International Journal of Advanced Smart Convergence Vol.13 No.2 88-93 (2024)
http://dx.doi.org/10.7236/IJASC.2024.13.2.88

IJASC 24-2-11

Edge Detection using Enhanced Cost Minimization Methods

Seong-Hoon Lee*

Professor, Division of Computer Engineering, Baekseok University, Korea
shlee@bu.ac.kr

Abstract

The main problem with existing edge detection techniques is that they have many limitations in detecting edges for complex and diverse images that exist in the real world. This is because only edges of a defined shape are discovered based on an accurate definition of the edge. One of the methods to solve this problem is the cost minimization method. In the cost minimization method, cost elements and cost functions are defined and used. The cost function calculates the cost for the candidate edge model generated according to the candidate edge generation strategy, and if the cost is found to be satisfactory, the candidate edge model becomes the edge for the image. In this study, we proposed an enhanced candidate edge generation strategy to discover edges for more diverse types of images in order to improve the shortcoming of the cost minimization method, which is that it only discovers edges of a defined type. As a result, improved edge detection results were confirmed.

Keywords: *Image processing, Pixels, cost function, candidate edges, Artificial intelligence.*

1. Introduction

Edges are short linear segments called "Edgls (edge elements), which are boundaries between two regions with different and constant gray levels. Edge detection involves various mathematical methods that aim to identify edges and curves in digital images where image brightness changes rapidly or has discontinuities [1, 2]. The problem of finding discontinuities in a one-dimensional signal is called step detection, and the problem of finding signal discontinuities over time is called change detection. Edge detection can be considered a fundamental tool in image processing, machine vision, and computer vision, especially in the areas of feature detection and feature extraction [4-6]. Because it was difficult to accurately define various edges in existing edge detection methods, research has focused on discovering specific edges. This causes the problem of discovering only a limited range of edges.

In the cost minimization method considering the cost function, the edge definition is defined only with general features in order to discover various edges. Additionally, a cost function is defined and used to evaluate the edge model, and this cost function is defined as the linear sum of weighted cost factors. The edge model

Manuscript Received: April. 18, 2024 / Revised: April. 26, 2024 / Accepted: May. 3, 2024
Corresponding Author: shlee@bu.ac.kr
Tel: +82-42-550-2433
Author's affiliation: Professor, Division of Computer Engineering, Baekseok University, Korea

that minimizes the cost function defined in this way is regarded as the extracted edge for a given picture [14]. In other words, a random candidate edge model is created for the image for which an edge is to be found and the cost is calculated. Continuing to generate a candidate edge model in the direction of reducing the cost according to the candidate edge generation strategy, this is repeated until the cost of the candidate edge model reaches the desired level. However, the existing candidate edge generation strategy 4 moves the two candidate edge models S' corresponding to the current edge model S only in a single direction (90 degrees) as a result of moving 90 degrees in both directions with respect to the central pixel. It has the disadvantage of only discovering the model at the location and not being able to discover edges of other shapes. Therefore, a strategy is needed to discover various types of edges by diversifying these movement directions. In other words, in order to discover overall edges for various images with a small number of iterations, a more diverse candidate edge generation strategy is needed.

Additionally, among the existing cost factors, ‘0’ and ‘1’ were uniformly applied to the cost for the number of pixels constituting a corner and the cost for fragmentation without considering neighboring pixels within the window. Therefore, in order to apply a more accurate cost, a cost that takes neighboring pixels into account must be applied. In this study, we redefined existing cost factors to discover various edges more efficiently, added and proposed a new candidate edge generation strategy, and compared the performance with existing methods by selecting various types of images.

2. Edge Detection Method

Existing methods for discovering boundaries between regions in an image include parallel processing techniques (linear, non-linear, optimal techniques) and sequential processing techniques using previous knowledge. The parallel processing technique is a method of determining whether a set of multiple points is an edge or not by considering the gray levels of not only the set but also several neighboring sets. In principle, the edge detection operator is applied to all pixels in the image at the same time. Parallel processing techniques can be broadly divided into linear methods, nonlinear methods, and optimal techniques. Linear methods include high-emphasis spatial frequency filtering, directional differentiation, and gradient methods, and nonlinear methods include Rosenfeld method and Herskovitz and Binford method. Additionally, optimal approach methods include Griffith's method and Chew's method. Sequential processing methods refer to when edge detectors are applied to the image sequentially, with the results at one point depending on the results at the previously examined point. These methods include heuristic search, dynamic programming, and guided edge detection methods.

