Korea Advanced Institute of Science and Technology

School of Electrical Engineering

EE535 Digital Image Processing Spring 2018

Student’s Name: Dinh Vu

Student’s ID: 20184187

**Homework 4**

1. **Image Transform**
   1. ***DFT***
      1. *Theoretical background and programming strategy*

1-D DFT algorithm:

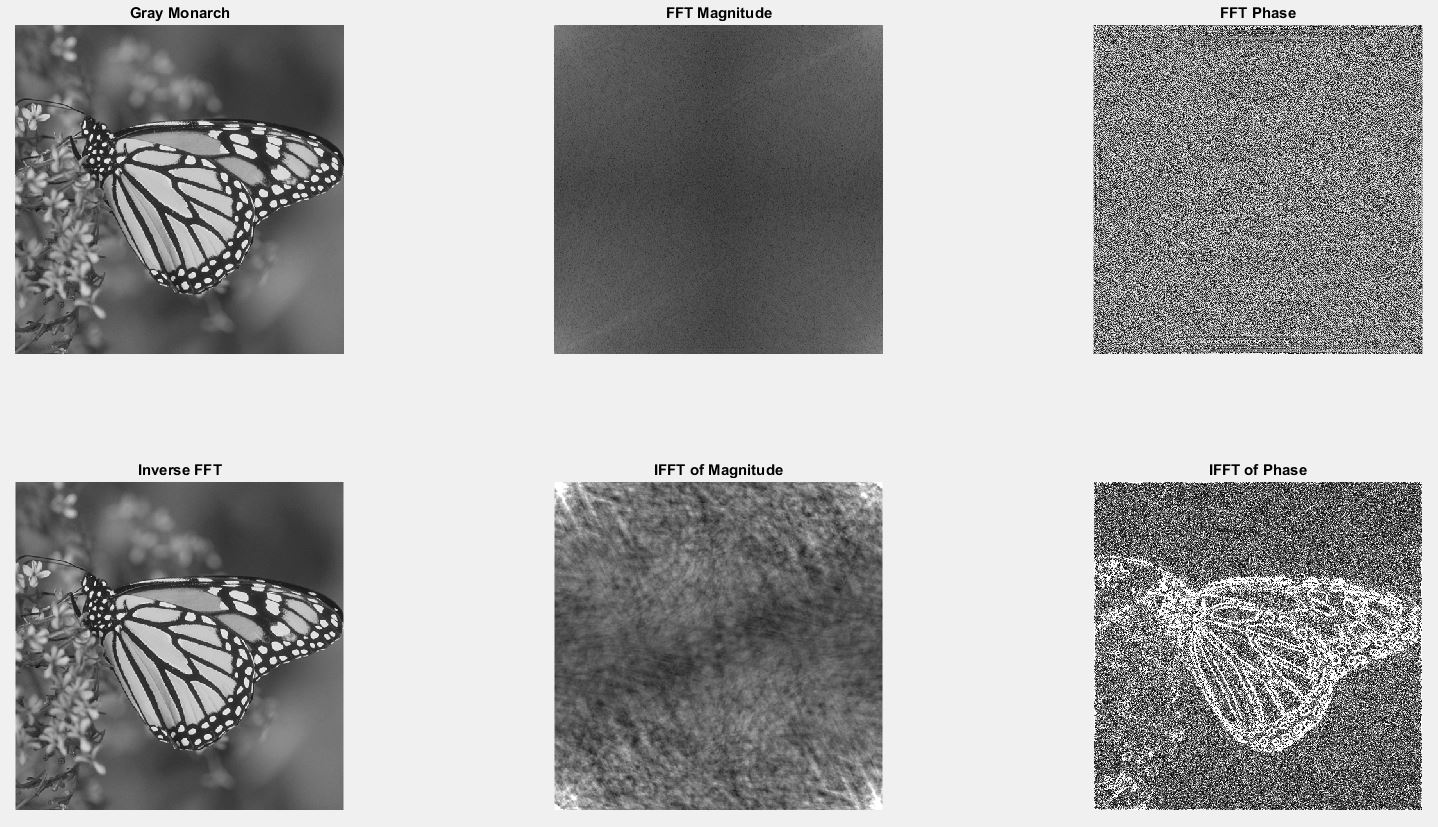
Where k = 0, 1, 2, …, N – 1.

