Types of Image Noise

1) **Salt and pepper noise**

2) **Gaussian noise**

3) **Speckle noise**

4) **Uniform noise**

Salt and pepper noise

- **Salt-and-pepper noise** is a form of **noise** sometimes seen on images. ... This **noise** can be caused by **sharp and sudden disturbances** in the image signal. It presents itself as sparsely occurring **white and black pixels**. An effective **noise** reduction method for this type of **noise** is a **median filter** or a **morphological filter**.

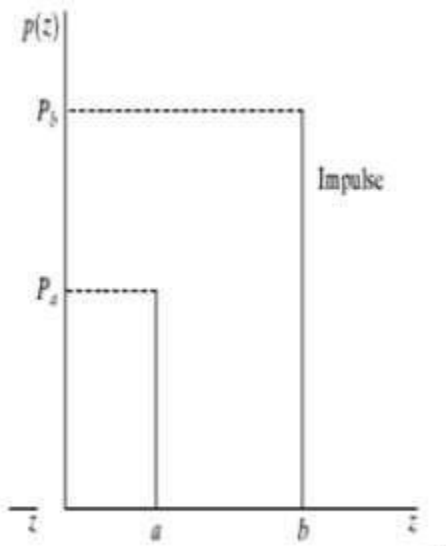
Salt and pepper noise (cont.)

Reasons:

1) By memory cell failure.
2) By malfunctioning of camera's sensor cells.
3) By synchronization errors in image digitizing or transmission.

Impulse noise:

$$p(z)=\begin{cases} p_a & for \quad z=a \\ p_b & for \quad z=b \\ 0 & otherwise \end{cases}$$



Where: p_a, p_b are the Probability Density Function (PDF), p(z) is distribution salt and pepper noise in image

Salt and pepper noise (cont.)

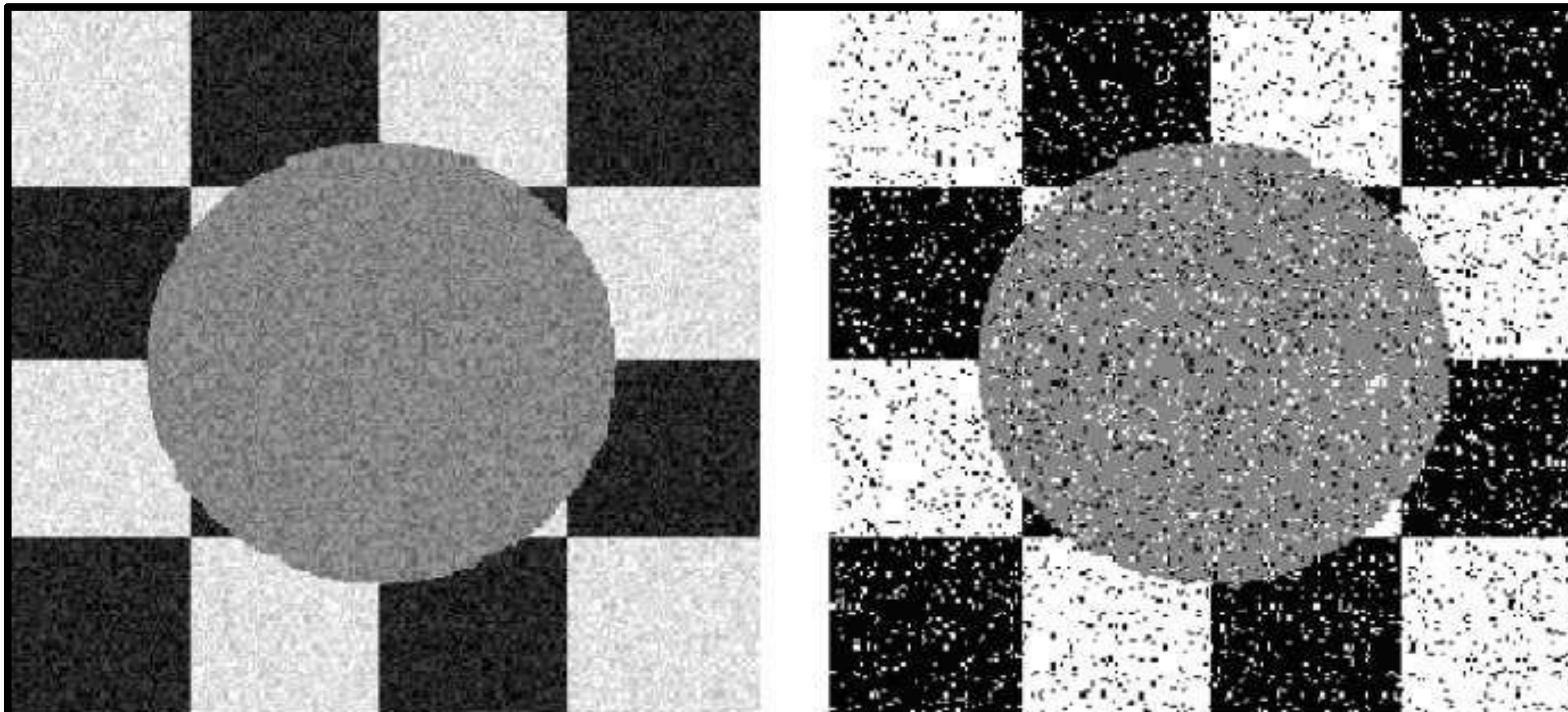


Figure. 1.11 Example of Impulse Noise

Image with Salt
and Pepper

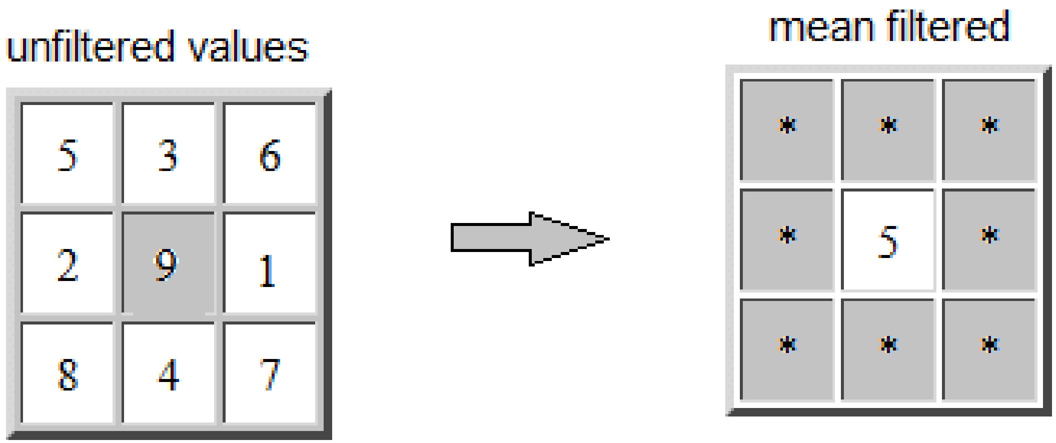
Original Image

Salt and pepper noise (cont.)

- filtering techniques :
  - ✓Mean (convolution) filtering
  - ✓Median filtering
  - ✓Gaussian filtering

- Gaussian filter is a linear type of filter which is based on Gaussian function. **It preserves edge while removing noise**. Deep Convolutional neural network (CNN) is able to handle Gaussian denoising at a certain noise level.

- Median filter is a non-linear type of filter.



