Sobel Edge Detection Image Using MATLAB

Ahmed Mustafa Taha Alzbier [a], Hang Cheng [b]

Changchun University of Science and Technology, Jilin, China

[a]AMT4047@gmail.com, [b]22804414@qq.com

Abstract

This work is intended to implement the edge detection for digital image, so that it may be carried out to a big contour identification of an image. Edge detection is one of the most commonly used operations in image processing and pattern recognition, the reason for this is that edges form the outline of an object. An edge is the boundary between an object and the background, and indicates the boundary between overlapping objects. This means that if the edges in an image can be identified accurately, all of the objects can be located and basic properties such as area, perimeter, and shape can be measured. Since computer vision involves the identification and classification of objects in an image, edge detection is an essential tool. Efficient and accurate edge detection will lead to increase the performance of subsequent image processing techniques, including image segmentation, object-based image coding, and image retrieval. The Sobel edge detector uses a pair of 3 x 3 convolution masks, one estimating gradient in the x-direction and the other estimating gradient in y–direction, and in this paper we will discuss sobel edge detection

Keywords

Edge, Edge Detection, Sobel Operator, Image Processing, Threshold.

1. Introduction

Edge detector is one of the most important tools in computer vision. The edge detection process serves to simplify the analysis of images by drastically reducing the amount of data to be processed, while at the same time preserving useful structural information about object boundaries [1]. Edge detection in color image is far more challenging task than gray scale images as color space is considered as a vector space. Almost 90% of edge information in a color image can be found in the corresponding grayscale image .However, the remaining 10% can still be vital in certain computer vision tasks [2]. The problem is that in general edge detectors behave very poorly. While their behavior may fall within tolerances in specific situations, in general edge detectors have difficulty adapting to different situations. The quality of edge detection is highly dependent on lighting conditions, the presence of objects of similar intensities, density of edges in the scene, and noise. Since different edge detectors work better under different conditions, it would be ideal to have an algorithm that makes use of multiple edge detectors, applying each one when the scene conditions are most ideal for its method of detection. In order to create this system, you must first know which edge detectors perform better under which conditions [3].

The basic idea of edge detection is as follows: First, use edge enhancement operator to highlight the local edge of the image. Then, define the pixel "edge strength" and set the threshold to extract the edge point set. However, because of the noise and the blurring image, the edge detected may not be continuous. So, edge detection includes two contents. First is using edge operator to extract the edge point set. Second is removing some of the edge points from the edge point set, filling it with some another and linking the obtained edge point set into lines [4].

The four steps of edge detection are:

Smoothing: suppress as much noise as possible, without destroying the true edges.

Enhancement: apply a filter to enhance the quality of the edges in the image (sharpening).

Detection: determine which edge pixels should be discarded as noise and which should be retained (usually, thresholding provides the criterion used for detection).

Localization: determine the exact location of an edge (sub-pixel resolution might be required for some applications, that is, estimate the location of an edge to better than the spacing between pixels). Edge thinning and linking are usually required in this step [5].

2. Edge detection techniques

There are many ways to perform edge detection. However, the majority of different methods may be grouped into two categories:

Gradient: The gradient method detects the edges by looking for the maximum and minimum in the first derivative of the image.

Laplacian: The Laplacian method searches for zero crossings in the second derivative of the image to find edges. An edge has the 1D shape of a ramp and calculating the derivative of the image can highlight its location. Suppose we have the following signal, with an edge shown by the jump in intensity Fig. 1 below [12], [13]:

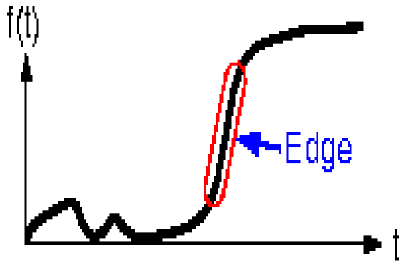


Fig. 1 Intensity graph of a signal

If we take the gradient of this signal (which, in one dimension, is just the first derivative with respect to time t) we get Fig. 2 [12], [14]. Clearly, the derivative shows a maximum located at the centre of the edge in the original signal. This method of locating an edge is characteristic of the “gradient filter” family of edge detection filters and includes the Sobel method. A pixel location is declared an edge location if the value of the gradient exceeds some threshold. As mentioned before, edges will have higher pixel intensity values than those surrounding it. So once a threshold is set, you can compare the gradient value to the threshold value and detect an edge whenever the threshold is exceeded.

