# A Fast CBIR System of Old Ornamental Letter

Mathieu Delalandre[1], Jean-Marc Ogier[2], and Josep Lladós[1]

[1] CVC, Barcelona, Spain
{mathieu;josep}@cvc.uab.es
[2] L3i, La Rochelle, France
jean-marc.ogier@univ-lr.fr

**Abstract.** This paper deals with the CBIR of old printed graphics (of XVI° and XVII° centuries) like the headpieces, the pictures and the ornamental letters. These graphical parts are previously segmented from digitized old books in order to constitute image databases for the historians. Today, large databases exist and involves to use automatic retrieval tools able to process large amounts of data. For this purpose, we have developed a fast retrieval system based on a Run Length Encoding (RLE) of images. We use the RLE in an image comparison algorithm using two steps: one of image centering and then a distance computation. Our centering step allows to solve the shifting problems usually met between the segmented images. We present experiments and results about our system concerning the processing time and the retrieval precision.

## 1 Introduction

This paper deals with the topic of CBIR[3] applied to the document images. During the last years many works have been done for the retrieval of journals, forms, maps, drawings, musical scores, etc. In this paper we are interested on a new retrieval application: the one of old printed graphics. Since the Digital Libraries development in the years 90's numerous works of digitization of historical collections have been done. These old books are composed of text but also of various graphical parts like the headpieces, the pictures and the ornamental letters. From the whole digitized pages these graphical parts are segmented (in a manual or automatic way [1]) in order to constitute image databases for the historians. Some example of famous databases are the Ornamental Letters DataBase[4] and the International Bank of Printers' Ornaments[5].

These databases are composed today of thousands of images that involves to use automatic retrieval tools able to process large amounts of data. Past works have been already proposed on this topic [2] [3] [4] [5]. In [2] the authors propose a system to retrieve similar illustrations composed of strokes using orientation radiograms. The radiograms are computed at a $\frac{\pi}{4}$ interval and used as image signatures for the retrieval. [3] looks for common sub-parts in the old figures. The segmentation is done by a local computation of the Hausdorff distance using a sliding window. A voting algorithm

[3] Content Based Image Retrieval

[4] http://www.bvh.univ-tours.fr/oldb.asp (10 000 images)

[5] http://www2.unil.ch/BCUTodai/app (3000 images)

manages the location process. In [4] a statistical scheme is proposed to retrieve the ornamental letters according a style criterion. Patterns (ie. gray-level configuration of a pixel neighborhood) are extracted from the image and next ranked to build a curve descriptor. The retrieval is done by a curve matching algorithm. In [5] the layout of the ornamental letters is used for the retrieval. A segmentation process extracts the textured and uniform regions from images. Minimum Spanning Tree (MST) are next computed from these regions and used as image signatures for the retrieval.

All the existing systems are dedicated to a specific kind of retrieval (style, layout, etc.). In this paper we focuss on the application of wood plug tracking illustrated in the Figure 1. Indeed, from the 16th to the 17th centuries the plugs, used to print the graphics in the old books, were mainly in wood. These wood plugs could be re-used to print several books, be exchanged between printing houses, or duplicated in the case of damage. So to retrieve, in automatic way, printings produced by a same wood plug could be very useful for the historian people. It could solve some dating problems of books as soon as to highlight the existing relations between the printing houses.

