A COMPARATIVE EVALUATION OF COMPETITIVE LEARNING ALGORITHMS FOR EDGE DETECTION ENHANCEMENT

Tuba Sirin[1], Mehmet Izzet Saglam[2], Isin Erer[3], Muhittin Gokmen[4], Okan Ersoy [5]

[1]Informatics Institute, Advanced Technologies in Engineering,
Computer Science and Engineering
[2]Informatics Institute, Advanced Technologies in Engineering,
Satellite Communication and Remote Sensing Program,
[3]Faculty of Electrical and Electronic Engineering,
Electronics and Communication Department,
[4]Faculty of Electrical and Electronic Engineering,
Computer Engineering Department,
Istanbul Technical University,
Maslak, Istanbul, 34469, TURKEY
[5]Purdue University, School of Electrical and Computer Engineering,
West Lafayette, Indiana, USA
[1]sirint@itu.edu.tr [2]misaglam@be.itu.edu.tr [3]ierer@ehb.itu.edu.tr [4]gokmen@cs.itu.edu.tr [5]ersoy@purdue.edu

ABSTRACT

Most edge detection algorithms include three main stages: smoothing, differentiation, and labeling. In this paper, we evaluate the performance of algorithms in which competitive learning is applied first to enhance edges, followed by an edge detector to locate the edges. In this way, more detailed and relatively more unbroken edges can be found as compared to the results when an edge detector is applied alone. The algorithms compared are K-Means, SOM and SOGR for clustering, and Canny and GED for edge detection. Perceptionally, best results were obtained with the GED-SOGR algorithm. The SOGR is also considerably simpler and faster than the SOM algorithm.

1. INTRODUCTION

Edge detection is a very important topic in image processing. Determining object boundaries in a still image is one aim of edge detection. The object boundaries are pixels at which sharp changes occur because of changes in surface orientation, depth, or physical properties of materials. The edge detectors should overcome the tradeoff between good localization and good detection.

There are many ways to perform edge detection. There are two kinds of edge detection methods based on filtering; gradient and Laplacian. In Fig. 1. we illustrate a 1-D example. $f(t)$ is the 1-D function of an input signal. The fast increment in the signal corresponds to an edge in the image. $f'(t)$ is the first derivative at this point, has a local extremum (maximum or minimum), and second derivative $f''(t)$ has a zero crossing.

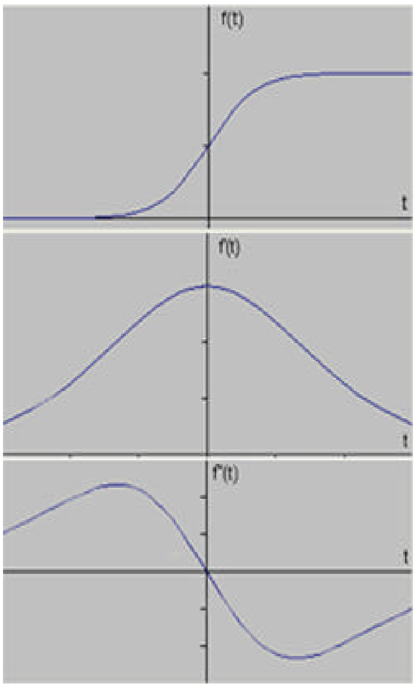


Fig. 1. Input signal $f(t)$, its first derivative $f'(t)$, and its second derivative $f''(t)$ for a typical 1-D edge.

The gradient-based methods use the maximum and minimum in the first derivative of the image to find the edges. An edge thinning algorithm may be necessary to improve the results since the initial edges are usually thick. In these methods, discontinuities in the edges are a common problem.

The Laplacian-based methods typically generate too many edge points, and may generate many false edge contours. One advantage of Laplacian-based methods is that an edge thinning algorithm may not be necessary.

Canny [1] developed an optimization theory approach to edge detection by considering three main criteria for the edge operator: 1. Good detection, 2. Good localization, 3. Only one response to one edge.

The Canny edge detector is a specific algorithm (filter) which includes these constraints in mathematical form. First, the Canny filter smoothens the image with Gaussian convolution. Edges are marked at maximum of the gradient after convolving the image with the optimal filter.

