

# Goal-Directed Evaluation of Binarization Methods

Øivind Due Trier, *Student Member, IEEE*, and Anil K. Jain, *Fellow, IEEE*

**Abstract**—This paper presents a methodology for evaluation of low-level image analysis methods, using binarization (two-level thresholding) as an example. Binarization of scanned gray scale images is the first step in most document image analysis systems. Selection of an appropriate binarization method for an input image domain is a difficult problem. Typically, a human expert evaluates the binarized images according to his/her visual criteria. However, to conduct an objective evaluation, one needs to investigate how well the subsequent image analysis steps will perform on the binarized image. We call this approach *goal-directed evaluation*, and it can be used to evaluate other low-level image processing methods as well. Our evaluation of binarization methods is in the context of digit recognition, so we define the performance of the character recognition module as the objective measure. Eleven different locally adaptive binarization methods were evaluated, and Niblack's method gave the best performance.

**Index Terms**—Objective evaluation, performance evaluation, binarization, segmentation, document images.

## I. INTRODUCTION

THERE is currently a substantial and growing interest in the field of document image analysis (*or document image understanding*). Several research groups are designing systems for extracting relevant information from documents as diverse as engineering drawings, maps, magazines and newspapers, forms, and mail envelopes [1], [2], [3], [4], [5], [6], [7], [8].

One of the most popular architectures preferred in designing large and complex vision systems is the sequential, modular, and feedforward processing of information. Typically, a given component module in the system takes an intermediate output of another module and transforms it into representations which make useful information more explicit. Among the range of off-the-shelf module alternatives, a system designer is confronted with the formidable problem of selecting the 'correct' module to obtain the desired solution for a given application at hand. As more and more computer vision systems are being developed, there is an intense need for a systematic characterization and evaluation of the performance of these systems. This paper is an attempt to explore an objective evaluation of suitability of a given system component for a practical vision system.

Manuscript received Aug. 26, 1994. Recommended for acceptance by M. Mohiuddin.

A preliminary version of this paper appeared in *Proceedings of NSF/ARPA Workshop on Performance Versus Methodology in Computer Vision*, pp. 206-217, Seattle, Wash., June 24-25, 1994.

Ø.D. Trier is with the Department of Informatics, University of Oslo, P.O. Box 1080 Blindern, N-0316 Oslo, Norway; e-mail: trier@ifi.uio.no. This work was done while Trier was visiting Michigan State University.

A.K. Jain is with the Department of Computer Science, Michigan State University, A726 Wells Hall, East Lansing, MI 48824-1027; e-mail: jain@cps.msu.edu.

To order reprints of this article, e-mail: transactions@computer.org, and reference IEEECS Log Number P95138.

Performance evaluation of low-level image processing routines, such as binarization, segmentation, edge-detection, thinning, etc., is inherently difficult. The usual approach is to define a set of criteria, and give scores for each criterion. Different weights may be assigned to the individual criteria. The criteria may be human visual criteria, or machine-computed criteria. Recent performance evaluations for binarization methods [9] and thinning [10] follow this approach. These efforts are indeed valuable as they sort out the good methods from the poor. However, the main drawback of these evaluation methods is that for a given application domain, where a low-level image processing routine is to be used, one can not determine from the subjective evaluation which routine will perform the best. The low-level module used may not conform to the requirements of the higher level image understanding module. Also, for complex images, the design of low-level operations such as binarization and thinning is a difficult problem. This calls for *goal-directed* evaluation, where an image understanding module using the results of the low-level image processing routine in question is used for quantitative evaluation. This avoids the use of subjective criteria, the problem of assigning weights to the different criteria, and determining which criteria are important and which are not.

Our evaluation methodology is not necessarily new and has been used to evaluate, for example, feature extraction methods in an optical character recognition (OCR) system [11]. Our contribution is to provide a formal framework for this evaluation methodology for low-level vision modules, and demonstrate its usefulness in a particular application.

Note that to evaluate feature extraction methods for OCR, the goal-directed evaluation method, in our opinion, is the *only* sensible scheme. For low-level vision modules, however, the goal-directed evaluation method is an alternative to other evaluation methods. Also, note that for some computer vision systems a goal-directed evaluation of the modules may be difficult to conduct, for example, evaluation of edge detectors for a vision-guided mobile robot. The difficulty lies in constructing an unbiased, objective measurement of the system performance.

Haralick [12] has proposed a scientific framework for performance characterization of image analysis methods, and Jaisimha et al. [13] used this methodology to evaluate thinning algorithms. However, Haralick's evaluation methodology [12] is based on using synthetic images, which may fail to give valid results for real images if the assumptions about the image formation and degradation models do not hold. Therefore, the goal-directed approach is a more appropriate evaluation procedure.

In most document image analysis systems, the scanned gray level image is binarized (labeling each pixel as print or back-

ground) prior to further processing. This paper will focus on performance evaluation of binarization methods for map images. Maps are central to a large number of public and private organizations. An example is hydrographic maps, which are hand-drawn maps of the coast and seabed, indicating the depth values and contours of constant depth (Fig. 1). These maps need to be digitized for use in computer-aided navigation systems in order for ships to operate safely. The depth values (numbers) represent extremely valuable information as it is very expensive to collect this data. Manual entry of the depth numbers and their positions is a tedious, time-consuming, and error-prone task. Therefore, it is essential to develop computer systems which are capable of automatically capturing as many of the depth numbers in these maps as possible.

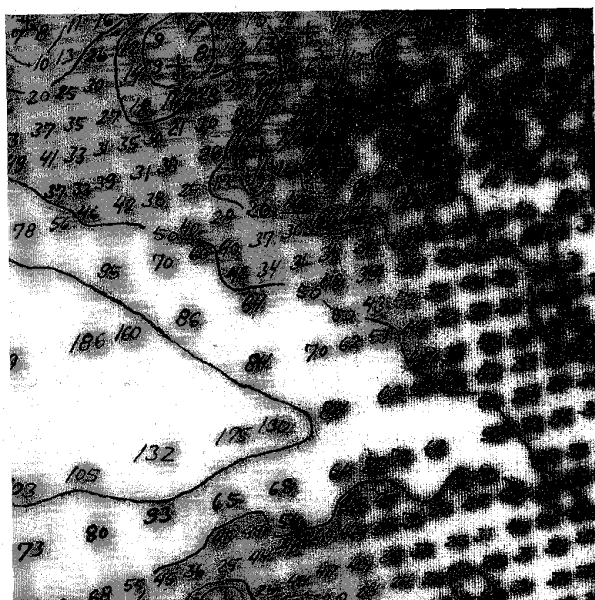


Fig. 1. A part of a hydrographic map, showing depth values and contours of constant depth. The  $2,000 \times 2,000$  pixels subimage shown here represents a  $2 \times 2$  inch<sup>2</sup> area on the paper map. The size of a typical paper map is in the range 1000–2000 inch<sup>2</sup>.

We utilize an experimental OCR module for this study. This OCR module, like most commercially available systems, requires binary images as input. Experiments have shown that when hydrographic maps are scanned binary, quite a few numerals in some areas are fragmented, while in other areas they are connected to each other or to contour lines or grid lines in the map, making the recognition task very difficult. Thus, gray level scanning, followed by a locally adaptive binarization using the best available method, is necessary.

This application domain provides an excellent setting for evaluating the large number of locally adaptive binarization methods available in the literature. The 11 most promising locally adaptive binarization methods we could identify were applied to the test images. The resulting binary images were fed into the OCR module, and the results were quantified in terms of misclassification rates and reject rates.

