Programming Assignment 2: Color Transfer Between Images

Student: Meltem Kaya

Student Id: 21827555

Overview of The Problem

One of the most crucial visual components of an image is color. A common approach in digital image processing known as "color transfer" involves transferring the color distribution of one target picture to another such that the altered target image might have a color distribution resembling the source image. Color transfer is a useful technique to change the look of a source image according to the color pattern of a target image. In this assignment, we will implement the color transfer method of Reinhard et al. and analyze RGB histograms of input and output images. After applying color transfer method, we will try to improve results by transfering the color between the similar regions of source and target images experimentally.

The Solution Approach

First, I read source and target files and convert their color spaces from RGB to LAB using the equations. After that I calculated mean and standard deviations of each channels l, α, β. To transfer color of target source onto source image, I made some calculations as stated below:

Substract mean values from each channel of source image.

> l* = l_source - mean(l_source) α* = α_source - mean(α_source) β* = β_source - mean(β_source)

Then i calculated transferred color image result using the equation:

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

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

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

PART 1. Experimental Results on Color Convertion

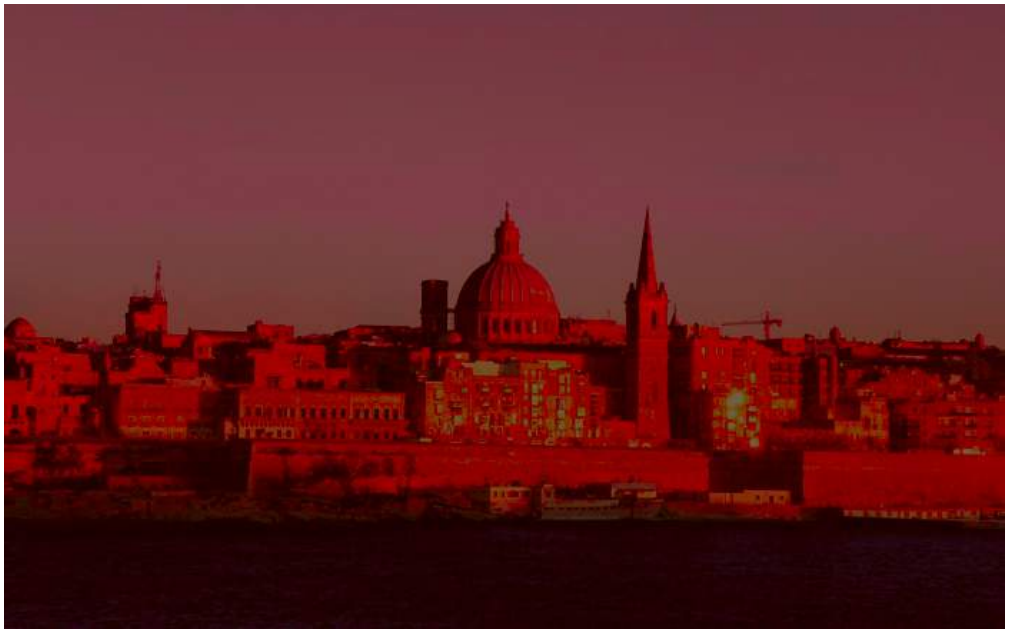
1. Applying OpenCV CIE LAB **vs** Implementing lαβ conversion as stated in referenced pdf [1]

When i either applied OpenCV CIE LAB or lαβ conversion applying myself as stated in referenced pdf **[1]**, for many inputs, the L*a*b and the L-alpha-beta implementations achieve similar quality images, even though there may be slight differences. However, My experiment resulted in some instances have been found where the L*a*b implementation produces a poor quality image but the L-alpha-beta implementation does not and closer to expected output images.

Here are some positive samples:

1.1.Sample result image: res_01

1.1.1.OpenCV COLOR_RGB2LAB



1.1.2.lαβ conversion as stated in referenced pdf

1.1.3 Expected output/result image

1.2.Sample result image: res_56

1.2.1.OpenCV COLOR_RGB2LAB

1.2.2.lαβ conversion as stated in referenced pdf

1.2.3 Expected output/result image

1.3.Sample result image: res_29

1.3.1.OpenCV COLOR_RGB2LAB

1.3.2.lαβ conversion as stated in referenced pdf

1.3.3 Expected output/result image

Since the results of the implementation of lαβ conversion as stated in referenced pdf gives closer results comparing to OpenCV CIE LAB, I continued using lαβ conversion which I implemented.

PART 2. Experimental Results on Handling Outliers

After applying color transfer methods, I got some outlier values such as more than 255, less than 0. Since RGB color components are standardly defined on a scale from 0 to 255, I have to handle outliers to obtain smooth results which are outside the range of [0,255]. I tried two different methods in my experiment to get better results.

> **1.** Min-Max scaling which scales images between 0 and 255.

> **2.** Setting outliers to nearest treshold. For example,If outlier value is bigger than 255, set to 255. Likely if outlier value is less than 0, set to 0.

Let's see the results of these two approaches on some samples.

2.1.Sample result image: res_01

2.1.1.Unhandled Tresholds/None of techniques are applied on

2.1.2.After Min-Max Scale

2.1.3.After setting outliers to nearest treshold

2.1.4 Expected output/result image

2.2.Sample result image: res_29

2.2.1.Unhandled Tresholds/None of techniques are applied on

2.2.2.After Min-Max Scale

2.2.3.After setting outliers to nearest treshold

2.2.4.Expected output/result image

2.3.Sample result image: res_08

2.3.1.Unhandled Tresholds/None of techniques are applied on

2.3.2.After Min-Max Scale

2.3.3.After setting outliers to nearest treshold

2.3.4.Expected output/result image

As visualized in resulted images after different approaches, min-max scale appraoch makes some differences positively but also reduces the luminosity of most of the images. I continued with setting outliers to nearest treshold because that approach enhanced my result quality and not to cause darker images unlike min-max scaling.

Part 3.Dividing the source and target images to equals rectangular regions and transfer colors between the most similar regions between source and target images

In this part, I used NCC algorithm to find similar regions. I separated images into 4 parts, actually that is enough to see differences whether the image quality enhanced or not. In general, as we can see in histogram equations of images below there are two main reason of distortion of images.

1. The difference between target and source image compositions
2. Since the approach of Reinhard et al. depends on global color statistics, large patches with comparable pixel intensity values significantly affect overall color transfer.
3. Without using deep learning techniques we cannot obtain expected results. Color transferring is not as much as easy as we do some calculations on.

