Implementation of Color-to-gray via Nonlinear Global Mapping

Srushti Kokare
(Portland State University -2019)

Abstract

Color images often have to be converted to grayscale for reproduction, artistic purposes, or for subsequent processing. This paper presents the re-implementation of the color-to-gray algorithm with non-linear global mapping. The implementation is a replication of the Robust Color-to-gray via non-linear Global Mapping paper. This method was able to produce convincing results in grayscale that preserves feature discriminability and color ordering. The goal was to understand the algorithm and reproduce the visual appearance of the color images in grayscale. Experiments were carried out on similar images as in the main paper. The results seem to work very good for some images but at the same time does worse for few images.

Introduction

Color images are computationally difficult because complexity of model increases with increase in number of colors of the image. On the other hand, the complexity of grayscale images is less than the color images. Grayscale image processing can be done on resource efficient systems which are most widely used in industrial systems. If more accurate features are extracted from gray images, they can be applicable to majority problems within less time.

Color to grayscale transformation has various applications in common tasks including black and white printing, single channel processing, non-photorealistic rendering with black and white media and amateur photography. A more common example is rendering color images to a monochrome device, or a color device in monochrome mode. This conversion is a dimensionality reduction problem where three dimensional (or more) RGB channel is converted to a single dimension.

Main problem with conversion into grayscale is to reproduce the intent of the color with its contrast in the original image. While transforming the color image to grayscale it is equally important to preserve the discriminability among the features. The goal is to produce images that are visually appealing. There will be some loss of information during the conversion, so we must aim to save as much information as possible from the color image.

There are some already existing methods for conversions divided into two categories: 1) Local Mapping 2) Global Mapping. The mapping of color pixels to gray pixels depends on the local distribution of colors in local mapping, it is spatially varying. Some color in the input image can map to different grayscale values in the output depending on the spatial surround. It helps restore chrominance details which were not considered in previous algorithms. Local Mapping conversion guarantees preservation of features accurately. While in global mapping the mapping of color to gray is same throughout the overall image pixels. There is a constant mapping between similar values to grayscale over the image. This global mapping conversion guarantees the homogenous conversion.

Related Work

There are several algorithms developed and implemented to solve the problem of color to gray conversion. One of the approaches I discussed in the introduction is the local and global mapping approaches. Local mapping proposed a high chromatic component to luminance [9]. There is different approach using image gradients [9]. This work mainly depends on the most of the previous approaches. Among all the approaches they have taken the non-linear mapping approach. The problems of linear mapping and global mapping are addressed in depth in the Kim, [8] et al. paper. I have tried to implement exactly similar to the Kim, et al. paper. My implementation refers some of the technique from the Smith's Apparent grayscale for the equations and formulas provided for the Helmholtz-Kohlrausch Effect. This work is based on the Nayatani's paper [1] that provides equations for converting CLE LUV color space to Helmholtz-Kohlrausch lightness predictor that is used in computing the color difference between pixels. The algorithm uses Nayatani's method without modification.

Design Objectives

I have set the same objectives from the Kim paper to preserve the visual appearance of a color image. They are as follows:

- Mapping Consistency: From the global mapping technique discussed above, every color pixel is mapped to same grayscale pixel
- Feature Discriminability: Local as well global features should be preserved in the gray image
- Ordering Preservation: Preserve the order of the original color
- Lightness fidelity: The variation in perception should remain same in the both the source and target image

Algorithm

1) Non-Linear Global Mapping

The paper [8] proposes an algorithm that uses global mapping between color-to-gray images. This non-linear mapping uses the lightness, chroma and hue angle of the input image to prevent inhomogeneous conversions of constant color regions. The equation for global mapping is given by:

$$g(x,y) = L + f(\Theta) * C \qquad (1)$$

Where g(x,y) is a simple nonlinear mapping of a color value at a pixel (x,y), L is lightness, *f(Θ) is hue angle*, C is chroma. These values are obtained from CIE LCH color space. These equations are further solved with many small equations which we will discuss further. The hue angle is defined as:

$$f(\Theta) = \sum_{k=1}^{n} (A_k \cos k\Theta + B_k \sin k\Theta) + A_0 \qquad (2)$$

A_k, B_k, A_0 are unknown parameters that we will solve in energy optimization. The local differences are produced similarly in color and grayscale values to distinguish features, calculated by *f(Θ)*. They have used image gradients to represent local differences. As a result, it was necessary to define the measurements of the color differences in 1D, since color is 3D and grayscale is 1D. A normalized distance between two colors in CIE L*a*b* colorspace was used for measurement; however conversion color to gray the relationship cannot be preserved. The value of chroma C influences lightness fidelity to adjust changes in lightness.

