Gradient Center Tracking: A Novel Method for Edge Detection and Contour Detection

Yipei Su, Xiaojun Wu(✉), and Xiaoyou Zhou

Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China
wuxj@hit.edu.cn

Abstract. Detecting complete contours with less clutters is a very challenging task in edge detection. This paper presents a new lightweight edge detection method, Gradient Center Tracking (GCT), to detect the main contours including the boundary and the structural lines of the objects. This method tracks the center curve of contours in the gradient image and detects edges while tracking. It makes full use of the edge correlation and contour continuity to choose edge candidates, then computes the gradient intensities of the candidates to select the real edge. In this method, the intensity of the edge is redefined as the Directional Weighted Intensity (DWI) which helps to present the result with more complete contours and less clutters. The GCT method outperforms $Canny$ detector and shows better results than several learning based methods. The comparison results are shown in our experiments and a typical scheme to apply the GCT method is also provided.

Keywords: Edge detection · Complete contours · Less clutters

1 Introduction

Edge detection, which aims to extract visually salient edges and object boundaries from natural images, is one of the most studied problems in computer vision. It is usually considered as a low-level technique, and varieties of high-level tasks have greatly benefited from the development of edge detection, such as object detection [7,23] and image segmentation [4,17,26]. Broadly speaking, edge detection methods can be generally grouped into two categories. (1) the classical edge detection methods based on brightness gradient and image filters represented by *Roberts* [20], *Prewitt* [18], *Sobel* [9], zero-crossing [15], and *Canny* [2]. (2) the modern methods, including methods based on the probability distributions and cluster [1,10,14], and methods based on learning [5,19,24]. Methods in the first category are usually lightweight and fast with good detection results in many cases. The *Canny* detector is nearly the most widely used edge detection method even now. However, these methods simply apply a high-pass filter to detect edges, which lead to lots of redundancy. Furthermore, these

J.-H. Lai et al. (Eds.): PRCV 2018, LNCS 11256, pp. 397–407, 2018.
https://doi.org/10.1007/978-3-030-03398-9_34

methods show poor effect in a complex situation as their limited use of features. The proposed method in this paper, Gradient Center Tracking (GCT), is also based on brightness gradient and image filter, however, we redefine the gradient intensity of an edge by Directional Weighted Intensity (DWI) and set a tracking strategy Inherited Edge Detection (IED) with special starting points to effectively detect contours with less clutters. Directional structure elements help to complete our detected contours comparing to *Canny* results. In the second category, the modern methods [6,13] show the state-of-the-art performance in some specific high-level applications such as object segmentation. However, there are still two main problems remained. (1) The modern edge detection methods are always much more complex in computation and demand higher-performance equipment. (2) They fail to meet the single response rule, which means that only one point should be detected for each given edge. This rule is declared by *Canny* and widely accepted in this filed. On the contrary, our GCT method is lightweight and fully meet the single response rule. Furthermore, for the contour detection task, both these two categories are two-step methods that they first detect edges independently, then connect them into contours by post process, such as topology coding [21]. On the contrary, our GCT method gets contours and edges simultaneously, which means no post process is needed and the detection result can apply to some subsequent work directly, such as segmentation task.

In recent years, the research on edge detection has developed into different directions to serve different applications. For examples, most of the learning-based methods are used in the object segmentation task [3,6,13]. While classical edge detection methods like *Canny* and lots of other improved method based on *Canny* [8,11,12,16,22,25] are widely adopted in the situation with simpler scene and high real-time requirements, such as the detecting tasks in industrial application. This paper presents a new lightweight edge detection method, Gradient Center Tracking (GCT), to extract the main contours including the boundary and the structural lines of the objects in a gray image. This method tracks the center curve of a contour in the gradient image and detect edges while tracking. It presents the result with more complete contours and less clutters. It outperforms *Canny* detector and shows better results than several learning based methods in some application, such as the industrial scene.

The main contributions of this paper are as follows. (1) Propose a novel method, GCT, to unite edge detection and contour detection. (2) Propose a new local searching scheme based on the edge correlation and contour continuity. (3) Redefine the gradient intensity of an edge as weighted intensity along a certain direction (DWI).

