CHROMINANCE EDGE PRESERVING GRAYSCALE TRANSFORMATION WITH APPROXIMATE FIRST PRINCIPAL COMPONENT FOR COLOR EDGE DETECTION

Salih Dikbas, Tarik Arici, and Yucel Altunbasak

School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, Georgia 30332–0250
Email: {salih, tariq, yucel}@ece.gatech.edu

ABSTRACT

Edges that are visible in color images may not be detected in the corresponding grayscale image. This is due to the neighboring objects having different hues but the same intensities. Hence, a color edge preserving grayscale conversion algorithm is proposed that helps detect color edges using only the luminance component. The algorithm calculates an approximation to the first principal component to form a new set of luminance coefficients instead of using the conventional luminance coefficients. This method can be directly applied to all existing grayscale edge detectors for color edge detection. Processing only one channel instead of three channels results in lower computational complexity compared to other color edge detectors. Experimental results on test images show similar edge detection capabilities to typical color edge detectors at reduced complexity levels.

Index Terms— Color edge detection, Image edge analysis

1. INTRODUCTION

Edge detection is one of the fundamental tasks in image processing and computer vision because of its wide use in several techniques such as segmentation, object recognition, tracking, stereo analysis, data hiding, and image coding. The efficacy of the subsequent techniques is heavily affected by the accuracy of edge detection. Conventionally, grayscale images have been used to detect the edges in an image. Pursuit of good edge detection algorithms led to such grayscale edge detectors as Canny, Cumani, and Compass [1–3]. Edges of the spatially neighboring objects with different hues but equal grayscale values cannot be detected using grayscale transformation since the color cue is lost during grayscale conversion.

To obtain more meaningful edges, there has been an increased interest in color edge detection. Humans can differentiate thousands of colors compared to about two dozen shades of gray; hence, grayscale images do not carry all the edge information that human visual system (HVS) can detect. In [4], it is stated that luminance component makes up 90% of all edge points in a color image but the remaining 10% can be critical for subsequent techniques that rely on edges in an image; in some cases the additional information provided by color is of utmost importance. Multi-dimensional nature of color makes it more challenging to detect edges in color images, and often increases the computational complexity threefold compared to grayscale edge detection; hence, color edge detection algorithms accept from the beginning that all of the efforts are to find the remaining 10% of the edges. Importance of color edge detection also becomes more apparent in low contrast images [5].

Color edge detection techniques fall into two main categories. Techniques in the first group [6–10] calculate gradients in each color component separately, then either fuses the gradients immediately or detect edges in each component separately before fusing to detect color edges. Techniques in the second group [2,3,11–15] treat each pixel as a three-tuple vector and apply vector processing techniques without decoupling color components to obtain the edge map. A comprehensive analysis of color edge detectors can be found in [5, 16].

There is no universally accepted ”color edge” definition. Literature in this field suggest the following three definitions: (1) an edge exists if there is an edge in the corresponding grayscale image, (2) an edge exists if at least one of the color components has an edge, and (3) an edge exists if some norm (generally L_1, L_2, or L_∞) of the gradient from each color component exceeds a threshold value.

In this paper, we propose a transformation that preserves chrominance edges. This transformation effectively reduces the dimensionality of color space from three to one dimension for detecting color edges along with the already attainable edges from grayscale image. The proposed method is based on principal component method. One advantage of using this method is that it enables the use of many existing grayscale edge detection techniques to detect color edges. From one perspective, the proposed method can be seen as a preprocessing step in grayscale conversion. It finds the weighting coefficients for each color component; hence, enabling edge detection to find color edges that may be impossible to find in standard grayscale images. The proposed method detects the color edges as other color edge detectors, but at a reduced computational complexity.

The organization of the paper is as follows: Section two gives the necessary mathematical framework of the method, and explains the proposed method in detail. In Section three, results of the proposed method are presented along with a discussion comparing with other methods. Then, the paper is concluded in Section four with remarks.

2. GRAYSCALE CONVERSION

Before describing the proposed method, this section describes the necessary mathematical foundations it relies on. First, principal component method is discussed. Second, approximation to finding principal component vector is presented. Finally, proposed algorithm making use of these ideas is presented.

