CPE409 Image Processing

Part 6
Image Restoration and Reconstruction

Assist. Prof. Dr. Caner ÖZCAN

Things which we see are not by themselves what we see...
It remains completely unknown to us what the objects may be by themselves
and apart from the receptivity of our senses. We know nothing but our
manner of perceiving them. ~Immanuel Kant

Outline

5. Image Restoration and Reconstruction
   - A Model of the Image Degradation/Restoration Process
   - Noise Models
   - Restoration in the Presence of Noise Only—Spatial Filtering
   - *Periodic Noise Reduction by Frequency Domain Filtering*
   - *Linear, Position-Invariant Degradations*
   - *Estimating the Degradation Function*
   - *Inverse Filtering*
   - *Minimum Mean Square Error (Wiener) Filtering*
   - *Constrained Least Squares Filtering*
   - *Geometric Mean Filter*
   - *Image Reconstruction from Projections*

Image Restoration

- Image restoration: recover an image that has been degraded by using a prior knowledge of the degradation phenomenon.

- Model the degradation and applying the inverse process in order to recover the original image.

Noise Sources

- The principal sources of noise in digital images arise during **image acquisition and/or transmission**
  - Image acquisition e.g., light levels, sensor temperature, etc.
  - Transmission

    e.g., lightning or other atmospheric disturbance in wireless network
- With the exception of spatially periodic noise, we assume
  - Noise is independent of spatial coordinates
  - Noise is uncorrelated with respect to the image itself

Gaussian Noise

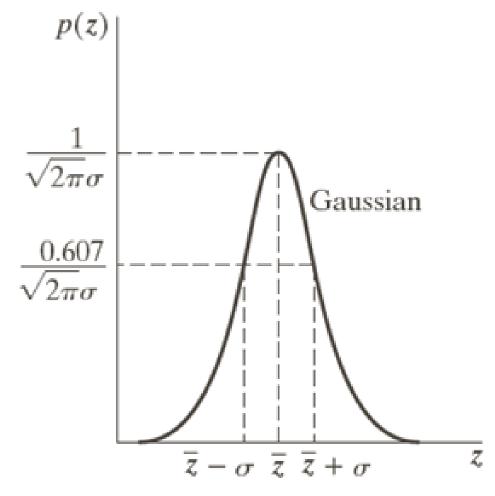
The PDF of Gaussian random variable, z, is given by

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

where, z represents intensity

$\bar{z}$ is the mean (average) value of z

σ is the standard deviation



Gaussian Noise

The PDF of Gaussian random variable, z, is given by

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

- 70% of its values will be in the range

$$\left[(\mu-\sigma),(\mu+\sigma)\right]$$

- 95% of its values will be in the range

$$\left[(\mu-2\sigma),(\mu+2\sigma)\right]$$

Rayleigh Noise

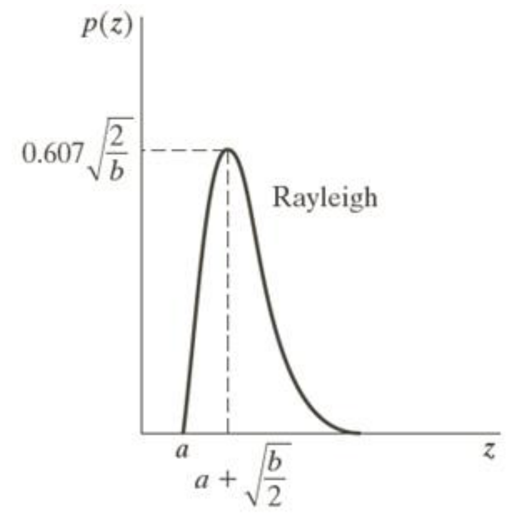
The PDF of Rayleigh noise is given by

$$p(z) = \begin{cases} \frac{2}{b}(z-a)e^{-(z-a)^2/b} & \text{for } z \geq a \\ 0 & \text{for } z < a \end{cases}$$

The mean and variance of this density are given by

$$\bar{z} = a + \sqrt{\pi b / 4}$$

$$\sigma^2 = \frac{b(4-\pi)}{4}$$



Erlang (Gamma) Noise

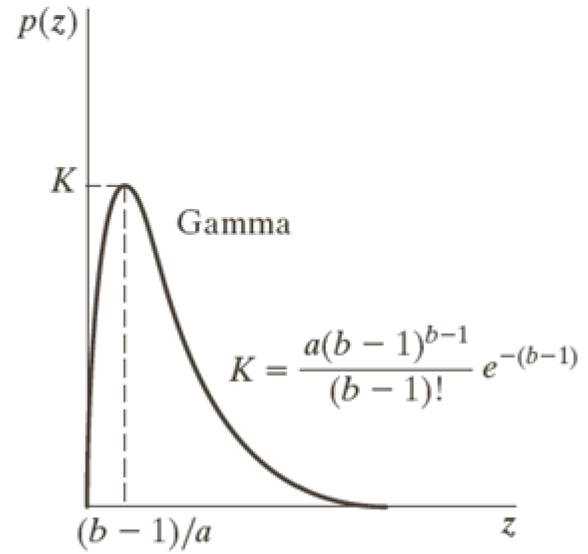
The PDF of Erlang noise is given by

$$p(z) = \begin{cases} \dfrac{a^b z^{b-1}}{(b-1)!} e^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < a \end{cases}$$

The mean and variance of this density are given by

$$\bar{z} = b / a$$

$$\sigma^2 = b / a^2$$



Exponential Noise

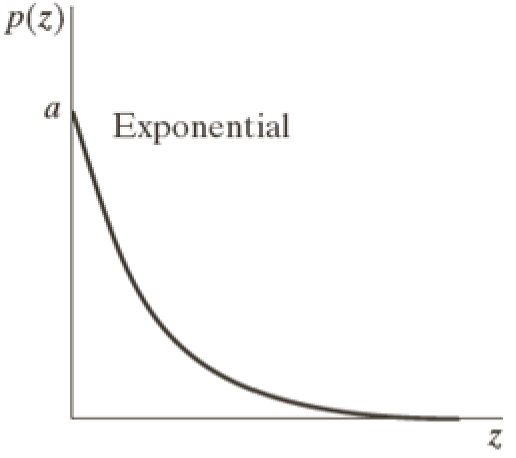
The PDF of exponential noise is given by

$$p(z) = \begin{cases} ae^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < a \end{cases}$$

