An efficient multiple pass pixel connectivity labeling method for object detection

Faisel G. Mohammed

Abstract— In this paper object detection (OD) model within the image was introduced. The simultaneously labels connected components (to be referred to as merely as components in the recent research work) and their contours in binary images was presented. The main step of this method is to use a contour tracing technique to detect the external contour and possible internal contours of each component, and also to identify and label the interior area of each component. Labeling is done in a single pass over the image, while contour points are revisited more than once, but no more than a constant number of times. Moreover, no re-labeling is required throughout the entire process, as it is required by other algorithms. Experimentation on various types of images (characters, half-tone pictures, photographs, newspaper, etc.) shows that suggested method outperforms methods that use the equivalence technique. Current algorithm not only labels components but also extracts component contours and sequential orders of contour points, which can be useful for many applications.

Index Terms— Object detection, counter tracing, image segmentation, and binary image representation.

———————————— ◆ ————————————

1 Introduction

Object recognition is an essential part of computer vision for improving machine intelligence or interacting with physical objects.

Various labeling algorithms have been proposed and a survey of various algorithms can be found in [1].A common property for all these algorithms are that they are memory access intense. Furthermore, all algorithms have to handle the same obstacle, i.e. label collisions. In a binary image, a typical label collision occurs when a u shaped object is encountered. Since an image typically is scanned from top to bottom and from left to right, it is not possible to know that the two pillars in the u is part of the same object until the bottom part of the u is encountered. [2].

Researchers often face the need to detect and classify objects in images. Technically, image objects are formed out of components that in turn are made of connected pixels. It is thus most equitable to first detect components from images. When objects have been successfully extracted from their backgrounds, they also need to be specifically identified. For the latter purpose, component contour is often a useful resource for identifying objects. There are methods that identify objects from either chain codes or Fourier descriptors, which are derived from object contours. There are also methods that match object contours against certain stochastic models. These methods demonstrate that both component and contour labeling is an effective method for detecting and identifying two-dimensional objects [3].

Also, one of the most fundamental operations in pattern recognition is the labeling of connected components in a binary image. The labeling algorithm transforms a binary image into a symbolic image in order that each connected component is assigned a unique label [4].

———————————————

Faisel G. Mohammed is currently assistant prof. has PhD degree in astronomy in Baghdad University, Iraq, PH-+9647903317690. E-mail: faiselgm73@gmail.com

2. Labeling connectivity algorithm

The standard labeling algorithm will compute the weakly connected components for a tree-loop graph. The algorithm uses a strategy presented in [5]. Initially, array ***F*** is the input tree-loop graph with nodes ***i*** and edges (***i, F[i]***), ***1<i< n***. When the algorithm finishes, each input tree-loop has been transformed into a star. The root of the star is the smallest node in the cycle of the input tree-loop.

```
CONNECTED-TREELOOP
begin
  for all i:= to n do in parallel
    begin
      S[i] := F[i];
      for k := 1 to log n do4
        begin
          if F[S[I']] < F[i] then F[i] := F[S[I']];
         en;[i] := S[S[i]];
         for m := 1 to logn do
          F[i] := F[F[i]];
    end
end
```

In this paper, we assume the results of division and logarithm are integers, as this will not affect the correctness of our algorithms [4].

For more explanation of the above algorithm see [4].

There is other faster method to do this connectivity labeling. In the next section the demonstration of proposed method is an efficient form of the above algorithm and faster has fully implementation and low complexity.

3. Proposed OD description

The proposed method labels each component using fast and low complexity contour tracing technique. This method is based on the principle that a component is fully determined by its contours. This method also provides a procedure for

ISSN 2229-5518

finding all component pixels. An image will scan in the same way as it would be encountered by a scanner, i.e., from top to bottom and from left to right per each line.

When an external or internal contour is encountered, then the contour-tracing procedure will use to complete the contour and assign a label, say L, to all pixels on the contour. When the contour is traced back to its starting point, scanning will resume at that point. Later on, when the contour pixels labeled L are visited again, then the same label L will assign to black pixels that lie next to them.

This method is applicable in areas in which must detect components and also classify them by means of certain contour features.

