# Graphics Recognition from Binary Images: One Step or Two Steps

Jiqiang Song, Min Cai, Michael R. Lyu, and Shijie Cai*
Dept. Computer Science & Engineering, the Chinese University of Hong Kong, Hong Kong, China
{jqsong, mcai, lyu}@cse.cuhk.edu.hk
* State Key Lab. of Novel Software Technology, Nanjing University, Nanjing, China
sjcai@netra.nju.edu.cn

**Abstract**

*Recognizing graphic objects from binary images is an important task in many real-life applications. Generally, there are two ways to do the graphics recognition: one-step methods and two-step methods. The former recognizes graphic objects from binary images directly, while the latter consists of vectorization and postprocessing. Neither of them is perfect enough to handle all difficulties. This paper first reviews popular graphics recognition methods to understand their advantages and disadvantages. Next, the performance comparison between two classes of methods is made in two important aspects: the time efficiency and the graphics quality, and the experimental results of time-efficiency comparison of 7 popular methods are also reported. Finally, we propose a new hybrid graphics-recognition paradigm to integrate the advantages of both one-step methods and two-step methods and minimize their disadvantages. The proposed paradigm is capable of recognizing straight lines, arcs, circles and curves efficiently, and is helpful for extracting text images in text-graphics touching cases.*

## 1. Introduction

Graphics recognition is an important task in many real-life applications. This process includes the low-level recognition and the high-level recognition. The former is to recognize graphic objects, including straight lines, arcs/circles and curves, which plays a fundamental role in the whole process. The latter involves structural analyses by domain knowledge. The graphics recognition addressed in this paper is the low-level graphics recognition.

Research on graphics recognition has been conducted for a couple of decades, and many methods have been proposed. However, as Tombre commented [1]," None of these methods works. ... Actually, the methods do work, but none of them is perfect." This paper is not to propose another new method for graphics recognition, but to discuss the argument between two classes of graphics recognition methods, one-step methods and two-step methods, and try to propose a solution by integrating the advantages of both classes. The binary image considered in this paper is the line image containing straight lines, arcs/circles, curves, and texts as well.

Existing graphics recognition methods could be classified into one-step methods and two-step methods by their paradigms. The former class recognizes graphic objects directly from a raster image, whereas the latter class first converts the raster image into a low-level vector representation, and then employs vector-based algorithms to recognize the final graphic objects. Both classes claimed that they had overcome the difficulties of the other [2,7,10,12,13]; however, they also have their own unsolved difficulties. Actually, both classes do have significant progresses in recent years. Thus, the most promising way is perhaps to well integrate their advantages.

## 2. Reviews

### 2.1 Two-step methods

The first step of two-step paradigm is raster-to-vector conversion, i.e. vectorization. The output of this step, i.e. short vectors, should represent the shape of original raster image as faithfully as possible. However, they are not necessary corresponding with ground-truth graphic objects [2]. Thus, the postprocessing is necessary to rebuild graphic objects from vectors with geometric constraints.

**2.1.1 Vectorization.** Almost all two-step methods base their vectorization on some skeletonization (or thinning) methods or some medial-axis approaches to get the skeletons or the medial axes of the image. The polygonization is performed on the skeleton points to remove redundant points, and then some line-fitting and arc-fitting algorithms are employed to convert the original image into its low-level vector format represented by short line segments and short arcs [15].

Lam *et al* [3] gave a comprehensive survey of thinning algorithms. They classified thinning methods into two categories: iterative thinning methods and non-iterative thinning methods. The former category is easy to implement but time-consuming. Moreover, its weaknesses include producing distortions at junctions, being noise sensitive, being rotation-variant and the missing of line-thickness information. The latter category includes Distance Transform (DT) based methods, contour-based methods, and graph-structure-based methods. Sequential-DT-based [4] methods need only two passes to generate the skeletons. The main weakness of DT-based methods is also the distortions at nonlinear areas, e.g. junctions, text-graphics touching or graphics-graphics touching. Contour-based methods [5] match the contour pairs to generate their medial axes so that they do not visit all pixels and need only one pass. Therefore, they are fast, rotation-invariant and do not have the spurious effects bothering the thinning methods. However, matching contours in complex cases is the primary difficulty for these methods [2]. Graph-structure-based methods [6] utilize some graph-like structures to represent line-like shapes and their topologic relations in the original image. The most popular technique to construct the graph-like structure is Run-Length Encoding (RLE). These methods work well for sparse line images containing mainly horizontal and vertical lines. Their main weakness is that the vector quality heavily depends on the scan direction of RLE.

Sparse-pixel-tracking method [7] is much faster than above-mentioned methods since it only visits a small portion of pixels in an image. But it still has two main disadvantages. First, tracking terminates if the size of intersections is larger than the predefined tracking step. Second, the constraint of length-direction consistency is not robust to keep the continuity of shapes.

