ECE 533 FALL 2015 LAB 2
THE TWO-DIMENSIONAL FOURIER TRANSFORM

In this lab, we will explore the 2-dimensional Fourier transform, develop our intuition for the Fourier coefficients and their meanings, and perform linear filtering operations in the Fourier domain.

You may use either MATLAB or Python for this and all coursework. However, examples will be provided in MATLAB.

A. The 2d DFT

1. What is the 2d DFT of a δ (delta) function?
2. What is the 2d DFT of a Gaussian function (approximately)?
3. What is the 2d DFT of a rectangle function?

$$\text{rect}[m,n] = \begin{cases} 1, & |m| < 1 \text{ and } |n| < 1 \\ 0, & \text{otherwise} \end{cases}$$

4. **Roles of different coefficients:**
   (a) Compute the 2d DFT of an image, set the DC (zero-frequency) component to zero, and then compute the 2d Inverse DFT. What do you observe? Provide a mathematical explainanation for what you see.
   (b) Compute the 2d DFT of an image, set `F[0,k]=0` for all k, and then compute the 2d Inverse DFT. What do you observe? Provide a mathematical explanation for what you see.
   (c) Compute the 2d DFT of an image, set `F[k,0]=0` for all k, and then compute the 2d Inverse DFT. What do you observe?
   (d) Compute the 2d DFT of an image, set $F[k_1, k_2] = 0$ if $|k_2| > c$ for some (small) constant c, and then compute the 2d Inverse DFT. What do you observe?
   (e) Compute the 2d DFT of an image, set $F[k_1, k_2] = 0$ if $|k_1| > c$ for some (small) constant c, and then compute the 2d Inverse DFT. What do you observe?
   (f) Compute the 2d DFT of an image, set $F[k_1, k_2] = 0$ if $|k_2| > c$ OR $|k_1| > c$ for some (small) constant c, and then compute the 2d Inverse DFT. What do you observe?
   (g) Compute the 2d DFT of an image, set $F[k_1, k_2] = 0$ if $|k_2| < c$ AND $|k_1| < c$ for some (small) constant c, and then compute the 2d Inverse DFT. What do you observe?
5. **Translation and rotation:**
   (a) Demonstrate how you can translate an image in the Fourier domain. Start with the following code snippet:

```
M = 256;
[x,y] = meshgrid(linspace(-2,2,M));
f1 = rect(y).*rect(5*x);
f2 = circshift(f1,round([-M/4,M/4]));
[u,v] = meshgrid(0:(M-1));
Fd = fft2(f1).*???
f3 = real(ifft2(Fd));
figure(1);imagesc([f1; f2; f3]);axis image;colormap gray;
```

(b) Demonstrate how you can rotate an image in the Fourier domain. Start with the following code snippet:

```
M = 256;
[x,y] = meshgrid(linspace(-2,2,M));
f1 = rect(y).*rect(5*x);
f2 = imrotate(f1,45,'crop');
F1 = fftshift(fft2(fftshift(f1)));
F3 = fftshift(???);
f3 = fftshift(abs(ifft2(F3)));
figure(1);imagesc([f1; f2; f3]);axis image;colorbar
```

6. **Magnitude and phase:**

(a) Compute the 2d DFT of an image of your choice. Store the magnitude and phase information separately:

```
IM1 = fft2(im1);
mag1 = abs(IM1);
phase1 = angle(IM1);
im1_test = real(ifft2(mag1.*exp(i*phase1)));
```

(b) Replace `mag1` with random numbers (leaving the phase intact) and reconstruct the image. What do you see?

(c) Replace `phase1` with random numbers (leaving the magnitude intact) and reconstruct the image. What do you see?

(d) Compute the magnitude and phase of the DFT coefficients of a second image of the same size. Swap their two phases and reconstruct the two images separately. What do you see?

7. k**-term approximations:** In this exercise, we will explore approximating our image with k Fourier components for different k.

