**LAPORAN PRAKTIKUM PENGOLAHAN CITRA DIGITAL**

**13. LOW-PASS FILTERS IN THE FREQUENCY**

**DOMAIN**



**Disusun oleh :**

**Nama : Andhika Rizky Cahya Putra**

**NPM : 2226250071**

**Kelas : IF4A**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS ILMU KOMPUTER DAN REKAYASA**

**UNIVERSITAS MULTI DATA PALEMBANG**

**2024**

**TUTORIAL : LOW-PASS FILTERS IN THE FREQUENCY**

**DOMAIN**

**Goal**

The goal of this tutorial is to demonstrate how to implement low-pass filters in the

frequency domain.

**Objectives**

* Learn how to generate and apply an ideal low-pass filter.
* Learn how to generate and apply a Gaussian low-pass filter.
* Learn how to generate and apply a Butterworth low-pass filter.

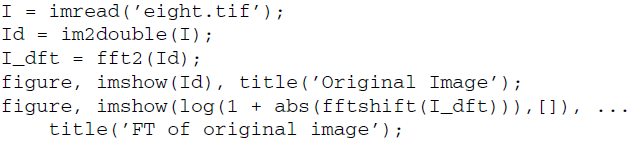
**What You Will Need**

* distmatrix.m
* fddemo.m

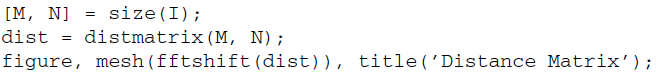
**Procedure**

As we have learned in Tutorial 12, the distmatrix function returns a two dimensional array, which should be of the same size as the image being processed. The values in the array represent the distance from each pixel to the center of the image. To begin, we will use this matrix to generate an ideal low-pass filter.

1. Load the eight image, generate a FT, and display it.



1. Generate a distance matrix with size equal to the input image.



**Question 1** Verify that the size of the distance matrix is in fact equal to the size of the image.

|  |
| --- |
|  |

**Question 2** What happens if we display the distance matrix without shifting?

|  |
| --- |
|  |

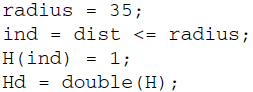
**Ideal LPF**

To create an ideal low-pass filter, we will start out with a matrix of all zeros and then set specific values to 1 that will represent all frequencies for which we will allow to pass through. Since we are defining an ideal filter, we can simply define the radius of the filter and then set any values within that radius to 1, while all others remain zero.

1. Create initial filter with all values of zero.



1. Create the ideal filter.



**Question 3** Explain how the previous code sets all values within a given radius to 1.

|  |
| --- |
|  |

We can visualize the filter’s frequency response by displaying it as an image.

1. Display the filter’s frequency response.

****

To apply the filter to the image, we simply multiply each value of the filter by its corresponding frequency value in the FT image.

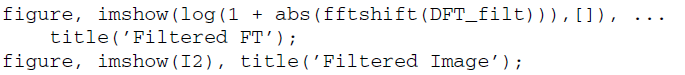
1. Apply the filter to the FT image.



**Question 4** Why do we take only the real values when converting the FT of the filtered image back to the spatial domain?

|  |
| --- |
|  |

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



Question 5 How does the filtered image compare to the original image? Can you see any noticeable artifacts?

|  |
| --- |
|  |

To see how the choice of radius affects the filtered image, we can use the frequency domain demo, fddemo (developed by Jeremy Jacob and available at the book’s companion web site).

1. Load the frequency-domain demo.



The default filter is the ideal low pass. To modify the cutoff value, select the magenta circle within the filter profile and drag it to a desired radius. The value of the radius is displayed below the filter profile.

**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?

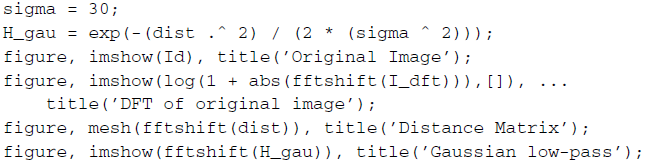
|  |
| --- |
|  |

1. Close the demo.

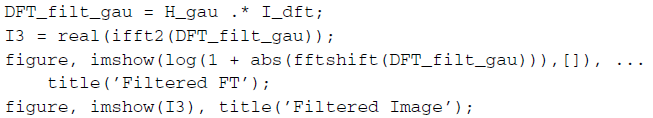
**Gaussian LPF**

The Gaussian low-pass filter is usually specified by providing a value for the standard deviation σ. We can use the distance matrix previously generated to create a Gaussian filter.

1. Create a Gaussian low-pass filter with σ = 30.



1. Filter the FT image with the Gaussian low-pass filter and display the filtered image.



**Question 7** Compare the output images between the ideal filter and the Gaussian filter. What are their similarities? What are their differences?

|  |
| --- |
|  |

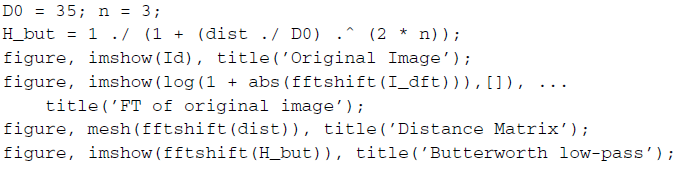
**Question 8** Start the frequency-domain demo (fddemo) once again, and this time select the Gaussian low-pass filter. Experiment with different values o sigma (standard deviation). How does this value affect the output image?

|  |
| --- |
|  |

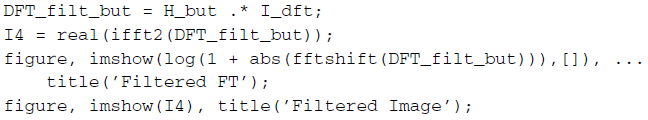
**Butterworth LPF**

The Butterworth low-pass filter is usually specified by providing two parameters: the order of the filter, n, and the cutoff value, D0. In our implementation of the ideal low-pass filter, earlier in this tutorial, we set the cutoff value to 35. For comparison purposes, we will use the same value for our Butterworth filter.

1. Generate a third-order Butterworth filter, where the cutoff value is 35.



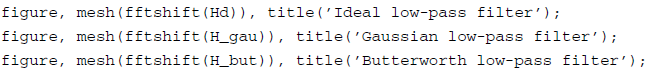
1. Filter the image with the Butterworth low-pass filter and display the resulting image.



**Question 9** Compare the ideal-filtered FT image and the Butterworth-filtered

FT image.

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



**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?

|  |
| --- |
|  |

**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?

|  |
| --- |
|  |