$5 + 3 + 6 + 2 + 9 + 1 + 8 + 4 + 7 = 45$

$45 / 9 = 5$

123	125	126	130	140
122	124	126	127	135
118	120	150	125	134
119	115	119	123	133
111	116	110	120	130

Neighbourhood values:

115, 119, 120, 123, 124,
125, 126, 127, 150

Median value: 124

Since the median value must actually be the value of one of the pixels in the neighborhood, the median filter does not create new unrealistic pixel values when the filter overlaps an edge. For this reason **the median filter is much better at preserving sharp edges than the mean filter**.

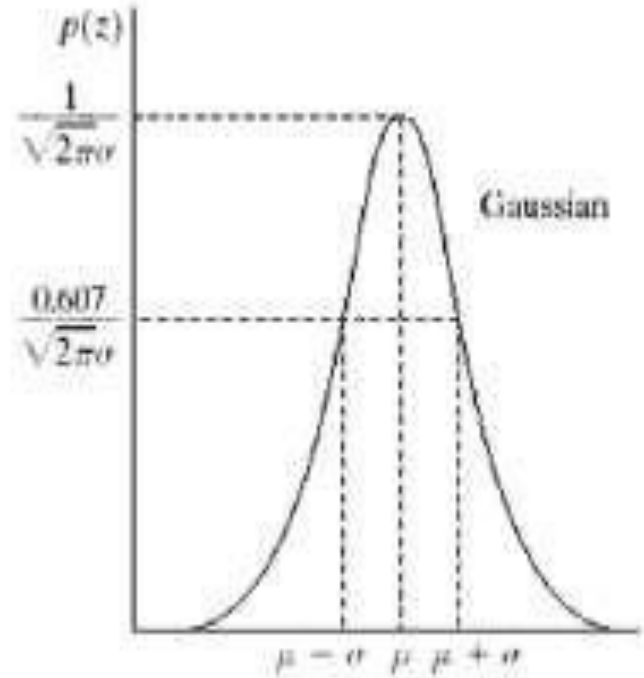
Gaussian Noise

Gaussian noise is caused by random fluctuations in the signal , its modeled by random values add to an image

This noise has a probability density function [pdf] of the normal distribution. It is also known as **Gaussian distribution.**

Gaussian noise:

$$p(z)=\frac{1}{\sqrt{2\pi\sigma}}e^{-(z-\mu)^2/2\sigma^2}$$



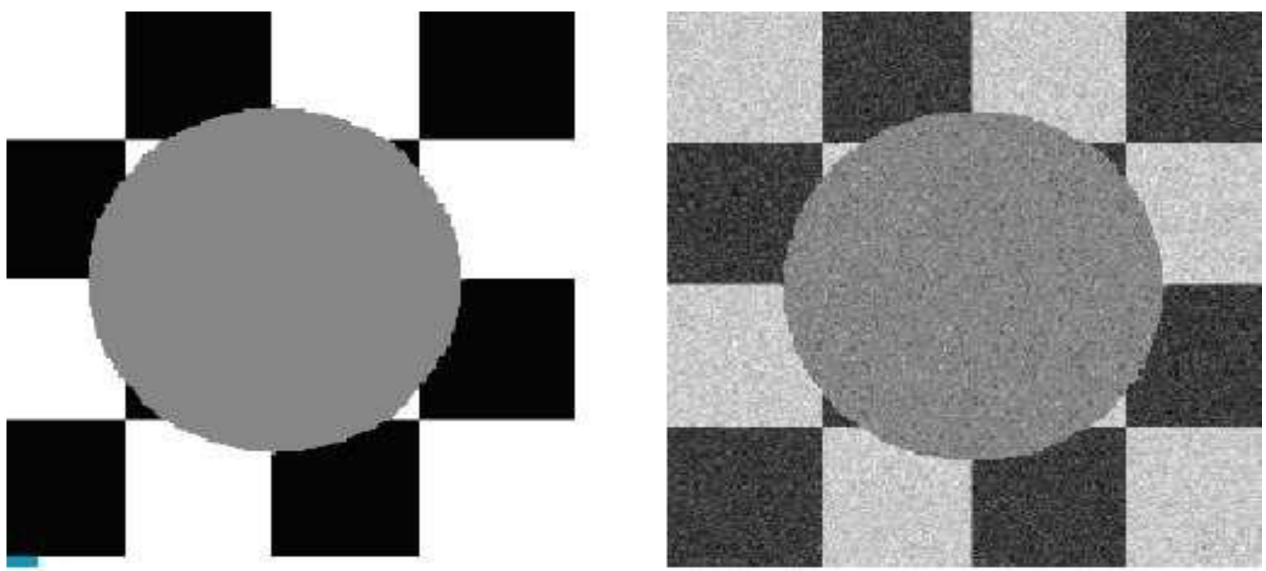
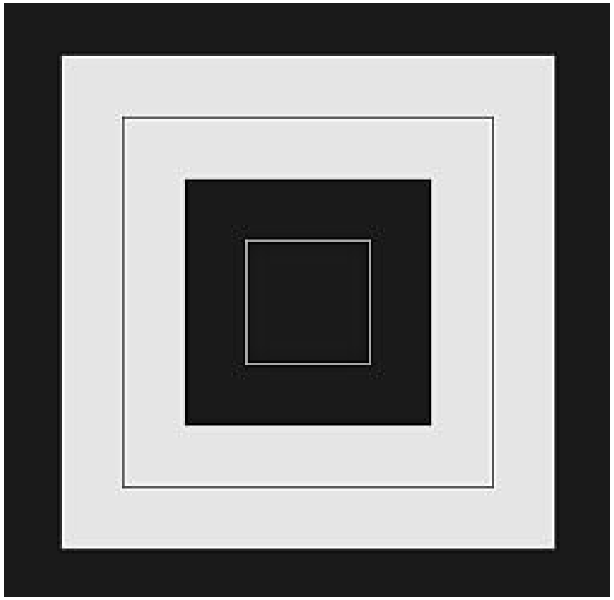
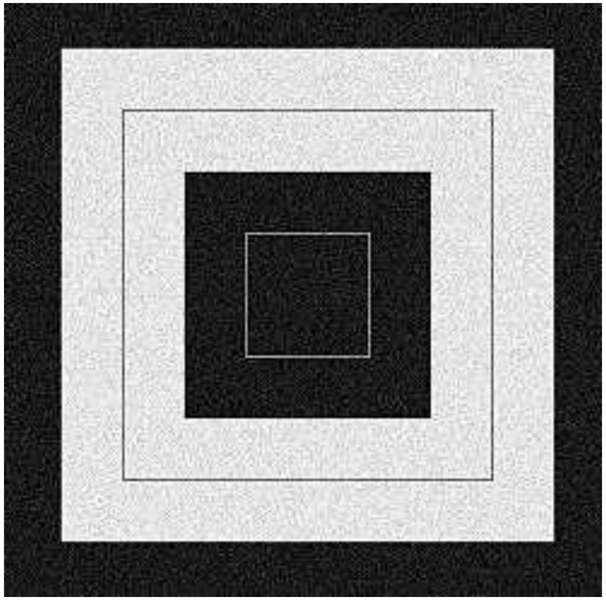


Figure. 1.9 Example of Gaussian Noise

Gaussian Noise (cont.)



Without Noise



With Gaussian Noise

Original Image

Image with Gaussian Noise

Sources of Gaussian Noise

- **Cause during image acquisition .**

  e.g. Sensor noise caused by poor illumination and/or high temperature

- **Transmission**

  e.g. Electronic circuit noise.

Gaussian Noise (cont.)

- filtering techniques :

  ✓Mean (convolution) filtering

  ✓Median filtering

  ✓Gaussian filtering

Speckle Noise

- Speckle noise can be modeled by random values multiplied by pixel values of an image
- results from random fluctuations in the return signal from an object that is no bigger than a single image-processing element.

It increases the mean grey level of a local area.

