A Method for Detecting Boundary Edges Based on a Local Image-Feature Integration Method

Hisashi Shimodaira

Abstract

In an image, boundaries between regions represent the shapes of objects in the scene. Finding the boundaries is a problem of fundamental importance in the area of computer vision such as object recognition whose accuracy depends on the quality of the estimates of the boundaries. This paper proposes a method for detecting boundary edges based on a local image-feature integration method using a modified Canny edge detector. In the proposed method, first by edge saliency indices, salient edges are selected from the output of the modified Canny edge detector. Next, by boundary likelihood indices, edges with high boundary likelihood are selected from the salient edges. The major features of our method are as follows: the algorithm is simple and easy to implement; the computational complexity is not so high; the parameter values once tuned for several images can be used in common for other images; the good thresholds for selecting boundary edges are automatically computed. We have performed experiments on various natural images. The results have shown that our method works effectively and can produce relatively good estimates of boundary edges and its performance is robust.

Keywords: Region boundary, Boundary detection, Contour detection, Edge detection, Edge saliency, Canny

1. Introduction

1.1. Background and purpose

In an image, the regions of which characteristics are almost uniform or homogeneous correspond to the projections of objects or parts of objects and the background in the real scene. Finding the boundaries between the regions in the image is a problem of fundamental importance in computer vision such as object recognition, because it relies on the region and boundary data representing the shapes and contours of objects in the image and consequently its accuracy depends on the quality of the estimates of such data.

In this paper, we distinguish a boundary and an edge in an image, though both represent a curve representing an abrupt change in the image features. That is, a boundary is defined as a border between different regions in the image plane, whereas an edge is defined as an abrupt change in some low-level image features such as brightness or color. Since a boundary can be represented as an edge between regions, we call it a boundary edge in this case. The standard edge detector such as Canny's edge detector [1] considers a small patch centered on an image pixel, computes local maximums of oriented brightness gradients, and outputs their magnitudes and

locations as edges. By its locality, besides true boundary edges, it produces many edges caused by noise and subtle changes in brightness even in almost uniform regions and it inevitably produces many high-contrast edges in textured regions. Therefore, the process which detects edges with high boundary likelihood is needed for their subsequent use. On the other hand, as a result of the standard region segmentation techniques such as the split-and-merge and region-growing algorithms, boundaries can be produced. However, the produced boundaries are ragged and often erroneous due to their limited discrimination power. Therefore, the data on edges with high boundary likelihood is needed in order to detect more accurate boundaries.

In this paper, we propose a method for detecting edges with high boundary likelihood of a natural image from the output of a modified Canny edge detector based on a boundary-likelihood index. Since Canny's edge detector whose algorithm is easy to implement and performance is superior has been most popular and widely used [2], developing such a method for enhancing the quality of its output by eliminating non-boundary edges is important and much desired.

1.2. Previous work

Various approaches to producing high-quality edges or detecting boundaries or contours have been reported in the extensive literatures; these approaches can be roughly classified into several groups as follows. We briefly review major ones.

The first group is based on various techniques to produce high-quality edges. It includes the techniques for the use of the multi-scale [3 - 5] and anisotropic diffusion [6 - 8]; the relaxation labeling [9 - 11]; a surround suppression technique for suppressing texture edges [12, 13]; and a non-boundary local energy model and local thresholding technique [14].

The second group is based on various techniques to detect boundary edges using the combination of the local features of edge pixels and edge elements. As the local features, length, gradient magnitude, contrast, continuity, proximity, curvature, co-circularity, etc are used. In these techniques, as the final procedure for selecting boundary edges or contours, thresholding, optimization, relaxation procedure, etc are used. It includes the hysteresis thresholding method based on gradient magnitudes and continuity [1]; the generalization of the hysteresis thresholding based on confidence measures [15]; the edge evaluation in a length and average gradient magnitude space based on statistics [16, 17]; the edge evaluation based on distances between textures [18]; and the edge element grouping techniques based on proximity [19], curvature and globally salient structures [20], co-circularity and combinatorial optimization [21], smoothness and relaxation labeling [22], co-circularity and surround inhibition [23], and continuity and adaptive pseudo dilation [24].

The third group is based on various techniques to detect boundaries using the learning processes of prior knowledge on boundaries. It includes the techniques for the learning processes of human-marked boundaries based on statistical inference [25, 26], conditional random field [27, 28], boosted edge learning [29], and artificial neural networks [30, 31]. It also includes the techniques for the learning processes of object specific low-, mid-, high-level information [32].

The fourth group is based on various approaches using region segmentation. It includes the techniques for integration of region-growing and edge detection [33], fusion of color and edge information [34], graph partitioning [35], and pairwise-similarity and variational cost function [36].

1.3. Our approach

Our approach falls within the second group and it is classified into the method considering only the features of each edge regardless of the reciprocal relation of edges. The recent related literatures show that the boundary

detectors based on local features are not complete and universal. Generally, the output containing the more true boundaries is apt to contain the more non-boundaries and the output containing the fewer non-boundaries is apt to contain the fewer true boundaries. The major problems with these approaches are as follows.

(1) The first is what features to use to detect edges with high boundary likelihood. Since boundaries are borders between regions with different image characteristics, use of only local edge features which are derived from a small neighborhood of each pixel such as gradient magnitudes, co-circularity of edge elements, etc cannot provide reliable indicators of boundary likelihood. Therefore, in order to obtain better results, the approaches considering image features in a much larger neighborhood such as the reduction effects of saliency of edges surrounded by textures [12, 13], distances between textures [18], and texture gradients [26] are required.

(2) The second is how to tune parameters which are inevitably needed to adjust the processing condition. In most of such systems, they must be tuned to obtain desirable results for each image. In the third group, such parameter tuning operations are performed by the learning processes, which is troublesome and computationally expensive, based on human-marked boundaries or object specific information.

(3) The third is how to select the final output. Most of the thresholding methods require the user to choose threshold values, especially two values in the hysteresis thresholding [1, 15], manually for each image and thus they have practical difficulties of how to choose these values. The processing such as the relaxation labeling, global optimization, etc is computationally expensive.

(4) The fourth is that the algorithms of most of the modern approaches are very complex, difficult to implement, and computationally demanding.

For the second, third, and fourth problems, we need to devise methods which are as simple to implement, less troublesome to tune, and computationally cheap as possible.

In this paper, we propose a method for detecting boundary edges by using the modified Canny edge detector and a local image-feature integration method (LIFIM). Our goal is to devise methods which can resolve the foregoing problems and to present a boundary edge detector which can produce output containing as many boundary edges and a small number of non-boundary edges as possible. Our method consists of three stages. First, by the modified Canny's edge detector, edge candidate pixels are selected. Next, by the processes of following edges, computing the edge saliency indices (ESIs) of the detected raw edges, and automatically thresholding them, salient edges are selected. Finally, by the processes of computing the boundary likelihood indices (BLIs) of the salient edges and automatically thresholding them using a depth-curvature based valley seeking method (DCVSM), boundary edges are selected. The edge saliency index (ESI) is constructed by a linear combination of the three terms based on a mean-fitting normalization (MFN) method: the length, mean gradient magnitude, and smoothness measure of a raw edge. The last term is computed by an entropy-based smoothness measure (EBSM). The boundary likelihood index (BLI) is represented by the product of ESI and the measure of diversity of a salient edge, which is computed by the gray level distributions of its relatively large neighboring area. Our method has several parameters to be tuned. It is designed so that the parameter values once tuned for several images can be used in common for other images by adjusting the differences between major characteristics of each image automatically.

The contributions of this paper are LIFIM integrating its components of ESI based on the MFN method, EBSM, and BLI; and DCVSM. The major features of our system are as follows: the algorithm is simple and easy to implement; the computational complexity is not so high; the parameter values once tuned can be used in common for other images; the good thresholds for selecting boundary edges are automatically computed; and the performance is robust and relatively good estimates of boundary edges in natural images can be obtained.

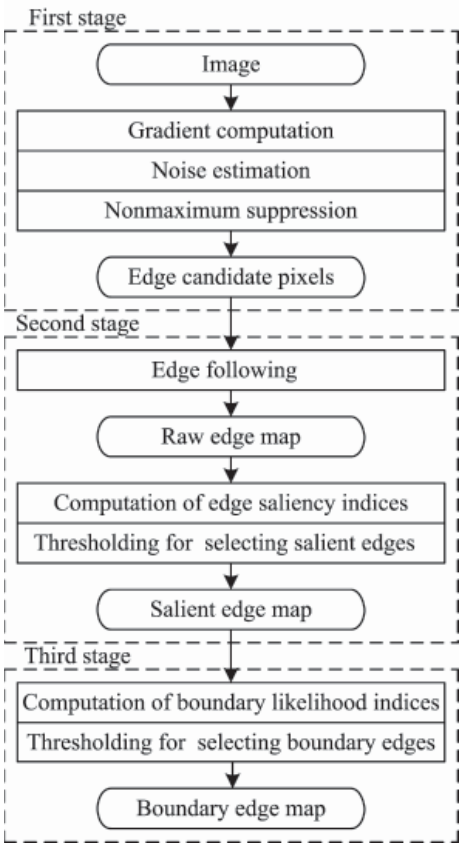


Fig. 1. The flow of the proposed boundary edge detection method.

We implemented our method and performed experiments to investigate and evaluate its performance on the three standard image data sets: the RuG [37], Berkeley [38], and South Florida [39] data sets. We will show as many resultant boundary edge maps as possible with the quantitative evaluations using the human-marked boundaries.

1.4. Paper organization

The rest of the paper is organized as follows. In Section 2, we present our boundary edge detection method. In Section 3, the results of the experiments are presented. In Section 4, we summarize the experimental results, compare our results with those of the previous papers, and discuss the future work. In Section 5, the conclusions are described.

