EE 569: Homework #1

Issued: 1/13/2020 Due: 11:59PM, 1/28/2020

General Instructions:

1. Read *Homework Guidelines* and *MATLAB Function Guidelines* for the information about homework programming, write-up and submission.
2. If you make any assumptions about a problem, please clearly state them in your report.
3. You need to understand the USC policy on academic integrity and penalties for cheating and plagiarism. These rules will be strictly enforced.

Problem 1: Image Demosaicing and Histogram Manipulation (50%)

(a) Bilinear Demosaicing (10%)

To capture color images, digital camera sensors are usually arranged in form of a color filter array (CFA), called the Bayer array, as shown in Figure 1. Since each sensor at a pixel location only captures one of the three primary colors (R, G, B), the other two colors have to be re-constructed based on their neighbor pixel values to obtain the full color. Demosaicing is the process of translating this Bayer array of primary colors into a color image that contains the R, G, B values at each pixel.

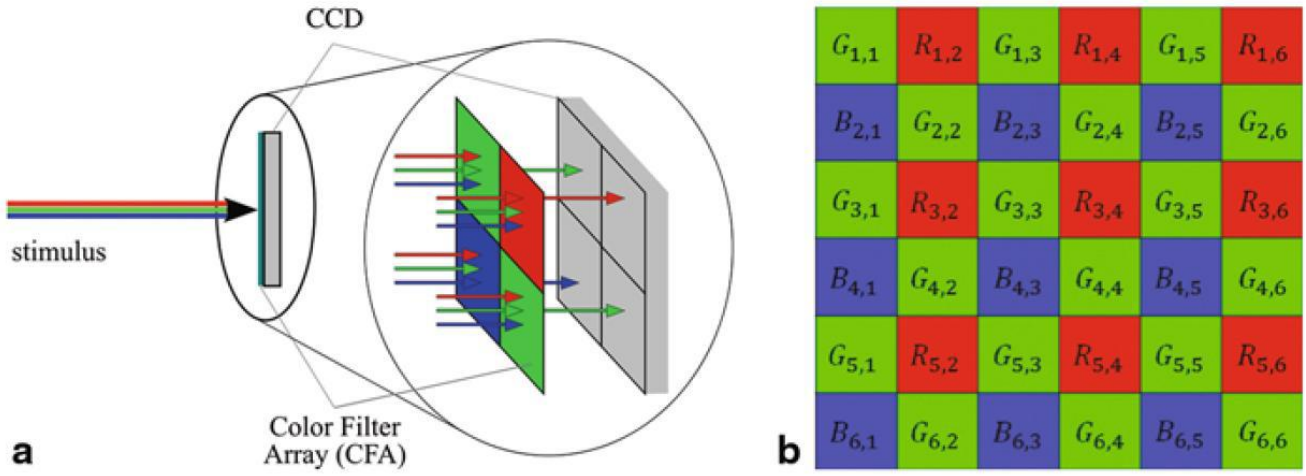

Figure 1: (a) Single CCD sensor covered by a CFA and (b) Bayer pattern [1].

Implement the simplest demosaicing method based on bilinear interpolation. Exemplary demosaicing results are given in Figure 2. With this method, the missing color value at each pixel is approximated by bilinear interpolation using the average of its two or four adjacent pixels of the same color. To give an example, the missing blue and green values at pixel $R_{3,4}$ are estimated as:

$$\hat{B}_{3,4} = \frac{1}{4}(B_{2,3} + B_{2,5} + B_{4,3} + B_{4,5})$$

$$\hat{G}_{3,4} = \frac{1}{4}(G_{3,3} + G_{2,4} + G_{3,5} + G_{4,4})$$

As for pixel $G_{3,3}$, the blue and red values are calculated as:

$$\hat{R}_{3,3} = \frac{1}{2}(R_{3,2} + R_{3,4})$$
$$\hat{B}_{3,3} = \frac{1}{2}(B_{2,3} + B_{4,3})$$

(a)

(b)

Figure 2: (a) The original image and (b) the demosaiced image by bilinear interpolation.

