# Digital Image Processing, 2024 Spring

# Homework 1

Name: Zhou Shouchen
Student ID: 2021533042

Due 23:59 (CST), Apr. 7, 2024

## Problem 1:

(a) Figure 1 is the histogram image of grain.tif.

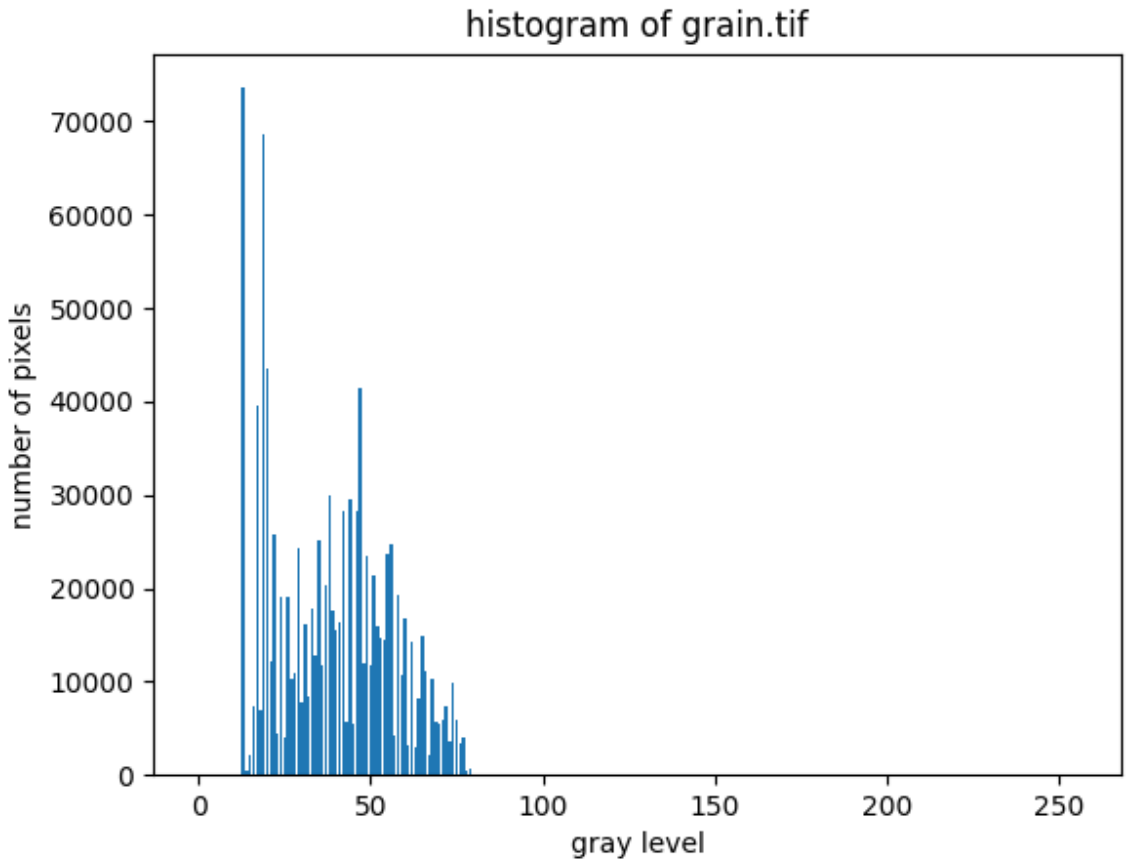


Figure 1. Histogram of grain.tif

(b) The left image of Figure 2 is the histogram equalized image, and the right one is the histogram of that histogram equalized image.

(c) The left image of Figure 3 is the CLAHE processed image, and the right one is the histogram of that CLAHE processed image.

And for the details of CLAHE, the image is padded with the edge values, since during the CLAHE process, it may need to access the pixels outside the image.