One should keep in mind that binarization is only one of several steps in a digit recognition system (Fig. 2). To design the best possible recognition system, one should optimize or tune the other steps involved as well. The quality of the scanned image depends on the resolution used, quality of the scanner hardware, and calibration of the scanner. A wide range of feature extraction methods exist [14]. Similarly, a variety of classifiers may be used, including statistical [15] as well as syntactical [16]. Besides, factors such as memory requirements and execution times may limit the permissible solutions. So, the task of designing the best possible digit recognition system is indeed challenging.

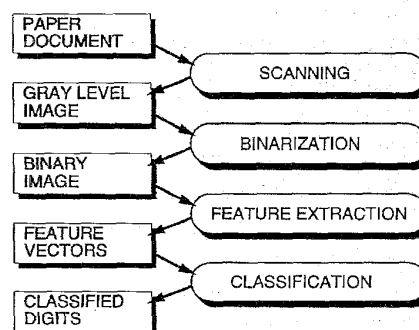


Fig. 2. Steps in a digit recognition system.

The rest of the paper is organized as follows. Section II briefly describes the binarization methods used in this evaluation. Section III describes the design of our experiments, with a description of the test images, binarization procedure, and details of the classifier. The results are described in Section IV, and compared with the results from the evaluation study of Trier and Taxt [9], where five different human visual criteria were used. The last section contains discussion and an outline of future work.

## II. BINARIZATION METHODS

*Global* binarization methods calculate a single *threshold* value for the entire image. Pixels having a gray level darker than the threshold value are labeled print (black), otherwise background (white).

*Locally adaptive* binarization methods, on the other hand, compute a threshold for each pixel on the basis of information contained in a neighborhood of the pixel. Some of the methods calculate a threshold surface over the entire image. If a pixel  $(x, y)$  in the input image has a higher gray level than the threshold surface evaluated at  $(x, y)$ , then the pixel  $(x, y)$  is labeled as background, otherwise it is labeled as print. Other methods do not use explicit thresholds, but search for print pixels in a transformed image.

We tested the following 11 locally adaptive binarization methods:

- 1) Bernsen's method [17],
- 2) Chow and Kaneko's method [18], [19],
- 3) Eikvil et al.'s method [20],

- 4) Mardia and Hainsworth's method [21],
- 5) Niblack's method [22],
- 6) Taxt et al.'s method [23],
- 7) Yanowitz and Bruckstein's method [24],
- 8) White and Rohrer's Dynamic Threshold Algorithm [25],
- 9) Parker's method [26],
- 10) White and Rohrer's Integrated Function Algorithm [25],  
and
- 11) Trier and Taxt's method [27].

These methods were selected because they have been frequently referred to in the literature, or appeared to be promising. The first eight methods use explicit thresholds or threshold surfaces, while the last three methods search for print pixels after having located the edges. All these methods are briefly described below.

To demonstrate that global binarization was inadequate to threshold the map images properly, we also tested four global binarization methods:

- 1) Abutaleb's method [28],
- 2) Kapur et al.'s method [29],
- 3) Kittler and Illingworth's method [30], and
- 4) Otsu's method [31].

The latter two are known to be optimal under two different normality assumptions [32]. Cho et al.'s improvement of Kittler and Illingworth's method [33] was not used. Abutaleb's method seemed to be the most promising global method, since it uses some local information to generate an extra feature for each pixel, followed by a two-dimensional thresholding to classify the pixels.

There is a relationship between the window size used in a locally adaptive binarization method and the size of the objects of interest in the image. If a too small window is used, then it will sometimes be positioned totally within the character and line strokes, and the binarization method may falsely label true print pixels as background. With a large a window, the method will behave too much like a global thresholding method.

Our experience is that if an appropriate resolution is chosen when scanning the images, the window size parameter appears to be fairly invariant for a given method. Further, slight modifications of the window size, or the other parameters for that matter, seem to have limited effect on the binarization result. Still, the parameters may need to be tuned if the binarization methods are to be used in a different application domain, for example, as preprocessing in medical image analysis.

#### A. Bernsen's Method

For each pixel  $(x, y)$ , the threshold  $T(x, y) = (Z_{low} + Z_{high})/2$  is used, where  $Z_{low}$  and  $Z_{high}$  are the lowest and highest gray level pixel values in a square  $r \times r$  neighborhood centered at  $(x, y)$ . However, if the contrast measure  $C(x, y) = (Z_{high} - Z_{low}) < \ell$ , then the neighborhood consists only of one class, print or background. In our images, wide print areas rarely occur, so the pixel is labeled background in such cases. We found  $\ell = 15$  and  $r = 15$  to be good choices.

#### B. Chow and Kaneko's Method

We used Nakagawa and Rosenfeld's implementation [19] of this algorithm. This method calculates a thresholding surface. The image is divided into non-overlapping windows, and the histograms for each window are tested for bimodality. As part of the bimodality test, each histogram is approximated by a mixture of two Gaussian distributions. For each bimodal window, a threshold is calculated based on the estimated parameters ( $\hat{\mu}_1$ ,  $\hat{\mu}_2$ ,  $\hat{\sigma}_1$ , and  $\hat{\sigma}_2$ ) of the bimodal mixture distribution. These thresholds are interpolated to give thresholds for the unimodal windows. The window thresholds are smoothed, and the thresholds for the individual pixels are determined by a bilinear interpolation of the window thresholds.

#### C. Eikvil et al.'s Method

The pixels inside a small window  $S$  are thresholded on the basis of clustering of the pixels inside a larger concentric window  $L$  (Fig. 3). We let  $S$  and  $L$  slide across the image in steps equal to the side of  $S$ . We used windows of sizes  $3 \times 3$  and  $15 \times 15$  pixels. For all the pixels inside  $L$ , Otsu's threshold  $T$  [31] is calculated to divide the pixels into two clusters. If the two estimated cluster means  $\hat{\mu}_1$  and  $\hat{\mu}_2$  are further apart than a user-specified limit,  $\|\hat{\mu}_1 - \hat{\mu}_2\| \geq \ell$ , then the pixels inside  $S$  are binarized using the threshold value  $T$ . When  $\|\hat{\mu}_1 - \hat{\mu}_2\| < \ell$ , all the pixels inside  $S$  are assigned to the class with the closest updated mean value. We used  $\ell = 15$ .

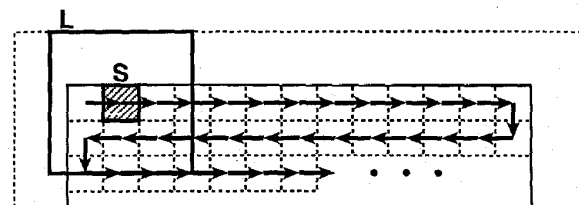


Fig. 3. Principle of Eikvil et al.'s method. The large window  $L$  with the small window  $S$  in the center is moved across the image in a zig-zag fashion, in steps equal to the size of  $S$ . Each pixel inside  $S$  is labeled print or background on the basis of the clustering of the pixels inside  $L$ .

#### D. Mardia and Hainsworth's Method

This method first makes an initial binarization, using, for example, Otsu's method [31]. Then several steps are iterated until convergence is reached. First, the estimated mean  $\hat{\mu}_i$  and the number of pixels  $n_i$  in both print and background of the current binary image are calculated. Then, a threshold  $t^*$  based on these values is calculated, and, for each pixel, a weighted mean,  $G$ , of the pixel and its eight neighbors is computed. If  $G \leq t^*$ , then the pixel is classified as "print," otherwise "background". The weights may be equal, or they may be calculated in a more sophisticated (and time-consuming) manner [21]. We found that the two approaches gave equivalent binarization results for our images.

The last step of the iteration is to smooth the binary image using a  $3 \times 3$  median filter.

