(Gonzalez 3rd edition)

1. Problem 4.14 (10%)

From Eq. (4.5-7),

$$F(\mu,\nu)=\Im\left[f(t,z)\right]=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(t,z)e^{-j2\pi(\mu t+\nu z)}dt\,dz.$$

From Eq. (2.6-2), the Fourier transform operation is linear if

$$\Im\left[a_1f_1(t,z)+a_2f_2(t,z)\right]=a_1\Im\left[f_1(t,z)\right]+a_2\Im\left[f_2(t,z)\right].$$

Substituting into the definition of the Fourier transform yields

$$\begin{aligned}\Im\left[a_1f_1(t,z)+a_2f_2(t,z)\right] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[a_1f_1(t,z)+a_2f_2(t,z)\right] \\ &\quad\times e^{-j2\pi(\mu t+\nu z)}dt\,dz \\ &= a_1\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(t,z)e^{-j2\pi(\mu t+\nu z)}dt\,dz \\ &\quad + a_2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f_2(t,z)e^{-j2\pi(\mu t+\nu z)}dt\,dz \\ &= a_1\Im\left[f_1(t,z)\right]+a_2\Im\left[f_2(t,z)\right].\end{aligned}$$

where the second step follows from the distributive property of the integral. Similarly, for the discrete case,

$$\begin{aligned}\Im\left[a_1f_1(x,y)+a_2f_2(x,y)\right] &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[a_1f_1(x,y)+a_2f_2(x,y)\right]e^{-j2\pi(ux/M+vy/N)} \\ &= a_1\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f_1(x,y)e^{-j2\pi(ux/M+vy/N)} \\ &\quad + a_2\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f_2(x,y)e^{-j2\pi(ux/M+vy/N)} \\ &= a_1\Im\left[f_1(x,y)\right]+a_2\Im\left[f_2(x,y)\right].\end{aligned}$$

The linearity of the inverse transforms is proved in exactly the same way.

$$F(\mu,\nu)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(t,z)\,e^{-j2\pi(\mu t+\nu z)}dt\,dz \qquad (4.5\text{-}7)$$

$$\begin{aligned}H\left[a_if_i(x,y)+a_jf_j(x,y)\right] &= a_iH\left[f_i(x,y)\right]+a_jH\left[f_j(x,y)\right] \\ &= a_ig_i(x,y)+a_jg_j(x,y)\end{aligned} \qquad (2.6\text{-}2)$$

2. Problem 4.15 (10%)

Select an image of your choice and compute its average value:

$$\bar{f}(x,y) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y).$$

Compute the DFT of $f(x,y)$ and obtain $F(0,0)$. If $F(0,0) = MN\bar{f}(x,y)$, then $1/MN$ was included in front of the IDFT [see Eqs. (4.5-15), (4.5-16) and (4.6-21)]. Similarly, if $F(0,0) = \bar{f}(x,y)$ the 1/MN term was included in front of the DFT. Finally, if $F(0,0) = \sqrt{MN}\bar{f}(x,y)$, the term $1/\sqrt{MN}$ was included in the formulation of both the DFT and IDFT.

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\,e^{-j2\pi(ux/M+vy/N)} \qquad (4.5\text{-}15)$$

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)\,e^{j2\pi(ux/M+vy/N)} \qquad (4.5\text{-}16)$$

$$\begin{aligned} F(0,0) &= MN\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) \\ &= MN\bar{f}(x,y) \qquad (4.6\text{-}21) \end{aligned}$$

3. Problem 4.22 (10%)

Unless all borders on of an image are black, padding the image with 0s introduces significant discontinuities (edges) at one or more borders of the image. These can be strong horizontal and vertical edges. These sharp transitions in the spatial domain introduce high-frequency components along the vertical and horizontal axes of the spectrum.

