

IMAGE PROCESSING AND INTERPRETATION

CS474.674.1001

PROGRAMMING ASSIGNMENT NO: 2

A Report

By

PRATIK WALUNJ

JALPA KAILA

Division of Work Statement

Pratik Walunj and Jalpa Kaila divided the work for this assignment as follows:

Coding:

- Pratik handled coding for Questions 2, 3 and 4.
- Jalpa handled coding for Questions 1, 2 and 5.

Report Write-up:

- Pratik contributed to Section 2.3, 2.4, 2.5, 3.2, 3.3 and 3.4.
- Jalpa contributed to Section 2.1, 2.2, 2.5, 3.1, 3.2, and 3.5.

We collaborated on coding and provided mutual input for report sections to ensure balanced contribution in both aspects.

TABLE OF CONTENTS

	Page
CHAPTER 1. MOTIVATION	1
CHAPTER 2. THEORY	2
2.1 Spatial Filtering (Correlation)	2
2.2 Averaging and Gaussian Smoothing	5
2.3 Median Filtering	6
2.4 Unsharp Masking and High Boost Filtering	7
2.5 Gradient and Laplacian	8
CHAPTER 3. RESULT AND DISCUSSION	11
3.1 Spatial Filtering (Correlation)	11
3.1.1 Correlation	11
3.1.2 Correlation with input and pattern Image	12
3.1.3 Correlation for finding location	12
3.2 Averaging and Gaussian smoothing	13
3.2.1 Smoothing Lenna and sf image using 7x7 test image 15x15 averaging filter	13
3.2.2 Smoothing Lenna and sf image using 7x7 test image 15x15 Gaussian filter	15
3.3 Median Filtering	17
3.3.1 Implement median Filtering	17
3.3.2 Corrupt Lenna and boat image with salt and pepper noise	18
3.3.3 Median filtering on corrupted Lenna and boat image	20
3.3.4 Median filtering and averaging filtering comparison	24
3.4 Unsharp masking and high boost filtering	25

3.4.1 Implementation of unsharp masking and high boost filtering	25
3.4.2 Experimenting with different k - values	26
3.5 Gradient and Laplacian	27
3.5.1 Sharpening Lenna and sf image using Prewitt, Sobel, and Laplacian mask	27
3.5.2 Gradient magnitude and partial derivative for prewitt and sobel mask	28
CHAPTER 4 REFERENCES	31

LIST OF FIGURES

	Page
Figure 1. Spatial Filtering on Input image	2
Figure 2. Spatial Filtering on Input image with $K \times K$ window	3
Figure 3. Size of $K \times K$ window	3
Figure 4. Correlation and Convolution	4
Figure 5. Average Filter Mask	5
Figure 6: Gaussian Filter Mask	6
Figure 7: Median Filter	7
Figure 8: Gradient magnitude and direction	8
Figure 9: Roberts, Sobel, and Prewitt masks for gradients	8
Figure 10: Laplacian masks	9
Figure 11: Results of Laplacian and gradient (Sobel) masks in input image	10
Figure 12: Spatial Filtering on input image Lenna	11
Figure 13: Result of Correlation on Input and pattern image	12
Figure 14: Result of Correlation for finding location	13
Figure 15: Averaging smoothing with 7×7 mask	14
Figure 16: Averaging smoothing with 15×15 mask	15
Figure 17: Gaussian Smoothing 7×7 mask	16
Figure 18: Gaussian Smoothing 15×15 mask	17
Figure 19: Original Image of boat and Lenna	18
Figure 20: Lenna image with salt and pepper noise with noise 30% and 50%	18
Figure 21: Boat image with salt and pepper noise with noise 30% and 50%	19

Figure 22: Lenna image with 30% noise and 50% noise with 7x7 and 15x15 median Filter	20
Figure 23: Boat Image with 30% noise and 50% noise with 7x7 and 15x15 median Filter	21
Figure 24: Lenna image with 30% and 50% noise and 7x7 and 15x15 averaging Filter	23
Figure 25: Boat image with 30% and 50% noise and 7x7 and 15x15 averaging Filter	24
Figure 26: Unsharp Masking image sharpening performed on Lenna Image	25
Figure 27: Unsharp Masking image sharpening performed on f_16 Image	25
Figure 28: High Boost Filtering Performed on Lenna Image	26
Figure 29: High Boost Filtering Performed on F_16 Image	27
Figure 30: Lenna and sf Image with Prewitt Sobel and Laplacian Mask	28
Figure 31: Lenna Image with Partial Derivative of Prewitt and Sobel Mask	29
Figure 32: sf Image with Partial Derivative of Prewitt and Sobel Mask	30

CHAPTER 1. MOTIVATION

The motivation behind this programming assignment in image processing and interpretation is to learn and practice essential techniques for working with digital images. In this assignment, the focus is on understanding and applying various methods to enhance, manipulate, and analyze images.

We aim to get a deeper understanding of image processing concepts such as spatial filtering, smoothing, sharpening, and gradient operations. These techniques are like tools in a digital toolbox, and we want to learn how to use them effectively. For example, spatial filtering involves looking at a small part of an image at a time and making decisions based on what's in that small part.

We'll experiment with different types of filters, like correlation, averaging, Gaussian, median, unsharp masking, and high boost filtering. Filters are like special effects we can apply to images to change their appearance. For instance, averaging helps to reduce noise and make images smoother, while unsharp masking makes images look sharper.

By applying these filters and examining their impact on images, we hope to gain hands-on experience in image processing. We'll also look at how these techniques can be used in real-world applications, like finding patterns in images or enhancing details in photographs.

Overall, this assignment serves as a practical learning opportunity, where we get to apply what we've learned about image processing. It's a chance to develop our skills in working with digital images and interpreting the results. Through collaborative effort and experimentation, we aim to better understand the world of image processing.

CHAPTER 2. THEORY

2.1 Spatial Filtering-Correlation

Spatial Filter is defined by:

1. Neighborhood pixel-Which pixel to process
2. Operation-how to process the pixels in the specified neighborhood.

It has a has a square shape $K \times K$ (we call it a “window”). e.g., 3×3 or 5×5 . A new value is obtained by processing the pixels in the window. Stored at the corresponding center location of the window in the output image.

As you can see, in figure 1, the spatial filtering is applied to the input image by considering the fixed area of an input image and after processing the spatial filtering on that area. A new pixel value is obtained, and it stored at the corresponding center location of the window in the output image.

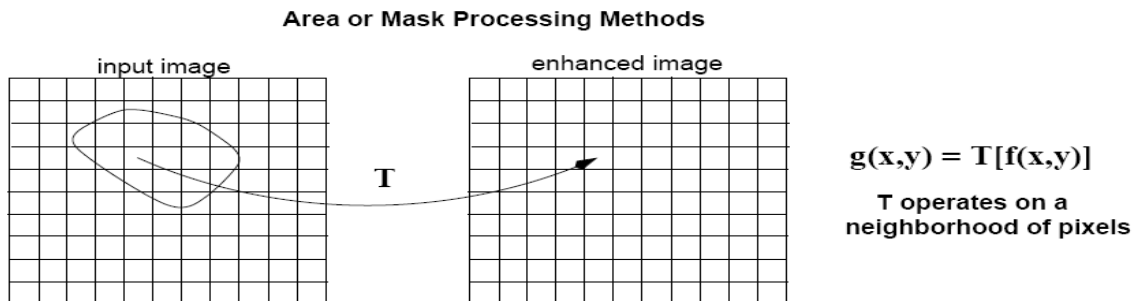


Figure 1: Spatial Filtering on Input image

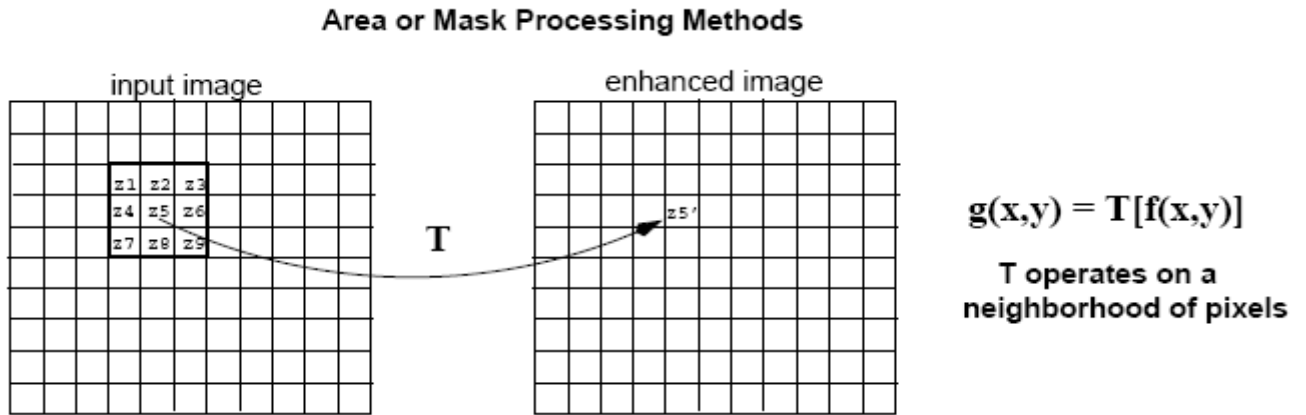


Figure 2: Spatial Filtering on Input image with $K \times K$ window

Depending on the operator used, a filter can be linear or non-linear:

Example: $z'_5 = 5z_1 - 3z_2 + z_3 - z_4 - 2z_5 - 3z_6 + z_8 - z_9 - 9z_7$ (linear)

Example: $z'_5 = \max(z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8, z_9)$ (non-linear)

Two common linear operators are:

- **Correlation:** The Output of correlation is weighted sum of input pixels. The weights are defined by a $K \times K$ mask (has the same size as the window):

w1	w2	w3
w4	w5	w6
w7	w8	w9

Figure 3: Size of $K \times K$ window

$$z'_5 = R = w_1z_1 + w_2z_2 + \dots + w_9z_9$$

Figures 2 and 3 represent the size of window in input image and size of $K \times K$ window applied on it, in results we get pixel value that is applied to corresponding center pixel of the image. For handling the locations which are close to boundaries, usually we pad it with zeros, or we can skip the first and last few rows and columns.

Correlation is often used in applications where we need to measure the similarity between image and pattern.

Example template matching.

- **Convolution:** Convolution is like correlation except the mask is first flipped both horizontally and vertically.

