A New Connected-Component Labeling Algorithm

Yuyan Chao[1], Lifeng He[2], Kenji Suzuki[3], Qian Yu[4], Wei Tang[5]
1.Shannxi University of Science and Technology, China & Nagoya Sangyo University, Aichi, Japan, chao@nagoya-su.ac.jp
2.Aichi Perfectural University, Aichi, Japan, helifeng@ist.aichi-pu.ac.jp
3.The University of Chicago, Chicago, USA, Suzuki@uchicago.edu
4.Nagoya Institute of Technology, Nagoya, Japan
5.Shannxi University of Science and Technology, China

Abstract. This paper proposes a new first-scan method for two-scan labeling algorithms. In the first scan, our proposed method first scans image lines three by three with a leaving line, and for foreground pixels among each three lines, assigns them provisional labels, and finds and resolves label equivalences among them. Then, it processes the leaving lines from top to bottom one by one, and for each line, assigns foreground pixels on the line provisional labels, finding and resolving label equivalences between the foreground pixels and those on the lines immediately above and below the current line. Experimental results demonstrated that our method is more efficient than conventional label-equivalence-based labeling algorithms.

Keywords: connected component; labeling; pattern recognition

1. Introduction

Labeling of connected components in a binary image is one of the most fundamental operations in pattern analysis, pattern recognition, computer (robot) vision, and machine intelligence 0. Especially in real-time applications such as traffic-jam detection, automated surveillance, and target tracking, faster labeling algorithms are always desirable.

Many algorithms have been proposed for addressing this issue, because the improvement of the efficiency of labeling is critical in many applications. For ordinary computer architectures and 2D images, there are mainly two types of labeling algorithms: (1) Raster-scan algorithms [3-9], and (2) Label propagation algorithms [10-11]. According to experimental results on various types of images, the algorithm proposed in Ref. [7], which is an improvement on the two-scan algorithm proposed in Ref. [6], is the most efficient one, and has been used for various applications [12-13]. For convenience, we denote this algorithm as HCS_I *algorithm*.

The HCS_I algorithm is a two-scan labeling algorithm. it uses equivalent label sets and a representative label table to record equivalent labels and resolve the label equivalences. For convenience, an equivalent label set with the representative label u is denoted as $S(u)$, and the representative label of a provisional label s is t, denoted as $T[s] = t$.

In the first scan, this algorithm uses the mask shown in Fig. 1 (a), which consists of three scanned neighbor of the current foreground pixels, to assign provisional labels to foreground pixels, and to record and resolve label equivalences. At any moment, all equivalent provisional labels are combined in a equivalent label set with the same representative label.

For the case where the current foreground pixel follows a background pixel (Fig. 1 (b)), if there is no label (foreground pixel) in the mask, this means that the current foreground pixel does not connect with any scanned foreground pixel, and the current foreground pixel belongs to a new connected component. The algorithm assigns a new provisional label m to the current foreground pixel, which is initialized to 1, and establishes the equivalent label set $S(m)=\{m\}$; it sets the representative label table as $T[m] = m$, and $m = m+1$ for later processing. Otherwise, i.e., if there are some foreground pixels in the mask, all of such foreground pixels and the current foreground pixel belong to the same connected component. Therefore the current foreground pixel can be assigned any of the labels in the mask. On the other hand, for the case where the current foreground pixel follows another foreground pixel (Fig. 1 (c)), the current foreground pixel can be assigned the same label of that foreground pixel.

In any cases, if there are provisional labels belonging to different equivalent label sets in the mask, all provisional labels in those sets are equivalent labels, and they will be combined together.

As soon as the first scan is finished, all equivalent labels of each connected component have been combined into an equivalent label set with a unique representative label. In the second scan, by replacement of each provisional label with its representative label, all foreground pixels of each connected component will be assigned a unique label.

2. Outline of Our Proposed First-Scan Method

For an $N \times M$ binary image, we use $b(x, y)$ to denote the pixel value at (x, y) in the image, where $1 \le x \le N$, $1 \le y \le M$, and $v(x, y)$ for the value of $b(x, y)$. For convenience, we suppose that the value of foreground pixels is 1 and that of background pixels is 0. Moreover, all pixels in the edge of an image are considered to be background pixels. Because we don't make any processing for background pixels, we will not discuss the processing for background pixels.