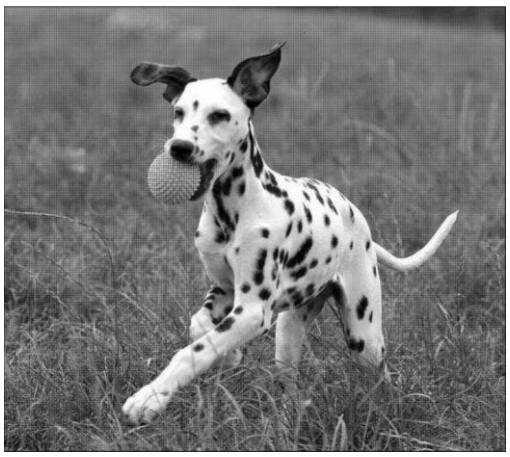

Figure 3: The dog image with the CFA sensor input.

(1) Apply the bilinear demosaicing to the *Dog* image in Figure 3 and show your results.
(2) Compare your demosaiced image with the ground truth color image *Dog_ori*. Do you observe any artifacts? If yes, explain the cause of the artifacts and provide your ideas to improve the demosaicing performance.

(b) Malvar-He-Cutler (MHC) Demosaicing (20%)

Malvar et al. [2] proposed an improved linear interpolation demosaicing algorithm. It yields a higher quality demosaicing result by adding a 2nd-order cross-channel correction term to the basic bilinear demosaicing result. Both the bilinear and the MHC demosaicing results of the *Fruit_Shop* image are shown in Figure 4.

Figure 4: Demosacing results of Fruit_Shop image: the CFA input (left), the bilinear demosaicing result (middle) and the MHC demosaicing result (right).

The MHC algorithm is stated below.
To estimate a green component at a red pixel location, we have

$$\hat{G}(i,j) = \hat{G}^{bl}(i,j) + \alpha\Delta_R(i,j)$$

where $\hat{G}^{bl}$ is the bilinear interpolation result and the 2nd term is a correction term. For the 2nd term, α is a weight factor, and Δ_R is the discrete 5-point Laplacian of the red channel:

$$\Delta_R(i,j) = R(i,j) - \frac{1}{4}\left(R(i-2,j) + R(i+2,j) + R(i,j-2) + R(i,j+2)\right)$$

To estimate a red component at a green pixel location, we have

$$\hat{R}(i,j) = \hat{R}^{bl}(i,j) + \beta\Delta_G(i,j)$$

where Δ_G is a discrete 9-point Laplacian of the green channel.

To estimate a red component at a blue pixel location,

$$\hat{R}(i,j) = \hat{R}^{bl}(i,j) + \gamma\Delta_B(i,j)$$

where Δ_B is a discrete 9-point Laplacian of the blue channel. The weights control how much correction is applied, and their default values are:

$$\alpha = \frac{1}{2}, \beta = \frac{5}{8}, \gamma = \frac{3}{4}$$

The above formulas can be generalized to missing color components at each sensor location. Consequently, the MHC demosaicing can be implemented by convolution with a set of linear filters. There are eight different filters for interpolating the different color components at different locations as illustrated in Figure 5 [2].

