The Fast Scheme for Document Page Segmentation in OCR using Window and Optimum Image

WICHIAN PREMCHAISWADI*, PHAISARN SUTHEEBANJARD*,
NUCHREE PREMCHAISWADI**
*Graduate School of Information Technology, Siam University
235 Petchkasem Road, Phasi-charoen, Bangkok 10163, Thailand
wichian@siam.edu
**Faculty of Information Technology, Dhurakij Pundit University
110/1-4 Prachachuen Road Laksi, Bangkok 10210, Thailand
nucharee@dpu.ac.th

Abstract: - This paper presents the speed-up method for document page segmentation which is one of the most important processes in an Optical Character Recognition (OCR) system. In this proposed scheme, a window size of 12 by 12 pixels is used to find a black pixel and its contour border. Then, the optimum image is created from these borders of characters where the 12x12 pixels of the original picture are represented by 1 pixel in the optimum image. Therefore, the number of pixels is reduced to 1/144 times the original image but still keeps the original image structure correctly. Finally, the optimum image is used for block extraction process to provide the faster work result. The experimental results show that the proposed scheme can significantly speed up the processing time of the document page segmentation process.

Key-Words: - Document Page Segmentation, Optimum Image, Window, OCR

1 Introduction

Document page segmentation is one of the most important processes using in OCR for identification areas in the image of a document page. There are many document page segmentation techniques have been proposed which can be classified as; top-down [1, 2], bottom-up [3] and mixed method [4, 5]. A contour edge following algorithm by using the 32x32 pixels window is used in [5] where the window is considered as located on the black pixels if at least 10 black pixels are in the window. This algorithm is faster with fewer overheads than algorithms that need to access all pixel of a document. However, if an area contains no black pixel, it must check every pixel in 32x32 pixels window or 1024 pixels.

In this paper, a mixed approach based on optimum image is proposed. This method consists of three steps as follows:

1. Finding black pixels in an image by using window of type "I" to reduce computation time.
2. Create the optimal picture of the original image.
3. Block extraction.

2 Finding Black Pixels

In order to reduce the computation time in finding the black pixels, many types and sizes of windows as shown in Fig.1 are tested to find the optimum window used in the proposed scheme.

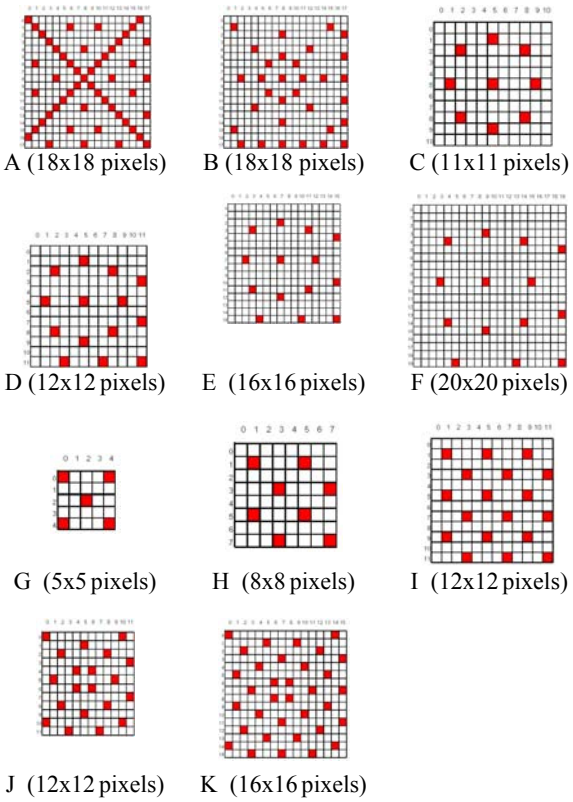


Fig. 1. Window type A to K

The characteristics of each window type are shown in Table 1.

