Homework 4

Noise Filter

2020311381 Sanghyeok Choi

**Goal of Project** : Understanding the concept of FIR linear noise filter and non-linear noise filters

**Tools :** Gaussian noised image, Salt & Pepper noised image, Median filter, Hamming window

1. Gaussian noised image :

Gaussian noised images add noise that has gaussian random distribution. It sometimes call AWGN(Additive white Gaussian noise) instead. It is strictly stationary process. Every noise is independent each other. Many processes consider this AWGN.

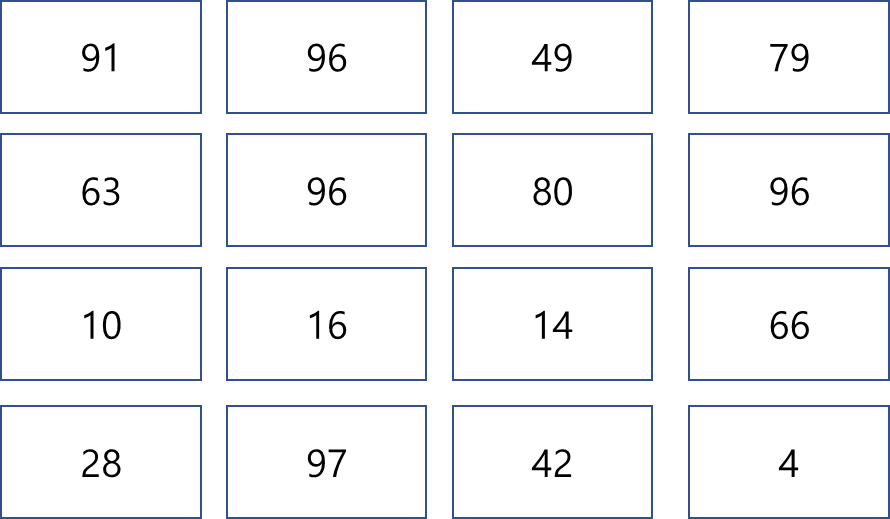
2. Salt & Pepper noised image :

It is also known as impulse noise. This noise can be caused by sharp and sudden disturbances in the image signal. It presents itself as sparsely occurring white and black [pixels](https://en.wikipedia.org/wiki/Pixel). An effective [noise reduction](https://en.wikipedia.org/wiki/Noise_reduction) method for this type of noise is a [median filter](https://en.wikipedia.org/wiki/Median_filter) or a [morphological filter](https://en.wikipedia.org/wiki/Mathematical_morphology).

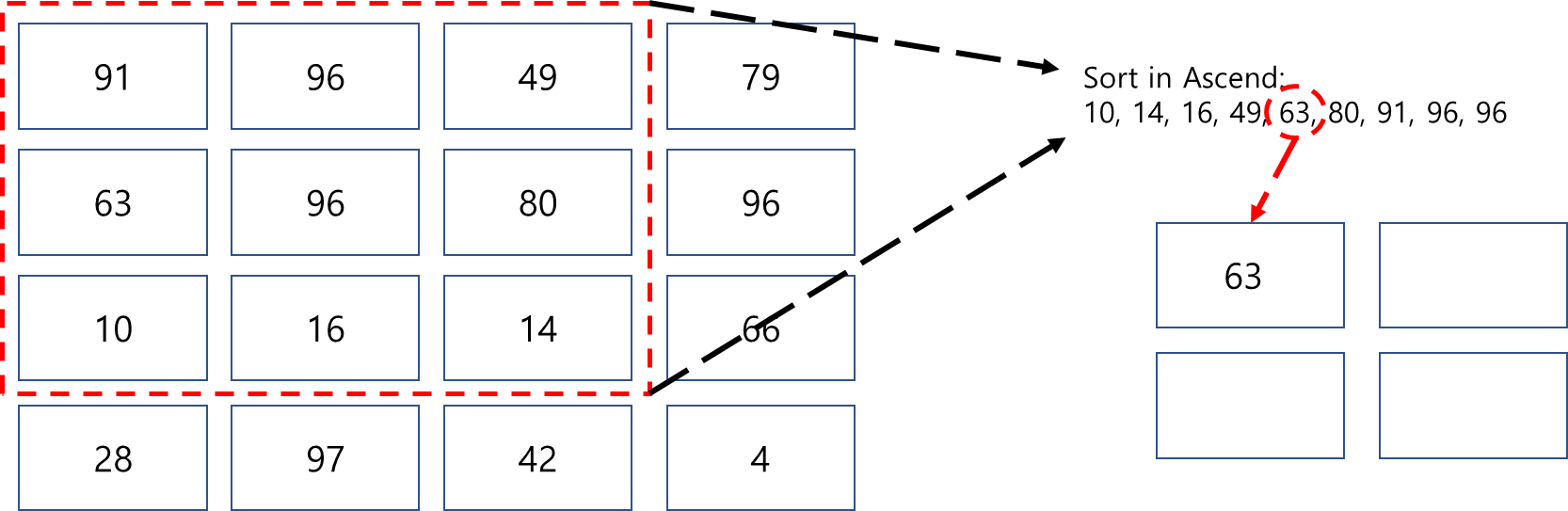
3. Median filter :

