Indian Institute of Science Education and Research Bhopal

Computer Vision(DSE/EECS-312)

Assignment-1: Answer Sheet

Name: Aditya Mishra
Roll No.: 21013
Date of submission: September 12, 2024 Marks Obtained:

Please follow the instructions given in the assignment carefully.

Please provide your detailed answers and any explanations or diagrams directly below each question in the 'Answers' section.

1. Apply the filters mentioned below on the image attached and analyze their impact. Describe what you found after applying each filter and why certain phenomena are happening. ***(Marks:)***

1.

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

2.

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

Since the above filters are of dimension 3×3. Construct the same filters of dimension 5×5 and do the above experiments.

Answer: The filter 1 is responsible for detecting vertical edges in an image. When the filter is placed on an image, it calculates the difference across columns (difference of the intensity of pixels to the right of current pixel and the intensity of pixels to the left of current pixel) and ignores the middle column (column with 0s ignores the current pixel). Figure 1 demonstrates its working by detection the vertical edges of the given image.

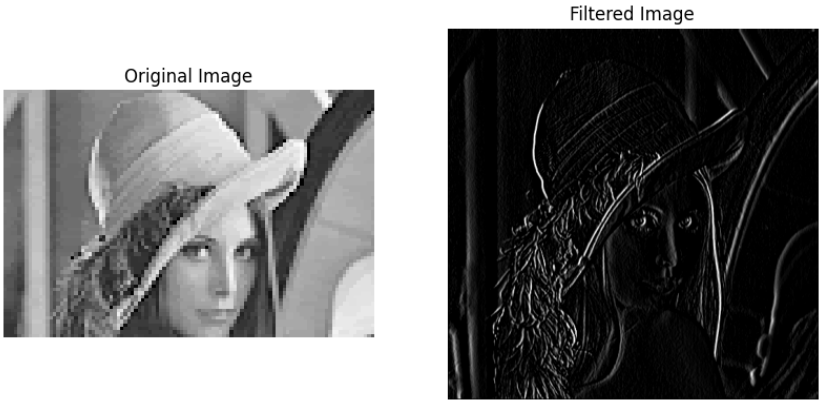

Figure 1: Vertical edge detection using 3×3 filter.

Similarly, the filter 2 is used for detecting the horizontal edges in an image. When the filter is placed on an image, it calculates the difference across rows (difference of the intensity of pixels above the current pixel and the intensity of pixels below the current pixel) and ignores the middle row (row with 0s ignores the current pixel). The working of horizontal edge detection filter is illustrated by Figure 2.

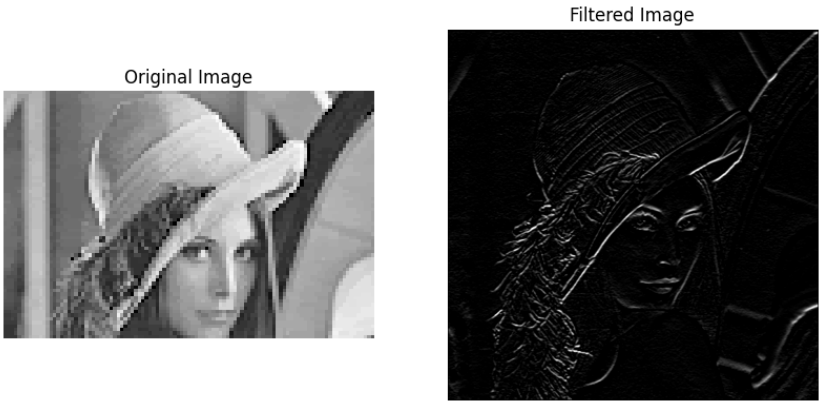

Figure 2: Horizontal edge detection using 3×3 filter.

The 3×3 vertical and horizontal edge detection filter can be extended to 5×5 dimension in the following way:

3.

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

4.

