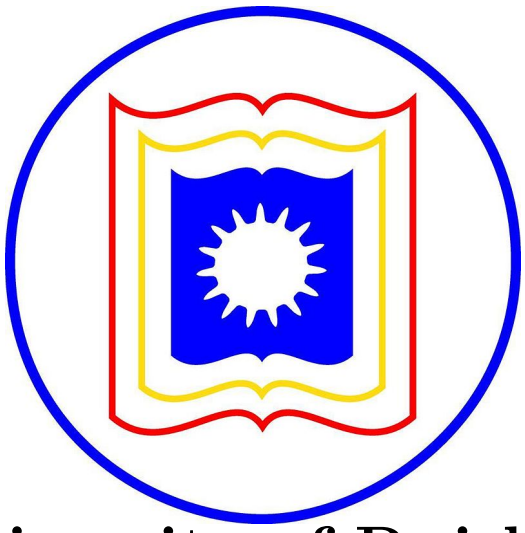


University of Rajshahi

Department of Computer Science & Engineering

CSE4182 - Digital Image Processing Lab

Submitted by : MD. Mehedi Hasan
Roll ID : 1810976114
Instructor : Sangeeta Biswas, Associate Professor
Date : August 14, 2022

Abstract

This is a notebook for Digital Image Processing Lab. This will contain all the assignments and their explanations.

Contents

Chapter 1

Assignment 11
Fourier Transformation for Image Processing

1.1 Introduction

Fourier Transform is used to analyze the frequency characteristics of various filters. For images, 2D Discrete Fourier Transform (DFT) is used to find the frequency domain as Digital images are not continuous.Ordinary DFT is slow so we choose A fast algorithm called Fast Fourier Transform (FFT) is used for calculation of DFT.

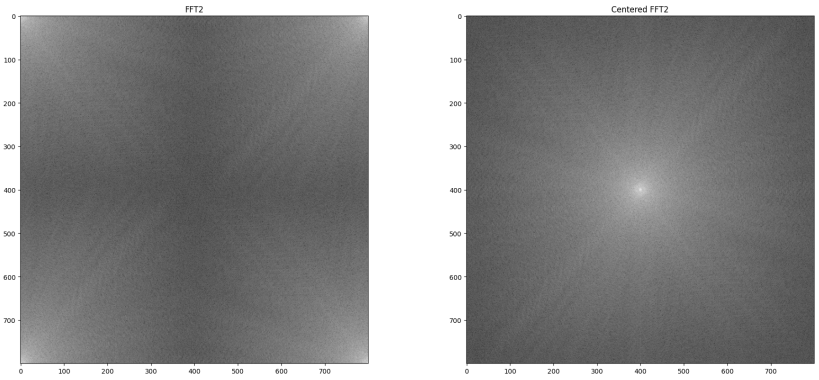
1.2 Applying Fourier Transform in Image Processing

Our main aim is to use Fourier transform to reduce the computational complexity for convolution. There are lots and lots of applications in computing, physics, sound mixing etc.

We will be following these steps :
1) Fast Fourier Transform to transform image to frequency domain.
2) Moving the origin to centre for better visualisation and understanding.
3) Apply filters to filter out frequencies.
4) Reversing the operation did in step 2.
5) Inverse transform using Inverse Fast Fourier Transformation to get image back from the frequency domain.

(a) RGB (b) Gray

Figure 1.1: Sample Image



(a) FFT (b) Centered FFT

Figure 1.2: Applying FFT on Image

1.2.1 Sample Image

First we have to read the image. Here we are using matplotlib.pyplot package to read the image. Then image is converted in grey scale format using CV2 package

```
rgbImg = plt.imread('dahlia.jpg')
gray =  cv2.cvtColor(rgbImg,cv2.COLOR_RGB2GRAY)
```

1.2.2 Applying FFT

Now we are going to apply FFT from numpy package. This means that we are taking the discrete image pixel values and are transforming it to a frequency domain and visualising it. The whole information is reserved but transformed to another domain.Here we can see that there are white spots over corners which represents the low frequency components.

```
gray =  cv2.cvtColor(rgbImg,cv2.COLOR_RGB2GRAY)
ftImg = np.fft.fft2(gray)
centeredfti_img = np.fft.fftshift(ftImg)
magnitude_spectrum = 100 * np.log(np.abs(ftImg))
centered_magnitude_spectrum = 100 * np.log(np.abs(centeredfti_img))
```

Then we take the origin from corner to centre. This results in such a way that the centre part contains the low frequency components where as other contains high frequency

1.2.3 Applying Filters

Here we will use four different types of filter and will try to analyse the different outputs.