It is rank order filter that output value is depends on the ranking of the pixels according to their grey values inside the filter window. Median filter use center rank value.

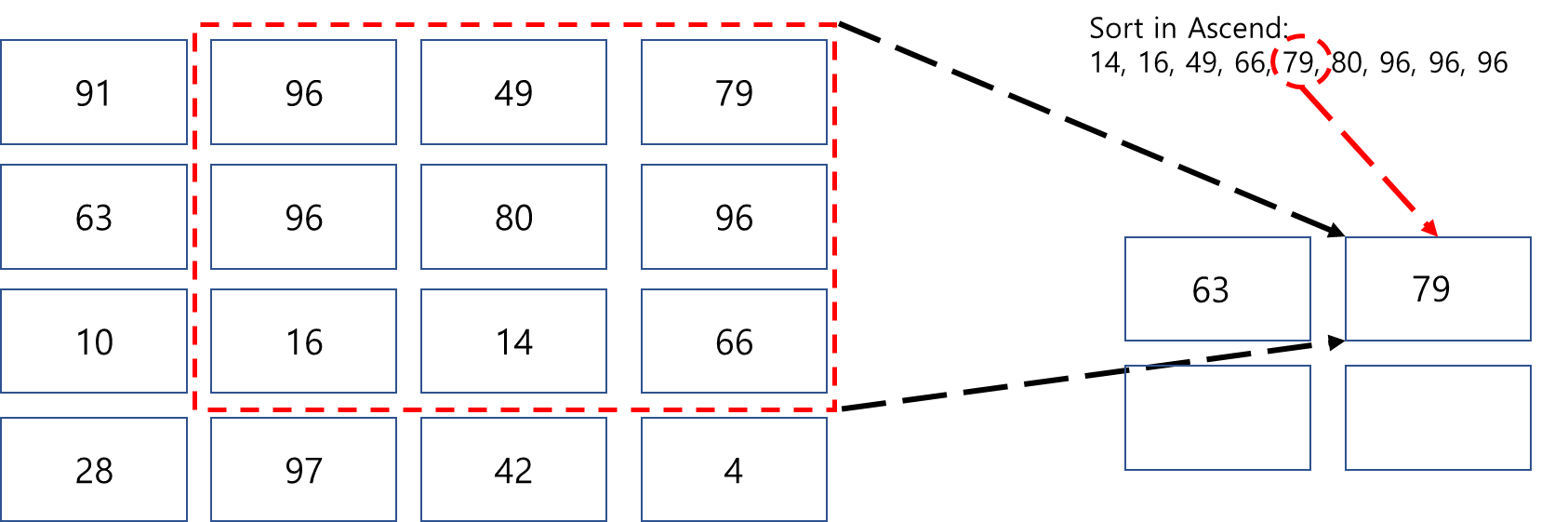
Let’s consider 4x4 image and median filter is 3 x 3.



