

CS443

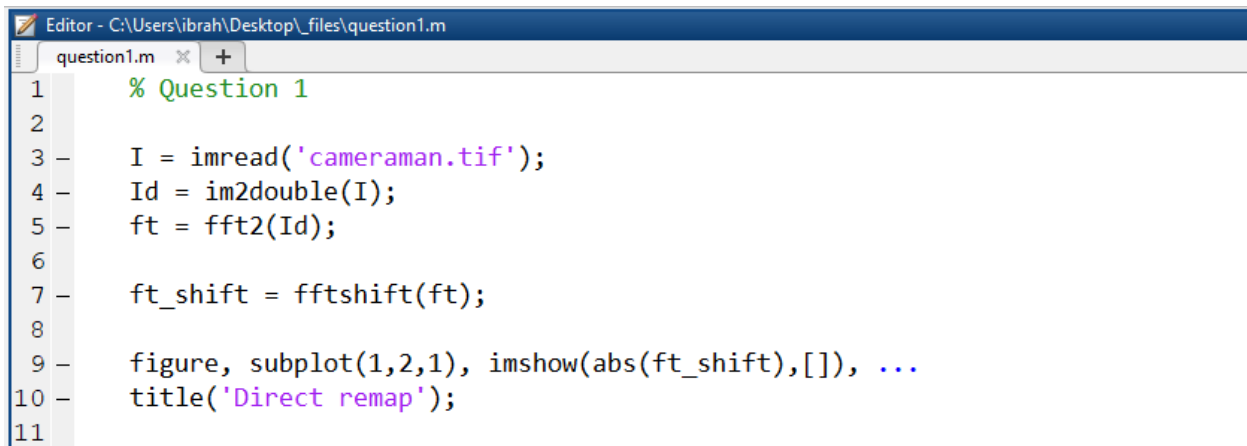
Digital Image Processing

Assignment #4

11.1: 2D Fourier Transform

Procedures

1. Load the cameraman image, convert it to double (one of the data classes) accepted as an input to `fft2`), and generate its FT.
2. Shift the FT array of results.
3. Display the FT results, remapped to a grayscale range.



```
Editor - C:\Users\ibrah\Desktop\files\question1.m
question1.m  x  +
1  % Question 1
2
3 - I = imread('cameraman.tif');
4 - Id = im2double(I);
5 - ft = fft2(Id);
6
7 - ft_shift = fftshift(ft);
8
9 - figure, subplot(1,2,1), imshow(abs(ft_shift),[]), ...
10 - title('Direct remap');
11
```

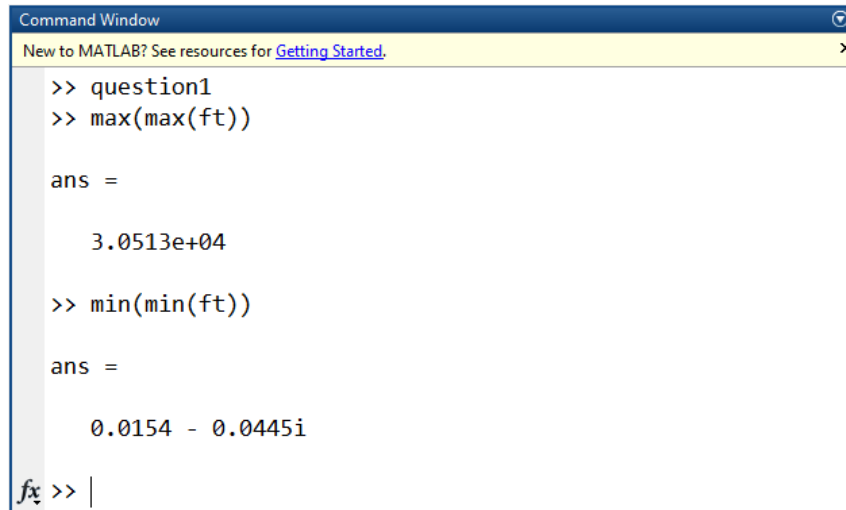
Code Fragment 1.1: Procedures 1, 2 and 3 applied.

Question 1

What are the minimum and maximum values of the resulting discrete Fourier transform coefficients for the **cameraman** image?

Answer 1

The minimum and the maximum values can be obtained by `max(max(ft))` and `min(min(ft))` functions as shown in the Command 1.1. ■



```

Command Window
New to MATLAB? See resources for Getting Started.
>> question1
>> max(max(ft))

ans =

    3.0513e+04

>> min(min(ft))

ans =

    0.0154 - 0.0445i
fx >> |

```

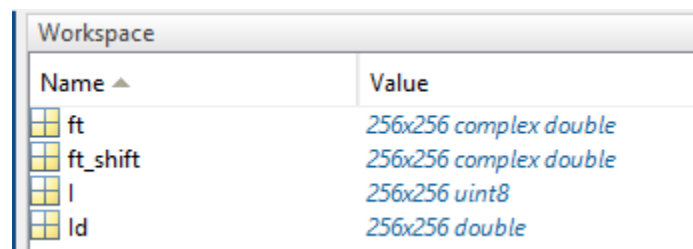
Command 1.1: Values of `max(max(ft))` and `min(min(ft))`

Question 2

Why are we required to use the **abs** function when displaying the `ft_shift` image?

Answer 2

As shown in Figure 2, `ft_shift` is a 256×256 *complex double variable*. In order for us to plot this, it must be converted into *double*. For that operation, we use **abs** function. ■



Name ▲	Value
ft	256x256 complex double
ft_shift	256x256 complex double
I	256x256 uint8
Id	256x256 double

Figure 1.1: Variable names and their values

4. Display the log of the shifted FT image.

```

11
12 - subplot(1,2,2), imshow(log(1 + abs(ft_shift)), []), ...
13 - title('Log remap');

```

Code Fragment 1.2: Procedure 4 applied.

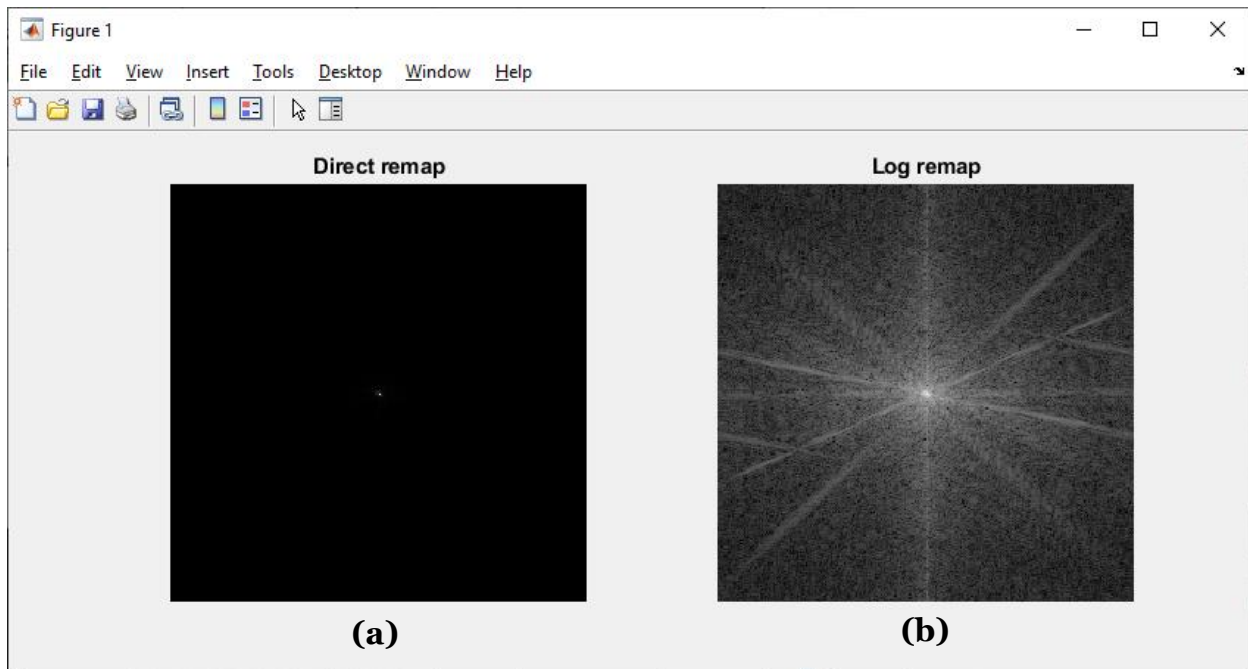


Figure 1.2: Output of Procedures [1-4]
(a) Direct remap, (b) Log remap

Question 3

How did we remap the image to a different (narrower) grayscale range?

Answer 3

Different (narrower) grayscale range has been remapped by use of log transformation as shown in Code Fragment 2. ■

```

9 - figure, subplot(1,2,1), imshow(abs(ft_shift),[]), ...
10 - title('Direct remap');
11
12 - subplot(1,2,2), imshow(log(1 + abs(ft_shift)), []), ...
13 - title('Log remap');

```

Code Fragment 1.3: Procedure 3 and 4 are shown.

Question 4

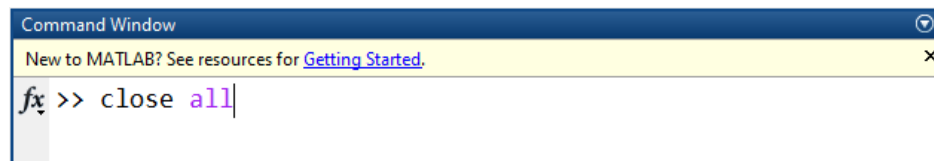
How does the **log remap** compare to the **direct remap**?

