FAST BLOCK-BASED ALGORITHMS FOR CONNECTED COMPONENTS LABELING

Diêgo J. C. Santiago[1], Tsang Ing Ren[1], George D. C. Cavalcanti[1] and Tsang Ing Jyh[2]

[1]Center for Informatics, Federal University of Pernambuco
Recife, PE, Brazil – www.cin.ufpe.br/~viisar
{djcs, tir, gdcc}@cin.ufpe.br
[2]Alcatel-Lucent, Belgium
{ing-jyh.tsang}@alcatel-lucent.com

ABSTRACT

Block-based algorithms are considered the fastest approach to label connected components in binary images. However, the existing algorithms are two-scan which would need more comparisons if they were used as one-and-a-half-scan algorithms. Here, we proposed a new mask that enables the design of a block-based one-and-a-half-scan algorithm without any extra comparison. Furthermore, three new efficient algorithms for connected components labeling are presented: a block-based two-scan, a pixel-based one-and-a-half-scan and a block-based one-and-a-half-scan. We conducted experiments using synthetic and realistic images to evaluate the performance of the proposed methods compared to the existing methods. The proposed block-based one-and-a-half-scan algorithm presents the best performance in the realistic images dataset composed of 1290 documents. Our block-based two-scan algorithm proved to be the fastest in the synthetic dataset, especially in low density images.

Index Terms— Connected components labeling, block-based, one-and-a-half-scan, image processing, image analysis

1. INTRODUCTION

Object detection and classification is a common problem in computer vision. In binary images analysis, objects are extracted by means of Connected Components Labeling (CCL), which distinguish objects in an image by assigning a unique label for each connected component.

CCL is usually a step between low-level image processing (filtering) and high-level image processing (detection, recognition). A common CCL post processing is features extraction. Features such as area, bounding box, spatial moments and their derivatives can be extracted from the image in a single pass [9]. Nevertheless, other features, for instance, the contour [8], convex hull and their derivatives need at least a second pass. The use of a second pass only through the region of interest is more convenient instead of the whole image.

Many labeling algorithms have been proposed. For ordinary computer architectures and pixel-based images representation, we divide them in two types depending on the way they access the binary input image and the label image: irregular [1,10,11] and regular access [14]. The algorithms of the first type lead to poor cache behavior when executed on modern processors. This is due to the irregular access of image data. As a result, they are slower in large and dense images [3,15].

In the regular access type, the number of scans usually determines the speed of the algorithm. They are multi-scan [4], four-scan [13], two-scan [3,5,6,7,12,14,15] and one-and-a-half-scan [8]. Consequently, the two-scan and one-and-a-half-scan are the fastest of this type. Based on the first scan image operation, these algorithms are also classified into three classes as pixel-based [4,6,7,13,14,15], run-based [5,8] and block-based [3,12] masks. These masks determine the number of comparisons or times for checking the neighboring pixels, which we should minimize in order to speed up the algorithm.

In general, the fastest labeling algorithms belong to the block-based connected components labeling class [3]. Still, these algorithms do not provide enough information to perform the second pass through only the foreground pixels. He *et al.* [8] resolves this issue by introducing a run-based one-and-a-half-scan labeling algorithm. This algorithm scans the background pixels once and the foreground pixels twice. In the second scan, by using the recorded run data, it assigns each foreground pixel a final label directly, without scanning any background pixel.

This paper presents three new efficient algorithms for connected components labeling: a block-based two-scan, a pixel-based one-and-a-half-scan and a block-based one-and-a-half-scan. A new block-based mask is created in order to minimize the number of comparisons and provisional labels created in the first scan. This mask enables the design of block-based one-and-a-half-scan algorithms without any extra comparison.

The rest of this paper is organized in the following way. The next section reviews the pixel-based and the block-based algorithms. Section 3 introduces our proposed algorithms. We show experimental results and analysis in Section 4. Finally, we present some conclusions in Section 5.