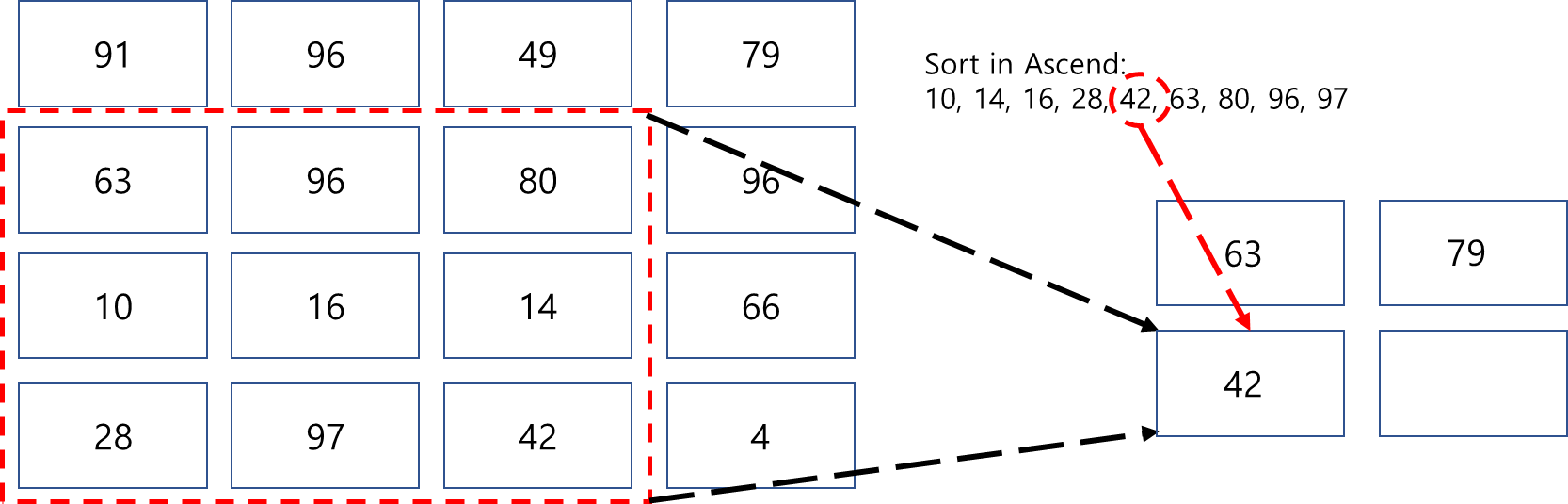
First output :



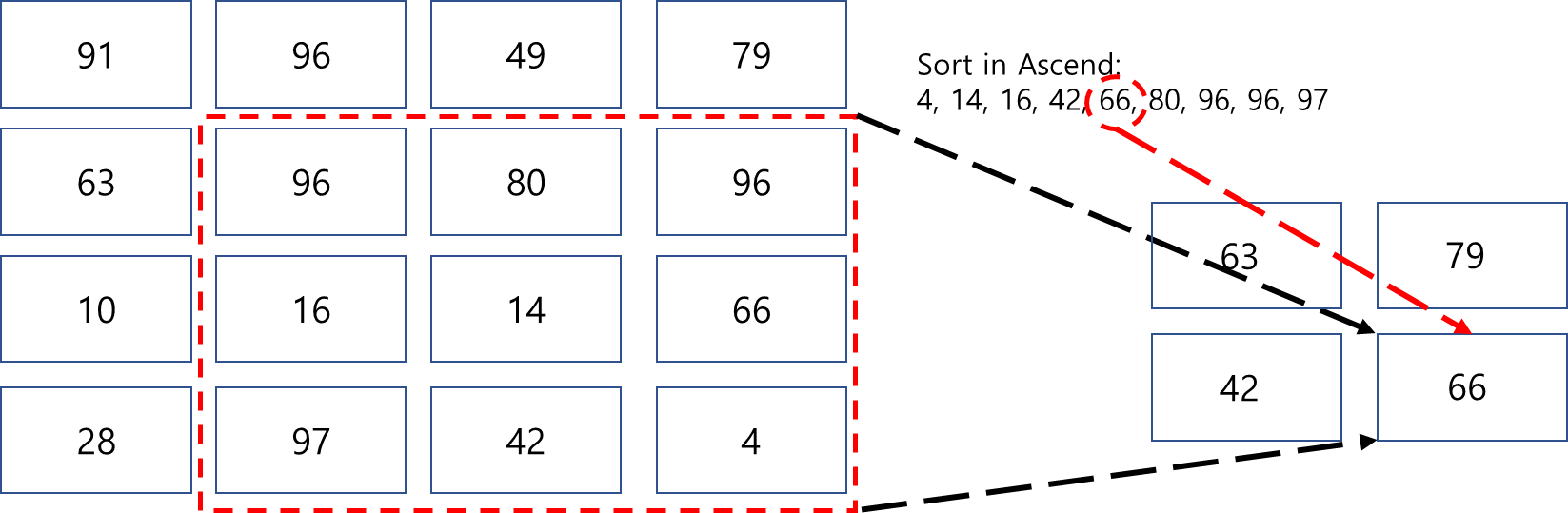
Second output :



Third output :



Fourth output :



In each iteration, median filter find correspond center rank value and copy to correspond output.