The mean and variance of this density are given by

$$\bar{z} = 1/a$$

$$\sigma^2 = 1/a^2$$



Uniform Noise

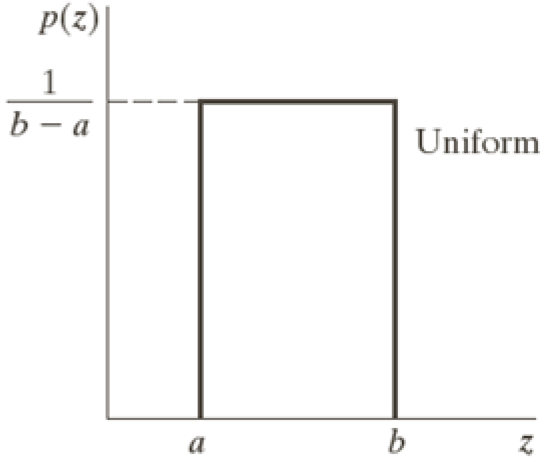
The PDF of uniform noise is given by

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

The mean and variance of this density are given by

$$\bar{z} = (a+b)/2$$

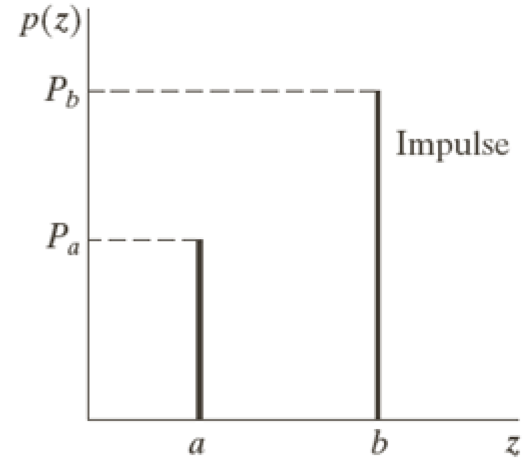
$$\sigma^2 = (b-a)^2 / 12$$



Impulse (Salt-and-Pepper) Noise

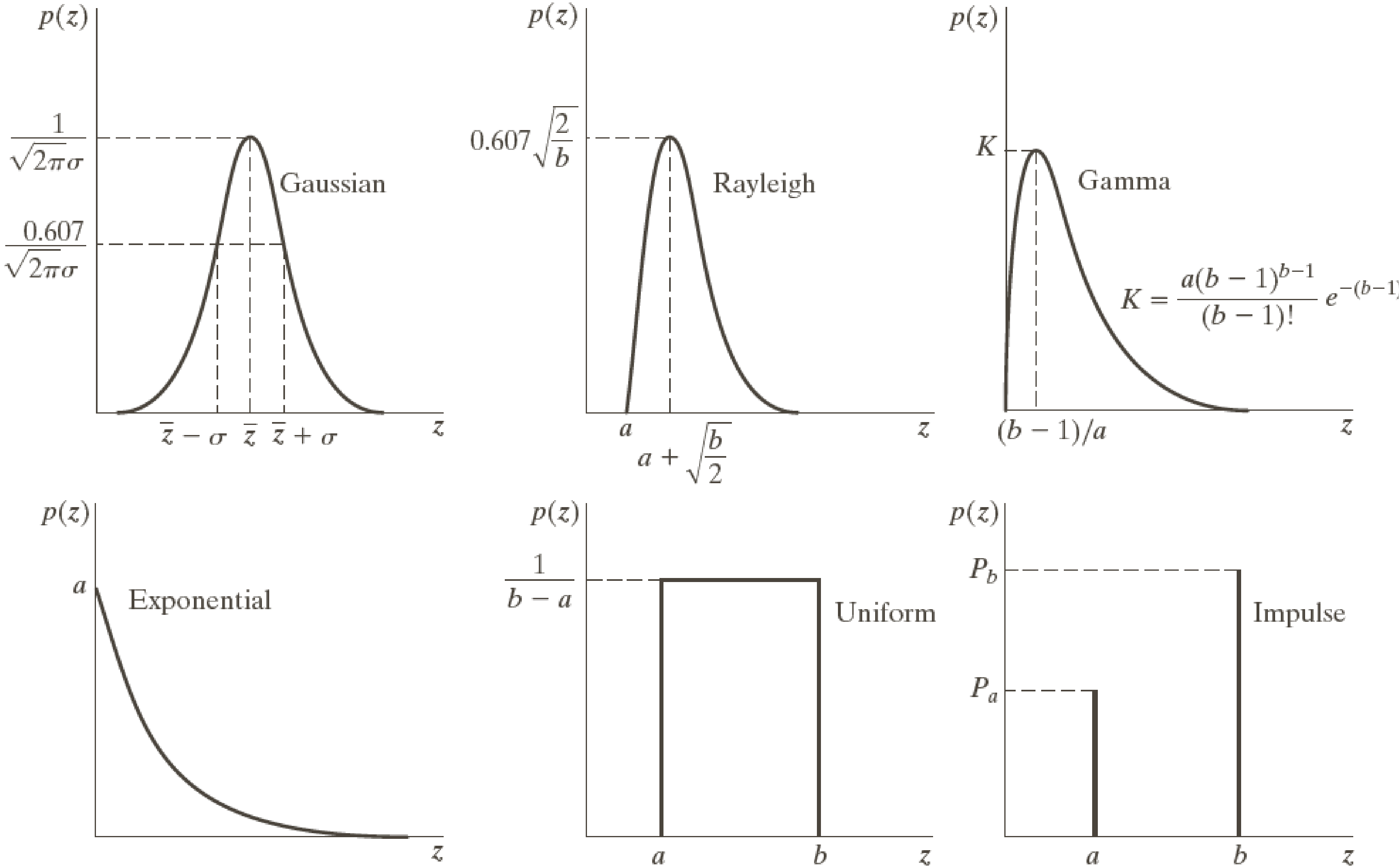
The PDF of (bipolar) impulse noise is given by

$$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$$



if $b > a$, gray-level b will appear as a light dot, while level a will appear like a dark dot.

If either P_a or P_b is zero, the impulse noise is called *unipolar*



a	b	c
d	e	f

FIGURE 5.2 Some important probability density functions.

Code Examples – Gaussian Noise

```
img = cv2.imread("input.jpg")
cv2.imshow("Original",img)

def gaussian_noise(image):
   row,col,ch = image.shape
   mean = 0
   var = 0.05
   sigma = var**0.5

   gauss = np.random.normal(mean,sigma,(row,col,ch))
   gauss = gauss.reshape(row,col,ch)
   noisy = image + gauss

   return noisy

img = cv2.imread("input.jpg")
img = img/255
noise_img = gaussian_noise(img)
cv2.imshow("Gaussian Noise",noise_img)
cv2.waitKey(0)
```

Code Examples - Salt&Pepper Noise

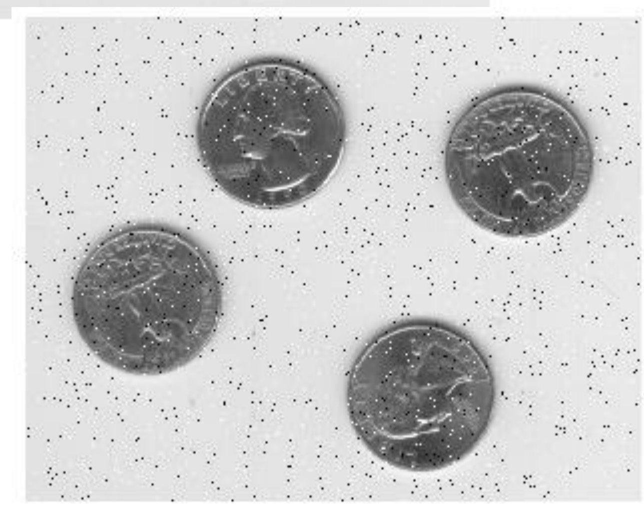
```
def saltPepperNoise(image):
    row,col,ch = image.shape
    s_vs_p = 0.5
    amount = 0.04
    noisy = np.copy(image)

    num_salt = int(np.ceil(amount*image.size*s_vs_p))
    corrds = [np.random.randint(0,i-1,num_salt) for i in image.shape]
    noisy[corrds] = 1

    num_pep = int(np.ceil(amount*image.size*s_vs_p))
    corrds = [np.random.randint(0,i-1,num_pep) for i in image.shape]
    noisy[corrds] = 0

    return noisy

img = cv2.imread("input.jpg")
img = img/255
noise_img = saltPepperNoise(img)
cv2.imshow("Gaussian Noise",noise_img)
cv2.waitKey(0)
```

Code Examples

- Read a grayscale image and display it.

```
img = cv2.imread("eight.jpg");
cv2.imshow("Original",img)
```

```
J = saltPepperNoise(img)
cv2.imshow("Gaussian Noise",J)
```



Examples of Noise: Original Image

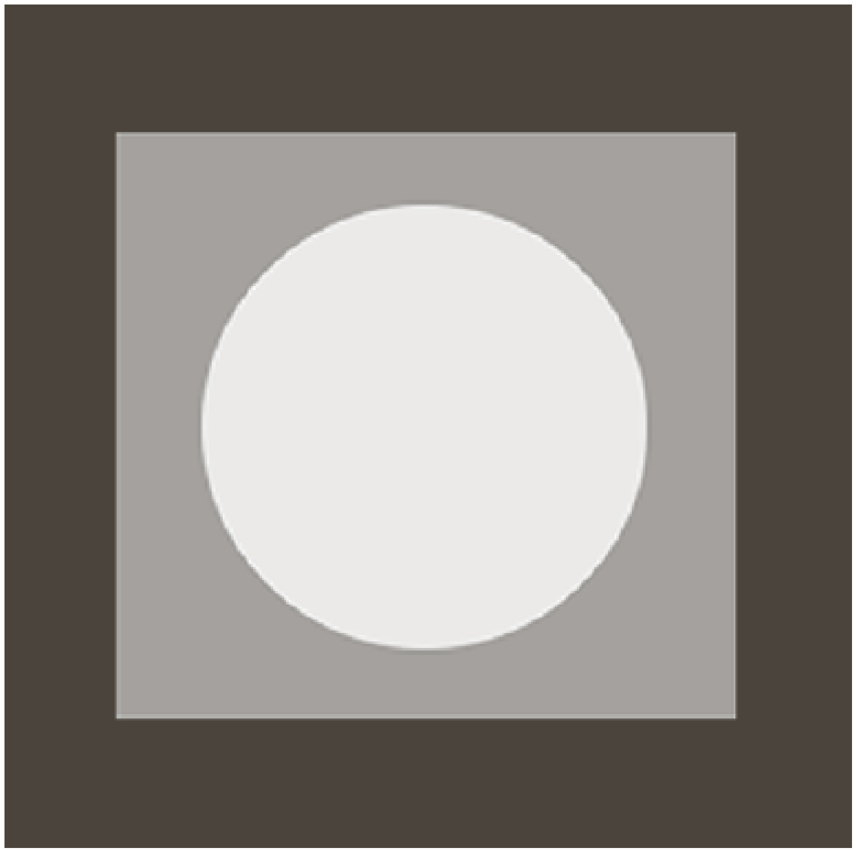


FIGURE 5.3 Test pattern used to illustrate the characteristics of the noise PDFs shown in Fig. 5.2.