### E. Niblack's Method

The idea of this method is to vary the threshold over the image, based on the local mean and local standard deviation. The threshold at pixel  $(x, y)$  is calculated as

$$T(x, y) = m(x, y) + k \cdot s(x, y), \quad (1)$$

where  $m(x, y)$  and  $s(x, y)$  are the sample mean and standard deviation values, respectively, in a local neighborhood of  $(x, y)$ . The size of the neighborhood should be small enough to preserve local details, but at the same time large enough to suppress noise. We found a  $15 \times 15$  neighborhood to be a good choice. The value of  $k$  is used to adjust how much of the total print object boundary is taken as a part of the given object;  $k = -0.2$  gave well-separated print objects.

### F. Taxt et al.'s Method

The image is divided into non-overlapping windows of size  $32 \times 32$  pixels. The histogram in each window is approximated by a mixture of two Gaussian distributions. The parameters of the mixture are estimated using an expectation-maximization (EM) algorithm [34]. In each window, the pixels are classified using the quadratic Bayes' classifier. The EM algorithm requires initial values for the estimated class means  $\hat{\mu}_1$  and  $\hat{\mu}_2$ , class variances  $\hat{\sigma}_1^2$  and  $\hat{\sigma}_2^2$ , and mixing weight  $\hat{\pi}$ . We used  $\hat{\pi} = 0.5$ , and the other four parameter estimates were obtained by clustering [35] 100 randomly selected pixels.

This method gives abrupt changes at the window borders. Eikvil et al.'s method [20] is in fact an improvement of this method. In Eikvil et al.'s method the use of two windows avoids the abrupt changes, and Otsu's method is used as a fast approximation of the EM algorithm. However, since five parameters have to be estimated by this method, a larger window than in In Eikvil et al.'s method has to be used.

### G. Yanowitz and Bruckstein's Method

A potential surface passing through the image at local maxima of the gradient is used as a thresholding surface. The gradient magnitude image is computed, using, for example, Sobel's edge operator [36]. (Jain and Dubuisson [37] have used the Canny edge detector.) This image is thresholded and thinned to one-pixel wide lines to identify edge pixels. The input image is smoothed by a  $3 \times 3$  mean filter. A potential surface is constructed by an iterative interpolation scheme [24]. The interpolated surface is made to pass through the edge pixels. An interpolation residual  $R(x, y)$  is calculated for each non-edge pixel, and the new pixel value  $P(x, y)$  at iteration  $(n + 1)$  is calculated as

$$P_{n+1}(x, y) = P_n(x, y) + \frac{\beta \cdot R_n(x, y)}{4}, \quad (2)$$

$$R_n(x, y) = P_n(x-1, y) + P_n(x+1, y) + P_n(x, y+1) + P_n(x, y-1) - 4P_n(x, y). \quad (3)$$

Yanowitz and Bruckstein recommend that the value of  $\beta$  lie in the interval  $1.0 < \beta < 2.0$  for fast convergence (over-relaxation).

However, we found that  $\beta > 1.0$  resulted in divergence, so  $\beta = 1.0$  was chosen.

The image is binarized using the threshold surface. Finally, false print objects are removed by the postprocessing step described below.

### H. White and Rohrer's Dynamic Thresholding Algorithm

In this method, a biased running average is used as a threshold at each pixel location. The horizontal running average  $H$  at pixel  $(x, y)$  is

$$H(x, y) = f \cdot Z(x, y) + (1 - f) \cdot H(x - 1, y), \quad (4)$$

where  $Z$  is the input image, and the weighing factor  $f$  is a non-linear function of  $[Z(x, y) - H(x - 1, y)]$ , with  $0 < f < 1$  for all arguments. The vertical running average  $V$  at pixel  $(x, y)$  is

$$V(x, y) = g \cdot H(x, y) + (1 - g) \cdot V(x, y - 1), \quad (5)$$

where the weighing factor  $g$  is a non-linear function of  $[H(x, y) - V(x, y - 1)]$ , also with  $0 < g < 1$  for all arguments. In order to use information on both sides of the pixel to be thresholded, a number of look ahead pixels are used. The threshold  $T$  at  $(x, y)$  is

$$T(x, y) = b(V(x + \ell_x, y + \ell_y)), \quad (6)$$

where  $b$  is a bias function. We used  $\ell_x = 8$  and  $\ell_y = 1$  as the number of look ahead pixels in  $x$ - and  $y$ -direction, respectively. The nonlinear functions  $f$  and  $g$  were kindly provided as lookup-tables by White [38]. The bias function  $b$  has to be individually tuned for each class of images.

### I. Parker's Method

This method first detects edges, and then the interior of objects between edges is filled. First, for each pixel  $(x, y)$  in the input image  $Z$ , calculate

$$D(x, y) = \min_{i=1, \dots, 8} (Z(x, y) - Z(x_i, y_i)), \quad (7)$$

where  $(x_i, y_i)$ ,  $i = 1, \dots, 8$  are the eight eight-connected neighbors of  $(x, y)$ . Thus, the negative of the gradient in the direction of the brightest neighbor is found. Then  $D$  is broken up in regions of size  $r \times r$ , and for each region, the sample mean and standard deviations are calculated. Both values are smoothed, and then bilinearly interpolated to give two new images  $M$  and  $S$ , originating from the mean values and standard deviations, respectively. Then for all pixels  $(x, y)$ , if  $M(x, y) \geq m_0$  or  $S(x, y) < s_0$ , then the pixel is regarded as part of a flat region, and remains unlabeled!; else, if  $D(x, y) < M(x, y) + k \cdot S(x, y)$ , then  $(x, y)$  is labeled print; else,  $(x, y)$  remains unlabeled. The resulting binary image highlights the edges. This is followed by pixel aggregation and region growing steps to locate the remaining parts of the print objects. We used the following parameters:  $r = 16$ ,  $m_0 = -1.0$ ,  $s_0 = 1.0$ , and  $k = -1.0$ .

### J. White and Rohrer's Integrated Function Algorithm

This method applies a gradient-like operator, called the activity, on the image. Pixels with activity below a manually set threshold  $T_A$  are labeled '0.' The other pixels are further tested.

1. In Parker's paper [26], the test is  $M(x, y) \geq -1$  or  $S(x, y) \geq -1$ , which is fulfilled by all pixels, since  $S(x, y)$  is never negative.

If the Laplacian edge operator of the pixel is positive, then the pixel is labeled '+,' otherwise '-.' This way, a *three-level label-image* is constructed, with legal pixel values '+,' '0,' and '-.' The idea is that in a sequence of labels, edges are identified as '-+' transitions or '+-' transitions. Object pixels are assumed to be '+' and '0' labeled pixels between a '-+' and '+-' pair. A  $2 \times 2$  region is classified at a time, requiring that all the four pixels are inside either horizontal or vertical object pixel sequences.

**K. Trier and Taxt's Method**

Trier and Taxt [27] have suggested the following three improvements to White and Rohrer's Integrated Function Algorithm:

- 1) The input image is smoothed by a  $5 \times 5$  mean filter.
- 2) The use of search vectors is abandoned. Instead, all '+'-marked regions are labeled print, and '-'-marked regions are labeled background. A '0'-marked region is labeled print if a majority of the pixels that are 4-connected to it are '+'-marked, otherwise it is labeled background.
- 3) The postprocessing step of Yanowitz and Bruckstein's method is used to remove false print objects.