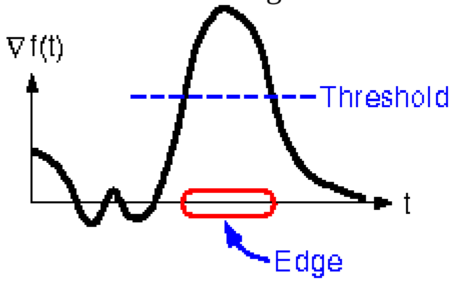


Fig. 2 First derivative of the signal [2], [5]

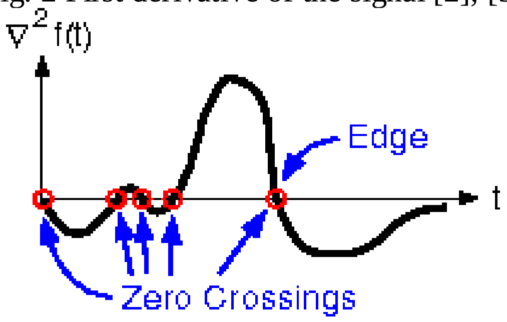


Fig. 3 Second derivative of the signal [7]

Furthermore, when the first derivative is at a maximum, the second derivative is zero. As a result, another alternative to finding the location of an edge is to locate the zeros in the second derivative. This method is known as the Laplacian and the second derivative of the signal is shown in Fig. 3 [15]:

The process of edge detection attenuates high fluctuations in color, i.e. dramatic change in intensity [16],[17]. Matlab includes the built-in function edge designed for edge detection. It supports the following types of edge detectors: Sobel, Prewitt, Roberts, Log (Laplacian), Canny and Zero cross.

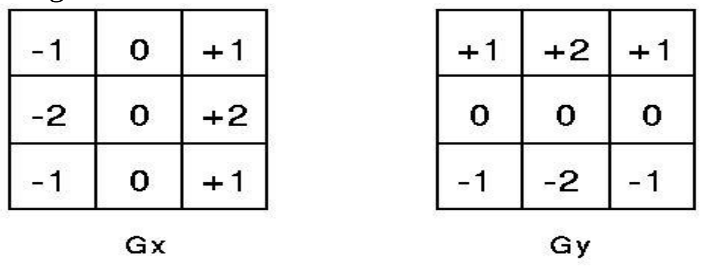
3. Sobel edge detection

Edge detection is the process of localizing pixel intensity transitions. The edge detection has been used by object recognition, target tracking, segmentation, and etc. Therefore, the edge detection is one of the most important parts of image processing. There mainly exist several edge detection methods: Sobel, Prewitt, Roberts and Canny. In this paper, Sobel which is an edge detection method is considered. The Sobel edge detector uses two masks, one vertical and one horizontal. These masks are generally used 3×3 matrices. Especially, the matrices which have 3×3 dimensions are used in matlab. Sobel has two main advantages: it has some smoothing effect to the random noise of the image [6]:

Since the introduction of the average factor, it has some smoothing effect to the random noise of the image.

Because it is the differential of two rows or two columns, so the element of the edge on both sides has been enhanced, so that the edge seems thick and bright

In the airspace, edge detection is usually carried out by using the local operator. What we usually use are orthogonal gradient operator, directional differential operator and some other operators relevant to 2nd order differential operator. Sobel operator is a kind of orthogonal gradient operator. Gradient corresponds to first derivative, and gradient operator is a derivative operator. Here, the image is convolved with only two kernels, one estimating the gradient in the x direction GX .the other the gradient in the y-direction Gy [8], a convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time. The actual Sobel masks are shown below Fig.4[7]:



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

Gx

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

Gy

Fig. 4 The actual Sobel masks

The magnitude **of the gradient is then calculated using the formula [8] [9]:**

$$|G| = \sqrt{Gx2+ Gy2} \qquad (1)$$

and is often approximated with [10]:

$$|G| = |Gx| + |Gy| \qquad (2)$$

which is much faster to compute.

