Improved Document Image Segmentation Algorithm using Multiresolution Morphology

Syed Saqib Bukhari[a], Faisal Shafait[b], and Thomas M. Breuel[a]

[a] Technical University of Kaiserslautern, Kaiserslautern, Germany,
[b] German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany

ABSTRACT

Page segmentation into text and non-text components is an essential preprocessing step before OCR operation. If this is not done properly, an OCR classification engine produces garbage text due to the presence of non-text components. This paper describes improvements to the text/image segmentation algorithm described by Bloomberg,[1] which is also available in his open-source Leptonica library.[2] The modifications result in significant improvements over Bloomberg's algorithm on UW-III, UNLV, ICDAR 2009 page segmentation competition test images and circuit diagram datasets.

1. INTRODUCTION

In document images, basic shapes of text characters are limited in number, but shapes of the non-text components including halftone, drawing, logos, graphs, etc. are unlimited. Therefore, OCR engines treat both text and non-text components differently, such that they only recognize text components and then arrange recognized text characters and images of non-text components in an output document using layout information. Document image segmentation, that is the problem of separating text and non-text components, is one of the most important preprocessing steps before feeding the specific contents to an optical character recognition (OCR) system. Otherwise, an OCR engine produces lot of garbage characters originating from non-text components. Document image segmentation approaches presented in the literature can be generally classified into following groups: (i) multiresolution morphology based segmentation,[1] ii) block or zone based classification,[3] iii) pixels based classification[4] and iv) connected component based classification.[5]

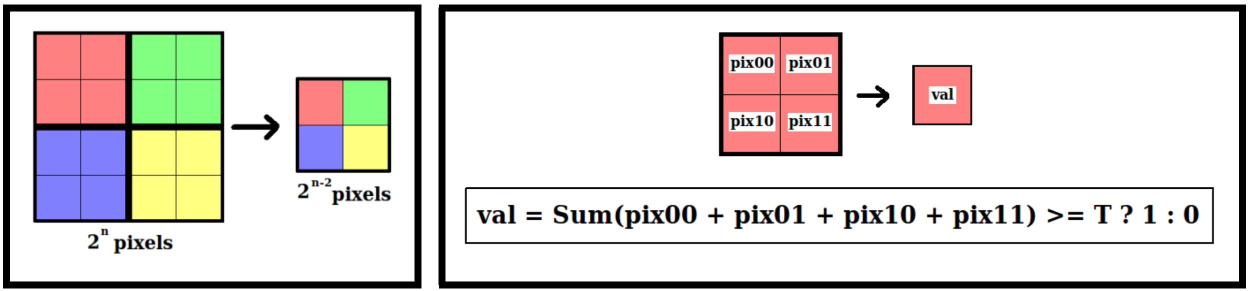
Bloomberg[1] described an approach to page segmentation based on multiresolution morphology. Bloomberg's approach is simple and performs well for separating halftone images from text. Furthermore, an open source implementation is available as part of the Leptonica library.[2] Bloomberg's text/image segmentation approach was specifically designed for separating text and halftone components. It is often unable to differentiate between text and non-text components other than halftones, like drawings, graphs, maps etc. Here, we introduce improvements in Bloomberg's text/image segmentation algorithm to generalize it for separating text and non-text components including halftones, drawings, graphs, maps, etc.

The paper is organized as follows. In Section 2, Bloomberg's text/image segmentation algorithm is described in detail. In Section 3, our modifications to Bloomberg's algorithm is explained. Section 4 deals with the experimental results and Section 5 discusses our conclusions.