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

These 5×5 vertical and horizontal edge detection filters are different from the 3×3 filters as they analyze a larger area of 25 pixels as compared to 9 pixels of 3×3 filters. The broader area make these filters sensitive to edges that span a wider region and tends to smooth out small fluctuations in pixel intensity. Figure 3 demonstrates the working of 5×5 vertical edge detection filter and Figure 4 shows the working of 5×5 horizontal edge detection filter. It is evident from the figures that they are less sensitive to fine details or noise.The edges detected by a 5×5 filters appears to be smoother and more gradual, while the 3×3 filter produces sharper and more abrupt transitions in the filtered image.

2. Sobel Edge Detection Implementation. ***(Marks:)***

   (a) Write a Python function to apply the Sobel filter given below to detect edges along the x-direction and y-direction. Combine these to compute the gradient magnitude image.

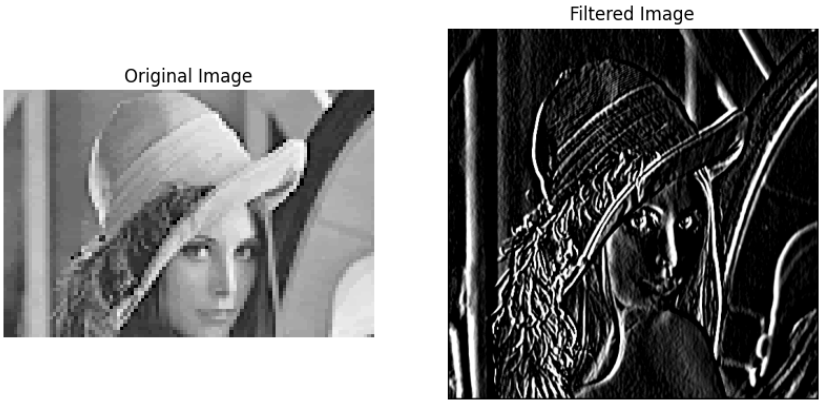

Figure 3: Vertical edge detection using 5×5 filter.

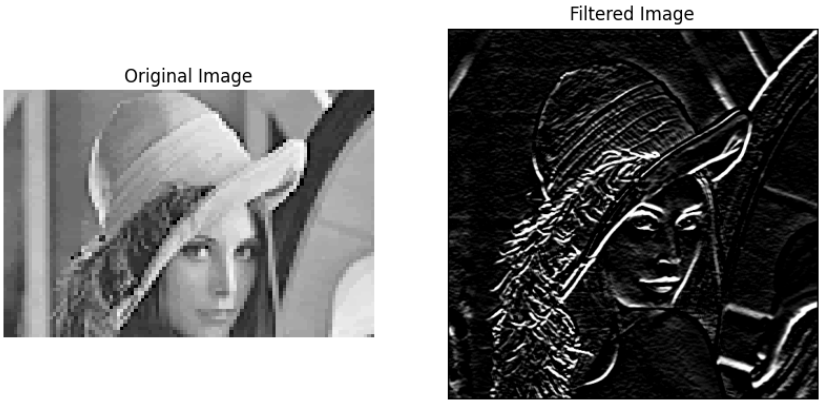

Figure 4: Horizontal edge detection using 5×5 filter.

1.

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

2.

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

3.

1	2	3	2	1
2	3	5	3	2
0	0	0	0	0
-2	-3	-5	-3	-2
-1	-2	-3	-2	-1

4.

-1	-2	0	2	1
-2	-3	0	3	2
-3	-5	0	5	3
-2	-3	0	3	2
-1	-2	0	2	1

(b) Apply your function to an image and display the gradient magnitude image alongside the original. Vary the size of the kernel and document the effects.

(c) Manually implement thresholding on the gradient magnitude to create a binary edge image. Experiment with different thresholds and show the results.

(d) Apply the Sobel edge detector on a noisy image (you may add synthetic noise to an attached clean image). Discuss how noise affects edge detection and the visual quality of the output images.

