Color-to-Grayscale Algorithms effect on Edge Detection – A Comparative Study

Ijaz Ahmad[1], Inkyu Moon[1], and Seok Joo Shin[1,*]
[1]Dept. of Computer Engineering, Chosun University, 309 Pilmun-daero, Dong-gu, Gwangju, 61452 South Korea
[*]Corresponding author: sjshin@chosun.ac.kr

Abstract

In image processing, color images are converted into grayscale to perform edge detection, without considering the color-to-grayscale algorithms in details. We have evaluated the impact of various color-to-grayscale algorithms in edge detection. This study shows that edges are not only dependent on the methods used for edge detection but also on the color-to-grayscale conversion algorithms. We have implemented ten different color-to-grayscale conversion algorithms in MATLAB R2016a and the resultant grayscale images were further tested with eight different edge detection algorithms. The experimental results shows that the Lightness color-to-grayscale conversion algorithm achieves higher performance among evaluated methods.

Keywords: edge detection, color-to-grayscale algorithms

1. Introduction

Edge detection is of significant importance in image processing. Edge is an abrupt change in intensity and characterizes boundaries. Edge detection in grayscale images is a well-established field, while in color images it has not gained that much focus [1]. Recently, different techniques have been experimented for edge detection in color images, e.g. monochromatic-based techniques (in which an edge detection method is applied separately on every channel and then the result of these three images are fused) and vector-valued techniques (with the main focus on, to treat the channels of color images holistically) [1] [2]. However, these techniques face two different problems, the prior technique is simple but has high computational cost while the later complicates the algorithm. To minimize the computational cost and simplify the complexity of edge detection algorithms on color images, instead of full color images, grayscale images are used for edge detection.

Many edge detection algorithms have been proposed [3] [4] [5] [6] [10] to make the edge detection process efficient for grayscale as well as color images. Also, there are number of methods used for color-to-grayscale conversion [5] [7] [8] [9] [11]. Each conversion method results different in image recognition [8] and image description tasks [9]. In this paper, we have replicated the results of [8] and [9] for edge detection in color images and demonstrated that edge detection is also dependent on color-to-grayscale conversion methods. The edge detection process of color images is shown in Fig. 1.

In image processing different methods have been devised for color-to-grayscale conversion. Among them the most commonly used techniques are based on weighted averages of red, green and blue channel (e.g. Intensity and Luminance), some other methods generate a more accurate perceptually representation with respect to brightness by adopting alternative methods than that of averaging (e.g. Luma and Lightness) [8]. The grayscale image obtained from each of these methods is a function of the pure Red, Green and Blue channels. In the grayscale image a three dimensional color pixel value is represented with a single value, thus resulting in information loss. Also, different color-to-grayscale techniques not necessarily results in same grayscale image [9], thus in this paper we have shown that the edges obtained from these methods are also different.

2. Color-to-Grayscale conversion

In this section we briefly describe ten color-to-grayscale conversion algorithms of a linear time complexity used in this study. Each of these techniques is a function G with an input color image R of dimension 3*n*m and an output of grayscale image R of dimension n*m. The pixel values of the input color image and the output grayscale values are in the range of 0 to 255. The 0 represents black while 255 represents white [8] [9]. In

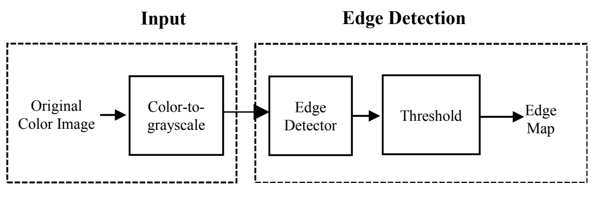

Fig. 1. Edge detection in color images

Table I: Color-To-Grayscale Conversion Algorithms

Name	Equation
GIntensity	$(R+G+B)/3$
GLuminance	$0.3*R+0.59*G+0.11*B$
GMax_Value	$max(R,G,B)$
GMin_Value	$min(R,G,B)$
GLuster	$(max(R,G,B)+\min(R,G,B))/2$
GLuma	$0.2126R'+0.7152G'+0.0722B'$
GLightness	$\frac{(116f(Y)-16)}{100}$, $Y=0.21R+0.71G+0.07B$
GRed	$extractR(R,G,B)$
GGreen	$extractG(R,G,B)$
GBlue	$extractB(R,G,B)$

this paper we have adopted the symbol convention and functions' name from [8], as listed in Table I.