The angle of orientation of the edge (relative to the pixel grid) giving rise to the spatial gradient is given by:

$$\theta = \arctan(Gy/Gx) \qquad (3)$$

In this case, orientation 0 is taken to mean that the direction of maximum contrast from black to white runs from left to right on the image, and other angles are measured anti-clockwise from this.

Often, this absolute magnitude is the only output the user sees ,the two components of the gradient are conveniently computed and added in a single pass over the input image using the pseudo-convolution operator shown in Fig. 2.

P1	P2	P3
P4	P5	P6
P7	P8	P9

Fig. 5 Pseudo-convolution masks used to quickly compute approximate gradient magnitude

Using this mask the approximate magnitude is given by:

$$|G| = |(P_1 + 2 \times P_2 + P_3) - (P_7 + 2 \times P_8 + P_9)| + |(P_3 + 2 \times P_6 + P_9) - (P_1 + 2 \times P_4 + P_7)| \quad (4)$$

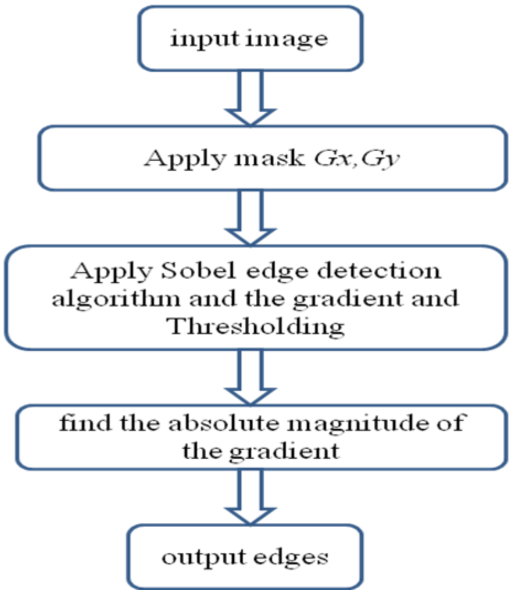

input image
Apply mask Gx,Gy
Apply Sobel edge detection algorithm and the gradient and Thresholding
find the absolute magnitude of the gradient
output edges

Fig. 6 Sobel edge detection method

4. Thresholding

In many vision applications, it is useful to be able to separate out the regions of the image corresponding to objects in which we are interested, from the regions of the image that correspond to background. Thresholding often provides an easy and convenient way to perform this segmentation on the basis of the different intensities or colors in the foreground and background regions of an image. In addition, it is often useful to be able to see what areas of an image consist of pixels whose values lie within a specified range, or band of intensities (or colors). Thresholding can be used for this as well.