Examples of Noise: Noisy Images

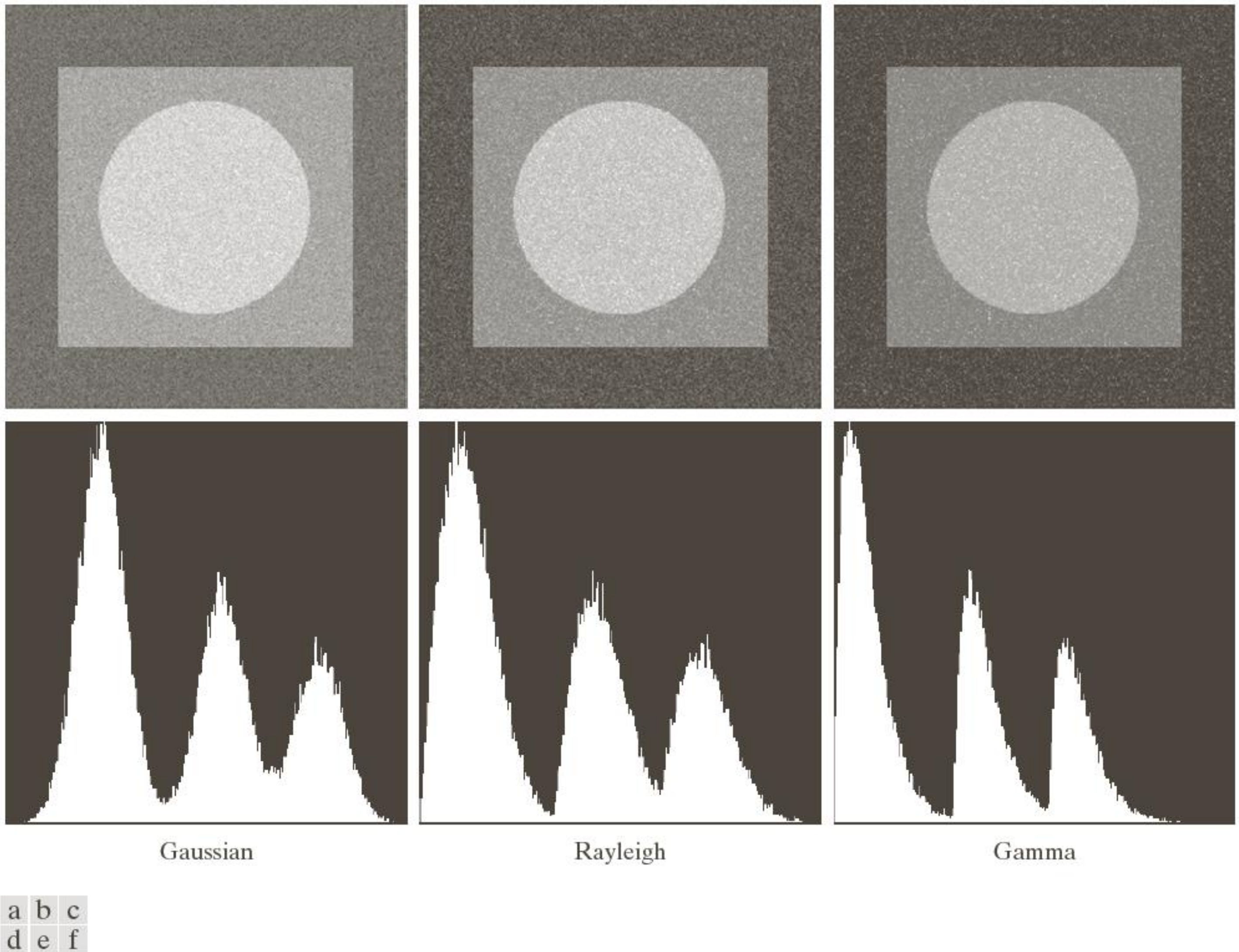


FIGURE 5.4 Images and histograms resulting from adding Gaussian, Rayleigh, and gamma noise to the image in Fig. 5.3.