**2.1.2 Postprocessing.** Postprocessing should handle all typical problems, including reconnecting broken graphics, intersection and endpoint location, and text/ graphics segmentation, to convert low-level vectors into fine graphic objects. Line-fitting methods [8] are popular in this step. Since they depend on approximating a sequence of adjacent line segments, a ground-truth line will not be recovered correctly if some parts of it are missing or have serious distortions. Some other methods [9] are based on Hough Transform (HT). They are not sensitive to partial misses or moderate distortions of a line, but the actual connectivity of contributed points should be verified. Their disadvantages are the heavy computation and large memory requirement. Recently, Liu and Dori [10] proposed a vector-based generic line-detection algorithm, which performs a stepwise extension from a starting segment with current line-shape and line-style constraints. With these constraints, redundant searching is avoided. This algorithm thus has a better performance than above two methods. However, partial misses or serious distortions of a line still affect its performance.

## 2.2 One-step methods

Kovalevsky [11] proposed a pixel-tracking algorithm for straight lines and arcs, which uses the narrowest strip to constrain the tracking direction. It can track through intersections and be non-sensitive to noise if the initial direction of strip is correct. However, it cannot assure that. Moreover, it is not aware of the line thickness.

HT is also used to detect straight lines, circles and ellipses in images directly. The advantage of HT-based methods is they can extract desired shapes from noisy environment. These methods are usually applied to small-size images, e.g. the microscopic image of blood cells.

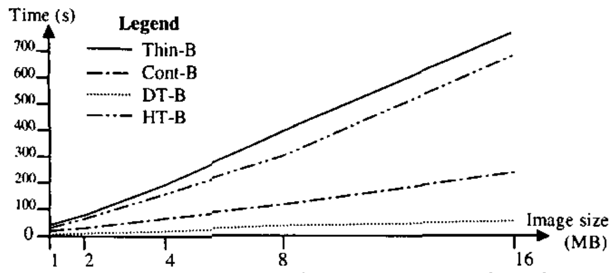
Chiang *et al* [12] proposed a region-based method to recognize straight lines from images directly, which uses the Maximum Inscribed Circle (MIC) to detect the characteristic of straight line. It recognizes straight lines with junctions directly to eliminate the distortions at junctions. But using MIC to detect the characteristic is not accurate. Moreover, iterative region test is not efficient and this method cannot handle arcs and dashed lines.

The latest one-step method is global line vectorization method [13], which recognizes straight lines (both solid and dashed), arcs and circles in their entirety from images directly. This method is very fast owing to the efficient tracking and the progressive simplification of the image. It has good line quality and can recognize line thickness. However, this method cannot recognize too short lines since the seed segment cannot be found.

## 3. Performance Analyses

### 3.1 Time efficiency

Although all two-step methods have vectorization and postprocessing steps, their computation complexities differ from each other. We have performed an experiment on the time efficiency of five two-step methods and two one-step methods. Only the first step of two-step method is implemented, which is enough to show the difference. Testing data are five scanned images of engineering drawings whose sizes vary from A4 (1MB) to A0 (16MB). The processing time of each method over different image sizes is shown in Fig. 1, which is tested on a PC with PIII500 CPU and 256MB RAM. The curves for time-consuming methods and time-efficient methods are shown in the left figure and the right figure, respectively. In following tables and figures, **Thin-B** stands for an iterative-thinning-based method, **DT-B** stands for a sequential-DT-based method, **Cont-B** stands for a contour-based method, **RLE-B** stands for a horizontal-