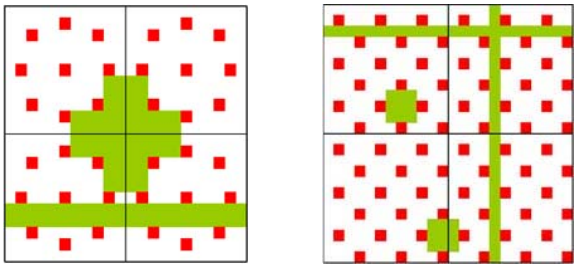
Table 1. Characteristics of each window type.

Type	Size	Scanned pixels	Percentage
A	18X18	52	16.049
B	18X18	32	9.877
C	11X11	9	7.438
D	12X12	14	9.72
E	16X16	14	5.469
F	20X20	14	3.5
G	5x5	5	20
H	8x8	8	12.5
I	12x12	18	12.5
J	12x12	20	13.889
K	16x16	34	13.281

The test results show that window of type A to F can not process correctly while the window of type G to K can process correctly. Therefore, window type "I" is selected for scanning in the proposed scheme.

Window type "I" is a window size of 12 by 12 pixels and only 18 points is used for checking black pixels, therefore it needs to check only 12.5% of the window.

The accuracy of window type "I" can be shown by comparing it with other window types such as window type "C" as shown in Fig. 2.



Window type "C" Window type "I"

Fig. 2. non checked points in window type "C" and "I"

Fig. 2 shows some of the non checked areas of these two windows. It can also see that the maximum size of the non checked area of window type "I" is only 3 by 3 pixels. The comparison is shown in Table 2.

Table 2. Compare non checked points in window type "C" and "I"

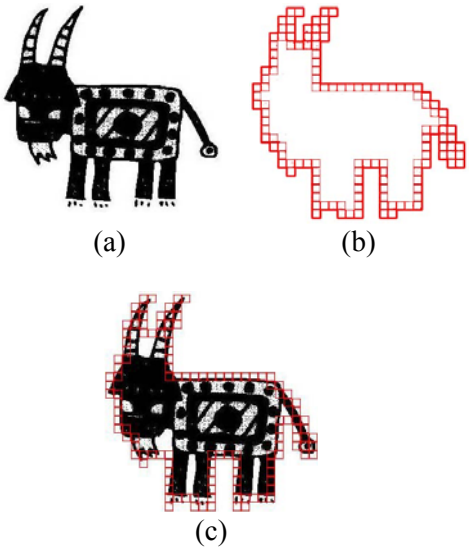
Window Type	Non Checked points in window				
	1px	2px	3px	4px	5px
C	1x∞	2x∞	3x10	4x10	-
I	1x∞	2x3	3x3	-	-

Fig. 2 and Table 2 show that a window type "I" can detect black pixels better than that of window type "C".

2.1 Identify Text and Pictures

This process is used for identify text and picture areas of the image. It starts with finding black pixels with raster scan by window type "I" for the entire image. After the first black pixel is found, contour edge following with chain of 1 pixel is used to find the size of the block of these black pixels. In order to classify the block of black pixels into picture, character or noise, we use the criteria as the following:

1. If the width or height of the block of black pixels is more than threshold (in this research, we used 80 pixels), the block will be classified as a picture. Then, contour edge following with window type "I" is used to find perimeter of this picture as shown in Fig. 3.



(a) (b)

(c)

Fig. 3. (a) Original image, (b) Perimeter of picture, (c) Perimeter of picture on original image

2. If the block of black pixels is not a picture and its width or height is more than 4 pixels, this block is classified as a character. The area of each character is represented by using rectangle as shown in Fig. 4.

(a) (b)

Fig. 4. Characters are represented by rectangles (a) Original image, (b) Rectangle of characters

3. If the block of black pixels is not a picture and not a character, the block is classified as noise. Then the block is deleted.

3 Create Optimum Image

An optimum image is an image that is used for representing the original image by reducing its size to 1/144 times the original image. This method applied only to the area of characters. The window size of 12 by 12 pixels, window type "I", is used to scan the original image, and this window will be represented by 1 pixel in the optimum image. Therefore, the optimum image will have its width and height about 1/12 times the original image. The example of creating an optimum image is shown in Fig. 5.