You can see output size is smaller than input size. It is because of kernel size that

output size = input size – filter size + 1. Input size is 4 and filter size is 3 so output size = 4 -3 + 1 = 2.

How can you fit with input size? Just put zero padding in every boundary. You put zero padding in each direction that mean output size = input size – filter size + 2\*padding size + 1. When output size and input size is same, padding size = (filter size – 1)/2 = 1. So consider zero padding, input become



Input size become 4 x 4 to 6 x 6 as you can see. You can apply median filter in upper image values.

Actually it has to consider stride that step size of the filter when traversing the image. Finally output size = (input size – filter size + 2\*padding size)/stride + 1. However In this experiment, stride = 1, so do not care about it.

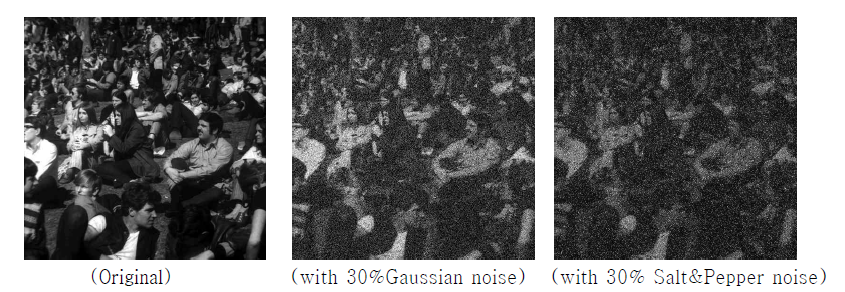
4. Hamming window :

In [signal processing](https://en.wikipedia.org/wiki/Signal_processing) and [statistics](https://en.wikipedia.org/wiki/Statistics), a window function is a mathematical function that is zero-valued outside of some chosen [interval](https://en.wikipedia.org/wiki/Interval_(mathematics)), normally symmetric around the middle of the interval, usually near a maximum in the middle, and usually tapering away from the middle. Hamming window is one of Hann window that w[n] = . When cut-off frequency is , It means H(w) = . Then h[n] = . Consider zero phase, w[n] is zero phase because it is symmetric but h[n]. So shift h[n] for (M-1)/2.

Finally 1D - Hamming filter = w[n]h[n]. 2 -D Filter is separable, do Matrix product correspond 1D – Hamming filter each other.

**Process :**

In this project, I use given images( Original image, with 30% Gaussian noise, with 30% Salt & Pepper noise )



< Fig 1 >

First I use median filter. Median filter usually use in Salt & Pepper noise. When apply median filter, 30% Salt & Pepper noise would be clear image but 30% Gaussian noise are not. Second I use Hamming filter. Consider filter size is 5 x 5, use w[n] & . For 5 points (e.g. n = 0,1,2,3,4).

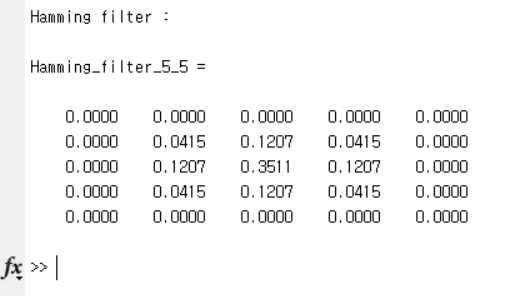
w[n] = [0.08, 0.54, 1 ,0.54, 0.08] & = [0, 0.318, 0.5, 0.318, 0]. Then 1 -D filter is dot product of each signal that [ 0, 0.1719, 0.5, 0.1719, 0 ]. Consider sum of filter weight is 1 for compare with original image, divide each weight to sum of weights = >[0, 0.2037, 0.5936, 0.2037, 0].

