A Novel Stroke Width Based Binarization Method to Handle Closely Spaced Thick Characters

P. Pavan Kumar, Atul Negi, B.L. Deekshatulu, Chakravarthy Bhagvati, Arun Agarwal

ABSTRACT

Signboards and billboards provide a challenge to image segmentation methods, since these images may also have pictures and graphical objects, apart from text objects. Methods that often succeed in more traditional text block segmentation situations do not perform well here since estimation of text lines and character widths etc fail due to the short sample sizes. Further, extraction of characters of different font sizes, which can be found in the real world and signboard images, remains a problem. In this paper, as a solution to the mentioned problem, we propose two stroke width based binarization approaches. These approaches can be used to eliminate extraneous objects based upon estimates of stroke width. We compare our methods with several other stroke width based binarization methods. We observe that the previous approaches fail, when there are closely spaced thick characters. We show that our second approach is able to extract closely spaced thick characters better than any of the other methods.

Keywords

Image Segmentation, Thresholding, Stroke Width, Thick Characters, Document Images.

1. INTRODUCTION

The problem of image segmentation has received a considerable attention in the literature [13], [14]. In the context of new applications such as camera based document analysis [15] these methods prove inadequate. More specifically they are found failing when segmenting characters, of different font sizes, which can be encountered in real world or signboard images. For example, in a vehicle license plate image, a part of vehicle is also seen along with the license plate. There is a need to suppress other parts, retaining only digits on the license plate. We consider such segmentation as a kind of binarization task.

Binarization techniques are broadly classified into two categories: Global and Local. Global binarization techniques like Otsu's[1] perform thresholding based on an image histogram. It assumes two peaks and hence two classes, in the image histogram and chooses a threshold that maximizes between-class variance. Global thresholding techniques can, therefore, be used when the image background is simple. If the background is complex, i.e., image histogram has more than two peaks, Cheriet et al. [2] proposed a recursive approach for image segmentation by extending Otsu's algorithm. This algorithm applies Otsu's method recursively, eliminating brightest object at each iteration, until darkest object remains. Here, the object of interest must be darkest in the whole image. This method fails, when the image has low contrast or when the gray values of the object overlap

with those of the background. Local binarization techniques, on

the other hand, take into account local image statistics like mean, standard deviation etc. One of such locally adaptive thresholding methods is Niblack's[3], which computes a threshold for each pixel based on mean and standard deviation within a local neighbourhood region around that pixel. But, it strictly depends on neighbourhood region and the parameter used in the expression for evaluating the threshold. The values to be given to that parameters not known, since it depends on the complexity of the image. Bernsen [4] eliminates the dependence of the parameter in estimating the threshold. But, this method suffers from the "ghost" phenomenon as discussed in [5]. Edge based techniques can also be used to extract character objects. One of them [6] takes into account "Zero Crossings" property. This technique is extremely sensitive to noise or gray level variations within the character stroke, since this method tends to fill regions between character edges, by using second derivatives. This method is very much dependent on a threshold, that has to be chosen manually, to retain significant edges. These edge based techniques are used, after applying noise removal filters like Gaussian filter, median filter etc, mentioned in [6]. Again, the problem with noise removal filters is that they blur edges and remove small details. Hence, edge preserving noise removal filters like anisotropic diffusion filter [7], coplanar filter[8], were used. These techniques help little, since there are various kinds of noise and a noise removal filter generally does not remove all of the noise.

We observe that all the methods discussed above, fail, if pictures and graphical objects are also present in the image, apart from the character objects. Here, it appears more appropriate to use stroke width based binarization methods [9], [10], [11], which are intended to eliminate these other objects. Kamel and Zhao's logical level technique [9] and Djeziri et al.'s filiformity technique [10] fail, when there are closely spaced thick characters, as they do not use the gray level information inside the neighbourhood region around a pixel. Cheriet et al.'s method [11] uses that information and hence attempts to solve the problem of closely spaced thick characters. But, as we show later, it does not eliminate that problem completely.

In the present work, as a unified solution to the above mentioned problems, we propose two methods that can be used to eliminate objects greater than a predefined width, retaining only the character objects. For that, we exploit the property that character objects have a predefined range of stroke widths. We also focus on the problem of extracting closely separated thick characters. We apply our methods on several natural scene and signboard images. Results show that our first approach's behaviour towards closely spaced thick characters is similar to Cheriet et al.'s stroke

model based method [11] and superior to Kamel and Zhao's logical level technique [9] and Djeziri's filiformity techniques [10] and Our second approach shows superior results as compared to all the above methods.