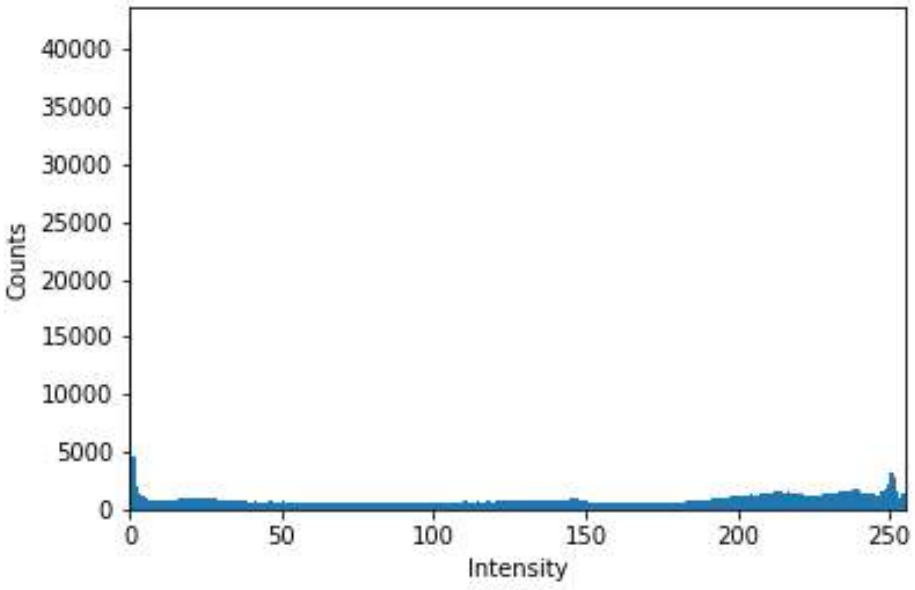
Let's see some examples:

Sample 1:

Sample one is partially closer to expected result. Because the target and source images have similar compositions. There is a car in the middle of the target image and same as on the source image. That is why we get a good result. But as we can see from the histogram equations of inputs, we have little differences of intensity. That causes the lower luminosity on the output Image. On the other hand, when we cropped and then applied color transferring method, We obtain better result at the bottom of the picture, because intensity values of these part are closer.

