2009 International Conference on Advances in Recent Technologies in Communication and Computing

A Statistics and Local Homogeneity based Color Edge Detection Algorithm

Soumya Dutta
Department of Electronics and Communication
Netaji Subhash Engineering College
Kolkata - 700152, India
soumya.nsec@gmail.com

Bidyut B. Chaudhuri
Computer Vision and Pattern Recognition Unit
Indian Statistical Institute
Kolkata - 700108, India
bbc@isical.ac.in

Abstract— **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. A novel color edge detection algorithm is proposed in this paper. On the basis of standard deviation calculation of pixels the discontinuity among the pixels are detected. Then the image is segmented into a binary image with a fixed threshold where black pixels signify homogeneous region and white pixels signify edges. Finally, a thinning technique is applied to extract thin edges. The proposed method is applied over large database of color images both synthetic and real life images and performance of the algorithm is evident from the results and is comparable with other edge detection algorithms.**

Keywords-color edge detection; edge thinning; standard deviation.

I. 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 gray scale image. However, the remaining 10% can still be vital in certain computer vision tasks. [2]. Furthermore human perception of color image is more enriched than an achromatic picture [3]. Several color models are present such as RGB color model, YUV model, CMY color model, CMYK color model, HIS color model [4].

One of the earliest color edge detectors is proposed by Navatia. The image data is transformed to luminance and two chrominance components and Huckel's edge detector is used to find the edge map in each individual component independently, except for the constraint of having the same orientation [3]. Shiozaki proposed a method on the basis of entropy calculation [5]. In 1999 M. A. Ruzon and Carlo Tomasi proposed a method of detecting color edges with compass operator [6] and in 2001 with color distribution [9]. They considered a disc at each pixel location. The disc was divided into several pairs of opposite semi discs by rotating the diameter over 180^o with an interval of 15^o. The color distribution of the pixels in each such semi disc was found after doing vector quantization. The distance between two semi discs generated by a single diameter was the distance between their color distributions. The distance between two distributions was found using the Earth Mover's Distance (EMD). The edge magnitude at the pixel was the maximum distance among all the distances found between each pair of opposite semi discs created rotating the diameter. Cumani proposed an extension of second directional derivative approach to color images [7]. In this paper a method is discussed to detect edges in color images with a threshold process that automatically selects a value for a given picture and after that a thinning technique is applied to generate edge map.

This paper is organized in the following way. Section II depicts the steps of the proposed algorithm and mathematical models behind it. In section III results of the proposed method is shown with comparisons. The discussion ends with concluding remarks in section IV.

II. STEPS OF PROPOSED METHOD

In this paper, we present a novel method of edge detection. We have assumed color image as a data set and each pixel as a data point of that data set. As RGB color model represents high correlation within pixels, we have chosen RGB model for our analysis. Proposed method consists of four distinct steps and they are discussed below.

A. Smoothing by Median Filter

A traditional **median filter** is based upon moving a window over an image and computing the output pixel as the median of the gray values within the input window. If the window is J x K in size we can order the J*K pixels in gray level values from smallest to largest. If J*K is odd then the median will be the (J*K+1)/2 entry in the list of ordered gray values. Note that the value selected will be exactly equal to one of the existing gray values so that no round off error will be involved if we want to work exclusively with integer gray values. Median filters are quite popular because, for certain random types of noise, they provide excellent noise reduction capabilities, with considerably less blurring than linear smoothing filters of similar size which helps to preserve edges. In the proposed method we have used an **adaptive median filter**. One main reason for using adaptive median filter is that it seeks to preserve detail of the image while smoothing the non-impulse noise, something that the "traditional" median filter does not do [4].

The working of an adaptive median filter is divided into two levels and as follows:

```
Level A.  A1 = $Z_{med} - Z_{min}$
A2 = $Z_{med} - Z_{max}$
If A1 > 0 AND A2 < 0, Go to Level B
Else increase the window size
If $S_{xy} < S_{max}$ repeat Level A
Else output $Z_{xy}$

Level B.  B1 = $Z_{xy} - Z_{min}$
B2 = $Z_{xy} - Z_{max}$
If B1 > 0 AND B2 < 0, output $Z_{xy}$
Else output $Z_{med}$
```

where Z_{xy} = gray value at coordinate (x,y), Z_{max} = maximum gray value, Z_{min} = minimum gray value, S_{xy} = window size, S_{max} = maximum allowed window size. Maximum allowed window size for our experiment was 9X9 starting from 3X3 window.