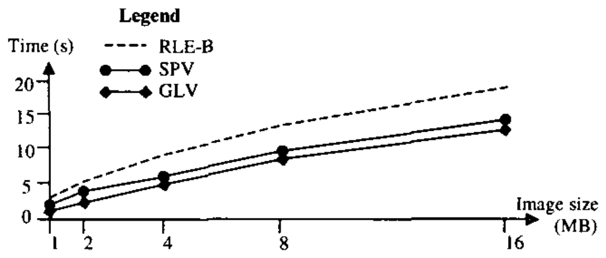


**Figure 1. Processing-time-to-image-size curves**

RLE-based method, **HT-B** stands for a straight-line-HT-based method, **SPV** stands for the sparse pixel vectorization method, and **GLV** stands for the global line vectorization method. Note that the HT is not performed on the original image, but on the skeleton image generated by Chamfer 3-4 DT. From Fig. 1, we conclude that the iterative-thinning-based method is most time-consuming, and the HT-based method is the next. The contour-based method and the sequential-DT-based method are more time-efficient due to avoiding the iteration. The RLE-based method is much faster than above methods. However, since it only performs a horizontal RLE, the lines nearly horizontal are not converted well. SPV method is the fastest one of evaluated two-step methods because it only visits a small portion of pixels. GLV method is most time-efficient for four reasons: 1) it does not visit all pixels; 2) the tracking path is generated by the time-efficient Bresenham algorithm; 3) the image is progressively simplified; 4) it dose not generate and store so many medial vectors as two-step methods do (referring to Table 1). The curves of the iterative-thinning-based method and the contour-based method are linear because they visit all pixels. The curve of the HT-based method is linear when the image size varies from 1M to 8M; however, it costs more time for the 16M image since the accumulative values of many random alignments of pixels exceed the peak threshold. The curves of the DT-based method, the RLE-based method, SPV and GLV are not linear. They are proportional to the number of lines instead.

| Method | *# 1* | *# 2* | *# 3* | *# 4* | *# 5* |
|---|---|---|---|---|---|
| **Thin-B** | 3692 | 7061 | 12459 | 25132 | 39797 |
| **DT-B** | 3942 | 7754 | 13367 | 26289 | 41332 |
| **Cont-B** | 7724 | 14816 | 25337 | 50177 | 78750 |
| **RLE-B** | 2128 | 4078 | 6936 | 13316 | 19402 |
| **SPV** | 3747 | 6944 | 11707 | 21189 | 33923 |

**Table 1. Vector numbers in the above experiment. # *N* stands for the vector number of image *N*.**

For two-step methods, the postprocessing is also a time-consuming step. Suppose the low-level vector result contains $N$ vectors, the time complexity of searching and merging vectors is $O(N^2)$, which is not linear to the image size. Table 1 shows the vector numbers of the above experiment. For the vector result of SPV on the 16M image, it takes 309 seconds to simply compute the collinearity of every two vectors. However, one-step methods do not have such trouble. Thus, from the computation-complexity point of view, one-step methods can be more efficient than two-step methods.

## 3.2 Graphics quality

Usually, the time spent on manual editing is much longer than the time on automatic graphics recognition. Knowing that, Chhabra and Philips [14] proposed a performance evaluation protocol based on the editing cost defined as the weighted sum of four types of errors.

$$w1 \cdot false\_alarms + w2 \cdot misses + w3 \cdot one2many + w4 \cdot many2one$$

They set $w1$ and $w2$ to be 1, but $w3$ and $w4$ are not indicated due to their complexity. Actually, the editing time of a one-to-many/many-to-one error is proportional to the value of 'many'. The main causes of these errors are junctions and complex cases, e.g. text-graphics touching.

Handling junctions has been a long-time difficulty for two-step methods due to the weakness of the vectorization step. At junctions, thinning generates distortions; contour following fails to get contour pairs; graph-structure analysis cannot handle well complex cases; sparse-pixel tracking cannot assure track through them correctly. The underlying reason for their weaknesses is because they only analyze a pixel area locally to generate the skeleton of this area. However, these skeletons may not conform to what the draftsman means. Two-step methods always apply geometric constraints in the second step. This is somewhat late since the vectors have been distorted in the first step. Contrarily, one-step methods try to apply geometric constraints to the pixel analysis directly. They know which direction to go and when to stop, so that they can handle junctions well. Since the final parameters are determined globally, the resulting graphics matches its raster image better. One-step methods work well for straight lines and arcs/circles; however, this way cannot be applied to curves since they do not have a fixed geometric constraint. Two-step methods are much better for curves.

## 4. Hybrid paradigm

Since neither one-step methods nor two-step methods are perfect for all kinds of graphic objects, the better way is to integrate their advantages and minimize their disadvantages as much as possible. The advantages of one-step methods are as follows.

(1) They recognize a graphic object with fixed geometric constraint in its entirety without being affected by junctions and text-graphics touching cases.
(2) They are time-efficient as long as the geometric constraints could be implemented efficiently.
(3) Progressive image simplification avoids the repetitive detection and decreases the junction complexity.

The advantages of two-step methods are as follows.

(1) They are capable of recognizing all kinds of graphic objects, especially for those without fixed geometric constraints. Searching its components in vector level is much more efficient than detecting it in pixel level.
(2) They handle well graphic objects with no junction or only a few simple junctions.

Therefore, we propose a new hybrid paradigm that can recognize all kinds of graphic objects efficiently (Fig. 2). A geometric-constrained one-step method is first applied to the image to recognize straight lines, arcs and circles. These graphic objects are recognized fast and accurately and their pixels are removed from the image after the recognition. Then the text images are extracted by some pixel-level text/graphics segmentation method. Since most text-graphics touching cases have been separated by the removal of recognized objects, the text segmentation performance on current image is much better than that on the original image. Finally, a two-step method is applied to the remained image to recognize curves. At this time, it will be more efficient for two reasons: 1) the image has been simplified largely so that only the curve image is left; 2) the original junctions among curves and recognized objects disappear so that the distortions are avoided.