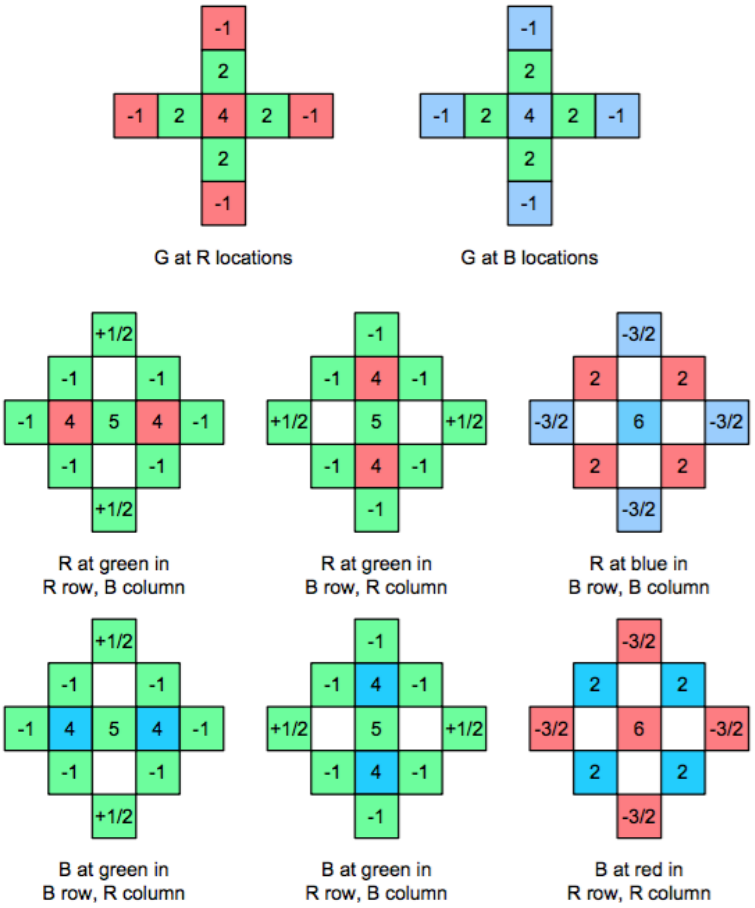

Figure 5: **Filter coefficients.**

(1) Implement the MHC linear demosaicing algorithm and apply it to the *Dog* image in Figure 3. Show your results.
(2) Compare the MHC and the bilinear demosaicing results and explain the performance differences between these two algorithms in your own words.

(c) Histogram Manipulation (20%)

Implement two histogram equalization techniques:

- Method A: the transfer-function-based histogram equalization method,
- Method B: the cumulative-probability-based histogram equalization method

to enhance the contrast of the *Toy* image in Figure 6 below.

(1) Plot the histograms of the red, green and blue channels of the original image. The figure should have the intensity value as the x-axis and the number of pixels as the y-axis.
(2) Apply Method A to the original image and show the enhanced image. Plot the transfer function for each channel.
(3) Apply Method B to the original image and show the enhanced image. Plot the cumulative histogram for each channel.
(4) Discuss your observations on these two enhancement results. Do you have any idea to improve the current result?

Note that MATLAB users CANNOT use functions from the Image Processing Toolbox except displaying function like imshow().

Figure 6: Toy image

Problem 2: Image Denoising (50 %)

In this problem, you will implement a set of denoising algorithms to improve image quality. You can use the PSNR (peak-signal-to-noise-ratio) quality metric to assess the performance of your denoising algorithm. The PSNR value for R, G, B channels can be, respectively, calculated as follows:

$$\text{PSNR (dB)} = 10\log_{10}\left(\frac{\text{Max}^2}{\text{MSE}}\right)$$

$$\text{where MSE} = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(Y(i,j)-X(i,j)\right)^2$$

X : Original Noise-free Image of size $N \times M$

Y : Filterd Image of size $N \times M$

Max: Maximum possible pixel intensity = 255

Remove noise in the image in Figure 7(b), compare it with the original image in Figure 7(a), and answer the following questions:

(a) Basic denoising methods (10%)

(1) What is the type of embedded noise in Figure 7(b)?

(2) Apply a linear filter of size N by N to the noisy image. Compare the performance of two choices of the filter parameters – the uniform weight function and the Gaussian weight function.

Figure 7: The original and noisy corn images.

(b) Bilateral Filtering (10%)

In most low-pass linear filters, we often see degradation of edges. However, using some nonlinear filters, we can preserve the edges. Bilateral filters are one such kind of filters. A discrete bilateral filter is given by:

$$Y(i,j) = \frac{\sum_{k,l} I(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}$$

$$w(i,j,k,l) = exp\left(-\frac{(i-k)^2+(j-l)^2}{2\sigma_c^2} - \frac{\|I(i,j)-I(k,l)\|^2}{2\sigma_s^2}\right)$$

where (k,l) is the neighboring pixel location within the window centered around (i,j), I is the image with noise, Y is the filtered image. σ_c and σ_s are the spread parameters.

(1) Implement the bilateral denoising filter and apply it to the noisy image.
(2) Explain the roles of σ_c and σ_s. Discuss the change in filter's performance with respect to the values of σ_c and σ_s.
(3) Does this filter perform better than linear filters you implemented in Problem 3(a)? Justify your answer in words.

