Department of Computer Science and Engineering

Lab-report:05

Course Name: Digital Image Processing

Course Code: CSE438

Section No: 03

Submitted To:

Prof. Dr. Ahmed Wasif Reza

Department of Computer Science and Engineering

East West University

Submitted by:

Student's ID: 2020-3-60-012

Student's Name: Sadia Islam Prova

Date of submission: 23-4-24

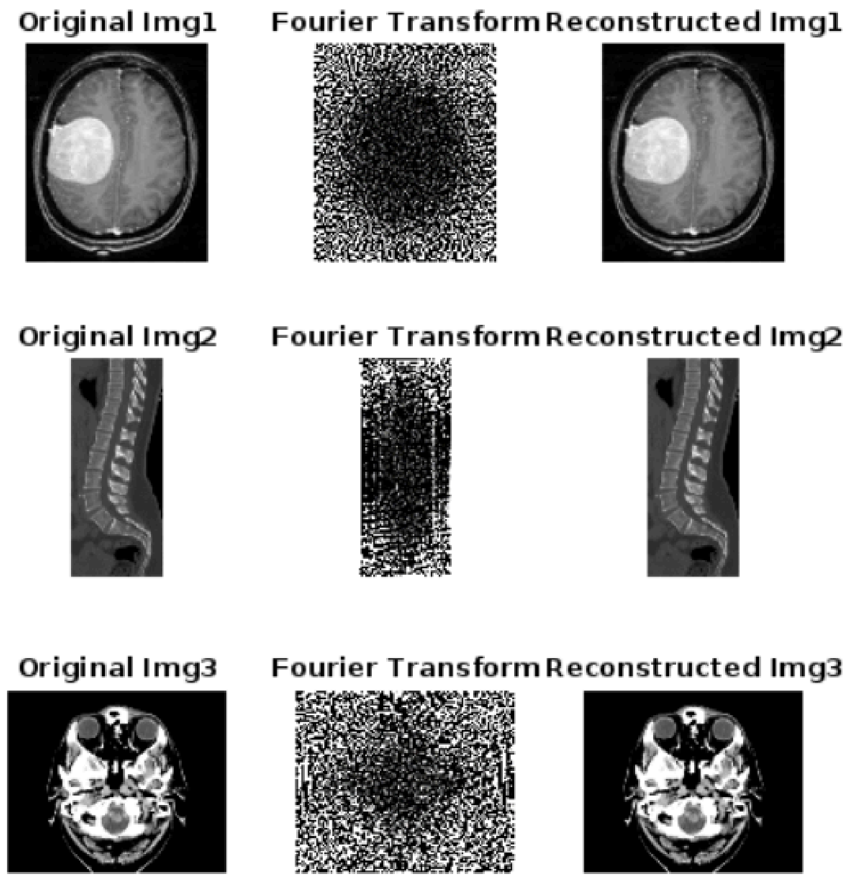
Problem 1: Apply Fourier transform to transform any image (above) from the spatial domain to the frequency domain. Apply inverse Fourier transform to transform the image from the frequency domain to the spatial domain.

Code:

```
gray_img1 = imread('gray_image1.png');
gray_img2 = imread('gray_image2.png');
gray_img3 = imread('gray_image3.png');
gray_img1 = im2double(gray_img1);
gray_img2 = im2double(gray_img2);
gray_img3 = im2double(gray_img3);
fourier_img1 = fft2(gray_img1);
fourier_img2 = fft2(gray_img2);
fourier_img3 = fft2(gray_img3);
reconstructed_img1 = ifft2(fourier_img1);
reconstructed_img2 = ifft2(fourier_img2);
reconstructed_img3 = ifft2(fourier_img3);
figure;
subplot(3, 4, 1);
imshow(gray_img1);
title('Original Img1');
subplot(3, 4, 2);
imshow(real(fourier_img1), []);
title('Fourier Transform');
subplot(3, 4, 3);
imshow(reconstructed_img1);
title('Reconstructed Img1');
subplot(3, 4, 5);
imshow(gray_img2);
title('Original Img2');
subplot(3, 4, 6);
```

```
imshow(real(fourier_img2), []);
title('Fourier Transform');
subplot(3, 4, 7);
imshow(reconstructed_img2);
title('Reconstructed Img2');
subplot(3, 4, 9);
imshow(gray_img3);
title('Original Img3');
subplot(3, 4, 10);
imshow(real(fourier_img3), []);
title('Fourier Transform');
subplot(3, 4, 11);
imshow(reconstructed_img3);
title('Reconstructed Img3');
```

Output:



Problem 2: Apply three types of high pass filtering in the frequency domain in Figure 1 and find out which one is better to produce the enhanced image (sharpen) for the given image (output must show all steps as shown in Figure 2).

