LETTER

A New Connected-Component Labeling Algorithm*

Xiao ZHAO[†], ***Nonmember***, **Lifeng HE**[†,††a)], ***Member***, **Bin YAO**[†], ***and*** **Yuyan CHAO**[†††], ***Nonmembers***

SUMMARY This paper presents a new connected component labeling algorithm. The proposed algorithm scans image lines every three lines and processes pixels three by three. When processing the current three pixels, we also utilize the information obtained before to reduce the repeated work for checking pixels in the mask. Experimental results demonstrated that our method is more efficient than the fastest conventional labeling algorithm.

key words: *connected component, labeling, pattern recognition, image analysis*

1. Introduction

Connected-component labeling in a binary image assigns all pixels belonging to each connected component (each object) with a unique label. By connected-component labeling, we can distinguish the difference objects in the image and then further extract the features of those objects. Therefore, connected-component labeling is indispensible for pattern recognition, pattern analysis, computer (robot) vision, and machine intelligence [1].

Many algorithms have been proposed for addressing this problem [2]–[8]. The fast connected-component labeling algorithm, *FCL algorithm* for short, proposed in [4] is a famous one. This algorithm was improved in [6] by utilizing the information obtained before for processing the current pixels. For convenience, we call this algorithm *the IFCL algorithm*.

The IFCL algorithm was further improved in [8], which is a configuration-transition-based one. This algorithm scans alternate image lines, processes pixels two by two, and also utilizes the information obtained before for processing the current pixels. This algorithm is the most efficient labeling algorithm up to now. For convenience, we call this algorithm *the CTB algorithm*.

This paper presents an improvement of the CTB algorithm. The proposed algorithm scans image lines every three lines, processes pixels three by three, and also utilizes the information obtained before for processing the current pixels. In this way, the number of times needed to check the neighbor pixels for processing a pixel can be further reduced; thus, the efficiency of labeling can be improved. Experimental results demonstrated that the proposed algorithm is more efficient than the CTB algorithm.

2. Reviews of the *CTB* Algorithm

The CTB algorithm proposed in [8] is a label-equivalence-based two-scan algorithm. In the first scan, it assigns each object pixel a provisional label. At any time, all provisional labels assigned to each object are recorded as *equivalent labels* by use of an *equivalent label set*. For each equivalent label set, the smallest label in the set is denoted as the *representative label*. For convenience, the representative label of label s is represented by $R[s]$; an equivalent label set with the representative label w is denoted as $S(w)$ (thus, for any $t \in S(w)$, $R[t] = w$). Whenever two provisional labels u and v, where $u \in S(m)$ and $v \in S(n)$ respectively, are found to be equivalent (i.e., belong to the same connected component) in the processing, we know that all labels in $S(m)$ and $S(n)$ should be equivalent (belong to the same connected component), thus, the two equivalent label sets should be combined, i.e., $S(r) = S(m) \cup S(n)$, where r is the smaller one of m and n.

After the first scan, all equivalent labels assigned to an object will be combined in the same equivalent label set with a unique representative label. Then, in the second scan, by replacing each label with its representative label, all pixels of each object will be assigned a unique label.

As introduced above, the CTB algorithm scans image lines alternate lines, and processes pixels two by two. The mask used in the CTB algorithm is shown in Fig. 1. For pro-

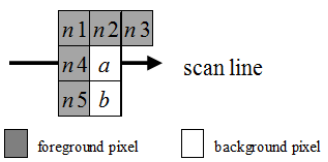

Fig. 1 Mask used in the CTB algorithm.

Manuscript received June 15, 2015.
Manuscript revised July 24, 2015.
Manuscript publicized August 5, 2015.

[†]The authors are with Artificial Intelligence Institute, College of Electrical and Information Engineering, Shaanxi University of Science and Technology, Xi'an, Shaanxi 710021, China.

[††]The author is with Graduate School of Information Science and Technology, Aichi Prefectural University, Nagakute-shi, 480–1198 Japan.

[†††]The author is with Graduate School of Environment Management, Nagoya Sangyo University, Owariasahi-shi, 488–8711 Japan.

