Real-time Edge Segment Detection with Edge Drawing Algorithm

Cihan Topal, Ozgur Ozsen and Cuneyt Akinlar
Department of Computer Engineering
Anadolu University
Eskisehir, TURKEY
{cihant, oozsen, cakinlar}@anadolu.edu.tr

Abstract—**Edge Drawing (ED) is an edge detection method that works by computing a set of anchor points, which are most likely to be edge elements, and linking them with a predefined set of rules which we call smart routing. ED can thus obtain one-pixel wide, contiguous and smooth edges in real-time. In most cases, obtaining the edges in segment form (i.e. blob of connected edgels) is desired. Because edge segments also include the connectivity information among the pixels, they contain useful features about the geometric shape of the detected edges. In contrast to conventional detectors, ED is able to output the resulting edge map as vectorial data, i.e., array of edge segments instead of a 2D binary image. In this work, this capability of ED is explained and discussed with examples.**

I. INTRODUCTION

Edge detection studies constitute a very crowded set in the image processing literature [1-6]. Since edge detection is an essential step in most image processing methods, there are many studies on evaluating their efficiency [7-12]. In these studies, smooth, contiguous, thin and well-localized edges are qualified as the high quality edges. There are also studies which seek to improve the perceptual structure of produced edge maps by several post-processing techniques [13-21]. These methods take a binary edge map as input and try to improve the edge quality by performing rule-based techniques with pixel-wise templates. Thus, they track and test the edge structures with the templates and try to link possible discontinuities. Although these methods help optimize the modal characteristics of the edges with morphological operations, they cannot provide a certain solution to having a high-quality edge map.

There is also a substantial demand for detecting the edges in segment fashion; in other words, within the connectivity information among the edge elements (edgels). After an edge segment is obtained, it can be chain-coded with the relative position vectors in-between the consecutive edgels [22]. Thus significant information about the geometric structure and orientation of the segment can be derived by analyzing the distribution and moments of the chain code [23-26]. Thereby, very useful features can be obtained for detection [27-29], recognition [30, 31], and registration [32, 33] applications.

If one wants to obtain the edge segments using the methods mentioned in the previous paragraph, s/he would first try to obtain the binary edge map, and then apply a connected component analysis on the edge image. Such analysis typically results in segments with ragged, discontiguous and multi-pixel wide edges. This is because the conventional edge detectors evaluate pixels in an isolated manner without considering the edge status of neighboring pixels and they generate edge maps consisting of discontinuous and unattended pixels.

On the contrary, Edge Drawing (ED) is a novel detector that obtains one-pixel wide, smooth and contiguous edges in real-time [34]. It works by computing a set of points most likely to lie on an edge, called anchors, and linking them by a set of predefined rules which we call smart routing. In other words, ED draws the edges by using the information derived from the image; whereas, conventional methods obtain the edges by filtering and eroding the image elements. The inspiration for the method name comes from the smart routing operation, i.e., drawing the edges between the anchor points just like a human would do.

Obtaining edges in high level qualifications is not the only competence of ED, but also getting the edge map in vectorial fashion, i.e., array of segments instead of a 2D binary edge image. By its high-level cognitive reasoning algorithm, ED behaves as if it has a perceptual distinguishing ability just like humans. Thus, it can detect edge segments as separated pixel chains even if those chains are connected in a complex structure, i.e. having multiple branches.

In the rest of the paper, a short overview of the ED algorithm is presented in Section 2, details about extracting edges in vectorial form with ED is explained in Section 3, experimental results are given in Section 4, and the paper is concluded in Section 5.

II. EDGE DRAWING (ED) OVERVIEW

ED consists of four main steps:

1. Noise reduction with Gaussian filtering,
2. Determination of edge areas and edge direction map construction,
3. Extraction of anchor points,
4. Linking anchor points with smart routing.

A. Noise Reduction

The very first step of ED is similar to most of the image processing methods. We simply convolve the image with a Gaussian kernel to reduce the effect of noise.

B. Finding the Edge Areas and Direction Map

After smoothing, the first order horizontal and vertical derivatives of the image are computed to determine the edge areas and the edge direction map. Edge areas can be defined as the regions where the pixels have a gradient magnitude greater than a certain threshold. Although the Sobel operator [1] serves well in this task, one-dimensional simpler kernels can also be used for the sake of performance. Because the major efficiency of ED comes with the edge drawing procedure, various gradient operators, e.g., Prewitt, Scharr, etc., would also work fine.