Examples of Noise: Noisy Images

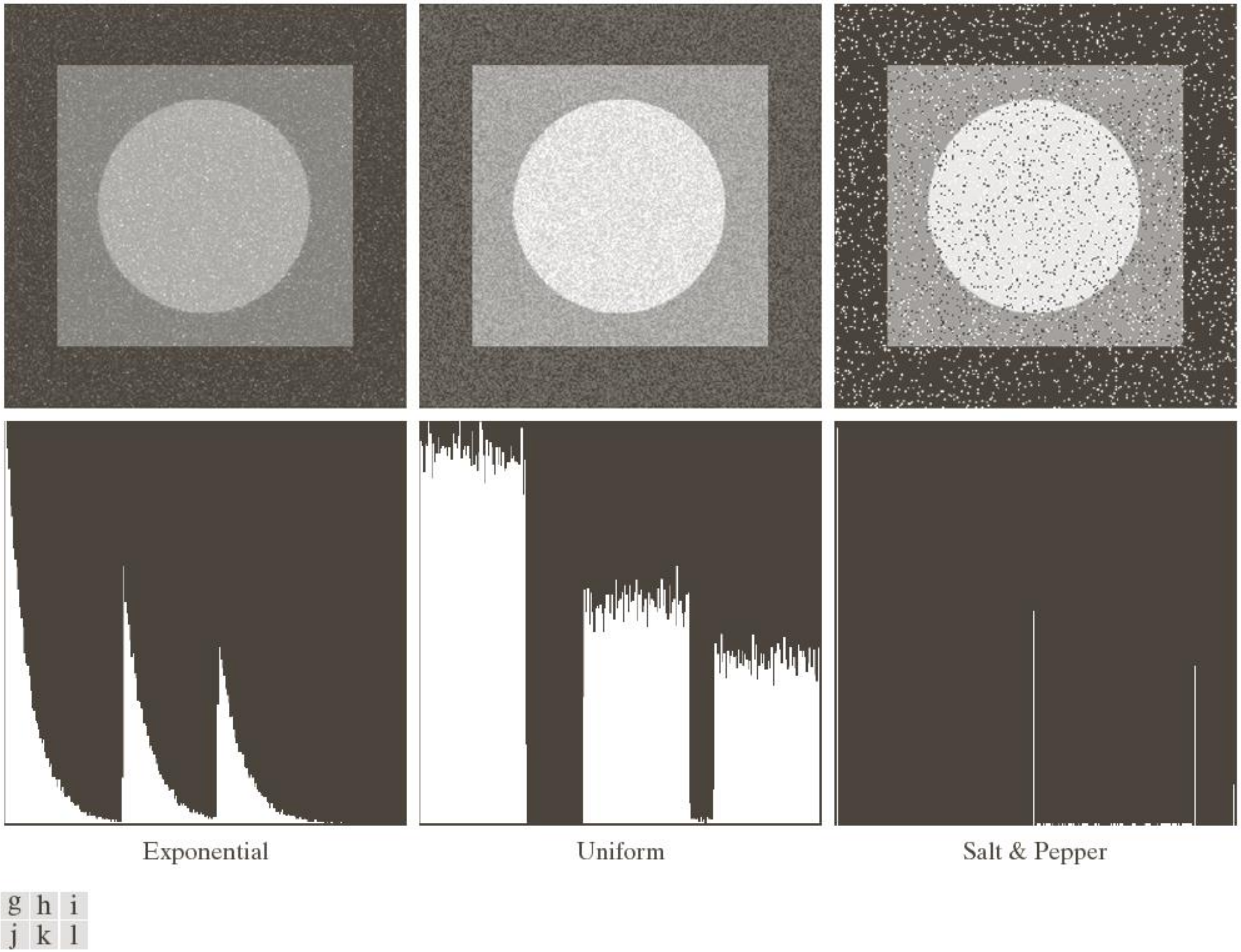
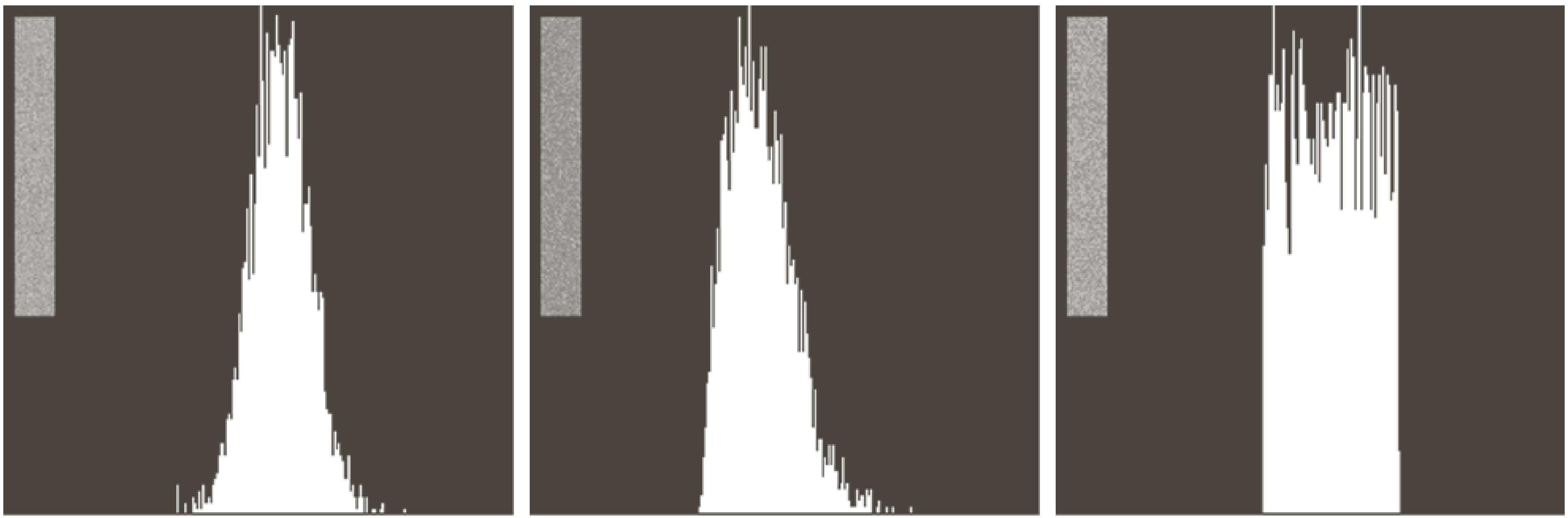


FIGURE 5.4 *(Continued)* Images and histograms resulting from adding exponential, uniform, and salt and pepper noise to the image in Fig. 5.3.

Estimation of Noise Parameters

The shape of the histogram identifies the closest PDF match



a b c

FIGURE 5.6 Histograms computed using small strips (shown as inserts) from (a) the Gaussian, (b) the Rayleigh, and (c) the uniform noisy images in Fig. 5.4.

Spatial Filtering: Mean Filters

Let S_{xy} represent the set of coordinates in a rectangle subimage window of size $m \times n$, centered at (x, y).