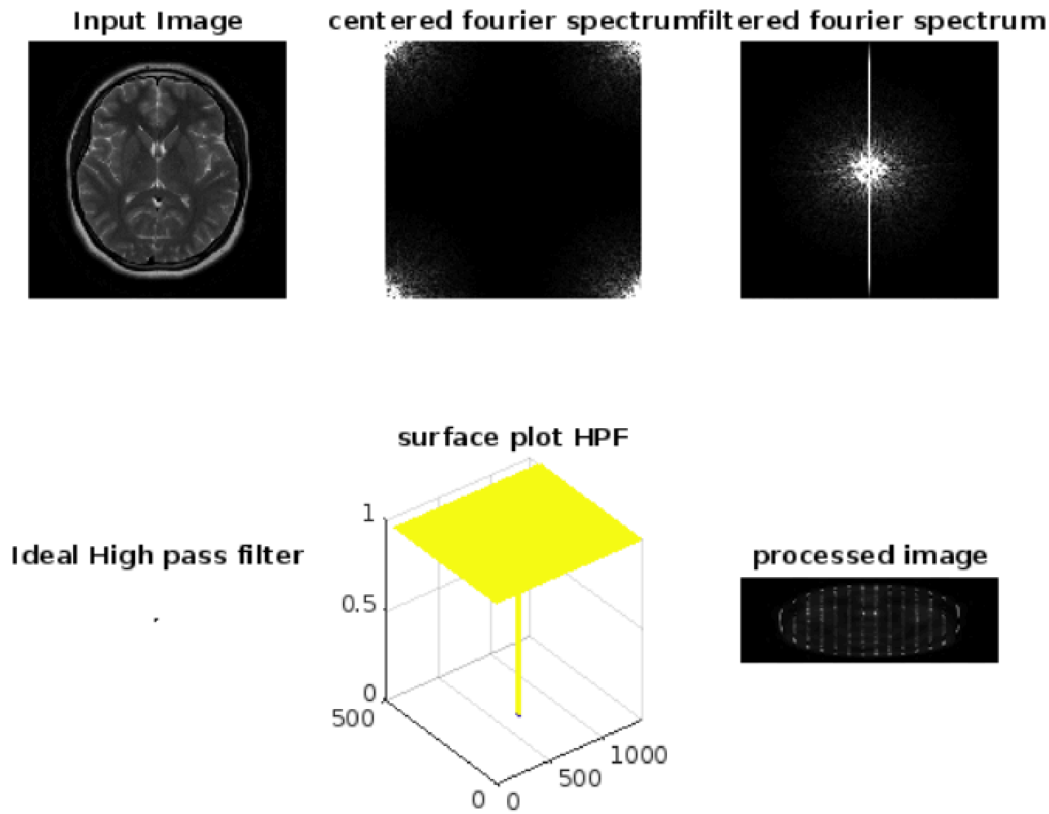
i. Ideal high pass filter (IHPF)

Code:

```
clc;

clear;

close all;


a = imread('gray img.png');

a = im2double(a);

subplot(2,3,1);

imshow(a);
```

```
title('Input Image');
[m,n] = size(a);  D0 = 10; A = fft2(a);  subplot(2,3,2);
imshow(uint8(abs(A)));
title('centered fourier spectrum');
A_shift = fftshift(A);  A_real = abs(A_shift);  subplot(2,3,3);
imshow(uint8(A_real));
title('filtered fourier spectrum');
A_high = zeros(m,n);
D = zeros(m,n);
for u=1:m
   for v=1:n
       D(u,v) = sqrt((u-(m/2))^2 + (v-(n/2))^2);
       if D(u,v) <= D0            A_high(u,v) = 0;
           filt(u,v) = 0;
       else
           A_high(u,v) = A_shift(u,v);
           filt(u,v) = 1;
       end
   end
end
subplot(2,3,4);
imshow(filt)
title('Ideal High pass filter')
subplot(2,3,5);
mesh(filt)
title('surface plot HPF')
B = fftshift(A_high);  B_inverse = ifft2(B); B_real = abs(B_inverse);%Taking
subplot(2,3,6);
imshow(B_real);
title('processed image');
```

