1. Figures of the Fourier magnitude spectra of the *bird* image after applying Laplacian filtering.

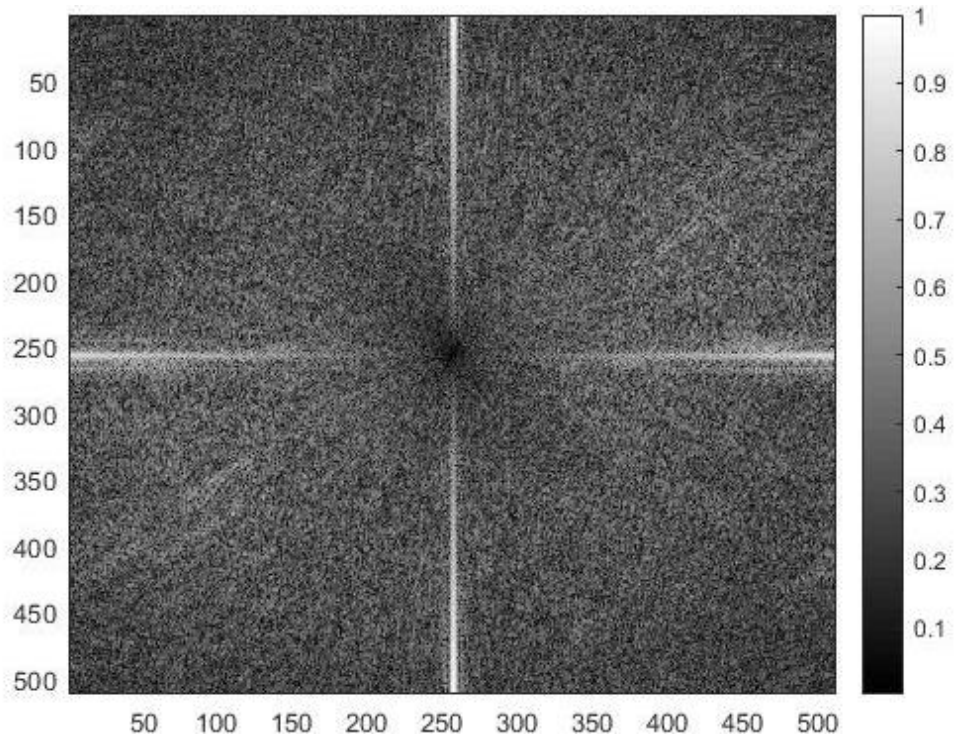

Figure 1 Fourier magnitude spectra of the *bird* image after applying Laplacian filtering (normalize and log scale).

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

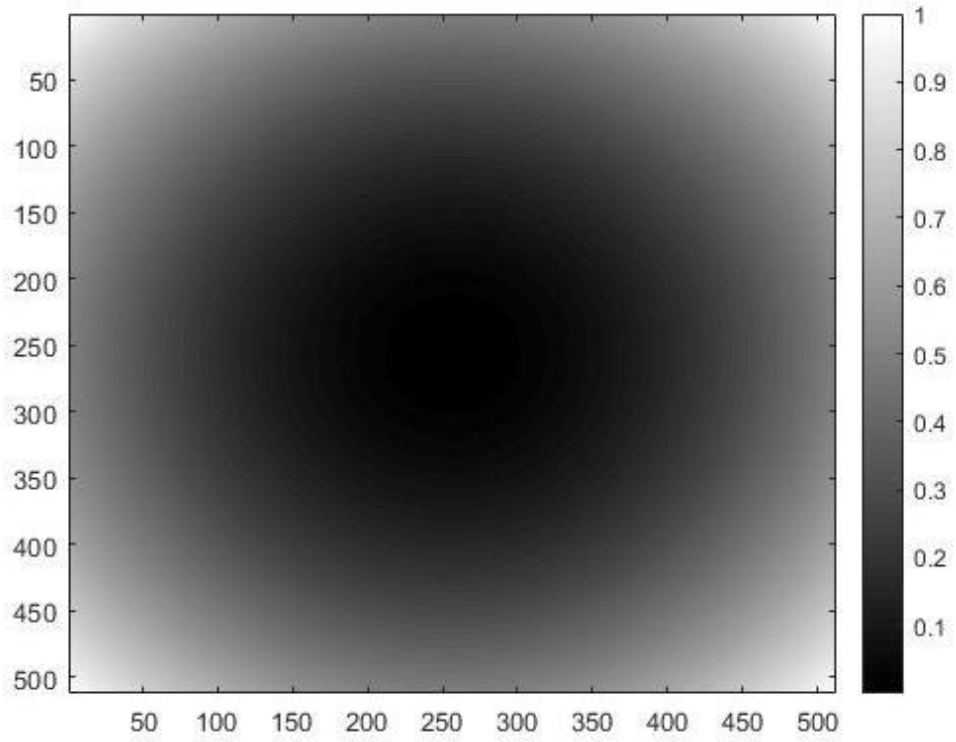

Figure 2 Fourier magnitude of Laplacian filter *H(u,v)*.