4. Problem 4.23 (10%)

(a) The averages of the two images are computed as follows:

$$\bar{f}(x,y) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)$$

and

$$
\begin{aligned}
\bar{f}_p(x,y) &= \frac{1}{PQ}\sum_{x=0}^{P-1}\sum_{y=0}^{Q-1} f_p(x,y) \\
&= \frac{1}{PQ}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) \\
&= \frac{MN}{PQ}\bar{f}(x,y)
\end{aligned}
$$

where the second step is result of the fact that the image is padded with 0s. Thus, the ratio of the average values is

$$
r = \frac{PQ}{MN}
$$

Thus, we see that the ratio increases as a function of PQ, indicating that the average value of the padded image decreases as a function of PQ. This is as expected; padding an image with zeros decreases its average value.

(b) Yes, they are equal. We know that $F(0,0) = MN\bar{f}(x,y)$ and $F_p(0,0) = PQ\bar{f}_p(x,y)$. And, from part (a), $\bar{f}_p(x,y) = MN\bar{f}(x,y)/PQ$. Then,

$$
\begin{aligned}
\frac{F_p(0,0)}{PQ} &= \frac{MN}{PQ}\frac{F(0,0)}{MN} \\
F_p(0,0) &= F(0,0).
\end{aligned}
$$

5. Problem 4.26 (10%)

(a) From Chapter 3, the Laplacian of a function $f(t,z)$ of two continuous variables is defined as

$$
\nabla^2 f(t,z) = \frac{\partial^2 f(t,z)}{\partial t^2} + \frac{\partial^2 f(t,z)}{\partial z^2}.
$$

We obtain the Fourier transform of the Laplacian using the result from Problem 4.25 (entry 12 in Table 4.3):

$$
\begin{aligned}
\Im\left[\nabla^2 f(t,z)\right] &= \Im\left[\frac{\partial^2 f(t,z)}{\partial t^2}\right] + \Im\left[\frac{\partial^2 f(t,z)}{\partial z^2}\right] \\
&= (j2\pi\mu)^2 F(\mu,\nu) + (j2\pi\nu)^2 F(\mu,\nu) \\
&= -4\pi^2(\mu^2+\nu^2)F(\mu,\nu).
\end{aligned}
$$

We recognize this as the familiar filtering expression $G(\mu,\nu) = H(\mu,\nu)F(\mu,\nu)$, in which $H(\mu,\nu) = -4\pi^2(\mu^2+\nu^2)$.

(b) As the preceding derivation shows, the Laplacian filter applies to *continuous* variables. We can generate a filter for using with the DFT simply by sampling this function:

$$H(u,v) = -4\pi^2(u^2+v^2)$$

for $u = 0,1,2,\ldots,M-1$ and $v = 0,1,2,\ldots,N-1$. When working with centered transforms, the Laplacian filter function in the frequency domain is expressed as

$$H(u,v) = -4\pi^2([u-M/2]^2 + [v-N/2]^2).$$

In summary, we have the following Fourier transform pair relating the Laplacian in the spatial and frequency domains:

$$\nabla^2 f(x,y) \Leftrightarrow -4\pi^2([u-M/2]^2 + [v-N/2]^2)F(u,v)$$

where it is understood that the filter is a sampled version of a continuous function.

(c) The Laplacian filter is isotropic, so its symmetry is approximated much closer by a Laplacian mask having the additional diagonal terms, which requires a -8 in the center so that its response is 0 in areas of constant intensity.

TABLE 4.3 (*Continued*)