Speckle Noise

The distribution noise can be expressed by:

$$\boxed{g(n,m) = f(n,m) * u(n,m) + \xi(n,m)}$$

Where **g(n,m),** is the observed image , **u(n,m)** is the multiplicative component . and **&** ξ **(n,m)** is the additive component of the speckle noise.

Original Image

Image with Noise

Speckle Noise (cont.)

- filtering techniques :
  - ✓Mean (convolution) filtering
  - ✓Median filtering

Uniform Noise

- The **uniform noise** is caused by **quantizing the pixels** of image to a number of distinct levels is known as **quantization noise.**
- Uniform noise can be analytically described by :

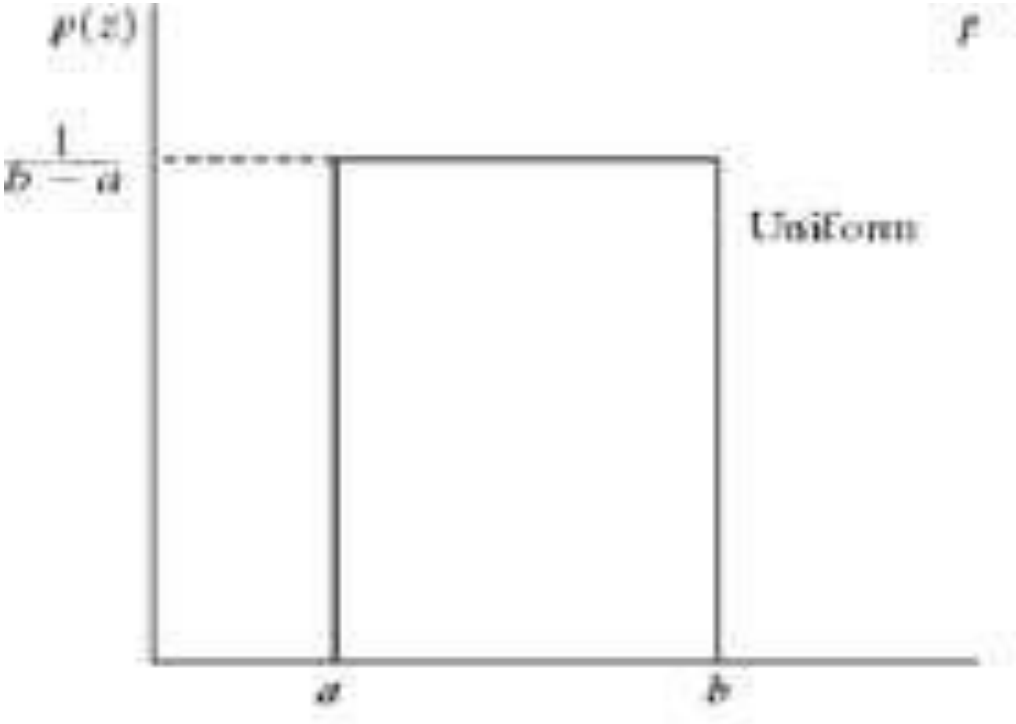
Uniform noise:

$$p(z) = \begin{cases} \frac{1}{(b-a)} & \text{if } a \le z \le b \\ 0 & \text{otherwise} \end{cases}$$

- The gray level values of the noise are evenly distributed across a specific range

Uniform Noise (cont.)

- Quantization noise has an approximately uniform distribution



Uniform Noise (cont.)

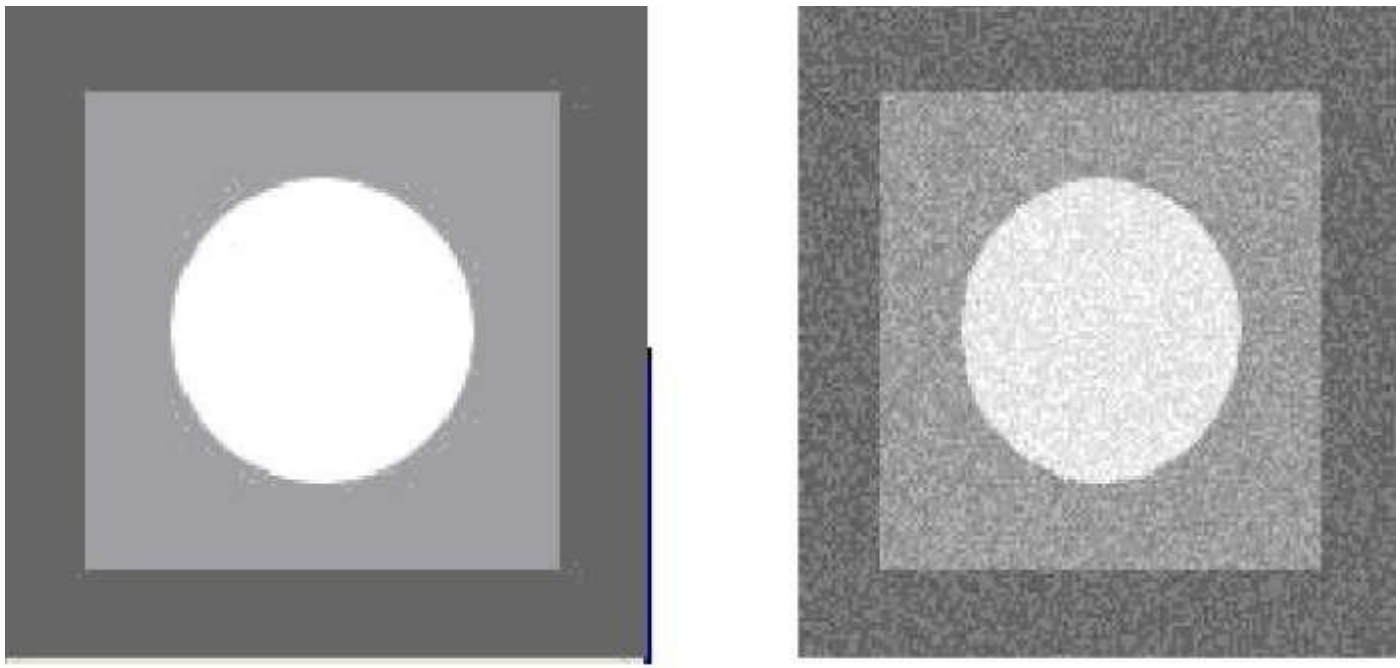
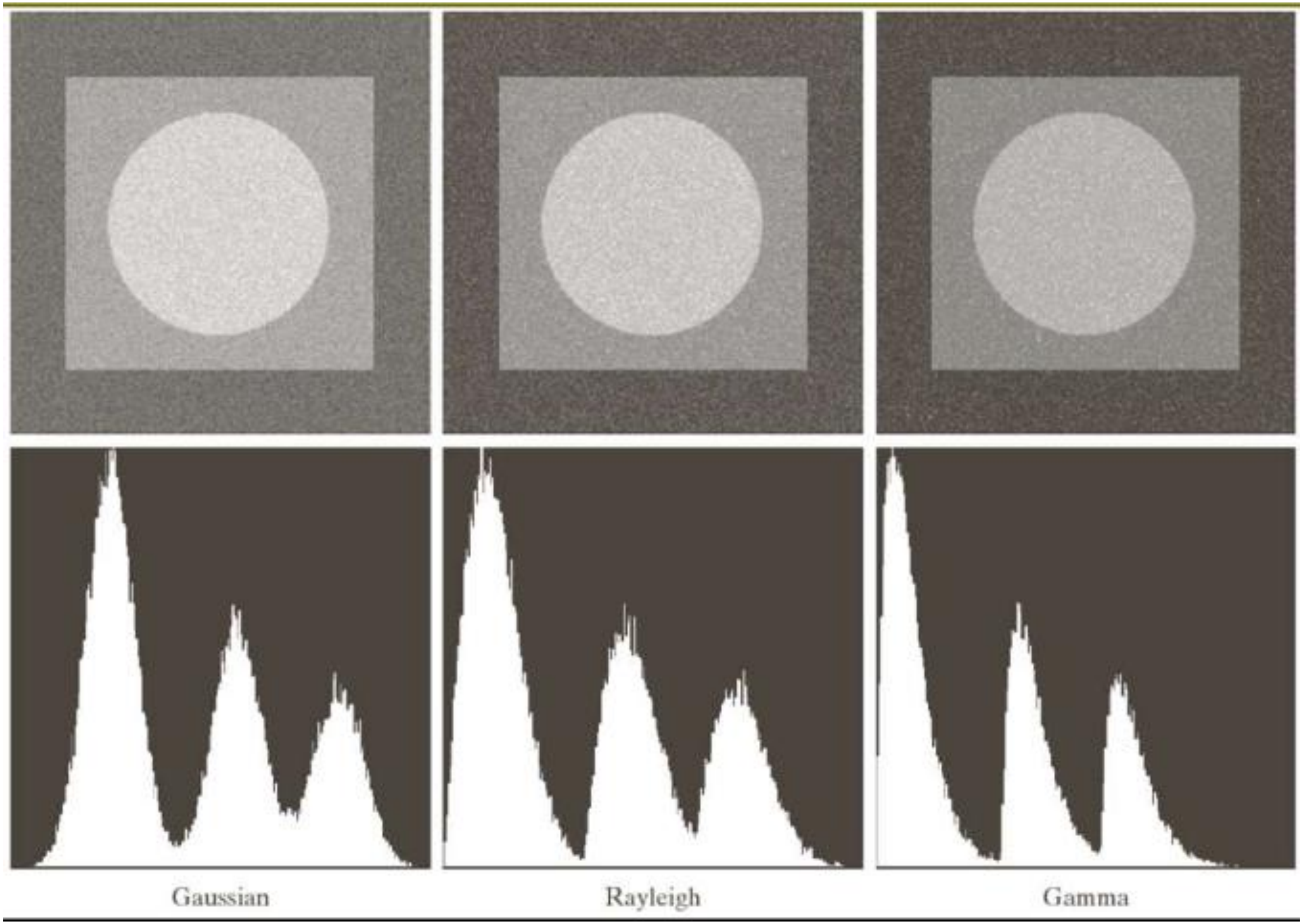