Answer:

(a) Sobel filter is commonly used operator for edge detection in computer vision and image processing. It has two kernels, one for detecting edges in the x-direction and another for detecting edges in the y-direction. It also comes in different sizes. Given above are 3×3 and 5×5 filters. Filter 1 is responsible for detecting vertical edges and Filter 2 is used to detect horizontal edges. Figure 5 illustrates the working of filter 1 and 2 on the given image by detecting vertical and horizontal edges. Filter 1 (x-direction Sobel filter) is responsible for recognizing the vertical edges in the image and the Filter 2 (y-direction Sobel Filter) is used to detect horizontal edges in the given image. Then they are combined to compute the gradient magnitude image using the equation 1:

$$GM = \sqrt{G_x^2 + G_y^2} \tag{1}$$

where G_x and G_y denotes the gradient along x-direction and y-direction, respectively.

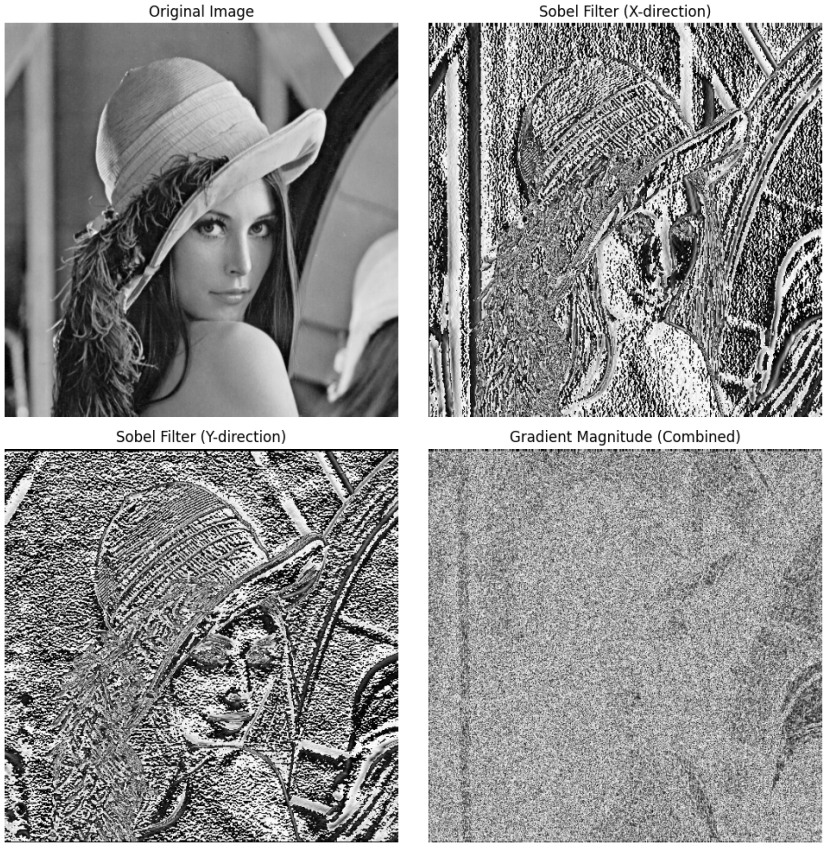

Figure 5: Filter 1 and 2: 1 is used to detect vertical edges (x-direction Sobel filter) and 2 detects horizontal edges (y-direction Sobel filter).

Similarly, Filter 3 is a 5×5 x-direction Sobel filter responsible for detecting vertical edges and Filter 4 is 5×5 y-direction Sobel filter used to detect horizontal edges. Figure 6 demonstrates the working of Filter 3 and Filter on the given image by

detecting horizontal and vertical edges and then uses gradient in x-direction and y-direction to compute the gradient magnitude image using equation 1.