The common term is .

The first term sum of DFT of the even index elements, the second term is about odd index. Both term can be easily transformed to vectorization to archive high performance in Matlab.

* + 1. *Result and Analysis result*

In the following Figure 1.1, it is noticeable to see that low frequency components are gathered at 4 corner of log magnitude of FFT image, the other regions have gray color so there are still unwanted frequency components.



**Figure 1.1.** Result of problem 1.1

The IFFT image look similar to the original image. It is very hard to see what information in the IFFT of magnitude image but the IFFT of phase image shows well-defined contours from the original image. It indicates that the phase component of DFT image carries more information than the magnitude of DFT image.

* 1. ***DCT***
     1. *Theoretical background and programming strategy*

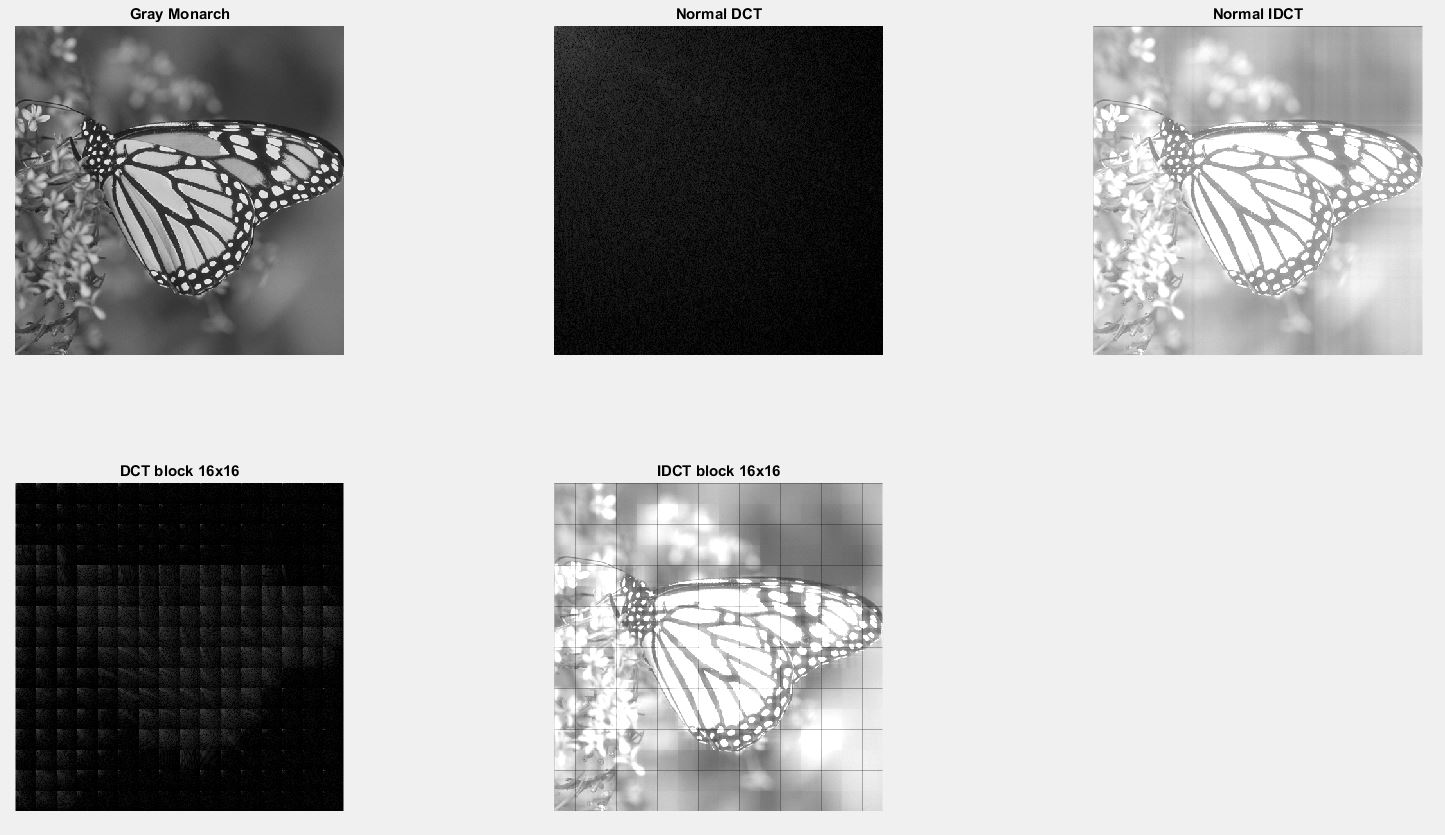
1-D DCT from DFT by using symmetric property:

Because image is 2 dimension and DCT is symmetric. So, we can use formula:

Where

* + 1. *Result and analysis result*

Figure 1.2 presents the result of problem 1.2. The normal DCT image isolates the low frequency components at the top-left corner where has gray color, while there is only black in the rest of image meaning that the unwanted frequency components are eliminated. The normal IDCT image is more blurred than the original image.



**Figure 1.2.** Result of problem 1.2

When DCT with block 16×16, the top-left corner of each block contains the low frequency components in the block and the others are removed. Therefore, the shape of object is still seen in the 16×16-DCT image. The 16×16-IDCT image is similar to the normal IDCT image except that there is boundaries of 16×16-blocks.

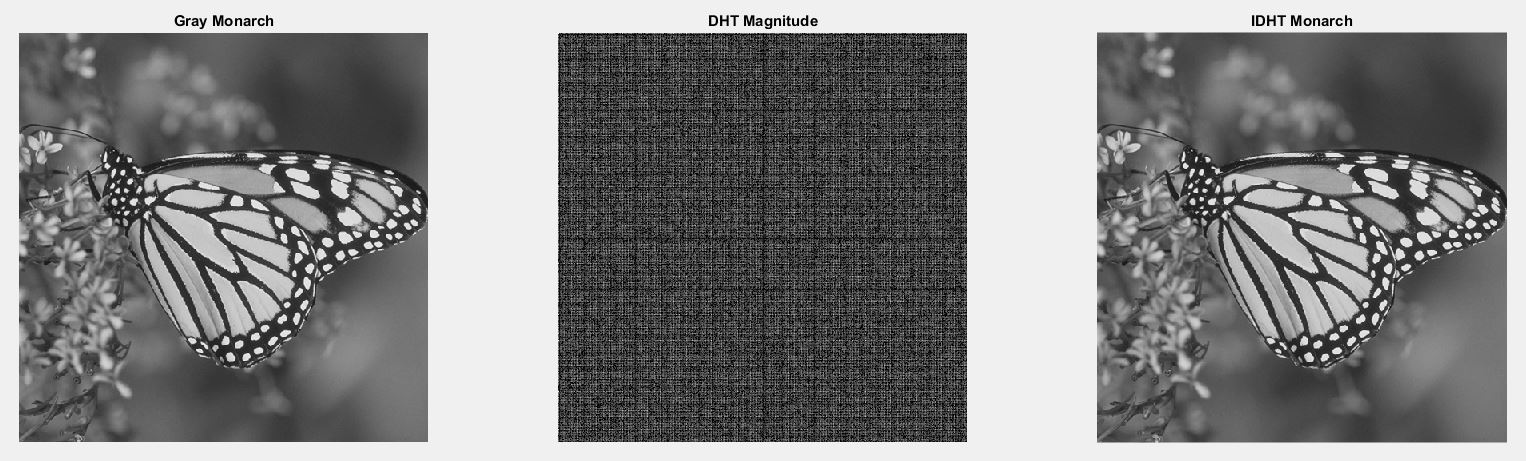
* 1. ***DHT***
     1. *Theoretical background and programming strategy*

First, the N×N kernel is created by for loop according to the property of Hadamard kernels:

Then forward DHT and inverse DHT by using the same kernel Hn, respectively:

V = HnU and U = HnV

* + 1. *Result and analysis result*



**Figure 1.3.** Result of problem 1.3

The IDHT image is the same the original image.

