```matlab
% Problem 4 - Image Deblurring by Frequency Domain Inverse Filter Design.
clear all;

% functionality of a particular functions.
% fft2() - It will return the two-dimensional fourier transform of a
% matrix using the fast fourier trtansform.
% fftshift() - It will rearrange a Fourier transform x by shifting the
% zero-frequency component to the center of the array.
% ifft2() - It will return the two-dimensional inverse fourier transform of a
% matrix using the fast fourier transform.
% mesh() - It will create a 3D surface that has a sloid edge colors and no face colors.

input_image = double(imread("text.tif")); % reading the original image. This image is 8
bits/pixel gray-scale

size_of_filter = 21; % We are simply defining the filter size and it is used further in
the below code.
standard_deviation = 1; % We are defining the value of standard deviation as per the
given problem.

[vector_col,vector_row] = meshgrid(-floor(size_of_filter/2):floor(size_of_filter/2), -
...
    floor(size_of_filter/2):floor(size_of_filter/2)); % It will return 2-D grid
coordinates based on the
% coordinates in vectors. This basically means that we are using the cropped image of
the central part, so that
% input image and the output image, bith has the same size.
gaussian_filter = exp(-(vector_col.^2 + vector_row.^2)/(2*standard_deviation^2))/
(2*pi*standard_deviation^2);
% Here, we are just
% normilizing the gaussian filter such that the sum of all the coefficients will be
equal to 1. This is given in
% the problem statement. We are just normalizing the gaussian filter.
blurring_filter = gaussian_filter./sum(gaussian_filter(:)); % created a blurring filter
which will be used
% to give blurring effect to an image.

blurred_image = conv2(input_image, blurring_filter, 'same'); % we applied 2D
convolution between the original
% input image and the gaussian filter to get a blurred image as a resulting image.
Hence, we have applied
% conv2() function on the input image.

% Frequency domain of the gaussian filter.
gaussian_filter_freq_domain = fft2(blurring_filter); % Here, we are applying 2D-DFT to
the spatial domain
% Gaussian filter to obtain frequency domain gaussian filter.  Hence, we have applied
fft2() function on the
% blurred image.
```

```
%Inverse Gaussian Filter
freq_domain_inverse_gaussian_filter = 1./gaussian_filter_freq_domain; % Here, we have↙
created a frequency
% domain inverse gaussian filter by simply taking reciprocal value of each and every↙
pixel value of frequency
% domain Gaussian filter coefficients. The '.'indicates that we have taken all the↙
values from the image matrix.
deblurring_spatial_domain_inverse_filter = real(ifft2↙
(freq_domain_inverse_gaussian_filter)); % Here, as per the
% problem statement we have applied the inverse 2D-DFT to generate spatial domain↙
inverse gaussian filter. For
% this we have implemented ifft2() method.

% convolution between the blurred image and the deblurred image obtained from spatial↙
domain inverse filter.
deblurred_image = conv2(blurred_image, deblurring_spatial_domain_inverse_filter,↙
'same'); % we applied 2D
% convolution between the spatial domain gaussian filtered image and the spatial domain↙
inverse gaussian filter
% to get a spatial domain blurred image as a resulting image. Hence, we have applied↙
conv2() function on the
% input image.

% Intensity transformation and DC shift for corresponding images.
original_image_fft = fft2(input_image); % Here, we are applying 2D-DFT to the input↙
image to obtain
% fourier transformed image.  Hence, we have applied fft2() function on the input↙
image.
original_image_intensity_transform = log(1 + abs(original_image_fft)); % Apply log↙
transform on the 2ddft
% image and make it log shifted image.
original_image_frequency = fftshift(original_image_intensity_transform./ ...
    max(original_image_intensity_transform(:))); % DC shift for the corresponding↙
images.

blurred_image_fft = fft2(blurred_image); % Here, we are applying 2D-DFT to the blurred↙
image to obtain
% fourier transformed image.  Hence, we have applied fft2() function on the blurred↙
image.
blurred_image_intensity_transform = log(1 + abs(blurred_image_fft)); % Apply log↙
transform on the 2ddft
% image and make it log shifted image.
blurred_image_frequency = fftshift(blurred_image_intensity_transform./max↙
(blurred_image_intensity_transform(:)));
% DC shift for the corresponding images.

