IMAGE PROCESSING

presented to:-

DR.Ayman Soliman

Presented by [**G1**]:-

Enji Hassan

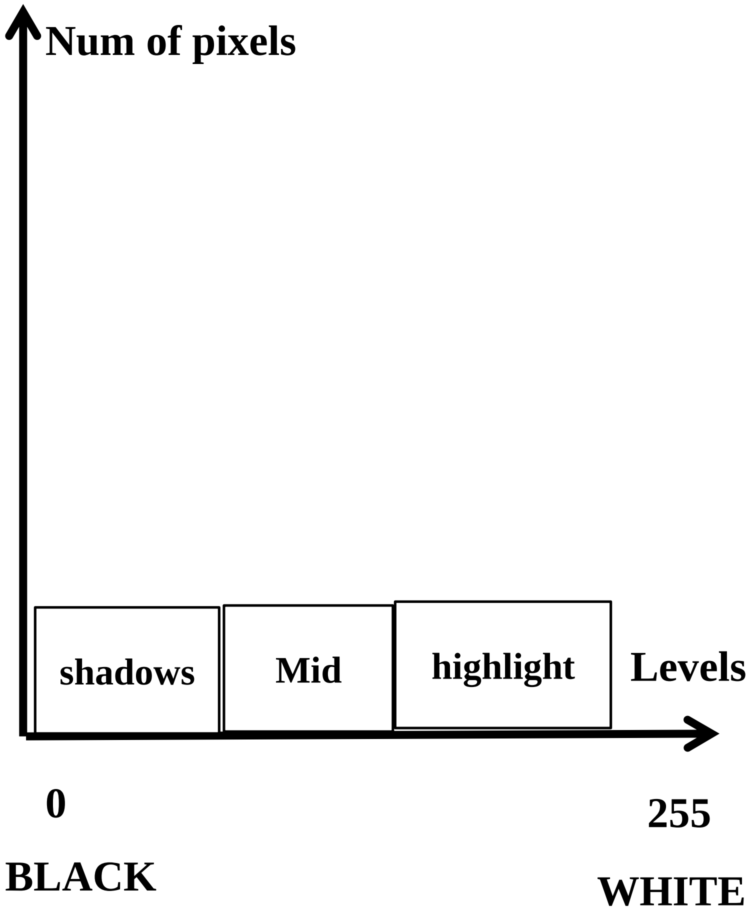
Elaf Mohamed

Salma Hany

➢ Histogram processing:-

In digital image processing, the histogram is used for graphical representation of a digital image. A graph is a plot by the number of pixels for each tonal value. Nowadays, image histogram is present in digital cameras. Photographers use them to see the distribution of tones captured.

In a graph, the horizontal axis of the graph is used to represent tonal variations whereas the vertical axis is used to represent the number of pixels in that particular pixel. Black and dark areas are represented in the left side of the horizontal axis, medium grey color is represented in the middle, and the vertical axis represents the size of the area.



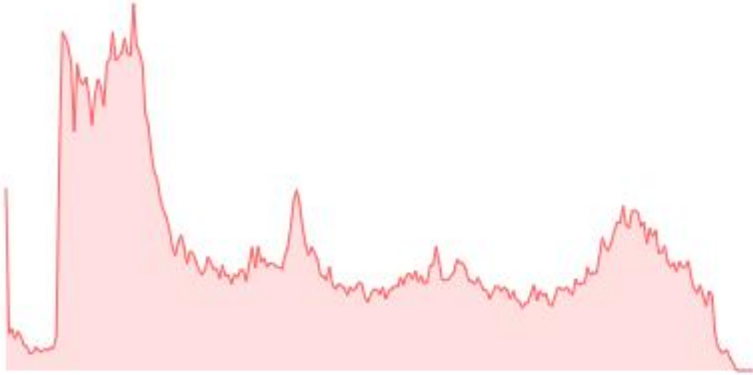
❑ **How It Works:**

The operation is very simple. The image is scanned in a single pass and a running count of the number of pixels found at each intensity value is kept. This is then used to construct a suitable histogram.

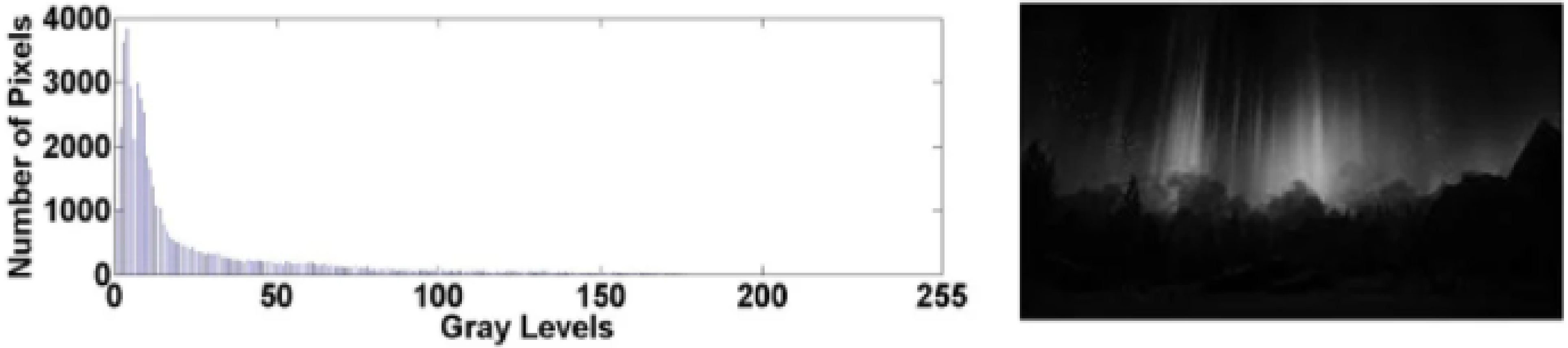
❑ **Applications of Histograms:**

a. The brightness of the image can be adjusted by having the details of its histogram.

b. If we have input and output histogram of an image, we can determine which type of transformation is applied in the algorithm.

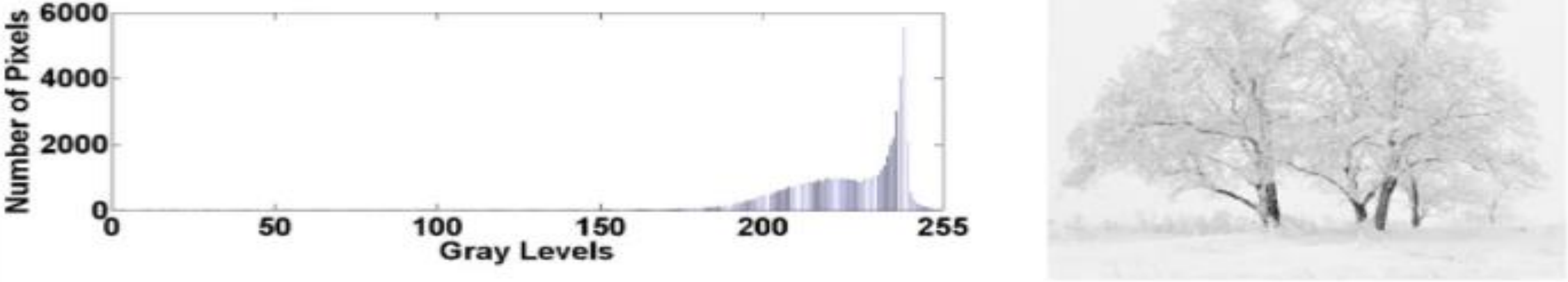


Histogram of the above scenery


Number of Pixels
4000
3000
2000
1000
0
0
50
100
150
200
255
Gray Levels

(a)

(b)


Number of Pixels
6000
4000
2000
0
0
50
100
150
200
255
Gray Levels

(a)

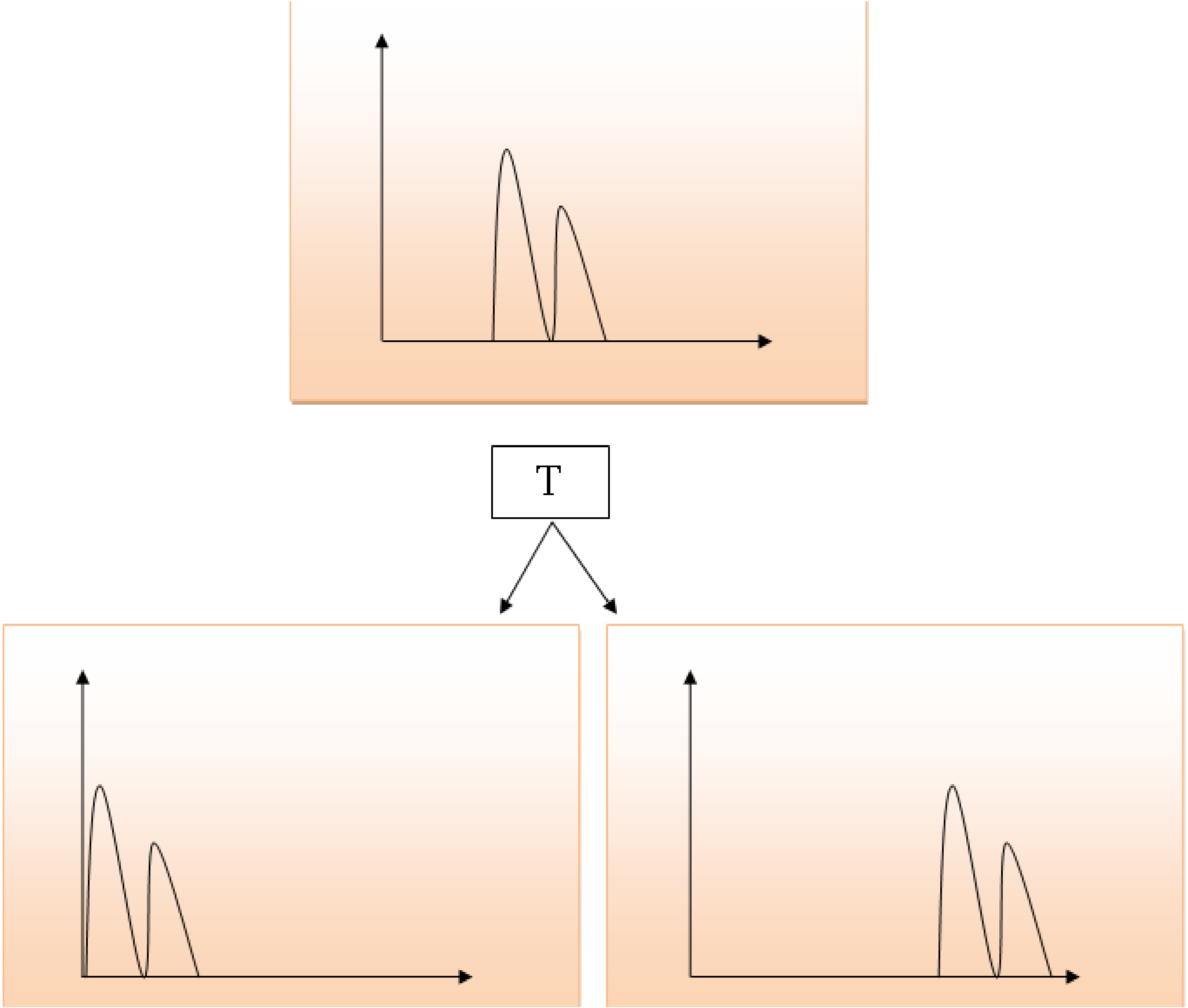
(b)

- ❑ **Histogram Processing Techniques:-**

a. **Histogram Sliding**
b. **Histogram Stretching**
c. **Histogram Equalization**

❑ Histogram Sliding:-

In Histogram sliding, the complete histogram is shifted towards rightwards or leftwards. When a histogram is shifted towards the right or left, clear changes are seen in the brightness of the image. The brightness of the image is defined by the intensity of light which is emitted by a particular light source.



This technique consists of simply adding or subtracting a constant brightness value to all pixels in the image. The overall effect is an image with comparable contrast properties, but higher or lower average brightness, respectively. imadd and imsubtract functions can be used for histogram sliding.

When implementing histogram sliding, you must make sure that pixel values do not go outside the greyscale boundaries. An example of histogram sliding is given below:

Matlab code:-

```
A = imread(‘Penguins_grey.jpg’);
imshow(A),title(‘Original Image’);
B=im2double(A);
bright_add = 0.2;
imhist(A), title(‘Original Histogram’);
C=B+bright_add;
imshow(C),title(‘New Bright Image’);
imhist(C), title(‘New Histogram’);
```

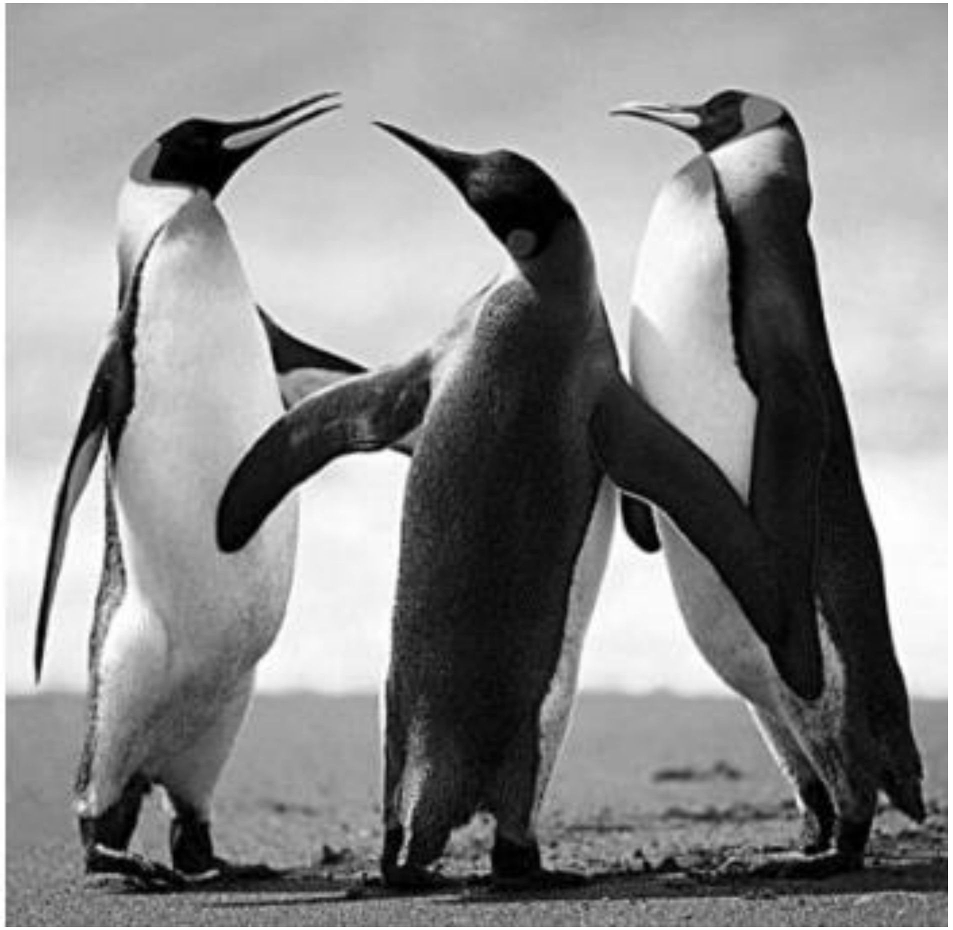


FIG 10

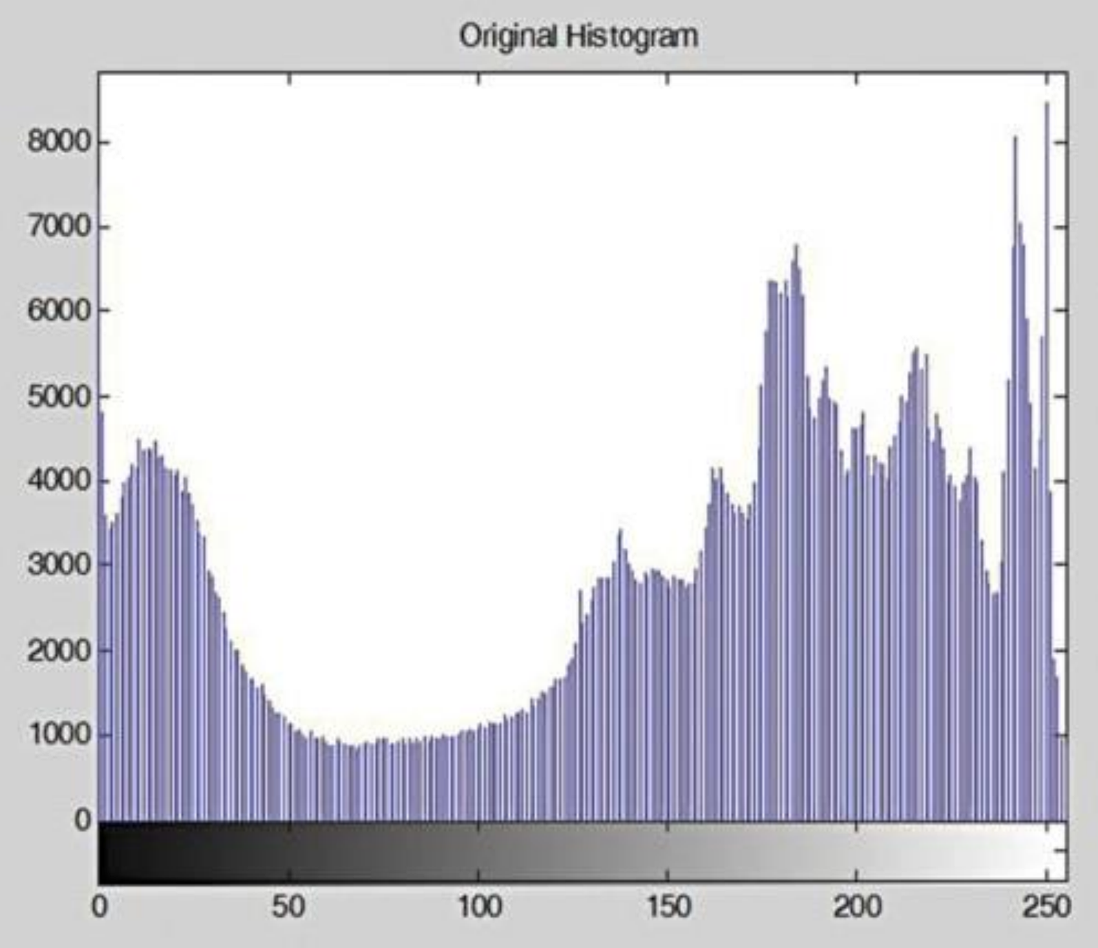

Original Histogram
8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250

FIG 11

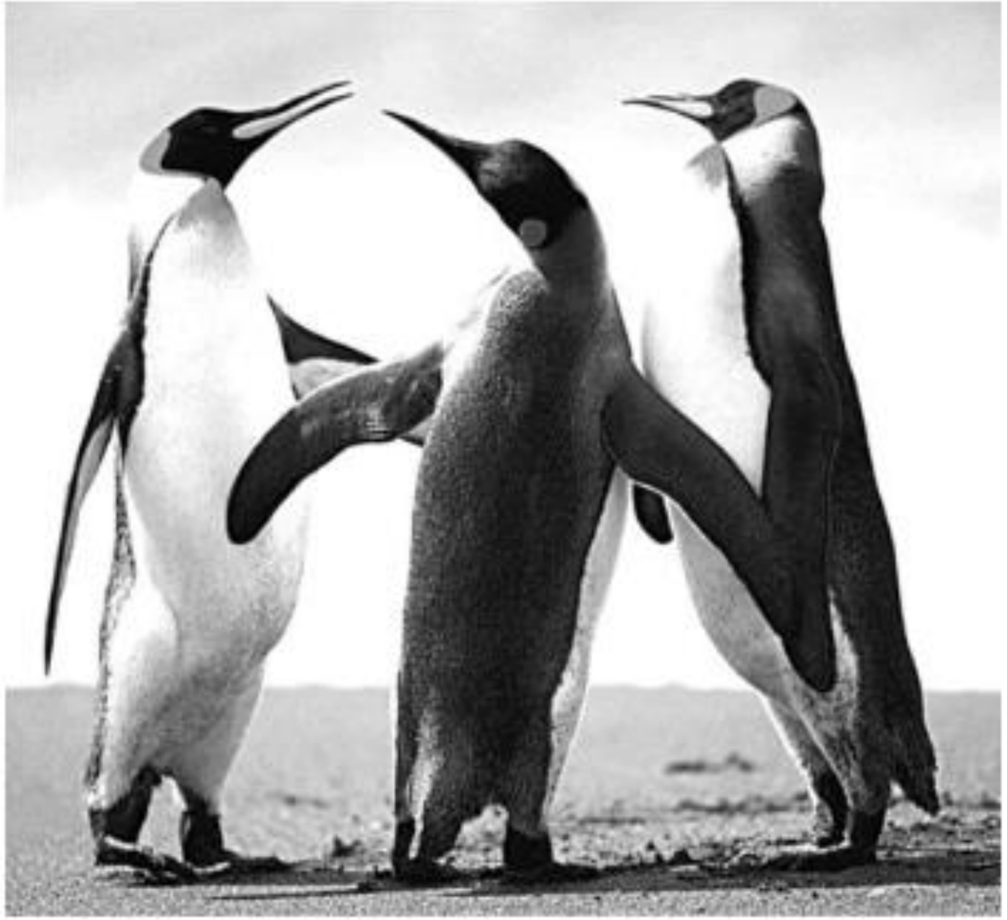


FIG 12

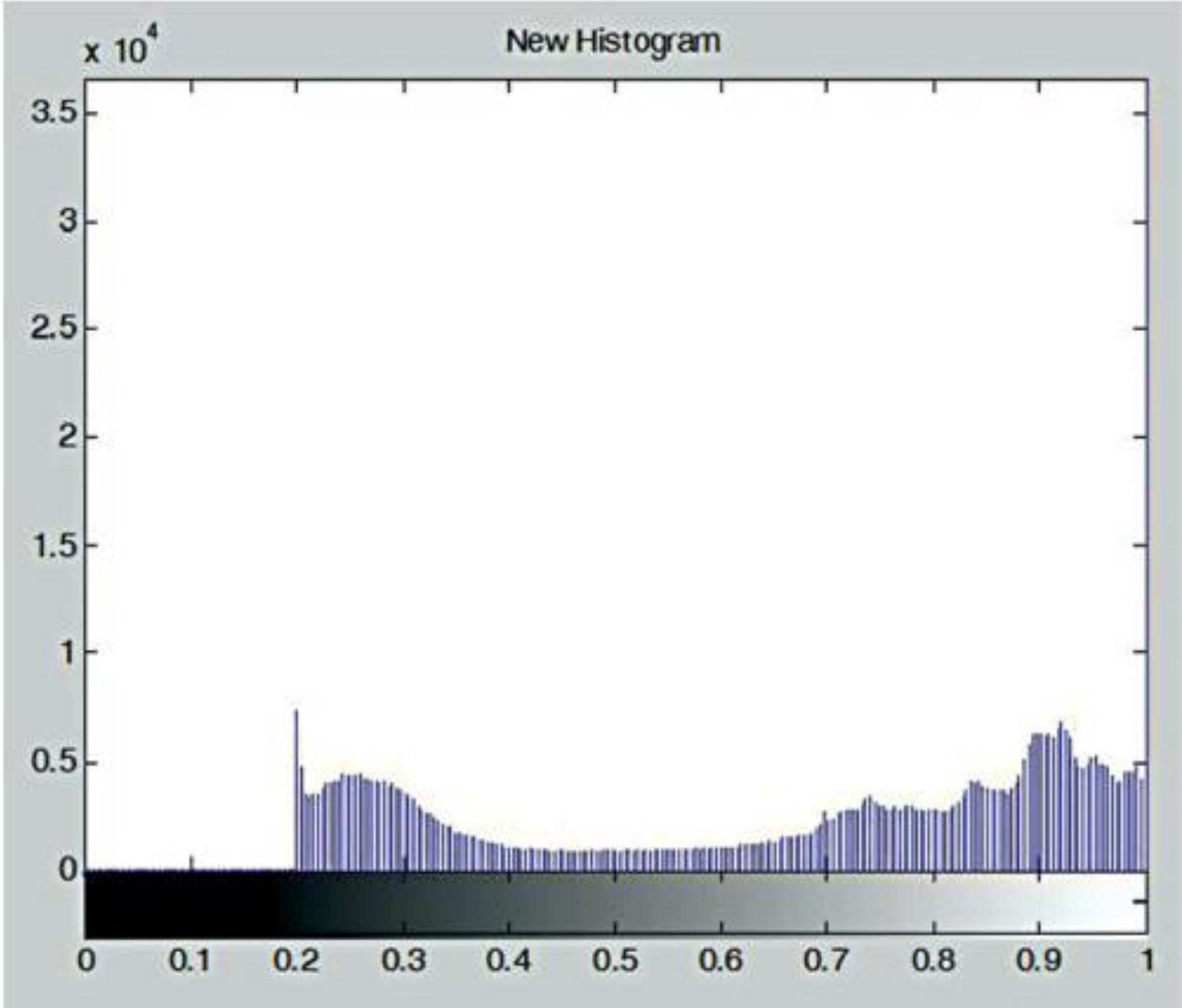


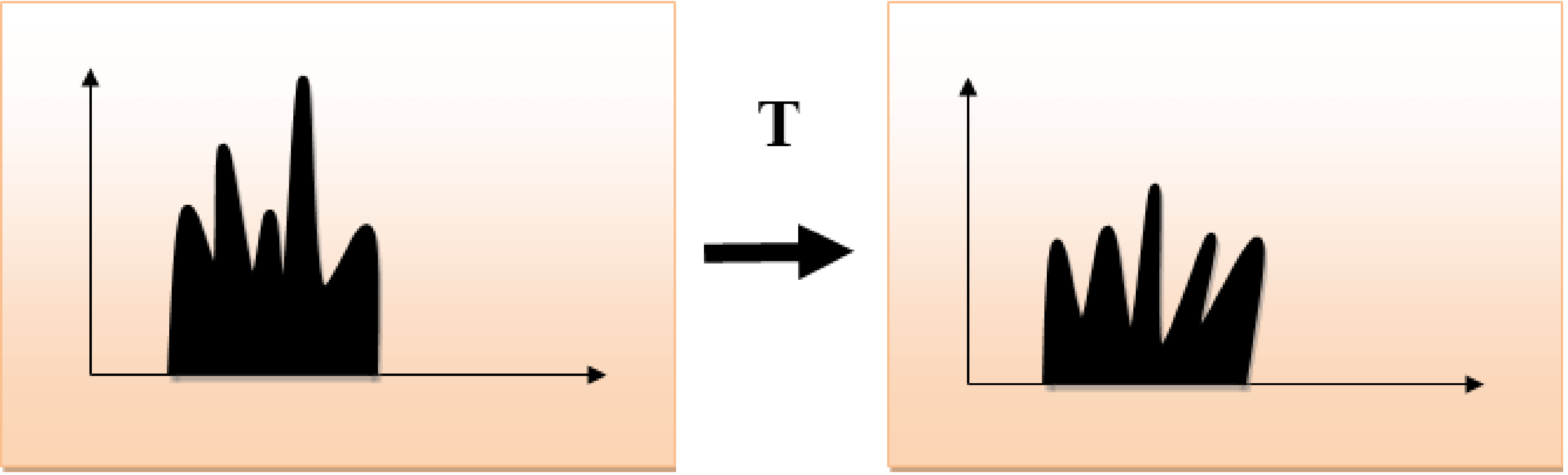
FIG 13

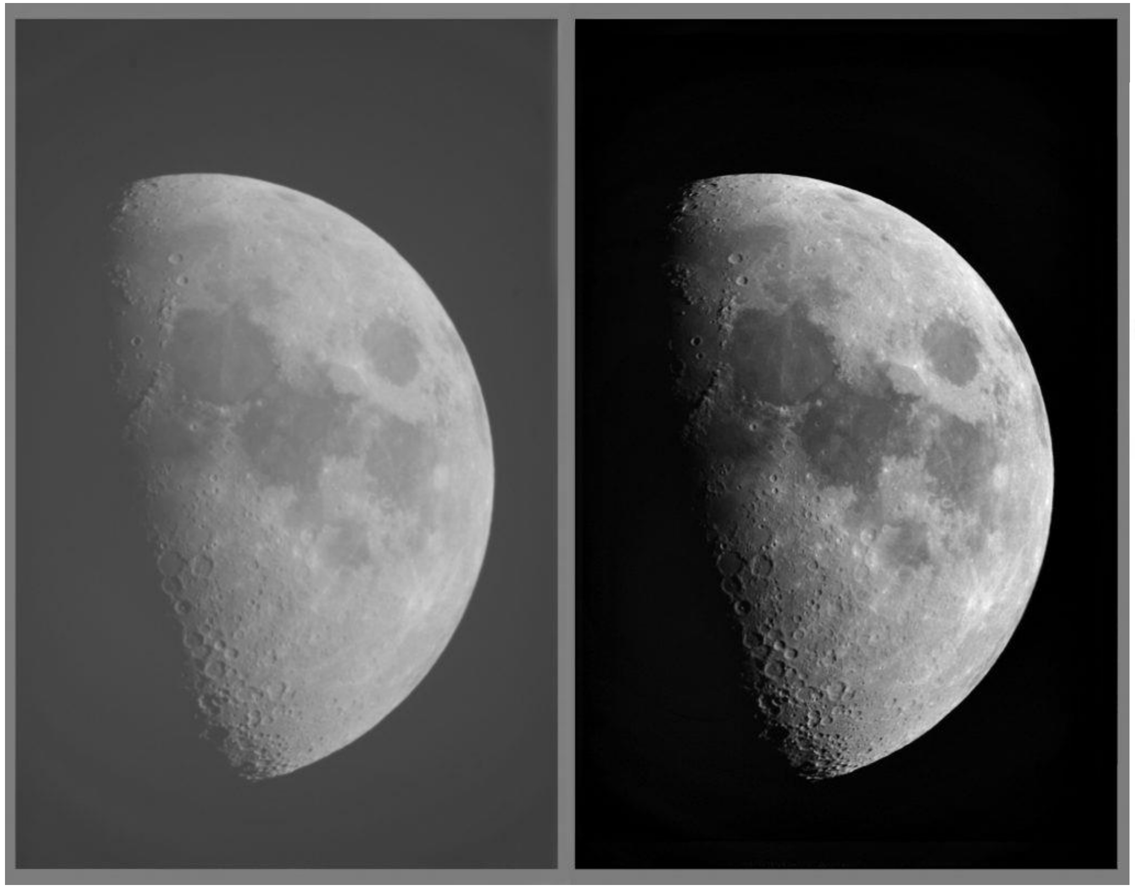
Histogram Stretching:-

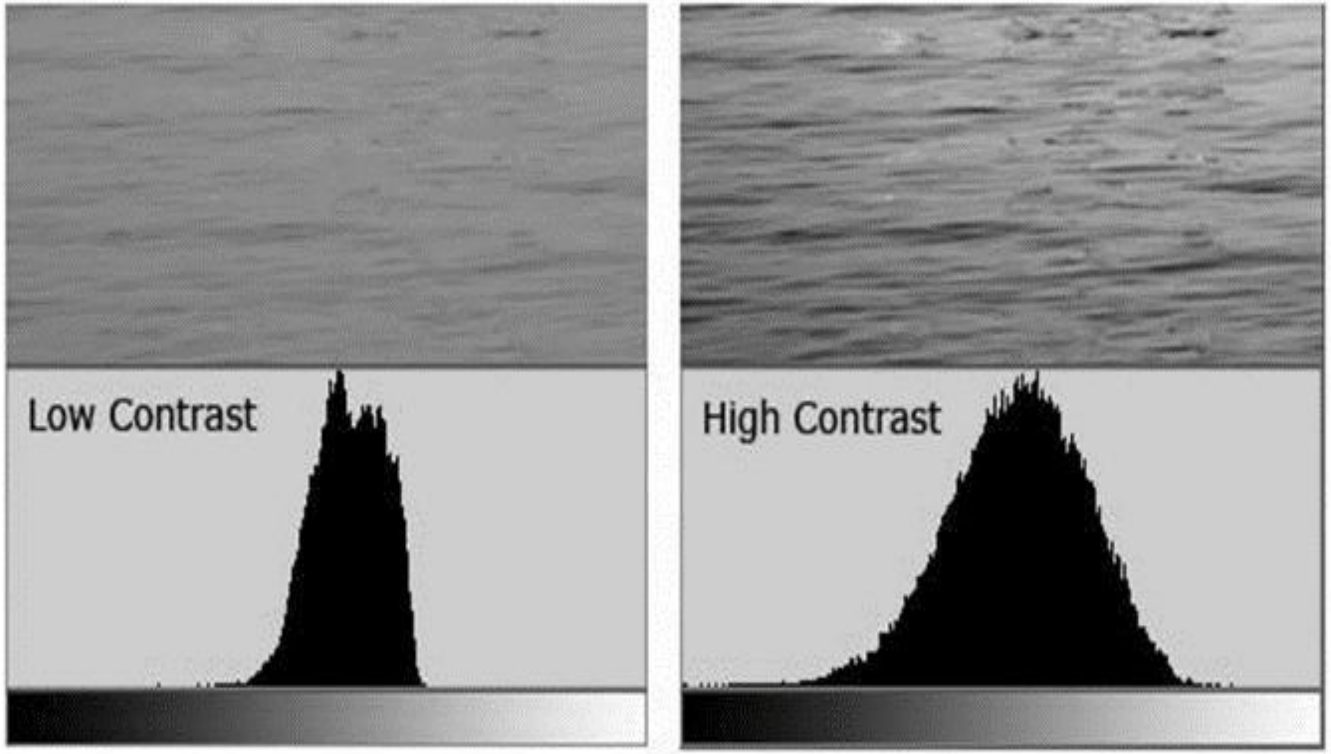
In histogram stretching, contrast of an image is increased. The contrast of an image is defined between the maximum and minimum value of pixel intensity.

If we want to increase the contrast of an image, histogram of that image will be fully stretched and covered the dynamic range of the histogram.

From histogram of an image, we can check that the image has low or high contrast.





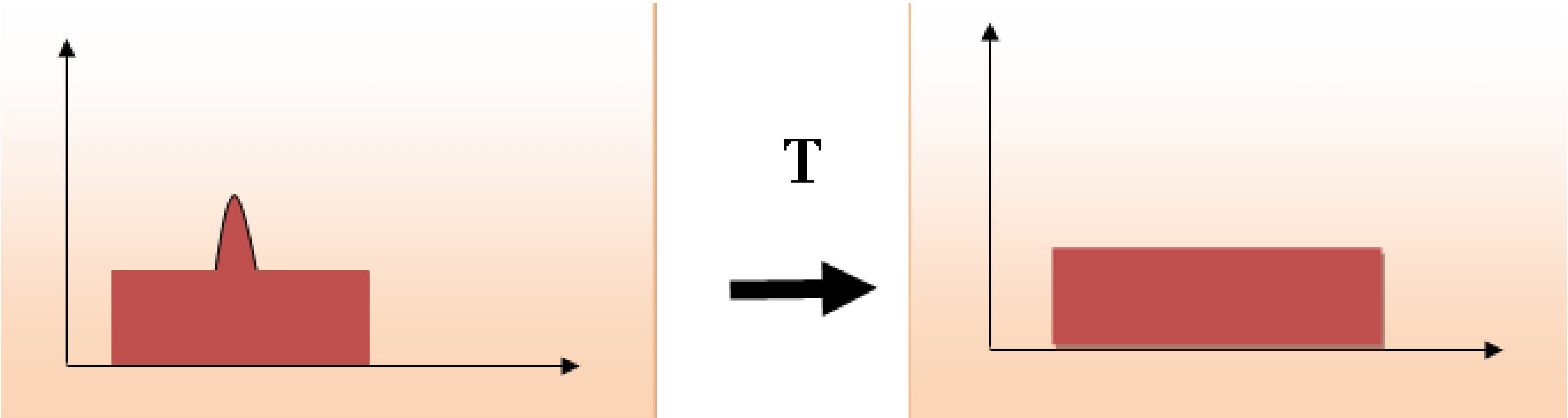

Low Contrast
High Contrast

❑ Histogram Equalization:-

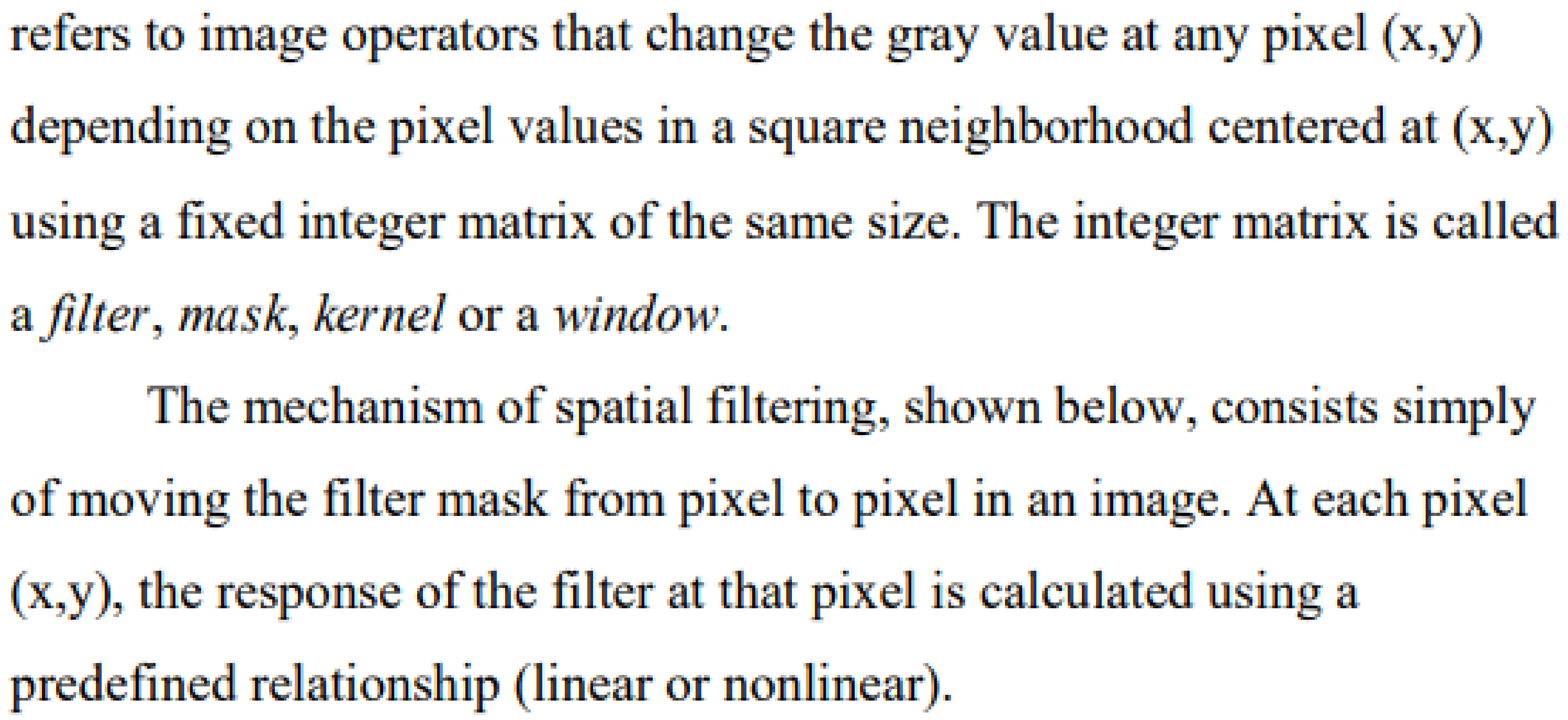
Histogram equalization is used for equalizing all the pixel values of an image. Transformation is done in such a way that uniform flattened histogram is produced.

Histogram equalization increases the dynamic range of pixel values and makes an equal count of pixels at each level which produces a flat histogram with high contrast image.

While stretching histogram, the shape of histogram remains the same whereas in Histogram equalization, the shape of histogram changes and it generates only one image.



➢ Fundamentals of spatial filtering:-



