Color Grading for Photography

Michal Conos
Jan 6 · 4 min read

Color grading is a very important stage of the photo post processing which gives your photographs the final touch: the mood, the style and the overall feeling about the image. It can be achieved in many ways, e.g.: by use of built-in filters, stylization plugins or by manually adjusting color curves. Important part of this process is the color theory. Especially achieving of the color harmony, based on various color schemes. In a nutshell basic harmony color schemes are: monochromatic, analogous and complementary.

Example of complementary color scheme. Source image taken from Day to Night Tutorial, adjusted with this code.

In the following series of articles I want to focus on more automated color transfer methods based on statistics and machine learning. Unlike in the manual process where we have to decide whether to grade the picture more into warm or to cool colors, here

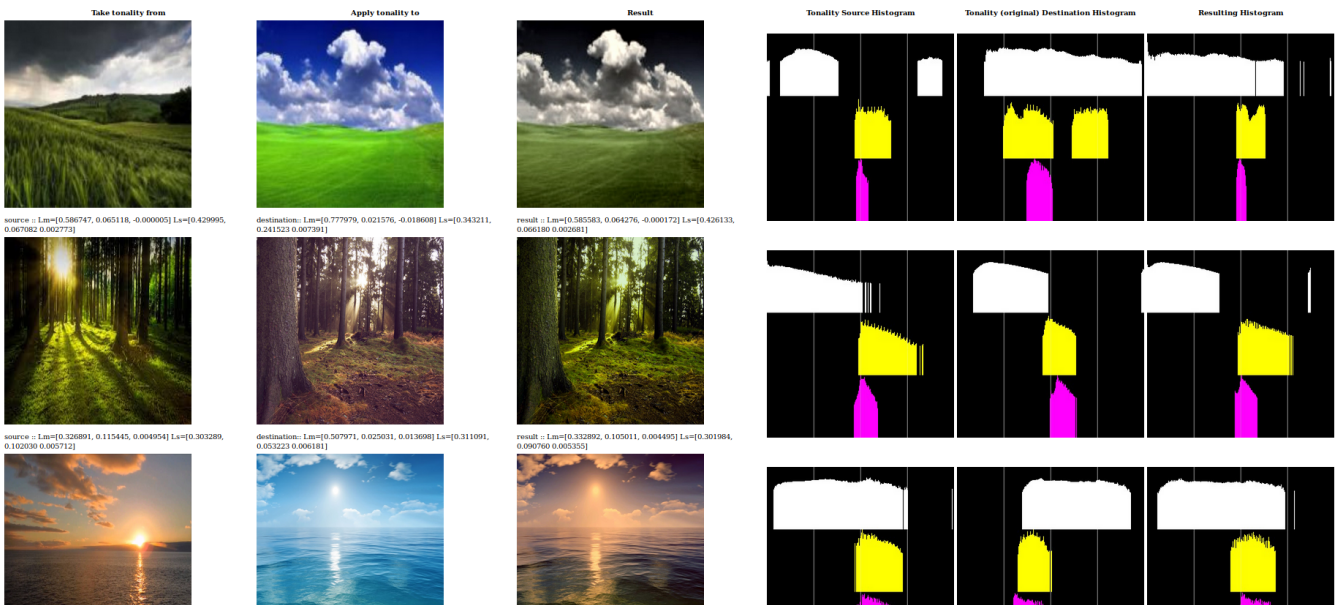
we grab the picture which has the desired look and feel and we transfer those colors into our image to get the final effect. Today we take a closer look at method first published by Eric Reinhard et al., in 2001, in paper named Color Transfer between Images. Almost every paper doing some sort of color transfer is citing this paper. It's a baseline for your results to compare with.

Method Overview

Reinhard's method operates on a global image histogram in the Ruderman's et al. uncorrelated $L\alpha\beta$ color space. Other methods as summarized in Colour space for colour transfer paper might use this $L\alpha\beta$ color space or sometimes the related and widely used CIELAB color space. The issue of transforming the image in correlated RGB space is described in this paper. The basic idea behind Reinhard's at al. approach is to compute arithmetic mean and variance for both source and target image in all 3 color space dimensions. This way we get the mean luminance L and color in $\alpha\beta$ color space. The formula for the output color first substracts the mean from the image we want to adjust μ□, then multiplies it by the variance ratio σ□ / σ□ which transforms the image variance one into another and finally adds the reference image mean μ□ to shift the colors to the desired space. The implementation of this algorithm can be found here. In this implementation two coefficients are introduced to show what happens when we only shift colors (no variance ratio applied) or when only the variance ratio is applied without any color shift.

