1. Figures of the Fourier magnitude spectra of the degraded image-pj4 (motion blurring).

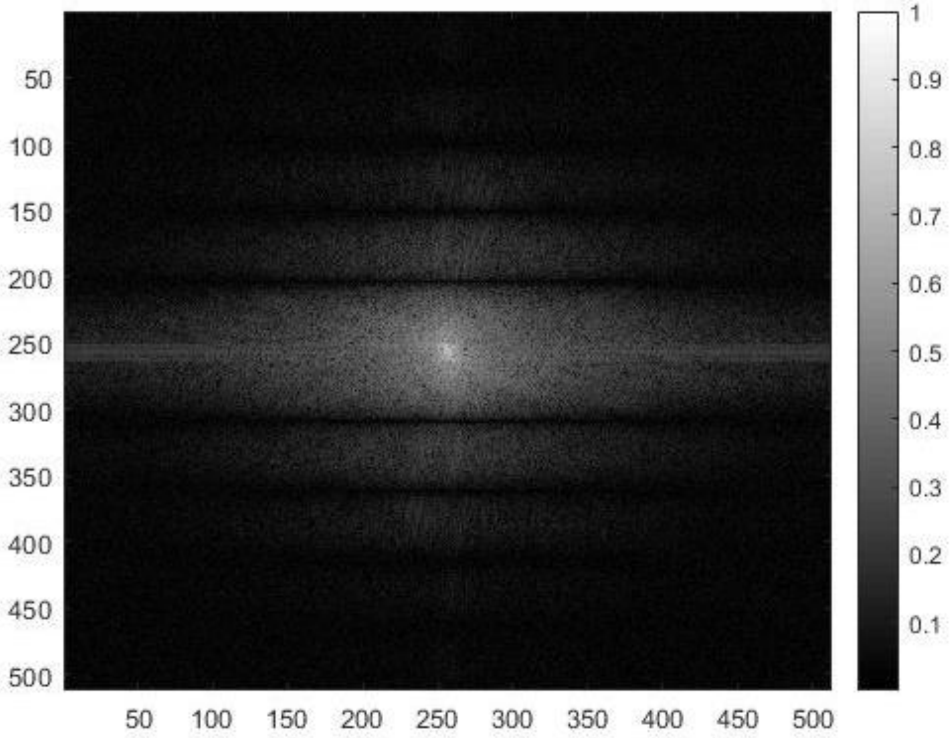


Figure 1 Log Fourier magnitude spectra of the degraded image-pj4 (motion blurring).

2. Figure of the Fourier magnitude of Laplacian filter *H(u, v).*

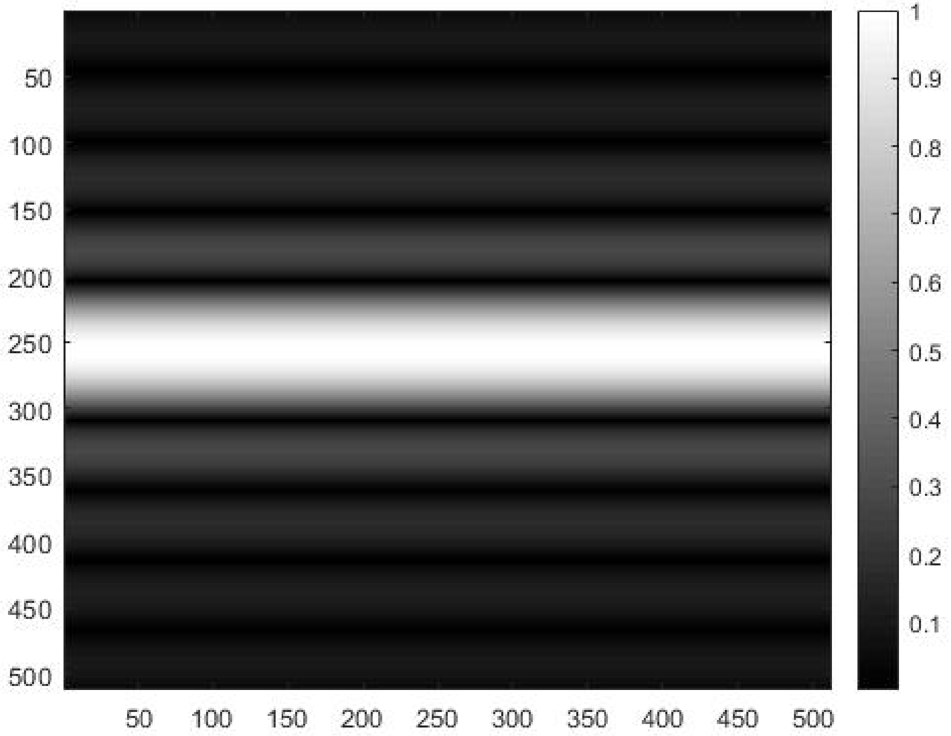
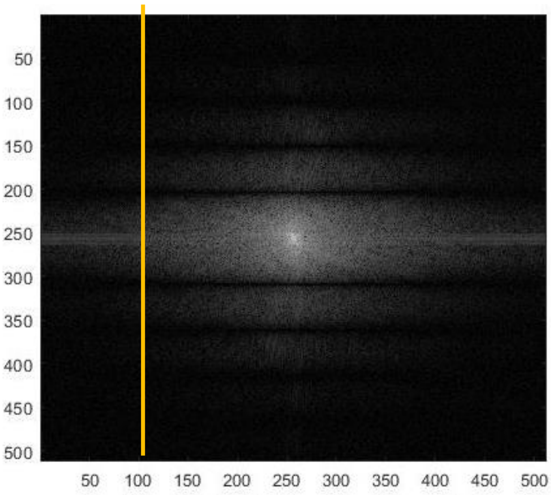
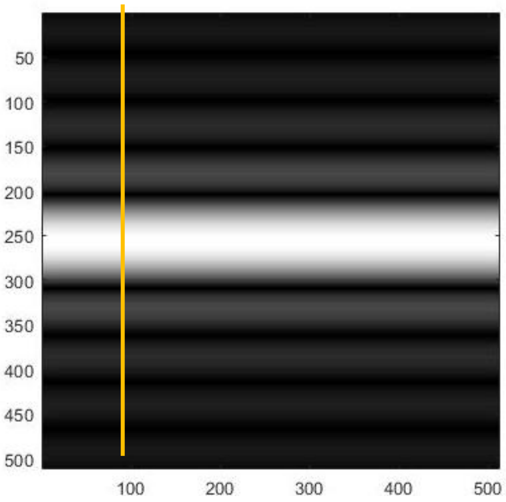


