THE OPTIMIZATION OF EDGE AND LINE DETECTORS FOR FOREST IMAGE ANALYSIS

Zhiling Long, Joseph Picone

Institute for Signal and Information Processing
Department of Electrical and Computer Engineering, Mississippi State University
Mississippi State, MS 39762, USA

and

Victor A. Rudis

Forest Inventory and Analysis Unit, Southern Research Station, USDA Forest Service
Starkville, MS 39760-0928, USA

ABSTRACT

Automated image analysis for forestry applications is becoming increasingly important with the rapid evolution of satellite and land-based remote imaging industries. Features derived from line information play a very important role in analyses of such images. Many edge and line detection algorithms have been proposed, but few, if any, comprehensive studies exist that evaluate performance in a scientifically meaningful way. In this paper, we introduce an objective evaluation paradigm. We also demonstrate, using this paradigm, improved performance on edge and line detection. We reduced the detection error rate from 42% to 29% for 159 manually labeled forest images.

Keywords: edge and line detection, image processing, forestry imaging, evaluation methodologies.

1. INTRODUCTION

For the past several years, we have been developing an automated image analysis system for forestry imaging [1][2]. This system extracts straightforward features such as color, and uses more subtle measurements such as entropy and line length distributions, to produce estimates of subjective measures such as scenic beauty. The system also performs image segmentation and classifies segments based on their content (trees, sky, foliage, etc.). The density of long and short lines in a forest image are important features for automated forest image classification and segmentation [1].

To generate such features reliably, these detectors had to be optimized for the unique character of forest images. The optimization is difficult for two reasons. First, we need to quantify the performance of edge and line detectors. Subjective evaluations may be reliable, but are inefficient because they usually involve the use of human subjects and require tremendous amounts of time. Second, there is no universally accepted objective metric. Although some evaluation metrics have been widely used [3], they all have shortcomings with respect to our application. For example, the metric proposed by Kitchen and Rosenfeld [4] only measures the continuity and thinness of detected edges, while we are also interested in the mismatch rate and the false alarms introduced by the detectors.

The first step in our evaluation methodology is to manually transcribe lines. From this transcription, we create a set of reference line data which is used to evaluate detector performance. We can transcribe lines either from a virtual perspective or from a physical perspective. Examples with both perspectives are shown in Figure 1. With the virtual perspective we use perception of lines to assume their continuation, e.g., we label outlines of tree trunks behind shrubs as well as those visible parts. With the physical perspective, we make no such assumptions about the continuity of obstructed lines. Although a virtual perspective is more consistent with human perception, it requires a much more intelligent line detection algorithm. The physical perspective fits the existing state of the art better, and hence was the criterion chosen for our study.

2. METRIC DESIGN

To measure the performance of an algorithm, we need to compare lines detected by the algorithm to the reference data. This is not trivial since line orientation and length can differ by small amounts due to computational issues, and yet perfectly acceptable from a visual perspective. Hence, some form of an evaluation metric is required that is forgiving of such small errors. In our metric, we evaluated such properties as location, length, and slope.

We determine the location of a line by finding the coordinates of its middle point. Suppose the coordinates of the two end points of a line are (x_0, y_0) and (x_1, y_1), respectively, then the length and slope are computed as

$$Length = \sqrt{(y_1 - y_0)^2 + (x_1 - x_0)^2} \tag{1}$$

and

$$Slope = \operatorname{atan}\left(\frac{y_1 - y_0}{x_1 - x_0}\right), \tag{2}$$

where $x_1 \neq x_0$.

The evaluation can be described as a two-stage procedure. In the first stage, we process all reference lines and find the best match for each reference line in the set of line segments hypothesized for a given image. In the second stage of the evaluation, we tabulate errors by considering the similarity between the reference line and its best match in the hypothesized data.

To find the best matches (i.e., the first stage), we search through all detected lines and compute the distance from the middle point of each detected line to a reference line. The minimum of this distance is the best match for that reference line. We count all detected lines without matches in the reference data as **insertion errors**.

To evaluate the best matches (i.e., the second stage), we need to handle four situations. First, the best match and the associated reference line are close parallel lines of approximately the same length. This is considered a correct detection. Second, we can have close parallel lines of unequal lengths. It is a correct detection if the difference in lengths is within 25% of the length of the reference line. Third, we can have close non-parallel lines of approximately the same length. These are considered a correct detection if the angle between the two lines is less than 20 degrees. Finally, non-parallel lines of unequal lengths are considered a correct detection if the angle between the two lines is less than 20 degrees and the length difference is within 25% of the length of the reference line.