It is observed from the Figure 5 and 6 that 3×3 Sobel filter captures fine-grained edges and high-frequency details. This results in sharper, more defined edges, but it also introduces more noise and less smoothness. On the other hand, 5×5 Sobel filter provides a smoother and more continuous edge detection. It filters average out more information, so the edges appear smoother and less noisy, but the fine details are blurred compared to the 3×3 filter.

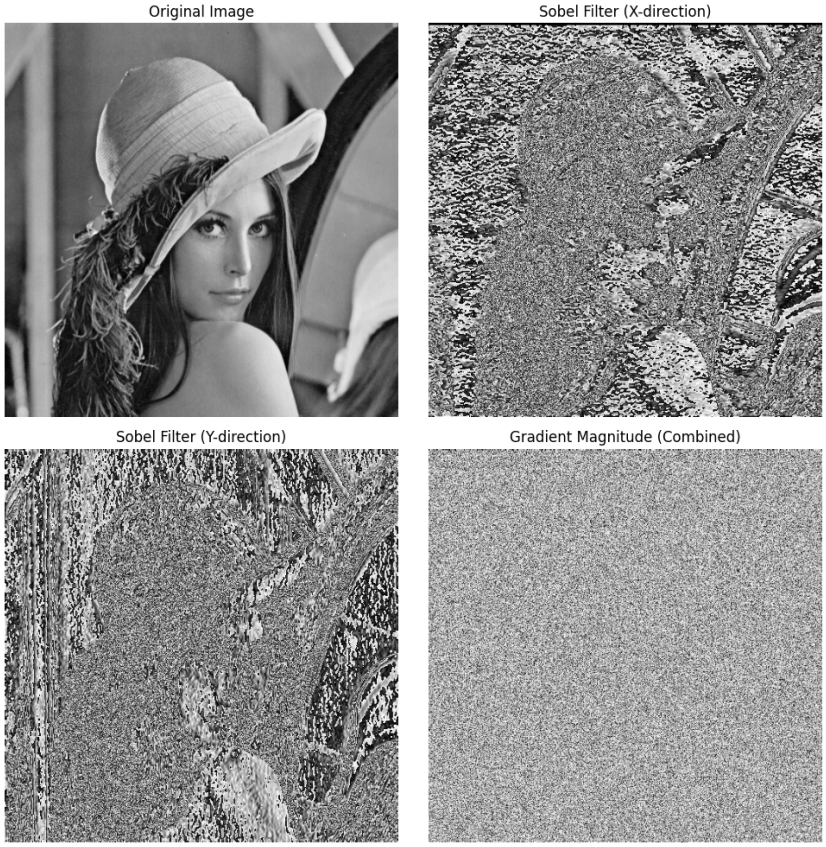

Figure 6: Filter 3 and 4 are 5×5 Sobel filter used for detecting vertical edges (x-direction) and horizontal edges (y-direction), respectively.

(b) In part (a), we already applied function to the given image and displayed the gradient magnitude image alongside the original. We also, display the changes in the Figure 7. Figure 7 showcases the gradient magnitude image computed from 3×3 and 5×5 Sobel filter. The gradient magnitude image created by the 3×3 Sobel filter appears noisier and shows more high-frequency variations. Conversely, the gradient magnitude image computed by 5×5 Sobel filter is smoother, showing fewer high-frequency variations.

(c) A binary image is an image where each pixel is either black (0) or white (255). This results from thresholding an image, where pixels with values below the threshold are set to 0 (black) and those above the threshold are set to 255 (white). Figure 8, 9, 10 and 11 showcases the manual implementation of thresholding on the gradient magnitude to create a binary edge image. The different value of threshold chosen by me are: 5, 10, 15 and 20. Upon choosing the threshold value of 20, we get completely black image. It suggests that there are no pixels having value greater