2. BLOOMBERG'S TEXT AND HALFTONE IMAGE SEGMENTATION

Multiresolution morphology is the main technique used in Bloomberg's text/image segmentation algorithm. Bloomberg[1] first defined the outline of the text/image segmentation algorithm using basic morphological operations before introducing his multiresolution morphology based algorithm, such that: i) an image can be closed with a sufficiently large structuring element intending to solidify halftone components, ii) then the image can be opened with an even larger structuring element intending to remove the text blobs and to preserve some portions of halftone components, iii) the residual portions or seeds of the halftone image can be used for generating the halftone mask from the original image. Bloomberg[1] has highlighted the importance of the multi-scale image

bukhari@informatik.uni-kl.de,faisal.shafait@dfki.de and tmb@informatik.uni-kl.de



(a) 4 × 1 Threshold Reduction (b) Formula of 4 × 1 Threshold Reduction

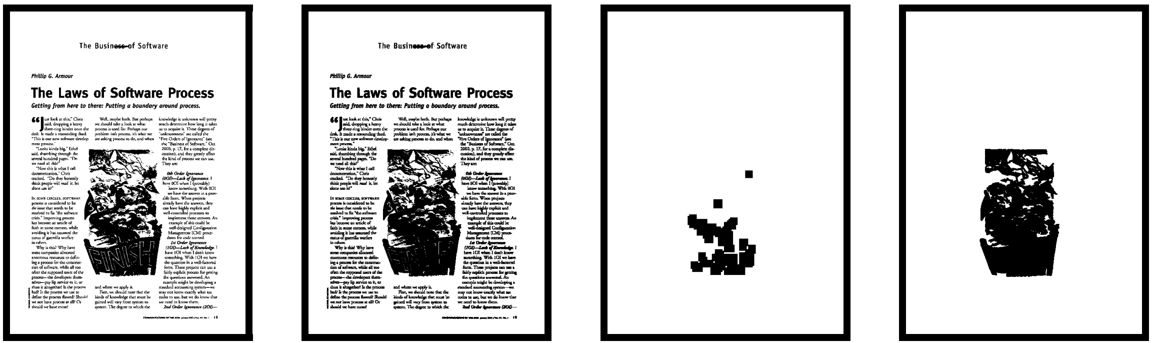
Figure 1. Definition of multiresolution morphology based threshold reduction operation: (a) each 2×2 block of four pixels is subsampled to one pixel (4×1 Reduction). (b) the value of subsampled or reduced pixel is '1' if the sum of the values of four pixel within 2×2 block is greater than or equal to the threshold (T), otherwise '0'. The threshold can be set between one and four.

representation by emphasizing that it can be used for efficient analysis of image contents as well as speeding up image processing operations (like morphology). He updated the aforementioned basic outline of the text and halftone segmentation algorithm using multi-scale image representation such that: i) an image can be closed or dilated before subsampling, in order to coalesce halftone components, ii) the image can be opened or eroded before further subsampling to intend to preserve only halftone portions. As it is expensive to use large structuring element at full or high image resolution, he introduced the key concept of "threshold reduction" for implementing the subsampling based text/image segmentation algorithm. The threshold reduction is defined as follows.

Threshold Reduction: consider a binary image where each foreground pixel is represented by '1' and each background pixel is represented by '0'. The image is tiled into 2×2 pixel blocks. Each 2×2 block of four pixels is replaced by a single pixel in subsampled image. The value of each subsampled pixel is either '1' or '0' depending on the chosen threshold, that ranges between one and four. The subsampled pixel value is '1' if the sum of the values of four pixels is greater than or equal to the threshold, otherwise '0'. The subsampling operation of each 2×2 block into single pixel with the threshold equal to one mimics the dilation of image with 2×2 structuring element followed by subsampling of upper-left pixel of each 2×2 pixel block. Similarly, subsampling with the threshold equal to four mimics the erosion of image with 2×2 structuring element followed by subsampling of upper-left pixel of each 2×2 pixel block. Besides thresholds of one for dilation and four for erosion, the threshold can be set equal to two or three as well. This type of threshold selection is referred as threshold convolution or rank order filter. Bloomberg[1] referred the combination of threshold convolution followed by subsampling as "threshold reduction". After a single threshold reduction (also called 4×1 reduction) operation, the number of image pixels is reduced from 2^n to 2^{n-2}. The concept of threshold reduction is illustrated in Figure 1. Bloomberg's text/image segmentation algorithm is described below.