## 5. Conclusions

This paper classifies low-level graphics-recognition methods into one-step methods and two-step methods, and discusses their advantages and limitations. The comparative performance analyses are performed considering both the time efficiency and the graphics quality. The experimental results of time-efficiency comparison of 7 popular methods are reported. Since neither of two classes is capable of completing the raster-to-graphics conversion both fast and accurately, we propose a hybrid paradigm to integrate the advantages of both classes, which can recognize straight lines, arcs, circles and curves more efficiently, and is also helpful for the text-image extraction.

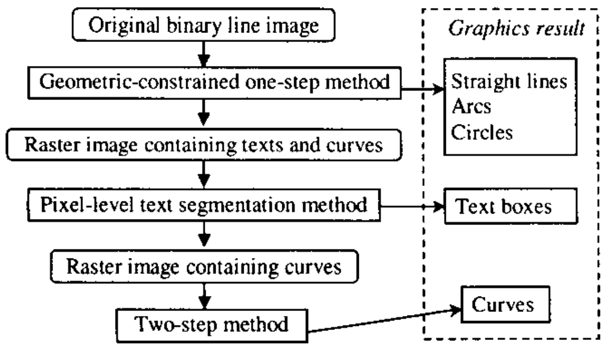

Original binary line image
Geometric-constrained one-step method
Raster image containing texts and curves
Pixel-level text segmentation method
Raster image containing curves
Two-step method
Graphics result
Straight lines
Arcs
Circles
Text boxes
Curves

**Figure 2. A hybrid graphics-recognition paradigm**

## Acknowledgement

The work described in this paper was fully supported by two grants from the Hong Kong Special Administrative Region: the Hong Kong Research Grants Council under Project No. CUHK4222/01E, and Innovation and Technology Fund, under Project No. ITS/29/00.

## References

[1] K. Tombre. "Analysis of engineering drawing: state of the art and challenges." In [16], pp257-264.

[2] K. Tombre and S. Tabbone. "Vectorization in graphics recognition: to thin or not to thin." In *Proceedings of ICPR'00*, Vol.2, pp 91-96, 2000.

[3] L. Lam, S.W. Lee, and C.-Y. Suen. "Thinning methodologies – a comprehensive survey." ***PAMI-14***, (9): 869-885, 1992.

[4] G. Borgefors. "Distance transformation in digital images." ***CVGIP*** 34: 344-371, 1986.

[5] C.-C. Han and K.-C. Fan. "Skeleton generation of engineering drawings via contour matching." *Pattern Recognition*, 27(2): 261-275, 1994.

[6] T. Pavlidis. "A vectorizer and feature extractor for document recognition." ***CVGIP***, 35:111-127, 1986.

[7] D. Dori, W. Liu. "Sparse pixel vectorization: an algorithm and its performance evaluation." ***PAMI-21***, (3): 202~215, 1999.

[8] O. Hori and S. Tanigawa. "Raster-to-vector conversion by line fitting based on contours and skeletons." In *Proceedings of ICDAR'93*, pp. 353-358, Tsukuba, Japan, 1993.

[9] T. Risse. "Hough Transform for Line Recognition: Complexity of Evidence Accumulation and Cluster Detection." ***CVGIP***, 46(3): 327-345, 1989.

[10] W. Liu and D. Dori. "A generic integrated line detection algorithm and its object-process specification." *Computer Vision and Image Understanding*, 70(3): 420-437, 1998.

[11] V.A. Kovalevsky. "New definition and fast recognition of digital straight segments and arcs." In *Proceedings of ICPR'90*, Vol.2, pp. 31-34, 1990.

[12] J. Chiang, S. Tue, and Y. Leu. "A new algorithm for line image vectorization." *Pattern Recognition*, 31: 1541-1549, 1998.

[13] J. Song, F. Su, C. Tai, J. Chen, and S. Cai, "Line Net Global Vectorization: an Algorithm and Its Performance Evaluation." In *Proceedings of CVPR 2000*, Vol.2, pp.383-388.

[14] A. K. Chhabra and I. T. Phillips. "The second international graphics recognition contest – raster to vector conversion: a report." In [16], pp 390-410.

[15] A. Chhabra, "Graphic Symbol Recognition: An Overview." In [16], pp. 68-79.

[16] K. Tombre, and A.K. Chhabra (eds). *Graphics Recognition - Algorithms and Systems*. **LNCS 1389**, Springer-Verlag, Berlin, Germany, 1998.