3. Edge detection algorithms

There are two main approaches for edge detection based on derivative: **first order derivative-based** or **gradient-based** edge detector detects an edge in image by locating where the intensity reaches a local extremum or a local maximum, e.g. Sobel operator, Robert operator, Prewitt operator; **second order derivative-based** or **Laplacian-based** edge detector locates an edge where a zero crossing occurs, e.g. Laplacian of Gaussian, Difference of Gaussian [4] [6].

We have implemented six first order derivative-based edge detection operators: Sobel Operator, Prewitt Operator, Robert Operator, Kirsch Compass Operator, Robinson Compass Operator, and Frei-Chen Compass Operator. [6] The first three edge detection operators are common first order derivative edge detectors, in which two masks are separately convolved with the image to measure the strength of edges in horizontal and vertical direction, as in (1)(2). The result of both edges are combined to find the total amount of edge at a particular pixel, as in (3).

$$E^{h}_{i,j} = -x_{i-1,j-1} - 2x_{i-1,j} - x_{i-1,j+1} + x_{i+1,j-1} + 2x_{i+1,j} + x_{i+1,j+1} \quad (1)$$

$$E^{v}_{i,j} = -x_{i-1,j-1} - 2x_{i,j-1} - x_{i+1,j-1} + x_{i-1,j+1} + 2x_{i,j+1} + x_{i+1,j+1} \quad (2)$$

$$E^{total}_{i,s} = ((E^{h}_{i,s})^2 + (E^{v}_{i,s})^2)^{1/2} \quad (3)$$

Since, we have addition and multiplication by negative numbers, in (1), (2) and (3), in order to avoid incorrect intensity values (i.e. values greater than 255 and less than 0) thresholding is applied as the next step. Thresholding is done as in (4).

$$E^{total}_{i,s} = \begin{cases} 0, & if\ E^{total}_{i,s} \le 0 \\ 255, & if\ E^{total}_{i,s} > 255 \end{cases} \quad (4)$$

The last three detectors are compass operators, in which eight masks are separately convolved with the image to detect edge strength in all eight directions. The convolution process is carried out similar to (1) or (2) with different indexes value. Instead of combining all the results, the max value among the eight values is selected as edge pixel. Finally, thresholding is applied to get correct intensity value for an edge [4] [6].

We have implemented two second order derivative-based detectors Laplacian of Gaussian (LOG) and Difference of Gaussian (DOG). In LOG approach first an input image is smoothened with a Gaussian filter and then

