Journal of Computer Science 6 (10): 1099-1107, 2010
ISSN 1549-3636
© 2010 Science Publications

Connected Component Labeling Using Components Neighbors-Scan Labeling Approach

Akmal Rakhmadi, Nur Zuraifah Syazrah Othman, Abdullah Bade,
Mohd Shafry Mohd Rahim and Ismail Mat Amin
Department of Computer Graphics and Multimedia,
Faculty of Computer Science and Information Systems,
University Technology Malaysia, 81310 UTM, Skudai, Johor Bahru, Malaysia

Abstract: Problem statement: Many approaches have been proposed in previous such as the classic sequential connected components labeling algorithm which is relies on two subsequent raster-scans of a binary image. This method produced good performance in terms of accuracy, but because of the implementation of the image processing systems now requires faster process of the computer, the speed of this technique's process has become an important issue. **Approach:** A computational approach, called components neighbors-scan labeling algorithm for connected component labeling was presented in this study. This algorithm required scanning through an image only once to label connected components. The algorithm started by scanning from the head of the component's group, before tracing all the components neighbors by using the main component's information. This algorithm had desirable characteristics, it is simple while promoted accuracy and low time consuming. By using a table of components, this approach also gave other advantages as the information for the next higher process. **Results:** The approach had been tested with a collection of binary images. In practically all cases, the technique had successfully given the desired result. Averagely, from the results the algorithm increased the speed around 67.4% from the two times scanning method. **Conclusion:** Conclusion from the comparison with the previous method, the approach of components neighbors-scan for connected component labeling promoted speed, accuracy and simplicity. The results showed that the approach has a good performance in terms of accuracy, the time consumed and the simplicity of the algorithm.

Key words: Image processing, connected components labeling

INTRODUCTION

One of the purposes of image processing is to identify and recognize the shape of an object on a digital image. From this identification, the results can be used for certain purpose, such as pattern recognition. Therefore, some techniques are needed to support the identification process to make it easier and faster. The technique that can be used for this purpose is the connected component labeling technique.

Connected component labeling is one of the most common operations in virtually all image processing applications. In machine vision most objects have surfaces. The points in a connected component form a candidate region to represent an object. The image object, which is the component, is separated from the background image on binary image. Then each component is labeled and displayed as output images (Lee *et al*., 2007). Points belonging to a surface project to spatially closed points. The notion of 'spatially closed' is captured by connected components in digital images.

The classic sequential connected components labeling algorithm dates back to the early days of computer vision and image processing (Jain *et al*., 1995; Rosenfeld and Kak, 1982; Klette and Zamperoni, 1996). The algorithm relies on two subsequent raster-scans of a binary image. This method produced good performance in terms of accuracy, but because of the implementation of the image processing systems now requires faster process of the computer, the speed of this technique's process has become an important issue.

Improvement on the accuracy and labeling speed has been reported in various researches (Lee *et al*.,

Corresponding Author: Mohd Shafry Mohd Rahim, Department of Computer Graphics and Multimedia, Faculty of Computer Science and Information Systems, University Technology Malaysia, 81310 UTM, Skudai, Johor Bahru, Malaysia

2007; Falcao *et al*., 2005; Sossa and Guzman, 2000; Yang and Zhang, 2003; Rosenfeld and Pfaltz, 1966). For instance, Di Stefano and Bulgarelli (1999) and Jonas *et al*. (1997) proposed a comparatively more efficient and much simpler algorithm, that shows an improvement in the efficiency of the labeling process. Other connected component labeling algorithms have also been proposed by (Yang and Zhang, 2003; Haralick and Shapiro, 1991; Samet and Tamminen, 1986; Dillencourt *et al*., 1992), where most of the algorithms rely on two subsequent raster-scans of a binary image.

In the first scan a temporary label is assigned to each foreground pixel, based on the values of its neighbors (in the sense of 4-connectivity or 8-connectivity) that are already visited. When a foreground pixel with its foreground neighbors carrying different labels is found, the labels associated with the pixels in the neighborhood are registered as being equivalent. The second scan replaces each temporary label by the identifier of its corresponding equivalence class. Another technique is using scan line clustering method. One of the methods was proposed by Yang and Zhang (2003), which scans the pixel line by line and searching the neighbors of the pixels.