2.1 Algorithm

Bloomberg's algorithm is based on the aforementioned threshold reduction (multiresolution morphology) concept and basic morphological operations. It also uses the trivial 1×4 expansion operation in which each pixel value is copied into 2×2 pixel block of four pixels. Bloomberg's halftone mask image generation algorithm is described as follows. Consider a binary image in which the foreground pixel value is '1' and the background pixel value is '0'. At first an input image is processed by two threshold reduction operations with thresholds equal to one. This operation subsamples the input image from 2^n to 2^{n-4} pixels by preserving the density of low as well as high frequency components within document image. This image can then be referred to as a 16×1 subsampled image, as shown in Figure 2(b). The subsampled image is further reduced by two threshold reduction operations with thresholds equal to four and three respectively and then followed by morphological opening by using a 5×5 structuring element. These further threshold reductions of the 16×1 subsampled image and morphological opening are intended to remove the text components and preserve some portions of halftone components, as shown in Figure 2(c). The image in Figure 2(c) is referred to as the seed image. The seed image is expended by using two 1×4 expansions to become equal in size to the 16×1 subsampled image of Figure 2(b). Finally, the halftone mask image is generated by comparing the 16×1 subsampled image (Figure 2(b)) with the seed image



(a) input image (b) 16 × 1 subsampled image (c) seed image (d) halftone mask image

Figure 2. Snapshots of Bloomberg's text and halftone image segmentation algorithm.

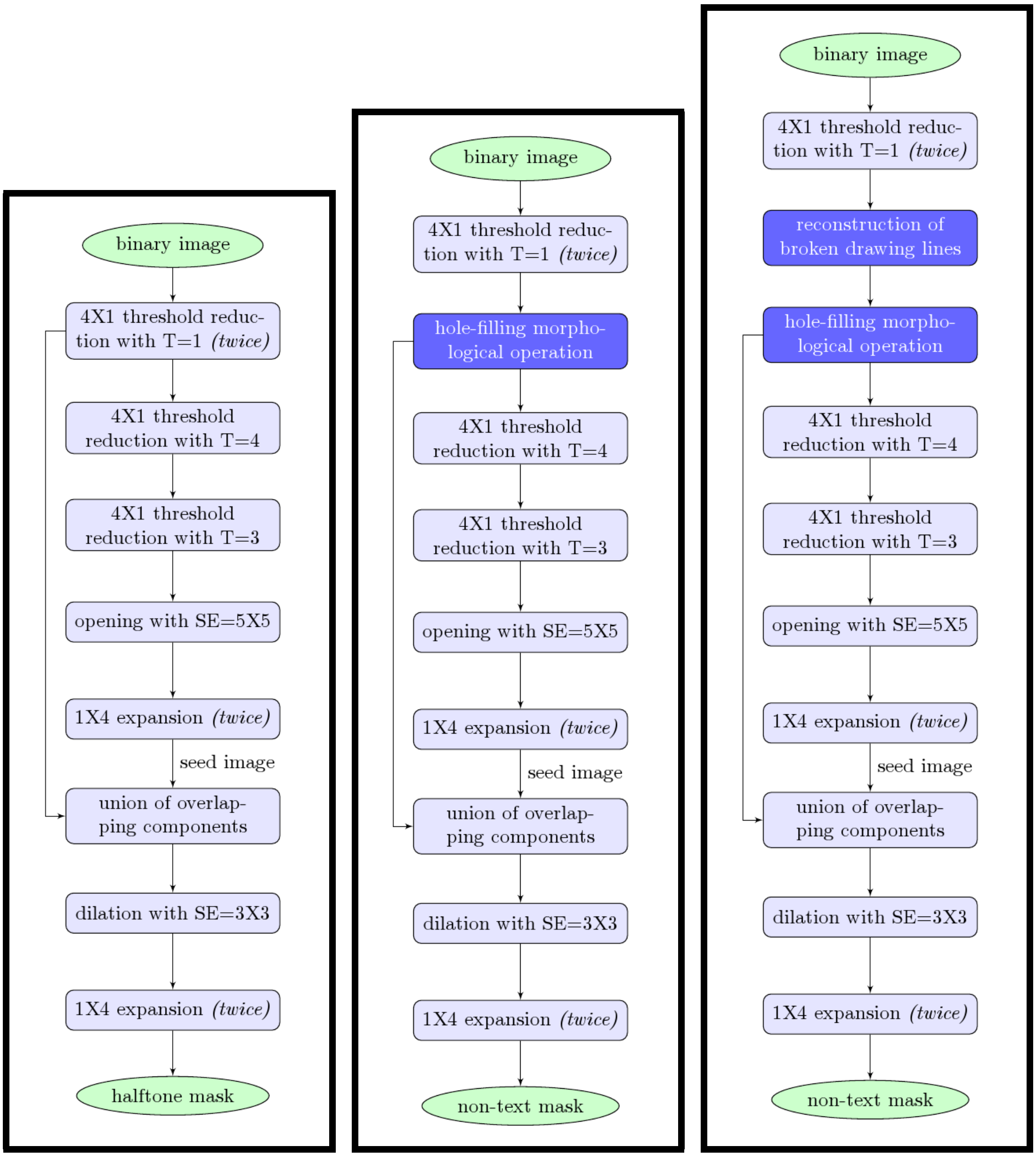
(Figure 2(c)) and selecting only fully or partially overlapping components between them. After morphological dilation (structuring element 3 × 3), the halftone mask image is expended by two 1 × 4 expansions to become equal to the dimension of the input image. The halftone mask image is shown in Figure 2(d). The data flow diagram of Bloomberg's text/image segmentation algorithm is shown in Figure 3(a).