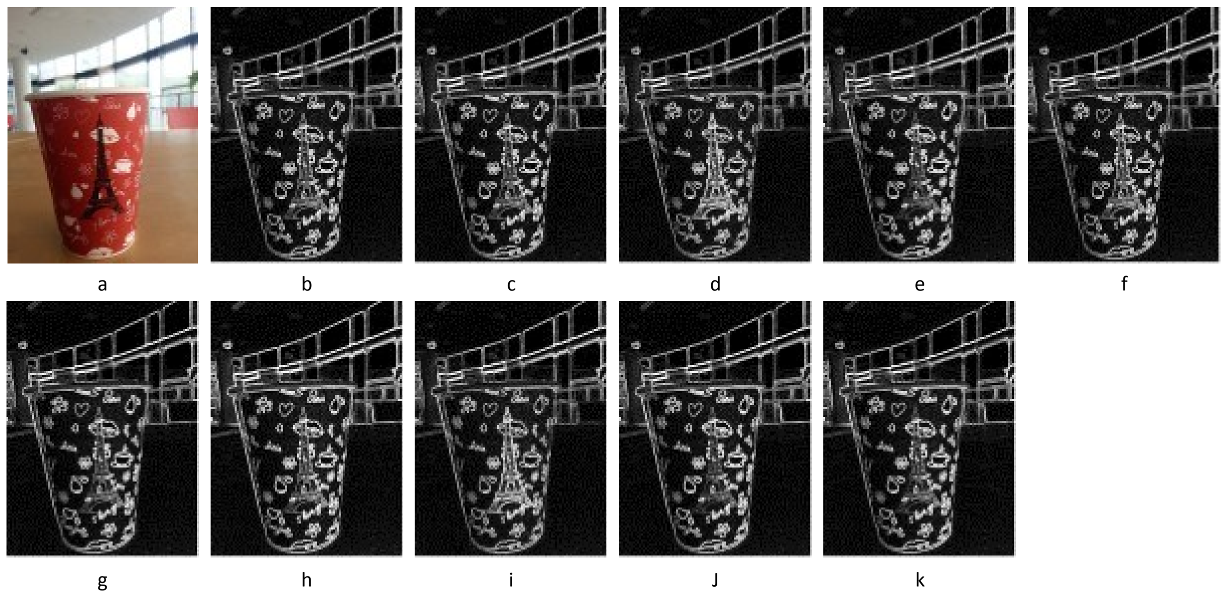

Fig. 2. Edge maps using Sobel edge detection

Table II. Comparison of GLightness with other color-to-grayscale conversion methods

Methods	Sobel			Prewitt			Robert			Kirsch			Robinson			Frei-Chen			LOG			DOG		
	F	*F'*	*E*	*F*	*F'*	*E*	*F*	*F'*	*E*	*F*	*F'*	*E*	*F*	*F'*	*E*	*F*	*F'*	*E*	*F*	*F'*	*E*	*F*	*F'*	*E*
Gintensity	0.7	0.3	0.4	0.7	0.3	0.4	0.3	0	0.3	0.7	0	0.7	0.7	0	0.7	0.7	0.3	0.4	0.7	0	0.7	0.7	0	0.7
GLuminance	0.7	0	0.7	0.7	0	0.7	1	0	1	0.7	0.7	0	0.7	0	0.7	0.7	0	0.7	0.7	0	0.7	1	0	1
GMax_Value	1	-0.3	1.3	1	0.4	0.6	0.7	0.3	0.4	1	0	1	0.7	0.4	0.3	0.7	-0.3	1	1	0.4	0.6	0.4	0	0.4
GMin_Value	0.7	-0.3	1	1	-0.3	1.3	0.7	0	0.7	1	0.7	0.3	0.7	-0.3	1	0.7	-0.3	1	0.3	0	0.3	0	0	0
GLuster	0.3	0	0.3	0.7	0.7	0	1	0.3	0.7	0.7	0	0.7	0.7	0.7	0	0.7	0.4	0.3	0.3	0	0.3	0	-0.3	0.3
GLuma	1	0	1	0.7	0	0.7	1	0	1	0.7	0	0.7	0.7	0	0.7	1	0	1	0.7	0	0.7	0.3	0.4	-0
GRed	0.3	-0.3	0.6	0.7	0.4	0.3	0.7	0.3	0.4	1	0	1	0.7	0	0.7	0.3	0.3	0	0.3	0	0.3	0	-0.3	0.3
GGreen	1	0.3	0.7	0.7	0.3	0.4	0.3	0	0.3	0.7	0.7	0	0.3	0	0.3	0.7	0.3	0.4	0.3	0.3	0	0.7	0.7	0
GBlue	0.3	0	0.3	1	0.7	0.3	0.7	0	0.7	0.7	0	0.7	0.7	0	0.7	0.7	-0.3	1	0.3	0.3	0	0.4	0	0.4

the resultant image is convolved with Laplacian filter to detect edges. In DOG two different Gaussian filters are applied to smoothen the image, the difference of both images is calculated as the edge map image [3] [4] [6].

4. Implementation and performance evaluation

In order to study the effect of color to grayscale algorithms on the performance of edge detection in image processing, we have implemented different color-to-grayscale methods and edge detection algorithms in MATLAB R2016a. We have performed our experiments on the sample image shown in Fig. 2(a). It consists of a cup as focused object, while the windows and chairs out of focus in the background with a noticeable degree of blurriness. The color-to-grayscale algorithms of TABLE I, are applied on the sample image to obtain grayscale images, the edge maps of the resultant grayscale images are constructed using the edge detection technique discussed in section III. Fig. 2 (b-k) shows the edge maps of Sobel operator for the grayscale resultant images of sample image.

We evaluated the relative effect of the color-to-grayscale methods on the performance of edge detection techniques by considering the important qualitative parameters; Edges preservation (P), Edge enhancement (H), and Noise avoidance (N). The grading criteria as in (5), is used to evaluate the effect of color-to-grayscale algorithms. The designed parameters P, H and N are assigned with the weights of 0.7, 0.3 and -0.3, respectively.

$$\mathrm{F} = 0.7(\mathrm{P}) + 0.3(\mathrm{H}) - 0.3(\mathrm{N}) \qquad (5)$$

Based on visualizing our experimental results we found that the edge maps obtained for the GLightness grayscale images shows greater details. We have confirmed our observations through the difference of edge maps, evaluating (5) for each method. Further, TABLE II lists the comparison results of the GLightness method with other conversion methods for all edge detection algorithms. The performance grade of GLightness from other method is given by F for an edge detection method, while F' denotes the vice versa. The efficiency of GLightness method with respect to other methods is given by E for each edge detection algorithm. Fig. 3, shows the accumulative efficiency of GLightness with respect to other conversion methods for all edge detection algorithms.

We obtained the difference of the edge maps for a color-to-grayscale conversion algorithm from all other methods and vice versa. Fig. 4, lists the differences of the edge maps obtained for the grayscale images of the conversion methods given in TABLE I, from the edge maps obtained for GLightness method for Sobel edge detection and vice versa. The difference of a method from GLightness is denoted with DGLightness while the difference of GLightness from a method is denoted with D followed by the method name.

We observed that the difference of edge maps obtained for other conversion methods from GLightness method is significant, as shown Fig. 4 (a-i). GLightness preserved extra edges with respect to Fig. 4 (j, k, l, m, o, and q), while it has enhanced background edges with respect to Fig. 4 (l, n, o, p, and r). In case of Fig. 4 (j, k, l, m, and p), GLightness has better outlined the object, and preserved edges in the object. On the contrary, the edge