In this study, we describe a Components Neighbors-Scan algorithm to label connected components. The algorithm is an alternative approach in connected component labeling that is relatively low time consuming. This simple and easy to understand algorithm extends to its easy implementation. In many applications it is desirable to compute component characteristics while performing the labeling process. This is enabled in the presented algorithm, as it can capture the characteristics (such as size, position and bounding rectangle) of the connected components, conveniently making it easier to compute.

This study is organized as follows: The Introduction discusses the basic ideas of the Components Neighbors-Scan algorithm and explains the implementation of the algorithm, respectively. In the Materials and Methods, the Components Neighbors-Scan algorithm will be discussed in more detail to give a clear understanding in term of its methods. The speed process of the algorithm will be presented in the Result and Discussion using the natural image data for showing the performance. Finally the Conclusions are drawn.

Related study: Several optimizations of connected component labeling algorithms have also been attempted by (Gonzalez and Woods, 1992; He *et al*., 2007). Some of them enhanced the time consumed for processing the connected label while others reduced the complexity of the algorithm.

Rosenfeld and Pfaltz (1966) developed the very first algorithm in. The algorithm scans the image from left to right and top to bottom. This algorithm uses 4-neighbor forward raster scan mask as shown in Fig. 1.

The main pixel scanned is in the “e” position of the mask. The main position is the decision value for triggering the labeling process. The mask will be moved to the next scanning position if the value of the current pixel position e is 0. On the other hand if the main pixel e is 1 and all other 4-nbr positions are 0 then a new label is assigned to pixel “e”. If the four a, b, c, d neighbors are not zero, all of the pixels will be assigned with the lowest label. These data label are stored in a matrix array. With this matrix array the redundancies in labels are recognized after the labels are identified. From this matrix the redundancies are removed and replaced with the certain class label. After that the pixels will be replaced with the final equivalence classes in the second scan pass. Because of the large data contained in the matrix the process speed are affected (Fig. 2).

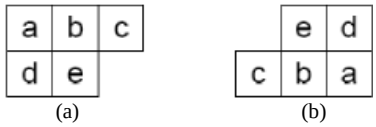

Fig. 1: The labeling mask of eight connected components. (a) Forward raster scan; (b) Backward raster scan

```
// lp, lq, lx: labels assigned to p, q, x
// B: background, F: foreground.
// FIRST SCAN:
for(i=1; i<NROWS-1; i++)
for(j=1; j<NCOLS-1; j++) {
if (I[i,j]==F) {
lp = I[i-1,j];
lq = I[i,j-1];
if(lp == B && lq == B) {
NewLabel++;
lx = NewLabel;}
else if((lp != lq)&&(lp != B)&&(lq != B)){
// REGISTER EQUIVALENCE (lp,lq)
lx = lq;}
else if(lq != B) lx = lq;
else if(lp != B) lx = lp;
I[i,j] = lx;} }
// FIND EQUIVALENCE CLASSES
// SECOND SCAN
```

Fig. 2: The classical algorithm

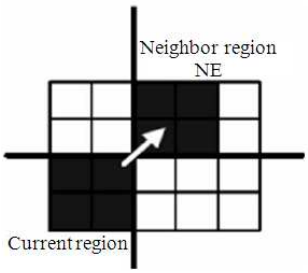

Fig. 3: Merge of the last pixel in the first row of the current region and the last pixel in the first column of the neighbor region on the north-east side

To reduce the processing time (Rosenfeld and Pfaltz, 1966; Jung-Me *et al*., 2000) reduces the size of the equivalent matrix array of the Rosenfeld and Pfaltz (1966) method. The enhancement starts by dividing the original image into N×N small regions and applying the Rosenfeld and Pfaltz (1966) algorithm to each region independently to generate local labels for each region. After that the regions will be connected with the region neighbors by resolving the region boundary. Jung-Me *et al*. (2000) uses N×N pointer array Label_List[i] as the array that maintain all the labels in the image. Label_List[i] points to the array for Region[i] where each array element is a global label within the entire image and the index for each array element is a local label within Region[i]. Jung-Me *et al*. (2000) shows that his algorithm is faster than Resenfeld and Pfaltz (1966) algorithm. There are three conditions to resolve label equivalences on region boundaries when merging regions with its neighbor regions in Jung-Me *et al*. (2000) method:

- Merge pixels values of neighboring regions with the first pixel of the current region
- Merge pixels in the first column of the current region and last column of the neighbor region on the west direction
- Merge the pixels in first row of the current region and the last row of the neighbor region on north direction of the current region. But according to this algorithm, an object similar to the Fig. 3 isn't recognized as belonging to a same single region

According to (Yapa and Harada, 2008), the forward and backward algorithm repeats passes through a binary image b(x,y) in the forward and backward raster directions alternatively. For example (Yapa and Harada, 2008), a binary image b(x,y) consists of pixel values F_o as the objects and F_B as the background and label m as the provisional label. In this method there are two scan masks used, firstly the forward scan mask as shown in Fig. 1a and secondly is the backward scan mask as shown in Fig. 1b. The first process is the forward scanning that uses the forward mask (Fig. 1a). In each step of scanning the mask will perform its local process which is scanning its four connected component in a, b, c and d position (Fig. 1):

$$g(x,y)=\begin{cases} F_B & \text{if } b(x,y)=F_B \\ m,(m=m+1) & \text{if}\forall\{i,j\in M_S\}g(x-i,y-j)F_B \\ g_{min}(x,y) & \text{otherwise} \end{cases}$$

$$g_{min}(w,y)=\min[\{g(x-i,y-j)\,|\,i,j\in M_S\}]$$

Where:

(m = m+1) = An increment of m
min(.) = Operator calculating the minimum value
MS = The region of the mask except the object pixel, i.e., b(x-1,y-1), b(x,y-1), b(x+1,y-1) and b(x-1,y) (Yapa and Harada, 2008)

The second process is the backward scanning that uses the forward mask (Fig. 1b). It is the same as the forward scanning, in each step of the scanning the mask will perform its local process which is scanning its four connected component in a, b, c and d position (Fig. 1b):

$$g(x,y)=\begin{cases} F_B & \text{if } b(x,y)=F_B \\ \min[\{g(x-i,y-j)\,|\,i,j\in M_S\}] & \text{otherwise} \end{cases}$$

The forward and backward scans are repeated alternatively until no provisional labels change and then the final labeled image can be obtained by assigning unique labels for each connected region (Yapa and Harada, 2008).

An improvement of the two scanning method was also proposed by Di Stefano and Bulgarelli (1999), they described a two-scan algorithm for labeling connected components in binary images in raster format. Unlike the classical two-scan approach, their algorithm processes equivalences during the first scan by merging equivalence classes as soon as a new equivalence is found as shown in Fig. 5. They show that algorithm significantly improves the efficiency of the labeling process with respect to the classical approach. The data structure used to support the handling of equivalences is a 1Darray. This renders the more frequent operation of finding class identifiers very fast, while the less-frequent class merging operation has a relatively high computational cost.

```
// C has been initialised ( C[i]=i );
// FIRST SCAN
for(i=1; i<NROWS-1; i++)
for(j=1; j<NCOLS-1; j++){
if (I[i,j] == F){
lp = I[i-1,j];
lq = I[i,j-1];
if(lp == B && lq == B){
NewLabel++;
lx = NewLabel;}
else if((C[lp]!=C[lq])&&(lp != B)&&(lq != B)){
for(k=0; k<=Newlabel; k++)
if (C[k] == C[lp]) C[k]=C[lq];
lx = lq;}
else if(lq != B) lx = lq;
else if(lp != B) lx = lp;
I[i,j] = lx;}}
// SECOND SCAN
for(i=1; i<NROWS-1; i++)
for(j=1; j<NCOLS-1; j++)
if (I[i,j] != B) I[i,j]=C[I[i,j]];
```