Name	DFT Pairs
7) Correlation theorem†	$f(x,y) \star h(x,y) \Leftrightarrow F^*(u,v)\,H(u,v)$ $f^*(x,y)h(x,y) \Leftrightarrow F(u,v) \star H(u,v)$
8) Discrete unit impulse	$\delta(x,y) \Leftrightarrow 1$
9) Rectangle	$\text{rect}[a,b] \Leftrightarrow ab\dfrac{\sin(\pi ua)}{(\pi ua)}\dfrac{\sin(\pi vb)}{(\pi vb)}e^{-j\pi(ua+vb)}$
10) Sine	$\sin(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $j\frac{1}{2}\left[\delta(u+Mu_0, v+Nv_0) - \delta(u-Mu_0, v-Nv_0)\right]$
11) Cosine	$\cos(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $\frac{1}{2}\left[\delta(u+Mu_0, v+Nv_0) + \delta(u-Mu_0, v-Nv_0)\right]$
The following Fourier transform pairs are derivable only for continuous variables, denoted as before by t and z for spatial variables and by μ and ν for frequency variables. These results can be used for DFT work by sampling the continuous forms.	
12) *Differentiation* (The expressions on the right assume that $f(\pm\infty, \pm\infty) = 0$.)	$\left(\dfrac{\partial}{\partial t}\right)^m\left(\dfrac{\partial}{\partial z}\right)^n f(t,z) \Leftrightarrow (j2\pi\mu)^m(j2\pi\nu)^n F(\mu,\nu)$ $\dfrac{\partial^m f(t,z)}{\partial t^m} \Leftrightarrow (j2\pi\mu)^m F(\mu,\nu);\ \dfrac{\partial^n f(t,z)}{\partial z^n} \Leftrightarrow (j2\pi\nu)^n F(\mu,\nu)$
13) *Gaussian*	$A2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)} \Leftrightarrow Ae^{-(\mu^2+\nu^2)/2\sigma^2}$ (A is a constant)

†Assumes that the functions have been extended by zero padding. Convolution and correlation are associative, commutative, and distributive.

6. Problem 4.27 (10%)

(a) The spatial average (excluding the center term) is

$$g(x,y)=\frac{1}{4}\left[f(x,y+1)+f(x+1,y)+f(x-1,y)+f(x,y-1)\right].$$

From property 3 in Table 4.3,

$$\begin{aligned} G(u,v) &= \frac{1}{4}\left[e^{j2\pi v/N}+e^{j2\pi u/M}+e^{-j2\pi u/M}+e^{-j2\pi v/N}\right]F(u,v) \\ &= H(u,v)F(u,v) \end{aligned}$$

where

$$H(u,v)=\frac{1}{2}\left[\cos(2\pi u/M)+\cos(2\pi v/N)\right]$$

is the filter transfer function in the frequency domain.

(b) To see that this is a lowpass filter, it helps to express the preceding equation in the form of our familiar centered functions:

$$H(u,v)=\frac{1}{2}\left[\cos(2\pi[u-M/2])/M)+\cos(2\pi[v-N/2]/N)\right].$$

Consider one variable for convenience. As u ranges from 0 to $M-1$, the value of $\cos(2\pi[u-M/2]/M)$ starts at -1, peaks at 1 when $u=M/2$ (the center of the filter) and then decreases to -1 again when $u=M$. Thus, we see that the amplitude of the filter decreases as a function of distance from the origin of the centered filter, which is the characteristic of a lowpass filter. A similar argument is easily carried out when considering both variables simultaneously.

7. Problem 4.28 (10%)

(a) As in Problem 4.27, the filtered image is given by:

$$g(x,y)=f(x+1,y)-f(x,y)+f(x,y+1)-f(x,y).$$

From property 3 in Table 4.3,

$$\begin{aligned} G(u,v) &= F(u,v)e^{j2\pi u/M}-F(u,v)+F(u,v)e^{j2\pi v/N}-F(u,v) \\ &= [e^{j2\pi u/M}-1]F(u,v)+[e^{j2\pi v/N}-1]F(u,v) \\ &= H(u,v)F(u,v) \end{aligned}$$

where $H(u,v)$ is the filter function:

$$\begin{aligned} H(u,v) &= \left[(e^{j2\pi u/M}-1)+(e^{j2\pi v/N}-1)\right] \\ &= 2j\left[\sin(\pi u/M)e^{j\pi u/M}+\sin(\pi v/N)e^{j\pi v/N}\right]. \end{aligned}$$

(b) To see that this is a highpass filter, it helps to express the filter function in the form of our familiar centered functions:

$$H(u,v) = 2j\left[\sin(\pi[u - M/2]/M)e^{j\pi u/M} + \sin(\pi[v - N/2]/N)e^{j\pi v/N}\right].$$

The function is 0 at the center of the filter $u = M/2$). As u and v increase, the value of the filter decreases, reaching its limiting value of close to $-4j$ when $u = M-1$ and $v = M-1$. The negative limiting value is due to the order in which the derivatives are taken. If, instead we had taken differences of the form $f(x,y) - f(x+1,y)$ and $f(x,y) - f(x,y+1)$, the filter would have tended toward a positive limiting value. The important point here is that the dc term is eliminated and higher frequencies are passed, which is the characteristic of a highpass filter.

