Published in Proceedings of International Conference on Image Processing ICIP'98

ADAPTIVE DOCUMENT IMAGE THRESHOLDING USING FOREGROUND AND BACKGROUND CLUSTERING

Andreas E. Savakis

Eastman Kodak Company
901 Elmgrove Rd.
Rochester, New York 14653
savakis@kodak.com

ABSTRACT

Two algorithms for document image thresholding are presented, that are suitable for scanning document images at high-speed. They are designed to operate on a portion of the image while scanning the document, thus, they fit a pipeline architecture and lend themselves to real-time implementation. The first algorithm is based on adaptive thresholding and uses local edge information to switch between global thresholding and adaptive local thresholding determined from the statistics of a local image window. The second thresholding algorithm is based on tracking the foreground and background levels using clustering based on a variant of the K-means algorithm. The two approaches may be used independently or may be combined for improved performance. Results are presented illustrating the algorithms' performance for document and pictorial images.

1. INTRODUCTION

High-speed scanners used in production scanning of document images may process over one hundred pages per minute. The speed and performance requirements imposed on these systems dictate the use of dedicated hardware for image processing, and require algorithms that are not only effective, but also efficient and lend themselves to real-time implementation. Document images are typically captured in gray scale (eight bits per pixel) by a linear array charge-coupled device and are converted to binary (one bit per pixel) output images. In most cases, the documents consist of text or line graphics on a relatively uniform background, thus, converting them to binary form is suitable for output and storage, because it significantly reduces file size and transfer bandwidth requirements without loss of important document information.

Given the system speed and cost requirements, a pipeline architecture implementation of image processing algorithms may be used, where only a small part of the image is kept in memory and is processed, while another part of the image is read by the scanner camera. This minimizes the cost of memory while maintaining high system throughput. However, memory restrictions and scanning speed requirements preclude the use of binarization algorithms that require *a priori* knowledge of the full image, thus, a number of well-known algorithms cannot be used.

Image binarization algorithms are often geared for use with pictorial instead of document images. For example, halftoning algorithms, such as dithering and error diffusion [1], are well suited for processing pictorial images because they are designed to effectively represent shades of gray. However, they are not optimal for processing document images, because they produce broken characters that are difficult to interpret. Thresholding algorithms are better suited for separating characters and graphics from backgrounds, and often enhance the appearance of low contrast text.

Previous work on image thresholding is described in a number of papers and their references [2-7]. Generally, the methods may be broadly categorized in techniques employing the histogram, moment preserving, entropy coding, and locally adaptive methods. In high-speed scanning, implementation constraints prevent the use of methods that require full access to the entire image, such as global and iterative methods that use the histogram or statistics of the whole image. The types of algorithms that are better suited for high-speed scanners are locally adaptive thresholding methods, e.g. [8-10].

In this paper, two algorithms for document image thresholding are presented that operate within the above constraints and lend themselves to real-time implementation. In the next section, the adaptive threshold processor is presented, followed by the background tracking algorithm based on foreground and background clustering (FBC). These two algorithms may be used independently or jointly. Results are presented which illustrate the relative merits of the two algorithms.

2. ADAPTIVE THRESHOLD PROCESSOR

The adaptive threshold processor (ATP) algorithm was developed by Lee [11]. The operation of ATP is based on switching between global and local thresholding. Global thresholding is used on the relatively uniform background regions and is based on a fixed user-supplied threshold. Local thresholding is used near intensity transitions that are due to lines or characters in the document.

Switching between local and global thresholding depends on whether the processed pixel is on the background region or near an edge transition. To determine whether a pixel lies near an edge transition, smoothed local gradient information is compared to a user supplied gradient threshold G. The smoothed local gradient L is computed by averaging the results of the Sobel edge detector over a local window centered at the pixel of interest. The size of the local window depends on the computational constraints, memory limitations, and the amount of the noise present in the scanned documents. If $L>G$, then it is assumed that the pixel is located on an edge neighborhood, and the locally adaptive threshold is used. Otherwise, the fixed global threshold is used.

The locally adaptive threshold is computed as the average of the maximum and minimum value in a second local window centered at the pixel of interest. The size of the second local window is slightly larger than the size of the first local window. Assuming that the window size is chosen, ATP requires two user inputs, the global threshold and the gradient threshold.

3. ADAPTIVE THRESHOLDING WITH FOREGROUND AND BACKGROUND CLUSTERING

In the foreground and background clustering (FBC) approach to document image binarization, each pixel is assigned to a foreground cluster or a background cluster. In cases involving typical document images, the foreground consists of characters and graphics and is fairly well separated from the background. The relative intensity values of the foreground and background clusters is not important. Pixel clustering is based on a variant of the K-means algorithm due to McQueen [12], where the cluster means are updated each time a data point is assigned to a cluster. Since only one background and one foreground are assumed, K=2, i.e. only two clusters are considered, which makes the overall implementation easier.