3. MODIFICATION TO BLOOMBERG'S ALGORITHM

Bloomberg's text/image segmentation algorithm is specifically designed for separating text and halftone image from a document image. It is unable to discriminate between text and drawing type non-text components and therefore fails to separate both of them from each other. Here we first describe why Bloomberg's algorithm is unable to distinguish between text and non-text components except text and halftone components. Then we introduce modifications to Bloomberg's text/image segmentation algorithm to improve it for efficiently separating text and non-text components, including halftones, drawings, maps, graphs, etc.

Bloomberg's algorithm is intended to preserve some portion(s) of the halftone image components in the seed image, which is later used for generating the halftone image mask. The seed image is generated by using four consecutive threshold reduction operations with thresholds equal to one, one, four and three respectively. The threshold reduction with the threshold equals to one preserves low as well as high frequency details of an image while a threshold greater than one drops fine or minor image details. On one hand, if a non-text component in an original image contains some solid bunch of pixels, then it will be preserved in the seed image, even after the threshold reductions with thresholds greater than one. On another hand, if a non-text component only composed of thin drawing lines, then it will vanish after the high value threshold reductions in the seed image. Therefore, Bloomberg's algorithm often fails to separate text and non-text components where non-text components do not contain any solid bunch of pixels, as shown in Figure 4.

Hole-filling morphological operation (first modification): we have observed that non-text components, such as drawings, maps, graphs and even halftones, are often composed of hollow contours of geometric and irregular shapes, as shown in Figure 4(a). The threshold reduction operations with high thresholds remove these hollow contours. But if these hollow contours can be filled before the high value threshold reduction operations, then they will remain present in the seed image. Another consequence of using the image filling operation is that it only fills hollow shape image components and preserve other text and non-text components as before. For this purpose, the well known "hole-filling" morphological operation is used, which is briefly described here as, (i) an input image with foreground pixels '1' and background pixels '0' is used as mask image, ii) the filled-image is initialized with all '0' pixels except the top-left pixel with '1', iii) the filled-image is dilated using a 3 × 3 structuring element, iv) after dilation, all of the pixels that are '0' in the mask image are set to '0' in the filled-image, v) dilation followed by resetting of the filled-image's pixels is repeated until no more changes are made to the filled-image. The hole-filling based modified Bloomberg's algorithm is briefly illustrated in Figure 3(b). The text/image segmentation results of the original Bloomberg's' algorithm and modified version are shown in Figures 4 and 5 respectively for comparison. Unlike the original Bloomberg's algorithm, the improved version can accurately separate text and non-text images including halftones, drawings, logos, graphs, maps etc.