Answer 4

When **log remap** compared to **direct remap**, it is obviously seen from the Figure 3, that **log remap** shows a detailed information, including lines and shining areas rather than having just a single bright point. With **log transformation**, we remove distorted brightness and get a balanced output. ■

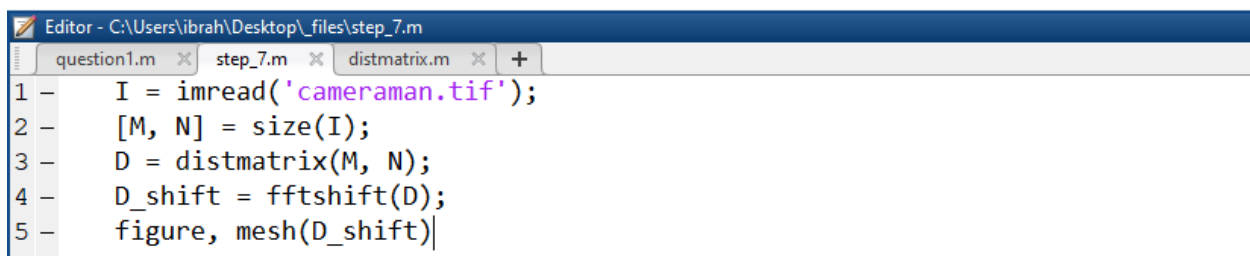
Distance Matrices

5. Close any open figures.



Command 1.2: Procedure 5 applied

6. Generate a distance matrix that is the same size as the image I.
7. Create a 3D mesh plot of the distance matrix.



Code Fragment 1.4: Procedure 6 and 7 applied.

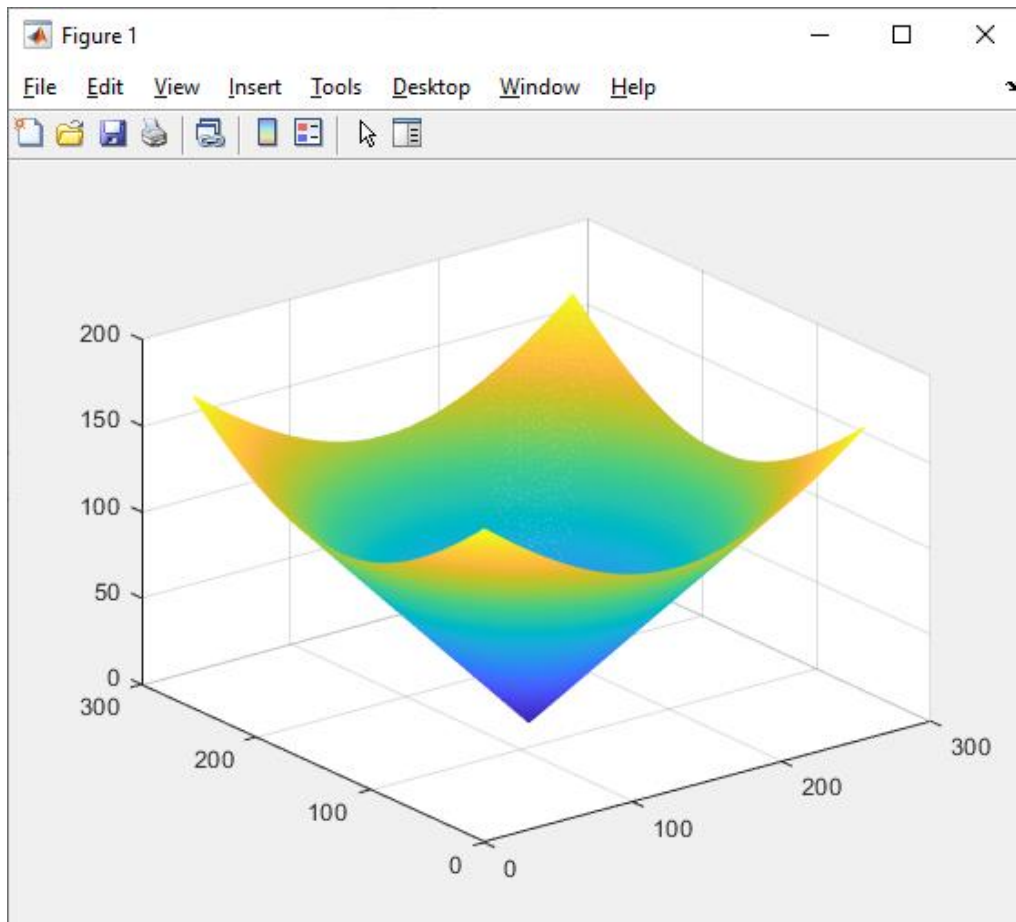


Figure 1.3: Output of Procedures [6-7], 3D mesh plot of distance matrix

Question 5

Explain the shape of the 3D plot.

Answer 5

The 3D Mesh Plot in Figure 3 shows the response function of frequency domain filter and it is similar to the low-pass filter equivalent to average filter in the spatial domain. ■

11.2: Low-pass Filters In the Frequency Domain

Procedures

1. Load the **eight** image, generate a FT, and display it.
2. Shift the FT array of results.

```

Editor - C:\Users\ibrah\Desktop\files\lowpass.m
lowpass.m  x  +
1 - I = imread('eight.tif');
2 - Id = im2double(I);
3 - I_dft = fft2(Id);
4 - figure, imshow(Id), title('Original Image');
5 - figure, imshow(log(1 + abs(fftshift(I_dft))),[]), ...
6 - title('FT of original image');
7
8 - [M, N] = size(I);
9 - dist = distmatrix(M, N);
10 - figure, mesh(fftshift(dist)), title('Distance Matrix');

```

Code Fragment 2.1: Procedure 1 and 2 applied.

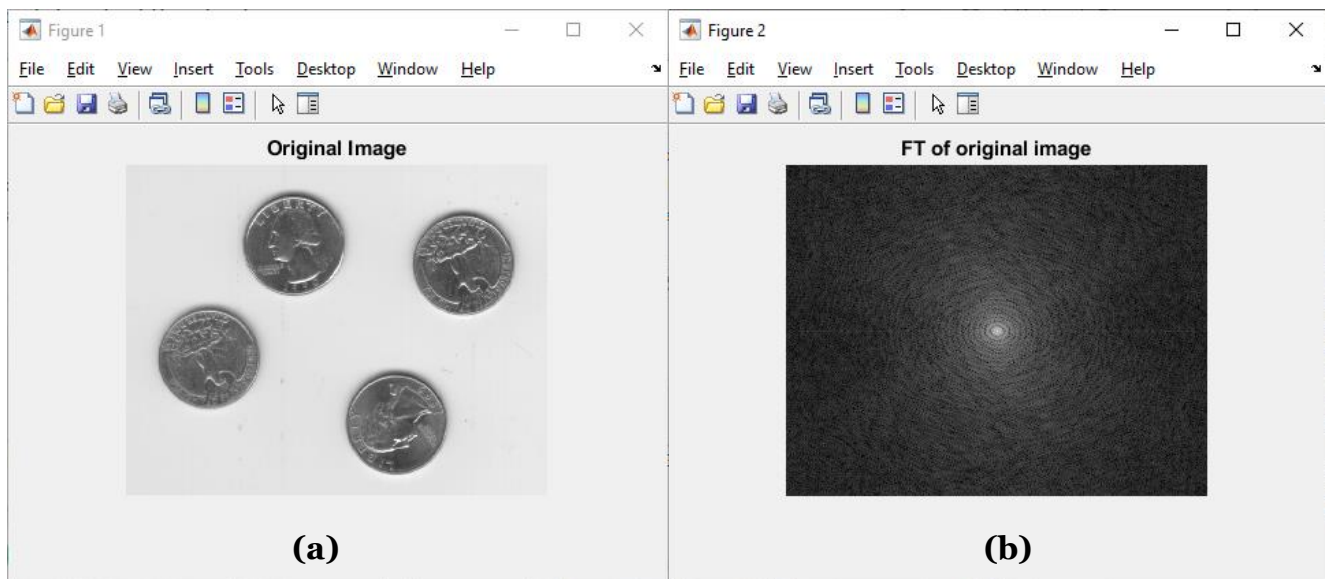
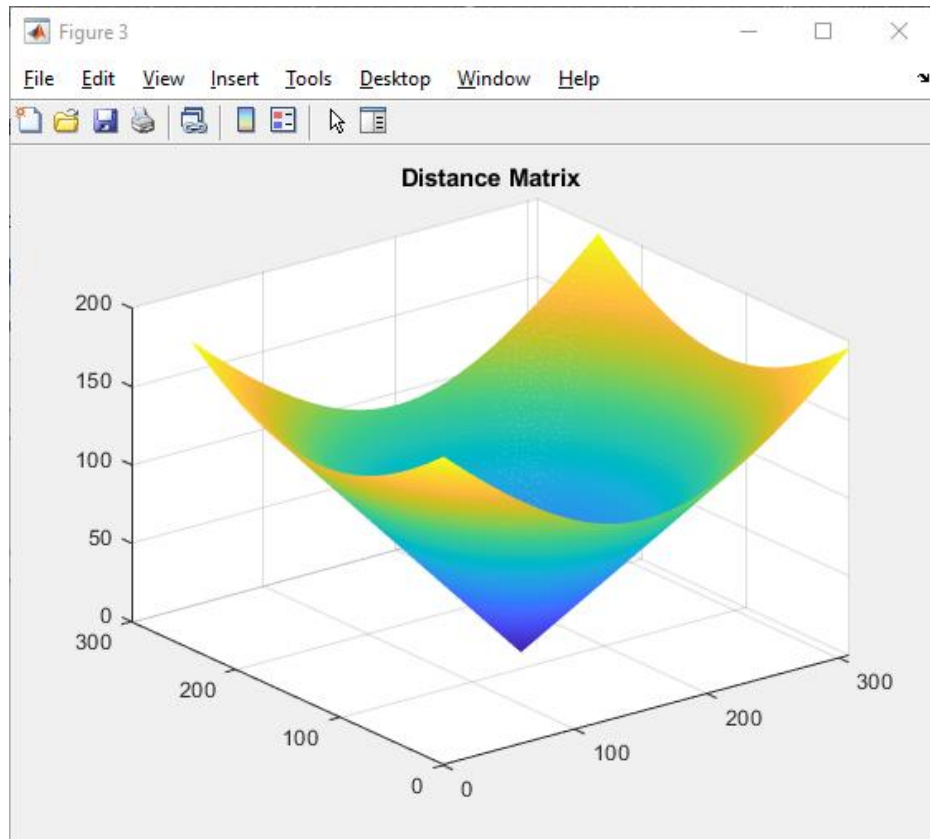


Figure 2.1: Output of Procedures [1-2]
 (a) Original
 (b) Fourier Transform of Original Image

Figure 2.2: Distance Matrix of **eight.tif****Question 1**

Verify that the size of the distance matrix is in fact equal to the size of the image.