Figure. 1.7 Example of Uniform Noise

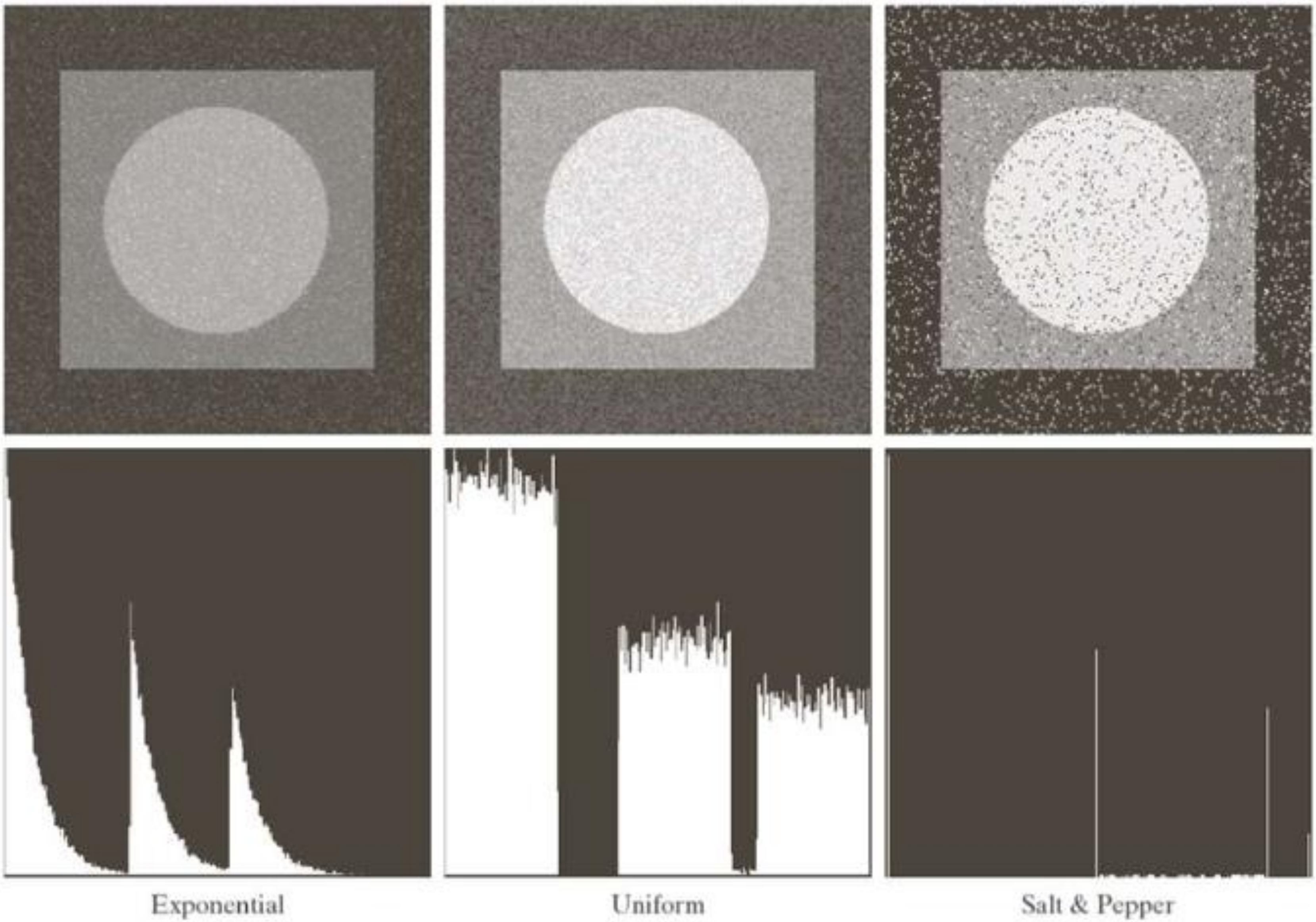
Some important noise models and their use in practice

- Gaussian noise
  - Easy to use mathematically. Therefore often used in practice even if the model is not perfect.
  - Electronic circuit noise and sensor noise due to poor illumination* or high temperature. *= rather poisson noise
- Rayleigh noise
  - Occurs in range imaging. Can model skewed histograms.
- Gamma and Exponential noise
  - Occurs in laser imaging. Can be used for approximating skewed histograms.
- Uniform noise
  - Not so practical, but can be useful in random number generation in simulations.
- Salt-and-pepper (impulse) noise
  - Quick transients due to as faulty switching during imaging
- Periodic noise
  - Electrical or electromechanical interference during image acquisition
  - Newspaper printing