Our first-scan method consists of two parts: Scan 1-A and Scan 1-B. In Scan1-A, from line 3, it scans image lines every four other lines, i.e., in the order of line 3, line 7, line 11, … (the black lines in Fig. 2). For each current line being processed, it assigns to the foreground pixels in the line and its neighbor lines (the gray lines in Fig. 2) provisional labels and resolves the label equivalences among them. By Scan 1-A, all foreground pixels in each area consisting of black and gray lines in Fig. 2 will be assigned provisional labels, and the label equivalences among them will be resolved.

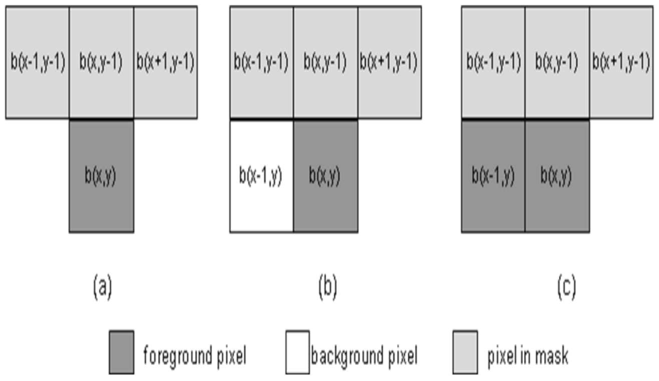

Fig. 1 Mask for the eight-connected connectivity.

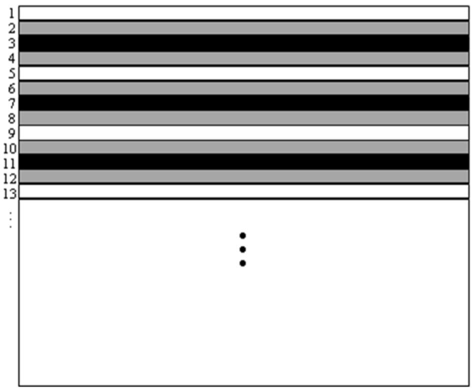

Fig. 2 First scan in our proposed method.

Then, Scan 1-B scans the lines unprocessed in the Scan 1-A (the white lines in Fig. 2) in the order of line 5, line 9, line 13, …. For each current line, it assigns to the foreground pixels in the line provisional labels and resolves the label equivalences among them and those in their neighboring lines. When Scan 1-B is finished, all foreground pixels in the given image will be assigned provisional labels, and the label equivalences among them will be resolved, i.e., all equivalent labels will be combined into an equivalent label set with a unique representative label, similar to the result of the first scan in the HCS_I algorithm.

In the second scan, similar to other two-scan labeling algorithms, by replacing the provisional label of each foreground pixel with its representative label, we can complete the whole labeling process.

In Scan 1-A, our method uses the mask shown in Fig.3 to assign to the current pixel *b*(*x*, *y*), its neighbor above, *b*(*x*, *y*−1)and its neighbor below, b(*x*, *y*+1)provisional labels, and to resolve the label equivalences in the mask. The following four cases can be considered.

Case 1: The current pixel *b*(*x*, *y*) is a background pixel that follows another background pixel (Fig.4 (b), (c), (e), (f)). When the neighbor above *b*(*x*, *y*-1) (the neighbor below *b*(*x*, *y*+1)) of the current pixel is a foreground pixel, we can process it as follows: if its neighbor left is a foreground pixel, assigning that pixel's label to it, otherwise, assigning a new provisional label to it.

