ISSN:2249-5789
Kiran Jot Singh et al, International Journal of Computer Science & Communication Networks,Vol 2(1), 83-86

Edge Detection Algorithm for Color Images using Logical Operations

Kiran Jot Singh[1], Monika Aggarwal[2]
Student[1], Assistant Professor[2]
BGIET, Sangrur, Punjab, India
kiranjot.693@gmail.com[1],monikaaggarwal@gmail.com[2]

Abstract

This paper is about a algorithm used for edge detection using logical operations. First image is converted into binary image and then it is rotated left, right, up and down. Afterwards logical AND operation is performed between rotated and original image and results are stored in different variables. After that XOR operation is performed between the results that are stored and then OR operation is performed on all results obtained after XOR operation. These operations are repeated for different values of threshold to get the final result constituting of only edges of that image. This method gives better or comparable output as given by other edge detection algorithms. This algorithm can be applied to variety of colored images.

1. Introduction

Edge detection is a procedure of finding significant changes in a image. Edge detection is basically the points where sharp changes in brightness occur typically from the border between different objects [1]. Edges characterize object boundaries and therefore useful for segmentation, registration, and identification of objects in scenes [2]. Many edge detectors are used these days but canny edge detection algorithm gives best output as compared to other edge detection algorithms such as Sobel, Prewitt and Robert [3]. Canny's edge detection algorithm is more complex then these algorithms and uses two threshold vales. So its computational expenses are more. According to study done by Carol L. Novak and Steven A. Shafer [4] 90% of the edges are about the same in gray level and in color images. It implies that 10% of the edges are left over in gray level images. Since color images give more information than gray-level images, this 10% left over edges may be extracted from color images.

In this proposed algorithm computation is done on four matrices. Three matrices are the red, green and blue matrices of RGB image and fourth matrix is combined form of all three matrices obtained by converting image to gray scale image. Now all these four matrices are converted into binary matrices at different threshold values. Afterwards these matrices are shifted left, right, up and down and then logical operations are performed on these matrices. At last all results are summed together to get final image in which only edges are present. This algorithm is tested on different color spaces such as RGB, YUV and HSV and the results are compared with the results of the other methods.

In Section 2, various color spaces and their inter conversion is discussed. In section 3 shift operations are discussed. In sections 4 logical approaches is defined. The proposed methodology is presented in Section 5 and the experimental results are presented in Section 6. Section 7 concludes the work.

2. Color Spaces

A color space relates to number of actual colors, and is a three dimensional object which contains all color combinations. Color spaces can be either dependent to or independent of a given device. Device-dependent spaces express color relative to some other color space whereas independent color spaces express color in absolute terms. Each dimension in color space represents some aspect of color, such as lightness, saturation or hue, depending on the type of space[5].

2.1. RGB Model

An RGB color space is any additive color space based on RGB color model. A particular RGB color space is defined by the three chromaticity's of red, green and blue additive primaries. RGB is a convenient color model for computer graphics because the human visual system works similar to an RGB color space.

2.2. HSV & HSL Model

HSL and HSV are two related representations of Points in RGB color space which attempt to describe the perceptual color relationships more accurately than

RGB. HSL stands for hue, saturation and lightness. HSV stands for hue, saturation and value.

2.3. YUV Model

YUV model defines a color space in one luminance (Y) and two chrominance (UV) components. YUV models human perception of color in a different way from the standard RGB model. Y stands for luminance (brightness) component and UV stands for the chrominance (color) components. The transformation from RGB into YUV is given by Since the luminance and chrominance components are separate, the YUV space is vigorously used to broadcast video systems and hence it is also used in image and video processing.

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.3 & 0.59 & 0.11 \\ -0.15 & -0.29 & 0.44 \\ 0.61 & -0.52 & -0.096 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

3. Shift Operations

In this algorithm shift operations are used to detect edges of a color image. Image is shifted left, right, up and down. Following operation are performed for shifting the image.