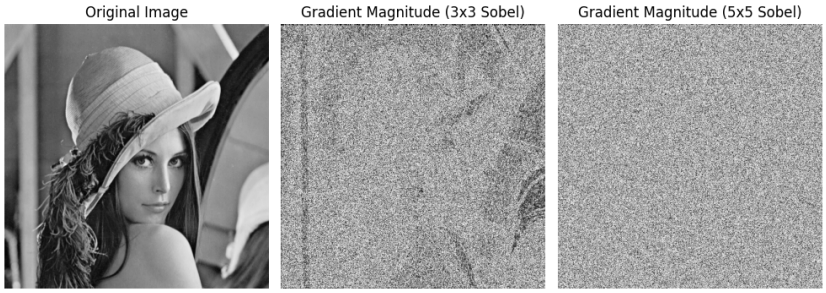

Figure 7: Visualization of different Sobel filters in computing gradient magnitude image along with original image.

or equal to 20. Upon choosing the value of threshold among 5, 10 and 15, we see binary images. The binary image with threshold of 5 (Figure 8) have majority of white pixels, showing that majority of the pixels have value more than 5.

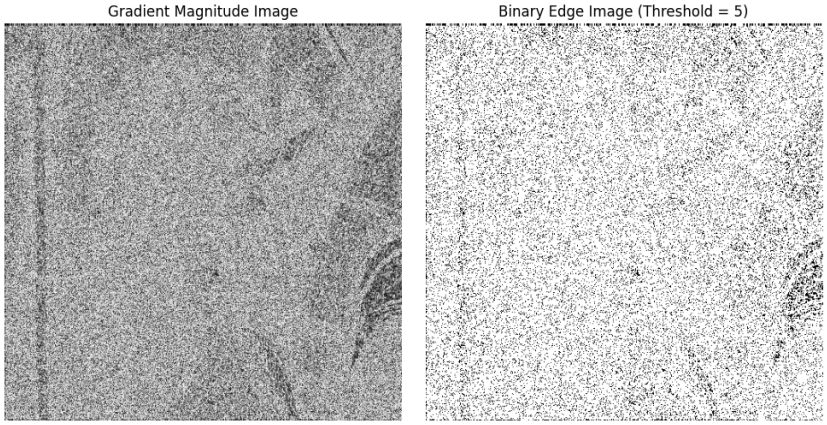

Figure 8: Gradient magnitude image to binary edge with threshold of 5.

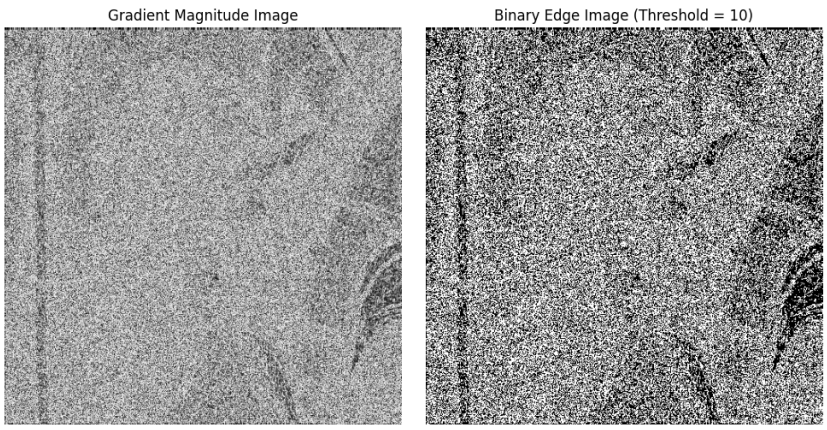

Figure 9: Gradient magnitude image to binary edge with threshold of 10.