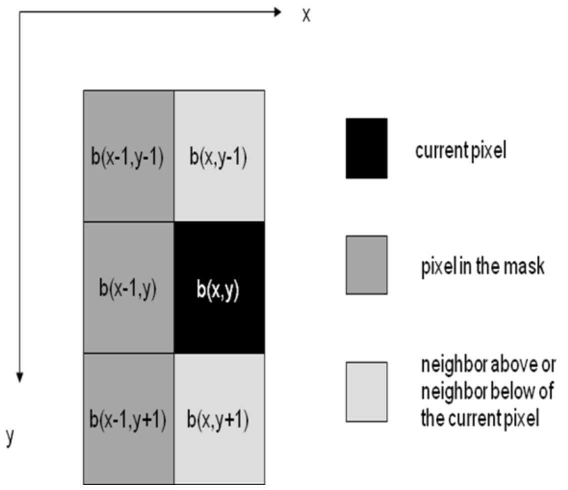

Fig. 3 Masks used in our proposed method for Scan 1-A.

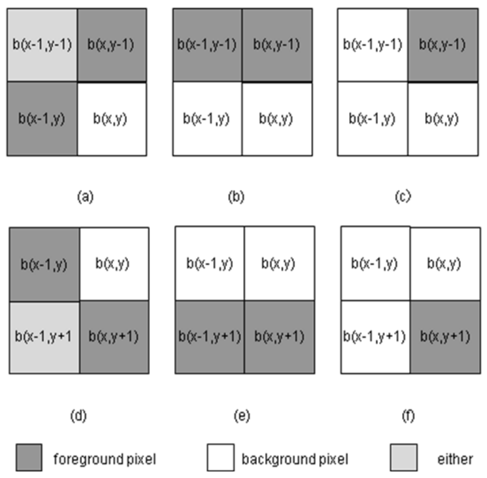

Fig. 4 Cases where the current pixel is a background pixel. in Scan 1-A.

Case 2:The current pixel is a background pixel following a foreground pixel, i.e., *b*(*x*-1, *y*) is a foreground pixel (Fig. 4 (a), (d)).When the neighbor above *b*(*x*, *y*-1) (the neighbor below, *b*(*x*, *y*+1)) of the current pixel is a foreground pixel, we can just assign *b*(*x*-1, *y*)'s label to it.

Case 3:The current pixel is a foreground pixel following a background pixel (Fig. 5 (b)-(e)). The current foreground pixel and its neighbor above *b*(*x*, *y*-1) and neighbor below *b*(*x*, *y*+1) can be processed as follows: (1) if both of *b*(*x*-1, *y*-1) and *b*(*x*-1, *y*+1) are foreground pixels, we resolve the label equivalence between the two corresponding labels, and assign their representative label to the current pixel; (2) if either *b*(*x*-1, *y*-1) or *b*(*x*-1, *y*+1) is a foreground pixel, then assigning its label to the current label;(3) if none of *b*(*x*-1, *y*-1) and *b*(*x*-1, *y*+1) is a foreground pixel, then assigning a new label to the current pixel. Moreover, we assign the current pixel's label to its neighbor above (its neighbor below) if that pixel is a foreground pixel.

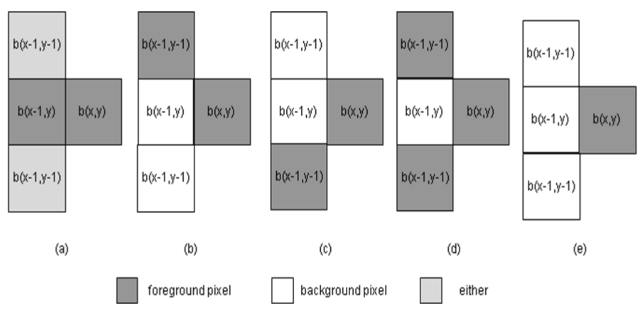

Fig. 5 Cases where the current pixel is a foreground pixel in Scan 1-A.

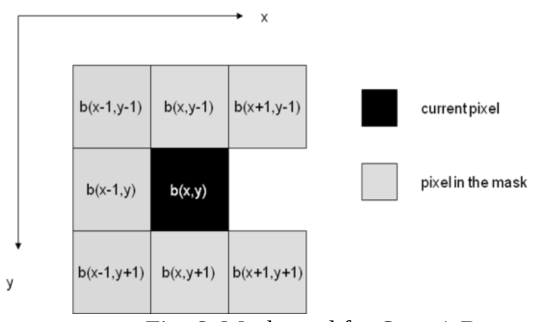

Fig. 6 Mask used for Scan 1-B.