B. *Standard Deviation Calculation For Homogeneity Measurement And Discontinuity Detection*

Edges exist in a color image where abrupt changes of RGB values occur. If a region in an image is homogeneous then the RGB values of that region will have a correlation among them and a small standard deviation. In probability theory and statistics, standard deviation is a measure of the variability or dispersion of a population, a data set, or a probability distribution. A low standard deviation indicates that the data points tend to be very close to the same value (the mean), while high standard deviation indicates that the data are "**spread out**" over a large range of values. So, if for a particular sub image (say a 3X3 neighborhood), the value of the standard deviation changes significantly from its previous positional value then there is a possibility of existence of an edge. Here we have used a 3X3 neighborhood of pixels to calculate standard deviation and homogeneity comparison.

1	1	1
1	1	1
1	1	1

Figure 1. 3X3 mask used to capture the RGB values

To reduce the computation overhead first we have calculated a transformed value for each pixel which converts three component valued pixels into a single valued attribute and then calculated standard deviation over that transformed value.

$$\text{Pixel(i,j)=Red(i,j)+2*Green(i,j)+3*Blue(i,j)} \qquad (1)$$

If $x_1, x_2, x_3, x_4 \ldots x_n$ are the transformed values of the pixels then standard deviation is calculated as follows:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2} \qquad (2)$$

where n is the no. of points, here no. of pixels and $\bar{x}$ is the arithmetic mean of the values x_i, defined as:

$$\bar{X} = (x_1 + x_2 + \ldots\ldots + x_n) \qquad (3)$$

C. *Threshold Technique*

Threshold technique is very important task in edge detection algorithms. The accuracy of an algorithm is dependent on the choice of threshold parameters. The criteria of selection of a parameter for a given image are that the resultant edge map should satisfy the following [8]:

- ❖ It should contain most of the prominent edges;
- ❖ It should not contain too much spurious edges;
- ❖ It should be meaningful and visibly pleasing;

The proposed method uses single threshold value T. There are several ways to obtain a fixed parameter value. A very simple way is to observe the edge maps for a set of selected images and take that value which is producing acceptable edge maps for all the selected images [8]. When such an experiment is done over more than 50 selected different kinds of images it produced acceptable edge maps at T=35 Thus the threshold value is set at T=35.

D. *Edge Thining*

The edge map produced in this way contains edges more than one pixel thick. So a thinning technique is applied to create thin edges. Two 3X3 masks are applied in thinning operation.

0	0	0
-1	2	-1
0	0	0

0	-1	0
0	2	0
0	-1	0

Figure 2. Thinning masks

An edge thinning scheme is presented in the proposed method. Two masks as shown in figure 2 are used to create thin edges. These two masks are moved over the edge image produced after thresholding. Left one works in the horizontal direction and the right one works in the vertical direction. This procedure creates thin pixel edge response by suppressing a thick edge response from both sides (vertically or horizontally) and eventually keeps only thin edge response which is more accurate and visibly soothing. Effect of the thinning masks is clearly observable from the figures given below:

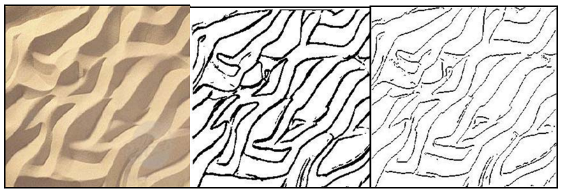

Figure 3. (a) Sand image, (b) multi pixel thick edge map after threshold, (c) final edge map after using thinning masks

III. Results And Comparison

Comparisons with several existing methods have been made to show the efficacy of the proposed method. Here we have considered the block image for comparison. Edge maps produced with Sobel operator, Laplace operator, Mexican hat operator, Cumani operator with different parameters, and finally with the proposed method are shown. It clearly shows that the output of the proposed method is easily comparable and in some cases better than the existing methods.