2.1. Principal component Analysis

Principal component analysis (PCA) is typically used for two purposes: (1) to de-correlate a data set, and (2) to reduce the dimensionality of the data set. For a color image $\mathbf{f}$ of size $M \times N$, each pixel location $[m, n]$ is represented by a three-tuple color vector $\mathbf{f}[m, n]$ for $m = 1, 2, \cdots, M$ and $n = 1, 2, \cdots, N$. Each color component can be represented as f_i, for $i = 1, 2, 3$. Then, the maximum-likelihood (ML) estimate of the mean is calculated as

$$\bar{\mathbf{f}} = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} \mathbf{f}[m, n]. \tag{1}$$

And, the ML covariance matrix estimate is

$$C = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} (\mathbf{f}[m, n] - \bar{\mathbf{f}})(\mathbf{f}[m, n] - \bar{\mathbf{f}})^T, \tag{2}$$

where C is a 3×3 real and symmetric matrix. Then, this matrix is used to solve for eigenvectors $\mathbf{v_1}, \mathbf{v_2}, \mathbf{v_3}$ corresponding to eigenvalues $\lambda_1, \lambda_2, \lambda_3$ such that $\lambda_1 \geq \lambda_2 \geq \lambda_3$. De-correlated color components can be written as

$$g_i[m, n] = \mathbf{v_i}^T \, \mathbf{f}[m, n], \quad i = 1, 2, 3 \tag{3}$$

where $m = 1, 2, \cdots, M$ and $n = 1, 2, \cdots, N$. As a result of applying PCA, the data is projected along the directions where it varies most; the variation of g_i is greater than the variation of g_j for $i < j$.

2.2. Principal component vector computation

There are several numerical methods [17] that can be used to compute the eigenvalues and eigenvectors of a matrix. First k components can be used to represent the data for many statistical purposes as most of the variance is contained in the first k principal components depending on the eigenvalues. If most of the variance is contained in the first principal component, then $k = 1$ can be used to represent the data, in which case, $\lambda_1 \gg \lambda_j$, for $j = 2, 3$, is satisfied.

Since solving for only the first principal component serves our purpose, we can eliminate eigenvalue calculation for all the components and suffice with an approximate first component value calculation method. Starting by an estimate of principal vector $\mathbf{v_1}$, power iteration method can be used to find a good estimate to the the actual principal component. The following derivation shows how an approximate first principal component is calculated.

Let $\mathbf{C}$ be an $n \times n$ covariance matrix with eigenvalues ordered as $|\lambda_1| \geq |\lambda_j| \geq \cdots |\lambda_n|$, with corresponding eigenvectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}$. Let $\mathbf{v}^{(0)}$ be a normalized vector not orthogonal to $\mathbf{v}_1$, where the superscript denotes the iteration number in parenthesis. Then, $\mathbf{v}^{(0)}$ can be written in terms of the eigenvectors of C as

$$\mathbf{v}^{(0)} = a_1 \mathbf{v_1} + a_2 \mathbf{v_2} + \cdots + a_n \mathbf{v_n} \tag{4}$$

for a set of coefficients $\{a_i\}$, where $a_1 \neq 0$. Then, an estimate of the first principal component at the $k + 1$th iteration is defined by power method recursion as $\mathbf{v}^{(k+1)} = C\mathbf{v}^{(k)}$. Then, taking advantage of the property that the principal vectors which are transformed by the matrix C will be scaled in the direction of the corresponding eigenvalue and using induction one can write

$$\begin{aligned}
\mathbf{v}^{(1)} &= C\mathbf{v}^{(0)} = \lambda_1 \left(a_1 \mathbf{v_1} + a_2 \left(\tfrac{\lambda_2}{\lambda_1} \right) \mathbf{v_2} + \cdots + a_n \left(\tfrac{\lambda_n}{\lambda_1} \right) \mathbf{v_n} \right) \\
\mathbf{v}^{(2)} &= C\mathbf{v}^{(1)} = \lambda_1^2 \left(a_1 \mathbf{v_1} + a_2 \left(\tfrac{\lambda_2}{\lambda_1} \right)^2 \mathbf{v_2} + \cdots + a_n \left(\tfrac{\lambda_n}{\lambda_1} \right)^2 \mathbf{v_n} \right) \\
&\vdots \\
\mathbf{v}^{(k+1)} &= C\mathbf{v}^{(k)} = \lambda_1^{k+1} \left(a_1 \mathbf{v_1} + a_2 \left(\tfrac{\lambda_2}{\lambda_1} \right)^{k+1} \mathbf{v_2} + \cdots + a_n \left(\tfrac{\lambda_n}{\lambda_1} \right)^{k+1} \mathbf{v_n} \right)
\end{aligned} \tag{5}$$

where $k \geq 0$. As $k \to \infty$, $\mathbf{v}^{(k)} \to \mathbf{v_1}$ because of the ordering of the eigenvalues.