The horizontal and vertical gradients (G_x and G_y) are first obtained for each pixel by the derivative operator. Next, a gradient magnitude image is obtained by simply adding G_x and G_y, i.e., $G = |G_x| + |G_y|$. Finally, a threshold is applied to determine if a pixel lies on an edge area. In Fig. 2.a gradient magnitude image of the Peppers image is shown.

Simultaneously with the gradient image computation, edge direction map is also constructed in order to route the linking process of anchor points at the last step. The direction map is constructed in the same size with the image by comparing the $|G_x|$ and $|G_y|$ values. The edge passing through the pixel is assumed to be horizontal if $|G_x| < |G_y|$, vertical otherwise. Direction information is not computed for pixels whose gradient magnitudes are smaller than the threshold. The block diagram shows the first two steps of the ED algorithm in Fig. 1.

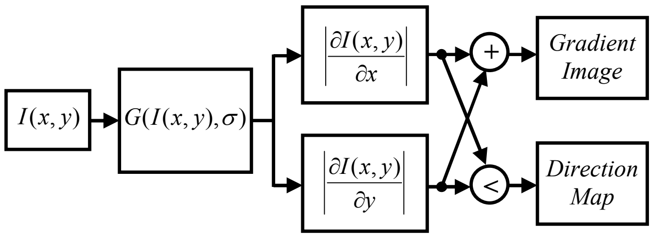

$I(x,y)$
$G(I(x,y),\sigma)$
$\left|\frac{\partial I(x,y)}{\partial x}\right|$
$\left|\frac{\partial I(x,y)}{\partial y}\right|$
+
<
Gradient Image
Direction Map

Figure 1. Construction of gradient magnitude and edge direction maps.

C. Extraction of Anchor Points

The next step is the extraction of some points which must be lying on an edge with a high probability, called anchors. Anchors can be defined as local maxima points of the gradient magnitude image and are selected by scanning the image in horizontal and vertical directions.

The density of the anchors can be tuned on demand. For instance, if only the detection of the major edges is desired, anchors can be extracted sparsely. If a lot of detail is required, anchors can be extracted more densely. Fig. 2.b shows anchors extracted from the Peppers image with a scan interval of 16, i.e., every 16th row or column is scanned.

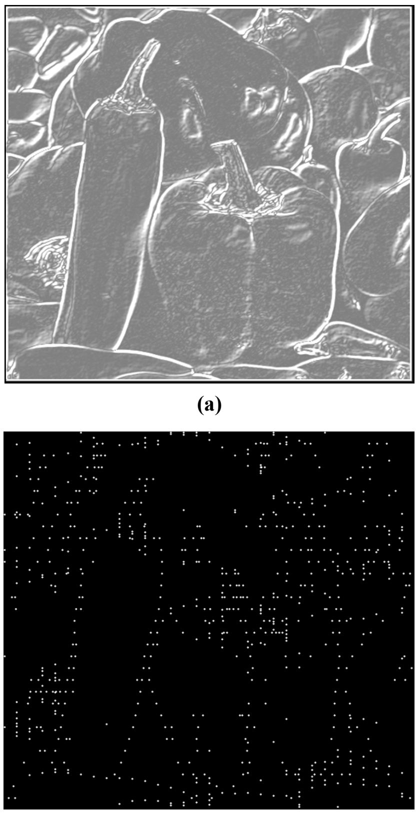


Figure 2. **a)** Gradient magnitude image of Peppers, **b)** Anchors extracted from (a).

D. Linking of Anchor Points by Edge Drawing

In the last step of ED, anchors are linked together with the help of the gradient magnitude image and direction map with a smart routing algorithm.

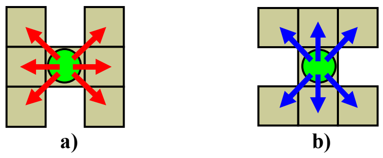


Figure 3. **a)** Horizontal, **b)** vertical proceedings in smart routing.