2. RELATED WORK

In an image, we can define two types of connectivity: four-connectivity and eight-connectivity [2]. In the following discussion, we adopt the eight-connectivity.

The classical labeling algorithm proposed by Rosenfeld et al. in [14] is pixel-based. It scans the image twice, pixel by pixel, in raster order. In the first scan, by using a mask, a temporary label is assigned to each foreground pixel based on the values of its neighbors already visited by the scan. The mask used is shown in Figure 1.a. When a foreground pixel with two foreground neighbors carrying different labels is found, the labels associated with the pixels in the neighborhood are registered as equivalent. Therefore, after completion of the first scan, equivalent labels are sorted into equivalence classes and a unique class identifier is assigned to each class. A second scan is then run over the image, pixel by pixel, so as to replace each temporary label by the class identifier of its equivalence class.

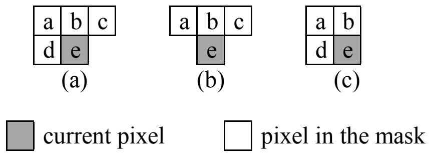

Figure 1. Masks used in pixel-based algorithms.

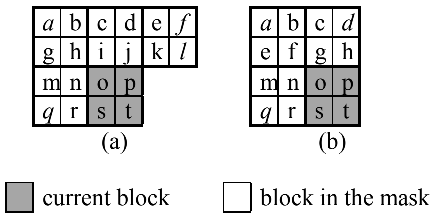

Figure 2. Masks used in block-based algorithms.

Wu *et al.* [15] defined an interesting optimization that reduced the number of comparisons to determine the provisional label. They exploited a decision tree so as to minimize the number of neighboring pixels to be visited when it evaluates the label of the current pixel. In an eight-connectivity, among all the neighboring pixels, only one of them is sufficient to determine the label of the current pixel [6]. Thus, we just need to resolve the label equivalences [6].

An improved algorithm was proposed by He *et al.* in [7], this method divides the foreground pixels into two groups: foreground pixel following background pixel and foreground pixel following foreground pixel. And this operation is done without any extra work. Thus, the number of pixels in the mask can be reduced to three as illustrated in Figure 1.b.

The first block-based algorithm was proposed by Grana *et al.* [3]. In the first pass, they scan an image by moving over an extended mask of five 2 × 2 blocks as shown in Figure 2.a. Some pixels do not provide an eight-connection between blocks of the mask (*a*, *f*, *l* and *q*) and can be ignored. Since all the pixels of a 2 × 2 square are connected to each other, the algorithm labels each block rather than each pixel. Because of this, in the second pass, it needs to access the binary and label image, in order to evaluate the final label. If the provisional label is 0, i.e. all the pixels of the block are background, it does not need to access the binary image.

Sutheebanjard *et al.* [12] proposed a new mask, which is shown in Figure 1.c, in order to improve performance of high density images by creating a balanced decision tree. This algorithm performs slower than He *et al.* [7]. However, the extension algorithm with block-based mask, shown in Figure 2.b, performs faster than Grana *et al.* [3], except in low density images. The pixels *a, d* and *q* in the mask are ignored, once they are not connected to the other blocks.

3. PROPOSED ALGORITHMS

A one-and-a-half scan algorithm was proposed by He *et al.* [8] for run-based mask. Here, we propose one-and-a-half scan algorithms for pixel-based and block-based mask. A new block-based mask was created so as to enable the design of block-based one-and-a-half algorithms without changing the number of comparisons in the first scan.

For an NxM-sized image, we use $L(x,y)$ to denote the label pixel at (x,y). We use l to denote the value of the last label created and *lab* the value of the last label assigned. As in most labeling algorithms [1,5,6,7,8,10,13,15], we assume that all pixels on the border of an image are background pixels.

3.1. New block-based mask

The new mask proposed is shown in the Figure 3. In the first scan, the proposed algorithm scans an image by moving that mask composed of four 2 × 1 blocks, labeling 2 pixels at the same time. The pixels *a* and *f* are not connected to the other blocks and are ignored. As the algorithm of He *et al.* in [7], the last pixel *g* is known without extra work and does not belong to the mask. The second scan is the same of the pixel-based algorithms.