Output:



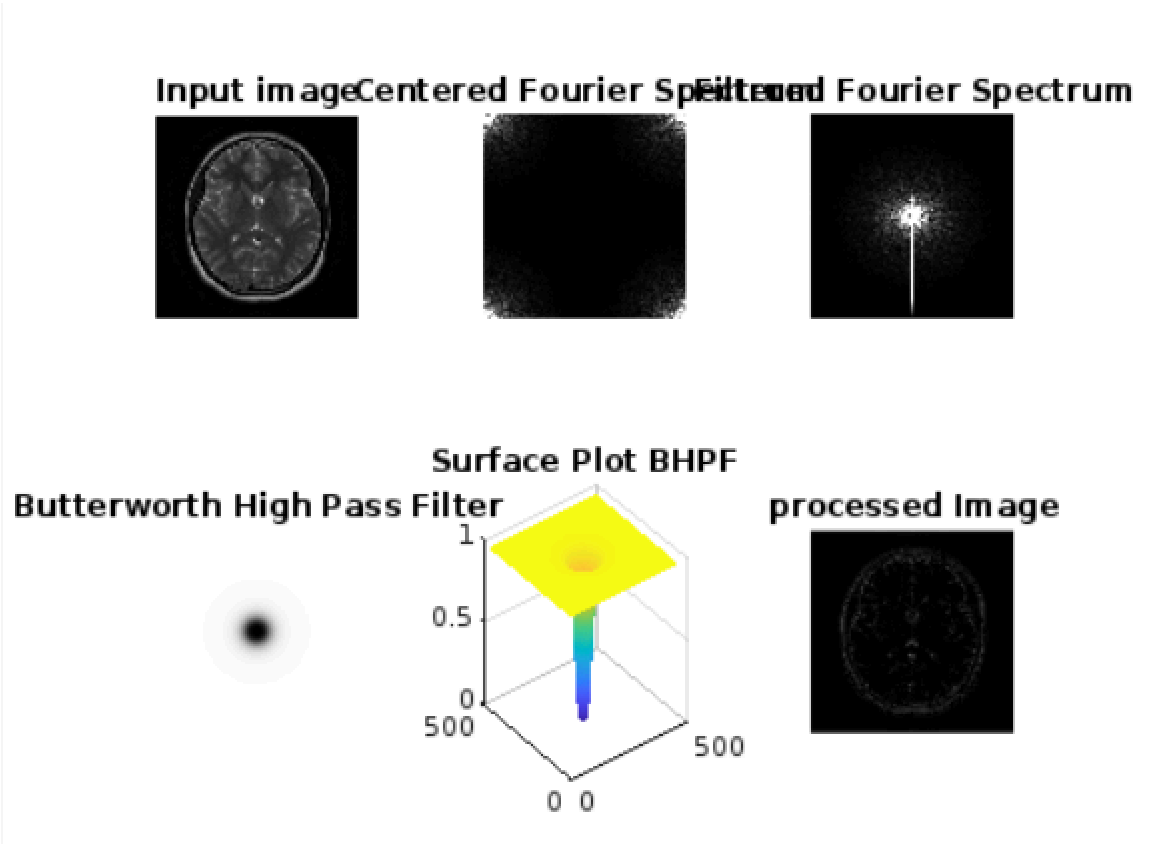
ii. Butterworth high pass filter (BHPF)

Code:

```
clc;

clear;

close all;

% Read the image

a = imread('gray img.png');

a = rgb2gray(a);

a = im2double(a);

[m,n] = size(a);

subplot(2,3,1)

imshow(a)
```

```
title('Input image');
A = fft2(a);
subplot(2,3,2);
imshow(uint8(abs(A)));
title('Centered Fourier Spectrum');
A_shift = fftshift(A);
A_real = abs(A_shift);
subplot(2,3,3)
imshow(uint8(A_real));
title('Filtered Fourier Spectrum');
D0 = 30;
order = 2;
for u = 1:m
   for v = 1:n
       D = sqrt((u - m/2)^2 + (v - n/2)^2);
       H(u,v) = 1 / (1 + (D0 / D)^(2*order));
   end
end
H_high = H .* A_shift;
H_high_real = H .* A_real;
H_high_shift = fftshift(H_high);
H_high_image = ifft2(H_high_shift);
subplot(2,3,4)
imshow(H)
title('Butterworth High Pass Filter');
subplot(2,3,5);
mesh(H)
title('Surface Plot BHPF')
subplot(2,3,6)
imshow(abs(H_high_image));
title('Butterworth High Pass Filtered Image');
```