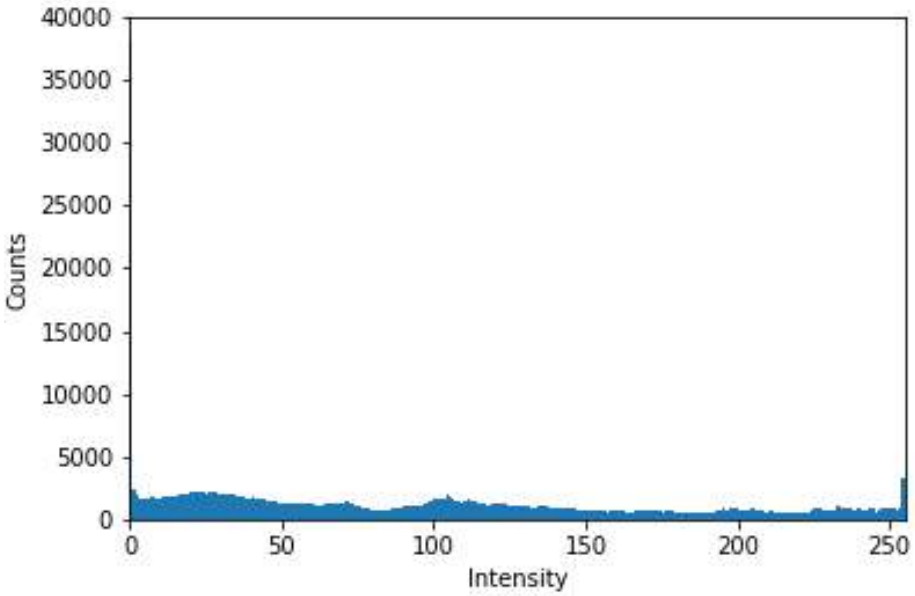
3.1.1.Source Image

Histogram of Source Image



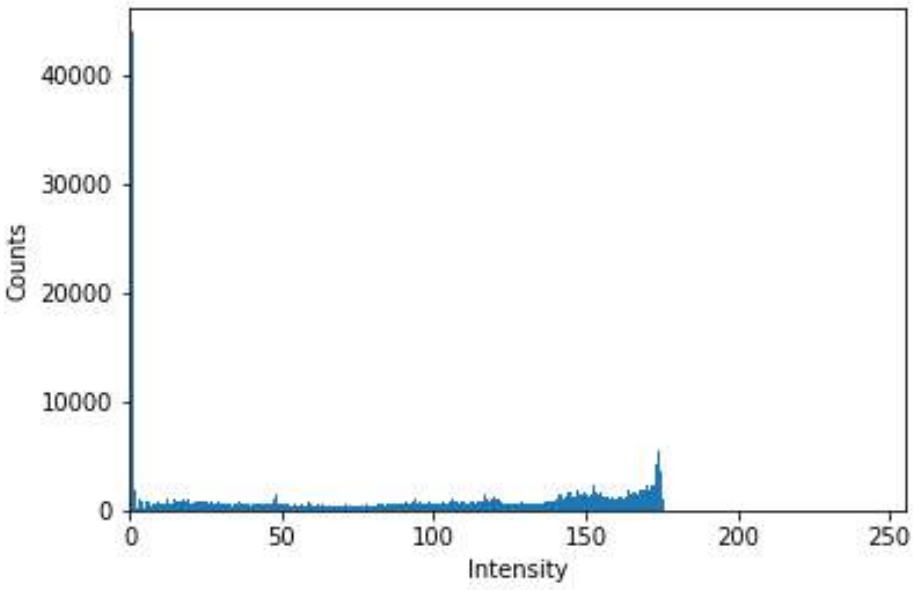
3.1.2.Target Image

Histogram of Target Image



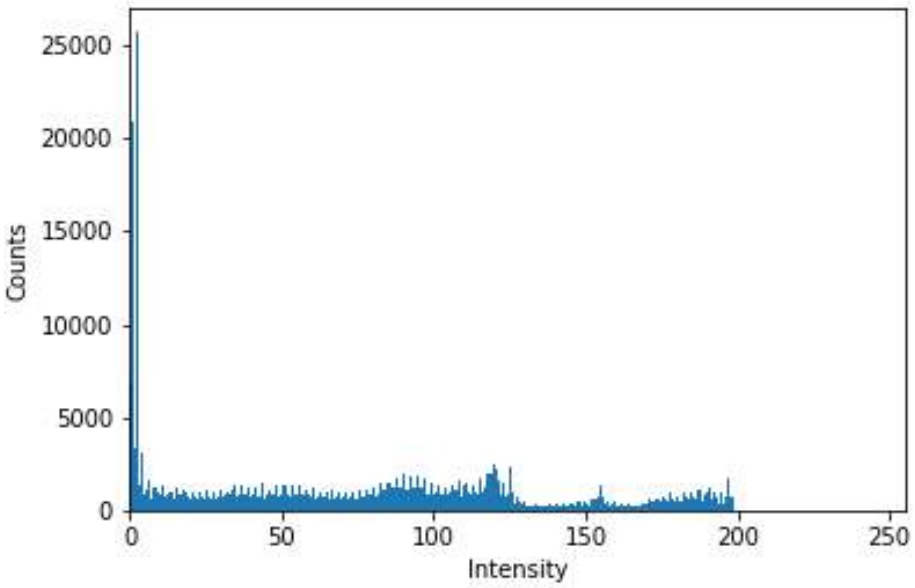
3.1.3.My Result After Color Transfer

Histogram of My Result After Color Transfer Image



3.1.4.My Result Image After Dividing Input Images, Then Color Transfer

Histogram of M Result Image After Dividing Input Images, Then Color Transfer



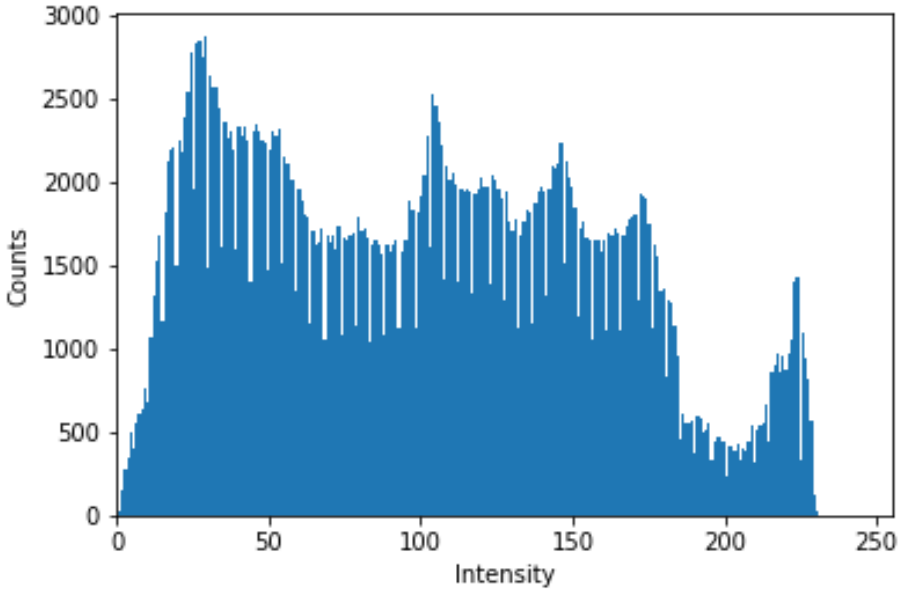
3.1.5.Expected output/result image

Sample 2:

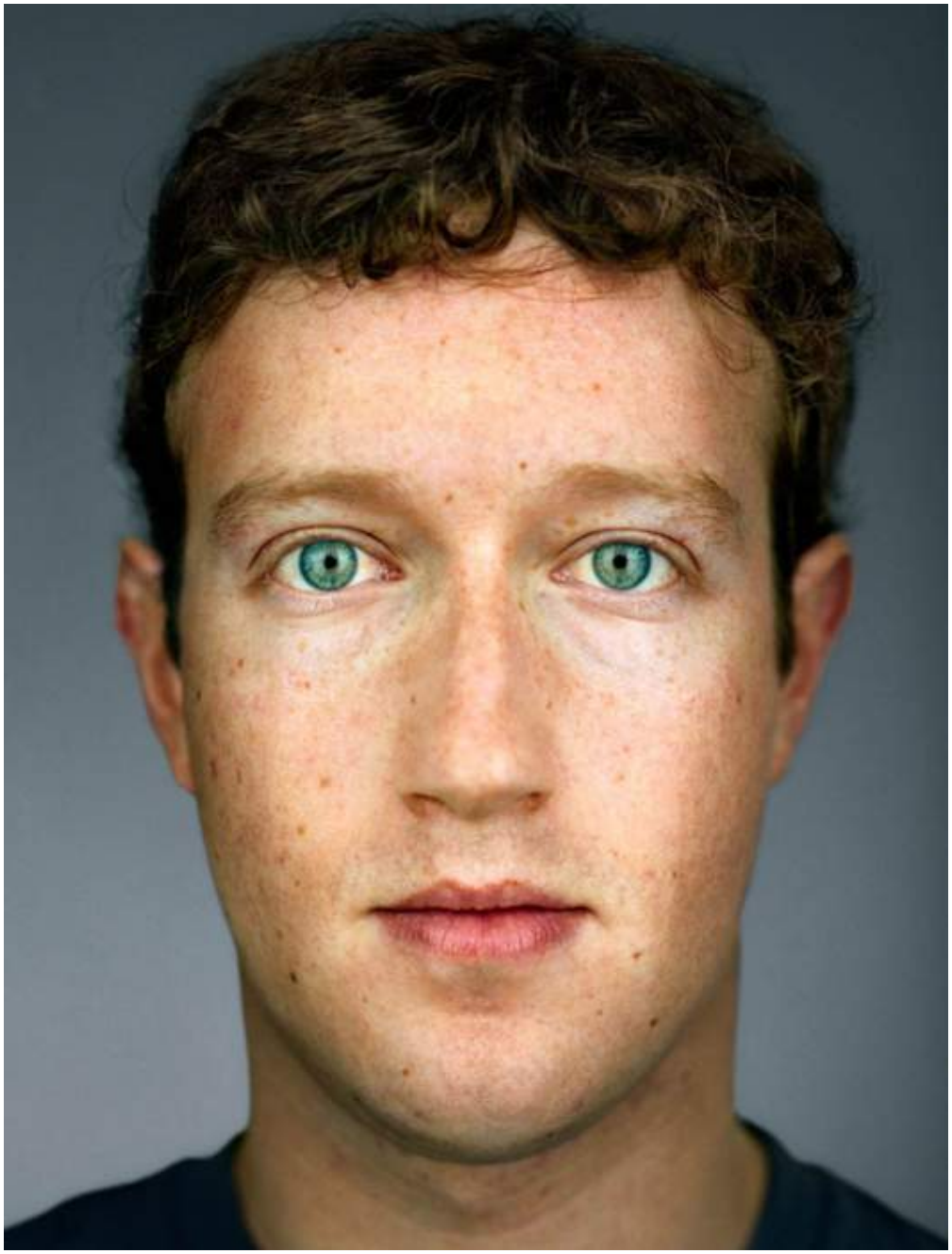
As we can see, Even source and target images has similar compositions, we couldn't obtain any significant result after applying color transfer using source and target images. Because the overall intensity of the source image dominates the mean and std deviation and also as expected, the result image. When we divide input images then apply color transfer on each part, we obtain the right side of image gain similar look as expected output. thanks to dividing input images into 4 parts, we reduced the global effect of intensity value of the source image and we got locally better results.