**L. Postprocessing Step**

As in the evaluation study of Trier and Taxt [9], the postprocessing step used in Yanowitz and Bruckstein's method [24] was used to improve some of the other methods as well. The postprocessing step is used to improve the binary image by removing "ghost" objects (Fig. 4), and works as follows. The average gradient value at the *edge* of each printed object is calculated. Objects having an average gradient below a threshold  $T_p$  are labeled as misclassified, and are removed. The main steps of the algorithm are given below:

- 1) Smooth the input image by a  $3 \times 3$  mean filter to remove noise.
- 2) Calculate the gradient magnitude image  $G$  of the smoothed image, using, e.g., Sobel's edge operator [36].
- 3) Select a value for  $T_p$ . So far, there is no well-founded automatic method to specify  $T_p$  for a given image.  $T_p$  has to be specified by trial and error. For all the methods but one,  $T_p = 100$  was used.
- 4) For all the 4-connected print components, calculate the average gradient of the edge pixels. Edge pixels are print pixels that are 4-connected to the background. Remove print components having an average edge gradient value below the threshold  $T_p$ .

**III. DESIGN OF EXPERIMENTS**

To perform an objective evaluation of the binarization results, we have used a character recognition module. Values of recognition rates, reject rates, and error rates are used to compare the different binarization methods.

**A. Test Images**

Two test images ( $3,416 \times 5,240$  pixels, and  $5,060 \times 4,180$  pixels) were obtained by scanning two parts of a hydro-

graphic map at 1,000 dots per inch (dpi) with 256 gray levels. A  $1,024 \times 1,024$  portion of one of these images is shown in Fig. 5a. The hydrographic map, provided by the Norwegian Mapping Authorities, shows depth values and contours from one of the Norwegian coasts. A typical digit in the map is  $50 \times 50$  pixels large.

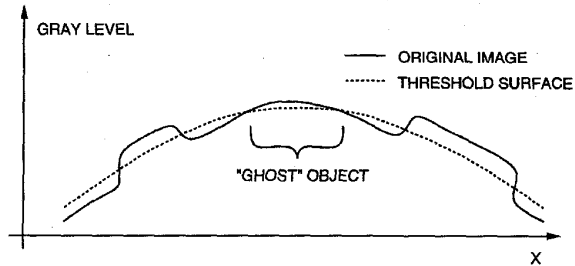


Fig. 4. Cross-section through an image, illustrating how "ghost" objects might occur. Based on a figure in Yanowitz and Bruckstein [24].

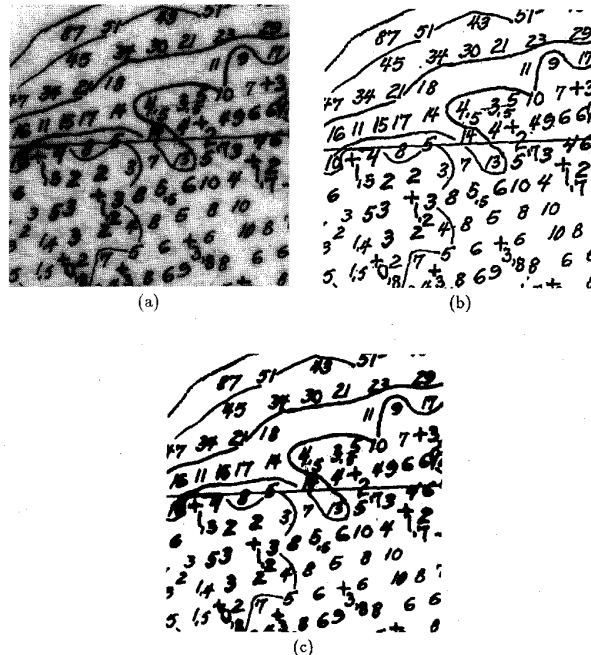


Fig. 5. A  $1,024 \times 1,024$  portion of the test image, which was taken from a hydrographic map. (a) Original image; (b) Binarized with Niblack's method [22] with postprocessing step, the "best" locally adaptive method; (c) Binarized with Otsu's method [31], perhaps the "best" global method.

**B. Binarization Procedure**

Due to limited processor memory and the fact that our binarization software loads the entire input image into the main memory, both test images had to be split into several parts before being binarized. This is a realistic scenario in a large-scale system as well, where the map may be  $1m^2$  or larger, resulting in an approximately 1.6 gigabyte image file for 1,000 dpi scanning resolution.

The test images were split into subimages overlapping their neighbors by 128 pixels. This is probably sufficient to preserve

smoothness across subimage borders for locally adaptive binarization. However, for the global methods, some abrupt changes may be expected at the borders. On the other hand, calculating a separate threshold for each subimage improves binarization slightly, as subimages with generally darker or lighter gray levels than the average may get a more appropriate threshold. The possibility of enforcing the same threshold on all the subimages was not exploited.

### C. Description of Character Recognition Module

Features were extracted from the contour curve of the digits by using Kuhl and Giardina's elliptic Fourier descriptors [39]. Taxt et al. [40] showed that these features give better classification results than some other methods based on closed contours of symbols of known rotation. In our maps, all digits have approximately the same rotation, but the size varies, so scale-invariant features were used.

The motivation behind the elliptic Fourier descriptors is that a periodic function is obtained by following the closed contour of a symbol successively. Periodic functions are well-suited for Fourier series expansions. The closed contour,  $(x(t), y(t))$ ,  $t \in [0, T]$ , where  $T$  is the length of the contour, has the  $N$ th order approximation:

$$\hat{x}(t) = A_0 + \sum_{n=1}^N \left[ a_n \cos \frac{2n\pi t}{T} + b_n \sin \frac{2n\pi t}{T} \right] \quad (8)$$

$$\hat{y}(t) = C_0 + \sum_{n=1}^N \left[ c_n \cos \frac{2n\pi t}{T} + d_n \sin \frac{2n\pi t}{T} \right], \quad (9)$$

with  $\hat{x}(t) \equiv x(t)$  and  $\hat{y}(t) \equiv y(t)$  in the limit when  $N \rightarrow \infty$ . The coefficients are

$$A_0 = \frac{1}{T} \int_0^T x(t) dt, \quad (10)$$

$$C_0 = \frac{1}{T} \int_0^T y(t) dt, \quad (11)$$

$$a_n = \frac{2}{T} \int_0^T x(t) \cos \frac{2n\pi t}{T} dt, \quad (12)$$

$$b_n = \frac{2}{T} \int_0^T x(t) \sin \frac{2n\pi t}{T} dt, \quad (13)$$

$$c_n = \frac{2}{T} \int_0^T y(t) \cos \frac{2n\pi t}{T} dt, \quad (14)$$

$$d_n = \frac{2}{T} \int_0^T y(t) \sin \frac{2n\pi t}{T} dt. \quad (15)$$

The starting point, where  $t = 0$ , can be arbitrarily chosen, and it is clear from (12)-(15) that the coefficients are dependent on this choice. To obtain features that are independent of the particular choice of the starting point, we calculate the *phase shift from the first major axis* as

$$\theta_1 = \frac{1}{2} \tan^{-1} \frac{2(a_1 b_1 + c_1 d_1)}{a_1^2 - b_1^2 + c_1^2 - d_1^2}. \quad (16)$$

Then the coefficients can be rotated to zero phase shift:

$$\begin{bmatrix} a_n^* & b_n^* \\ c_n^* & d_n^* \end{bmatrix} = \begin{bmatrix} a_n & b_n \\ c_n & d_n \end{bmatrix} \begin{bmatrix} \cos n\theta_1 & -\sin n\theta_1 \\ \sin n\theta_1 & \cos n\theta_1 \end{bmatrix}. \quad (17)$$

To obtain size-invariant features, the coefficients can be divided by the magnitude,  $E$ , of the semimajor axis, given by

$$E = \sqrt{a_1^{*2} + c_1^{*2}}. \quad (18)$$

We then obtain the following features  $(\bar{a}_n^*, \dots)$ :

$$\begin{bmatrix} \bar{a}_n^* & \bar{b}_n^* \\ \bar{c}_n^* & \bar{d}_n^* \end{bmatrix} = \frac{1}{\sqrt{a_1^{*2} + c_1^{*2}}} \begin{bmatrix} a_n^* & b_n^* \\ c_n^* & d_n^* \end{bmatrix}. \quad (19)$$

The low-order coefficients contain the most information, and should always be used. We used a total of 12 features:  $\bar{a}_n^*$ ,  $\bar{b}_n^*$ ,  $\bar{c}_n^*$ , and  $\bar{d}_n^*$ ,  $n = 1, 2, 3$ . The small variations in rotation occurring among symbols from the same class will be reflected in the class-conditional density functions.