Under the Hood

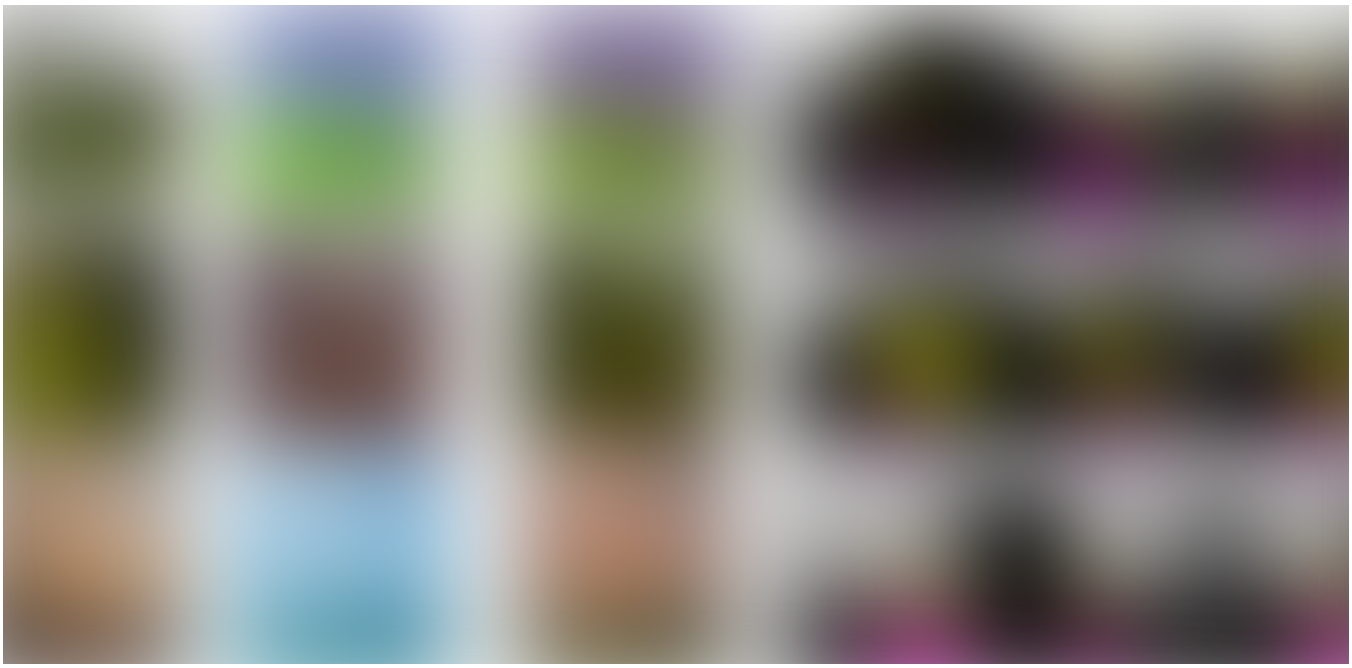
Let's take a closer look on the mechanics of this method, starting with the real world examples:



Results from Reinhard, 2001 Color Transfer between Images, Lαβ color space shown on the right. Both variance and mean applied in full range.

Now let's see what it looks like if do the mean shift only, the code we used:

```
/**
 * Apply tonality from source to dest.
 */
apply_color_reinhard2001(source, dest, source.zone, dest.zone, 0.0f
/*variance coefficient*/, 1.0f /*mean coefficient*/);
```



Apply the mean shift only, the distribution is only shifted left or right. The luminance contrast remains the same (white histogram on the right)

What happens if we adjust the variance without applying the shift?

```
/**
 * Apply tonality from source to dest.
 */
apply_color_reinhard2001(source, dest, source.zone, dest.zone, 1.0f
/*variance coefficient*/, 0.0f /*mean coefficient*/);
```

. . .

Variance scale only, no mean shift

Implementation Summary

1. Transfer RGB color space into $L\alpha\beta$ color space. This part can be found in color_conversion.c source code

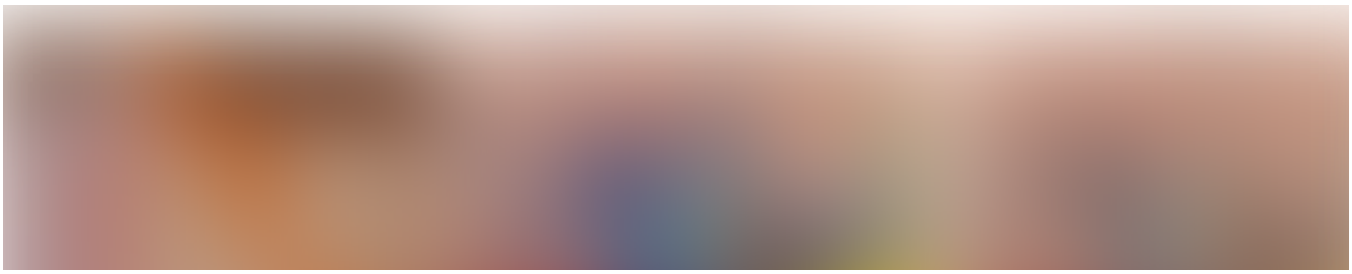
2. Next step is to compute mean and variance for L, α, β in both source and destination images: $\mu = (1/N) \sum L$, $\sigma = (1/\sqrt{N}) \sqrt{\sum (L - \mu)^2}$ where N is the total number of samples.

3. The output color is then computed as: $L_o = \sigma\square(L\square - \mu\square) / \sigma\square + \mu\square$ for luminance, the α, β are computed in the same way with mean and variance for α, β respectively.

4. The final step is to transform $L\alpha\beta$ into RGB color space.

Conclusion

This method works pretty well in cases where the image compositions are similar. Here is an example of a failed result, drawing from Kitaj, R.B, The Oak Tree, 1991 as the target to apply the colors to, and Nemcsics, A., Europe, 2003 as a color source:



From left to right: Nemcsics, A., Europe, 2003, Kitaj, R.B, The Oak Tree, 1991, result image

To address this issue we have to use some local color tone mapping technique, which first clusters the image in color space in order to extract the dominant colors in both source and target images. Let's talk about this next time.

)

Colors Photography Image Processing