A Speed Enhancement Method for Document Page Segmentation 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 a speed enhancement method for document page segmentation which is one of the most important processes in an Optical Character Recognition (OCR) system. A mixed approach based on an optimum image is proposed. This method consists of four steps: (1) finding black pixels in an image by using a window of type "I" to reduce computation time (2) identify text and picture area (3) create the optimal image from the original image and (4) block extraction. In this proposed method, instead of using one pixel, 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 window of the original picture is 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 structures correctly. Finally, the optimum image is used for a block extraction process to provide a 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

1 Introduction

Document page segmentation is one of the most important processes used in OCR for the identification of areas in the image of a document page. Many document page segmentation techniques have been proposed which can be classified as; top-down [1, 2], bottom-up [3] and mixed methods [4, 5]. A contour edge following algorithm 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 less overheads than algorithms that need to access all the pixels of a document. However, if an area contains no black pixels, it must check every pixel in 32x32 pixels window or 1024 pixels.

In this paper, a mixed approach based on an optimum image is proposed. This method consists of four steps.

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

2 Finding Black Pixels

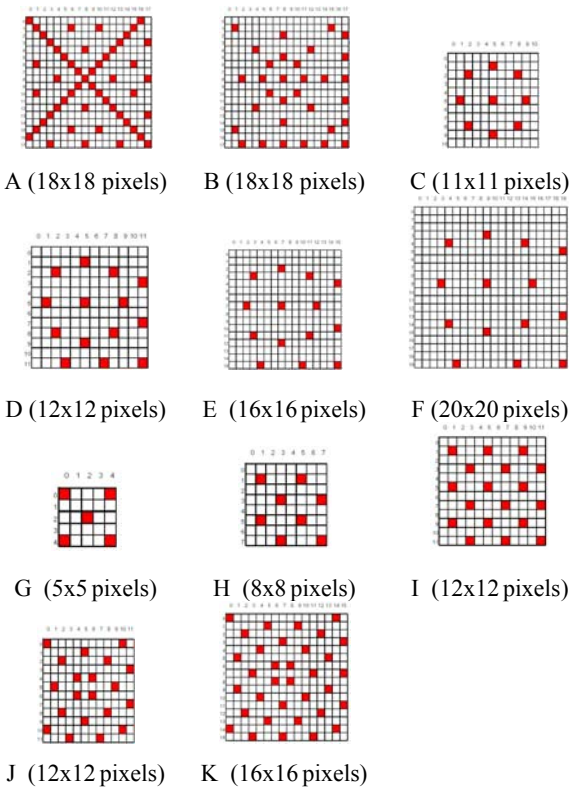


Fig. 1. Window type A to K

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

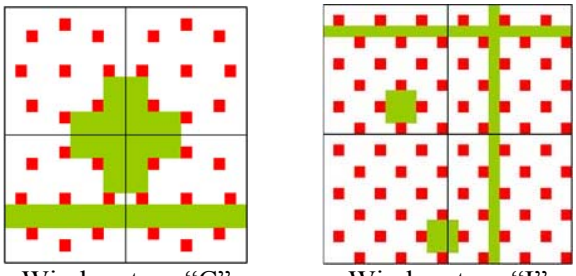
Table 1. Characteristics of each window type.

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

These windows are used for finding the optimal widow. From the test results, it was found that windows of type A to F can not process correctly while the windows of type G to K can process correctly. Finally, window type "I" is selected for scanning in the proposed scheme.

Window type "I" is a window size of 12 by 12 pixels and 18 points are used for checking for black pixels, so the method only needs to check 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"

From Fig. 2, some of the non checked areas are shown. We can also see that the maximum size of the non checked area of window type "I" is only 3 by 3 pixels. The comparison of non checked points of window type "C" and "I" is shown in Table 2.

Table 2. Compare non checking point in window type "C" and "I"

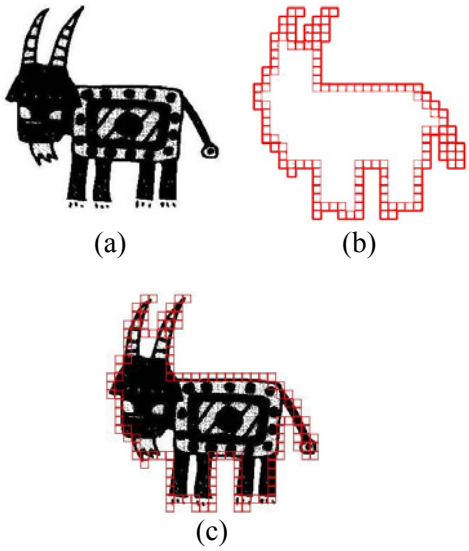
Window Type	Non Checking point in window				
	1px	2px	3px	4px	5px
C	1x∞	2x∞	3x10	4x10	-
I	1x∞	2x3	3x3	-	-