The following steps describe the way this approach works:

Step 1. Region selection

Divide the document into all inclusive mutually exclusive subregions. Select the document subregion for which the threshold will be computed, and a region containing the subregion that will be used to determine the threshold subregion. For example, the region may consist of N contiguous scanlines, where the subregion is the center M scanlines, with $M<N$.

Step 2. Initialization

Initialize the background cluster mean and the foreground cluster mean to be the same as computed for the previous subregion. If there is no previous subregion, set the two initial cluster means with a large separation between them.

For each pixel inside the region, iterate between steps 3 and 4:

Step 3. Pixel Assignment

Assign each pixel to the nearest cluster.

Step 4. Cluster Mean Update

After each new pixel assignment update the relevant cluster mean.

Step 5. Threshold Calculation

After all pixels in the region have been assigned , set the threshold for the subregion equal to the average of the foreground and background cluster means.

The size of the region and subregion determine the amount of memory that is necessary, the speed of processing, and the adaptivity to local image variations.

It is possible to combine the ATP and FBC thresholding approaches for improved results. This is accomplished by replacing the fixed threshold mode of the ATP with the threshold computed by FBC thresholding.

4. RESULTS

The ATP and FBC algorithms have been tested for a variety of images. An example of thresholding a document image using ATP is shown in Figure 1. The original image is shown in Figure 1a. The binary images obtained by fixed thresholding and ATP are shown in Figures 1b and 1c respectively. As expected, ATP outperforms fixed thresholding, because it adapts to local image properties. The ATP adaptivity regions are illustrated in Figure 1d, where black corresponds to pixels where the global fixed threshold was used, and

white corresponds to pixels where the locally adaptive threshold was used. It is clear that the local threshold is used near sharp intensity transitions, as a result of comparing the smoothed gradient to the gradient threshold. Figure 1e shows the result obtained by FBC thresholding.

In general, ATP is better than FBC at extracting low contrast characters from the background, but it is also more sensitive to noise. On the other hand, FBC is better at tracking the background level, and does not require that the user set any threshold values.

The advantages of ATP and FBC may be combined, by using the former as a method for capturing character information and the latter as a method for tracking the background level. By using FBC to automatically track the background, the fixed threshold input to ATP is eliminated. Figure 1f shows the result obtained by the combination of ATP and FBC, by using FBC to compute the global threshold input to the ATP method. This makes the overall system more user friendly, and minimizes errors that may result due to improper setting of the fixed threshold parameter.

Both ATP and FBC are designed to operate on document images and are not ideally suited for binarizing pictorial images. For completeness, an example of processing a pictorial image is shown in Figure 2. This example illustrates that FBC is better for thresholding pictorial images, while ATP generates edge artifacts.

In some cases, compound images consisting of text regions and pictorial regions need to be processed. It is preferable in these cases to use image segmentation and process each type of region separately using a suitable algorithm, such as adaptive thresholding for text regions and error diffusion for pictorial images.

Acknowledgments

I would like to express my appreciation to Y. Lee and P. Rudak of Eastman Kodak Company for their helpful comments and support during the course of this research.

REFERENCES

[1] R. Ulichney, "Digital Halftoning," MIT Press, Cambridge MA, 1987.

[2] J. Weszka and A. Rosenfeld, "Threshold Evaluation Techniques," *IEEE Trans. Systems Man and Cybernetics*, pp. 622-629, 1978.

[3] P. Palumbo, P. Swaminathan, and S. Srihari, "Document Image Binarization: Evaluation of Algorithms," *SPIE Applications of Digital Image Processing IX*, vol. 697, pp. 278-285, 1986.

[4] P.K. Sahoo, S. Soltani, A.K.C. Wong, and Y.C. Chen, "A survey of thresholding techniques," *Computer Vision, Graphics and Image Processing*, vol. 41, pp. 233-260, 1988.

[5] M. Kamel and A. Zhao, "Extraction of binary Character/Graphics images from grayscale document images," *CVGIP: Graphical Models and Image Processing*, vol. 55, pp. 203-217, 1993.

[6] O.D. Trier and A. Jain, "Goal-directed evaluation of binarization methods," *IEEE Trans. PAMI*, pp. 1191-1201, Dec. 1995.

[7] A.T. Abak, U. Baris, and B. Sankur, "The Performance Evaluation of Thresholding Algorithms for Optical Character Recognition," pp. 697-700, *ICDAR 97*, Ulm, Germany, 1997.

[8] J.D. Yang, Y.S. Chen, and W.H. Hsu, "Adaptive thresholding algorithm and its hardware implementation," *Pattern Recognition Letters*, pp. 141-150, 1994.

[9] W. Niblack, "An Introduction to Digital Image Processing," pp. 115-116, Prentice Hall, 1986.

