ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.11, No.3, 1–6

Efficient algorithm for obtaining connected components in bi-level images

Yan Liu[a)] **and Guoqing Gu**

Department of Computer Science and Technology,
East China Normal University, Shanghai, China

a) *anne.liuyan@gmail.com*

Abstract: We propose an efficient method for labeling connected components in bi-level images. The proposed algorithm uses a Deterministic Finite Automaton to obtain chain codes and label component boundaries, and use a tracing technique that is simpler than existing methods. Experiments on various types of images show that the proposed method improves contour tracing efficiency.

Keywords: connected component, boundary tracing, chain code, automaton, boundary processing

Classification: Fiber optics, Microwave photonics, Optical interconnection, Photonic signal processing, Photonic integration and systems

References

[1] A. Rosenfeld: Computer Graphics **12** (1978) 135.
[2] I. Chakravarty: CGIP **15** (1981) 182.
[3] T. Pavlidis: *Algorithms for Graphics and Image Processing* (Computer Science Presse, Rockville, 1982).
[4] M. W. Ren, J. Y. Yang and S. Han: Image and Vision Computing **20** (2002) 125.
[5] F. Chang, C. J. Chen and C. J. Lu: CVIU **93** (2004) 206.
[6] S. D. Kim, J. H. Lee and J. K. Kim: CVGIP **41** (1988) 114.
[7] H. Freeman: ACM Comput. Surveys **6** (1974) 57.
[8] E. Bribiesca: Pattern Recognition **32** (1999) 235.
[9] Y. K. Liu, W. Wei and P. J. Wang: Pattern Recognition **40** (2007) 2908.
[10] S. C. Hermilo, E. Bribiesca and M. R. D. Ramon: Pattern Recognition **40** (2007) 1660.

1 Introduction

Connected component labeling (CCL) of a bi-level image is a fundamental operation in the segmentation process. CCL has applications in the fields of image understanding, character recognition, and geometric modeling. With the development of real-time systems, improving the CCL performance to meet the demand of applications on such large amount of images is desirable.

DOI: 10.1587/elex.11.20130748
Received September 24, 2013
Accepted January 06, 2014
Publicized January 22, 2014
Copyedited February 10, 2014

Rosenfeld proposed a number of algorithms for pixel-to-vector conversion [1]. However, these methods do not consider the case of intersecting lines [2]. Pavlidis proposed the well-known TP algorithm [3], which exhibited errors, such as loss of inner contours, connectivity, and traversing some boundary pixels more than once [4].

Several algorithms based on label equivalences have been proposed. Chang et al. [5] proposed an algorithm that outperforms other methods that use label equivalences and ordinary computers. In this paper, we select Chang's method to represent existing methods that are based on label equivalences.

Freeman et al. developed the chain code to separately encode each connected component [7, 8]. According to the classification of Pavlidis [3], the chain code belongs to class 3, has less data redundancy, and has a higher ratio of lossless compression [9, 10].

Kim et al. [6] proposed a two-step algorithm. This method computes chain codes from run-length coding representation, preserves the connectivity information between runs, and never loses inner or outer contours.

In this paper, we propose an efficient algorithm based on Deterministic Finite Automaton (DFA). Automaton traces the connected components in a bi-level image and obtains the chain code of the contours at the same time. Thus the proposed algorithm reduces computational time and its labeling technique is simpler than existing techniques.

2 Proposed method

The proposed algorithm consists of four parts, namely, automaton definition, starting procedure, tracing technique, and termination conditions.

2.1 Automaton definition

After analyzing all types of image contours, we classify contour situation into four states, as shown in Fig. 1. Each state can transit to a unique state under a triggering condition that meets the terms of DFA. The proposed algorithm considers the four pixel states shown in Fig. 1 as DFA internal states. The pixels labeled by a and b are taken as input of DFA.

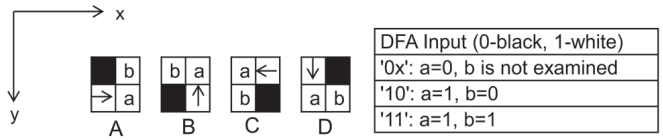

Fig. 1. States of automaton

The proposed method uses a set of 2×2 templates as shown in Fig. 1. The solid arrow represents a white pixel on the outside boundary located at coordinates (i, j). Positions with label a and label b are pixels used to determine the next state of the automaton. Pixel a should be examined first. Pixel b can be examined only when pixel a is white.

