Image Restoration and Reconstruction

NGUYỄN HẢI TRIỀU[1]

[1]Bộ môn Kỹ thuật phần mềm,

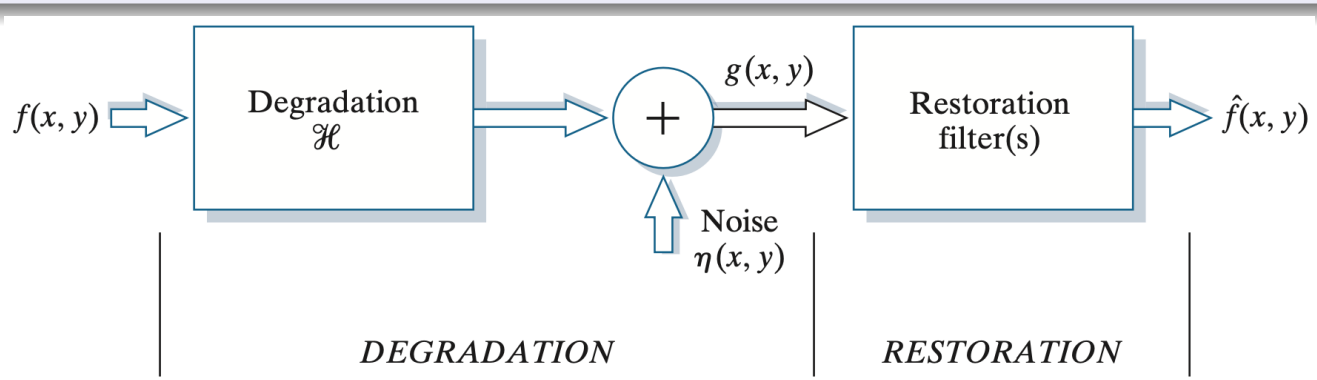
Khoa Công nghệ thông tin, Trường ĐH Nha Trang

NhaTrang, December 2023

Restoration techniques–Khôi phục ảnh

The principal goal of restoration techniques is to improve an image in some predefined sense. So sánh với cải thiện ảnh:

- Cải thiện ảnh là một quá trình chủ quan xuất phát từ mong muốn của người dùng.
- Trong khi đó, khôi phục ảnh lại đến từ quá trình khách quan, bắt buộc phải khôi phục lại một ảnh bị xuống cấp (image degradation).



Hình 1: we model image degradation as an operator $\mathcal{H}$ that, together with an additive noise term $\eta(x, y)$, operates on an input image $f(x, y)$ to produce a degraded image $g(x, y)$.

Nhiễu

Nhiễu xuất hiện trong quá trình thu nhận ảnh, số hóa và truyền. Nguyên nhân do

- Cảm biến ảnh có thể bị ảnh hưởng bởi các điều kiện môi trường.
- Nhiễu có thể can thiệp vào ảnh trong quá trình truyền ảnh.
- Lượng tử hóa, Số hóa.

Ảnh nhiễu được biểu diễn bằng biểu thức:

$$g(x, y) = f(x, y) + \eta(x, y),$$

trong đó, $f(x, y)$ là ảnh gốc, $\eta(x, y)$ là nhiễu, $g(x, y)$ là ảnh sau khi bị nhiều tác động.

Nếu xác định mô hình nhiễu, ta có thể **tách nhiễu để khôi phục ảnh**.

Nhiễu Gaussian

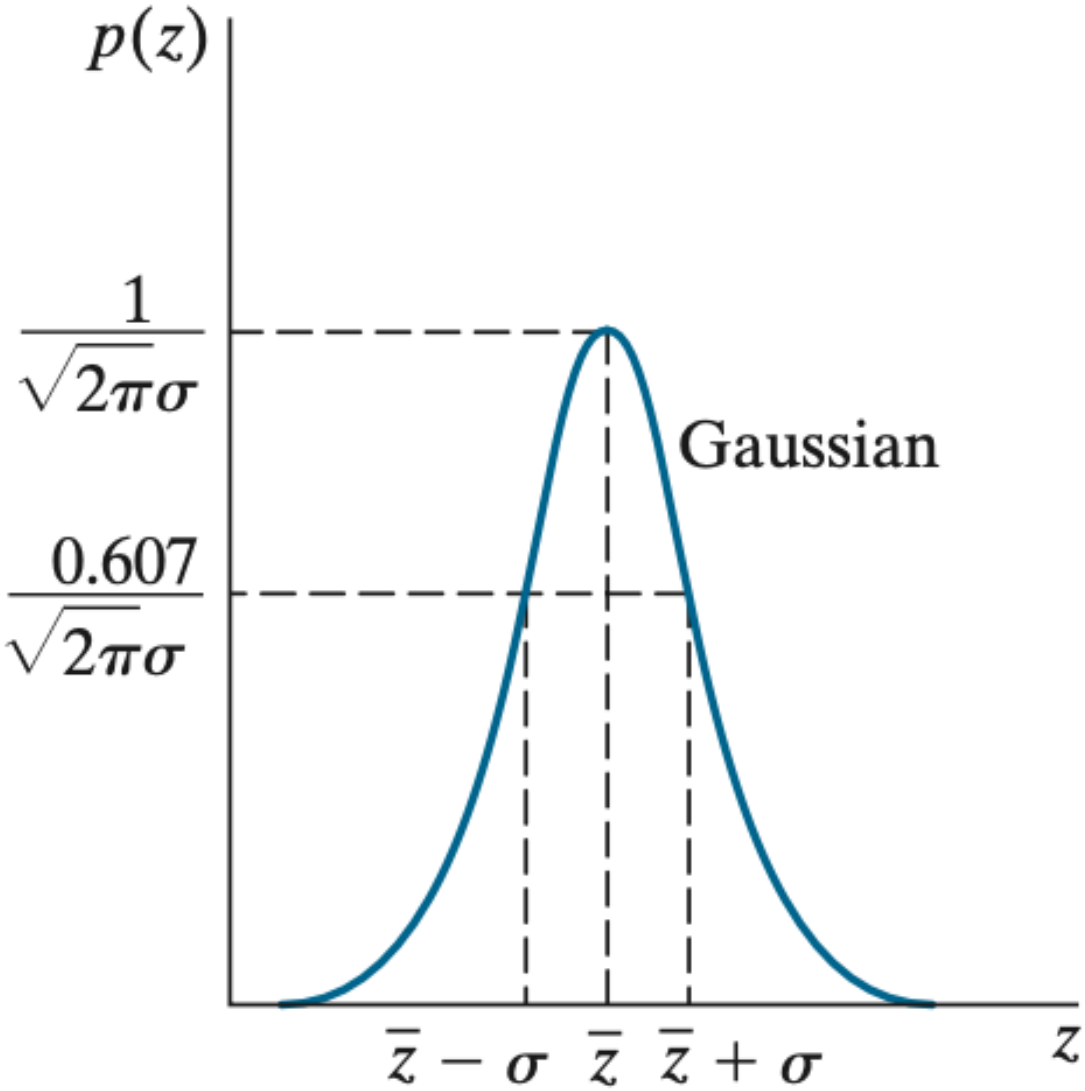
Nhiễu Gaussian có thể biểu diễn công thức toán học trong cả hai miền không gian và miền tần số.

- Nguyên nhân do: nhiễu mạch điện tử, thiết bị cảm biến, ánh sáng kém, nhiệt độ bộ cảm biến cao.
- Phân bố xác suất của nhiễu được tính bằng hàm Gaussian

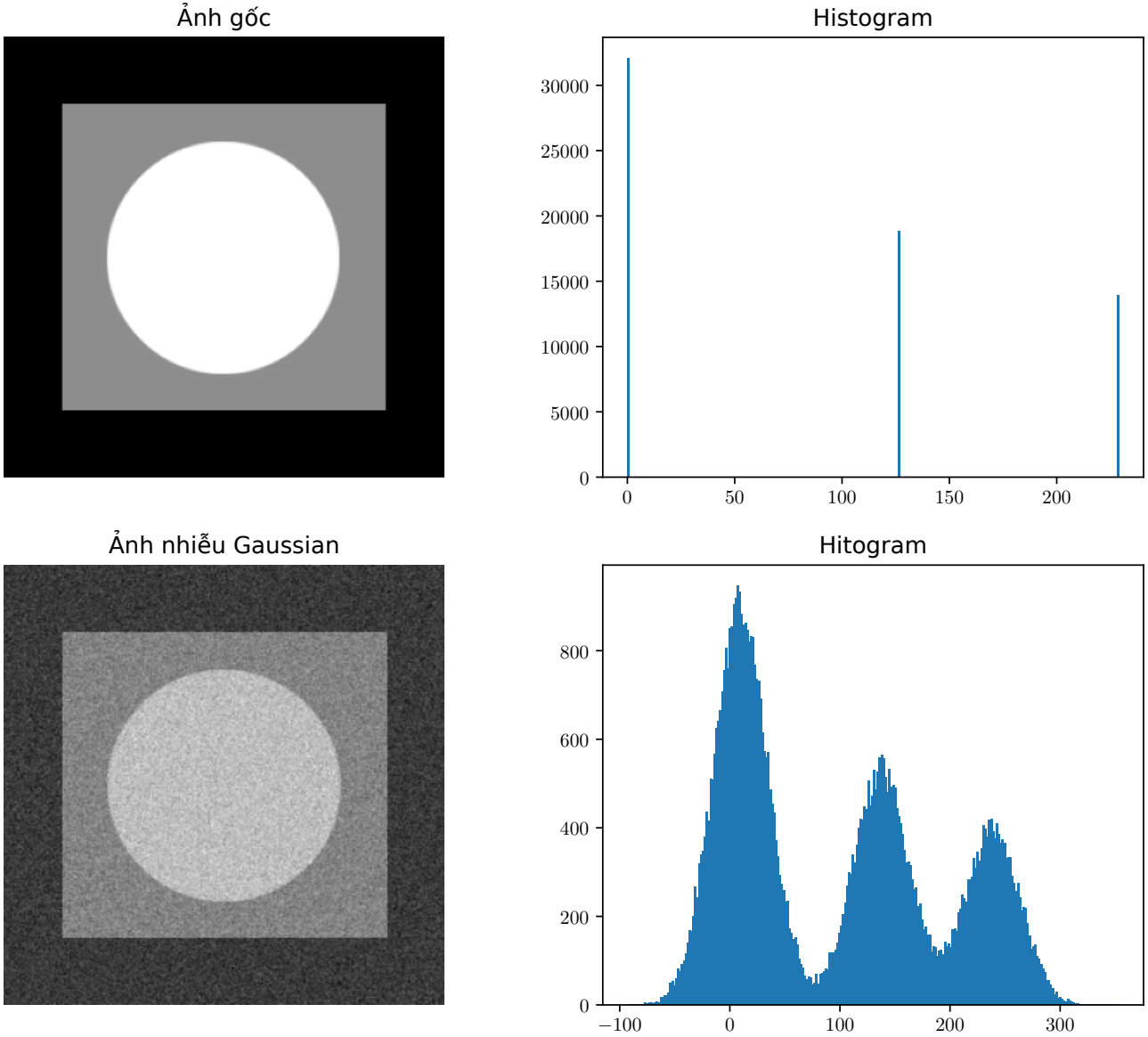
The probability density function (PDF) of a Gaussian random variable z

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-\bar{z})^2}{2\sigma^2}}, \quad -\infty < z < \infty.$$

Trong đó, z giá trị mức xám (cường độ sáng) điểm ảnh, $\bar{z}$ giá trị trung bình của z, σ là độ lệch chuẩn (standard deviation) của z, σ^2 là phương sai (variance) của z.



Hình 2: The PDF of a Gaussian.



Hình 3: Ví dụ mô tả ảnh bị nhiễu Gaussian.

Nhiễu xung (Impulse noise)

Impulse noise: đặc trưng bởi một điểm ảnh có giá trị mức xám khác biệt lớn so với những điểm lân cận. Chia làm 2 loại:

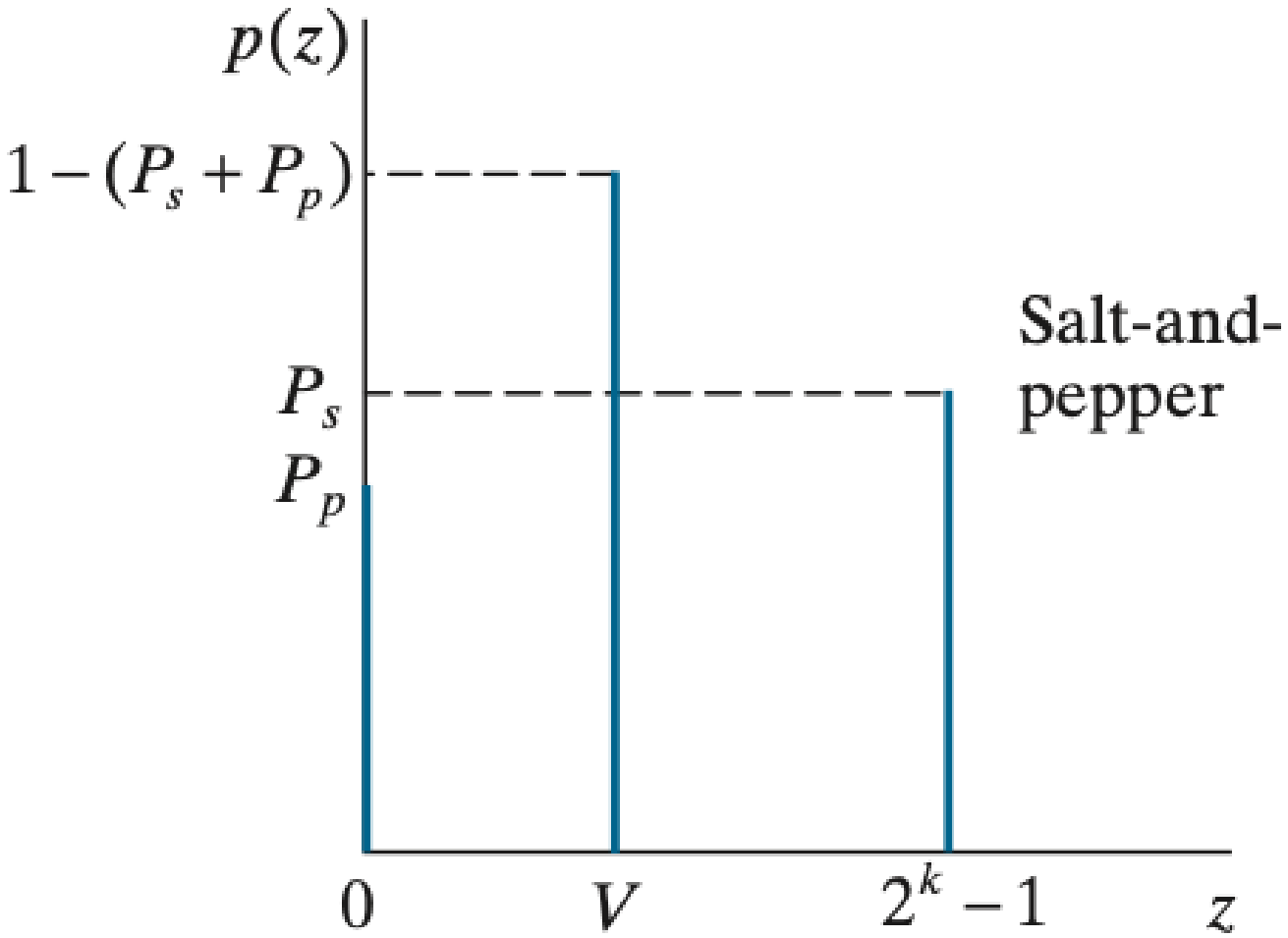
- Xung nhiễu âm → điểm ảnh đen (pepper) → nhiễu hạt tiêu.
- Xung nhiễu dương → điểm ảnh trắng (salt) → nhiễu muối tiêu.

Khi ảnh bị bão hòa bởi nhiễu xung thì xuất hiện các điểm đen-trắng được gọi là **nhiễu muối tiêu** (salt-and-pepper noise).