Case 4: The current pixel is a foreground pixel following a foreground pixel, i.e., *b*(*x*-1, *y*) is a foreground pixel (Fig. 5 (a)). We just assign *b*(*x*-1, *y*)'s label to the current pixel. Moreover, we assign the same label to its neighbor above (its neighbor below) if that pixel is a foreground pixel.

After Scan 1-A, from line 5, in the order of line 5, line 9, line 13, …, Scan 1-B scans the lines unprocessed in Scan 1-A. It does nothing for background pixels. For each foreground pixel, it uses the mask shown in Fig. 6 to assign to the pixel a provisional label and resolves the label equivalences in the mask. Because the foreground pixels that are connected each other in the mask, such a combination is called a connected part, belonging to the same connected component, and their provisional labels are equivalent labels and belong to the same equivalent label set; thus, a connected part can be considered as if a single foreground pixel. For this reason, checking the pixels in the mask in the order of the largest number of the neighbors of a pixel first will reduce the number of times for checking pixels in the mask; thus, it leads to efficient processing [9].

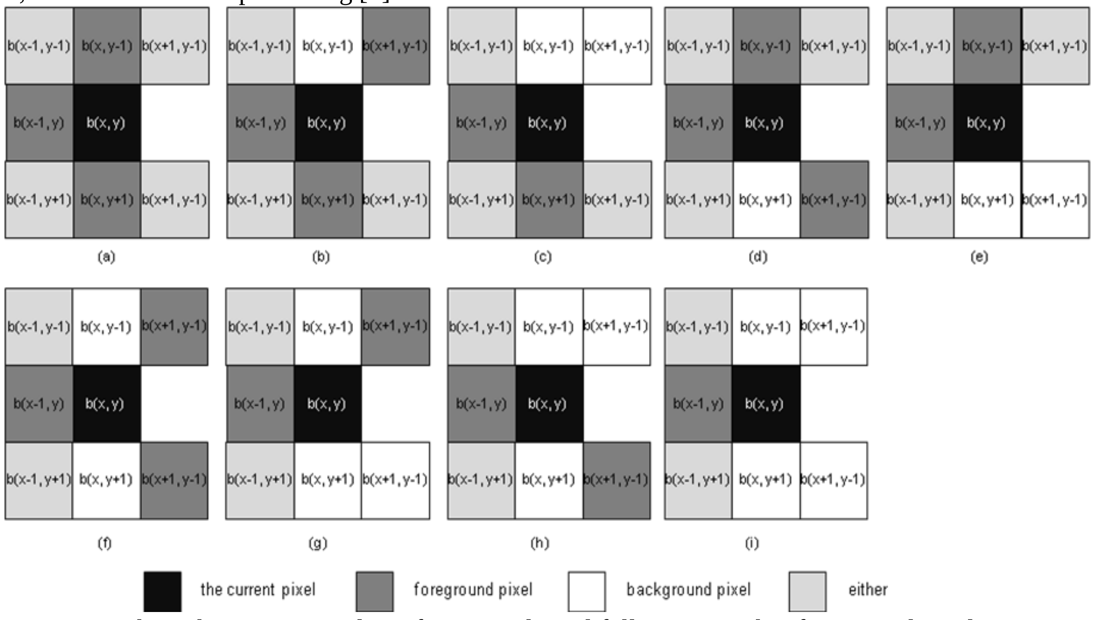

Fig. 7 Cases where the current pixel is a foreground pixel following another foreground pixel in Scan 1-B.