* 1. ***Compression Efficiency of DFT, DCT and DHT***

In the compression, DCT provide the best performance because DCT removed most of high frequency components and only keep the low frequency component at the top-left corner which mean except the top-left corner, the other pixels have zero value in DCT image. It will boost compression rate rapidly.

The second best transformation is DFT which includes the positive value at 4 corners while the values of other positions are very small. The performance of compression DHT image is worst due to the various value of each pixel in the transformed image.

* 1. ***Wavelet Transform***
     1. *Theoretical background and programming strategy*

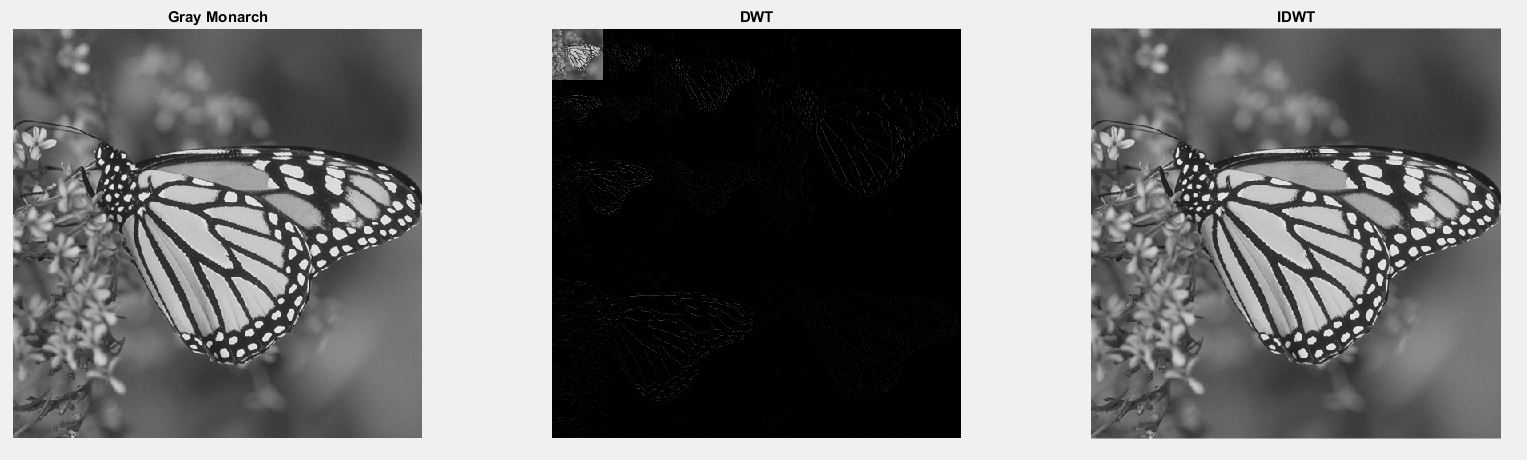


**Figure 1.4.** Three-level wavelet transform

To forward DWT, down sampling by 2 for the original image and haft of sample is average of sum of 2 row-order pixels continuously and half of sample is average of subtract between them. To inverse DWT, up sampling by 2.

* + 1. *Result and analysis result*

The result of problem 1.5 is shown in Figure 1.5 below. The DWT image contains resize the original image in the top-left corner, and the contour of object is expressed in difference block sample. To see more clearly, the DWT image should be zoomed in.



**Figure 1.5.** Result of problem 1.5

Because, the process includes filtering and sampling, there is not any loss information. Hence, the IDWT image is reconstructed to be similar to the original image.

1. **Image Enhancement**
   1. ***Point operations***
      1. *Theoretical background and programming strategy*

Histogram equalization:

Where:

* pdf is Probability Distribution Function.
* cdf is Cumulative Distribution Function.