Now 2-D filter is separable that mean it is matrix product of 1 – D filter. So final filter =

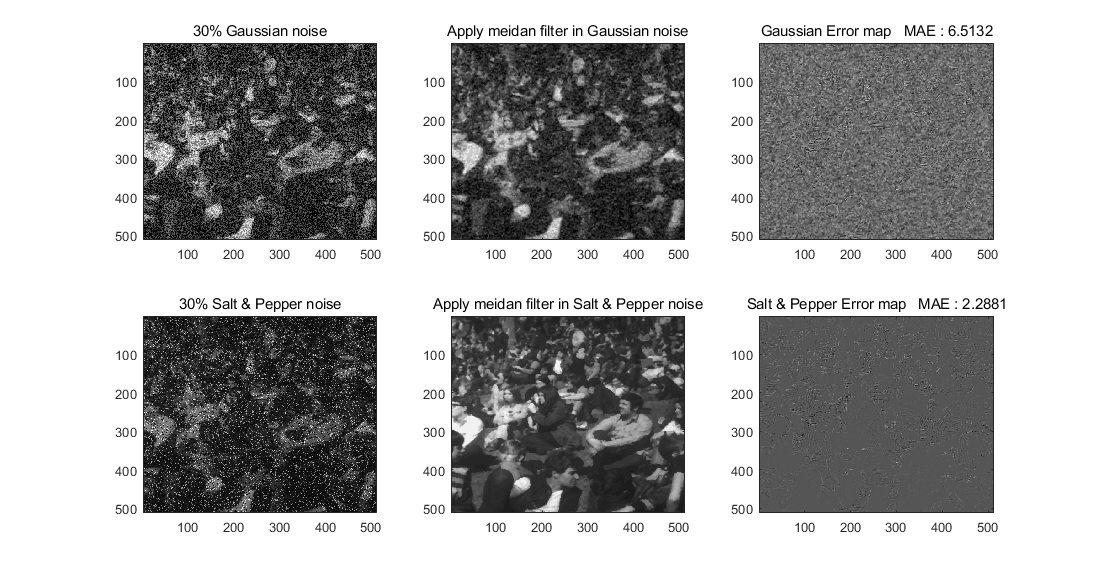
= .

I apply this method by “Matlab” tool.