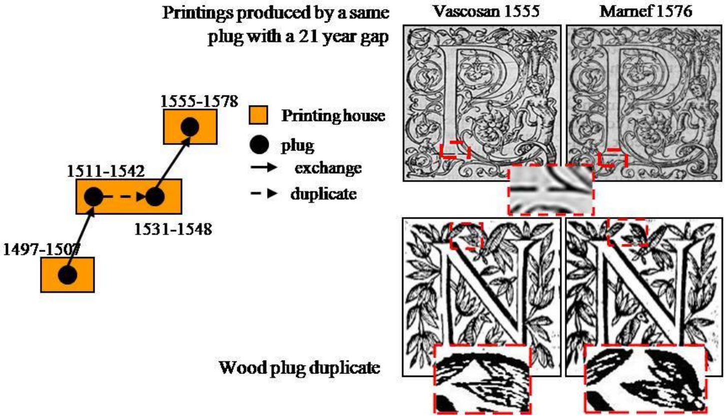


**Fig. 1.** Wood plug tracking

This retrieval application is an image comparison task [6]. Indeed, the images produced by a same wood plug present similarities at pixel level. However, it raises a complexity problem. First in regard to the amount of data, building a comparison index between thousands of image could require days of computation. Next in regard to the copyright aspects. The images belong to specific Digital Libraries or private collections; in order to allow crossed queries between these databases real-time retrieval processes are required. In regard to these specificities we have developed a system to perform a fast retrieval of images. This one uses two main steps as shown in the Figure 2: one of Run Length Encoding of images and the other of image comparison (centering and distance computation). We will present both of them in the next sections 2 and 3. Then, in the section 4 we will present some experiments and results about our system. At last, we will conclude and will give perspectives in the section 5.

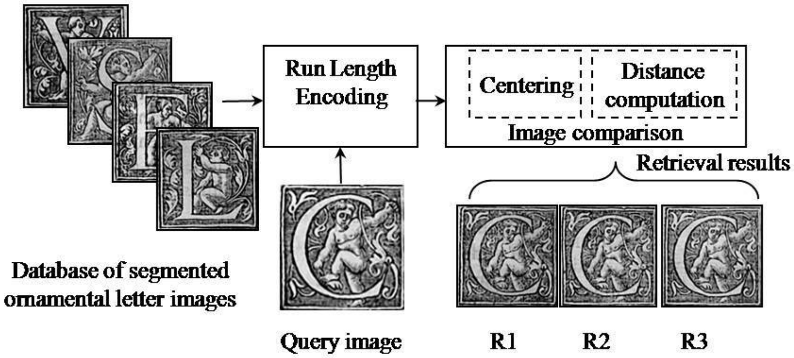


**Fig. 2.** Our system

## 2 Run-Length Encoding

Our purpose is to retrieve, in a fast way, the images similar to a query in a large database. To do it it is necessary to decrease the processing time of the retrieval system. One way is to exploit a special hardware architecture [7] like the pipe-line processor or the mesh-connected computer. It is an expensive solution which makes difficult the spreading of the developed systems. The other way is to use a compressed data structure to represent the images in order to decrease their handling times. It exists few works dealing with this topic, some examples are the ones of [8] and [9]. In [8] the authors use a connected-component based representation to perform a fast retrieval of images based on their layout. The system of [9] employs a contour based representation of images in order to perform fast neighboring operations like the erosion, the dilatation or the skeletonization.

For our problematic we have considered the run based representation [10]. The run is well known data structure. As explained in the Definition 1, it encodes successive pixels of same intensity into a single object. The conversion of a raster based representation to a run based representation is called Run-Length Encoding (RLE) in the literature. The Figure 3 gives an example of RLE.

**Definition 1.** *A run is maximal sequence of pixels defined by* ***o*** *the orientation (either vertical or horizontal),* ***(x,y)*** *the starting point,* ***l*** *the length and* ***c*** *its color.*

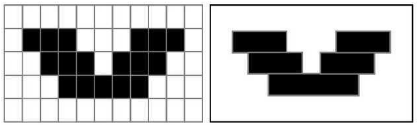


**Fig. 3.** Example of Run-Length Encoding (RLE)

In the past the run based systems have been widely used for the document recognition. The first system has been proposed by [11]. Then, several ones have been developed during the years 90's [10] and nowadays they are used for various applications: handwriting recognition [12], symbol recognition [13], structured document segmentation [14], etc. Concerning the use of run for the complexity only the following systems have been proposed up to day: [15] for the contour extraction, [16] for the image comparison and [17] for the morphology operations. Using the RLE, the final sizes of images are reduced. The compression rate defines itself as the ratio between the number of run and the one of pixel. Next, the algorithms have to work on the RLE space to perform faster operations. However, different criteria can influence the compression rate. First, the RLE have to be extracted from binary images and the previous step of color quantization will have a great impact on the encoding results. Next, the RLE can be applied to the foreground and/or to the background of images. In this way, three encodings can be considered as presented in the Figure 4 and each of them will give a different compression result. Finally the RLE can be also done in different ways: vertical, horizontal, zig-zag or others. According to the content of an image this will change the result of the RLE.