(a) Write a function that lets you input an image and an integer k, and outputs a new image which approximates the input image with k DFT coefficients. Here is some code to help you get started:

```
f = double(rgb2gray(imread('ImageName.jpg')));
figure;imagesc(f);axis image;colormap gray;axis off
F = fft2(f);
```

(b) If you choose the k DFT coefficients by choosing those with the largest magnitudes, how does the mean squared error (MSE) of the approximation vary as a function of k? Generate a plot using a test image of your choice. Describe your observations.

(c) Repeat the above experiment, but use the DFT coefficients of a *noisy* version of your image and compute the MSE relative to the *original, noise-free* version of your image.

(d) Can you think of an alternative way to choose the k DFT coefficients? How does the MSE performance of your method compare with the MSE performance of the magnitude-based method? Why?

8. Write a function which computes the 2d DFT of an image. Compare your result with the output of the MATLAB or Python 2d DFT command. (They should be identical.) HINT: it is possible (and faster!) to do this without any `for` loops. +10 bonus points if you complete this without loops.

B. Fourier-domain filtering

1. In the first part of this lab, you examined the 2d DFT of a rect function and a Gaussian function. Now consider filtering an image with either of these functions. What does the Fourier content of the two filters tell you about their respective effect on an image? (HINT: Using the TestImage with a filter with a large neighborhood can be helpful.)

2. Filter an image with an *ideal low-pass filter*:

$$H[k,\ell] = \begin{cases} 1, & \text{if } k^2 + \ell^2 \leq D_0 \\ 0, & \text{otherwise.} \end{cases}$$

What is its effect on an image for different values of D_0? Note that by applying this ideal LPF in the Fourier domain, you are convolving with a corresponding filter in the spatial domain. What is the corresponding spatial-domain filter? How does its structure correspond to effects and artifacts you see in the filtered images?

3. Consider the below two zero-padded images. Is there a reason to prefer one type of padding over the other when performing filtering via the DFT?


4. Design a high-pass filter to emphasize the ridges in the ThumbPrint image.

5. Design a band-reject filter to remove the artifacts in the MoirePattern image.

6. Design a band-reject filter to remove the artifacts in the PeriodicInterference image.

7. A binary image contains straight lines oriented horizontally, vertically, at 45° and at -45°. Give a set of 3×3 masks that can be used to detect 1-pixel breaks in these lines. Assume that the intensities

of the lines and background are 1 and 0, respectively. Here is a code snippet you can use to generate such an image:

```
M = 64;
f = zeros(64);
f(5,:) = 1;
f(36,:) = 1;
f(59,:) = 1;
f(:,10) = 1;
f(:,24) = 1;
f(:,45) = 1;
k = (1:M:(M^2)) + (1:M);k = k(find((k>0).*(k<=M^2)));
f(k) = ones(size(k));
k = (1:M:(M^2)) - (1:M);k = k(find((k>0).*(k<=M^2)));
f(k) = ones(size(k));
noise = imnoise(f,'salt & pepper',0.05);
f(find(noise==0)) = 0;
figure(2);imagesc(f);axis image; colormap gray;colorbar
```

C. Homework problems

1. Consider a 3×3 spatial mask that averages the four closest neighbors of a point $[m, n]$, but excludes the point itself from the average. Find the equivalent filter $H[k, \ell]$ in the frequency domain.

2. Book question 4.22.

3. Book question 4.33.

4. Book question 4.36.

5. Assume that the Sobel masks are used to obtain g_x and g_y. Show that in this case, the magnitude of the gradient computed using

$$M_1(x, y) = \sqrt{g_x^2 + g_y^2}$$

and

$$M_2(x, y) = |g_x| + |g_y|$$

give identical results.

D. Deadline

Reports, including images, descriptions, and code, should be turned in via Moodle. Turn in the result of "publish" in Matlab or iPython Notebooks. Written problem solutions should also be submitted digitally. If you photograph or scan hand-written solutions, make sure they are legible. Due by 2:30pm on Sept. 21.