The rest of the paper is organized as follows. In section 2, we discuss some of the stroke width based binarization methods and their limitations. In section 3, we present the two proposed approaches. In section 4, we show experimental results and we conclude the paper in section 5.

2. SOME STROKE WIDTH BASED BINARIZATION METHODS

In our paper, we have mainly focused on the following three stroke width based binarization methods: Kamel and Zhao's logical level technique [9], Djeziri et al.'s filiformity method [10] and Cheriet et al.'s stroke model based method [11]. All the methods assume dark characters on bright background. All these approaches try to capture features for text like objects. After evaluating features, they apply binarization on the resultant feature image. All these methods use different post-processing steps, to get a final binary image, from the feature image. To facilitate comparison among these methods, we have used Otsu method [1] as a post-processing step, for all the methods, including the proposed methods. That is, after getting a feature image, Otsu's method is applied on that image to obtain the binary image.

2.1 Logical Level Technique

In Kamel and Zhao's method [9], for each pixel p in the original image f, regions of size $(2W+1)\times(2W+1)$, which are W pixels away from it, are considered, on both sides of it, in the four possible directions around that pixel, to evaluate feature value for that pixel. Here, W refers to the width of the stroke to be extracted. Using these neighbourhood regions, feature values are calculated at each pixel using the following expressions:

$$L_d(p) = Min\{Avg_{q \in N_i}(f(q)), Avg_{q \in N_j}(f(q))\} - f(p)$$

$$(1)$$

$$LW(p) = Max_{d=0}^{3}\{L_d(p)\} \qquad (2)$$

Here, $d = 0,1,2,3$ refers to four directions $\{0, \pi/4, \pi/2, 3\pi/4\}$ respectively, that approximate most possible directions around a pixel. N_i and N_j are two $(2W+1)\times(2W+1)$ squares, whose centres are $2\times W$ pixels away from p on the opposite sides, in a particular direction. $f(p)$ is the gray value of f at pixel p. The neighbourhood regions around a pixel can be seen in Fig 1.
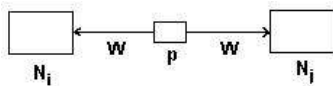

Figure 1: Neighbourhood regions around a pixel in Kamel and Zhao algorithm

Logic level technique(LLT) tends to break character strokes near darker neighbors as discussed by Oh [12]. To remove that anamoly, this method uses bright-average scheme, which uses only the maxima in columns and rows of a neighbourhood to compute average, instead of all the values in that neighbourhood.

2.2 Filiformity Technique

Djeziri et al.'s method[10] also considers the neighbourhood regions on both sides, in four directions and W pixels away from the current pixel. But, unlike Kamel and Zhao method, it takes maximum of gray values of pixels, instead of average, in the neighbourhood regions. The feature values in this method are evaluated using the following expressions:

$$S_{Dd} = min(M_1, M_2) - M_u \qquad (3)$$

$$g_r(p) = Max_{d=0}^{3}\{SDd\} \qquad (4)$$

Here also, $d = 0,1,2,3$ refers to four directions as given above, that approximate most possible directions around a pixel. M_1 and M_2 are maxima of gray value of the pixels on two sides of outside of the internal window, along the current direction given by d. M_u is the maximum of the gray values of pixels inside the internal window, in the orthogonal direction to the current direction. The internal window corresponds to width of the stroke. The neighbourhood around a pixel is shown in Fig.2.
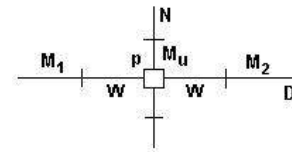

Figure 2: Neighbourhood region for the evaluation of Filiformity feature expression

2.3 Stroke Model Based Method

Unlike the previous two methods, this method[11] uses the pixel information inside the neighbourhood region, which can be observed in its feature expression given below:

$$DWd = Max_{i=1}^{W-1}\{Min(f(x-i), f(x+W-i))\} - f(x)$$

$$(5)$$

$$DEW(p) = Max_{d=0}^{3}\{DWd\} \qquad (6)$$

Here, as usual, $d = 0,1,2,3$ refers to same four directions as given above, that approximate most possible directions around a pixel. As usual, W refers to width of the stroke. D_{Wd} gives feature value at pixel p in the direction corresponding to d and $DE_{W}(p)$ gives final feature value assigned to pixel p.

