been studied for more than thirty years now. Early works such as (Bunke, 1982) or (Lin et al, 1985) were already focused on the recognition of line drawings for further automatic process. This research line continued until the midninetieths (Blostein, 1996; Yu et al, 1997) when most of the research efforts were re-focused on the treatment of on-line sketched drawings (Szwoch, 2007; Yuan et al, 2008) although some recent research in those lines can still be found (Vasudevan et al, 2008). To our best knowledge, no commercial patent retrieval system uses image understanding techniques for the retrieval of non-textual information in patent documents. The only efforts in that direction come from the Image Mining for Patent Exploration (**IMPEx**) Project [1] which its main objective is the extraction of semantic information from patent images.

In CLEF-IP 2012 (Piroi et al, 2012) a new image-based task was proposed. The flowchart recognition task deals with the interpretation of flowchart line-drawing images. The participants were asked to extract as much structural information as possible in these images and return it in a predefined textual format for further processing for the purpose of patent search. Three different institutions participated in such task (Mörzinger et al, 2012; Rusiñol et al, 2012; Thean et al, 2012).

The state of the art in graphical document understanding presents an important flaw, most of the methods (e.g. (Bunke, 1982; Lin et al, 1985; Lamiroy et al, 2001; Valveny and Lamiroy, 2002)) were just evaluated qualitatively, thus making very difficult to actually assess the methods' performances. Initiatives such as the CLEF-IP 2012 flowchart recognition task are really beneficial for the community since they allow to track the progress and to fairly compare different methods under the same conditions. This paper is an extended version of our previous working notes paper (Rusiñol et al, 2012) that introduced our proposed flowchart recognition methodology. Our method follows the ideas sketched in (Lamiroy et al, 2001) and (Valveny and Lamiroy, 2002) in which the authors proposed a framework allowing to automatically generate a structured output file (XML-like) from the analysis of input graphical documents. Specifically, we have extended the method details in order to make the paper more self-contained. We have also included a quantitative evaluation section in which we compare the results of all the participant's methods in CLEF-IP 2012. This comparison with (Mörzinger et al, 2012) and (Thean et al, 2012) shows how our contribution outperforms the other participant's methods in both the ability of detecting the flowchart structure and the ability of automatically transcribe the text within the flowchart. Finally, an implementation[2] of our baseline system has been made available in order to allow the interested readers to test and study our approach.

[1] http://www.joanneum.at/?id=3922

[2] Code available at: `http://www.cvc.uab.es/~marcal/demos/flowchart.html`

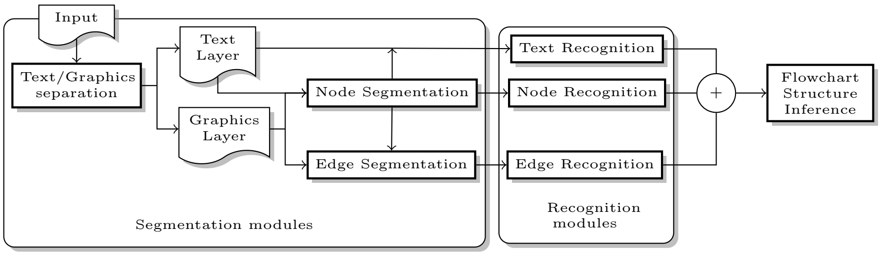


Fig. 1 System's architecture overview.

The rest of this paper has been organized as follows. Section 2 overviews the proposed architecture and details each of the modules that comprises the system's pipeline. In section 3 we present and analyze the obtained results and finally in section 4 we draw our concluding remarks.

2 Architecture Description

We have designed the architecture of our recognition system as follows. As we can see in Fig. 1, we structure the system pipeline in separate modules dealing with the different steps of the flowchart recognition problem.

Many of the methods used in each step require a number of parameters to be fixed beforehand and depend on the quality of flowchart image in terms of image resolution, noise, etc. and also whether images are black&white, gray scale or color. In this section we provide a rough overview of methods and meaning of the parameters required while we provide the actual values used on the dataset in the section of experimental results, section 3.

The input of the system is an already segmented flowchart image that appeared in a patent document, as the one shown in Fig 2a). In a first step, we apply a text/graphic separation module aimed to separate the textual elements from the graphical ones. We apply an OCR engine on the text layer while we analyze the nodes and edges on the graphical layer. Then, for the node and edge segmentation, we apply two different strategies resulting in two alternative segmentation modules: a pixel-based and a vectorial-based approach. The vectorial-based approach requires a conversion module that transforms the raw pixel image into a vectorial representation. By submitting two different runs we wanted to assess the strengths and weaknesses of using two different primitives, pixels and vectors, for node and edge segmentation.

The output of the node segmentation module is a list of bounding-boxes of the detected nodes. These locations are subsequently fed to the recognizer module which is in charge of establishing the type of node (e.g. circle, oval, rectangle, etc.). Herein, we have used two different node descriptors, namely a descriptor based on geometric moments (Zhang and Lu, 2002) and the Blurred Shape Model (BSM) descriptor (Escalera et al, 2009), resulting in two alternatives for the node recognition module. The modules analyzing edges are devoted to assess which nodes are connected and classify the edges in terms of their style.

Finally and taking together the results of text, node and edge recognition, we apply a flowchart structure inference module in order to correct certain *syntactic* errors and build the final graph structure representing the flowchart.

The combination of the two alternative node and edge segmentation modules with the two alternative node recognition modules results to four system variants producing the four submitted runs to the CLEF-IP 2012 flowchart recognition task summarized in Table 1.

Table 1 Submitted runs produced by the four different system variants.