*This work was supported in part by the Grant-in-Aid for the National Natural Science Foundation of China under Grant No. 61471227, the Scientific Research (C) of the Ministry of Education, Science, Sports and Culture of Japan under Grant No. 26330200, and the Scientific Research of Shaanxi Province of China under Grant No. 2014K11-02-01-13.

a) E-mail: helifeng@ist.aichi-pu.ac.jp (Corresponding author)

DOI: 10.1587/transinf.2015EDL8135

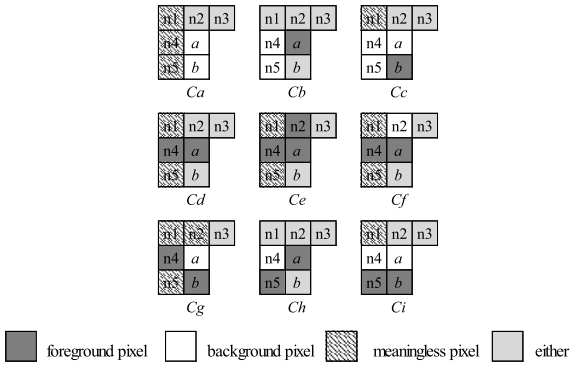

Fig. 2 Configurations of the mask and the two current pixels a and b.

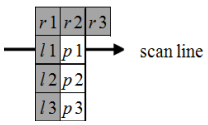

Fig. 3 Mask used in our algorithm.

cessing the current pixels a and b, the CTB algorithm considers the nine configurations shown in Fig. 2, where meaningless pixels means that they will not be used for processing the current pixels.

For example, for the configuration Cd, where a is an object pixel, b is either, $n4$ is an object pixel, $n2$ and $n3$ are unknown pixels, and $n1$ and $n5$ are meaningless whether they are, it is processed as follows.

(1) The current pixel a is assigned the label of pixel $n4$, and if pixel b is a foreground pixel, it can also be assigned the same label;

(2) If $n2$ is an object pixel, nothing else needs to be done, the configuration will transmit to one of Ca, Cd, and Cg, according to the next two pixels to be processed;

(3) Else if $n3$ is an object pixel, then $n4$ and $n3$ are connected by a, it resolves the label equivalence between the label of $n4$ and that of $n3$, and the configuration will transmit to one of Ca, Ce, and Cg, according to the next two pixels to be processed;

(4) Else, nothing needs to be done, the configuration will transmit to one of Ca, Cf, and Cg, according to the next two pixels to be processed;

Other configurations can be analyzed in a similar way. Obviously, when a pixel in the mask is checked, the pixel will not be checked again when processing the next two pixels.

3. Our Proposed Algorithm

We extend the CTB algorithm by processing image lines every three lines. The mask used in our proposed algorithm is shown in Fig. 3.

Because any pixel is either a foreground pixel or a background pixel, there are eight cases for the three current pixels $p1$, $p2$ and $p3$. Moreover, in the case where $p2$ is a foreground pixel, similar in the CTB algorithm, if $p3$ is a

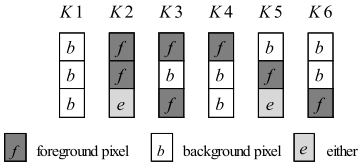

Fig. 4 Six cases needed to be considered for the three current pixels $p1$, $p2$ and $p3$.

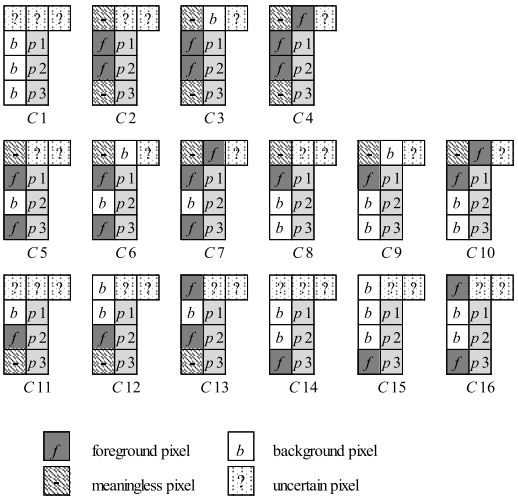

Fig. 5 Configurations of the mask and the three current pixels $p1$, $p2$ and $p3$.

foreground pixel, we only need to assign the label of $p2$ to it, and otherwise nothing needs to be done. Therefore, we only need to consider six cases, as shown in Fig. 4.