2.4 Limitations of the previous methods

The feature expression for Kamel and Zhao's method [9] takes

more time to get evaluated, since it has to find average of $(2W + 1) \times (2W + 1)$ gray values especially for larger windows. In our experimentation, we observed that the usage of larger windows(to get thick characters), around a current pixel removes some of the character's boundary pixels, since the windows at the boundary pixels may fall on the surrounding characters, which affects feature values at those pixels. In this way, this method fails to extract character strokes when there are closely spaced thick characters.

Djeziri et al.'s method[10] eliminates the problem of time complexity, by reducing the size of neighbourhood region, to be considered around a pixel. But this method also fails, when there are closely spaced thick characters, due to the same reasons, as discussed above.

We have observed that Cheriet et al.'s method [11] eliminates the problem of closely spaced thick characters, to some extent. This is because, to evaluate feature value at each pixel, this considers pixels within the neighbourhood region. This is, in contrast to the above two methods, which completely ignores the inside of neighbourhood region and considers only pixels outside of the neigbourhood region, and hence fails when larger neighbourhoods have to be used, to extract thick characters. As we have observed in our experimentation, Cheriet et al.'s method does not eliminate the problem of closely spaced thick characters completely, as shown in fig. 4(e), where character **T** has lost some of its pixels to the background, the reason for which is discussed in the later sections. We have extended the Cheriet et al.'s method[11] to overcome its limitations. For that, we have proposed two approaches which are discussed in the next section.

3. PROPOSED METHODS
3.1 Basic Idea
Given the limitations of the previous methods, we have proposed two methods to evaluate features that are modelled to extract character objects. The basic idea behind these two approaches is the same. Since any character has some width W, any pixel in that character should encounter a surrounding background pixel, up to W pixels away from it. If the gray value of a character pixel is subtracted from its surrounding background pixel, the resultant value is positive, since we are assuming darker characters on brighter background. If the same subtraction operation is applied to a background pixel, it may get negative values or smaller positive values, since the background is a larger homogeneous region and the surrounding pixels have similar gray values. Of course, background pixels with negative values can be eliminated from further analysis. This property can be used to distinguish between character and background pixels. Therefore, character pixels get larger positive values and background pixels get smaller positive values and the corresponding feature image exhibits a histogram with two peaks, with the first peak corresponding to background and the second one corresponding to the character objects. Now, character objects can be isolated using simple global thresholding method like Otsu [1], since the feature image obtained shows a greater bimodality of the histogram.

But there are some important points that should be considered to design a feature expression. First one is that the direction, in which, the curent pixel has the background(i.e., direction of the stroke at that pixel)is not known and so the above mentioned subtraction operation should be performed in the four directions $\{0, \pi/4, \pi/2, 3\pi/4\}$, which approximate all the possible directions around a pixel in a digital image. Then, maximum of those four responses should be retained at that pixel in the feature image. Because, maximum response for a character pixel, is obviously obtained in the orthogonal direction of stroke, at that pixel.

Second point to be considered is that a character pixel has background pixels on both sides of it, in the direction orthogonal to the stroke direction at that pixel. Finally, width of a character is not uniform, in its entirety or there may be characters of different sizes in the original image. Given the maximum width of a character in the image, feature expression should be able to represent all the character objects of width less than or equal to that maximum width. We have designed our feature expressions by exploiting all these points.

For both of our approaches, after getting feature image, we have applied Otsu's method [1] on it, to get final binary image. Both of our approaches consider pixels inside the neighbourhood region, around the current pixel,to evaluate feature value at that pixel and hence addressed the problem of closely spaced thick characters. We have observed in our experimentation that our first approach is showing similar behaviour, as that of Cheriet et al.'s method, and its our second approach that is more robust to closely spaced thick characters and giving better results than all the other methods.