Id	Run	Segmentation	Descriptor
$R1$	CVC-UAB.BSM	Pixel-based	BSM
$R2$	CVC-UAB.GMOMENTS	Pixel-based	Geometric moments
$R3$	CVC-UAB.VECTORIALBSM	Vectorial-based	BSM
$R4$	CVC-UAB.VECTORIALGMOMENTS	Vectorial-based	Geometric moments

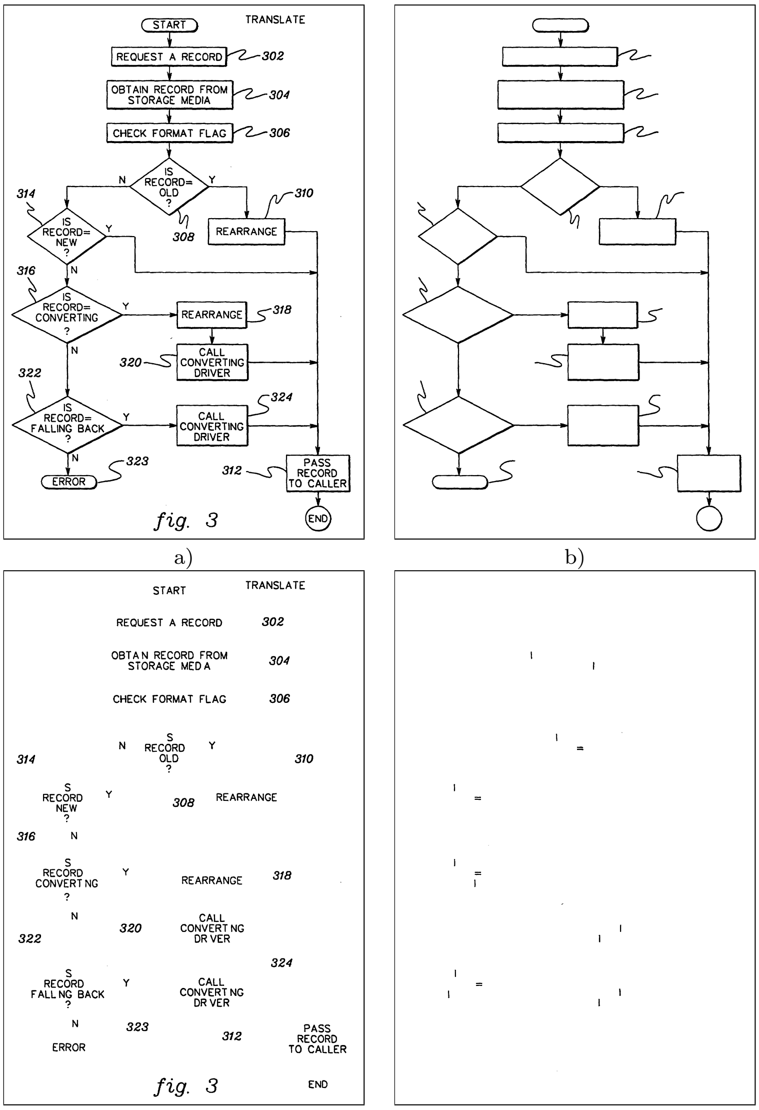
Let us describe in the following sections each of these specific modules.

2.1 Segmentation modules

The segmentation module is composed of the following steps applied sequentially, one after the other: *text/graphics separation*, *raster-to-vector conversion*, *node segmentation* and *edge segmentation*. In addition, for node and edge segmentation modules, we have analyzed two different alternatives depending on the primitives we use to segment nodes and edges from the flowchart images. On the one hand, we have proposed segmentation modules directly working with the raw pixels from the graphical layer. On the other hand, we have proposed a segmentation strategy working on the vectorial representation of the graphical layer.

Text/graphics separation. We have applied the text/graphics separation algorithm proposed by (Tombre et al, 2002) which yields acceptable results in a variety of mixed-type documents. This approach is based on the well-known approach of (Fletcher and Kasturi, 1988), which in turn is based on the analysis of connected components (CCs). A CC is a maximal set of spatially connected pixels that share a common property (herein, the same color). Although Tombre's method might not be the best one (it is really hard to assess which is the best text/graphics separation technique in a general sense), both Fletcher and Tombre's methods are used as de-facto standards within the document analysis community since they have been proven to be stable and generic in different scenarios (Lladós and Rusiñol, 2013).

The applied text/graphics separation module proceeds as follows: Given an input image, structural features such as height and width ratios are computed from CCs. Four thresholds determine whether a CC corresponds to a graphical element or a textual element. Moreover, the CCs that the system is not certain



a) b) c) d)

Fig. 2 Example of the text/graphics separation module. a) Original image, b) graphical layer, c) textual layer, d) undetermined layer.

about, are assigned to an undetermined layer via a rejection criterion. The output of the module is an image consisting of three separate layers, namely the textual, the graphical and the undetermined one.

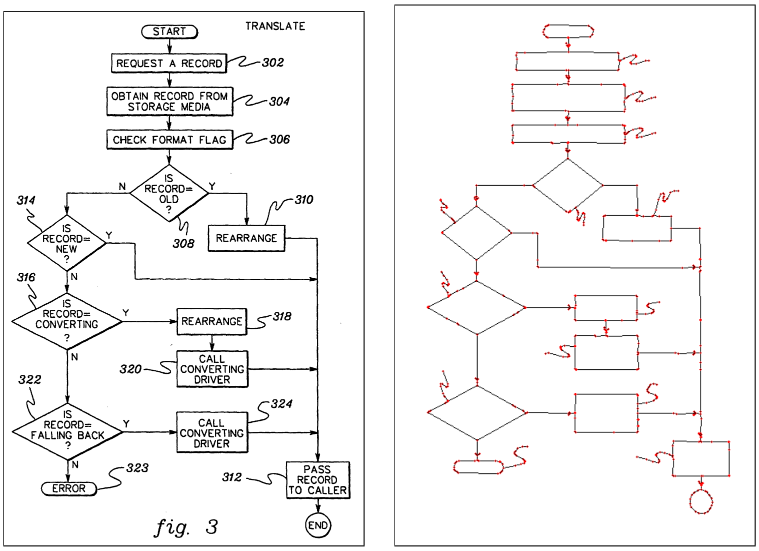
Two of these four thresholds depend on geometric properties of CCs. The first threshold T_1 sets a maximum size threshold for bounding boxes of CCs. More precisely, let A_{mp} and A_{avg} respectively be the most populated CC area and the CC average area, then $T_1 = n \max\{A_{mp}, A_{avg}\}$, where n is a parameter fixed beforehand. The second threshold T_2 is used to fix a maximum elongation threshold of bounding boxes. Thus, CCs with a ratio $\frac{height}{width}$ in the range $[\frac{1}{T_2}, T_2]$ and both magnitudes *height* and *width* below $\sqrt{T_1}$ are considered as *text*. Then, the algorithm computed the best enclosing rectangle of each CC classified as *text* and using two more thresholds to reclassify each component in the third layer if the algorithm consider that the CC is "small and elongated" component. These two new threshold concerns the density of the CC, with respect to the area of the best enclosing rectangle and the elongation of this rectangle. We can see an example of the results obtained by this text/graphics separation module in Fig. 2.