Figure 2 Log Fourier magnitude of degradation model *H(u, v).*

One-dimension signal of G(u, v) and H(u, v)



Log Fourier spectrum of G(u, v)



Log Fourier spectrum of H(u, v)

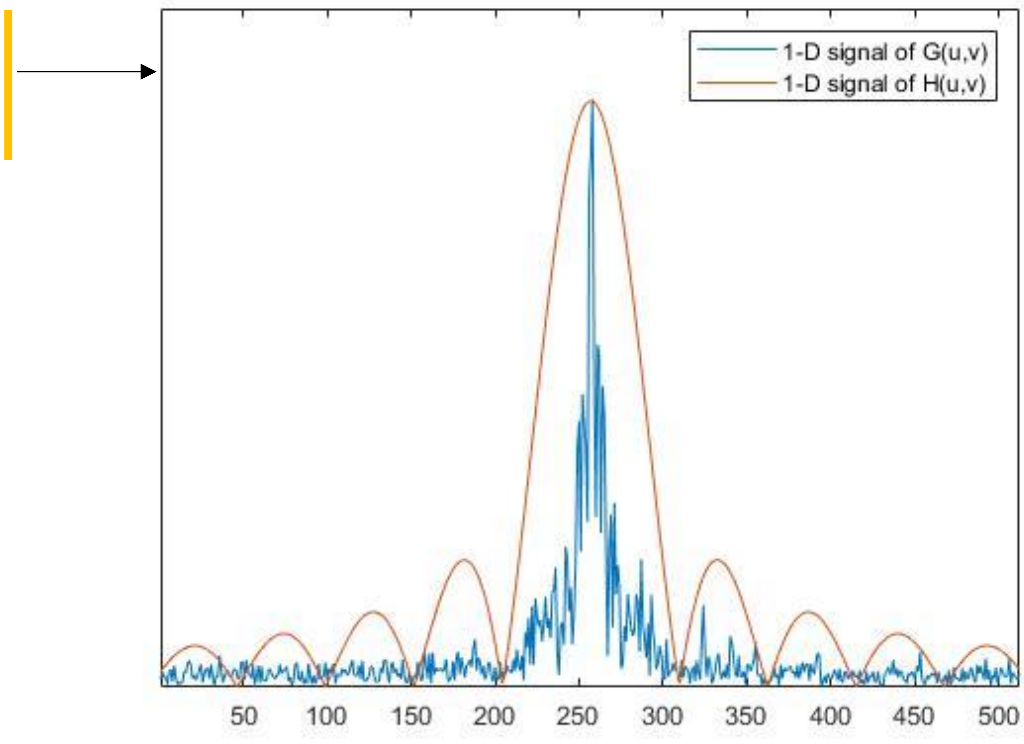


Figure 3 One-dimension of G(u, v) and H(u, v).

Model parameters: θ=90° , T = 1, a = -0.019, b = 0

3. output image (restoration).