On the other hand, because the three current pixels $p1$, $p2$ and $p3$ will become the three pixels $l1$, $l2$, and $l3$ when processing the next three pixels, the states of $l1$, $l2$, and $l3$ must be one of states shown in Fig. 4. Moreover, (1) when processing $l1$, $l2$, and $l3$, $r3$ did not occur in the corresponding mask, therefore, it is an unknown pixel when processing $p1$, $p2$, and $p3$; (2) if $l1$ is a foreground pixel, then what $r1$ is does not influence the labeling result, thus, $r1$ is a meaningless pixel; (3) if $p1$ is a background pixel, then we will not go to check $r3$, similarly, if $l1$ is a background pixel, when processing $p1$, $p2$, and $p3$, $r2$ will be an unknown pixel; (4) for other cases, each of $r1$ and $r2$ could be an unknown pixel, a foreground pixel or a background pixel.

According to the above analysis, we can conclude the 16 configurations for the mask, as shown in Fig. 5 For convenience, hereafter we use a shorthand notation, $\{r1, r2, r3, l1, l2, l3\}$, to denote a configuration of the mask. Moreover, we use b to denote a background pixel, f for a foreground pixel, – for a meaningless pixel, and ? for an uncertain pixel. Thus, for example, the configuration $C2$ in Fig. 5 can be represented by $\{–, ?, ?, f, f, –\}$.

The processing for the configuration $C2$, for example, can be made according to the combination of the three current pixels as follows:

(1) For the case $K1$, nothing needs to be done. The con-

figuration will transit to configuration $\{?, ?, ?, b, b, b\}$, i.e., $C1$;

(2) For the case $K2$, where $p1$ and $p2$ are connected to $l1$ and $l2$, we assign the label assigned to $l1$ or $l2$ to $p1$ and $p2$, and if $p3$ is an object pixel, we assign the same label to it. Moreover, if $r2$ is an object pixel, nothing else needs to be done. The configuration will transit to either $\{f, ?, ?, f, f, f\}$ or $\{f, ?, ?, f, f, b\}$, both are an instance of $\{-, ?, ?, f, f, -\}$, i.e., $C2$; otherwise, if $r2$ is a background pixel and $r3$ is an object pixel, then the object pixel $l1$ is connected to $r3$ by $p1$, thus, the label assigned to $l1$ and that assigned to $r3$ are equivalent labels. We resolve the label equivalence between the two labels. The configuration will transmit to either $\{b, f, ?, f, f, f\}$ or $\{b, f, ?, f, f, b\}$, both are an instance of $\{-, f, ?, f, f, -\}$, i.e., $C4$; lastly, if both $r2$ and $r3$ are background pixels, nothing else needs to be done. The configuration will transmit to either $\{b, b, ?, f, f, f\}$ or $\{b, b, ?, f, f, b\}$, both are an instance of $\{-, b, ?, f, f, -\}$, i.e., $C3$.

(3) For the case $K3$, where $p1$ and $p3$ are connected by $l2$, we assign the label assigned to $l2$ to $p1$ and $p3$. If $r2$ is an object pixel, nothing else needs to be done. The configuration will transit to $\{f, ?, ?, f, b, f\}$, an instance of $\{-, ?, ?, f, b, f\}$, i.e., $C5$; else if $r2$ is a background pixel and $r3$ is an object pixel, then the object pixel $l1$ is connected to $r3$ by $p1$, thus, the label assigned to $l1$ and that assigned to $r3$ are equivalent labels. We resolve the label equivalence between the two labels. Then, the configuration will transmit to $\{b, f, ?, f, b, f\}$, an instance of $\{-, f, ?, f, b, f\}$, i.e., $C7$; lastly, if both $r2$ and $r3$ are background pixels, nothing else needs to be done, and the configuration will transit to $\{b, b, ?, f, b, f\}$, an instance of $\{-, b, ?, f, b, f\}$, i.e., $C6$.

(4) For the case $K4$, we assign the label assigned to $l1$ to $p1$. If $r2$ is an object pixel, nothing else needs to be done. The configuration will transit to $\{f, ?, ?, f, b, b\}$, an instance of $\{-, ?, ?, f, b, b\}$, i.e., $C8$; else if $r2$ is a background pixel and $r3$ is an object pixel, then the object pixel $l1$ is connected to $r3$ by $p1$, thus, the label assigned to $l1$ and that assigned to $r3$ are equivalent labels. We resolve the label equivalence between the two labels. Then, the configuration will transmit to the $\{b, f, ?, f, b, b\}$, an instance of $\{-, f, ?, f, b, b\}$, i.e., $C10$; lastly, if both $r2$ and $r3$ are background pixels, nothing else needs to be done, and the configuration will transit to $\{b, b, ?, f, b, b\}$, an instance of $\{-, b, ?, f, b, f\}$, i.e., $C9$.