The gradient magnitude image is raster scanned for the anchor points and the first encountered one is picked up. Then the edge direction of the selected anchor point is checked from the direction map. According to the edge direction, horizontal (Fig.3.a) or vertical (Fig.3.b) routing is performed over the gradient magnitude image. During routing, neighboring pixels in the linking direction are checked, and the one having the greatest gradient magnitude is selected as the next hop. The routing continues until we hit the boundary of the thresholded gradient magnitude image or a previously computed edge segment is encountered. Thus obtaining one-pixel wide, contiguous and smooth edges is guaranteed. Localization is another important edge quality metric for edges and it is also provided by selecting the pixel which has the maximum response to the gradient operator.

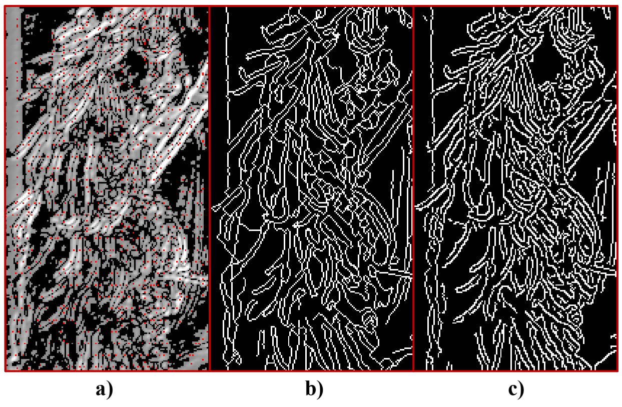


Figure 4. Close view of feathers on Lena's hat, a) thresholded gradient magnitudes and anchors (red dots), b) edge drawing result, c) OpenCV Canny result.

Fig. 4 shows the edge results of a very tough region in the Lena image; the feathers on the Lena's hat. Fig.4.a shows the thresholded gradient magnitude image with extracted anchors (red dots), and Fig.4.b shows the edge map by ED. OpenCV Canny (available online at http://opencv.willowgarage.com/) result is shown in Fig.4.c. Both algorithms' parameters were set to obtain the same level of detail in the final edge map; however, as we decrease Canny's threshold values to get more details, the noise becomes dominant instead of edges. Note that there are notches, unattended pixel formations and multi-pixel wide edges in Canny's output while ED's result consists of one-pixel wide, smooth and contiguous pixels. For a more detailed definition of ED algorithm, please refer to the [34].

III. Edge Segment Detection with ED

Besides detecting high quality edges in real-time, ED is also able to output the result in vectorial fashion, i.e., as a list of edge segments. In most applications, edges have to be detected in segment form to apply further operations on the edge features [23-33]. In most studies, the traditional trend on edge segment detection is edge detection followed by a connected component analysis. Thus, edge blobs consisting of connected edge elements can be detected and further geometric and structural analysis (such as line fitting, curve approximation, etc.) can be done.

To get the maximum efficiency from the edge segments, obtaining them regardful to the determined edge qualifications is a must. On the contrary, a connected contour might be detected in multiple portions or notch-like formations on the edge induce incoherent positional vectors and irrelevant chain code. Since ED picks pixels one by one in the order they are met, it is also able to accumulate the chain code information by simply recording the positional difference vectors between the last two collected edgels. Thus, the chain code representation of the edges obtained by ED comprises reliable information about the modal structure of the segment. Behind any doubt, the efficiency of ED comes from its authentic high-level cognitive reasoning.

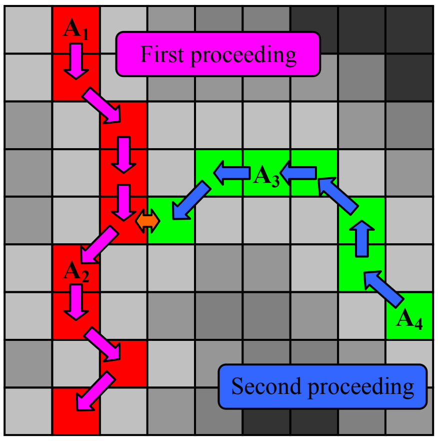


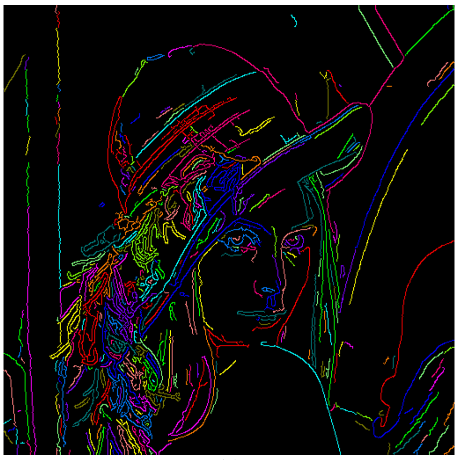
Figure 5. Illustration of ED's segment detection strategy.