If the current pixel is a foreground pixel following another foreground pixel, there are nine subcases shown in Fig. 7. In Scan 1-B, we process the current pixel *b*(*x*, *y*) as follows: (1) because *b*(*x*-1, *y*) is a foreground pixel, we assign *b*(*x*-1, *y*)'s label to the current foreground pixel; (2) because the number of connected parts does not depend on whether *b*(*x*-1, *y*-1) and/or *b*(*x*-1, *y*+1) is a background pixel or a foreground pixel, we do not need to check either of them; (3) according to the number of neighbors of each pixel in the mask, the order for checking the pixels except for *b*(*x*-1, *y*-1), *b*(*x*-1, *y*+1), and *b*(*x*-1, *y*) is *b*(*x*, *y*-1)$\rightarrow$ *b*(*x*, *y*+1) $\rightarrow$ *b*(*x*+1, *y*-1) $\rightarrow$ *b*(*x*+1, *y*+1). If *b*(*x*, *y*-1) is a foreground pixel (e.g., Fig. 7 (a)), *b*(*x*, *y*-1) and *b*(*x*-1, *y*) belong to the same connected part, we need to do nothing for the pixel. Moreover, in this subcase, whether *b*(*x*+1, *y*-1) is a foreground pixel or not does not change the number of connected parts in the mask, we also need to do nothing about this pixel. On the other hand, if *b*(*x*, *y*-1) is a background pixel and *b*(*x*+1, *y*-1) is a foreground pixel (e.g., Fig. 7 (b)), we need to resolve the label equivalence of *v*(*x*-1, *y*) and *v*(*x*+1, *y*-1). Then, *b*(*x*, *y*+1) and *b*(*x*+1, *y*+1) can be processed in a similar way.

On the other hand, in the case where the current foreground pixel *b*(*x*, *y*) follows a background pixel, there are 21 subcases, as shown in Figure 8. Except for *b*(*x*-1, *y*), according to the number of neighbors of each pixel in the mask, the order for checking pixels is *b*(*x*, *y*-1)$\rightarrow$ *b*(*x*, *y*+1) $\rightarrow$ *b*(*x*-1, *y*-1) $\rightarrow$ *b*(*x*-1, *y*+1) $\rightarrow$ *b*(*x*+1, *y*-1) $\rightarrow$ *b*(*x*+1, *y*+1). For each pixel, if there are more than one connected part in the mask (e.g., Figure 8 (a)-(d)), we need to resolve the label equivalences among them, and assign to the current foreground pixel its representative label. On the other hand, if there is only one connected part in the mask (e.g., Figure 8 (e)), we only need to assign to the current foreground pixel its representative label. Lastly, if there is no foreground pixel in the mask (Figure (u)), we only need to assign to the current pixel a new provisional label.

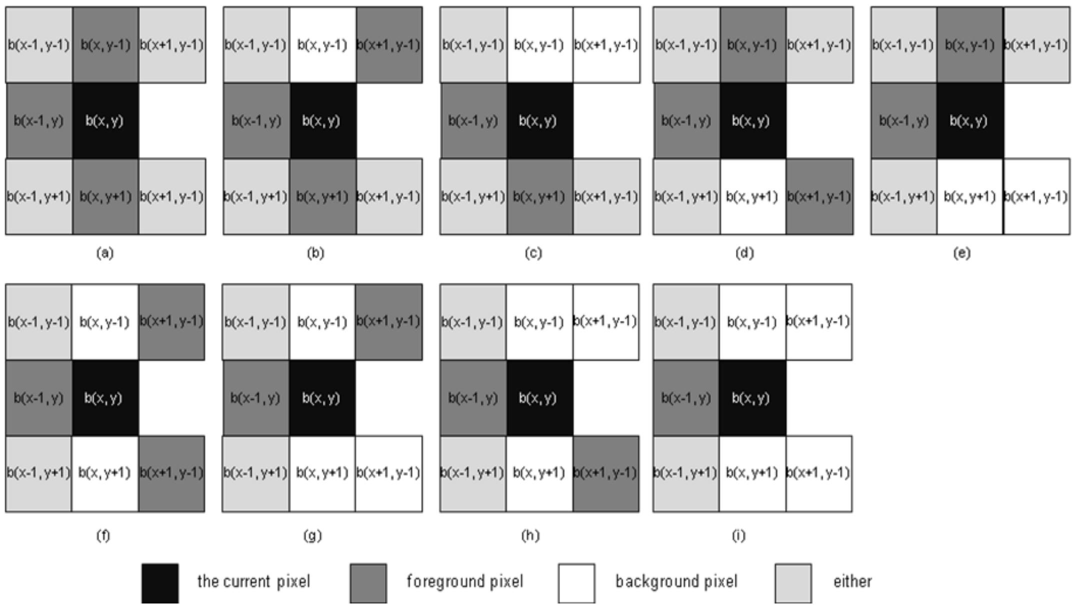

Fig. 8 Cases where the current pixel is a foreground pixel following a background pixel in Scan 1-B.