Output:



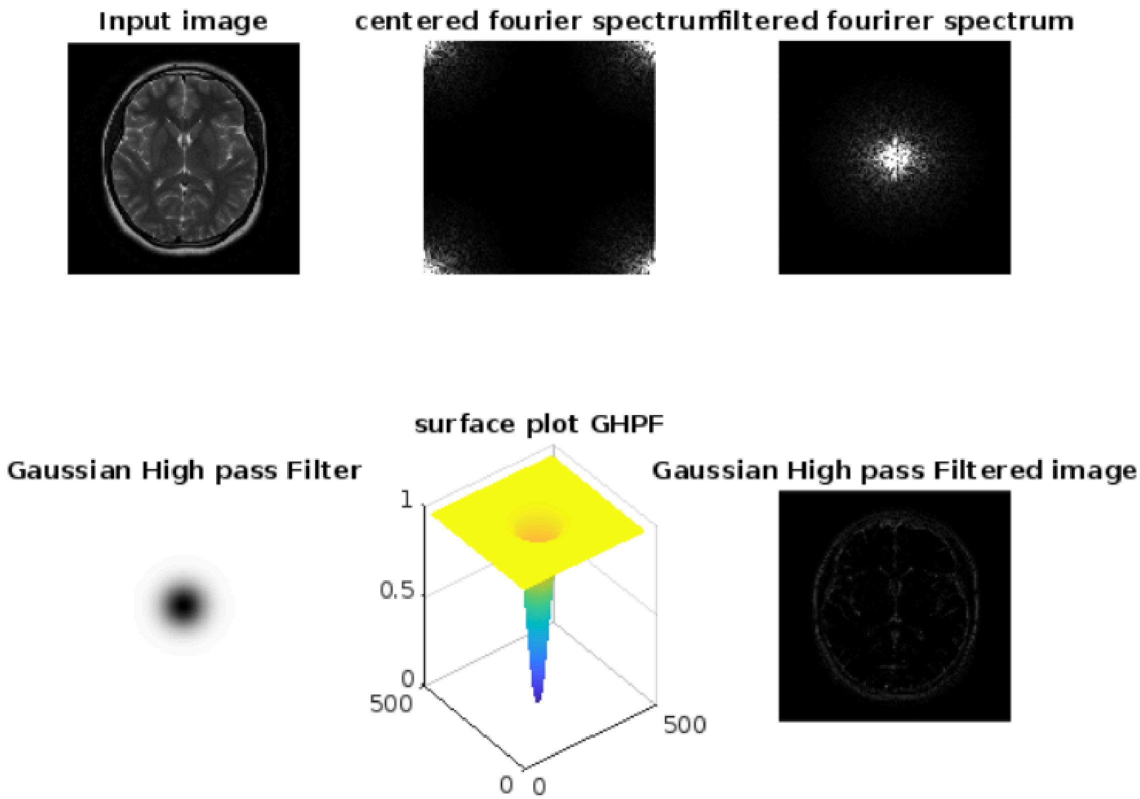
iii. Gaussian high pass filter (GHPF)

Code:

```
clc;

clear;

close all;

a = imread('gray img.png');

a = rgb2gray(a);

a = im2double(a);

[m,n] =size(a);

subplot(2,3,1)

imshow(a)

title('Input image');

A = fft2(a);

subplot(2,3,2);
```

```
imshow(uint8(abs(A)));
title('centered fourier spectrum');
A_shift = fftshift(A);
A_real = abs(A_shift);
subplot(2,3,3)
imshow(uint8(A_real));
title('filtered fourirer spectrum');
D0 = 30;
for u=1:m
   for v=1:n
       D = sqrt((u-m/2).^2+(v-n/2).^2);
       H(u,v) = 1 - exp(-(D^2)/(2*D0.^2));
   end
end
H_high = H.*A_shift;
H_high_real = H.*A_real;
H_high_shift = fftshift(H_high);
H_high_image = ifft2(H_high_shift);
subplot(2,3,4)
imshow(H)
title('Gaussian High pass Filter');
subplot(2,3,5);
mesh(H)
title('surface plot GHPF')
subplot(2,3,6)
imshow(abs(H_high_image));
title('Gaussian High pass Filtered image');
```

Output:



For producing enhanced images (sharpening), the Butterworth High Pass Filter (BHPF) is better than IHPF and BHPF.