Document analysis and recognition (DAR) in particular is an area for which our method is beneficial. High-order objects, such as half-tone pictures, characters, textlines, and text regions, need to be classified in order to effectively perform DAR.

Components are the basic ingredients of all high-order objects. Labeling components is therefore a commonly used technique for extracting high-order objects. The objective of DAR is not simply to extract high-order objects, but to recognize individual characters found within textual areas. There are many methods that employ certain contour features for classifying characters [1].

The next pseudo codes show in details the implementation of proposed algorithm. The essential remarkable was showed before each ambiguous step. You can directly use this and implement in any programming language and you can email me on the next email to get the full source code:

Email: faiselgm73@gmail.com

Start algorithm:

Set label (detected object) as matrix have dimensions tmpwidth & tmpheight

```
Set rep as no of iteration - 1
amount := 1000
begin
for j := 0 to rep
 n := 1
 fwdscan: (forwarded scan)
 for x := 1 To tmpwidth – 2 do begin
  for y := 1 To tmpheight – 2 do begin
    mask[0] := label[x - 1, y – 1]
    mask[1] := label[x, y – 1]
    mask[2] := label[x + 1, y – 1]
    mask[3] := label[x - 1, y]
    mask[4] := label[x, y]
    if imgBW[x, y] := 1 then
     temp:=mask[0] or mask[1] or mask[2] or mask[3]
     if temp := 0 then begin
        label(x, y) := n; bscan = 1;  n := + 1
      else
        min := mask[0]
        for i := 1 to 4 do begin
            if min := 0 then min = mask[i]; cont
          if mask[i] < min and mask[i] <> 0 then
          min = mask(i); cont;
          end
          label[x, y] := min ; bscan := 1
        end
    end
 end
end
backscan:
for x := 1 To tmpwidth – 2 do begin
  for y := 1 To tmpheight – 2 do begin
     if label((tmpwidth - 1) - x, (tmpheight - 1) - y)
        <> 0 then begin
       mask[0] = label[[tmpwidth – 1] – [x –
                  1], [tmpheight – 1] – [y – 1]]
       mask[1] = label[[tmpwidth – 1] - x,
                   [tmpheight – 1] – [y – 1]]
       mask[2] = label[[tmpwidth – 1] – [x + 1],
                   [tmpheight – 1] – [y – 1]]
       mask[3] = label[[tmpwidth – 1] – [x – 1],
                   [tmpheight – 1] – y]
       mask[4] = label[[tmpwidth – 1] - x, [tmpheight
                   - 1] – y]
       min = mask[0]
       for i := 1 To 4 do begin
          If min = 0 Then min = mask(i): GoTo cont2
   IIf mask(i) < min And mask(i) <> 0 Then min = mask[i]
         cont2:
       end
 label((tmpwidth- 1)- x, (tmpheight- 1)- y)=  min
   end
  end
end
end
finish:
```

set how many of label number 'no' has been got

```
Set count(amount, 4)
for y := 0 To tmpheight – 1 do begin
  for x := 0 To tmpwidth – 1 do begin
   count[label[x, y], 0] := count[label(x, y), 0) + 1
Set all the min & max coordinate of x and y to min 1st..
   if count[label[x, y], 1] := 0 Then do begin
          count[label[x, y], 1] := x
          count[label[x, y], 2] := x
          count(label(x, y), 3) = y
          count(label(x, y), 4) = y
    end
```

update each coordinate

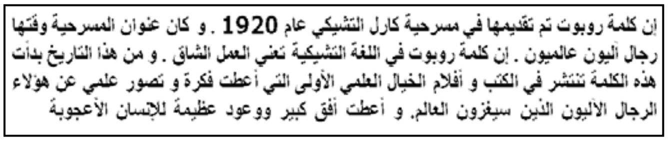
update coordinate x min (if x < than the previous value)

```
   if x < count[label[x, y], 1] and count[label[x, y], 1) <>
         0 then count(label[x, y], 1) = x
   if x > count[label[x, y], 2] and count[label[x, y], 2] <>
         0 then count[label[x, y], 2] = x
   if y < count[label[x, y], 3] and count[label[x, y],
         3] <> 0 then count[label[x, y], 3] = y
   if y > count[label[x, y], 4] and count[label[x, y], 4]
          <> 0 then count[label[x, y], 4] = y
   end
end
```

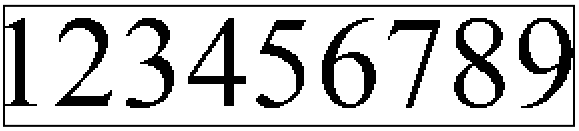
IJSER

4. Experiments

Activities of proposed method are tested with the six types of test images shown in fig.1. The test environment is an Intel Pentium5 1.7 GHz personal computer with 1G SDRAM. Figure2 show the resulted labeling objects within tested images.