Before the detailed description, it is necessary to explain the relationship among an edge, contours and edges. In this paper, an edge is defined as a detected point in an image, and "edges" means the detection result which can be divided into several contours according to visual perception. The details of GCT method are stated in Sect. 2 and the results of our experiment are presented in the Sect. 3.

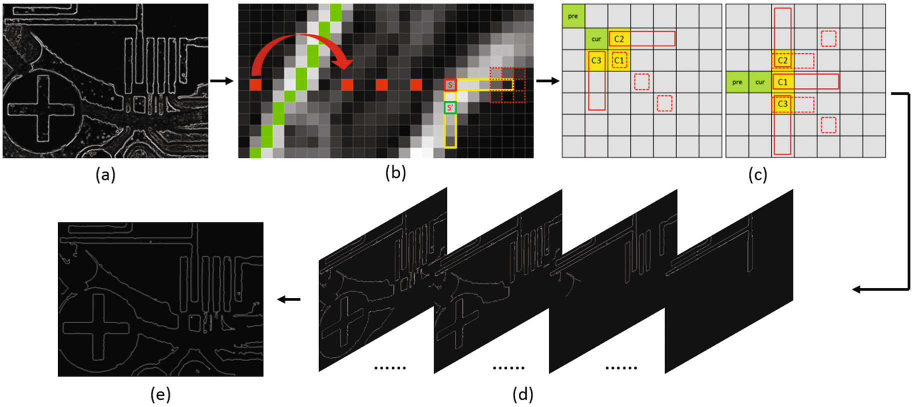

Fig. 1. Illustration of the *GCT* method. From (a) to (e): gradient image, searching new starting point, *IED* and *DWI*, tracking all the contours, detection result. Firstly, blur the source image to get a gradient image, then search a new starting point and apply the inherited edge detection method (*IED*) and a new defined edge intensity (*DWI*) to repeatedly find the "next edge" to complete the current contour. All the contours form the detection result.

2 Gradient Center Tracking Method

This paper proposes the Gradient Center Track method (*GCT*) to make full use of the edge correlation and contour continuity. Following this idea, the GCT method is designed as shown in Fig. 1. A given image will first be smoothed by *Sobel* or other detectors to compute the gradient image. Then *GCT* method begins by searching starting points in the gradient image and extend the contour by tracking the center of the high intensity band. While tracking, the Inherited Edge Detection (*IED*) strategy is applied to decide which point should be the next edge in current contour. A new definition of edge intensity marked as Directional Weighted Intensity (*DWI*) is used here. *GCT* method will repeatedly search a new starting point and track contours until all the contours are found.

2.1 Starting Point Selection

The first step of *GCT* method is to find a starting point. Figure 2 shows the strategy to select the starting points. It will first find a rough staring point and then modify it to be a real one. The *GCT* method sets a high enough threshold T_s in the gradient intensity and search the whole gradient image to choose the rough starting points. During this searching process, it skips the points that have been marked as edges already. Usually, a rough starting point is not a real edge, even though it is always close to the center of the gradient band.

In order to find a better starting point, the *GCT* method extends the rough starting point along the positive direction of the image coordinate axes, and then compute the weighted intensity to choose the best starting point. For example,

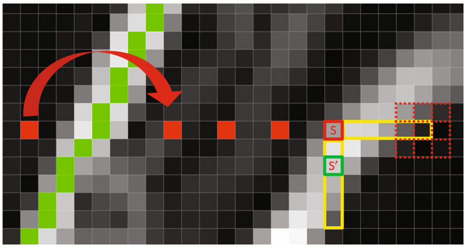

Fig. 2. Illustration of starting point searching in the gradient image. Every small square represents a pixel point, and there are two strong contours in this small picture. The searching will skip the first contour as it has already been detected (paint in green), then choose the point S (the single point with red box) and extend along the positive directions (the yellow boxes), next value each extended points by weighted intensity (the 3×3 region with red boxes), finally modify S into $S^{'}$ (the single point with green boxes). (Color figure online)

if the rough starting point is $P(x, y)$, then all the points $P(x+1, y)$, $P(x+2, y)$ $\ldots$ $P(x+t, y)$ and $P(x, y+1)$, $P(x, y+2)$ $\ldots$ $P(x, y+t)$ (t is set to be 5 in our experiment) will be valued by weighted intensity in a small region (a 3×3 region is used in the experiments), and the point with the highest intensity will be chosen to be the real starting point.