2. Our boundary edge detection method

2.1. Overview of our method

The overview of our method is shown in Fig. 1. Our method consists of the following three stages. The first stage consists of the following three processes. First, an image is smoothed by using a gaussian filter and then

the gradient at each pixel is computed. Next, at the noise estimation process, a threshold above which a gradient magnitude is considered "significant" is determined. Finally, the nonmaximum suppression is performed and edge candidate pixels are selected. The second stage consists of the following three processes. First, edges are followed by searching eight-connected pixels to each other in the edge candidate pixels and the raw edge map is produced. Next, ESI of each raw edge is computed. Finally, the histogram of ESI values is formed and the salient edge map is produced by a threshold which is automatically determined by analyzing the concavity of the histogram. If the salient edge map is satisfactory, we can stop further processing and accept it as the boundary edge map. The third stage consists of the following two processes. First, BLI of each salient edge is computed. Next, the histogram of BLI values is formed and the final boundary edge map is produced by a threshold which is automatically computed by DCVSM.

Our LIFIM is based on ESI and BLI. The edge saliency index (ESI) is represented by the edge information, i.e. a linear combination of the three terms: the length, mean gradient magnitude, and EBSM of a raw edge. The boundary likelihood index (BLI) is represented by the information on a salient edge and its neighboring area, i.e. the product of ESI and the measure of diversity which is computed by the gray level distributions of the two subsets of the observation window. Our strategy to detect boundary edges is first to select edges containing as more boundary edges and fewer non-boundary edges as possible by ESI and next to select edges with high boundary likelihood as accurately as possible by BLI. In the latter, edges of which the image characteristics on both sides are more different are selected from the output of the former. This strategy allows us to reduce the complexity of computation of BLI and to facilitate computation of thresholds in DCVSM.

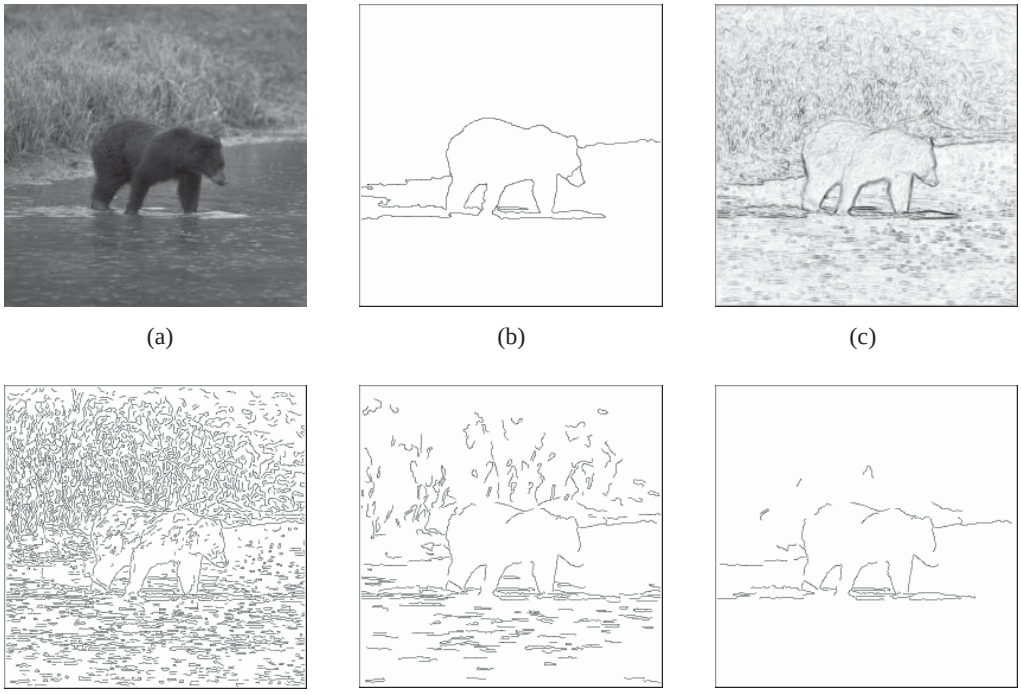
Our method is different from Canny's edge detector in that pixels with insignificant gradient magnitudes identified by the noise estimation process are eliminated from the edge candidate pixels and any thresholds are not used in the edge following process. In this meaning, Canny's edge detector is modified and used in our method.

We show the major output of these processes for the bear image of the RuG data set [37]. The original image and human-marked boundary edge map are shown in Fig. 2 (a) and (b), respectively.

2.2. Selection of edge candidate pixels

In order to diminish the effects of noise, the smoothing operation is applied to the gray levels of the image of interest before computing the gradients. Concretely, the input image is smoothed by convolving it with a two-dimensional linear gaussian filter of which standard deviation and size are σ_g and W_g, respectively. The scale of smoothing (σ_g) must be selected suitably. Smoothing with large scale filters can hide or blur fine structures and subtle features; consequently this results in increase in missing edges. On the other hand, smoothing with small scale filters is sensitive to edge signals but also prone to respond to noise; consequently this results in increase in spurious edges due to noise, etc. Since our method uses the noise estimation process, we adopted a strategy of using a relatively small value of σ_g in order not to eliminate significant changes of image features. We empirically found that σ_g = 1.0 and W_g = 5 are best for the images which we tested. The results shown in this paper are those which were obtained by using these values.

Using the gray levels of the smoothed image, the gradient vector at each pixel is computed following Fleck's method [40] which can reduce the bias in computation of gradients. Her method uses a 3×3 pixel area centered on a pixel of interest and a mask [-1, 0, 1] to compute first differences in the four directions: horizontal, vertical, and two diagonal directions. The X and Y components of the gradient vector on the x and y axes, respectively are computed considering the projection of the diagonal differences on to the x and y axes. The gradient magnitude is computed as the magnitude of the gradient vector (X, Y) and the gradient direction is



(a) (b) (c)

(d) (e) (f)

Fig. 2. The input and major output of each process for the bear image: (a) original image, (b) human-marked boundary edge map, (c)gradient magnitude, (d) raw edge map, (e) salient edge map, and (f) final boundary edge map (F = 0.788).

computed by using the arctangent of the ratio of X and Y. The image of the gradient magnitudes of the bear image is shown in Fig. 2 (c); it is expressed by inversely transforming their values into the gray levels.

The gradient magnitudes computed in such a way contain small values due to noise. At the noise estimation process, the histogram of the gradient magnitudes is formed and a threshold above which a gradient magnitude is considered “significant” is determined. The threshold τ is computed by the modified Voorhees method [41] by

$$\tau = \sigma_n \sqrt{-2 \ln P_n} , \tag{1}$$

where P_n , which is given as input, is the probability that pixels with gradient magnitudes due to noise are contained in the edge candidate pixels, if the threshold is used; and σ_n is the mode of the Rayleigh distribution that the histogram of gradient magnitudes due to noise takes on. We assumed that σ_n is equal to the gradient magnitude at which the histogram exhibits the first maximum. The pixels with a gradient magnitude equal to or above τ are treated as the objects for the search of local maximums in the nonmaximum suppression process. In other word, the pixels with a gradient magnitude below τ are eliminated from the candidates of edge pixels. As a result, the edge following process becomes stable. Note that the threshold τ is determined considering the characteristics of each image. The threshold τ corresponds to the lower threshold in the hysteresis thresholding [1]. By this processing, a suitable threshold can be determined according to the image characteristics and we are relieved from choosing the threshold manually.

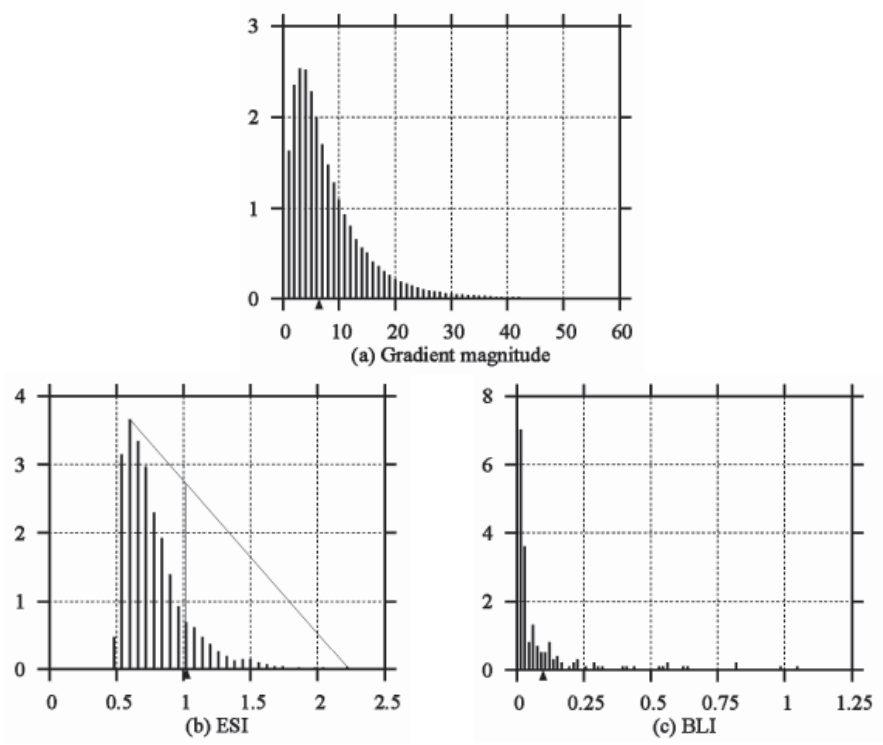


Fig. 3. Histograms for the bear image: (a) gradient magnitudes, (b) ESI values of the raw edges with BS_1 = 0.06, and (c) BLI values of the salient edges with BS_2 = 0.015. The ordinates: for (a), the number of pixels in 10,000; for (b) and (c), the number of edges in 100 and 10, respectively.

The histogram of the gradient magnitudes with a bin size 1.0 for the bear image is shown in Fig. 3 (a). We adopted a strategy of using relatively large values of P_n, i.e. relatively small values of τ in order not to eliminate significant structures of image features. We empirically found that P_n = 0.1 is best for the images which we tested. The results shown in this paper are those which were obtained by using this value. For the bear image, since σ_n = 3, the threshold 6.4 was obtained.