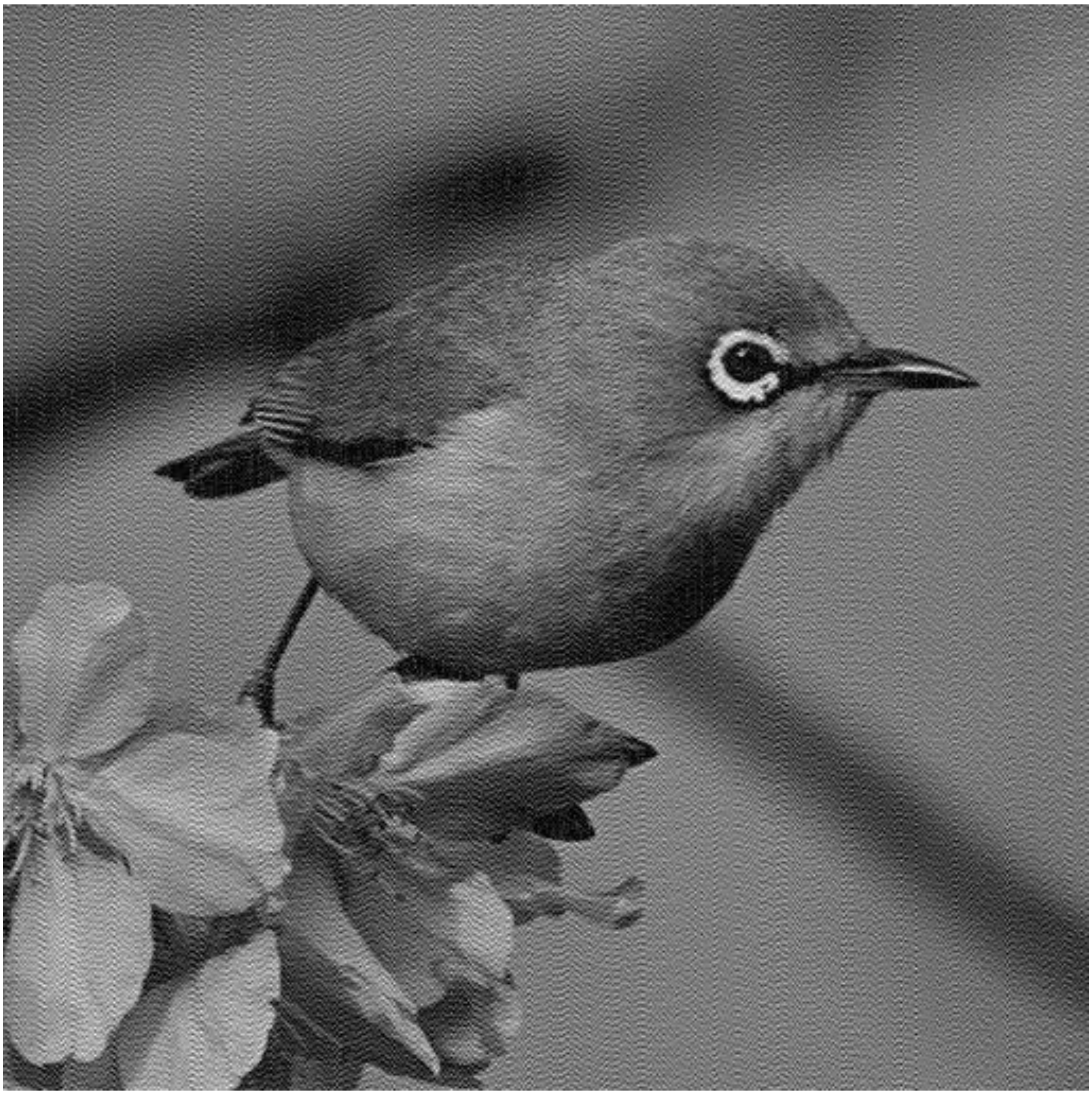


Figure 4 Output image.

Source code:

```
% Clear all command window, temporary variables and close all MATLAB window
close all; clear; clc;

% Read the image, data type: uint8
img_blur = imread('image-pj4 (motion blurring).tif');
% change img_blur type to double and normalize to [0, 1]
imgd = double(img_blur)/255;

% Get Fourier transform of input image
X = fft2(imgd);
% Shift zero-frequency component to center of spectrum
X = fftshift(X);

% Show the degraded image
figure;
imshow(img_blur);
title('original degraded image');

% Show the log Fourier magnitude spectra of the degraded image and normalize
figure;
imagesc(log(abs(X)+1)./log(max(abs(X(:))+1)));
colorbar;
colormap gray;
title('Fourier magnitude of the degraded image');

% Degradation function design (get a,b,T, theta by trail and error)
T = 1; theta = 90; a = -0.019;
b = 0; % because this direction of linear motion of degraded image is 90, just setting b=0
% b = abs(a/tan(theta * pi / 180));

H = zeros(512,512);
for u = 1:512
    for v = 1:512
```

```
        k = pi*((u-257)*a + (v-257)*b);
        if k == 0
            K = 1; % accroding to L'Hospital's rule sin(k)/k = 1, when
the k --> 0
            H(u,v) = T * K * exp(-1j * k);
        else
            H(u,v) = T / k * sin(k) * exp(-1j * k);
        end
    end
end

% Show the log of degradation function and normalize
figure;
imagesc(log(abs(H)+1)./log(max(abs(H(:))+1)));
colorbar;
colormap gray;
title('Fourier magnitude of the filter');

% Use designed degradation function to do inverse filtering
F = X ./ H;

% Show the restoration image
output = uint8(255*mat2gray(abs(ifft2(ifftshift(F)))));
figure;
imshow(output);
title('restored image');
```