An Image with various noise …



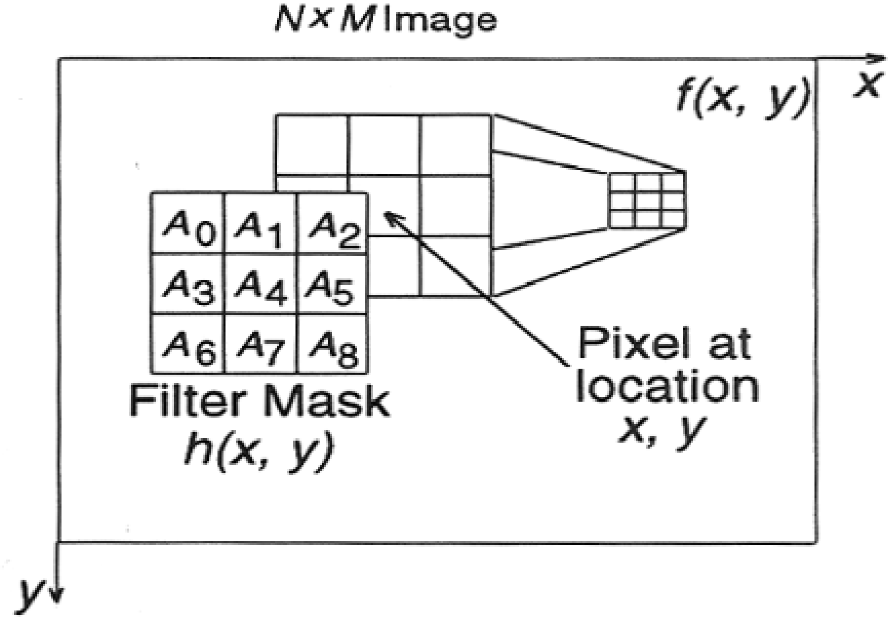
An Image with various noise …



Spatial Filtering: Graphical illustration

Spatial filtering of image f(x,y) using neighborhood mask h(x,y):

Mask:
h(x,y)= 3x3 filter
with weights as shown

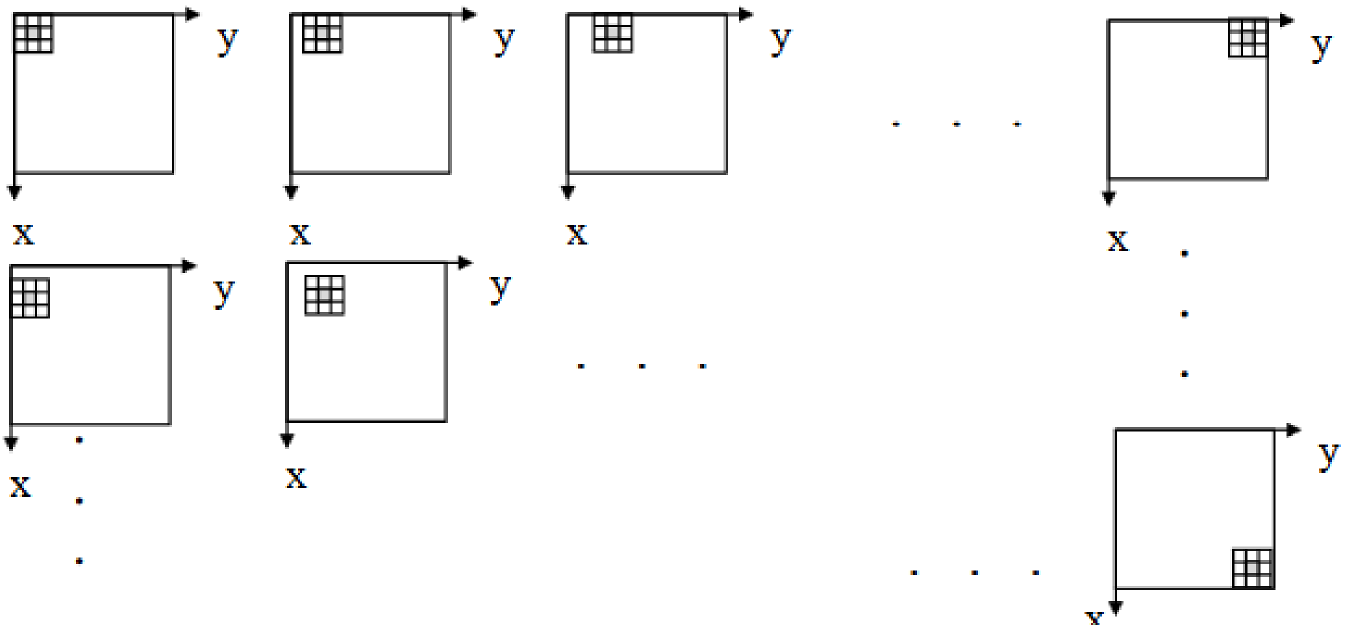


Spatial convolution:

$$g(x,y) = A_0 f(x-1,y-1) + A_1 f(x,y-1) + A_2 f(x+1,y-1) + A_3 f(x-1,y) + A_4 f(x,y) + A_5 f(x+1,y) + A_6 f(x-1,y+1) + A_7 f(x,y+1) + A_8 f(x+1,y+1)$$

⇨ Corresponds to multiplication of each pixel under mask with corresponding filter weight, and replacement of this value at the point of coordinates (x,y).

Neighborhood process, scanning all image
➡ computationally intensive.
Mask size or shape: any, but arbitrary shapes will play role in result (i.e. Math. Morphology).
Typically: small square/rectangular 2D array, odd number of elements (eg 3x3, 5x5 neighborhoods) to ease programming, centered at pixel being filtered.



- Examples of spatial averaging masks

k \ l	0	1
0	1/4	1/4
1	1/4	1/4

2x2 window

k \ l	-1	0	1
-1	1/9	1/9	1/9
0	1/9	1/9	1/9
1	1/9	1/9	1/9

3x3 window

k \ l	-1	0	1
-1	0	1/8	0
0	1/8	1/2	1/8
1	0	1/8	0

5 points weighted averaging

e.g. Assuming white noise with zero mean and variance σ_η^2.

$$y(m,n) = u(m,n) + \eta(m,n)$$

Then the spatial average:

$$v(m,n) = \frac{1}{N_W} \sum_{(k,l)\in} \sum_{W} u(m-k, n-l) + \bar{\eta}(m,n)$$

assuming equal weight

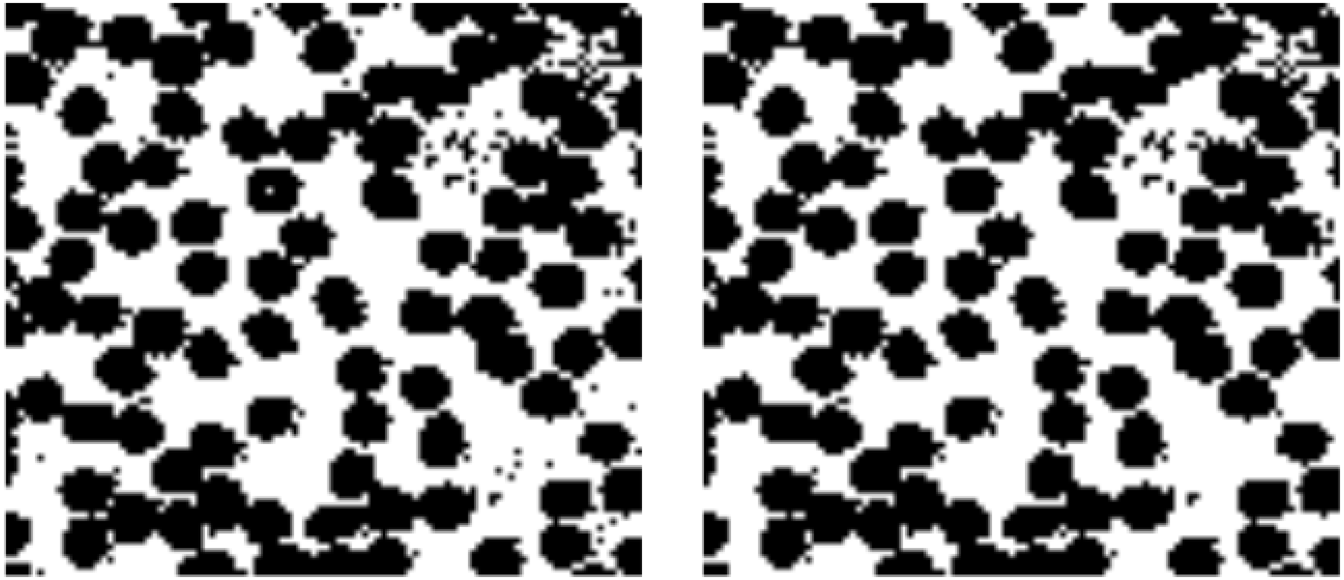
where $\bar{\eta}(m,n)$ is the spatial average of $\eta(m,n)$.