The Generalized Edge Detector (GED) is a more recent method which is a hybrid model comprised of the linear combination of membrane and thin-plate functionals [2].

Goal of segmentation is to reduce the amount of data by categorizing or grouping similar data items together. There are several segmentation methods in the literature; such as Self-Organizing Map (SOM), K-Means, and Vector Quantization (VQ). We have developed a new segmentation method called Self-Organizing Global Ranking (SOGR) algorithm discussed in Section 3.3.

A grayscale edge detector (Canny) was used with SOM in multispectral images previously [3].

The paper is organized in six sections. The GED is discussed in the next section. The clustering algorithms are discussed in Section 3. The two-stage algorithm is described in Section 4. The comparative experimental results are discussed in Section 5. Conclusions and future research are given in Section 6.

2. GENERALIZED EDGE DETECTION

The relationship between regularization theory and convolution with filters are explored by the Generalized Edge Detector. A general framework used to convert an ill-posed problem to a well-posed problem by restricting the class of admissible solutions using the constraints such as smoothness is regularization theory [4]. There are two kinds of edge detector operators. The first one is related to minimizing a membrane functional $E_m(f)$. The operators in the second group are related to minimizing a plate functional $E_p(f)$. The one-dimensional hybrid energy functional was considered, and the corresponding filter associated with it was found. The related functionals are

$$E_m(f)=\iint_\Omega (f(x,y)-d(x,y))^2 dxdy + l\iint_\Omega (f_x^2+f_y^2)dxdy \quad (1)$$

$$E_p(f)=\iint_\Omega (f(x,y)-d(x,y))^2 dxdy + l\iint_\Omega (f_{xx}^2+2f_{xy}^2+f_{yy}^2)dxdy \quad (2)$$

The function that minimizes the one-dimensional hybrid functional

$$E_{lt}(f;x)=\int_\Omega \left\{ b(f-d)^2 + l\left[(1-t)f_x^2+t f_{xx}^2\right]\right\}dx \quad (3)$$

is found by solving the associated Euler-Lagrange equation given by

$$l\,t\,f_{xxxx} - l(1-t)f_{xx} + f = d \quad (4)$$

3. SEGMENTATION

Segmentation algorithms separate a dataset into different membership areas. Categorizing or grouping similar data items together is the goal of segmentation. Segmentation algorithms such as Self Organizing Map [5], K-Means [6], Vector Quantization, and Self Organizing Global Ranking (SOGR) algorithm [7] has many applications, for example, in remote sensing, data compression, signal processing, and data mining.

3.1 K-Means

K-means is one of the first and still popular segmentation methods [6]. The criterion function to be minimized in K-means is the average squared distance of the data items x_k from their nearest segment centroids:

$$E_k = \sum_k \left\| x_k - m_{c(x_k)} \right\|^2 \quad (5)$$

where $c(x_k)$ is the index of the centroid that is closest to x_k. For minimizing the cost function, one possible algorithm begins by initializing a set of K cluster centroids denoted by $m_i, i = 1,...,K$, and iteratively updates them until convergence.

3.2 Self-Organizing Map

The self-organizing map (SOM) [5] has been used in a wide variety of applications. The procedure for learning a SOM is as follows:

Initialization: three methods used for this purpose are

1. Random initialization
2. Sample initialization
3. Linear initialization

For example, in the first method, before training with the SOM, the weight (reference) vectors are initialized with random values.

Training: There are two stages in the training:

a. First, a sample vector is chosen from the input data vectors randomly. Then, the similarity between it and all the weight vectors of the map is calculated, and a winner is chosen by

$$\left\| x - m_c \right\| = \min_i \left\{ \left\| x - m_i \right\| \right\} \quad (6)$$

Up to this point, the training process is called winner node search stage.

b. In the adaptation stage, we update the weight vectors in the map as shown below:

$$m_i(t+1) = \begin{cases} m_i(t) + a(t)\left[x(t) - m_i(t)\right] & i \in N_c \\ m_i(t) & \text{otherwise} \end{cases} \quad (7)$$

This adaptation procedure moves the prototypes of the best matching unit (BMU) and its topological neighbors towards the sample vector.