In the nonmaximum suppression process, the gradient magnitude at a pixel of interest is compared with those at the two virtual points located at the opposite sides in the gradient direction. If the gradient magnitude at the pixel of interest is lager than those at the two virtual points, then the pixel is selected as an edge candidate pixel and stored into the edge candidate pixel map. In the conventional scheme, the two virtual points are defined as the intersections of the gradient direction with the sides of the 3×3 pixel area and their gradient magnitudes are interpolated using the two adjacent pixels on the side. We present a new interpolation scheme where the distance between the pixel of interest and the virtual points is a given constant value R. The locations of the virtual points can be computed using the unit vector in the gradient direction and their gradient magnitude can be computed by the bilinear interpolation using those of its four nearest neighboring pixels. In this scheme, any value of R can be used, if necessary. Compared to the conventional scheme, our scheme is easy to implement, because it is isotropic for gradient directions. In such a sense, we call this scheme the isotropic interpolation scheme. We empirically found that R = 1 is best for the images which we tested. The results shown

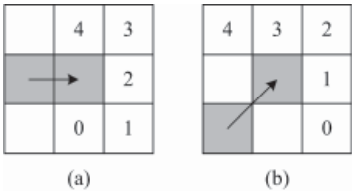


Fig. 4. Search for a next edge pixel. Examples of the directions of edge segments and the corresponding five neighboring pixels for searching a next edge pixel: (a) horizontal direction and (b) diagonal direction.

in this paper are those which were obtained by using this value.

2.3. Edge following method

In Fig. 1 in [28], Ren et al. showed the characteristics of the empirical distribution of the tangent angle changes along human marked-boundaries in natural images: the distribution is sharply peaked around zero, i.e. in most cases boundaries are smooth; and the distribution has a heavy tail, i.e. occasionally boundaries do make sudden turns, though most of tangent changes are between -90 and 90 degrees. Our edge following method for detecting edges is based on such empirical data.

In the beginning of the edge following processing, the image is divided into blocks for selecting the starting point. As the starting point for following an edge, the pixel with the maximum gradient magnitude at the time in the block is selected. If the processing of an edge is finished, the starting point is selected from the remaining pixels in the block. An edge starting at a block can be followed into another block, if necessary. The processing is performed for each block in order. This strategy has the following advantages: since significant edges with larger gradient values have priority in the edge following processing, the processing becomes stable; it allows us to produce a unique set of edges and to reproduce the same result.

An edge is followed and detected from the edge candidate pixels as follows. A pixel constituting the edge is followed by searching five neighboring pixels that exist at the angles between -90 and 90 degrees from the current edge direction formed by the pair of the last and the second last pixels of the edge as illustrated in Fig. 4. If a pixel is determined to belong to the edge, it is given the flag to avoid the overlapping of edges. Each of the five neighboring pixels is given one of the five pattern labels of 0, 1, 2, 3, and 4 corresponding to an angle from the foregoing edge direction as shown in Fig. 4. This label is used to express the changes of positions of consecutive pixels of the edge and to compute its smoothness measure. The edge data is stored into the edge buffer using the chain cord of each pixel.

In the case where multiple edges intersect at one pixel, since the decision on which one is a true boundary edge should be independent on each edge, we adopt a strategy that the edge processing is stopped and each edge is followed separately. Thus, the continuation condition of the edge following processing is that only one candidate pixel is found except for the following case. If two candidate pixels found are four-connected to each other, the pixel that is four-connected to the pixel of interest is chosen as the next edge pixel. A threshold L_l for the lower limit of edge length is used to eliminate edges whose length is too short to compute the smoothness measure. In the experiments, $L_l = 5$ was used.

Since the pixels with small gradient magnitudes due to noise are eliminated from the search object, the processing is stable. The raw edge map detected for the bear image is shown in Fig. 2 (d). The number of the raw edges is 2,338.

2.4. Edge saliency measure

We define the edge saliency as a measure to detect high-quality edges containing boundaries. Considering the results of the previous papers, we decided the image features to represent the edge saliency based on the facts obtained from observing natural images by visual perception. Generally, in most of natural images, contours and boundaries representing major objects in the image are long and their gradient magnitudes are relatively large. Also, edges in textured regions are wiggly and not smooth. For example, we can see these in the bear image shown in Fig. 2 (c) and (d). In addition, for human marked-boundaries, Ren et al. [28] showed that in most cases, boundaries are smooth and occasionally do make sudden turns, though most of tangent changes are between -90 and 90 degrees. Considering these, we selected the length, mean gradient magnitude, and smoothness of an edge as the edge saliency measure.

For each raw edge i, the length L_i and the mean gradient magnitude G_i are computed by

$$L_i = \sum_{k=1}^{m-1} d_k \ , \tag{2}$$

$$G_i = \frac{1}{m} \sum_{j=1}^{m} g_j \ , \tag{3}$$

where d_k is the length of an edge segment k formed by two adjacent pixels, which is 1.0 for the horizontal and vertical edge segments and $\sqrt{2}$ for the diagonal one; g_j is the gradient magnitude at a pixel j; and m is the total number of pixels of the edge.

For the measure to represent the smoothness or regularity of shapes or edges, the techniques using the following values have been reported: the curvature and curvature variation of an edge by Sha'ashua [20], the chord length of an edge curve by Vasselle [42], and the local angel distribution of contour segments by Chen [43].

We define the smoothness of an edge as a measure to represent the frequency of the directional changes along the edge curve. Since the position of a pixel is discrete and the constituents of an edge are eight-connected pixels, it should be defined considering the following things. The smoothest extreme is the case where the edge curve is a horizontal or vertical straight line. The non-smoothest extreme is the case where the edge curve is completely random as in the Brownian motion curve. All positional changes in edge pixels correspond to the smoothness levels between these extremes. The smoothness level of an arc or oblique line should be high, because it is composed of horizontal or vertical short segments which are stepwise connected to each other.

On the basis of the foregoing consideration, we call a pair of two consecutive edge pixels an edge segment and define the smoothness of an edge by using the entropy of the probability distribution computed from the combinations of patterns of consecutive edge segments as follows. In the edge following process, each edge pixel, which constitutes the current edge segment, is assigned one of the five pattern labels which is given corresponding to an angle from the direction of the last edge segment as shown in Fig. 4. The directional changes between consecutive edge segments can be represented by the combinations of these five pattern labels. The total number of the pattern combinations M is 25, 125, and 625 corresponding to the number of the combination 2, 3, and 4, respectively. If we make such pattern combinations along an edge in succession, a sequence to represent the smoothness of the edge is produced. By examining the frequency of each pattern combination in this sequence and computing the probability, we can compute the entropy that represents the degree of the randomness of the sequence and evaluate the smoothness of the edge using it as follows.

The entropy of the probability distribution of the pattern combinations for an edge i is defined as

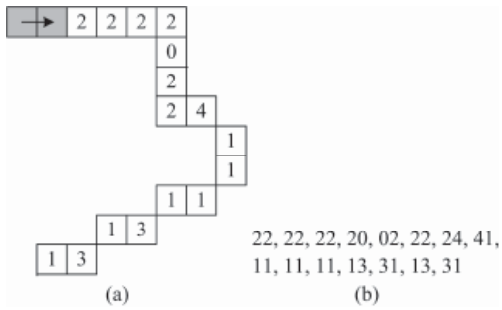


Fig. 5. An example of edges: (a) the positions of edge pixels and their pattern labels, and (b) the sequence of two-pattern combinations.

$$h_i = -\sum_{j=0}^{n-1} p_j \log p_j \,, \tag{4}$$

where p_j is the probability of a pattern combination j which is computed by dividing its frequency by the total number of the counted pattern combinations and n is the total number of kinds of the pattern combinations with a nonzero frequency that occurred in the sequence. The maximum entropy that represents the complete randomness is defined as

$$h_{\max} = \log M. \tag{5}$$

The relative value H_i of the entropy to the maximum value is given by

$$H_i = \frac{h_i}{h_{\max}}. \tag{6}$$

The value H_i can represent the degree of the randomness of the sequence and thus that of the edge curve. In the case where it is completely random, $H_i = 1$; in the case where it is completely smooth, $H_i = 0$.

We define the entropy-based smoothness measure (EBSM) of the edge as

$$S_i = 1 - H_i. \tag{7}$$

In the case where the edge curve is completely smooth, $S_i = 1$; in the case where it is completely random, $S_i = 0$.

Let's take an example of an edge shown in Fig. 5. In (a), the positions of the edge pixels are depicted. The first shaded pixel is the starting pixel and the starting direction of the edge is determined by the second shaded pixel. Each of the subsequent pixels is given one of the five pattern labels of 0, 1, 2, 3, and 4 in the foregoing way. From the top of the edge, we make the combinations of two consecutive pattern labels to constitute the sequence. We call this the two-pattern combination. In (b), such a sequence is shown. Each two-pattern combination represents the directional changes between two consecutive edge segments. Therefore, the sequence of the two-pattern combinations can represent the degree of the smoothness of the edge. For example, at the part of the horizontal straight line, the sequence is 22, 22, and 22. At the part of the shape such as an arc, the sequence is 11, 11, and 11. At the part of the oblique straight line, the sequence is 13, 31, 13, and 31. The part of 20, 02, 22, 24, and 41 represents an almost random change. Note that the two-pattern combinations such as 20 and 02 represent the different directional changes in edge segments, respectively and thus must be treated as different pattern combinations. For the edge and the sequence of the two-pattern combination shown in Fig. 5, $S_i = 0.399$.

Table 1: Statistics of the edge saliency measures of the raw edges of the bear image.

Attribute	Length	Gradient	Smoothness
Maximum	136.4	50.2	1.000
Minimum	5.0	6.6	0.145
Mean	15.8	13.6	0.534
Std. Dv.	12.1	5.5	0.197

2.5. Edge saliency index

The next problem is how to represent the edge saliency index of an edge by combining these measures. The length, mean gradient magnitude, and EBSM of the raw edges are produced by very different methods; consequently the ranges and distributions of their values are very different. We can see these in Table 1 for the bear image. According to the analysis of many natural images, the values of the length and mean gradient magnitude of the raw edges have the following characteristics in common.

(1) Most of their values are relatively small and there are only a few large values. Consequently, the mean lies in the much lower portion of the distribution range and most of their values distribute in the neighborhood of the mean.