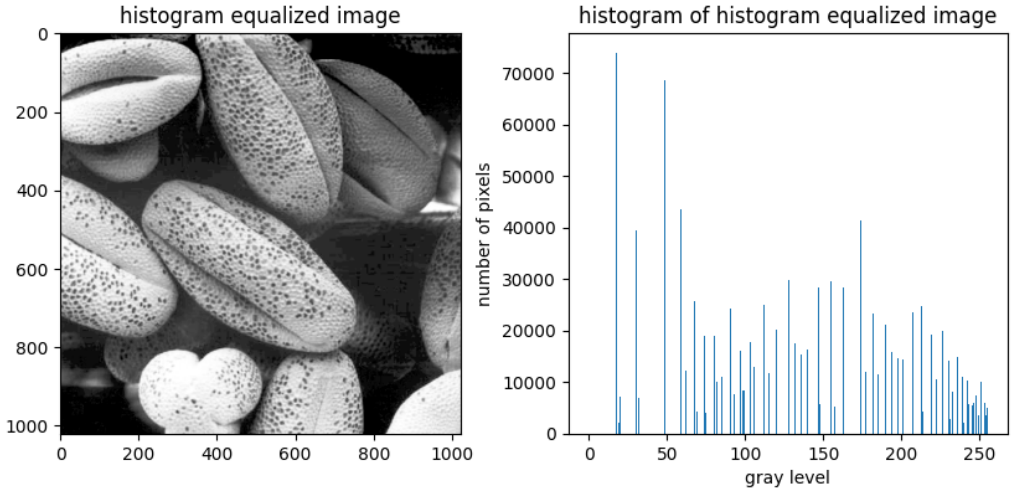


Figure 2. Histogram equalized image and its histogram

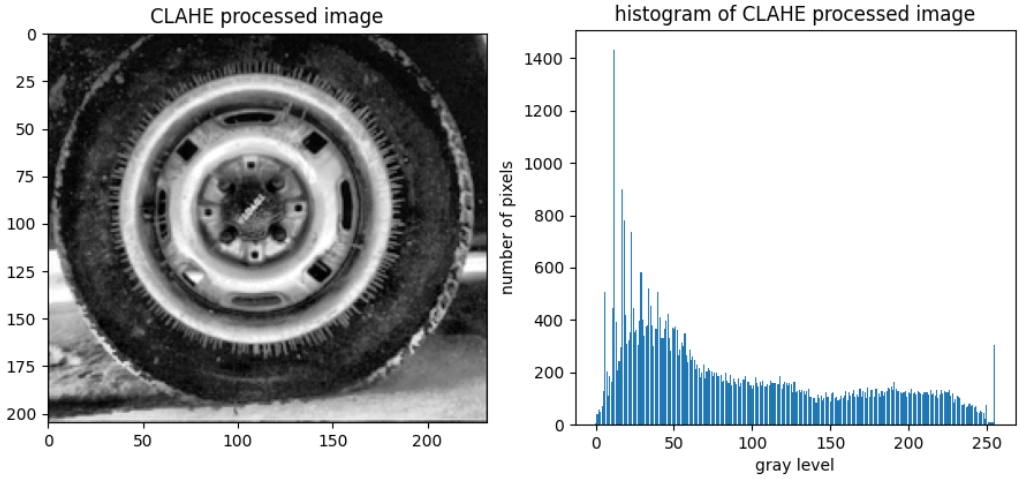


Figure 3. CLAHE processed image and its histogram

## Problem 2:

For the details of this problem: the image is padded with the edge values during the convolution, since during the convolution process, it may need to access the pixels outside the image.

(a) The Laplacian kernel is that:

$$\nabla^2 f = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

We can seperate the Laplcian kernel along the $x$-direction and $y$-direction, and we can simplify them into 1-D:

$$\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$$

The processed image corresponding to the kernel above all shown in Figure 4.