3.2 First Approach
The motivation behind going for this approach is that even though Cheriet et al.'s method is considering the pixels inside the neighbourhood region, it sometimes fails to to pick up the surrounding background pixel, for a character pixel. The reason for its failure, is that it is not searching for a background
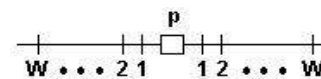

Figure 3: Neighbourhood region for the evaluation of Proposed feature expressions

pixel symmetrically, on both sides in a particular direction, that can be seen from (5). If two stroke patterns occur periodically, the way it searches captures the surrounding character pixel, instead of background pixel especially if larger neighbourhoods are used. It is that search with a periodicity of W units, which loses some of the pixels of character **T** in fig. 4(e) to the background. To account for that, our first approach tries to seek a background pixel, for a character pixel using symmetric search as given by (7). After computing feature values for all the pixels in the original image, Otsu method [1] is applied on the resultant feature image to get the final binary image. The neighbourhood around a pixel for both of our approaches is shown in Fig.3.

$$FWd = \underset{i=1}{\overset{W}{Max}}\{Min(f(x-i), f(x+i))\} - f(x) \qquad (7)$$

$$FE1W(p) = \underset{d=0}{\overset{3}{Max}}\{FWd(x)\} \qquad (8)$$

As usual, $d = 0,1,2,3$ refers to four directions given earlier, that approximate most possible directions around a pixel in a digital image. W gives the width of the stroke. The above expression gives the estimation of whether the pixel p belongs to the stroke or the background. F_{Wd} gives feature values in the direction corresponding to d. $f(x)$ is original image in 1-D, the direction of which is given by d, with respect to pixel p. $FE1_W(p)$ is the feature value finally assigned to pixel p, which is taken as a maximum of F_{Wd}, for $d = 0,1,2,3$. By observing the above expression, the positive values of feature expression indicate the existence of a darker stroke on brighter background and that stroke has width less than or equal to W. Pixels with negative values may correspond to other objects like pictures, graphics etc, which can be eliminated from further analysis. But smaller positive values may also occur due to sharply varying backgrounds, as discussed earlier. For the reasons mentioned above, we have chosen Otsu[1] to perform global thresholding on the feature image. Hence, our method first applies feature expression on the original image to get feature image. Later, Otsu method is applied on feature image to get final binarized output.

This approach has shown better behaviour on closely spaced thick characters than that of Kamel and Zhao's technique and Djeziri et al.'s method, and similar behaviour to that of Cheriet et al.'s method as shown in Fig.4. In Fig.4(a), we can see that Otsu's[1] is picking up the entire background and urges the need to go for stroke width based binarization methods. Kamel and zhao's (Fig.4(c)) and Djeziri et al.'s (Fig.4(d)) methods have lost most of the character pixels to the background. It can be observed that our first approach (Fig.4(f)) has no breaks in character **T**. But it has lost some boundary pixels of character **U** to the background. This motivated us to revise our first feature expression, to focus further on the problem of closely spaced thick characters, which we have handled by our second approach to a greater extent.

3.3 Second Approach

Eventhough, our first approach is looking for a surrounding background pixel symmetrically, it fails when a boundary character pixel has some surrounding character pixel at a distance less than the given stroke width, in all the possible four directions around it. Due to this, it gets negative or smaller positive value and hence it is marked as background by Otsu method. We have found that this phenomenon is happening because we are searching for background pixels on both sides of a character pixel simultaneously. As a solution to the above problem, we have decoupled the search for a background pixel on both sides. Our second approach looks for the background pixels of a character pixel separately, as our final intention is to see that a character pixel should have background on both sides in the direction orthogonal to the direction of stroke at that pixel. Finally, we have arrived at the following feature expression, which takes care of all the above mentioned limitations.