(2) The maximum values of their values are mostly outliers. The maximum value of each image is very different from each other, especially in the length measure.

On the other hand, for EBSM, the differences between images are not so large.

Although the normalization method with their maximum values is conventionally used to adjust the values with the different ranges and distributions, it does not work well due to the foregoing characteristics of these measures. Therefore, we devised a new normalization method based on the mean of the measure values called the mean-fitting normalization (MFN) method. We tested several ideas such as a linear combination of the measures and a product of the measures with exponents. We empirically found the following linear combination is best.

The edge saliency index (ESI) of each raw edge is constructed by a linear combination of the length, mean gradient magnitude, and EBSM considering the differences of the ranges and distributions of their values. The values of the length L_i and mean gradient magnitude G_i of an edge i are adjusted by the MFN method as follows:

$$L_i^* = \min\left(1,\ K_m \frac{L_i}{L_m}\right) , \tag{8}$$

$$G_i^* = \min\left(1,\ K_m \frac{G_i}{G_m}\right) , \tag{9}$$

where L_m and G_m are the means of the values of L_i and G_i of all the raw edges, respectively; and K_m is the normalization factor to adjust the normalization degree of the values based on the mean. Here, the same value of K_m is used in Eqs. (8) and (9) to reduce the number of parameters to be tuned. The edge saliency index (ESI) of a raw edge i is defined as

$$E_i = L_i^* + W_g\, G_i^* + W_s\, S_i , \tag{10}$$

where W_g and W_s are the parameters to adjust the contribution of each term.

As an example to explain the meaning of the MFN method, Fig. 6 illustrates the relationships of L_i and L_i^* in the MFN and in the conventional normalization method. A range R in the neighborhood of the mean

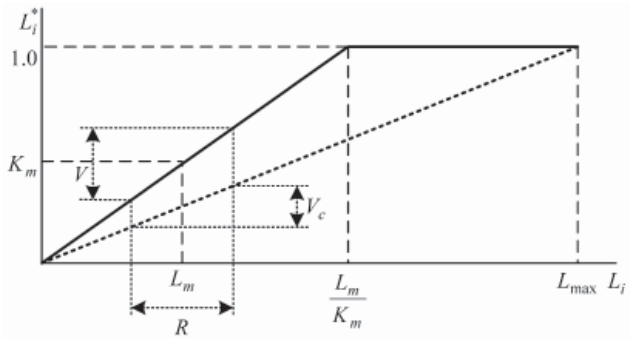


Fig. 6. The relationships of L_i and L_i^* in the MFN method and in the conventional normalization method; L_{max} is the maximum of L_i.

corresponds to the range V of the L_i^* values. In contrast, in the conventional normalization method, L_i^* is the values normalized by L_{max} and R corresponds to the range V_c of the L_i^* values. The range V is larger than V_c, if K_m is larger than L_m/L_{max}.

The MFN method works as follows.

(1) It adjusts the values of L_i and G_i so that they become appropriate magnitudes to add.

(2) By using appropriate values of K_m, it allows us to assign a suitable value to L_i and G_i in the neighborhood of the mean by which it is easy to distinguish one from another.

(3) By assigning the same value K_m to L_i and G_i corresponding to the mean of each image, most of the values of L_i^* and G_i^* become similar magnitudes for any images, because most of the values of L_i and G_i distribute in the neighborhood of the mean. This leads to the effect that the differences between these characteristics of each image are automatically adjusted. Since the differences of EBSMs between images are also not so large, it becomes feasible to use the same parameter values of W_g and W_s in common for any image. Consequently, since most of the E_i values also become similar magnitudes for any image, it becomes feasible to use the same values of the parameters in common in the subsequent processes to select boundary edges.

In contrast, as the normalization factor, the conventional normalization method uses the maximum value of each distribution which has the foregoing characteristics. Consequently, the normalized values become mostly small ones which are difficult to distinguish one from another and the locations of the ranges containing most of the values are considerably different from each other. Therefore, different parameter values need to be used for each image to obtain good results.

The proposed ESI can produce suitable values according to the values of the constituent terms. Its advantage is that it can provide a flexible expression of edge saliency for us. For example, an edge with a relatively small gradient magnitude but a long and smooth configuration, which can be a boundary of an object, can have a relatively large value of ESI. Another advantage of the proposed ESI is as follows. The values of W_g, W_s and K_m need to be tuned trial and error according to the subjective criteria and the practical requirements of the user, because recognition of whether an edge is a boundary or not depends on them in some degree. However, the parameter values once tuned by a person can be used for almost all natural images as the default values for the person, because ESI is constructed so that the differences of the image characteristics to be measured between each image are automatically adjusted in the foregoing way.

2.6. Thresholding method for selecting salient edges

Since the values of ESI of the raw edges are real numbers, its histogram is formed by using bins of a size BS_1. Generally, it consists of two populations: the first population of which elements are largely non-boundary edges with low ESI values and the second population of which elements are the mixture of non-boundary edges and boundary edges with high ESI values. The shape is almost unimodal as shown in Fig. 3 (b). That is, the peak of the first population is very high and the peak of the second population is either very low or it is submerged within the first population. In the former case, a shallow valley between both the populations can exist. In the latter case, a shoulder representing the discontinuity point of the histogram can exist. Therefore, we can determine a good threshold at the bottom of the valley or at the root of the shoulder by using a suitable BS_1.

We can determine such a threshold by analyzing the concavity structure of the histogram [44] as follows. A straight line $\bar{h}$ is drawn from the top of the histogram bar of the first peak to the top of the histogram bar of the last filled bin. Let *h(i)* be the height of the histogram bar and $\overline{h(i)}$ be the value of $\bar{h}$ at a bin number *i*, respectively. Compute the vertical distance between the top of the histogram bar *h(i)* and the straight line $\bar{h}$, that is, $\overline{h(i)}$ - *h(i)* as shown in Fig. 3 (b). We can determine the position of a good threshold by the point *i* at which this vertical distance becomes maximum, because this point is the vertically deepest concavity point. This is empirically justified by the following consideration. As *i* increases from the position of the first peak, *h(i)* drops faster than $\overline{h(i)}$, especially at the bottom of the valley or at the root of the shoulder; consequently $\overline{h(i)}$ - *h(i)* will become maximum at these points. Actually, the heights of histogram bars fluctuate locally and there may be some spurious concavities as shown in [44]. However, the fluctuations and the corresponding spurious concavities decrease by using a relatively large bin size. Therefore, we can determine a good threshold by using a suitable BS_1.

We empirically found that $BS_1 = 0.06$ is best for the images which we tested. The results shown in this paper are those which were obtained by using this value. For the bear image, the threshold 1.08 is obtained (see Fig. 3 (b)) using the parameter values described in Section 3. By using this threshold, the salient edge map shown in Fig. 2 (e) is produced. The number of the salient edges is 261.

2.7. Boundary likelihood index

Since Canny's edge detector is based on only gray levels in a small 3×3 pixel area, ESI which is composed of the information on only edges is not sufficient to distinguish boundary edges from non-boundary edges mainly due to texture. Therefore, the salient edges produced at the second stage contain not only boundary edges but also many non-boundary edges. The problem is now to compose an index to decide whether an edge corresponds to a true boundary and should be kept or whether it is due to texture and should be suppressed. Since texture is image characteristics of a relatively large area, we must use image characteristics derived from larger areas in order to distinguish edges due to texture. To this end, we use the measure of diversity proposed by Zamperoni [45], which is defined as a distance between the vectors of the rank-ordered gray levels in the two symmetrical subwindows which are set at each edge point. We construct a boundary likelihood index (BLI) to distinguish boundary edges from non-boundary edges by combining the foregoing ESI and the measure of diversity. More concretely, we define BLI of each edge as the product of ESI and the measure of diversity.

The measure of diversity is defined as follows. The observation window W of a $L_w \times L_w$ pixel area centered on a current edge pixel P is set as shown in Fig. 7 (a) and the four pairs of the symmetrical subwindows W_{ku}, W_{kv}, $k = 1,\cdots\cdots,4$ are defined in the observation window W as shown in Fig. 7 (b). The number 4 is chosen as a compromise between the angular resolution of edge directions and the computational complexity. The gray levels observed in each subwindow form a histogram as shown in Fig. 8 (a) and form the vector of its

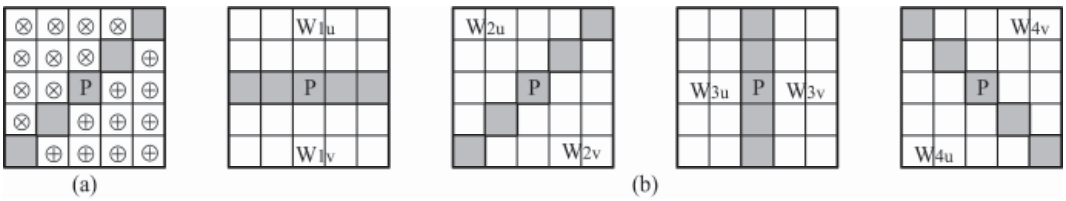


Fig. 7. Observation window: (a) the observation window W centered on the current pixel P, and (b) the four pairs of the symmetrical subwindows W_{ku}, and W_{kv}, k = 1,·····, 4 of the observation window W.

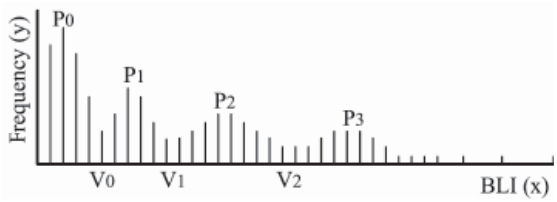


Fig. 8. Histogram and vector: (a) the histogram of the gray levels of one of the subwindows, and (b) the vector of the rank-ordered gray levels of one of the subwindows plotted against the rank i (i = 1, , N).

rank-ordered gray levels as shown in Fig. 8 (b). Let $U_k = (u_{k1,}\ u_{k2,}\cdots\cdots,\ u_{kN,})$ be the vector of the rank-ordered gray levels of the pixels of the subwindow W_{ku} and $V_k = (v_{k1,}\ v_{k2,}\cdots\cdots,\ v_{kN,})$ be the same for the other side subwindow W_{kv}, where N is the number of pixels in each subwindow. The Tanimoto distance [46] for the vectors of each subwindow pair is defined as

$$D_k = 1 - \frac{(U_k, V_k)}{\|U_k\|^2 + \|V_k\|^2 - (U_k, V_k)}, \tag{11}$$

where (U_k, V_k) is the scalar product of U_k and V_k, and $\|U_k\|$ and $\|V_k\|$ are the Euclidean norms of U_k and V_k, respectively. The measure of diversity of an edge pixel P is defined as

$$D_p = \max_{k=1,\ldots,4} D_k. \tag{12}$$

Taking the function max in Eq. (12) results in finding the edge direction by using the relatively large area of the observation window.