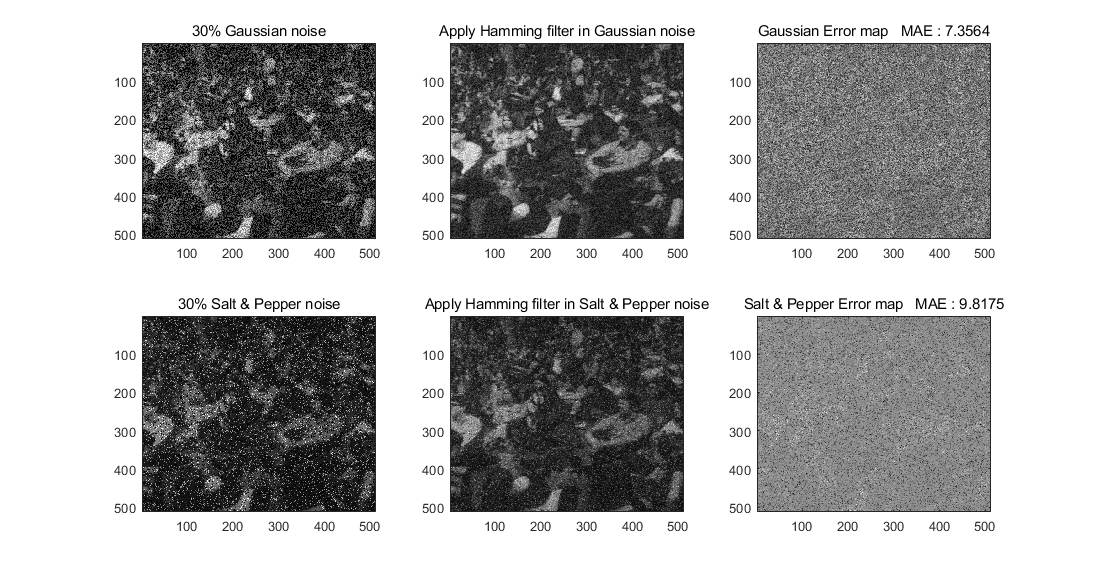
**Result :**



< Fig 2 >



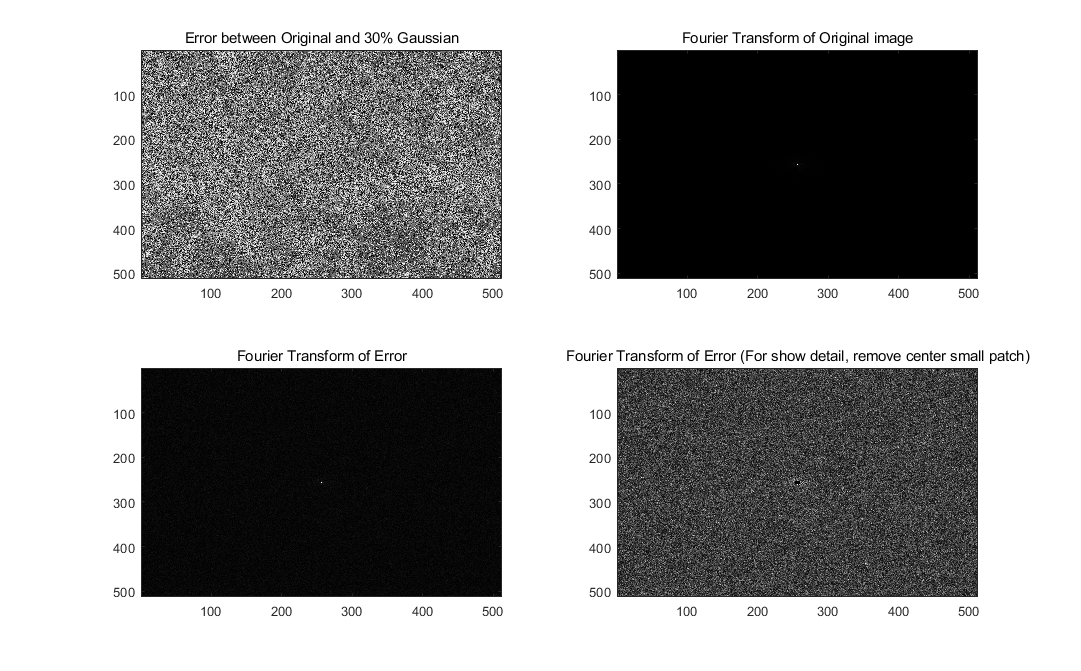
< Fig 3 >



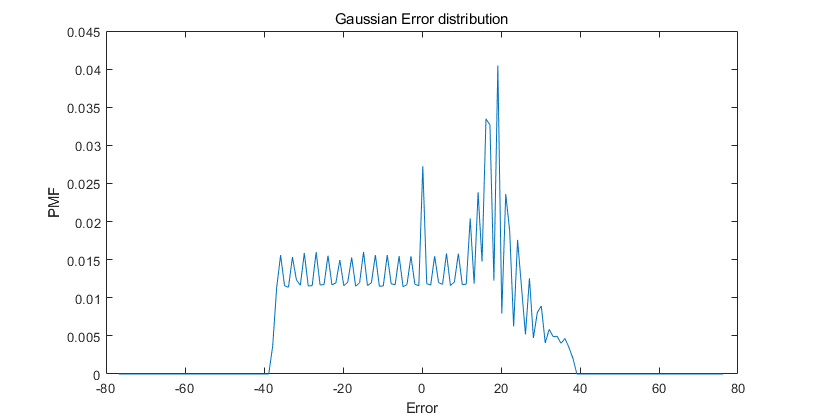
< Fig 4 >

< Fig 2 > show Hamming filter. When you see < Fig 3 >, median filter is good at Salt & Pepper noise. In < Fig 3 >, each images is worse than median filter. Also gaussian noise’s error is less than Salt & Pepper case in Fig 3