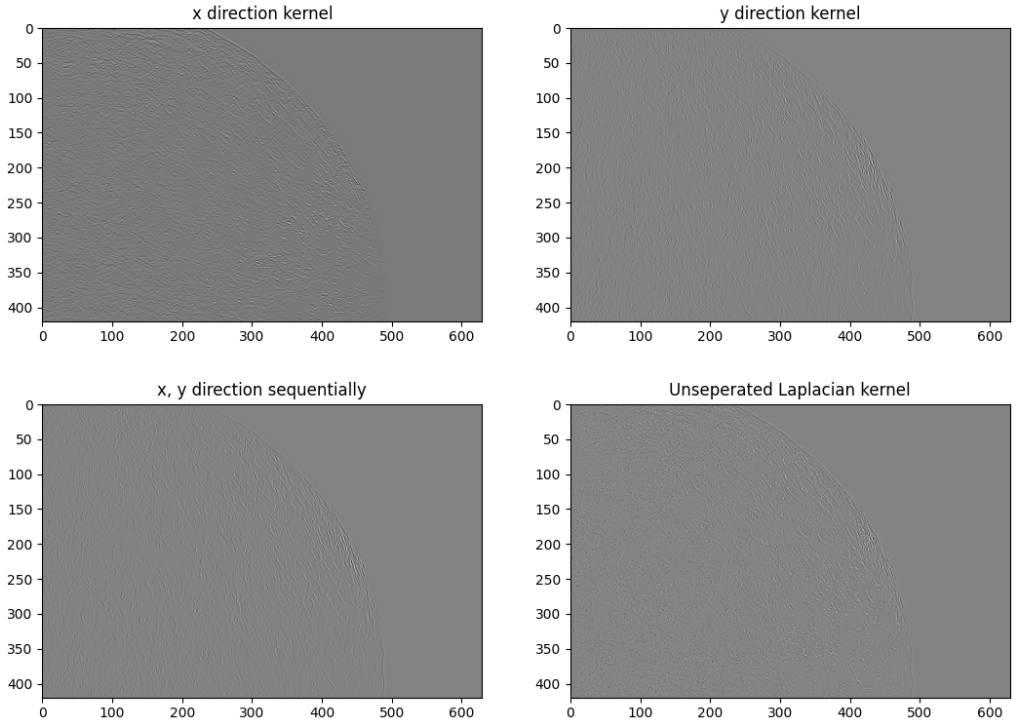


Figure 4. Separated Laplacian kernels processed image

(b) The Sharpened image with Laplacian kernel is shown in Figure 5.

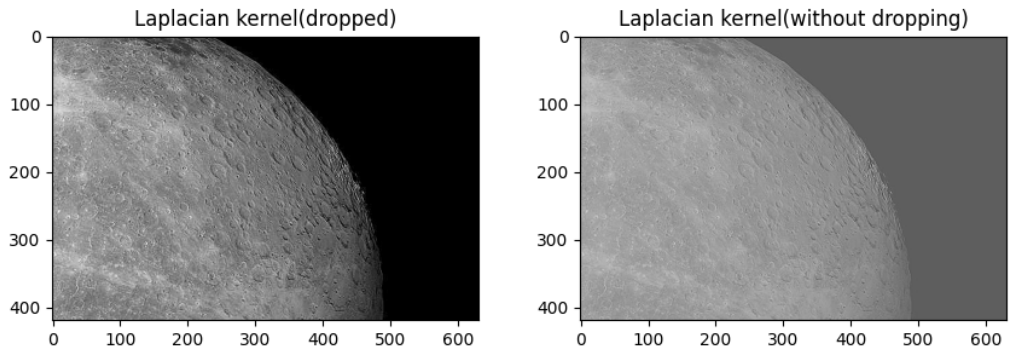


Figure 5. Unseparated Laplacian kernel processed image

The image sharpening with Laplacian could be seen as sharpened with the kernel

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

In order to make the right side of the image to be black instead of being gray, the $< 0$ and $> 255$ part are abandoned, i.e. turning the $< 0$ part into 0, and turning the $> 255$ part into 255, which is shown in the left image.
And the right image is doing no additional operations, slighting mapping $[\min, \max] \to [0, 255]$.