The value of the measure of diversity D_p is between zero and one according to the difference between the image characteristics of the subwindow pair irrespective of whether the pixel is in a textured region or in a non-textured region. In the ultimate cases, if they are identical, it becomes zero; if they are completely different, it becomes one. Consequently, a smaller value of D_p indicates that the edge pixel lies in a region with similar image characteristics. In contrast, a larger value of D_p indicates that the edge pixel lies at a border between regions with correspondingly different image characteristics. If an edge is due to texture and is not a border of different textures, the vectors of the rank-ordered gray levels of both the subwindows become similar and D_p becomes a smaller value. Therefore, we can use D_p as a measure to distinguish boundaryedges from non-boundary edges. It is preferable that the window size L_w is as small as possible in the range where the gray levels of the pixels in the subwindow can capture the image characteristics of textures. We empirically found that L_w = 17 is best for the images which we tested. The results shown in this paper are those which were obtained by using this value.

The next problem is how to compute the measure of diversity D_i of an edge i on the basis of the D_p values of the constituent edge pixels. Since a long edge may pass through some different regions, the average of the D_p values of the entire constituent edge pixels is not suitable to represent in what region the edge lies. Thus, we compute the moving average D_{aj} of the D_p values at each edge pixel j and define D_i by the maximum value of the D_{aj} values:

$$D_i = \max_j \{D_{aj}\} . \tag{13}$$

We empirically found that the moving average computed by using an operator size equal to the mean of the pixel numbers of the entire salient edges gave good results. The results shown in this paper are those which were obtained by using D_i computed in such a way.

Now, we define the boundary likelihood index (BLI) of a salient edge i as

$$B_i = E_i * D_i . \tag{14}$$

This equation means that BLI is a value of ESI which is reduced according to the differences between the image characteristics of the areas at either side of the edge. That is, the smaller the difference is, the smaller the magnitude of B_i becomes compared to E_i, and vice versa.

Since the values of BLI are real numbers, its histogram is formed by using bins of a size BS_2. For the bear image, the histogram in the case of $BS_2 = 0.015$ is shown in Fig. 3 (c). The part above the threshold 1.08 of the ESI histogram shown in Fig. 3 (b) corresponds to the histogram in Fig. 3 (c). From these, we can see that Eq. (14) can effectively produce suitable values of BLI to distinguish boundary edges from non-boundary edges by the effect of D_i, as the produced boundary edge map shows later.

2.8. Thresholding method for selecting boundary edges

Generally, the histogram of the BLI values of the salient edges consists of two populations: the first population of which elements are largely non-boundary edges with low BLI values and the second population of which elements are largely boundary edges with high BLI values. The shape is almost unimodal as shown in Fig. 3 (c). That is, the peak of the first population is very high and clear, whereas the histogram bars of the second population are fluctuated and sparse. There exist multiple peaks and valleys and consequently the border of the two populations is not clear. The problem is how to determine a suitable threshold to separate these two populations.

Over the years, many histogram-based thresholding methods have been proposed. We tested the methods proposed by Rosenfeld [44], Otsu [47], Kittler [48] and so on, which we considered promising for this problem. However, we found that these methods do not work properly and in most cases produce too large threshold values. On the other hand, Sahasrabudhe [49] proposed a threshold selection technique by detecting a deep valley using the geometric mean of the depth values of the valley, which seems promising for this problem.

Inspired by the Sahasrabudhe method, we devised and present a depth-curvature based valley seeking method (DCVSM) in order to solve this problem. Our method consists of the following two stages: gaussian filter smoothing and deep valley-seeking stages. We assume that a suitable threshold is at the bottom of the deepest and sharpest valley.

At the former stage, in order to reduce small fluctuations in the histogram bar heights, the histogram is convolved with a gaussian filter of which standard deviation is σ and filter size is W. We assume that a suitable threshold exists in the range where the BLI values are smaller than the mean of the values; we control the degree of smoothing by the number of the local minimums of the histogram bar heights in that range.

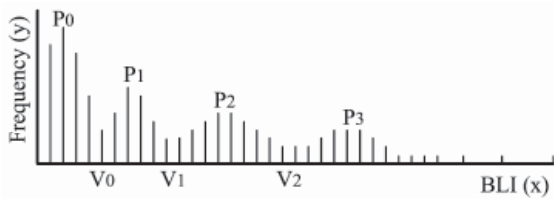


Fig. 9. The histogram of BLI values and the notations.

That is, the convolution is performed at least one time and repeated until such a number of local minimums becomes equal to or below the given number of local minimums.

At the latter stage, the candidates of thresholds are detected from the valleys of the smoothed histogram and the most suitable one is selected by a suitability measure, which is represented by the product of the measure for the depth of a valley and the curvature at the bottom of the valley. Let P_0, P_1, …., P_n be consecutive peak values in the smoothed histogram forming valleys of which values are V_0, V_1, …., V_n. See Fig. 9. The valley V_k is formed by the peaks P_k and P_{k+1} and represented by a parabola through these three points defined as

$$y = a * x^2 + b * x + c \,. \tag{15}$$

By Eq. (15), the bottom of the valley $V_{k\,b}$ is located and the curvature at the bottom of the valley and the threshold x_t are computed by

$$C = 2 * a \,, \tag{16}$$

$$x_t = -\frac{b}{2 * a} \,. \tag{17}$$

Its depths with respect to the two peaks are $(P_k - V_{k\,b})$ and $(P_{k+1} - V_{k\,b})$, respectively. The measure for the depth of this valley is defined by the square of the geometric mean of these two values as

$$D = \left(P_k - V_{kb}\right) * \left(P_{k+1} - V_{kb}\right). \tag{18}$$

The geometric mean is more desirable for selecting a balanced valley. The suitability measure S of this threshold is defined as

$$S = C * D \,. \tag{19}$$

The algorithm of DCVSM is given below.

Let N_e and N_g be the number of the local minimums of the histogram bar heights which exist in the range where the BLI values are smaller than the mean of the values and the given number of local minimums, respectively.

Step 1: Compute the bin number which corresponds to the mean of the initial BLI values.

Step 2: Convolve the histogram with the gaussian filter defined by σ and W.

Step 3: Compute N_e.

Step 4: Compare N_e with N_g. If N_e is larger than N_g, then go to Step 2, otherwise go to Step 5.

Step 5: Normalize the histogram so that the summation of the bar heights becomes one.

Step 6: Detect the peaks and valleys in the normalized histogram.

Step 7: Compute x_t, C, D, and S for each valley.

Step 8: Search the valley corresponding to the maximum value of S (S_{max}) and select its location x_t as the most suitable threshold T_b.

The shape of the histogram of BLI values changes extremely with the bin size used to form it. If the bin size

is very small, the histogram is sparse and there are many shallow valleys. If the bin size becomes larger, the first peak becomes higher and a few deep and sharp valleys emerge. If the bin size becomes very large, there exists no valley and the shape becomes unimodal. In the case of the most suitable bin size, the deep and sharp valley which can separate two populations can emerge. Therefore, the bin size is very important in selecting a suitable threshold. At present, however, it is difficult to determine the most suitable bin size for a given image theoretically, because the suitability of a threshold should be evaluated by the boundary edge map produced using it. However, we empirically found that a good threshold exhibits a large value of S_{max}, though the threshold corresponding to the largest value of S_{max} does not necessarily produce the most satisfactory boundary edge map. Therefore, we can use S_{max} as a cue to select a good threshold. We empirically found that the suitable bin size exists between 0.001 and 0.03 for the images which we tested. Therefore, we adopted the following strategy: we compute thresholds using bin sizes at an interval 0.001 in this range and select several thresholds exhibiting larger values of S_{max} from them as candidates; then, we produce boundary edge maps by using these candidate thresholds and select the most satisfactory one from them. If we want more desirable results, we can use a threshold tuned manually in the neighborhood of such an automatically computed threshold.

Our DCVSM has the following features.

(1) It can detect a threshold at the bottom of the deepest and sharpest valley near the first peak in the histogram.

(2) It can detect a good threshold which can separate the two populations by using the smoothing of the histogram to eliminate small fluctuations and controlling its degree semi-automatically.

(3) The accurate location of a threshold can be computed by using parabolic interpolation.

(4) The best threshold can be selected by changing the bin size and using S_{max} as the cue for the suitability.

The values of the parameters σ, W, and N_g need to be determined in order not to eliminate essential changes in histogram bar heights. We empirically found that σ = 1.0, W = 3, N_g = 3 are best for the images which we tested. The results shown in this paper are those which were obtained by using these values. For the bear image, the best threshold T_b = 0.0963 is obtained in the case of the bin size BS_2 = 0.015 (see Fig. 3 (c)) using the parameter values described in Section 3. By using this threshold, the boundary edge map shown in Fig. 2 (f) is obtained. The number of the selected boundary edges is 48. Comparing Fig.2 (f) to (e), we can see that most of the non-boundary edges in the textured regions are eliminated by the effect of D_i in Eq. (14).

2.9. Comparison with related work

Our approach is classified into the method considering only the features of each edge, regardless of the reciprocal relation of edges, in order to detect boundaries. In the related work, for example, the following characteristics were used: gradient magnitude and continuity [1], length and contrast [9], gradient magnitude and surrounding texture edges [12, 13], gradient magnitude and edge presence confidence measure [15], length and gradient magnitude [16, 17], distances between textures called the measure of diversity [18]. Our approach is distinguished from these methods in following respects.