Filter 1

In this low pass filter only the center portion has high values which diminishes going beyond center.

```
r , c = gray.shape
m , n = r//2 , c//2
black_img = np.zeros((r,c),dtype=np.uint8)
filter1 = cv2.circle(black_img.copy(),(m,n),25,(255,255,255),-1)
```

As we have already seen the centre contains low frequency components. Thus it removes high frequency component when we multiply and keep low frequency.

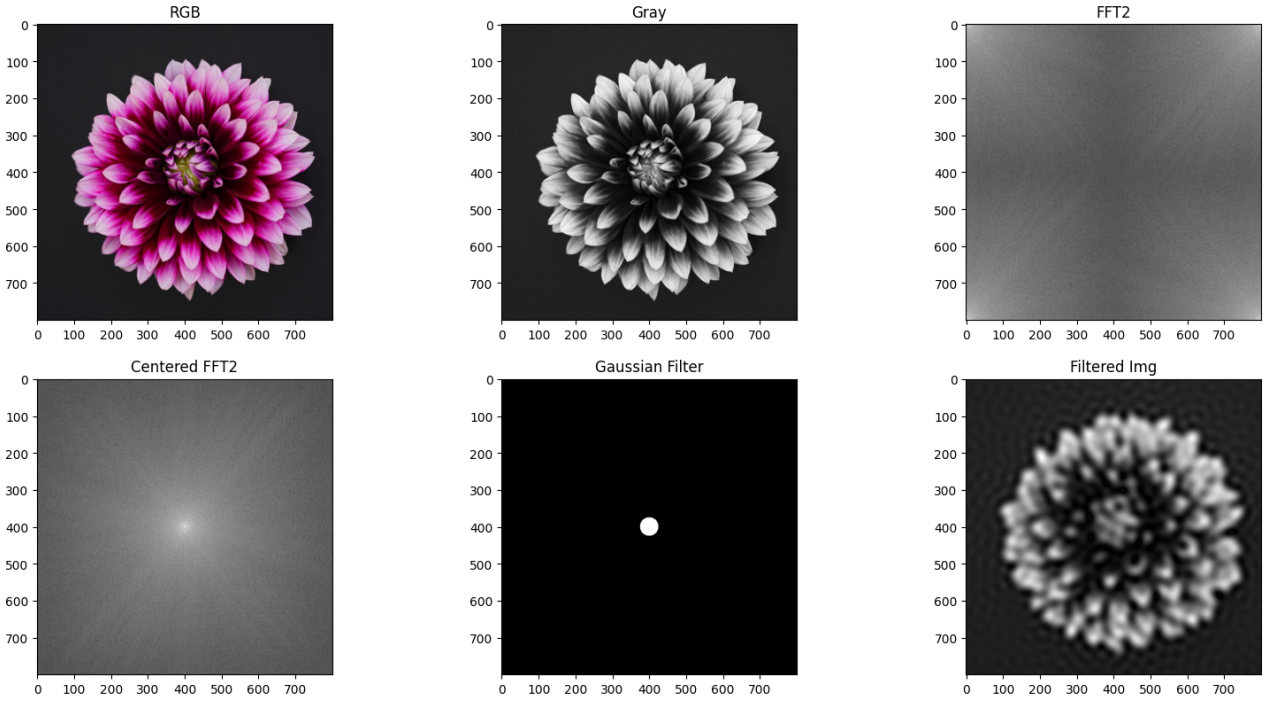


Figure 1.3: Low Pass Filter -Center

Filter 2

In this High pass filter only the center portion has low values which diminishes going beyond center.

```
r , c = gray.shape
m , n = r//2 , c//2
white_img = np.ones((r,c),dtype=np.uint8)
filter2 = cv2.circle(white_img.copy(),(m,n),25,(0,0,0),-1)
```

As we have already seen the centre contains low frequency components. Thus it removes low frequency component when we multiply and keep high frequency.

