Homework 4

ECE 253
Digital Image Processing

November 13, 2015

Instructions :

- Homework 4 is due by 11:59 PM, November 17, 2015.
- Submit your homework electronically by email to arangesh@ucsd.edu with the subject line *ECE 253 HW4*.
- The email should have one PDF file of your writeup attached. Make sure it includes your full name, PID, and email. This file must be named ECE_253_hw4_lastname_studentid.pdf.
- All problems are to be solved using MATLAB unless mentioned otherwise.
- You should avoid using loops in your MATLAB code unless you are explicitly permitted to do so.
- Append source code, floats, matrices, and image outputs, to your writeup where applicable. Simply pasting your code in the report should suffice, but make sure it is indented correctly.

Problem 1. 2D Discrete Fourier Transforms

(i) **Basic Images**
Read in each of the following images and pad them with zeros to size 512x512 (you may pad them to the right and below of the image). Then compute the 2D DFT for each padded image (using fft2). Include the unpadded original image alongside the corresponding 2D DFT magnitude directly next to each other. When displaying the 2D DFT magnitude, use *fftshift* to move the DC component to the center and use

```
imagesc(...); colorbar;
```

- FreqHorizLow.png
- FreqHorizMed.png
- FreqHorizHigh.png
- FreqVertMed.png
- FreqAngle1.png
- FreqAngle2.png
- FreqAngle3.png

Note: Instead of manually padding, you may also input the size into fft2. Type *help fft2*. Do not accidentally use fft in place of fft2. Subplot highly recommended.

(ii) **Checkerboard Image**
Repeat 1(i) for *FreqCheckerboard.png*. If you have taken a DSP class in the past, a 1-D square wave signal is composed of a base frequency along with repeating decaying harmonics of that frequency. You should notice this in the 2-D DFT of the checkerboard image as well.

Inspecting the 2D DFT magnitude of the checkerboard along with the 2D DFT magnitude results from 1(i)... Which two images when summed together, e.g.

```
newimage = double(image1)+double(image2);
```

would best match the checkerboard image? Read those two images in, convert them to double-type, sum them together, and repeat 1(i) using this new image. Are the results what you had expected? Why or why not?

(iii) **Frequency Filters**

From the previous assignment, we saw the unsharp function:

```
function im_out = unsharp( im_in, maskA, weight )
[a,b] = size(maskA)
maskB = zeros( size(maskA) )
maskB(ceil(a/2), ceil(b/2)) = 1
maskC = maskB - maskA
maskD = maskB + weight * maskC
im_out = conv2(im_in, maskD, same)
```

Given the low pass filter:

```
maskA = (1/16)*[1 2 1; 2 4 2; 1 2 1];
```

Compute the 2D DFT (zero padding to size 512x512) for maskA, maskB, maskC, and maskD, and display their 2D DFT magnitude. Label the figures with as maskA, ..., maskD as well as their filter type, e.g. low-pass filter, identity filter, high-pass filter, high-boost filter.

Things to turn in:

- All images should have colorbars next to them
- All DFT magnitude images should have the DC frequencies in the center of the image.
- 14 images from 1(i): 7 unpadded images and their corresponding 2D DFT magnitude
- 4 images from 1(ii): 2 unpadded images and their corresponding 2D DFT magnitude
- 4 images from 1(iii): maskA-D's 2D DFT magnitude, properly labeled.
- Response for 1(ii)
- Code for 1(i), 1(ii), 1(iii)

Problem 2. Butterworth Notch Reject Filtering in Frequency Domain

This problem will follow Figure 4.64 in section 4.10.2 Notch Filters of Gonzalez & Woods 3rd Edition.