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

3 Identify Text and Picture Area

The process described above is used to identify text and picture areas of the image. It begins by finding black pixels with a raster scan of window type "I" for the entire image. After the first black pixel is found, a contour edge following technique with a chain of 1 pixel is used to find the size of the block of these black pixels. Then, these blocks have to be classified as picture, character or noise. In order to classify the block of black pixels into picture, character or noise, the proposed method uses the criteria as the followings:

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



(a) (b)

(c)

Fig. 3. (a) An original image
(b) A perimeter of the picture,
(c) A perimeter of picture on the original image.

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

(a) (b)

Fig. 4. Store character by rectangle
(a) Original image, (b) Rectangle of character

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

4 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 is used only for the area containing characters. The window size of 12 by 12 pixels, window type "I", is used to scan the original image, and the window will be represented by using a 1 pixel optimum image. Therefore, the optimum image will have its width and height about 1/12 times the original image. The example of creating the optimum image is shown in Fig. 5.

(a) (b)

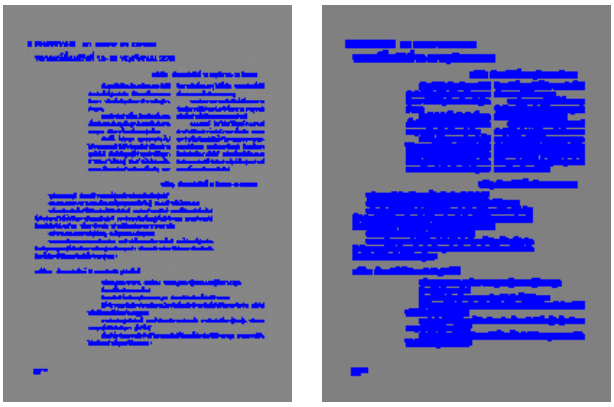
Fig. 5. (a) An original image (2400x3300 pixels)
(b) An optimum image of (a) (200x275 pixels)

5 Block Extraction

The proposed process is used to extract blocks from the optimal image generated from the method described in the previous section. It consists of 2 steps as follows: (1) joining character blocks and (2) block separation.

5.1 Joining Character Blocks

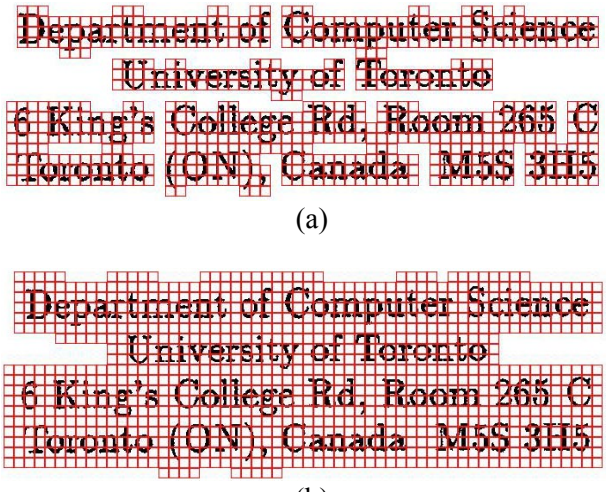
The dilation technique is used with a mask size of 3x3 pixels for connecting adjacent characters of the optimal image into the same block. The example of using this technique is shown in Fig. 6.



(a) (b)

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

Each pixel in an optimum image represents the 12x12 pixels in the original image, so dilation of the optimum image with a mask size of 3x3 pixels has an 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. 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 technique is used with a chain of 1 pixel to find the perimeter of the block. To show the efficiency and the limitation of the use of window type "I", we can compare it with the use of a window of size 32 by 32 pixels. In Fig. 8 shows that the window type "I" can not separate characters having a distance less than 45 pixels while Fig.9 shows that a window of size 32X32 pixels can not separate character having a distance less than 62 pixels.

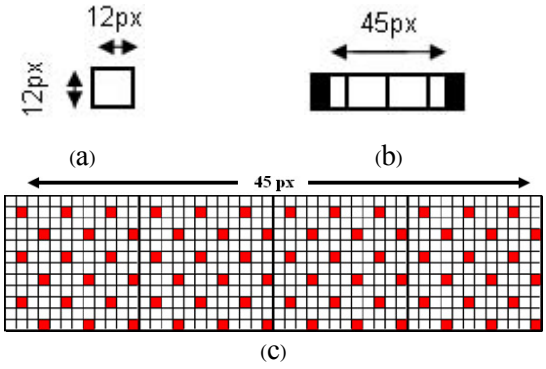


Fig. 8. (a) window type "I"
(b) minimum distance for this method, 45 pixels
(C) measuring of 45 pixels.

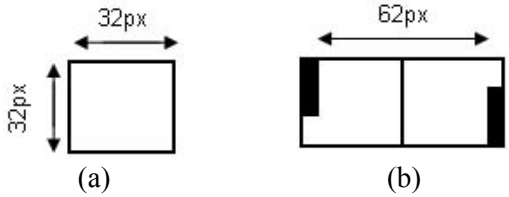


Fig. 9. (a) The 32x32 pixels window,
(b) minimum distance for this method, 62 pixels,

For the proposed technique, many cases have been tested with the window type "I" and the limitation of the technique are shown in Fig. 10. There are 7 cases in Fig. 10, case (a) to case (g). The details of each case are shown below.

(a) The space between blocks is 34 pixels, the blocks 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 blocks can be separated.

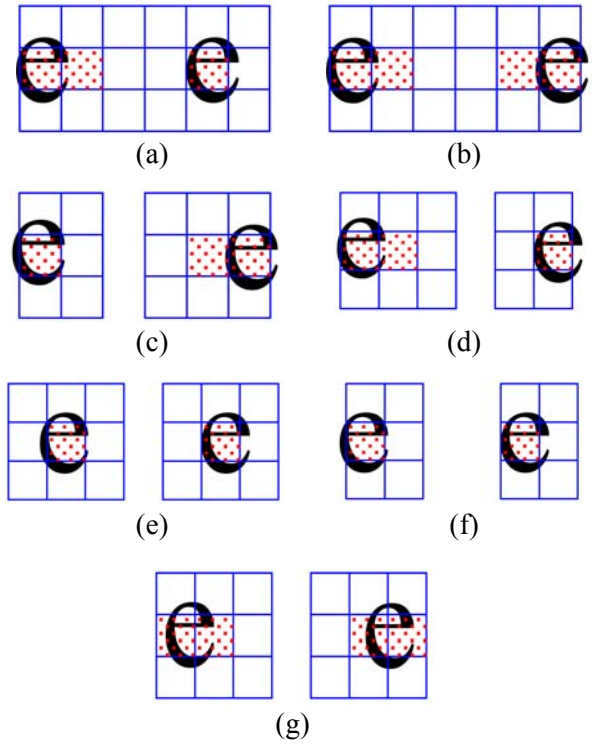


Fig. 10. The use of windows type "I"

The results can be summarized into 3 categories

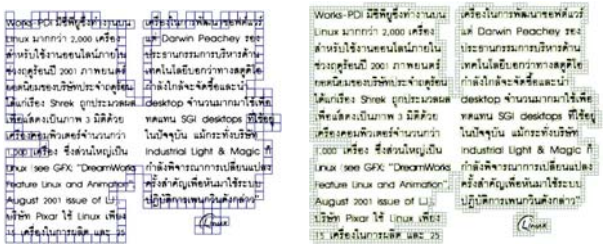
1. If the space between blocks is less than 35 pixels, the technique can not separate blocks.
2. If the space between blocks is between 35 and 45 pixels, the technique may or may not separate blocks. Fig. 10 (b) shows that the technique can not separate blocks while in Fig. 10 (e) it can separate block.
3. If the space between blocks is more than 45 pixels, the technique can separate blocks.

Fig. 11 and 12 show examples of applying a window of size 32x32 pixels and window type "I" on a real document.

(a) (b)

Fig. 11. (a) using window 32X32
(b) using window type "I"

Fig. 11 shows that window type "I" can separate the blocks correctly while the used of window size of 32x32 pixels can not separate the block correctly. However Fig. 12 shows that both methods can not separate blocks correctly because the space between the two blocks is too small.