Note that $\bar{\eta}(m,n)$ has zero mean and $\bar{\sigma}_\eta^2 = \sigma_\eta^2 / N_W$

i.e. Noise power is reduced by a factor of N_W.

Remark: Spatial averaging introduces a distortion in the form of blurring.

Variances= *the squares of the standard deviations, in the values of the input or output images.*



1	1	1
1	0	1
1	1	1

⇒

1	1	1
1	1	1
1	1	1

;

0	0	0
0	1	0
0	0	0

⇒

0	0	0
0	0	0
0	0	0

X	X	X
X	L	X
X	X	X

⇒

X	X	X
X	X	X
X	X	X

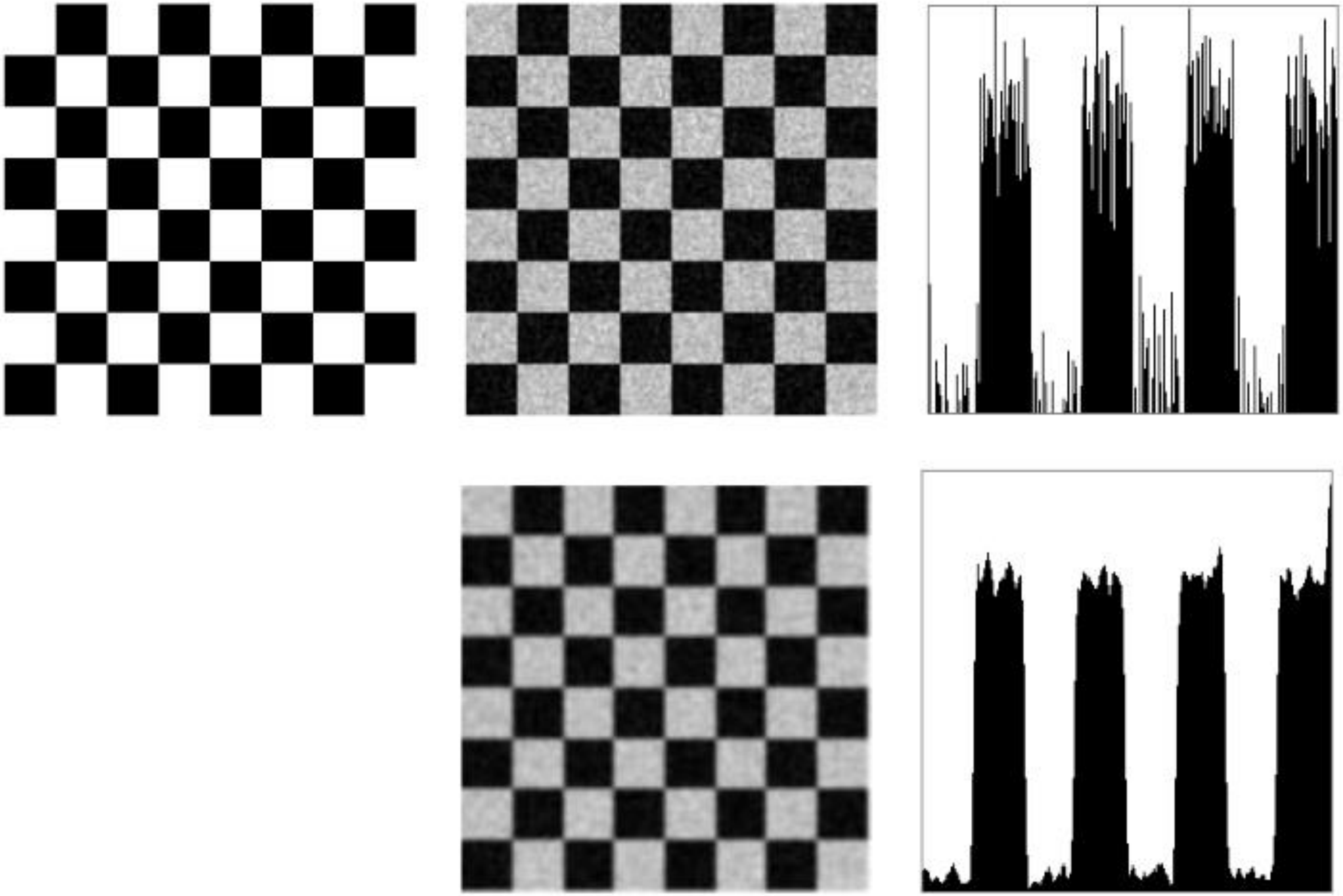
;

	X	
X	L	X
	X	

⇒

	X	
X	X	X
	X	

: Binary image of red blood cells (top left) with salt and pepper noise removed (top right). Middle row: templates showing how binary pixel neighborhoods can be cleaned. Bottom row: templates defining isolated pixel removal for a general labeled input image; (bottom left) 8-neighborhood decision and (bottom right) 4-neighborhood decision.



: Ideal image of checkerboard (top left) with pixel values of 0 in the black squares and 255 in the white squares; (top center) image with added Gaussian noise of standard deviation 30; (top right) pixel values in a horizontal row 100 from the top of the noisy image; (bottom center) noise averaged using a 5x5 neighborhood centered at each pixel; (bottom right) pixels across image row 100 from the top.

directional smoothing

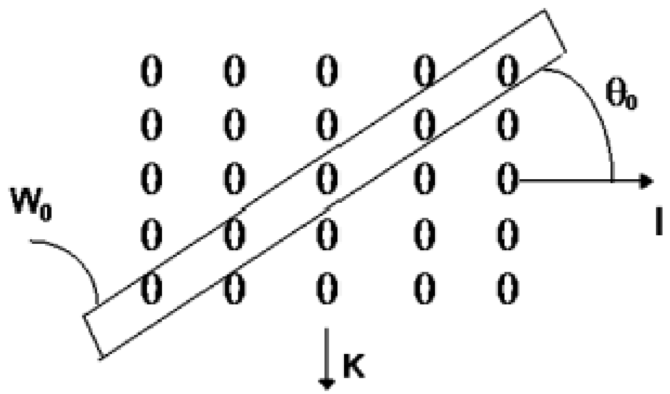
- To protect edges from blurring while smoothing.

- spatial averages are calculated in several directions , and the direction giving the smallest changes before and after filtering is selected.

$$v(m,n:\theta) = \frac{1}{N_\theta} \sum_{(k,l)\in W_\theta} \sum y(m-k,n-l)$$

The direction (θ) is found such that $|y(m,n) - v(m,n:\theta^*)|$ is minimum

- Then $v(m,n) = v(m,n:\theta^*)$ gives the desired result.