Answer 1

Size of the distance matrix is the same as size of the image, as seen in Command 2.1. ■

```

Command Window
New to MATLAB? See resources for Getting Started.

>> lowpass
>> size(dist)

ans =

    242    308

>> size(I)

ans =

    242    308

fx >>

```

Command 2.1: Value of **size(dist)** is the same as **size(I)**

Question 2

What happens if we display the distance matrix without shifting?

Answer 2

Displaying the matrix without shifting yields an error as the x, y and z axes cannot be shown with a complex number in the coordinate plane. ■

Ideal LPF

3. Create initial filter with all values of zero.
4. Create the ideal filter.

```
13 - H = zeros(M, N);  
14  
15 - radius = 35;  
16 - ind = dist <= radius;  
17 - H(ind) = 1;  
18 - Hd = double(H);
```

Code Fragment 2.2: Procedure 3 and 4 applied.

Question 3

Explain *how* the previous code sets all values within a given radius to 1.

Answer 3

Initial step is defining variable **H**, which holds zeros of variables **M** and **N**. Then, variable **radius** is initialized as **35**, and the values from **dist** less than or equal to that radius are stored in variable **ind**, consisting of values **1s** and **0s**. At the last line of Code Frag. 2.2, **H(ind)** is set to **1**. ■

5. Display the filter's frequency response.
6. Apply the filter to the FT image.

```
20 - figure, imshow(fftshift(H)), title('Ideal low-pass filter');  
21  
22 - DFT_filt = Hd .* I_dft;  
23 - I2 = real(ifft2(DFT_filt));
```

Code Fragment 2.3: Procedure 5 and 6 applied.

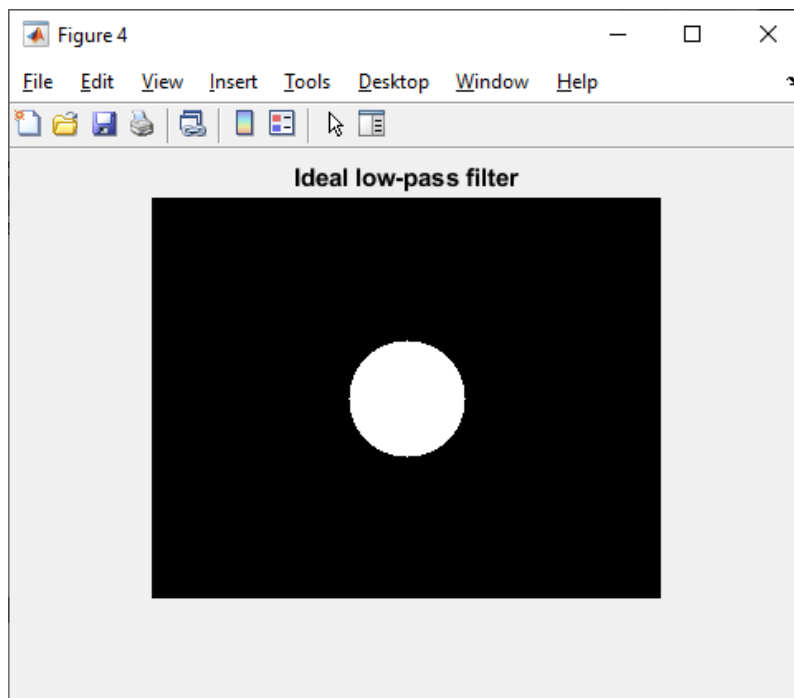


Figure 2.3: Output of Procedure [5-6].
Ideal low-pass filter of **eight.tif**

Question 4

Why do we take only the real values when converting the FT of the filtered image back to the spatial domain?

Answer 4

FT contains complex numbers, we have to take the real values as having complex numbers creates problem in displaying image. ■

7. Display both the filtered FT image and the final filtered image.

```
24 - subplot(3, 2, 5), imshow(log(1 + abs(fftshift(DFT_filt))),[]), ...
25 - title('Filtered FT');
26 - subplot(3, 2, 6), imshow(I2), title('Filtered Image');
```

Code Fragment 2.4: Procedure 7 applied.

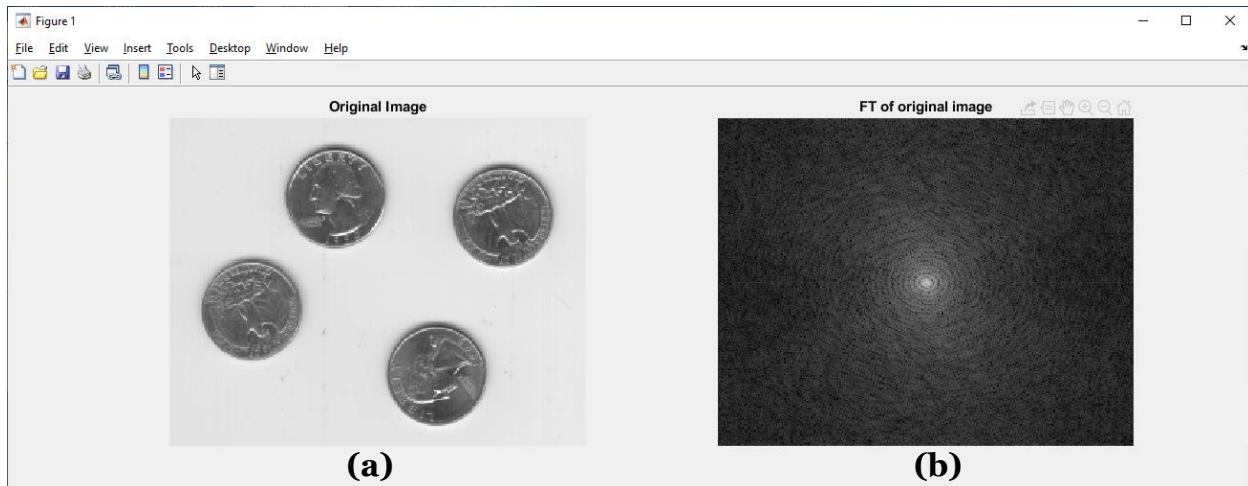


Figure 2.4: Output of Procedure 7
(a) Original Image, (b) FT of Original Image **eight.tif**

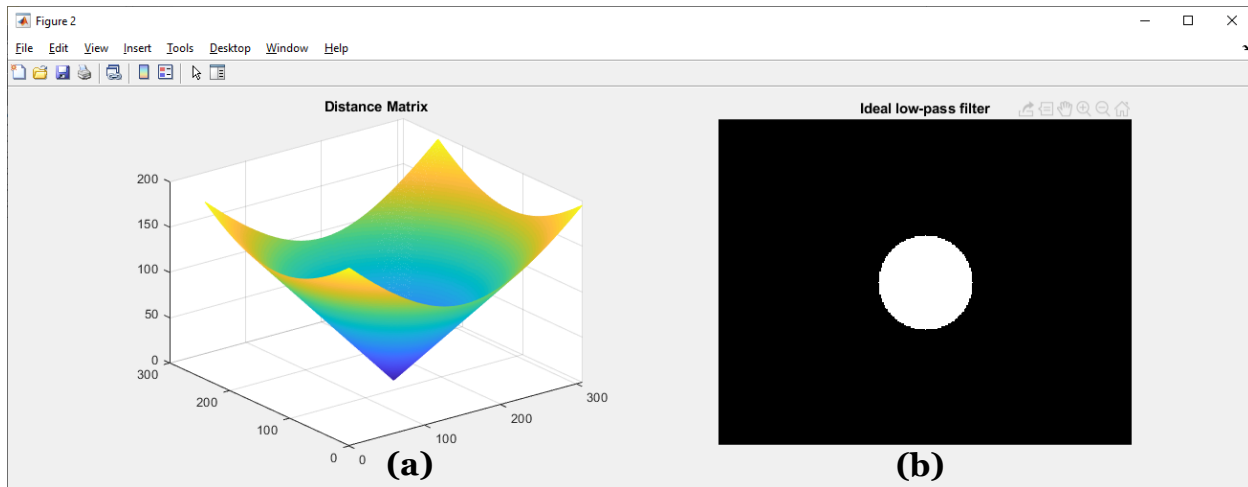


Figure 2.5: Output of Procedure 7
(a) Distance Matrix, (b) Ideal Low-pass Filter of **eight.tif**

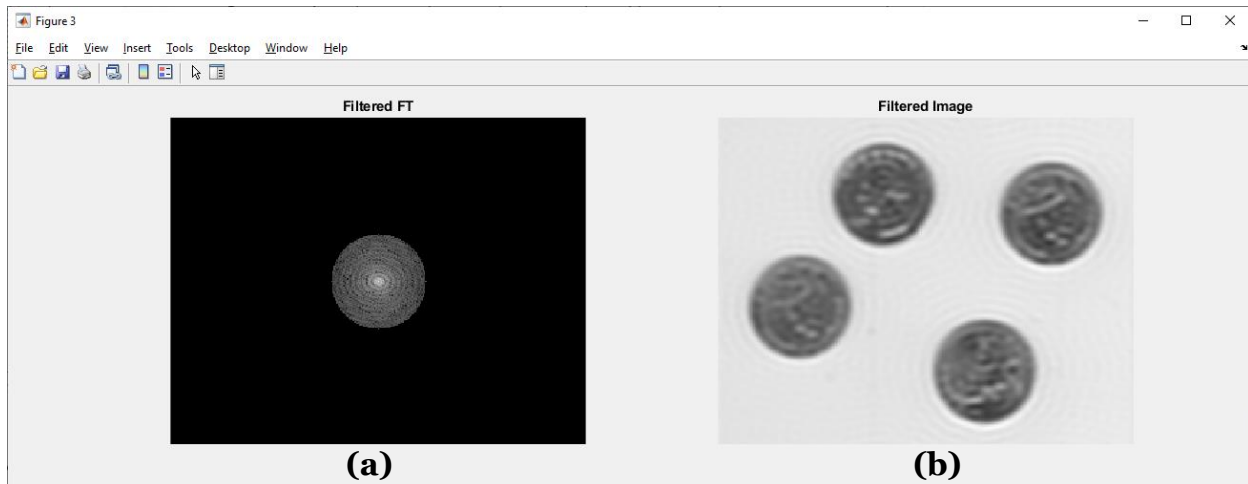


Figure 2.6: Output of Procedure 7
(a) Filtered FT, (b) Filtered Image of **eight.tif**

Question 5

How does the filtered image compare to the original image? Can you see any noticeable artifacts?