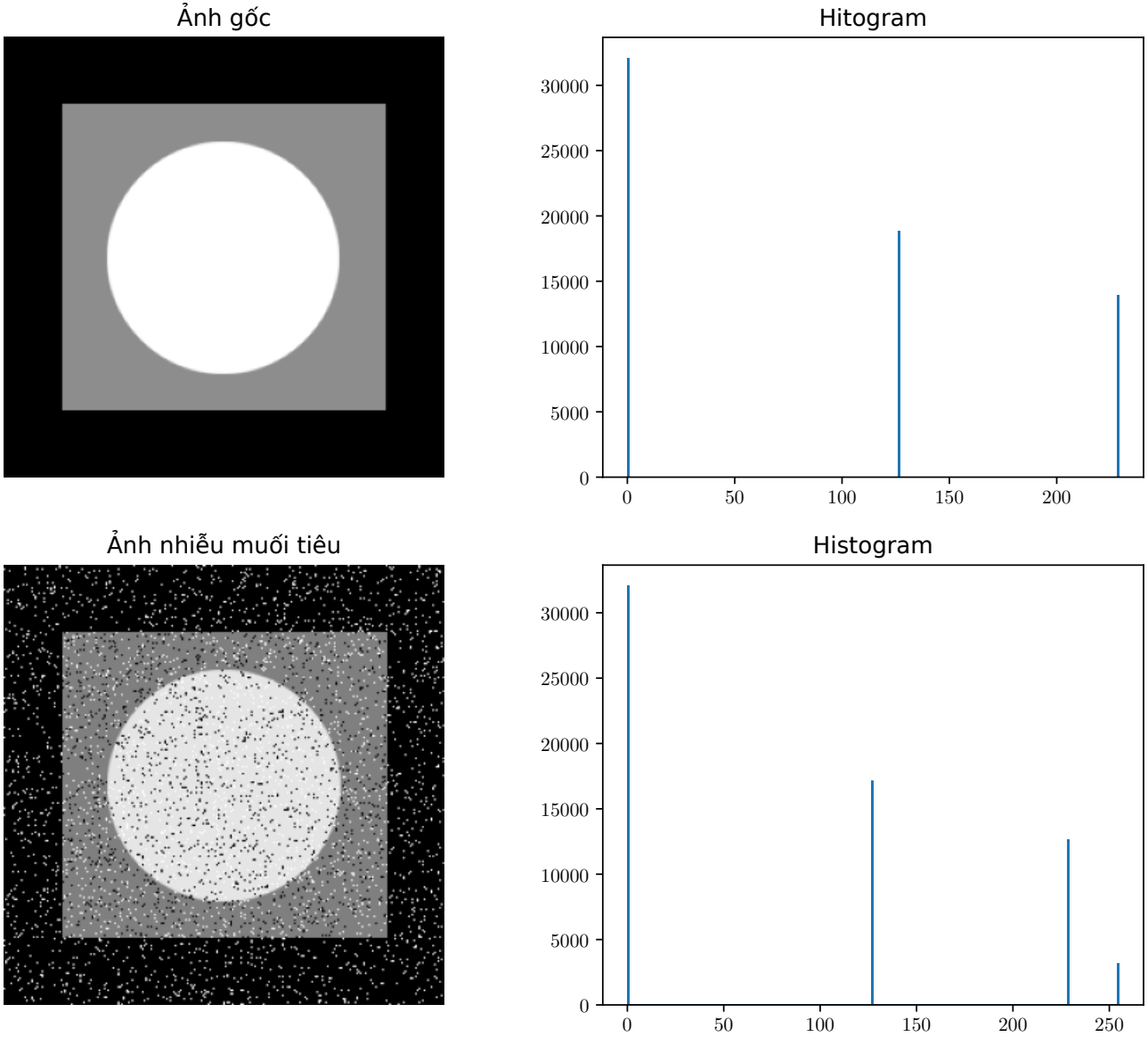
The PDF of salt-and-pepper noise is given by

$$p(z)\begin{cases} P_s, & \text{for } z = 2^k - 1, \\ P_p, & \text{for } z = 0, \\ 1 - (P_s + P_p), & \text{for } z = V. \end{cases}$$

where, k represents the number of bits used to represent the intensity values in a digital image. V is any integer value in the range $0 < V < 2^k - 1$.



Hình 4: The PDF of salt-and-pepper noise



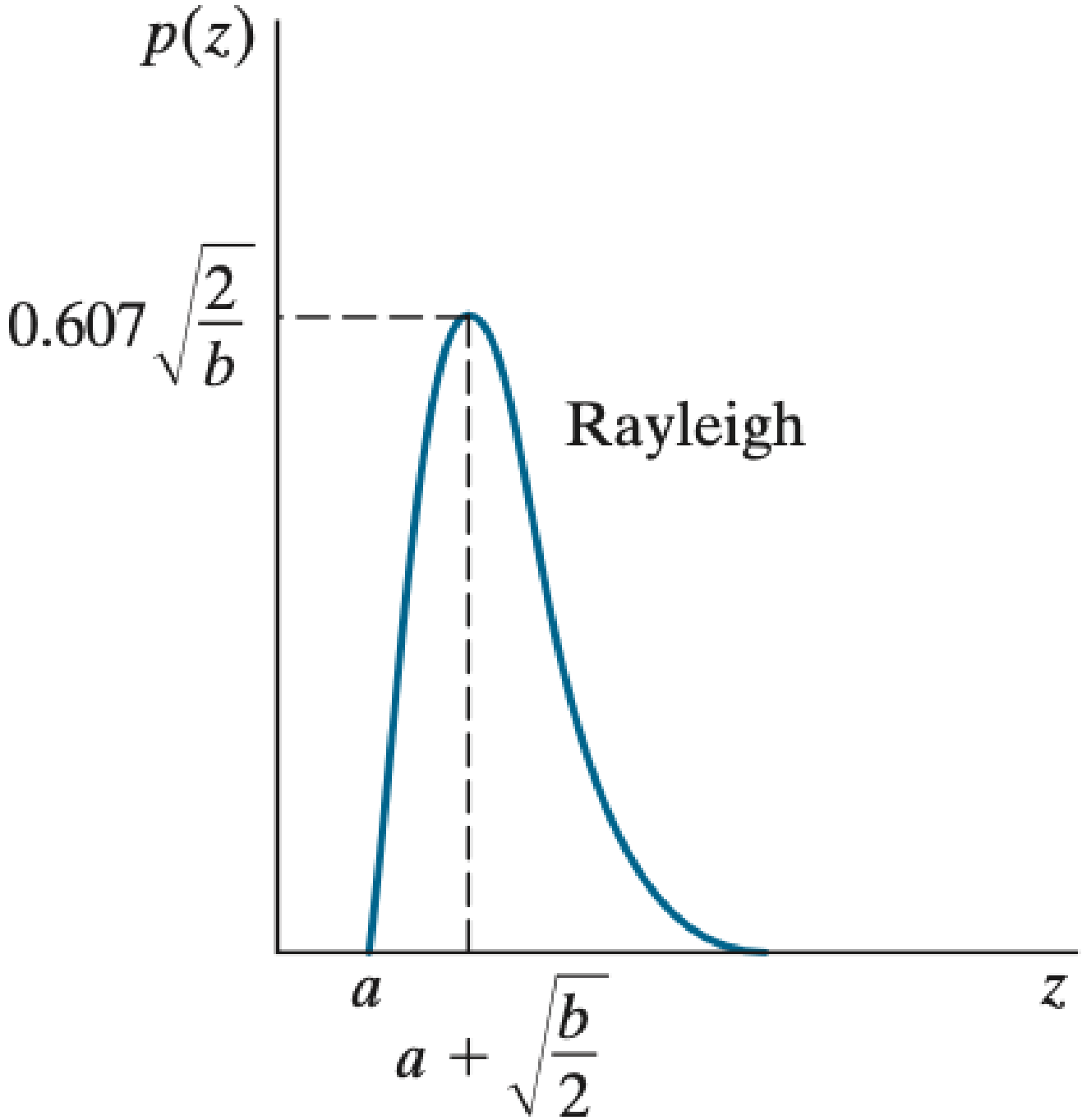
Hình 5: Ví dụ mô tả ảnh bị nhiễu muối tiêu.