3.2.1.Source Image

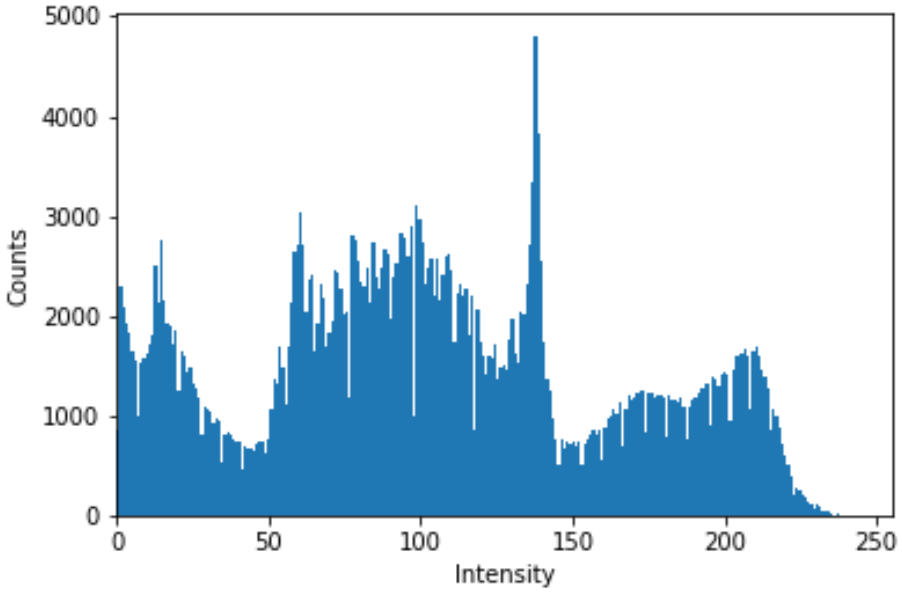
Histogram of Source Image



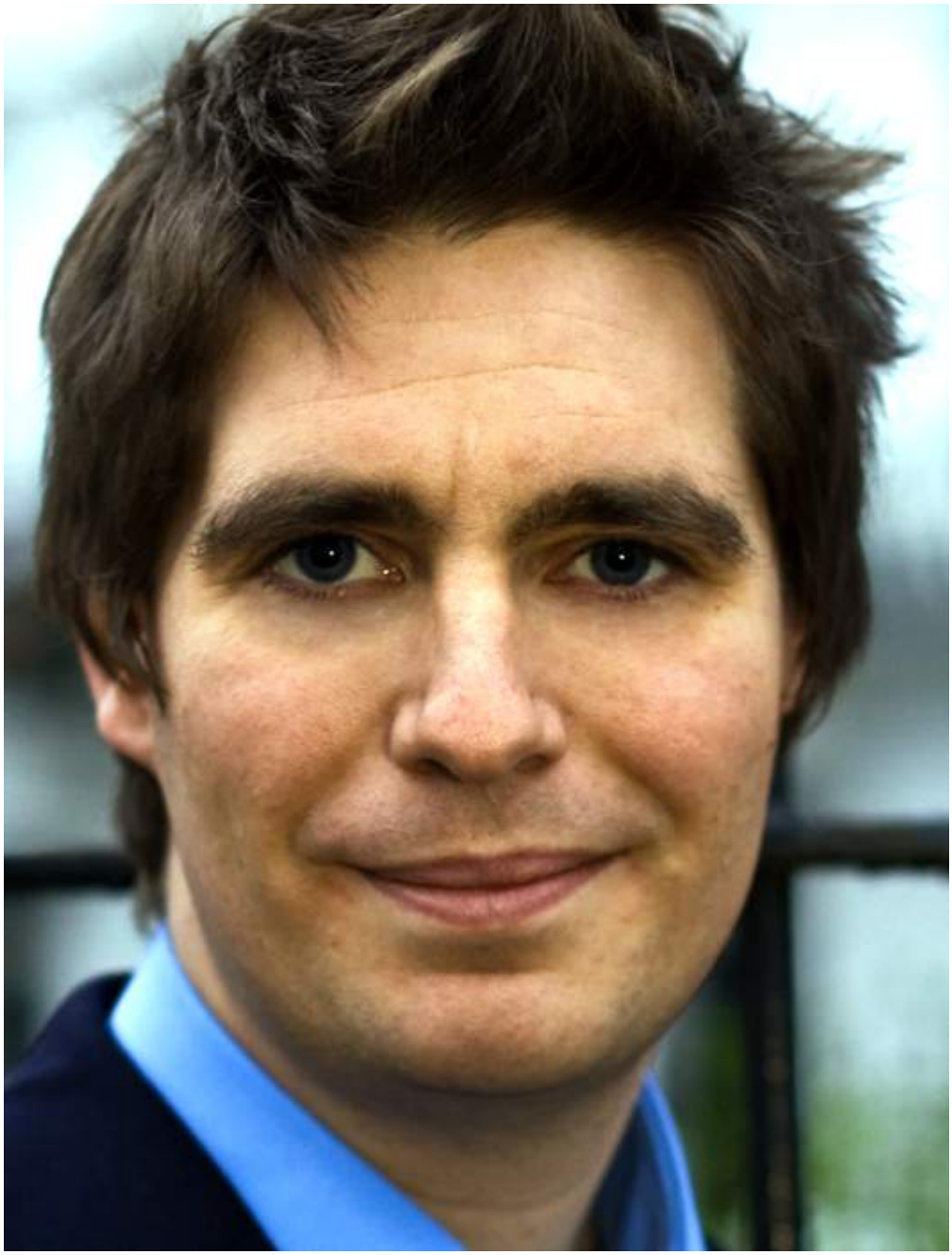
3.2.2.Target Image



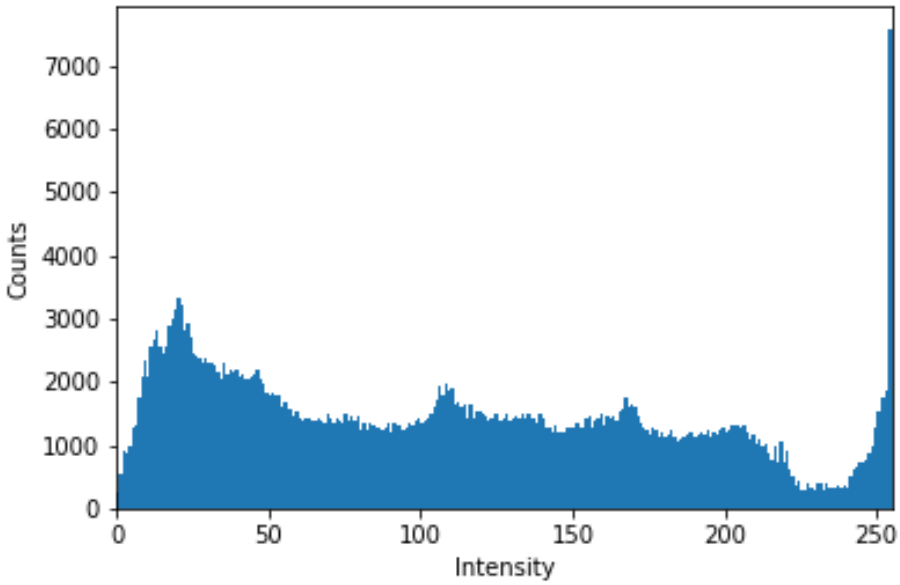
Histogram of Target Image



3.2.3.My Result After Color Transfer

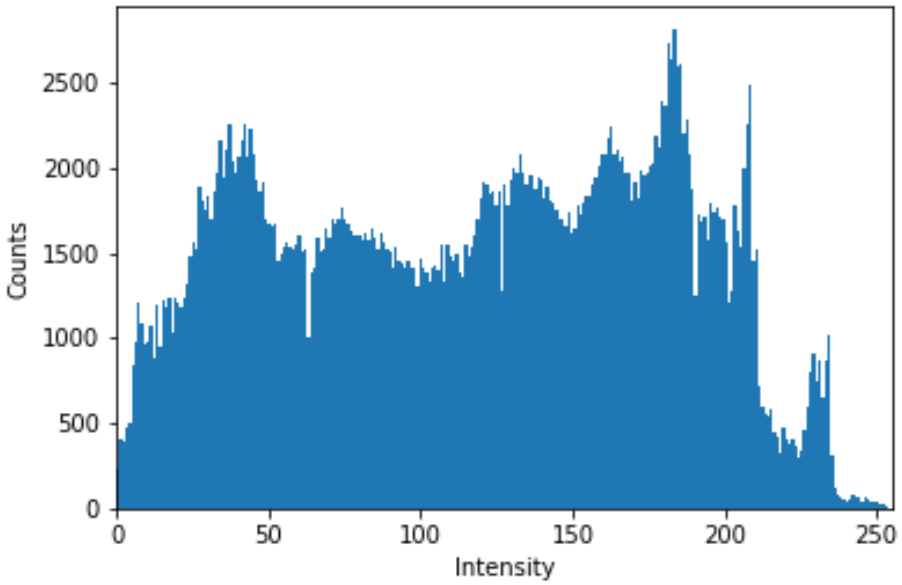


Histogram of My Result Image After Color Transferring



3.2.4.My Result After Dividing Input Images, Then Color Transfer

Histogram of My Result Image After Dividing Input Images, Then Color Transfer



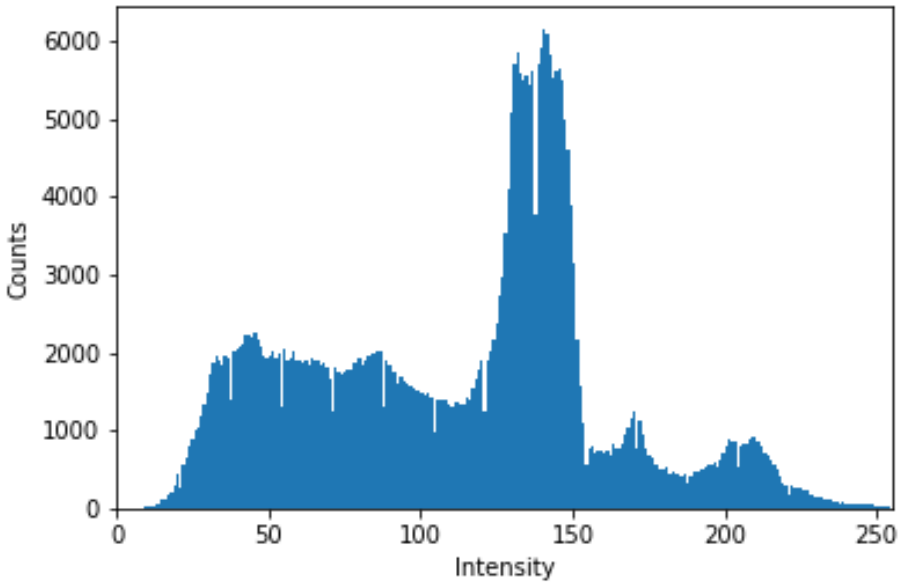
3.2.5.Expected output/result image

Sample 3:

As we can see neither compositions of target and source images are similar nor histograms. The intensity of source image dominates the statistical calculations and gives wrong results. The huge difference between compositions of source and target images doesn't work together in a good way. The source is full with homes, the target is full with darker sky and sea.