Answer 5

Figure 2.4 (a), which is the original image, contains much details compared to Figure 2.6 (b), which is Filtered Image. This implies that there exists noticeable artifacts, that is filtered image loses the details and it is blurry compared to original image. ■

Question 6

Experiment with different values for the radius of the filter. How does your choice of radius affect the amount of ringing in the output image?

Answer 6

Decreasing the radius makes the filtered image blurry. ■

Gaussian LPF

10. Create a Gaussian low-pass filter with $\sigma = 30$.
11. Filter the FT image with the Gaussian low-pass filter and display the filtered image.

```

Editor - C:\Users\ibrah\Desktop\files\part2_gaussian.m
lowpass.m  part2_gaussian.m  +
1 - I = imread('eight.tif');
2 - Id = im2double(I);
3 - I_dft = fft2(Id);
4
5 - [M, N] = size(I);
6 - dist = distmatrix(M, N);
7
8 - sigma = 30;
9 - H_gau = exp(-(dist.^2) / (2 * (sigma ^ 2)));
10 - subplot(1, 2, 1), imshow(Id), title('Original Image');
11 - subplot(1, 2, 2), imshow(log(1 + abs(fftshift(I_dft))), [], ...
12 - title('DFT of original image');
13 - figure, subplot(1, 2, 1), mesh(fftshift(dist)), title('Distance Matrix');
14 - subplot(1, 2, 2), imshow(fftshift(H_gau)), title('Gaussian low-pass');
15
16 - DFT_filt_gau = H_gau .* I_dft;
17 - I3 = real(ifft2(DFT_filt_gau));
18 - figure, subplot(1, 2, 1), imshow(log(1 + abs(fftshift(DFT_filt_gau))), ...
19 - [], title('Filtered FT');
20 - subplot(1, 2, 2), imshow(I3), title('Filtered Image');

```

Code Fragment 2.5: Procedure 10 and 11 applied.

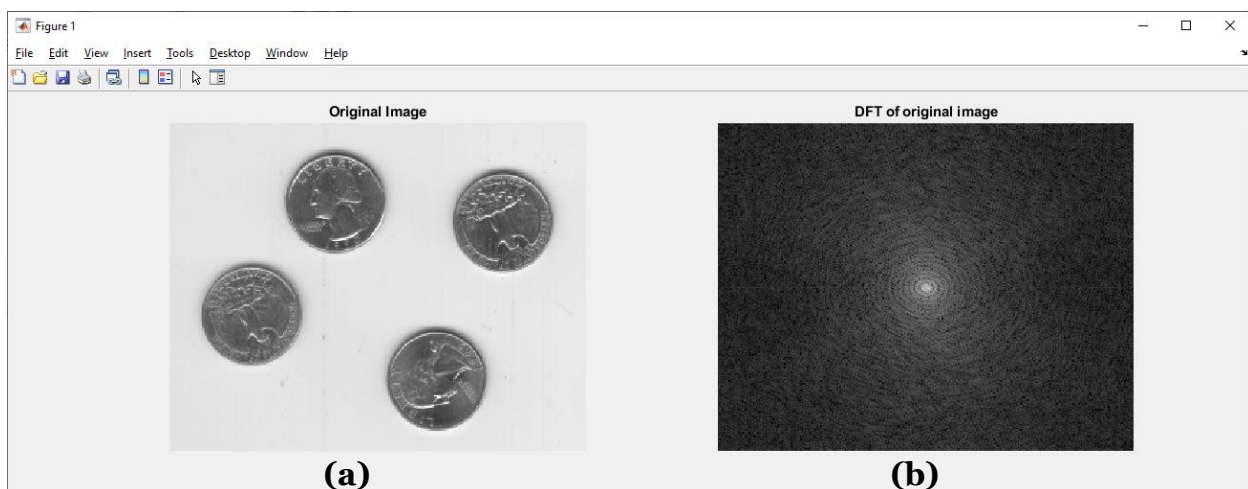


Figure 2.7: Output of Procedure [10-11]
 (a) Original Image, (b) DFT of Original Image **eight.tif**

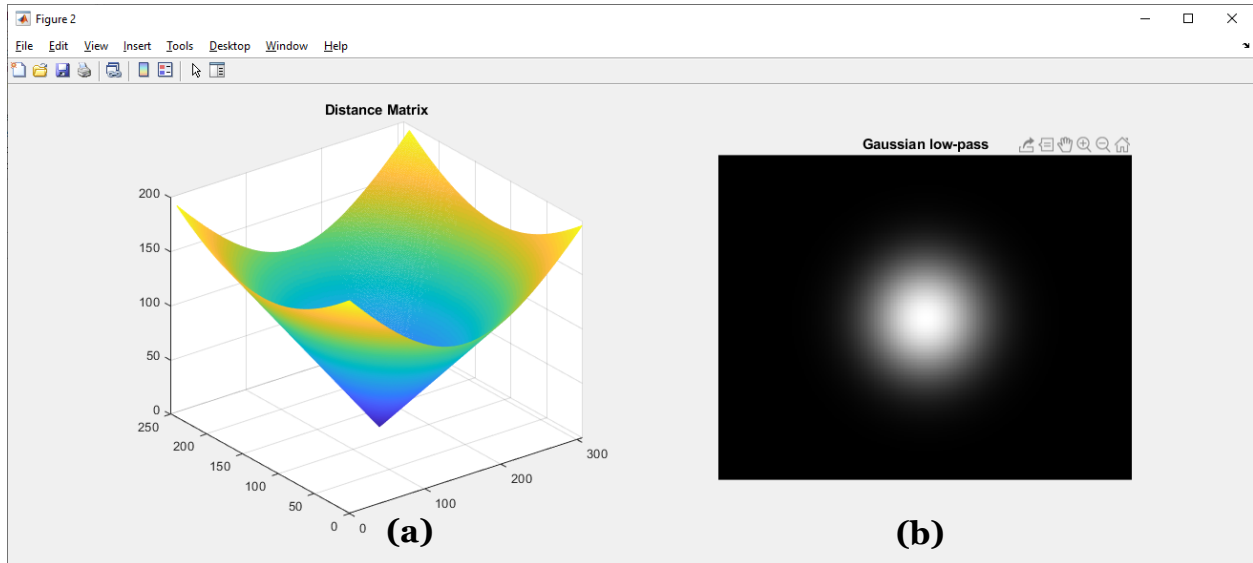


Figure 2.8: Output of Procedure [10-11]
(a) Distance Matrix, (b) Gaussian Low-pass Filter of **eight.tif**

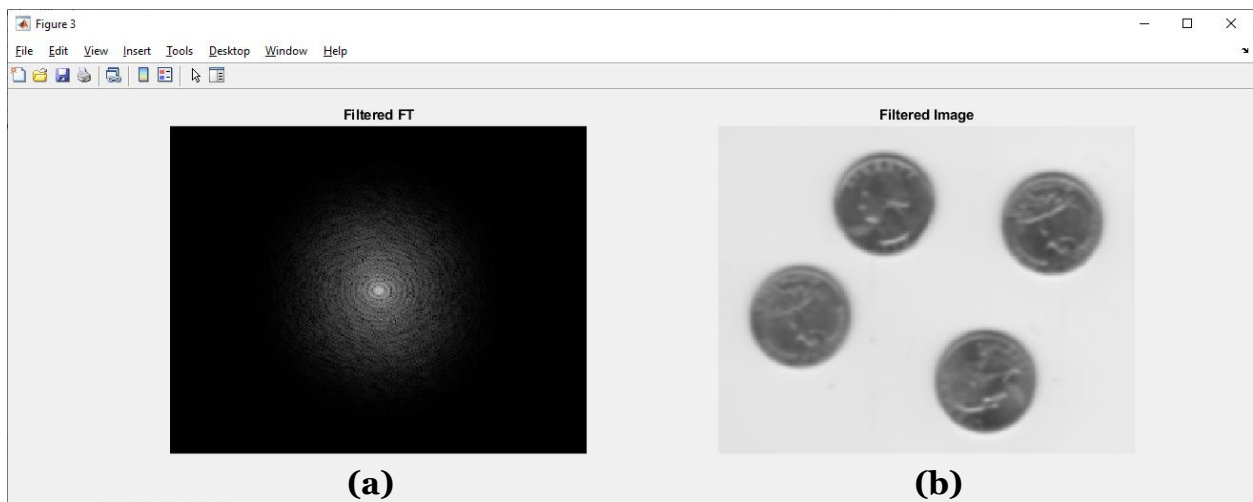


Figure 2.9: Output of Procedure [10-11]
(a) Filtered FT, (b) Filtered Image of **eight.tif**

Question 7

Compare the output images between the ideal filter and the Gaussian filter. What are their similarities? What are their differences?

Answer 7

Taking the output images of Ideal filter and Gaussian filter into consideration, it can be concluded that in the behavior seem more natural in Gaussian Low-pass filter and transition occurs gradually whereas in the Ideal Low-pass filter, there exists ringing artifacts. ■

Butterworth LPF

12. Generate a third-order Butterworth filter, where the cutoff value is 35.
13. Filter the image with the Butterworth low-pass filter and display the resulting image.

```

Editor - C:\Users\Vibrah\Desktop\_files\Butterworth_LPF.m
lowpass.m  part2_gaussian.m  Butterworth_LPF.m  +
1 - I = imread('eight.tif');
2 - Id = im2double(I);
3 - I_dft = fft2(Id);
4
5 - [M, N] = size(I);
6 - dist = distmatrix(M, N);
7
8 - D0 = 35; n = 3;
9 - H_but = 1 ./ (1 + (dist ./ D0) .^ (2 * n));
10 - subplot(1, 2, 1), imshow(Id), title('Original Image');
11 - subplot(1, 2, 2), imshow(log(1 + abs(fftshift(I_dft))),[]), ...
12 - title('FT of original image');
13 - figure, subplot(1, 2, 1), mesh(fftshift(dist)), title('Distance Matrix');
14 - subplot(1, 2, 2), imshow(fftshift(H_but)), title('Butterworth low-pass');
15
16 - DFT_filt_but = H_but .* I_dft;
17 - I4 = real(ifft2(DFT_filt_but));
18 - figure, subplot(1, 2, 1), imshow(log(1 + abs(fftshift(DFT_filt_but))),[]),
19 - title('Filtered FT');
20 - subplot(1, 2, 2), imshow(I4), title('Filtered Image');|

```

Code Fragment 2.6: Procedure 12 and 13 applied.

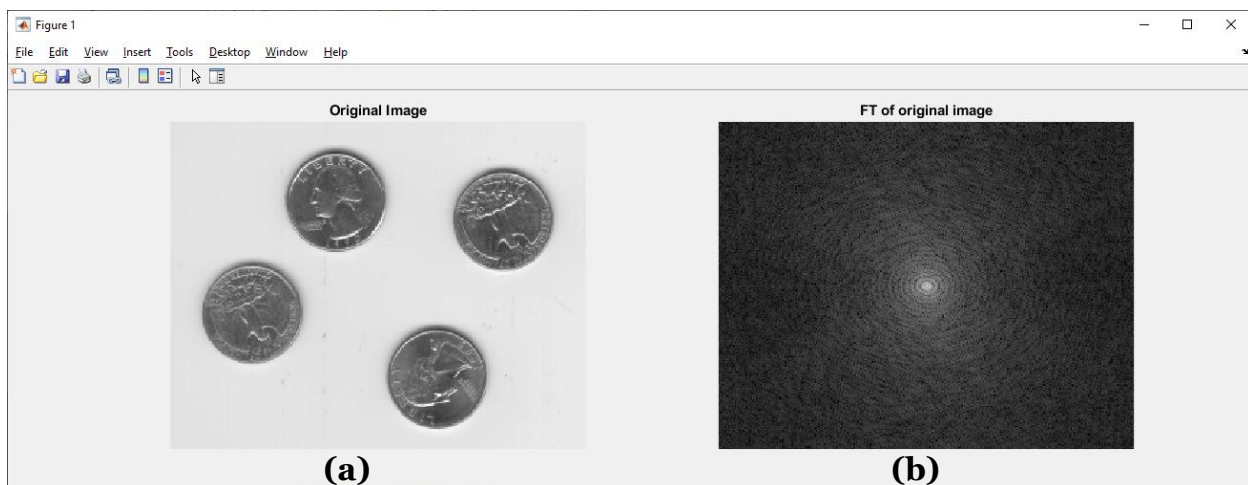


Figure 2.10: Output of Procedure [12-13]
(a) Original Image, (b) FT of Original Image **eight.tif**

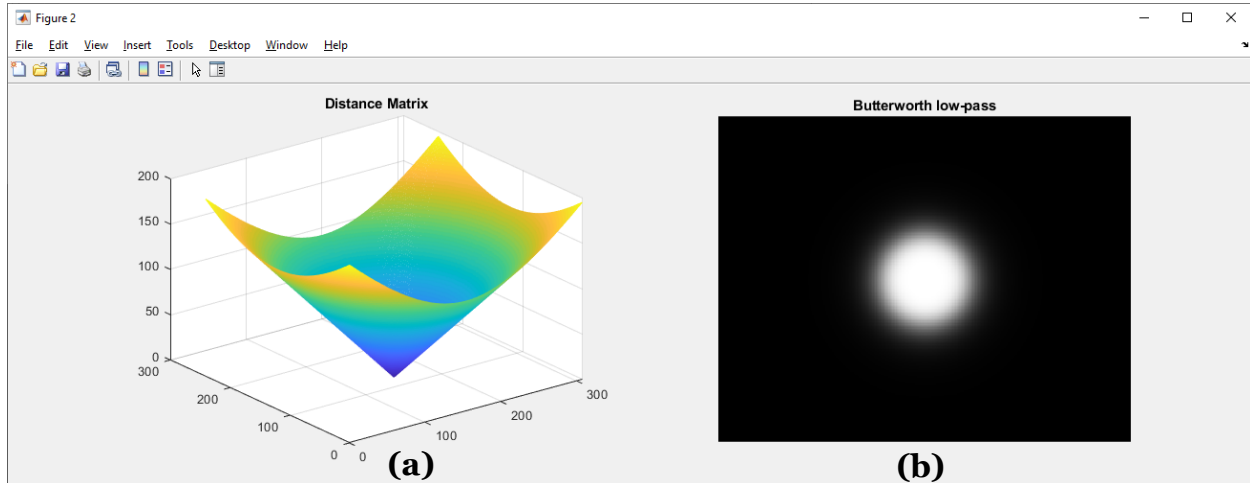


Figure 2.11: Output of Procedure [12-13]
 (a) Distance Matrix, (b) Butterworth Low-pass Filter of **eight.tif**

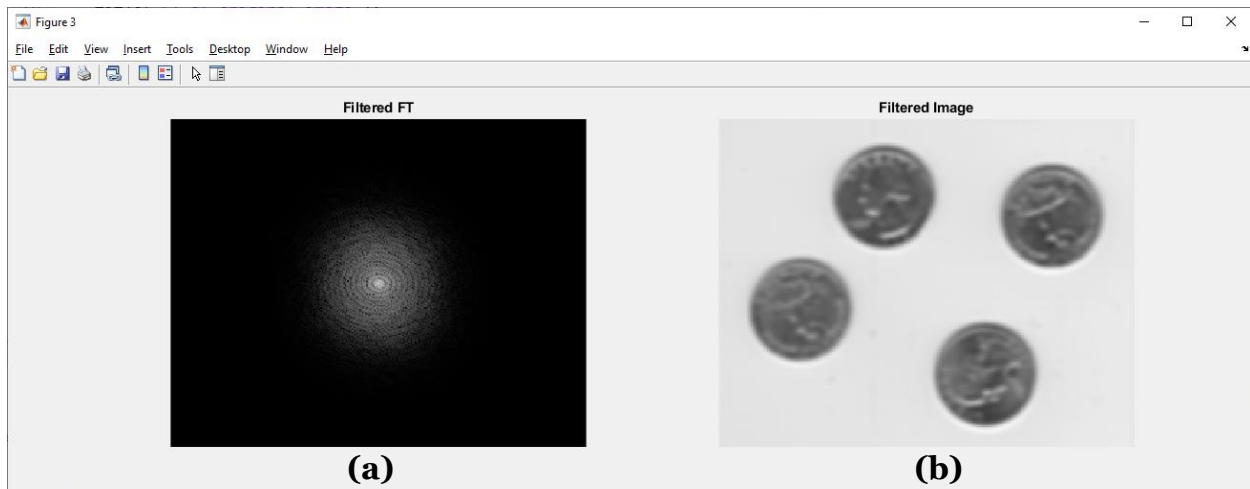


Figure 2.12: Output of Procedure [12-13]
 (a) Filtered FT, (b) Filtered Image of **eight.tif**

Question 9

Compare the ideal-filtered FT image and the Butterworth-filtered FT image.

Answer 9

Comparing the Ideal-Filtered FT image and Butterworth-Filtered FT image, it can be concluded that in the Butterworth-Filtered image, there exists less ringing than the Ideal-Filtered image. ■

14. Display all three filters as meshes in 3D and use the Rotate 3D option of function **imshow** to explore them in detail.

```
Editor - C:\Users\ibrah\Desktop\files\hw11_2_p14.m
lowpass.m  part2_gaussian.m  Butterworth_LPF.m  hw11_2_p14.m  +
1 - I = imread('eight.tif');
2 - Id = im2double(I);
3 - I_dft = fft2(Id);
4
5 - [M, N] = size(I);
6 - dist = distmatrix(M, N);
7
8 % Ideal Low-Pass Filter
9 - H = zeros(M, N);
10 - radius = 35;
11 - ind = dist <= radius;
12 - H(ind) = 1;
13 - Hd = double(H);
14
15 % Gaussian Low-Pass Filter
16 - sigma = 30;
17 - H_gau = exp(-(dist.^2) / (2 * (sigma ^ 2)));
18
19 % Butterworth Low-Pass Filter
20 - D0 = 35; n = 3;
21 - H_but = 1 ./ (1 + (dist ./ D0).^ (2 * n));
22
23 - subplot (1,2,1),mesh(fftshift(Hd)),title('Ideal low-pass filter');
24 - subplot (1,2,2),mesh(fftshift(H_gau)),title('Gaussian low-pass filter');
25 - figure, mesh(fftshift(H_but)),title('Butterworth low-pass filter');
```

Code Fragment 2.6: Procedure 14 applied.

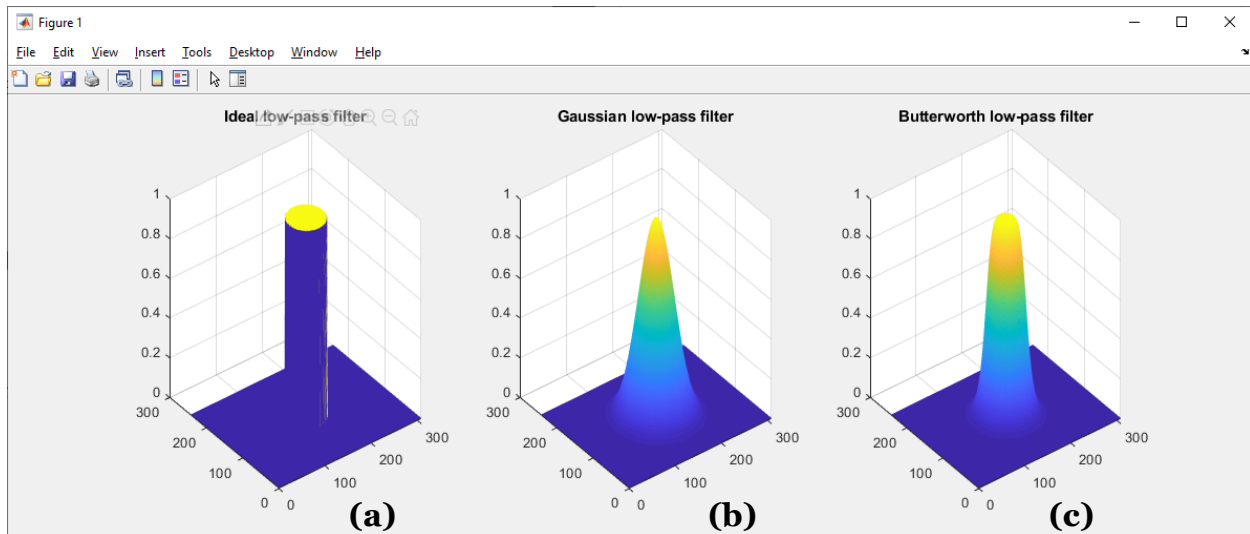


Figure 2.13: 3D Mesh Plot outputs of Procedure [14]
 (a) Ideal LPF, (b) Gaussian LPF and (c) Butterworth LPF of **eight.tif**

Question 10

Implement the Butterworth filter again, but this time using a much higher order, such as 20. How does this output compare to the ideal filter?