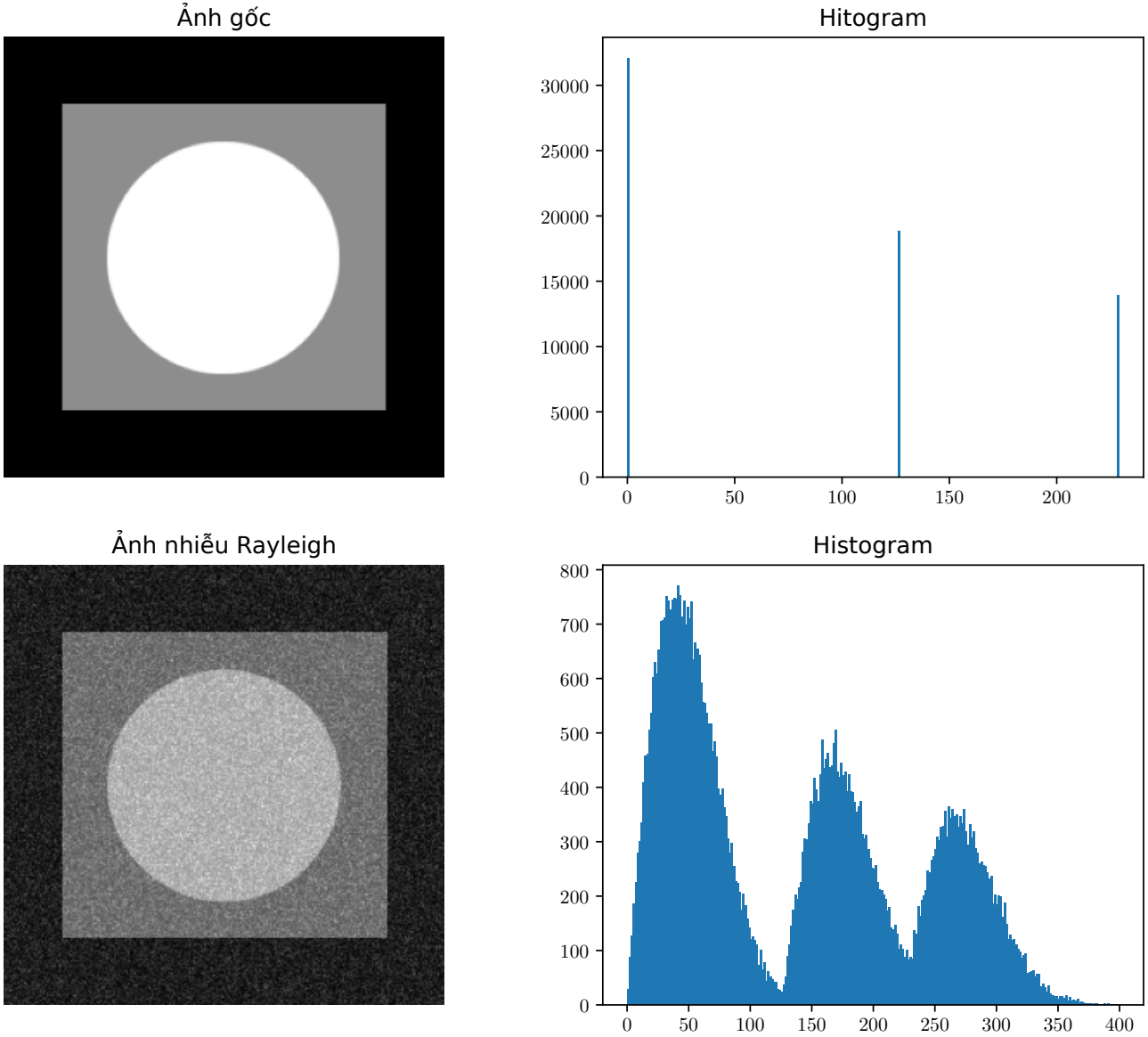
Nhiễu Rayleigh–Rayleigh Noise

The PDF of Rayleigh noise is given by

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

. The mean and variance of z when this random variable is characterized by a Rayleigh PDF are $\bar{z} = a + \sqrt{\pi b/4}$, $\sigma^2 = \frac{b(4-\pi)}{4}$. Trong đó, a, b là hằng số nguyên dương và $a > b$.