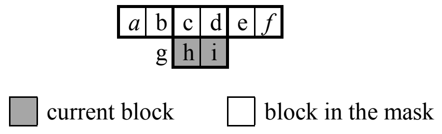

Figure 3. The mask used in our first-scan method.

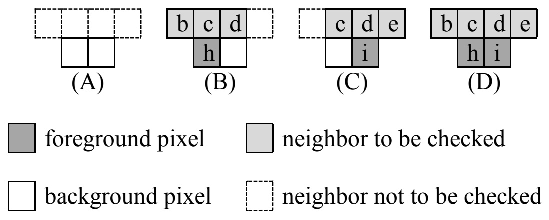

Figure 4. Four possible cases in our mask regarding the two pixels in the block.

There are four cases in our mask to be considered regarding the two pixels being labeled, as shown in Figure 4. Case (A) does not need any operation. In Case (B), the *procedure 1* proposed by He *et al.* in [7] is performed whether the current block is after a background pixel or the *procedure 2* proposed by He *et al.* in [7] whether the current block is after a foreground pixel. In the Case (C), the *procedure 1* is performed.

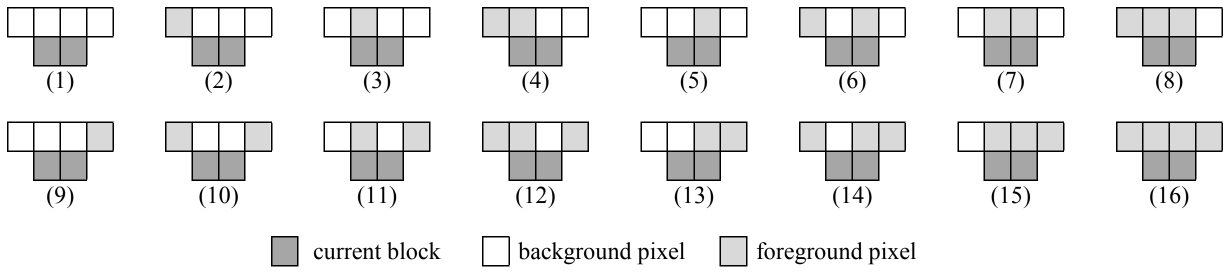

Figure 5. Sixteen possible cases in our mask when two foreground pixel in the block are considered.

Table 1. Operations performed in the sixteen cases where the current block has two foreground pixels.

Case	Operations after a background pixel	Operations after a foreground pixel
(1)	*lab*=*l*, *l* = *l* +1	-
(2)	*lab*=*L*(*b*)	-
(3)	*lab*=*L*(*c*)	-
(4)	*lab*=*L*(*c*)	-
(5)	*lab*=*L*(*d*)	*resolve*(*lab*,*L*(*d*))
(6)	*lab*=*L*(*b*), *resolve*(*lab*,*L*(*d*))	*resolve*(*lab*,*L*(*d*))
(7)	*lab*=*L*(*c*)	-
(8)	*lab*=*L*(*c*)	-
(9)	*lab*=*L*(*e*)	*resolve*(*lab*,*L*(*e*))
(10)	*lab*=*L*(*b*), *resolve*(*lab*,*L*(*e*))	*resolve*(*lab*,*L*(*e*))
(11)	*lab*=*L*(*c*), *resolve*(*lab*,*L*(*e*))	*resolve*(*lab*,*L*(*e*))
(12)	*lab*=*L*(*c*), *resolve*(*lab*,*L*(*e*))	*resolve*(*lab*,*L*(*e*))
(13)	*lab*=*L*(*d*)	*resolve*(*lab*,*L*(*d*))
(14)	*lab*=*L*(*b*), *resolve*(*lab*,*L*(*d*))	*resolve*(*lab*,*L*(*d*))
(15)	*lab*=*L*(*c*)	-
(16)	*lab*=*L*(*c*)	-

Two procedures, *procedure 3* and *procedure 4*, which are summarized in the Table 1, were designed in order to process Case (D). For this case, there are sixteen cases in our mask, as shown in Figure 5.

In Case (D), for the current block following a background pixel, we use the *procedure 3*. A new provisional label is assigned in Case (1). In the other cases, any label in the mask is assigned. In Cases (6), (10), (11), (12) and (14), we need to consider label equivalence, which is done by the operation *resolve*.

For the current block following a foreground pixel in case (D), *procedure 4* is used. The pixel is labeled with the last label assigned *lab*. Label equivalence must be considered in Cases (5), (6), (9), (10), (11), (12), (13) and (14).

3.2. One-and-a-half-scan algorithms

In order to perform a one-and-a-half-scan, the algorithms must record run data in the first scan. A run data store coordinates of start and end points of the run. The second scan is performed likewise the algorithm in [8].

For pixel-based one-and-a-half-scan, the first scan of the algorithm in [7] is modified in order to record the run data. The run start point is found as soon as a *procedure 1* is performed, whereas the run end point as soon as the last *procedure 2* is performed.

For block-based one-and-a-half-scan, the first scan of the algorithm in Section 3.1 is also modified. Finding the run start and end point is more complex in this algorithm. We should also consider both the current block following a background pixel and a foreground pixel.

For block following a background pixel, Case (A) does not need any operation. In Case (B), the run starts and ends in pixel *h*. In Case (C), the run starts in pixel *i*. And, in Case (D), the run starts in pixel *h*.

For block following a foreground pixel, in Case (A), the run ends in pixel *g*. In Case (B), the run ends in pixel *h*. In Case (C), the run ends in pixel *g* and other run starts in pixel *h*. Case (D) does not need any operation.

4. EXPERIMENTAL RESULTS

The experiment was performed on an Intel Core 2 Quad Q8300, 2.5 GHz, RAM 4 GB, 4 cores, using a single core for the processing. All experimental results presented in this section were obtained by averaging the execution time for 100 runs, from those 50 runs on Ubuntu 11.10 using the compiler GCC 4.6.3 -O2 and 50 runs on Windows Seven using the Microsoft Visual Studio 2010 compiler /O2.

All algorithms used for the comparison, which were implemented in C, are listed in Table 2. The programs of the BBDT (Block-based with Decision Tree) and sBBDT (squared Block-Based with Decision Tree) was improved and adapted from the OpenCV compliant version, available online at http://imagelab.ing.unimore.it/imagelab/labeling.asp and http://phaisarn.com/labeling, respectively, so as to consider all the pixels on the border of an image are background pixels. In tests, the border has size two. The source code of all algorithms can be downloaded from http://cin.ufpe.br/~djcs/labeling.

In order to evaluate the performance of the proposed first and second scan algorithm and then to avoid the effect of the equivalences resolution operation, the equivalences resolution of all algorithms followed the Union-Find technique as presented by He *et al.* in [5].

Table 2. Algorithms used in the comparisons.

Algorithm	Class	Reference
BBDT	Block-based Two-scan	Ref. [3]
sBBDT	Block-based Two-scan	Ref. [12]
EFS	Pixel-based Two-scan	Ref. [7]
BTS	Block-based Two-scan	Section 3.1
FOS	Pixel-based One-and-half-scan	Section 3.2
BOS	Block-based One-and-half-scan	Section 3.2

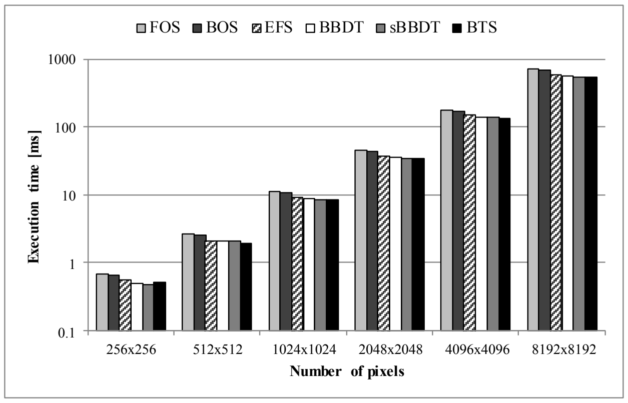

Figure 6. Performance of the algorithms scaling the size.

