Literature review on edge detection methods

Yujeong Lee
elee17@earlham.edu
Earlham College
Richmond, Indiana

KEYWORDS

Edge detection, Sobel, Canny, convolution filter, Gaussian smoothing filter, thresholding, hysteresis

.

1 INTRODUCTION

Edge detection is a common method used in computer vision to identify boundaries of objects in a given image. A pixel is identified as an edge point when there is rapid intensity change in the image, such as when an object surface discontinues or there is change in brightness. A line is defined as a set of edges located contiguously in any orientation. An optimal edge detector will identify only the true edges while discarding irrelevant details and noise in the image. The terminology 'true edge' refers to a real edge in the original image.

Edge detectors can generally be classified into autonomous detectors that functions without external results, and contextual detectors that are dependent on contextual information. [8].

Some early edge detectors include the Sobel operator [3], the zero-crossing [5], and the Canny edge detector [1]. The process of edge detection generally consists of smoothing, differentiation, and labeling, where the order of smoothing and differentiation steps can be reversed [8]. Differentiation methods can generally be classified into gradient based methods or the Laplacian based methods [8].

Figure 1: The color difference on a panda's fur displays a form of rapid intensity change, which is used to identify edges. The collage images are screen captured from 0:31 and 0:33 of the video. (https://youtu.be/uihBwtPIBxM)

CS388 Methods for Research and Dissemination, , Earlham College

2 PREPARING THE IMAGES

2.1 Gray level conversion

The very first step in any of the common edge detection methods is to convert the input image into grey scale, a "black and white" image, where the grey level ranges from 0 to 255.

Figure 2: An image like the one on the left is converted into grey scale like the one on the right. (https://youtu.be/uihBwtPIBxM)

2.2 Smoothing

Smoothing of the input image is necessary because edge detection is sensitive to noise. Smoothing filters reduce noise and details of the input image to yield more relevant results for edge detection. Smoothing filters take the grey level image as an input, and they output a visually blurred image. Many edge detectors, including Sobel, zero-crossing, and Canny, smooth the input image before or after its operation.

2.2.1 Regularization theory. One significant challenge in scaling the smoothing filter is setting the parameter that does not negatively impact the edge detection performance. An ideal smoothing filter and its parameters reduce as much noise as possible in the image while retaining as much information as possible. In combating this challenge, Ziou et al. [8] describes regularization theory, which finds the optimal smoothing filter by calculating the optimal compromise between noise reduction and information loss. Some examples of proposed regularization filters are convolution of the image with its derivatives, Green function, and the Gaussian [8]. I will review the Gaussian smoothing filter, a common convolution filter used in the smoothing process.

2.2.2 Gaussian smoothing operator. An example of a simple filter is the mean filter, where each pixel is replaced by the mean value of its neighbors and itself [6]. A 3x3 kernel will examine 1 pixel and its 8 neighboring pixels, while a 5x5 kernel will examine the current pixel and 24 surrounding neighbors. In edge detection, it is common to use the Gaussian smoothing filter instead, which is more complex than the mean filter. The Gaussian smoothing filter is similar to the mean filter in that each pixel and its neighbors are examined under a kernel, and the pixel is replaced by a new value. However, instead of a mean value, the Gaussian takes a weighted average, where the central pixel values have more weight than the neighbors [6]. The output of such smoothing process is a visually blurred image.

3 EDGE DETECTOR METHODS

3.1 Sobel Operator

Maini et al. [4] describes the Sobel operator. The Sobel operator takes a grey leveled and optionally smoothed image as its input. The Sobel operator uses a gradient based method. Sobel uses a pair of 3x3 convolution kernels that are 90° to each other, and each kernel filters the image in vertical and horizontal directions. The Gx kernel responds to edges running vertically, and the Gy kernel to edges running horizontally. The kernels' two separate measurements of the gradient are combined to produce the absolute magnitude and the orientation of the gradient [3].

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

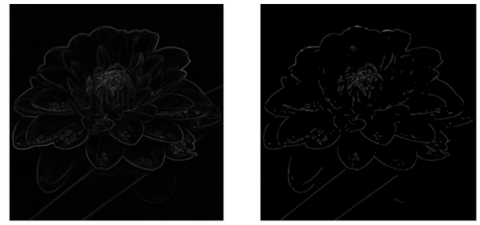
Gy

Figure 3: A pair of convolution kernels used by Sobel edge detector. One kernel is the other rotated by 90°. (https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm)

Sobel x direction Sobel y direction

Figure 4: The two separate gradient output from the pair of convolution kernels, which will be further combined to produce the absolute magnitude and the orientation. (https://youtu.be/uihBwtPIBxM)

Because the Sobel operator is sensitive to noise, users of the operator can filter the image using a threshold. As an example, when the threshold is set as 100 from the grey level range of 0 - 255, any identified edge points below the value 100 will be discarded. The output image will only include edges above the 100 value threshold.



