2010 Third International Joint Conference on Computational Science and Optimization

A New Contour Tracing Algorithm in Eight-Connected Binary Images

Ding Sun
Department of Computer Science and Technology
East China Normal University,
Shanghai, 200241, China.
E-mail: sunnysunding@gmail.com

Yan Liu
Department of Computer Center
East China Normal University,
Shanghai, 200241, China.
E-mail: yliu @cc.ecnu.edu.cn

***Abstract*—A new contour tracing algorithm in eight-connected binary images is presented in this paper. The algorithm adopts a unique contour labeling method using automaton. Connectivity information is preserved and inner and outer contours can be distinguished. Furthermore, it only labels west side of the outer contour and east side of the inner contour while traversing the image simultaneously. And it is also capable of generating all three types of chain code after tracing the contours. It is demonstrated that it increases the coding efficiency for binary image effectively and outperforms the algorithms which compute chain code from run-length code.**

Keywords-binary image; contour tracing; labeling method; automaton; chain code

I. INTRODUCTION

The encoding efficiency for binary images to represent shapes of objects is crucial in image storage and transmission. Two kinds of boundary description algorithm are mainly used in binary image. One is based on run-length code which records each consecutive 'run' [1]. And one of its applications is run-length coding (RLC) usually used in image compression. The other one is chain-code [2], which is widely adopted for representing digital curves in image analysis and processing. It possesses structured representation and high ratio of data compression. It plays an essential role in moment calculation, geometric feature detection, document image analysis and character recognition.

Images are distinguished into 4 classes in [3], and chain coding images belong to class 3 which possesses much lower information redundancy comparing to class 1 and class 2. And it has been theoretically and experimentally proved that chain code provide a compact representation and preserve all the information of the images [4, 5].

Obtaining chain code from bitmap representation of the image is prerequisite for using it to detect image features such as corners, perimeter, moment, centers and projection. Several pixel-to-vector conversion algorithms and a boundary traversing algorithm are described in [6]. The well known TP algorithm is proposed in [3], but some errors including losing inner contours, connectivity and traversing some of the boundary pixels more than one time can be found. A monochrome image compression algorithm based on chain-link encoding of object boundaries is presented in [7]. The algorithm searches for objects sequentially but it is time consuming and requires a huge stack to shrink object for finding the next one. Then, instead of using label method, a two-step algorithm to compute chain code from run-length coding representation is given in [1]. This method preserves the connectivity information between runs and never loses the inner or outer contours, although it requires a higher level of algorithmic complexity and memory usage. Computing chain code from run-length coding representation is well approved until algorithms proposed in [8], [9]. Those two demonstrate that using labeling method is also capable of keeping the connectivity information intact. In addition, an improved automaton is given in [11] to refine the algorithm in [8] using labeling method.

In applications of pattern recognition, the objects are commonly defined as eight-connected for it implies a fewer chain code, a fewer space and time consuming. Consequently, this paper proposes a novel labeling method in contour tracing using automaton specifically for eight-connected binary image. This algorithm possesses the advantages as follows

- Connectivity information is preserved
- Inner and outer contours can be obtained and distinguished in complex region
- Labeling operation and tracing contour can be executed simultaneously
- Low Algorithm complexity
- Low memory usage

This paper is organized as follows. Section 2 analyzes the two algorithms proposed in [1] and [9] which are both capable of preserving the connectivity and any other feature of the image. New contour tracing algorithm is given in Section 3. It only labels west side of the outer contours and east side of the inner contours while tracing the image simultaneously. Comparison of the performance between using labeling method and computing from run-length representation are shown in Section 4. And at last, the conclusions are drawn in Section 5.

It seems trivial to improve the encoding efficiency for binary images utilizing today's computing power. When dealing with a large number of images or in pervasive environment where the computing ability and available memory is limited, it is meaningful and advantageous for us

978-0-7695-4030-6/10 $26.00 © 2010 IEEE
DOI 10.1109/CSO.2010.183

to enhance the performance of the foundational algorithms especially.

II. RELATED WORK

In order to precisely present the shape and feature of the image, chain code should be obtained accurately. Some algorithm uses the labeling method to avoid tracing the same boundary pixels repeatedly. However, some flaws have been found in these methods. Therefore another algorithm, instead of using labeling method, computing chain code from run-length coding representation is proposed in [1].