(a) (b)

Fig. 12. both methods can not separate blocks
(a) window 32x32 (b) window type "I"

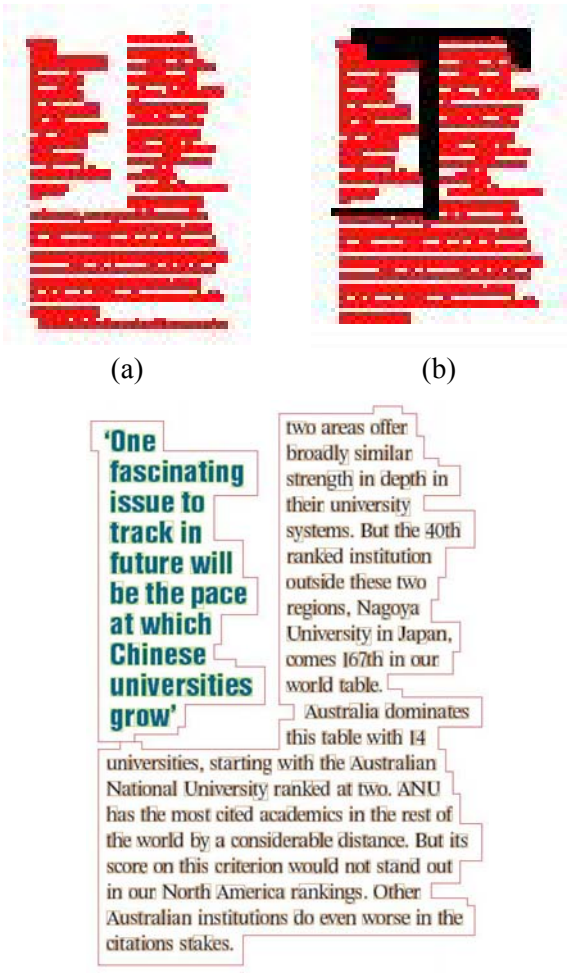
5.2 Blocks Separation

The vertical tracing of space between blocks is used for block separation. This method is applied with the optimum image instead of the original image. The advantage is that it can provide faster computation time. In this process, if the tracing reaches the bottom of the block, it means that the block is extracted. If the tracing can not reach the bottom of the block but it is deeper than a threshold, then the process starts checking on the left hand side. If it can reach the left boundary of that block, the block is extracted. Fig. 13 shows the example of using this method.

6. Experimental Results

The proposed method was implemented on a Sempron 2400, 960 MB of RAM under Window XP SP2 operating system. The MS Visual Basic.Net 2003 was used for the experiment. Ten tested documents with A4 size were scanned at 300 dpi for using in the experiment. Fig. 14 and 15 show examples of the documents and the results of page segmentation by using the propose method.

The method can also be applied with other languages which have structures more complex than the English language, such as the Thai Language. Furthermore, it can also be used with all types of document image structures as shown in Fig. 14 and Fig.15.



(a) (b)

(c)

Fig. 13. (a) Optimum image before dilation,
(b) Position checking for block extraction
(c) Result of block extraction.

(a) Original image (b) Result image

Fig. 14. An example of test document 1 .

(a) Original image (b) Result image

Fig.15. An example of test document 2.

The efficiency and effectiveness of the proposed technique is shown by comparing the processing time of this technique with the method that is used with a window size of 32x32 pixels called method-1. The test results are shown in Table 3 and Fig. 16.

Table 3. Comparison of computation time of method-1 and proposed scheme.

Case No.	Image Size	Method-1 (sec)	Propose 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

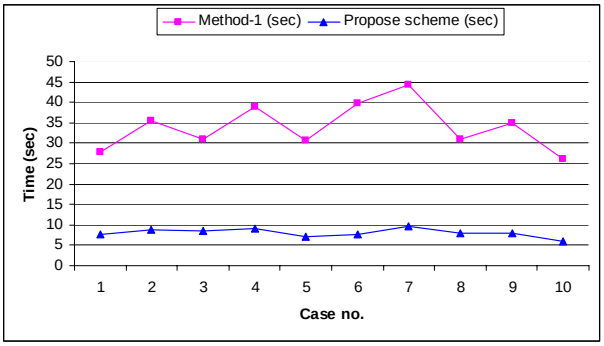


Fig.16 Comparison of computation time of method-1 and the proposed scheme

6 Conclusion

The fast and efficient technique for document page segmentation in an OCR application by using window and optimum image has been presented. This method consists of four steps: 1) Finding black, 2) Identify text and picture area, 3) Create optimal image and 4) Block extraction. The test results show that the proposed technique of using window type "I" and optimum image is faster and more accurate than that of other methods. The technique can also be applied with other languages which have structures more complex than the English language, such as the Thai Language. Furthermore, it can also be used with all types of document image structures as shown in Fig. 14 and Fig.15. Therefore, the technique could significantly speed-up the processing time for document segmentation.

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