Answer 10

After the implementation of Butterworth Filter again, the output image gets blurrier and its ringing artifacts becomes more visible. ■

Question 11

Experiment with the Butterworth filter by using the frequency-domain demo (fddemo). What is the advantage of using this filter over the previous two?

Answer 11

Sharpness can be controlled with Butterworth filter. A High-pass filter is holding the frequencies outside of radius. There is gradual change, this will reduce ringing effect. ■

11.3: High-pass filters in the frequency domain

Procedures

1. Load the **eight** image, generate a FT, and display it.
2. Generate a distance matrix based on the size of the input image.

Ideal HPF

3. Create the ideal high-pass filter.

```

Editor - C:\Users\ibrah\Desktop\_files\hw4_11_3_p3.m
distmatrix.m  hw4_11_3_p3.m  +
1 - I = im2double(imread('eight.tif'));
2 - I_dft = fft2(I);
3 - figure, imshow(I), title('Original Image');
4 - figure, imshow(log(1 + abs(fftshift(I_dft))),[]), ...
5 - title('FT of original image');
6
7 - [M, N] = size(I);
8 - dist = distmatrix(M, N);
9
10 - H = ones(M, N);
11 - radius = 30;
12 - ind = dist <= radius;
13 - H(ind) = 0;

```

Code Fragment 3.1: Procedure 1-3 applied.

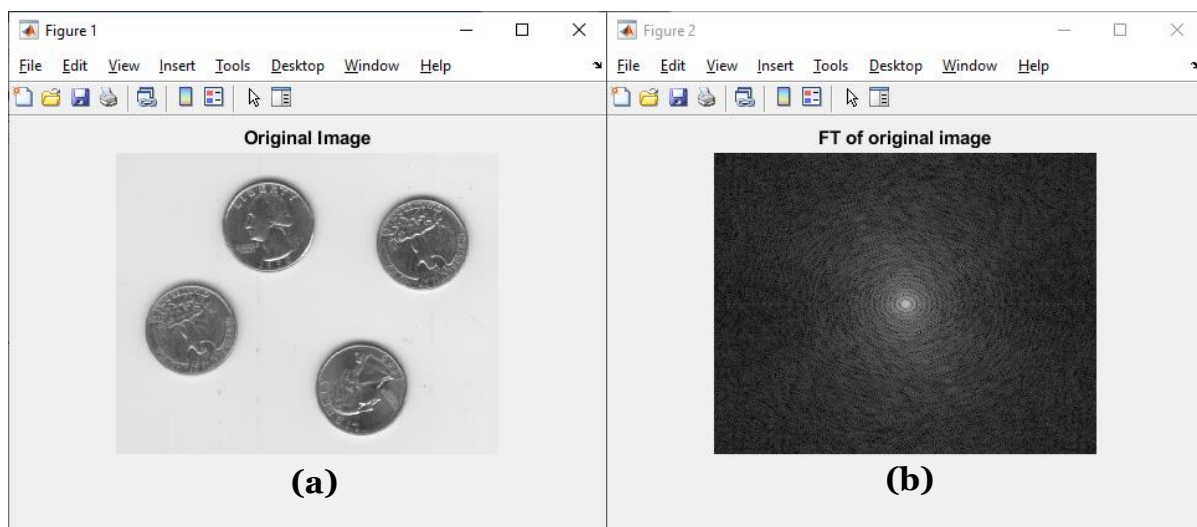


Figure 3.1: Outputs of Procedure [1-3].
(a) Original Image, (b) FT of the Original image

Question 1

Explain how the previous code generates an ideal high-pass filter.

Answer 1

Similar to Ideal Low-pass filter, the values less than or equal to 1, will be assign to **ind**, and 0 (zero) values will be assigned to **H(ind)** and rest of the values remains 1 (one). ■

4. Apply high-frequency emphasis filtering to the high-pass filter.
5. Apply the filter to the FT image and display the results.
6. Display the filter as an image and as a 3D mesh in separate figures.

```

15 - a = 1; b = 1;
16 - Hd = double(a + (b .* H));
17
18 - DFT_filt = Hd .* I_dft;
19 - I2 = real(ifft2(DFT_filt));
20 - figure, imshow(log(1 + abs(fftshift(DFT_filt))),[]), ...
21 - title('Filtered FT');
22 - figure, imshow(I2), title('Filtered Image');
23
24 - figure, imshow(fftshift(Hd),[]), title('Filter as an image');
25 - figure, mesh(fftshift(Hd)), zlim([0 2]), title('Filter as a mesh');|

```

Code Fragment 3.2: Procedure 4-6 applied.

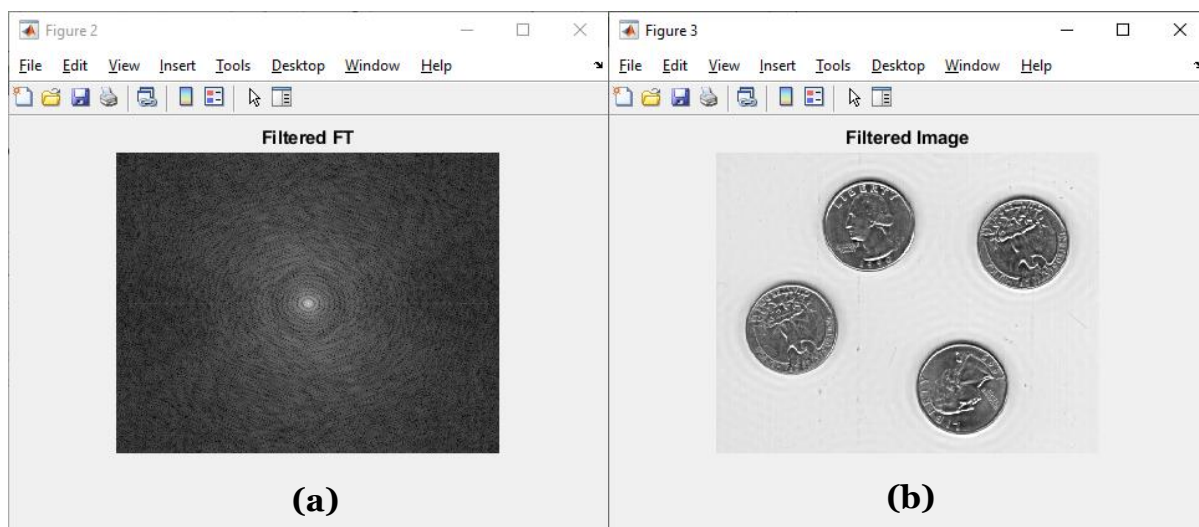


Figure 3.2: Outputs of Procedure [4-6].
(a) Filtered FT, (b) Filtered image

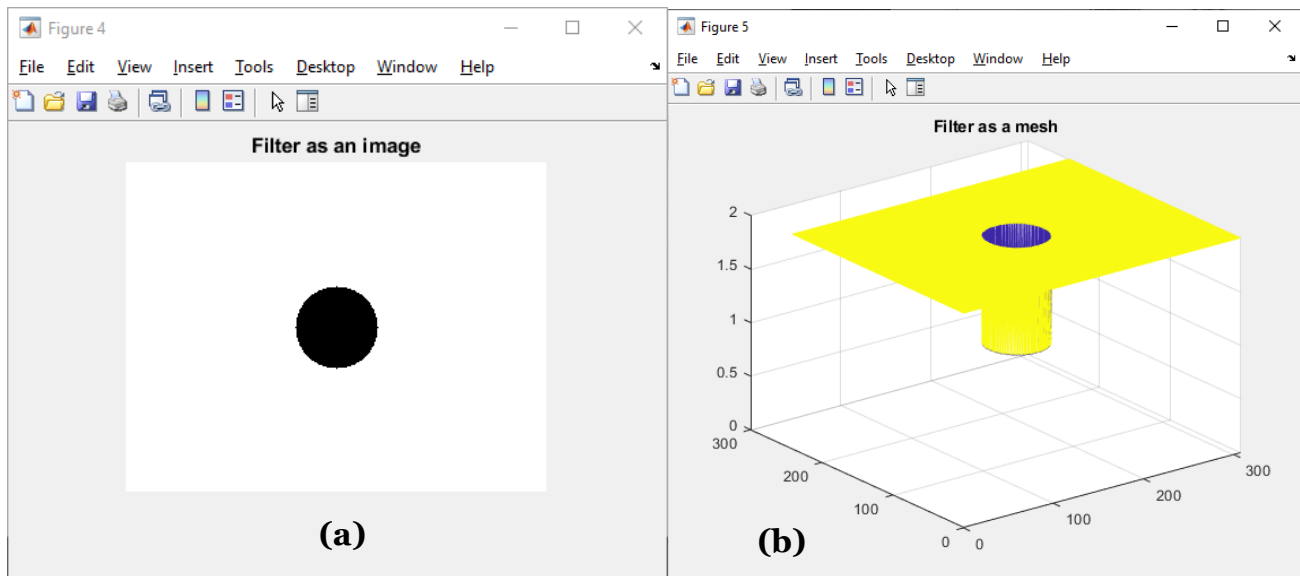


Figure 3.3: Outputs of Procedure [4-6] (Cont.)

(a) Filter as an image, (b) Filter as a mesh

Question 2

When displaying the filter as an image, why must we scale the output for display purposes?

Answer 2

Scaling of the image is needed in order to get image without losing its full size. No-scaling will be resulted in having white screen containing several dots. ■

Question 3

How does the filtered image compare to the original image?

Answer 3

Compared to the original image, filtered image have more sharpness, though the ringing artifacts is not yet been eliminated. ■

Question 4

How does the new image compare to the original image?

Answer 4

The cut-off value has been increased from 10 to 30. Thus, the sharpness has changed. ■

Question 5

How does the output compare between using a cut-off value of 30 and 10?

Answer 5

The change in the cut-off value affects the ringing. The ringing effect loses its visibility in the cut-off value of 30. ■

Question 6

What happens to the filtered image as we increase the cut-off value (beyond 30) with respect to the original image?

