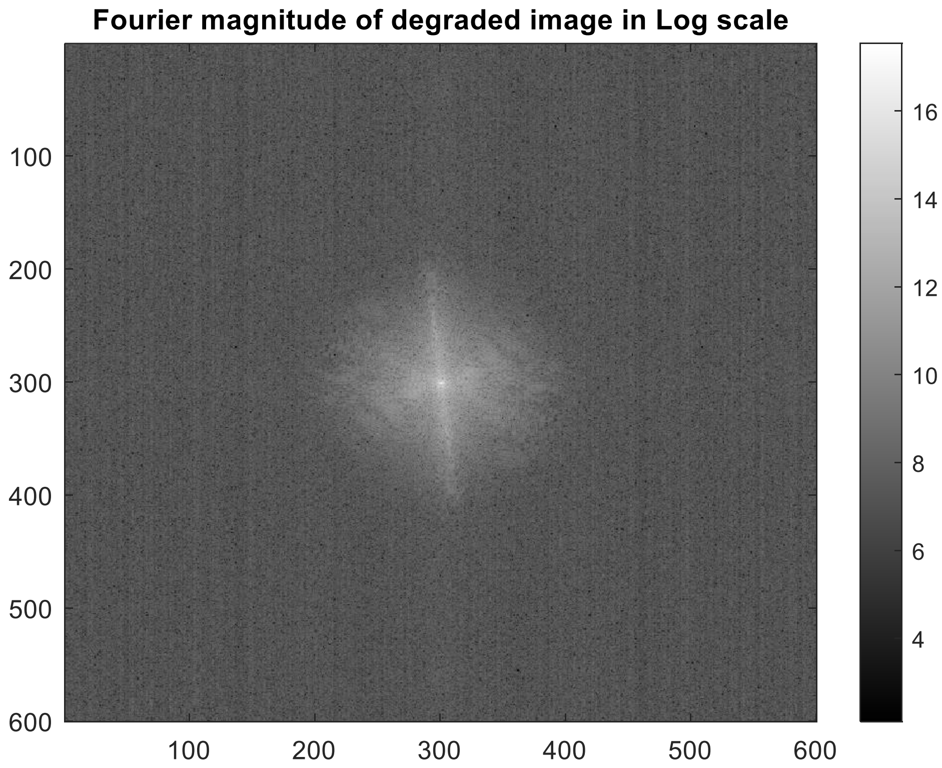
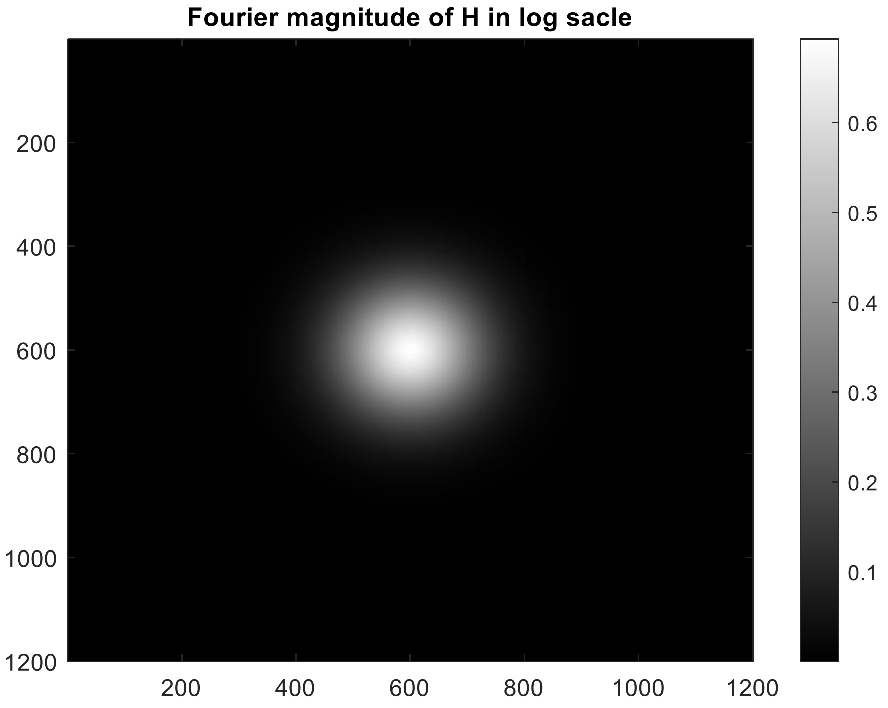
Project 3 solution