(c) The Sharpened image with unsharpen mask is shown in Figure 6.
The first two rows (marked filter1 in the title) are smoothed with the kernel

$$\text{kernel } 1 = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

And the last two rows (marked filter2 in the title) are smoothed with the kernel

$$\text{kernel } 2 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Suppose the origin image is $f(x, y)$.
The first column are the smoothed images processed by kernel1 and kernel2, mark as $\overline{f(x, y)}$.
The second column are the unsharped masks processed by kernel1 and kernel2. i.e. the difference between the origin image and the smoothed image. i.e. $g_{mask}(x, y) = f(x, y) - \overline{f(x, y)}$.
The third and the forth column are the sharpened images with different $k$. i.e. $g(x, y) = f(x, y) + k \cdot g_{mask}(x, y)$
The third column is the sharpened image with $k = 1$, and the forth column is the sharpened image with $k = 4.5$.

And the fifth column is the origin image.

For the difference bewteen the $1, 2$ rows and $3, 4$ rows is that: the $1, 3$ rows are doing normalization with $[\min, \max] \to [0, 255]$, and the $2, 4$ rows are setting $< 0$ and $> 255$ part are abandoned, i.e. turning the $< 0$ part into 0, and turning the $> 255$ part into 255.
We could see that as $k$ grows, more high frequency of the image increases, make the image sharper.