Raster-to-vector conversion. Some of the submitted runs are based on the analysis of vectorial data instead of working on the raw image domain. Thus we need a module that converts the raster image into a vectorial representation. We have used the raster-to-vector conversion method described in (Tombre et al, 2000). This algorithm first computes the skeleton of the image by means of a distance transform. Chains of pixels are then polygonally approximated by using the algorithm proposed in (Rosin and West, 1989) which is based on a recursive split-and-merge technique. We can see an example of the obtained results after the raster-to-vector conversion in Fig. 3.

Node and edge segmentation. The node segmentation module takes as input the graphical and textual layers obtained through the text/graphics separation module and outputs a list of bounding-boxes where the nodes are found. Thus, we have first segmented the symbolic nodes (i.e. *oval*, *rectangle*, *double-rectangles*, *parallelogram*, *diamond circle* and *cylinder*) and also the textual nodes (*no-box* nodes), corresponding to text which is not enclosed by any graphical entity.

In the current system CCs of interest are the ones corresponding to the white area inside the nodes and the main problem is to discriminate those from connected components corresponding to the background. After a preliminary step where very small and very large CCs are filtered, the remaining CCs are labeled as node candidates. However, not all the remaining components correspond to nodes since the white areas produced by loops formed by edges connecting nodes are also detected as candidates. In order to discriminate the real node components from the rest, we have used a couple of structural features:

– **Solidity**: computed as the ratio between the number of pixels in the CCs and the area of the convex hull of the CCs. Since nodes tend to be convex, objects below a solidity threshold are rejected. In our implementation, the solidity threshold was experimentally set to $t_{sol} = 0.89$.
– **Vertical symmetry**: computed as the ratio between the amount of pixels in the right and the left part of the CCs. Since nodes tend to be vertically symmetric, objects below a symmetry threshold are rejected. In our implementation, the symmetry threshold was experimentally set to $t_{sym} = 0.78$.



a) b)

Fig. 3 Example of the raster-to-vector conversion module. a) Original image, b) vectorial representation applied to the graphical layer.

Distinctively from the pixel-based segmentation approach, the vectorial-based node segmentation takes as input the vectorial representation of the images obtained after the raster-to-vector conversion, see Fig. 3 b). It is based on the exploration of loops in the planar graphs obtained from the vectorial images. In these graphs, nodes are the vectorial lines and edges are the connection points between these lines. The loop extraction process is driven by the implementation of the optimal algorithm for finding regions in a planar graph (Jiang and Bunke, 1993). As in the pixel-based segmentation method, we have also applied the solidity and vertical symmetry features introduced above to rule out inconsistent node candidate instances. Although both approaches are rather similar, in the sense that they both look at “closed things”, the difference is the input data they process, either the raw pixels or extracted vectors. When extracting such vectors, we decided not to use any pre-processing step at pixel level, the raster-to-vector conversion has been applied to the raw pixels from the graphical layer. Even the smallest gap in nodes or edges provokes the loose of the connectivity between loops and thus, that certain loop structures are not correctly retrieved. To overcome this issue, well-aligned vectors with small gaps are connected. Nevertheless, in our implementation some of those small gaps are still not correctly detected. In contrast, in the pixel-based approaches we applied a pre-processing step before the CCs analysis devoted to fill such small gaps by simply performing an *opening* operation with a linear structuring element.

The *no-box* nodes are extracted by analyzing the textual layer. We have removed text regions falling within detected nodes. Then, *no-box* nodes are the reminding CCs that have an edge that links them to a recognized node. Otherwise, text block outside nodes are disregarded or considered to be the title of the drawing if they are located in the upper part of the drawing. Those CCs are grouped together in terms of proximity by applying a mathematical morphology operations. Computing a *closing* operation with a rectangular structural element of size 25 × 100. We can see an example of the final node segmentation output in Fig. 4 b).

Finally, we have done edge segmentation in a similar fashion in both the pixel-based and the vectorial-based approaches. Candidate edges are the CCs obtained from the graphical layer after removing from it the detected nodes. We can see an example in Fig. 4 c).

2.2 Recognition modules

Modules devoted to recognition tasks take as input the result of applying segmentation modules. Three different modules have been developed, namely *text recognition*, *edge recognition* and *node recognition.*

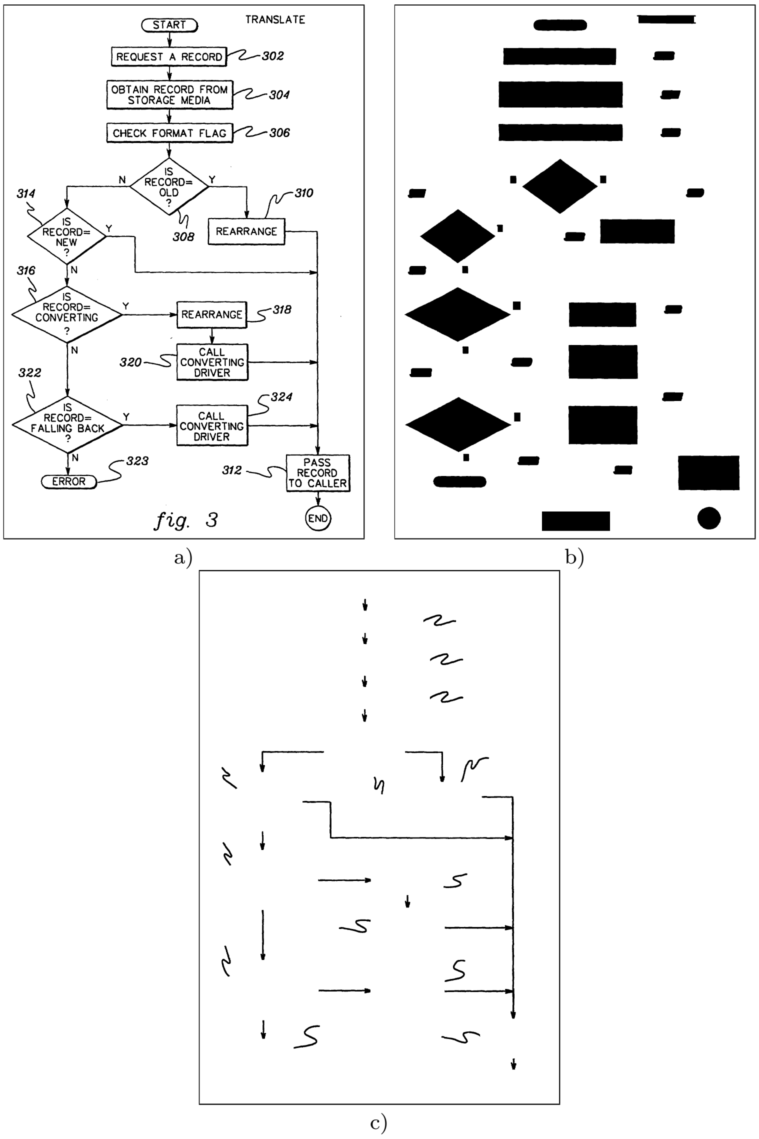
Text recognition. The module dealing with text recognition receives as input the textual layer obtained by the text/graphics separation module and the bounding-boxes arisen from the node segmentation module. The content of each bounding-box is processed by the commercial OCR from ABBYY[3]. No further post-processing steps are applied.