The edge detection techniques described above make it difficult to accurately define the edge shape to be discovered. Therefore, by defining the shape of the edge to be discovered for a specific input image and discovering only the defined edge, it is not possible to effectively discover various types of edges that exist in the real world. Therefore, research is being conducted on various edge detection techniques, and one of these studies is a cost minimization method.

3. Cost Minimization Method

In general, the problem with previous edge detection techniques is that they use a strict definition of the edge, so there are limitations to the images to which they can be applied. Therefore, rather than defining edges strictly, it is appropriate to define only the general characteristics of edges to discover various types of edges. Additionally, since many algorithms do not consider the local edge structure of neighbors when determining a corner pixel, it was not possible to extract and use information from the neighboring edge structure during the decision process of the corner pixel. Therefore, a method to solve these shortcomings is a cost minimization method.

3.1 Improving Dissimilarity

The first process to discover corners is dissimilarity enhancement, where points in the image that can be considered corner pixels are enhanced, and this improved image D can be written as a formula (D = { d(i, j); 1 ≤ i, j ≤ N }). It is a set of pixels where each pixel value is proportional to the degree of local dissimilarity at that pixel location. At this time, the value of each pixel is between ‘0’ and ‘1’, and pixels with a large value close to ‘1’ are good candidates to be considered corner pixels. Two important requirements are needed for this image enhancement: a title that defines regions and a function that can measure the dissimilarity of these regions. The regional dissimilarity measurement function can be defined and used by the user as a scale that can measure the degree of regional dissimilarity. In this paper, the dissimilarity measurement function between two regions R1 and R2 is denoted as F(R1, R2). The function is defined as the difference in average gray levels present in two regions.

From this improved image, edges are discovered by finding an edge model with low cost through a minimization operation on the cost function expressed as a weighted sum of five elements. From this improved image, the cost function is calculated for all pixels. Before describing the cost function, we first look at the definitions required.

Table 1. Definition for cost function

Image G	G = {g(i, j); (1≤i, j≤n)}, each pixel g(i, j) has a gray level ranging from 0 to 255
Edge S	S = { s(i, j); (1 ≤ i, j ≤ n) }, Each pixel s(i, j) has a binary value of 0 or 1. If s(i, j) is 1, it is called an edge pixel, and if s(i, j) is 0, it is called a non-edge pixel.
Corner E	A component that constitutes a set of corner pixels included in the corner model (configuration)
Thin edge	An edge composed only of edge pixels that are not included in the cycle
Thick edge	An edge composed only of edge pixels included in the cycle
Cycle	A path with the same beginning and end

3.2 Cost Function

The cost function assigns a cost to each edge model so that the model with the lowest cost matches the best model according to the edge concept. Mutual contradictory actions occur between cost elements that reflect the characteristics of the edges. For example, if all the edges in the image are long and continuous, poor locality and incorrect edges may appear. On the other hand, if all the edges in the image are short and discontinuous, there may be good locality and clear edges. However, the overall surface is not reflected. There are drawbacks. Therefore, to resolve these conflicting requirements, we combine each desirable edge characteristic with a degree of importance. This can be satisfied by a cost function, which is obtained as the sum of the costs for each point forming the edge model. In other words, the cost function is defined as a linearly weighted sum of cost elements. The definitions of the cost function and cost elements were identical to those in the reference literature [14].

As described previously, the edge model that minimizes the cost function is considered to be the extracted edge for a given picture. In other words, a random candidate edge model is created for the image for which the edge is to be found and the cost is calculated. Continuing to generate a candidate edge model in a way that reduces the cost according to the candidate edge generation strategy, this is repeated until the cost of the candidate edge model reaches the desired level. At this time, the candidate edge model that is satisfied is called the edge for the corresponding image.

3.3 Candidate Edge Model Generation Strategy

The method of creating a candidate edge model is created by combining the following six strategies. A candidate edge model for the image is created using the candidate edge model generation strategy until a satisfactory edge is generated. And in each strategy, the position l is selected using raster scan method. When X_k = S is the current edge model and Y_k = S′ is the candidate edge model, each strategy is as follows.