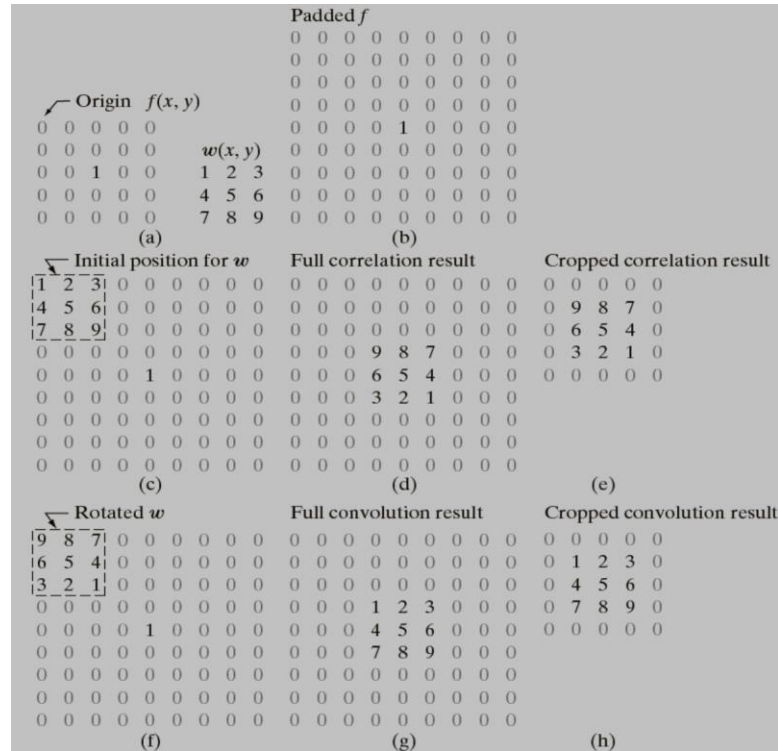


Figure 4: (a, b) input image $f(x, y)$, 3 x 3 Mask $w(x, y)$ and Input image padded with 0. (c, d, e) Correlation processing on input image with mask. (f, g, h) Convolution processing on input image with mask.

Convolution processing is like correlation, but the mask $w(x, y)$ is first flipped both horizontally and vertically.

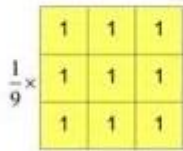
As you can see in figure 4, the position of the pixel changed both horizontally and vertically then it applied to input image. If $w(i, j)$ is symmetric (example: $w(i, j) = w(-i, -j)$), then convolution is equivalent to correlation.

2.2 Averaging and Gaussian Smoothing

Smoothing is also called Low pass Filtering. It is useful for reducing the noise and removes the detail from the image. The element of the mask must be positive. Common smoothing filters are Averaging, Gaussian and Median Filtering.

Averaging: In Averaging mask weight are all equal to 1 and result divide by K^2 , Here K represents the size of mask. As you can see in figure 5 (a) and (b) represent the Averaging mask with different K x K window. In the case of the 3x3 matrix it is divided by 9 and in case of 5x5 matrix the result is divided by 25. Mask size determines the degree of smoothing. In the case of Averaging, it's loss the details. In averaging each pixel is divided by total number of pixel values.

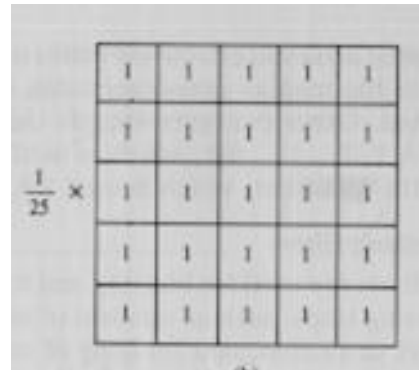
Averaging Filter - Example



A 3x3 grid of yellow squares, each containing the number 1. To the left of the grid is the expression $\frac{1}{9} \times$.

1	1	1
1	1	1
1	1	1

(a)



A 5x5 grid of light gray squares, each containing the number 1. To the left of the grid is the expression $\frac{1}{25} \times$.

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

(b)

Figure 5: (a) Average filter mask 3x3 and (b) Average filter mask 5x5

Gaussian Smoothing: Mask Weights are sampled value of 2D gaussian function. This value decreases as they get further away from the center of the mask. Gaussian filtering preserves the detail better than averaging. In Gaussian smoothing, standard deviation (σ) controls the amount of smoothing since it determines the mask size. As you can see in figure 6, the values of gaussian mask in both 7x7 and 15x15 decreases as it get further away from center value.

$$G_{\sigma}(x, y) = \frac{1}{2\pi\sigma^2} \exp -\frac{x^2 + y^2}{2\sigma^2}$$

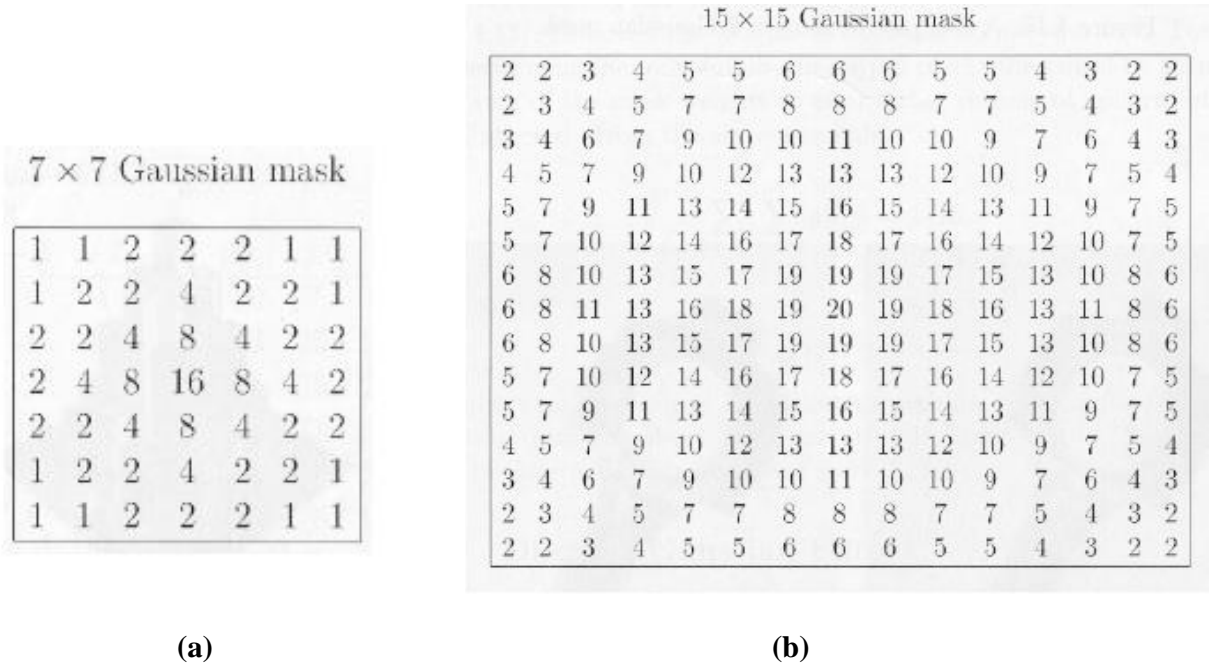


Figure 6: (a) Gaussian filter mask 7x7 and (b) Gaussian filter mask 15x15

2.3 Median Filtering

Median filtering is a widely used image processing technique employed to reduce noise and enhance the quality of digital images. It is particularly effective in removing various types of noise, such as salt-and-pepper noise.

In this the fundamental principle is that it replaces each pixel with the median value of neighborhood of pixel median filtering can cause loss of sharpness particularly where the noise and image feature overlap, median filtering is computationally expensive.

As you can see in figure 7, the pixels from input image are sorted by sorting algorithm and from sorted array median is found. Then that median value is applied to every corresponding pixel in input image.

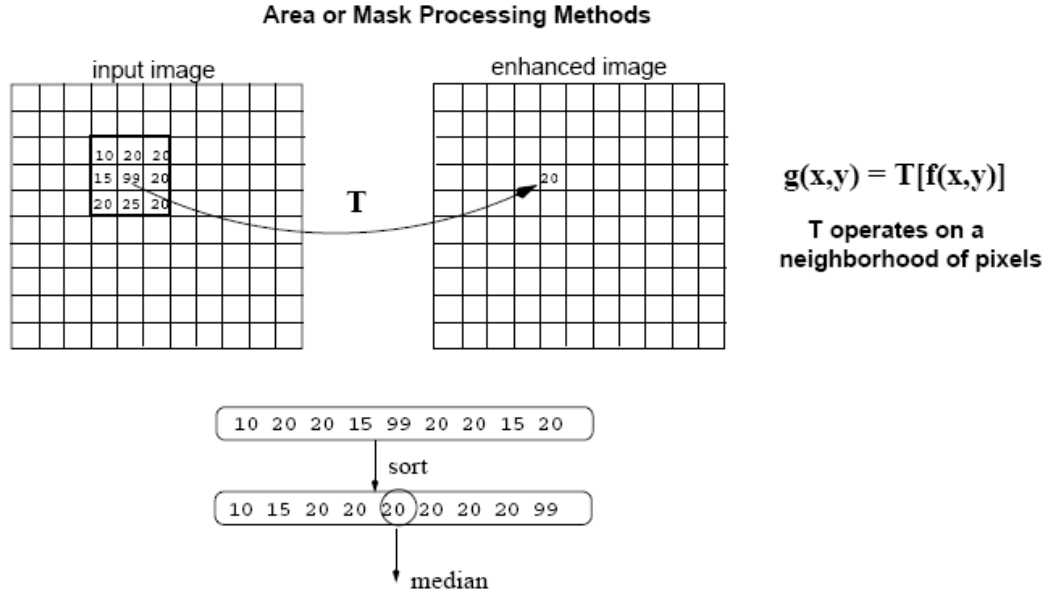


Figure 7: Median Filter

2.4 Unsharp Masking and High Boost Filtering:

It is the sharpening technique in which the image smooth image of original image is subtracted from the original image then we get gmask, then the weighted sum of gmask is added to original image.

$$g_{mask}(x, y) = f(x, y) - f_{LP}(x, y)$$

$$g(x, y) = f(x, y) + kg_{mask}(x, y), \quad k \geq 0$$

If $k = 1$ then the sharpening is called unsharp masking and if the value of $k > 1$ then it is called high boost Filtering.