Arithmetic mean filter

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t)\in S_{xy}} g(s,t)$$

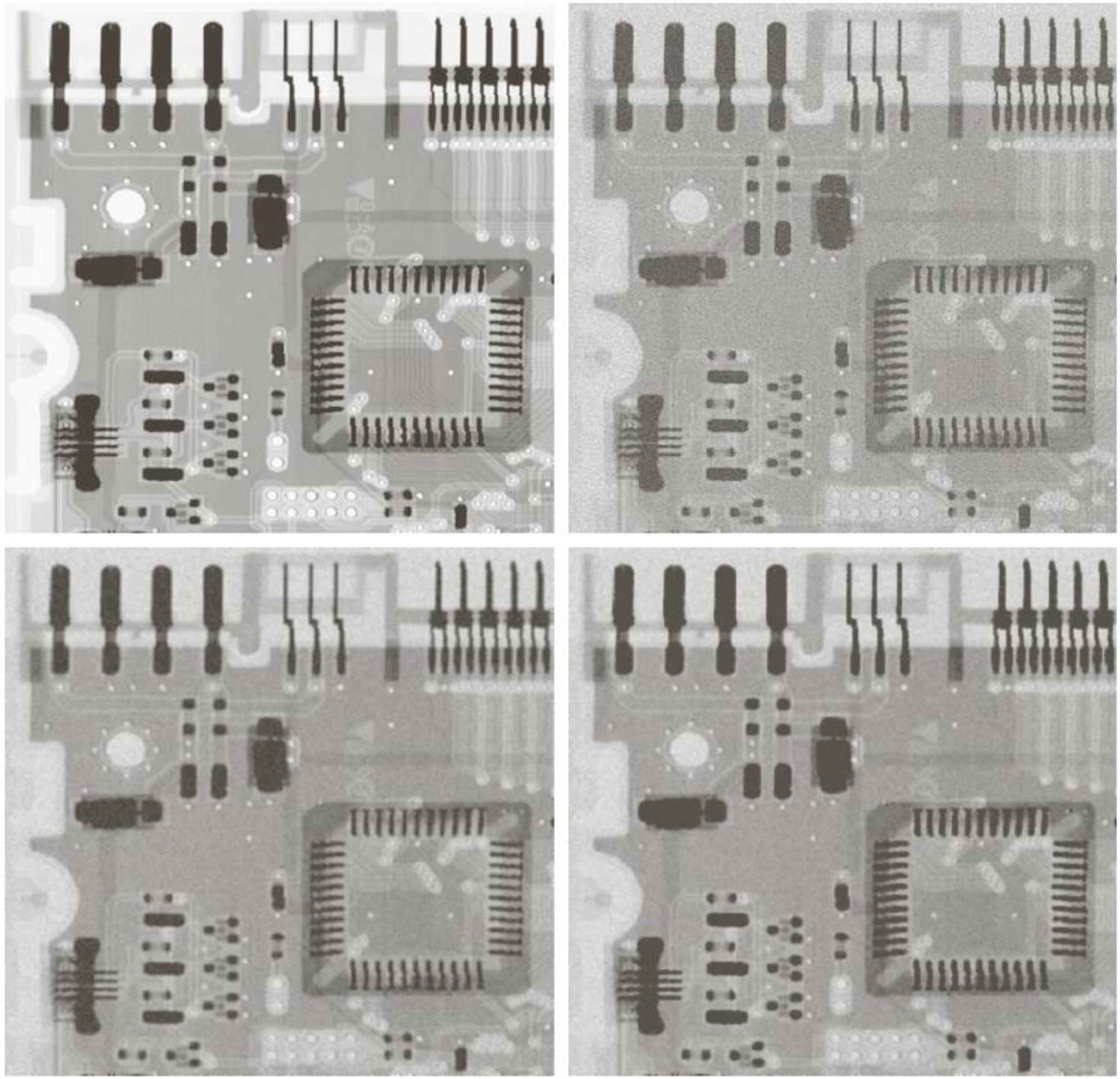
Spatial Filtering: Mean Filters

Geometric mean filter

$$\hat{f}(x,y) = \left[\prod_{(s,t)\in S_{xy}} g(s,t) \right]^{\frac{1}{mn}}$$

Generally, a geometric mean filter achieves smoothing comparable to the arithmetic mean filter, but it tends to lose less image detail in the process

Spatial Filtering: Example



a b
c d

FIGURE 5.7 (a) X-ray image. (b) Image corrupted by additive Gaussian noise. (c) Result of filtering with an arithmetic mean filter of size 3×3. (d) Result of filtering with a geometric mean filter of the same size. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Spatial Filtering: Order-Statistic Filters