① Single pixel change: S' = M_1(S, l), a candidate edge model is created by complementing the pixel value at position l within the current edge model S.

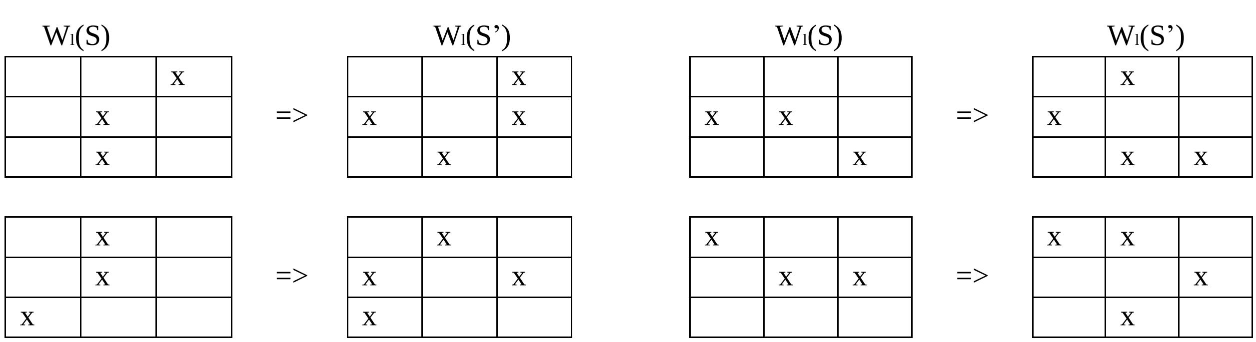
② Pixel change at two locations: S' = M_2(S, l), the positions of neighboring pixels are randomly selected and the pixel values of positions l and l' within the current edge model S are conservative to create a candidate edge model.

③ Single pixel movement: S' = M_3(S, l), if the edge structure in the window of the current model belongs to one of the edge structures in Figure 1 below, the structure of the candidate edge model is as shown on the right. If it is not included in any edge structure in the picture, the candidate edge model becomes the current edge model.

④ Move multiple pixels: S' = M_4(S, l), this strategy is the same as strategy 3 except that the positions of more pixels are moved.

⑤ Window area change: S' = M_5(S, l), in the current edge model S, a candidate edge model S' is created by randomly changing pixel values for each position included in the window.

⑥ Move the central pixel 180 degrees in both directions: S' = M_6(S, l), in strategy 6, the central pixel is moved 180 degrees in both directions to create a candidate edge model W_l(S') for the current edge model window (W_l(S)), as shown in Figure 1. If the edge structure in the window of the current model belongs to one of the edge structures in Figure 1 below, the structure of the candidate edge model is as shown on the right. If it is not included in any edge structure, the candidate edge model becomes the current model. In this strategy, the candidate edge model S' is the same as the current edge model S except for the pixels corresponding to the window (3*3) in the current model.



x		
	x	
	x	

=>

x		
x		x
	x	

		x
x	x	

=>

	x	x
x		
	x	

	x	
	x	
		x

=>

	x	
x		x
		x

	x	x
X		

=>

	x	
		x
x	x	

Figure. 1 8 windows ($W_l(S, l)$ and the corresponding candidate edge model structure ($W_l(S', l)$)

4. Performance Evaluation

The data format used in the implementation was a two-dimensional array with a size of 15*15, and data between '0' and '255' corresponding to the gray level value was used, which is assumed to be the result of scanning. The image composed of gray level values is shown in Table 2.