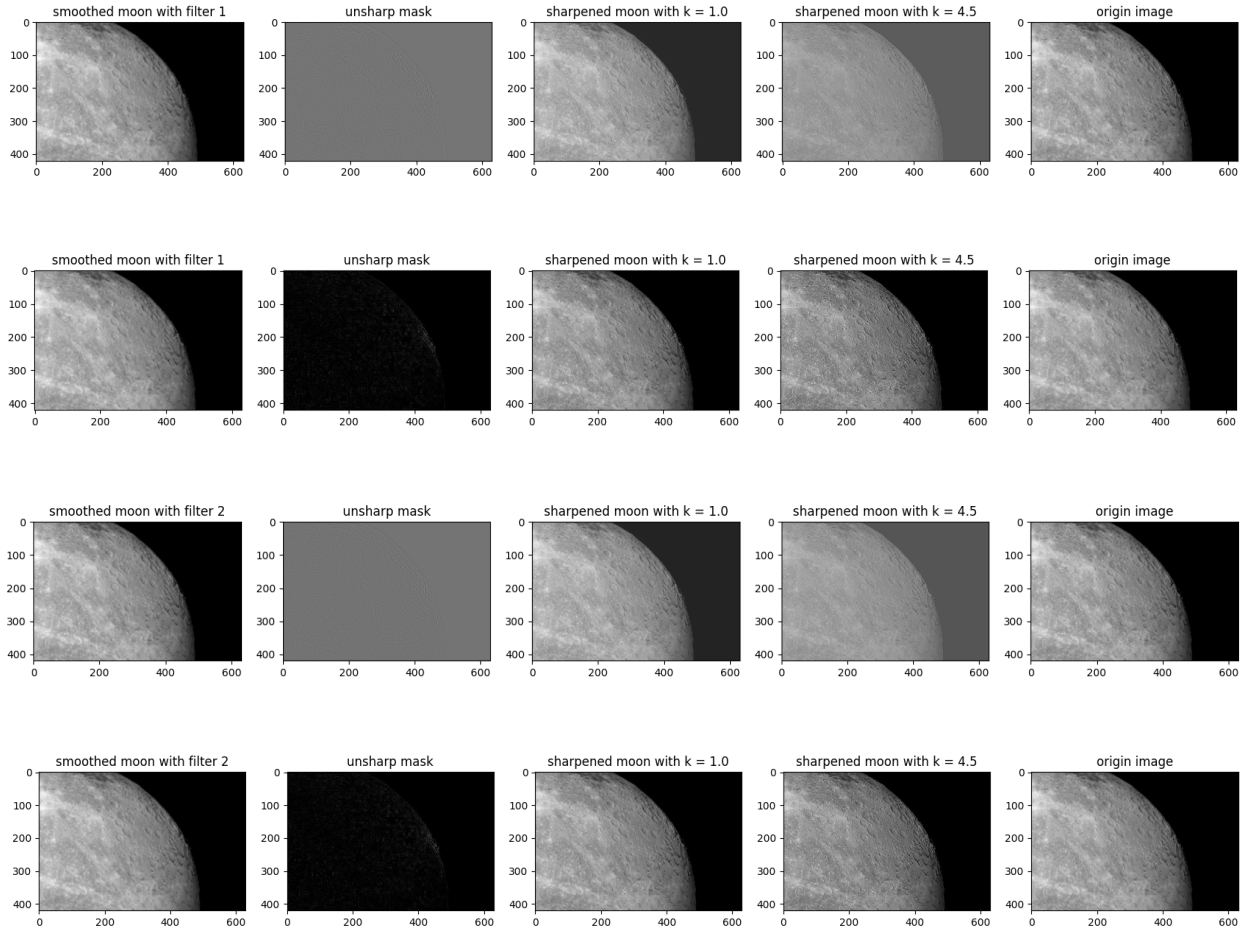


Figure 6. unsharpen mask processed image

## Problem 3:

For the details of this problem: the image is padded with the edge values during the convolution, since during the convolution process, it may need to access the pixels outside the image.

The processed image by the median filter is shown in Figure 7. The median filter is selected with the size of $3 \times 3, 5 \times 5, 7 \times 7$.

And we could see that the median kernel with size $3 \times 3$ has the best performance in this case.
The kernel has the proper size that remove the salt and pepper noise.

From the result, we could see that all the median filter processed images eliminate the salt and pepper noise, but the kernel size $3 \times 3$ has the best performance. The kernel size $5 \times 5$ and $7 \times 7$ make the processed images much blur than the $3 \times 3$ kernel size as they have the too big filter size to make information losses.