Stages one and two are repeated during the training process. The clusters that correspond to characteristic features are formed into the map automatically. A number of clusters are generated.

3.3 Self-Organizing Global Ranking Algorithm

The Self Organizing Global Ranking (SOGR) algorithm is a simple iterative algorithm[7]. It is similar to SOM in terms of updating codebook vectors, but it is quite different from SOM in terms of definition of neighborhood. In the SOGR, neighbor codebook vectors are chosen globally based on similarity ranking, and hence they can be anywhere in the relevant space of vectors.

During iterations, the neighbor codebook vectors are chosen based on similarity ranking. Hence, a distance measure is required. The initial codebook vectors can be chosen randomly from the input data. During an iteration, a pixel (input vector) is chosen randomly, and then the distances between all codebook vectors and this pixel are calculated. Assuming the integer R is the neighborhood size, The closest R codebook vectors are chosen as the (winning) neighbors. After that, we update the codebook vectors by

$$m_i(t+1) = \begin{cases} m_i(t) + b_i(t)\left[x(t) - m_i(t)\right] & i \in R \\ m_i(t) & otherwise \end{cases} \quad (8)$$

where b_i is the learning rate. It is chosen to decrease with time, for example, exponentially. It is advisable to choose it differently for each neighborhood size R. For large R, the learning rate should be chosen smaller. Furthermore, when the iteration number increases, R should be decreased, for example, linearly. These changes help move in the direction of the global minimum and avoid local minima.

In order to use any of the above competitive learning algorithms in supervised learning, it is necessary to label the final codebook vectors with appropriate classes. After the procedure of updating the codebook vectors, the distances between each training pixel and all codebook vectors are calculated. The nearest codebook vector is chosen the winner, and a counter in the node related to the winning codebook vector is increased by one for the corresponding class. This process is repeated for all the training pixels. After this process, the label of the counter which is the maximum for a codebook vector is chosen as the label of the codebook vector. Thus, the counter with the maximum value indicates the class label for the codebook vector.

4. COMPETITIVE LEARNING-EDGE DETECTION

We evaluate the performance of algorithms in which competitive learning is applied first to enhance edges, followed by an edge detector to locate the edges. In this way, more detailed and relatively more unbroken edges can be found as compared to the results when edge detector is applied alone. The algorithms compared are

1. Clustering Algorithms: K-Means, SOM and SOGR
2. Gray Scale Edge Detection Algorithms: Canny and GED

5. EXPERIMENTAL RESULTS

The first comparative evaluations were done with the checkerboard image shown in Figure 2(a). The results with different combinations of algorithms are shown in Figures 2(b) thru 2(j).

Various amounts of noise were also added to the checkerboard image. The corresponding results are shown in Figures 3(a) thru 3(i), and Figures 4(a) thru 4(i).

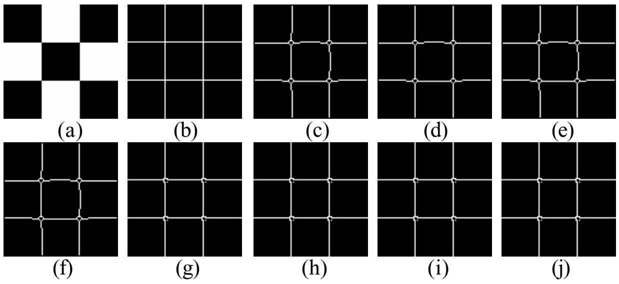


Fig. 2. (a)The checkerboard image, (b)Exact edges, (c)Result of Canny operator, (d)Result of K-Means-Canny, (e)Result of SOM-Canny, (f)Result of SOGR-Canny, (g)Result of GED, (h)Result of K-Means-GED, (i)Result of SOM-GED, (j) Result of SOGR-GED.