A. *Computing Chain Code from Run-Length Coding Representations*

After the previous algorithms using labeling method fail to maintain all the connectivity information and lose some of the inner or outer contours when tracing complex image, an algorithm is proposed in [1] solves these problems.

The algorithm first detects every run at each row and records every single run's serial number and the corresponding start-pixel coordinates and end-pixel coordinates in the ABS-table. Through this method, the run-length code of the image is obtained. Then, a 3×3 pixel box has been adopted to detect the relationship between the objective pixel and its eight-connected surrounding pixels. All the runs are categorized into 5 classes. And likewise, it records each run's serial number and the corresponding class in a table called COD-table. Then, the algorithm searches the ABS-table and COD-table sequentially. Based on the coordinates and class of each run recorded on the table, the starting pixel of the contour is recognized and the contour is successfully followed. And the chain code is generated while following the contour.

This method maintains connectivity information between runs. And the extracting memory requirements of boundary tracking methods, which need to access the entire image data, can be overcame by the simultaneous chain coding of all region boundaries, but requiring a higher level of algorithmic complexity [10]. As a matter of fact, when the objective image becomes larger and more complex which means the number of runs is growing bigger, the efficiency of the algorithm declines dramatically and the memory usage increase strikingly. Actually, the realization of the algorithm becomes questionable when processing a complex region with a number of pixels larger than 10 M.

B. *Labeling All Sides of Contours with Tracer*

Computing chain code from run-length coding representation used widely until a number of algorithms with revised labeling methods are given. And the algorithm proposed in [9] distinguishes itself from the group. It proves that tracing contours using labeling method also can preserve the connectivity information without losing the inner and outer contour.

In [9], it assumes that the value of background pixel as 0 and is represented by white color and the value of object points as 1 represented by black color. And the direction of outer contour is counterclockwise and the direction of inner contour is clockwise; traversal sequence of an image is from up to down and left to right. The background point neighboring and out of the outer contour is called OB. The background neighboring and enclosed by the contour is called IB. The method labels the object point on the traced contour to 3 and labels OB or IB to 2. During traversal, if the change of value of two consecutive points is from 2 to 1, a new outer contour starting pixel is found; otherwise, if the change is from 1 to 0 or 3 to 0, a new inner contour starting pixel is found. And it is proved that the labeling operation can be done simultaneously while tracing the contour.

However, this algorithm labels all the pixels of outer and inner contours requiring a higher level of time complexity and memory usage. And the output of the tracing processing is only one kind of the chain code, the Freeman chain code. Furthermore, it does not compare the performances between the algorithm using labeling method and the algorithm extracting chain code from run-length code.

III. NEW CONTOUR TRACING ALGORITHM USING AUTOMATON

The research content of the algorithm can be generally divided into three parts. The first part demonstrates how the automaton follows the contour and helps generate the three types of chain code. The second part analyzes the uniqueness of the labeling operation. And how to distinguish the outer and inner contour of the labeled boundary are mentioned in the third part.

This algorithm assumes the value of pixel, the direction of outer/inner contours and the traversal sequence are the same as [9].

A. *Contour Tracing Automaton*

In Fig. 1, the pixels marked with B is the inside boundary and the pixels marked with O is the outside boundary. This algorithm adopts the automaton which traverses on the outside boundary of the eight-connected image. It assumes that the direction of the automaton is counterclockwise. The arrow demonstrates the moving direction of the automaton. And there are four kinds of position relations between the automaton and the object pixel which are shown in Fig. 2. So, we can acquire the set of internal state which is $Q = \{State\_a, State\_B, State\_C, State\_D\}$. The input of the automaton, from the scale 0 to 1, represents whether the automaton's current position is one of the object pixels.

	0	0	0	0	0	0	
0	0	B	B	B	B	0	0
0	B					B	0
0	B					B	0
0	B					B	0
0	B					B	0
0	B					B	0
0	B					B	0
0	B					B	0
0	0	B	B	B	B	0	0
	0	0	0	0	0	0	

Figure 1. Inside and outside boundary of an binary image

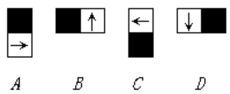

Figure 2. Four kinds of position relations in eight-connected image

Labeling process should be capable of speculating the next internal state of the automaton based on the current state and then converting itself to the corresponding state in order to make sure the automaton follow the contour accurately. The state transition graph of tracing automaton [12] is shown in Fig. 3[8] illustrating State_A. Automation detects whether the pixel marked with '?' is an object pixel. The result becomes the input of the automaton, and as the automaton moves, six possible final state of the automaton is demonstrated. $S_n (n = 1,2,..., 6)$ Each serial number represents the corresponding final state and the coordinates which is the output of the automaton shows how far the automaton moves.