[10] J. Sauvola, T. Seppanen, S. Haapakoski, and M. Pietikainen, "Adaptive Document Binarization," pp. 147-152, *ICDAR 97*, Ulm, Germany, 1997.

[11] Y. Lee, J. Basile, and P. Rudak, "Multi-windowing technique for thresholding an image using local image properties," US Patent No. 5,583,659, Eastman Kodak Company.

[12] A. Jain, and R. Dubes, *Algorithms for Clustering Data*, pp. 96-101, Prentice Hall, 1988.

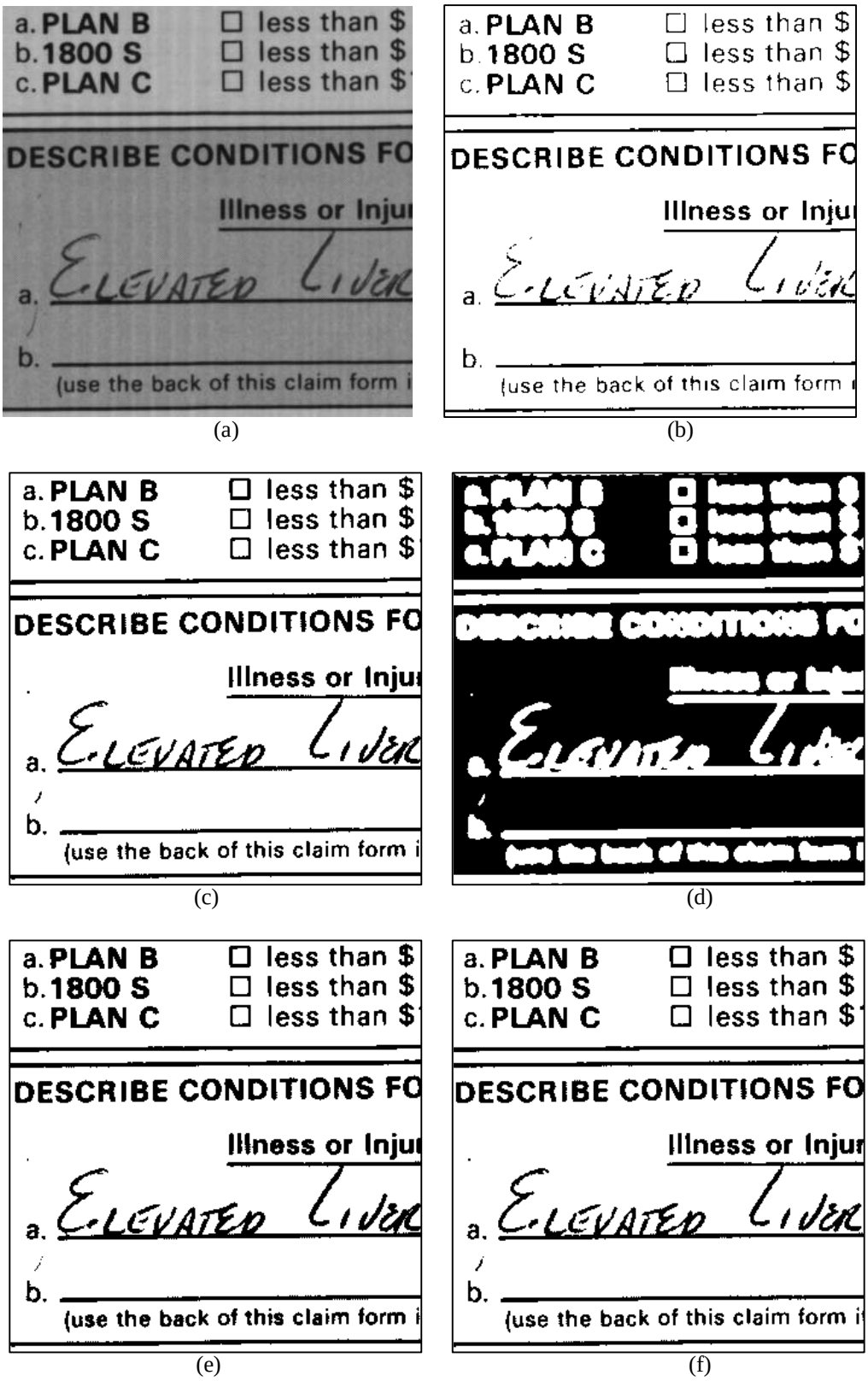

Figure 1: (a) Original gray scale image; Thresholded image region using: (b) fixed threshold; (c) ATP; (d) Adaptivity regions for ATP; Thresholded image region using: (e) FBC; (f) Combination of FBC and ATP.

Figure 3: Top left:Original; Top Right: Thresholded with fixed threshold.
Bottom left: Thresholded with ATP; Bottom Right: Thresholded with FBC.