(d) We choose to introduce Gaussian Noise and then use 3×3 x-direction and y-direction Sobel filter for vertical and horizontal edge detection, respectively. Figure 12 demonstrates the introduction of Gaussian noise and then edge detection using Sobel filter. Original Image displays the natural features and textures without any added noise. On the other hand, Gaussian noise introduces random variations in

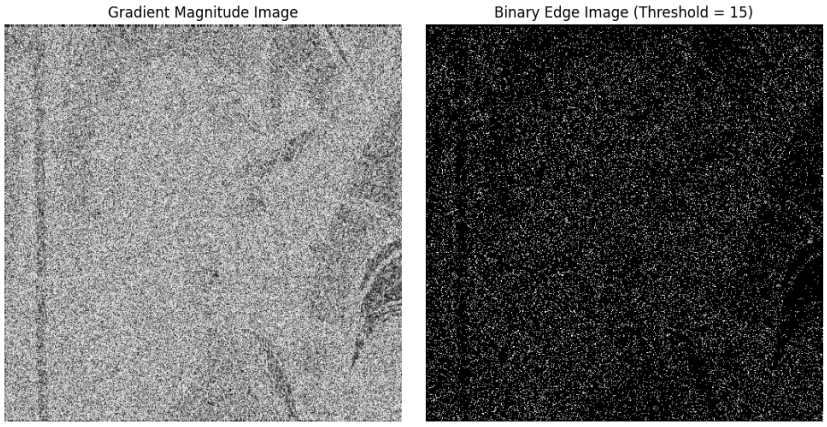

Figure 10: Gradient magnitude image to binary edge with threshold of 15.

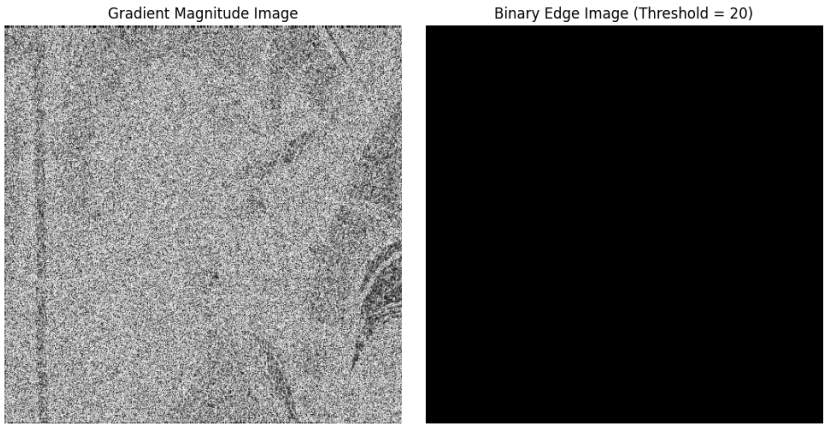

Figure 11: Gradient magnitude image to binary edge with threshold of 20.

pixel values, leading to grainy pixels. This noise hide the details and make the image appear vague.

It is observed from Figure 12 that adding Gaussian noise to the image reduces the clarity of the edges detected by the Sobel filter. The noisy image introduces artifacts that interferes with edge detection and lead to a less precise representation of edges. The Sobel filter still highlight areas with high gradients, but the additional noise causes the detected edges to be less accurate.

3. Laplacian of Gaussian Edge Detection (follow the class notes). ***(Marks:)***
   (a) Implement Gaussian smoothing from scratch. Apply your Gaussian filter given below to smooth an image before edge detection.
   (b) Develop the Laplacian filter and apply it to the smoothed image from 3 (a) to detect edges via zero-crossings. Describe how you detect zero-crossings in your implementation.
   (c) Display the edges detected from the smoothed image alongside the edges detected from the non-smoothed image. Discuss the differences and the impact of noise.