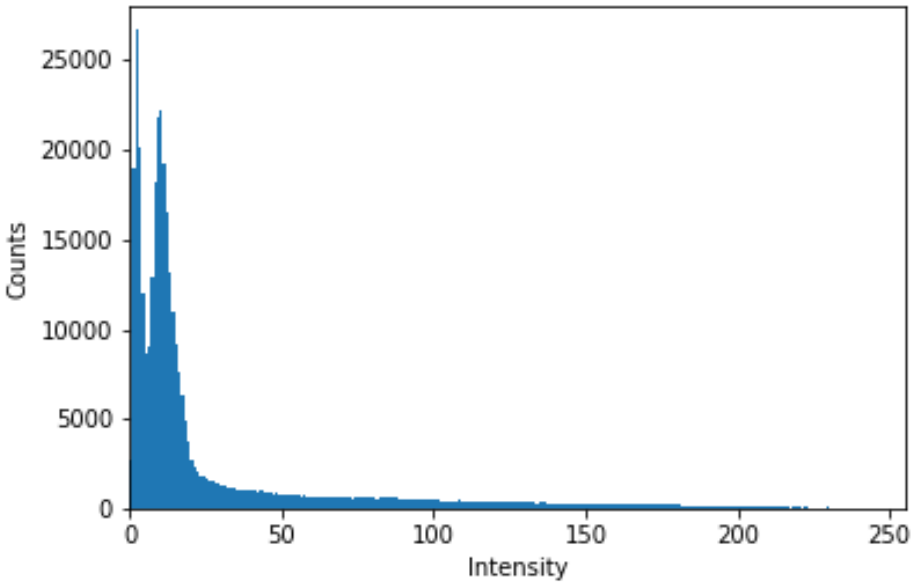
3.3.1.Source Image

Histogram of Source Image



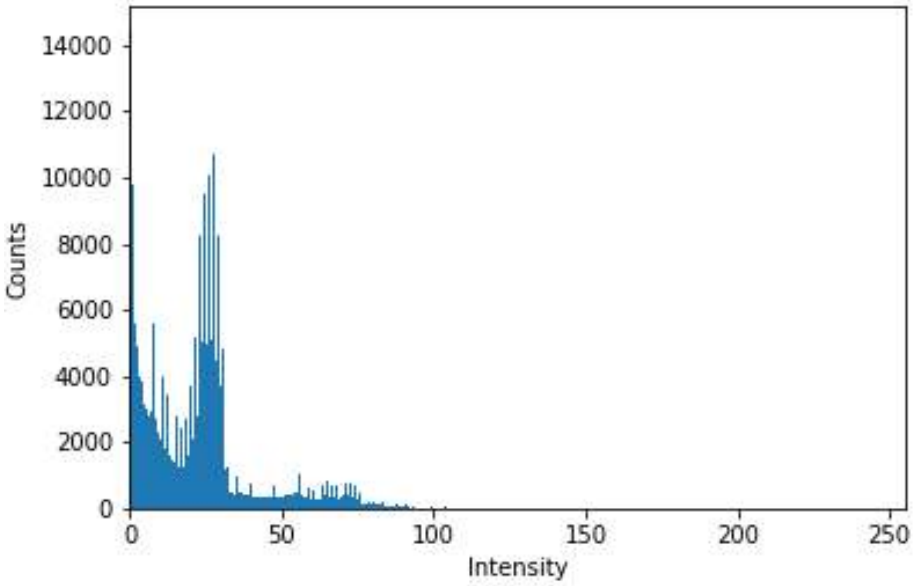
3.3.2.Target Image

Histogram of Target Image



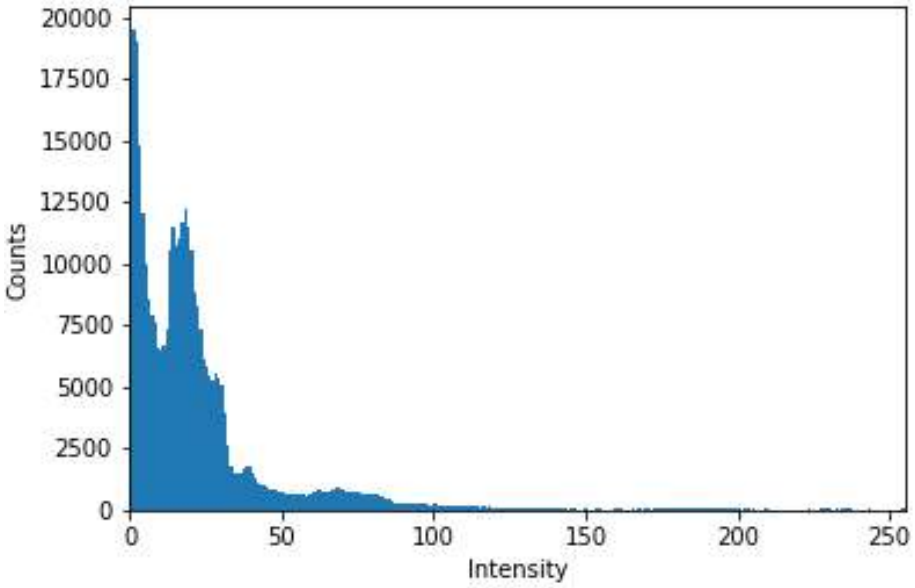
3.3.3.My Result After Color Transfer

Histogram of My Result After Color Transfer Image



3.3.4.My Result After Dividing Input Images, Then Color Transfer

Histogram of M Result Image After Dividing Input Images, Then Color Transfer



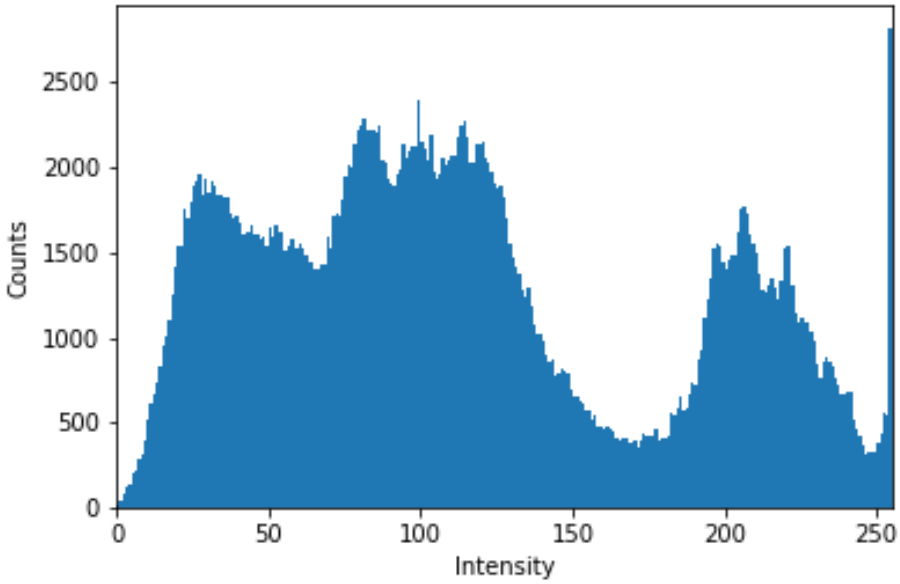
3.3.5.Expected output/result image

Sample 4:

In this sample, we are working on two pictures with a similar appearance but still we couldn't get the expected result, even couldn't get a good luminosity. The reason of that problem is as we compared in previous examples, The difference between intenisty of source and target image decides which input image will dominate each other. This is a failed sample. Since the difference between intensities are distributed on the overall image, we couldn't obtain any better result after dividing input images into 4 part then color transfer.