8. **Fourier Spectrum and Average Value** (10%)
   (a) Download Fig. 4.41(a) from the course web site and compute its (centered) Fourier spectrum.
   (b) Display the spectrum.
   (c) Use your result in (a) to compute the average value of the image.

Sol:

(a)

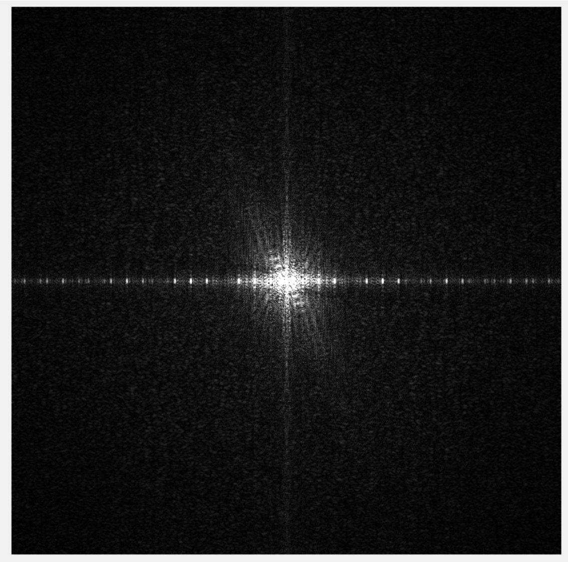
(Matlab code)

```
clear all;clc;
img=imread('Fig0441(a)(characters_test_pattern).tif');
% FFT of image
transFFT_A=fft2(img);
transFFT_B=fftshift(transFFT_A);
imshow(abs(transFFT_B),[-12 300000]);
avg = transFFT_A(1,1) / (W*H);
disp(avg);
```

(Python code)

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('Fig0441(a)(characters_test_pattern).tif', cv2.IMREAD_GRAYSCALE)

f = np.fft.fft2(img.astype(np.float32))
fshift = np.fft.fftshift(f)

magnitude_spectrum = 20*np.log(np.abs(fshift))

plt.subplot(121), plt.imshow(img, cmap='gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122), plt.imshow(magnitude_spectrum, cmap='gray')
plt.title('Magnitude Spectrum'), plt.xticks([]), plt.yticks([])
plt.show()

mean_value = np.mean(img)
print("Average Value:", mean_value)
```

(b)



(c)

$$Average\ Value = 207.314699$$

9. **Edge Detection Combined with Smoothing and Thresholding** (10%)
   (a) Extend the program from HW #2 p.9(a) to compute the Sobel gradient using the masks in Fig. 3.41 (d) & (e). Your program should implement Eq. (3.6-12), and have the option of outputting a binary image by comparing each gradient point against a specified threshold, T.
   (b) Download Fig. 2.35(c) from the course web site. By combining smoothing with a 3 x 3 mask from HW #2 p.9(a) and your program from (a), process Fig. 2.35(c) and produce a binary image that isolates (segments) the large blood vessel in the center of the image. This will require repeated trials of smoothing and choices of T. Looking at the histogram (HW #2 p.8) of the gradient image before it is thresholded will help you select a value for T.

Sol:

(a)

(Matlab code)