deblurred_image_fft = fft2(deblurred_image); % Here, we are applying 2D-DFT to the↙
deblurred image to obtain
% fourier transformed image.  Hence, we have applied fft2() function on the deblurred↙
image.
```

```
deblurred_image_intensity_transform = log(1 + abs(deblurred_image_fft)); % Apply log↙
transform on the 2ddft
% image and make it log shifted image.
deblurred_image_frequency = fftshift(deblurred_image_intensity_transform./ ...
    max(deblurred_image_intensity_transform(:))); % DC shift for the corresponding↙
images.


% Plot for images and figures

figure % figure creates figure graphics objects. figure objects are the individual↙
windows on the screen
% in which MATLAB displays graphical output.
subplot(2,3,1); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=2, n=3, p=1.
imshow(uint8(input_image)); % It will display the gray-scale image in the figure and
% it will convert each and every pixel value of the input image into the range of 0 to↙
255.
title('Original Text Image'); % It will add the specified title for the current plot.

subplot(2,3,2); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=2, n=3, p=2.
imshow(uint8(blurred_image)); % It will display the gray-scale image in the figure and
% it will convert each and every pixel value of the input image into the range of 0 to↙
255.
title('Blurred Text Image'); % It will add the specified title for the current plot.

subplot(2,3,3); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=2, n=p=3.
imshow(uint8(deblurred_image)); % It will display the gray-scale image in the figure↙
and
% it will convert each and every pixel value of the input image into the range of 0 to↙
255.
title('Deblurred Text Image'); % It will add the specified title for the current plot.

subplot(2,3,4); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=2, n=3, p=4.
imshow(original_image_frequency,[]); % It will display the gray-scale image in the↙
figure and
% it will convert each and every pixel value of the input image into the range of 0 to↙
255.
title('Original Image Frequency'); % It will add the specified title for the current↙
plot.

subplot(2,3,5); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
```

```
% position specified by p. Here, m=2, n=3, p=5.
imshow(blurred_image_frequency,[]); % It will display the gray-scale image in the↙
figure and
% it will convert each and every pixel value of the input image into the range of 0 to↙
255.
title('Blurred Image Frequency'); % It will add the specified title for the current↙
plot.

subplot(2,3,6); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=2, n=3, p=6.
imshow(deblurred_image_frequency,[]); % It will display the gray-scale image in the↙
figure and
% it will convert each and every pixel value of the input image into the range of 0 to↙
255.
title('Deblurred Image Frequency'); % It will add the specified title for the current↙
plot.


%%%%%
figure % figure creates figure graphics objects. figure objects are the individual↙
windows on the screen
% in which MATLAB displays graphical output.
subplot(2,2,1); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=1.
mesh(blurring_filter); % It will create a 3D surface that has a sloid edge colors and↙
no face colors.
title('Spatial domain Gaussian Blur Filter'); % It will add the specified title for the↙
current plot.

subplot(2,2,2); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=p=2.
mesh(abs(fftshift(fft2(blurring_filter)))); % It will create a 3D surface that has a↙
sloid edge colors and no
% face colors.
title('Magnitude of Fourier Domain GBF'); % It will add the specified title for the↙
current plot.


subplot(2,2,3); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=3.
mesh(deblurring_spatial_domain_inverse_filter); % It will create a 3D surface that has↙
a sloid edge colors and no
% face colors.
title('Spatial domain Gaussian Deblur Filter'); % It will add the specified title for↙
the current plot.
```

```
subplot(2,2,4); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=4.
mesh(abs(fftshift(freq_domain_inverse_gaussian_filter))); % It will create a 3D surface↙
that has a sloid edge colors and
% no face colors.
title('Magnitude of Fourier Domain GDF'); % It will add the specified title for the↙
current plot.
```