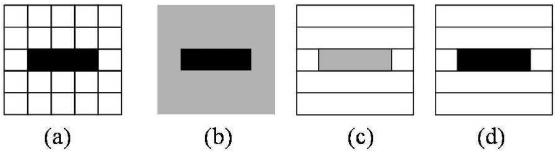


**Fig. 4.** RLE types
(a) raster (b) foreground RLE (c) background RLE (d) foreground/background RLE

For our application we have chosen to use the foreground/background encoding. This seems more adapted for the comparison of ornamental letter images where the objects (letter, character, etc.) appear as soon as on the foreground and the background of the images. Next, we perform this encoding from gray-level images by applying a binarization step with a fixed threshold. Indeed, the processed images by our system have been previously cleaned (lighting correction, filtering, etc.) by the digitalization platforms of old books. Finally, due to the property of visual symmetry of the ornamental letter images (on on both sides of letters) we have chosen to apply a vertical RLE.

## 3 RLE based Image Comparison

In this section we present our image comparison algorithm based on the RLE. As presented in the introduction part, the images processed by our system have been previously segmented from old books. It introduces shifting problems between the images which makes more harder their comparison. In order to solve this problem our comparison algorithm uses two steps: one of image centering and one other of distance computation. We present each of them in what follows.

Our image centering step exploits horizontal and vertical projection histograms of pixels. The Figure 5 gives examples of such projection histograms. These ones are built during the indexing step (with the RLE) from the black pixels of images. We use the black pixels because the segmentation process adds background areas around the ornamental letter. The centering parameters deduce themselves by the foreground analysis.

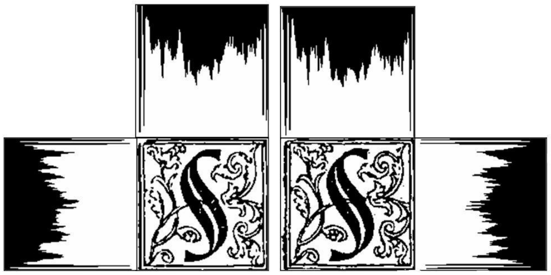


**Fig. 5.** Vertical and horizontal projection histograms

We center next two images together by computing the distances between their histograms (vertical and then horizontal). To do it we have chosen the distance presented in the Equation **1**. It is a weighted distance between two histograms $g$ and $h$. We have chosen the weighting because it increases the robustness of the comparison when strong amplitude variations appear in the histograms [18]. Our images could be of different sizes, so we compute our weighted distances using an offset in pixel (from $0$ to $l - k$). $k$ and $l$ are the lengths of $g$ and $h$ with $l$ the higher value ($h$ is the largest histogram). Considering two images to center, $g$ and $h$ are chosen when staring the centering step by finding the minimum/maximum widths and heights. The delta to use, ether x or ether y, corresponds to the found minimum weighted distance among the computed offsets (from $0$ to $l - k$). The previous Figure 5 corresponds to the deltas $d_x = 1$ and $d_y = 4$.

$$\begin{matrix} g_{1,2,..,k} \\ h_{1,2,..,l} \\ k \leq l \end{matrix} \qquad delta = \min\left( \bigcup_{j=0}^{l-k} \sum_{i=1}^{k} \frac{|(h_i - g_{i+j})|}{h_i} \right) \tag{1}$$

In a second step we compute a distance between our images. This distance is obtained by a “simple” comparison pixel to pixel [6]. However, to compute it our comparison works obviously from the RLE representation of images. We present here the algorithm that we use[6].

[6] Presentation based on the LaTeX package Pseudocode [19].

**Pseudo-algorithm 3.1:** DISTANCE$(i_1, i_2, d_x, d_y)$

```
$s \leftarrow 0$
$x_1 \leftarrow x_2 \leftarrow 0$
$a_1 \leftarrow a_2 \leftarrow 0$
for each line $L1$ at $y$ of $i_1$ and $L2$ at $y + d_y$ of $i_2$
  do
    $p_1 \leftarrow$ NEXT$(L1)$
    $x_1 + = p_1.length$
    $p_2 \leftarrow$ NEXT$(L2)$
    $x_2 + = p_2.length$
    while $(p_1 \neq end) \vee (p_2 \neq end)$
      do
        while $x_1 \geq (x_2 + d_x)$
          do
            if $p_2.color = p_1.color$
              then $s + = p_2.lenght - a_2$
            $p_2 \leftarrow$ NEXT$(L2)$
            $x_2 + = p_2.lenght$
            $a_1 + = p_2.lenght - a_2$
            $a_2 = 0$
        while $(x_2 + d_x) \geq x_1$
          do
            if $p_1.color = p_2.color$
              then $s + = p_1.lenght - a_1$
            $p_1 \leftarrow$ NEXT$(L1)$
            $x_1 + = p_1.lenght$
            $a_2 + = p_1.lenght - a_1$
            $a_1 = 0$
$s \leftarrow s / (\min(i_1.width, i_2.width) \times i_1.height)$
```