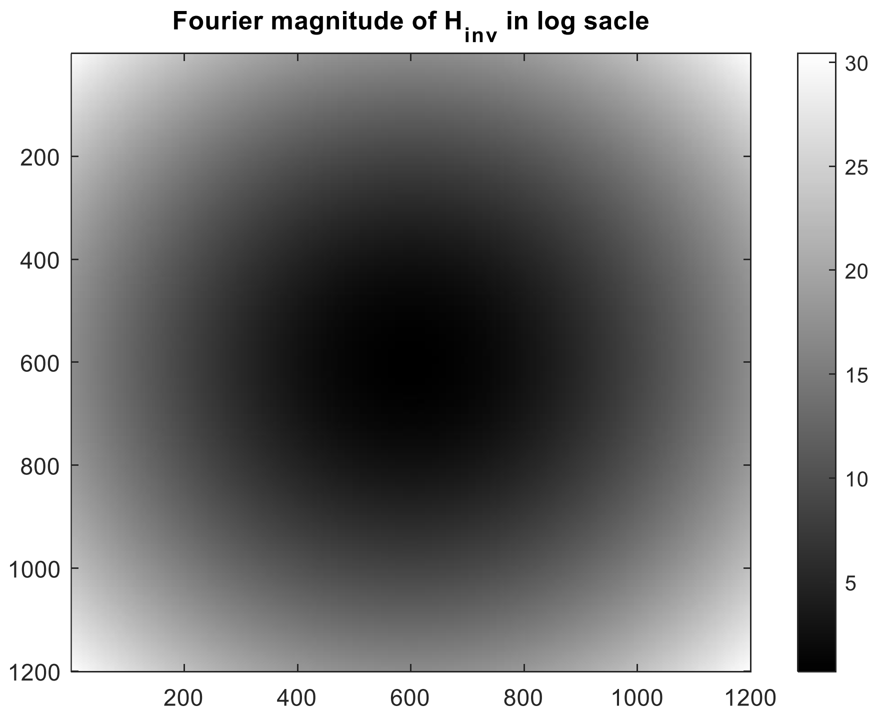
1. Figure of the Fourier magnitude spectrum of the degraded image, “Bird 2 degraded” (15%)



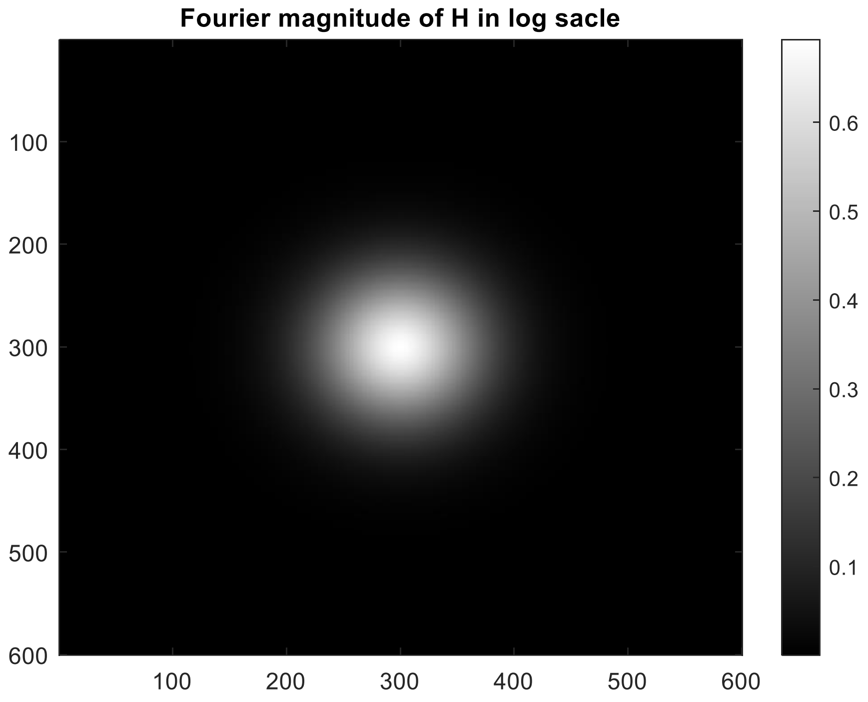
2. Figure of the Fourier magnitude (frequency response) of degradation model $H(u,v)$ (15%)