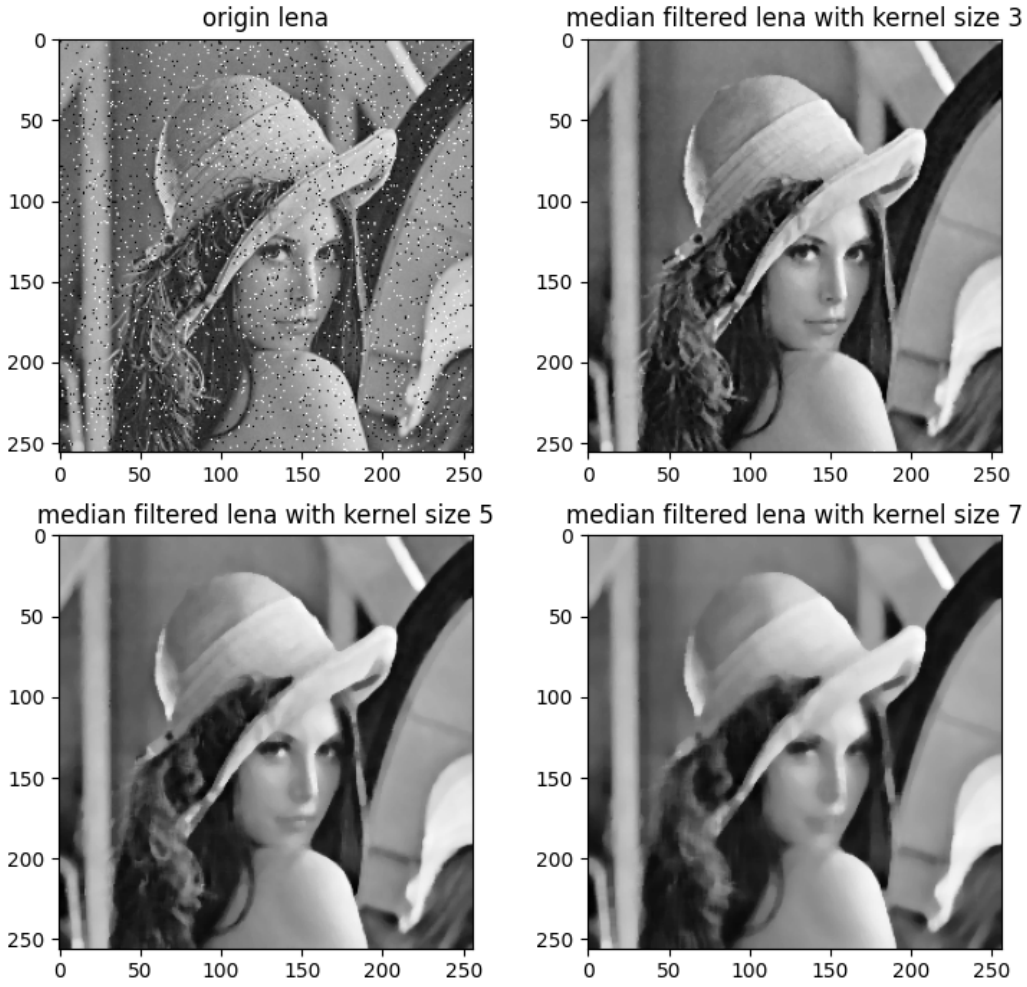


Figure 7. Median filter processed image

The processed image by the and Gaussian filter is shown in Figure 8.

Since the Gaussian filter's elements are $G(s,t) = Ke^{-\frac{s^2+t^2}{2\sigma^2}}$, but with the the normalize factor, the value of $K$ would be eliminate, so $K$ does not matter. So we can just take $K = 1$ and adjust the kernel size.

And we could see that no matter how the kernel size changes, the processed image is still blur than the median filter processed image. The Gaussian filter is not suitable for the salt and pepper noise removal.

