INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY

HYDERABAD

# Digital Image Processing

Monsoon 2018

## Final Project Report



# Color Transfer Between Images

By
Surya Soujanya Kodavalla
201530140
Project no. 34

Mentor
Ashwin Pathak

# Table of contents

# Introduction

## Problem statement

One of the most common tasks in image processing is to alter an image's color. Often this means removing a dominant and undesirable color cast, such as the yellow in photos taken under incandescent illumination. The issue addressed here a is more general form of color correction that borrows one image's color characteristics from another.

## Motivation

Creating a method to get more generalized and desirable results, even with vastly different images.

## Overview

The aim is to get more generalized and desirable results while transfering color from the source image to the target image. To do this, a color space with relatively independent axes is considered. The lαβ color space is considered for this purpose. Both the source and target images are converted to this color space initially. All the operations on the images are conducted in the lαβ color space itself and then converted back to RGB space again. Simple operations are applied on the images using their standard deviations and means.

# Approaches considered

## Method 1

The goal is to do color transfers with a simple algorithm. The important part is to choose a suitable color space and then apply simple operations there. Usually, algorithms in computer graphics and computer vision benefit a lot with the appropriate choice of color space. When a typical three channel image is represented in any of the most well-known color spaces, there will be correlations between the different channels' values. This implies that if we want to change the appearance of a pixel's color in a coherent way, we must modify all color channels in tandem. This complicates any color modification process. What we need is an orthogonal color space without correlations between the axes. Little correlation between the axes lets us apply different operations in different color channels with a guarantee that undesirable cross channel artifacts won't occur.

Ruderman et al. developed a color space, called lαβ, which minimizes correlation between channels for many natural scenes. This space is based on data-driven human perception research that assumes the human visual system is ideally suited for processing natural scenes. This is the color space known to have least correlation among the axes hence it will be used for the color transfer between images.

Since the main focus is on RGB images, first a method needs to figured out to convert images from the RGB space to the lαβ space and vice versa.

lαβ is a transform of LMS cone space. We first convert the image to LMS space in two steps. The first is a conversion from RGB to XYZ tristimulus values. This conversion depends on the phosphors of the monitor that the image was originally intended to be displayed on. Because that

information is rarely available, we use a device-independent conversion that maps white in the chromaticity diagram (CIE xy) to white in RGB space and vice versa. The standard conversion matrix accordingly modified. So we convert the image from RGB to lαβ in 4 steps-

1. RGB -> XYZ
2. XYZ -> LMS
3. Take log of L,M and S channels giving L',M' and S' respectively
4. L'M'S' -> lαβ

We follow the same procedure, in reverse and use exponents, to convert an image from lαβ to RGB space after processing it for the color transfer. Here are a few examples of conversion of images from one color space to another.

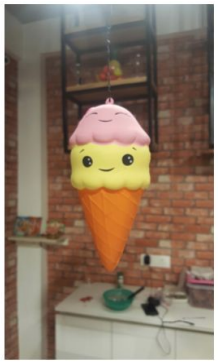
Input image

RGB to lαβ

lαβ to RGB

Input image

RGB to lαβ



lαβ to RGB

Now coming to the part of color processing, the goal is to make one image take on another images' look and feel. This implies that some aspects of the distribution of data points in lαβ space to transfer between images. For this, we just need the mean and standard deviations along each of the three axes. These values are computed for each axis, individually, in both the source and target images.

First, subtract the means of the target image from the data points of the target image.

$$l^* = l - \langle l \rangle$$
$$\alpha^* = \alpha - \langle \alpha \rangle$$
$$\beta^* = \beta - \langle \beta \rangle$$

Then, scale the data points comprising the synthetic image by factors determined by the respective standard deviations.

$$l' = \frac{\sigma_t^l}{\sigma_s^l} l^*$$
$$\alpha' = \frac{\sigma_t^\alpha}{\sigma_s^\alpha} \alpha^*$$
$$\beta' = \frac{\sigma_t^\beta}{\sigma_s^\beta} \beta^*$$

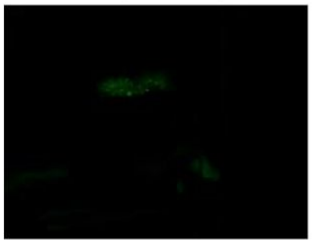
Next, instead of adding the averages that we previously subtracted, add the means computed for the source image. And as noted earlier, we convert the image from lαβ to RGB space by reversing the steps and using exponents instead of taking log. Here are a few examples, with outputs shown at each step of the process -

Source

Target

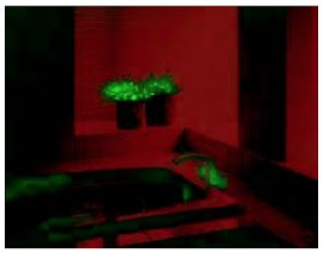
Source in lαβ

Target in lαβ

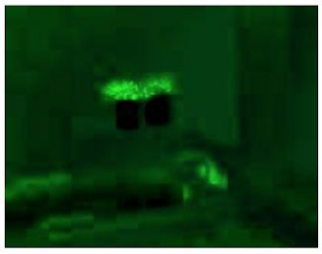


Subtracting the means

Scaling with ratio of standard deviations



Adding the means



Final output with color transfer from source to target
after converting from lαβ to RGB

### Method 2

Another alternative and more intuitive method would be to match the histograms that can be computed along each axis. Histogram matching begins by computing the histograms of both the source and target images. Next, the cumulative histograms are computed. The histogram of the source is then inverted. Pixels values in a particular channel in the target image are then converted by first looking up their corresponding frequency in the cumulative target histogram. This frequency is then mapped to a different pixel value by applying the inverted cumulative source histogram.