Our algorithm uses the vertical runs to compare two given images $i_1$ et $i_2$. It browses all the lines $L1$ and $L2$ of these images at the coordinates $y$ and $y + d_y$. For each couple of line, it browses alternately the runs using two variables $\{p_1, p_2\}$. The Figure 6 explains this run browsing. Two markers $\{x_1, x_2\}$ are used to indicate the current positions of the browsing. The browsed line is already the one of lower position (tacking into account the $d_x$ offset of the centering step). The latest read run of the upper position is used as reference run. The runs of the browsed line are summed using a variable $s$ if they are of the same color than the reference run. During the alternately browsing two stacks $\{a_1, a_2\}$ are used. These last ones allow to deal the browsing switch ($L1 \leftrightarrows L2$). For this purpose, they sum the browsed distances on each line using the reference runs.

## 4 Experiments and Results

In this section we present experiments and results about our system. For this purpose we have tested our system on the Ornamental Letter Database[4]. In this database we have selected 2048 gray level images digitized from 250 to 350 dpi. The full size of these images (in uncompressed mode) is of 268 Mo (a mean size of 131 Ko per image). We present here experiments and results of our system concerning three criteria: the compression rate, the comparison time and the retrieval precision.

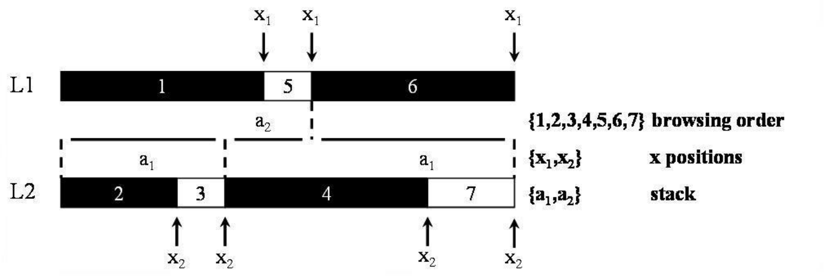


**Fig. 6.** Run browsing

In a first step we have computed the RLE of images in order to obtain their compression rates. The Figure 7 shows our results. We have obtained a mean rate $\overline{r_c} = 0.88$ on the whole database with minimum and maximum ones of $0.75$ and $0.95$. These results show that RLE has reduced of 88 % the sizes of images, so from 8 to 9 times. The Figure 7 gives also examples of ornamental letter image corresponding to the characteristic rates min, mean and max. The better rates are obtained for the images composed of strongly homogeneous regions whereas the lower ones correspond to textured images (which produce lot of heterogeneous regions).

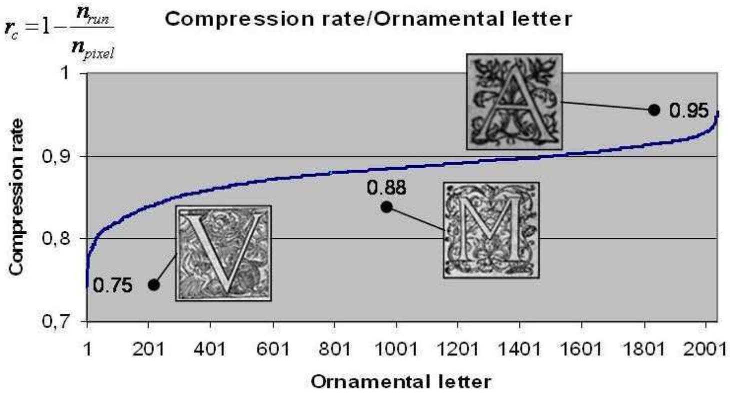
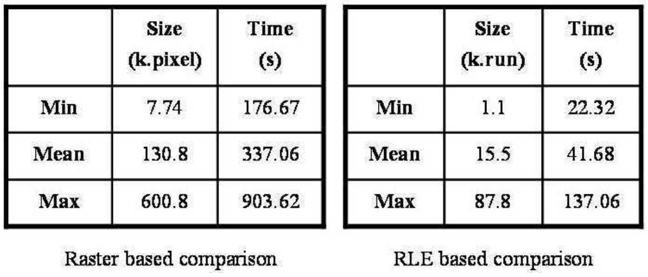