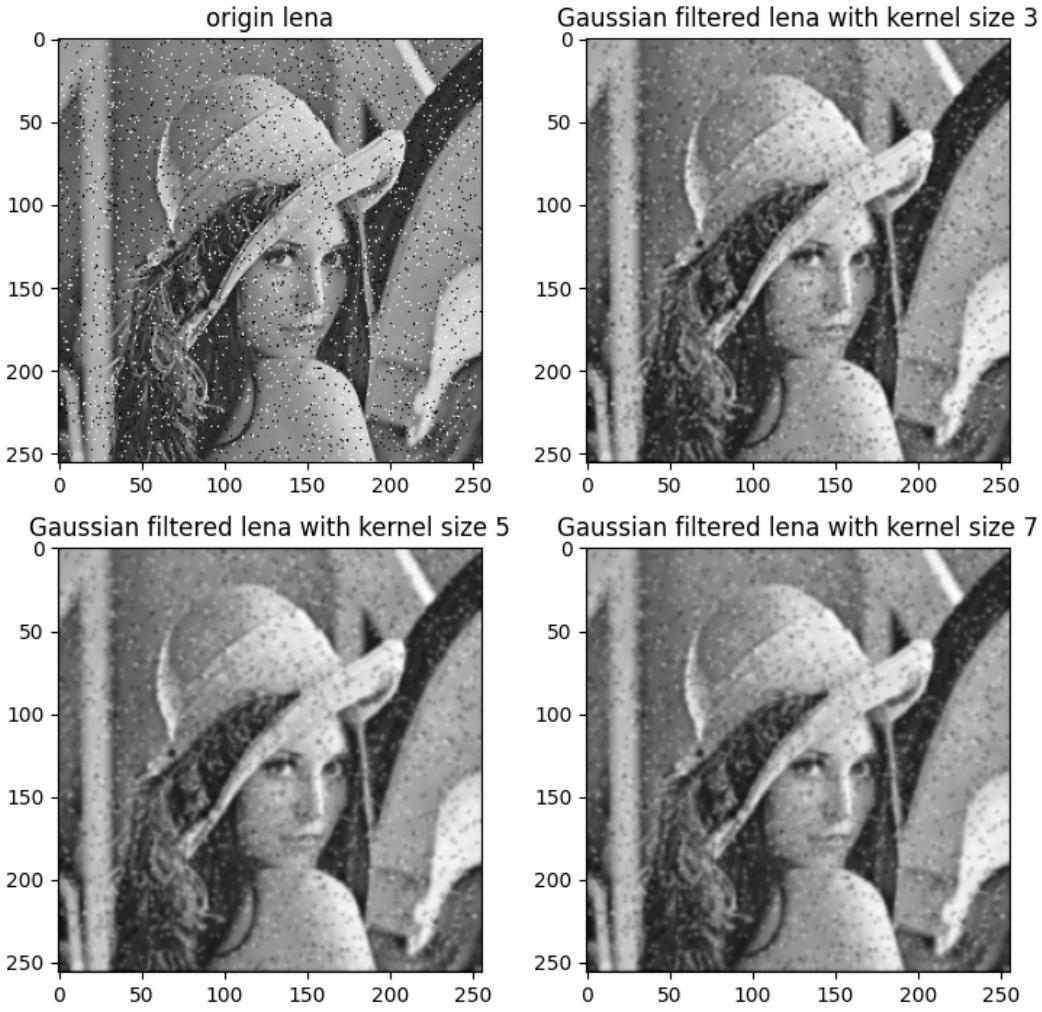


Figure 8. Gaussian filter processed image

Analyse:

Salt-and-pepper noise is an impulse noise. It is extramly high(255) or low(0) pixel values.

Median Filter:

The median filter works by replacing each pixel value with the median value of all pixel in the patch. Since it

is non-linear, it effectively removes noise. Since it replaces inpulse noise with the median value of the patch, it can perserve image's details and sharpness to recover the image from the noise.

The small-size filters ($3 \times 3$) can remove the noise while perserving the sharpness and details of the image. But for the high density of noise, it may not be sufficient to remove the noise, but in our Lena's case, it removes noises sufficiently.

The large-size filters ($5 \times 5$ or larger) may remove the high density of noise, but it may lead to the loss of image detail and make the image appear blurry.

Gaussian Filter:

The Gaussian filter is a linear smoothing filter that replaces each pixel value with a weighted average of the patch, the weights are normalized Gaussians. This gives the greatest weight to the central pixel, preserve the origin center's information sufficiently, and remove the noise by considering other pixels in the patch. It works well on the Gaussian noises, but it does not work very well in our case's salt and pepper noise, as it is a linear filter, and weight the most to the center pixel, so it cannot remove the salt-and-pepper noise sufficiently, and cause the image blur.

The kernel size will effect the amplitude of each pixels' weights. With larger kernel, the amplitude of each part, especially for the center part becomes less, so it cause the result image smoother, and cause the salt-and-pepper noise not so obvious. But as the kernel size gets larger, the image also get much blurrer. From ours results, it seems that $5 \times 5$ is better than the others, it somehow removes the salt-and-pepper noise, and make the image not too blur.

So above all:

Median filters are generally more suited for removing salt-and-pepper noise than Gaussian filters, especially when the noise density is high. And median filters also better preserve the sharpness and the recovery details.

The selection of kernel size should be case by case. Small kernel may not remove the noise sufficiently, while large kernel may cause the images loss the sharpness and details, cause the image blur. In this case, the $3 \times 3$ median filter is the best.