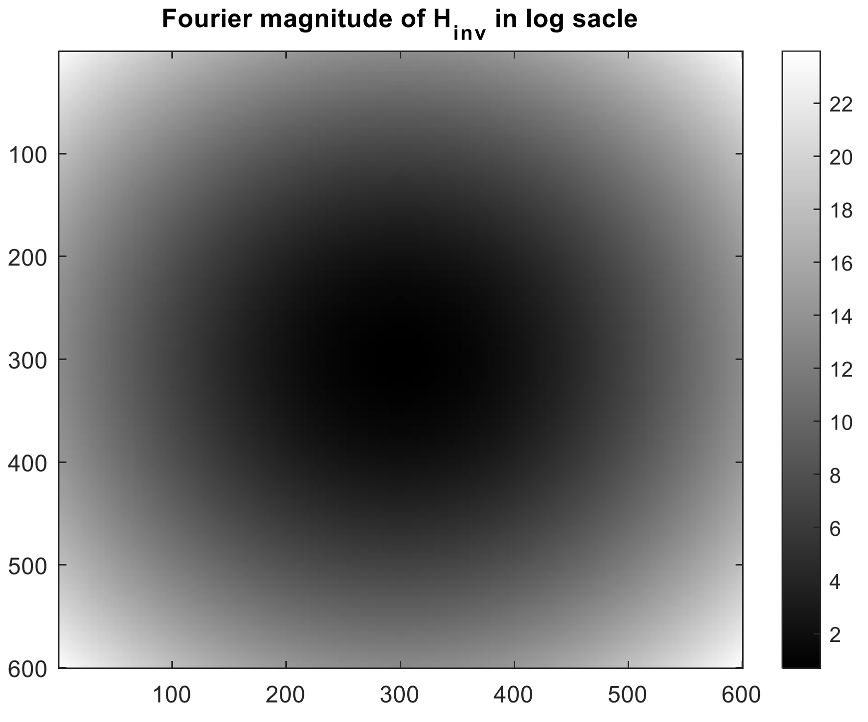
(1). Using zero-padding (k=0.0004):





(2). Without zero-padding (*k*=0.001):





3. Figures of the output images using different radii (50, 85, 120) of inverse filtering (30%) (BLPF: Butterworth low-pass filter, ILPF: Ideal low-pass filter)

(1). Using zero-padding (*k*=0.0004):

Figure 1. Using radii(50) of BLPF

Figure 2. Using radii(50) of ILPF

Figure 3. Using radii(85) of BLPF

Figure 4. Using radii(85) of ILPF

Figure 5. Using radii(120) of BLPF

Figure 6. Using radii(120) of ILPF

(2). Without zero-padding (k=0.001):

Figure 7. Using radii(50) of BLPF

Figure 8. Using radii(50) of ILPF

Figure 9. Using radii(85) of BLPF

Figure 10. Using radii(85) of ILPF

Figure 11. Using radii(120) of BLPF

Figure 12. Using radii(120) of ILPF

4. Model parameter *k* (10%)

Ans: *k*=0.001

(Using zero-padding: you will get 10 points if the k value is 0.0006~0.0002

Without zero-padding: you will get 10 points if the k value is 0.001~0.0006)