Sobel output Canny output

Figure 5: The output of Canny displays reduced noise and more distinct edge lines compared to that of Sobel. Please zoom into the image for better comparison. (https://youtu.be/sRFM5IEqR2w)

3.2 Canny Edge Detector

John Canny draws from several existing edge detection techniques to present an improved multi-stage algorithm. He develops the three criteria for an optimal edge detector: 1) "good detection" with low error rate to maximize signal-to-noise ratio, 2) "good localization" where identified edge points near, if not at the center or, the true edge, and 3) "only one response to single edge" [1].

To perform Canny edge detection, the input image is first smoothed to reduce noise using the Gaussian smoothing operator. The smoothed image is processed by a "2-dimensional first derivative operator" [1], such as the Sobel operator or the Roberts Cross operator, which produces gradient direction and edge orientation representing intensity of each pixel. Because the Sobel operator is sensitive to noise [3], common implementations of the Canny edge detector, such as that of OpenCV [6], directly takes the output of the Sobel operator as an input for the next step.

The edges identified by the gradient magnitude in Sobel are not uniform in thickness. Canny employs the "non-maximum suppression" method to essentially reduce the edge lines to one pixel wide [1]. Non-maximum suppression identifies the "local maximum in its neighborhood in the direction of gradient" and suppresses any other detected edges in the neighborhood that is not the local maximum [6]. Canny draws similarity in his method of identifying the local maximum with Marr-Hildreth edge detector's "zero-crossings in the Laplacian operator" [5]. However, he contrasts the two methods by stating that "in two dimensions, … the directional properties of [Canny's] detector enhance its detection and localization performance" [1].

Once the local maxima have been identified, Canny employs two-level threshold method, which is referred to as hysteresis [1]. As mentioned above, using one threshold to discard pixels under a certain value has been in use. However, the resulting images are still troubled by noise and details that have not been discarded.

In facing this challenge, Canny [1] sets a high threshold and a low threshold. When an edge point's pixel is higher than the high threshold, it is confirmed by the algorithm as an edge. When it is below the low threshold, the edge point is discarded. When an edge point's pixel value is between the high and low thresholds, there are two rules in deciding whether or not to discard the edge point. When an edge point's pixel value is between the high and low thresholds, the algorithm first looks for verified edge points in the neighborhood. 1) If it is connected to a verified line of edge points, the contested edge point is classified as an edge. 2) If the edge points are not connected to any verified edge points/lines, then it is discarded. This process of hysteresis and is displayed to be a successful method according to Maini et al. [4].

3.3 Recent Developments

3.3.1 Holistically-Nested Edge Detection. Xie et al. [7] presents a Holistically-Nested Edge Detection (HED). The challenges the authors try to improve are "holistic image training and prediction" and "multi-scale and multi-level feature learning" [7]. The algorithm is built upon fully convolutional neural networks and deeply-supervised nets. The method was applied to the BSD500 dataset and the NYU Depth dataset, and it is evaluated through measuring accuracy using the F-score and its time complexity. The authors present enhanced accuracy and speed compared to traditional methods.

3.3.2 Edge detection using structured forests. Dollár et al. [2] presents a faster edge detection method using structured forests, a method described by Xie et al. [7] as the best structured learning edge detection. The method is achieved by "learning structured random decision forest that robustly uses structured labels to select splits in the trees". To prevent overfitting, the method takes several parameters, including structured forest splitting parameters, image and channel blurring, and varying number of trees and data quantity [2]. After edge detection, the authors further employ sharpening and multi-scale detection to improve the results. This new method was applied to the Berkeley Segmentation Dataset and the Benchmark dataset, and the accuracy was measured by the fixed contour threshold (ODS), the per-image best threshold (OIS), and the average precision (AP). The results present enhanced accuracy and time compared to traditional methods.

4 CONCLUSION

I provided an overview of preparing the image and running them through the Sobel operator and the Canny edge detector. I also provided a brief description of two recent methods, the Holistically-Nested Edge Detection and edge detection with structured forests.

REFERENCES

[1] John Canny. 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), 679–698.

[2] Piotr Dollár and C Lawrence Zitnick. 2014. Fast edge detection using structured forests. *IEEE transactions on pattern analysis and machine intelligence* 37, 8 (2014), 1558–1570.

[3] Robert Fisher, Simon Perkins, Ashley Walker, and Erik Wolfart. 2000. Sobel Edge Detector. Retrieved September 8, 2020 from https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm

[4] Raman Maini and Himanshu Aggarwal. 2009. Study and comparison of various image edge detection techniques. *International journal of image processing (IJIP)* 3, 1 (2009), 1–11.

[5] David Marr and Ellen Hildreth. 1980. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207, 1167 (1980), 187–217.

[6] OpenCV. [n.d.]. Canny Edge Detection. Retrieved September 8, 2020 from https://docs.opencv.org/trunk/da/d22/tutorial_py_canny.html

[7] Saining Xie and Zhuowen Tu. 2015. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*. 1395–1403.

[8] Djemel Ziou, Salvatore Tabbone, et al. 1998. Edge detection techniques-an overview. *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii* 8 (1998), 537–559.