1) Consider the two-dimensional case.

   a) Write the Inverse Fourier Transform formula in 2D (express $f(x,y)$ function of $F(u,v)$ in the continuous case).

   Answer: The Inverse Fourier Transform formula in two dimensions is:

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{i2\pi(ux+vy)}dudv \quad .$$

   b) Assume f is twice differentiable. Using (a), find the Fourier transform of the mixed partial derivative $\frac{\partial^2 f}{\partial x \partial y}$, function of F.

We first will take the partial derivative $\frac{\partial f}{\partial x}$ of the function inverse Fourier Transform:

$$\frac{\partial f}{\partial x} = i2\pi u e^{i2\pi(ux+vy)}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)dudv \quad . \tag{1}$$

We then take the parital derivative $\frac{\partial}{\partial y}$ of the equation $\frac{\partial f}{\partial x}$ and we get:

$$\frac{\partial^2 f}{\partial x \partial y} = (i2\pi)^2 uv e^{i2\pi(ux+vy)}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)dudv \quad . \tag{2}$$

2) We have seen in continuous variables that $\mathcal{F}(\delta) \equiv 1$, thus δ and 1 form a Fourier pair; we also have that $\mathcal{F}(1) = \delta$. Using the second property and the translation property, show that the Fourier transform of $f(x) = \sin(2\pi u_0 x)$, where u_0 is a real number, is

$$F(u) = (i/2)\Big[\delta(u+u_0) - \delta(u-u_0)\Big] \quad .$$

(Hint: You could express f function of exponentials.)

Answer: We first apply the Fourier transformation to the function given:

$$\mathcal{F}[f(x)] = \int_{-\infty}^{\infty} \sin(2\pi u_0 x)e^{-i2\pi(ux)}dx \quad .$$

We then convert the sine function into exponentials:

$$\sin\theta = \frac{e^{i\theta} - e^{-i\theta}}{2i}$$

and substitute it into our formula, which gives us the following:

$$\int_{-\infty}^{\infty} \frac{-i}{2}\left(e^{i2\pi(u_0 x)} - e^{-i2\pi(u_0 x)}\right)e^{-i2\pi(ux)}dx \tag{3}$$

$$\Rightarrow \frac{-i}{2}\int_{-\infty}^{\infty}(1)e^{-i2\pi(ux)}e^{i2\pi(u_0 x)}dx - \left(\frac{-i}{2}\right)\int_{-\infty}^{\infty}(1)e^{-i2\pi(ux)}e^{-i2\pi(u_0 x)}dx \tag{4}$$

Using the translation property of the Fourier Transform and the delta function, we can then write:

$$\Rightarrow \frac{-i}{2}\mathcal{F}(1)(u-u_0) - \left(\frac{-i}{2}\right)\mathcal{F}(1)(u+u_0) \tag{5}$$

$$\Rightarrow \frac{i}{2}\left[\delta(u+u_0) - \delta(u-u_0)\right] \tag{6}$$

3) We have seen in continuous variables that $\mathcal{F}(\delta) \equiv 1$, thus δ and 1 form a Fourier pair; we also have that $\mathcal{F}(1) = \delta$. Using the second property and the translation property, show that the Fourier transform of the continuous function $f(x, y) = A\sin(2\pi u_0 x + 2\pi v_0 y)$ is

$$F(u, v) = A\frac{i}{2}\Big[\delta(u + u_0, v + v_0) - \delta(u - u_0, v - v_0)\Big] \quad .$$

(Hint: You could express f as a function of exponentials)

Answer: We first apply the Fourier transformation to the function given:

$$\mathcal{F}[f(x)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} A\sin(2\pi u_0 x + 2\pi v_0 y)e^{-i2\pi(ux+vy)}dxdy \quad .$$

We then convert the sine function into its exponentials:

$$\sin\theta = \frac{e^{i\theta} - e^{-i\theta}}{2i}$$

which gives us:

$$\frac{-Ai}{2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\Big[e^{i(2\pi u_0 x + 2\pi v_0 y)} - e^{-i(2\pi u_0 x + 2\pi v_0 y)}\Big]e^{-i2\pi(ux+vy)}dxdy \tag{7}$$

$$\Rightarrow \frac{-Ai}{2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(1)e^{i(2\pi u_0 x + 2\pi v_0 y)}e^{-i2\pi(ux+vy)}dxdy - \frac{-Ai}{2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(1)e^{-i(2\pi u_0 x + 2\pi v_0 y)}e^{-i2\pi(ux+vy)}dxdy \tag{8}$$

and after applying the translation property of the Fourier Transformation, we then get:

$$\Rightarrow \frac{-Ai}{2}\mathcal{F}(1)(u - u_0, v - v_0) - \left(\frac{-Ai}{2}\right)\mathcal{F}(1)(u + u_0, v + v_0) \tag{9}$$

$$\Rightarrow A\frac{i}{2}\Big[\delta(u + u_0, v + v_0) - \delta(u - u_0, v - v_0)\Big] \quad . \tag{10}$$