Original + Noise

3x3 Average

Directional Smoothing
(2x5, 5x2, diagonalx2)

Median filtering

- Input pixel is replaced by the **median** of the pixels contained in a window around a pixel

$$v(m, n) = median \{y(m - k, n - l), (k, l) \in W\}$$

- The algorithm requires arranging the pixels in an increasing or decreasing order and picking the middle value.
- For **Odd window size** is commonly used [3*3; 5*5; 7*7]
- For **even window size the average of two middle values** is taken.

- Median filter properties:

1- Non-linear filter

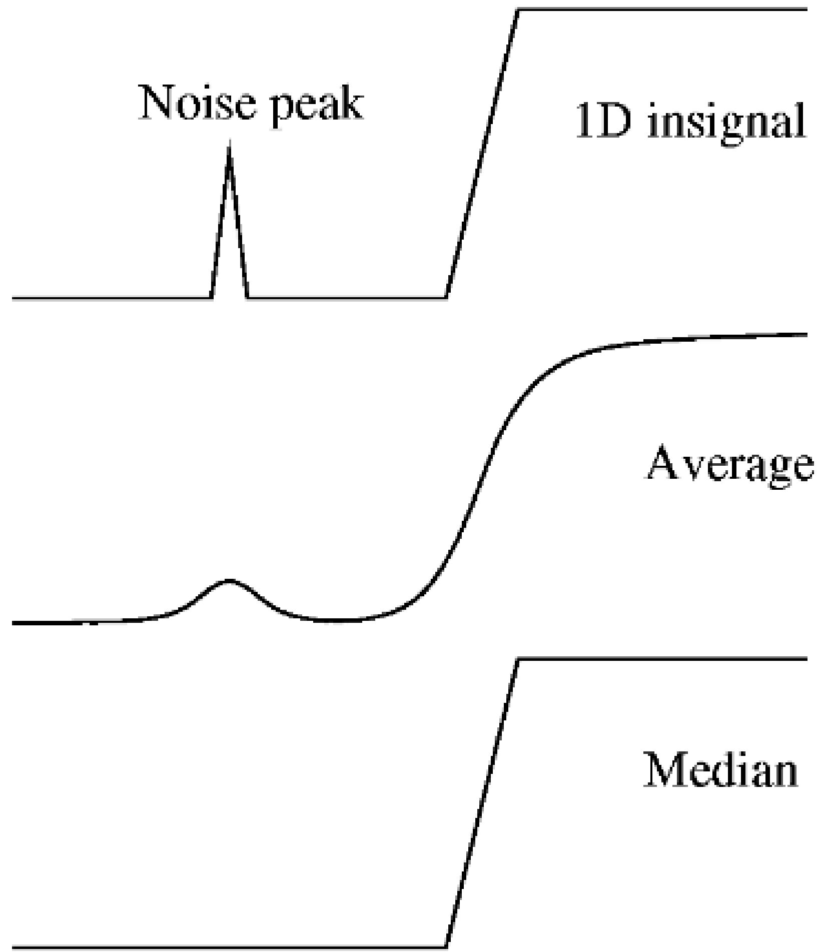
$$\text{median}\{x(m) + y(m)\} \neq \text{median}\{x(m)\} + \text{median}\{y(m)\}$$

2-Performes very well on images containing binary noise , poorly when the noise is Gaussian.

3-performance is poor in case that the number of noise pixels in the window is greater than or half the number of pixels in the window.

Properties of the median filter

- Edges are preserved.
- Noise is suppressed (especially salt-and-pepper noise).
- Thin lines are destroyed.
- Smooth surfaces arise.



30	10	20
10	250	25
20	25	30

10, 10, 20, 20, 25, 25, 30, 30, 250

median

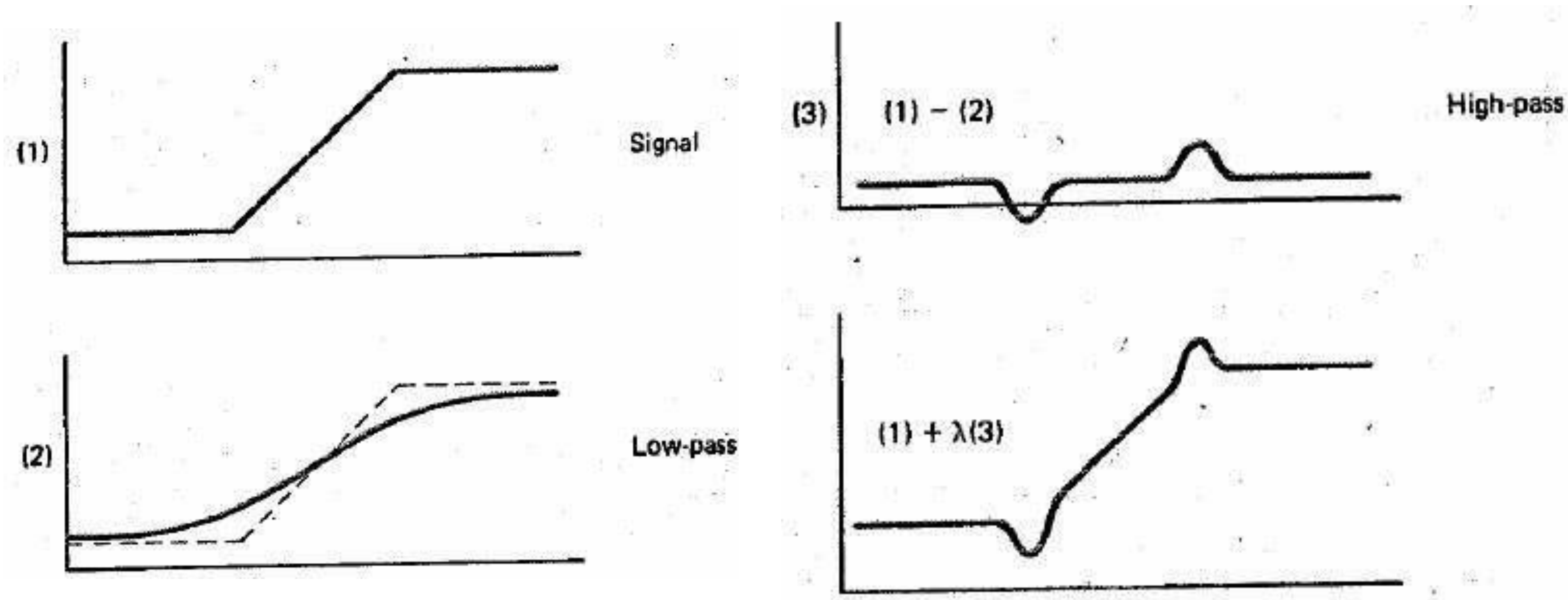
Median + Average: average the k central values.

$$10, 10, \underbrace{20, 20, 25, 25, 30}_{24}, 30, 250$$

Combined Approach

Unsharp Masking and Crispening

- The unsharp masking technique is used commonly in printing industry for crispening the edges.
- It is applied by subtracting an unsharp or smoothed or low-pass filtered version of an image from the original image.
- It is equivalent to adding the gradient, or high-pass signal to the image as shown in figure.



Unsharp masking and crispening cont.

- Unsharp masking operation can be represented by :

$$v(m,n) = u(m,n) + \lambda g(m,n)$$

Where $\lambda > 0$ and $g(m,n)$ is a suitably defined gradient at (m,n).

$$g(m,n) \triangleq u(m,n) - \tfrac{1}{4}[u(m-1,n) + u(m,n-1) + u(m+1,n) + u(m,n-1)]$$

- A commonly used gradient function is the discrete laplacian.

Low-pass filters are useful for: noise smoothing and interpolation .

High-pass Filters are useful in: extracting edges and in sharpening images.

Band-pass filters are useful in: the enhancement of edges and other high- pass characteristics in the presence of noise.

