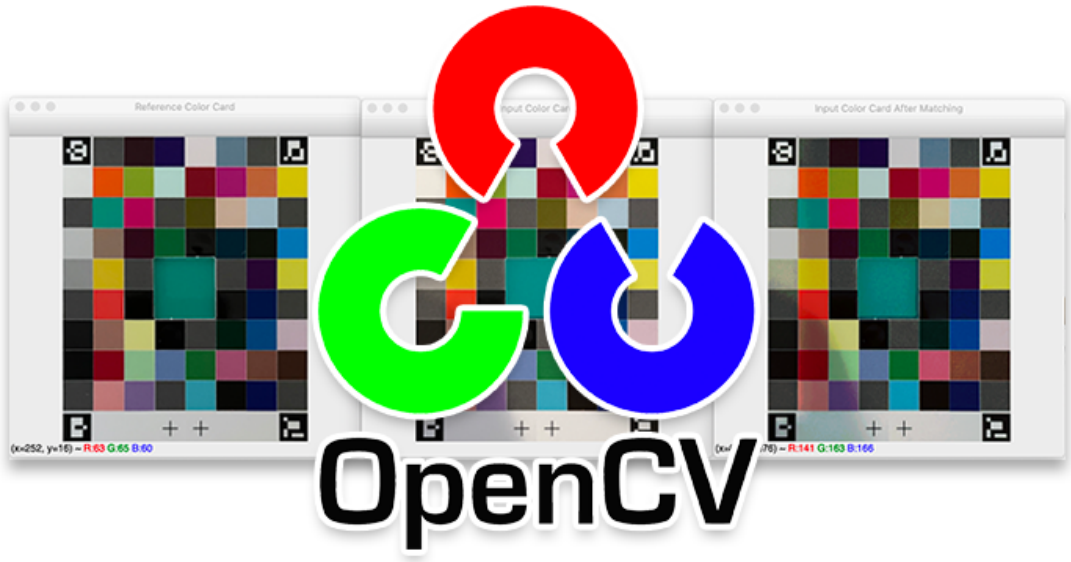
Automatic color correction with OpenCV

"Adrian at PyImageSearch" <adrian@pyimagesearch.com>
收件人:navicester@163.com
时　间:2021-2-16 0:15:00
附　件:

Hi,

In today's new guide, you will learn how to perform automatic color correction with OpenCV using a color matching/balancing card:



Last week we discovered how to perform histogram matching. Using histogram matching, we can take the color distribution of one image and match it to another.

A practical, real-world application of color matching is to perform basic color correction through color constancy. The goal of color constancy is to perceive the colors of objects correctly regardless of differences in light sources, illumination, etc. (which, as you can imagine, is easier said than done).

Photographers and computer vision practitioners can help obtain color constancy by using color correction cards, like this one:

Using a color correction/color constancy card, we can:

1. Detect the color correction card in an input image
2. Compute the histogram of the card, which contains gradated colors of varying colors, hues, shades, blacks, whites, and grays
3. Apply histogram matching from the color card to another image, thereby attempting to achieve color constancy

In this tutorial, we'll build a color correction system with OpenCV by putting together all the pieces we've learned from previous tutorials on:

1. Detecting ArUco markers with OpenCV and Python
2. OpenCV Histogram Equalization and Adaptive Histogram Equalization (CLAHE)
3. Histogram matching with OpenCV, scikit-image, and Python

By the end of the guide, you will understand the fundamentals of how color correction cards can be used in conjunction with histogram matching to build a basic color corrector, regardless of the illumination conditions under which an image was captured.

To learn how to apply basic color correction with OpenCV to your own images, just keep reading the full tutorial here.

Adrian Rosebrock
Chief PyImageSearcher