Signal Processing in Practice
Assignment 5

Saptarshi Mandal

5th February 2024

1 Experiment 1: Low Pass Gaussian Filter

Images often get corrupted by white Gaussian noise. A common method to remove this noise is to use a Gaussian low-pass filter. This filter essentially smooths out the image by considering the average of nearby pixels, with more weight given to closer pixels. Think of it like blurring slightly, but in a controlled way.

Here's how it works mathematically:

- Noisy image: y(m,n), where m and n denote pixel coordinates.
- Denoised image: $\hat{x}(m, n)$, the image we aim to obtain after noise reduction.
- Filter kernel: w(k,l), a matrix representing the "weight" given to each pixel's contribution during averaging. This depends on the distance from the central pixel (k and l are offsets).
- Constant factor: C, chosen to ensure that the sum of all weights in the kernel adds up to 1 (for proper normalization).
- Gaussian function: This function determines how the weights decay with distance, based on a parameter σ^2 that controls the level of smoothing. Higher σ^2 means more blurring.

The filtering process involves convolving the noisy image with the filter kernel:

$$\hat{x}(m, n) = \sum_{k=-P}^{P} \sum_{l=-P}^{P} w(k, l) y(m + k, n + l), \quad \text{where} \tag{1}$$

Here P defines the size of the filter (e.g., P = 1 for a 3x3 kernel, P = 2 for a 5x5 kernel, etc.)

$$w(k, l) = C \exp\left(-\frac{k^2 + l^2}{2\sigma^2}\right) \quad \text{and} \quad \sum_{k=-P}^{P} \sum_{l=-P}^{P} w(k, l) = 1. \tag{2}$$

Images corrupted with white Gaussian noise are often denoised using a simple Gaussian low pass filter. If the noisy image is $y(m,n)$ and the desired denoised image is $\hat{x}(m,n)$, the denoised image is obtained as

$$\hat{x}(m,n) = \sum_{k=-P}^{P} \sum_{l=-P}^{P} w(k,l) y(m+k, n+l), \quad \text{where} \tag{3}$$

$$w(k,l) = C \exp\left(-\frac{k^2+l^2}{2\sigma^2}\right) \quad \text{and} \quad \sum_{k=-P}^{P} \sum_{l=-P}^{P} w(k,l) = 1. \tag{4}$$

For different noise levels introduced in the same image, the best among $sigma = 0.1, 1, 2, 4, 8$ for the gaussian filter was found considering both mean squared error (MSE) and Structural Similarity Index (SSIM). The following are the results in a tabular manner:

Noise Level	**MSE**	**SSIM**
low noise	0.1	0.1
low-medium noise	0.1	1
medium noise	1	1
medium-high noise	1	2
high noise	4	4

Table 1: MSE and SSIM values for different noise levels

From this table we can infer that:

- As the noise level increases, both MSE and SSIM values worsen, indicating a decrease in image quality. This is expected, as noise addition corrupts the original image information.
- For low noise, any standard deviation might not significantly impact the results. Since the noise level is low, the image is already close to the original noise-free image, and any filtering might not make a noticeable difference.
- For low-medium noise, a small standard deviation might be optimal to preserve detail while smoothing noise. A small standard deviation kernel can smooth out some noise while preserving most of the image details.
- For medium noise, a moderate standard deviation likely balances noise reduction and detail preservation. A moderate standard deviation kernel can strike a balance between smoothing noise and preserving image details.
- For medium-high and high noise, larger standard deviations are necessary for strong smoothing but might introduce blur. As the noise level gets higher, larger standard deviations are needed to effectively remove noise. However, this can also lead to blurring of image details.

Figure 1: Original Image

The original image is given in Figure. 1

The variation of optimal value of σ for different noise levels based on MSE and SSIM is given in Figure 2