High-boost filtering allows for more control over the enhancement process compared to unsharp masking. By adjusting the boosting factor, you can control the strength of the enhancement effect.

2.5 Gradient and Laplacian

Gradient:

Gradient is used to compute the image derivatives. The gradient is a vector which has magnitude and direction. Gradient magnitude provides information about edge strength. Gradient direction is perpendicular to the direction of the edge (useful for tracing object boundaries).

$$\text{grad}(f) = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

$$\text{magnitude}(\text{grad}(f)) = \sqrt{\frac{\partial f^2}{\partial x} + \frac{\partial f^2}{\partial y}}$$

$$\text{direction}(\text{grad}(f)) = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

Figure 8: Gradient computes the partial derivatives in x and y direction. The magnitude and direction of gradient in x and y direction.

To compute the derivatives in x and y direction, gradient uses Roberts, Sobel, and Prewitt masks. Gradient represents the 1st derivatives. The intensity changes like edges of gradient can be detected by local maxima and minima of the first derivatives. Figure 9 represents the Robert, Sobel and Prewitt mask using which the gradient is computed. So, gradient requires 2 masks in x and y direction for computations.

1	0
0	-1

0	1
-1	0

Robert Mask

-1	0	1
-1	0	1
-1	0	1

1	1	1
0	0	0
-1	-1	-1

Prewitt Mask

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Sobel Mask

Figure 9: Roberts, Sobel, and Prewitt masks for gradients

Laplacian:

Laplacians represent the 2nd derivative. The intensity changes like edges in Laplacian can be detected by zero crossing of the 2nd derivative where 2nd derivative changes the sign. Laplacians uses one mask for computation in both x and y direction instead of two masks in case of gradients.

0	1	0	1	1	1
1	-4	1	1	-8	1
0	1	0	1	1	1

0	-1	0	-1	-1	-1
-1	4	-1	-1	8	-1
0	-1	0	-1	-1	-1

Figure 10: Laplacian masks

Laplacians localize the edges better compare to gradient. Higher order derivatives are more sensitive to noise. Laplacian is less computationally expensive because it uses one mask for both x and y directions. Laplacian can provide edge magnitude information but no information about edge direction. Figure 10 represents the different Laplacian masks. Figure 11 shows the results after applying the Laplacian and Gradient-Sobel mask on input image. We can see that Laplacian provides more edge magnitude information compared to gradient masks.



(Input image)



(Laplacian)



(Gradient-Sobel)

Figure 11: Results of Laplacian and gradient (Sobel) masks in input image

CHAPTER 3. RESULTS AND DISCUSSION

3.1 Spatial Filtering: Correlation

3.1.1 Correlation

This section 3.1.1 shows the result of Spatial Filtering on input image. We used Mask size 7x7 for the input. The masks and their values we use here are given as a input variable. The Spatial Filtering is applied on Input image using the 7x7 mask size. Figure 12 represents the result after spatial filtering.

The correlation uses the sample mask for partial filtering and this 7x7 mask applied to all places of input image and then multiplies with the input image pixel values and final sum will produce new pixels which applied to center of the corresponding pixel for output image.

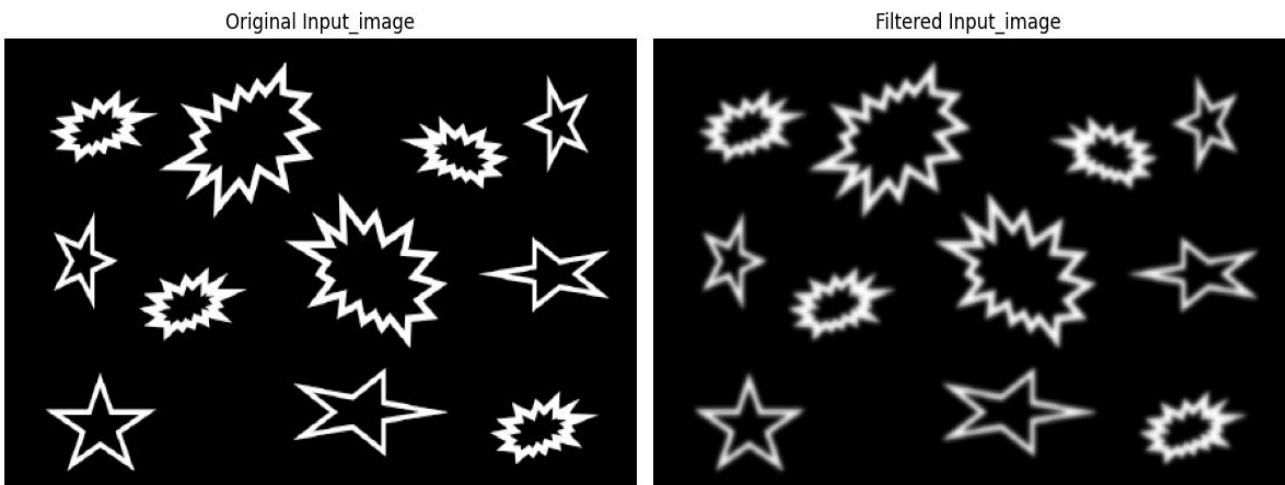


Figure 12: Left: Input image **Right:** Output image after performing Spatial Filtering on input image.

3.1.2 Correlation with Input and Pattern image

Section 3.1.2 shows the result of correlation between two images. The given Image which is on right side is taken as an input image and image on left side is taken as pattern image. The correlation in spatial filtering is specifically used for template or pattern matching.

The working of the Correlation is to apply the pattern image pixels to all place of input image and here we can skip some rows and columns or padded with 0 for handling the corner pixels of the image. Figure 13 shows the results after applying the pattern image to all places of input image. We can see that after applying the correlation the pixels in input image that match with the pattern image looks brighter compared to other pixels.

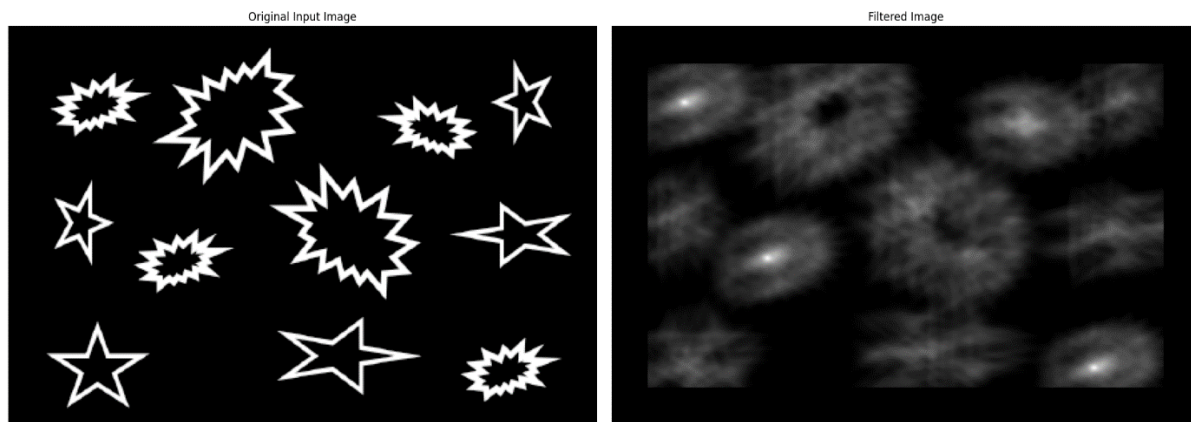


Figure 13: Top Left: Input image. **Right:** Filtered Output image after performing Correlation on Input image.

3.1.3 Correlation for finding Location.

This section 3.1.3 shows the result of location of the pattern shown in pattern image in input image. The main application of the correlation is template or pattern matching. Using that we can find the location of a pattern shown in pattern image in input image.

We can find the pattern by using the method normalizing cross-correlation, which is measure the similarity between two images at different location. This involves the sliding the pattern image over input image, computing the correlation at each position.

While sliding the template image over input image, we will calculate the correlation score at each position. keep a tract of maximum correlation score and its corresponding position, it gives us a best position match for given template image over input image.

The threshold can also be used to find maximum score we find in correlation. The threshold will filter out weak matches and only consider strong matches. To visualize the results, we draw the rectangle or mark the detected position in input image to indicate where the pattern matches.

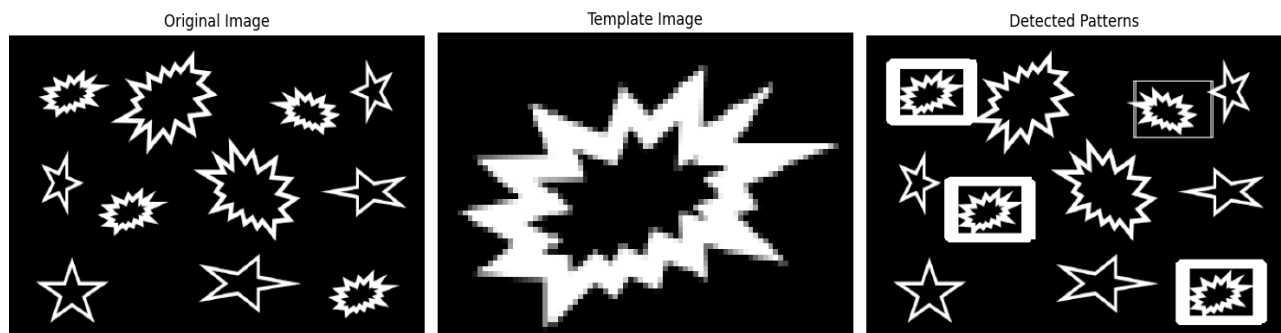


Figure 14: 1: Input Boat image. 2: Template image. 3: Detected template pattern in input image after correlation.

3.2 Averaging and Gaussian smoothing

3.2.1 Smoothing Lenna and sf image using 7x7 and 15x15 Averaging filter.

This section 3.2.1 shows the result of Smoothing after applying the 7x7 and 15x15 Averaging filter masks on Lenna and sf images. As you can see in Figure 15 and 16, first we give input Lenna and sf image. We applied 7x7 and 15x15 Filtering masks for smoothing.

As we understand in section 2.2 of chapter 2, Averaging reduces the noise and removes the details from the image. As a result, we can see that in figures 15 and 16 that after applying the 7x7 and 15x15 average filtering mask on input Lenna and sf images, it removes the detail from the image and blurs both Lenna and sf image. Compared to 7x7 masks, the results we got after applying 15x15 masks is more blurred Lenna and sf image.