Median filter

$$\hat{f}(x,y) = \underset{(s,t)\in S_{xy}}{median}\{g(s,t)\}$$

Max filter

$$\hat{f}(x,y) = \max_{(s,t)\in S_{xy}}\{g(s,t)\}$$

Min filter

$$\hat{f}(x,y) = \min_{(s,t)\in S_{xy}}\{g(s,t)\}$$

Spatial Filtering: Order-Statistic Filters

Midpoint filter

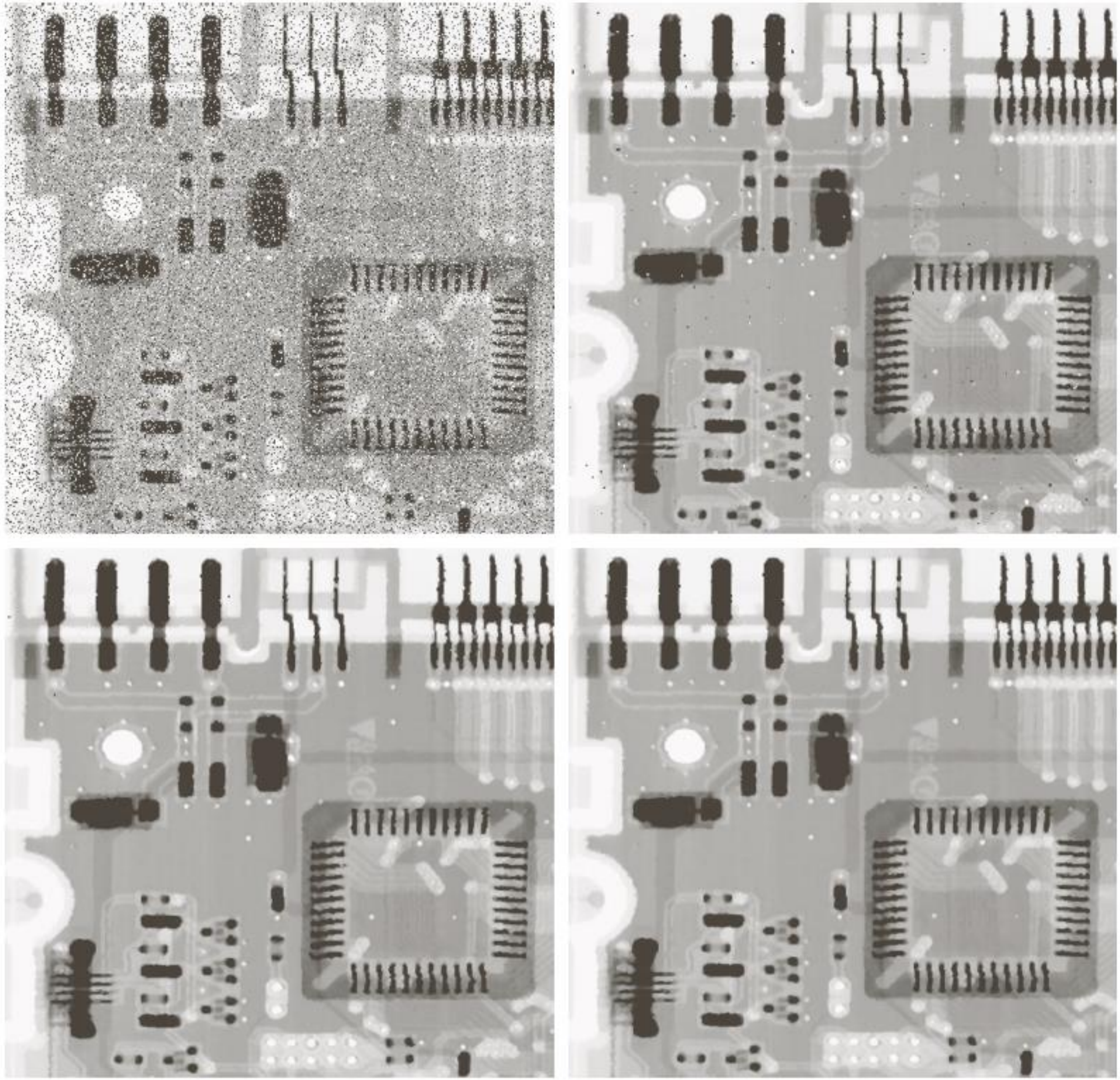
$$\hat{f}(x,y)=\frac{1}{2}\left[\max_{(s,t)\in S_{xy}}\{g(s,t)\}+\min_{(s,t)\in S_{xy}}\{g(s,t)\}\right]$$

Spatial Filtering: Order-Statistic Filters

Alpha-trimmed mean filter

$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} \left\{ g_r(s,t) \right\}$$

We delete the $d/2$ lowest and the $d/2$ highest intensity values of $g(s,t)$ in the neighborhood S_{xy}. Let $g_r(s,t)$ represent the remaining $mn-d$ pixels.



a b
c d

FIGURE 5.10 (a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.1$. (b) Result of one pass with a median filter of size 3×3. (c) Result of processing (b) with this filter. (d) Result of processing (c) with the same filter.