Fig. 4: Di Stefano and Bulgarelli (1999) algorithm

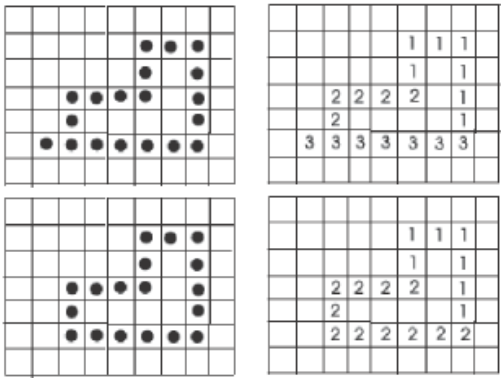

Fig. 5: Binary and labeling shapes

In their labeling algorithm equivalences are processed directly in the first scan so that equivalence classes are always maintain updated during the scan. This is obtained by associating a new equivalence class with each new label and by merging the corresponding classes as soon as a new equivalence is found. They perform the merging operation using a simple data structure called Class Array (C). C is a one dimensional array as large as the maximum label value and containing for each label value its corresponding class identifier (i.e., C[i] is the equivalence class associated with label i). C is initialized by posing C[i] = i; i = 0<maxlabel; this means that, initially, each possible label is assumed to belong to a distinct class. It is used to reserve one entry in C to express in the class identifiers domain the relation among pixels implied by the label value used to mark foreground pixels. This is done by setting C[B] = lB, i.e., typically C[B] = B. When two labels, li and lj, are found to be equivalent during the first scan, the corresponding classes, C[li] and C[lj], are merged. The merging consists in first setting one of two class identifiers to be the survivor and the other to be deleted and then storing the survivor identifier into the entries in the Class Array equal to the deleted one. The choice of the survivor can be arbitrary (for example: Always retain C[li], always retain C[lj], retain min(C[li]; C[lj])...).

Keeping equivalence classes in their correct, updated state during the first scan allows the check for a new equivalence to be carried out in the class domain rather than in the label domain. More precisely, when a conflict between li and lj is found, the algorithm checks whether C[li] and C[lj] are different or not and then handle the equivalence (i.e., merge the two classes) only in the former case.

After completion of the first scan the Class Array holds the class identifier associated with each temporary label and thus can be used as a look-up table in the second scan to change label values into their corresponding class identifiers. Figure 4 shows the C code of the Di Stefano and Bulgarelli (1999) algorithm in the 4-connectivity case.

The merging operation in the algorithm is used to automatically note all the equivalences between the merged classes members which are implied through the transitive property. The algorithm will not handle the new equivalences that are already found. It has been exploited in the first pass.

Samet and Tamminen (1986) and Dillencourt *et al*. (1992) has also presented ideas of merging equivalence classes and performing the check for conflicts on class identifiers. They propose a general labeling algorithm capable of handling in a unified way a wide range of image representation schemes (such as two-dimensional arrays, run lengths, bintrees, quadtrees...) and making use of a tree structure based on father links to represent each equivalence class.

Suzuki *et al*. (2003) has seen some weaknesses in this two scanning algorithm especially for the time consumed if the image has complex form and also large pixel data. He proposed an improvement by adding one dimensional table called label connection table to memorize label equivalences. It improves the algorithms' performance. This algorithm propagates provisional labels not only on the image but also on the label connection table. The table can reduce the number of forward and backward scans to complete the labeling by reflecting the connectivity of provisional

labels at a geometrical distance, in the connection table.

In the Suzuki *et al*. (2003) algorithm, he determines the provisional label for each pixel at the position "e" as follows:

$$g(x,y)=\begin{cases} F_B & \text{if } b(x,y)=F_B \\ m,(m=m+1) & \text{if}\forall\{i,j\in M_S\}g(x-i,y-j)F_B \\ T_{min}(x,y) & \text{oterwise} \end{cases}$$

$$T_{min}(x,y)=\min[\{T[g(x-i,y-j)]\Big|i,j\in M_S\}]$$

Then the connection label table always updated with the provisional label assignment with:

$$\begin{cases} \text{non}-\text{operational} & \text{if } b(x,y)=F_B \\ T[m]=m & \text{if}\forall\{i,j\in M_S\}g(x-i,y-j)F_B \\ T[g(x-i,y-j)]=T_{min}(x,y) & \text{if } g(x-i,y-j)\neq F_B \end{cases}$$

Suzuki *et al*. (2003) algorithm successfully improves the conventional forward and backward algorithm. It can reduce the process time in the conventional forward and backward algorithm. But Wu *et al*. (2005) still see an opportunity to improve this Suzuki *et al*. (2003) forward and backward algorithm by reducing the number of neighbors examined during the scanning steps and reducing the cost of union find algorithm by array based equivalence matrix rather than pointer based rooted trees.

Wu *et al*. (2005) focuses on the neighbors pixels of the image which are correlated. In his theory each pixels in an image which are belong to one or many object would be related to each other. With a suitable data structures it can scan less than four. From this strategy, he proposed an algorithm to examine neighbor pixels based on a decision tree and same as Suzuki's algorithm, Wu *et al*. (2005) also store the label equivalence information.