The expression $\mathbf{v}^{(k+1)} = C\mathbf{v}^{(k)}$ can be rewritten as $\mathbf{v}^{(k+1)} = C^{k+1}\mathbf{v}^{(0)}$ as well. In this case, an estimate of the first principal component at the $k + 1$th iteration is defined in terms of the matrix C and the initial estimate $\mathbf{v}^{(0)}$.

Then, an estimate of the first principal component $\mathbf{v}^{(k)}$ at the kth iteration is expressed as $\mathbf{v}^{(k)} = C^k \mathbf{v}^{(0)}$, and for $k \geq 0$ approaches $\mathbf{v}_1$ as k approaches ∞.

Note that, when dealing with eigenvectors normalization of an eigenvector is done using L_2 norm. In this paper, however, L_1 norm is used instead since the components of the vector are used as weighting coefficients of R,G, and B color components for grayscale conversion.

An immediate consequence of (5) is that the error decreases in the $\mathcal{O}(|\frac{\lambda_2}{\lambda_1}|^k)$. Experiment with several images that are used in image processing literature suggest that $k = 3$ is a good choice of tradeoff and gives an error less than 0.001.

2.3. Proposed Method

When a color image is converted to a grayscale image, generally $\mathbf{v_0} = [0.299 \quad 0.587 \quad 0.114]$ vector is used for weighting red, green, and blue color components of RGB images, respectively. These weights correspond to the sensitivity of human visual system (HVS) to each of the RGB primaries. Some of the color spaces, i.e., hue-based color spaces, use $\mathbf{v_0} = [0.333 \quad 0.333 \quad 0.333]$ to calculate the grayscale intensity value. Regardless of the selection, grayscale conversion fails to preserve color edges in certain situations, e.g. Fig. 1(a). In this case, the grayscale image does not carry the color edge information at all as shown in Fig. 1(b). Color edge detectors, on the other hand, can easily detect these edges as shown in Fig. 1(d).

In using the proposed grayscale images for edge detection, the purpose is not to generate monochrome images for visual perception but to generate monochrome images that carry the color edge information. Hence, different set of coefficient could be used if they enable preservation of color edges. Principal component answers that problem by supplying optimal vector for each image so that the resulting grayscale image has maximum variation; and, consequently color edge information is preserved during the grayscale conversion.

The proposed method obtains the grayscale image using approximate first principal component as the weights of each color components. In RGB color space, f_i represents each color component for $i = R, G, B$. Then, the mean vector is found using (1) and then the covariance matrix is found using (2). Approximate principal component is obtained by $\mathbf{v_1} = C^k \mathbf{v}^{(0)}$. Hence, the grayscale image can be obtained using (3) for $i = 1$.

2.4. Computational Complexity Analysis

Color canny edge detector processes three color components before forming the color gradient. First, each component is filtered with gaussian to smooth the image. Then, each smoothed color component is filtered with the derivative of gaussian. Separability of the gaussian is employed to reduce the computational complexity. Smoothing requires $2N \times M \times w_\sigma$ multiplications and $2N \times M \times (w_\sigma - 1)$ additions for each color component, where w_σ is the length of the smoothing filter. The length of the smoothing filter is at least 3 and increases as σ increases. For example, $w_1 = 9$, $w_{1.5} = 13$ and $w_2 = 17$. Filtering with the derivative of gaussian requires $N \times M \times w_\sigma^2$ multiplications and $N \times M \times (w_\sigma^2 - 1)$ additions for each color component. Hence, the increased computational complexity of color canny is $2N \times M \times (w_\sigma^2 + 2w_\sigma)$ multiplications and $2N \times M \times (w_\sigma^2 + 2w_\sigma - 2)$ additions.

The proposed method requires the calculation of the covariance matrix estimate. Since C is symmetric, only six elements of the matrix need to be calculated. Calculation of (1) requires $3N \times M$ additions, and the calculation of (2) requires additional $6N \times M$ multiplications and $6N \times M$ additions. Taking the kth power of C requires $k \times 15$ multiplications and $k \times 10$ additions. Hence, the increased computational complexity of the proposed method is $6N \times M + 15k$ multiplications and $6N \times M + 10k$ additions.