(1) ESI of each edge is a linear combination of its length, mean gradient magnitude, and EBSM which can provide a flexible expression of edge saliency for us. The length and mean gradient magnitude are adjusted based on the MFN method so as to consider the differences between these characteristics of each image automatically.

(2) BLI of each edge is represented by the product of ESI and the measure of diversity considering the image characteristics of the relatively large neighboring area of the edge. Consequently, it allows us to detect edges with high boundary likelihood considering both the edge saliency and the image characteristics around the

edge.

Chen [43] presented a measure for representing the structural complexity of a shape contour by using the local angles between two consecutive contour segments and evaluating the entropy of the probability distribution computed from their histogram. Our EBSM is distinguished from Chen's method in the following respect: we define an edge segment formed by a pair of two consecutive edge pixels and patterns representing the directional changes in consecutive edge segments; for each edge, we use the histogram for the combinations of the patterns of multiple consecutive edge segments to evaluate its smoothness measure. Use of the combinations of such multiple consecutive patterns allows us to represent the more complex variations of the positions of edge pixels.

Our DCVSM is distinguished from Sahasrabudhe's method [49] in that it is devised so as to detect a suitable and accurate threshold as follows: the function of smoothing histograms is added to eliminate small fluctuations of histograms and the smoothing degree is controlled semi-automatically, and the accurate location of a threshold can be determined by combining the depth and curvature of a valley using parabolic interpolation.

3. Experiments and the results

We performed the experiments on the RuG [37], Berkeley [38], and South Florida [39] data sets using the Java implementation of our method under Windows XP. For computation of EBSM, the two-pattern combination was used.

We evaluated the selected boundary edge maps quantitatively by the F-measure using human-marked boundaries as the ground truth. Let *DB*, *GT*, and *CB* be the number of the pixels of the boundary edges which are actually detected by the algorithm, the number of the pixels of the ground truth edges, and the number of the correctly detected pixels in *DB*, respectively. The precision *P* and the recall *R* are defined by the ratios of *CB* to *DB* and *CB* to *GT*, respectively. The F-measure is defined as

$$F = \frac{2PR}{P + R}. \tag{20}$$

Human-marked boundaries are different according to the observer's subject and not perfect; some boundaries are missing and the locations are displaced. Therefore, in computing the values of *CB*, the localization tolerance needs to be considered. We used a distance of d_{max} as the localization tolerance as in [26], where as the value of d_{max}, one percent of the image diagonal (2.88 pixels) was used. Since the locations of our selected boundary edges are represented in pixels, we used d_{max} = 3 pixels. Since the Berkeley data set is region segmentation data and the extracted boundary edges from it are two pixel wide, we counted one pixel of the ground truth boundary edges among d_{max}.

Our method has several parameters whose values the user needs to determine. The parameter values which are hardly dependent on the characteristics of each image and thus can be determined mainly based on the strategy to obtain good solutions are described in the foregoing Sections. The remaining parameters to be tuned are W_g and W_s in Eq. (10), and K_{m} in Eqs. (8) and (9). We tuned these parameter values using the following five images of the RuG data set: bear, elephant, rhino, golfcart, and tire. We selected these images containing various objects with the aim of obtaining suitable parameter values which can be used in common for other various images. We tuned these values trial and error according to our subject so that the best boundary edge maps were obtained. As a result, the best values were W_g = 0.8, W_s = 0.5, and K_m = 0.3.

We performed the experiments by using these parameter values in common for all images which we tested and produced the boundary edge maps by using the threshold values computed by DCVSM automatically. We

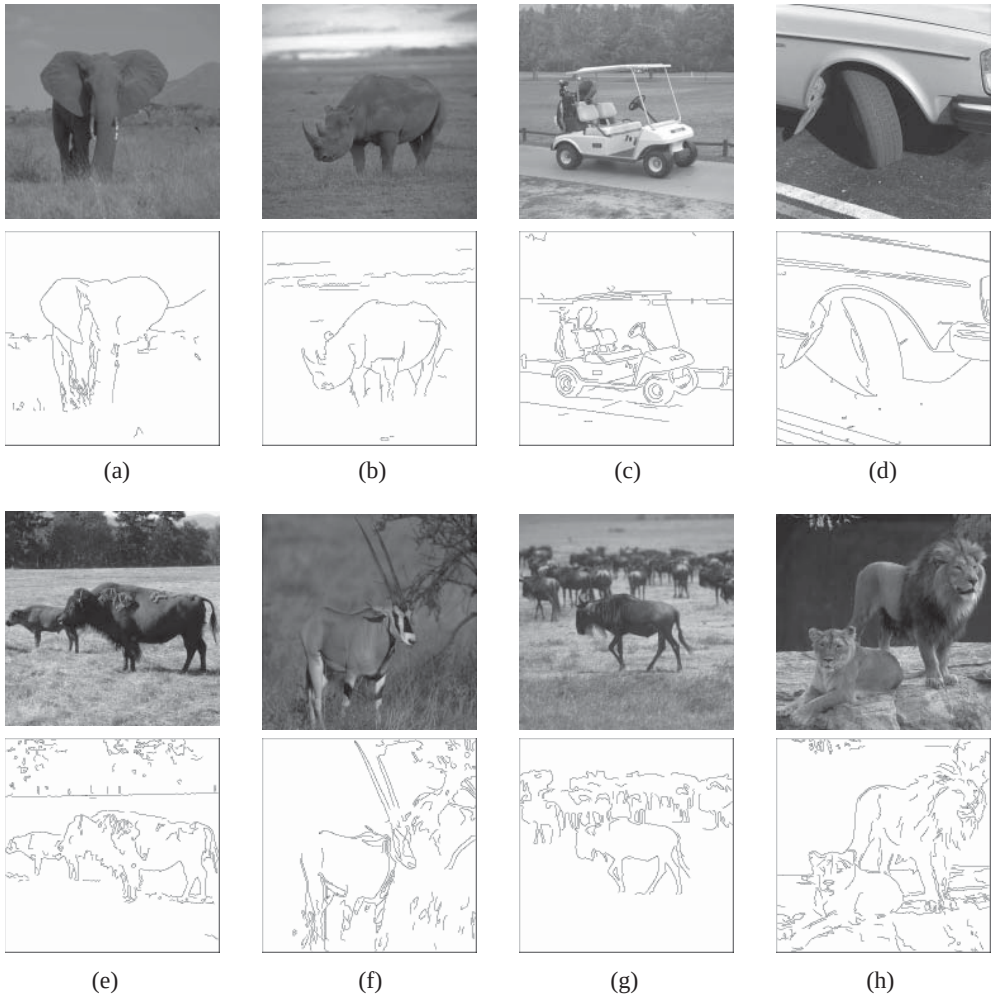


Fig. 10. The results on the RuG data set: (a) elephant_2 (F = 0.791), (b) rhino (F = 0.748), (c) golfcart (F = 0.784), (d) tire (F = 0.758), (e) buffalo (F = 0.699), (f) gazelle (F = 0.766), (g) gnu (F = 0.785), and (h) lions (F = 0.758).

selected the best boundary edge maps in the way described in Section 2.8. Here, we show the obtained best boundary edge maps with the F-measure values (F). We report on as many images containing different objects as possible in order to show the effectiveness of our method and compare our results qualitatively with the results in the previous papers. Figures 2 and 10 show those on the RuG data set. Figure 11 shows those on the Berkeley data set. In the Berkeley data set, since multiple segmentation data by different users are provided, we show which one we used. Figure 12 shows those on the South Florida data set.

In addition, we show the results of the experiments for the relationships between the performance and the parameter values on the bear image. Figure 13 shows the relationships between the F-measure, precision, and recall and the parameter values W_g, W_s, and K_m. The fluctuations in the F-measure are due to the fluctuations in the threshold values caused by the changes of shapes of the histograms of BLI which were formed using

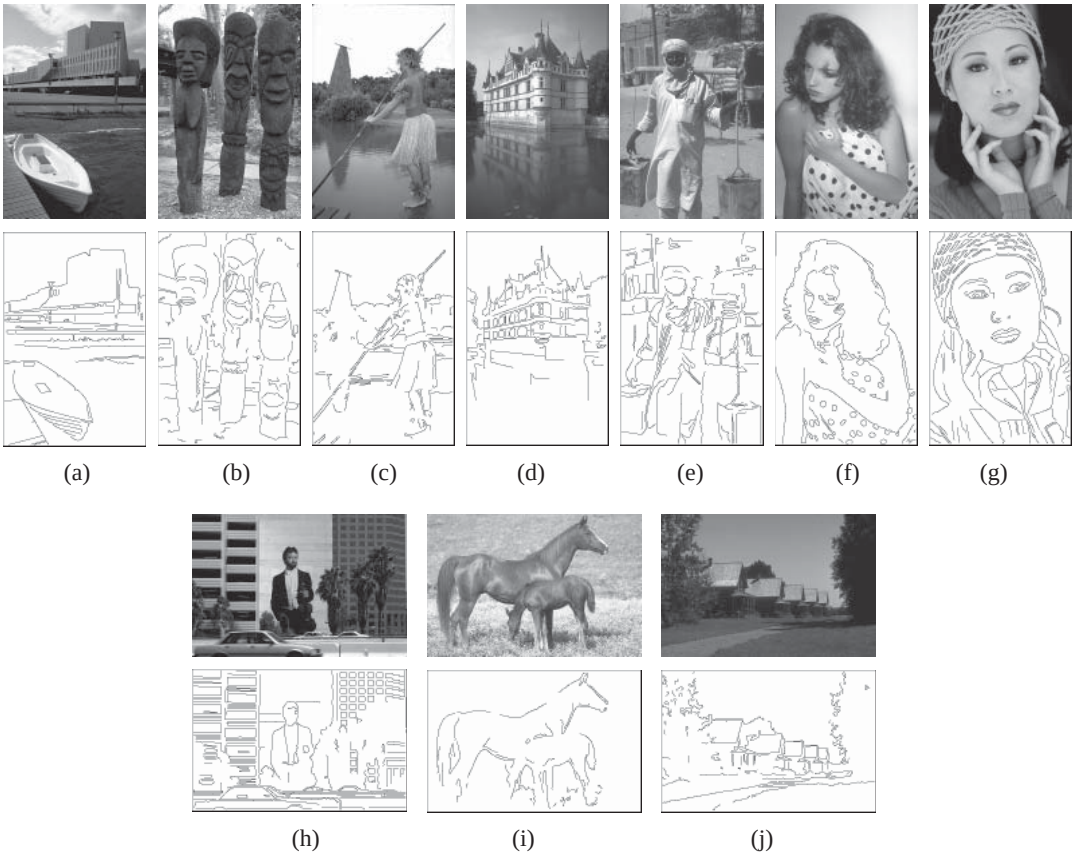
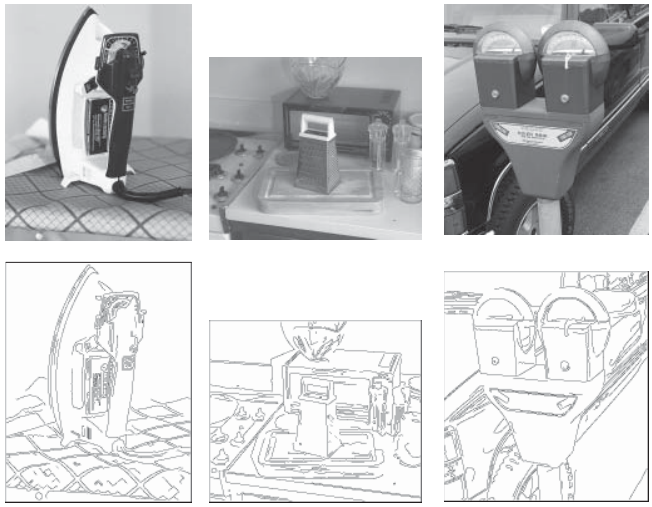