**Discussion :**

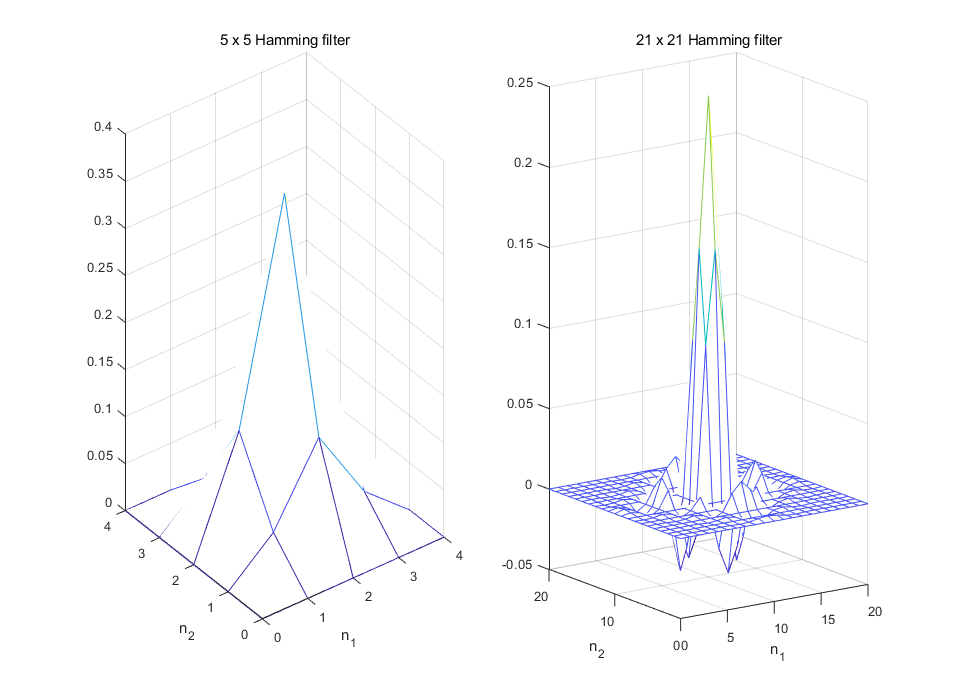


< Fig 5 >

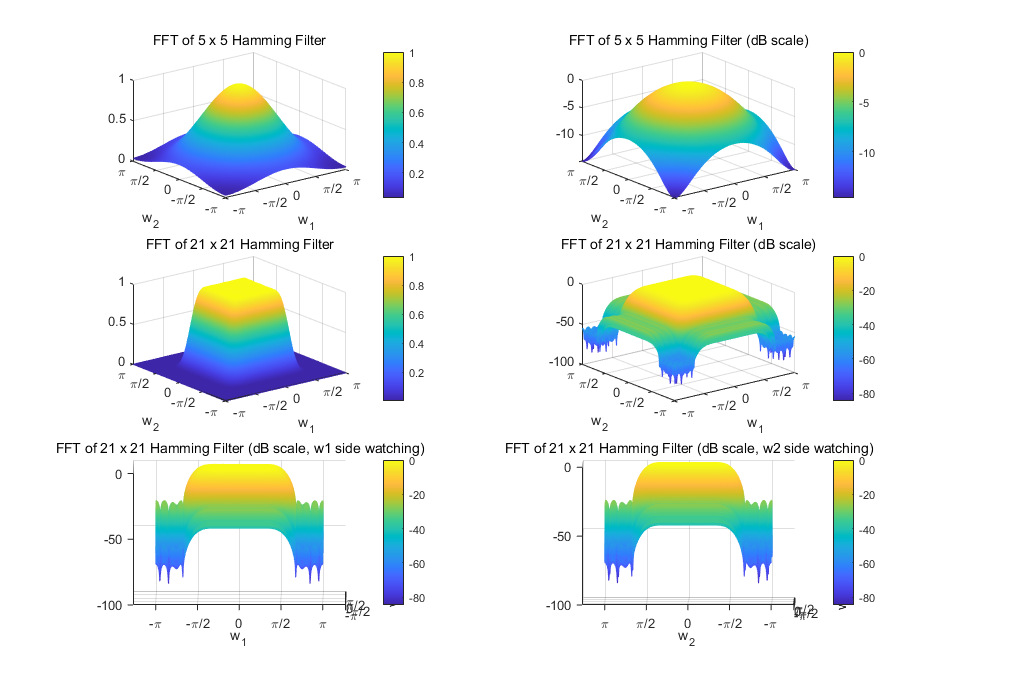


< Fig 6 >

< Fig 5 > & <Fig 6> are analysis for gaussian error. In figure 5, left – top image is error of 30 % gaussian noise and Original image. Gaussian noise’s Fourier transform is gaussian noise. Left – bottom image is frequency domain of error, but as you can see intensity of center region is too big to watch other region. You can watch top – right image is frequency domain of original image and indicate original image’s center region’s intensity is much bigger than other region. It effects bottom – left image because I get error by subtraction of gaussian noised image and original image, but professor mention that there are unknown error by convert images. So error that I get is not exact error so original image’s frequency domain could effects in error’s frequency domain that I get. Bottom-right image remove center region ‘s high intensity value. As I mention, gaussian noise’s frequency domain is gaussian noise, bottom-right image looks like gaussian noise. < Fig 6 > is error distribution that subtraction of original image and 30% gaussian noised image. It might be gaussian distribution but as professor mention, there are unknown error so it is not exact gaussian noise.

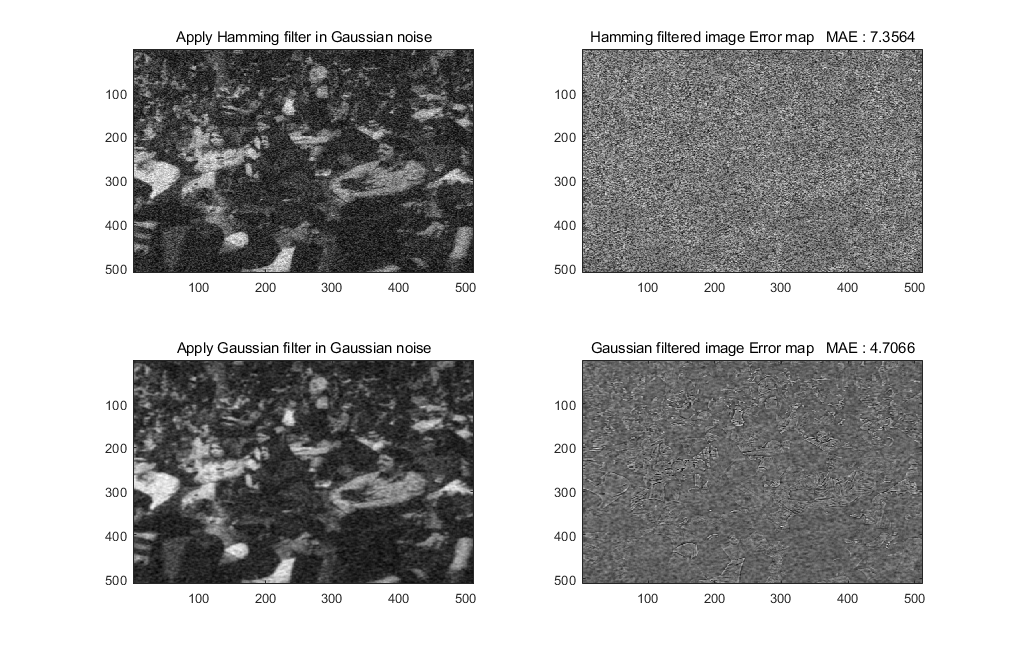


< Fig 7 >



< Fig 8 >

