ARIZONA STATE UNIVERSITY
SCHOOL OF ELECTRICAL, COMPUTER, AND ENERGY ENGINEERING

EEE 508 Spring 2017
Hands-On Assignment #2

Assigned: 2 February 2017
Due Date: 16 February 2017

Reading: Read Chapters 3, 4, and 6 in the OpenCV book.

EXERCISE 2.1:

(5 points) Perform the following experiment with MATLAB (*For help, use the commands* **help** *and* **lookfor***; for information about these commands, run Matlab and type* **help help** *and* **help lookfor**):

1. First, you need to download the image **cameraman.jpg** from the Course Web Page under Weekly Schedule by clicking on IMAGES.

2. Run Matlab and load the selected image using the **imread()** function.
   Note: you might need to include your working directory in the Matlab PATH – type **help path** for more information on how to do this.

3. Display the image using the **imshow()** function.

4. Generate the following 3×3 2-D filter impulse response as a matrix in Matlab:

   $$h(n_1, n_2) = \begin{cases} 1/9, & 0 \leq n_1, n_2 \leq 2 \\ 0, & \text{elsewhere} \end{cases}$$

   Call the generated matrix **h**.

5. Use the function **conv2()** to filter the image using the filter with impulse response $h(n_1, n_2)$. Call the resulting image **y1**. Display and generate a printout of **y1**.

6. Repeat the above step with the **filter2()** command. Call the result **y2**. Compare **y1** and **y2**. Using the **type()** command, list and study the function **filter2()**.

7. Show that $h(n_1, n_2)$ is separable; using Matlab, generate the resulting filtered image using the 1-D **conv()** function. Call the result **y3**. Display and print **y3**. Compare **y3** and **y1**.

8. Is the filter of Step 4 $h(n_1, n_2)$ lowpass or highpass? (JUSTIFY your answer). If it is highpass, show how you can transform it into a lowpass filter with impulse response $h_2(n_1, n_2)$; otherwise, show how you can transform it into a highpass filter $h_2(n_1, n_2)$. Scale $h_2(n_1, n_2)$ properly so that the image mean is unchanged by filtering (Hint: mean of image is multiplied by $H_2(0, 0)$, the Discrete-Time Fourier Transform of $h_2(n_1, n_2)$ evaluated at $\omega_1 = \omega_2 = 0$).

9. Store the resulting filter impulse response $h_2(n_1, n_2)$ as a Matlab matrix **h2**. Repeat Step 5 using **h2** instead of **h**. Call the resulting filtered image **y4**. Display and print **y4**. What is the effect of filtering with $h_2(n_1, n_2)$?

10. Submit on Blackboard under Assignments a zipped folder called Matlab_yourname.zip, where yourname is replaced by your actual name, by attaching the zipped folder using the "Browse for Local File" button. The zipped folder should contain a pdf file with answers to the questions above, your Matlab code implementing Steps 2 to 9, and outputs generated by the code.

EXERCISE 2.2:

(5 points) Repeat the experiment in Problem 1 with Python (Python installation instructions are provided on Blackboard):

1. First, you need to download the image **cameraman.jpg** from the Course Web Page under Weekly Schedule by clicking on IMAGES.
2. Run your Python IDE (e.g., Spyder) and load the selected image using the **mpimg.imread()** function.
   Note: you need to include your working directory for opening the image. Also, you must define the modules that you need before calling the function. For instance, to use function **mpimg.imread()**, you first need to define
   import matplotlib.image as mpimg
   For this exercise, also define the following modules:
   import matplotlib.pyplot as plt
   import numpy as np
   from scipy import ndimage
3. Display the image using the **plt.imshow()** function. (Make sure to define the module **import matplotlib.pyplot** as **plt** first)
4. Generate the following 3×3 2-D filter impulse response as a matrix in Python:

$$h(n_1, n_2) = \begin{cases} 1/9, & 0 \leq n_1, n_2 \leq 2 \\ 0, & \text{elsewhere} \end{cases}$$

   Call the generated matrix **h**.
5. Use the function **signal.convolve2d()** to filter the image using the filter with impulse response $h(n_1, n_2)$. Call the resulting image **y1**. Display and generate a printout of **y1**.
6. Repeat the above step with the **ndimage.convolve()** command. Call the result **y2**. Compare **y1** and **y2**. Using the **type()** command, list and study the function **filter2()**.
7. Generate the resulting filtered image using the 1-D **signal.convolve()** function. Call the result **y3**. Display and print **y3**. Compare **y3** and **y1**.
8. If $h(n_1, n_2)$ is a lowpass filter, show how you can transform it into a highpass filter $h_2(n_1, n_2)$; otherwise, show how you can transform it into a lowpass filter. Scale $h_2(n_1, n_2)$ properly so that the image mean is unchanged by filtering (Hint: mean of image is multiplied by $H_2(0, 0)$, the Discrete-Time Fourier Transform of $h_2(n_1, n_2)$ evaluated at $\omega_1 = \omega_2 = 0$).
9. Store the resulting filter impulse response $h_2(n_1, n_2)$ as a matrix **h2**. Repeat Step 5 using **h2** instead of **h**. Call the resulting filtered image **y4**. Display and print **y4**. What is the effect of filtering with $h_2(n_1, n_2)$?

10. Submit on Blackboard under Assignments a zipped folder called Python_yourname.zip, where yourname is replaced by your actual name, by attaching the zipped folder using the "Browse for Local File" button. The zipped folder should contain your Python code implementing Steps 2 to 9, and outputs generated by the code.

EXERCISE 2.3:

(6 points) Perform the following experiment with OpenCV:

(a) First, you need to download the 256×256 image **zebras.jpg** from the Course Web Page under Weekly Schedule by clicking on IMAGES.

(b) Load and display the selected image as described in Chapter 2 of the OpenCV book.

(c) Compute the two-dimensional 256×256 DFT matrix $F(K_1, K_2)$ of the loaded image as discussed in Chapter 6 of the OpenCV book (or lookup online for C++ version or the latest OpenCV version). Truncate the DFT matrix $F(K_1, K_2)$ by setting the DFT coefficients corresponding to $L \leq K_1 \leq (255 - L + 1)$ and $L \leq K_2 \leq (255 - L + 1)$ to zero, where L is a suitably selected integer value. Compute and display the corresponding image by taking the inverse DFT of the truncated DFT matrix.
Find the value $L_{min,DFT}$ by decreasing the value of L until the image starts to get noticeably degraded; report the obtained value $L_{min,DFT}$ and display the corresponding image.

(d) Compute the two-dimensional DCT matrix $D(K_1, K_2)$ of the loaded image as discussed in Chapter 6 of the OpenCV book. Truncate the DCT matrix $D(K_1, K_2)$ by setting the DCT coefficients corresponding to $K_1 \geq L$ and $K_2 \geq L$ to zero (keep only the first coefficients corresponding to $0 \leq K_1 < L$ and $0 \leq K_2 < L$), where L is is a suitably selected integer value. Compute and display the corresponding image by taking the inverse DCT of the truncated DCT matrix.
Find the value $L_{min,DCT}$ by decreasing the value of L until the image starts to get noticeably degraded; report the obtained value $L_{min,DCT}$ and display the corresponding image.

Submit the implemented code, a Readme file containing compilation and running instructions, the input image and the reconstructed images named after their corresponding L, $L_{min,DFT}$, and $L_{min,DCT}$ values, as a zipped folder called DFT_DCT_yourname.zip under Assignments in Blackboard.