Conclusively, the two major concerns in the connected component labeling algorithm are the time consumed and the complexity of the algorithm. In order to obtain a good speed for the labeling process, hardware upgrade had also been attempted by the researchers; however, an enhancement on the algorithm itself is much more desired, which is exactly what this study is presenting.

MATERIALS AND METHODS

The target of this approach is to make the process run faster without any failures in labeling the components. The approach constructed must be simple in order to make the computational process steps shorter. The approach has to avoid the complexity of the conditional statements process in the algorithms. This approach is constructed without modifying the hardware parts, so it will focus on the algorithm parts.

The first step of this algorithm is started with identifying the first pixel p_0 (x, y) of the component. This first pixel henceforth in this study will be called as the head of the component. There are some ways to get this components' head. It could be by using the heads' rule which is if the pixel component are scanned with the heads' rules from the certain direction it does not have any single neighbors ($p_1 + p_2 + p_3 + p_4 = 0$), or it could be using a common scanning that find any common pixels in the scanning which are not belong to any group label (p_0 (x, y) = -1). -1 is given for the components that are not included in any group. In this approach there are two kind of propagation used which are the one which is propagating on the image pixels and the one which is propagating on array tables H[j, i] and C[j, i]. The head table is needed if the head's components table is given from the previous phase. This head's component table H[j, i] is useful for speeding up the process, because it can help the scanning process to go directly to the head of the components. It can also reduce the number of scanning processes. It is the same as the (Di Stefano and Bulgarelli, 1999; Jung-Me *et al*., 2000; Yapa and Harada, 2008; Suzuki *et al*., 2003; Wu *et al*., 2005; Samet and Tamminen, 1986), the process of labeling in this algorithm is not propagating only on the connected components but also on the array table. The difference is on the array's dimension. Di Stefano and Bulgarelli (1999), only one dimensional array is used while this algorithm uses two dimensional arrays. It is good to use two-dimensional arrays, because it can also capture other information about the pixels such as the position and the size of the object, so it can ease the next image-processing process. The two dimensional array of head H[j, i] and the component C[j, i] will be explained in a one-dimensional array H[i] and C[i]. It is aimed to give a clear understanding of this algorithm's processes.

This algorithm is different than the current method that uses double scanning forward and backward (He *et al*., 2007), in this algorithm only the forward scanning is used. In the current method, mostly the main scanning mask will use the mask that is used by (Rosenfeld and Pfaltz, 1966). In this algorithm, the scanning mask in (Rosenfeld and Pfaltz, 1966) is only used in the pre-labeling process if using the component's head table. But if the component's head table is not used, then the scanning mask

(Rosenfeld and Pfaltz, 1966) would not be used. The main scanning in this algorithm is implementing the 8-connected component Components Neighbors-Scan. On each pixel C[i], it will scan its neighbors in clockwise direction. The component that are connected to the main component scanned C[i], can be defined using distance measurement such as the Euclidian distance, or just by justifying the closest pixel. For example, in the 8-connected component are the pixels that are located beside the main component as shown in Fig. 6.

So the closest connected component in 8-connected component (in raster image coordinate) will be:

$P_1 = p_0(x-1, y)$
$P_2 = p_0(x-1, y-1)$
$P_3 = p_0(x, y-1)$
$P_4 = p_0(x+1, y-1)$
$P_5 = p_0(x+1, y)$
$P_6 = p_0(x+1, y+1)$
$P_7 = p_0(x, y+1)$
$P_8 = p_0(x-1, y+1)$

The variable p_n is indicating the pixel position while the x and y are the coordinate of the pixel. Referring to the pixels position in graphic coordinate, the pixel p_1 position is on the West of the main pixel p_0 where the coordinate will be x-1 while y is still the same as the main pixel position, pixel p_2 is on the North West of the main pixel p_0 with the x-1 and y-1, pixel p_3 is on the North of the main pixel p_0 where the coordinate x the same as the p_0 position while y-1, pixel p_4 is on the North East of the main pixel p_0 where x+1 and y-1, pixel p_5 is on the East of the main pixel p_0 where x+1 and y has the same position with p_0, pixel p_6 is on the South East of the main pixel p_0 with x+1 and y+1, pixel p_7 is on the South of the main pixel p_0 where the coordinate x the same as the main pixel while y+1 and the last pixel p_8 is on the South West of the main pixel p_0 where the coordinate will be x-1 and y+1. These positions are very important to be identified for the next neighbor scanning process.