(a) (b)

Figure5. (a) An original image (2400x3300 pixels) (b) An optimum image (200x275 pixels)

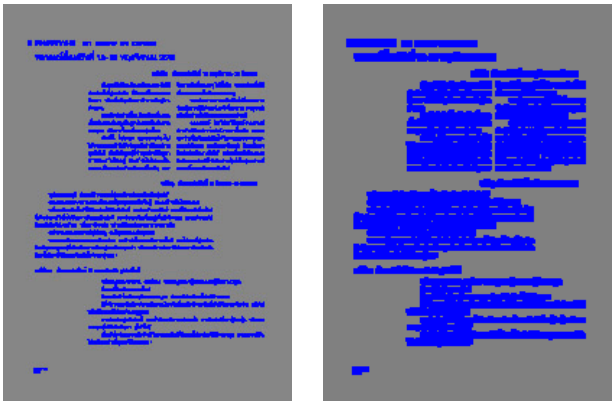
4 Block extraction in Optimum Image

The process consists of 2 steps as follows:

4.1 Identify Blocks

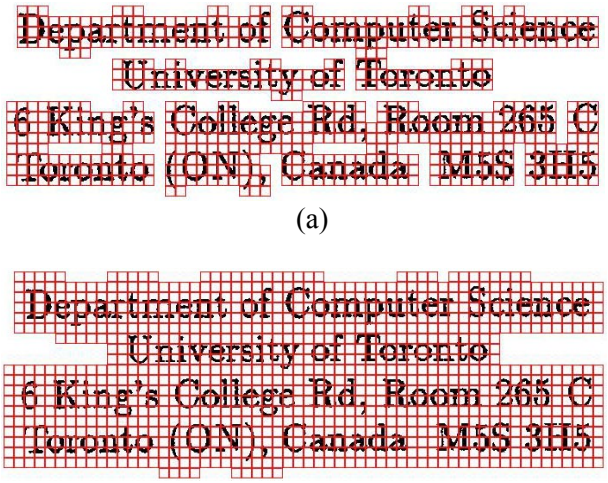
The dilation technique is used with a mask of size 3x3 pixels for connecting adjacent characters of the optimal image into the same block as shown in Fig. 6.

Each pixel in optimum image represent 12x12 pixels in the original image, so dilation the optimum image with a mask size of 3x3 pixels has effect like dilation of 36x36 pixels in the original image. The process can group characters into the same block as shown in Fig. 7.



(a) (b)

Fig. 6. (a) Optimum image, (b) Dilation of (a) with a mask size of 3x3 pixels.



(a)

(b)

Fig. 7. (a) Group of characters before dilation, (b) after dilation

After dilation of the optimum image with a mask size of 3x3 pixels, contour edge following with chain of 1 pixel is used to find perimeter of the block. To show the efficiency and the limitation of the use of window type "I", we compare it with the use of window size of 32 by 32 pixels. Fig. 8 shows that the window size of 32X32 pixels can not separate character having a distance less than 62 pixels.

For the propose scheme, many cases are tested with the window of type "I" and the results to show the limitation of the scheme are shown in Fig. 9.

(a) (b)

Fig. 8. (a) The 32x32 pixels window,
(b) minimum distance for this method, 62 pixels,