Figure 15: Top 1: Input Lenna image. **Top 2:** Smoothed Lenna image using Averaging 7x7 mask. **Bottom 1:** Input sf image. **Bottom 2:** Smoothed sf image using Averaging 7x7 mask.

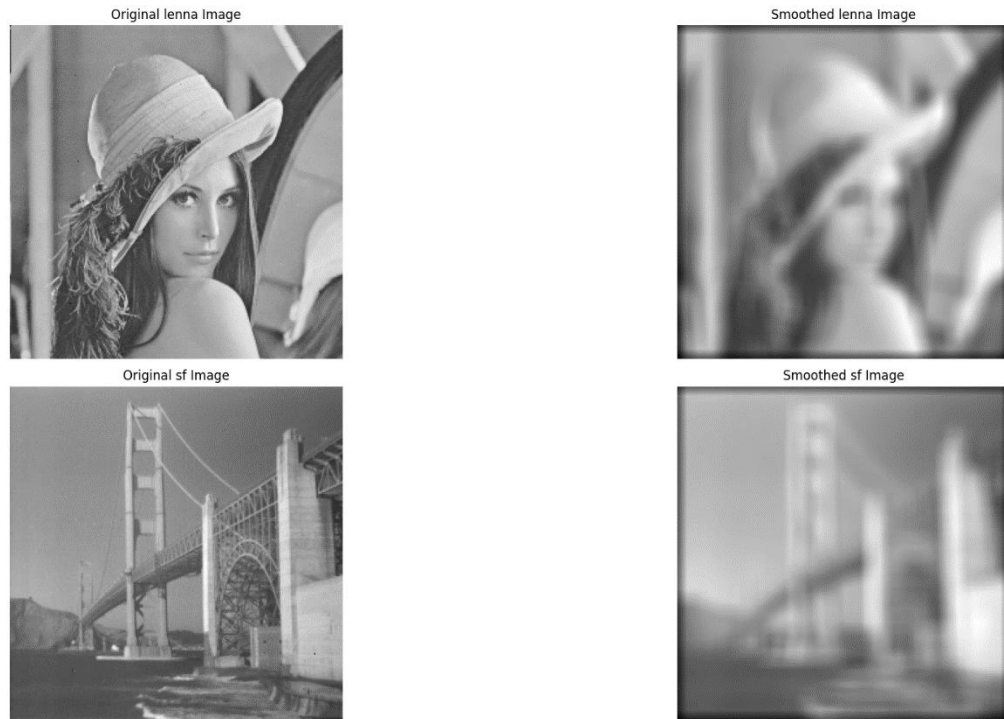


Figure 16: Top 1: Input Lenna image. **Top 2:** Smoothed Lenna image using Averaging 15x15 mask.
Bottom 1: Input sf image. **Bottom 2:** Smoothed sf image using Averaging 15x15 mask.

3.2.2 Smoothing Lenna and sf image using 7x7 and 15x15 Gaussian filter.

This section 3.2.2 shows the result of Smoothing after applying the 7x7 and 15x15 Gaussian filter masks (Provided in question) on Lenna and sf images. As you can see in Figure 17 and 18, first we give input Lenna and sf image. We applied 7x7 and 15x15 Gaussian Filtering masks for smoothing.

As we understand in section 2.2 of chapter 2, Gaussian preserves the details of the image and results are better than Averaging filter. As a result, we can see that in figures 17 and 18 that after applying the 7x7 and 15x15 Gaussian filtering mask on input Lenna and sf images, it smoothed the image and preserves the better detail of the image

compared to Averaging filter. Compared to 7x7 masks, the results we got after applying 15x15 Gaussian masks are more blurred Lenna and sf image.



Figure 17: Top 1: Input Lenna image. **Top 2:** Smoothed Lenna image using Gaussian 7x7 mask. **Bottom 1:** Input sf image. **Bottom 2:** Smoothed sf image using Gaussian 7x7 mask.



Figure 18: Top 1: Input Lenna image. **Top 2:** Smoothed Lenna image using Gaussian 15x15 mask.
Bottom 1: Input sf image. **Bottom 2:** Smoothed sf image using Gaussian 15x15 mask.

3.3 Median Filtering:

3.3.1 Implementation of Median Filtering :

Implementing median filtering involves processing each pixel in an image by replacing its intensity value with the median value of the pixel values within a local neighborhood or kernel. As you can see in figure 19, 20 and 21, we first applied salt and pepper noise 30% and 50% on original boat and Lenna image and corrupt it. With noise 50% it will degrade the quality of image compared to 30%.

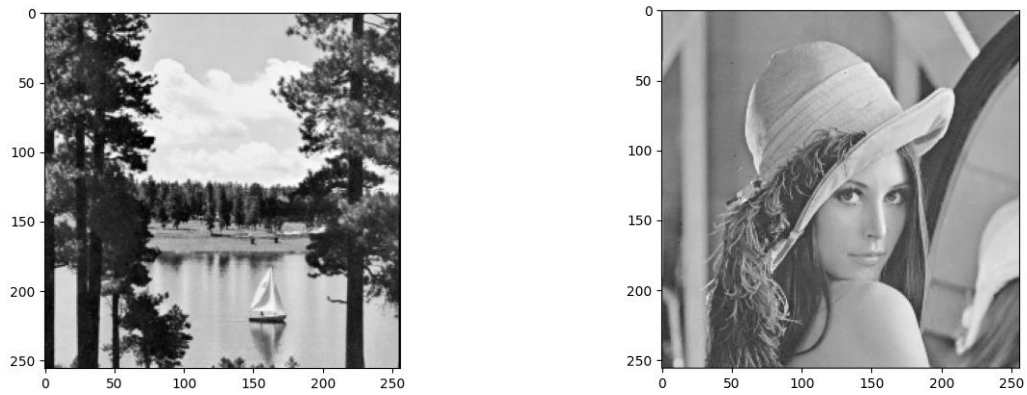


Figure 19: Original Image of boat and Lenna

3.3.2 Corrupt Lenna and boat Image with salt and pepper noise

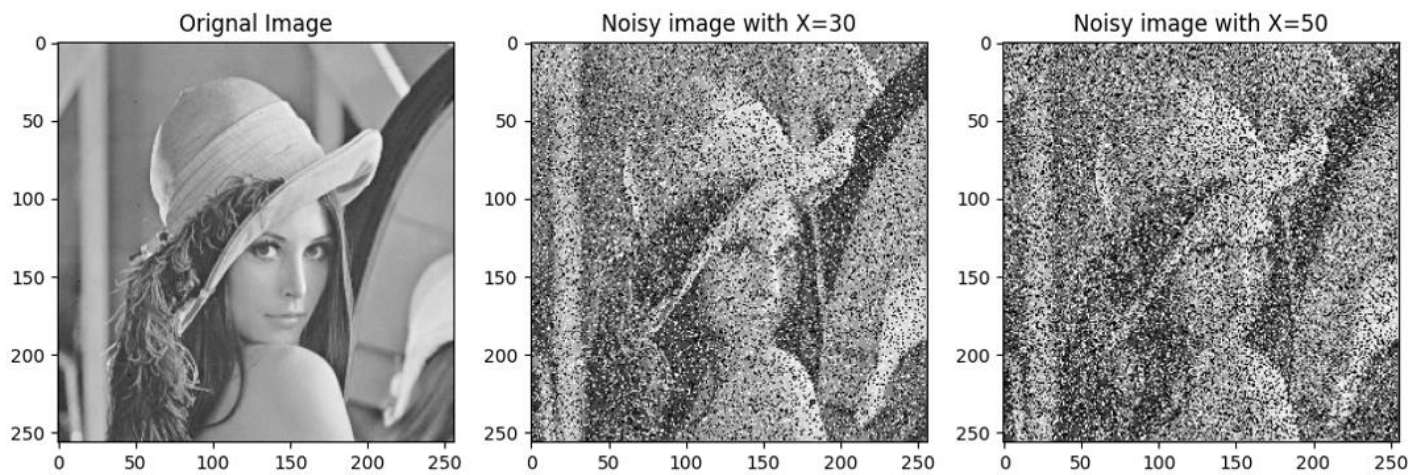


Figure 20: Image Showing Lenna salt and pepper noise with noise 30% and 50%

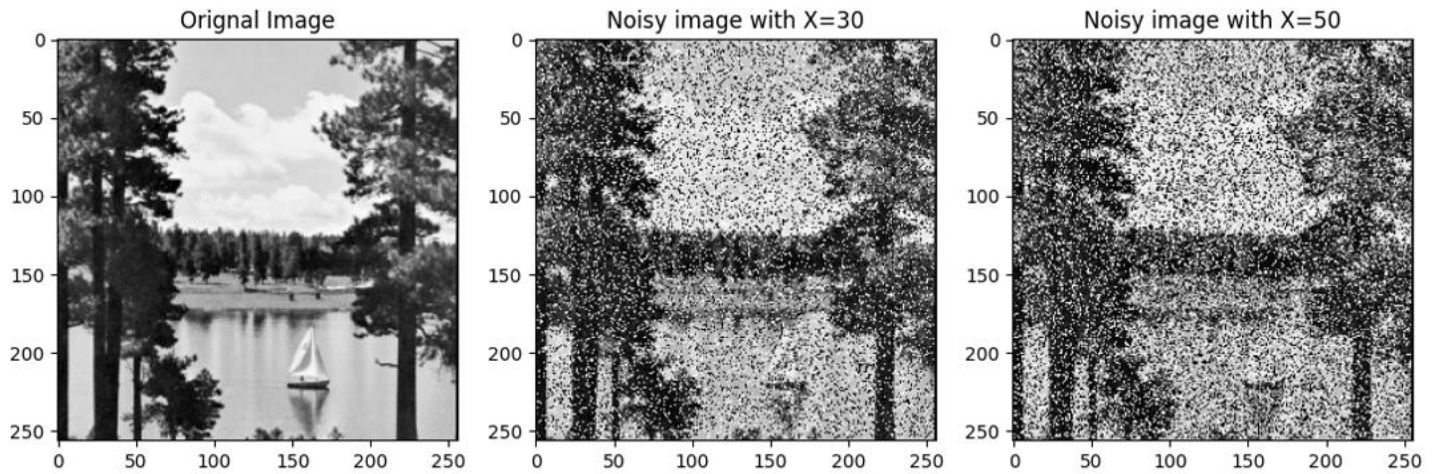


Figure 21: Image Showing Boat salt and pepper noise with noise 30% and 50%

We have applied the function to introduce salt and pepper noise into the image. The random occurrence of white and black pixels observed in the image is commonly referred to as "salt and pepper noise."