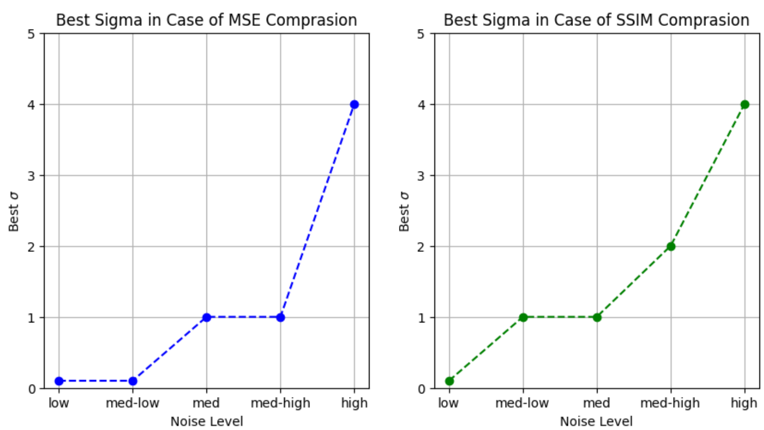

Figure 2: Plot for optimal sigma

A set of images denoised using the optimal-sigma Gaussian filter using the image quality assessment metrics: MSE (Mean Squared Error) and SSIM (Structural Similarity Index) is given in Figure 3

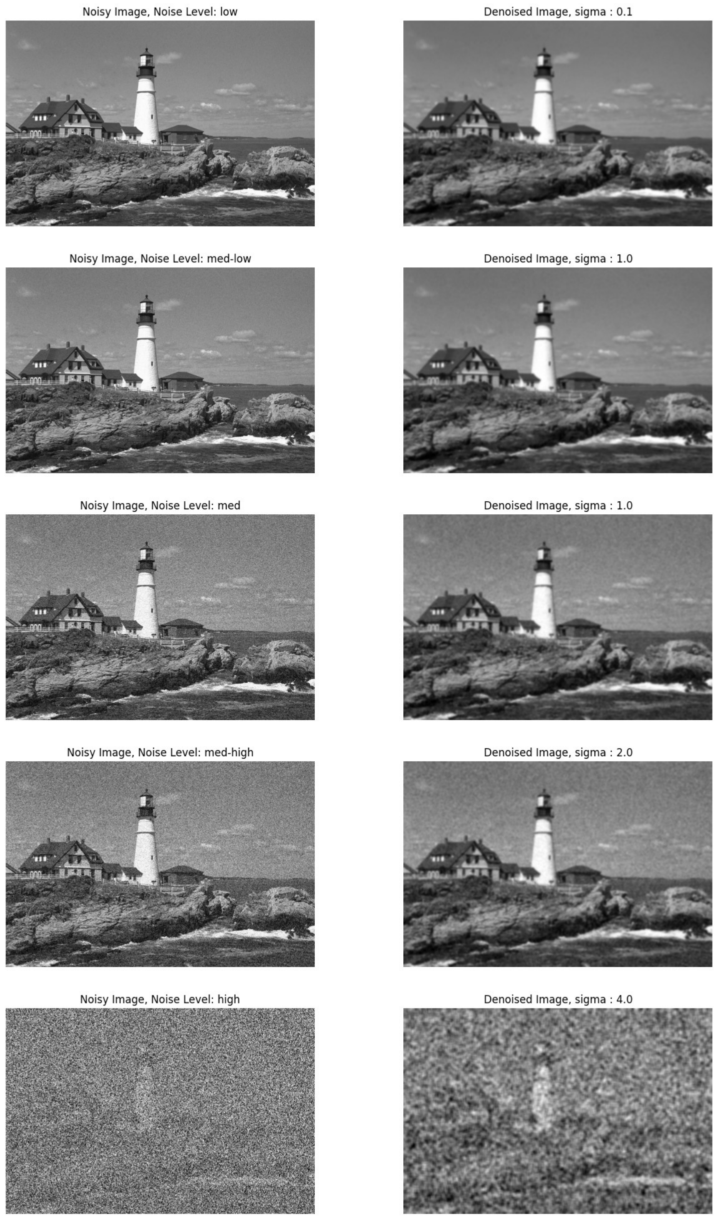

Figure 3: Optimal Denoised Image for each Noise Level and the Optimal σ value

2 Experiment 2: Bilateral Filtering

The bilateral filter is defined as

$$g(m,n) = \frac{1}{C(m,n)} \sum_{k=-P}^{P} \sum_{l=-P}^{P} G(k,l) H(f(m,n) - f(k,l)) f(m+k, n+l), \quad (5)$$

where

$$H(x) = \exp\left(-\frac{x^2}{2\sigma_H^2}\right)$$

where x denotes the intensity difference

$$G(k,l) = \exp\left(-\frac{k^2+l^2}{2\sigma_G^2}\right),$$
$$C(m,n) = \sum_{k=-P}^{P}\sum_{l=-P}^{P} G(k,l)H(f(m,n)-f(k,l)).$$