2.2 Define an Edge by Directional Weighted Intensity

For most of the exist edge detection methods, a prescribed gradient intensity threshold is required and the edges are the points with gradient intensity higher than the threshold. They use only the intensity of the point itself to compare with the threshold. However, this definition is based on a hypothesis that all the real edges have the local highest intensity. But it is difficult to make this assumption come true in many cases. For example, with noise in it, sometimes the intensity of the real edge may be a little lower than its neighbor points, although both the intensity of them are higher than the threshold. In this situation, the real edge will be abandoned for its non-maximum intensity according to the general edge definition.

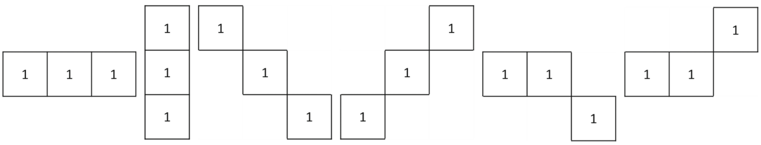

Fig. 3. Structural elements for direction expansion.

In this paper, the intensity of the edge is redefined as the weighted intensity along a directional, marked as DWI. The direction depends on the precious edge in the same contour. This definition contains the edge correlation and contour continuity, which can better represent the gradient intensity of a real edge to a certain extent. The advantage of this definition is showed in our experiment in Sect. 3. In our method, to match this definition, we set structural elements, such as rectangle kernel and diagonal kernel, to compute the DWI along a certain local direction, which helps to track the contours in our Inherited Edge Detection method (IED). Figure 3 shows the Directional kernels.

2.3 Inherited Edge Detection Method

Ignoring the edge correlation and contour continuity, most of the gradient-based edge detection methods detect edges individually. However, this paper takes full consideration of them by proposing the Inherited Edge Detection method (IED). Edge correlation and contour continuity is the fact that each edge in a contour is connected to its last edge and next edge along the contour. We find that edges in the same contour are most likely to distribute in a line segment along the contour curve, especially in a very small region. While tracking a contour, the position of the next edge is related to the positions of the current edge and the previous edge. This helps us to find the points with the higher probability which named candidate points in this paper, rather than searching 8-neighbor points or 4-neighbor points. After that, the remaining work is to find the real edge point among the candidate points.

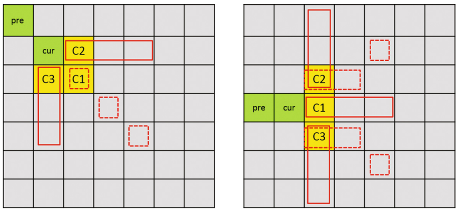

Fig. 4. Illustration of the inherited detection method (IED). The previous edge and current edge form a small line (the green points), which leads to the candidate points (the yellow points). Each kernel consists of three pixels and each candidate point uses one kernel to compute the DWI. For horizontal and vertical line, apply one more kernel to candidate point C_2 and C_3. (Color figure online)

$$P_{next} = \arg\max(DWI(C_1), DWI(C_2), DWI(C_3)) \quad (1)$$

The Fig. 4 shows how the IED works. First of all, the starting point is set to be the first edge of the contour, also be the previous edge at this moment. Then it selects the second edge in 8-neighborhood to be the current edge at this moment. Next it will choose edge candidates based on the positions of the previous edge and the current edge. Focusing on a 3×3 region, the point, lying in the extended line formed by the previous edge and the current edge, is the first candidate point C_1. Then the two closest points to C_1 are the other two candidates C_2 and C_3. Afterward, the IED computes the DWI of each edge candidate and the next edge P_{next} which is defined in Eq. (1) should be the one with the highest DWI. Finally, the previous edge and current edge is moved forward to find new next edge.