The notion of proximity must be determined empirically using a threshold. To determine if a detected line is close to the associated reference line, we compute the distance from the middle point of the former to the latter. If that distance is within 5 pixels, then we consider these lines as close. We then apply the criteria described above to determine the nature of the match.

Using these criteria, we will have two types of errors: insertion and detection. Insertion errors were described previously. Detection errors represent all errors for which the reference line and the best match do not pass the criteria described above (location, length and slope).

3. ALGORITHMS

We now describe the specific edge and line detection algorithms used in this study. Canny proposed several criteria for edge detector design and derived corresponding optimal operators by numerical methods [5]. He also introduced a novel edge detection scheme on the basis of these optimal operators.

3.1. Edge Detection

Canny's edge detection algorithm works in the following way. First, a symmetric two-dimensional Gaussian mask is convolved with the original image to smooth out noise present in the source. We used a 3x3 Gaussian mask:

$$\frac{1}{\sqrt{2\pi\sigma^2}}\begin{bmatrix} \exp\left(-\frac{1}{\sigma^2}\right) & \exp\left(-\frac{1}{2\sigma^2}\right) & \exp\left(-\frac{1}{\sigma^2}\right) \\ \exp\left(-\frac{1}{2\sigma^2}\right) & 1 & \exp\left(-\frac{1}{2\sigma^2}\right) \\ \exp\left(-\frac{1}{\sigma^2}\right) & \exp\left(-\frac{1}{2\sigma^2}\right) & \exp\left(-\frac{1}{\sigma^2}\right) \end{bmatrix} \tag{3}$$

Second, differentiations in both horizontal and vertical directions are performed. The corresponding masks are:

Horizontal: Vertical:

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{4}$$

Third, gradient values are calculated on the basis of both differentiations at each pixel of the image.

$$Gradient = \sqrt{\left(\frac{d}{dx}\right)^2 + \left(\frac{d}{dy}\right)^2} . \tag{5}$$

An edge pixel is defined to be a local maximum of the gradient values.

The final step in edge detection is to suppress non-maximal gradient values. To accomplish this last step, an adaptive thresholding technique known as “hysteresis” is used. First set two threshold values. If the gradient value of a pixel is greater than the higher threshold, then the pixel is treated as an edge pixel. If a pixel is less than the higher threshold, greater than the lower threshold, and connected to a previously identified edge, then the pixel is subsequently treated as an edge pixel. Otherwise, it is a non-edge pixel. Hysteresis greatly reduces broken edge contours.

3.2. Line Detection

Our approach to line detection is simple. We postprocess the output of the edge detector and compare edge lengths with a threshold parameter. For an edge to be a line, its length must be above the line threshold. In this initial effort, we are only interested in those vertical lines which represent tree stems, shrubs and grasses. Therefore, we do not detect lines in the horizontal direction.

4. OPTIMIZATION EXPERIMENTS

Our optimization paradigm is summarized as follows. First, we create a set of reference data by manually transcribing “significant” lines in images from a physical perspective. A line is considered significant if it is easily distinguished from the surrounding scene. Typically, these lines are located at places where there is a great discontinuity in intensity. Next, we use our objective metric for performance evaluation. We sweep through ranges of parameter values searching for the combinations that give us the best overall performance.

We prepared two data sets for experimentation. Data set 1 contained 165 images randomly chosen from the training set 01 of USFS Pre-phase 01 image data. Data set 2 consisted of 159 images randomly chosen from the test set 01 of Pre-Phase 01 [6].

We experimented with key parameters for both the edge and the line detectors. These parameters included the Gaussian standard deviation (σ in the Gaussian mask function), the low and high edge thresholds for edge detection [5], and the line threshold for line detection.

Detector performance was assessed using a combination of two resulting values. First, we considered the error rate, which is the ratio of detection errors to the total number of reference lines. Second, we considered the insertion rate, which is the total number of insertion errors. Both errors have been described previously in the metric design. A system which achieves a low error rate and a low insertion rate simultaneously is desirable. However, one interesting phenomenon we noticed from the experiments was that, when we lowered the thresholds and the Gaussian standard deviation, the error rate tended to decrease, and the insertion rate increased. The reason for this trade-off is that lower thresholds result in more lines, which simultaneously increases the chance for both correct matches and undesirable insertions. Figure 3 - 6 illustrate the results with various experimental conditions.