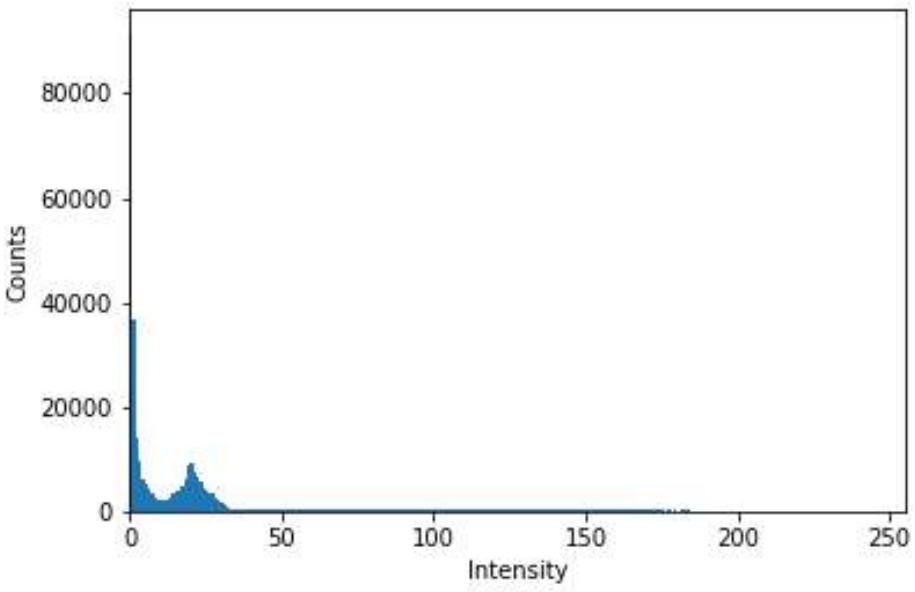
3.4.1.Source Image

Histogram of Source Image



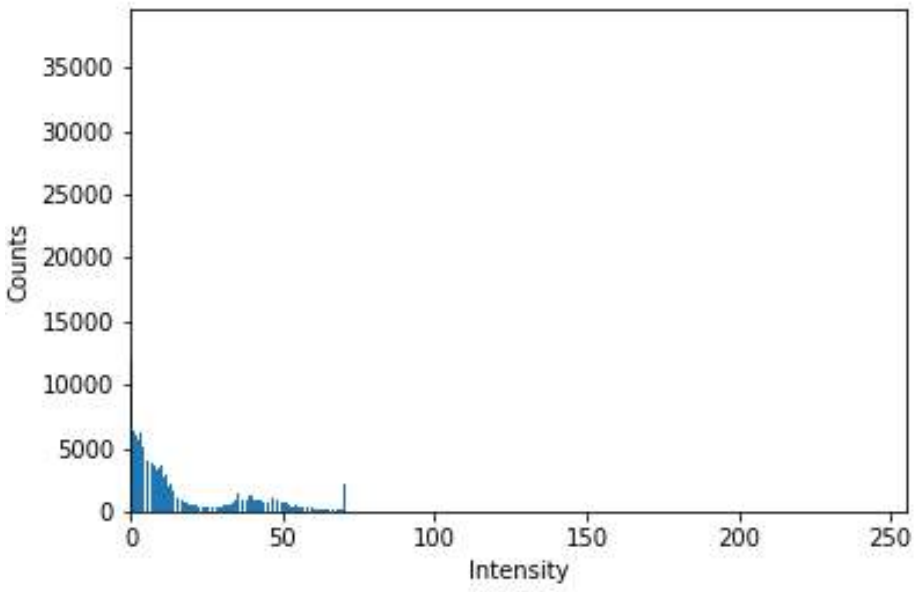
3.4.2.Target Image

Histogram of Target Image



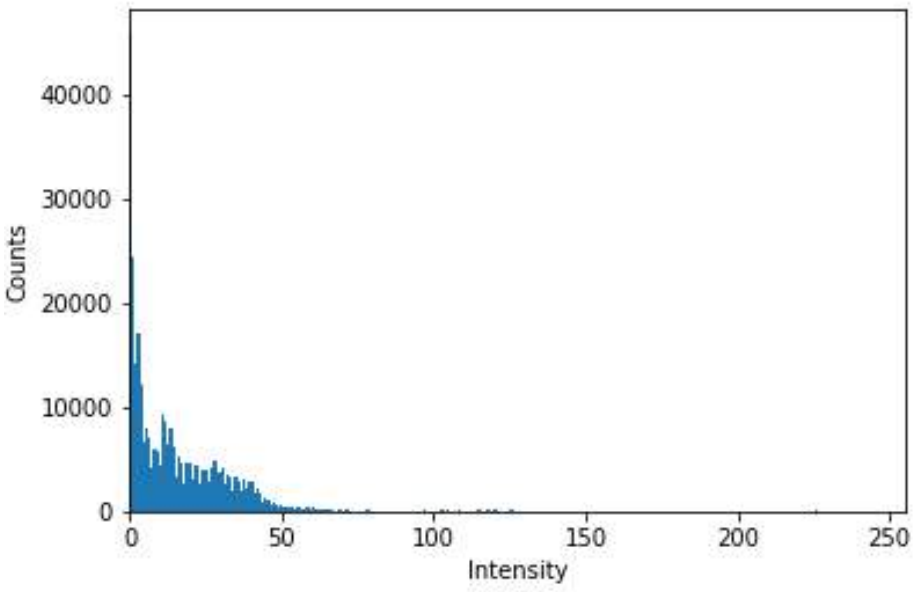
3.4.3.My Result After Color Transfer

Histogram of My Result After Color Transfer Image



3.4.4.My Result After Dividing Input Images, Then Color Transfer

Histogram of M Result Image After Dividing Input Images, Then Color Transfer



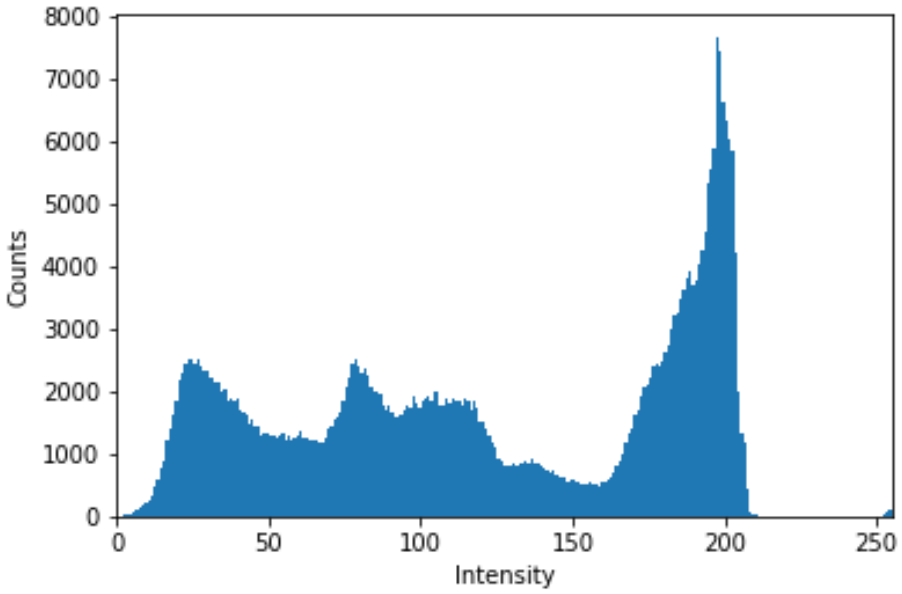
3.4.5.Expected output/result image

Sample 5:

As we can see neither compositions of target and source images are similar nor histograms. The difference between intensity value is the reason of dominating the statistical calculations and giving wrong results. The huge difference between compositions of source and target images doesn't work together in a good way. Since the difference between intensities are distributed on the overall image, we couldn't obtain any better result after dividing input images into 4 part then color transfer.