ED ensures having one-pixel wide contiguous edges by extending edge map with edgel selections from neighbor pixels in smart routing step. Thus, robust and resistive edge structure can also be made up against noise. Another very important property of ED's high-level cognitive reasoning is its property of detecting adjoint segments as separate segments in a manner of multiple pixel chains.

This property is explained with the illustration in Fig. 5. In the smart routing step, an anchor point is selected and the edge is started to be drawn with the help of gradient magnitudes and direction map. During this operation, two conditions are being checked to stop the smart routing and abort the detection procedure; 1) moving out the edge areas, 2) encountering a previously detected edge. In Fig. 5, the first linking starts from the anchor point A1 and proceeds down. Then another anchor A2 is encountered and the operation continues until the image boundary is met. After that, another proceeding procedure is launched from A4 and proceeds through the left. In this second proceeding, A3 anchor is encountered and the operation continues until it touches to the previously detected edge segment, i.e. red edge segment. Thus red and green edge segments can be detected separately.

If we detect the edges with another method and apply a connected component analysis (CCA), we would detect just one edge segment. Then if we would apply a post-processing in order to find the junction pixels and partition the edge segments, we would detect 3 separated edge segments, i.e. we would detect the red edge segment in two parts. These scenarios are valid in case we are able to detect the edges with the same structural quality with another method. If the resulting edge map includes discontinuities or notches, very different results could be obtained far away from the expected ones.

In Fig. 6, the efficiency of ED's high-level cognitive reasoning is presented with a simple experiment. In Fig. 6.a Lena's edge segment results are indicates in different colors obtained by OpenCV implementation of Canny algorithm and 8-neighborhood connected component analysis (8N CCA). In the Fig. 6.b, the same experiment is run with the Edge Drawing algorithm.


(a)

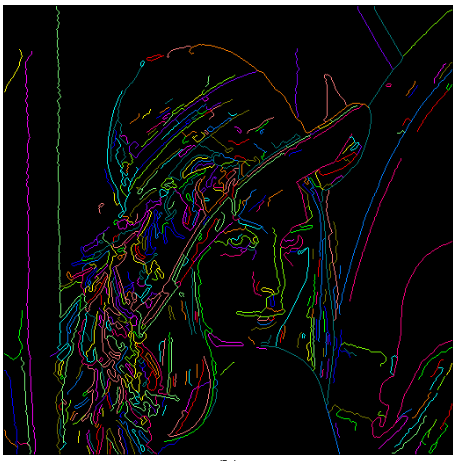

(b)

Figure 6. **a)** Edge segments of Lena by Canny + 8N CCA, **b)** Edge segments obtained by Edge Drawing algorithm.

Please note that the edge quality in ED's result is very appreciable with the contiguous, one-pixel wide and smooth edge segments. The effect of ED's high-level cognitive reasoning is also quite noticeable on the Lena's hat's feathers. In the Canny + 8N CCA result, obtained edge segments consist of multiple chains and looks like a regional blob of pixels. In the meantime, ED detects the same area more efficiently, i.e., each segment is an individual chain of edgels. Because in Fig. 6.a multiple chains of edgels are detected as one segment, the total number of detected edge segments is smaller than the ED's result. In the experiment, 386 edge segments are obtained in Fig. 6.a, whereas 531 segments are obtained in the Fig. 6.b.

IV. Edge Segment Detection with ED

In this section, our goal is to perform edge segment detection tests and compare the ED's results with OpenCV Canny followed by Connected Component Analysis (CCA) for four different images.

Fig.7.a shows Canny's edge segments and Fig.7.b shows ED's edge segments, where each color represents a different edge segment. Fig.7.c and Fig.7.d shows the same edge segments but they are colored based on their lengths. Specifically, segments longer than 200 pixels are colored red, segments with length between 50 and 200 pixels are colored yellow, segments with length between 10 and 50 pixels are colored green and segments shorter than 10 pixels are colored blue (see Fig. 7.e for color legend).

TABLE I. Number of Edge Segments in Fig. 7.a and b.