2) Energy Function

They have used the energy function to minimize the image gradients between the color and grayscale pixel values. The equation is given as:

$$E_s = \sum_{(x,y) \in \Omega} \| \nabla g(x,y) - G(x,y \|^2 \qquad (3)$$

where *(x,y)* is a pixel in Ω and ∇g is a gradient between pixels described as:

$$\nabla g(x,y) = (g(x+1,y) - g(x-1,y), \\ g(x,y+1) - g(x,y-1))$$

and G describes the difference between input colors c, defined by:

$$G(x,y) = \begin{pmatrix} G^x & (x,y) \\ G^y & (x,y) \end{pmatrix} T = \begin{pmatrix} c(x+1,y) & \ominus & c(x-1,y) \\ c(x,y+1) & \ominus & c(x,y-1) \end{pmatrix} T$$

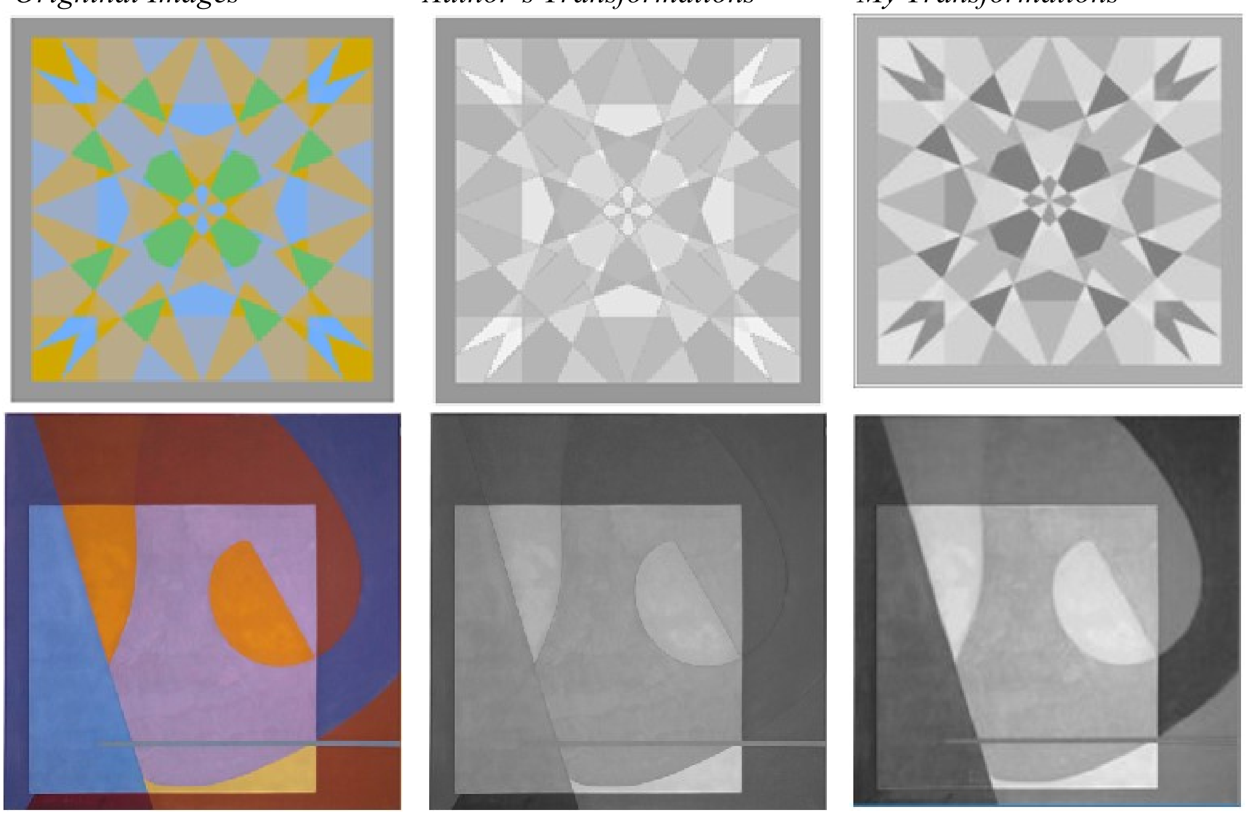
The color difference operator $\ominus$ is defined by,