Because of the space symmetry of the internal state of the automaton, the state transition graph of State_B, C and D can be obtained by rotate the state transition graph of State_A by 90°, 180° and 270°. Because of the finite and closeness of the operation, the tracing automaton is a deterministic finite automaton (DFA). [11] manages to narrow down the number of 6 possible final states to 3. The revised state transition graph of State_A is shown in Fig. 4. And likewise, the state transition graph of State_B, C and D can be acquired by rotating State_A's.

Instead of generating Freeman chain code exclusively in [9], new algorithm is capable of generating all three types of chain code after tracing the contour: Freeman chain code, crack code [13], and vertex code can be generated through the output of the automaton.

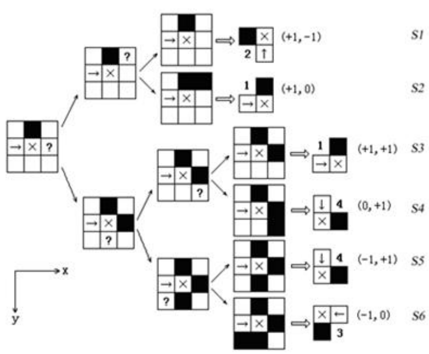

Figure 3. The state transition graph of 8-connected tracing automaton

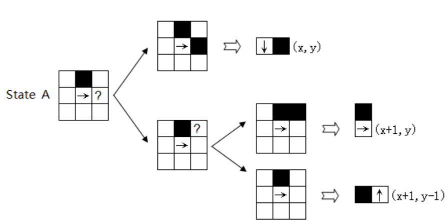

Figure 4. The revised state transition graph of State_A

B. Labeling operation

At the sake of saving run time and reduce the memory usage, the labeling operation proposed in this paper only labels one side of the contour. Since we assume that the traversal sequence of an image is from up to down and left to right and the direction of automaton is counterclockwise, we only label the west side of the outer contour and east side of the inner contour instead of labeling all sides of the contours. A boundary pixel is recognized only when the black pixel's left neighbor is a white pixel. In other words, in the assumed traversal method, the west side of the outer contour and the east side of the inner contour will first be traced more than once without labeling. As a matter of fact, it has been experimentally proved that all the information will be reserved if we just label the west side of the outer contour and eastside of the inner contour. And the labeling operation can be executed simultaneously while automaton tracing the contours.

For the reason that the new algorithm only labels west side of the outer contour and east side of the inner contour, the labeling time is reduced remarkably comparing to the algorithm proposed in [9]. And if the image is read only, which means the labeling processing requires allocating new memory, the memory usage of the algorithm we proposed is comparatively low.

C. Method to distinguishing the outer and inner contour

In traditional algorithms which adopt labeling method, flaws like losing connectivity information or inner/outer contours are mostly made by inappropriate labeling operation. In this algorithm, we distinguish each contour by detecting the features of its lowest pixel which has a white neighbor. That is to say, among all the pixels on the bottom line of each contour, the one whose left neighbor is white determines whether it is an outer contour or an inner contour. Assuming the pixel's coordinate is (x, y), we detect the pixel with the coordinate $(x-1, y-1)$: if it is a white pixel, then it is an outer contour; if it is a black pixel, then it is an inner contour. As shown in Fig. 5. The contour includes pixel A is an outer contour, the contour includes B is an inner contour. Fig. 6 shows the original objective image; Fig. 7 shows the algorithm labels the west side of the outside contour and east side of the outer contour with red pixel; Fig. 8 shows the contours of the original image where connectivity information has been preserved.