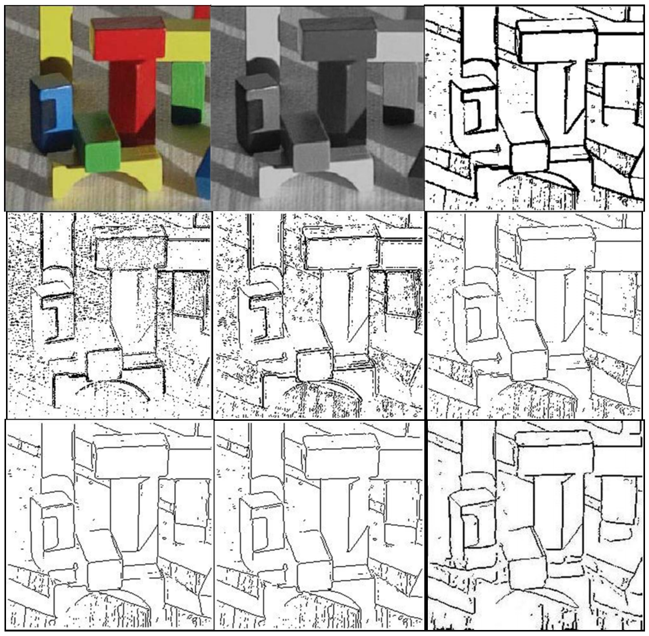

Figure 4. (a) Original block image, (b) gray image, (c) result of Sobel operator, (d)result with Laplace operator, (e) result with Mexican hat operator(σ=1.0), (f) result with Cumani operator with Gaussian mask (σ = 0.5), (g) result with Cumani operator with (σ=1.0), (h) Cumani operator with (σ= 1.0) applied to the intensity block image, (i) result of proposed method with T=1.2t.

Undoubtedly, as the above results demonstrate, the proposed method performed better than Sobel operator, Laplace operator, Mexican hat operator. In figure 4.(h) Cumani operator is applied to the intensity block image (σ= 1.0) produced good result. Proposed method in figure 4.(i) produced satisfactory output with less spurious edges than others with T=1.2t and contains most of the important edges.

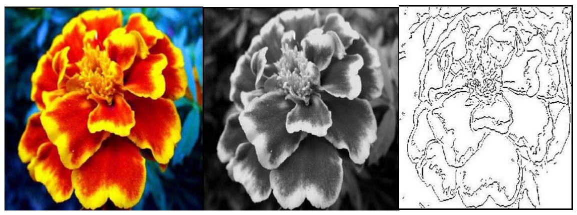

Figure 5. (a) Original flower image, (b) gray image(c) final edge map produced by proposed method

IV. Conclusion And Future Work

The proposed method, when with this fixed parameter values, tested on different images, it produced stable and fairly good results. Consistent acceptable outputs over different kinds of real life images have proved robustness of the presented scheme. Thus, the proposed method may be handy for any computer vision task where extraction of edge maps is required for a large set of images for feature extraction or for any other work. Though the scheme is computationally and logically less complex than most of the existing edge detection algorithms, still we need to show the results. So the next venture will be comparing those algorithms with the proposed one and analyze the performance on the basis of parameters like computing time, execution complexity and accuracy of the system output in presence of noise.

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] R. Nevatia, “A color edge detector and its use in scene segmentation,” *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-7, no. 11, pp. 820–826, Nov. 1977.

[4] Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing, Second Edition.

[5] A. Shiozaki, “Edge extraction using entropy operator,” *Computer Vis.,Graph., Image Process.*, vol. 36, no. 1, pp. 1–9, Oct. 1986.

[6] M. A. Ruzon and C. Tomasi, “Color edge detection with compass operator,” in *Proc. IEEE Conf. Computer Vision Pattern Recognition*, Jun. 1999, vol. 2, pp. 160–166.

[7] A. Cumani, “Edge detection in multispectral images,” *CVGIP: Graph. Models Image Process*, vol. 53, no. 1, pp. 40–51, Jan. 1991.

[8] S. K. Naik and C. A. Murthy, “Standardization of Edge Magnitude in Color Images,” *IEEE Trans. Image processing,* vol. 15, no. 9, Sept. 2006.

[9] M. A. Ruzon and C. Tomasi, “Edge, junction, and corner detection using color distributions,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol.23, no. 11, pp. 1281–1295, Nov. 2001.00X.