## Work Performed

### Phases of the project

The project was done in three different phases.

Initially, the source and target were converted into the CIE Lab color space from RGB instead of the prefered lαβ color space for convenience. Then the color processing part was coded. The

color processing was carried out on the images in the CIE Lab color space itself. After being color processed, the images were converted back to the RGB color space.

In the next phase, the images were carefully converted into the lαβ color space before being color processed.

The third phase included studying methods for better color transfer from various sources other than just the paper and modifying the previously implemented color processing method to a more generalized form to be able to get better results with less compatible images. This phase is discussed more in the failure cases and the discussion sections of this report.

## Experiments

Multiple images were processed and experimented with throughout each phase to get a better understanding of the transfer of colors from the source to the target. Based on these results, a better and generalized code was built. Both photographs and images from the internet and photographs from real life were used in the experimentation process.

## Software programming

The entire project was coded in MATLAB. There are multiples codes though, one with the basic method and then the generalizations and rectifications for failure cases.

# Results

The result's quality depends on the images' similarity in composition.

## Successes

More the similarity between the target and source images, better is the quality of the obtained result.

## Failure cases

This algorithm extracts a color from source image and then covers a monochrome layer of this color to the target image. It doesn't necessarily work as expected for all images, especially as the difference between the composition of the images increases. In particular, if the composition of the source and target images is very different, the algorithm may fail. In general, transformations whereby the source image has a fairly small color palette (such as night images, sunsets, and everything with a limited number of different colors) tend to work well.

## Comparison across color spaces

The algorithm can be carried out on the images in various color spaces. But the results are better when conducted in lαβ color space due to the relatively independent axes.


Comparison across color spaces

# Discussion

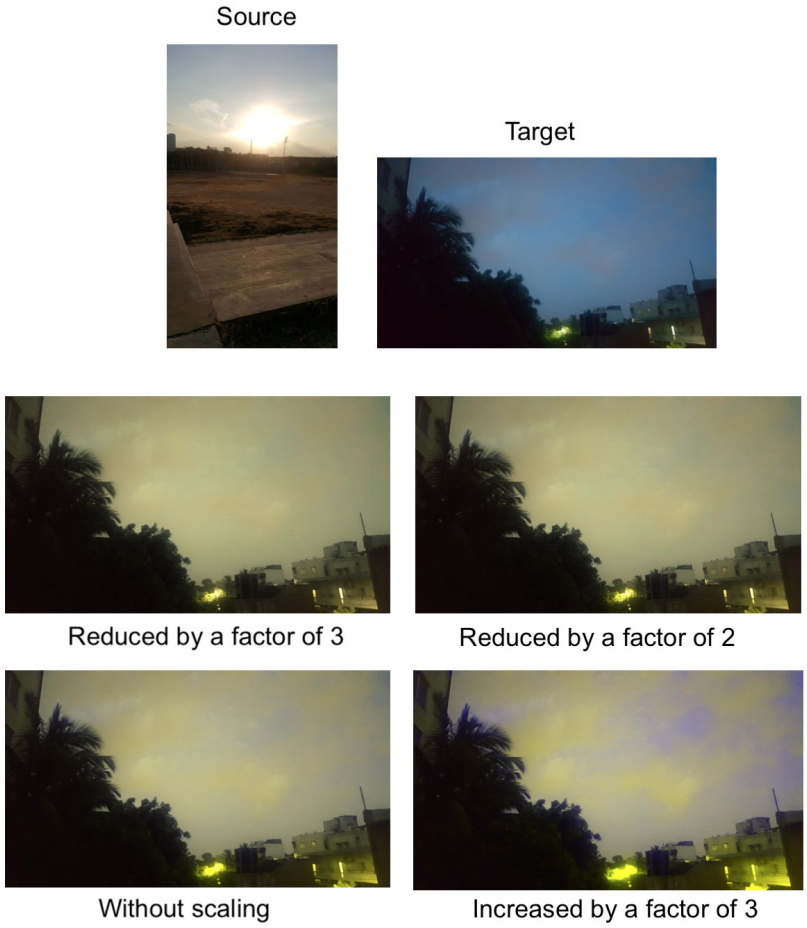
## Failure cases and rectification

To get better results with the failure cases, here are few methods that can be implemented -