5. Source code:

```
% Load the degraded image, data type: uint8
img = imread('Bird 2 degraded.tif');
figure
imshow(img);
title('degraded image')

% Get the image size, and change data type to "double"
[M,N] = size(img);
img = double(img);

% Fourier transform for degraded image, and apply Centering
G = fft2(img);
G = fftshift(G);

% Show the Fourier magnitude spectra in Log scale
figure
imagesc(log(abs(G)+1));
colorbar
colormap gray
title('Fourier magnitude of degraded image in Log scale')


%% Use Zero-padding

% Zero-padding, the result image size: 1200*1200
pad = zeros(M,N);
img_pad = [img pad; pad pad];
```

```matlab
% Fourier transform for zero-padding image, and apply Centering
G_pad = fft2(img_pad);
G_pad = fftshift(G_pad);

% Design inverse filter, H_inv
H_inv = zeros(2*M,2*N);
k = 0.0004; %k value should be 0.0006~0.0002 if you use zero-padding
radius = [50 85 120];
radius_pad = 2*radius; % Double the radius because of zero-padding
cu = 0.5*(2*M); % Center of zero-padding image
cv = 0.5*(2*N);

for rr = 1:length(radius_pad)

for u = 1:2*M
    for v = 1:2*N
        H_inv(u,v) = exp(k*((u - cu)^2 + (v - cv)^2)^(5/6));
    end
end

% Show Fourier magnitude of inverse filter in Log scale
figure
imagesc(log(abs(H_inv)+1));
colorbar
colormap gray
title('Fourier magnitude of H_i_n_v in log sacle')

% Ideal low-pass filter
H_ILPF = zeros(2*M,2*N);
for u = 1:2*M
    for v = 1:2*N
        if (sqrt((u - cu)^2 + (v - cv)^2) <= radius_pad(rr))
            H_ILPF(u,v) = 1;
        end
    end
end
```

```
% Butterworth low-pass filter
H_BLPF = zeros(2*M,2*N);
n = 10;   % order of BLPF
beta = 1;
for u = 1:2*M
    for v = 1:2*N
        D(u,v) = sqrt((u - cu)^2 + (v - cv)^2);
        H_BLPF(u,v) = 1 / (1 + beta*(D(u,v)/radius_pad(rr))^(2*n));
    end
end

% Cascade the inverse filter with the ideal LPF and Butterworth LPF
H_inv_ILPF = H_inv .* H_ILPF;
H_inv_BLPF = H_inv .* H_BLPF;

% Restore the image
F_ILPF = G_pad .* H_inv_ILPF;
F_BLPF = G_pad .* H_inv_BLPF;

% Get the output image back to spatial domain, and change data type
to "uint8"
output_ILPF_abs = uint8(255*mat2gray(abs(ifft2(ifftshift(F_ILPF)))));
output_ILPF_abs = output_ILPF_abs(1:M,1:N);
figure
imshow(output_ILPF_abs)
text = ['Output image using radii(' num2str(radius(rr)) ') of inverse
filtering (ILPF)'];
title(text)

output_BLPF_abs = uint8(255*mat2gray(abs(ifft2(ifftshift(F_BLPF)))));
output_BLPF_abs = output_BLPF_abs(1:M,1:N);
figure
imshow(output_BLPF_abs)
text = ['Output image using radii(' num2str(radius(rr)) ') of inverse
filtering (BLPF)'];
title(text)
end
```

```
%% Without Zero-padding
clear H_inv_ILPF H_inv_BLPF F_ILPF F_BLPF output_ILPF_abs
output_BLPF_abs

% Design inverse filter, H_inv
H_inv = zeros(M,N);
k = 0.001; %k value should be 0.001~0.0006 if you don't use zero-
padding
cu = 0.5 * M;   % Center of input image
cv = 0.5 * N;

for rr = 1:length(radius)

for u = 1:M
    for v = 1:N
        H_inv(u,v) = exp(k*((u - cu)^2 + (v - cv)^2)^(5/6));
    end
end

% Show Fourier magnitude of inverse filter in Log scale
figure
imagesc(log(abs(H_inv)+1));
colorbar
colormap gray
title('Fourier magnitude of H_i_n_v in log sacle')

% Ideal low-pass filter
H_ILPF = zeros(M,N);
for u = 1:M
    for v = 1:N
        if (sqrt((u - cu)^2 + (v - cv)^2) <= radius(rr))
            H_ILPF(u,v) = 1;
        end
    end
end

% Butterworth low-pass filter
```

```
H_BLPF = zeros(M,N);
n = 10;   % order of BLPF
beta = 1;
for u = 1:M
    for v = 1:N
        D(u,v) = sqrt((u - cu)^2 + (v - cv)^2);
        H_BLPF(u,v) = 1 / (1 + beta*(D(u,v)/radius(rr))^(2*n));
    end
end

% Cascade the inverse filter with the ideal LPF and Butterworth LPF
H_inv_ILPF = H_inv .* H_ILPF;
H_inv_BLPF = H_inv .* H_BLPF;

% Restore the image
F_ILPF = G .* H_inv_ILPF;
F_BLPF = G .* H_inv_BLPF;

% Get the output image back to spatial domain, and change data type
to "uint8"
output_ILPF_abs = uint8(255*mat2gray(abs(ifft2(ifftshift(F_ILPF)))));
output_ILPF_abs = output_ILPF_abs(1:M,1:N);
figure
imshow(output_ILPF_abs)
text = ['Output image using radii(' num2str(radius(rr)) ') of inverse
filtering (ILPF)'];
title(text)

output_BLPF_abs = uint8(255*mat2gray(abs(ifft2(ifftshift(F_BLPF)))));
output_BLPF_abs = output_BLPF_abs(1:M,1:N);
figure
imshow(output_BLPF_abs)
text = ['Output image using radii(' num2str(radius(rr)) ') of inverse
filtering (BLPF)'];
title(text)
end
```