Actually, this inherited method is not limited to a 3×3 kernel. We also tried other sizes such as 5×5, 7×7. However, the size of 3×3 always performs better. In practice, the GCT method uses the first edge and the second edge twice in a contour. For the latter, it exchanges the roles of them and start new tracking along the opposite direction of the contour.

2.4 Local Threshold

Canny method sets a high threshold and a low threshold to filter out the edges. Points connected to the determined edge with the intensity higher than the low threshold will be chosen to be the edge. The defect of this method is that the thresholds, especially the low threshold are difficult to set in different applications. Too low a threshold can lead to too much noise or other useless edges, and too high a threshold can make the contours incomplete. The essence of the problem is that the thresholds of *Canny* method are global thresholds, not local thresholds.

In this paper, for the edge detection, our tracking based method is natural equipped with the advantage of local thresholds. Firstly, while detecting, it focuses on a local region and chooses the point with highest DWI. This strategy shows a similar effect to non-maximum suppression but simpler. Secondly, it searches the starting point for every contour, and once a contour is chosen, the edges in this contour will be detected. For example, it is assumed that P and Q are two points where P is located in a detected contour while Q is not. In this situation, even the intensity of Q is higher than P, Q will not be detected as an edge. The advantage of this strategy is obviously that only the edges in strong contours will be detected, meanwhile, the very week contours and others like small spots in the image will be discarded. Deeply, for the detected contours, they tend to be more complete than the result using other detectors like *Canny*. Experiments in Sect. 3 show the advantage of this strategy.

2.5 Ending Conditions and Coding the Contour

The GCT method needs an ending threshold T_e which is always much lower than the starting point, even lower than the low threshold of *Canny* in the same situation. The first ending condition is set by T_e. The tracking will stop when:

- the intensities of the candidates are all lower than the ending threshold T_e;
- it reaches the boundary of the image;
- it hits the points that have been marked as the edges.

Each contour will be coded with a unique number from the very beginning. For the third ending condition, the *GCT* records the number of the hit contour. This record table will help to merge the contours and compute some statistics to get the feature of the contours, such as the length of the contour or the average intensity of the contour. It's useful if the user wants to do any subsequent work based on edge detection or contours detection.

3 Experiment

The experiment consists of two parts. It first compares the detection results of our *GCT* method to one classical method *Canny* [2] and two learning-based methods *SE* [6], *RCF* [13]. Facing the fact in this field that there is not a standard and uniform evaluation method to compare different edge detection methods, especially for industrial application, we tried to make the comparative experiment in this paper more comprehensive. The remaining part of this section will show the experimental results when adjusting and improving the *GCT* method with smooth filters, thresholds and kernels. Finally, we provide a typical scheme of *GCT* method for general application.

3.1 Comparison Among Different Edge Detection Methods

As *Canny* is the most typical gradient-based edge detection method and our *GCT* method is based on the brightness gradient as well, we compare these two methods in same situation. The starting and ending thresholds of our *GCT* method are set to be equal to the high and low thresholds of *Canny* respectively. Meanwhile, both of these two methods use *Gaussian* filter with the same kernel size 3×3. Another two edge detection methods *SE* and *RCF* are learning-based methods and *RCF* achieved state-of-the-art performance on the BSDS500 benchmark. However, in industrial scene they fail to perform as well as in natural scene. The original results of these learning-based methods are always coarse. Here, we add extra non-maximum suppression to their results to thin the contours before comparison. Note that the results of our *GCT* method are original detection results without any post process. Deeply, our *GCT* result is already merged into contours, but others are still independent edge points.

The test images in our experiments vary from simple structures to complex. As shown in Fig. 5, our *GCT* method detects much more clean contours of the objects than *Canny* and outperforms the *SE* and *RCF* in most of the contours. To compare the details, we choose some regions of interest and enlarge them to watch the contours in pixel level. Notice that in the realization of *GCT* method, it is set to ignore the outermost 5 pixels of the source image which leads to losing some edges at the outer boundary. A better result can be achieved by using other boundary strategy such as adding another several columns and rows before detecting. Even though, the results show that the detected contours of our *GCT* method are more complete than others in most of the regions.