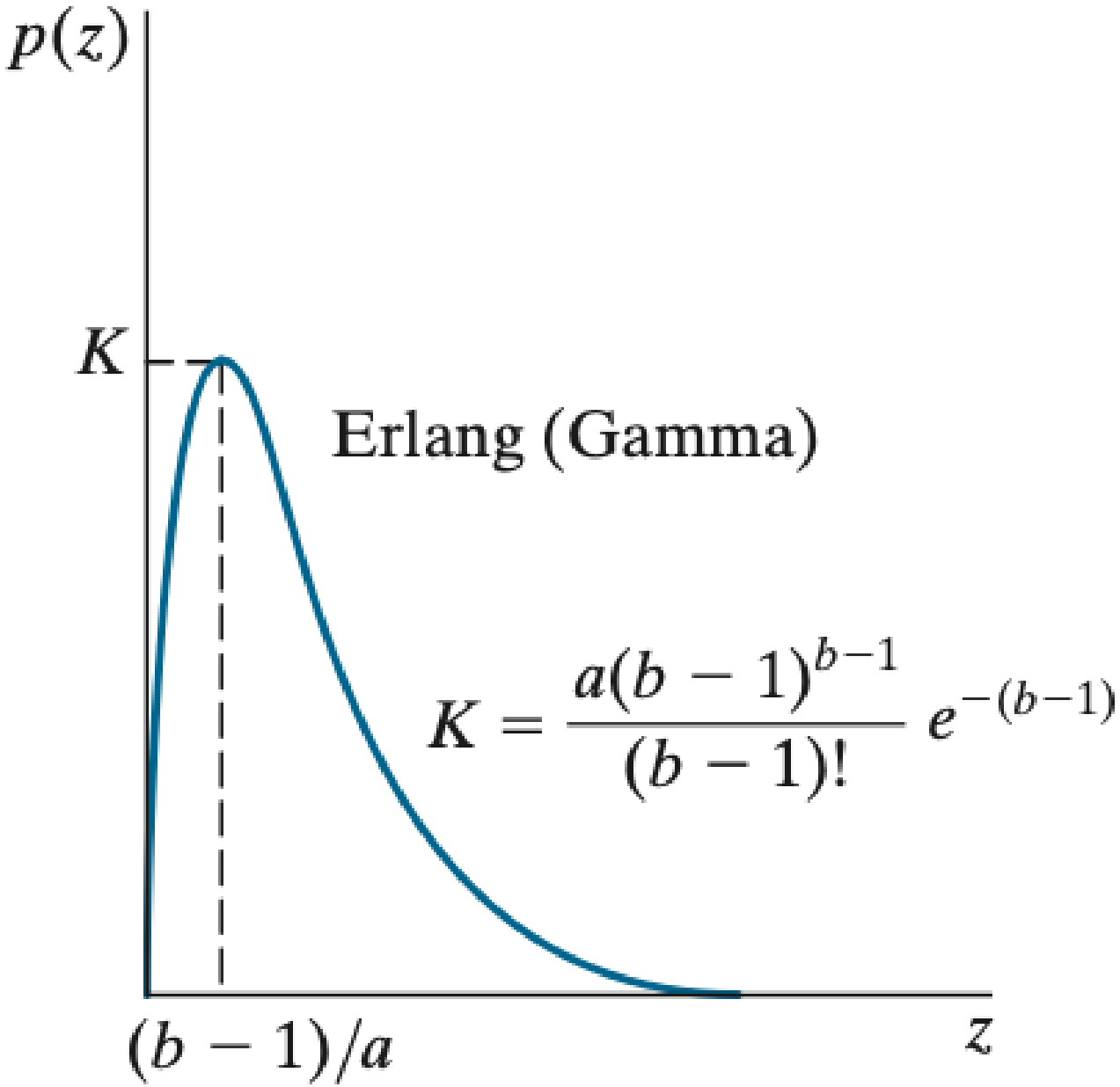

$p(z)$
$0.607\sqrt{\frac{2}{b}}$
Rayleigh
a
$a+\sqrt{\frac{b}{2}}$
z



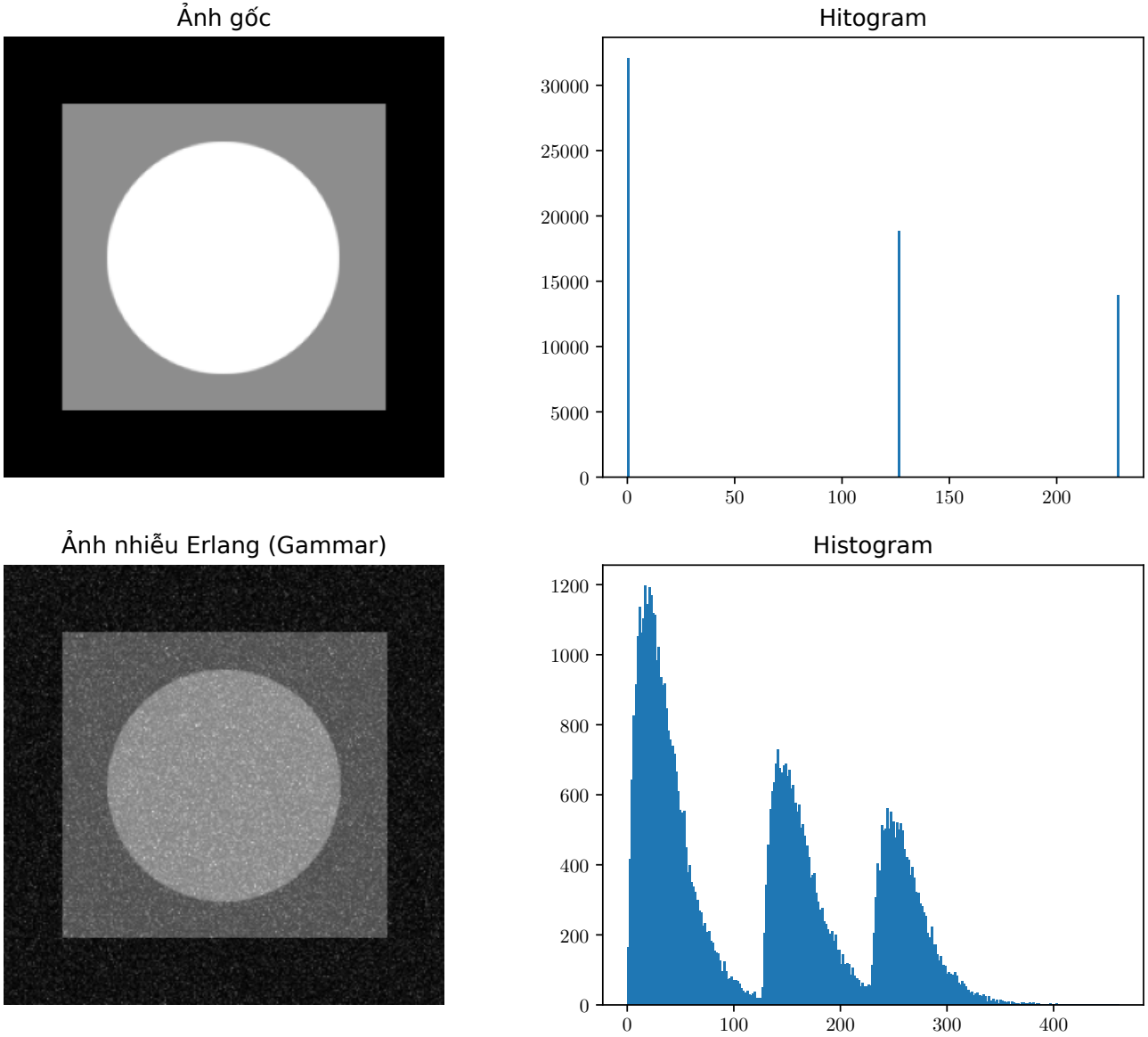
Hình 7: Ví dụ mô tả ảnh bị nhiễu Rayleigh.

Nhiễu Rayleigh–Rayleigh Noise

The PDF of Erlang noise is $p(z) = \begin{cases} \frac{a^b z^{b-1}}{(b-1)!} e^{-az}, & z \geq 0, \\ 0, & z < 0. \end{cases}$. Trong đó: giá trị trung bình và phương sai của z $\bar{z} = \frac{b}{a}$, $\sigma^2 = \frac{b}{a^2}$. Trong đó, a, b là hằng số nguyên dương và $a > b$. Ký hiệu "!" là phép toán tính giai thừa.



