White Balancing

Due on May 9th, 2014

Digital Photography - Spring 2014

Exercise 3

1 Images

There are four images available on Moodle for this exercise. You will apply different white-balancing algorithms on different images.
Hint: Make sure all your images have been converted to double precision and normalized by the maximum value 255 (you can use *im2double* function for that purpose). Moreover, remember to apply gamma correction and then normalization before displaying the image.
Hint: Gamma correction:

$$V_{out} = AV_{in}^{1/\gamma} \tag{1}$$

where in common case $A = 1$. We give you the value for γ in the template: $exe3\_template\_2014.m$.

2 Introduction

The human visual system perceives the color of objects as relatively constant under varying illumination conditions. Color constancy and white balancing algorithms try to achieve such effect. Color constancy algorithms estimate the illuminant while white balancing algorithms aim at discarding the influence of illuminant. All white balancing algorithms process the RGB channels independently. In this exercise, you will implement four white-balancing methods and study their differences.

3 Gray World Algorithm

Note: For this part, you need to apply all the requested steps on the first and second images (Im1 and Im2).

The Gray World algorithm is based on the assumption that the average reflectance spectrum of all objects in an image is flat. It means that in a correctly white-balanced image, the average intensity in all color channels (red, green, and blue) is the same. Consequently, in a non white-balanced image, the shift from gray (equal average in all color channels) corresponds to the color of the illuminant. This algorithm forces the means of color channels to be equal.
3.1 Find the three average values for the color channels in the image (a_R, a_G, and a_B).
3.2 Compute the average of a_R, a_G, and a_B. We call this new value a.
3.3 Rescale each channel of the image so that its average is a.
3.4 Run the algorithm on both images, display the original images and the result images on the same figure using the command "subplot".
3.5 What are the limitations of this algorithm?

4 Weighted Gray World Algorithm

Note: For this part, you need to apply all the requested steps on the second images Im2.

As you saw, the Gray World algorithm has some limitations, especially if we have an image with a predominant color. To solve this problem, Gershon et al. proposed the Weighted Gray World algorithm. In this method, the image is segmented into patches of equal colors before estimating the illuminant. The color of each segment is then counted only once in the averaging, so that the patches of different sizes have equal weight.

4.1 Calculate the histogram of each color channel with 255 bins.

Hint: Look up the "imhist" function.

4.2 Calculate the mean code value in each channel by counting each color only once regardless of how many times the color appears in the image. Use the histogram information, and sum over all the bins in which there exists at least one pixel.

Hint: You can use the "sign" function to find out which bins in the histogram contain at least one pixel.

4.3 Use the mean value you get in 4.2 to re-scale Im2, using the same way as Gray World. Compare the result with the Gray World result.

4.4 What is the advantage of the Weighted Gray World over the Gray World algorithm?

5 MaxRGB Algorithm

Implement the MaxRGB white balancing algorithm and apply it to images Im 1, 2, and 3. This algorithm assumes that there exists a highly reflective patch in the image that reflects light equally at all wavelengths. Consequently the brightest point in the image belongs to a white patch. Hence, the color of this point is the color of the illuminant we want to estimate and discard.

5.1 Find the brightest point in the image. This pixel is the one that has the highest (R+G+B)/3 among the other pixels.

5.2 Discard the color of the illuminant by rescaling the image channels so that the red, green, and blue values of the white point (the pixel found in 5.1) will be the same.

5.3 First apply the MaxRGB algorithm on the images. Then apply gamma correction on them. Display all the results on one figure. Justify your results.

5.4 As you can see, finding the white point of the image automatically is not always easy. Thus, some image processing softwares, like Photoshop, ask the user to localize the white point of the scene (we know that the human visual system is capable of recognizing the white point even in the unbalanced image). Change your MaxRGB algorithm such that the program asks the user to click on the white point of the image. The MaxRGB algorithm then uses this point to white balance the image. For this part, you can use the command "ginput".

Hint: Note that MATLAB image coordinates are not the same as matrix coordinates. It means that the first output of ginput is the index of the corresponding **column**, and the second output is the corresponding **row**.

Comment on the performance of the MaxRGB algorithm when the white point is selected manually. Compare the results with the automatic white point selection.

5.5 As a pre-processing step for the MaxRGB method, apply a median filter **only** to the third image (Im 3) and find the white point of the pre-processed image. In the original image, re-scale each channel based on the newly-found maxima, so that the maximum for each channel is the same and equal to the max of the image (for this question, first apply the median filter, then MaxRGB, and finally the nonlinear processing. See Figure 1).

To apply the median filter, you can use MATLAB function "medfilt2".

5.6 What's the advantage of above mentioned pre-processing step (median filtering)?

5.7 When does it give good results?

5.8 What are its limitations?

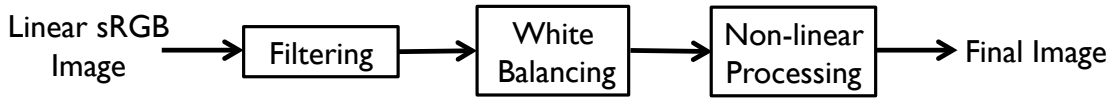

Figure 1: The order of operations you need to apply in 5.5.

6 Illuminant Estimation

We can directly do white balancing if we know the color of the illuminant. In $exe3.mat$ we provide the illuminant spectrum A, the camera sensitivity R, the transformation matrix from camera sensitivities to sRGB T, and the corresponding photo Im4 taken under the illuminant A. Note that Im4 is in the linear sRGB color space.
6.1 Compute the RGB values of the illuminant in the camera space.
6.2 Transform the RGB value from the camera space to the linear sRGB space.
Hint: Use the transformation matrix T.
6.3 Use the linear sRGB value of the illuminant to white balance Im4.
Hint: Divide the RGB channels of Im4 by the linear sRGB value of the illuminant, respectively.
6.4 Use Gray World algorithm to white balance Im4. Compared with the results in 6.3, which one is better?
6.5 What are the limitations of this method?

7 HAND IN

- Include your commented working code in a zip file
- Date and time: **May 9th, 2014 at noon**

NOTE: The main filename must have the following format: $FirstName\_LastName\_ExerciseNo$. Please structure your code such that each part is easily identifiable, by following the numbering in this description.

References

[1] R. Gershon, A. Jepson, J. Tsotsos. "From [R, G, B] to surface reflectance: computing color constant descriptors in images" *Perception*. vol. 17, pp. 755-758, 1988.

[2] G. D. Finlayson, S. D. Hordley, P. M. Hubel. "Color by correlation: a simple, unifying framework for color constancy" *Pattern Analysis and Machine Intelligence, IEEE Transactions on*. Vol. 23, pp. 1209 - 1221, 2001.