Answer 6

The increase in the cut-off value (beyond 30) will result in losing the peak sharpness point, meaning that the sharpness will have a decreasing trend again. ■

Gaussian HPF

12. Generate a Gaussian high-pass filter.
13. Apply high-frequency emphasis filtering to the high-pass filter and display the filter.
14. Apply the filter and display the results.

```

Editor - C:\Users\ibrah\Desktop\_files\hw4_11_3p12.m
distmatrix.m  hw4_11_3_p3.m  hw4_11_3p12.m  +
1 - I = im2double(imread('eight.tif'));
2 - I_dft = fft2(I);
3 - figure, imshow(I), title('Original Image');
4 - figure, imshow(log(1 + abs(fftshift(I_dft))),[]), ...
5 - title('FT of original image');
6
7 - [M, N] = size(I);
8 - dist = distmatrix(M, N);
9
10 - sigma = 30;
11 - H_gau = 1 - exp(-(dist.^2) / (2 * (sigma ^ 2)));
12
13 - H_gau_hfe = a + (b .* H_gau);
14 - figure, mesh(fftshift(H_gau_hfe)), zlim([0 2]), ...
15 - title('Gaussian high-pass filter');
16
17 - DFT_filt_gau = H_gau_hfe .* I_dft;
18 - I3 = real(ifft2(DFT_filt_gau));
19 - figure, imshow(I), title('Original Image');
20 - figure, imshow(log(1 + abs(fftshift(I_dft))),[]), ...
21 - title('FT of original image');
22 - figure, imshow(log(1 + abs(fftshift(DFT_filt_gau))),[]), ...
23 - title('Filtered FT');
24 - figure, imshow(I3), title('Filtered Image');

```

Code Fragment 3.3: Procedure 12-14 applied.

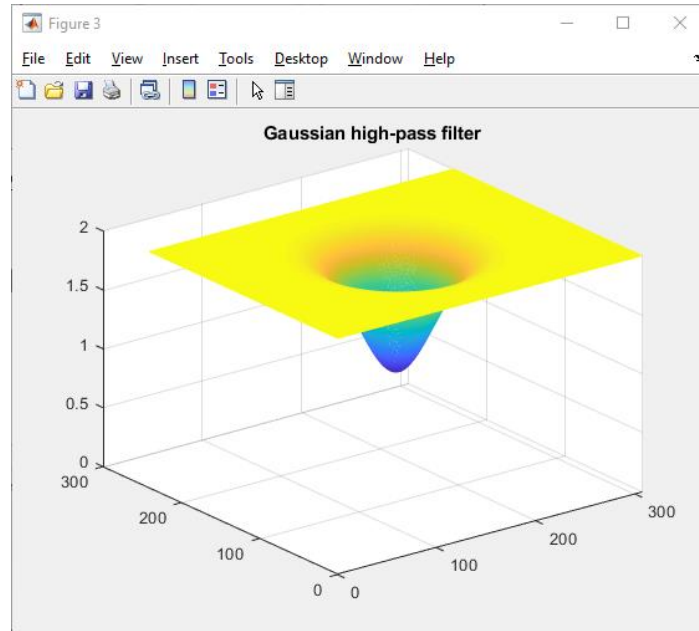


Figure 3.4: Outputs of Procedure [13-14] Gaussian High-pass Filter

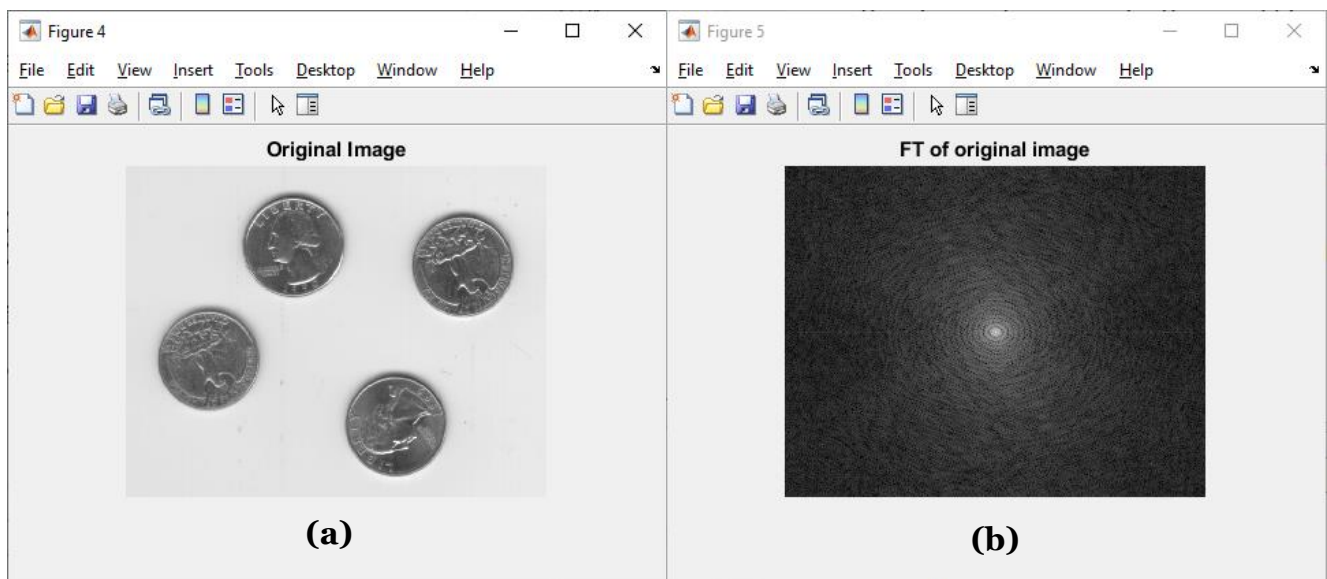


Figure 3.5: Outputs of Procedure [13-14],
(a) Original Image (b) FT of Original Image

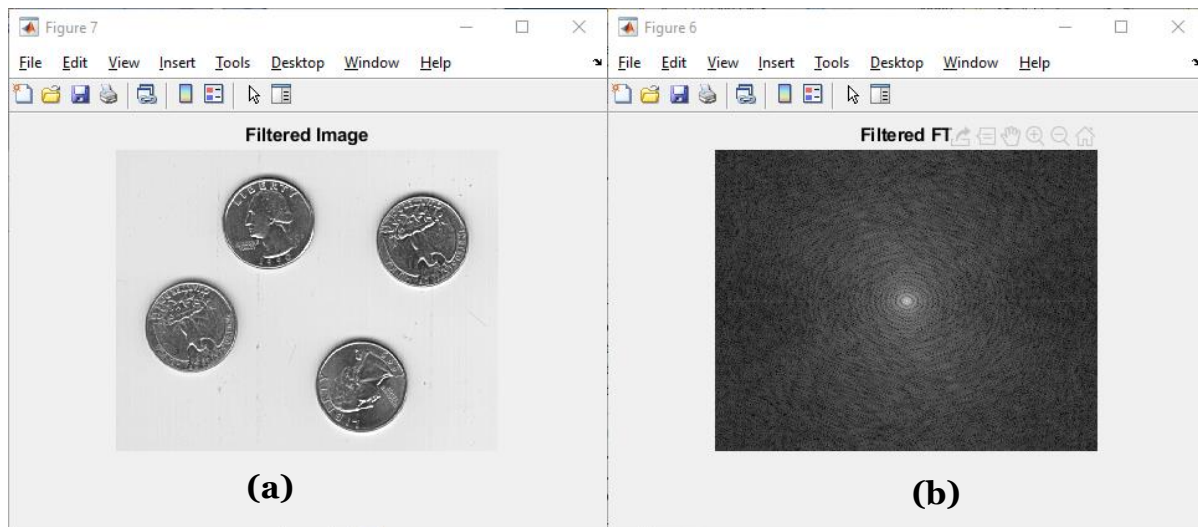


Figure 3.6: Outputs of Procedure [13-14],
(a) Filtered Image (b) FT of Filtered Image

Question 7

How does the Gaussian-filtered image compare to the ideal-filtered image?

Answer 7

While it makes the image sharpen, it does not create ringing effect. ■

Question 8

What happened to the filter?

Answer 8

The value of sigma has been updated. ■

Question 9

How was the filtered image affected?

Answer 9

Having this operation, the image becomes clear. ■

Question 10

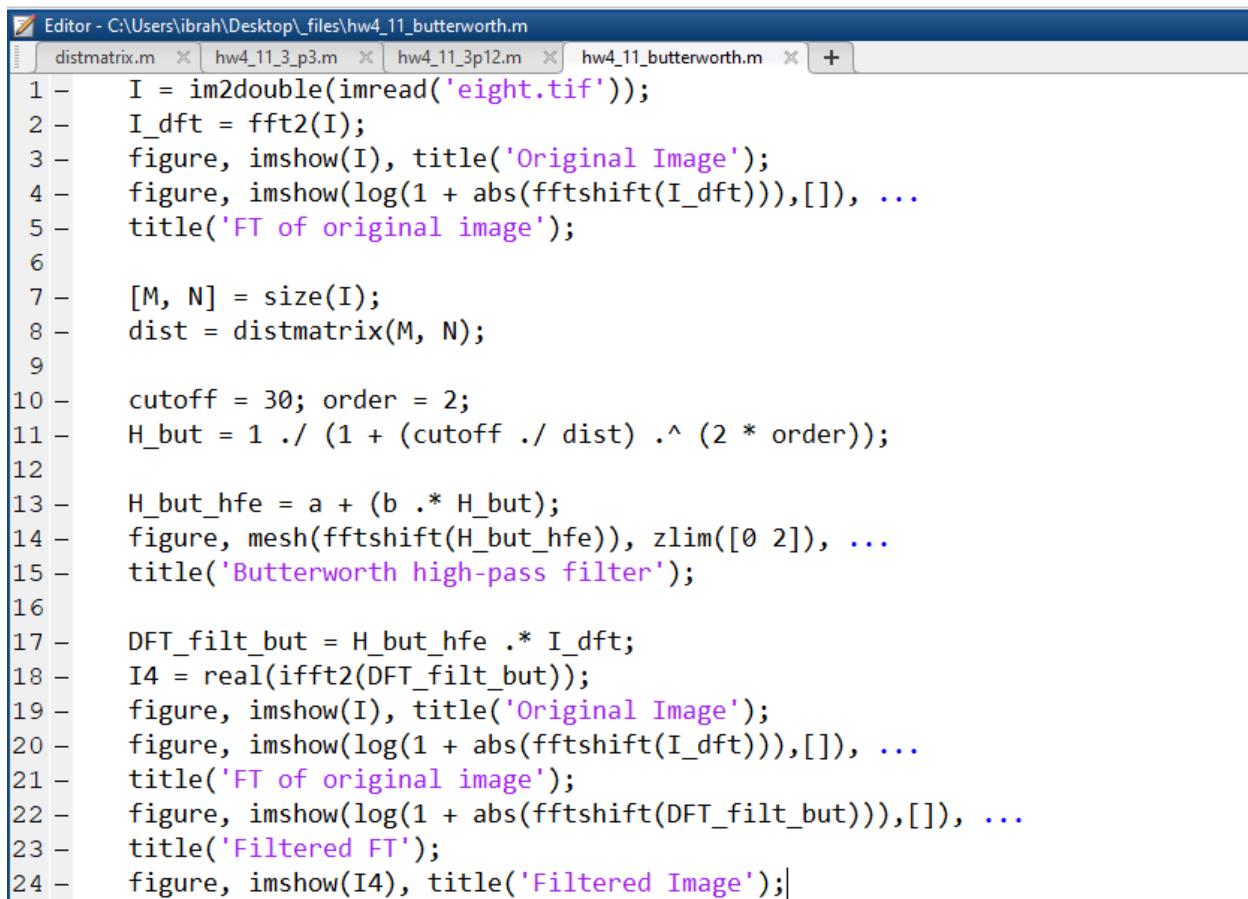
In general, how does the filter change when the standard deviation of the filter is increased or decreased?