As soon as the first scan is finished, all provisional labels assigned to each connected component have been combined in an equivalent label set with a unique representative label. During the second scan, similar to all conventional two-scan labeling algorithms, by replacing each provisional label with its representative label, we can complete labeling.

3. Comparative Evaluation

We implemented the HCS_I algorithm and our algorithm with the C language on a PC-based workstation (Intel Pentium D 3.0 GHz + 3.0 GHz CPUs, 2 GB Memory, Mandriva Linux OS). Because our method is a new first-scan method (as we described above, the second scan of our method is exactly the same with the HCS_I method), we will compare the performances of the two methods only on the first scan. All data in this section were obtained by averaging of the execution time for 10,000 runs with a single core. Images used for testing included of four types: noise images, natural images, texture images, and medical images.

Noise images consist of forty one 512X512-sized noise images were generated by thresholding of the images containing uniform random noise with 41 different threshold values from 0 to 1000 in steps of 25.

On the other hand, 50 natural images, including landscape, aerial, fingerprint, portrait, still-life, snapshot, and text images, obtained from the Standard Image Database(SIDBA)developed by the University of Tokyo (http://sampl.ece.ohio-state.edu/data/stills/sidba/index.htm) and the image database of the University of Southern California (http://sipi.usc.edu/database/), were used for realistic testing of labeling algorithms. In addition, seven texture images, which were downloaded from the Columbia-Utrecht Reectance and Texture Database (http:// www 1.cs. columbia.edu/CAVE/software/curet/index.php), and 25 medical images obtained from a medical image database of The University of Chicago were used for testing. All of these images were 512×512 pixels in size, and they were transformed into binary images by a standard thresholding method.

Fig. 9 shows the speed-up of our method compared to the HCS_I method on the 512×512 noise images, where the vertical axis is defined as $(t_1-t_2)/t_1$, where t_1 is the execution time of the HCS_I algorithm and t_2 is that of the proposed method.

The experimental results on the natural images, the medical images, and the textural images are shown in Tab. 1.

Tab. 1 Comparation on various types of images [ms].

Image		HCS_I	Ours
Natural	Max.	1.83	1.57
	Mean.	0.96	0.89
	Min.	0.44	0.39
Medical	Max.	0.98	0.95
	Mean	0.73	0.70
	Min.	0.59	0.29
Textural	Max.	1.48	1.36
	Mean	1.09	0.95
	Min.	0.79	0.53

4. Discussion

Because our algorithm processes an image piecewise, it is not as efficient as the MECTSL algorithm, which does that successively. Therefore, our algorithm is not as efficient as the MECTSL algorithm for the noise images whose densities are lower than 10%, where the advantage of our algorithm for resolving connectivity cannot be exerted. For high-density noise images, because our method processes an image every four lines in Scan1-A, the number of provisional labels assigned by our algorithm is much larger than that assigned by the HCS_I algorithm. For example, for the highest density noise image, the number of provisional labels assigned by our algorithm and that assigned by the HCS_I algorithm are 129 and 2, respectively. For each provisional label, we need to apply some operations to establish a new equivalent label set and initialize the representative label table. Moreover, because the connectivity of the foreground pixels in this case is simple, the advantage of our method for resolving connectivity cannot be exerted, and the effect of the efficiency of our algorithm becomes weaker and weaker with the increase of the density of an image from 80%.