For Shifting image to Right:-

```
Img1(1:g-1,1:h)=img(2:g,1:h);
Img1(g:g,:)=img(1:1,:);
```

For Shifting image to Left:-

```
Img2(2:g,1:h)=img(1:g-1,1:h);
Img2(1:1,:)=img(g:g,:);
```

For Shifting image to Up:-

```
Img3(1:g,2:h)=img(1:g,1:h-1);
Img3(:,1:1)=img(:,h:h);
```

For Shifting image to Down:-

```
Img4(1:g,1:h-1)=img(1:g,2:h);
Img4(:,h:h)=img(:,1:1);
```

g = no of rows , h = no of columns

Let us assume a binary image:-

$$Img = \begin{matrix} 0&0&0&0&0&0 \\ 0&1&1&1&1&0 \\ 0&1&1&1&1&0 \\ 0&1&1&1&1&0 \\ 0&0&0&0&0&0 \end{matrix}$$

After performing shift right operation we get

$$Img1 = \begin{matrix} 0&0&0&0&0&0 \\ 0&0&1&1&1&1 \\ 0&0&1&1&1&1 \\ 0&0&1&1&1&1 \\ 0&0&0&0&0&0 \end{matrix}$$

After performing shift left operation we get

$$Img2 = \begin{matrix} 0&0&0&0&0&0 \\ 1&1&1&1&0&0 \\ 1&1&1&1&0&0 \\ 1&1&1&1&0&0 \\ 0&0&0&0&0&0 \end{matrix}$$

After performing shift up operation we get

$$Img3 = \begin{matrix} 0&1&1&1&1&0 \\ 0&1&1&1&1&0 \\ 0&1&1&1&1&0 \\ 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \end{matrix}$$

After performing shift down operation we get

$$Img4 = \begin{matrix} 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \\ 0&1&1&1&1&0 \\ 0&1&1&1&1&0 \\ 0&1&1&1&1&0 \end{matrix}$$

4. Logical Approach

After we get rotated matrices logical operations are performed on then to detect edges. First logical AND operation is performed between original image matrix and rotated matrices and after that logical XOR operation is performed between the results. Then results of XOR operation are summed together using logical OR operation.

Img_And1 = (Img) AND (Img1)

$$Img\_And1 = \begin{matrix} 0&0&0&0&0&0 \\ 0&1&1&1&0&0 \\ 0&1&1&1&0&0 \\ 0&1&1&1&0&0 \\ 0&0&0&0&0&0 \end{matrix}$$

Img_And2 = (Img) AND (Img2)

$$Img\_And2 = \begin{matrix} 0&0&0&0&0&0 \\ 0&0&1&1&1&0 \\ 0&0&1&1&1&0 \\ 0&0&1&1&1&0 \\ 0&0&0&0&0&0 \end{matrix}$$

Img_And3 = (Img) AND (Img3)

$$Img\_And3 = \begin{matrix} 0&0&0&0&0&0 \\ 0&1&1&1&1&0 \\ 0&1&1&1&1&0 \\ 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \end{matrix}$$

Img_And4 = (Img) AND (Img4)

$$Img\_And4 = \begin{matrix} 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \\ 0&1&1&1&1&0 \\ 0&1&1&1&1&0 \\ 0&0&0&0&0&0 \end{matrix}$$

ISSN:2249-5789
Kiran Jot Singh et al, International Journal of Computer Science & Communication Networks,Vol 2(1), 83-86

Img_Xor1 = (Img_And1) XOR (Img_And2)

```
           0 0 0 0 0 0
           0 1 0 0 1 0
Img_Xor1=  0 1 0 0 1 0
           0 1 0 0 1 0
           0 0 0 0 0 0
```

Img_Xor2 = (Img_And3) XOR (Img_And4)

```
           0 0 0 0 0 0
           0 1 1 1 1 0
Img_Xor2=  0 0 0 0 0 0
           0 1 1 1 1 0
           0 0 0 0 0 0
```

ImgFinal = (Img_Xor1) OR (Img_Xor2)

```
           0 0 0 0 0 0
           0 1 1 1 1 0
ImgFinal = 0 1 0 0 1 0
           0 1 1 1 1 0
           0 0 0 0 0 0
```

5. Proposed Methodology

We combined shift operations and logical approach to extract the edges of the color image. The block diagram is shown in Fig. 1.