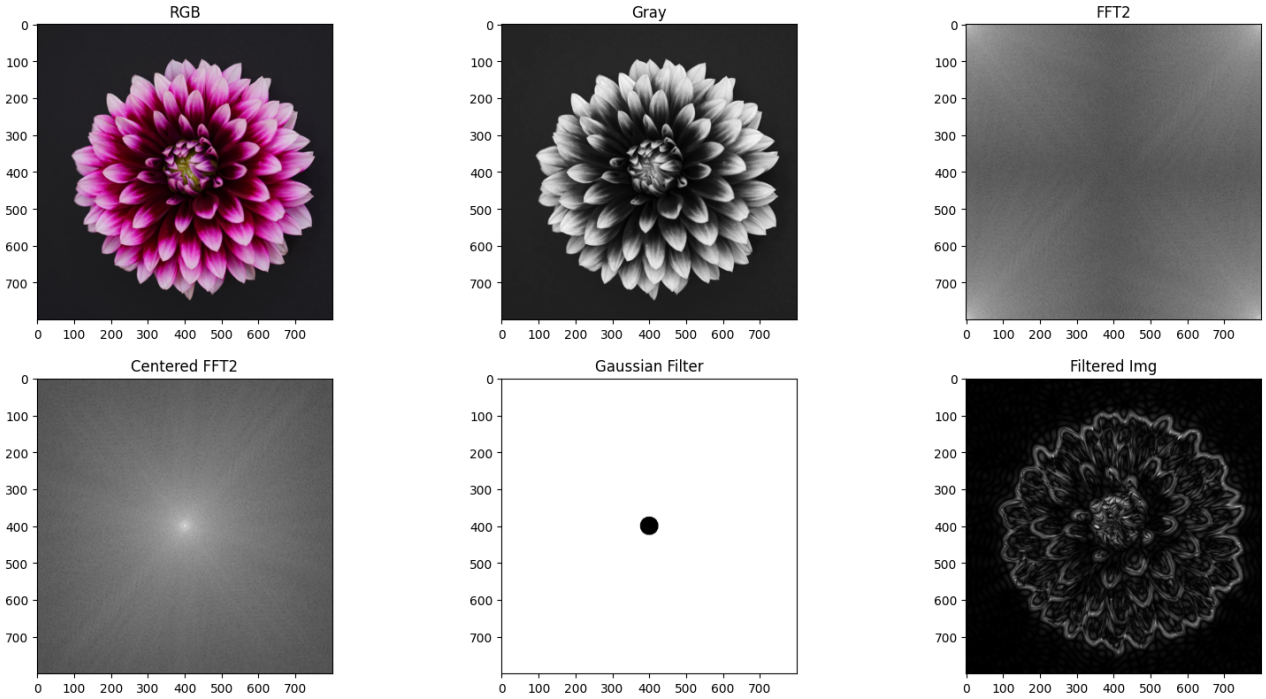


Figure 1.4: High Pass Filter -Center

Filter 3

In low pass filter only the vertical line in the center portion has high values.

```
r , c = gray.shape
m , n = r//2 , c//2
black_img = np.zeros((r,c),dtype=np.uint8)
filter3 = cv2.line(black_img.copy(),(m,0),(m,c),(255,255,255),9)
```

As we have already seen the line contains low frequency components. Thus it removes high frequency component when we multiply and keep low frequency.