A quadratic classifier [15] was used, assuming multivariate Gaussian distributions for each of the ten classes. The mean vector and covariance matrix for each class were estimated from the training set. Symbol candidates can be rejected in two ways, by being classified as either *doubt* or *outlier* (Chapter 1 and 6 in Hjort [41]). The doubt criterion is introduced by using the following *loss function*  $L$  in a Bayesian classifier [41]:

$$L(k, c) = \begin{cases} 0 & \text{if } c = k \quad (\text{correct decision}) \\ d & \text{if } c = D \quad (\text{being in doubt}) \\ 1 & \text{otherwise} \quad (\text{wrong decision}), \end{cases} \quad (20)$$

where  $k \in \{1, \dots, K\}$  is the true class,  $c \in \{1, \dots, K, D\}$  is the class assigned by the classifier,  $K$  is the number of classes, and  $d < 1$  is the cost of a reject. It can be shown (Chapter 1 in Hjort [41]) that the optimal classification rule is

$$C_0(x) = \begin{cases} D & \text{if every } P(k|x) \leq (1-d) \\ k & \text{if } P(k|x) = \max_c P(c|x) > (1-d) \end{cases} \quad (21)$$

Note that if  $d \geq 1 - \frac{1}{K}$ , the doubt decision will never be used.

The outlier criterion is used to detect symbol candidates that are not from one of the  $K$  classes (Chapter 6 in Hjort [41]). A hypothesis test is used with the null hypothesis

$$\mathcal{H}_0: x \text{ is from one of the } K \text{ classes}, \quad (22)$$

and alternative hypothesis

$$\mathcal{H}_1: x \text{ is not from one of the } K \text{ classes}. \quad (23)$$

The significance level,  $\epsilon$ , of the test is determined as

$$Pr(x \text{ is classified as outlier} \mid \mathcal{H}_0 \text{ is true}) \leq \epsilon, \quad (24)$$

meaning that, in the long run, only  $(\epsilon \times 100)\%$  of the true digits will be classified as outliers. The outlier criterion can be approximated as follows [41]. A symbol candidate  $x$  is classified as an outlier if

$$W \geq \gamma_{p,1-\epsilon} \quad (25)$$

$$\text{where } W = \min_k (x - \mu_k)^T \Sigma^{-1} (x - \mu_k), \quad (26)$$

$$Pr(\chi_p^2 \geq \gamma_{p,1-\epsilon}) = \epsilon, \quad (27)$$

$p$  is the dimensionality of the feature space, and  $\gamma_{p,1-\epsilon}$  is the  $\epsilon$ -percentile of the  $\chi_p^2$  (Chi-square with  $p$  degrees of freedom) distribution. In the approximation [41], a common covariance matrix,  $\Sigma$ , is used, which may be estimated by [42]:

$$\hat{\Sigma} = \frac{n_1 \hat{\Sigma}_1 + \dots + n_k \hat{\Sigma}_k}{n_1 + \dots + n_k}, \quad (28)$$

where  $n_i$  is the number of samples used to estimate the covariance matrix  $\Sigma_i$  of class  $i$ .

We used  $d = 0.35$  in the doubt criterion, and  $\epsilon = 0.01$  in the outlier criterion.

Symbol candidates were extracted from the test set images by considering all the 8-connected components. Those symbols that had an upright enclosing rectangle within specific upper and lower bounds were accepted as symbol candidates.

**D. Training of Classifier**

The binarized versions of the two images were divided in two parts each, providing two training subimages of size  $3,416 \times 3,930$  and  $5,060 \times 2,090$  pixels. The remaining  $3,416 \times 1,310$  and  $5,060 \times 2,090$  subimages comprised the test set. During training, only the characters that were successfully binarized were used to train the classifier. Numerals with broken loops, numerals attached to other objects, fragmented numerals, self-touching numerals, and otherwise severely degraded numerals, were not included (Fig. 6). The effective size of the training set varied considerably from method to method (Table I). Some methods (e.g., White and Rohrer’s Integrated Function Algorithm) gave a large number of severely degraded digits in the training set images, resulting in a considerably smaller number of training patterns.

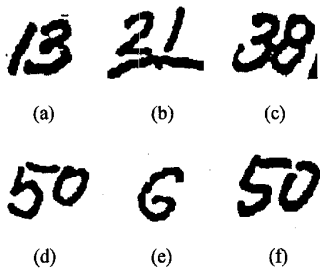


Fig. 6. Examples of digits rejected in the training session; The self-touching ‘3’ in (a), the ‘2’ attached to a line in (b), both digits in (c), the fragmented ‘5’ in (d), the broken-loop ‘6’ in (e), and the broken-loop ‘0’ in (f).

The postprocessing step essentially removes noisy blobs without affecting the digits. Therefore, for the methods with two variants, with or without postprocessing, it was sufficient to train on the binarized images obtained when postprocessing was used. The results of the training session were separate class descriptions corresponding to each binarization method, describing each class with a 12-dimensional feature vector and a  $12 \times 12$  covariance matrix.

**E. Evaluation Procedure**

For each binarization method  $j$ , the resulting class descriptions were loaded into the classifier. Then the test set images obtained by the same binarization method were input to the classifier in turn. Each extracted symbol candidate was either marked with one of the 10 class labels ‘0’–‘9,’ or marked “doubt” or “outlier.” We counted the following quantities from the classification results:

*True number of digits,  $t$ .* This is the number of digits actually contained in the test set. As in all the other quantities below, digits overlapping the image boundaries were not counted.

*Number of correctly classified digits,  $c(j)$ .* This is the number of digits that were correctly classified.

*Number of misclassified digits,  $m_d(j)$ .* This is the number of digits that were misclassified as another digit. This frequently happens for severely degraded symbols (e.g., fragmented symbols, symbols with open loops, symbols with small objects attached, and self-touching symbols).

*Number of rejected digits,  $r_d(j)$ .* This is the number of digits that were marked as “outlier” or “doubt” by the classifier. Most pairs of connected digits fall into this category, as do some fragmented symbols, symbols with noisy blobs attached, and digits with open loops.

*Number of misclassified nondigits,  $m_n(j)$ .* This is the number of nondigit symbol candidates that were incorrectly classified into one of the ten digit classes. Most frequently, this occurs for a component that is a short vertical or nearly-vertical line, thus mistaken for a ‘1’.

*Number of rejected nondigits,  $r_n(j)$ .* This is the number of nondigit symbol candidates that were correctly rejected by the classifier, marked as either an “outlier” or “doubt”.

*Number of ignored digits,  $i_d(j)$ .* This is the number of digits that were not regarded as symbol candidates by the segmentation part of the character recognition module. Typical cases are digits crossed by a grid line, and digits touching a contour line.