(c) Non-Local Means (NLM) Filtering (10%)

The non-local mean filter utilizes the pixel value from a larger region rather the mean of a local window centered around the target pixel. A discrete non-local mean filter with Gaussian weighting function is as follows:

$$Y(i,j) = \frac{\sum_{k=1}^{N'}\sum_{l=1}^{M'} I(k,l)f(i,j,k,l)}{\sum_{k=1}^{N'}\sum_{l=1}^{M'} f(i,j,k,l)}$$

$$f(i,j,k,l) = \exp\left(-\frac{\left\|I(N_{i,j}) - I(N_{k,l})\right\|_{2,a}^2}{h^2}\right)$$

where I, Y are the noisy and filtered images respectively, $N_{x,y}$ is the window centered around location (x,y), and h is the filtering parameter, $N' \leq N$ and $M' < M$ denote the window size of your choice.
The Gaussian weighted Euclidian distance between window $I(N_{i,j})$and $I(N_{k,l})$ is defined as:

$$\left\|I(N_{i,j}) - I(N_{k,l})\right\|_{2,a}^2 = \sum_{n_1,n_2 \in \aleph} G_a(n_1,n_2)\left(I(i-n_1,j-n_2) - I(k-n_1,l-n_2)\right)^2$$

$$G_a(n_1, n_2) = \frac{1}{\sqrt{2\pi}a} \exp\left(-\frac{n_1^2 + n_2^2}{2a^2}\right)$$

where $\aleph$ denotes the local neighborhood centered at the origin, $n_1, n_2 \in \aleph$ denotes the relative position in the neighborhood window. $a > 0$ is the standard deviation of the Gaussian kernel.

(1) Implement the NLM filter and apply it to the noisy image. Try several filter parameters and discuss their effect on filtering process. Clearly state your final choice of parameters in your report.
(2) Compare the performance of NLM with filters used in Problem 3(a) and Problem 3(b).

(d) Block matching and 3-D (BM3D) transform filter (10%)

In this part, you will get familiar with another state-of-the-art denoising algorithm proposed in [3].

(1) Please explain the BM3D algorithm in your own words.
(2) Implement the BM3D filter (Write your own code or use any available online source code but include the source in your reference) to denoise the noisy image *House* in Figure 7. Show the final noise-removed image, and compare the PSNR value and visual quality with those in (a).

Note: It is recommended that you use the code provided by the authors on their website [4]. Their code is written in MATLAB.

(e) Mixed noises in color image (10%)

Figure 8 (b) is a noisy color image corrupted with mixed types of noises. Please identify noise types in the image and answer the following questions:

(1) What types of noises are there?
(2) Should you perform filtering on individual channels separately for both noise types?
(3) What filters would you like use to remove mixed noise? Can you cascade these filters in any order? Justify your answer.

(a)

(b)

Figure 8: (a) the original pepper image (b) the pepper image with mixed noises.

Appendix:

Problem 1: Image Demosaicing and Histogram Manipulation

Dog.raw	600x532	8-bit	gray
Dog_ori.raw	600x532	24-bit	color(RGB)
Toy.raw	560x400	24-bit	color(RGB)

Problem 2: Image Denoising

Corn_gray.raw	320x320	8-bit	gray
Corn_noisy.raw	320x320	8-bit	gray

Reference Images

All images in this homework are from Google images [5] or the USC-SIPI image database [6].

References

[1] M. E. Celebi et al. (eds.), Color Image and Video Enhancement.

[2] Malvar, Henrique S., Li-wei He, and Ross Cutler. "High-quality linear interpolation for demosaicing of Bayer-patterned color images." *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 3. IEEE, 2004.

[3] Dabov, Kostadin, et al. "Image denoising with block-matching and 3D filtering." *Image Processing: Algorithms and Systems, Neural Networks, and Machine Learning*. Vol. 6064. International Society for Optics and Photonics, 2006.

[4] [Online] Available: http://www.cs.tut.fi/~foi/GCF-BM3D/

[5] [Online] http://images.google.com/

[6] [Online] http://sipi.usc.edu/database/