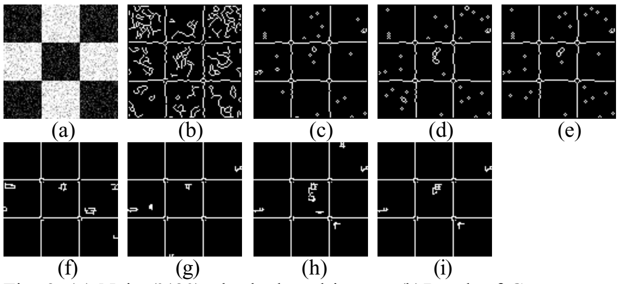


Fig. 3. (a) Noisy(%20) checkerboard image, (b)Result of Canny operator, (c)Result of K-Means-Canny, (d)Result of SOM-Canny, (e) Result of SOGR-Canny, (f)Result of GED, (g)Result of K-Means-GED, (h)Result of SOM-GED (i)Result of SOGR-GED

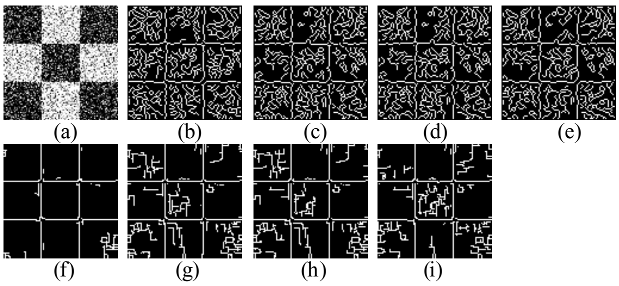


Fig. 4. (a) Noisy(%50) checkerboard image, (b)Result of Canny operator, (c)Result of K-Means-Canny, (d)Result of SOM-Canny, (e) Result of SOGR-Canny, (f)Result of GED, (g)Result of K-Means-GED, (h)Result of SOM-GED (i)Result of SOGR-Canny.

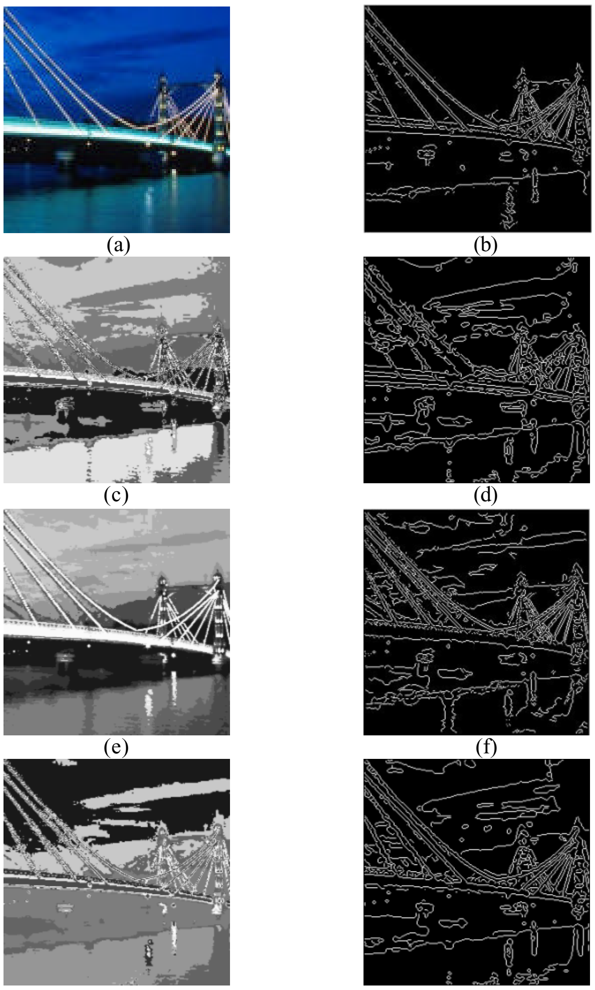
Error rates for false edges are given in Table 1. It is observed that the K-Means–Canny or SOM – Canny or SOGR-Canny works less erroneously than using Canny alone. Similarly, K-Means–GED or SOM – GED or SOGR-GED works less erroneously than GED alone at reasonable noise ratios. However, at very high noise, GED based methods have less error rate than Canny based methods.