refers to image operators that change the gray value at any pixel (x,y) depending on the pixel values in a square neighborhood centered at (x,y) using a fixed integer matrix of the same size. The integer matrix is called a *filter*, *mask*, *kernel* or a *window*.

The mechanism of spatial filtering, shown below, consists simply of moving the filter mask from pixel to pixel in an image. At each pixel (x,y), the response of the filter at that pixel is calculated using a predefined relationship (linear or nonlinear).

❑ **Classification on the basis of linearity:**

a. **Linear Spatial Filter**

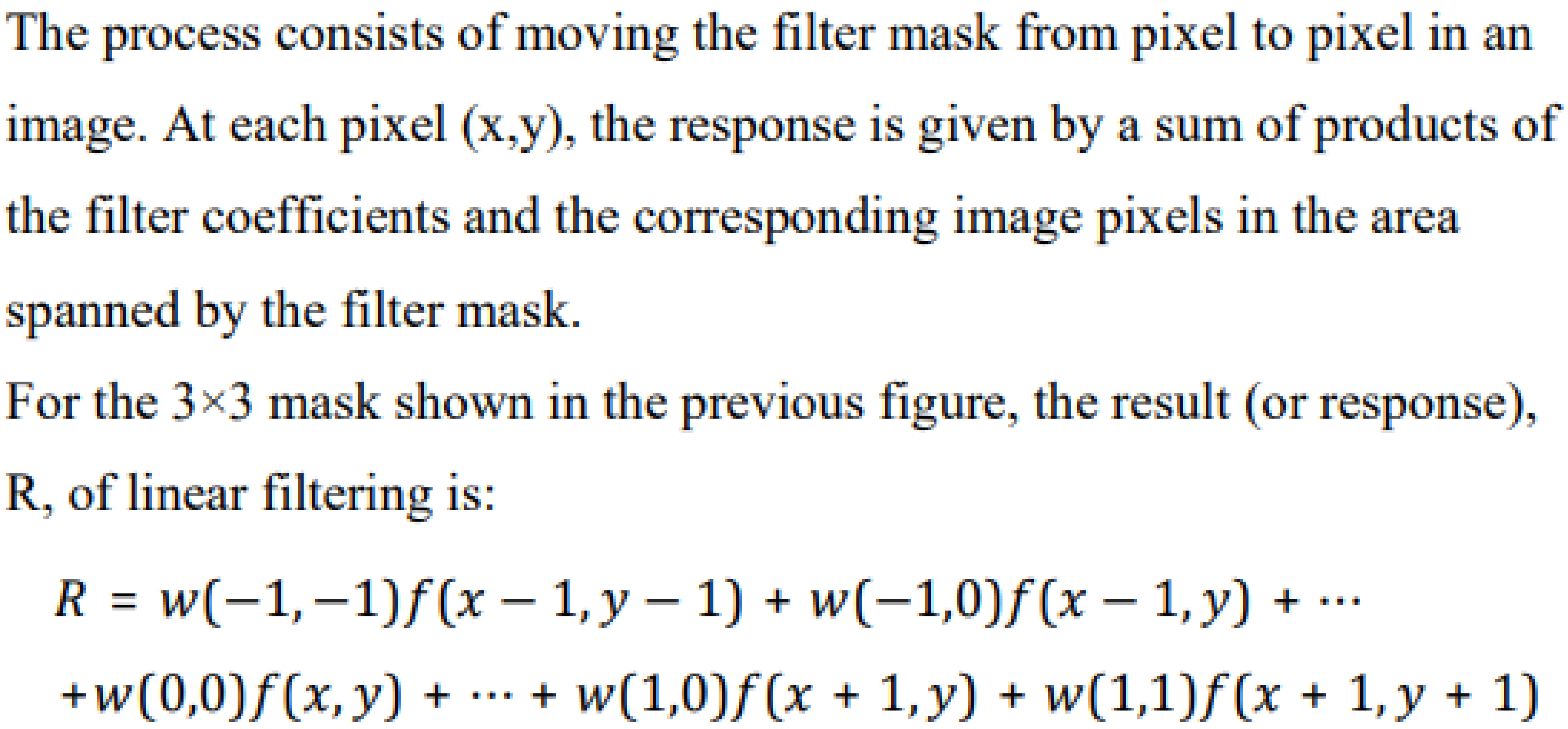
b. **Non-linear Spatial Filter**

❑ **General Classification:**

a. **Smoothing Spatial Filter**

b. **Sharpening Spatial Filter**

❑ Linear Spatial Filter:-



The process consists of moving the filter mask from pixel to pixel in an image. At each pixel (x,y), the response is given by a sum of products of the filter coefficients and the corresponding image pixels in the area spanned by the filter mask.

For the 3×3 mask shown in the previous figure, the result (or response), R, of linear filtering is:

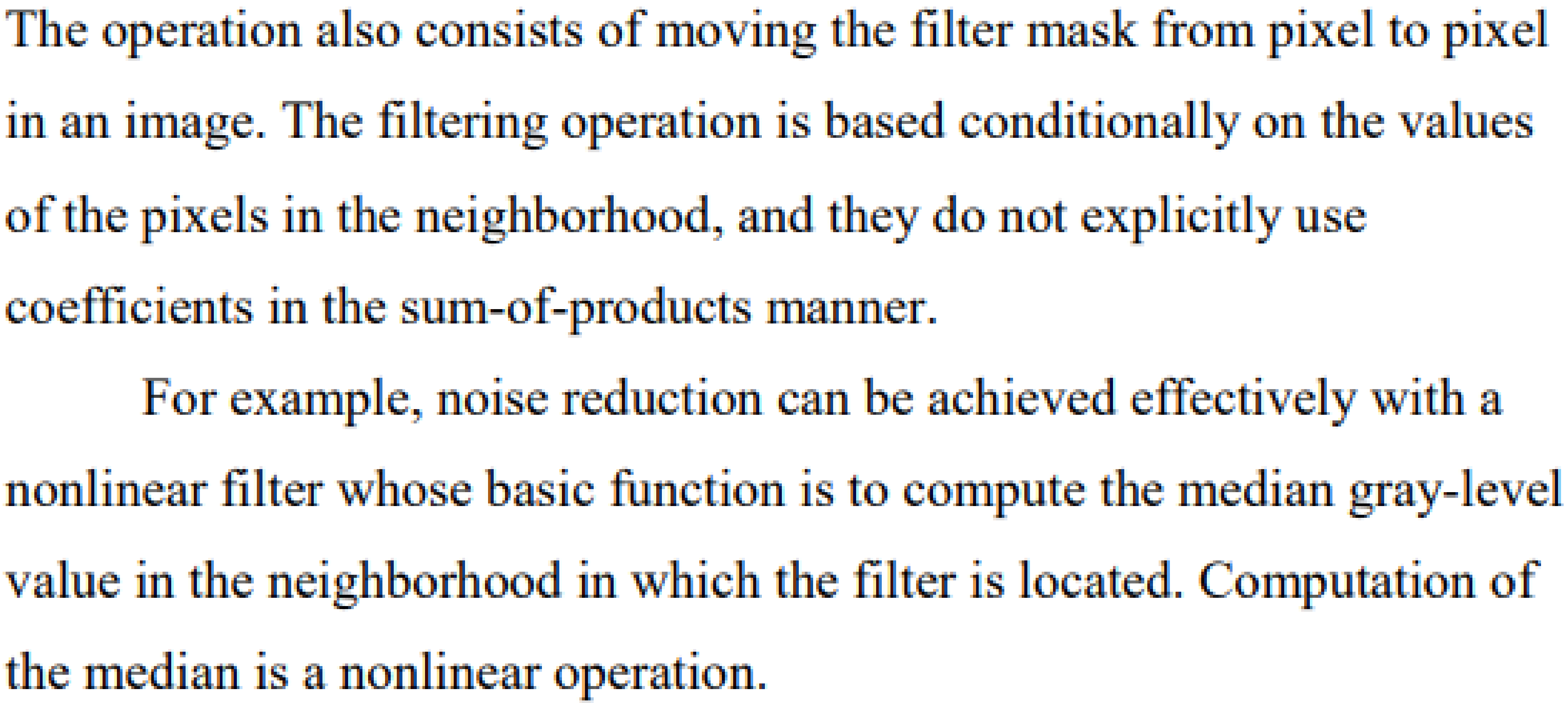
$$R = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \cdots$$
$$+w(0,0)f(x,y) + \cdots + w(1,0)f(x+1,y) + w(1,1)f(x+1,y+1)$$

In general, linear filtering of an image *f* of size *M*×*N* with a filter mask of size *m*×*n* is given by the expression:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t)f(x+s,y+t)$$

where a = (*m* - 1)/2 and b = (*n* - l)/2. To generate a complete filtered image this equation must be applied for x = 0,1, 2,..., *M*-1 and y = 0,1, 2,..., *N*-1.