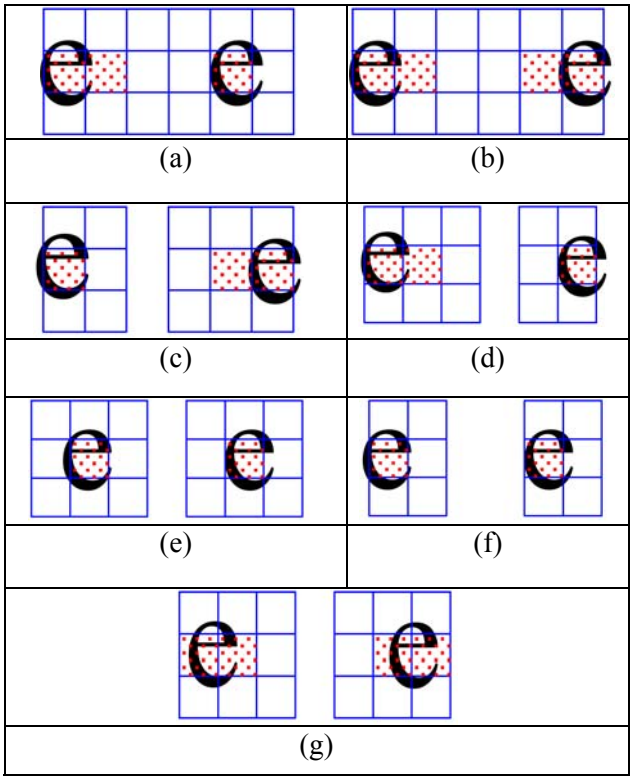


Fig. 9. The use of Windows Type "I"

The results are as the following.

(a) The space between blocks is 34 pixels, the block can not be separated.
(b) The space between blocks is 45 pixels, the blocks can not be separated.
(c) The space between blocks is 46 pixels, the blocks can be separated.
(d) The space between blocks is 46 pixels, the blocks can be separated.
(e) The space between blocks is 35 pixels, the blocks can be separated.
(f), (g) The space between blocks is 47 pixels, the block can be separated.

From the results, they can be divided into 3 categories as follows.

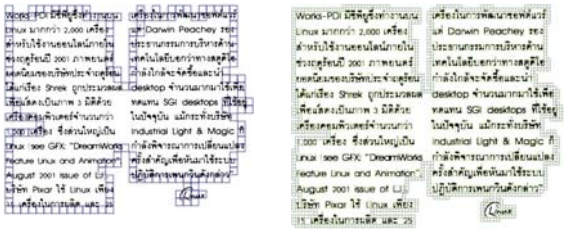
1. If the space between blocks is less than 35 pixels, the scheme can not separate blocks.
2. If the space between blocks is between 35 and 45 pixels, the scheme may or may not separate blocks. Fig. 9 (b) shows that the scheme can not separate block while in Fig. 9 (e) it can separate blocks.
3. If the space between blocks is more than 45 pixels, the scheme can separate blocks

Fig. 10 and 11 shows the example of applying the window size of 32x32 pixels and window type "I" on a real document.

(a) (b)

Fig. 10. (a) window size of 32X32 pixels can not separate blocks
(b) window type "I" can separate blocks

In Fig. 10, it shows that window type "I" can separate the blocks correctly while the used of window 32x32 can not separate the block correctly. For Fig. 11 both methods can not identify block correctly because these two blocks have too small space between blocks.



(a) (b)

Fig. 11. both method can not separate block
(a) window 32x32 (b) window type "I"

4.2 Separate Blocks

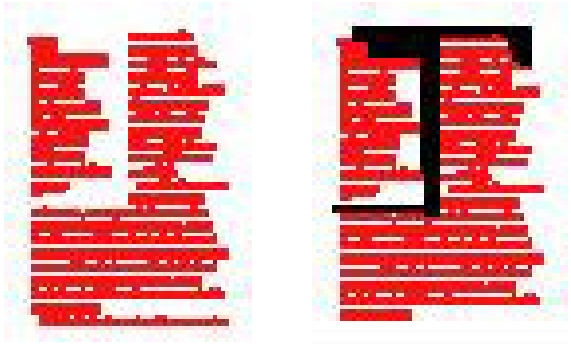
This paper used vertical tracing of space between blocks. The method is applied with the optimum image instead of using original image. The advantage is that it has faster computation time. If the separator reaches the bottom of the block, block is extracted. If the separator can not reaches the

bottom of the block but it is deep more than threshold, start checking on the left hand side if it can reaches to left boundary of that block, the block is extracted. Fig. 12 shows the example of using this method.