In the second set of evaluations, the algorithms were tested on the bridge image shown in Figure 5(a). Perceptionally, the best results were obtained with the

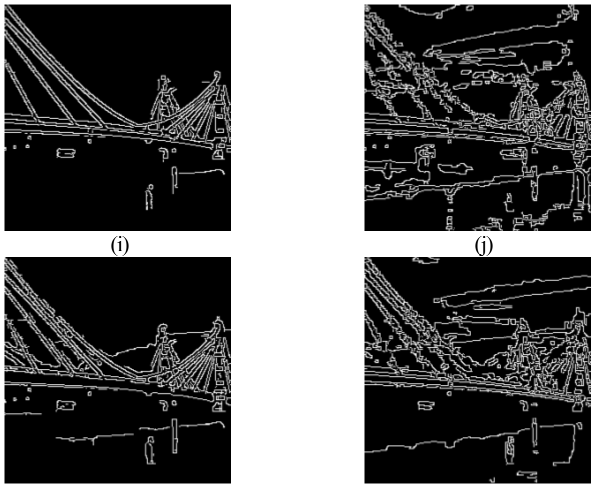
SOGR-Canny or SOGR-GED algorithms, the last one also being somewhat better than the first one. The SOGR algorithm is also considerably simpler and faster to compute with than the SOM algorithm due to simple definition of neighborhood. Similar conclusions were also reached in classification applications previously [7].

Table 1. Error Analysis on the checkerboard image.

%	ERROR RATE		
Noise Ratio	**0**	**20**	**50**
CANNY	4.1667	13.4766	21.5386
K-MEANS -CANNY	2.8212	5.3277	20.4861
SOM – CANNY	4.1667	5.8594	20.1931
SOGR – CANNY	4.1667	4.9588	19.5313
GED	0.3906	1.3455	2.3220
K-MEANS – GED	0.3906	1.0525	8.7674
SOM – GED	0.3906	1.5625	8.8976
SOGR – GED	0.3906	1.1502	8.9952



(a) (b) (c) (d) (e) (f) (g) (h)



(i) (j) (k) (l)

Fig. 5. (a) The bridge image, (b) Result of Canny, (c) Segmented image with K-Means, (d)Result of K-Means-Canny, (e) Segmented image with SOM, (f) Result of SOM-Canny, (g) Segmented image with SOGR, (h) Result of SOGR-Canny, (i) Result of GED, (j) Result of K-Means-GED, (k) Result of SOM-GED, (l) Result of SOGR-GED.

6. CONCLUSIONS AND FUTURE WORK

A comparative evaluation of two-stage edge detection algorithm involving clustering (segmentation) followed by an edge detector is presented. The algorithms involved were K-Means, SOM, and SOGR for segmentation, and Canny and GED algorithms for edge detection. The performance of the two-stage algorithm is superior to one stage edge detection. In general, the best results were obtained with the SOGR-GED algorithm at reasonable noise levels. The SOGR is also much simpler and faster to compute with than the SOM algorithm due to simpler definition of neighborhood which does not involve overhead computations.The two-stage algorithm is also more sensitivite to noise than the one-stage algorithm. At high noise, GED by itself shows better performance. Future work will involve reduction of such sensitivity, for example, by prefiltering of noise.

REFERENCES

[1] J.F. Canny, “A Computational Approach to Edge Detection,” IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol. 8, 1986, pp. 679-698.

[2] M. Gokmen and A.K. Jain, “-Space Representation of Images and Generalized Edge Detector,” IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.9, 1997, pp. 545-563.

[3] P.J. Toivanen, J. Ansamaki, J.P.S. Parkkinen, J. Mielikainen, “Edge Detection in Multispectral Images Using The Self Organizing Map,” Elsevier Pattern Recognition Letters, vol. 23, 2003, pp. 2987-2994.

[4] M. Bertero, T. Poggio, and V. Torre, “Ill-posed Problems in Early Vision,” Technical Report, MIT AI Lab., AI Memo 924, 1987.

[5] T. Kohonen, The Self-Organizing Map, Proc. IEEE, vol. 78, no. 9, Sept. 1990, pp. 1464-1480.

[6]MacQueen, “Some methods for classification and analysis of multivariate observations,” *Proceedings of Fifth Berkeley Symposium on Math. Statist. and Prob.*, 1967, pp. 281-297.

[7] M.I. Saglam, O. Ersoy, and I. Erer, “Self Organizing Global Ranking And Its Applications,” *Proceedings of the Conference on Smart Engineering Design (ANNIE 04)* in St. Louis, November 2004.