Table 2. Initial image with gray level

244	245	247	250	251	255	236	240	251	239	241	246	251	244	231
239	233	243	253	249	250	247	244	251	235	255	253	247	240	241
250	240	240	236	242	243	248	245	246	237	248	249	240	243	245
240	250	250	241	242	243	234	235	246	231	238	239	240	241	251
250	237	240	231	11	13	9	13	16	17	11	239	240	245	250
240	220	240	241	7	9	14	11	11	7	8	229	245	246	250
231	233	250	251	5	6	4	5	6	7	9	239	240	244	249
233	244	250	242	6	8	9	10	11	10	12	244	240	245	241
250	232	233	231	10	13	14	12	7	11	12	238	250	243	251
240	240	245	249	11	12	15	12	6	10	12	234	239	241	245
241	243	244	253	13	10	15	18	13	17	12	234	230	241	244
233	249	243	239	234	231	236	233	231	236	231	244	237	241	245
248	241	249	236	233	233	240	233	237	239	234	233	250	243	252
244	238	238	241	242	249	241	246	241	246	249	245	244	248	248
249	249	250	243	248	244	250	241	248	244	243	244	246	248	251

Additionally, the weights for cost factors used in this experiment are listed in Table 3.

Table 3. Weight for cost factors

Cost factor	Cc	Cd	Ce	Cf	Ct
weight	0.5	2.0	1.0	3.0	6.51

This result shows the result after performing the process 10 times. In the existing edge detection method, the four pixels ((2.2), (2.6), (9.3), (9.5)) were not found even though they were edges. On the other hand, in the proposed method, three pixel parts ((2.4), (8.8), (9.7)) were not found even though they were edges.

5. Conclusions

Existing edge detection techniques define the shape of the edge to be discovered for a specific input image and discover only the defined edge, so they cannot effectively discover various types of edges that exist in the real world. One method to solve this problem is the cost minimization method. In this method, cost factors are defined and the cost function is defined as the sum of their linear weights. The edge detection process generates a random candidate edge model for the image for which the edge is to be found. This process is repeated until the cost of the candidate edge model reaches the desired level, and a specific candidate generation strategy is applied. In this study, we attempted to discover various types of edges that exist in the real world by adding an edge candidate generation strategy. In the experiment, edge detection results between the existing method and the proposed method were derived for images expressed with specific gray level values. The results showed that the proposed method increases the probability of accurate edge detection compared to the existing method for a specific image.

Acknowledgement

This paper was supported by 2024 Baekseok University Research Fund

References

[1] L. Tony, “Edge detection and ridge detection with automatic scale selection”, International Journal of Computer Vision, Vol. 30, no 2, pp. 117-154, 1998.
[2] L. Tony, “Edge detection”, Encyclopedia of Mathematics, EMS Press. 2001.
[3] M. H. Asghari, B. Jalali, “Physics-inspired image edge detection”, 2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP), 2014. DOI: 10.1109/GlobalSIP.2014.7032125.
[4] T. P. Agustín, K. Krissian, A. F. Miguel, S. C. Daniel, "Accurate subpixel edge location based on partial area effect", Image and Vision Computing, Vol.31, Issue 1, pp.72-90, 2013.
[5] Sugata Ghosal, Rajiv Mehrotra, “Orthogonal moment operators for subpixel edge detection”, Pattern Recognition, Vol,26, Issue 2, pp.295-306, 1993.
[6] R. D. Jules and T. Takamura, "Alternative Approach for Satellite Cloud Classification: Edge Gradient Application", Advances in Meteorology, 2013. https://doi.org/10.1155/2013/584816
[7] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05), 2005.
[8] W. Zhang and F. Bergholm, “Multi-Scale Blur Estimation and Edge Type Classification for Scene Analysis”, International Journal of Computer Vision, Vol.24, pp.219–250, 1997.
[9] M. Petrou and J. Kittler. Optimal Edge Detector for Ramp Edges. IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(5) pp. 483-491, 1991.
[10] R. Mehrotra and S. Zhan. A Computational Approach to Zero-Crossing-Based Two Dimensional Edge Detection. CVGIP: Graphical Models and Image Processing, Vol.58, pp.1-17, 1996.
[11] D. J. Williams and M. Shah. “Edge Characterization Using Normalized Edge Detector”, Computer Vision, Graphics and Image Processing, Vol.55, pp.311-318, 1993.
[12] S. Tabbone and D. Ziou. “Elimination of False Edges by Separation and Propagation of Thresholds”, In 13th Conference on Signal Processing and Images, 1991.
[13] H. L. Tan and S. B. Gelfand, “A Cost Minimization Approach to Edge Detection using Simulated Annealing”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No.1, 1991.
[14] H. L Tan and E. J. Delp, “Edge Detection by Cost Minimization”, Purdue University, 1988.