Histogram modification techniques:

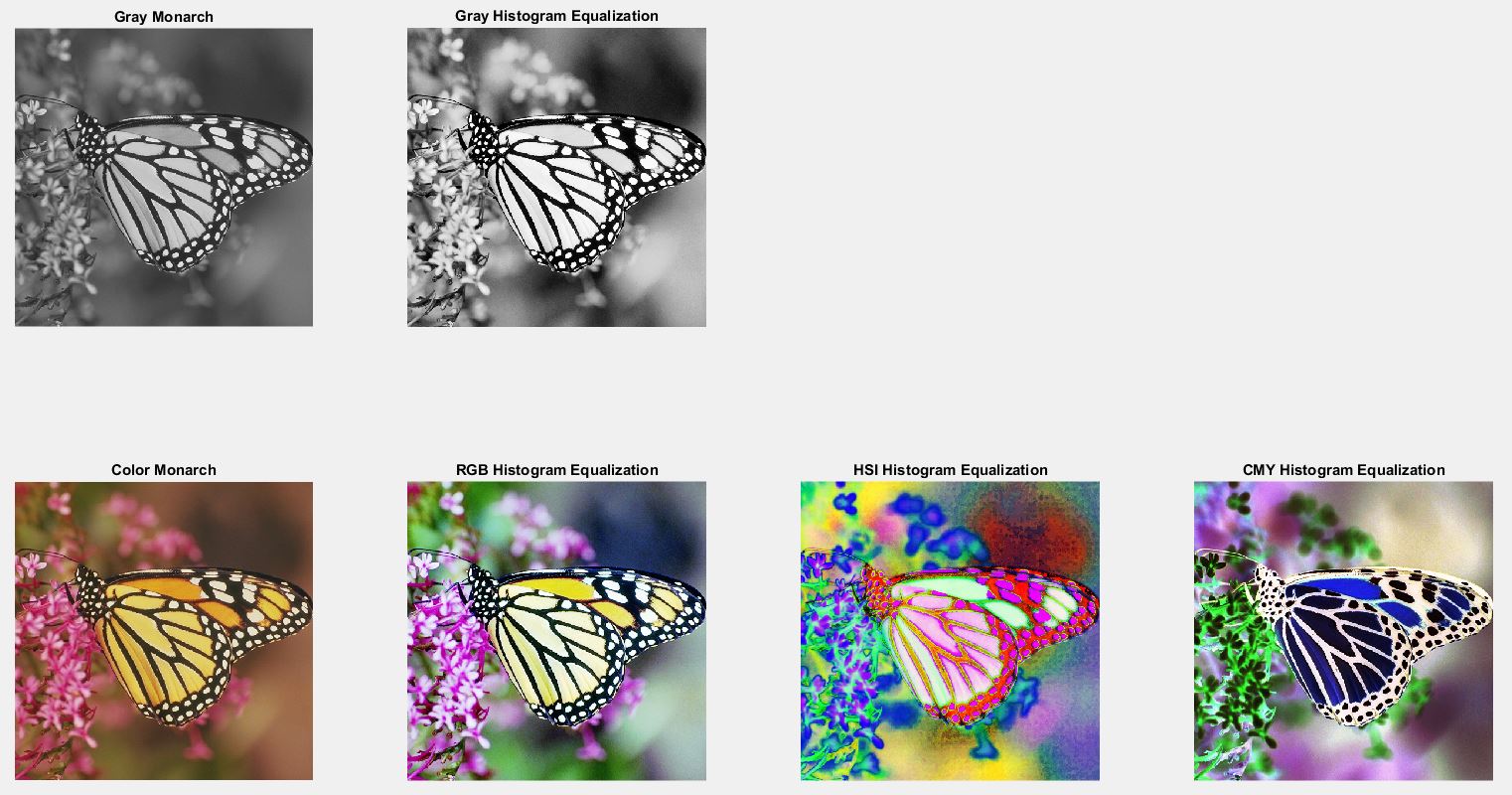
With exponential distribution:

First, pdf values must be got by counting individual value then divide to the total number of pixels in the image. After, calculating cdf by accumulation sum of pdf. Finally, mapping the cdf and the value of pixel in the original image by (2.1) and (2.2) for problem 2.1a and 2.1b, respectively.