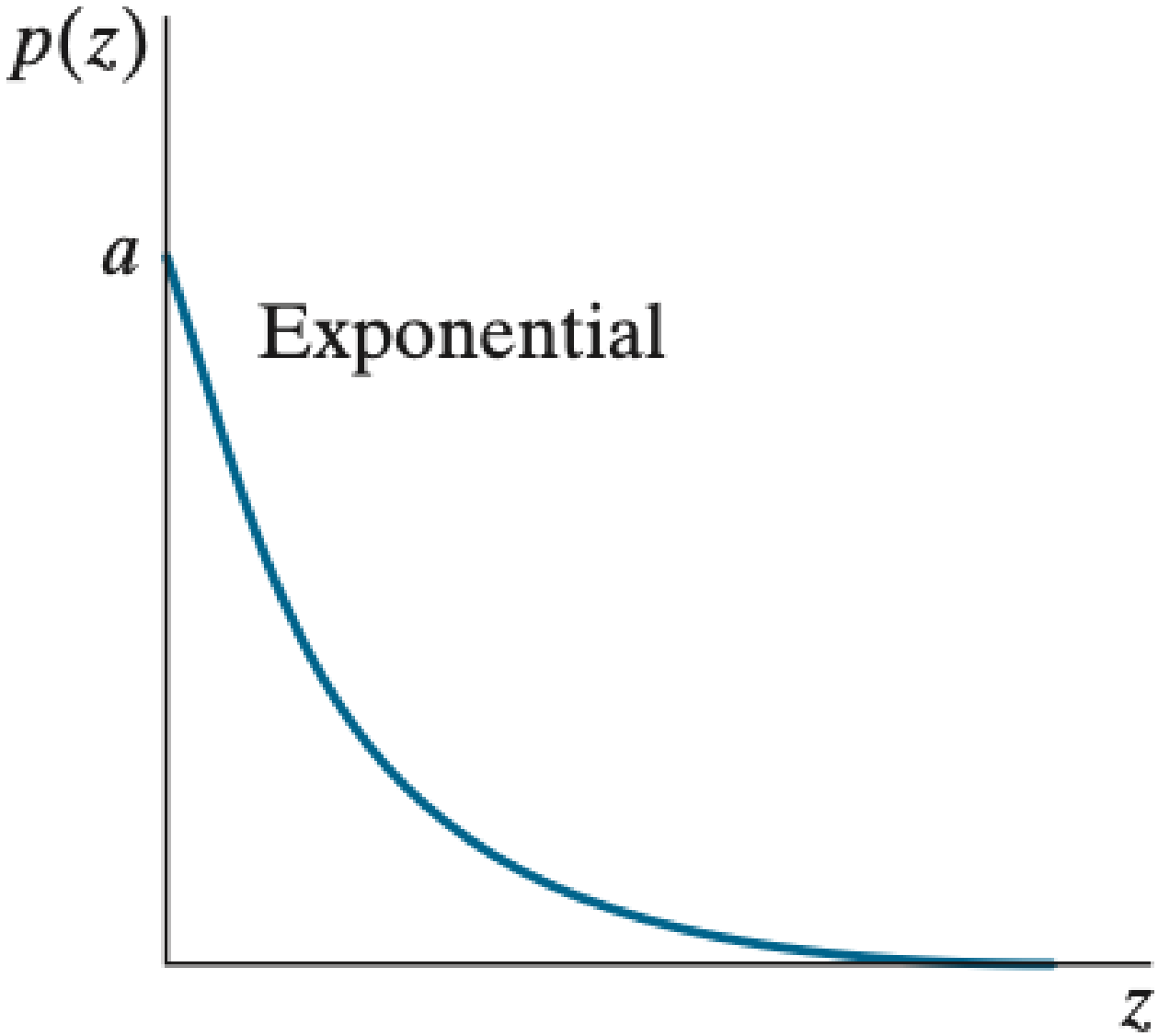
Hình 8: The PDF of Erlang Noise



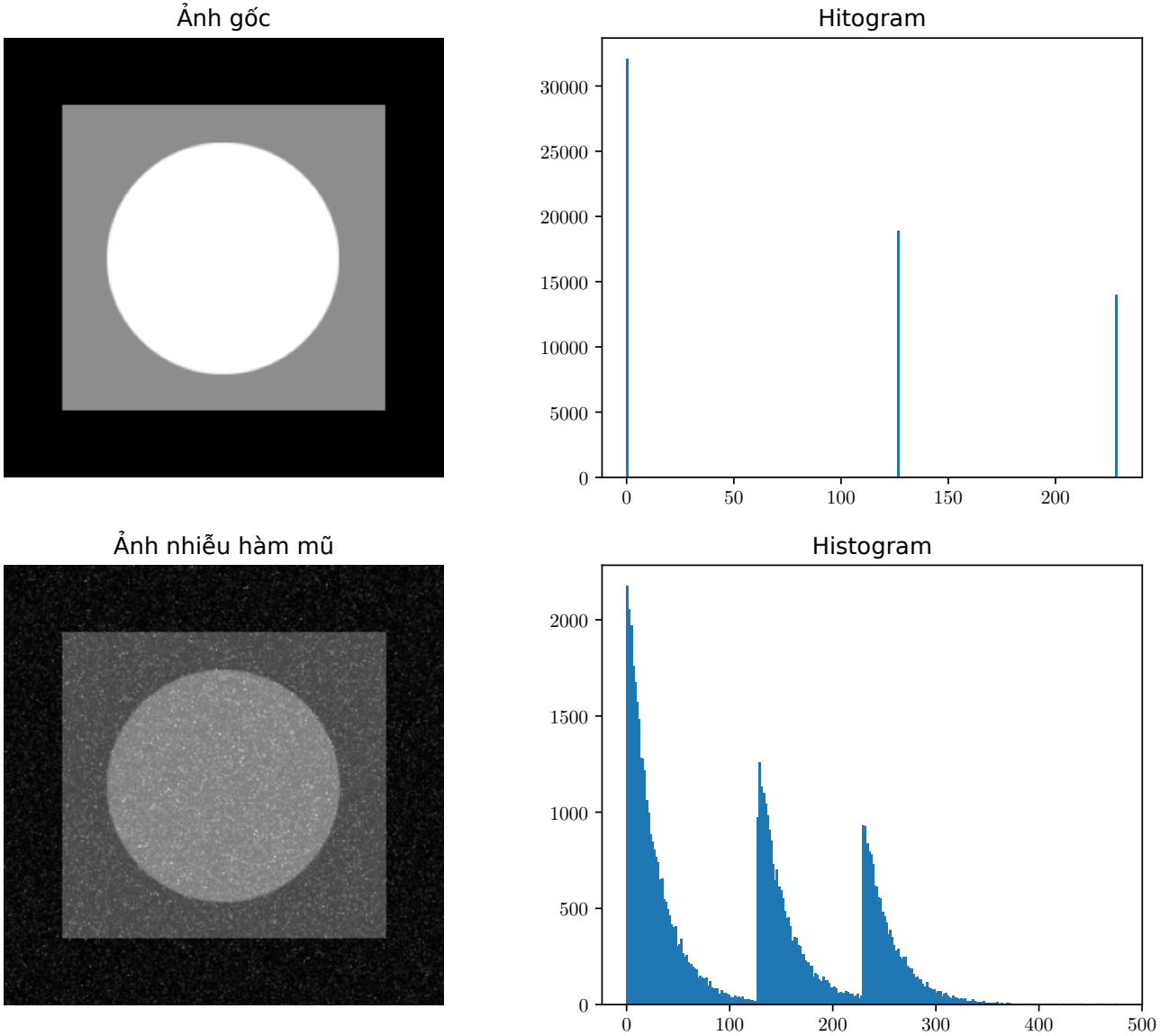
Hình 9: Ví dụ mô tả ảnh bị nhiễu Erlang.

Nhiễu hàm mũ–Exponential Noise

The PDF of exponential noise is $p(z) = \begin{cases} ae^{-az}, & z \geq 0, \\ 0, & z < 0. \end{cases}$. Trong đó: $a > 0$, giá trị trung bình và phương sai của z: $\bar{z} = \frac{1}{a}$, $\sigma^2 = \frac{1}{a^2}$. Đây là trường hợp đặt biệt của nhiều Erlang khi $b = 1$.