```
clear all;clc;
mask1 = zeros(3, 3);
for i = 1:3
    for j = 1:3
        fprintf('mask1_%d%d', i, j);
        mask1(i, j) = input(':');
    end
end
mask2 = zeros(3, 3);
for i = 1:3
    for j = 1:3
        fprintf('mask2_%d%d', i, j);
        mask2(i, j) = input(':');
    end
end
src = imread('Fig0235(c)(kidney_original).tif');
d = conv2(im2double(src),mask1);
e = conv2(im2double(src),mask2);
finish = abs(d + e);
T = graythresh(finish);
R = im2bw(finish, T);
```

```
subplot(1, 2, 1);
imshow(src);
subplot(1,2,2);
imshow(R);
```

(Python code)

```
import cv2
import numpy as np

mask1 = np.zeros((3, 3))
for i in range(3):
    for j in range(3):
        mask1[i, j] = float(input(f'mask1_{i+1}{j+1}: '))

mask2 = np.zeros((3, 3))
for i in range(3):
    for j in range(3):
        mask2[i, j] = float(input(f'mask2_{i+1}{j+1}: '))

src = cv2.imread('Fig0235(c)(kidney_original).tif', cv2.IMREAD_GRAYSCALE)
d = cv2.filter2D(src, -1, mask1)
e = cv2.filter2D(src, -1, mask2)
finish = np.abs(d + e)
T, R = cv2.threshold(finish, 0, 255, cv2.THRESH_BINARY | cv2.THRESH_OTSU)

cv2.imshow('Original Image', src)
cv2.imshow('Result Image', R)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

(b)

10. **Highpass Filtering Combined with Thresholding** (10%)

(a) Implement the Gaussian highpass filter of Eq. (4.9-4). You must be able to specify the size, M x N, of the resulting 2D function. In addition, you must be able to specify the location of the center of the Gaussian function.

(b) Download Fig. 4.57(a) from the course web site and use your program from (a) combined with thresholding to approximate the results in Fig. 4.57 (Note that you will be using a Gaussian instead of a Butterworth filter).

Sol:

(a)

(Matlab code)

```
clc;clear all;
img=imread('Fig0457(a)(thumb_print).tif');
transFFT=fft2(img);
transFFT=fftshift(transFFT);
% Gaussian Filter
[M N]=size(transFFT);
Cutoff = input('Cut off frequency¡G');
X=0:N-1;
Y=0:M-1;
[X Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
Hi=1-exp(-((X-Cx).^2+(Y-Cy).^2)./(2*Cutoff).^2);
filter=transFFT.*Hi;
afterFFT=ifftshift(filter);
answer=ifft2(afterFFT);
answer=uint8(answer);
% thresholding
[x y]=size(answer);
th_answer = zeros(x,y);
for dx=1:x
    for dy=1:y
        if answer(dx,dy)>0
            th_answer(dx,dy)=255;
        else
            th_answer(dx,dy)=0;
        end
    end
end
  figure(1)
  imshow(img);
  figure(2)
  imshow(answer);
  figure(3)
  imshow(th_answer);
```

(Python code)

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread('Fig0457(a)(thumb_print).tif', 0)

transFFT = np.fft.fft2(img)
transFFT = np.fft.fftshift(transFFT)

cutoff = float(input('Cut off frequency: '))
```

```
M, N = img.shape
x = np.linspace(0, N - 1, N)
y = np.linspace(0, M - 1, M)
X, Y = np.meshgrid(x, y)
Cx = 0.5 * N
Cy = 0.5 * M
Hi = 1 - np.exp(-((X - Cx)**2 + (Y - Cy)**2) / (2 * cutoff)**2)

filter = transFFT * Hi
afterFFT = np.fft.ifftshift(filter)
answer = np.fft.ifft2(afterFFT)
answer = np.uint8(answer.real)

th_answer = np.zeros(answer.shape, dtype=np.uint8)
th_answer[answer > 0] = 255

plt.figure(figsize=(12, 4))
plt.subplot(131)
plt.imshow(img, cmap='gray')
plt.title('Original Image')

plt.subplot(132)
plt.imshow(answer, cmap='gray')
plt.title('After FFT')

plt.subplot(133)
plt.imshow(th_answer, cmap='gray')
plt.title('Thresholded')

plt.show()
```

(b)