(i) Read in the image *Car.tif*, pad the image to 512x512, and display the 2D-FFT log magnitude (after moving the DC component to the center with fftshift):

```
imagesc(-256:255,-256:255,log(abs(imFFT))); colorbar;
xlabel('u'); ylabel('v');
```

You should see symmetric "impulse-like" bursts which we suspect is the cause of the Moire Pattern (the dot pattern from the newspaper image). We would like to filter out this pattern in the frequency domain using a Butterworth Notch Reject Filter given by:

$$H_{NR}(u,v) = \prod_{k=1}^{K} \left[\frac{1}{1+[D_0/D_k(u,v)]^{2n}}\right]\left[\frac{1}{1+[D_0/D_{-k}(u,v)]^{2n}}\right] \tag{1}$$

where

$$D_k(u,v) = \left[(u-u_k)^2 + (v-v_k)^2\right]^{1/2} \tag{2}$$

$$D_{-k}(u,v) = \left[(u+u_k)^2 + (v+v_k)^2\right]^{1/2} \tag{3}$$

Here, we have slightly modified the definition from the textbook slightly in that we removed $M/2$ and $N/2$ from the equation so that the center of the DFT image is 0 rather than $(M/2, N/2)$.

The parameter **K** is the number of "impulse-like" bursts you would like to remove (excluding their symmetric counterparts since the H_{NR} equation already includes it), e.g. 4 for this image rather than 8. The parameters **n** and $\mathbf{D_0}$ are scalar values that you may choose to control the transition and radius of the notch filters. The parameter $\mathbf{u_k}$ and $\mathbf{v_k}$ is the location of the k^{th} "impulse-like" bursts in the 2D DFT magnitude image. **u** and **v** are all possible (u,v) coordinate pairs that you want to generate H_{NR} for:

```
[u,v] = meshgrid(-256:255);
```

(ii) Repeat for *Street.png*, except for K=2 and remove the burst along the $u = 0$ axis and the $v = 0$ axis.

Things to turn in:

- All images should have colorbars next to them
- All DFT magnitude images should have the DC frequencies in the center of the image.
- 4 images from 2(i): 1 unpadded original image, the corresponding 2D DFT log-magnitude, the butterworth Notch Reject Filter in frequency domain $H_{NR}(u,v)$, the final filtered image.
- 10 parameters for 2(i): n, D_0, u_1, v_1, ..., u_4, v_4

- 4 images from 2(ii): 1 unpadded original image, the corresponding 2D DFT log-magnitude, the butterworth Notch Reject Filter in frequency domain $H_{NR}(u, v)$, the final filtered image.
- 6 parameters for 2(ii): n, D_0, u_1, v_1, u_2, v_2
- Code for 2(i), 2(ii)

Problem 3. Template Matching

(i) **Cross-correlation filter on a simple image**
Read in image *Letters.jpg* and template *LettersTemplate.jpg* and convert to grayscale, double type. Perform cross-correlation on the image with the template using convolution in both spatial and frequency domain. Display the resulting filtered images from both methods.

Note: you may need to shift the resulting image from the frequency domain method.

The following functions may be useful:
conv2, fft2, ifft2, circshift.

(ii) **Cross-correlation filter on a realistic image**
Repeat step (i) for image *StopSign.jpg* and template *StopSignTemplate.jpg.*

(iii) **Normalized cross-correlation**
From lecture, cross-correlation for template matching was shown to not work very well on an actual image. Apply normalized cross-correlation on the stop sign image using the corresponding template and display the resulting image with a colorbar. Also display the original image with a with a rectangular box (the same size as the template, similar to the lecture slides with the faces) at the location with the highest normalized cross-correlation score.

The following functions may be useful:
normxcorr2, rectangle or plot

(iv) Using the same template *StopSignTemplate.jpg* on another image with a stop sign taken at a much different camera angle or much closer/farther away, would normalized cross-correlation template matching still perform as well? Why or why not?

Things to turn in:

- All images should have colorbars next to them
- 2 images from 3(i): cross-correlation images from spatial and frequency methods.
- 2 images from 3(ii): cross-correlation images from spatial and frequency methods
- 2 images from 3(iii): normalized cross-correlation image and original image with rectangular box overlayed
- Response for 3(iv)
- Code for 3(i), 3(ii), 3(iii)