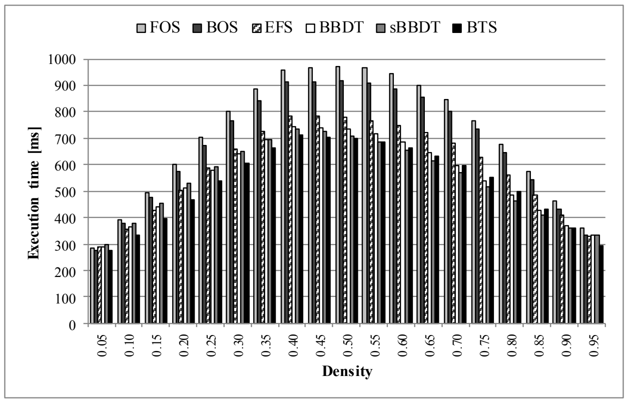

Figure 7. Performance of the algorithms varying the density.

4.1. Synthetic dataset

CCL algorithms are data dependent, and benchmarking such algorithms is not obvious. Analogously to most works [3,5,6,7,8,12,15], we used a synthetic dataset of black and white random noise square images. The synthetic dataset was composed by 114 images, with nineteen different densities, from 0.05 to 0.95, and six different images sizes: 256x256, 512x512, 1024x1024, 2048x2048, 4096x4096 and 8192x8192.

The experimental results demonstrated that our block-based two-scan algorithm is the fastest for images larger than 256x256 as shown in Figure 6. The existing block-based two-scan algorithm that uses the squared mask in Figure 2.b was the fastest for 256x256 images. The existing block-based algorithms are slightly impacted by cache memory, once they access four different image rows in order to define the provisional label, whereas ours access just two.

We also tested 8192x8192 images varying the density of the foreground pixels as shown in Figure 7. The experimental results show that our block-based algorithm was the fastest in twelve out of all nineteen densities, especially in low densities.

4.2. Analysis

The time complexity of the algorithms is determined mainly by the number per pixel of comparisons, merge operations, and memory accesses to both label and binary image. Table 3 shows those metrics that are measured from the synthetic image with size 8192x8192 and density 0.50. The number of memory accesses of one-and-a-half-algorithms depends on the number of runs, but the other numbers are the same of the two-scan version algorithms.

The algorithm sBBDT [3] performs the lowest number of comparisons and accesses to image. Nevertheless, it is the algorithm that performs the most number of merge operations. Our algorithms perform the least number of merge operations, once the condition of creating a provisional label is more restrict, and it is the second on fewer comparisons.

It is also important to point out that the algorithm EFS [7] and our algorithms access the binary image only once per pixel in order to determine whether the pixel is foreground or not. The other accesses are performed in the label image. In these algorithms, one array can be used for both the input image and the output image, which is not possible for the others.

Table 3. Analysis of the time complexity of the algorithms.

	Comparisons	Label image accesses	Binary image accesses	Total accesses	Merge operations
BBDT	1.842	1.681	2.530	4.211	0.028
sBBDT	**1.709**	1.685	2.397	**4.082**	0.034
EFS	1.875	3.375	1.000	4.375	0.028
BTS	1.781	3.281	1.000	4.281	**0.024**

4.3. Document image dataset

We also tested 1290 document images from the database Tobacco800 Document Image Database, available online at http://www.umiacs.umd.edu/~zhugy/tobacco800.html. It is a realistic dataset for document analysis. The number of pixels ranges from 1.8 to 9.2 MPixels, and the densities vary from 0.001 to 0.502. The experimental results demonstrated that our block-based one-and-a-half-scan consumes the lowest computation time as shown in Table 4.

Table 4. Execution time [ms] for document image dataset.

	EFS	BTS	sBBDT	BBDT	FOS	BOS
min	5.521	5.154	3.706	3.428	2.609	**2.240**
median	12.230	11.287	8.826	8.156	6.909	**6.184**
mean	14.496	13.363	10.785	10.018	8.637	**7.697**
max	40.464	37.521	37.515	**35.633**	39.962	38.827
std	8.720	7.946	6.591	6.110	5.487	4.946

5. CONCLUSION

In this paper, we proposed three algorithms for connected components labeling. A new block-based mask that reduces the number of comparisons, memory access and provisional labels was designed. The experimental results demonstrated that our block-based two-scan is more efficient for noise images, whereas our pixel-based and block-based one-and-a-half-scan is more efficient for document analysis. Because CCL algorithms are data dependent, for images with a lot of short runs, two-scan is better than one-and-a-half-scan. We expect that the more pixels labeling will accelerate the algorithm and, hence, those will be investigate in future works.

6. REFERENCES

[1] F. Chang, C.J. Chen, and C.J. Lu, “A linear-time component-labeling algorithm using contour tracing technique,” *Comput Vis Image Underst*, vol. 93, n. 2, pp. 206-220, 2004.

[2] R.C. Gonzalez, and R.E. Woods, *Digital Image Processing*, Prentice-Hall, New Jersey, 2002.

[3] C. Grana, D. Borghesani and R. Cucchiara, “Optimized block-based connected components labeling with decision trees,” *IEEE Transactions on Image Processing*, vol. 19, n. 6, pp. 1596-1609, 2010.

[4] R.M. Haralick, “Some neighborhood operations,” *Real Time Parallel Computing: Image Analysis*, Plenum Press, New York, pp. 11-35, 1981.

[5] L. He, Y. Chao, and K. Suzuki, “A run-based two-scan labeling algorithm,” *IEEE Transactions on Image*, vol. 17, n. 5, pp. 749–756, 2008.

[6] L. He, Y. Chao, K. Suzuki and K. Wu, “Fast connected-component labeling,” *Pattern Recognition*, vol. 42, pp. 1977-1987, 2008.

[7] L. He, Y. Chao, and K. Suzuki, “An efficient first-scan method for label-equivalence-based labeling algorithms,” *Pattern Recognition Letters*, vol. 31, n. 5, 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, n. 4, pp.557-579, 2011.

[9] J. Hoshen, “On the application of the enhanced Hoshen-Kopelman algorithm for image analysis,” *Pattern Recognition Letters*, vol. 19, pp.575-584, 1998.

[10] J. Martín-Herrero, “Hybrid object labeling in digital images. Machine,” *Vision and Applications*, vol. 18, pp. 1-15, 2007.

[11] Y. Shima, T. Murakami, M. Koga, H. Yashiro, and H. Fujisawa, “A high speed algorithm for propagation-type labeling based on block sorting of runs in binary images,” In *Proc. 10th Internat. Conf. Pattern Recogn.*, pp. 655–658, 1990.

[12] P. Sutheebanjard and W. Premchaiswadi, “Efficient Scan Mask Techniques for Connected Components Labeling Algorithm,” *EURASIP Journal on Image and Video Processing*, 2011:14.

[13] K. Suzuki, I. Horiba and N. Sugie, “Linear-time connected-component labeling based on sequential local operations,” *Comput Vis Image Underst.*, vol. 89, n. 1, pp.1-23, 2003.

[14] A. Rosenfeld and J.L. Pfaltz, “Sequential Operations in Digital Picture Processing,” *Journal of the ACM*, vol. 13, n. 4, pp. 471-494, 1966.

[15] K. Wu, E. Otto and K. Suzuki, “Optimizing two-pass connected-component labeling algorithms,” *Pattern Analysis and Applications*, vol. 12, pp. 117-135, 2008.