DOI: 10.1587/elex.11.20130748
Received September 24, 2013
Accepted January 06, 2014
Publicized January 22, 2014
Copyedited February 10, 2014

The tracer moves to the next state based on the current state and inputs. The transition chart of *stateA* is shown in Fig. 2. The outputs include the next state along with the next coordinates. Given the space symmetry of the internal states of the automaton, the state transition charts of *stateB*, *C*, and *D* can be obtained by rotating the state transition chart of *stateA* by $-90°$, $-180°$, and $-270°$, respectively. We obtain the DFA state diagram based on the state transitions among *stateA*, *B*, *C*, and *D*, as illustrated in Fig. 3.

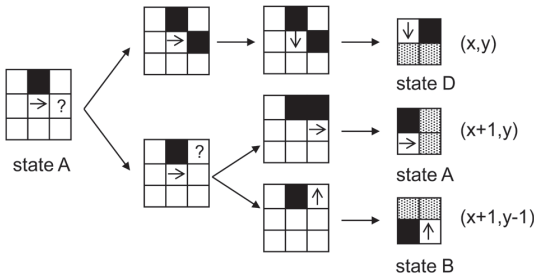

Fig. 2. Automaton state transition of *stateA*

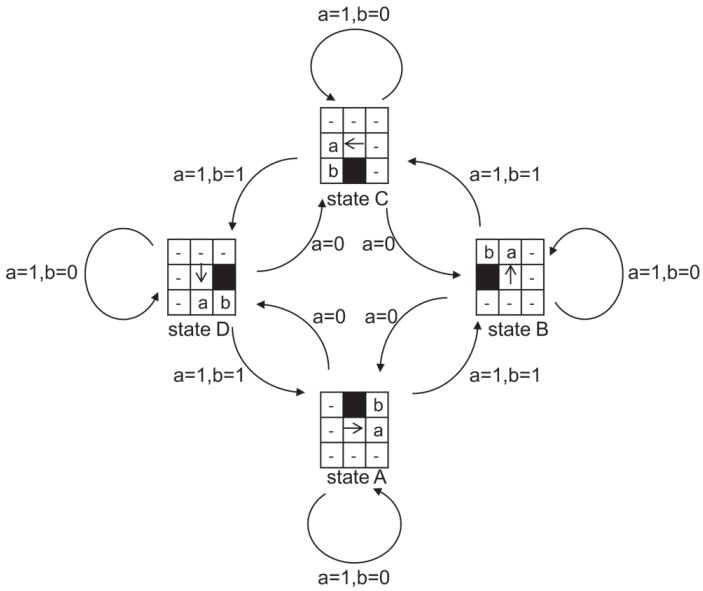

Fig. 3. DFA state diagram

As shown in Table I, the offsets of coordinates are outputs of the DFA. The tracer moves automatically according to the output position. Here, S_i and S_{i+1} represent the current state and the next state, respectively.

Table I. Output of DFA

S_i \ S_{i+1}	*stateA*	*stateB*	*stateC*	*stateD*
stateA	Δx=1,Δy=0	Δx=1,Δy=-1	-	Δx=0, Δy=0
stateB	Δx=0,Δy=0	Δx=0,Δy=-1	Δx=-1,Δy=-1	-
stateC	-	Δx=0,Δy=0	Δx=-1,Δy=0	Δx=-1,Δy=1
stateD	Δx=1,Δy=1	-	Δx=0, Δy=0	Δx=0, Δy=1

DOI: 10.1587/elex.11.20130748
Received September 24, 2013
Accepted January 06, 2014
Publicized January 22, 2014
Copyedited February 10, 2014

2.2 How to find a new contour

The proposed algorithm scans the image in raster fashion and traces each intermediate contour. If we regard a contour as a closed circle even if it is a single point, any point of the contour can be regarded as a starting point with the corresponding state. However, if we decide to start from *stateA*, we should search for unvisited pixels that match the condition of *stateA*.

2.3 How to follow a contour

A key aspect of the proposed method is that the tracer follows the contour and generates the chain codes automatically. Part of the function of the tracer is to transit to the next state based on the current state and the inputs, and to move forward to the next coordinates according to the DFA outputs.

The tracing algorithm, where *iState* is the current DFA state, $dx1$, $dy1$, $dx2$, $dy2$ are offsets of the coordinates in the x and y directions, respectively, is as follows.

```
While(bReturn!=True)
{  lblPt=GetPos_a(x,y,iState);    //value of pixel a
   if(lblPt==0)
      iState=(iState+3)modulo 4;
   else
   {  lblPt=GetPos_b(x,y,iState);//value of pixel b
      if(lblPt==0)
      {  x+=dx1[iState];
         y+=dy1[iState];
         bReturn=ReachStartPoint();
      }
      else
      {  x+=dx2[iState];
         y+=dy2[iState];
         iState=(iState+1)modulo 4;
         bReturn=ReachStartPoint();
}}}
```