3. Figure of output image.

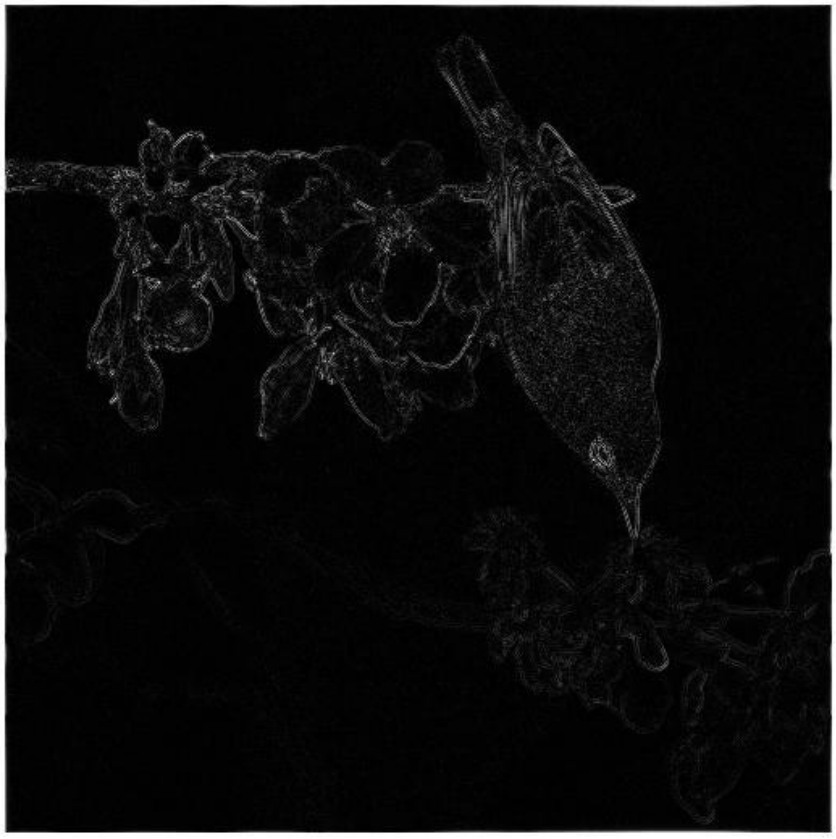

Figure 3 Output image.

4. Table of top 25 DFT frequencies *(u,v)* after Laplacian filtering.

top	(u, v)
1	[506,257]
2	[8,257]
3	[511,257]
4	[3,257]
5	[507,257]
6	[7,257]
7	[503,257]
8	[1,257]
9	[11,257]
10	[502,257]
11	[512,257]
12	[12,257]
13	[2,257]
14	[508,257]
15	[6,257]
16	[510,257]
17	[476,257]
18	[4,257]
19	[38,257]
20	[509,257]
21	[5,257]
22	[498,257]
23	[501,257]
24	[16,257]
25	[13,257]

Source code

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

% Read the image, data type: uint8
imdata = imread('Bird 1.tif');

% Show the input image (Bird 1.tif)
figure;
imshow(imdata);
title('Original Image');

% Get Fourier transform of input image and change the data type to
double
F = fft2(im2double(imdata));

% Shift zero-frequency component to center of spectrum
Fsh = fftshift(F);

% Get the absolute of the spectrum of input image (Fourier magnitude)
S = abs(Fsh);

% Implement Laplacian digital filter
H=[];
K=1/(256^2+256^2);
for u=1:256
    for v=1:256
        H(u,v)=K*(u^2+v^2);
    end
end
H_1=flipud(H);
H_2=fliplr(H);
H_3=flipud(H_2);
H_top = cat(2,H_3,H_1);
```

```
H_low = cat(2,H_2,H);
H_all = cat(1,H_top,H_low);

% Show Fourier magnitude of Laplacian filter H(u,v)
figure;
imagesc(H_all);
colorbar; % show colorbar
colormap gray; % Let the image present gray-level

% The spectra of output image is that the spectra of input image
multiply Laplacian filter H(u,v)
output_f = Fsh.*H_all;
% Get the output image using 2-D inverse fast Fourier transform
output=ifft2(output_f);
out=abs(output);
% Adjust the scale range to 0 - 255
out = out - min(out(:));
out = out ./ max(out(:)) .* 255;
% Show the output image
figure;
imshow(uint8(out));

% Normalize the scale, its range [0 - 1]
c2 = 1 / log(max(abs(output_f(:)))+1);
% Show the Fourier magnitude spectra of output image using log scale
figure;
imagesc(c2 .* log(abs(output_f)+1));
colorbar;
colormap gray;


% Find coordinates (u,v) of the top 25 frequency components from
output image
output_f_abs=abs(output_f); % Get the Fourier magnitude spectra of
output image
output_top25_freq=[]; % Initial variable to store results
% Sorting the frequency components from Fourier magnitude. It returns
value and index to M and I with descending
```

```
[M,I] = sort(output_f_abs(:), 'descend');
% Get the coordinates of top 25 frequency components
for kk=1:25
    if mod(I(kk),512) == 0
        row = 512;
    else
        row = mod(I(kk),512);
    end
    col = ceil(I(kk) / 512);
    output_top25_freq{kk,1} = [row, col];
end
```