Problem 3: Apply three types of low pass filtering in the frequency domain in Figure 1 and find out which one is better to produce the smoothen image for the given image (output must show all steps as shown in Figure 2).

i. Ideal low pass filter (ILPF)

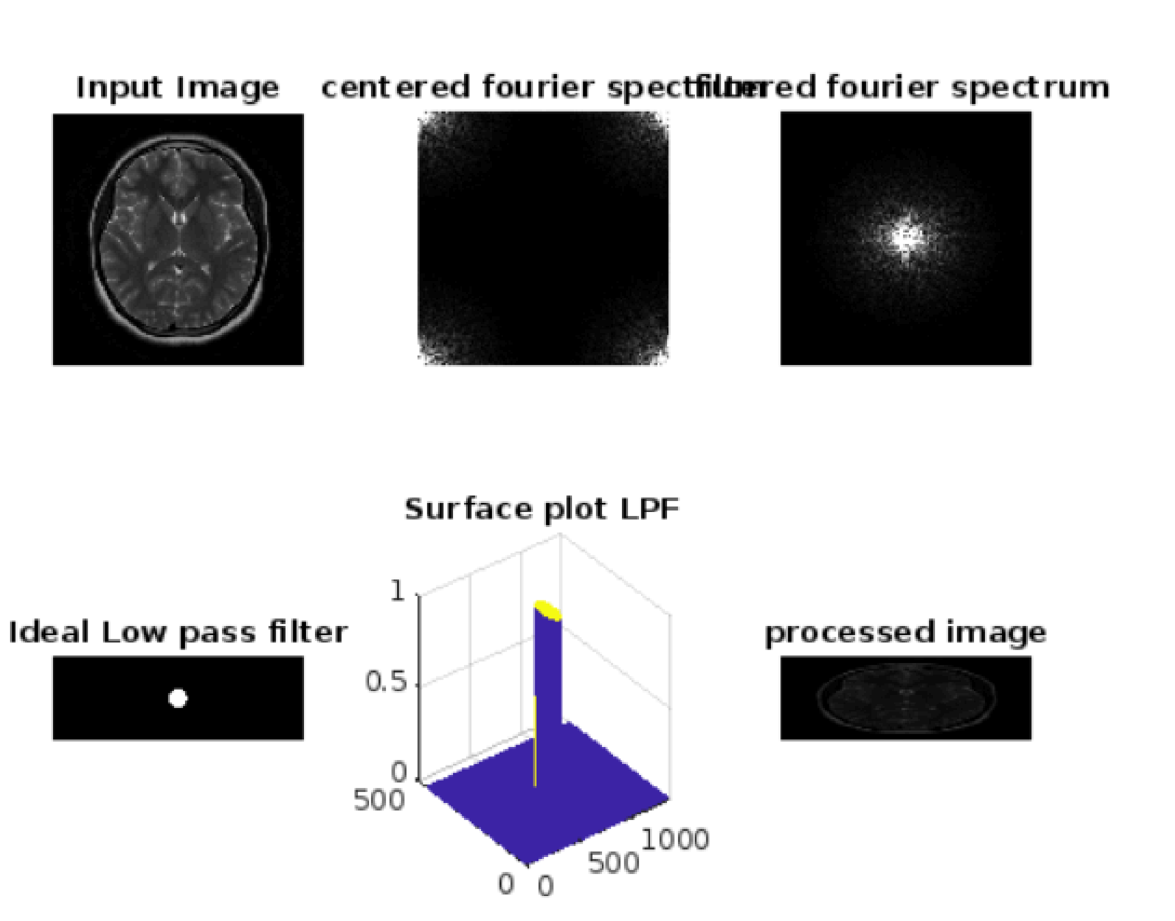
Code:

```
clc;

clear;

close all;

a = imread('gray img.png');

a = im2double(a);
```

```
subplot(2,3,1);
imshow(a);
title('Input Image');
[m,n] = size(a);
D0 = 50;
A = fft2(a);
subplot(2,3,2);
imshow(uint8(abs(A)));
title('centered fourier spectrum');
A_shift = fftshift(A);
A_real = abs(A_shift);
subplot(2,3,3);
imshow(uint8(A_real));
title('filtered fourier spectrum ');
A_low = zeros(m,n);
d = zeros(m,n);
for u = 1:m
   for v = 1:n
       d(u,v) = sqrt((u-(m/2))^2 + (v-(n/2))^2);
       if d(u,v) <= D0
           A_low(u,v) = A_shift(u,v);
           filt(u,v) = 1;
       else
           A_low(u,v) = 0;
           filt(u,v) = 0;
       end
   end
end
subplot(2,3,4);
imshow(filt)
title('Ideal Low pass filter')
subplot(2,3,5);
```

```
mesh(filt)
title('Surface plot LPF')
B = fftshift(A_low);
B_inverse = ifft2(B);
B_real = abs(B_inverse);
subplot(2,3,6);
imshow(B_real);
title('processed image');
```