The recognition rate  $C$ , reject rate  $R$ , and error rate  $M$  are defined as follows, corresponding to binarization method  $j$ :

$$C(j) = \frac{c(j)}{T(j)} \quad (29)$$

$$R(j) = \frac{r_d(j) + r_n(j) + i_d(j)}{T(j)} \quad (30)$$

$$M(j) = \frac{m_d(j) + m_n(j)}{T(j)}, \quad (31)$$

$$\text{where } T(j) = t + m_n(j) + r_n(j), \quad (32)$$

and  $t = c(j) + r_d(j) + m_d(j) + i_d(j)$ . The denominator  $T(j) = (t + m_n(j) + r_n(j))$  in (29)-(31) reflects the fact that the classifier sometimes treats nondigits (e.g., ‘+’ symbols and short lines) as symbol candidates. These are either correctly rejected or incorrectly classified as a digit. The recognition rate, reject rate, and error rate we have defined illustrate the burden on a human

TABLE I  
THE NUMBER OF DIGITS IN EACH CLASS THAT WERE ACCEPTED  
DURING THE TRAINING SESSIONS. THE ACTUAL TOTAL NUMBER OF DIGITS  
IN THE GRAY SCALE IMAGES WAS 3,188

method	No. of digits in each class										total No. of digits
	1	2	3	4	5	6	7	8	9	0	
<i>locally adaptive methods</i>											
Bernsen	478	334	256	240	232	190	168	137	142	165	2342
Chow/Kaneko	356	219	153	129	143	139	111	101	100	110	1561
Eikvil et al.	483	336	259	242	231	191	166	139	145	168	2360
Niblack	501	376	288	279	255	194	180	144	152	192	2561
Mardia/Hainsworth	410	253	193	184	176	155	142	106	110	132	1861
Taxt et al.	384	261	191	167	161	145	139	100	107	124	1779
Yanowitz/Bruckstein	485	350	267	263	238	191	173	137	145	181	2430
White/Rohrer DTA	482	355	273	252	247	188	173	138	145	174	2427
Parker	482	339	263	217	224	177	168	134	129	182	2315
White/Rohrer IFA	470	314	254	234	250	180	168	130	140	185	2325
Trier/Taxt	489	359	284	266	245	184	177	136	148	185	2473
<i>global methods</i>											
Abutaleb	281	138	112	73	90	81	96	56	66	50	1043
Kapur et al.	104	31	28	19	56	57	50	27	42	14	428
Kittler/Illingworth	160	64	67	46	76	61	66	38	49	27	654
Otsu	343	161	135	89	122	107	113	74	89	69	1302

TABLE II  
RESULTS OF USING THE CHARACTER RECOGNITION MODULE  
ON THE BINARIZED IMAGES. DOUBT AND OUTLIER CRITERIA:  
 $d=0.35$ ,  $e=0.01$ , RESPECTIVELY

binarization method	number of symbol candidates								rates [%]		
	$c$	$m_d$	$r_d$	$m_n$	$r_n$	$i_d$	$t$	$T$	$C$	$M$	$R$
<i>locally adaptive methods</i>											
Bernsen	1367	20	223	17	52	150	1760	1829	74.74	2.02	23.24
Chow/Kaneko	1101	16	380	14	44	263	1760	1818	60.56	1.65	37.79
Eikvil et al.	1354	21	236	15	56	149	1760	1831	73.95	1.97	24.08
Mardia/Hainsworth	1091	36	373	10	53	260	1760	1823	59.85	2.52	37.63
Niblack	1429	16	185	26	62	130	1760	1848	77.33	2.27	20.40
Taxt et al.	1307	22	258	16	47	173	1760	1823	71.70	2.08	26.22
Yanowitz/Bruckstein	1394	20	200	19	49	146	1760	1828	76.26	2.13	21.61
White/Rohrer DTA	1405	23	210	23	60	122	1760	1843	76.23	2.50	21.27
Parker	1375	25	223	24	49	137	1760	1833	75.01	2.67	22.31
White/Rohrer IFA	1399	52	209	42	175	100	1760	1977	70.76	4.76	24.48
Trier/Taxt	1395	22	215	22	62	128	1760	1844	75.65	2.39	21.96
<i>global methods</i>											
Abutaleb	852	23	419	10	42	466	1760	1812	47.02	1.82	51.16
Kapur et al.	489	25	622	5	37	624	1760	1802	27.14	1.66	71.20
Kittler/Illingworth	913	31	432	10	54	384	1760	1824	50.05	2.25	47.70
Otsu	1126	57	321	12	52	255	1759	1823	61.77	3.78	34.45
<i>modified locally adaptive methods</i>											
Bernsen, postproc.	1367	20	223	17	47	150	1760	1824	74.95	2.03	23.03
Eikvil et al., postproc.	1354	21	236	15	51	149	1760	1826	74.15	1.97	23.88
Niblack, postproc.	1429	16	185	26	45	130	1760	1831	78.04	2.29	19.66
Parker, postproc.	1375	25	223	23	45	137	1760	1828	75.22	2.63	22.15

operator who wants to verify the classification results of a character recognition system. One can argue that the term  $r_n(j)$  should be in the expression for recognition rate, and not reject rate. However, all the rejected symbols have to be manually checked, and the burden on an operator is the same regardless

of the rejected symbol being a true digit or nondigit.

We can also debate whether the term  $i_d(j)$  is to be included in the reject rate or the error rate. We chose to include it in the reject rate, since the ignored digits are not marked with a class label. Thus, they are easily distinguishable from the misclassified ones.

When comparing two binarization methods, one method may lead to a lower error rate, but higher reject rate than the other. Then we must tune the doubt and outlier criteria to make either the reject rates or the error rates of the two methods to be the same. Alternatively, we can compare the reject vs. error rate curves [43] for the two methods.

#### IV. RESULTS

The results of the experiments, using  $d = 0.35$  and  $\epsilon = 0.01$  in the doubt and outlier criteria, respectively, are shown in Table II. Chow and Kaneko's method had the lowest error rate, but a high reject rate. Two of the global methods, Kapur et al.'s method and Abutaleb's method, also had low error rates, but with very high reject rates. Of the other methods, Eikvil et al.'s method with postprocessing had the lowest error rate, while Niblack's method with postprocessing had the lowest reject rate, with a slightly higher error rate.

For the methods that appear in two variants, with or without the postprocessing step, the only difference was that without postprocessing, some additional noisy blobs which were large enough to be recognized as symbol candidates appeared in the binary image. Almost all these noisy blobs were correctly rejected by the character recognition module, thus slightly increasing the reject rate. Also, the error rate is insignificantly reduced, since  $T(j)$  has increased slightly.

Parker's method with postprocessing did in fact remove a few digits from the image when  $T_p = 100$  was used in the postprocessing step. When run with  $T_p = 75$ , the results shown in Table II were obtained. No other method had these problems with  $T_p = 100$ .

Next, the error vs. reject curves for the methods that seemed to have the best combination of low error rate and low reject rate were plotted (Fig. 7). The curves were obtained by changing the outlier criterion. We also tried changing the doubt criterion, but this was not very useful. In fact, only a very few symbol candidates were classified as "doubt" even when  $d \leq 0.05$ . The curves suggest that Niblack's method with postprocessing has the best performance, but only slightly better than Yanowitz and Bruckstein's method. We also plotted the data obtained from Table II for the best of the other binarization methods (Fig. 7). None of these methods are able to compete with Niblack's method with postprocessing.

The binarization results obtained by Niblack's method with postprocessing and by Otsu's method on a small part of the test set are shown in Figs. 5b and 5c. Niblack's method with postprocessing was able to separate a fairly large number of the digits from other print objects. Still some of the digits were either connected in pairs, or connected to lines and curves (Fig. 5b), and were rejected by the character recognition module. With Otsu's method, these defects were much more apparent (Fig. 5c). In addition, some of the symbols that were separated were instead self-touching or blurred, increasing the chance of being either misrecognized or rejected by the character recognition module.