3. RESULTS AND DISCUSSION

Experimental results assessing the edge detection are obtained for the images Tile, Paper, and Tulips. The first one is a synthetic image; it is generated such that each patch has the same intensities but different hues. The others are natural images. For comparison purposes, color variant of canny operator, which incorporates the Jacobian matrix that is discussed in [5], is implemented. Three different edges are obtained: (1) canny edge detector applied to regular grayscale image, (2) canny edge detector applied to grayscale image obtained by the proposed method, and (3) color canny operator applied to color image.

In all three edge detection, the same values of sigma, σ, and threshold values are used. Sigma values of 1, 1.5, and 1 are used for Tile, Paper and Tulips images, respectively. The high and low threshold values of the hysteresis are chosen automatically such that $P(X \le t_h) = 0.7$, and $t_l = 0.4t_h$.

Figs. 1(a)-(f) depicts the original Tile image, its regular grayscale version, its proposed grayscale version, and the corresponding edges obtained from them. For this image, it is impossible to detect color edges using regular grayscale image as shown in Fig. 1(e). Color canny can detect edges easily since it employs all three color channels. Using the proposed conversion algorithm grayscale image shown in Fig. 1(c) is obtained. Since the principal component is the direction of the largest variation on the data, this modified grayscale image clearly shows each color patch as distinct regions. For the edge detection of color and grayscale images in Fig. 1(g)-(i), $\sigma = 1.5$ used for good edge detection. One of the sheet in the image was not detected for the grayscale image due to close intensity value with the background. However, it was easily detected for the modified grayscale image and color image. The edges in Fig. 1(p)-(r) look similar. But, a close inspection reveals that edge formation in Fig. 1(p) and Fig. 1(r) are better than Fig.1(q), and Fig. 1(r) gives more meaningful edges than Fig.1(p). The proposed method produces similar results to color canny edge detector. However, the increased computational complexity of the proposed method is a fraction of the color canny's.

4. CONCLUSION

This paper presents a chrominance edge preserving grayscale conversion method. The proposed method uses the approximate first principal component vector as the grayscale conversion coefficients. To save computation time and to design a hardware implementable real-time algorithm, the proposed