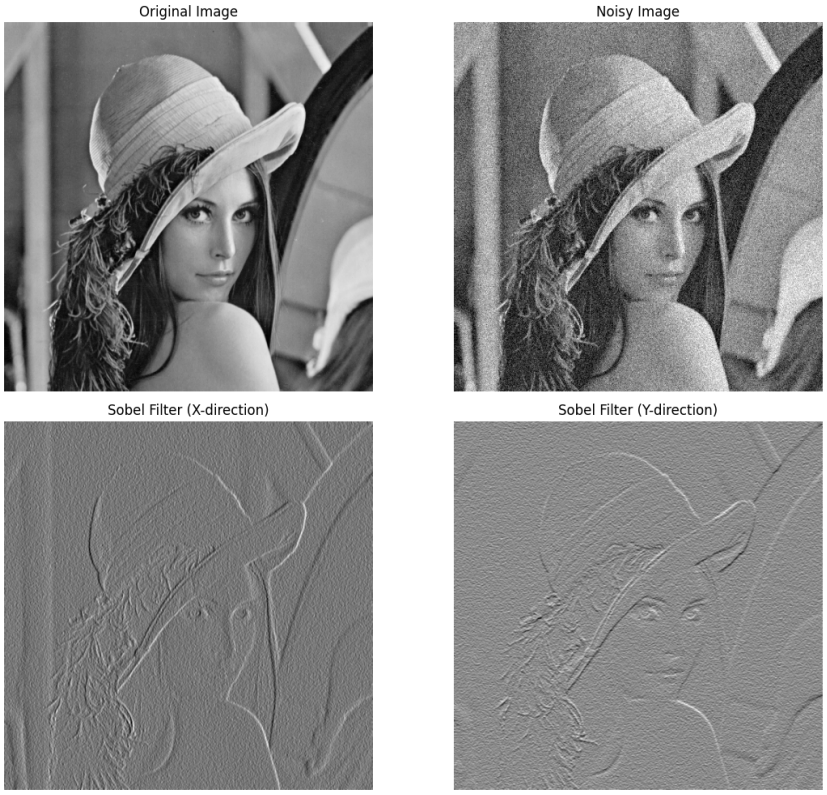

Figure 12: Edge detection using Sobel filter after introduction of Gaussian noise.

Answer:

(a) Gaussian filter is a linear filter that is used to blur an image. It is based on the Gaussian function (bell-shaped curve). It is particularly effective for reducing noise and detail in an image. Mathematically, it is denoted by equation 2

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{2}$$

Where G(x,y) represents the value of the filter at (x,y), σ is the standard deviation (determines the extent of the smoothing), and x and y are the pixel distances from the center of the filter. The smoothness can be clearly observed in the face and other parts of the image in Figure 13. The blur effect is clearly visible on the hairs of the lady in the image. Gaussian filter blurs the image and make it appear smoother. Both original and smoothed image has been provided for the comparison.

(b) Laplacian filter is a second-order derivative filter that is used for detecting edges by highlighting areas of rapid intensity change. It detects edges in all directions. The Laplacian equation is given by 3:

$$\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{3}$$

After applying the Laplacian filter to the smoothed image(from part (a)), edges are identified by locating zero-crossings in the Laplacian output. Zero-crossings are points where the Laplacian value changes sign by switching from positive to negative

Figure 13: Gaussian smoothing.

or vice versa. These transitions typically indicate the presence of edges in the given image. Figure 14 demonstrates the working of Laplacian filter.

The detection of zero-crossings can be understood by the following methodology:

1. In the first step, we locate surrounding pixels and examine each pixel's neighboring pixels (up, down, left, right, and diagonals) in the Laplacian-filtered image.
2. Secondly, we check for a change in sign, meaning that if a pixel is currently positive and any of its neighbors are negative (or vice versa), then a zero-crossing has occurred.
3. At last, we eliminate weak edges that can be the consequence of noise by applying a threshold to the absolute magnitude of the Laplacian values.

