# LAB # 11: Image Analysis in Frequency Domain

**Lab Objective:**

The objective of this lab is to understand Fourier Transform, apply it on images and understand the results.

**Lab Description:**

**Fourier series** tells us that any function can be represented as a sum of sines/cosines of different frequencies multiplied by a different coefficient. Similarly, non periodic functions can also be represented as the integral of sines/cosines multiplied by weighing function.

****The Fourier transform of function f(x) is given by,

Fourier transform pair for a function f (x, y) of two variables;

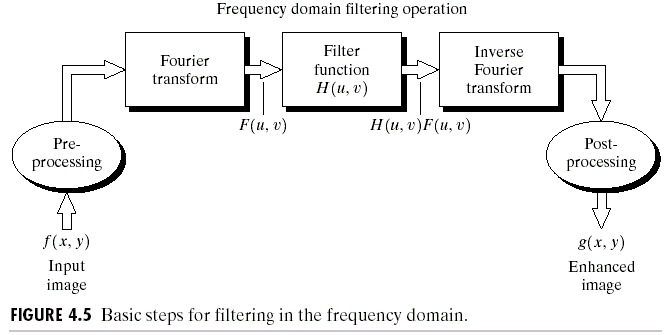
****

****

The filtering in frequency domain consists of following steps:

1. Compute F (u, v) the DFT of the image
2. Multiply F (u, v) by a filter function H (u, v)

Compute the inverse DFT of the result

****

## Some Useful Commands:

1. To obtain the Fourier Transform of an image: **my\_transformed\_image = numpy.fft.fft2(my\_image)**
2. To obtain the Inverse Fourier Transform of an image: **my\_inverse\_image = numpy.fft.ifft2(my\_image)**
3. To shift the DC component of a Fourier Transformed Image to center: **my\_shifted\_image = numpy.fft.fftshift(my\_transformed\_image)**
4. To shift the DC component back to the top left corner: **my\_inverse\_shifted\_image = np.fft.ifftshift(my\_shifted\_image)**
5. To calculate absolute of a value**: my\_absolute = numpy.abs(my\_image)**
6. To calculate the exponential of an element: **my\_exponential = numpy.exp(my\_input)**
7. To use the value of pi: **numpy.pi**
8. To denote a complex number: simply put j after it e.g. -1j
9. To multiply two matrices point by point: **my\_result = numpy.multiply(my\_image, my\_filter)**
10. To take lop transformation of the image: **my\_result = numpy.log(my\_image)**
11. To use math function: **import math, x = math.sqrt(25)**

## Steps:

1. Take Fourier Transform of the image.
2. Shift its Dc component to center.

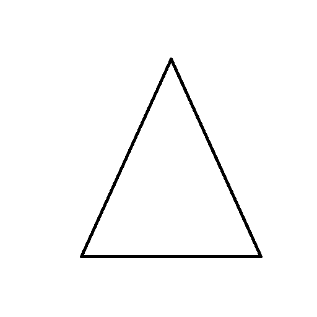
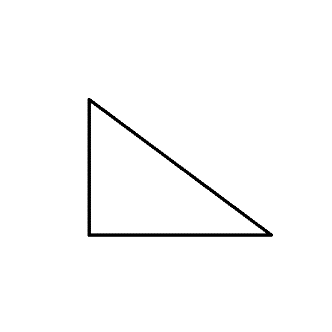
Perform following steps if you want visual representation of the fourier transform.

* + 1. Obtain absolute values of the output.
    2. Brighten up the image using log transformation.
    3. Normalize the image to obtain High Contrast (min, max) -> (0, 255).

1. Multiply it with the filter.
2. Take inverse shift to bring the dc component back to top left.
3. Take inverse Fourier transform to obtain frequency filtered image.

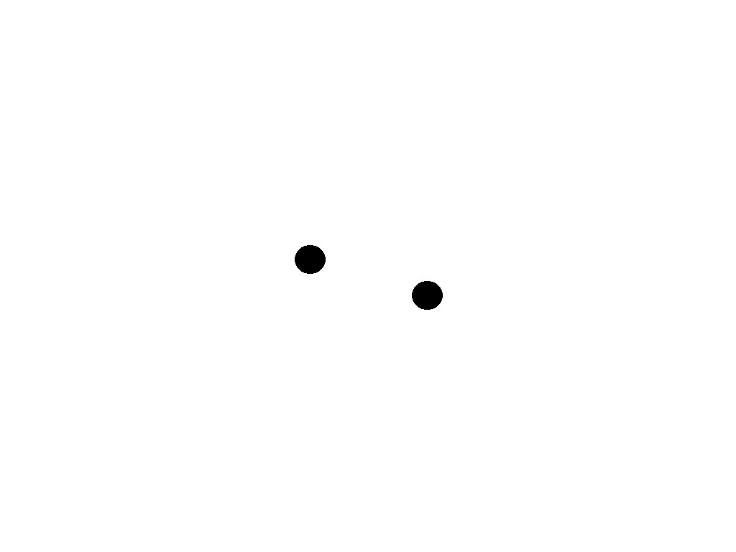
**Lab Task:**

**1:** Use the following figures to compute the FFT and display the results.

**Fig 1 Fig 2**

**2:**

**Fig 3 Fig 4**

* Use Fig 3 to compute the FFT with dc component centered.
* Now threshold Fig 4 by setting black color to zero and white color to one and multiply it with the FFT to filter out the noisy frequency component.
* Now obtain the inverse FFT to obtain noiseless image
* Enhance the contrast by setting 5th and 95th percentile of gray value to 0 and 255 respectively

**3:** Repeat Task 2 by making your own filter. Use Distance formula, if the distance between the pixel and center of image is less than 50 pixels, the pixel value of filter will be one otherwise Zero.

**HOME TASK:**

Repeat Task 2 but this time try to make the same filter in code.

**THINK!!**

1. Why most part of frequency output is black?
2. If a certain image has only one pixel value as shown below. What is its expected frequency output?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 |

1. In our case, why is it not possible to have a zero dc component?
2. In Task 2 why we still have the periodic noise on the edges but not inside?