$$c_i \ominus c_j = sign(c_i,c_j) \sqrt{\Delta Lij^2 + \alpha \left(\frac{\sqrt{\Delta a^*_{ij} + \Delta b^*_{ij}}}{\Re} \right)^2} \quad (4)$$

where c is represented by CIE L*a*b, $\Re$ is the normalization constant which is $\Re$ = 2.54 * √2 which acts to equalize chromatic contrast. Alpha parameter is set to 1.0 that controls the influence of chromatic contrast. The sign function preserves the relative color ordering between two pixels. For this implementation, I used the equations from Apparent Grayscale given in [Smith et al. 2008]. To calculate the sign of the value H-K effect predictor is given higher priority i.e. ΔL^{HK}.Helmholtz-Kohlrausch Effect: “A chromatic stimulus with the same luminance as a white reference stimulus will appear brighter than the reference” [5]. The calculations are based on Nayatani’s Variable Achromatic Color (VAC) approach there is a conversion between the achromatic luminance to the color stimulus. This a more common adjustment method used. However if $\Delta L^{HK}$is zero, we look next to the sign of ΔL, which if also is zero, we use the sign of $\Delta L^3 + \Delta a^{*3} + \Delta b^{*3}$ [1].

The coefficient to adjust the adaptive luminance dependency is given as:

Orignal Images *Author’s Transformations* *My Transformations*



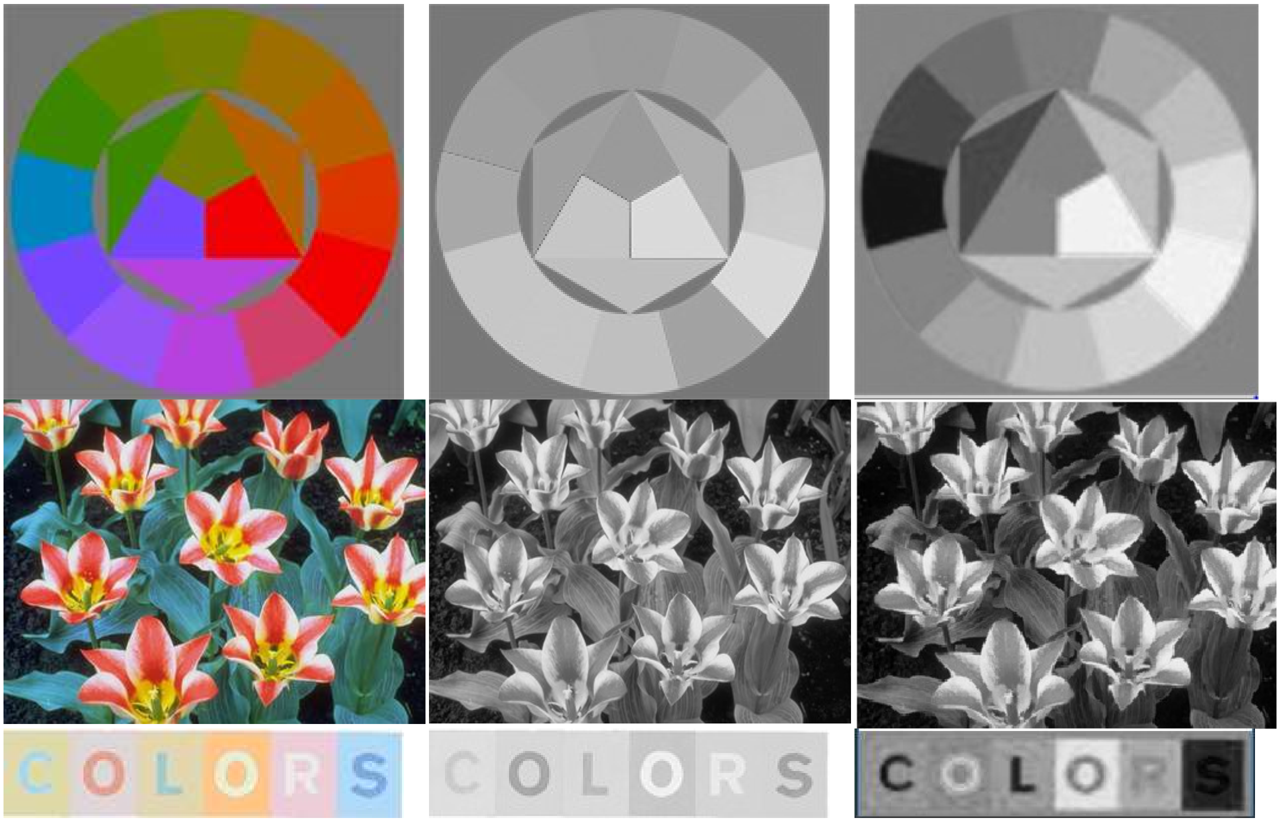


Figure ***1****Visual Comparison* ***:*** *Column 2 has results from author's paper. Column 3 shows my results*.

Origninal Images | *My Transformations* | *Reference Transformations (open CV)*

Figure 2: *Comparison of my conversions with OpenCV's conversions of color to gray*

$$K_{Br} = 0.2717\frac{6.469 + 6.362L_a^{0.4495}}{6.469 + L_a^{0.4495}}$$

Where u',v' is test chromaticity, u,v is chromaticity of reference white and La is adapting luminance which is set to constant value 20 as given in the Nayatani VAC model.
Saturation of the test chromatic light is given as:

$$s_{uv} = 13[(u' - u'_c)^2 + (v' - v'_c)^2]^{\frac{1}{2}}$$

Equation for theta and function of predicting the change of the *H-K* effect in different hues is given as:

$$\theta = tan^{-1}\frac{v' - v'_c}{u' - u'_c}$$

$$q(\theta) = -0.01585 - 0.03017\cos\theta - 0.04556\cos 2\theta - 0.02667\cos 3\theta - 0.00295\cos 4\theta + 0.14592\sin\theta + 0.05084\sin 2\theta + 0.14592\sin\theta + 0.05084\sin 2\theta - 0.01900\sin 3\theta - 0.00764\sin 4\theta$$

3) Optimization

The energy equation given in the equation (3), is reduced into a quadratic equation of unknown parameters,

$$f(\theta) = t^{\mathrm{T}}\mathrm{x}$$

This is further manipulated as,

$$E_s = \mathrm{x}^{\mathrm{T}}\mathrm{M}_s\mathrm{x} - 2b_s^T\mathrm{x} + C$$

Where:

$$\mathrm{M}_s = \sum_{\Omega}(\mathrm{uu}^2 + vv^2)$$

$$b_s = \sum_{\Omega}(pu + qv)$$

$$u = (C.t)_x$$

$$v = (C.t)_y$$

$$p = G^x - L_x$$

$$q = G^y - L_y$$

The value of n was decided to be 4 after performing various experiments. I have chosen the value of n as 4 in my implementation as well. The value of n corresponds to different bases of cosine and sine. The higher of values did not produce any significant change in the results. I also experimented with several larger values of n and found out no change in the results. The matrix M_s is a 9 X 9 matrix and b_s is a 9X1 vector from the paper.

The energy equation is regularized with λ term which improves the visibility of range when the M_s matrix is null or when the value of x exceeds beyond visible range. After adding the energy term,

$$E_{image} = E_s + \lambda E_r$$

that is minimized as,

$$\hat{x}_{image} = (M_s + \lambda I)^{-1} b_s$$

Results

I have implemented the algorithm in Python 3.6 using OpenCV libraries. The images are run sequentially one after the other. I have made use of python packages numpy, cv2, skimage and matplotlib. I tested my implemented on a 64-bit operating system, x64 based processor. I used the Pycharm Integrated IDE for code development and debugging. The processor used to run the experiments is the Intel® Core(TM) i7-6700 HQ CPU@ 2.60 GHz along with a 8.00 GB RAM. The computation time is proportional to the number of pixels in the image. As the size of the image increases there is more time required to compute the result.