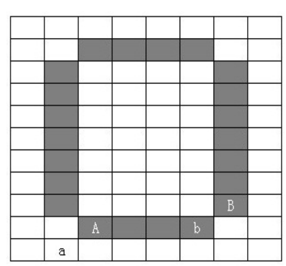

Figure 5. The detected pixel which determines the contour an outer contour or an inner contour

IV. EXPERIMENTS

Our new contour tracing algorithm using automaton (AAM) is compared with the algorithm exacting chain code from run-length code (ARL). We use six types of test images: halftone pictures, headlines, legacy documents, newspaper, photographs and textual contents. This experiment environment is an Intel Pentium III 1.02 GHz personal computer with 512 MB SDRAM. Runtime is the average of 100 times operation. Each document type adopts the same image with 4 different sizes: 0.04 M pixels, 0.16 M pixels, 0.64 M pixels, and 2.56 M pixels.

The comparison results are listed in Table 1. The performances of the two algorithms are shown in Figs. 9-14. (AAM stands for algorithm using automaton; ARL stands for algorithm compute chain code from run-length code) In these figures, the size of the test image (M pixels) is plotted along the horizontal axis, while the average processing time (ms) of each method is plotted along the vertical axis.

TABLE I. PERFORMANCES OF THE TWO METHODS BEING COMPARED

Document type	Image size (M pixels)	Algorithms AAM	ARL
		Average processing time (ms)	
Halftone pictures	0.04	15.7	20.4
	0.16	31.2	57.8
	0.64	93	125
	2.56	234.6	353.2
Headlines	0.04	4.5	4.7
	0.16	15.5	18.7
	0.64	46.4	51.6
	2.56	172	174.1
Legacy documents	0.04	2.8	3.1
	0.16	13.3	15.7
	0.64	46	50
	2.56	156.1	212.5
Newspaper	0.04	4.4	6.2
	0.16	14.7	28.1
	0.64	62	82.8
	2.56	203	359.3
Photographs	0.04	3.1	3.1
	0.16	14	15.6
	0.64	31.3	48.5
	2.56	141.1	201.3
Textual content	0.04	6.7	7.8
	0.16	30.8	35.9
	0.64	78.2	106.3
	2.56	250.8	546.9