- **Non-linear Spatial Filter:-**



The operation also consists of moving the filter mask from pixel to pixel in an image. The filtering operation is based conditionally on the values of the pixels in the neighborhood, and they do not explicitly use coefficients in the sum-of-products manner.

For example, noise reduction can be achieved effectively with a nonlinear filter whose basic function is to compute the median gray-level value in the neighborhood in which the filter is located. Computation of the median is a nonlinear operation.

Example:

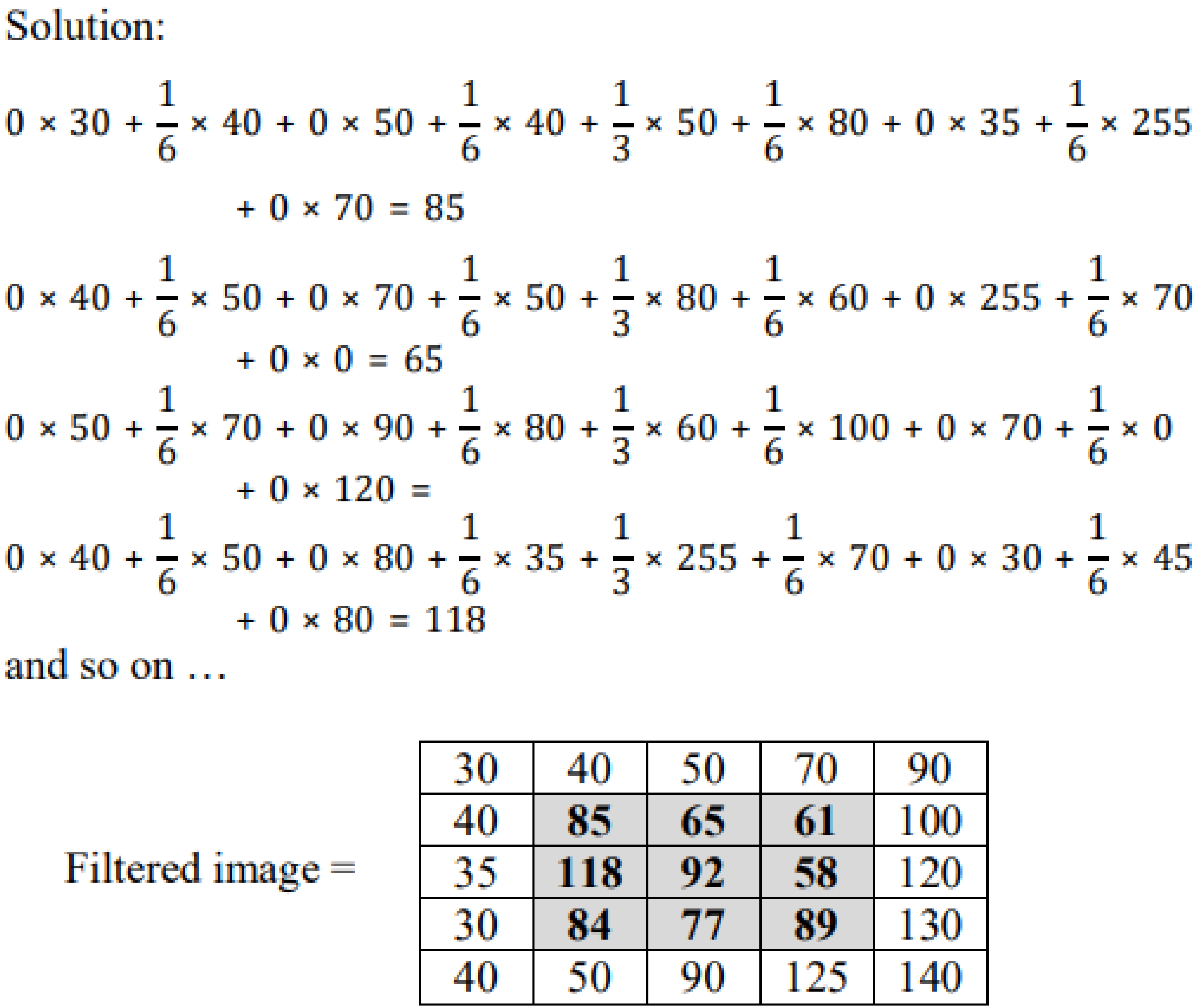
Use the following 3×3mask to perform the convolution process on the shaded pixels in the 5×5 image below. Write the filtered image.

0	1/6	0
1/6	1/3	1/6
0	1/6	0

3×3 mask

30	40	50	70	90
40	50	80	60	100
35	255	70	0	120
30	45	80	100	130
40	50	90	125	140

5×5 image



Solution:

$$0 \times 30 + \frac{1}{6} \times 40 + 0 \times 50 + \frac{1}{6} \times 40 + \frac{1}{3} \times 50 + \frac{1}{6} \times 80 + 0 \times 35 + \frac{1}{6} \times 255 + 0 \times 70 = 85$$

$$0 \times 40 + \frac{1}{6} \times 50 + 0 \times 70 + \frac{1}{6} \times 50 + \frac{1}{3} \times 80 + \frac{1}{6} \times 60 + 0 \times 255 + \frac{1}{6} \times 70 + 0 \times 0 = 65$$

$$0 \times 50 + \frac{1}{6} \times 70 + 0 \times 90 + \frac{1}{6} \times 80 + \frac{1}{3} \times 60 + \frac{1}{6} \times 100 + 0 \times 70 + \frac{1}{6} \times 0 + 0 \times 120 =$$

$$0 \times 40 + \frac{1}{6} \times 50 + 0 \times 80 + \frac{1}{6} \times 35 + \frac{1}{3} \times 255 + \frac{1}{6} \times 70 + 0 \times 30 + \frac{1}{6} \times 45 + 0 \times 80 = 118$$

and so on …

Filtered image =

30	40	50	70	90
40	**85**	**65**	**61**	100
35	**118**	**92**	**58**	120
30	**84**	**77**	**89**	130
40	50	90	125	140

- ## Smoothing spatial filters:-

Also called low pass filter. They include:

a. Averaging linear filters

b. Order-statistics non-linear filter

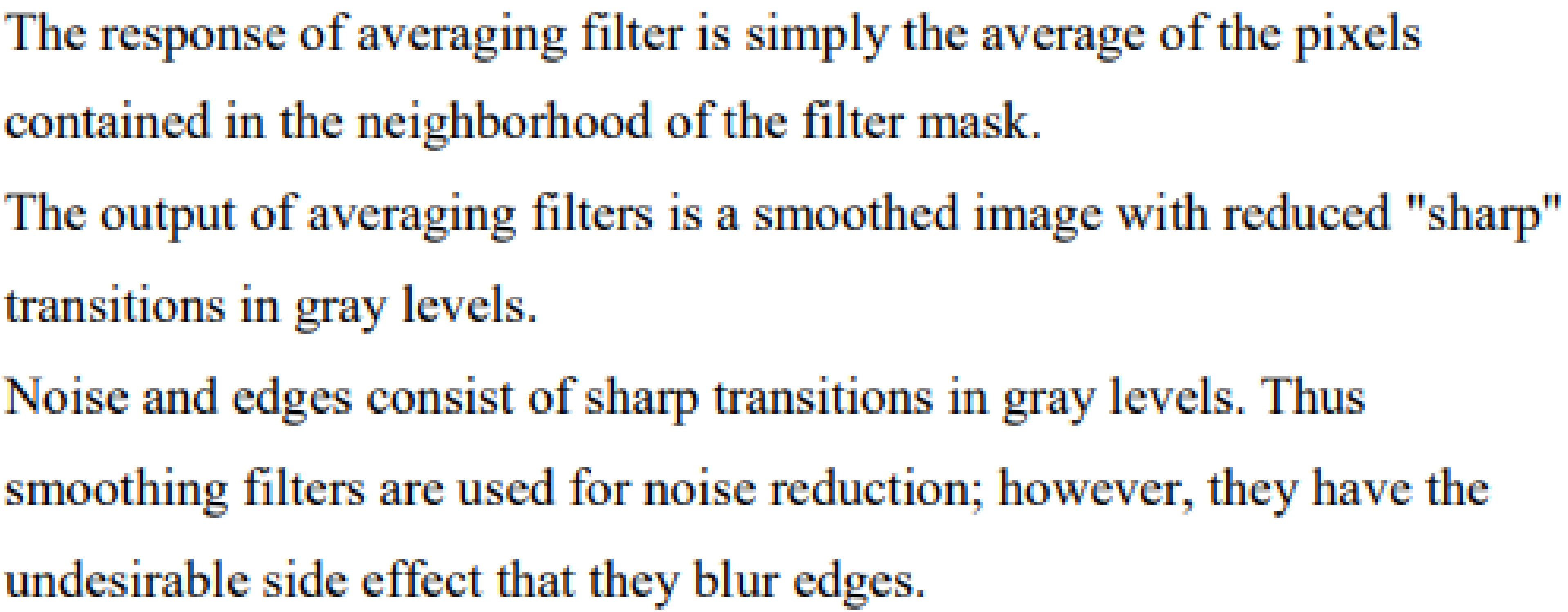
Smoothing Spatial Filters

are used for blurring and for noise reduction. Blurring is used in preprocessing steps to:

- remove small details from an image prior to (large) object extraction
- bridge small gaps in lines or curves.

Noise reduction can be accomplished by blurring with a linear filter and also by nonlinear filtering.

❑ Averaging linear filters:-



The response of averaging filter is simply the average of the pixels contained in the neighborhood of the filter mask.

The output of averaging filters is a smoothed image with reduced "sharp" transitions in gray levels.

Noise and edges consist of sharp transitions in gray levels. Thus smoothing filters are used for noise reduction; however, they have the undesirable side effect that they blur edges.

The figure below shows two 3×3 averaging filters.

$$\frac{1}{9}\times\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$

Standard average filter

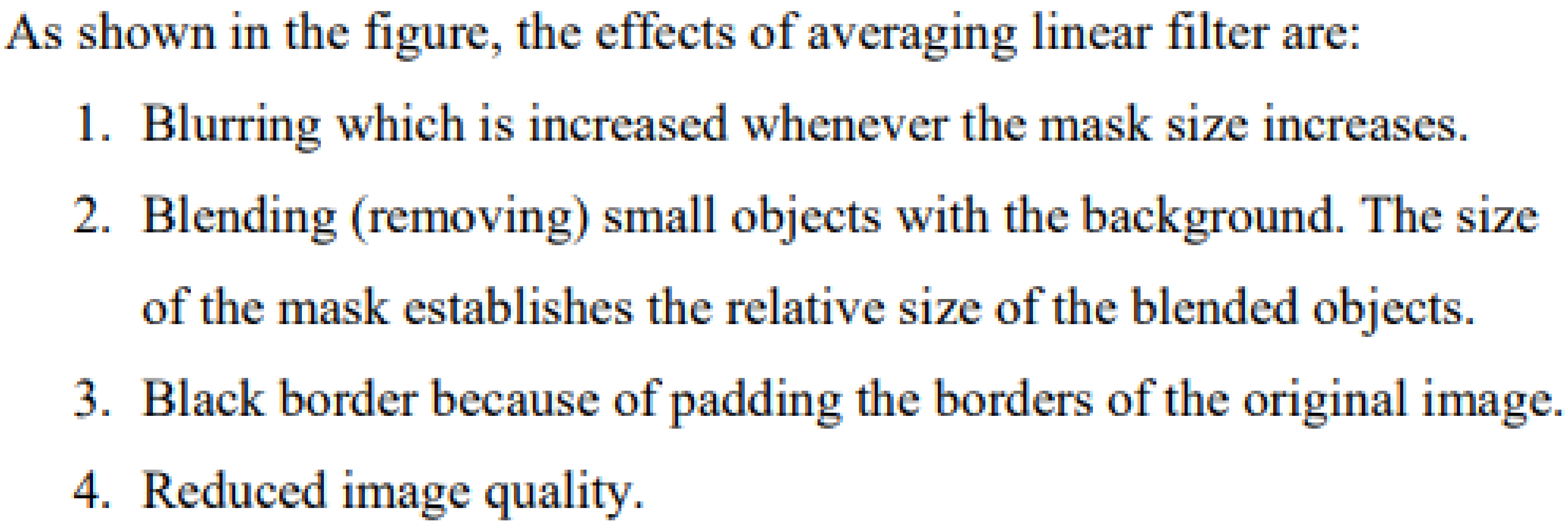
$$\frac{1}{16}\times\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline\end{array}$$

Weighted average filter

Averaging linear filtering of an image *f* of size *M*×*N* with a filter mask of size *m*×*n* is given by the expression:

$$g(x, y) = \frac{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x + s, y + t)}{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t)}$$

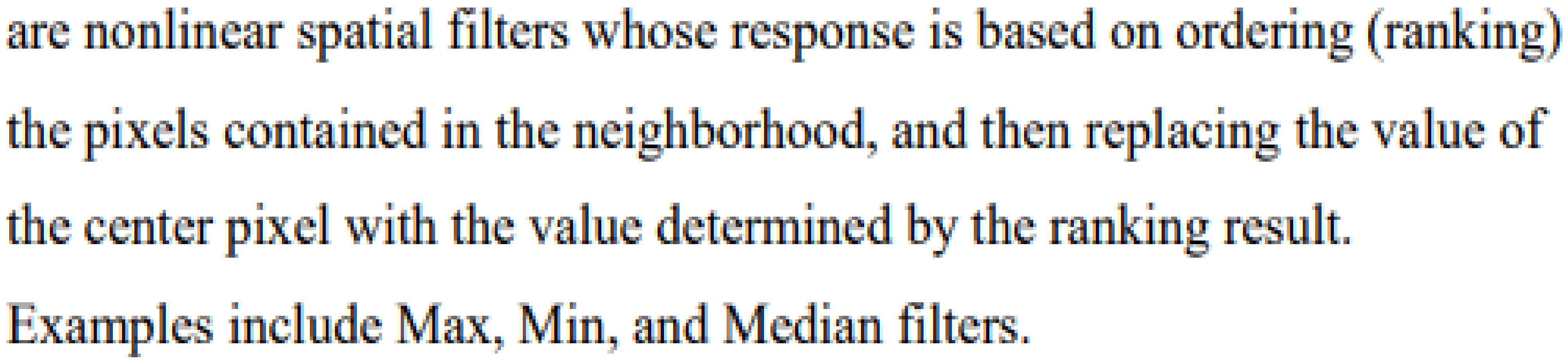
To generate a complete filtered image this equation must be applied for x = 0,1, 2,..., *M*-1 and y = 0,1, 2,..., *N*-1.