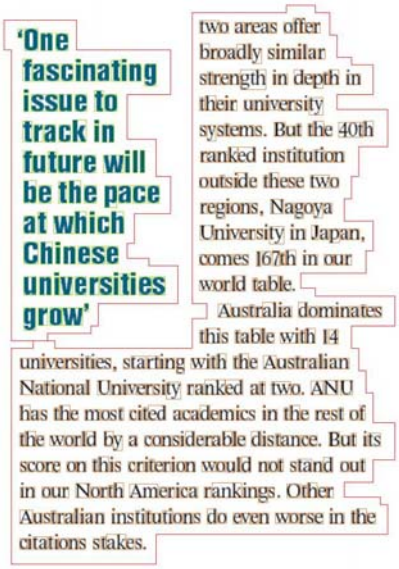
5. Experimental Results

Our method was implemented on Sempron 2400, 960 MB of ram under Window XP SP2 operating system using MS Visual Basic.Net 2003. The 10 test documents with A4 size are scanned at 300 dpi. Fig. 13 and 14 show examples of the documents and results of page segmentation by using the proposed scheme.

The efficiency and effectiveness of the propose scheme is shown by comparing the processing time of this scheme with the method that used the window size of 32x32 pixels called method1. The test results are shown in Table 3.

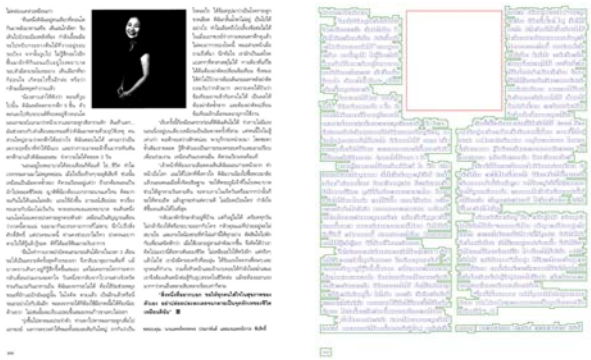


(a) (b)



(c)

Fig. 12. (a) An optimum image before dilation, (b) Position that checking for extracting blocks, (c) Output



(a) Original image (b) Result image

Fig. 13. An example of test document 1 .

(a) Original image (b) Result image

Fig.14. An example of test document 2.

Table 3 . Comparison of computation time of using method1 and propose scheme.

No.	Image Size	Method1(sec)	Proposed scheme (sec)
1	2400x3300	27.828	7.577
2	2400x3300	35.468	8.937
3	2400x3300	30.89	8.5
4	2800x3500	38.796	9.186
5	2500x3350	30.578	7.062
6	2700x3500	39.656	7.749
7	2800x3300	44.312	9.696
8	2300x3300	30.859	7.914
9	2400x3300	34.937	7.905
10	2400x3300	26.234	5.921

6 Conclusion

The fast and efficient scheme for document page segmentation in OCR application by using window and optimum image is presented. The tested results show that the method is faster and more accurate

than that of other methods. The scheme also can be used with all types of document image structures. Therefore, it can be used to speed-up the processing time of this process significantly.

References:

[1] Jiajun Wang, Yanling Li, Xianwu Huang and Zhenya He, "Page Segmentation and Classification Based on Pattern-list Analysis," International Symposium on Intelligent Multimedia, Video and Speech Processing, 2004, pp 735-738.

[2] Jaekyu Ha, Robert M. Haralick and Ihsin T. Phillips, "Recursive X-Y Cut using Bounding Boxes of Connected Components," IEEE, 1995, pp 952-955.

[3] Dimitrios Drivas and Adnan Amin, "Page Segmentation and Classification Utilising Bottom-Up Approach," IEEE, 1995, pp 610-614.

[4] P. Sutheebanjard, W. Premchaiswadi, N, Premchaiswadi, "A Fast Block Extraction Method for Document Page Segmentation," EECON26, Vol. 2, 6-7 Nov. 2003 pp. 1069 -1074

[5] B. Kruatrachue, P. Suthaphan, "A Fast and Efficient Method for Document Page Segmentation for OCR," Electrical and Electronic Technology, 2001. Vol. 1, 19-22 Aug. 2001 pp. 381 -383