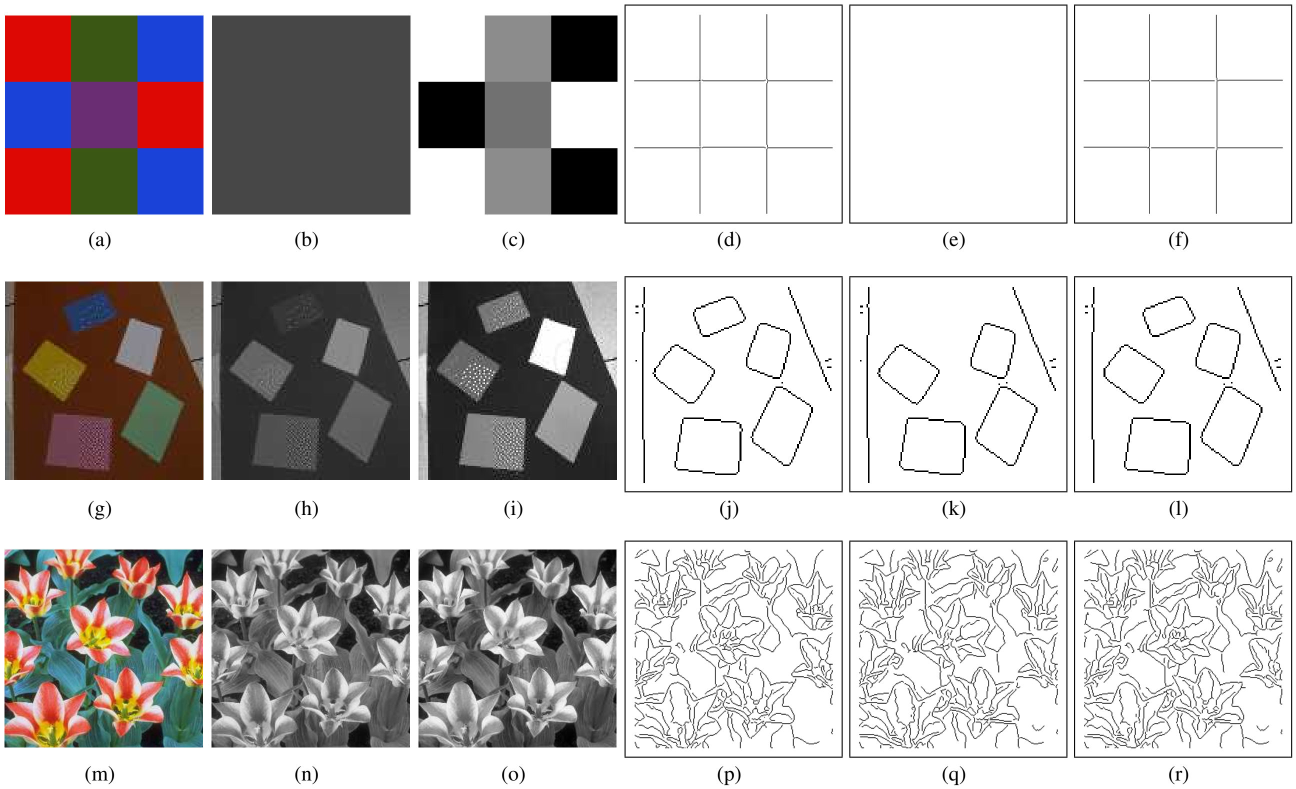

Fig. 1. The first column is the original images (Tile, Paper, Tulips.) The second column is the grayscale images. The third column is the grayscale images obtained by approximate principal component. The corresponding edges are in the fourth, fifth and sixth columns.

method avoids eigenvector decomposition by making use of power iteration. The conversion enables the edge detector to detect some edges of the grayscale image that are not detected using regular grayscale image. It offers a comparable performance with color edge detection without increasing the complexity threefold.

5. REFERENCES

[1] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, 1986.

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

[3] Aldo Cumani, "Efficient contour extraction in color images," in *ACCV '98: Proceedings of the Third Asian Conference on Computer Vision*, London, UK, 1998, vol. 1, pp. 582–589, Springer-Verlag.

[4] C. L. Novak and S. A. Shafer, "Color edge detection," in *Proc. Of DARPA Image Understanding Workshop*, 1987, pp. 35–37.

[5] A. Koschan and M. Abidi, "Detection and classification of edges in color images," *IEEE Signal Processing Mag.*, vol. 22, no. 1, pp. 64–73, January 2005.

[6] R. Nevatia, "A color edge detector and its use in scene segmentation," *IEEE Trans. Syst., Man, Cybernetics*, vol. 7, no. 11, pp. 820–826, 1977.

[7] G. Robinson, "Color edge detection," *Optical Eng.*, vol. 16, no. 5, pp. 479–484, September 1977.

[8] A. Shiozaki, "Edge extraction using entropy operator," *CVGIP*, vol. 36, pp. 1–9, 1986.

[9] J. Fan, W.G. Aref, M.S. Hacid, and A.K. Elmagarmid, "An improved automatic isotropic color edge detection technique," vol. 22, no. 13, pp. 1419–1429, November 2001.

[10] A. Koschan, "A comparative study on color edge detection," in *Proc. 2nd Asian Conf. Computer Vision-ACCV'95*, 1995, vol. III, pp. 574–578.

[11] R. Machuca and K. Phillips, "Applications of vector fields to image processing," vol. 5, no. 3, pp. 316–329, May 1983.

[12] S. Di Zenzo, "A note on the gradient of a multi-image," *CVGIP*, vol. 33, pp. 116–125, 1986.

[13] P.E. Trahanias and A.N. Venetsanopoulos, "Color edge detection using vector order statistics," vol. 2, no. 2, pp. 259–264, April 1993.

[14] P.E. Trahanias and A.N. Venetsanopoulos, "Vector order statistics operators as color edge detectors," vol. 26, no. 1, pp. 135–143, February 1996.

[15] J. Scharcanski and A.N. Venetsanopoulos, "Edge-detection of color images using directional operators," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 2, pp. 397–401, April 1997.

[16] Plataniotis K.N. Venetsanopoulos A.N. Zhu, S.-Y., "Comprehensive analysis of edge detection in color image processing," *Opt. Eng.*, vol. 38, no. 4, pp. 612–625, April 1999.

[17] Lloyd N. Trefethen and David Bau III, *Numerical Linear Algebra*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.