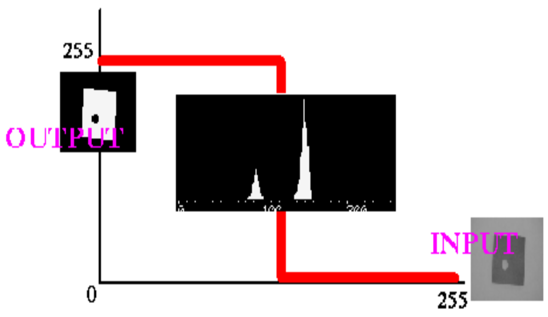
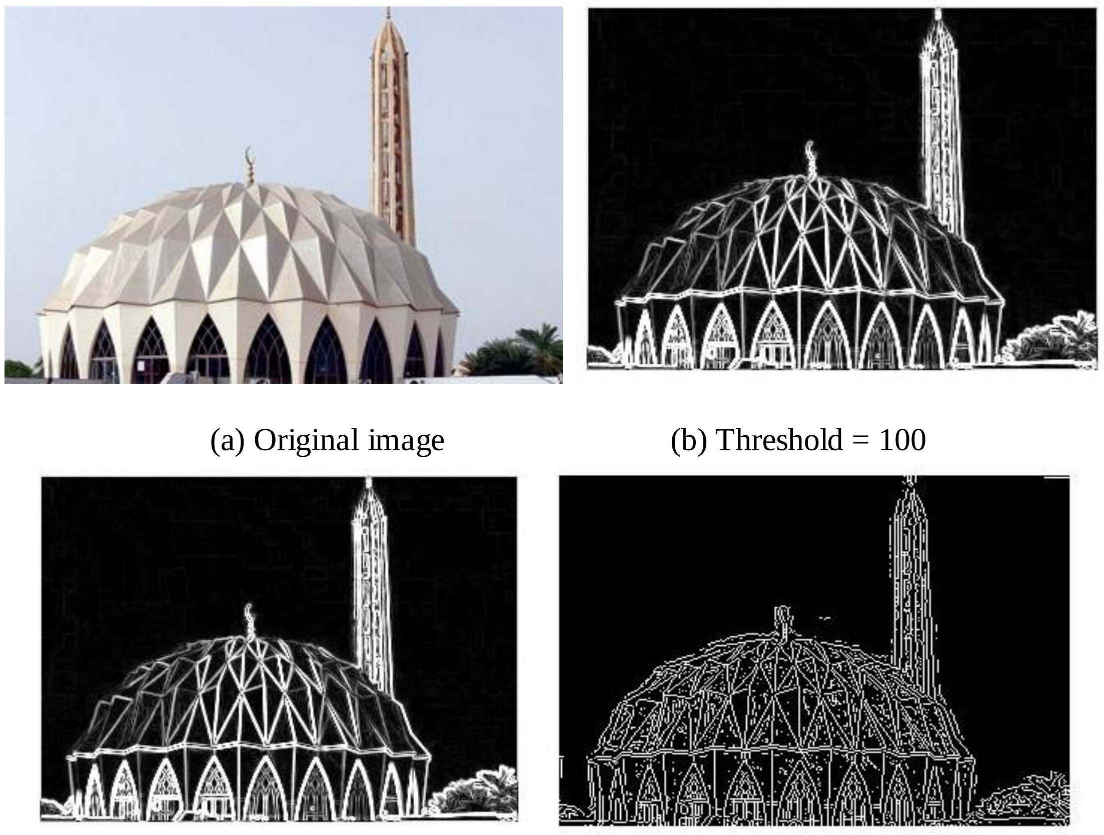


Fig. 7 Threshold, Density slicing

The input to a thresholding operation is typically a grayscale or color image. In the simplest implementation, the output is a binary image representing the segmentation. Black pixels correspond to background and white pixels correspond to foreground (or vice versa). In simple implementations, the segmentation is determined by a single parameter known as the intensity threshold. In a single pass, each pixel in the image is compared with this threshold. If the pixel's intensity is higher than the threshold, the pixel is set to say white in the output. If it is less than the threshold, it is set to black.

In more sophisticated implementations, multiple thresholds can be specified, so that a band of intensity values can be set to white while everything else is set to black. For color or multi-spectral images, it may be possible to set different thresholds for each color channel, and so select just those pixels within a specified cuboid in RGB space. Another common variant is to set to black all those pixels corresponding to background, but leave foreground pixels at their original color/intensity (as opposed to forcing them to white), so that that information is not lost [11].



(a) Original image (b) Threshold = 100

(c) Threshold = 150 (d) Edge detected Image

Fig. 8 (a) Original image (b, c) image after used soble gradient (d) Edge detected Image

5. Results and Discussion

The Sobel operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical direction and is therefore relatively inexpensive in terms of computations. On the other hand, the gradient approximation which it produces is relatively crude, in particular for high frequency variations in the image.

Edge detection helps in optimizing network bandwidth and it is needed to keep track of data flowing in and out of the network. It helps to extract useful features for pattern recognition. Although the Sobel operator is slower to compute, it’s larger convolution kernel smoothes the input image to a greater extent and so makes the operator less sensitive to noise. The larger the width of the mask, the lower its sensitivity to noise and the operator also produces considerably higher output values for similar edges. Sobel operator effectively highlights noise found in real world pictures as edges though, the detected edges could be thick and Sobel operator is highly recommended in massive data communication found in image data transfer.