- **Swatches** - If the source image has more grass and the target image has more sky in it, then the transfer of statistics will most probably fail and will not give desired results. To overcome this issue, select separate swatches of grass and sky and compute their statistics, leading to two pairs of clusters in lαβ space (one pair for the grass and sky swatches). Convert the whole rendering to lαβ space. Scale and shift each pixel in the input image according to the statistics associated with each of the cluster pairs. Compute the distance to the center of each of the source clusters and divide it by the cluster's standard deviation. This division is required to compensate for different cluster sizes. Blend the scaled and shifted pixels with weights inversely proportional to these normalized distances, yielding the final color. This approach can be applied to images with more than two clusters as well.
- **Higher moments** - Another possible extension would be to use higher moments such as skew and kurtosis, which are respective measures of the lopsidedness of a distribution and of the thickness of a distribution's tails. Imposing such higher moments on a second image would shape its distribution of pixel values along each axis to more closely resemble the corresponding distribution in the first image. While it appears that the mean and standard deviation alone suffice to produce practical results, using higher moments would give outputs of greater clarity. But the power transform and modulus transform are used to adjust the skewness and kurtosis of source image, make them more similar to the

higher moments' distribution of target image and generate the better result of color transfer.

- **Scaling** - In some cases, nudging some of the statistics in the right direction can sometimes make the result more visually appealing. Directly applying the method might result in a corrected image with too much or too less of a color in it. Altering the the standard deviation of the required channel (achromatic, yellow–blue or red–green) by a factor using trial and error, outputs with prefered appearances can be obtained. The scale value determines what color is preserved. A big value implies that the source image's color counts a lot and it will combine with the target color. The smaller the value, the more similar the color is to the pure color.




Scaling the yellow-blue channel

## Applications and Future Directions

Applications of this method of color transfer can range from subtle post-processing on images to improve their appearance to more dramatic alterations, like converting a daylight image into a night scene. We can modify the method being used here to make it more suitable to the required outputs, like using the range of colors measured in photographs to restrict the color selection to ones that are likely to occur in nature.


Image taken in the day made to look like a night time picture

The color transfer algorithm can be applied to images which are superimposed on backgrounds to make the blending look better.

This method of color transfer between images can be expanded over various images and also can be combined with several other applications and concepts to get amazing results.

For example, to recolor only a part of the image, like a flower, edge detection or even face detection can be done and then only that part can undergo color processing and exclude the background out of the process domain.

Application of color transfer need not be necessarily limited to images. It works even on videos. Based on the similarity of video frames, the basic algorithm of color transfer is applied. The methods talked about for improving the results can also be implemented on videos.

Another interesting application is the transfer of style and color from source image to the target image. The style of a particular painting or image can be transferred to another target image and combining this with color transfer gives fascinating results.

Color transfer can also be used to apply color to black and white images (grayscale images). Though it doesn't give extremely satisfying outputs, the image can be changed from grayscale to having a different look with color.

The opposite of converting color image to grayscale using the luminance channel of the Lαβ color space can also be done.

Also, looking at color transfer has given the comparison between color spaces and why lαβ is better. This can be used for other applications in color processing as well.

Another important aspect of this method is that it is important in augmented reality applications where real environments are merged with computer-generated data. To make the computer-generated elements mesh with the real scene, a fast algorithm is required to account for mismatches in the choice of surface materials as well as differences in the lighting. While the color transfer algorithm cannot account for missing or wrongly placed shadows or other light interactions between the real and virtual components, the computer-generated elements can be adjusted to have the same color statistics, which makes them fit in the scene better.

# Github Link

https://github.com/solareclipse27/DIP_Color-Transfer-between-Images

# Task Assignment

The project had three main tasks - the basic color processing, proper conversion from one color space to another and implementing a method to get better results.

Initially, the source and target were converted into the CIE Lab color space from RGB instead of the prefered lαβ color space for convenience. Then the color processing part was coded. The color processing was carried out on the images in the CIE Lab color space itself. After being color processed, the images were converted back to the RGB color space.

The next task was converting the images into the lαβ color space before being color processed and back into RGB color space to view the results.

For the third task, different methods were studied for better color transfer from various sources other than just the paper and modifying the previously implemented color processing method to a more generalized form to be able to get better results with less compatible images.

# References

[1] Smriti Kumar, Ayush Swarnkar, “Colorization of Gray Scale Images in lαß Color Space Using Mean and Standard Deviation”, *2012 IEEE Students’ Conference on Electrical, Electronics and Computer Science.*

[2] Daniel L. Ruderman, Thomas W. Cronin and Chuan-Chin Chiao, “Statistics of cone responses to natural images: implications for visual coding”, *J. Opt. Soc. Am. A/ Vol. 15, pp. 8, August, 1998.*

[3] Erik Reinhard, Michael Ashikhmin, Bruce Gooch, and Peter Shirley, “Color Transfer between Images”, *University of Utah, 2001.*

[4] *http://www.ee.oulu.fi/~gyzhao/research/color_transfer.htm*

[5] Color Imaging: Fundamentals and Applications, *1st Edition*, Erik Reinhard, Erum Arif Khan, Ahmet Oguz Akyuz, Garrett Johnson.

* The pictures used are either from google images or are originals.