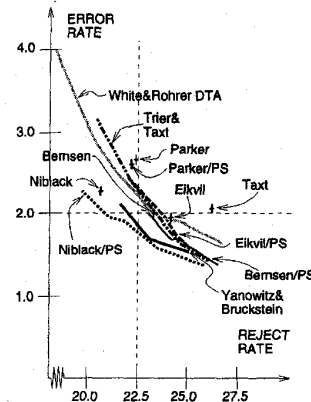


Fig. 7. Error rate vs. reject rate for the best methods.

#### V. DISCUSSION AND FUTURE WORK

We have described an evaluation methodology for low-level image analysis methods, which we called *goal-directed evaluation*. We used it to evaluate the eleven most promising locally adaptive binarization methods. Our experimental results indicated that Niblack's method with postprocessing step appears to be the best. However, it is desirable to use a hypothesis test to state whether this method is *significantly* better than the others. A test similar to McNemar's test [44], [45] may be used. McNemar's test cannot be used directly, since it assumes two possible outcomes for each classified digit, as opposed to three outcomes (correct, error, reject) in our case.

Computational issues have not been considered in this evaluation. In practical applications, execution times are often specified as absolute upper limits, and dictate which methods can be considered and which must be avoided or improved. Trier and Taxt [9] found that Niblack's method and Bemsen's method were the fastest, while Yanowitz and Bruckstein's method was the slowest of the best performing methods (Table III). However, Yanowitz and Bruckstein's method can be speeded up [24] at the cost of minor changes in the thresholding surface. Some of the algorithms which are slow in sequential implementation may actually turn out to be more amenable to parallel implementation.

The role of the postprocessing step also needs to be discussed. For most methods, it is of limited value in terms of the reject and error rates; the main effect was to remove small print objects (noisy blobs) from the binary image that would have been sorted out as too small by the character recognition module anyway. However, the choice of which module, binarization or character recognition, should be responsible for removing these noisy blobs can have an important impact on the execution times. The character recognition module slowed down substantially when Niblack's method and Parker's method were used without the postprocessing step.

The evaluation study of Trier and Taxt [9] using visual criteria (Table III) gave a slightly different ranking of the methods. Most importantly, the effect of the postprocessing step was significantly larger. However, we found that the presence of noisy blobs, if smaller than the lower bound on symbol can-

didate size did not effect the reject and error rates, although their presence did slow down the character recognition module considerably. Both evaluations identified Niblack's method with postprocessing as the best, and also identified the same top six methods, but in different order.

TABLE III  
THE EVALUATION RESULTS FROM THE STUDY OF TRIER AND TAXT [9], WHERE FIVE VISUAL CRITERIA WERE USED. THE BEST SCORE IS SHOWN IN **BOLDFACE**. EXECUTION TIME IS FOR A 512 × 512 PIXEL IMAGE ON A SILICON GRAPHICS INDY WORKSTATION

binarization method	score	rank	execution time
<i>locally adaptive methods</i>			
Yanowitz/Bruckstein	128	4	98 s
Trier/Taxt	124	5	6 s
Niblack	115	6	1 s
Eikvil et al.	115	7	15 s
Bernsen	108	8	1 s
White/Rohrer DTA	107	9	1.6s
Parker	97	11	30 s
Chow/Kaneko	97	12	66 s
Mardia/Hainsworth	89	14	14 s
Taxt et al.	85	16	83 s
Whire/Rohrer IFA	73	18	4 s
<i>global methods</i>			
Otsu	95	13	0.3s
Kapur et al.	86	15	0.3s
Abutaleb	85	17	1 s
Kittler/Illingworth	62	19	0.3s
<i>modified locally adaptive methods</i>			
Niblack, postproc.	<b>138</b>	<b>1</b>	3 s
Eikvil et al., postproc.	133	2	18 s
Bernsen, postproc.	132	3	3 s
Parker, postproc.	104	10	32 s

Strictly speaking, our evaluation has only identified the best binarization method for the given OCR system. One might think that using a different feature extraction method and/or a different classifier would give a different ranking of the binarization methods. If this is true, then one might have to do several iterations optimizing one module (Fig. 2) at a time. Still, there is no guarantee that the global optimum will be found by optimizing each module separately.

Another interesting goal-directed evaluation study would be to evaluate thinning algorithms in the context of handwritten digit or character recognition. Then one would avoid the subjective considerations used to evaluate criteria like shape preservation, and common defects such as spurious branches, dislocated junctions and corners, etc.

The substantial amount of time and effort needed in conducting a careful evaluation, whether goal-directed or not, may explain why more such studies have not been reported. Labeling the training patterns, and inspecting the classification results on the test set were indeed extremely tedious tasks, and required about five hours for each binarization method. Better software tools are needed to allow large scale evaluations.

In conclusion, the present goal-directed evaluation study of

eleven locally adaptive binarization methods indicates that Niblack's method, with the addition of the postprocessing step of Yanowitz and Bruckstein's method, is the best binarization method for our hydrographic images. However, we need to use a hypothesis test to decide whether this method is *significantly* better than the other binarization methods included in this study. We must point out that the results of this objective evaluation are in the context of handwritten digit recognition, and are not immediately applicable to other application domains in which binarized images are needed.

#### ACKNOWLEDGMENTS

We thank the Norwegian Mapping Authorities, Stavanger, Norway, for providing challenging data, Torfinn Taxt and Sarathcha Pankanti for useful discussions, and the anonymous referees for relevant comments. This work was supported by a NATO collaborative research grant (CRG 930289), and the Research Council of Norway.

#### REFERENCES

- [1] *Proc. First Int'l Conf. Document Analysis and Recognition*, Saint-Malo, France, 1991.
- [2] *Proc. Second Int'l Conf. Document Analysis and Recognition*, Tsukuba Science City, Japan, 1993.
- [3] R. Kasturi and L. O'Gorman, guest eds., "Special issue: Document image analysis techniques," *Machine Vision and Applications*, vol. 5, no. 3, pp. 141-248, 1992.
- [4] R. Kasturi and L. O'Gorman, guest eds., "Special issue: Document image analysis techniques," *Machine Vision and Applications*, vol. 6, no. 2-3, pp. 67-180, 1993.
- [5] L. O'Gorman and R. Kasturi, guest eds., "Special issue on document image analysis systems," *Computer*, vol. 25, no. 7, pp. 1-112, July 1992.
- [6] R. Plamondon, Guest ed., "Special issue: Handwriting processing and recognition," *Pattern Recognition*, vol. 26, no. 3, pp. 379-460, Mar. 1993.
- [7] T. Pavlidis and S. Mori, guest eds., "Special issue on optical character recognition," *Proc. IEEE*, vol. 80, no. 7, pp. 1,027-1,215, July 1992.
- [8] H. Tominaga, guest ed., "Special issue on postal processing and character recognition," *Pattern Recognition Letters*, vol. 14, no. 4, pp. 257-354, Apr. 1993.
- [9] Ø.D. Trier and T. Taxt, "Evaluation of binarization methods for document images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 3, pp. 312-315, Mar. 1995.
- [10] S.-W. Lee, L. Lam, and C.Y. Suen, "Performance evaluation of skeletonizing algorithms for document image processing," *Proc. First Int'l Conf. Document Analysis and Recognition*, pp. 260-271, Saint-Malo, France, 1991.
- [11] S.O. Belkasim, M. Shridhar, and A. Ahmadi, "Pattern recognition with moment invariants: A comparative study and new results," *Pattern Recognition*, vol. 24, no. 12, pp. 1,117-1,138, Dec. 1991.
- [12] R.M. Haralick, "Performance characterization in image analysis: Thinning, a case in point," *Pattern Recognition Letters*, vol. 13, no. 1, pp. 5-12, Jan. 1992.
- [13] M.Y. Jaisimha, R.M. Haralick, and D. Dori, "Quantitative performance evaluation of thinning algorithms under noisy conditions," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 678-683, Seattle, 1994.
- [14] Ø.D. Trier, A.K. Jain, and T. Taxt, "Feature extraction methods for character recognition—a survey," Technical Report, Michigan State Univ., Dec. 1994, to appear in *Pattern Recognition*.