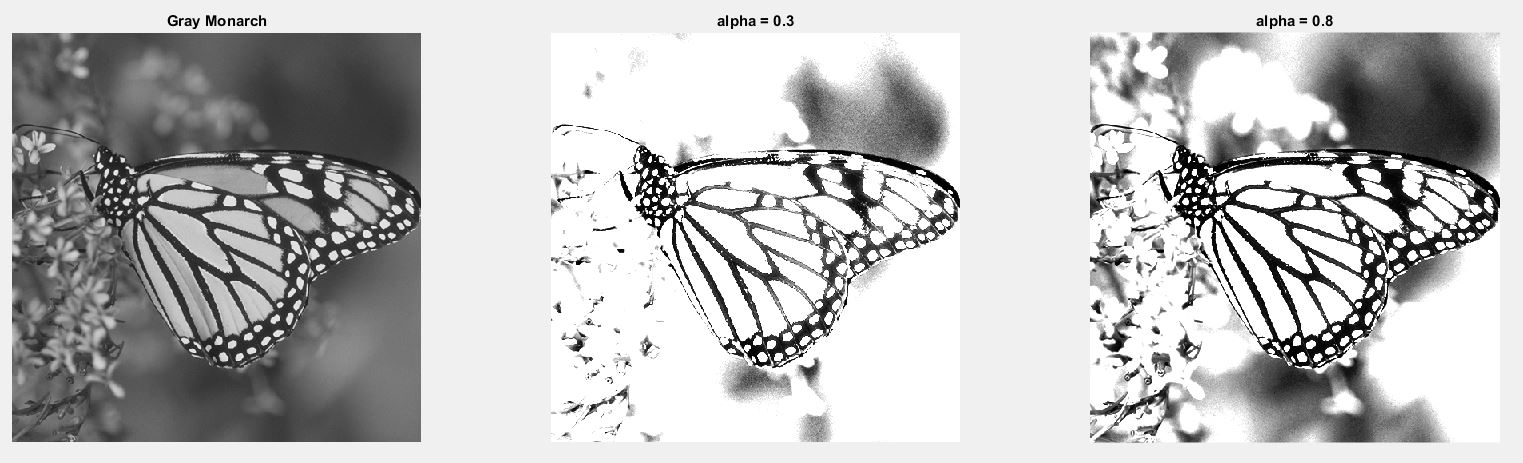
If the original image is RGB or HSI or CMY, it will be separated to 3 component image. After applying histogram equalization to each of them, they will be sum up.

* + 1. *Result and analysis result*

The results of problem 2.1 are displayed in Figure 2.1.



(a)



(b)

**Figure 2.1.** Result of problem 2.1

After histogram equalization for Gray Monarch and Color Monarch, the output images look brighter. However, in HSI and CMY, the color change so different to the original image cause by intensity equalization. The butterfly become pink and blue and the flowers is purple and green in HSI and CMY, respectively.

With difference values of alpha, histogram modification makes Gray Monarch brighter with difference ratio. If alpha increases, the intensity after histogram modification goes up, so the butterfly can be see more detail. In other word, if alpha go down, the intensity after histogram modification decreases, hence the butterfly is more faded. This is corresponding to (2.2).

* 1. ***Spatial Domain Filtering***
     1. *Theoretical background and programming strategy*
        + Gamma correction:
        + Mean filtering:

Where , W is size of window.

* Median filtering:

Zero padding to the original image in each direction by N/2. After that, move the N×N-window to cover all padding image. Then, read the intensity of each pixel in the window to an array. Sort by ascending order to this array and choose the value of the middle element.