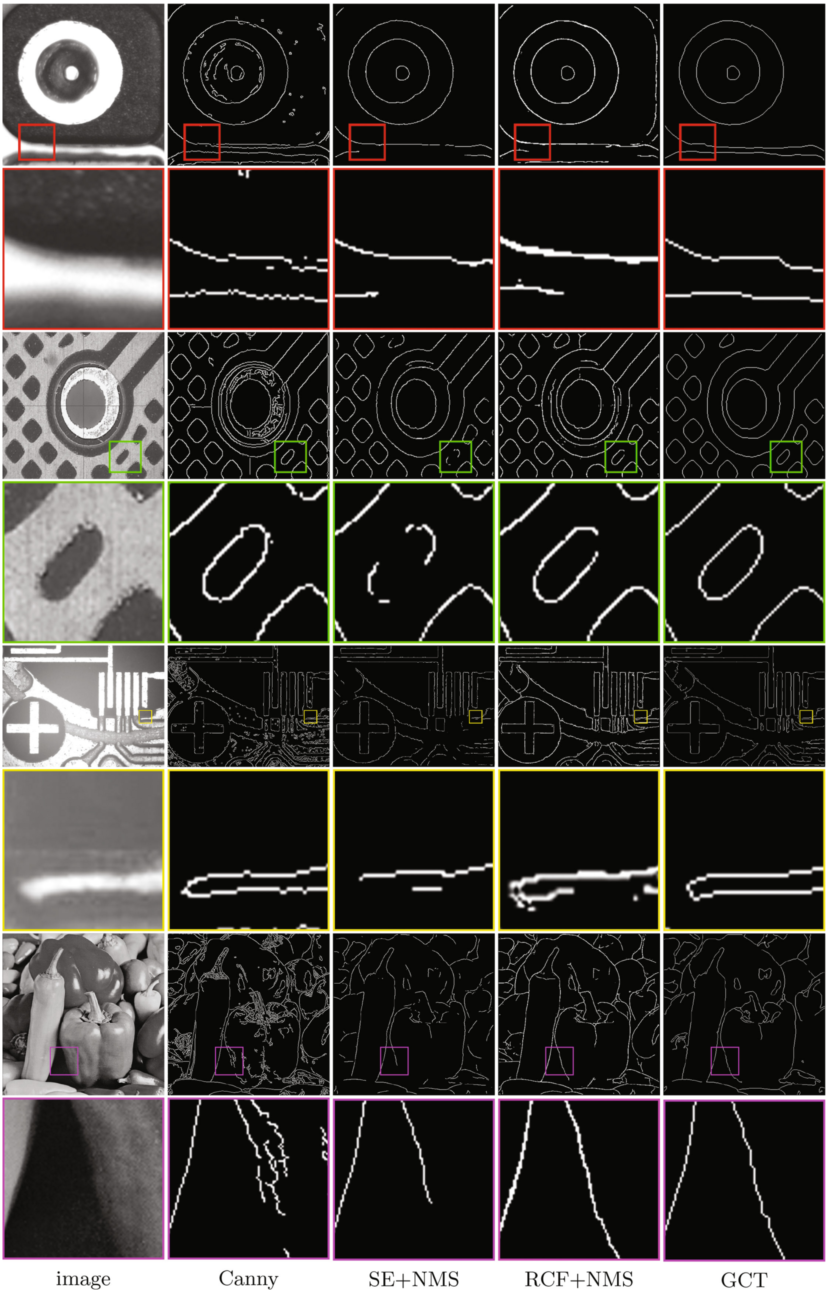

Fig. 5. Detection results of *Canny* [2], *SE* [6], *RCF* [13] and our *GCT* method in Industrial test images. *SE* and *RCF* are learning-based methods with coarse original results. We add extra non-maximum suppression (*NMS*) to *SE* and *RCF* to thin the contours, even though, our *GCT* contours are more clean with less clutters and more complete for the detected contours.

