Implement Auto White Balance Algorithm using FFCC

Overview

The color of a scene depends a lot on the color temperature of the illuminant in the scene. Our human vision system is good at chromatic adaptation. i.e. we automatically adjust the color temperature on what we see. However, for images captured by camera sensors, without applying white balance gains, the captured images might not look natural.

The process of recovering the color of the scene from sensor captured data to match human eye perception is referred to as “auto white balance”. In this process, we wish to estimate the color of the illuminant in the scene and transform the image such that the illuminant is mapped to the white point in the linear RGB space.

Related work

There are many existing approaches for illuminant estimations, such as gray-world, MaxRGB, Convolutional Color Constancy (CCC), Fast Fourier Color Constancy (FFCC), etc.

Gray-world:

Assume the world is on average gray [1, 2]. In this algorithm, the average chromaticity of a scene is taken as the illuminant. In most cases it works well, but it fails when large colored patches appear in the scene. An extreme example is an image of blue sky.

MaxRGB:

In this algorithm, the maximum values found in each channel, Rmax, Gmax and Bmax are used to form the illuminant [3].

Convolutional Color Constancy (CCC):

It is a learning-based algorithm, which applied a convolution kernel (learned from training data set) to the image log-chrominance histogram for illumants scoring, and pick the highest-scoring as the best fit illuminant for the scene [4].

Fast Fourier Color Constancy (FFCC):

It is based on CCC. Instead of apply convolution for the entire image in spatial domain, It performs a learned FFT convolution on a small and aliased toroidal log-chrominance histogram, apply a softmax, fit a de-aliased bivariate von Mises distribution to get the illuminant estimation [5, 6].

Objective

The goal of this project is to implement an auto white balance algorithm using FFCC and compare it to traditional white balancing methods, gray-world and MaxRGB, etc.

Milestones

Week6:

- Read through FFCC paper in depth to fully understand the proposed system.
- Modify/Polish the the proposal with more details of the FFCC approach.
- Generate images using traditional methods.

Week7:

- Download the FFCC training code and run it to get the convolution kernel.
- Coding:
    - implement a method to convert input image to a small and aliased toroidal log-chrominance histogram (64x64).
    - implement method to fit a de-aliased bivariate von Mises distribution.

Week8:

- Coding: finish up the auto white balance pipeline and debug.

Week9:

- Compare with output images using traditional white balancing methods.
- Come up with ideas to improve CCFF (optional).
- Prepare the presentation, and write the final report.

Week10:

- Presentation and submit final report.

References

[1] Weijer, et al, “Edge-based color constancy”, IEEE Transactions on Image Processing, vol. 16, no. 9 , September 2007
[2] G. Buchsbaum, “A spatial processor model for object colour perception,” J. Frank. Inst., vol. 310, 1980.
[3] Funt and Shi, “The Rehabilitation of MaxRGB”, Proceedings 18th IST Color Imaging Conference, Nov. 2010
[4] J. T. Barron. Convolutional color constancy. ICCV, 2015.
[5] Barron and Tsai, Fast Fourier Color Constancy, CVPR 2017
[6] Liba et al., Handheld Mobile Photography in Very Low Light, SIGGRAPH Asia 2019