Output:



ii. Butterworth low pass filter (BLPF)

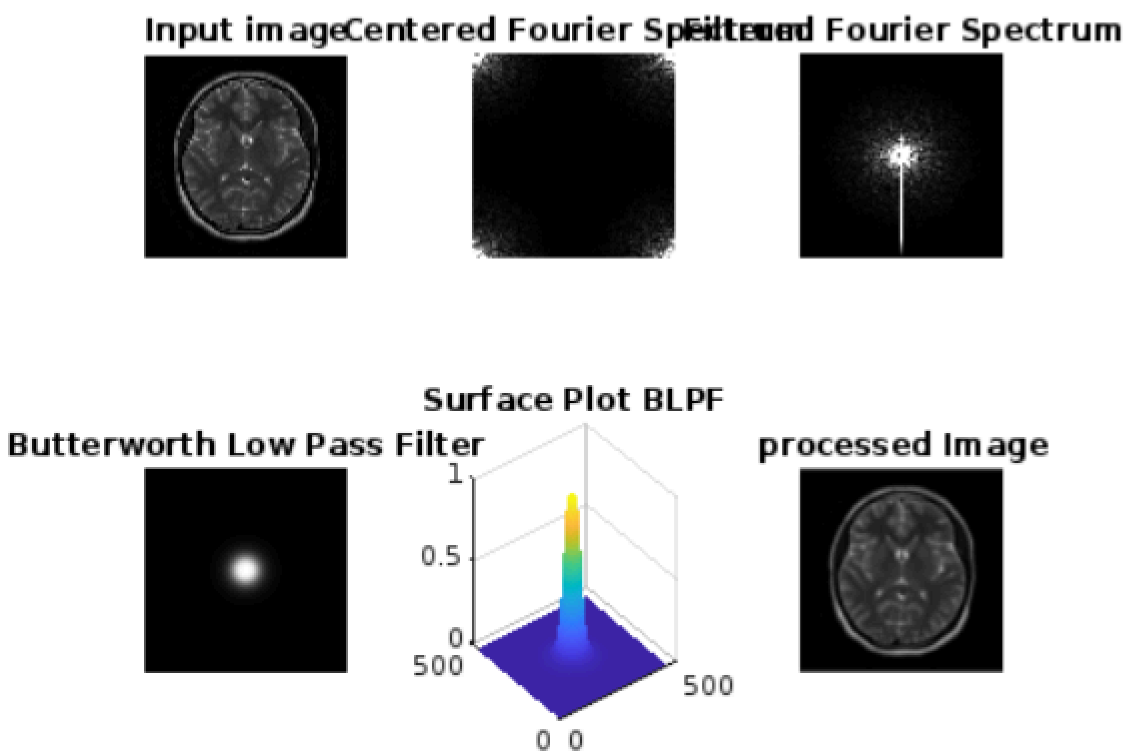
Code:

```
clc;
clear;
```

```
close all;
a = imread('gray img.png');
a = rgb2gray(a);
a = im2double(a);
[m,n] = size(a);
subplot(2,3,1)
imshow(a)
title('Input image');
A = fft2(a);
subplot(2,3,2);
imshow(uint8(abs(A)));
title('Centered Fourier Spectrum');
A_shift = fftshift(A);
A_real = abs(A_shift);
subplot(2,3,3)
imshow(uint8(A_real));
title('Filtered Fourier Spectrum');
D0 = 30;
order = 2;
for u = 1:m
   for v = 1:n
       D = sqrt((u - m/2)^2 + (v - n/2)^2);
       H(u,v) = 1 / (1 + (D / D0)^(2*order));
 end
end
H_low = H .* A_shift;
H_low_real = H .* A_real;
H_low_shift = fftshift(H_low);
H_low_image = ifft2(H_low_shift);
subplot(2,3,4)
imshow(H)
title('Butterworth Low Pass Filter');
```

```
subplot(2,3,5);
mesh(H)
title('Surface Plot BLPF')
subplot(2,3,6)
imshow(abs(H_low_image));
title('processed Image');
```