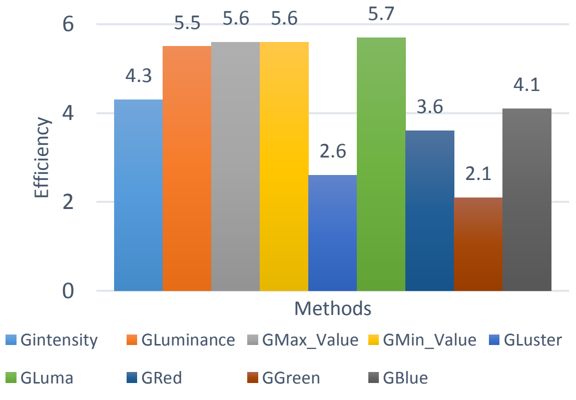

Fig. 3. Efficiency of GLightness method from others

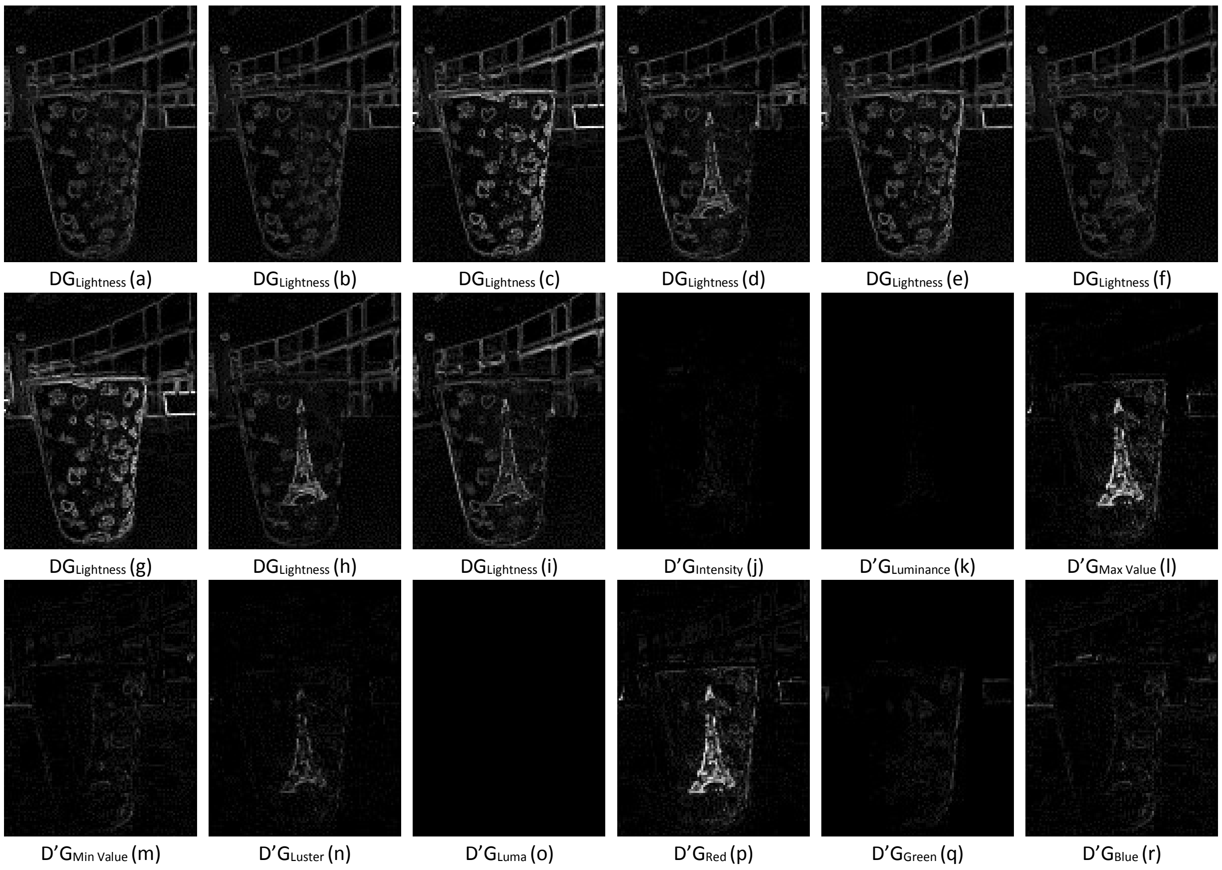

Fig. 4. Comparison results

maps obtained for the difference of GLightness method from the rest of methods is less perceptible and insignificant to contribute to edges as shown in Fig. 4 (j-r). All of the listed methods performed poor in preserving background edges and outline of the object. Fig. 4 (m, p, and r) resulted in some false edges. GLightness is successful in preserving edges in focused region, however methods except Fig. 4 (k and o) contributed in enhancement of those edges.

The comparison result of GLightness with other methods for Sobel edge detection are shown in Fig. 4. The high resolution images, grayscale images, edge maps, animation of the complete set, and comparison results images for edge detection methods can be found on [12].

5. Conclusion

Edge detection plays an important role in image processing applications. However, the effect of color-to-grayscale methods on edges has not been considered in details. We have studied the effect of different color-to-grayscale conversion algorithms on edge detection. Based on the experimental results, edge detection performed on the grayscale images obtained from the GLightness conversion algorithm performs best. The edge detection algorithm performed best in outlining the focused object and in edge detection in the blurred region of such grayscale images for all listed edge detection algorithms.

Acknowledgement

This research is supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2015R1D1A1A01059962)

References

[1] Andreas Koschan and Mongi Abidi, “Detection and classification of edges in color images”
[2] Saining Xie and Zhuowen Tu, “Holistically-Nested edge detection”
[3] Shashindar Ram Joshi and Roshan Koju, “Study and comparison of edge detection algorithms”
[4] Rafael C. Gonzales and Richard E. Woods, “Digital image processing”
[5] Hanumanthappa, S Regina Luordhu Suganthi, “Edge detection in color images—A comparative study”
[6] Al Bovik, “Handbook of image and video processing”
[7] Mark Grundland and Neil A. Dodgson, “The decolorize algorithm for contrast enhancing, color-to-grayscale conversion”
[8] Christoper Kanan and Garrison W. Cottrell, “Color-to-Grayscale: Does the method matter in image recognition”
[9] Samuel Macedo, Givanio Melo and Judith Kelner, “A comparative study of grayscale conversion techniques applied to SIFT descriptors
[10] Li Zhuo, Xiaochen Hu and Liying Jiang, “A color image edge detection algorithm based on color differences”
[11] Ali Gunes, Habil Kalkan and Efkan Durumus, “Optimizing the color-to-grayscale conversion for image classification”
[12] https://drive.google.com/open?id=0BzuRL2vZzmYVOUhTeUhLTFN1djQ