$$F1d = \underset{i=1}{\overset{W}{Max}}\{f(x+i)\} \qquad (9)$$

$$F2d = \underset{i=1}{\overset{W}{Max}}\{f(x-i)\} \qquad (10)$$

$$FWd = Min(F1d, F2d) - f(x) \qquad (11)$$

$$FE2W(p) = \underset{d=0}{\overset{3}{Max}}\{FWd\} \qquad (12)$$

Our second method also first applies feature expression on the original image to get feature image. Later, Otsu's method is applied on feature image to get final binarized output. Again, F_{Wd} gives feature values in the direction corresponding to d. $f(x)$ is original image in 1-D, the direction of which is given by d, with respect to pixel p. $FE2_W(p)$ is the feature value assigned to pixel p, which is taken as a maximum of F_{Wd}, for $d = 0,1,2,3$. It is the maximum operations used in evaluating F_{1d} and F_{2d}, that looks for the surrounding background pixels on both sides of a character pixel separately, in the direction corresponding to d. That is, search for the background pixels on both sides of a character pixel has been decoupled using F_{1d} and F_{2d}. Even, if larger neighbourhoods are taken around a pixel, so that the neighbourhood regions fall on the adjacent characters, that maximum operation still picks up its surrounding background pixel, from the gray value of which, current pixel gray value should be subtracted. The minimum operation used in (11), is used to find out whether a character pixel has a surrounding background pixel on both sides of it. In this sense, this method is more robust than any of the previous methods including our first approach, to the problem of closely spaced thick characters. This effect can be immediately observed from fig.4(g), where none of the character pixels are lost to the background.

4. EXPERIMENTAL RESULTS AND DISCUSSION

The first example we consider is a sign board image which is shown in Fig.5. We can see that Kamel and Zhao's and Filiformity method outputs have lost a lot of character pixels to the background. In stroke method, the characters **Q** and **A** have some breaks, which the proposed method outputs do not have. Another interesting natural scene image containing closely spaced thick characters is shown in Fig.6. We can observe breaks in the characters of all the outputs except in that of our second approach.
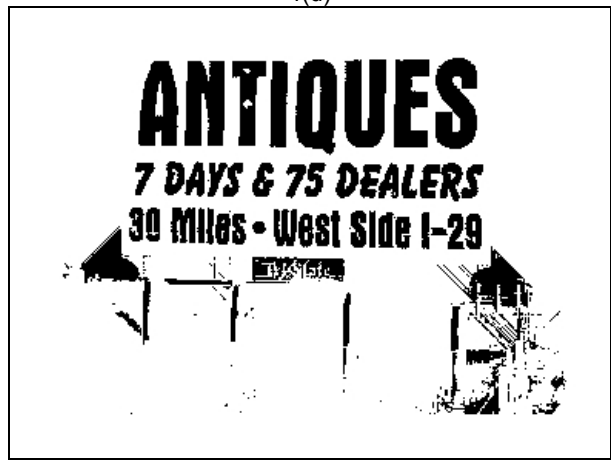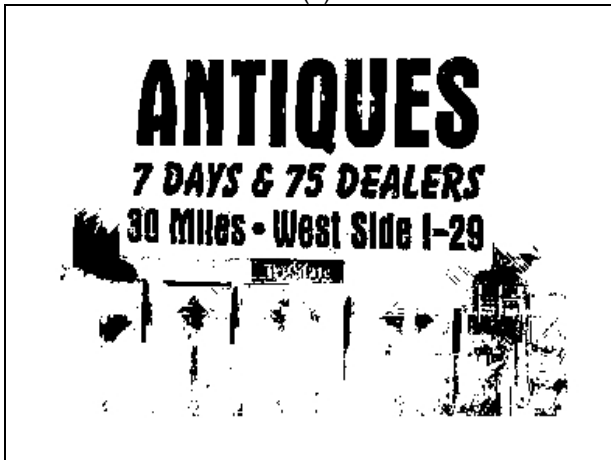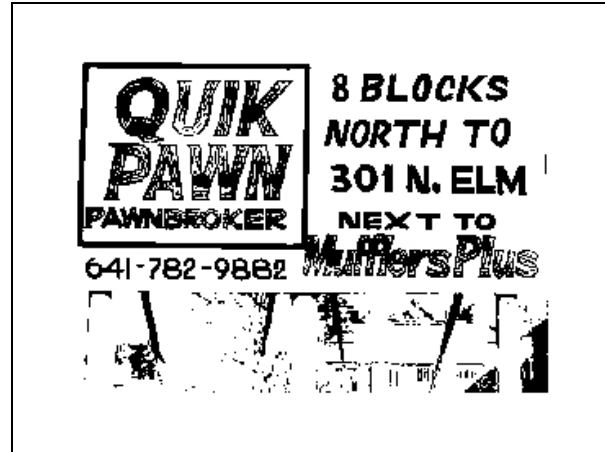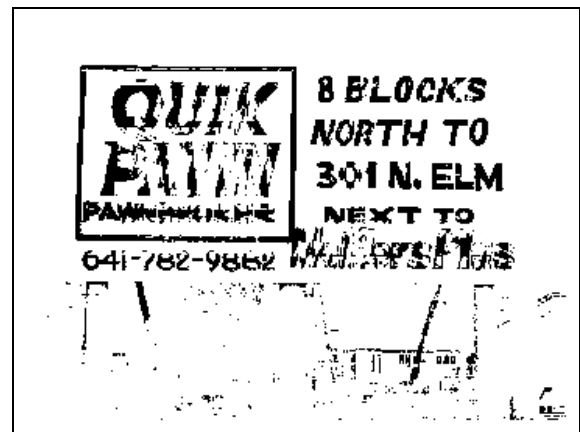
4(a)