Fig. 11. The results on the Berkeley data set: (a) no.78004 (F = 0.678 for #1130), (b) no.101085 (F = 0.561 for #1123), (c) no.101087 (F = 0.760 for #1119), (d) no.102061 (F = 0.580 for #1123), (e) no.271035 (F 0.545 for #1130), (f) no.198004 (F = 0.692 for #1123), (g) no.302003 (F = 0.562 for #1119), (h) no.119082 (F = 0.625 for #1124), (i) no.113044 (F = 0.676 for #1123), and (j) no.232038 (F = 0.633 for #1123).

different bin sizes. Figure 14 shows the results with the best-tuned parameter values of which only one parameter value was changed to the lower limit or the upper limit of the nearly optimal range of the parameter values. Figure 15 shows the relationships between the F-measure, precision, and recall and the threshold value T_b which was manually changed. Fig. 16 shows the precision-recall curve for the bear image. Table 2 shows the mean and standard deviation of the F-measure for the RuG data set (40 images). For reference, those by the Papari [24] and Grigorescu [12] methods which were read from Fig. 16 in [24] are shown.

The major difference of the computational complexity between the standard Canny edge detector and our method is the processes of the edge following and computation of ESI values, and the process of the computation of BLI values. For the RuG data set (40 images), the average computation times were 0.37 second and 0.15 second for the former and latter processes, respectively on a PC with a Pentium 4 CPU 3.20 GH_z. Since we adopt the strategy of computing the BLI values after selecting the salient edges, its complexity is not so high.



(a) (b) (c)

Fig. 12. The results on the South Florida data set: (a) no.101 (F = 0.778), (b) no.214 (F = 0.840), and (c) parkingmeter (F = 0.821).

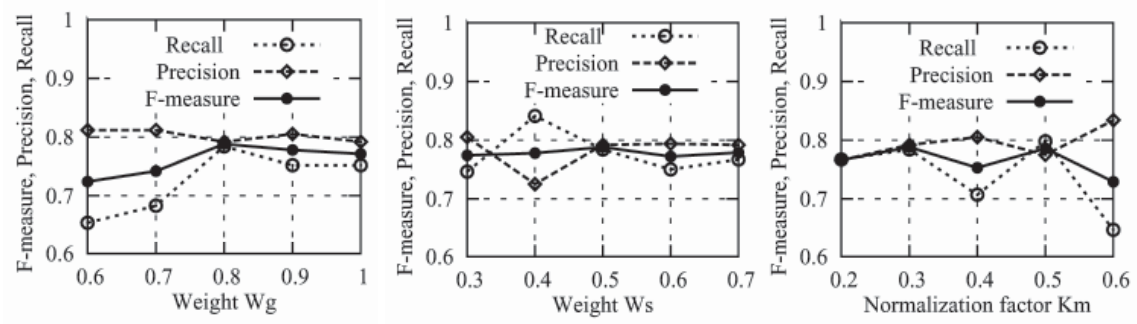


Fig. 13. The relationships between the F-measure, precision, and recall, and the parameter values W_g,, W_s, and K_m, respectively, for the bear image.

4. Summary of the results and Discussions

In evaluation of boundary detection methods, we should note the following. The desired boundaries are different depending on the use. Also, as human-marked boundaries indicate, boundaries perceived by the observers are different according to their subject. Therefore, the evaluation of boundary detectors is more or less dependent on these conditions.

The results of the experiments are summarized as follows.

(1) From Fig. 2 (d), (e), and (f), we can see that the relatively good estimate of the boundary edge map of the bear image is produced, because ESI and BLI work effectively to represent the saliency and the boundary likelihood of edges, respectively.

(2) From the optimum values of W_g and Ws, we can see that the length measure is most important and the mean gradient magnitude measure follows it. From Figs. 2 (f) and 14 (c), we can see that EBSM contributes to

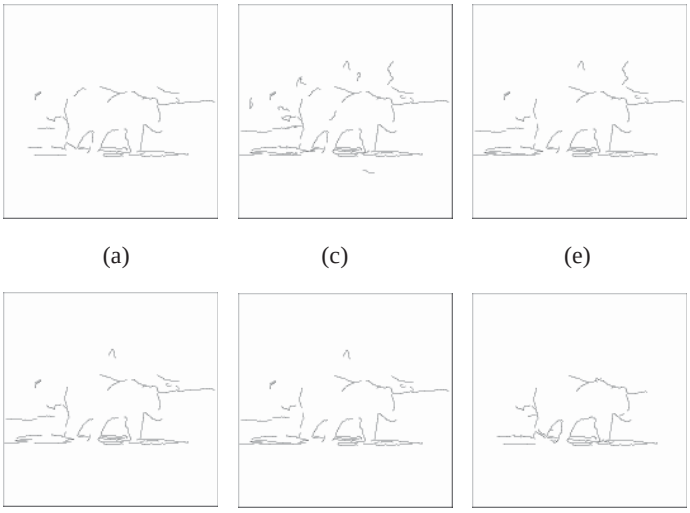


Fig. 14. The results on the bear image with the best-tuned parameter values of which only one parameter value was changed into the following value: (a) W_g = 0.6, (b) W_g = 1.0, (c) W_s = 0.0, (d) W_s = 0.7, (e) K_m = 0.2, and (f) K_m = 0.6.

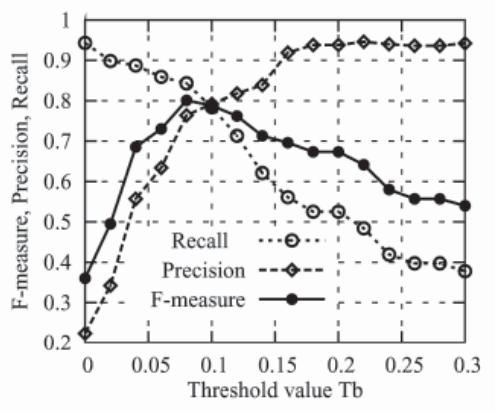


Fig. 15. The relationships between the F-measure, precision, and recall, and the threshold T_b for the bear image.

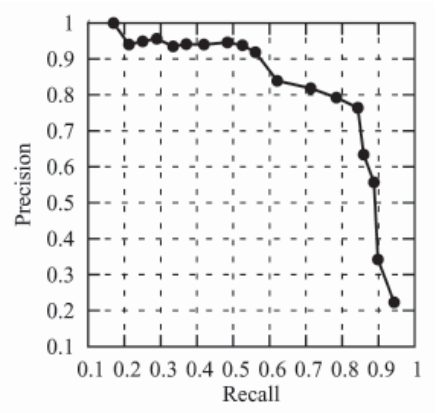


Fig. 16. The precision-recall curve for the bear image.

eliminating non-boundary edges which can not be eliminated by only the length and mean gradient magnitude measures.

(3) From Figs. 10, 11, and 12, we can see that the parameter values of K_m, W_g, and Ws tuned by the five images produce the relatively good boundary edge maps of the other images and thus ESI based on the MFN method works effectively to adjust the differences between the characteristics of natural images.

(4) From Figs. 13 and 14, we can see that there exist the relatively wide ranges of almost optimal values of K_m,

Table 2: The mean and standard deviation of the F-measure values for the RuG data set (40 images).

Method	Mean	Std. Dv.
Our method	0.67	0.12
Papari method [25]	0.57	0.12
Grigorescu method [14]	0.48	0.14

W_g, and Ws, and thus the performance is not so sensitive to the these parameter values. From Table 2, we can see that the standard deviation of the F-measure is relatively small and thus the performance is robust.

(5) From Figs. 3 (b) and (c), and Fig. 15, we can see that the thresholding methods based on the histogram concavity analysis and DCVSM work effectively to determine suitable thresholds automatically. In Fig. 15, the recall value is equal to 0.943 for T_b = 0.0; this shows that the salient edge map contains almost all the pixels of the ground truth edges, because the edge following method and the thresholding method based on the histogram concavity analysis work effectively. Fig. 15 also shows that the threshold T_b = 0.0963 determined by DCVSM is almost optimal.

(6) From these results, it can be seen that the local image-feature integration method (LIFIM) and its components work effectively to detect edges with high boundary likelihood in the natural images.