Node and edge recognition. We have developed two different approaches to perform node recognition. For both versions, we have used a nearest neighbor classifier built on a training dataset (Duda et al, 2001). Thus, the main difference between these two methods is the shape descriptor used in each version of the node recognition module.

In general, pattern recognition systems usually require shape descriptors invariant to similarity transforms (scale, translation, rotation) and even affine transforms. Therefore, many kinds of shape descriptors for graphics recognitions tasks, invariant to such transforms, have been proposed over the last decades (Zhang and Lu, 2004). However, in the context of flowchart recognition, the shape descriptors used to recognize the type of nodes just need to be invariant to translation and scale, while a lack of invariance to other transforms is actually beneficial as it results to increased discrimination capacities.

From the pool of shape descriptors, invariant to scale and translation but not to rotation and other affine transforms, we have selected a descriptor based on geometric moments (Zhang and Lu, 2002) and the Blurred Shape Model (BSM) descriptor (Escalera et al, 2009). Both descriptors are easy and fast to compute on node shape images. In addition, both descriptors have proven to perform reasonably well in pattern recognition problems with a low number of classes (type of nodes).

[3] ABBYY Finereader Engine 10: `http://www.abbyy.com/ocr_sdk/`



a) b)

c)

Fig. 4 Example of the node and edge segmentation modules. a) Original image, b) node layer, c) edge layer.

- **Geometric moments** have been widely used as shape descriptors since lower order moments represent certain well known fundamental geometric properties of the underlying image functions. We have used the geometric moments up to third order as a feature vector. The central $(p+q)$-th order moment for a digital image $I(x, y)$ is expressed by

$$\mu_{pq} = \sum_{x,y} (x - \bar{x})^p (y - \bar{y})^q I(x, y) \tag{1}$$

The use of the centroid $(\bar{x}, \bar{y})$ allow the descriptor to be invariant to translation. A normalization by the object area is used to achieve invariance to scale.

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \quad \text{where} \quad \gamma = \frac{p+q}{2} + 1 \tag{2}$$

- The **BSM descriptor** was originally created to perform handwritten musical score recognition but it has also been applied to other related document analysis with different degrees of success (Escalera et al, 2009). The BSM descriptor is a zoning-based descriptor. Shapes are divided into a 15 × 15 regular grid. Then, the area and the center of gravity are computed for each cell. The final BSM descriptor is constructed by weighting the areas computed by the inverse of distances between two gravity centers of adjacent cells. This weighted average is performed in order to achieve robustness to local variations of the shape under analysis. A normalization by the object area is used in order to achieve invariance to scale.

Finally, the step dealing with edge recognition only have to distinguish between directed and undirected edges. A simple analysis of the width of the edge stroke is enough to discriminate between these two classes of edges.

2.3 Flowchart Structure Inference

Once we have identified all the elements of a flowchart, we have to infer from the different relationships among elements which is the structure of the flowchart. More specifically, we have to assess which nodes are connected by an edge. Therefore, we have pair-wisely selected all the detected nodes and subsequently analyzed whether any element of the edge layer provokes that those two disjoint nodes merge into a single element. If this happens, then the two nodes are linked through this edge in the delivered graph structure.

Since node linking strongly depends on the extraction of CCs, the proposed system fails at detecting the *dotted* edges. Also, the proposed system tends to fail if the edge is broken by some text.

The final graph is syntactically analyzed to detected *point* nodes. When we find three or more nodes that are connected through exactly the same edge element, we add a new intermediate node of type *point.* We repeat this procedure up to there are no node junctions with cardinality higher than two.

3 Experimental Results

Let us first detail the CLEF-IP 2012 dataset for flowchart recognition and the evaluation protocol. We will then briefly detail the other participant methods and finally report the obtained results.

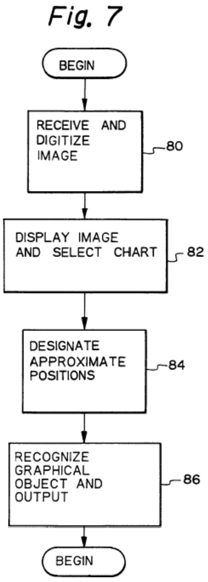
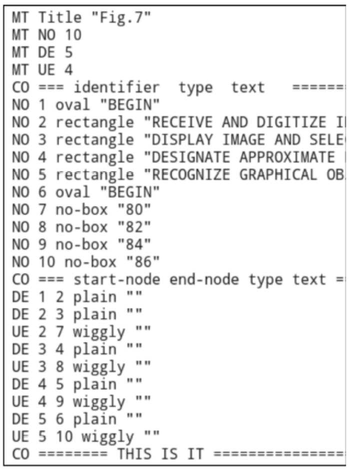