Image	Canny + CCA	Edge Drawing
Mandril	1966	2538
Boats	735	962
Peppers	418	451
Bicycle	798	912

From the figure, it is clear that ED's segments are clean and accurate as each one is a chain of pixels; whereas Canny + 8N CCA's edge segments are noisy and cover wider areas on the image. Because of this, Canny + 8N CCA's edge segments consist of many more edgels and form more complex structure. As a consequence, the number of detected segments is greater in ED's outputs (see Table 1). The images in Fig.7 c and Fig.7.d also prove that the number of ED's segments is greater; hence its segments are shorter. Note that the red regions are more common in the images in Fig. 7.c. This is a simple indication that multiple chains are detected as one segment and constitutes a crowded blob of edgels.

TABLE II. Running Time Experiments for Images in Fig. 7.

Image	Canny + CCA	Edge Drawing
Mandril	18 msec	15 msec
Boats	12 msec	9 msec
Peppers	11 msec	8 msec
Bicycle	12 msec	9 msec

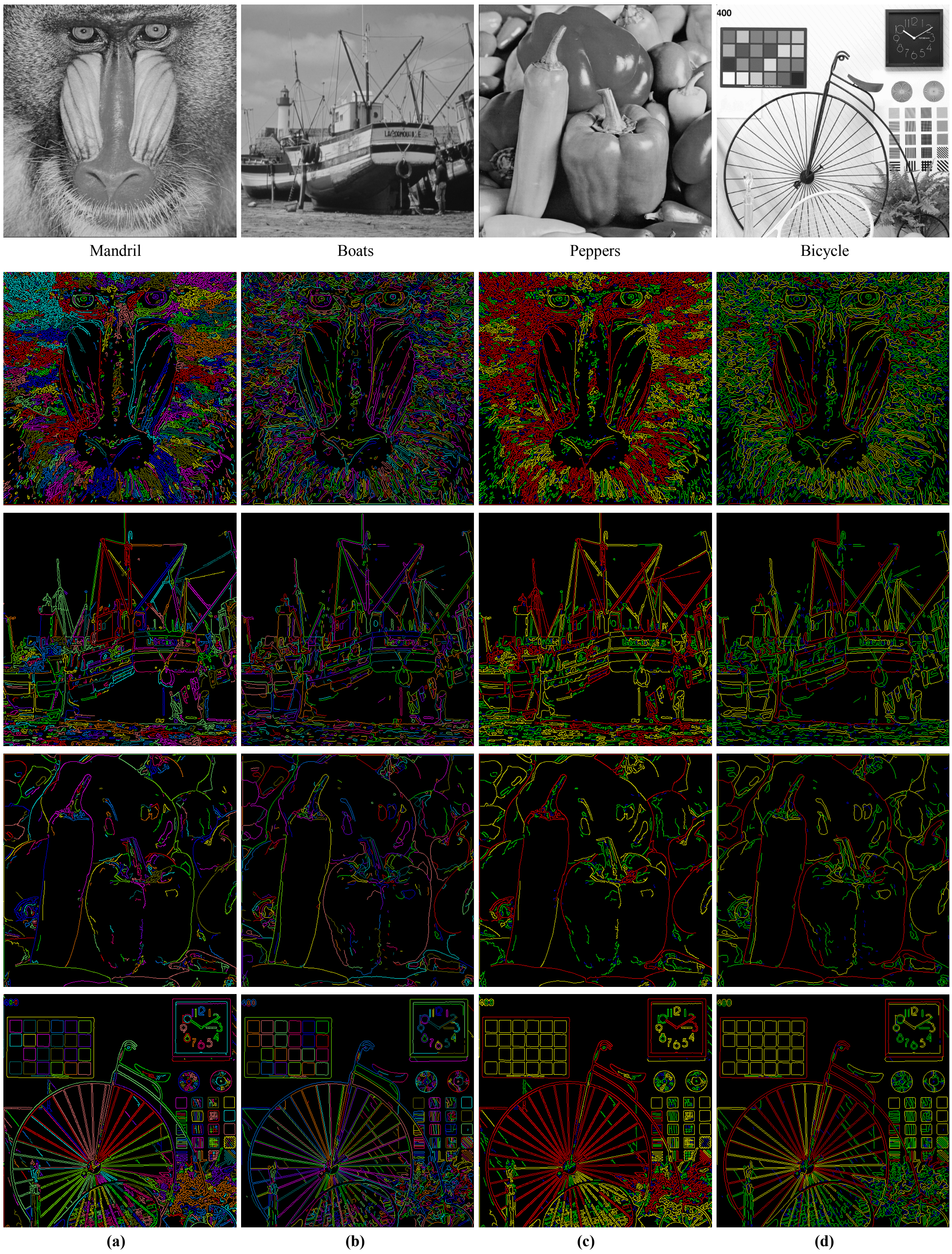
We conclude that not only ED produces higher quality edge segments, but also it is much faster than Canny + 8N CCA as shown in the list of running times in Table 2. All these experiments were performed in a single-threaded application on a typical desktop computer with a 2 GHz CPU and 2 GB of RAM. The reader can also perform many more experiments by visiting ED demo page at [35].