3.2 Different Scheme of GCT

In practice, we tried different schemes to meet different needs. In the comparison experiment with other detectors, the *Gaussian* filter is used to smooth the test images. However, there are some other choices, such as mean filter and bilateral filter. Figure 6 shows part of the detection results of *GCT* method with different smooth filters. The experiments show that all these three filters can help to detect the contours of the objects with a clean surroundings (the column 2 to column 4 in Fig. 6). In some situations, bilateral filter leads to less clutters, however, sometimes leads to the incompleteness of the edges. Bilateral filter takes much more running time which is more than three times as much as *Gaussian* filter takes under the experimental environment. The performance of *Gaussian* filter and mean filter with the same kernel size is almost the same. While using these two filters, the kernel size may affect the detection results. Which to choose is depends on the application scene and the size of the image. One useful suggestion is to use filters with larger size for larger images.

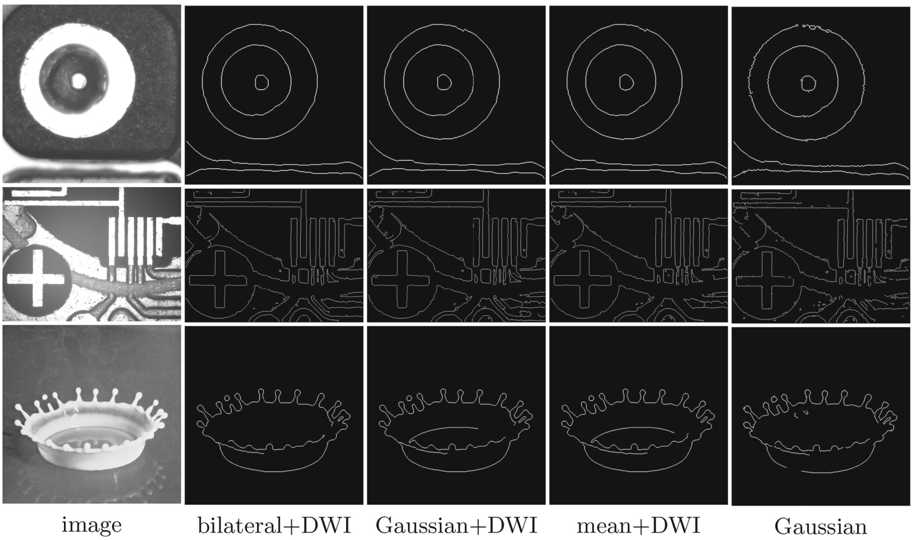

Fig. 6. In some situations, bilateral filter leads to less clutters, however, sometimes leads to the incompleteness of the edges. Using directional kernels to redefine the intensity of an edge (DWI) makes the contours more complete with less clutters.

While detecting the "next edge" in IED, we use the DWI to redefine the intensity of an edge. The last column in Fig. 6 presents the detecting results with the traditional strategy that only uses a single point value. On the contrary, other columns are the results with DWI. Experiments show the advantage that the results using DWI could better tracking the gradient center leading to more complete contours.

Ending threshold and starting threshold also affect the detection. One of the starting threshold and ending threshold is set to be constant, meanwhile the other one is adjusted to detect edges and compare the results of all images. The experiments show that the detection results are insensitive to the ending threshold. In practice, users can easily set the ending threshold as 20–40. On the contract, the detection result is sensitive to the starting threshold. Users can get detection results with different level by setting the starting threshold.

As a conclusion of this part, we could give a typical scheme of our *GCT* method:

– *Gaussian* filter with the size of 3×3 to smooth the given image
– *Sobel* detector with the kernel size of 3×3 to get the gradient image
– For a gray-scale image, the ending threshold could be 20–40, and the starting threshold could be 80–120
– Apply the inherited edge detection method with the kernel size of 3×3.

4 Conclusions

In this paper, we propose a novel pixel level edge detection method, *GCT*, by tracking the center curve of the edge band in the gradient image. This method is also a contour detection method for its detection result is presented by contours. We describe the edge correlation and contour continuity, and then put forward to the edge detection process, which is stated as the Inherited Edge Detection method (*IED*) in this paper. This paper also redefines the intensity of the edge by Directional Weighted Intensity (*DWI*), which helps to complete the contours. Comparing to the classical *Canny* method, our *GCT* method focus on the main structure of the object and achieves a much cleaner detection result without redundant clutters. Meanwhile, the detected contours are continuous and complete. Furthermore, our *GCT* method outperforms several learning-based methods in industrial scene. A typical scheme to apply the *GCT* method is also provided.