Fig. 7
BEGIN
RECEIVE AND DIGITIZE IMAGE
80
DISPLAY IMAGE AND SELECT CHART
82
DESIGNATE APPROXIMATE POSITIONS
84
RECOGNIZE GRAPHICAL OBJECT AND OUTPUT
86
BEGIN



```
MT Title "Fig.7"
MT NO 10
MT DE 5
MT UE 4
CO === identifier  type  text   ======
NO 1 oval "BEGIN"
NO 2 rectangle "RECEIVE AND DIGITIZE I
NO 3 rectangle "DISPLAY IMAGE AND SELE
NO 4 rectangle "DESIGNATE APPROXIMATE
NO 5 rectangle "RECOGNIZE GRAPHICAL OB
NO 6 oval "BEGIN"
NO 7 no-box "80"
NO 8 no-box "82"
NO 9 no-box "84"
NO 10 no-box "86"
CO === start-node end-node type text =
DE 1 2 plain ""
DE 2 3 plain ""
UE 2 7 wiggly ""
DE 3 4 plain ""
UE 3 8 wiggly ""
DE 4 5 plain ""
UE 4 9 wiggly ""
DE 5 6 plain ""
UE 5 10 wiggly ""
CO ======== THIS IS IT ===============
```

Fig. 5 An example of input image with its corresponding textual information (extracted from (Piroi et al, 2012)).

3.1 Dataset

The dataset used to evaluate our method is the one from CLEF-IP 2012 (Piroi et al, 2012). This public dataset contains 150 flowchart images cropped from real patent documents. All the images in the collection are in binary format and contain a single flowchart. The dataset is split into 50 training images with their corresponding textual information, see Fig. 5, and 100 test images. However, in the final benchmarking task at CLEF-IP 2012, only 44 of those test images have been used in the evaluation (Piroi et al, 2012).

As the ground-truth used to assess the systems' performance, an XML-like file describes the structure of each input flowchart in 3 parts:

- The tag **MT** refers to the meta information, such as the flowchart title, the number of nodes and the number of directed and undirected edges.
- The node information is referred with the tag **NO**, which is composed by the identifier, the type and the text appearing in it. There are 10 types of nodes available according to their shape: *oval, rectangle, double-rectangle, parallelogram, diamond, circle, point, cylinder, no-box* and *unknown.*
- The tags **DE** and **UE** correspond to the information of directed and undirected edges respectively. They are composed by the starting and ending node identifiers, their functionality and their associated text. The type of the edges can be either *plain* or *wiggly*, where *plain* refer to edges that connect nodes, and *wiggly* states for edges that relate nodes with text.

The available training data (50 flowcharts with their corresonding groundtuth released by the CLEF-IP 2012 organizers) has been used for parameter validation in the tasks of text/graphic separation, raster-to-vector conversion and node segmentation. In addition to that, we have manually labeled the types of a set of the nodes in the 50 training images. Such ground-truth allowed us to train the nearest-neighbor classifier in the node recognition step with a 4-fold cross-validation setup.

3.2 Evaluation Measures

The flowchart recognition task has been evaluated at three different levels. Namely, how well the flowchart structure has been recognized (*structural level*), how well the nodes and the edge types have been recognized (*recognition level*) and a third level that evaluates the text label transcription (*transcription level*).

In order to assess the methods' performance at structural level, a graph metric distance between the topic flowchart F_t and the submitted flowchart F_s is defined in terms of the most common subgraph, $mcs(F_t, F_s)$ (Bunke and Shearer, 1998), (Wallis et al, 2001). Formally, it is computed as follows

$$d(F_t, F_s) = 1 - \frac{|mcs(F_t, F_s)|}{|F_t| + |F_s| - |mcs(F_t, F_s)|}, \tag{3}$$

where $|F_i|$ denotes the size of the graph computed as the number of nodes plus the number of edges.

The most common subgraph measure can be interpreted as follows. When comparing a recognized flowchart F_s and the ground-truthed expected output F_t, the maximum common subgraph $mcs(F_t, F_s)$ measures how well the participant's output matches the expected graph. If the participant's method output is perfect, the maximum common subgraph is the flowchart itself and thus $mcs(F_t, F_s) = F_t = F_s$ and $d(F_t, F_s) = 0$. If the output is missing a node or some edges, the common structure shared between the output and the groundtruth will be smaller than F_t and consequently since $|mcs(F_t, F_s)| < |F_t|$, the final distance $d(F_t, F_s) > 0$ will increase as long as we keep missing elements. The same applies if we deliver an output with extra elements than the ground-truth.

The ability to recognize the nodes and the edges types of the different submitted runs is evaluated by the accuracy of the classification. Whereas the performance of the textual transcription is measures with a normalized edit distance between the automatically transcribed text and the yielded automatic transcription from the methods.

3.3 CLEF-IP 2012 Participant Methods

Thirteen different runs were submitted to the flowchart recognition task at CLEF-IP 2012 coming from three different institutions. We have detailed in this paper the four different runs ($R1$–$R4$) from the Computer Vision Center[4] (CVC) submitted at CLEF-IP 2012 (Rusiñol et al, 2012). An implementation[5] of our baseline system has been made available in order to allow the interested readers to test and study our approach.

The team from INRIA[6] submitted (Thean et al, 2012) a single run $I1$. After applying a text/graphics algorithm, nodes are segmented and features based on shape symmetry are used in order to recognize the different node shapes. The commercial OCR from ABBYY was also used here, although the authors proposed to use a post-correction technique that enhances the textual output.

The team from JOANNEUM Research [7] submitted (Mörzinger et al, 2012) eight different runs ($N1$–$N8$). The flowchart recognition procedure was based on a connected component analysis and posterior vectorization aimed at segmenting the nodes and finding the linking edges. Nine ad-hoc features were proposed to recognize the node shapes and the Transym OCR engine was used to transcribe the textual elements. The 8 different runs are the result of adapting each module of the system to increase the final performance. Thus, over the system baseline (run $N1$), different adaptations and parameters tuning are presented to mainly increase the node segmentation and recognition and the edge segmentation.

3.4 Results

Let us first evaluate the proposed system's performance qualitatively by looking at specific cases where we face some problems and then present the obtained quantitative results.