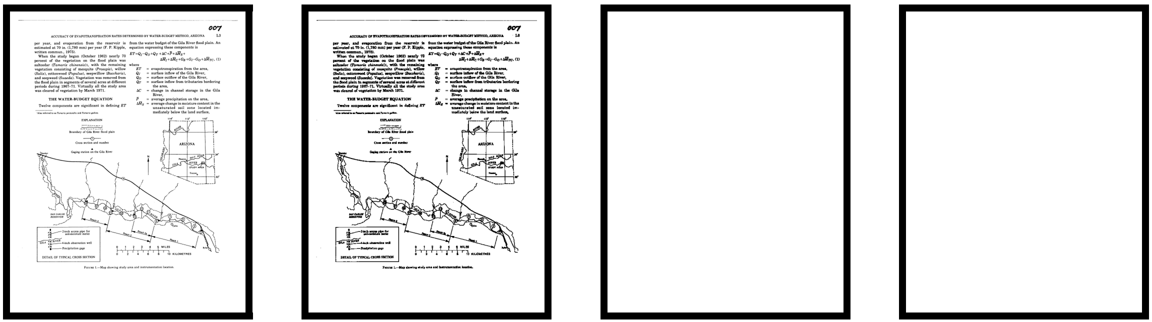

binary image
4X1 threshold reduction with T=1 (twice)
4X1 threshold reduction with T=4
4X1 threshold reduction with T=3
opening with SE=5X5
1X4 expansion (twice)
seed image
union of overlapping components
dilation with SE=3X3
1X4 expansion (twice)
halftone mask
binary image
4X1 threshold reduction with T=1 (twice)
hole-filling morphological operation
4X1 threshold reduction with T=4
4X1 threshold reduction with T=3
opening with SE=5X5
1X4 expansion (twice)
seed image
union of overlapping components
dilation with SE=3X3
1X4 expansion (twice)
non-text mask
binary image
4X1 threshold reduction with T=1 (twice)
reconstruction of broken drawing lines
hole-filling morphological operation
4X1 threshold reduction with T=4
4X1 threshold reduction with T=3
opening with SE=5X5
1X4 expansion (twice)
seed image
union of overlapping components
dilation with SE=3X3
1X4 expansion (twice)
non-text mask

(a) The original Bloomberg's algorithm

(b) First modified version

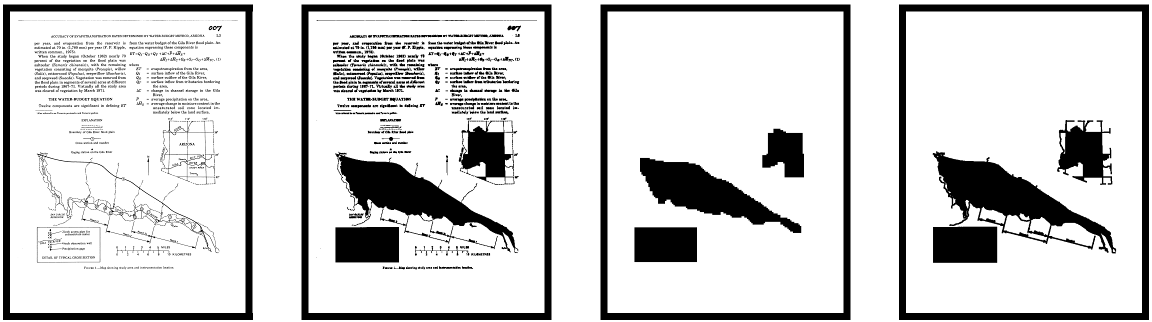
(c) Second modified version

Figure 3. Data flow diagrams of Bloomberg's text/image segmentation algorithm and our modified versions ('T': threshold; 'SE': structuring element).