4(b)
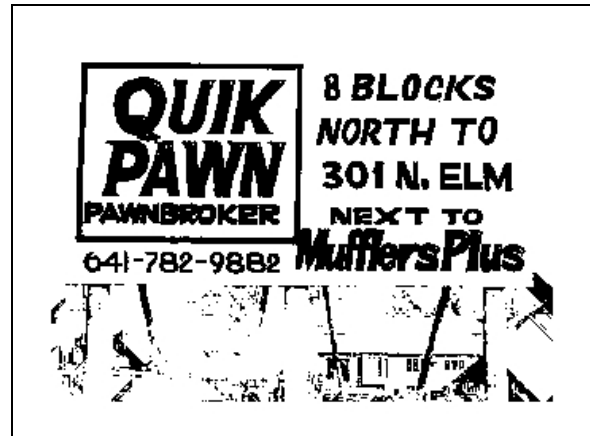

4(c)

4(d)

4(e)

4(f)

4(g)
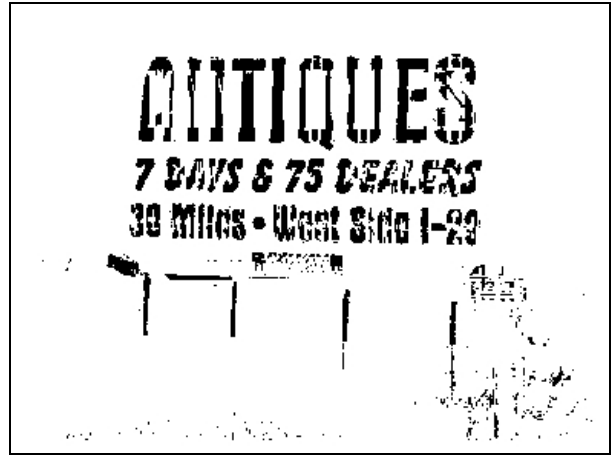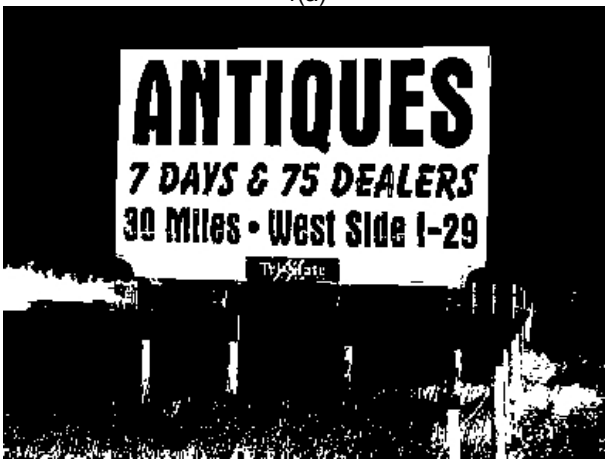
Figure 4: (a) Original Image (b) Otsu Output (c) Kamel and Zhao Output (d) Filiformity output (e) Stroke method output (f) First Proposed method output (g) Second Proposed method output
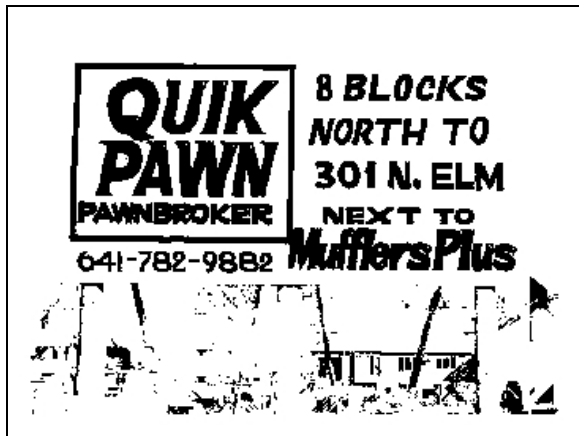