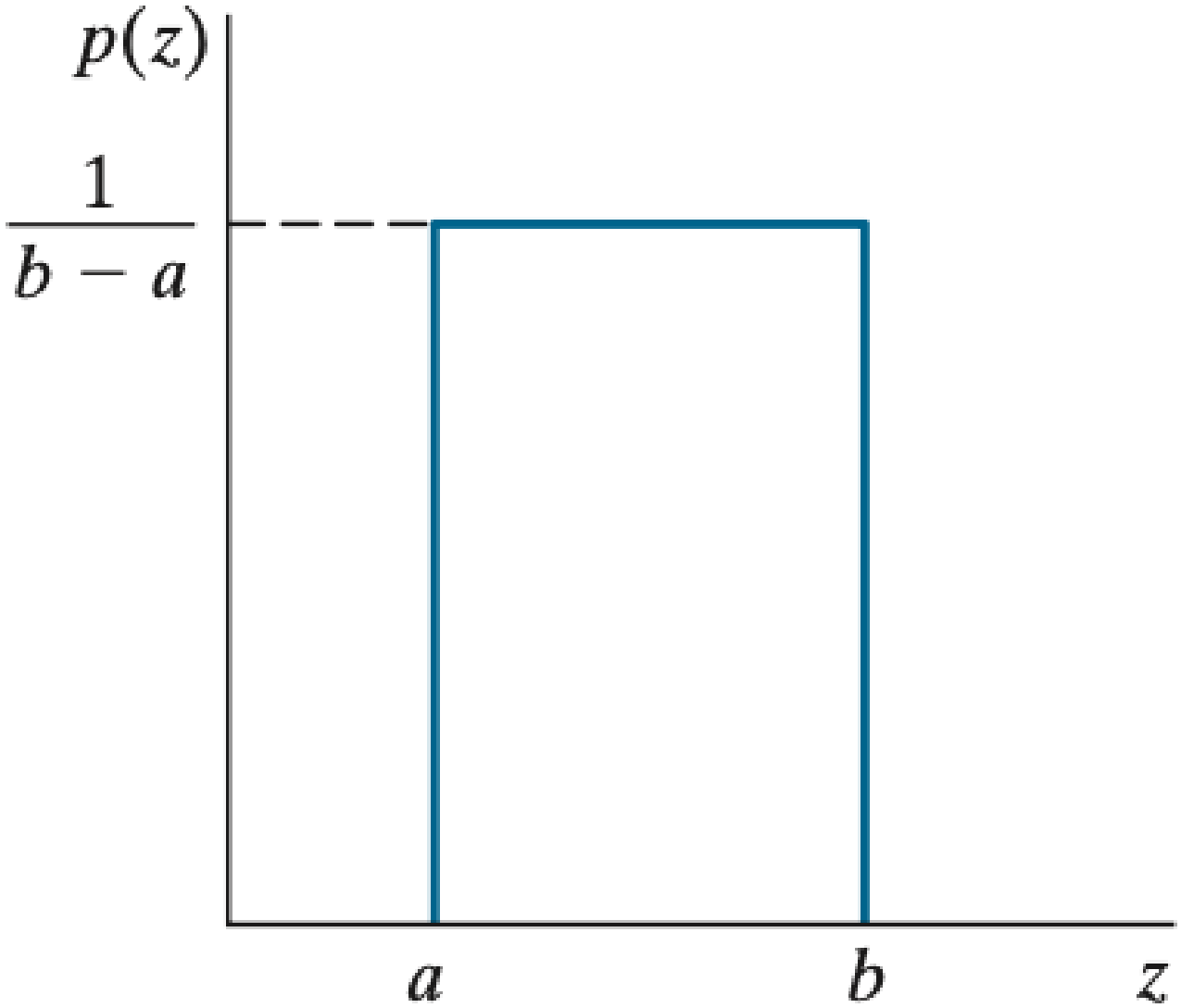
Hình 10: The PDF of Exponential Noise



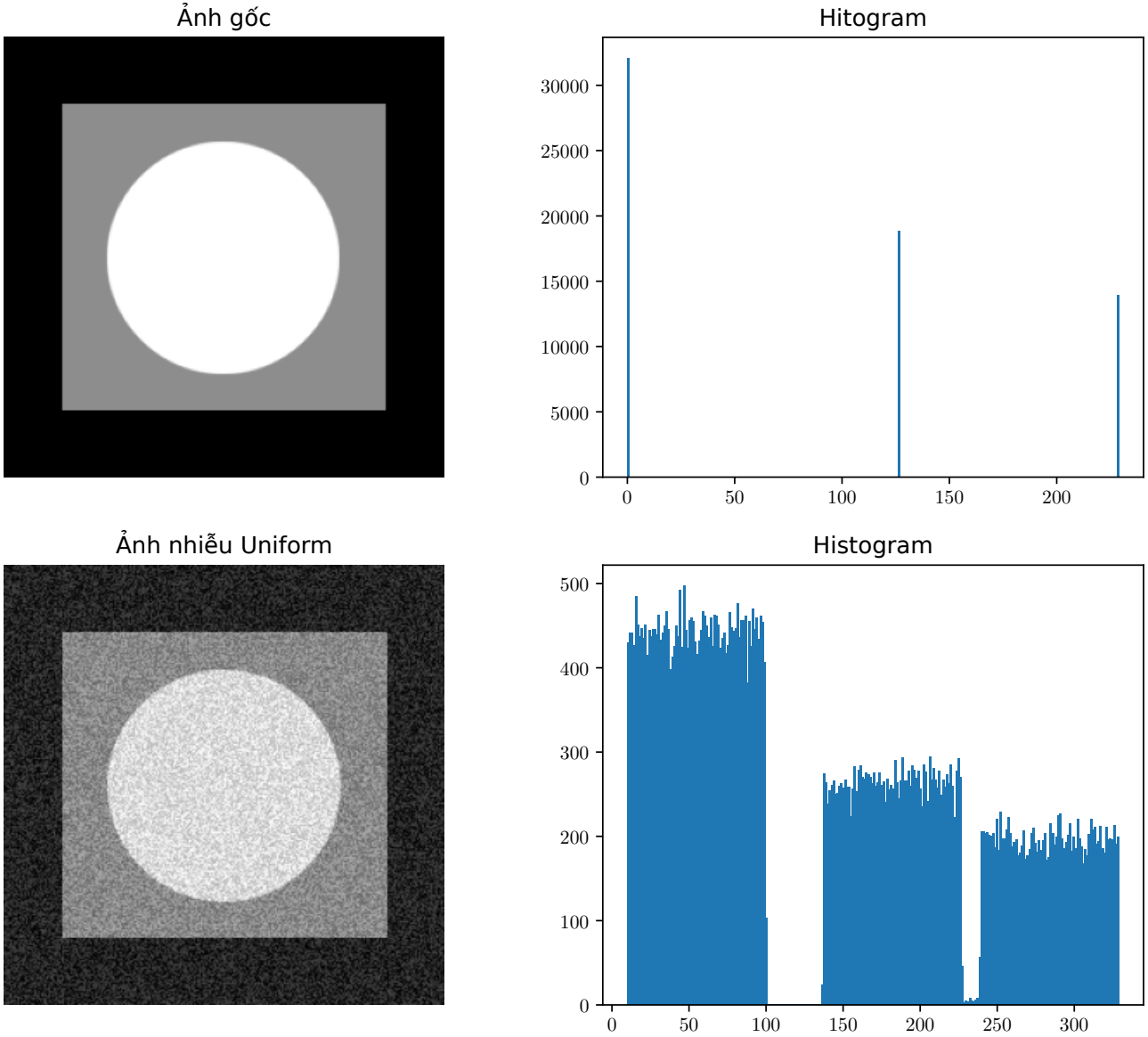
Hình 11: Ví dụ mô tả ảnh bị nhiễu hàm mũ.

Nhiễu đồng nhất–Uniform Noise

The PDF of Uniform Noise is $p(z) = \begin{cases} \frac{1}{b-a}, & a \leq z \leq b, \\ 0, & \text{otherwise.} \end{cases}$. Trong đó: $a, b > 0$, giá trị trung bình và phương sai của z: $\bar{z} = \frac{a+b}{2}$, $\sigma^2 = \frac{(b-a)^2}{12}$.



Hình 12: The PDF of Uniform Noise



Hình 13: Ví dụ mô tả ảnh bị nhiễu hàm đồng nhất.

Code minh hoạ một số hàm tạo nhiễu I

```
1  import numpy as np
2  import cv2
3  import matplotlib.pyplot as plt
4
5  # Đọc ảnh
6  img = cv2.imread('test_pattern.tif',0)
7  m,n = img.shape[:2]
8
9  # Thêm nhiễu Gaussian vào ảnh img
10 gia_tri_TB = 10
11 phuong_sai = 25
12 noise = np.random.normal(loc=gia_tri_TB,scale=phuong_sai,size=(m,n))
13 Gau_noisy_img = img + noise
14
15 # Thêm nhiễu Rayleigh vào ảnh img
16 phuong_sai = 40
17 noise = np.random.rayleigh(scale=phuong_sai,size=(m,n))
18 Ray_noisy_img = img + noise
19
20 # Thêm nhiễu Erlang (Gammar) vào ảnh img
```

Code minh hoạ một số hàm tạo nhiễu II

```
21 K = 2.0
22 phuong_sai = 18
23 noise = np.random.gamma(shape=K,scale=phuong_sai,size=(m,n))
24 Gam_noisy_img = img + noise
25
26 # Thêm nhiễu hàm mũ vào ảnh img
27 phuong_sai = 26
28 noise = np.random.exponential(scale=phuong_sai,size=(m,n))
29 Exp_noisy_img = img + noise
30
31 # Thêm nhiễu Uniform vào ảnh img
32 a,b = 10,100
33 noise = np.random.uniform(low=a,high=b,size=(m,n))
34 Uni_noisy_img = img + noise
35
36 # Thêm nhiễu muối tiêu (add salt and pepper) vào ảnh img
37 number_black = int(m*n*0.05) # định nghĩa số điểm đen
38 number_white = int(m*n*0.05) # định nghĩa số điểm trắng
39 # Lấy giá trị nguyên ngẫu nhiên trong đoạn 0..m
40 # Giá trị này sẽ biểu diễn tọa độđiểm đen theo hàng
```

Code minh hoạ một số hàm tạo nhiễu III

```
41 m_blacks = np.random.randint(0,m,number_black)
42 # Lấy giá trị nguyên ngẫu nhiên trong đoạn 0..n
43 # Giá trị này sẽ biểu diễn tọa độđiểm đen theo cột
44 n_blacks = np.random.randint(0,n,number_black)
45 # Lấy giá trị nguyên ngẫu nhiên trong đoạn 0..m
46 # Giá trị này sẽ biểu diễn tọa độđiểm trắng theo hàng
47 m_whites = np.random.randint(0,m,number_white)
48 # Lấy giá trị nguyên ngẫu nhiên trong đoạn 0..n
49 # Giá trị này sẽ biểu diễn tọa độđiểm trắng theo cột
50 n_whites = np.random.randint(0,n,number_white)
51
52 SP_noisy_img = np.copy(img) # Sao chép ảnh img đểtạo ảnh SP_noisy_img
53 # Thiết lập mức xám = 0 (điểm đen) cho điểm ảnh có tọa độ(m_blacks,n_blacks)
54 SP_noisy_img[m_blacks,n_blacks] = 0
55 # Thiết lập mức xám = 255 (điểm trắng) cho điểm ảnh có tọa độ
       (m_whites,n_whites)
56 SP_noisy_img[m_whites,n_whites] = 255
57
58 # 1. Hiển thị ảnh gốc, ảnh nhiễu và histogram
59 # Tạo cửa số 1 đểhiển thị ảnh cho nhiễu Gaussian
```