2.4 When a contour tracer should terminate

The proposed algorithm adopts two conditions for terminating a trace: that is, the tracer has to return to the starting point, and the current state has to be the same as the initial state. Hence, when the tracer reaches the starting point with the initial state, the tracer of that contour should terminate.

```
Bool ReachStartPoint()
{ If ( x=initX && y=initY && iState=initState )
     Return True;
}
```


DOI: 10.1587/elex.11.20130748
Received September 24, 2013
Accepted January 06, 2014
Publicized January 22, 2014
Copyedited February 10, 2014

3 Laboratory result

In the proposed technique, the tracer is capable of generating all existing types of chain codes while tracing a contour. In Table II, Freeman eight-direction code is used as a example. *state_C* and *state_N* are the current state and the next state respectively.

Table II. Output of Freeman 8-direction code

state_N \ *state_C*	*stateA*	*stateB*	*stateC*	*stateD*
stateA	0	1	-	-
stateB	-	2	3	-
stateC	-	-	4	5
stateD	7	-	-	6

A comparison of the performances is shown in Table III. The methods proposed by Kim and Chang require scanning every pixel in the region. Thus, the compared times are $O(n) + area$. The space Kim uses to store the connectivity information, the coordinates, and the marks are denoted by $5 \times n\_Run + 3 \times \mathrm{H}$. Kim also required an extra $3 \times \mathrm{W} \times \mathrm{H}$ space to store the runs, the labels, and the chain strings. Chang spends an extra $1 \times \mathrm{W} \times \mathrm{H}$ to save different labels.

Table III. Theoretical algorithm performance

	Kim's	Chang's	DFA method
Compared times	O(n)+area	O(n)+area	O(n)
Space used	$4 \times \mathrm{W} \times \mathrm{H} + 5 \times n\_Run + 3 \times \mathrm{H}$	$2 \times \mathrm{W} \times \mathrm{H}$	$\mathrm{W} \times \mathrm{H}$

The tracer traverses the image to trace components, so no component is lost. The automaton can obtain all contour points connected to each new contour point, so the component is integrated and has no defects. To test the correctness of the proposed method, we experimented with 300 bi-level images selected from the computer vision test images of CMU (http://www.cs.cmu.edu/~cil/v-images.html). The test set used consist of various types of images. The proposed method labels all components in the test images, with no contour loss. Moreover, the proposed method outperforms the other algorithms in all types of images in terms of computational speed.

The second experiment is conducted on the six types of test images as in Chang's study [5]. Each document has three sets of images: original 2.56 M pixel images, images decreased to 0.64 M pixels, and images decreased to 0.16 M pixels.

Kim's method uses a run-length code which decreases speed as a result of the significant space wasted in storing extra information. By contrast, Chang's algorithm traces components fast. However, it has to label the whole black region of the component to mark its boundary. This labeling

DOI: 10.1587/elex.11.20130748
Received September 24, 2013
Accepted January 06, 2014
Publicized January 22, 2014
Copyedited February 10, 2014

Table IV. Performance of compared methods

Document type	Image size	Average processing time (ms)		
		Chang's	Kim's	DFA method
Legacy documents	0.16	10.7	9.59	7.3
	0.64	33.1	27.42	21.3
	2.56	106.48	89.26	69.7
Headlines	0.16	10.77	9.5	6.8
	0.64	30.63	26.20	19.3
	2.56	100.14	90.37	66.2
Textual content	0.16	17.16	15.22	11
	0.64	39.25	34.87	25
	2.56	117.04	106.33	76.5
Halftone pictures	0.16	9.6	8.23	6.56
	0.64	30.15	25.4	19.84
	2.56	129.5	116.17	84.4
Newspaper	0.16	11.86	10.14	7.8
	0.64	36.74	30.89	23.4
	2.56	114.1	93.37	71.8
Photographs	0.16	9.5	8.13	6.2
	0.64	30.26	23.27	18.3
	2.56	112.13	95.65	69.8

technique results less efficiency in Chang's method. The proposed algorithm outperforms all these methods in terms of computational speed.

4 Conclusion

This paper proposes an efficient method for labeling connected components in bi-level images. The proposed algorithm uses automaton in labeling to avoid tracing contours more than once. Thus, the connectivity information is fully preserved. The experimental results show that the proposed method outperforms these algorithms in all types of images in terms of computational speed. The proposed algorithm improves bi-level contour tracing performance.

Acknowledgements

We would like to thank L. Z. Han and an anonymous reviewer that greatly improved the presentation of the paper. We also gratefully acknowledge the sponsor of National Natural Science Foundation of China (10635040).

DOI: 10.1587/elex.11.20130748
Received September 24, 2013
Accepted January 06, 2014
Publicized January 22, 2014
Copyedited February 10, 2014