V. Conclusions

ED is a novel, real-time edge segment detection algorithm. We can summarize the properties of ED as follows:

- detects one-pixel wide, smooth, contiguous and well-localized edges,
- works in real-time,
- is able to give the output in segment form without additional computational time,
- can accumulate the corresponding chain code of the segment if desired.

In this study, convenience of ED algorithm for real-time edge segment detection is examined and justified with the extensively performed experiments. Presented results show that the ED is a promising edge segment detection algorithm for real-time applications.



Color legend for edge length maps: 🟦 < 10 🟩 < 50 🟨 < 200 🟥 > 200

(e)

Figure 7. **a, b)** Edge segments obtained with Canny + 8N CCA and ED, respectively.
c, d) Edge length maps of Canny + 8N CCA and ED, respectively.
e) Color legend for edge segments in c and d.

REFERENCES

[1] E. Sobel, “Camera Models and Machine Perception”, PhD thesis, Stanford Univ., 1970.

[2] V. S. Nalwa and T. O. Binford, “On Detecting Edges”, IEEE Trans. PAMI, vol. 8, no. 6, pp. 699-714, Nov. 1986.

[3] J. Canny, “A Computational Approach to Edge Detection,” IEEE Trans. PAMI, vol. 8, no. 6, pp. 679- 698, Nov. 1986.

[4] L. A. Iverson and S.W. Zucker, “Logical/Linear Operators for Image Curves”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 17, no. 10, pp. 982-996, Oct. 1995.

[5] F. Bergholm, “Edge Focusing”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol.9, no.6, pp.726, Nov. 1987.

[6] C. A. Rothwell, J. L. Mundy, W. Hoffman, and V. -D. Nguyen, “Driving Vision by Topology”, Int’l Symp. Computer Vision, pp. 395- 400, Coral Gables, Nov. 1995.

[7] V. Ramesh and R. M. Haralick, “Performance Characterization of Edge Detectors”, SPIE, vol. 1,708, Applications of Artificial Intelligence X: Machine Vision and Robotics, pp. 252-266, 1992.

[8] J. R. Fram and E. S. Deutsch, “On the Quantitative Evaluation of Edge Detection Schemes and Their Comparison With Human Performance”, IEEE Trans. Computers, vol. 24, no. 6, pp. 616-628, June 1975.

[9] D. J. Bryant and D. W. Bouldin, “Evaluation of Edge Operators Using Relative and Absolute Grading”, Int’l Conf. Pattern Recognition and Image Processing, pp. 138-145, Chicago, 1979.

[10] R. N. Strickland and D. K. Cheng, “Adaptable Edge Quality Metric”, Optical Eng., vol. 32, no. 5, pp. 944-951, May 1993.

[11] T. Kanungo, M. Y. Jaisimha, J. Palmer, and R. M. Haralick, “A Methodology for Quantitative Performance Evaluation of Detection Algorithms”, IEEE Trans. Image Processing, vol. 4, no. 12, pp. 1,667-1,674, Dec. 1995.

[12] M. D. Heath, S. Sarkar, T. Sanocki, K. W. Bowyer, "A Robust Visual Method for Assessing the Relative Performance of Edge-Detection Algorithms”, IEEE Trans. on PAMI, vol.19, pp. 1338-1359, Dec. 1997.

[13] Shing-Min Liu, Wei-Chung Lin, Cheng-Chung Liang, ”An iterative edge linking algorithm with noise removal capability,” Int’l Conf. Pattern Recognition (ICPR), pp.1120-1122, 1988.