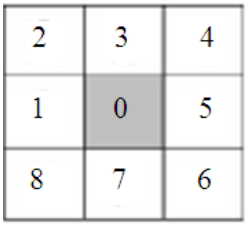

Fig. 6: 8-connected component

After the head of the component H[i] data is identified, the pixels are traced from the first heads array H[i = 0] to the last heads array H[i = n]. The heads will be the starting point of the Components Neighbors-Scan. The component heads will be processed if the head is not with any component labeled (-1), otherwise the process will skip to the next heads.

The pixel head H[i] will scan its neighbors to find the connected pixels which are the in p_1, p_2, p_3, p_4, p_5, p_6, p_7 and p_8.

Each neighbors found are stored in a component array C[i] (Fig. 7). After it finishes to scan the neighbors, the next process is to scan the C[i+n] neighbors to find the other neighbors of the current pixel C[i].

Similar as the head H[i], the components also scan its neighbors. In this stage the process does not include the head of the component, as it is already taken by the components (Fig. 8). Now the components scan their neighbors without the heads. The process will be stopped until it cannot find any available components that are not already join the other groups.

The algorithms constructed for this approach are also simple. It consists of two main neighbors scanning for the head and the components. The scanning is run on the array and on the image.

Component
C_1
C_2
C_3
C_4
.
.
.
C_n

Fig. 7: Component's table

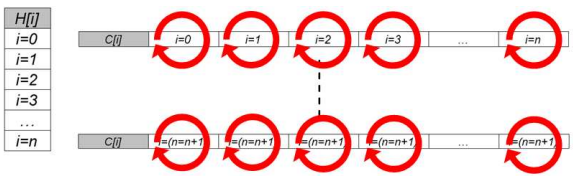

Fig. 8: Neighbors scanning

RESULTS

An implementation and experiment has been conducted using data of simple images and natural images to see the performance. The natural images used in this experiment are the standard images downloaded from Volume 3 (Miscellaneous) of the image data base of University of Southern California (USC, database link: http://sipi.usc.edu/services/database/database.cgi?volume=misc).

To analyze the accuracy of the presented approach, a simple shape of object in an image is used. It is aimed to give a clear view of the algorithm's step. Suppose there is a shape of object as Fig. 9. The first process starts from pixel 1 (red arrow) which has not join any other groups or -1. After identifying that the pixel is still in the free status or is not in any group, the system will find the pixel (blue arrow) that is likely to be the member of the head's components. This process is iterated continuously to the last pixel. From this simple shape of object experiments, it can be seen that the algorithm constructed give a good result without any errors. It proves that the algorithm of this connected component labeling algorithm works properly and gives results as expected.

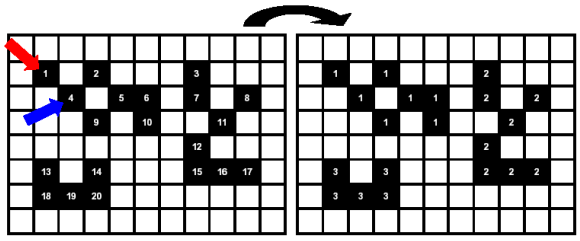

Fig. 9: Binary and component labeling

```
For i := 0 to n do
Begin
If  (H[i]=-1) then
Begin
Scan H[i] neighbors;
LabelingComponent;
Store C[j];
For j := 0 to n do
Begin
Scan C[j] neighbors;
LabelingComponent;
Store C[j];
End;
End;
End;
```

Fig. 10: Neighbors scan algorithm

With the algorithm working as expected, it is next tested with more various and bigger size of image to see its accuracy in handling data. Firstly the testing is started with numbers of simple shapes (Fig. 9-12). In this test, the algorithm gives results without any errors and failures. It successfully groups the objects with its clusters numbers. The speed of this process are also satisfactory.

Now the algorithm is given a more natural (real) data image (Fig. 13). In these data, the shapes of the component are more complex and it has numbers of groups.