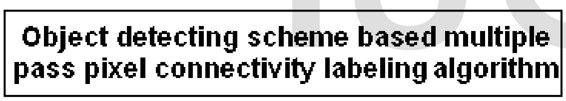


(a) Arabic (size=993×147 pixel)



(b)Numerals (size=288×65 pixel)

(c)License Palette (size=188×119 pixel)



(d)(size=288×65 pixel)

Fig 1. Original tested images

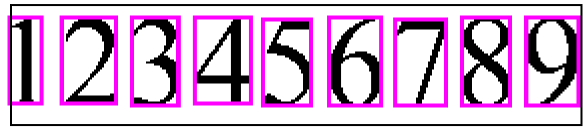
5. Conclusion

An efficient component-labeling algorithm that employs contour tracing technique was presented. This method scans a binary image only once and traces each contour pixel no more than a constant number of times. It is thus computationally effective (time consuming) in labeling connected components and also finding all contours and sequential orders of contour pixels. In experiments six types of images of various sizes were used as attested images.

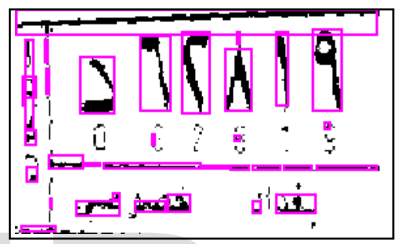
The results show that suggested method outperforms all of them in terms of computational speed.

Our method has the following advantages. First, it requires only one pass over the image. Contour points are visited more than once due to the aforementioned contour tracing procedure, but no more than a constant number of times. Second, it does not require any re-labeling mechanism. Once a labeling index is assigned to a pixel, its value is unchanged. Third, we obtain as by-products all contours and sequential orders of contour pixels. Fourth, experimental results show that low consuming time.

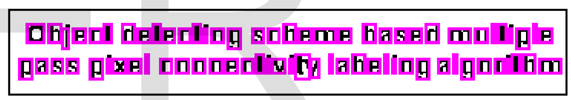
(a)Repetition=6, TC=1.007 sec



(b)Repetition=4, TC=0.168 sec.



(c)Repetition=7, TC=0.424 sec.



(d)Repetition=3, TC=0.275 sec.

Fig 2. Objected labeled results of the tested images showed in fig1, and TC referred to time consuming

References

[1] W. Kesheng, O. Ekow, and S. Arie, "*Optimizing connected components labeling algorithms*, "in SPIE Int. Symposium on Medical Imaging, San Diego, CA, USA, Feb 2005.

[2] K.Siva Nagi, Bhanu Murthy Bhaskara and B.R.Vikram, "*Modified Unsupervised Image Segmentation based on Gaussian Mixture Model for Traffic Surveillance Applications*". International Journal of Computer Applications (0975 – 8887) Vol. 44– No.3,P.p.41-48, April 2012.

[3] Fu Chang, , Chun-Jen Chen and Chi-Jen Lu, "*A linear-time component-labeling algorithm using contour tracing technique*", Elsevier, journal of Computer Vision and Image Understanding, Volume 93, Issue 2, Pp. 206-220, February 2004.

[4] Kenji Suzuki, Isao Horibaa and Noboru Sugieb, " *Linear-time connected-component labeling based on sequential local operations*", Elsevier, journal of Computer Vision and Image Understanding, Volume 89, Issue 1, Pp. 1-23, Jan. 2003.

[5] Yijie Han & Robert A. Wagner, "*An Efficient and Fast Parallel-Connected Component Algorithm*", Journal of the Association for Computing Machinery, Vol. 37, No. 3, pp. 626-642, July 1990.