(5) For the case $K5$, we assign the label assigned to $l2$ to $p2$. If $p3$ is an object pixel, we assign the same label to it. The configuration will transit to either $\{?, ?, ?, b, f, f\}$ or $\{?, ?, ?, b, f, b\}$, both are an instance of $\{?, ?, ?, b, f, -\}$, i.e., $C11$.

(6) For the case $K6$, we assign the label assigned to $l2$ to $p3$. The configuration will transit to $\{?, ?, ?, b, b, f\}$, i.e., $C14$.

According to the above analysis, after processing, the configuration $C2$ will transmit to one of the configurations $C1$, $C2$, $C3$, $C4$, $C5$, $C6$, $C7$, $C8$, $C9$, $C10$, $C11$, and $C14$.

Other configurations shown in Fig. 5 can be analyzed in a similar way. The next configurations to which a configuration may transmit are shown in Table 1.

Table 1 Configuration transition relationship.

Current configuration	Next configuration
$C1$, $C2$, $C5$, $C8$, $C11$, $C12$, $C13$, $C14$, $C15$, $C16$	$C1$, $C2$, $C3$, $C4$, $C5$, $C6$, $C7$, $C8$, $C9$, $C10$, $C11$, $C14$
$C3$, $C6$, $C9$	$C1$, $C2$, $C3$, $C4$, $C5$, $C6$, $C7$, $C8$, $C9$, $C10$, $C12$, $C15$
$C4$, $C7$, $C10$	$C1$, $C2$, $C5$, $C8$, $C13$, $C16$

4. Experimental Results

In this section, we compared our proposed algorithm with the IFCL algorithm and the CTB algorithm. Our proposed algorithm was implemented in a similar way as the CTB algorithm introduced in [8], and the codes of the IFCL algorithm and the CTB algorithm were provided by their authors.

All experiments were performed on a PC-based workstation (Intel Core i5-3470 CPU@3.20GHz, 4GB Memory, Ubuntu Linux OS). The three algorithms were implemented in C language and compiled by the GNU C compiler (version 4.2.3) with the option –O3. All experimental results presented in this section were obtained by averaging of the execution time for 5000 runs. Moreover, the labeling results for all image obtained by any algorithm are exactly the same.

Four types of images are used for testing in our experiment: artificial images, natural images, texture images, and medical images.

Artificial images contain specialized patterns (stair-like, spiral-like, saw-tooth-like, checker-board-like, and honeycomb-like connected components) and noise images. The dataset of noise images was downloaded from [8]. The size of the noise images is 1024×1024. There are nine different foreground densities (from 0.1 to 0.9), and ten images for each density (thus, 90 images in total).

Natural images, including landscape, aerial, fingerprint, portrait, still-life, snapshot, and text images, were obtained from the Standard Image Database (SIDBA) developed by the University of Tokyo [9] and the image database of the University of Southern California [10], there are 50 images in this type of images were used for realistic testing of labeling algorithms. In addition, seven texture images downloaded from the Columbia-Utrecht Reflectance and Texture Database [11], and 25 medical images obtained from a medical image database of the University of Chicago, were used for testing. Images in these three types were 512×512 pixels in size, and they were transformed into binary images using Otsu's threshold selection method in [12].

4.1 Execution Time versus the Densities of Images

Noise images with a size of 1024×1024 pixels were used for testing the execution time versus the density of the foreground pixels in an image. The results are shown in Fig. 6,

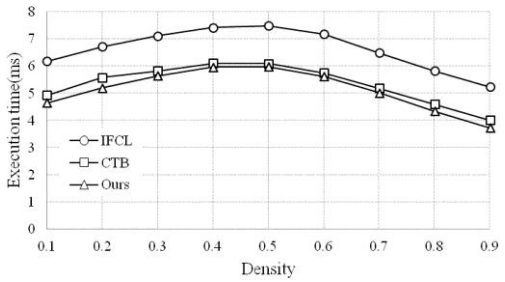

Fig. 6 Execution time (ms) versus the density of foreground pixels in noise images.

Table 2 Various execution times (*ms*) for various types of images.