The image has been denoised using a bilateral filter and a normal Gaussian filter. The outputs are shown below:

2.1 Specifications of the filters used

For the bilateral filter:

- σ_H : 100
- σ_G : 50
- kernel size = (11,11)

For the Gaussian filter:

- $\sigma = 100$
- kernel size = (5,5)

2.2 Analysis

It can be seen that Bilateral filtering outperforms Gaussian filtering for denoising images corrupted with white Gaussian noise from the results. Here are the possible reasons :

1. **Edge Preservation**:

   Gaussian filter: Considers only spatial proximity when averaging pixels, leading to blurring of edges along with noise reduction.

   Bilateral filter: Considers both spatial proximity and intensity similarity. Pixels farther away in space but similar in intensity contribute less to the average, helping preserve edge sharpness.

2. **Adaptive Smoothing**:

   Gaussian filter: Applies uniform smoothing across the entire image, often over-smoothing flat areas while under-smoothing noisy areas.

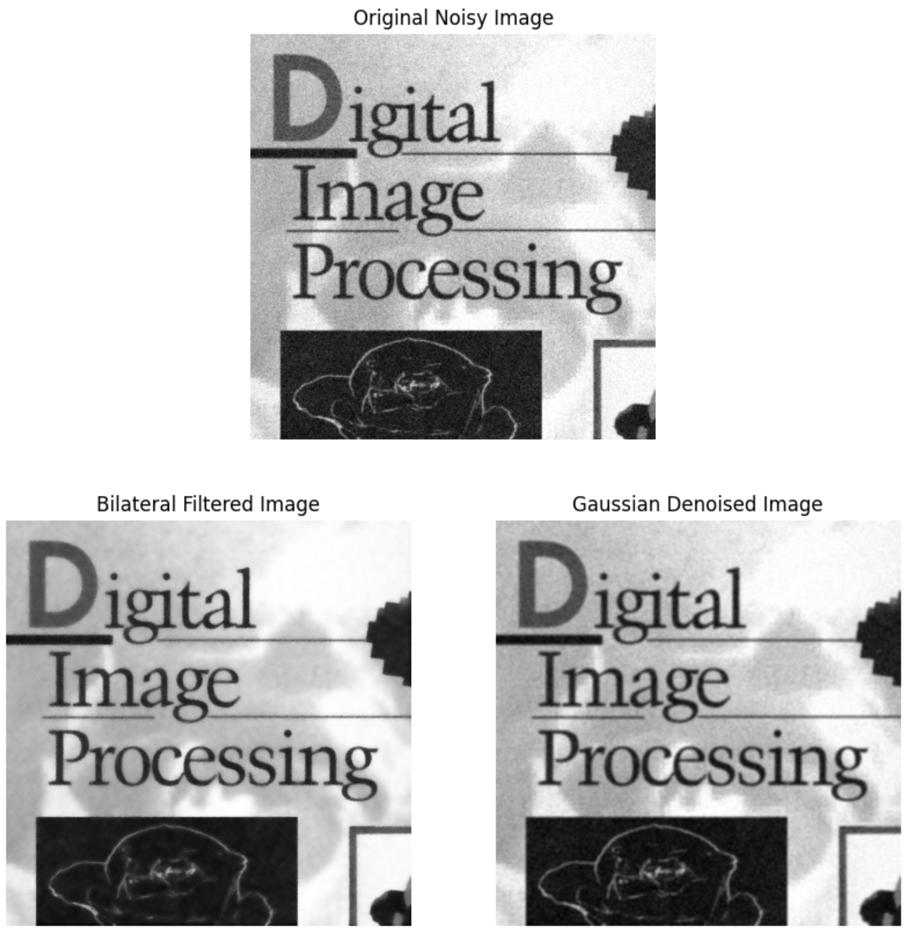

Figure 4: Bilateral Filtering

Bilateral filter: Adapts the smoothing strength based on local image content. In regions with high noise and low edge variation, it can apply more aggressive smoothing, while in areas with sharp edges, it uses milder smoothing to preserve details.