3.3.3 Median Filtering on corrupted Lenna and Boat Image

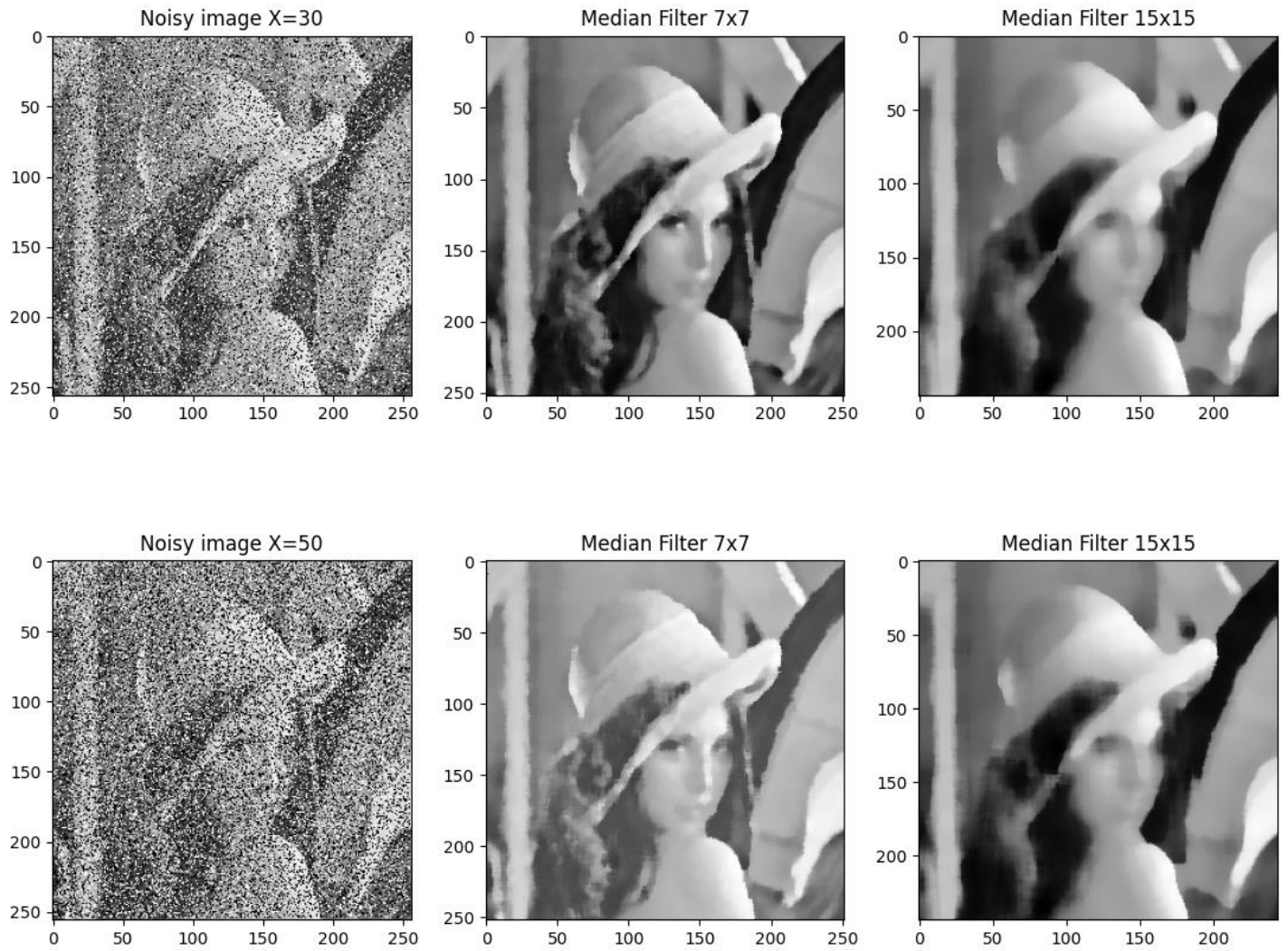


Figure 22: Lenna image with 30% noise and 50% noise with 7x7 and 15x15 median Filter.

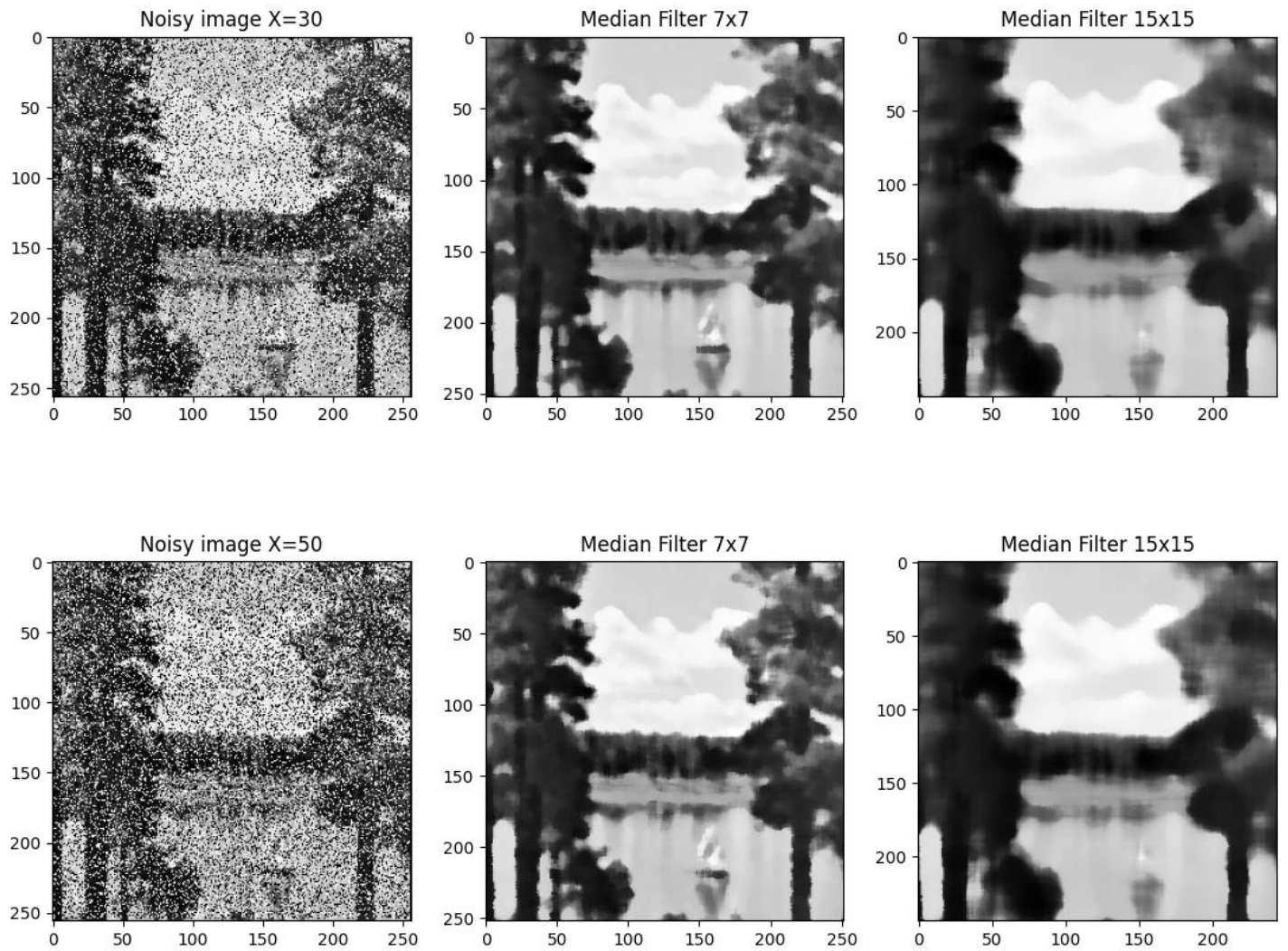


Figure 23: Boat Image with 30% noise and 50% noise with 7x7 and 15x15 median Filter.

In the initial stage, we used the original Lenna and boat images. We then applied salt and pepper noise to both images using a specific function.

For the third task, as you can see in figure 22 and 23, we applied median filtering to the Lenna and boat noisy images. We conducted two experiments: one with 30% noise and 7x7 filter size, and another with 30% noise and 15x15 filter size. Additionally, we tested the same scenarios with 50% noise. Here are the results:

1. 30% Noise and 7x7 Filter:

- The median filter performed well, effectively removing noise while preserving image quality.

2. 30% Noise and 15x15 Filter:

- With a larger filter size, the median filter still reduced noise but distorted the image slightly.

3. 50% Noise and 7x7 Filter:

- The median filter worked reasonably well, providing noise reduction and some image quality retention.

4. 50% Noise and 15x15 Filter:

- In this case, with a larger filter size, the median filter struggled to maintain image quality while removing noise.

In conclusion, the effectiveness of median filtering depends on the filter size. Smaller filter sizes yield better results, especially when dealing with high noise levels.