[14] F.R. Miller, J. Maeda, H. Kubo, “Template based method of edge linking using a weighted decision,” IEEE/RSJ Conf. on Intelligent Robots and Systems (IROS), pp.1808-1815, 1993.

[15] J. Basak, B. Chanda, D.D. Majumder, “On Edge and Line Linking with Connectionist Models,” IEEE Trans. Systems, Man and Cybernetics, vol.24, no.3, pp.413-428, March 1994.

[16] A. Hajjar, T. Chen, “A VLSI Architecture for Real-Time Edge Linking,” IEEE Trans. Pattern Anal. Mac. Intell. (PAMI), vol.21, no.1, pp.89-94 Jan 1999.

[17] Z. Wang, H. Zhang, “Edge Linking Using Geodesic Distance and Neighborhood Information,” IEEE/ASME Int’l Conf. Advanced Intelligent Mechatronics (AIM), pp.151-155, 2008.

[18] Ya-Ping Wong, V. Chien-Ming Soh, Kar-Weng Ban, Yoon-Teck Bau, ”Improved Canny Edges Using Ant Colony Optimization,” Int’l Conf. Computer Graphics, Imaging and Visualisation (CGIV), 2008.

[19] A. Jevtic, I. Melgar, D. Andina, ”Ant based edge linking algorithm,” IEEE Int’l Conf. Industrial Electronics (IECON), 2009.

[20] J. Rahebi, Z. Elmi, A. Farzamnia, K. Shayan, ”Digital image edge detection using an ant colony optimization based on genetic algorithm,” IEEE Int’l Conf. Cybernetics and Intelligent Systems (CIS), 2010.

[21] Q. Lin, Y. Han, and H. Hahn, ”Real-Time Lane Departure Detection Based on Extended Edge-Linking Algorithm,” Int’l Conf. Computer Research and Development, 2010.

[22] H. Freeman, “Computer processing of line drawing images,” Computer Surveys, vol. 6, pp. 57-98, 1974.

[23] S. Marshall, “Review of shape coding techniques”, Image and Vision Computing, 7(4):281–294, November 1989.

[24] L. Gupta and M. D. Srinath, “Contour sequence moments for the classification of closed planar shapes”, Pattern Recognition, 20(3):267–272, 1987.

[25] J. Iivarinen, M. Peura, J. Särelä, and A. Visa, “Comparison of Combined Shape Descriptors for Irregular Objects”, Proc. of the Eight British Machine Vision Conference, UK, 1997.

[26] S. Kiranyaz, M. Ferreira, M. Gabbouj, "A Generic Shape/Texture Descriptor Over Multiscale Edge Field: 2-D Walking Ant Histogram", IEEE Trans. Image Processing, vol.17, issue.3, pp.377, Mar. 2008.

[27] C. Akinlar, and C. Topal, ”EDLines: Realtime Line Segment Detection by Edge Drawing (ED),” IEEE Int’l Conf. on Image Processing (ICIP), Sep. 2011.

[28] J.S. Stahl, S. Wang, "Edge Grouping Combining Boundary and Region Information", IEEE Trans. Image Processing, vol.16, issue.10, pp.2590, Oct 2007.

[29] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, "Groups of Adjacent Contour Segments for Object Detection", IEEE Trans. PAMI, vol.30, no.1, Jan 2008.

[30] F. van der Heijden, "Edge and line feature extraction based on covariance models," IEEE Trans. Pattern Analysis and Machine Intelligence, vol.17, issue.1, pp.16, Jan 1995.

[31] Y. Gao, M.K.H. Leung, "Face recognition using line edge map", IEEE Trans. PAMI, Vol.24, issue.6, pp.764, Jun 2002.

[32] X. Dai and S. Khorram, "A Feature-Based Image Registration Algorithm Using Improved Chain-Code Representation Combined with Invariant Moments", IEEE Trans. on Geoscience and Remote Sensing, vol. 37, no. 5, Sep 1999.

[33] M. Xia, B. Liu, "Image registration by "Super-curves", IEEE Trans. Image Processing, vol.13, issue.5, pp.720, May 2004.

[34] C. Topal, C. Akınlar, and Y. Genç, "Edge Drawing: A Heuristic Approach to Robust Real-Time Edge Detection," Proc. of Int. Conference on Pattern Recognition (ICPR), pp. 2424-2427, 2010.

[35] Edge Drawing WebSite is available online at http://ceng.anadolu.edu.tr/cv/ (2011)