An optimal parameter set is one well balanced between the error rate and the insertion rate. By visual inspection of the error rate curves and the insertion rate curves, we found that with the following settings of parameters, we achieved a good balance between both the error rate and the insertion rate: 2.0 for the Gaussian standard deviation, 60 for the high edge threshold, 30 for the low edge threshold, and 40 for the line threshold. The error rate achieved with these settings was 29% on data set 2.

The corresponding insertion rate was 272,073 lines for all 159 images, as shown in Figure 2. Given that we had transcribed only significant lines (not all existing lines) as reference data, some of the

Optimization	Parameters				Performance	
	σ	High Edge Threshold	Low Edge Threshold	Line Threshold	Error Rate	Insertion Rate
Without	2.0	180	60	15	42%	701,877
With	2.0	60	30	40	29%	272,073

Table 1: A comparison of parameters and performance between the optimized system and the previous system.

inserted lines might actually be correct detection. Hence such an insertion rate seemed acceptable. A comparison of parameters and performance between the optimized system and the previous system is shown in Table 1.

5. SUMMARY AND INSIGHT

In this paper, we designed an objective metric to evaluate the performance of edge and line detectors, and then optimized the performance of our image analysis system using this metric. Our best system resulted in an error rate of 29%, and had an acceptable insertion rate.

Tree density and tree stem appearance are critically important to forest resource measurements, but forest users who value scenery indicate that the variety of natural features are also very important [7]. In our analysis, we observed that many of the inserted lines without matches to our vertical reference lines also occurred as edges of horizontal and other irregularly-shaped features, such as boundaries between sun-drenched and shadowed areas on the forest floor. We speculate that drawing polygon boundaries from detected lines could be a logical next step in quantifying multi-dimensional polygon features. To evaluate these features, one might manually transcribe and label polygon areas, and then compare them with polygon areas bounded by detected line boundaries. Scene features could then be assessed for density, diversity, orientation, and juxtaposition - attributes that likely play a role in scenic beauty, vegetation diversity, and other subjective, labor-intensive forest resource evaluations.

6. ACKNOWLEDGMENT

This work was supported by the United States Department of Agriculture (USDA) through the United States Forest Service (USFS) Southern Research Station, Ecosystem Management research for the Ouachita-Ozark National Forests.

7. REFERENCES

[1] N. Kalidindi, A. Lee, L. Zheng, Y. Hong, J. Picone, and V. A. Rudis, “Scenic Beauty Estimation of Forestry Images”, Proceedings of IEEE Southeastcon, New York, NY, USA: IEEE, 1997, pp. 337-339.

[2] V. A. Rudis, R. E. Thill, J. H. Gramann, J. Picone, N. Kalidindi and P. Tappe, “Understory Structure by Season Following Uneven-aged Reproduction Cutting: a Comparison of Selected Measures Two and Six Years after Treatment”, Forest Ecology and Management, vol. 114, no. 2-3, 1999, pp. 309-320.

[3] R. N. Strickland and D. Chang, “An Adaptable Edge Quality Metric”, Proceedings of SPIE, vol. 1360, Bellingham, WA, USA: SPIE, 1990, pp. 982-995.

[4] L. Kitchen and A. Rosenfeld, “Edge Evaluation Using Local Edge Coherence”, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 9, 1981, pp. 597-605.

[5] J. Canny, “A Computational Approach to Edge Detection”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 8, no. 6, 1986, pp. 679-698.

[6] Z. Long, and J. Picone, “USFS: DATA”, http://www.isip.msstate.edu/projects/usfs/data/index.html, Mississippi State University, 2000.

[7] T. A. Herrick, and V. A. Rudis, “Visitor Preference for Forest Scenery in the Ouachita National Forest”, Proceedings of the Symposium on Ecosystem Management Research in the Ouachita Mountains: Pretreatment Conditions and Preliminary Findings, GTR-SO-112, New Orleans, LA, USA: USDA-FS. Southern Forest Experiment Station, 1993, pp. 212-222.