Spatial Filtering

a b

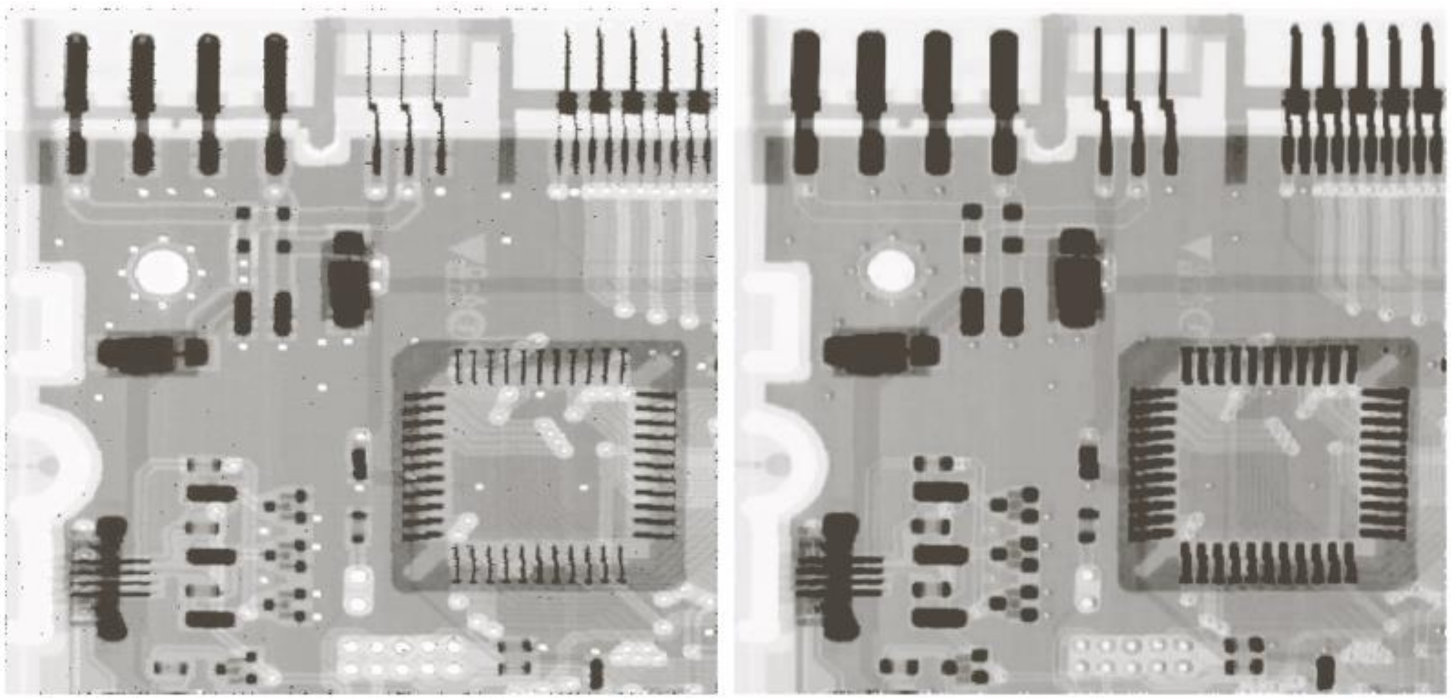
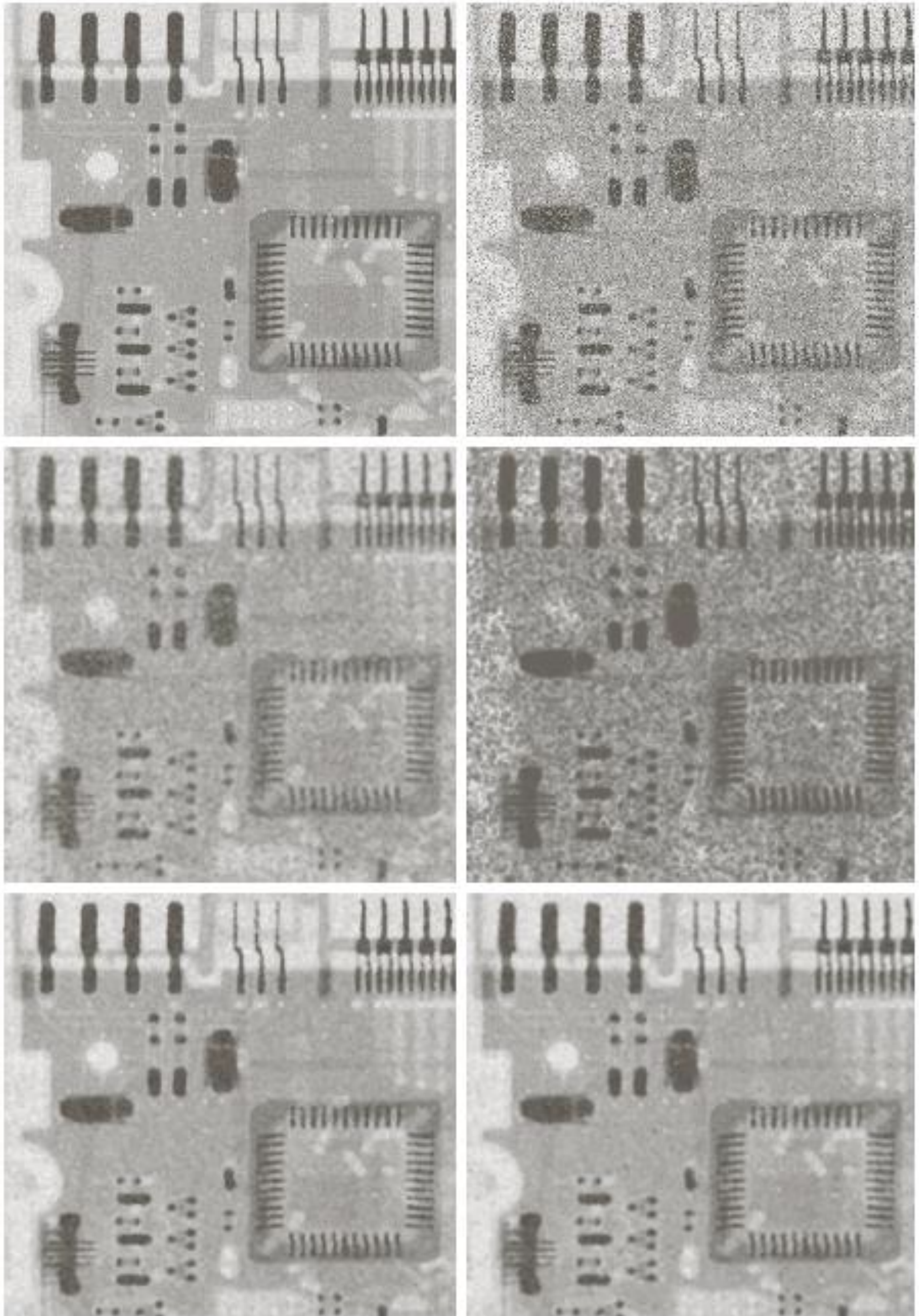


FIGURE 5.11 (a) Result of filtering Fig. 5.8(a) with a max filter of size 3 × 3. (b) Result of filtering 5.8(b) with a min filter of the same size.



a b
c d
e f

FIGURE 5.12
(a) Image corrupted by additive uniform noise. (b) Image additionally corrupted by additive salt-and-pepper noise. Image (b) filtered with a 5×5; (c) arithmetic mean filter; (d) geometric mean filter; (e) median filter; and (f) alpha-trimmed mean filter with $d = 5$.

References

- Sayısal Görüntü İşleme, Palme Publishing, Third Press Trans. (*Orj: R.C. Gonzalez and R.E. Woods: "Digital Image Processing", Prentice Hall, 3rd edition, 2008*).
- "Digital Image Processing Using Matlab", Gonzalez & Richard E. Woods, Steven L. Eddins, Gatesmark Publishing, 2009
- Lecture Notes, CS589-04 Digital Image Processing, Frank (Qingzhong) Liu, http://www.cs.nmt.edu/~ip
- Lecture Notes, BIL717-Image Processing, Erkut Erdem
- Lecture Notes, EBM537-Image Processing, F.Karabiber
- https://docs.opencv.org/
- https://senolomer0.medium.com/
- Bekir Aksoy, Python ile İmgeden Veriye Görüntü İşleme ve Uygulamaları, Nobel Akademik Yayıncılık