Although visually the system seems to perform quite well, we have identified several cases where the proposed modules fail, examples of which are presented in Fig. 6.

First of all, the node and edge segmentation modules are based on CCs. When either a node (Fig. 6a)) or an edge (Fig. 6b)) is broken it is usually not well segmented by any of the proposed methods. In addition, low-quality documents are hard to "read" by the OCR engine (Fig. 6c)). Finally, the text/graphics separation module is also based on the analysis of CCs, so when text characters overlap with graphical elements, they are not properly segmented. This is the case shown in Fig. 6d), where the character *F* of *FROM* and the character *D* from *FIELD* touch the diamond shape and thus are classified as graphics and assigned to the graphical layer.

The quantitative evaluation has been separated in three different levels. We first present in Fig. 7a) the structural level in which the methods are evaluated in terms of their abilities to correctly extract the flowchart structure.

[4] http://www.cvc.uab.cat/

[5] Code available at: `http://www.cvc.uab.es/~marcal/demos/flowchart.html`

[6] http://www.inria.fr/

[7] http://www.joanneum.at/

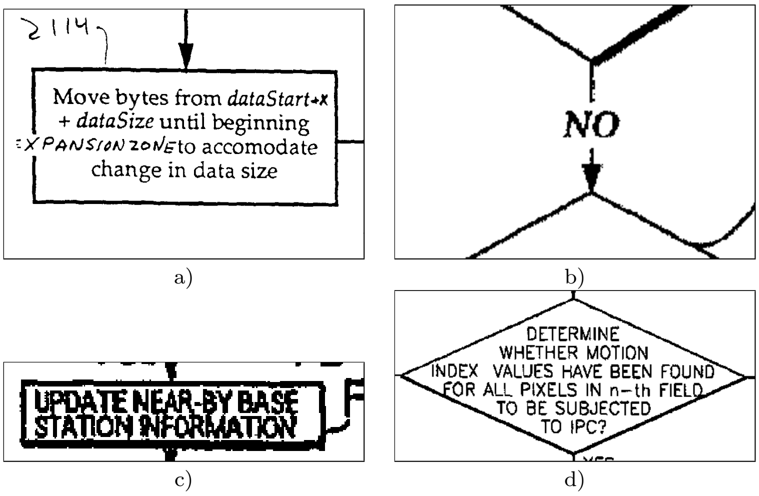

2114
Move bytes from dataStart+x
+ dataSize until beginning
EXPANSION ZONE to accomodate
change in data size
a)
NO
b)
UPDATE NEAR-BY BASE
STATION INFORMATION
c)
DETERMINE
WHETHER MOTION
INDEX VALUES HAVE BEEN FOUND
FOR ALL PIXELS IN n-th FIELD
TO BE SUBJECTED
TO IPC?
d)

Fig. 6 Problematic cases. a) Broken node, b) broken edge, c) low-quality text, d) text/graphics overlapping.

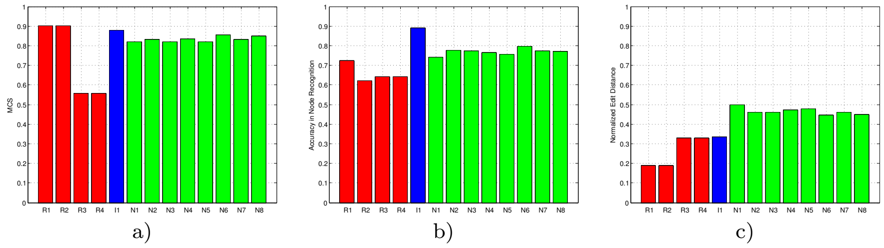
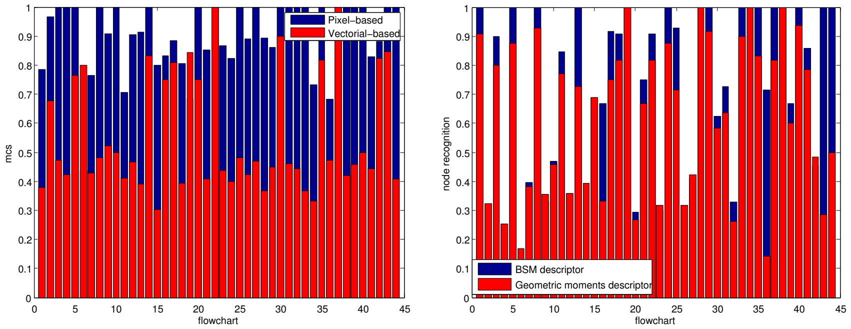


Fig. 7 Evaluation results at CLEF-IP2012 a) of the transcription level computed in terms of the normalized edit distance, b) of the structural level computed in terms of the most common subgraph, and c) of the recognition level computed in terms of the accuracy at node classification.

As detailed above, this structural correctness of the delivered results is computed in terms of the most common subgraph between the obtained results and the ground-truth graphs. At first glance we can see that the two runs ($R3$ and $R4$) that were based on the vectorial representation perform much worse than the ones working at pixel level ($R1$ and $R2$). This fact can be appreciated in Fig. 8a) where we show the maximum common subgraph achieved at each flowchart under evaluation for both pixel and vectorial-based runs. In the vectorial approaches we yield a $d = 0.56$ whereas the pixel-based approaches delivered a $d = 0.9$. Although both approaches are rather similar, in the sense that they both look at “closed things”, the difference is the input data they process, either the raw pixels or extracted vectors. When extracting such vectors, we decided not to use any pre-processing step at pixel level, the raster-to-vector conversion has been applied to the raw pixels from the graphical layer. Even the smallest gap in nodes or edges provoke to loose the



a)

b)

Fig. 8 Analysis of the obtained a) mcs and b) node recognition for each of the 44 flowcharts depending on the different modules. a) Pixel-based segmentation versus vectorial-based analysis. b) BSM node descriptor versus Geometric moments description.

connectivity between loops and thus provoked that certain structures are not correctly retrieved. In contrast, in the pixel-based approaches we applied a pre-processing step before the CCs analysis devoted to fill such small gaps, that fixed such problems. When comparing the performance of $R1$ and $R3$ with the rest of submitted runs, we can see in Table 2 that at structural level, the proposed architecture outperforms the rest of the runs.