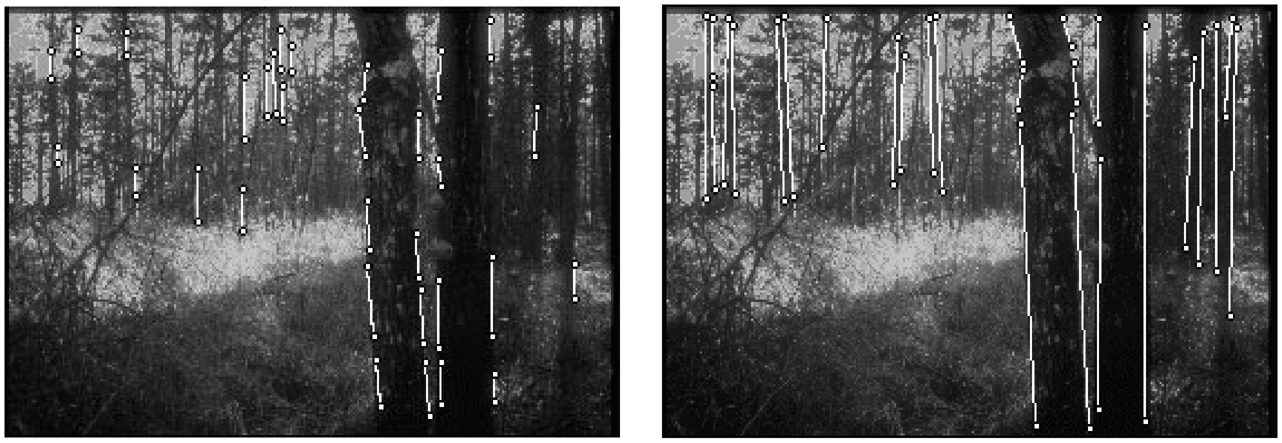

Figure 1: Example transcriptions with the two perspectives. The image on the left was transcribed from the physical perspective - only existing lines were labeled. The image on the right was transcribed from the virtual perspective, where obstructed lines, such as outlines of tree trunks behind shrubs, were also labeled. The white lines are the transcriptions.

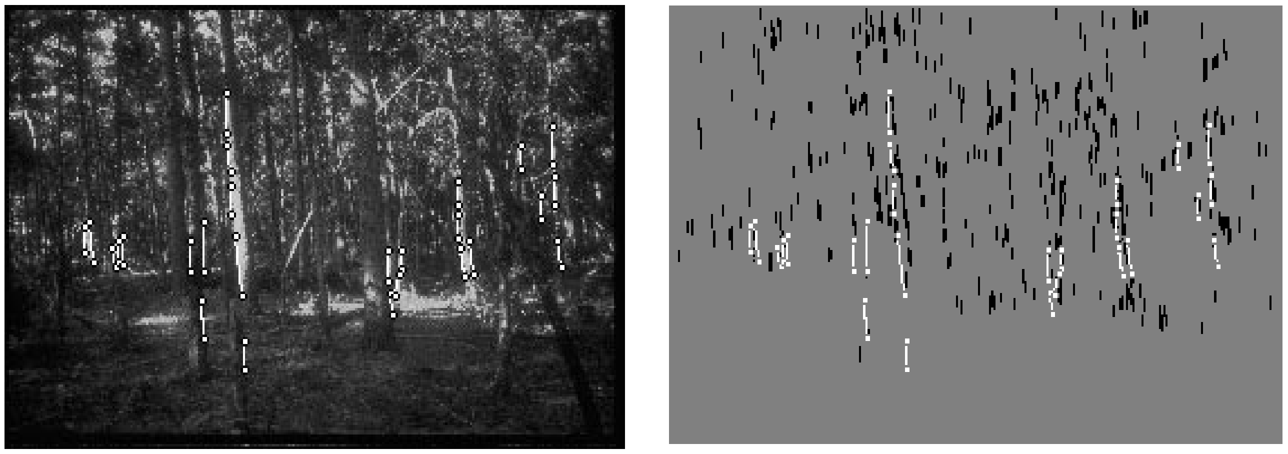

Figure 2: Example detection results with the optimal parameter setting. The image on the left is a manually transcribed image from data set 2, where the white lines are the transcriptions. The image on the right is the corresponding detection image under the optimal conditions (the detected lines are black). Note the inserted lines and the match lines.