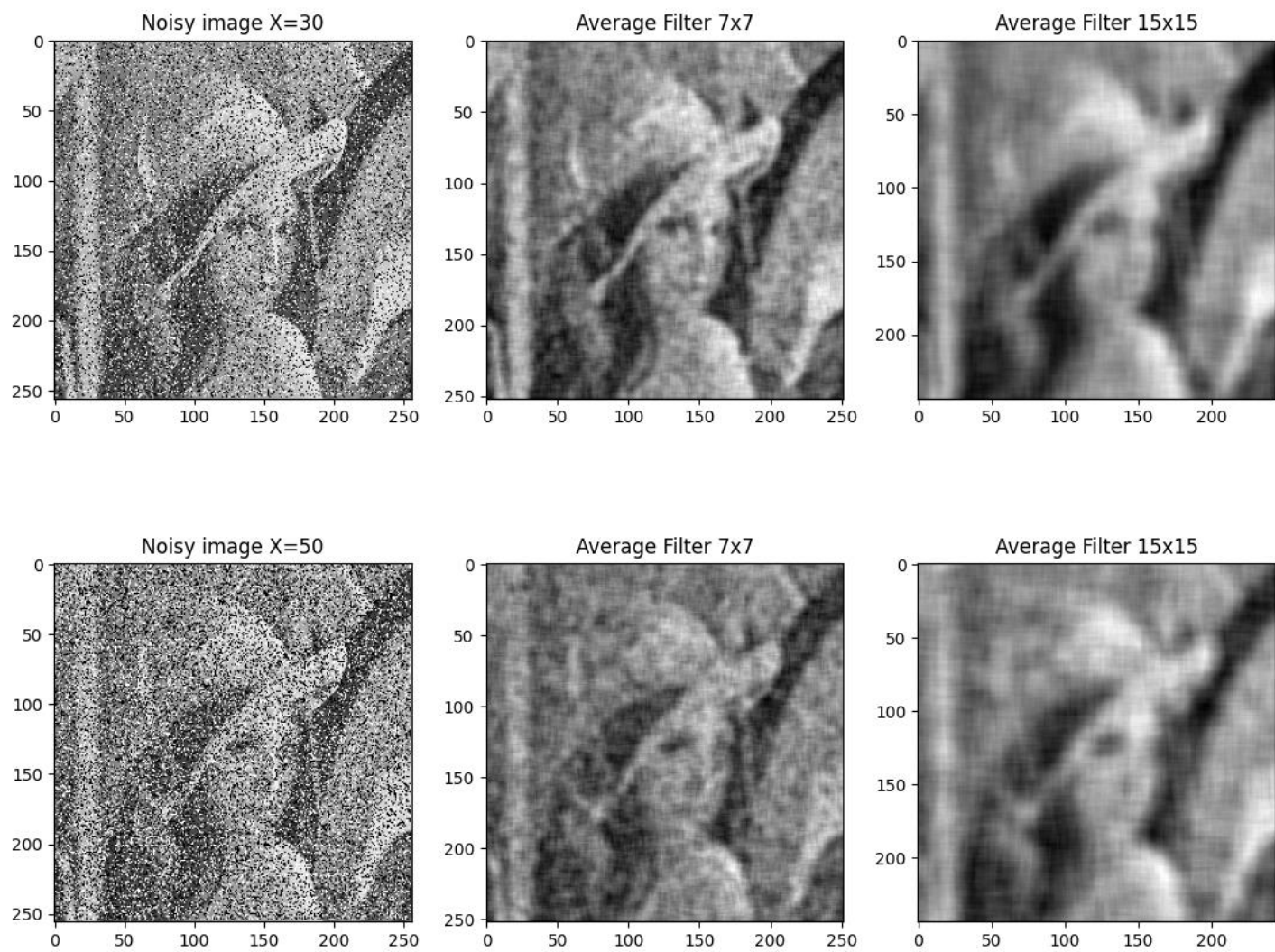


Figure 24: Lenna image with 30% and 50% noise and 7x7 and 15x15 averaging Filter.

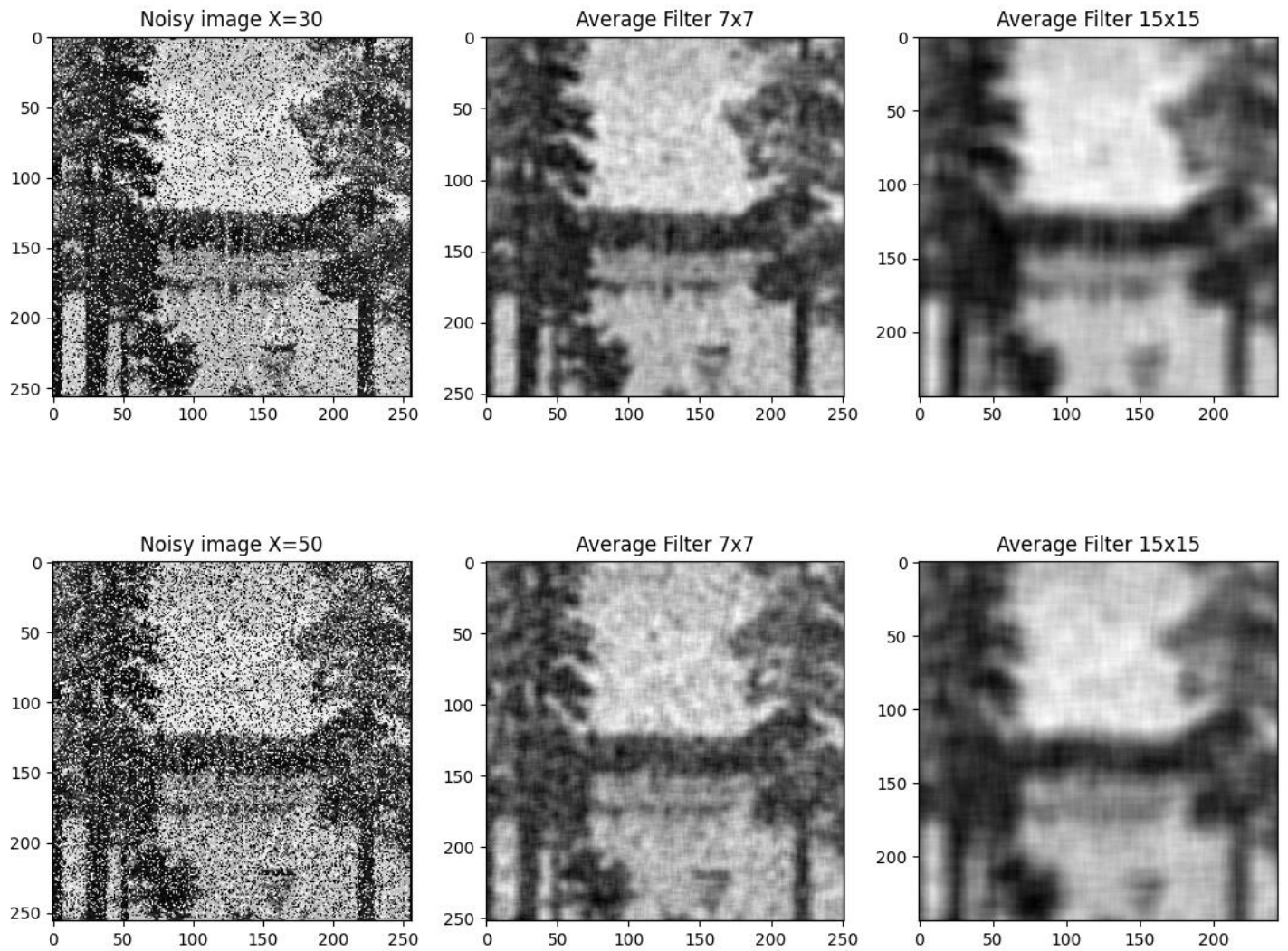


Figure 25: Boat image with 30% and 50% noise and 7x7 and 15x15 averaging Filter.

3.3.4 Median Filter and Average Filter Comparison

When comparing median filtering to average filtering, median filtering generally outperforms average filtering. As you can see in figure 24 and 25, The average filter struggles to remove all noise and may distort the image geometry, while median filtering effectively removes noise and preserves image quality.

3.4 Unsharp masking and High boost Filtering:

3.4.1 Implement of unsharp masking and high boost Filtering

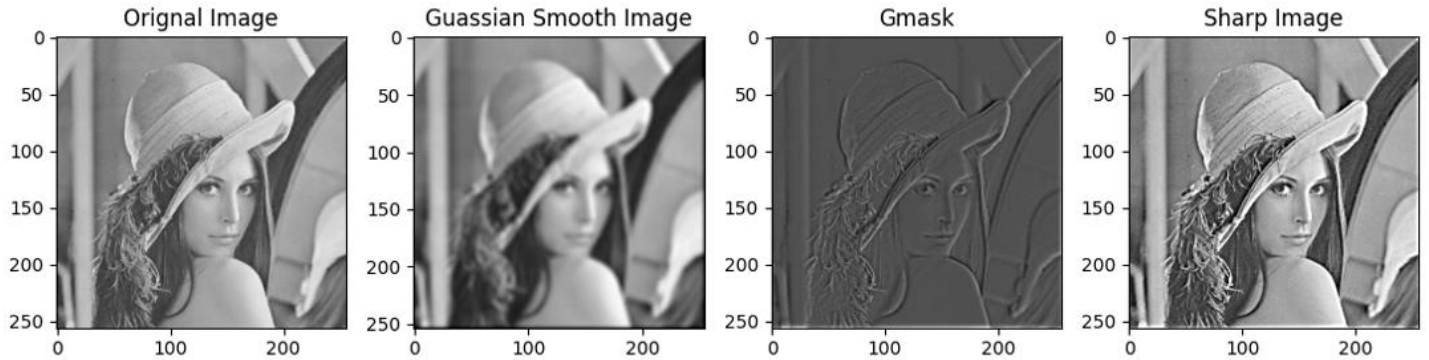


Figure 26: Unsharp masking performed on Lenna Image

In unsharp masking and High boost Filtering we take original image and subtract the smooth image from it then we add weighted (k) gmask to the original image to get the sharp image if the value of k is equal to 1 then the sharpening is called unsharp masking and if value of k is greater than 1 then the sharpening is called high boost Filtering. As you can see the results of unsharp masking on Lenna and f_16 images in figure 26 and 27, it sharpens the image. Figures 28 and 29 show the results of high boost filtering on Lenna and f_16 images and we experiment it with different values of k . As it shows better results as we increase the k value for sharpening.

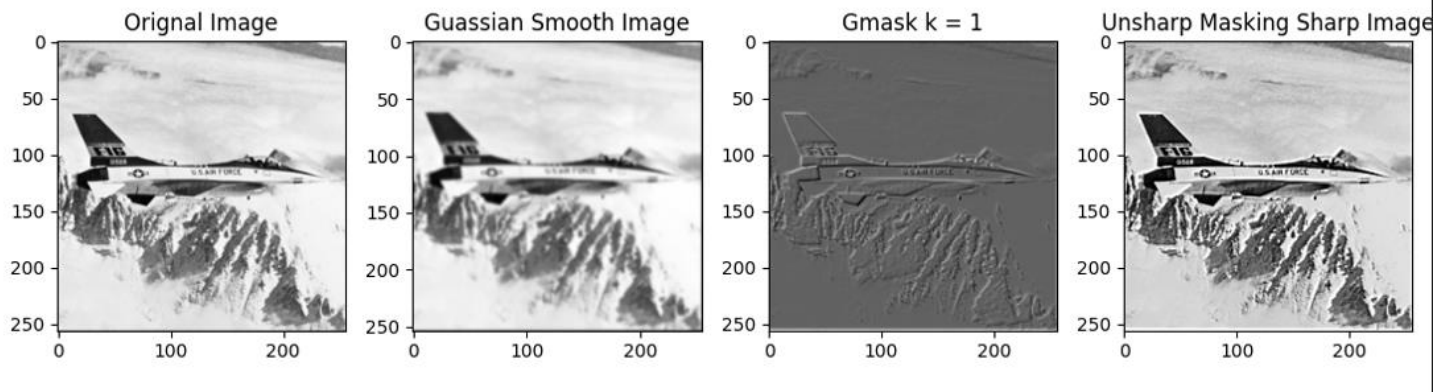


Figure 27: Unsharp masking performed on f_16 Image.