We present in Fig. 7b) the accuracies reached at node recognition. In that case the node recognition methodology presented in (Thean et al, 2012) corresponding to the $I1$ run is the one that performs the best. The squeezing operation the authors propose in order to normalize shapes presenting great variations such as ovals and diamonds into a canonical form, has proven its discriminative power with respect to the rest of submitted runs. In our submitted runs, we can see that the BSM descriptor $R1$ performs better than the geometric moments $R2$ for the nodes extracted with the pixel-based approach. We present in Fig. 8b) the reached accuracies per flowchart when using either the BSM or the geometric moment descriptor. We can see that in most of the flowcharts, the BSM steadily outperforms the geometric moment descriptor. Regarding the vectorial approach, both descriptors perform equally, but the amount of "detected" nodes is much smaller as pointed earlier in the structural level evaluation. The fact that better recognition rates have been achieved by the other participants, proves that in the specific scenario of flowchart recognition ad-hoc node descriptors outperform state-of-the-art descriptors proposed to be used for generic purposes.

Finally, regarding the transcription level, all the participants used off-theshelve OCR engines. In Fig. 7c) we can see the performances measures in terms of the normalized edit distance between the transcribed text and the ground-truth one. Here it is worth to note that in our configurations, we fed to the OCR the segmented regions of the documents in which we detected some text. The fact the run $I1$ use the ABBYY technology as well but obtains worse results than our runs $R1$ and $R2$ (0.33 in $I1$ against a 0.19 in $R1$–$R2$) highlights

Table 2 Summary of the evaluation results at CLEF-IP 2012 for the three participants

Method	Structural Level	Recognition Level	Transcription Level
	mcs	*accuracy*	*norm. edit distance*
CVC, Our proposal (Rusiñol et al, 2012)	**0.9026**	0.7250	**0.1888**
INRIA (Thean et al, 2012)	0.8789	**0.8909**	0.3337
JOANNEUM (Mörzinger et al, 2012)	0.8563	0.7982	0.4461

the importance in such scenarios to provide an accurate text segmentation to the OCR. It is also worth to note that in our approach we do not post-process the OCR output nor build any specific language model. As proven in (Thean et al, 2012) an addition of such post-processing steps or an ad-hoc tuning of the OCR engine for the specific purpose of “reading” flowchart’s text for sure will provide a much more accurate transcription output.

We provide in Table 2 a per-participant summary of the evaluation results in the CLEF-IP 2012 for all the three different levels. We can appreciate the proposed architecture outperform the rest of the submitted runs at structural and transcription level, whereas better recognition rates have been achieved by the other participants.

4 Conclusions

In this paper we have presented a modular flowchart recognition system. For building such modules we have proposed to apply state-of-the-art text/graphic recognition techniques, segmentation methods, general purpose shape descriptors, which are well-known in the graphics recognition community, and a commercial OCR. The large amount of existing literature in some of these modules has motivated us to test four different versions of the proposed architecture. Two methods working at pixel-based flowchart representation and two methods working at vectorial-based flowchart representation. In addition, we have tested two general purpose shape descriptors for the node recognition module. The reported results show that some of the configurations of the different presented modules (specially the R1 run) provide interesting results for the given problem of flowchart recognition.

The proposed method outputs a structured representation of flowchart images, which can later be semantically queried to retrieve related information conveyed in this sort of graphical drawings. Such graphics understanding techniques can be very beneficial in the patent search domain by providing a complementary retrieval ability to the image search by visual similarity approaches.

Thanks to the CLEF-IP flowchart recognition task, some flaws of the current method have been identified. First, since we have based the node segmentation on the seek of closed structures, the proposed methods are quite sensitive to noise when the nodes are broken either due to bad acquisition or to design purposes. It would be beneficial to test other segmentation approaches based on Gestalt laws that should not be that sensitive. On the the other hand, we have applied off-the-shelf well-known shape descriptors with the idea that being general-purpose they will perform well a well in such scenario. The evaluation has proven that using ad-hoc shape descriptors to the specific tasks of recognizing the shapes that are possible within the flowchart lexicon outperforms the generic shape descriptors. Further research on such dedicated descriptors would enhance the system's performance.

Acknowledgements The authors would like to thank Mihai Lupu, Florina Piroi and Allan Hanbury for their support and assistance in our participation in CLEF-IP 2012 task and for their advice and counsel in the field of retrieval in intellectual property domain. This work has been partially supported by the Spanish Ministry of Education and Science under projects TIN2009-14633-C03-03 and TIN2012-37475-C02-02, and by a research grant of the Universitat Autonòma de Barcelona (471-02-1/2010).

References

Adams S (2005) Electronic non-text material in patent applications – some questions for patent offices, applicants and searchers. World Patent Information 27(2):99–103

Bhatti N, Hanbury A (2012) Image search in patents: a review. International Journal on Document Analysis and Recognition DOI 10.1007/s10032-012-0197-5

Blostein D (1996) General diagram-recognition methodologies. In: Graphics Recognition Methods and Applications, Lecture Notes in Computer Science, vol 1072, Springer-Verlag, pp 106–122

Bunke H (1982) Attributed programmed graph grammars and their application to schematic diagram interpretation. IEEE Transactions on Pattern Analysis and Machine Intelligence 4(6):574–582

Bunke H, Shearer K (1998) A graph distance metric based on the maximal common subgraph. Pattern Recognition Letters 19(3–4):255–259

Codina J, Pianta E, Vrochidis S, Papadopoulos S (2008) Integration of semantic, metadata and image search engines with a text search engine for patent retrieval. In: Proceedings of the Workshop on Semantic Search at the Fifth European Semantic Web Conference, pp 14–28

Csurka G, Renders J, Jacquet G (2011) XRCE's participation at patent image classification and image-based patent retrieval tasks of the CLEF-IP 2011. In: CLEF 2011 Evaluation Labs and Workshop, Online Working Notes

Duda R, Hart PE, Stork DG (2001) Pattern Classification. Wiley-Interscience

Escalera S, Fornés A, Pujol O, Radeva P, Lladós GSJ (2009) Blurred shape model for binary and grey-level symbol recognition. Pattern Recognition Letters 30(15):1424–1433

Fletcher L, Kasturi R (1988) A robust algorithm for text string separation from mixed text/graphics images. IEEE Transactions on Pattern Analysis and Machine Intelligence 10(6):910–918