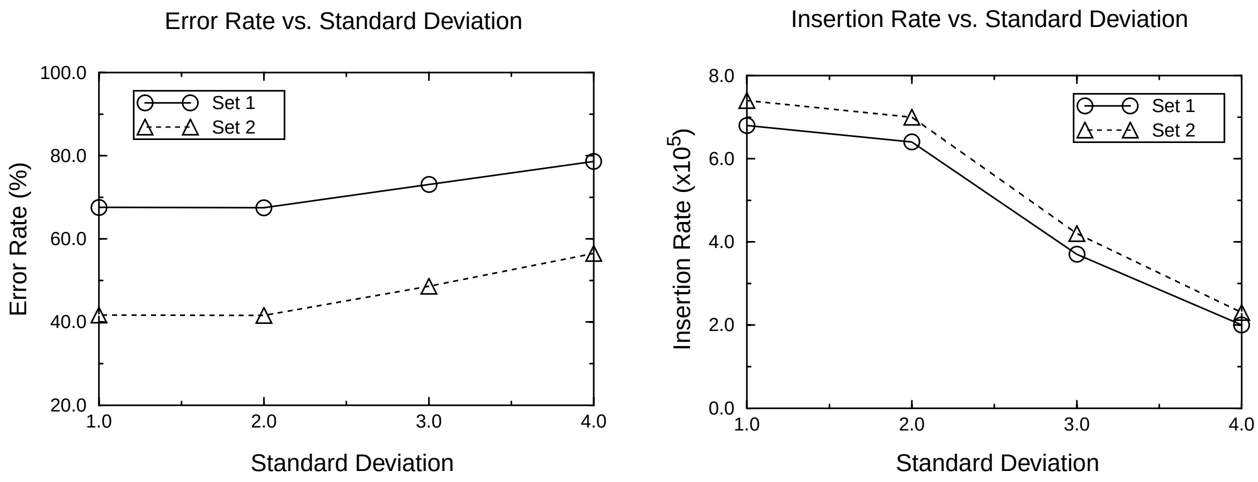

Figure 3: Results from optimization experiments with the Gaussian standard deviation involved in Canny edge detection algorithm. Data set 1 contained 165 images randomly chosen from the training set 01 of USFS Pre-phase 01 image data. Data set 2 consisted of 159 images randomly chosen from the test set 01 of Pre-Phase 01. The high edge threshold was 180; the low edge threshold was 60; and the line threshold was 15.

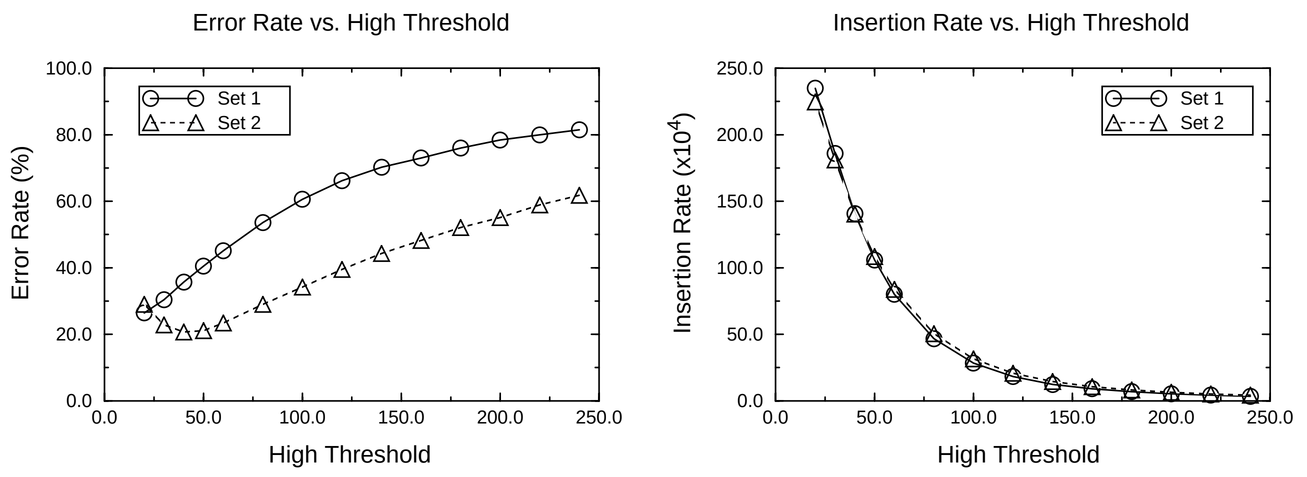

Figure 4: Results from optimization experiments with the high edge threshold involved in Canny edge detection algorithm. Data sets were the same as those in Figure 3. The Gaussian standard deviation was 2.0; the low edge threshold was set half of the high edge threshold as suggested in Canny's paper; and the line threshold was 25.

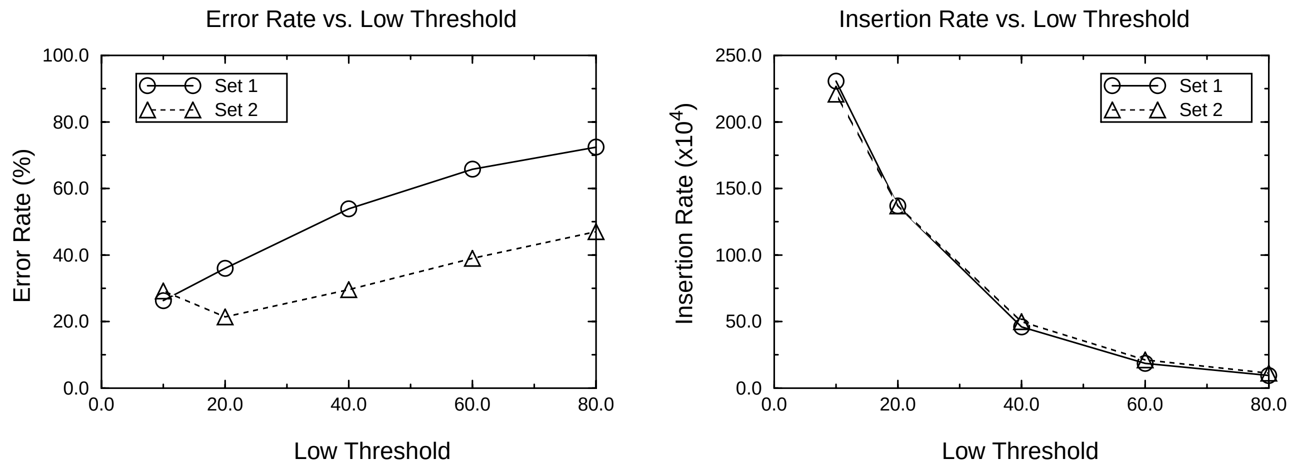

Figure 5: Results from optimization experiments with the low edge threshold involved in Canny edge detection algorithm. Data sets were the same as those in Figure 3 and 4. The Gaussian standard deviation was 2.0; the high edge threshold was 100; and the line threshold was 25.

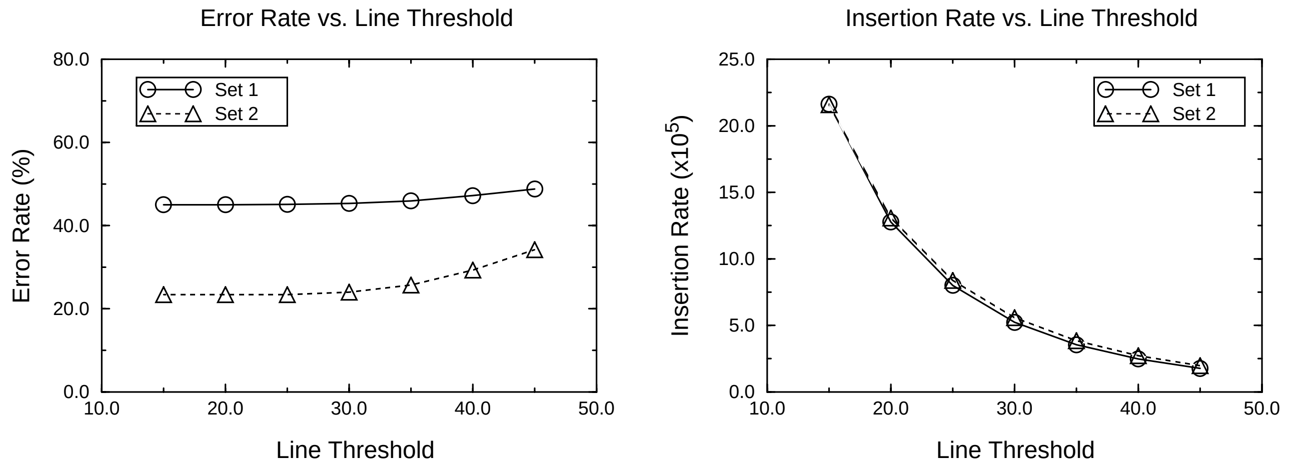

Figure 6: Results from optimization experiments with the line threshold involved in the line detection algorithm. Data sets were the same as those in Figure 3, 4 and 5. The Gaussian standard deviation was 2.0; the high edge threshold was 60; and the low edge threshold was 30.