Answer 10

The behavior of the filter is depending on the value of sigma which is the standard deviation. The sharpness increases when sigma decreases and vice versa. ■

Butterworth HPF

19. Generate a Gaussian high-pass filter.
20. Apply high-frequency emphasis filtering to the high-pass filter and display the filter.
21. Apply the filter and display the results.



```

Editor - C:\Users\ibrah\Desktop\files\hw4_11_butterworth.m
distmatrix.m  hw4_11_3_p3.m  hw4_11_3p12.m  hw4_11_butterworth.m  +
1 - I = im2double(imread('eight.tif'));
2 - I_dft = fft2(I);
3 - figure, imshow(I), title('Original Image');
4 - figure, imshow(log(1 + abs(fftshift(I_dft))),[]), ...
5 - title('FT of original image');
6
7 - [M, N] = size(I);
8 - dist = distmatrix(M, N);
9
10 - cutoff = 30; order = 2;
11 - H_but = 1 ./ (1 + (cutoff ./ dist) .^ (2 * order));
12
13 - H_but_hfe = a + (b .* H_but);
14 - figure, mesh(fftshift(H_but_hfe)), zlim([0 2]), ...
15 - title('Butterworth high-pass filter');
16
17 - DFT_filt_but = H_but_hfe .* I_dft;
18 - I4 = real(ifft2(DFT_filt_but));
19 - figure, imshow(I), title('Original Image');
20 - figure, imshow(log(1 + abs(fftshift(I_dft))),[]), ...
21 - title('FT of original image');
22 - figure, imshow(log(1 + abs(fftshift(DFT_filt_but))),[]), ...
23 - title('Filtered FT');
24 - figure, imshow(I4), title('Filtered Image');|

```

Code Fragment 3.4: Procedure 19-21 applied.

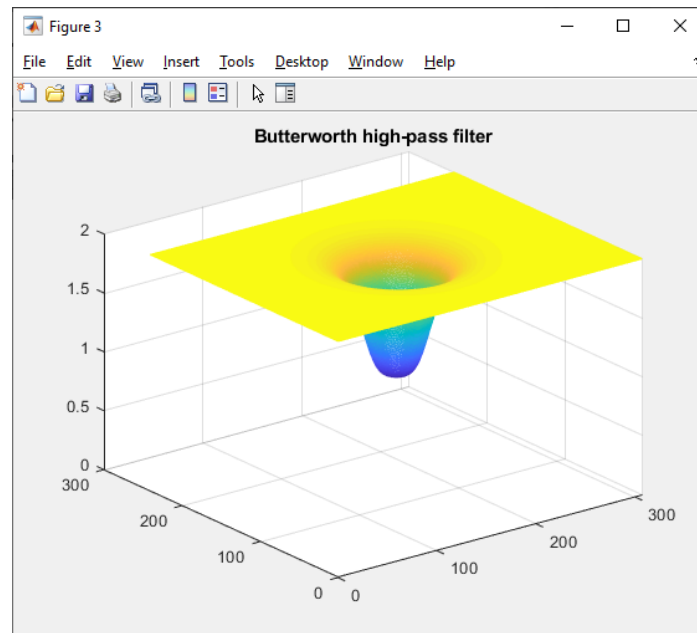


Figure 3.7: Outputs of Procedure [19-21],

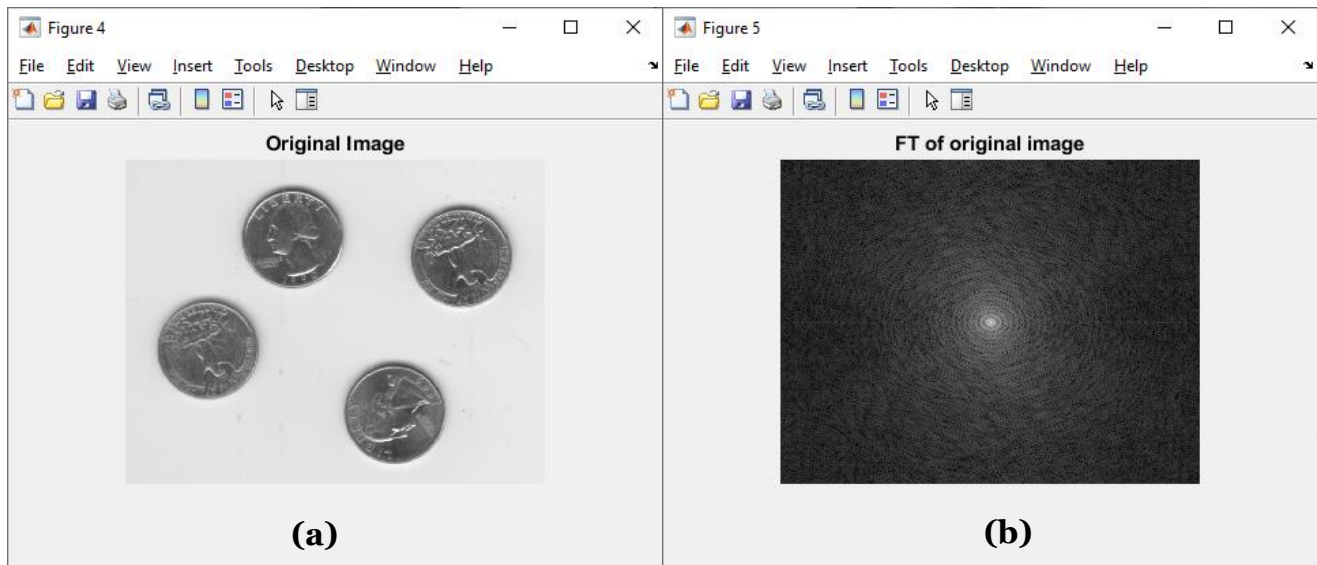


Figure 3.8: Outputs of Procedure [19-21] (Cont.)

(a) Original Image, (b) FT of Original Image

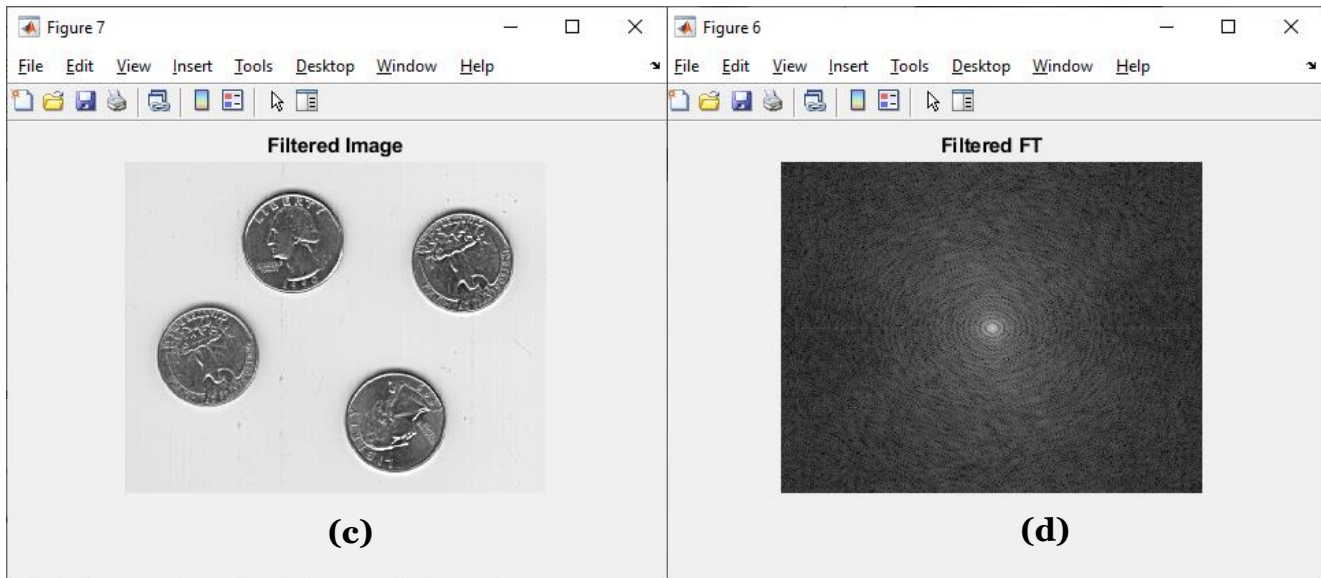


Figure 3.8: Outputs of Procedure [19-21] (Cont.)
(c) Filtered Image, (b) FT of Filtered Image

Question 11

How does the order parameter change the shape of the filter?

Answer 11

Having different values will produces images with different sharpness values. ■

Question 12

For large order values (such as 10), the Butterworth begins to take the shape of what other high-pass filters take?

Answer 12

Increasing the value (e.g. 20) will reduces the sharpness. ■