4) Assume that $\mathcal{F}(1) = \delta$ also holds in the discrete case (this can be shown). Using this property and the translation property, show that the Fourier transform of the discrete function $f(x, y) = \sin(2\pi u_0 x + 2\pi v_0 y)$ is

$$F(u, v) = \frac{i}{2}\Big[\delta(u + Mu_0, v + Nv_0) - \delta(u - Mu_0, v - Nv_0)\Big] \quad .$$

Answer: We first apply the discrete Fourier transform to the given function:

$$\mathcal{F}[f(x, y)] = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\sin(2\pi u_0 x + 2\pi v_0 y)e^{-i2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

We then can express our sine function as exponentials with the following:

$$\sin\theta = \frac{e^{i\theta} - e^{-i\theta}}{2i}$$

which, we then apply to our equation:

$$\Rightarrow \frac{-i}{2}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\Big[e^{i(2\pi u_0 x + 2\pi v_0 y)} - e^{-i(2\pi u_0 x + 2\pi v_0 y)}\Big]e^{-i2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)} \quad . \tag{11}$$

Thus, we then can write:

$$\Rightarrow \frac{-i}{2}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(1)e^{i(2\pi u_0 x+2\pi v_0 y)}e^{-i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}-\left(\frac{-i}{2}\right)\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(1)e^{-i(2\pi u_0 x+2\pi v_0 y)}e^{-i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \quad (12)$$

Using the translation property, we can then write:

$$\Rightarrow \frac{-i}{2}\mathcal{F}(1)(u-Mu_0, v-Nv_0)-\left(\frac{-i}{2}\right)\mathcal{F}(1)(u+Mu_0, v+Nv_0) \quad (13)$$

$$\Rightarrow \frac{i}{2}\Big[\delta(u+Mu_0, v+Nv_0)-\delta(u-Mu_0, v-Nv_0)\Big] \quad . \quad (14)$$

5) **Periodic Noise Reduction Using a Notch Filter**

a) Write a program that implements sinusoidal noise of the form given in the previous homework: $n(x,y) = A\sin(2\pi u_0 x + 2\pi v_0 y)$. The input to the program must be the amplitude, A, and the two frequency components u_0 and v_0.

Answer: The following is the code written to generate the noise of the image:

```
original_image = imread('image.jpg');
original_image_double = im2double(original_image);

[M, N] = size(original_image_double);

A = input('Amplitude:');
frequency1 = input('Frequency 1:');
frequency2 = input('Frequency 2:');

for i = 1:M
    for j = 1:N
        noise(i,j) = A * sin(2 * pi * frequency1 * i + 2 * pi * frequency2 * j);
    end
end
```

b) Download image 5.26(a) of size $M \times N$ and add sinusoidal noise to it, with $v_0 = 0$. The value of A must be high enough for the noise to be quite visible in the image (for example, you can take $A = 100$, $u_0 = 134.4$, $v_0 = 0$).

Answer: The following is the code written to add noise to the image:

```
for k = 1:M
    for l = 1:N
        B(k,l) = original_image_double(k,l) + noise(k,l);
    end
end

figure, imshow(B), title('Degraded Image');
```

c) Compute and display the degraded image and its spectrum (you may need to apply a log transform to visualize the spectrum).

Please see attached pages for image and spectrum.

```
fourier_transform = fft2(shift);
spectrum = abs(fourier_transform);

for i = 1:M
    for j = 1:N
        plot_spectrum(i,j) = 5 * log(1 + spectrum(i,j));
    end
end
```

d) Notch-filter the image using a notch filter of the form shown in Fig. 5.19(c), to remove the periodic noise.

Please see attached pages for image and spectrum.
Note: Some of the functions given in the code were pulled from StackOverflow.

```
fourier_transform2 = fftshift(fft2(original_image));
norm_img = @(img) (img - min(img(:))) / (max(img(:)) - min(img(:)));
gNotch = @(v,mu,cov) 1-exp(-0.5*sum((bsxfun(@minus,v,mu).*(cov\bsxfun(@minus,v,mu)))));
center_x = 129;
center_y = 129;

% distance of noise from center
wx1 = 149.5-129;
wx2 = 165.5-129;
wy  = 157.5-129;

% create notch filter
notch_filt = ones(M,N);

[y,x] = meshgrid(1:N, 1:M);
X = [y(:) x(:)].';
notch_filt = notch_filt .* reshape(gNotch(X,[center_x+wx1;center_y+wy],eye(2)*25),[M,N]);
notch_filt = notch_filt .* reshape(gNotch(X,[center_x+wx2;center_y+wy],eye(2)*25),[M,N]);
notch_filt = notch_filt .* reshape(gNotch(X,[center_x-wx1;center_y-wy],eye(2)*25),[M,N]);
notch_filt = notch_filt .* reshape(gNotch(X,[center_x-wx2;center_y-wy],eye(2)*25),[M,N]);

% apply filter
fourier_transform2 = fourier_transform2 .* notch_filt;

% compute inverse
ifft_ = ifft2(ifftshift(fourier_transform2));
restored_image = histeq(norm_img(ifft_));
```