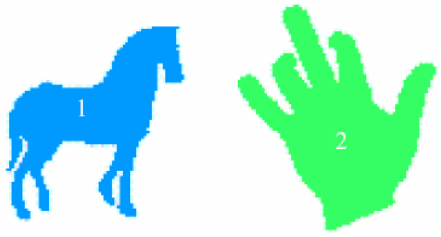

Fig. 11: Simple shape image

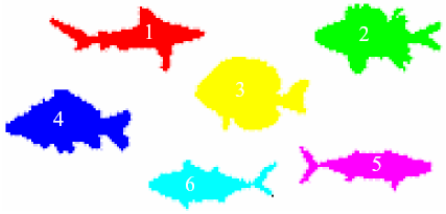

Fig. 12: Image with numbers of shapes

Fig. 13: Natural images sample

Fig. 14: Labeling results

In this test, the algorithm also gives a good result without any errors and failures. It successfully groups the objects. The speeds of process needed are also relatively fast (Fig. 14).

DISCUSSION

In discussion, the performance of the presented approach is compared with other available method that uses well-known two time scanning method (Di Stefano and Bulgarelli, 1999). The accuracy of the presented approach and the other methods is significantly accurate, according to their tests. All of the methods give a good result in accuracy issues.

Table 1: Speed test result

			Time (m sec)	
Nunber	Sample	Size	CNS	TTS
1	4.1.01	256×256	0	11
2	4.1.02	256×256	0	11
3	4.1.03	256×256	0	12
4	4.1.04	256×256	0	11
5	4.1.05	256×256	0	11
6	4.1.06	256×256	0	11
7	4.1.07	256×256	0	12
8	4.1.08	256×256	0	11
9	4.2.01	512×512	0	40
10	4.2.02	512×512	31	47
11	4.2.03	512×512	15	55
12	4.2.04	512×512	15	46
13	4.2.05	512×512	31	46
14	4.2.06	512×512	15	45
15	4.2.07	512×512	15	44
16	5.1.09	256×256	0	13
17	5.1.10	256×256	0	13
18	5.1.11	256×256	0	12
19	5.1.12	256×256	0	12
20	5.1.13	256×256	0	12
21	5.1.14	256×256	0	13
22	5.2.08	512×512	31	47
23	5.2.09	512×512	31	49
24	5.2.10	512×512	16	46
25	5.3.01	1024×1024	94	172
26	5.3.02	1024×1024	46	204
27	7.1.01	512×512	31	48
28	7.1.02	512×512	31	48
29	7.1.03	512×512	31	48
30	7.1.04	512×512	31	47
31	7.1.05	512×512	15	54
32	7.1.06	512×512	15	53
33	7.1.07	512×512	31	63
34	7.1.08	512×512	31	47
35	7.1.09	512×512	31	46
36	7.1.10	512×512	31	50
37	7.2.01	1024×1024	15	144
38	Boat	512×512	31	47
39	Elaine	512×512	15	46
40	Grey21	512×512	15	42
41	House	512×512	31	49
42	Numbers	512×512	15	46
43	Ruler	512×512	31	47
44	Testpat	1024×1024	109	168

The difference between this approach and the methods are in the steps of the algorithm and the time consumed for the process. For this experiment the algorithm was constructed to be as simple as possible. It is aimed to meet the connected component labeling requirements.

As it can be seen in the Fig. 10, the algorithm of this approach is relatively simple compare to the Fig. 2 and 4 algorithm. In the term of time consumed, the experiment result shows that the approach has a good speed of process. It is considerably fast compare to the other methods. It can be seen from the test result in Table 1.

From the natural image testing, the result shows that the presented approach is slightly faster than the two times scan method. Averagely, from the Table 1 result the algorithm increases the speed around 67.4% from the two times scanning method.

CONCLUSION

An approach of Components Neighbors-Scan connected component labeling technique in image processing is presented. The approach promotes speed, accuracy and simplicity.

An implementation and experiment has been done to measure its performance. In the experimental results, it shows that the approach has a good performance in terms of accuracy, the time consumed and the simplicity of the algorithm.

Future study: There are still some techniques for improving and implementing this connected component labeling approach.

For example for the speed of process, it can be enhanced with parallel processing by using multiple processors or it could use a multi-thread programming technique.

ACKNOWLEDGMENT

This research was financially supported by the Ministry Of Science Technology and Innovation (MOSTI) and Research Management Center (RMC) UTM using vote 79326, Malaysia.

REFERENCES

Dillencourt, M.B., H. Samet and M. Tamminen, 1992. A general approach to connected component labeling for arbitrary image representations. J. ACM, 39: 253-280. DOI: 10.1145/128749.128750

Di Stefano, L. and A. Bulgarelli, 1999. A simple and efficient connected components labeling algorithm. Proceeding of the International Conference on Image Analysis and Processing, Sept. 27-29, IEEE Computer Society, Washington DC., USA., pp: 322-327. http://portal.acm.org/citation.cfm?id=840794

Falcao, A.X., P.A.V. Miranda, A. Rocha and F.P.G. Bergo, 2005. Object detection by-connected seed competition. Proceedings of the 18th Brazilian Symposium on Computer Graphics and Image Processing, Oct. 9-12 IEEE Computer Society, USA., pp: 97-104. DOI: 10.1109/SIBGRAPI.2005.34

Gonzalez, R.C. and R.E. Woods, 1992. Digital Image Processing. 3rd Edn., Addison-Wesley, Reading, MA., ISBN: 13: 9780201508031, pp: 730.

Haralick, R.M. and L.G. Shapiro, 1991. Computer and Robot Vision. 1st Edn., Addison-Wesley, Reading, MA., ISBN: 13: 9780201108774, pp: 672.

He, L., Y. Chao and K. Suzuki, 2007. A linear-time two-scan labeling algorithm. Proceedings of the IEEE International Conference on Image Processing, Sept. 16-Oct. 19, IEEE Computer Society, San Antonio, TX., pp: 241-244. DOI: 10.1109/ICIP.2007.4379810

Jain, R., R. Kasturi and B.G. Schunck, 1995. Machine Vision. 1st Edn., McGraw-Hill, USA., ISBN: 0070320187, pp: 549.

Jonas, G., L. Velho and S. Levy 1997. Image Processing for Computer Graphic. 1st Edn., Springer, USA., ISBN: 10: 0387948546, pp: 352.

Jung-Me, P., C.G. Looney and C. Hui-Chuan, 2000. Fast connected component labeling algorithm using a divide and conquer technique. Proceeding of the Conference on Computers and their Applications, Mar. 29-31, ISCA, New Orleans, USA., pp: 373-376. http://www.pubzone.org/dblp/conf/cata/ParkLC00

Klette, R. and P. Zamperoni, 1996. Hand Book of Image Processing Operators. 1st Edn., Wiley, New York, ISBN: 10: 0471956422, pp: 416.

Lee, D.R., S.H. Jin, P.C. Thien and J.W. Jeon, 2007. FPGA based connected component labeling. Proceeding of the International Conference on Control, Automation and System, Oct. 17-20, IEEE Computer Society, Seoul, pp: 2313-2317. DOI: 10.1109/ICCAS.2007.4406746

Rosenfeld, A. and A.C. Kak, 1982. Digital Picture Processing. 2nd Edn., Academic Press, New York, ISBN: 10: 0125973020, pp: 349.

Rosenfeld, A. and J.L. Pfaltz, 1966. Sequential operations in digital processing. J. ACM, 13: 471-494. DOI: 10.1145/321356.321357

Samet, H. and M. Tamminen, 1986. An improved approach to connected component labeling of images. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June 1986, IEEE Computer Society, USA., pp: 312-318. http://www.cs.umd.edu/~hjs/pubs/SametCVPR86c.pdf

Sossa, H. and G. Guzman, 2000. New method to count objects into an image. Proceeding of the 15th International Conference on Pattern Recognition, Sept. 3-8, IEEE Computer Society, Washington DC, USA., pp: 470-472. DOI: 10.1109/ICPR.2000.905378

Suzuki, K., H. Isao and S. Noboru, 2003. Linear-time connected-component labeling based on sequencial local operations. Comput. Vis. Image Understand., 89: 1-23. DOI: 10.1016/S1077-3142(02)00030-9

Wu, K., O. Ekow and S. Arie, 2005. Optimizing connected component labeling algorithms. Proc. SPIE, 5747: 1965-1976. DOI: 10.1117/12.596105

Yang, Y. and D. Zhang, 2003. A novel line scan clustering algorithm for identifying connected components in digital images. Image Vis. Comput., 21: 459-472. DOI: 10.1016/S0262-8856(03)00015-5

Yapa, R.D. and K. Harada, 2008. Connected component labeling algorithms for gray-scale images and evaluation of performance using digital mammograms. Int. J. Comput. Sci. Network Secur., 8: 33-41.