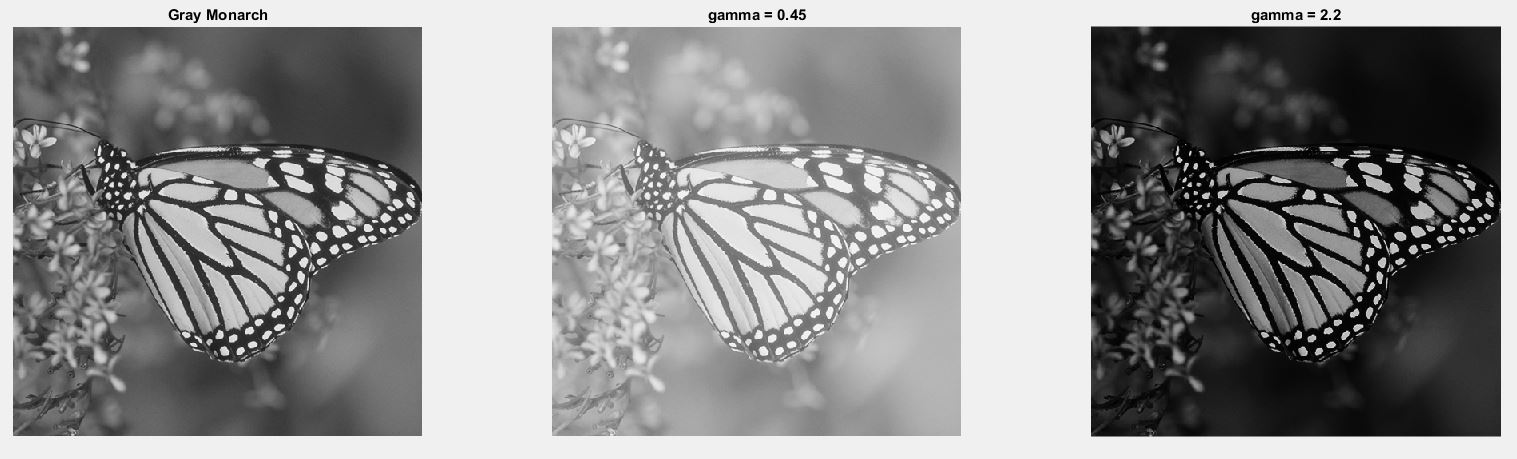
**Figure 2.2.** Median filtering with 3×3-window

* Directional smoothing:

Chose 4 directions: 0o, 45o, 90o, 135o. Read all value of pixel inside window follow each direction then take average value.

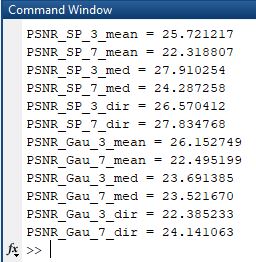
* + 1. *Result and analysis result*

The result of problem 2.2 is shown in Figure 2.3 below. Look at Figure 2.3a, if gamma is greater than 1 and increases, the intensity rises speedily, the result image is darker. If gamma is less than 1 and decrease, the intensity falls quickly and the result image is lighter.



(a)





(b)

**Figure 2.3.** Result of problem 2.2

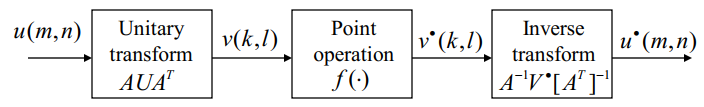
**Table 2.1.** PSNR values

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PSNR** | **Kernel 3** | | **Kernel 7** | |
| **Salt & pepper noise** | **Gaussian noise** | **Salt & pepper noise** | **Gaussian noise** |
| **Mean filtering** | 25.72 dB | 26.15 dB | 22.32 dB | 22.49 dB |
| **Median filtering** | 27.91 dB | 23.69 dB | 24.29 dB | 23.52 dB |
| **Directional smoothing** | 26.57 dB | 22.38 dB | 27.83 dB | 24.14 dB |

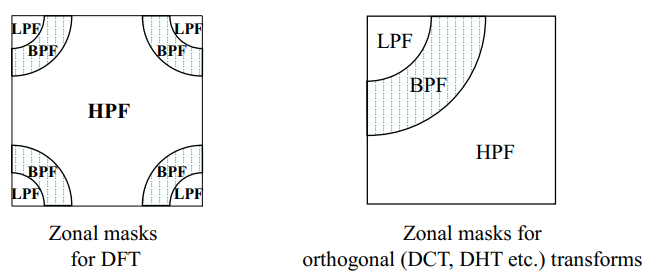
For kernel 3×3, median filter works best with salt & pepper noise while mean filter removes largest amount of Gaussian noise. Directional smoothing does well in salt & pepper noise.

For kernel 7×7, directional smoothing is the best choice in both types of noise. When size of kernel increases, the performance of all three techniques decreases in eliminating Gaussian noise.