As shown in the figure, the effects of averaging linear filter are:

1. Blurring which is increased whenever the mask size increases.
2. Blending (removing) small objects with the background. The size of the mask establishes the relative size of the blended objects.
3. Black border because of padding the borders of the original image.
4. Reduced image quality.

❑ Order-statistics non-linear filter:-



are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the neighborhood, and then replacing the value of the center pixel with the value determined by the ranking result.

Examples include Max, Min, and Median filters.

Median filter

It replaces the value at the center by the median pixel value in the neighborhood, (i.e. the middle element after they are sorted). Median filters are particularly useful in removing impulse noise (also known as salt-and-pepper noise). Salt = 255, pepper = 0 gray levels.

In a 3×3 neighborhood the median is the 5th largest value, in a 5×5 neighborhood the 13th largest value, and so on.

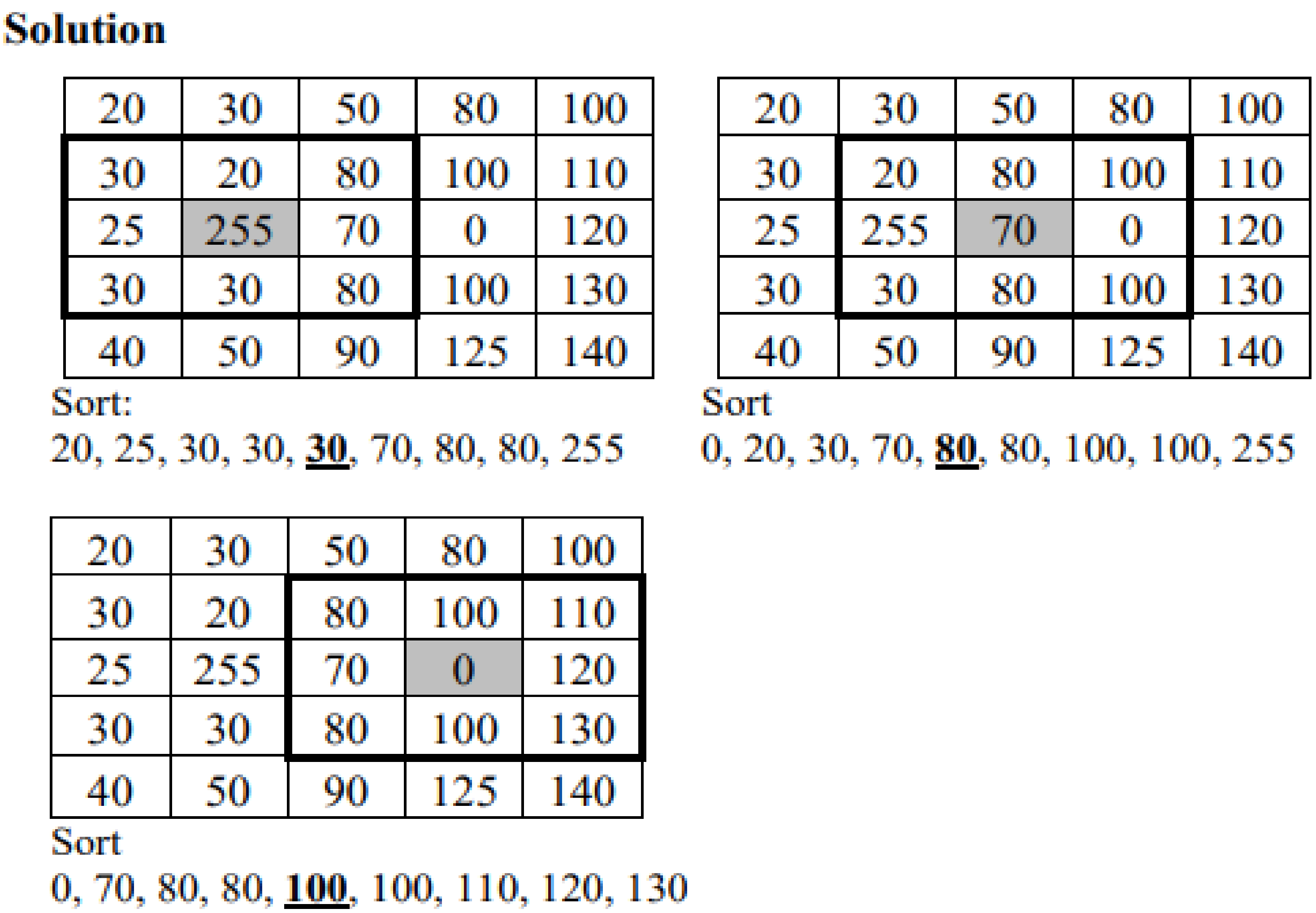
For example, suppose that a 3×3 neighborhood has gray levels (10, 20, 0, 20, 255, 20, 20, 25, 15). These values are sorted as (0,10,15,20,20,20,20,25,255), which results in a median of 20 that replaces the original pixel value 255 (salt noise).

Example:

Consider the following 5×5 image:

20	30	50	80	100
30	20	80	100	110
25	255	70	0	120
30	30	80	100	130
40	50	90	125	140

Apply a 3×3 median filter on the shaded pixels, and write the filtered image.



Solution

20	30	50	80	100
30	20	80	100	110
25	255	70	0	120
30	30	80	100	130
40	50	90	125	140

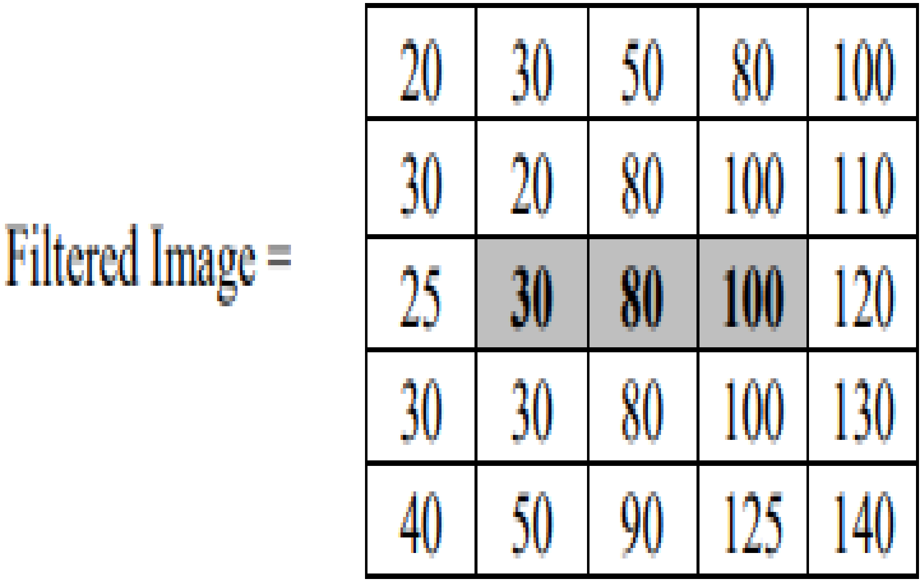
Sort:
20, 25, 30, 30, **30**, 70, 80, 80, 255

20	30	50	80	100
30	20	80	100	110
25	255	70	0	120
30	30	80	100	130
40	50	90	125	140

Sort
0, 20, 30, 70, **80**, 80, 100, 100, 255

20	30	50	80	100
30	20	80	100	110
25	255	70	0	120
30	30	80	100	130
40	50	90	125	140

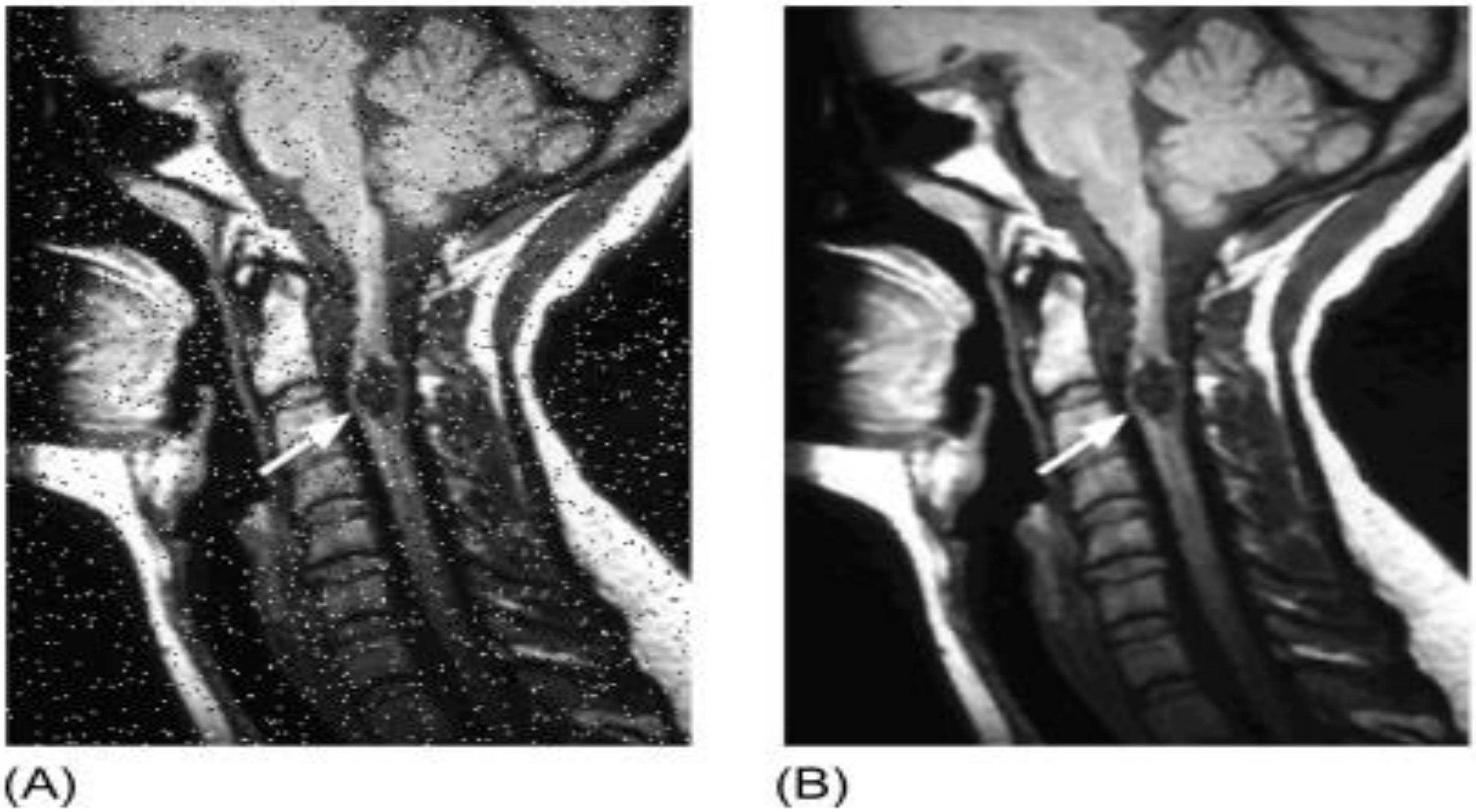
Sort
0, 70, 80, 80, **100**, 100, 110, 120, 130



Filtered Image =

20	30	50	80	100
30	20	80	100	110
25	**30**	**80**	**100**	120
30	30	80	100	130
40	50	90	125	140

❑ **Figure below shows an example of applying the median filter:-**

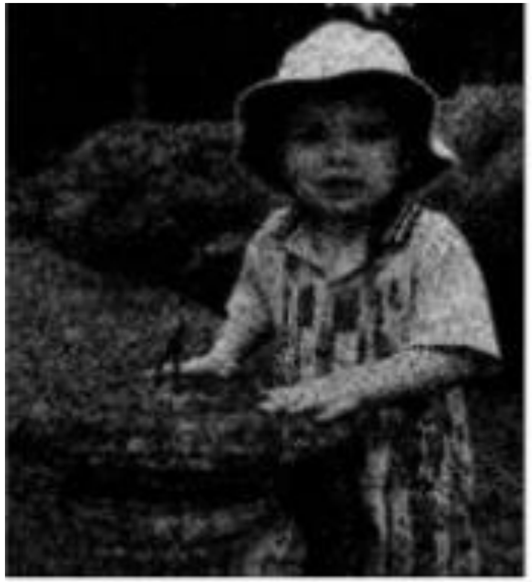


As shown in the figure, the effects of median filter are:

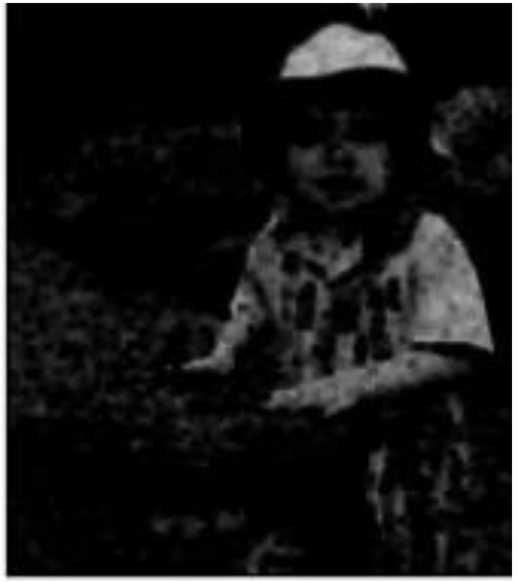
1. Noise reduction
2. Less blurring than averaging linear filter

Minimum Filter

- The 0th percentile filter is the min filter.
- Minimum filter selects the smallest value in the window and replace the center by the smallest value
- Using comparison the minimum value can be obtained fast.(not necessary to sort)
- It enhances the dark areas of image



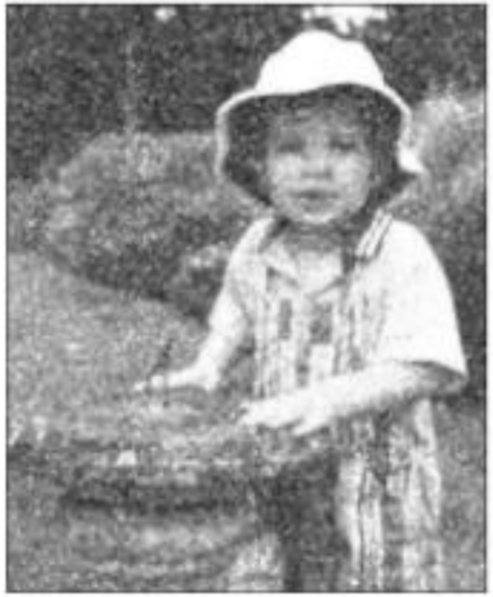
(mask size =3 x 3)



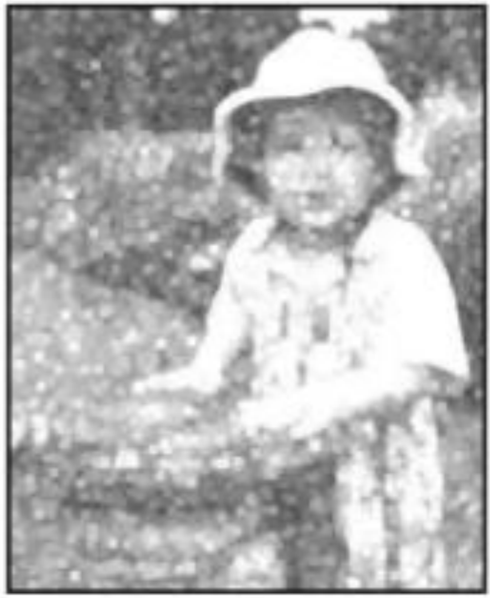
(mask size =7 x 7)