Output:



iii. Gaussian low pass filter (GLPF)

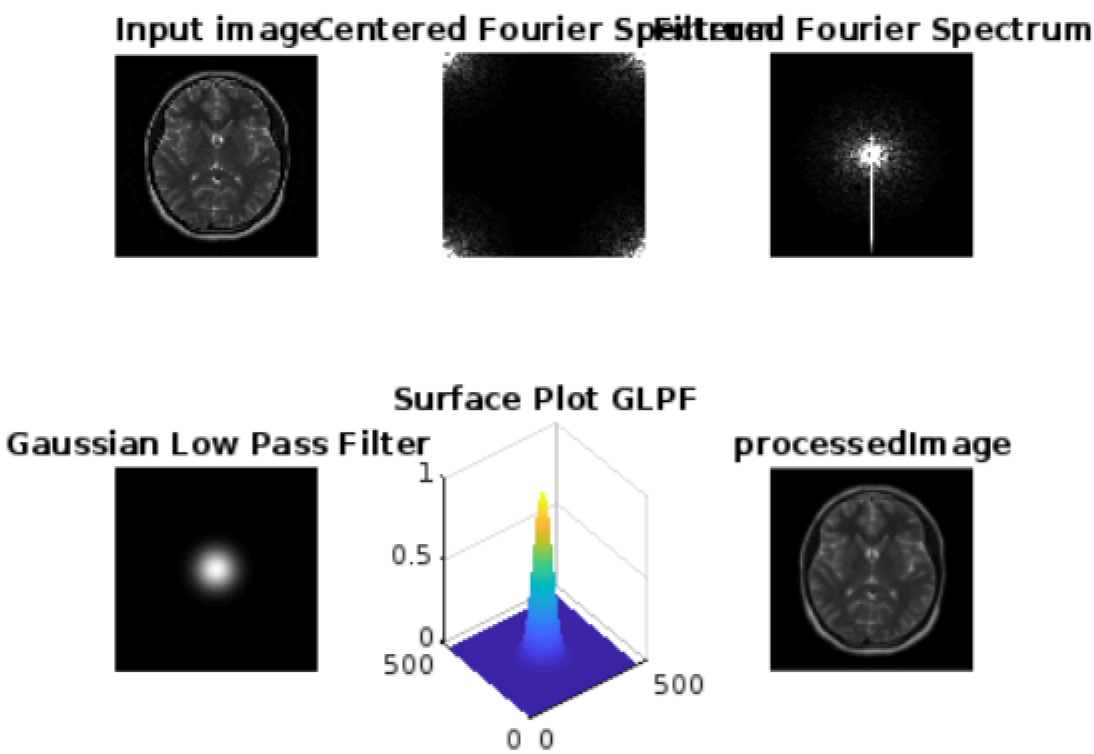
Code:

```
clc;
clear;
close all;
```

```
a = imread('gray img.png');
a = rgb2gray(a);
a = im2double(a);
[m,n] = size(a);
subplot(2,3,1)
imshow(a)
title('Input image');
A = fft2(a);
subplot(2,3,2);
imshow(uint8(abs(A)));
title('Centered Fourier Spectrum');
A_shift = fftshift(A);
A_real = abs(A_shift);
subplot(2,3,3)
imshow(uint8(A_real));
title('Filtered Fourier Spectrum');
D0 = 30;
for u = 1:m
   for v = 1:n
       D = sqrt((u - m/2)^2 + (v - n/2)^2);
       H(u,v) = exp(-(D^2) / (2*D0^2));
   end
end
H_low = H .* A_shift;
H_low_real = H .* A_real;
H_low_shift = fftshift(H_low);
H_low_image = ifft2(H_low_shift);
subplot(2,3,4)
imshow(H)
title('Gaussian Low Pass Filter');
subplot(2,3,5);
mesh(H)
```

```
title('Surface Plot GLPF')
subplot(2,3,6)
imshow(abs(H_low_image));
title('processedImage');
```

Output:



For producing smoothed images, the Gaussian Low Pass Filter (GLPF) is better than ILPF and GLPF.