References

1. Arbeláez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2011)
2. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **6**, 679–698 (1986)
3. Chen, L.C., Barron, J.T., Papandreou, G., Murphy, K., Yuille, A.L.: Semantic image segmentation with task-specific edge detection using CNNS and a discriminatively trained domain transform. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 4545–4554 (2016)
4. Cheng, M.-M., et al.: HFS: hierarchical feature selection for efficient image segmentation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 867–882. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_53

5. Dollar, P., Tu, Z., Belongie, S.: Supervised learning of edges and object boundaries. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1964–1971 (2006)
6. Dollar, P., Zitnick, C.L.: Fast edge detection using structured forests. IEEE Trans. Pattern Anal. Mach. Intell. **37**(8), 1558–1570 (2015)
7. Ferrari, V., Fevrier, L., Jurie, F., Schmid, C.: Groups of adjacent contour segments for object detection. IEEE Trans. Pattern Anal. Mach. Intell. **30**(1), 36 (2008)
8. Freeman, W.T., Adelson, E.H.: The design and use of steerable filters. IEEE Trans. Pattern Anal. Mach. Intell. **13**(9), 891–906 (1991)
9. Kittler, J.: On the accuracy of the sobel edge detector. Image Vis. Comput. **1**(1), 37–42 (1983)
10. Konishi, S., Yuille, A.L., Coughlan, J.M., Zhu, S.C.: Statistical edge detection: learning and evaluating edge cues. IEEE Trans. Pattern Anal. Mach. Intell. **25**(1), 57–74 (2003)
11. Lindeberg, T.: Edge detection and ridge detection with automatic scale selection. Int. J. Comput. Vis. **30**(2), 117–156 (1998)
12. Liu, H., Jezek, K.C.: Automated extraction of coastline from satellite imagery by integrating canny edge detection and locally adaptive thresholding methods. Int. J. Remote. Sens. **25**(5), 937–958 (2004)
13. Liu, Y., Cheng, M.M., Hu, X., Wang, K., Bai, X.: Richer convolutional features for edge detection, pp. 5872–5881 (2016)
14. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. In: International Conference on Neural Information Processing Systems, pp. 1279–1286 (2002)
15. Mehrotra, R., Zhan, S.: A computational approach to zero-crossing-based two-dimensional edge detection. Graph. Model. Image Process. **58**(1), 1–17 (1996)
16. Moore, D.J.: Fast hysteresis thresholding in Canny edge detection (2011)
17. Pont-Tuset, J., Barron, J., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 328–335 (2014)
18. Prewitt, J.M.S.: Object enhancement and extraction. Pict. Process. Psychopictorics **10**(1), 15–19 (1970)
19. Ren, X.: Multi-scale improves boundary detection in natural images. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5304, pp. 533–545. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88690-7_40
20. Roberts, L.G.: Machine perception of three-dimensional solids **20**, 31–39 (1963)
21. Suzuki, S., Be, K.: Topological structural analysis of digitized binary images by border following. Comput. Vis. Graph. Image Process. **30**(1), 32–46 (1985)
22. Tai, S.C., Yang, S.M.: A fast method for image noise estimation using Laplacian operator and adaptive edge detection. In: International Symposium on Communications, Control and Signal Processing, pp. 1077–1081 (2008)
23. Ullman, S., Basri, R.: Recognition by linear combinations of models. IEEE Trans. Pattern Anal. Mach. Intell. **13**(10), 992–1006 (1991)
24. Wang, R.: Edge detection using convolutional neural network. In: Cheng, L., Liu, Q., Ronzhin, A. (eds.) ISNN 2016. LNCS, vol. 9719, pp. 12–20. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40663-3_2
25. Wang, Z., Li, Q., Zhong, S., He, S.: Fast adaptive threshold for the canny edge detector. In: Proceedings of SPIE - The International Society for Optical Engineering, vol. 6044, pp. 501–508 (2005)
26. Wei, Y., et al.: STC: a simple to complex framework for weakly-supervised semantic segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2314–2320 (2017)