Digital Image Processing

Color Transfer Between Images

By -
Surya Soujanya Kodavalla
201530140
Project no. 34

Mentor-
Ashwin Pathak

Aim

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. The aim is to get more generalized and desirable results while transfering color from the source image to the target image.

Basic Pipeline

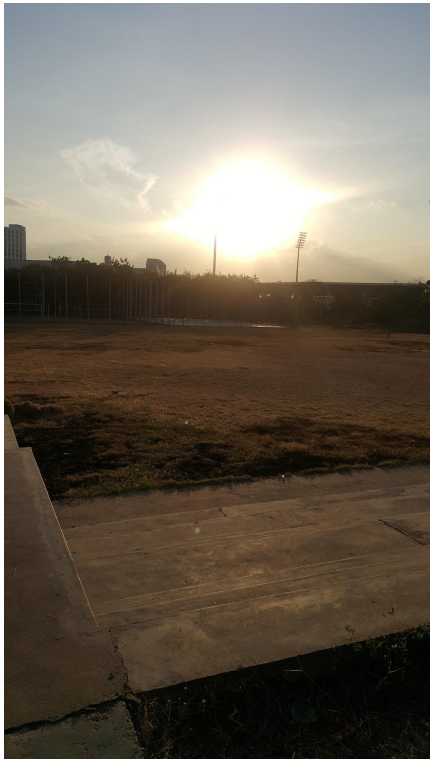
Convert target and source to lαβ color space.

Get means and standard deviations of each channel of source and target.

Carry out operations on the target image from the values obtained.

Convert target image back to initial color space (with new values).

Results



Source

Target

Output

Method

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.

Why lαβ?

Ruderman et al. developed a color space, called lαβ, which minimizes correlation between channels. This space is based on data-driven human perception research that assumes the human visual system is ideally suited for processing natural scenes. To convert an RGB image to lαβ color space, the steps are as follows-

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.

RGB -> lαβ

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3811 & 0.5783 & 0.0402 \\ 0.1967 & 0.7244 & 0.0782 \\ 0.0241 & 0.1288 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\mathbf{L} = \log L$$

$$\mathbf{M} = \log M$$

$$\mathbf{S} = \log S$$

$$\begin{bmatrix} l \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 0.5774 & 0.5774 & 0.5774 \\ 0.4082 & 0.4082 & -0.8165 \\ 0.7071 & -0.7071 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{L} \\ \mathbf{M} \\ \mathbf{S} \end{bmatrix}$$

lαβ -> RGB

$$\begin{bmatrix} \mathbf{L} \\ \mathbf{M} \\ \mathbf{S} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -2 & 0 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{6}}{6} & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} l \\ \alpha \\ \beta \end{bmatrix}$$

$L = 10^{L}$

$M = 10^{M}$

$S = 10^{S}$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 4.4679 & -3.5873 & 0.1193 \\ -1.2186 & 2.3809 & -0.1624 \\ 0.0497 & -0.2439 & 1.2045 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix}$$

RGB -> Iαβ -> RGB

Original

Iαβ

RGB

Color Processing

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

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

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

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

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

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

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

Results

Source

Target

Output

Comparison between color spaces

Source

Target

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 between color spaces

RGB

Iαβ

CIECAM97s

Analysing Results

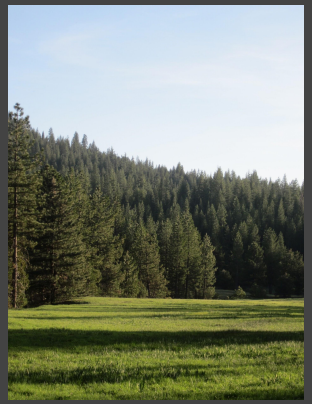
- **Successes**

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.

Incompatible images:



Weird output:

Improvising for better results

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

- **Swatches**
  - **Method** - For incompatible images the transfer of statistics will most probably fail and will not give desired results. To overcome this issue, select separate swatches and compute their statistics, leading to different clusters in lαβ space. 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.
  - **Issues in implementation** - Was not able to divide the image into swatches.

Improvising for better results

- **Higher moments**
  - **Method** - Using higher moments such as skew and kurtosis would give outputs of greater clarity. Imposing such higher moments on an image would shape its distribution of pixel values along each axis to more closely resemble the corresponding distribution in the first image.
  - **Issues in implementation** - Was not able to apply proper transform to adjust the skewness. 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.

Improvising for better results

- **Scaling**
  - **Method** - 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.

Results with different scaling factors

Source

Target

Results with different scaling factors

Scaling the yellow-blue channel by 2

No scaling

Scaling the red-green channel by 2

Applications and Future Scope

- Applications of this method of color transfer can range from subtle post-processing on images to improve their appearance to more dramatic alterations.
- The color transfer algorithm can be applied to images which are superimposed on backgrounds to make the blending look better.
- To recolor only a part of the image, 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.
- Another important aspect of this method is that it is important in augmented reality applications where real environments are merged with computer-generated data.

Thank you !!