Maximum Filter

- The maximum filter selects the largest value within of pixel values, and replace the center by the largest value.
- Using comparison the maximum value can be obtained fast.(not necessary to sort)
- Using the 100th percentile results in the so-called *max filter*
- it enhances bright areas of image



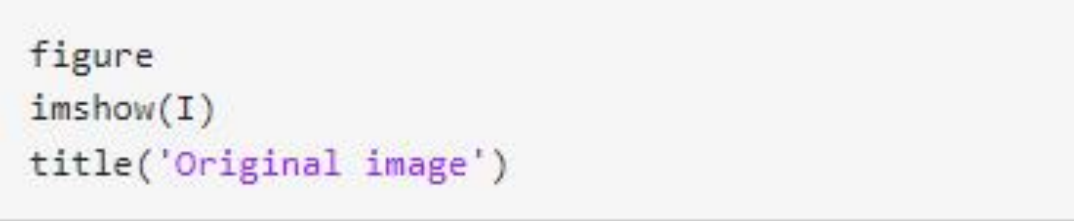
mask (3 x 3)



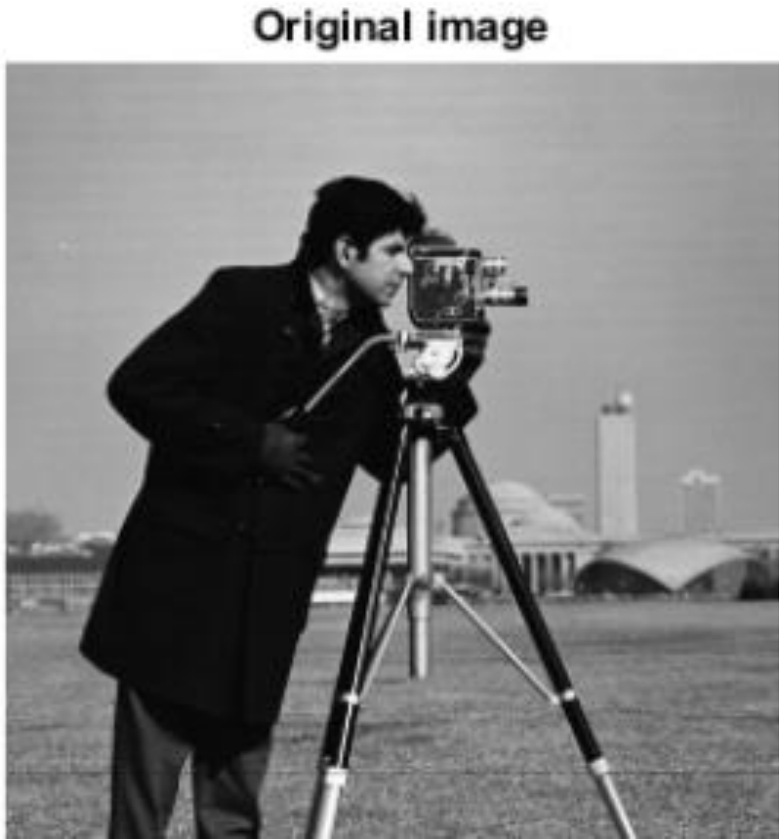
mask (7 x 7)

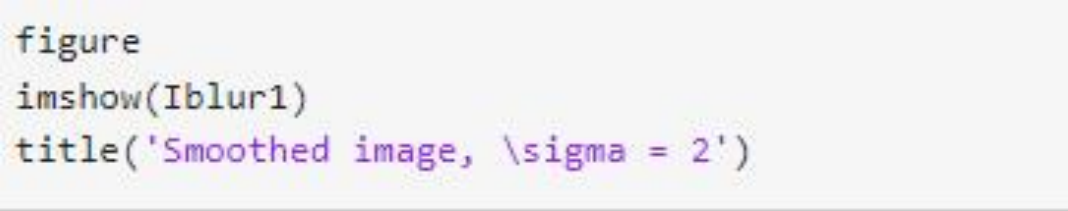
- **Matlab code:-**

```
I = imread('cameraman.tif');

Iblur1 = imgaussfilt(I,2);

Iblur2 = imgaussfilt(I,4);

Iblur3 = imgaussfilt(I,8);
```

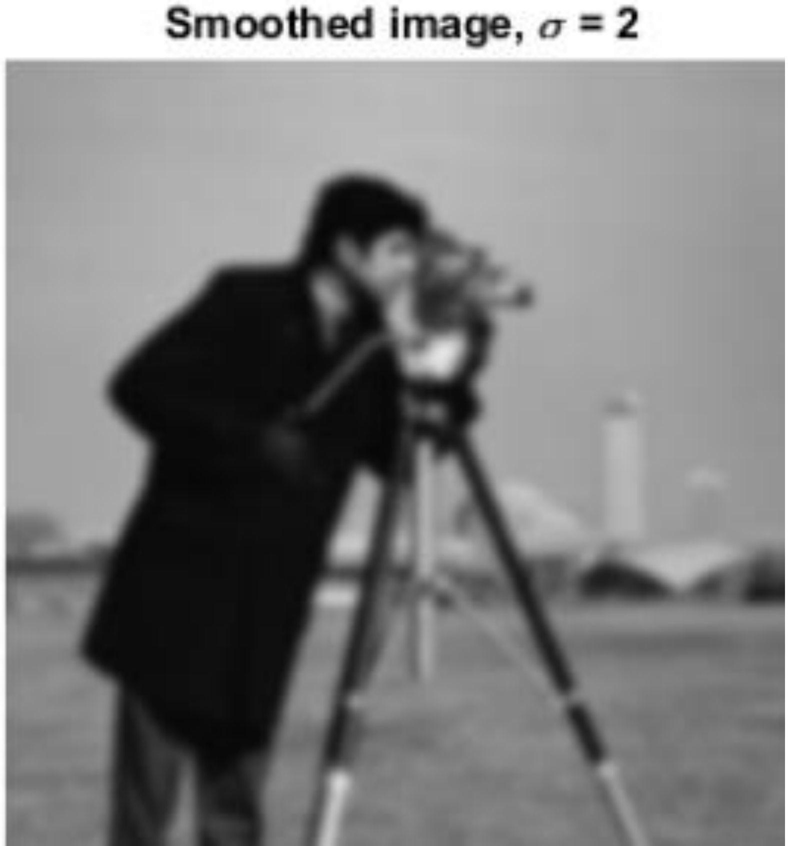


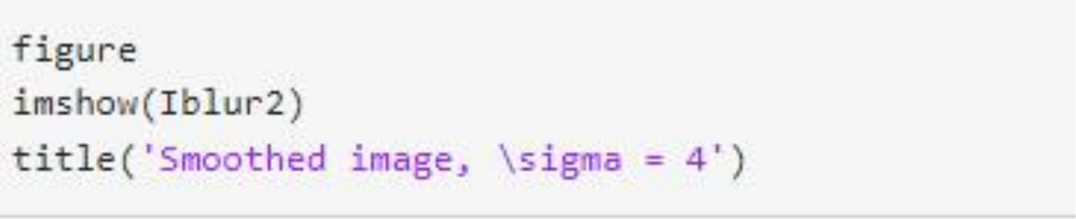
```
figure
imshow(I)
title('Original image')
```



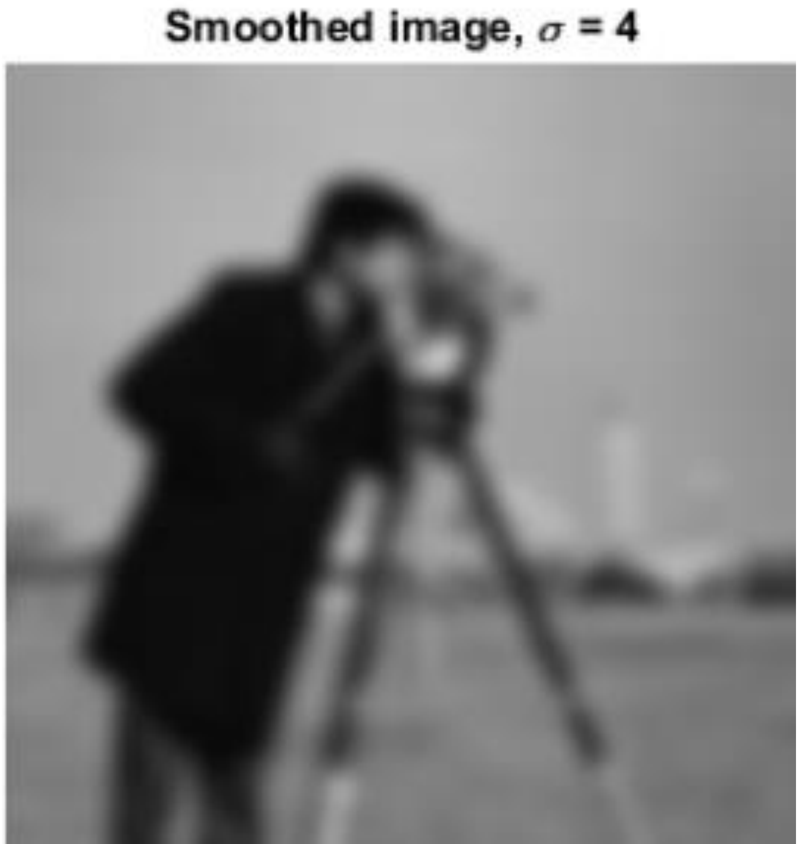


```
figure
imshow(Iblur1)
title('Smoothed image, \sigma = 2')
```

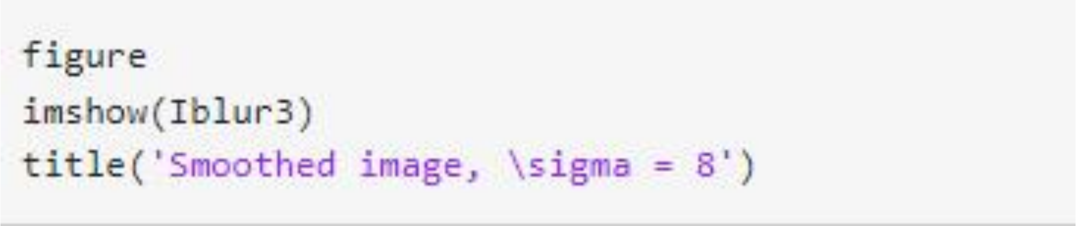




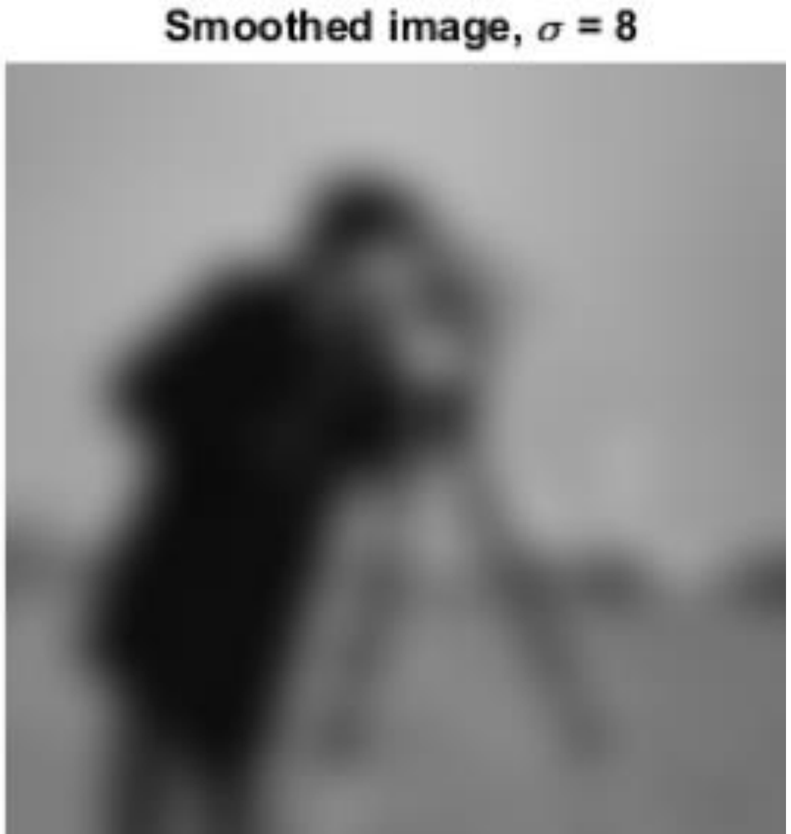
```
figure
imshow(Iblur2)
title('Smoothed image, \sigma = 4')
```



Smoothed image, σ = 4

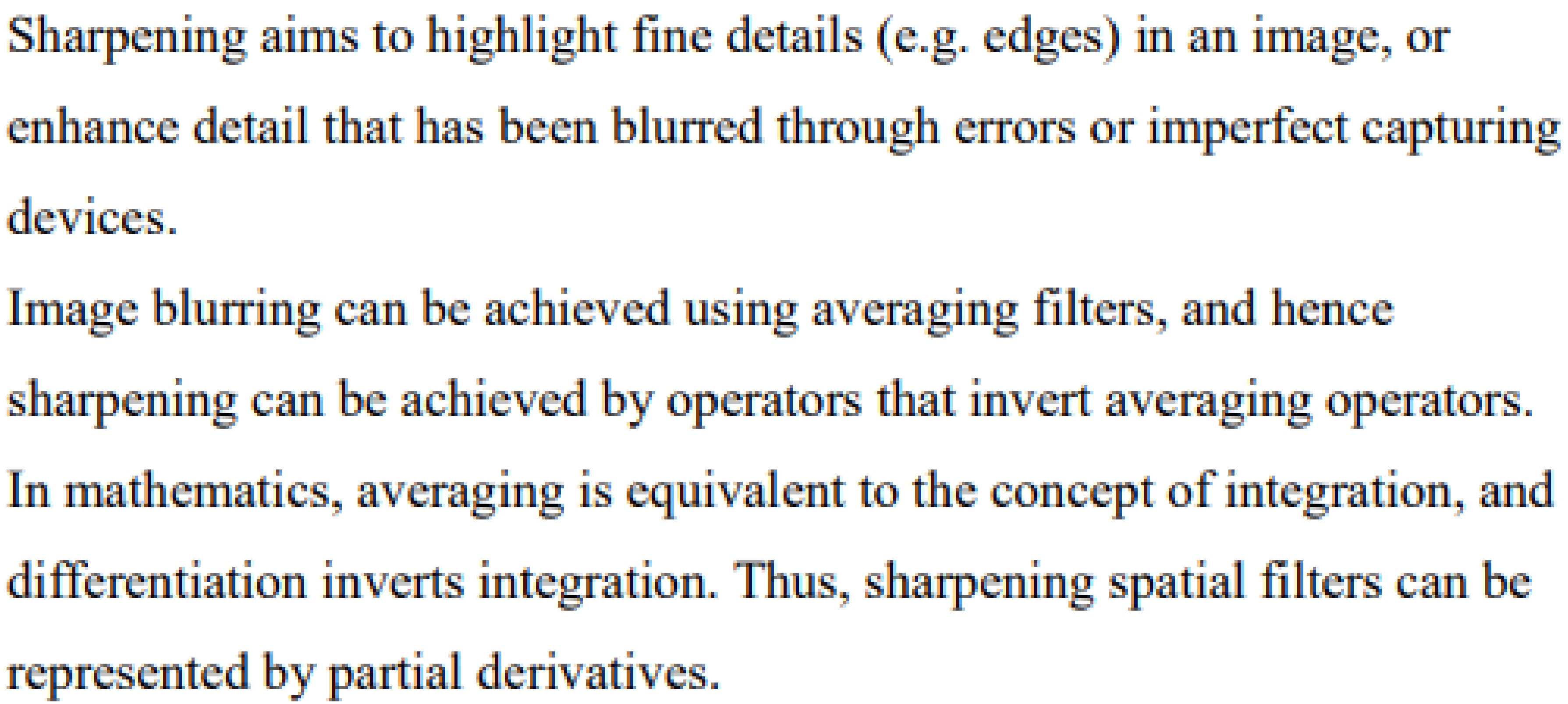


```
figure
imshow(Iblur3)
title('Smoothed image, \sigma = 8')
```



Smoothed image, σ = 8

➢ Sharpening spatial filters:-



Sharpening aims to highlight fine details (e.g. edges) in an image, or enhance detail that has been blurred through errors or imperfect capturing devices.