(a) input image (b) subsampled (16 × 1) image (c) seed image (d) halftone mask image

Figure 4. Bloomberg's text/image segmentation algorithm often fails to separate drawing type non-text components from text components.



(a) input image (b) subsampled (16 × 1) image followed by hole-filling operation) (c) seed image (d) non-text mask image

Figure 5. First modified Bloomberg's text/image segmentation algorithm: hole-filling based improved Bloomberg's algorithm produces accurate non-text mask for drawing type components as compared to the result of the original Bloomberg's algorithm as shown in Figure 4.

Reconstruction of broken drawing lines (second modification): we have also observed that sometimes non-text components consist of broken drawing lines, by choice or because of document digitization errors (like low resolution, bad binarization etc.). Hole-filling morphological operation only fills those hollow drawing components which are composed of unbroken contour lines. Non-text components with broken drawing lines remain unfilled, even after hole-filling operation. Therefore, even modified version of Bloomberg's algorithm misclassifies them as text components, which is shown in the top row of Figure 6. We can further improve Bloomberg's algorithm by reconstructing broken drawing lines before the hole-filling operation. At first, one might consider using a morphological closing operation with oriented structuring elements for drawing lines reconstruction. However, a morphological closing operation can not handle drawing line reconstruction and produces worse effect on the final non-text mask, as shown in the middle row of Figure 6. Here we introduce an efficient and easy way to implement a horizontal and vertical drawing line reconstruction algorithm, which can be generalized for a variety of line orientations. Our drawing line reconstruction algorithm is described as follows: i) horizontal and vertical lines from the morphologically thinned 16 × 1 subsampled image are identified using a morphological hit-miss transform using horizontal and vertical structuring elements respectively, ii) the broken horizontal lines are blended together through anisotropic Gaussian smoothing with $\sigma_x > \sigma_y$, iii) the smoothed image is converted into the binarized image using global thresholding, which produces connected horizontal lines, iv) these line are labeled using connected components analysis, v) the broken horizontal lines are labeled with respect to the labeling of connected horizontal lines. vi) finally, all of the broken lines with the same label are joined together, resulting in reconstructed horizontal drawing lines, vii) the same procedure is repeated for reconstructing vertical drawing lines by smoothing with $\sigma_x < \sigma_y$. The modified Bloomberg's algorithm with reconstruction of broken drawing lines before hole-filling operation is shown in Figure 3(c). The results of reconstructed drawing lines