- [15] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*. New York: John Wiley & Sons, 1973.
- [16] K.-S. Fu, *Syntactic Pattern Recognition and Application*. Englewood Cliffs, N.J.: Prentice Hall, 1982.
- [17] J. Bernsen, "Dynamic thresholding of grey-level images," *Proc. Eighth Int'l Conf. Pattern Recognition*, pp. 1,251-1,255, Paris, 1986.
- [18] C.K. Chow and T. Kaneko, "Automatic detection of the left ventricle from cineangiograms," *Computers and Biomedical Research*, vol. 5, pp. 388-410, 1972.
- [19] Y. Nakagawa and A. Rosenfeld, "Some experiments on variable thresholding," *Pattern Recognition*, vol. 11, no. 3, pp. 191-204, 1979.
- [20] L. Eikvil, T. Taxt, and K. Moen, "A fast adaptive method for binarization of document images," *Proc. First Int'l Conf. Document Analysis and Recognition*, pp. 435-443, Saint-Malo, France, 1991.
- [21] K.V. Mardia and T.J. Hainsworth, "A spatial thresholding method for image segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 919-927, 1988.
- [22] W. Niblack, *An Introduction to Digital Image Processing*, pp. 115-116. Englewood Cliffs, N.J.: Prentice Hall, 1986.
- [23] T. Taxt, P.J. Flynn, and A.K. Jain, "Segmentation of document images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, no. 12, pp. 1,322-1,329, 1989.
- [24] S.D. Yanowitz and A.M. Bruckstein, "A new method for image segmentation," *Computer Vision, Graphics and Image Processing*, vol. 46, no. 1, pp. 82-95, Apr. 1989.
- [25] J.M. White and G.D. Rohrer, "Image thresholding for optical character recognition and other applications requiring character image extraction," *IBM J. Research and Development*, vol. 27, no. 4, pp. 400-411, July 1983.
- [26] J.R. Parker, "Gray level thresholding in badly illuminated images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 813-819, 1991.
- [27] Ø.D. Trier and T. Taxt, "Improvement of 'integrated function algorithm' for binarization of document images," *Pattern Recognition Letters*, vol. 16, no. 3, pp. 277-283, Mar. 1995.
- [28] A.S. Abutaleb, "Automatic thresholding of gray-level pictures using two-dimensional entropy," *Computer Vision, Graphics and Image Processing*, vol. 47, pp. 22-32, 1989.
- [29] J.N. Kapur, P.K. Sahoo, and A.K.C. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram," *Computer Vision, Graphics and Image Processing*, vol. 29, pp. 273-285, 1985.
- [30] J. Kittler and J. Illingworth, "Minimum error thresholding," *Pattern Recognition*, vol. 19, no. 1, pp. 41-47, 1986.
- [31] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62-66, 1979.
- [32] T. Kurita, N. Otsu, and N. Abdelmalek, "Maximum likelihood thresholding based on population mixture models," *Pattern Recognition*, vol. 25, no. 10, pp. 1,231-1,240, 1992.
- [33] S. Cho, R. Haralick, and S. Yi, "Improvement of Kittler and Illingworth's minimum error thresholding," *Pattern Recognition*, vol. 22, no. 5, pp. 609-617, 1989.
- [34] D.M. Titterton, A.F.M. Smith, and U.E. Makov, *Statistical Analysis of Finite Mixture Distributions*. New York: John Wiley & Sons, 1985.
- [35] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data*, pp. 96-117. Englewood Cliffs, N.J.: Prentice Hall, 1988.
- [36] W.K. Pratt, *Digital Image Processing*, second edition, pp. 501-504. New York: John Wiley & Sons, 1991.
- [37] A.K. Jain and M.-P. Dubuisson, "Segmentation of X-ray and C-scan images of fiber reinforced composite materials," *Pattern Recognition*, vol. 25, no. 3, pp. 257-270, 1992.
- [38] Jim M. White, 1994, Private communication.
- [39] F.P. Kuhl and C.R. Giardina, "Elliptic Fourier features of a closed contour," *Computer Vision, Graphics and Image Processing*, vol. 18, pp. 236-258, 1982.
- [40] T. Taxt, J.B. Ólafsdóttir, and M. Dæhlen, "Recognition of handwritten symbols," *Pattern Recognition*, vol. 23, no. 11, pp. 1,155-1,166, 1990.
- [41] N.L. Hjort, "Notes on the theory of statistical pattern recognition," Report no. 778, Norwegian Computing Center, Oslo, Norway, Dec. 1986.
- [42] K.V. Mardia, J.T. Kent, and J.M. Bibby, *Multivariate Analysis*. London: Academic Press, 1979.
- [43] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. Boston: Academic Press, 1990.
- [44] S.A. Glantz, *Primer of Biostatistics*, third edition, pp. 311-314. New York: McGraw-Hill, 1992.
- [45] Tim Hesterberg, 1994, Private communication.



Øivind Due Trier received his MSc degree in computer science from the Norwegian Institute of Technology in 1991. He was with the Norwegian Defense Research Establishment from 1991 to 1992. Since 1992, he has been a PhD student at the Department of Informatics, University of Oslo, and will graduate in January, 1996. His current research interests include pattern recognition, image analysis, document processing, and geographical information systems.

He received the best paper award at the Norwegian National Conference on Image Analysis and Pattern Recognition, Aalesund, Norway, 1995, and a prize for one of the three best poster papers presented at the Third International Conference on Document Analysis and Recognition, Montreal, Canada, 1995.

Mr. Trier is a student member of the IEEE.



Anil K. Jain received a BTech degree in 1969 from the Indian Institute of Technology, Kanpur, and the MS and PhD degrees in electrical engineering from Ohio State University, in 1970 and 1973, respectively. He joined the faculty of Michigan State University in 1974, where he currently holds the rank of University Distinguished Professor in the Department of Computer Science. Dr. Jain served as program director of the Intelligent Systems Program at the National Science Foundation (1980-1981), and has held visiting appointments at Delft Technical

University, Holland, Norwegian Computing Center, Oslo, and Tata Research Development and Design Center, Pune, India.

Dr. Jain has published a number of papers on the following topics: statistical pattern recognition, exploratory pattern analysis, neural networks, Markov random fields, texture analysis, interpretation of range images, and 3D object recognition. Several of his papers have been reprinted in edited volumes on image processing and pattern recognition. He received the best paper awards in 1987 and 1991, and received certificates for outstanding contributions in 1976, 1979, and 1992 from the Pattern Recognition Society. Dr. Jain served as the editor-in-chief of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1991-1994), and currently serves on the editorial boards of *Pattern Recognition Journal*, *Pattern Recognition Letters*, *Journal of Mathematical Imaging*, *Journal of Applied Intelligence*, and *IEEE Transactions on Neural Networks*. He is the co-author of the book *Algorithms for Clustering Data* (Prentice Hall, 1988), has edited the book *Real-Time Object Measurement and Classification* (Springer-Verlag, 1988), and has co-edited the books, *Analysis and Interpretation of Range Images* (Springer-Verlag, 1989), *Neural Networks and Statistical Pattern Recognition* (North Holland, 1991), *Markov Random Fields: Theory and Applications* (Academic Press, 1993), and *3D Object Recognition* (Elsevier, 1993).

Dr. Jain is a fellow of the IEEE. He is currently a Distinguished Lecturer of the IEEE Computer Society's Asia-Pacific Lectureship Program.