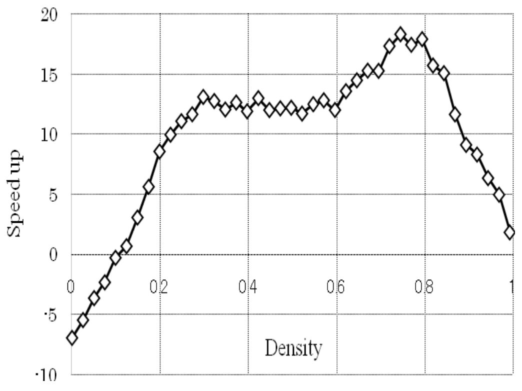

Fig. 9 Speed up on the densities of noise images.

5. Conclusions

In this paper, we presented a new method for the first scan of label-equivalence based two-scan labeling algorithms. In our proposed method, the first scan consists of two subscans: Scan 1-A and Scan 1-B. In Scan 1-A, we process image lines every four lines. For each current line being scanned, we assign to the foreground pixels in the line and its neighboring lines provisional labels and resolve the label equivalences among them. In Scan 1-B, we scan the lines that were unprocessed in Scan 1-A one by one. For each current line, we assign to the foreground pixels in the line provisional labels and resolve the label equivalences among them and those in its neighboring lines processed in Scan 1-A. By our method, the number of times for checking pixels for assigning provisional labels and processing label equivalences is decreased; thus, the efficiency of labeling is improved. Our experimental results demonstrated that our method was more efficient than the first scan of conventional label-equivalence-based two-scan labeling algorithms.

6. Acknowledgment

This research was partially supported by the Ministry of Education, Science, Sports and Culture, Japan, Grant-in-Aid for Scientific Research (C), 23500222, 2011.

References

[1] C. Ronsen and P. A. Denjiver. Connected Components in Binary Images: The Detection Problem, Research Studies Press, 1984.
[2] R. C. Gonzalez and R. E. Woods. Digital Image Processing. Addison Wesley, 1992.
[3] K. Suzuki, I. Horiba, and N. Sugie. Linear-time connected-component labeling based on sequential local operations. Computer Vision and Image Understanding, 89:1-23, 2003.
[4] A. Rosenfeld and J. L. Pfalts. Sequential operations in digital picture processing. Journal of ACM, 13(4):471-494, October 1966.
[5] L. He, Y. Chao, and K. Suzuki. A Run-based Two-Scan Labeling Algorithm. IEEE Transactions on Image Processing, 17(5):749-756, 2008
[6] L. He, Y. Chao, and K. Suzuki, K. Wu. Fast Connected-Component Labeling. Pattern Recognition, 42 (2009):1977
[7] L. He, Y. Chao, and K. Suzuki. An Efficient First-Scan Method for Label-Equivalence-Based Labeling Algorithms. Pattern Recognition Letters, 31:28-35, 2010.
[8] L. He, Y. Chao, and K. Suzuki. A Run-Based One-and-a-Half-Scan Connected-Component Labeling Algorithm. International Journal of Pattern Recognition and Artificial Intelligence, Vol. 24, No. 4(2010), pp.557-579.
[9] L. He, Y. Chao, and K. Suzuki. Two Efficient Label-Equivalence-Based Connected-Component Labeling

Algorithms for Three-Dimensional Binary Images. IEEE Transactions on Image Processing, 20(8), 2011, DOI:. 10.1109/TIP.2011.2114352.

[10] F. Chang, C. J. Chen, and C. J. Lu. A linear-time component-labeling algorithm using contour tracing technique. Computer Vision and Image Understanding, 93:206-220, 2004.

[11] Q. Hu, G. Qian and W. L. Nowinski, Fast connected-component labeling in three-dimensional binary images based on iterative recursion. Computer Vision and Image Understanding, 99:414-434, 2005

[12] A. Alexey, K. Tomas, W. Florentin, and D. Babette. Real-Time Image Segmentation on a GPU. Facing the Multicore-Challenge, Lecture Notes in Computer Science, 6310:131-142, 2011, Springer Berlin / Heidelberg.

[13] Christopher Wolfe, T. C. Nicholas Graham, and Joseph A. Pape. Seeing through the fog: an algorithm for fast and accurate touch detection in optical tabletop surfaces. In ACM International Conference on Interactive Tabletops and Surfaces (ITS '10). ACM, 73-82, 2010, New York, NY, USA.