Image Type		IFCL	CTB	Ours
Natural	Max.	1.77	1.43	1.39
	Mean	1.43	1.13	1.06
	Min.	1.19	0.90	0.83
Medical	Max.	1.46	1.15	1.08
	Mean	1.34	1.05	0.98
	Min.	1.27	0.98	0.90
Textural	Max.	1.66	1.37	1.18
	Mean	1.52	1.21	1.15
	Min.	1.32	1.07	1.02
Artificial	Max.	1.42	1.10	1.01
	Mean	0.84	0.66	0.60
	Min.	0.31	0.28	0.23

where the value for each density is the average execution time on the ten images with the density. For all density images, the CTB algorithm is faster than the IFCL algorithm, and our proposed algorithm is fastest.

4.2 Comparisons in Terms of the Maximum, Average, and Minimum Execution Times

Natural images, medical images, texture images, and artificial images with specialized shape patterns were used for this test. The results of the comparisons are shown in Table 2. From Table 2, we find that, for all types of images, the CTB algorithm is faster than the IFCL algorithm, and our proposed algorithm is still fastest.

5. Conclusion

In this paper, we extended the CTB algorithm by scanning images lines every three lines and processing pixels three by three. In our proposed algorithm, the information obtained during processing the current three pixels will be used to process the next three pixels. Thus, it can reduce the repeated work for checking pixels in the mask. Experimental results demonstrated that our algorithm outperformed the CTB algorithm.

However, in comparison to the length of the code of the IFCL algorithm, which is about 50 lines, and the CTB algorithm, which is about 250 lines, the length of the code of our proposed algorithm is about 1300 lines.

Our proposed algorithm can be further extended by scanning image lines every four lines, and processing pixels four by four. However, because the cases that should be considered will also greatly increase, the length of the code for the algorithm might exceed 5000 lines. We can see from Fig. 6 and Table 2 that the CTB algorithm improves the IFCL algorithm a lot, but our proposed algorithm does not improve the CTB algorithm so much. Therefore, it might not be possible, in practice, to further improve our proposed algorithm.

Acknowledgements

We thank the anonymous referee for his/her valuable comments that improved this paper greatly. We are grateful to the associate editor, Prof. Hirano, for his kind cooperation.

References

[1] R.C. Gonzalez and R.E. Woods, Digital Image Processing, Addison-Wesley, Reading, MA, 1992.
[2] J. Trein, A.T. Schwarzbacher, and B. Hoppe, "FPGA implementation of a single pass real-time blob analysis using run length encoding," MPC-Workshop, pp.71–77, Ravensburg-Weingarten, Germany, 2008.
[3] L. He, Y. Chao, and K. Suzuki, "A run-based two-scan labeling algorithm," IEEE Trans. Image Process., vol.17, no.5, pp.749–756, 2008.
[4] L. He, Y. Chao, K. Suzuki, and K. Wu "Fast connected-component labeling," Pattern Recognit., vol.42, no.9, pp.1977–1987, 2009.
[5] K. Wu, E. Otoo, and K. Suzuki, "Optimizing two-pass connected-component labeling algorithms," Pattern Anal. Appl., vol.12, no.2, pp.117–135, 2009.
[6] L. He, Y. Chao, and K. Suzuki, "An efficient first-scan method for label-equivalence-based labeling algorithms," Pattern Recognit. Lett., vol.31, no.1, pp.28–35, 2010.
[7] C. Grana, D. Borghesani, and R. Cucchiara, "Optimized block-based connected components labeling with decision trees," IEEE Trans. Image Process., vol.19, no.6, pp.1596–1609, 2010.
[8] L. He, X. Zhao, Y. Chao, and K. Suzuki, "Configuration-transition-based connected-component labeling," IEEE Trans. Image Process., vol.23, no.2, pp.943–951, 2014.
[9] The Standard Image Database [Online]. Available: http://sampl.ece.ohio-state.edu/data/stills/sidba/index.htm
[10] The Image Database of the University of Southern California [Online]. Available: http://sipi.usc.edu/database/
[11] The Columbia-Utrecht Reflectance and Texture Database [Online]. Available: http://www1.cs.columbia.edu/CAVE/software/curet/index.php
[12] N. Otsu, "A threshold selection method from gray-level histograms," IEEE Trans. Syst. Man Cybern., vol.9, no.1, pp.62–66, Jan. 1979.