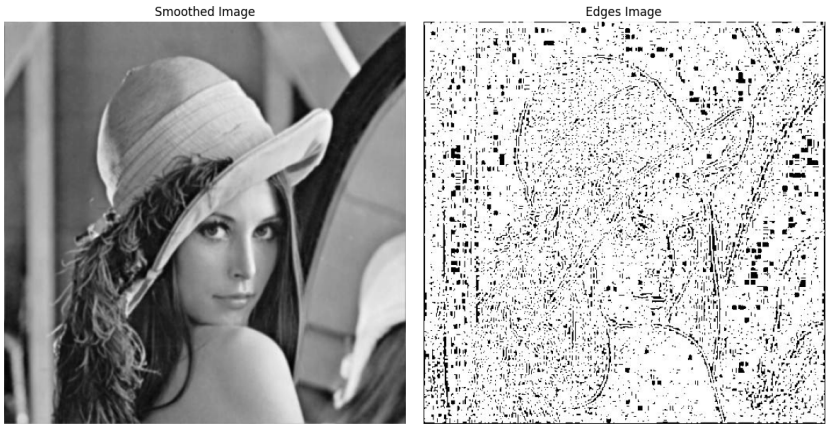

Figure 14: Edge Detection by zero-crossings.

(c) Figure 15 demonstrates the difference between non-smoothed (original image) and smoothed image (Gaussian smoothed image). Edge detection in Gaussian smoothed image is better as compared to original image because the noise gets eliminated upon Gaussian smoothing. Also, sharper boundaries can be seen in the case of Gaussian smoothed image. On the other hand, false edges are produced in the original image as a result of high-frequency noises.

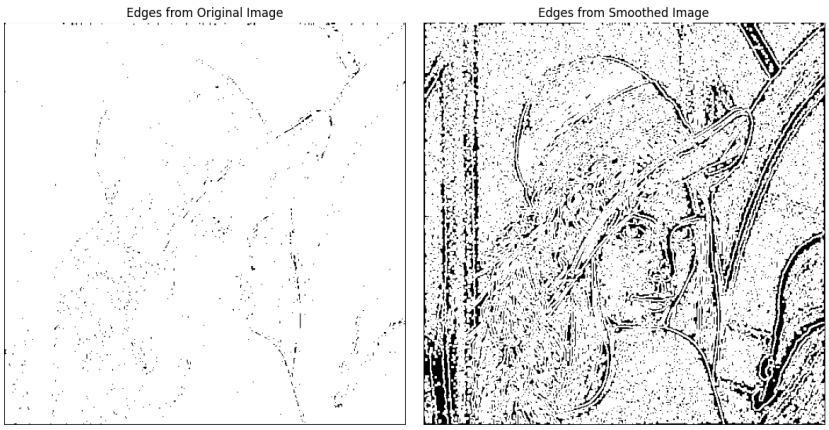

Figure 15: Zero-crossings edge detection in original and Gaussian smoothed image.

We summarize the difference observed in the edge detection from smoothed and original image below:

Table 1: Difference observed in the edge detection from smoothed and original image.

S. No.	Original Image	Smoothed Image
1	**Noise Sensitivity:** The original, non-smoothed image's edges are significantly noisier than the smoothed image in this case. Every small variation in pixel intensity caused by noise is frequently identified as an edge, creating a disordered edge map.	**Noise Reduction:** When we apply the Laplacian filter after the Gaussian smoothing filter, which produces a clearer and more consistent set of edges, the image's high-frequency noise is effectively reduced.
2	**False Edges:** False edges are produced by high-frequency noise, making it difficult to distinguish between noise artifacts and true object boundaries.	**Sharper Boundaries:** The primary object boundaries are preserved despite the noise reduction, resulting in a more defined edge map that shows the real outlines of the items in the image.
3	**Impact of noise:** An edge map that is noisy and fragmented is produced when noise is present because the Laplacian filter sees noise as edges.	**Impact of noise:** There are less false edges and a more precise identification of true boundaries when Gaussian smoothing is used, which essentially eliminates most of the noise before the Laplacian filter is applied. Though some fine features are lost in favor of noise reduction, the smoothed image offers more crisp edges.