.   5(a)

5(b)

5(c)

5(d)

5(e)

5(f)

Figure 5: A Signboard image (a) Original image (b) Kamel and Zhao output (c) Filiformity output (d) Stroke method output (e) First Proposed method output (f) Second Proposed method output

6(a)

6(b)

6(c)

6(d)

6(e)

6(f)

Figure 6: A Signboard image (a) Original image (b) Kamel and Zhao output (c) Filiformity output (d) Stroke method output(e) First Proposed method output (f) Second Proposed method output

5. CONCLUSION AND FUTURE WORK

In the present work, we have proposed two stroke width based binarization methods. These methods are designed to suppress pictures and graphical objects, commonly seen in real world and signboard images and at the same time, retain the character objects of different stroke widths. We have shown that the existing stroke width based binarization methods fail, when there are closely spaced thick characters. Experimental results have shown that our second approach handles the problem of closely spaced thick characters better than any of the existing methods. As a future work, the proposed methods, especially, the second approach, can be applied to other domains like automatic bank check processing, document image processing etc.

6. ACKNOWLEDGMENT

7. REFERENCES

[1] N. Otsu, "A threshold selection method from grey level histograms", IEEE Trans. Syst., Man,Cybern., SMC-8, pp. 62-66, 1978.

[2] M. Cheriet and J.N. Said and C.Y. Suen, "A recursive thresholding technique for image segmentation", IEEE Trans. Image Processing, vol. 7, pp.918-921, June 1998.

[3] W. Niblack, An Introduction to Digital Image Processing, Prentice Hall, 1986.

[4] J. Bernsen, "Dynamic thresholding of gray level images", Proc. 8th Int. Conf. on Pattern Recognition, vol. 2, pp. 1251-1255, 1986.

[5] Q.D. Trier and A. K. Jain, "Goal-directed evaluation of binarization methods", IEEE Trans. Pattern Anal.Machine Intell., vol. 17, pp. 1191-1201, 1995.

[6] Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing, Pearson Education Indian Reprint, 2nd edition, 2003.

[7] Pietro Perona and Jitendra Malik, "Scale space and edge detection using anisotropic diffusion", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 7, pp. 629-639, July 1990.

[8] Liying Fan, Chew Lim Tan, Lixin Fan, "Edge preserving prefiltering for document image binarization", Proc. ICIP 01,IEEE Computer Society Press, pp. 1070-1073, 2001.

[9] M. Kamel and A. Zhao, "Extraction of binary character/graphics images from gray scale document images", CVGIP:Graph Models Image Process., vol. 55, no. 3, pp. 203-217, 1993.

[10] S. Djeziri, F. Nouboud, R. Plamondon, "Extraction of signatures from check background based on filiformity criterion", IEEE Transactions on Image Processing, vol. 7, pp. 1425-1438, October 1998.

[11] Xiangyun Ye, Mohamed Cheriet, Ching Y. Suen, "Stroke-Model-Based Character Extraction from Gray-Level Document Images", IEEE Transactions on Image Processing, vol. 10, no. 8, August 2001.

[12] I.-S. Oh, "Document image binarization preserving stroke connectivity", Pattern Recognition Letters, vol. 16, pp. 743-748, 1995.

[13] P. K. Sahoo, S. Soltani, and A. K. C. Wong, "SURVEY: A survey of thresholding techniques", Comput. Vis. Graph. Image Process., vol. 41, pp. 233-260, 1988.

[14] N. R. Pal and S. Pal, "A review on image segmentation techniques", Pattern Recognit., vol. 26, pp. 1277-1294, 1993.

[15] Second International Workshop on Camera-Based Document Analysis and Recognition (CBDAR2007) September 22, 2007 Grand Hotel Rayon, Curitiba, Brazil. "http://www.m.cs.osakafu-u.ac.jp/cbdar2007/demo.shtml".