Code minh hoạ một số hàm tạo nhiễu IV

```
60  fig1 = plt.figure(figsize=(11, 9)) # Tạo vùng vẽ tỷ lệ 16:9
61  #Tạo 9 vùng vẽ con
62  (ax1, ax2), (ax3, ax4) = fig1.subplots(2, 2)
63  # Hiển thị ảnh gốc
64  ax1.imshow(img, cmap='gray')
65  ax1.set_title('Ảnh gốc')
66  ax1.axis('off')
67  # Hiển thị histogram ảnh gốc
68  ax2.hist(img.flatten(),bins=256)
69  ax2.set_title('Histogram')
70  # Hiển thị ảnh nhiễu Gaussian
71  ax3.imshow(Gau_noisy_img, cmap='gray')
72  ax3.set_title('Ảnh nhiễu Gaussian')
73  ax3.axis('off')
74  # Hiển thị histogram ảnh nhiễu gaussian
75  ax4.hist(Gau_noisy_img.flatten(),bins=256)
76  ax4.set_title('Hitogram')
77  plt.show()
78
79  # 2. Tạo cửa số 2 đểhiển thị ảnh cho nhiễu Rayleigh
```

Code minh hoạ một số hàm tạo nhiễu V

```
80  fig2 = plt.figure(figsize=(11, 9)) # Tạo vùng vẽ tỷ lệ 16:9
81  #Tạo 9 vùng vẽ con
82  (ax1, ax2), (ax3, ax4) = fig2.subplots(2, 2)
83  # Hiển thị ảnh gốc
84  ax1.imshow(img, cmap='gray')
85  ax1.set_title('Ảnh gốc')
86  ax1.axis('off')
87  # Hiển thị histogram ảnh gốc
88  ax2.hist(img.flatten(),bins=256)
89  ax2.set_title('Hitogram')
90  # Hiển thị ảnh nhiễu Rayleigh
91  ax3.imshow(Ray_noisy_img, cmap='gray')
92  ax3.set_title('Ảnh nhiễu Rayleigh')
93  ax3.axis('off')
94  # Hiển thị histogram ảnh nhiễu Rayleigh
95  ax4.hist(Ray_noisy_img.flatten(),bins=256)
96  ax4.set_title('Histogram')
97  plt.show()
98
99  # 3. Tạo cửa số 3 đểhiển thị ảnh cho nhiễu Erlang (Gamma)
```

Code minh hoạ một số hàm tạo nhiễu VI

```
00 fig2 = plt.figure(figsize=(11, 9)) # Tạo vùng vẽ tỷ lệ 16:9
01 #Tạo 9 vùng vẽ con
02 (ax1, ax2), (ax3, ax4) = fig2.subplots(2, 2)
03 # Hiển thị ảnh gốc
04 ax1.imshow(img, cmap='gray')
05 ax1.set_title('Ảnh gốc')
06 ax1.axis('off')
07 # Hiển thị histogram ảnh gốc
08 ax2.hist(img.flatten(),bins=256)
09 ax2.set_title('Hitogram')
10 # Hiển thị ảnh nhiễu Erlang (Gammar)
11 ax3.imshow(Gam_noisy_img, cmap='gray')
12 ax3.set_title('Ảnh nhiễu Erlang (Gammar)')
13 ax3.axis('off')
14 # Hiển thị histogram ảnh nhiễu Erlang (Gammar)
15 ax4.hist(Gam_noisy_img.flatten(),bins=256)
16 ax4.set_title('Histogram')
17 plt.show()
18
19 # 4. Tạo cửa số 4 đểhiển thị ảnh cho nhiễu nhiễu hàm mũ
```

Code minh hoạ một số hàm tạo nhiễu VII

```
20 fig2 = plt.figure(figsize=(11, 9)) # Tạo vùng vẽ tỷ lệ 16:9
21 #Tạo 4 vùng vẽ con
22 (ax1, ax2), (ax3, ax4) = fig2.subplots(2, 2)
23 # Hiển thị ảnh gốc
24 ax1.imshow(img, cmap='gray')
25 ax1.set_title('Ảnh gốc')
26 ax1.axis('off')
27 # Hiển thị histogram ảnh gốc
28 ax2.hist(img.flatten(),bins=256)
29 ax2.set_title('Hitogram')
30 # Hiển thị ảnh nhiễu nhiễu hàm mũ
31 ax3.imshow(Exp_noisy_img, cmap='gray')
32 ax3.set_title('Ảnh nhiễu hàm mũ')
33 ax3.axis('off')
34 # Hiển thị histogram ảnh nhiễu nhiễu hàm mũ
35 ax4.hist(Exp_noisy_img.flatten(),bins=256)
36 ax4.set_title('Histogram')
37 plt.show()
38
39 # 5. Tạo cửa số 5 đểhiển thị ảnh cho nhiễu nhiễu Uniform
```

Code minh hoạ một số hàm tạo nhiễu VIII

```
40  fig2 = plt.figure(figsize=(11, 9)) # Tạo vùng vẽ tỷ lệ 16:9
41  #Tạo 4 vùng vẽ con
42  (ax1, ax2), (ax3, ax4) = fig2.subplots(2, 2)
43  # Hiển thị ảnh gốc
44  ax1.imshow(img, cmap='gray')
45  ax1.set_title('Ảnh gốc')
46  ax1.axis('off')
47  # Hiển thị histogram ảnh gốc
48  ax2.hist(img.flatten(),bins=256)
49  ax2.set_title('Hitogram')
50  # Hiển thị ảnh nhiễu nhiễu Uniform
51  ax3.imshow(Uni_noisy_img, cmap='gray')
52  ax3.set_title('Ảnh nhiễu Uniform')
53  ax3.axis('off')
54  # Hiển thị histogram ảnh nhiễu Uniform
55  ax4.hist(Uni_noisy_img.flatten(),bins=256)
56  ax4.set_title('Histogram')
57  plt.show()
58
59  # 6. Tạo cửa số 6 đểhiển thị ảnh cho nhiễu nhiễu muối tiêu (nhiễu xung)
```

Code minh hoạ một số hàm tạo nhiễu IX

```
60 fig2 = plt.figure(figsize=(11, 9)) # Tạo vùng vẽ tỷ lệ 16:9
61 #Tạo 4 vùng vẽ con
62 (ax1, ax2), (ax3, ax4) = fig2.subplots(2, 2)
63 # Hiển thị ảnh gốc
64 ax1.imshow(img, cmap='gray')
65 ax1.set_title('Ảnh gốc')
66 ax1.axis('off')
67 # Hiển thị histogram ảnh gốc
68 ax2.hist(img.flatten(),bins=256)
69 ax2.set_title('Hitogram')
70 # Hiển thị ảnh nhiễu nhiễu muối tiêu (nhiễu xung)
71 ax3.imshow(SP_noisy_img, cmap='gray')
72 ax3.set_title('Ảnh nhiễu muối tiêu')
73 ax3.axis('off')
74 # Hiển thị histogram ảnh nhiễu muối tiêu (nhiễu xung)
75 ax4.hist(SP_noisy_img.flatten(),bins=256)
76 ax4.set_title('Histogram')
77 plt.show()
```

Tài liệu tham khảo

Rafael C. Gonzalez, Richard E. Woods
Digital Image Processing (2018), Fourth Edition, Global Edition, Pearson.

Ravishankar Chityala, Sridevi Pudipeddi
Image Processing and Acquisition using Python (2020), Second Edition, Chapman & Hall/CRC.

Phạm Nguyễn Minh Nhựt
Xử lý ảnh (2021), Trường Đại học Công nghệ Thông tin và Truyền thông Việt - Hàn.