3.4.2 Experimentation with different K values

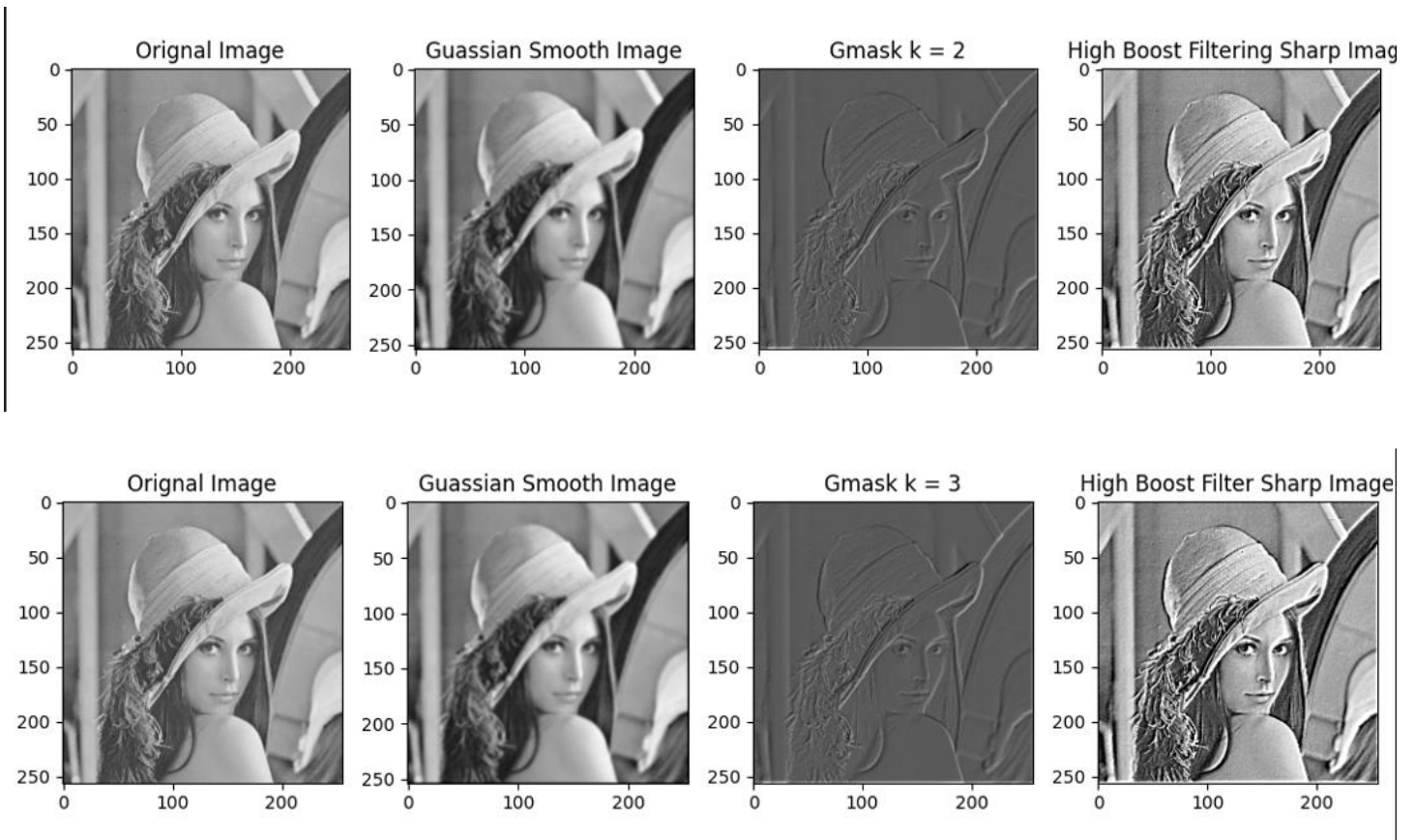
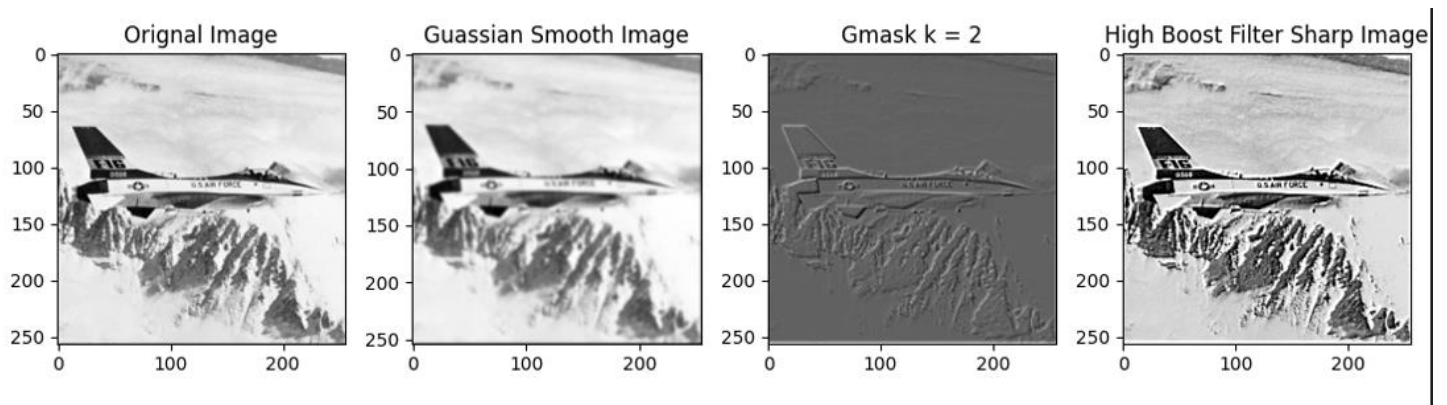


Figure 28: High Boost Filtering Performed on Lenna Image



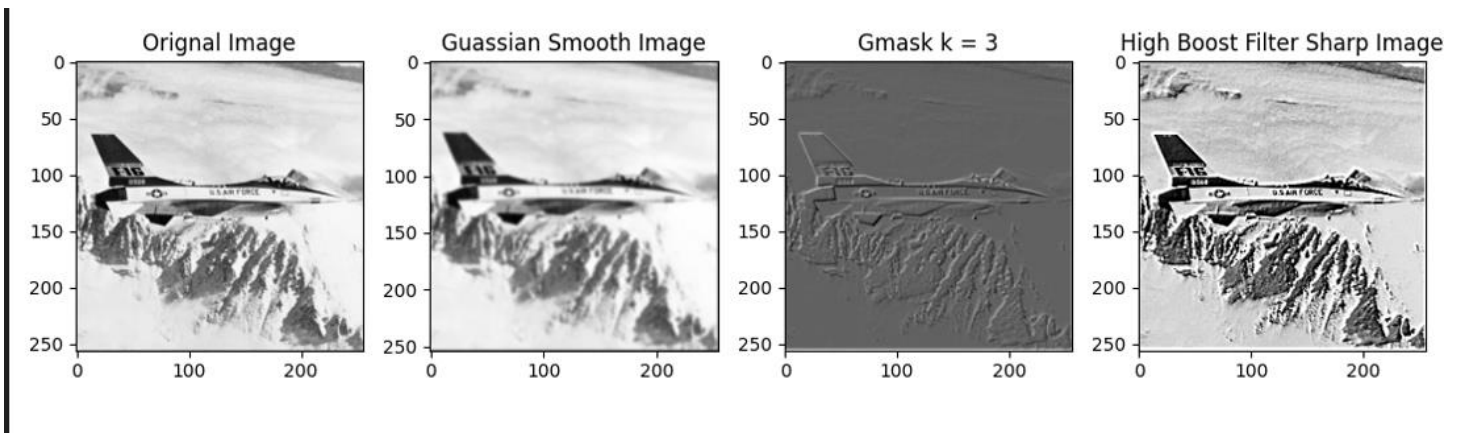


Figure 29: High Boost Filtering Performed on F_16 Image

3.5 Gradient and Laplacian:

3.5.1 Sharpening Lenna and sf image using Prewitt, Sobel, and Laplacian masks.

This section 3.5.1 shows the result of Sharpening after applying the Prewitt, Sobel and Laplacian masks on Lenna and sf images. As you can see in Figure 30, first we give input Lenna and sf image. We applied Prewitt, Sobel mask from gradient and Laplacian masks for Laplacian computation.

As we understand in section 2.5 of chapter 2, gradient uses two masks for x and y direction computation. Laplacian uses single masks for x and y direction computation.

The results show that Laplacian localizes the edges better as it performs 2nd derivative on image. Laplacian provides the magnitude information but no information about edge direction. While gradient perform 1st derivative and here, we used Prewitt and Sobel masks on input Lenna image and sf image. It performs the 1st derivative for computation on input images. Two masks are used for x and y direction computation in gradient using Prewitt and Sobel.