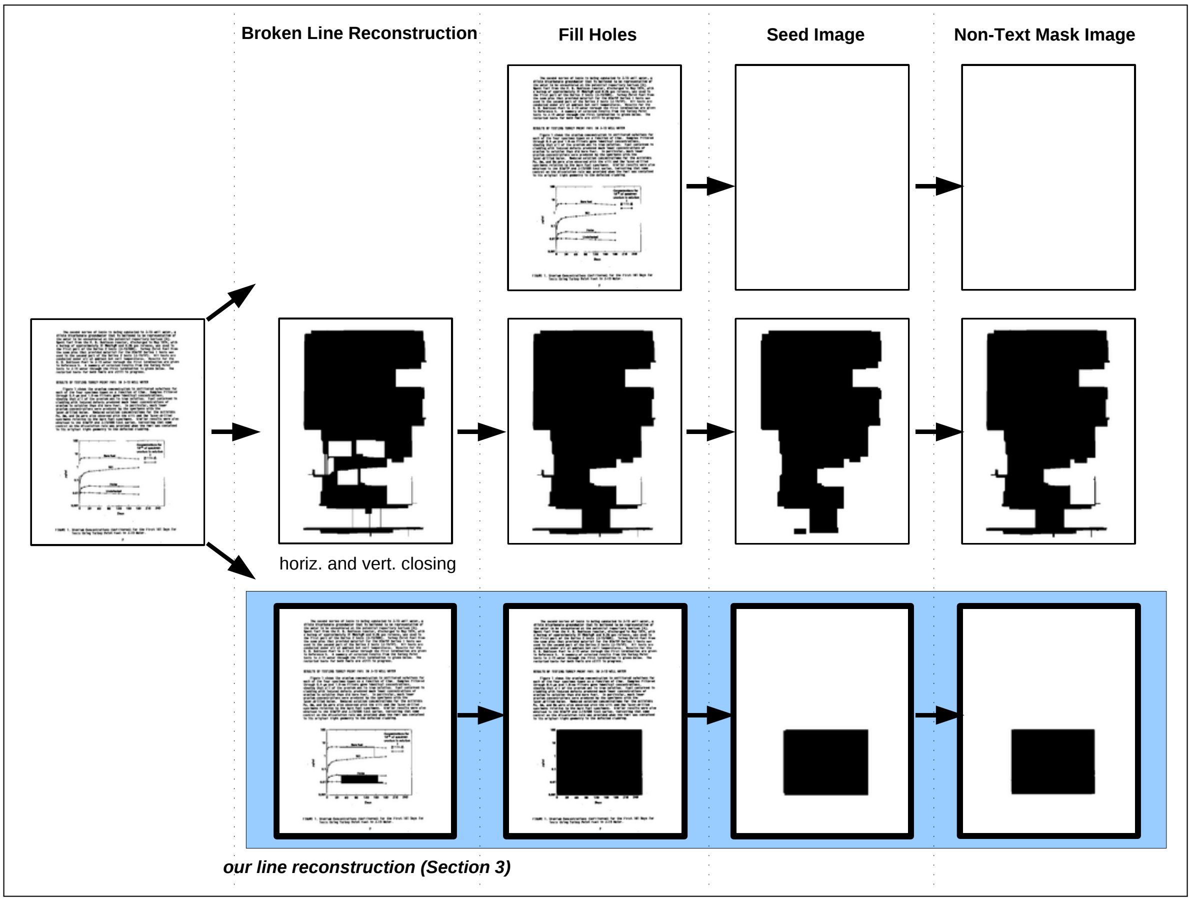


Figure 6. Second modified Bloomberg's text/image segmentation algorithm: reconstruction of broken drawing lines followed by hole-filling based modified Bloomberg's algorithm. **Top Row:** hole-filling operation on broken drawing lines image does not fill the hollow non-text components and therefore misclassifies non-text components as text. **Middle Row:** horizontal and vertical closing based line reconstruction produces a garbage image and a garbage non-text mask. **Bottom Row:** our broken line reconstruction method (described in Section 3) generates closed contour drawing shapes, which help in producing an accurate non-text mask.

using our aforementioned approach and its positive effect on final non-text mask separation are shown in the bottom row of Figure 6.

4. EXPERIMENTS AND RESULTS

We have compared the performance of Bloomberg's original text/image segmentation algorithm and our modified versions using standard datasets like UW-III, UNLV, ICDAR-2009 page segmentation competition test images and our private circuit diagrams images. The main reason for using different datasets is to compare the text/image segmentation accuracy of these algorithms on different types of document images with a variety of text and non-text components. A total of 95 documents, mainly composed of text and halftone components, were selected from the UW-III dataset. 100 documents, comprising of text, halftone, graphs, drawings, maps, etc, were selected from UNLV Magazine Sample 2 (category Z). Our circuit diagrams dataset composed of 10 images having text and drawing components. ICDAR 2009 dataset contains 8 test images with non-Manhattan layout, unlike the other selected datasets. For each dataset, pixel-level ground-truth images were generated using zone-level ground truth information. Each pixel in a ground-truth image contains either a text or non-text label.