The parameters in the algorithm are maintained as same in the Kim's paper [8]. There is absolutely no change.

The results given in the fig. 1 shows the comparison of results with the output results from the Kim's paper [8] with my results. In some images my results are able to distinguish very well between the darker pixels of the images than the main paper's results. But at the same time, the images did not properly follow the principle of lightness fidelity. The luminance was low in my results and therefore some lighter colors turned out to be darker in the output of my results.

I have done a comparison between the results of my implementation with results obtained from the OpenCV color-to-gray conversion. My results did quite well in comparison with openCV, the darker colors were visually more appealing in my results. These results are shown in the fig 2. Overall, I would say that my implementation obtained descent results but not too good.

Discussion and Future Work

I followed same algorithm steps as described by the Kim's paper. But there is a difference in their implementation and test execution environment. The authors have used c++ for coding along with the OpenCV. They have also run their tests in a parallel system with OpenMp. Whereas in my implementation I have used Python 3.6 which is slower to execute than c++. And also because i ran my execution sequentially, the author's computation speed could be definitely higher than mines. I had a intention to create a lightweight application for which I choose to use python. Python was a still a good choice for implementation in terms numpy library that supports matrix multiplications in less lines of code.

During my work, I came across various challenges. First thing is reading the input image from the cv2 library method. I read the image with a IMREAD_COLOR flag and started to run the algorithm. The results started to look very dark and lightness of the image was lost. The read image was a distortion in color and had a blue color spread across it. After debugging the problem, I found out that it was required for the image to convert from BGR format into RGB format.

There was a more saturation of colors observed when image was read with cv2 library and displayed with matplotlib. The pixels were darker and cannot be distinguished. After solving these issues, I was able to get the above results in the paper. They seem not to properly preserver the color ordering and the light pixels get darker in the grayscale. This is one of the things I observed could be worked on in future. From my observations the results have mostly preserved the global mapping but lost the local features. This observation was also seen in the Kim's paper.

I would study some techniques to reproduce these images into good grayscale conversions and implement the video conversion as a future work.

Acknowledgements

I would acknowledge the author of the paper for the algorithm they have presented. Along with this also like to acknowledge Prof. Feng Liu for his guidance.

References

[1] NAYATANI Y. : Simple estimation methods for the Helmholtz-Kohlrausch effect. Color Res. Appl. 22 (1997), 385–401.

[2] NAYATANI Y.: Relations between the two kinds of representation methods in the Helmholtz-Kohlrausch effect. Color Res. Appl. 23 (1998), 288–301.

[3] NOWAK R. D., BARANIUK R. G.: Adaptive weighted highpass filters using multiscale analysis. IEEE Transactions on Image Processing 7, 7 (1998), 1068 – 1074.

[4] NEUMANN L., CADIK M., NEMCSICS A.: An efficient perception-based adaptive color to gray transformation. In Proceedings of Computational Aesthetics 2007 (Banff, Canada, 2007), Eurographics Association, pp. 73– 80.

[5]SMITH, K., LANDES, P.-E., THOLLOT, J., AND MYSZKOWSKI,K. 2008. Apparent greyscale: A simple and fast conversionto perceptually accurate images and video. Computer GraphicsForum (Proc. Eurographics 2008) 27, 2, 193–200.

[6] GOOCH, A. A., OLSEN, S. C., TUMBLIN, J., AND GOOCH, B.2005. Color2gray: salience-preserving color removal. ACM Trans. Graphics (Proc. SIGGRAPH 2005) 24, 3, 634–639

[7] https://github.com/huis-clos/color2gray

[8] KIM, Y, ET AL. 2009. Robust color-to-gray via nonlinear global mapping. ACM SIGGRAPH Asia. 28, 5.

[9] BALA, R., AND ESCHBACH, R. 2004. Spatial color-to-grayscale transform preserving chrominance edge information. In Proc. Color Imaging Conference 2004, 82–86.

[10]CAD´IK, M. 2008.Perceptual evaluation of color-to-grayscale image conversions. Computer Graphics Forum (Proc. Pacific Graphics 2008) 27, 7, 1745–1754.

Appendix
Additional Results

Original Image

Transformed Image