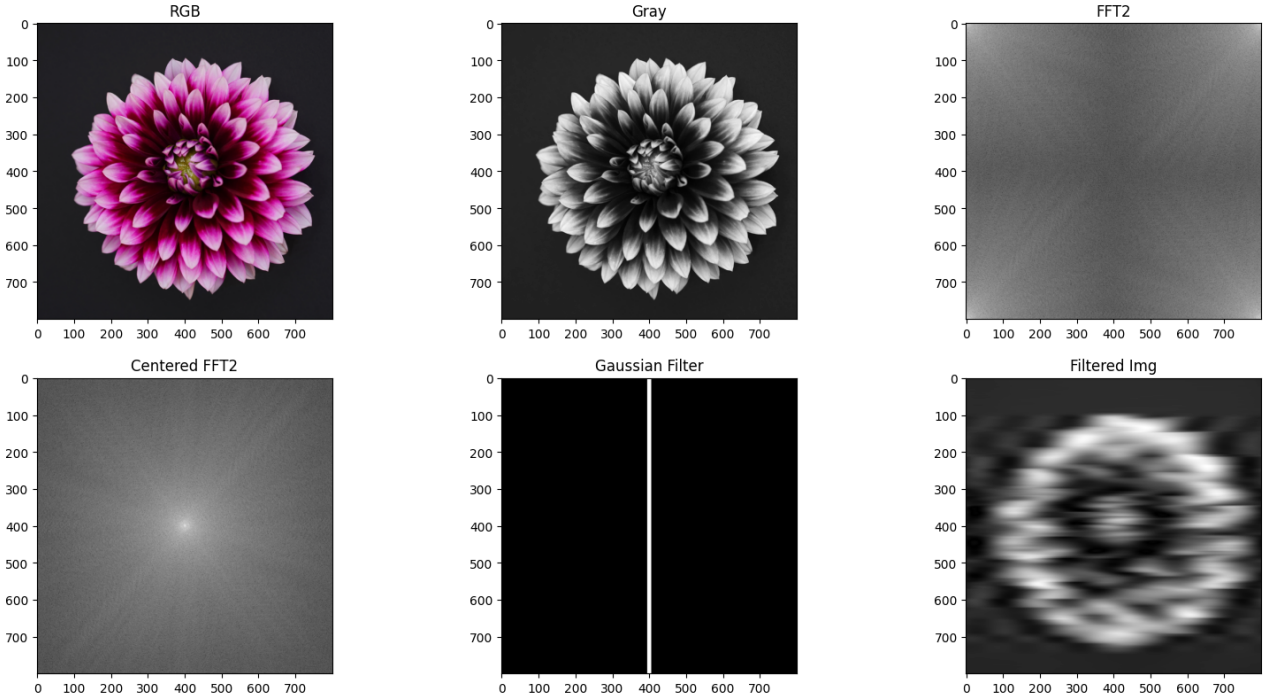


Figure 1.5: Low Pass Filter -Line

Filter 4

In this High pass filter only the horizontal line in the center portion has low values.

```
r , c = gray.shape
m , n = r//2 , c//2
white_img = np.ones((r,c),dtype=np.uint8)
filter4 = cv2.line(white_img.copy(),(0,n),(r,n),(0,0,0),9)
```

As we have already seen the line contains low frequency components. Thus it removes low frequency component when we multiply and keep high frequency.

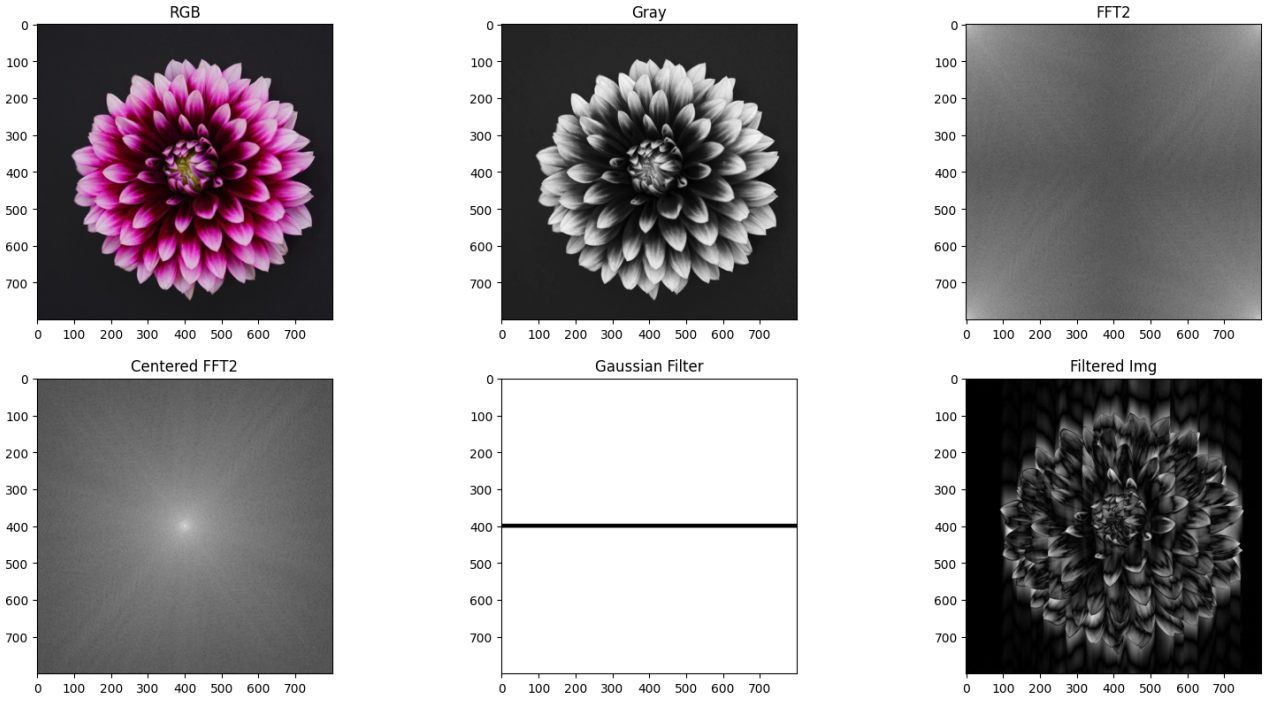


Figure 1.6: High Pass Filter -Line

1.2.4 Inverse FFT

Final step, We reverse back the image from the frequency domain by using inverse Fourier transformation.

1.3 Conclusion

The Fast Fourier Transform (FFT) is commonly used to transform an image between the spatial and frequency domain. Unlike other domains such as Hough and Radon, the FFT method preserves all original data. Plus, FFT fully transforms images into the frequency domain, unlike time-frequency or wavelet transforms.