**Fig. 7.** Compression rates of the ornamental letter images

We have next evaluated the retrieval times of our system. To do it we have performed a query with each of the images of the database. We have compared each of these queries with all the other images of the database. The comparison is done in two steps, one of image centering and then the distance computation. From the time results we have looked for the min, mean and max ones. In order to compare these values we have also done the same experiments but using a classical image comparison algorithm working from a raster based representation. The both algorithms have been implemented in C++ and tested on a laptop computer using a 2GHz Pentium processor working with a Windows XP operating System. Our results are presented in the Figure 8. We have obtained a mean time less to one minute with a our approach contrary to the several ones needed with the raster based comparison. In any cases, our approach allows to execute the queries within two minutes whereas the raster based comparison can take until a $\frac{1}{4}$ hour.



|  | Size (k.pixel) | Time (s) |
|---|---|---|
| Min | 7.74 | 176.67 |
| Mean | 130.8 | 337.06 |
| Max | 600.8 | 903.62 |

Raster based comparison

|  | Size (k.run) | Time (s) |
|---|---|---|
| Min | 1.1 | 22.32 |
| Mean | 15.5 | 41.68 |
| Max | 87.8 | 137.06 |

RLE based comparison

**Fig. 8.** Time retrieval results

At last we have performed, in a random way, some queries in order to evaluate the retrieval precision of our system. The Figure 9 gives an example of query result. In regard to this kind of result our system seems allowing an efficient retrieval of the ornamental letter images. Indeed, as explained previously the image is done at a pixel level. It gives a precise comparison of the images allowing to obtain good retrieval results. The remained retrieval problems concern the very damaged ornamental letter images which appear in the case of broken plugs, ripped parts, darkness papers, bad curvatures, etc.

## 5 Conclusion and Perspectives

In this paper we have presented a system dealing with the CBIR of old printed graphics (headpieces, pictures and ornamental letters of the XVI° and XVII° centuries). The aim of our system is to process large image databases. For this purpose, we have developed a fast approach based on a Run Length Encoding (RLE) of images. This one allows to reduce the image sizes and then their handling times for the comparison. The core part of our system is an image comparison algorithm. This one uses two steps: one of image centering following by a distance computation. Like this, the centering step allows to solve the shifting problems usually met between segmented images. We have presented different experiments and results about our system. We have shown how our system allows to compress from 8 to 9 times the image sizes, and therefore to reduce the needed retrieval time. We have also illustrated the retrieval precision of our system through an example of query result.

The perspectives concerning this work are of two types. In a first step we work now on a selection process of images based on global features. The key idea to this work is

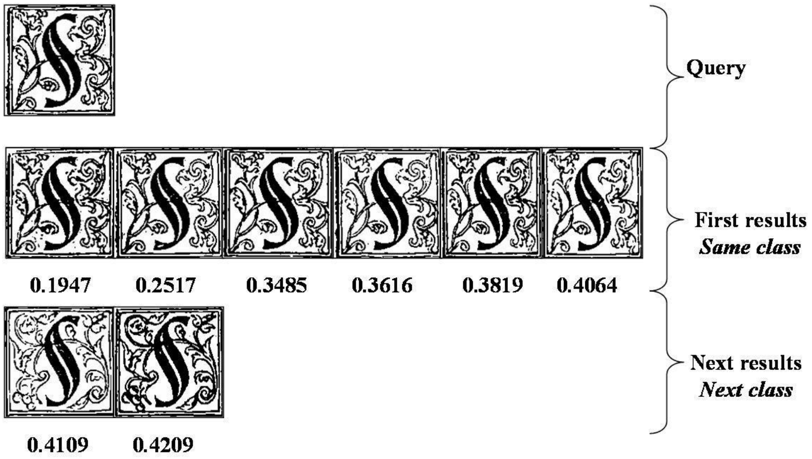


**Fig. 9.** Example of query result

to reduce previously the comparison space by rejecting the images too different from the query one in order to speed-up the retrieval. In a second step we want to evaluate our retrieval results. However, this needs to acquire the ground-truth from the ornamental letter images. Editing the ground-truth in an hand user way could be a long and harder work which could introduce lot of errors. Our key idea to solve this problem is use our system as a ground-truthing one in the way of [20]. It will provide the retrieval results to a user which will valid or will correct them in order to constitute the ground-truth. Like this, the user will be able to edit a ground-truth in the semi-automatic way.