Different types of metrics were used for the performance evaluation of text/image segmentation algorithm, as defined below:

Table 1. Performance evaluation of the original Bloomberg's text/image segmentation algorithm and our modified versions on UW-III dataset (95 document images) and ICDAR 2009 page segmentation competition test dataset (8 document images).

	UW-III			ICDAR-2009		
	Original	1st version	2nd version	Original	1st version	2nd version
non-text classified as non-text	95.36%	99.39%	99.51%	85.62%	91.44%	98.41%
text classified as text	99.79%	99.28%	99.19%	100%	99.11%	99.42%
segmentation accuracy	97.58%	**99.34%**	**99.35%**	92.81%	**95.28%**	**98.92%**

Table 2. Performance evaluation of the original Bloomberg's text/image segmentation algorithm and our modified versions on UNLV dataset (100 document images) and Circuits dataset (10 document images).

	UNLV			CIRCUITS		
	Original	1st version	2nd version	Original	1st version	2nd version
non-text classified as non-text	18.48%	72.98%	79.39%	0%	89.11%	90.31%
text classified as text	99.98%	97.97%	97.48%	100%	100%	96.67%
segmentation accuracy	59.23%	**85.48%**	**88.45%**	50%	**94.56%**	**93.49%**

1. **non-text classified as non-text:** percentage of intersection of non-text pixels in both segmented image and ground-truth image with respect to the total number of non-text pixels in the ground-truth image.
2. **text classified as text:** percentage of intersection of text pixels in both segmented image and ground-truth image with respect to the total number of text pixels in the ground-truth image.
3. **segmentation accuracy:** average percentage of non-text classified as non-text and text classified as text accuracy.

Based on the metrics defined above, the comparison among the original Bloomberg's text/image segmentation algorithm and our modified versions are shown in Table 1 and 2. It is clearly visible in Table 1 and 2 that our modified versions achieved better segmentation accuracy as compared to the original Bloomberg's algorithm.

5. DISCUSSION

Bloomberg's text/image segmentation algorithm[1] is specifically designed for text and halftone image separation. It is simple and fast approach and performs well on text and halftone image segmentation, but it is unable to segment text and non-text components other than halftones, such as drawings, graphs, maps, etc. In this paper, we have introduced modifications to the original Bloomberg's algorithm for making it a general text and non-text image segmentation approach, where non-text components can be halftones, drawings, maps, graphs, etc. We have evaluated the original Bloomberg's approach and our modified versions on standard datasets like UW-III, UNLV and ICDAR 2009 page segmentation competition test images as well as our circuit diagram dataset. The modifications result in significant improvements over the original Bloomberg's text/image segmentation algorithm, as shown in Table 1 and Table 2.

REFERENCES

[1] Bloomberg, D. S., "Multiresolution morphological approach to document image analysis," in [*Proc. Int. Conf. Documnet Analysis and Recognition (ICDAR 1991)*], 963–971 (1991).

[2] Bloomberg, D. S., "Leptonica: An open source c library for efficient image processing and image analysis operations." http://code.google.com/p/leptonica/.

[3] Keysers, D., Shafait, F., and Breuel, T. M., "Document image zone classification- a simple high-performance approach," in [*Proc. 2nd Int. Conf. Computer Vision Theory and Applications*], 44–51 (Mar. 2007).

[4] Moll, M. A., Baird, H. S., and An, C., "Truthing for pixel-accurate segmentation," in [*Document Analysis Systems, the Eighth IAPR Int. Workshop*], 379–385 (Sep. 2008).

[5] Bukhari, S. S., Shafait, F., and Breuel, T. M., "Document image segmentation using discriminative learning over connected components," in [*Proc. 9th IAPR Workshop on Document Analysis Systems*], 183–190 (2010).