< Fig 7 > & < Fig 8 > about filter by hamming window. < Fig 7 > is about image domain and < Fig 8 > is about frequency domain. In < Fig 7 >, left image cannot remind it is similar with sinc function because the number of signals is not big. Right image is filter that size is 21 x 21. It looks like sinc function. Now let’s see < Fig 8 >. First row and second row are frequency domain of 5 x 5 filter and 21 x 21, left images are normal scale while right images are dB scale. First row images are much more smooth than second row because it has much less signal. Third row images are watching different side of right image in second row. Each image remind it is common hamming windowed filter.



< Fig 9 >

Gaussian noised image can handle with gaussian filter. Gaussian filter is  [filter](https://en.wikipedia.org/wiki/Filter_(signal_processing)) whose [impulse response](https://en.wikipedia.org/wiki/Impulse_response) is a [Gaussian function](https://en.wikipedia.org/wiki/Gaussian_function). As you can see error is much less than other filter.

I learn filter convolution method in deep learning class. There would much more efficient filter in correspond images. But anyway I think DL’s convolution method is affected by these filter methods.

**Reference :**

<https://en.wikipedia.org/wiki/Salt-and-pepper_noise>

<https://en.wikipedia.org/wiki/Median_filter>

<https://en.wikipedia.org/wiki/Window_function>

<https://en.wikipedia.org/wiki/Gaussian_filter>