Hanbury A, Bhatti N, Lupu M, Mörzinger R (2011) Patent image retrieval: a survey. In: Proceedings of the Fourth Workshop on Patent Information Retrieval, pp 3–8

Huet B, Kern N, Guarascio G, Merialdo B (2001) Relational skeletons for retrieval in patent drawings. In: Proceedings of the International Conference on Image Processing, pp 737–740

Jiang X, Bunke H (1993) An optimal algorithm for extracting the regions of a plane graph. Pattern Recognition Letters 14(7):553 – 558

Lamiroy B, Najman L, Ehrard R, Louis C, Quelin F, Rouyer N, Zeghache N (2001) Scan-to-XML for vector graphics: an experimental setup for intelligent browsable document generation. In: Proceedings of the 4th IAPR International Workshop on Graphics Recognition, pp 312–325

Lew M, Sebe N, Djeraba C, Jain R (2006) Content-based multimedia information retrieval: State of the art and challenges. ACM Transactions on Multimedia Computing, Communications, and Applications 2(1):1–19

Lin X, Shimotsuji S, Minoh M, Sakai T (1985) Efficient diagram understanding with characteristic pattern detection. Computer Vision, Graphics, and Image Processing 30(1):84–106

List J (2007) How drawings could enhance retrieval in mechanical and device patent searching. World Patent Information 29(3):210–218

Lladós J, Rusiñol M (2013) Handbook of Document Image Processing and Recognition, Springer, chap Graphics Recognition Techniques

Lupu M, Schuster R, Mörzinger R, Piroi F, Schleser T, Hanbury A (2012) Patent images – a glass-encased tool: opening the case. In: Proceedings of the Twelveth International Conference on Knowledge Management and Knowledge Technologies

Mahmoudi F, Shanbehzadeh J, Eftekhari-Moghadam A, Soltanian-Zadeh H (2003) Image retrieval based on shape similarity by edge orientation autocorrelogram. Pattern Recognition 36(8):1725–1736

Mörzinger R, Horti A, Thallinger G, Bhatti N, Hanbury A (2011) Classifying patent images. In: CLEF 2011 Evaluation Labs and Workshop, Online Working Notes

Mörzinger R, Schuster R, Horti A, Thallinger G (2012) Visual structure analysis of flow charts in patent images. In: CLEF 2012 Evaluation Labs and Workshop, Online Working Notes

Piroi F, Lupu M, Hanbury A, Zenz V (2011) CLEF-IP 2011: Retrieval in the intellectual property domain. In: CLEF 2011 Evaluation Labs and Workshop, Online Working Notes

Piroi F, Lupu M, Hanbury A, Sexton A, Magdy W, Filippov I (2012) CLEF-IP 2012: Retrieval experiments in the intellectual property domain. In: CLEF 2012 Evaluation Labs and Workshop, Online Working Notes

Rosin P, West G (1989) Segmentation of edges into lines and arcs. Image and Vision Computing 7(2):109–114

Rusiñol M, de las Heras L, Mas J, Terrades O, Karatzas D, Dutta A, Sánchez G, Lladós J (2012) CVC-UAB's participation in the flowchart recognition task of CLEF-IP 2012. In: CLEF 2012 Evaluation Labs and Workshop, Online Working Notes

Samet H, Webber R (1985) Storing a collection of polygons using quadtrees. ACM Transactions on Graphics 4(3):182–222

Sidiropoulos P, Vrochidis S, Kompatsiaris I (2011) Content-based binary image retrieval using the adaptive hierarchicaldensity histogram. Pattern Recognition 44(4):739–750

Szwoch W (2007) Recognition, understanding and aestheticization of freehand drawing flowcharts. In: Proceedings of the Ninth International Conference on Document Analysis and Recognition, pp 1138–1142

Thean A, Deltorn J, Lopez P, Romary L (2012) Textual summarisation of flowcharts in patent drawings for CLEF-IP2012. In: CLEF 2012 Evaluation Labs and Workshop, Online Working Notes

Tiwari A, Bansal V (2004) PATSEEK: Content based image retrieval system for patent database. In: Proceedings of the Fourth International Conference on ElectronicBusiness, pp 1167–1171

Tombre K, Ah-Soon C, Dosch P, Massini G, Tabbone S (2000) Stable and robust vectorization: How to make the right choices. In: Graphics Recognition Recent Advances, Lecture Notes in Computer Science, vol 1941, Springer-Verlag, pp 3–18

Tombre K, Tabbone S, Pélissier L, Lamiroy B, Dosch P (2002) Text/graphics separation revisited. In: Document Analysis Systems, Lecture Notes in Computer Science, vol 2423, Springer-Verlag, pp 615–620

Valveny E, Lamiroy B (2002) Scan-to-XML: Automatic generation of browsable technical documents. In: Proceedings of the 16th International Conference on Pattern Recognition, pp 188–191

Vasudevan B, Dhanapanichkul S, Balakrishnan R (2008) Flowchart knowledge extraction on image processing. In: Proceedings of the IEEE International Joint Conference on Neural Networks, pp 4075–4082

Vrochidis S, Papadopoulos S, Moumtzidou A, Sidiropoulos P, Pianta E, Kompatsiaris I (2010) Towards content-based patent image retrieval: A framework perspective. World Patent Information 32(2):94–106

Vrochidis S, Moumtzidou A, Kompatsiaris I (2012) Concept-based patent image retrieval. World Patent Information 34(4):292–303

Wallis W, Shoubridge P, Kraetz M, Ray D (2001) Graph distances using graph union. Pattern Recognition Letters 22(6–7):701–704

Yu Y, Samal A, Seth S (1997) A system for recognizing a large class of engineering drawings. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(8):868–890

Yuan Z, Pan H, Zhang L (2008) A novel pen-based flowchart recognition system for programming teaching. In: Advances in Blended Learning, Lecture Notes in Computer Science, vol 5328, Springer-Verlag, pp 55–64

Zhang D, Lu G (2002) A comparative study of three region shape descriptors. In: Proceedings of the Digital Image Computing Techniques and Applications, pp 1–6

Zhang D, Lu G (2004) Review of shape representation and description techniques. Pattern Recognition 37:1–19