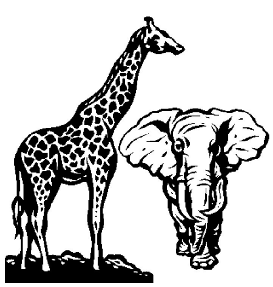

Figure 6. The original image

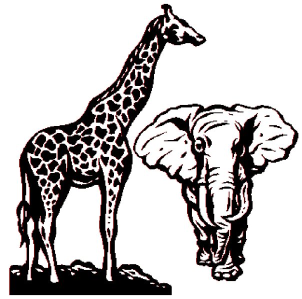

Figure 7. Original objective image after labeling process

Figure 8. Contours of the original image

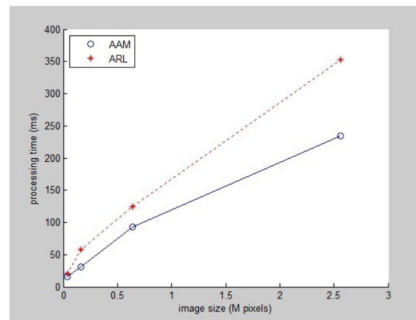

Figure 9. Performance of the two algorithms for halftone pictures

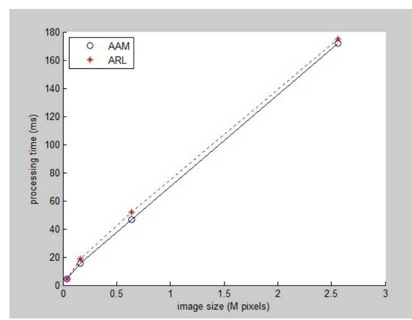

Figure 10. Performance of the two algorithms for headlines

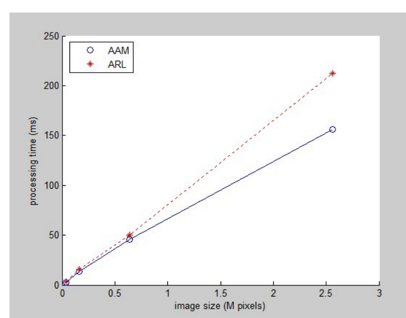

Figure 11. Performance of the two algorithms for legacy documents

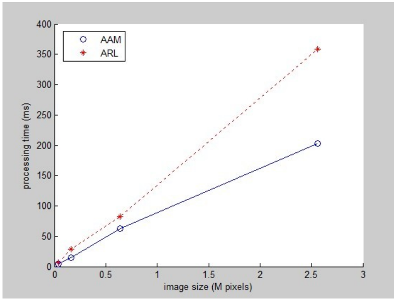

Figure 12. Performance of the two algorithms for newspaper

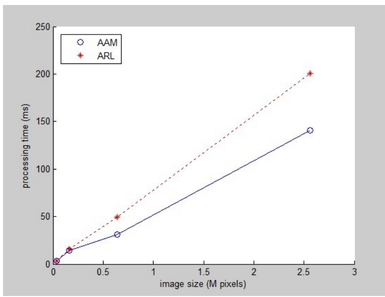

Figure 13. Performance of the two algorithms for photographs

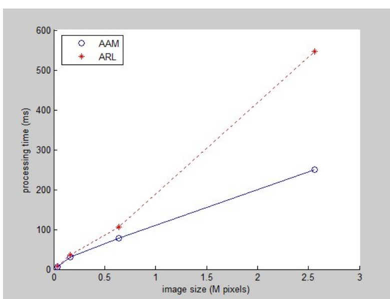

Figure 14. Performance of the two algorithms for textual content

V. CONCLUSION

This paper presents a new contour tracing algorithm in eight-connected binary images. We make full utilization of the automaton in the labeling method to avoid tracing contours more than once. The connectivity information is fully preserved and the inner and outer contours can be distinguished. The new algorithm only labels west side of the outer contour and east side of the inner contour and it is capable of generating all three types of chain code after tracing the contours. It is theoretically proved that our algorithm improves the current binary image encoding efficiency regarding algorithmic complexity and memory usage. In experiments on six types of images of various sizes, we compare our method with the algorithm computing chain code from run-length code. The results show that our algorithm outperforms that in all six types of binary image in terms of computational speed especially when the size of the image grows bigger and more complex.

ACKNOWLEDGMENT

Thanks to my tutor's instruction, support and patience. Thanks and profound appreciation to all my senior fellows especially Yan Liu for tutoring and helping me improve the experiment. And thanks to all my friends for helping.

REFERENCES

[1] Kim S D, Lee J H, Kim J K, A new chain-coding algorithm for binary images using run-length codes [J]. CVGIP, 1988, 41: 114-128.

[2] H. Freeman, Computer processing of line drawing images, Computational Surveys 6 (1974) 57-97.

[3] Pavlidis T, Algorithms for Graphics and Image Processing [M]. Rockville: Computer Science Press, 1982.

[4] Liu Y K, Wei W, Wang P J. Compressed vertex chain codes [J]. Pattern Recognition, 2007, 40: 2908–2913.

[5] Hermilo S C, Bribiesca E, Ramon M R D. Efficiency of chain codes to represent binary objects [J]. Pattern Recognition, 2007, 40: 1660-1674.

[6] A. Rosenfeld and A.C. Kak, Digital Image Processing, New York: Academic Press, cap 11, 1982.

[7] T.H. Morrin, Chain-Link Compression of Arbitrary Black-White Images, Computer Graphics and Image Processing, no. 5, pp. 172-189, 1976.

[8] Wang L, "The study on obtaining chain code and document layout analysis [D]", Department of Computer Science and Technology, East China Normal University, 2007.

[9] Mingwu Ren, Jingyu Yang, and Han Sun, Tracing boundary contours in a binary image, Image and Vision Computing 20 (2002) 125-131.

[10] Primo Zingaretti, Massimiliano Gasparroni, and Lorenzo Vecci, Fast Chain Coding of Region Boundaries, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 4, April 1998.

[11] Yan Liu, Weinan Gao, Zhiqing Wang, and Zhiyong Ju, An Improved Boundary Tracing Automaton in 8-Neigthborhood Digital Image, Journal of East China Normal University (Nature Science).

[12] Guoqing Gu, Yanbing Xu, A Method of Region Labeling for Digital Image [J]. Journal of University of Shanghai for Science and Technology, 2001, 23(4) 295-299.

[13] Bribiesca E, A new chain code [J]. Pattern Recognition, 1999, 32: 235-251.