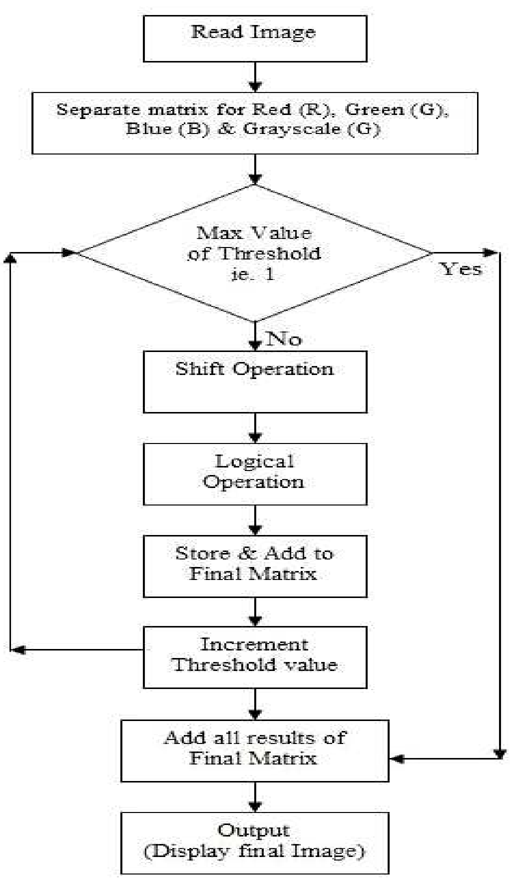

Fig.1 Block diagram for proposed methodology

Following are the steps for proposed method:-

1. Read the color image.
2. Separate Red, Green, Blue and Grayscale matrix.
3. Select a threshold value.
4. Apply shift operations.
5. Apply logical operations.
6. Save results in a new matrix.
7. Increment the threshold value.
8. Repeat step 4 to 6 for different value of thresholds.

6. Experimental Results

The results of the proposed method are presented in this section. This method is applied on both real world and synthetic images. Results obtained are better or comparable as compared to the results of other edge detection algorithms. Results for synthetic and real images are shown in Fig.2 and Fig. 3 respectively.

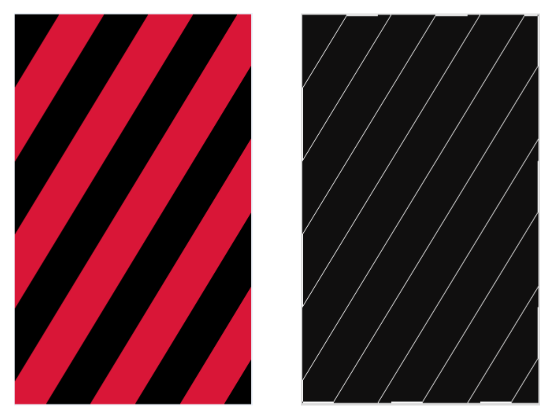

Fig. 2 Synthetic image and its corresponding edge image

Fig. 3 Real world image and its corresponding edge image

6. Conclusions

Method proposed in this paper extract edges from color images. The output given by this method is more accurate than the existing edge detectors. This algorithm uses only simple shift operations and logical operations to compute edges of an image. Time taken by this method is less as compared to other edge detection techniques. Experimental results show that its output is satisfactory in different cases.

6. References

[1] S Jayaraman, S Esakkirajan, T Veerakumar , "Edge Detection" *Digital Image Processing*, Tata McGraw Hill, 2010,pp.381

[2] Anil K. Jain, "Edge Detection", *Fundamentals of Digital Image Processing*, Prentice Hall of India,2002,pp.347

[3] Stephen M. Smith, J. Michael Brady, "SUSAN—A New Approach to Low Level Image Processing*", International Journal of Computer Vision* 23(1), 45–78 (1997)

[4] C.L Novak and S.A Shafer, "Color edge detection", *Proceedings of DARPA Image Understanding workshop*, Los Angeles, CA, USA, vol.1,pp.35-37,1987.

[5] T. N. Janakiraman, and P. V. S. S. R. Chandra Mouli," Color Image Edge Detection using Pseudo-Complement and Matrix Operations", *World Academy of Science Engineering and Technology*, 42 2008