Image blurring can be achieved using averaging filters, and hence sharpening can be achieved by operators that invert averaging operators. In mathematics, averaging is equivalent to the concept of integration, and differentiation inverts integration. Thus, sharpening spatial filters can be represented by partial derivatives.

Partial derivatives of digital functions

The first order partial derivatives of the digital image *f(x,y)* are:

$$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y) \quad \text{and} \quad \frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$$

The first derivative must be:

1) zero along flat segments (i.e. constant gray values).
2) non-zero at the outset of gray level step or ramp (edges or noise)
3) non-zero along segments of continuing changes (i.e. ramps).

The second order partial derivatives of the digital image *f(x,y)* are:

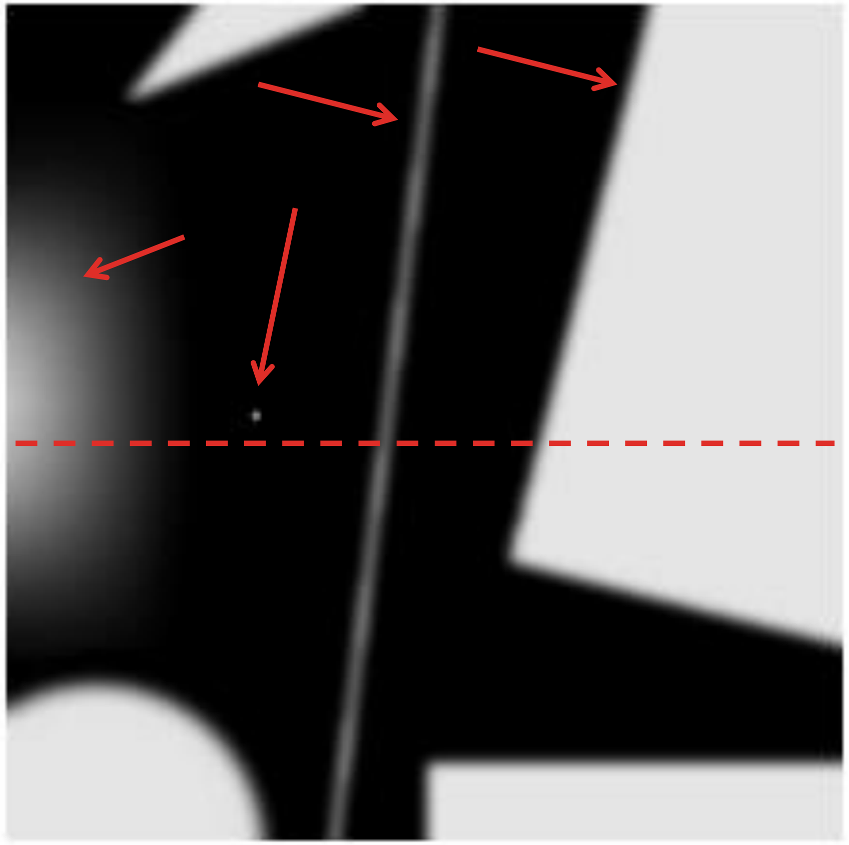
$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

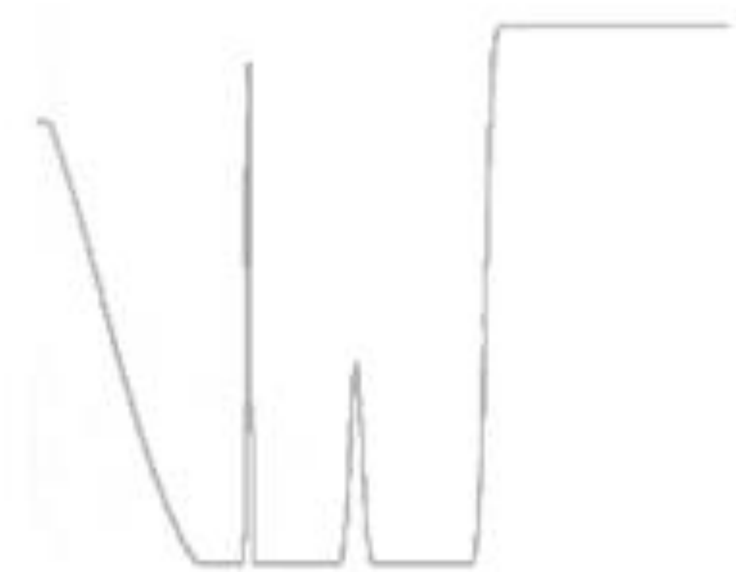
$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

The second derivative must be:

1) zero along flat segments.
2) nonzero at the outset and end of a gray-level step or ramp;
3) zero along ramps

Consider the example below:





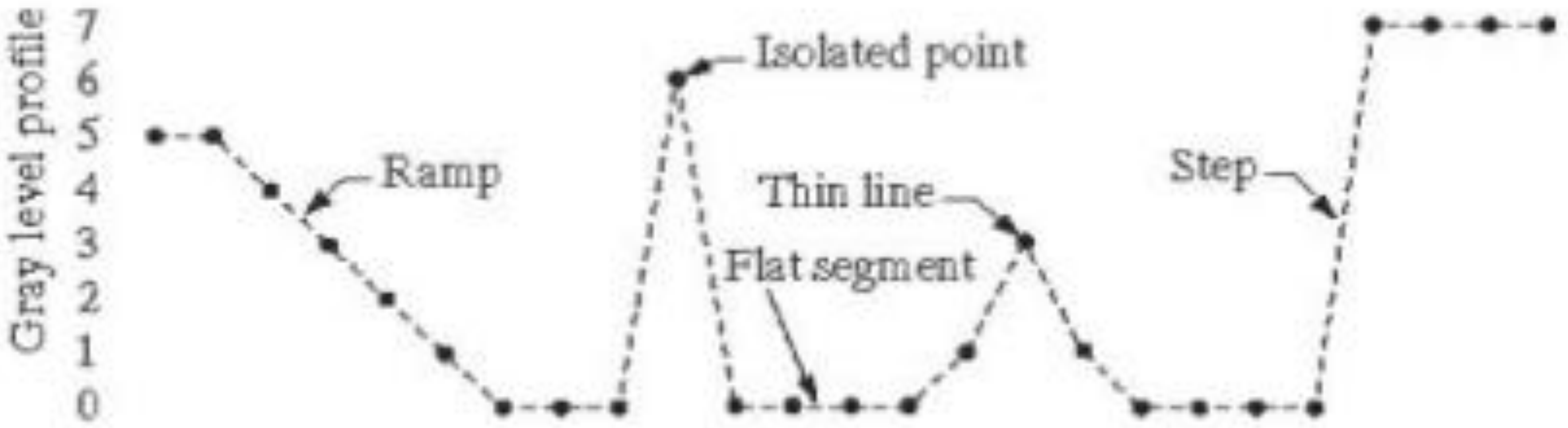
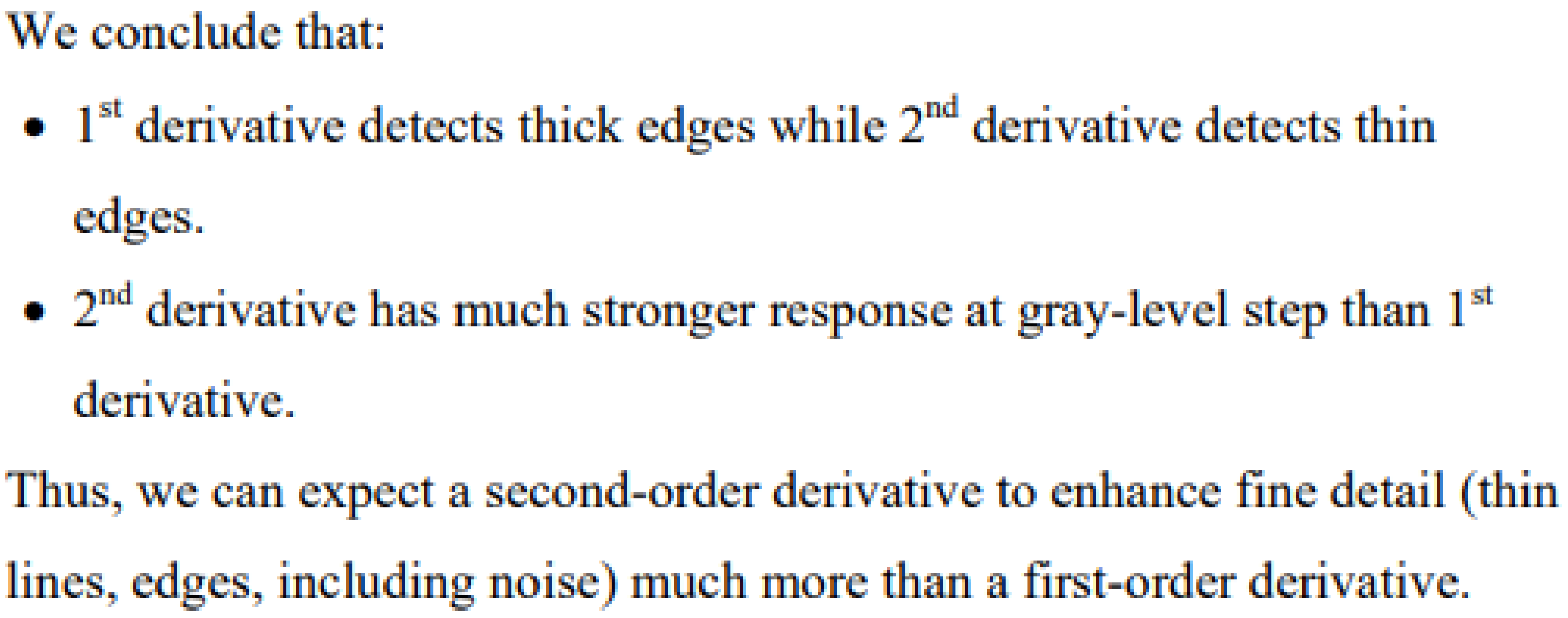


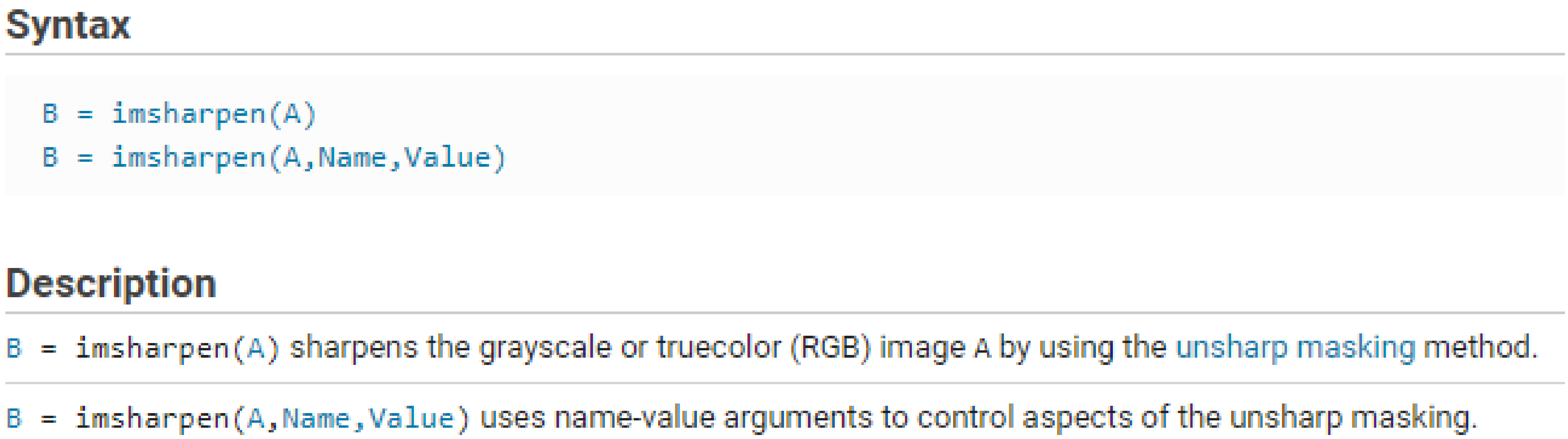
Image strip	5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7	•	•
First Derivative		−1	−1	−1	−1	−1	0	0	6	−6	0	0	0	1	2	−2	−1	0	0	0	7	0	0	0			
Second Derivative		−1	0	0	0	0	1	0	6	−12	6	0	0	1	1	−4	1	1	0	0	7	−7	0	0			



We conclude that:

- 1st derivative detects thick edges while 2nd derivative detects thin edges.
- 2nd derivative has much stronger response at gray-level step than 1st derivative.

Thus, we can expect a second-order derivative to enhance fine detail (thin lines, edges, including noise) much more than a first-order derivative.



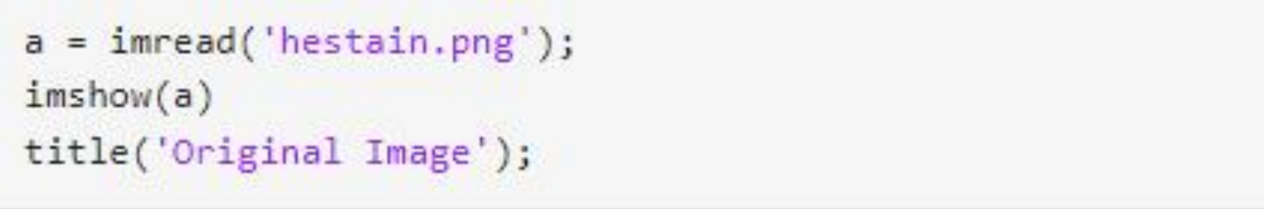
Syntax

```
B = imsharpen(A)
B = imsharpen(A,Name,Value)
```

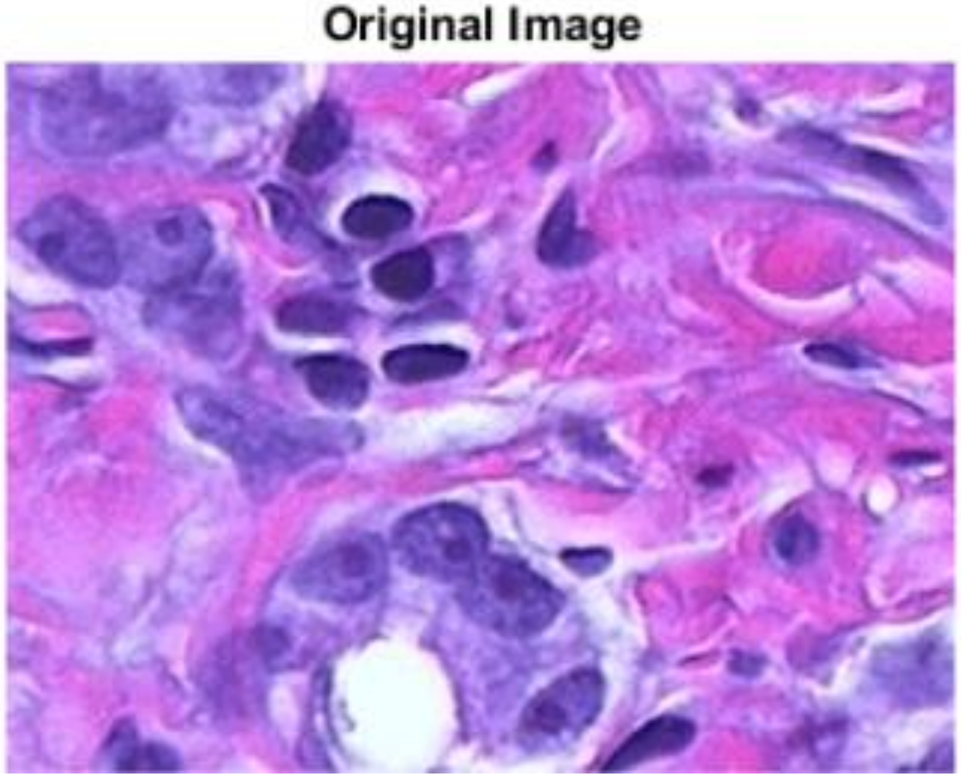
Description

`B = imsharpen(A)` sharpens the grayscale or truecolor (RGB) image `A` by using the unsharp masking method.