Example1

Fig. 6.21 The results of LPF (Fig. c), HPF (Fig. b),BPF (Fig. d) for a grey level image (Fig. a – original image)

- Example2

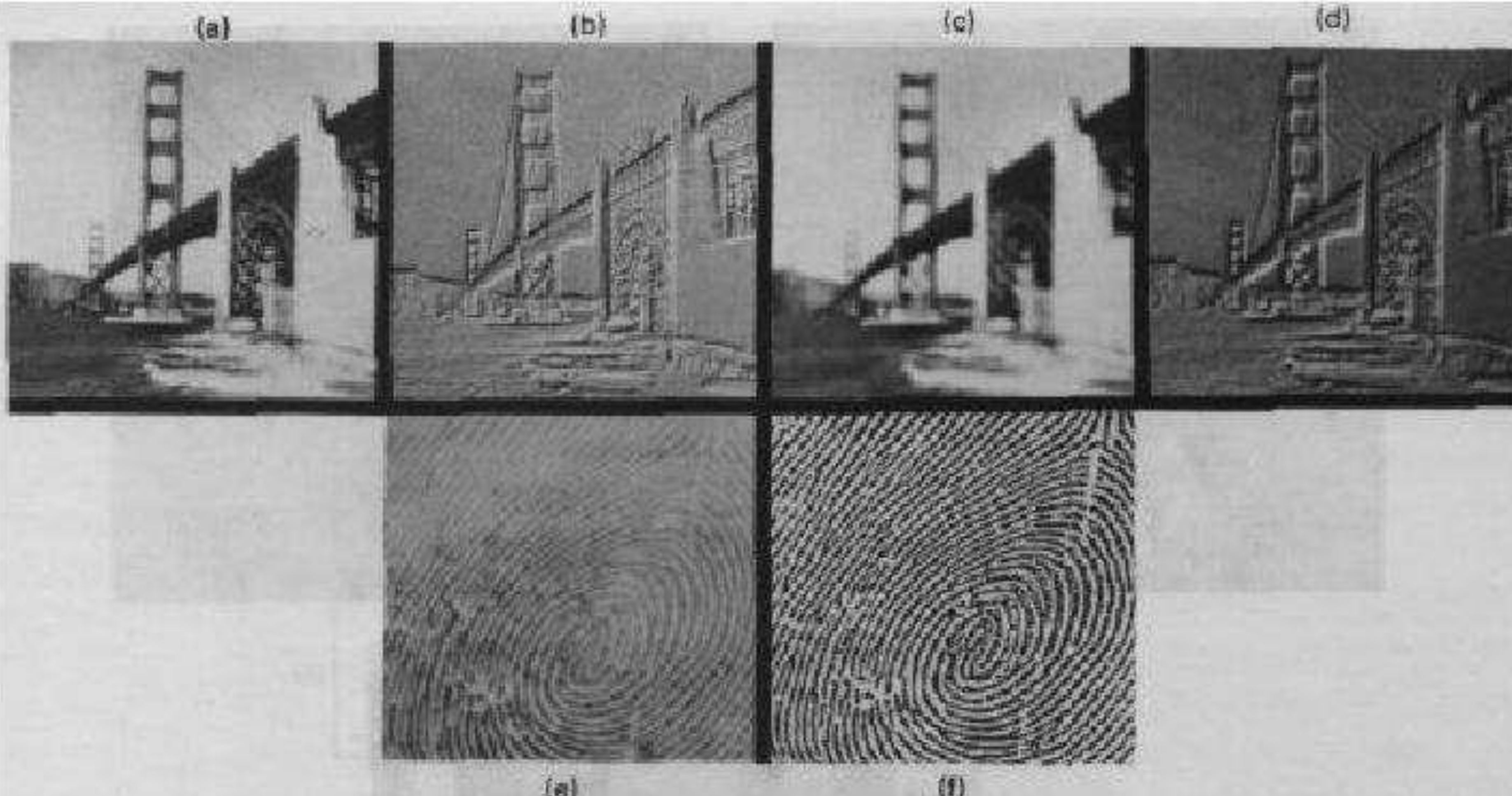


Figure 7.26 Spatial filtering examples.
Top row: original, high-pass, low-pass and band-pass filtered images.
Bottom row: original and high-pass filtered images.

Inverse contrast mapping & statistical scaling

- The ability of our visual system to detect an object in the uniform Background depends on it's size and the contrast ratio as:

$$\gamma = \sigma / \mu$$

•where μ is the average luminance of object;

σ is the standard deviation of the luminance of the object plus it's surround. Now consider the inverse contrast ratio transformation

$$v(m,n) = \frac{\mu(m,n)}{\sigma(m,n)}$$

Where $\mu(m,n)$ and $\sigma(m,n)$ are the local mean and standard deviation of $u(m,n)$ measured over a window W and are given by:

$$\mu(m,n) = \frac{1}{N_w (k,l) \in w} \sum\sum u(m-k, n-1)$$

$$\sigma(m,n) = \left[\frac{1}{N_w (k,l) \in w} \sum\sum [u(m-k, n-1) - \mu(m,n)]^2 \right]^{1/2}$$

This transformation generates an image , where the weak(low contrast) edges are enhanced.

- A special case of this transformation

$$v(m, n) = \frac{u(m, n)}{\sigma(m, n)}$$

- which scales each pixel by it's standard deviation to generate an image whose pixels have unity variance .

- This mapping is also called statistical scaling.

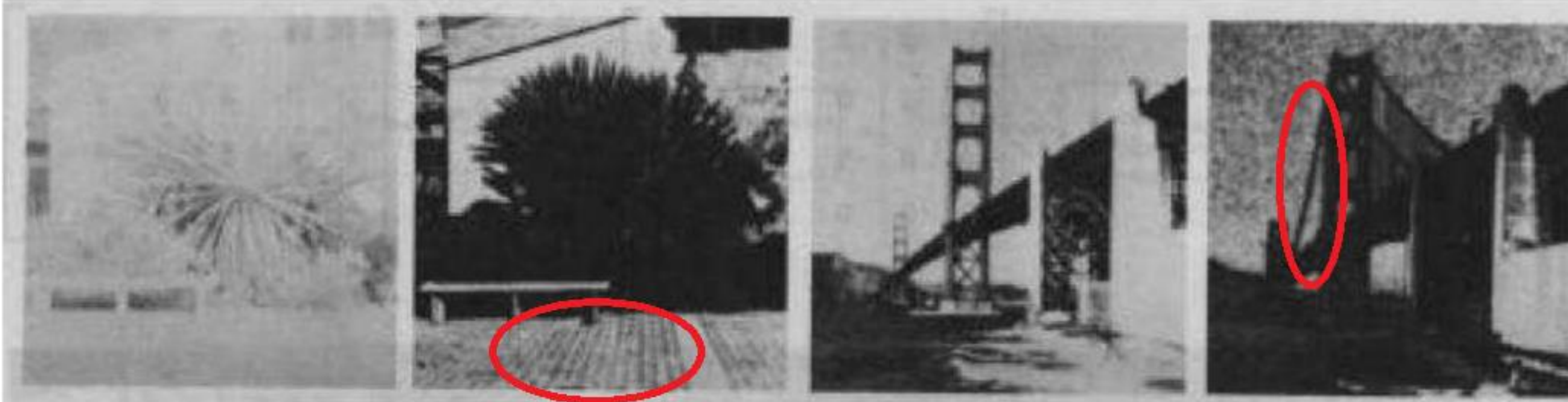


Figure 7.27 Inverse contrast ratio mapping of images. Faint edges in the originals have been enhanced. For example, note the bricks on the patio and suspension cables on the bridge.