* 1. ***Transform Domain Filtering***
     1. *Theoretical background and programming strategy*

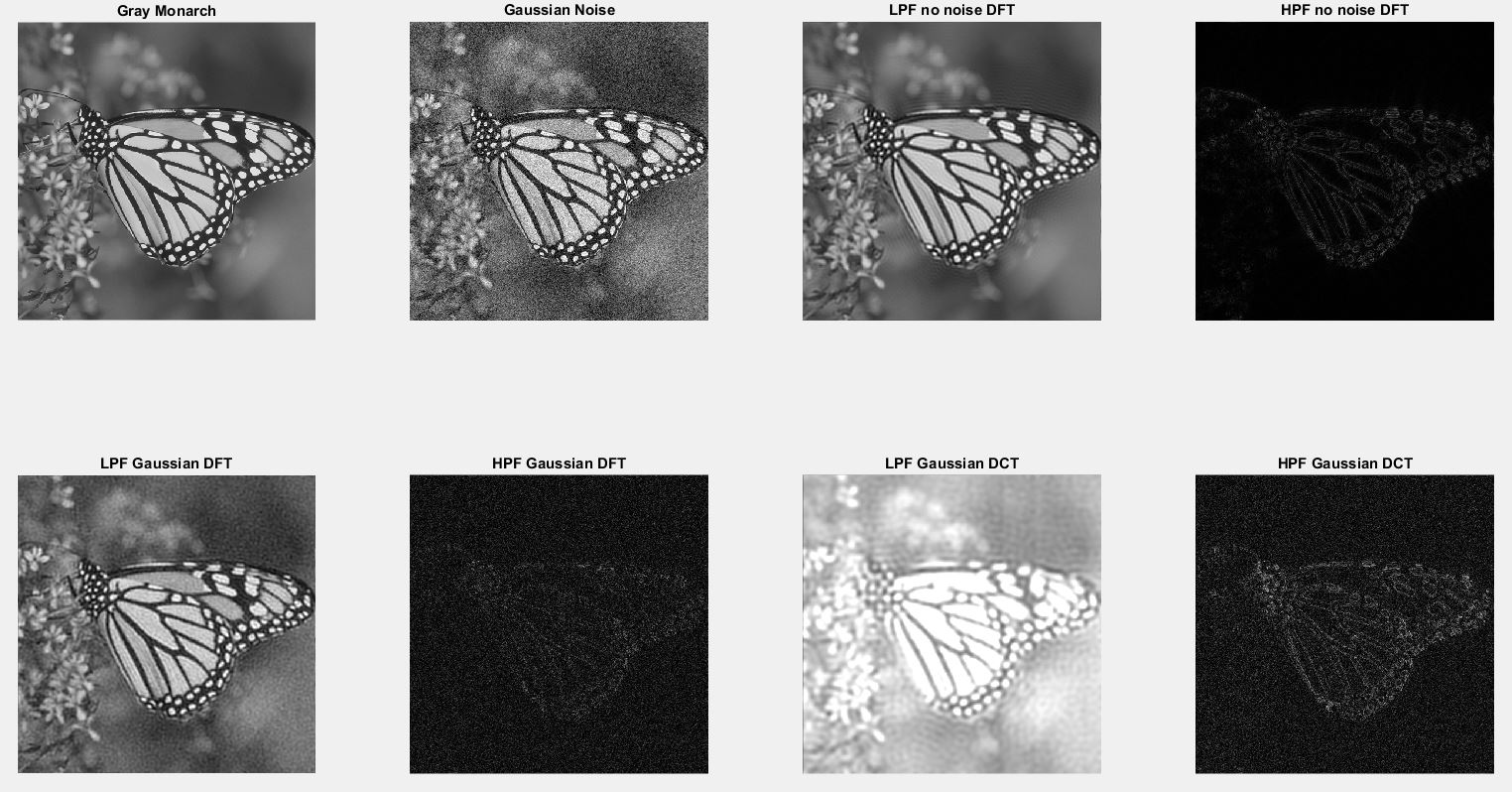


**Figure 2.4.** Transform operation



**Figure 2.5.** Generalized Linear Filtering

* + 1. *Result and analysis result*



**Figure 2.6.** Result of problem 2.3

HPF gives the output image almost black and remained edge of object because both types of transformation DFT and DCT contain low frequency components mostly.

With no-noise image, LPF makes image is brighter lightly than the original image. While with Gaussian noise, for DFT, the DFT image after LPF look nearly the original but more blurred, even very blurred for DCT image.