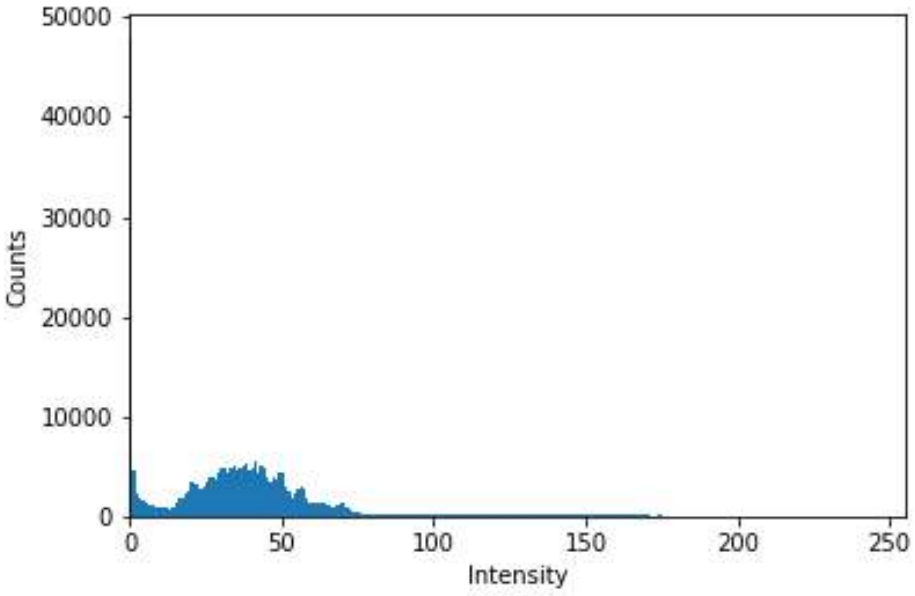
3.5.1.Source Image

Histogram of Source Image



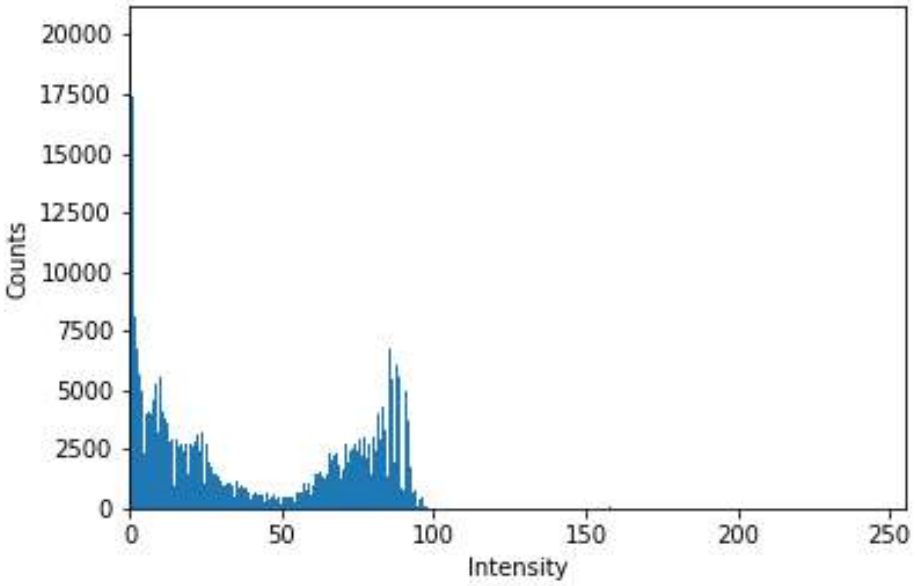
3.5.2.Target Image

Histogram of Target Image



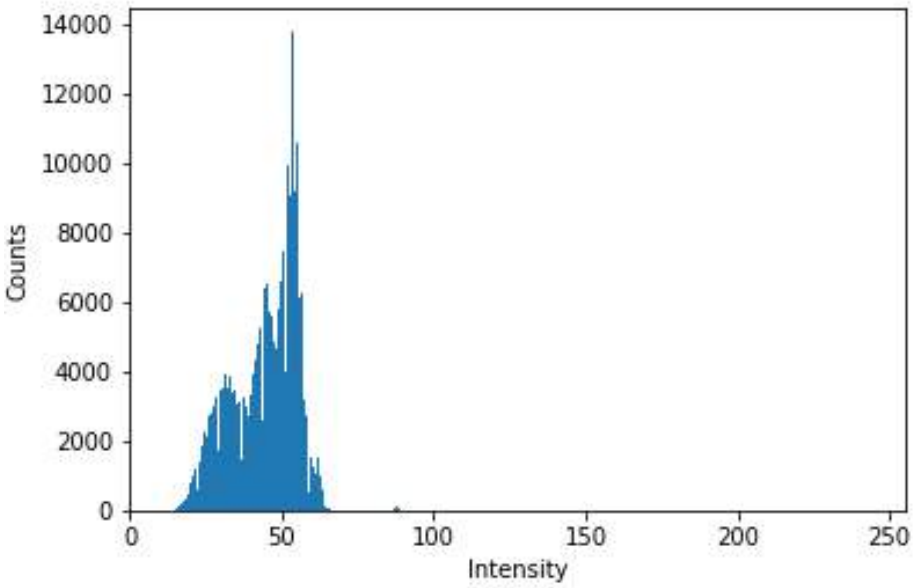
3.5.3.My Result After Color Transfer

Histogram of My Result After Color Transfer Image



3.5.4.My Result After Dividing Input Images, Then Color Transfer

Histogram of My Result Dividing Input Images, Then Color Transfer



3.5.5.Expected output/result image

Final Results

When we are transferring target image's appearance to source image, difference between image compositions may cause to fail because if we want to obtain a high quality images as a result, we have to choose similar compositions between source and target. To enhance the results we can separate input images into subparts how many number of different compositions exists on the image. The number of partititons are depends on the similarity between compositions of source and target. The approach of Reinhard et al. depends on global color statistics, that is why large patches with comparable pixel intensity values can significantly affect the mean and, consequently, the overall color transfer.

Alternative Algorithmic Apporaches to Get Better Results

> **Solution Approach 1**

> The first algorithmic strategy is instead of utilizing the complete image, compute the mean and standard deviation of the section of the source image that you wish to colorize. This method, it is said, will improve the way your mean and standard deviation depict the color space.

> **Solution Approach 2**

> The second algorithmic strategy is applying k-means to both pictures. The centroids between the two photos that are most comparable may be found by clustering on the pixel intensities of each image in the lαβ color space. Then, solely compute statistics for each of these areas. This will help address the overrepresentation issue in global statistics by giving a mean and standard deviation a more local impact.

References

[1] Erik Reinhard, Michael Ashikhmin, Bruce Gooch and Peter Shirley, 'Color Transfer between Images', IEEE CG&A special issue on Applied Perception, Vol 21, No 5, pp 34-41, September - October 2001.