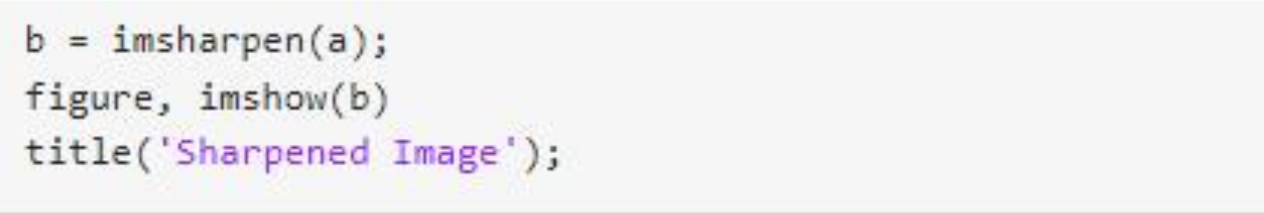
`B = imsharpen(A,Name,Value)` uses name-value arguments to control aspects of the unsharp masking.



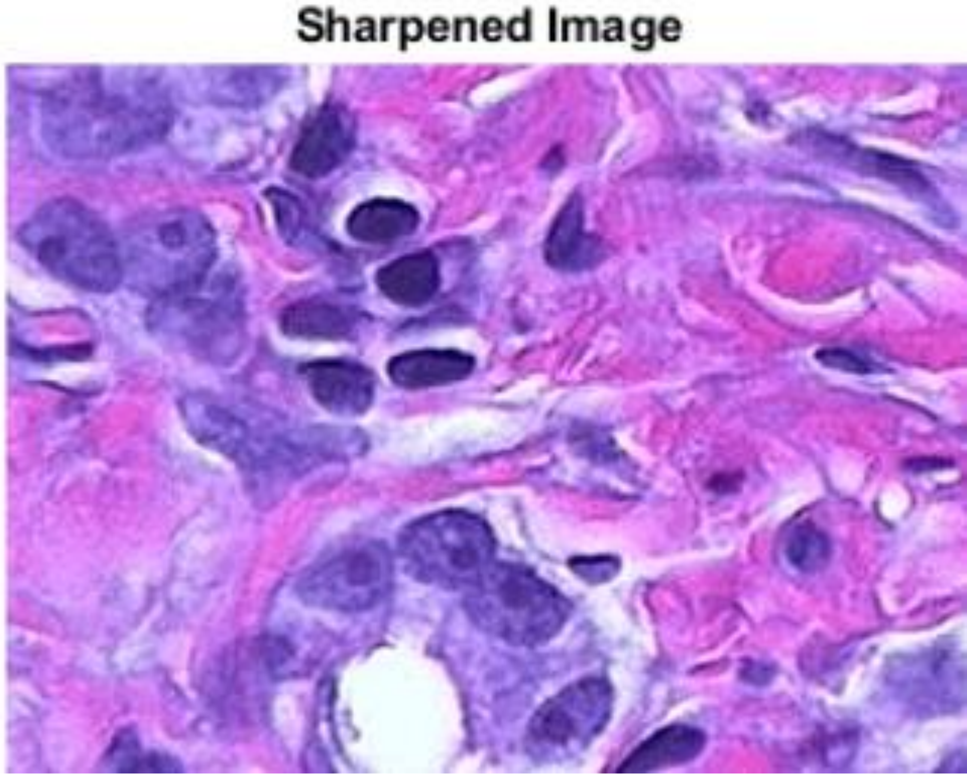
```
a = imread('hestain.png');
imshow(a)
title('Original Image');
```



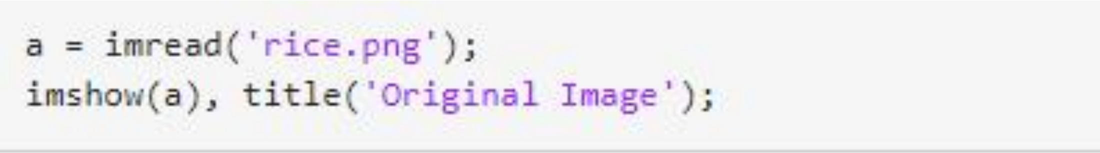
Original Image



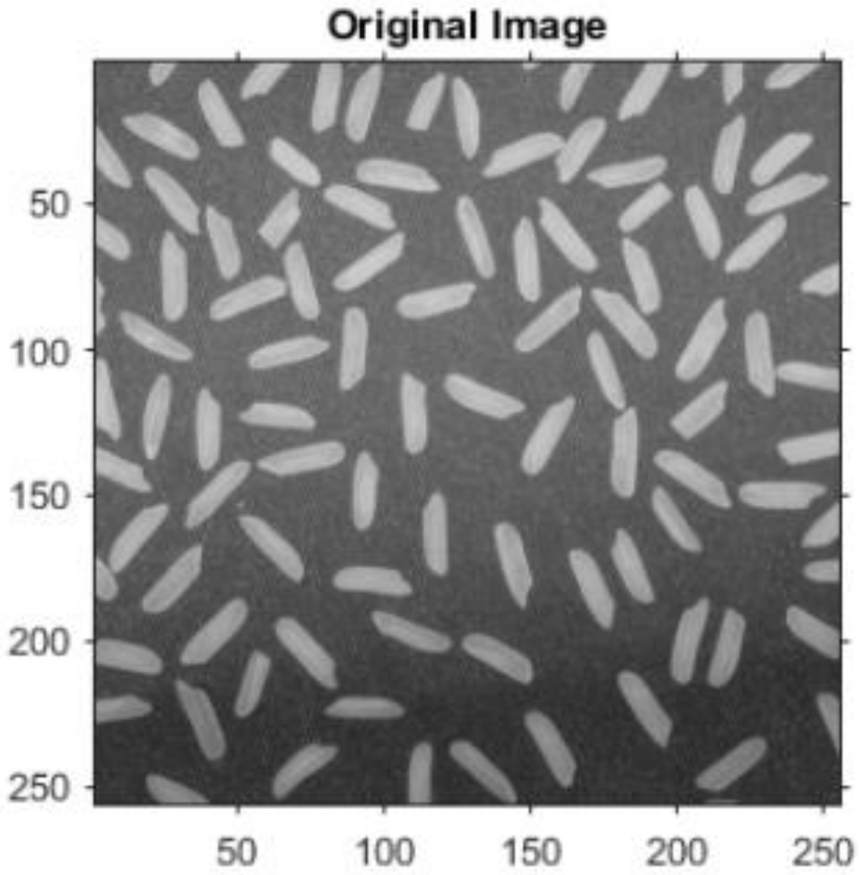
```
b = imsharpen(a);
figure, imshow(b)
title('Sharpened Image');
```

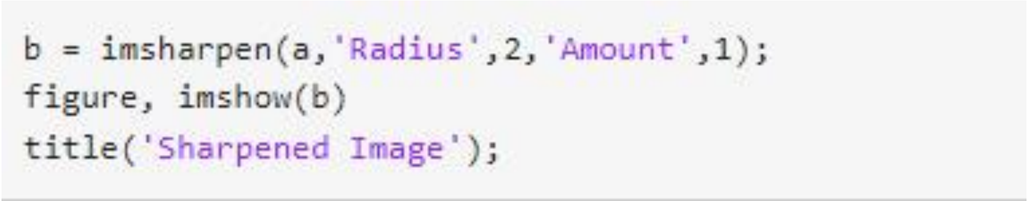


Sharpened Image

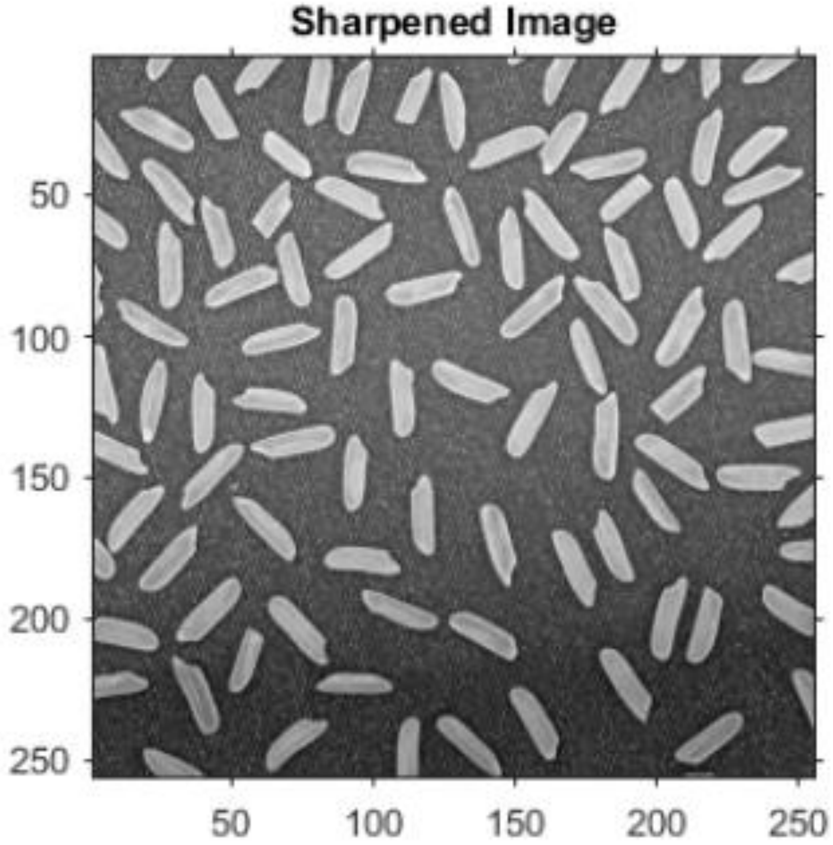


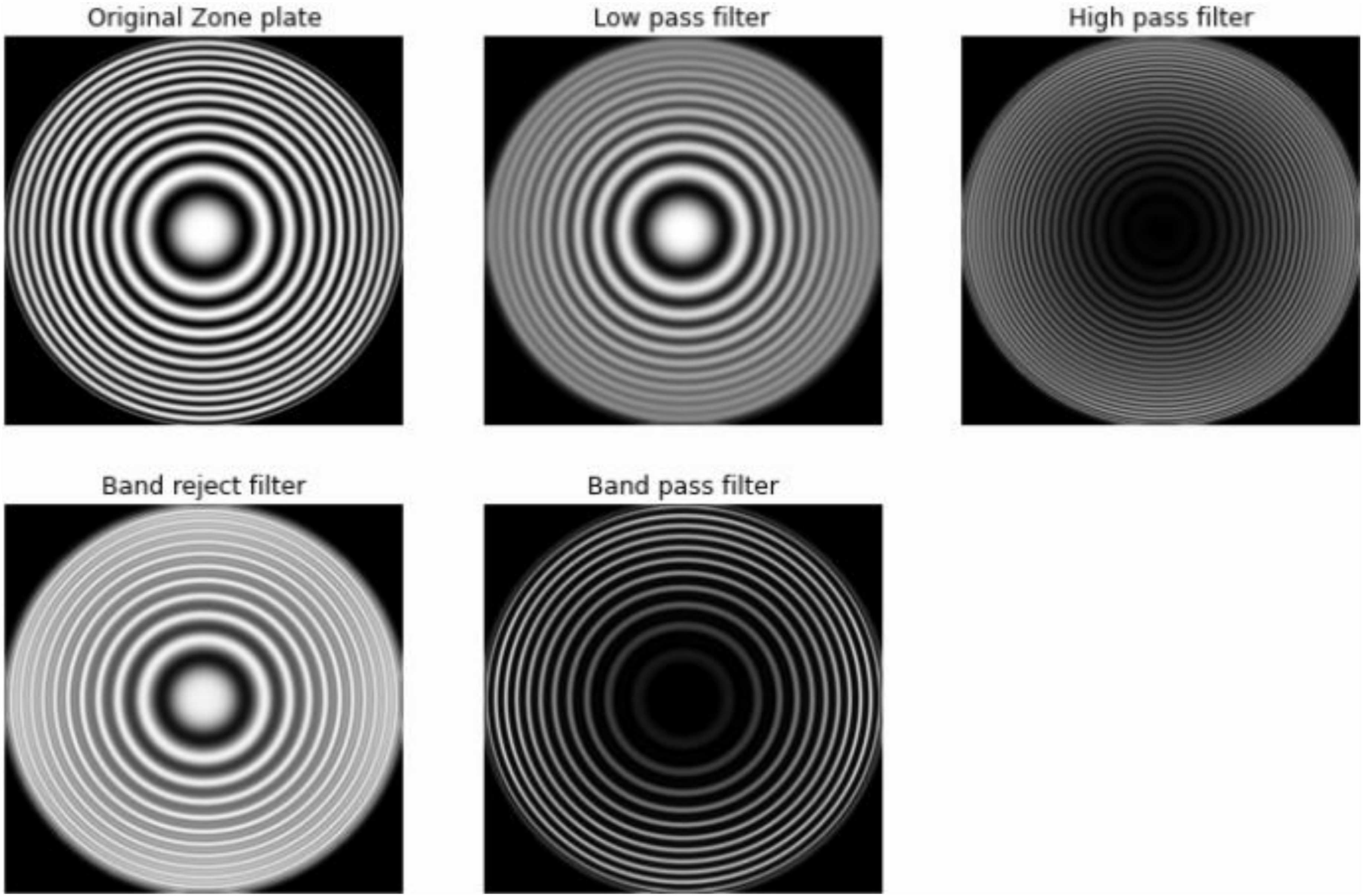
```
a = imread('rice.png');
imshow(a), title('Original Image');
```





```
b = imsharpen(a,'Radius',2,'Amount',1);
figure, imshow(b)
title('Sharpened Image');
```





Original Zone plate

Low pass filter

High pass filter

Band reject filter

Band pass filter

- **Highpass filter**

In spatial domain, a Highpass filtered image can be obtained by subtracting a Lowpass filtered image from the image itself (like unsharp mask).

- **Bandreject filter**

Similarly, a Bandreject filtered image can be obtained by adding a Lowpass filtered with a Highpass filtered image (at different threshold).

- **Bandpass filter**

And Bandpass filtered image can be obtained by subtracting the Bandreject filtered image from the image itself.

Kernels	Equation
Lowpass kernel	$lp(x, y)$
Highpass kernel	$hp(x, y) = \delta(x, y) - lp(x, y)$
Bandreject kernel	$br(x, y) = lp1(x, y) + hp2(x, y)$ $br(x, y) = lp1(x, y) + (\delta(x, y) - lp2(x, y))$
Bandpass kernel	$bp(x, y) = \delta(x, y) - br(x, y)$