6. Conclusion

The Sobel operator performs spatial gradient measurement on an image. Typically it is used to find the approximate absolute gradient magnitude at each point I of an input grayscale image. The Sobel edge detector uses a pair of 3 x3 convolution masks, one estimating gradient in the xdirection and the other estimating gradient in y–direction. It is easy to implement than the other operators. Transferring pixel array into statistically uncorrelated data set enhances the removal of redundant data, as a result, reduction of the amount of data required to represent a digital image. Considering data communication especially the internet, massive data transfer causes serious problems for interactive network users.

Edge detection helps in optimizing network bandwidth and it is needed to keep track of data flowing in and out of the network. It helps to extract useful features for pattern recognition. Although the Sobel operator is slower to compute, it's larger convolution kernel smoothes the input image to a greater extent and so makes the operator less sensitive to noise. The larger the width of the mask, the lower its sensitivity to noise and the operator also produces considerably higher output values for similar edges.

Threshold In many vision applications, it is useful to be able to separate out the regions of the image corresponding to objects in which we are interested, from the regions of the image that correspond to background.

Acknowledgments

Fund Project: Topics of major projects National Technology Support Program (2012BAF 12B22), Key scientific and technological projects of Jilin Province (20140204050GX), Major scientific and technological projects of Jilin Province (2012ZDGG004).

References

[1] J. F. Canny, .A computational approach to edge detection,. IEEE Trans. Pattern Anal. Machine Intell, vol. 8, no. 6, pp. 679.698,Nov. 1986.

[2] C. L. Novak and S. A. Shafer, “Color edge detection,” in Proc.DARPA Image Understanding Workshop, 1987, pp. 35–37.

[3] N. Ehsan, S. Sara and H. Hamid, "Edge Detection Techniques: Evaluations and Comparisons", Applied Mathematical Sciences, vol. 2, no. 31, pp. 1507 – 1520, 2008.

[4] G. Wenshuo, Y. Lei, Z. Xiaoguangand L. Huizhong, "An Improved Sobel Edge Detection", IEEE, 2010.

[5] Trucco and Jain et al., "Edge detection", Chapter 4 and 5.

[6] A. Elif, "Sobel Edge Detection Method for Matlab", Assistant Professor Elif Aybar Is With Porsuk Vocational School, Anadolu University, Eskisehir. E-Mail: Elaybar@Anadolu.Edu.Tr. Fax: 0 222 224 1390.

[7] Bill Green, “ Edge Detection Tutorial”, 2002.

[8] T. A. Al-Aish, "Edge Detection in Sensor Networks using Image Processing", Diala Jour., vol. 31 , Iraq, 2008.

[9] S. A. Salem, N. V. Kalyankar and S. D. Khamitkar, "Image Segmentation By Using Edge Detection", (IJCSE) International Journal On Computer Science And Engineering, vol. 2, no. 3, pp. 804-807, 2010.

[10] R. Fisher, S. Perkins, A. Walker and E. Wolfart, "Edge Detectors", 2003.

[11] R. Fisher, S. Perkins, A. Walker and E. Wolfart, “Thresholding” , 2003.

[12] J. P. G. Abel, "Edge Detectors: Visual Computing and Multimedia", Departamento de Inform ática, Universidade da Beira Interior Portugal, 2011.

[13] C. John, "A Yariational Approach To Edge Detection", AAAI-83 Proceedings, 1983.

[14] A. D. Jepson and D. J. Fleet, "Edgel Detection", 2011.

[15] J. M. Elham J. M., "Segmentation of SAR Images Using Edge-Detector Method," Al-Mustansiriya Journal of Science, vol. 19, no. 7, pp. 21-31, 2008.

[16] B. Saketand M. Ajay, "A Survey on Various Edge Detector Techniques", Published by Elsevier Ltd.doi: 10.1016/j.protcy. 2012.05.033, 2212- 0173, 2012.

[17] K. B. Samir, "Edge Detection from CT Images of Lung", (IJESAT) International Journal of Engineering Science & Advanced Technology vol.2, Issue 1, pp. 34–37, ISSN: 2250–3676, 2012.