## 6 Acknowledgments

This work was funded by the project Madonne[7] of the French ANR program "ACI Masse de Données" 2003 and the Spanish Ministry of Education and Science under grant TIN2006-15694-C02-02. The authors wish to thank Sébastien Busson (CESR, Tours, France) of the BVH project for his collaboration to this work.

## References

1. Ramel, J., Busson, S., Demonet, M.: Agora: the interactive document image analysis tool of the bvh project. In: Document Image Analysis for Libraries (DIAL). (2006) 145–155
2. Bigun, J., Bhattacharjee, S., Michel, S.: Orientation radiograms for image retrieval: An alternative to segmentation. In: International Conference on Pattern Recognition (ICPR). Volume 3. (1996) 346–350
3. Baudrier, E., Millon, G., Nicolier, F., Seulin, R., Ruan, S.: Hausdorff distance based multiresolution maps applied to an image similarity measure. In: Optical Sensing and Artificial Vision (OSAV). (2004) 18–21

[7] http://l3iexp.univ-lr.fr/madonne/

4. Pareti, R., Vincent, N.: Global discrimination of graphics styles. In: Workshop on Graphics Recognition (GREC). (2005) 120–128
5. Uttama, S., Hammoud, M., Garrido, C., Franco, P., Ogier, J.: Ancient graphic documents characterization. In: Workshop on Graphics Recognition (GREC). (2005) 97–105
6. Gesu, V.D., Starovoitov, V.: Distance based function for image comparison. Pattern Recognition Letters (PRL) **20**(2) (1999) 207–214
7. Kumar, V.: Parallel Architectures and Algorithms for Image Understanding. Academic Press (1991)
8. Biancardi, A., Mérigot, A.: Connected component support for image analysis programs. In: International Conference on Pattern Recognition (ICPR). Volume 4. (1996) 620–624
9. van Vliet, L., Verwer, B.: A contour processing method for fast binary neighbourhood operations. Pattern Recognition Letters (PRL) **7**(1) (1998) 27–36
10. Wenyin, L., Dori, D.: From raster to vectors : Extracting visual information from line drawings. Pattern Analysis and Applications (PAA) **2**(2) (1999) 10–21
11. Pavlidis, T.: A minimum storage boundary tracing algorithm and its application to automatic inspection. Transactions on Systems, Man and Cybernetics (TSMC) **8**(1) (1978) 66–69
12. Xue, H., Govindaraju, V.: Building skeletal graphs for structural feature extraction on handwriting images. In: International Conference on Document Analysis and Recognition (ICDAR). (2001) 96–100
13. Zhong, D., H.Yan: Pattern skeletonization using run-length-wise processing for intersection distortion problem. Pattern Recognition Letters (PRL) **20** (1999) 833–846
14. Shi, Z., Govindaraju, V.: Line separation for complex document images using fuzzy runlength. In: Workshop on Document Image Analysis for Libraries (DIAL). (2004) 306–313
15. Kim, S., Lee, J., Kim, J.: A new chain-coding algorithm for binary images using run-length codes. Computer Graphics and Image Processing (CGIP) **41** (1988) 114–128
16. Chan, Y., Chang, C.: Image matching using run-length feature. Pattern Recognition Letters (PRL) **22**(5) (2001) 447–455
17. Breuel, T.: Binary morphology and related operations on run-length representations. In: International Conference on Computer Vision Theory and Applications (VISAPP). (2008)
18. Brunelli, R., Mich, O.: On the use of histograms for image retrieval. In: International Conference on Multimedia Computing and Systems (ICMC). (1999) 143–147
19. Kreher, D., Stinson, D.: Pseudocode: A LATEX Style File for Displaying Algorithms. Department of Mathematical Sciences, Michigan Technological University, Houghton, USA. (2005)
20. Yang, L., Huang, W., Tan, C.: Semi-automatic ground truth generation for chart image recognition. In: Workshop on Document Analysis Systems (DAS). Volume 3872 of Lecture Notes in Computer Science (LNCS). (2006) 324–335