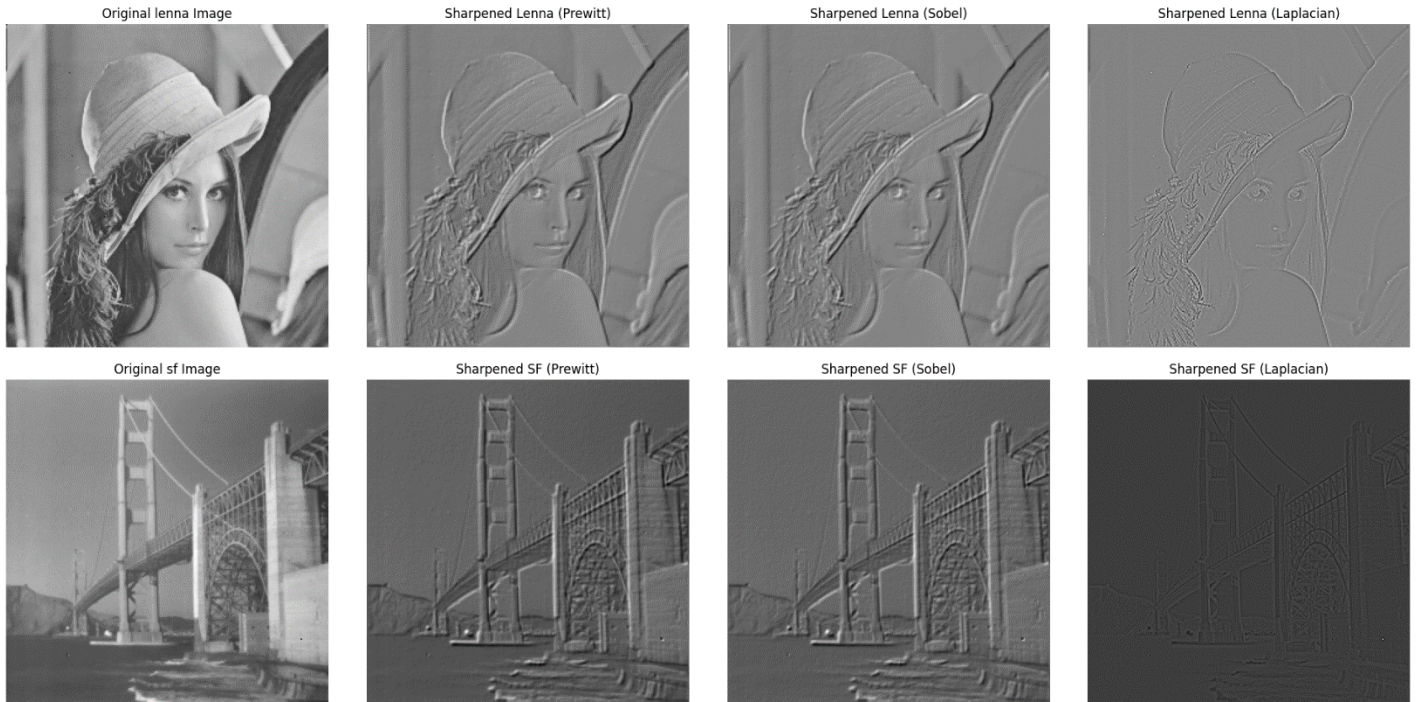


Figure 30: Top 1: Input Lenna image. **Top 2:** Sharpened Lenna using Prewitt masks. **Top 3:** Sharpened Lenna using Sobel masks. **Top 4:** Sharpened Lenna using Laplacian mask. **Bottom 1:** Input sf image. **Bottom 2:** Sharpened sf using Prewitt masks. **Bottom 3:** Sharpened sf using Sobel masks. **Bottom 4:** Sharpened sf using Laplacian mask.

3.5.2 Gradient Magnitude and Partial Derivatives

This section 3.5.2 shows the result of Gradient magnitude and partial derivatives of Lenna and sf image after applying the Prewitt and Sobel masks. Prewitt and Sobel masks use 2 masks in X and Y direction for computation. As you can see in figure 31, the Results after Partial derivatives applied on x and y direction of input Lenna image using Prewitt masks. The Gradient magnitude provides information about the strength of the edges of the Lenna image. The results of gradient magnitude in both cases Prewitt and Sobel masks provide the edge strength of the Lenna image.

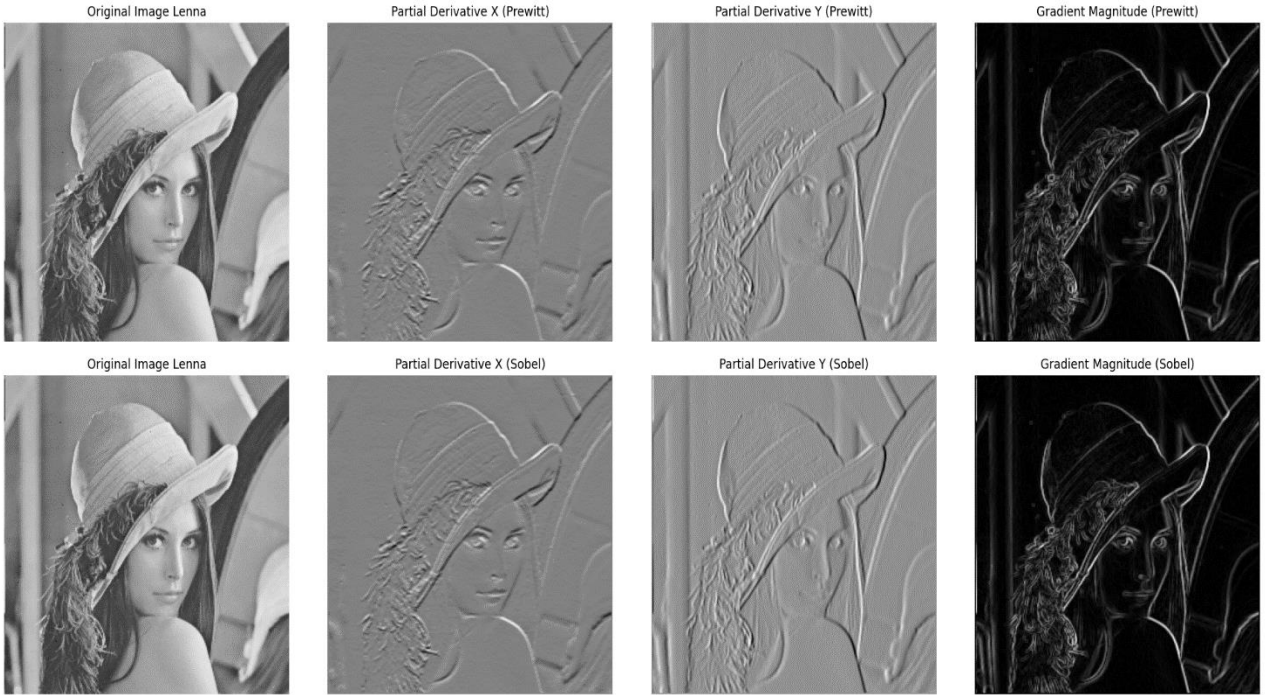


Figure 31: Top 1: Input Lenna image. **Top 2:** Partial derivative X on Lenna image using Prewitt masks. **Top 3:** Partial derivative Y on Lenna image using Prewitt masks. **Top 4:** Gradient Magnitude of Lenna using Prewitt mask. **Bottom 1:** Input Lenna image. **Bottom 2:** Partial derivative X on Lenna image using Sobel masks. **Bottom 3:** Partial derivative Y on Lenna image using Sobel masks. **Bottom 4:** Gradient Magnitude of Lenna using Sobel mask.

As you can see in figure 32, the Results after Partial derivatives applied on x and y direction of input sf image using Prewitt masks. The Gradient magnitude provides information about the strength of the edges of the sf image. The results of gradient magnitude in both cases Prewitt and Sobel masks provide the edge strength of the sf image.

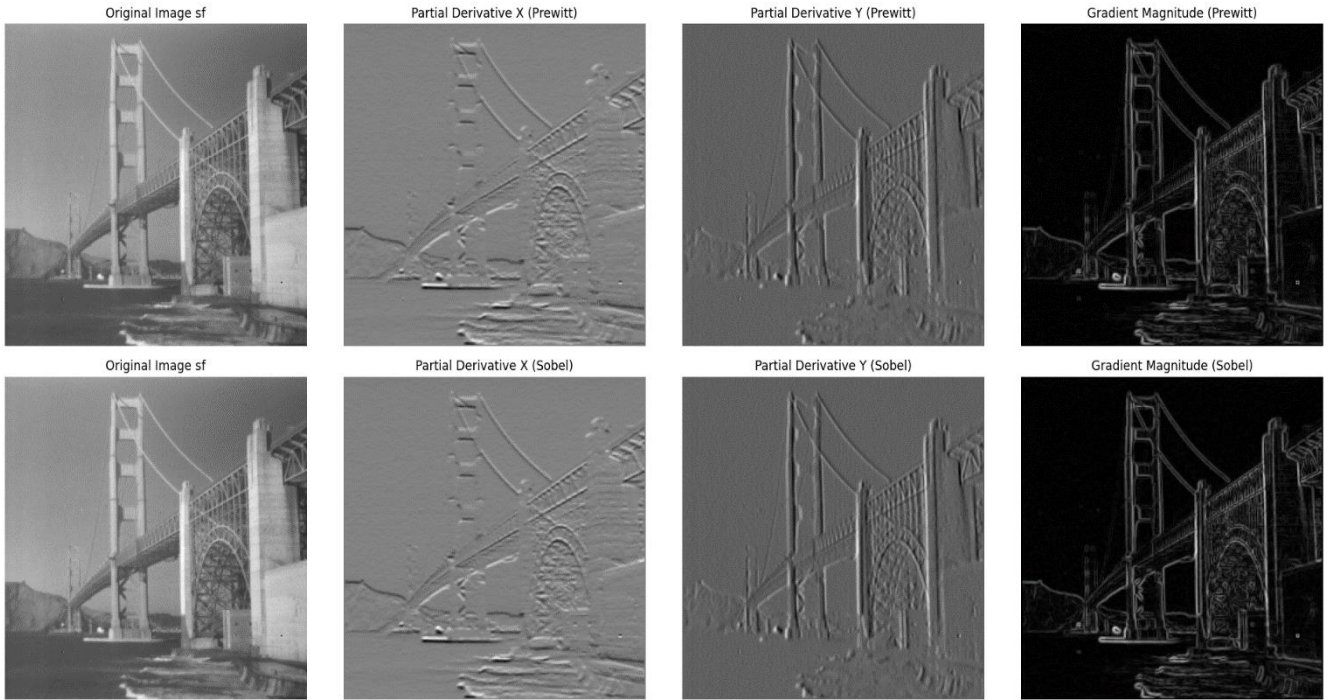


Figure 32: Top 1: Input sf image. **Top 2:** Partial derivative X on sf image using Prewitt masks. **Top 3:** Partial derivative Y on sf image using Prewitt masks. **Top 4:** Gradient Magnitude of sf using Prewitt mask. **Bottom 1:** Input sf image. **Bottom 2:** Partial derivative X on sf image using Sobel masks. **Bottom 3:** Partial derivative Y on sf image using Sobel masks. **Bottom 4:** Gradient Magnitude of sf using Sobel mask.

CHAPTER 4. REFERENCES

- 1) <https://www.cse.unr.edu/~bebis/CS474/>
- 2) R. Gonzalez and R. Woods [Digital Image Processing](#), 4th edition, Pearson, 2018.