Since the details of the implementation of the algorithms and the parameter values used are not clearly described in the previous papers and thus it is impossible to reproduce the same results reported in the previous papers, it is difficult to make quantitative comparisons between the results by our results and those in the previous papers. Therefore, for the convenience of the qualitative comparisons of the readers, we refer to the Figures in the previous papers which correspond to those by our methods: Figure 1 in [5]; Figs. 7 and 9 in [12]; Figures 9, 10, 11, 12, 17, 18, 19, and 20 in [13]; Figures 8, 9, and 10 in [14]; Figs. 11 and 12 in [15]; Figs. 3 and 7 in [17]; Figs. 15, 19, and 20 in [24]; Figs. 15 and 16 in [26]; Fig. 18 in [28]; and Figure 8 in [29]. From the qualitative comparisons with these previous results and the quantitative comparisons shown in Table 2, it can be seen that our results are almost comparable to most of those of these previous papers. It should be noted that our method considering only the features of each edge can produce the results almost comparable to most of those of the modern approaches based on the more complex techniques such as the edge element grouping techniques and the learning processes of prior knowledge on boundaries.

The algorithms of most of the modern approaches described in Section 1.2 are very complex, difficult to implement, and computationally demanding. In contrast, our method is simple and easy to implement, less troublesome to tune, and computationally cheap. Although our method uses several parameters that the user needs to tune, once the user tunes them using several images, they can be used to other images in common.

The limitation of our method is derived from Canny's edge detector which is based on the ideal step edge model and gray levels in a small 3×3 pixel area. The produced edges are apt to be fragmentary. Also, it is difficult to detect boundaries between textured regions. Resolving these problems is our future work. A possible idea to resolve the first problem is the edge completion using boundary edges detected by our method as the seeds.

5. Conclusions

We have proposed a method for detecting boundary edges based on the local image-feature integration method using the modified Canny edge detector, and performed experiments to investigate and evaluate its performance on the standard image data sets. The results have shown that our method works effectively and can

produce relatively good estimates of boundary edges in natural images and its performance is robust. Our method has the feature that once the user tunes the parameters using several images, they can be used to other images in common.

References

[1] J Canny, A computational approach to edge detection, IEEE Trans. Pattern Analysis and Machine Intelligence PAMI-8 (6) (1986) 679-698.
[2] OpenCV, http://opencv.willowgarage.com/wiki/, December 2009.
[3] F. Bergholm, Edge focusing, IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-9 (6) (1987) 726-741.
[4] P. Bao, L. Zhang, X. Wu, Canny edge detection enhancement by scale multiplication, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (9) (2005) 1485-1490.
[5] B. Sumengen, B.S. Manjunath, Multi-scale edge detection and image segmentation, in: Proceedings of the European Signal Processing Conference, September 2005.
[6] P. Perona, J. Malik, Scale-space and edge detection using anisotropic diffusion, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (7) (1990) 629-639.
[7] M.J. Black, G. Sapiro, D.H. Marimont, D. Heeger, Robust anisotropic diffusion, IEEE Transactions on Image Processing 7 (3) (1998) 421-432.
[8] K.H. Liang, T. Tjahjadi, Y.H. Yang, Bounded diffusion for multiscale edge detection using regularized cubic b-spline fitting, IEEE Transactions on System, Man, and Cybernetics PART B 29 (2) (1999) 291-297.
[9] J.M. Prager, Extracting and labeling boundary segments in natural scenes, IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-2 (1) (1980) 16-17.
[10] E.R. Hancock, J. Kittler, Edge-labeling using dictionary-based relaxation, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (2) (1990) 165-181.
[11] I. Matalas, R. Benjamin, R. Kitney, An edge detection technique using the facet model and parameterized relaxation labeling, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (4) (1997) 328-341.
[12] C. Grigorescu, N. Petkov, M.A. Westenberg, Contour and boundary detection improved by surround suppression of texture edges, Image and Vision Computing 22 (2004) 609-622.
[13] G. Papari, P. Campisi, N. Petkov, A. Neri, A biologically motivated multiresolution approach to contour detection, EURASIP Journal on Advances in Signal Processing 2007 (1) (2007) 1-28.
[14] J. Ng, A.A. Bharath, P.C.P. Kin, Extrapolative spatial models for detecting perceptual boundaries in color images, International Journal of Computer Vision 73 (2) (2007) 179-194.
[15] P. Meer, B. Georgescu, Edge detection with embedded confidence, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (12) (2001) 1351-1365.
[16] S. Venkatesh, P.L. Rosin, Dynamic threshold determination by local and global edge evaluation, Graphical Models and Image Processing 57 (2) (1995) 146-160.
[17] P.L. Rosin, Comments on “ground from figure discrimination”, Pattern Recognition Letters 24 (2003) 2761-2766.
[18] S. Philipp, P. Zamperoni, Segmentation and contour closing of textured and non-textured images using distances between textures, in: Proceedings of International Conference on Image Processing, September 1996, pp. 125-128.

[19] D.G. Lowe, Three-dimensional object recognition from single two-dimensional images, Artificial Intelligence 31 (1987) 355-395.

[20] A. Sha'ashua, S. Ullman, Structural saliency: the detection of globally salient structures using a locally connected network, in Proceedings of the Second International Conference on Computer Vision, Florida, USA, December 1988, pp. 321-327.

[21] L. Herault, R. Horaud, Figure-ground discrimination: a combinatorial optimization approach, IEEE Transactions on Pattern Analysis and Machine Intelligence 15 (9) (1993) 899-914.

[22] A. Amir, M. Lindenbaum, Ground from figure discrimination, Computer Vision and Image Understanding 76 (1) (1999) 7-18.

[23] Q. Tang, N. Sang, T. Zhang, Extraction of salient contours from cluttered scenes, Pattern Recognition, 40 (2007) 3100-3109.

[24] G. Papari, N. Petkov, Adaptive pseudo dilation for gestalt edge grouping and contour detection, IEEE Transactions on Image Processing 17 (10) (2008) 1950-1962.

[25] S. Konishi, A.L. Yuille, J.M. Coughlan, S.C. Zhu, Statistical edge detection: learning and evaluating edge cues, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (1) (2003) 57-74.

[26] D.R. Martin, C.C. Fowlkes, J. Malik, Learning to detect natural image boundaries using local brightness, color, and texture cues, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (5) (2004) 530-549.

[27] X. Ren, C.C. Fowlkes, J. Malik, Cue integration for figure/ground labeling, Proceedings of NIPS, 2005, pp. 1121-1128.

[28] X. Ren, C.C. Fowlkes, J. Malik, Learning probabilistic models for contour completion in natural images, International Journal of Computer Vision 77 (2008) 47-63.

[29] P. Dollar, Z. Tu, S. Belongie, Supervised learning of edges and object boundaries, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, New York, June 2006, pp. 1964-1971.

[30] H.M. Gonzalez-Velasco, C.J. Garcia-Orellana, M. Macias-Macias, F.J. Lopez-Aligue, M.I. Acevedo-Sotoca, Neural-networks-based edges selector for boundary extraction problems, Image and Vision Computing 22 (2004) 1129-1135.

[31] I. Kokkinos, R. Deriche, O. Faugeras, P. Maragos, Computational analysis and learning for a biologically motivated model of boundary detection, Neurocomputing 71 (2008) 1798-1812.

[32] S. Zheng, Z. Tu, A.L. Yuille, Detecting object boundaries using low-, mid-, and high-level information, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, USA, June 2007, pp. 2652-2659.

[33] T. Pavlidis, Y.T. Liow, Integrating region growing and edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (3) (1990) 225-233.

[34] E. Saber, A.M. Tekalp, G. Bozdagi, Fusion of color and edge information for improved segmentation and edge linking, Image and Vision Computing 15 (1997) 769-780.

[35] S. Wang, J.M. Siskind, Image segmentation with ratio cut, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (6) (2003) 675-690.

[36] L. Bertelli, B. Sumengen, B.S. Manjunath, F. Gibou, A variational framework for multiregion pairwise-similarity-based image segmentation, IEEE Trans. Pattern Analysis and Machine Intelligence 30 (8) (2008) 1400-1414.

[37] The RuG data set, http://www.cs.rug.nl/~imaging/, December 2009.

[38] D. Martin, C. Fowlkes, D. Tal, J. Malik, A Database of human segmented natural images and its application

to evaluating segmentation algorithms and measuring ecological statistics, in: Proceedings of the 8th International Conference on Computer Vision, vol. 2, Vancouver, Canada, July 2001 pp. 416—423, http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/
segbench/, December 2009.

[39] The South Florida data set, http://marathon.csee.usf.edu/edge/edgecompare_main.html/, December 2009.

[40] M.M. Fleck, Some defects in finite-difference edge finders, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (3) (1992) 337-345.

[41] H. Voorhees, T. Poggio, Detecting textons and texture boundaries in natural images, in: Proceedings of the First International Conference on Computer Vision, London, June 1987, pp. 250-258.

[42] B. Vasselle, G. Giraudon, A multiscale regularity measure as a geometric criterion for image segmentation, Machine Vision and Applications 7 (1994) 229-236.

[43] Y. Chen, H. Sundaram, Estimating complexity of 2D shapes, in: IEEE 7th Workshop on Multimedia Signal Processing, Shanghai, October 2005, pp. 1-4.

[44] A. Rosenfeld, P.D.L. Torre, Histogram concavity analysis as an aid in threshold selection, IEEE Transactions System, Man, and Cybernetics SMC-13 (3) (1983) 231-235.

[45] P. Zamperoni, Model-free texture segmentation based on distances between first-order statistics, Digital Signal Processing 5 (1995) 197-225.

[46] T. Kohonen, Self-Organization Maps, Springer-Verlag, Berlin, 1995.

[47] N. Otsu, A threshold selection method from gray-level histograms, IEEE Transactions on System, Man, and Cybernetics SMC-9 (1) (1979) 62-66.

[48] J. Kittler, J. Illingworth, Minimum error thresholding, Pattern Recognition 19 (1) (1986) 41-47.

[49] S.C. Sahasrabudhe, K.S.D. Guputa, A valley-seeking threshold selection technique, in: L. Schapiro (Ed.), Computer Vision and Image Processing, Academic Press, 1992, pp. 55-65.