UNIVERSITÄT TÜBINGEN
LEHRSTUHL FÜR COMPUTERGRAFIK
PROF. DR.-ING. HENDRIK P.A. LENSCH
JIEEN CHEN

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

17.05.2018

COMPUTATIONAL PHOTOGRAPHY
ASSIGNMENT 3

Submission deadline for the exercises: Thursday, 24. May 2018 before 11:59.

Instructions: Upload the source code to your solution **(no images please, just the plain code)** in the ILIAS system at:
ILIAS Computational Photography SoSe2018

3.1 White Balancing ($5 + 10 + 5 = 20$ points)

Often the colors of the final image are not correct and additional color processing is required. This process is usually called *white balancing*. One way to perform white balancing is to capture a photo of a target with known colors (Figure 1) and then do a least squares fit of measured color values to the desired ones.

Figure 1: Image of a checkerboard with wrong colors.

Your task is to perform a least squares fit by filling in the missing lines in the file '`white_balance.m`':

a) Fill the matrix `cam` with the measured `r,g,b` values for each color patch of the checkerboard. Matrix `coords` holds the upper left and bottom right (y, x) coordinates of a region in the upper left patch from where colors can be extracted, while `delta` is the offset in both x and y direction between two consecutive patches.

b) Compute the white balancing matrix `Mat`, using the matrices `b` and `cam`, applying the least squares fitting method.

c) Multiply each pixel of the original image with the calculated matrix `Mat` to perform white balancing. (**Hint:** to avoid using for-loops, the `reshape` function might be useful.)

General Tips:

- We use the Peak Signal to Noise Ratio (PSNR) between the ground truth and the recovered image to measure the image fidelity, which in Matlab is `peaksnr = psnr(A,ref)`

3.2 Demosaicing ($40 + 40 = 80$ points)

Nowadays cameras acquire color photographs is by means of using a color filter array (CFA). The most common of such CFAs is the Bayer pattern (see figure 2). In this configuration each pixel only senses one out of the three primary colors, hence discarding about $\frac{2}{3}$ of the incoming light. The recovery of the missed color information in each pixel is called demosaicing and requires quite tricky image processing to avoid the introduction of unsightly color artifacts.
A reasonable approach is to reconstruct the green channel first, since it has the best spatial distribution of the three color channels and therefore is less subject to noise. After the green channel is finished one can employ its information to reconstruct the red and blue channel.

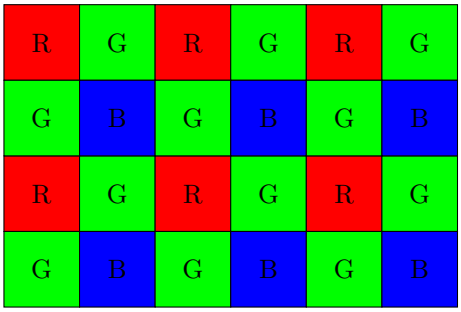

Figure 2: The Bayer pattern.

a) The straightforward method is to use the color information in a 2×2 neighborhood to restore the colors for all 4 pixels. This procedure is quite simple but has a severe drawback. Realize demosaicing by nearest neighbor interpolation.

b) The most common interpolation type is bilinear interpolation. The basic scheme is to interpolate a missing pixel by averaging existing pixels in a 3×3- neighborhood first in one direction and then averaging the computed averages along the other direction. Implement bilinear interpolation and compare it to nearest neighbor interpolation. (**Hint:** the `filter2` function might be useful.)

All exercises should be implemented in functions taking the bayer pattern image as input and return the demosaiced image, e.g. `image = function_name(bayer)`.
You don't need to implement correct boundary handling, since the results are only differing in a few boundary pixels and the induced overhead is just too high.
Give a short description of the drawbacks / artifacts of the different algorithms and explain the problem behind, e.g. why it happens.
Use `psnr` to measure the recovered image fidelity.

General Tips:

- If you want to extract the blue channel of a bayer patterned picture as a compacted array you can get it with `img(2:2:end,2:2:end,3)`, assuming the Bayer pattern layout shown in Figure 2.

Links:

- To get a clue about the demosaicing dilemma give this one a try
  http://www.cambridgeincolour.com/tutorials/camera-sensors.htm
- PSNR meatric
  https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio