

IMAGE THRESHOLDING

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



Contents

1. Simple Thresholding
2. Adaptive Thresholding
3. Otsu's Binarization



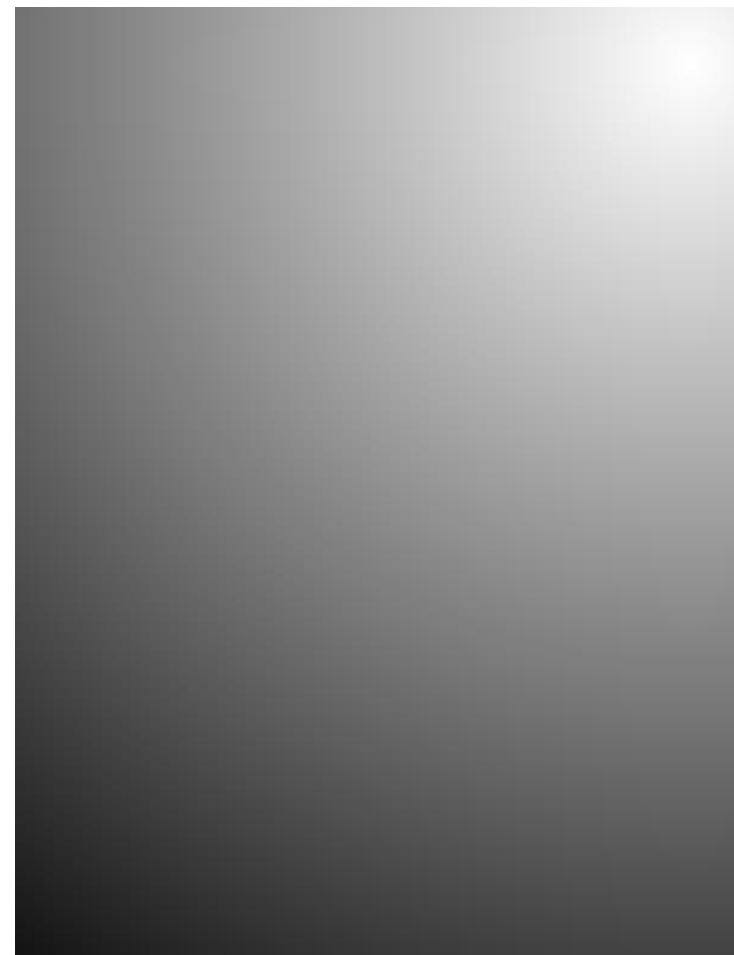
Image thresholding

REVIEW



Bài toán 0

- Bài toán: viết chương trình đọc và hiển thị ảnh mức xám có tên `gradient.tif`.



Bài toán 0

```
11.import matplotlib.pyplot as plt
```

```
12.import numpy as np
```

```
13.import cv2
```

```
14.img = cv2.imread('gradient.tif', 0)
```

```
15 plt.figure(figsize=(12, 12))
```

```
16 plt.subplot(1,3,1), plt.imshow(img, cmap='gray')
```

```
17 plt.title('Original')
```



Bài toán 0

```
11.import matplotlib.pyplot as plt
12.import numpy as np
13.import cv2
14.img = cv2.imread('gradient.tif', 0)
15.plt.figure(figsize=(12, 12))
16.plt.subplot(1,3,1),plt.imshow(img,cmap='gray')
17.plt.title('Original')
```

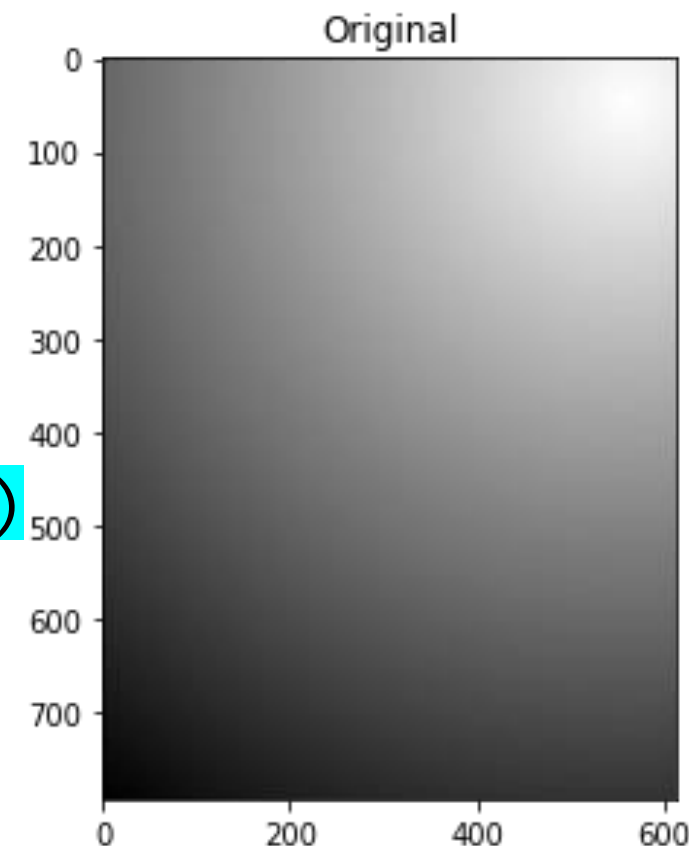


Image thresholding

SIMPLE THRESHOLDING





Phân ngưỡng ảnh đơn giản

— Hàm threshold:

- + tham số đầu tiên là 1 ảnh xám,
- + tham số thứ 2 là giá trị ngưỡng,
- + tham số thứ 3 maxval là giá trị được gán nếu giá pixel lớn hơn giá trị ngưỡng,
- + tham số thứ 4 là loại phân ngưỡng. Tùy theo các loại phân ngưỡng mà pixel được gán giá trị khác nhau.



Các loại phân ngưỡng

– THRESH_BINARY

- + Nếu giá trị pixel lớn hơn ngưỡng thì gán bằng maxval.
- + Ngược lại bằng gán bằng 0.

– THRESH_BINARY_INV

- + Nếu giá trị pixel lớn hơn ngưỡng thì gán bằng 0.
- + Ngược lại bằng gán bằng maxval.

– THRESH_TRUNC

- + Nếu giá trị pixel lớn hơn ngưỡng thì gán bằng maxval.
- + Ngược lại giữ nguyên giá trị.



Các loại phân ngưỡng

– THRESH_TOZERO

- + Nếu giá trị pixel lớn hơn ngưỡng thì giữ nguyên giá trị.
- + Ngược lại gán bằng 0.

– THRESH_TOZERO_INV

- + Nếu giá trị pixel lớn hơn ngưỡng thì gán giá trị bằng 0.
- + Ngược lại giữ nguyên.

Bài toán 1

— Bài toán: viết chương trình đọc và hiển thị ảnh mức xám có tên **threshold.png**.

+ Tạo ảnh mới từ ảnh ban đầu với ngưỡng ảnh là 128.

+ Tạo ảnh mới từ ảnh ban đầu bằng cách thay giá trị pixel lớn hơn ngưỡng thì gán bằng 0, ngược lại gán bằng maxval.

+ Hiển thị kết quả.



Bài toán 1



```
11.import matplotlib.pyplot as plt
```

```
12.import numpy as np
```

```
13.import cv2
```

```
14.img = cv2.imread('threshold.png', 0)
```

```
15.ret1, img1 = cv2.threshold(img, 127, 255,  
cv2.THRESH_BINARY)
```

```
16.ret2, img2 = cv2.threshold(img, 127, 255,  
cv2.THRESH_BINARY_INV)
```



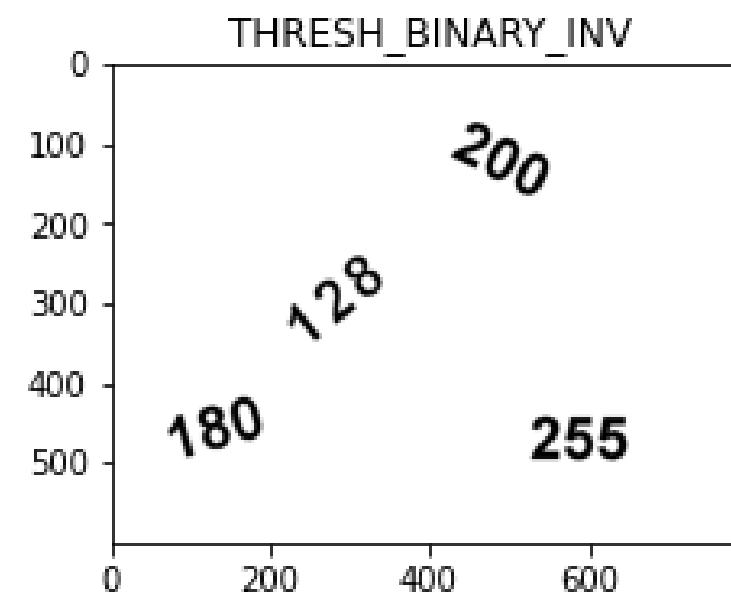
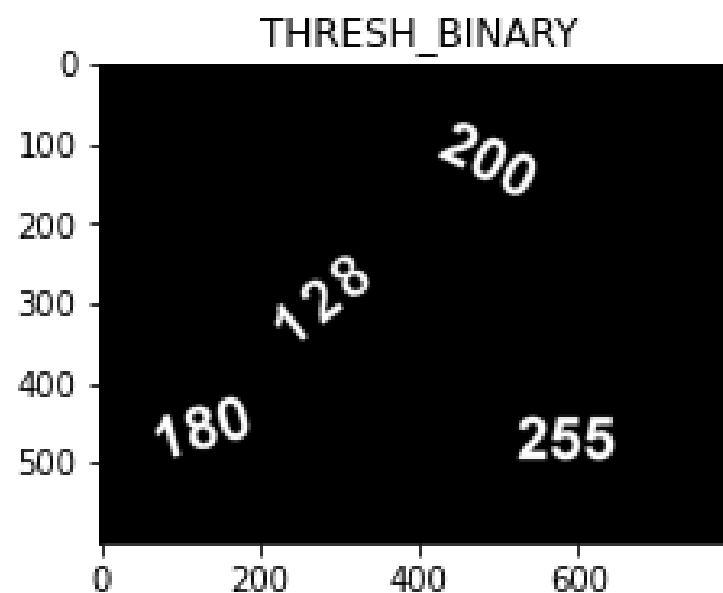
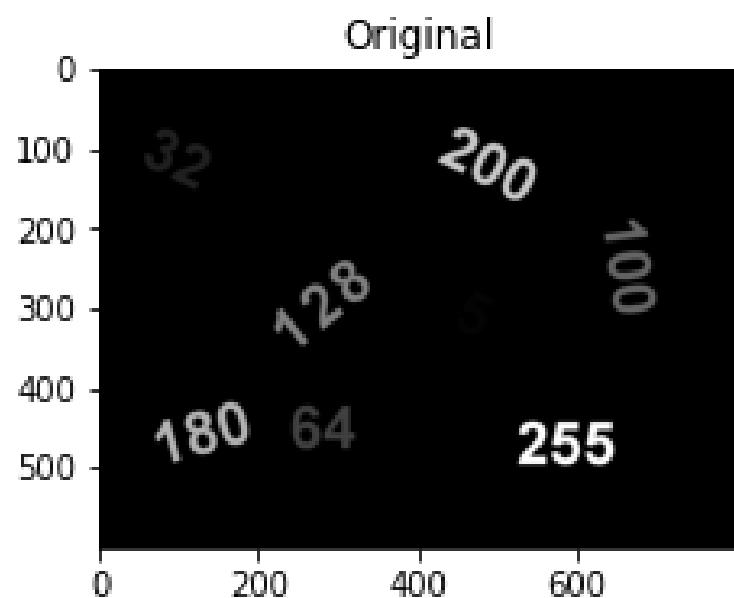
Bài toán 1



```
11. plt.figure(figsize=(12, 12))
12. plt.subplot(1,3,1), plt.imshow(img, cmap='gray')
13. plt.title('Original')
14. plt.subplot(1,3,2), plt.imshow(img1, cmap='gray')
15. plt.title('THRESH_BINARY')
16. plt.subplot(1,3,3), plt.imshow(img2, cmap='gray')
17. plt.title('THRESH_BINARY_INV')
```



Bài toán 1



Bài toán 2

— Bài toán: viết chương trình đọc và hiển thị ảnh mức xám có tên **threshold.png**.

+ Tạo ảnh mới từ ảnh ban đầu với ngưỡng ảnh là 127. Nếu giá trị pixel lớn hơn ngưỡng thì gán bằng maxval. Ngược lại giữ nguyên giá trị.

+ Hiển thị kết quả.



Bài toán 2



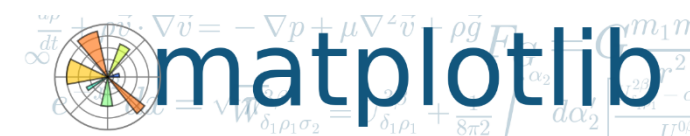
```
11.import matplotlib.pyplot as plt
```

```
12.import numpy as np
```

```
13.import cv2
```

```
14.img = cv2.imread('threshold.png', 0)
```

```
15.ret1, img1 = cv2.threshold(img, 127, 255,  
cv2.THRESH_TRUNC)
```



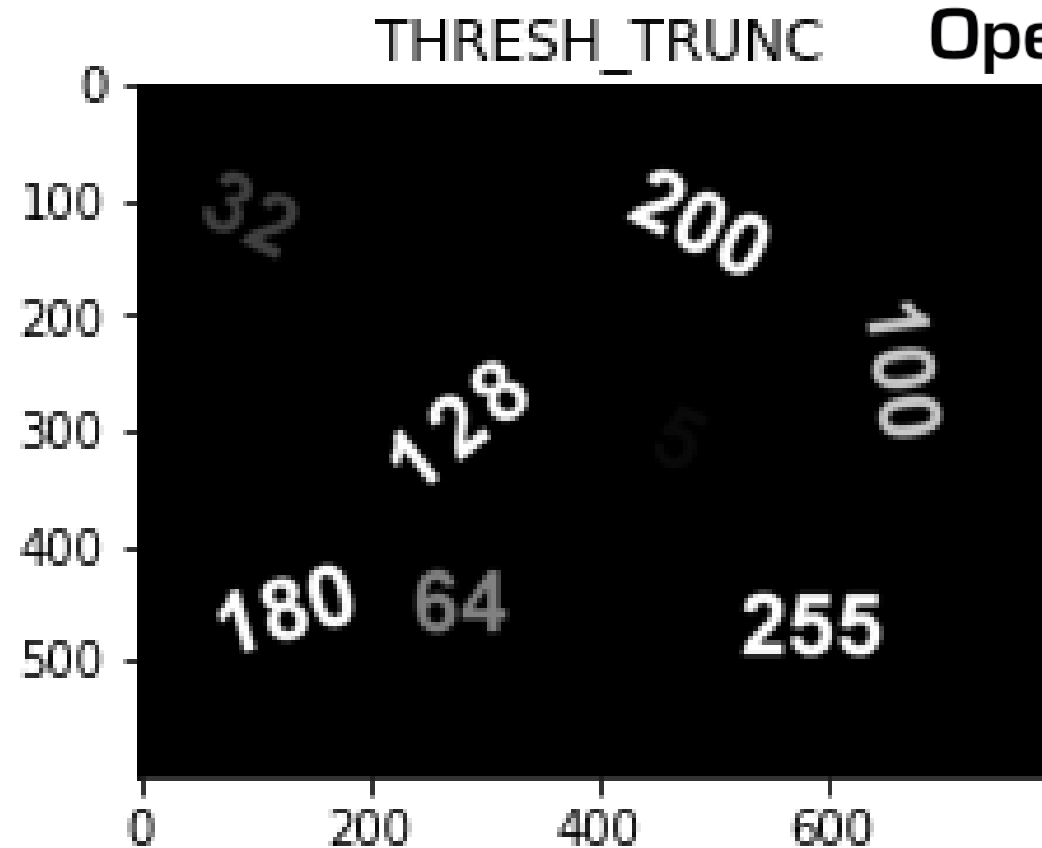
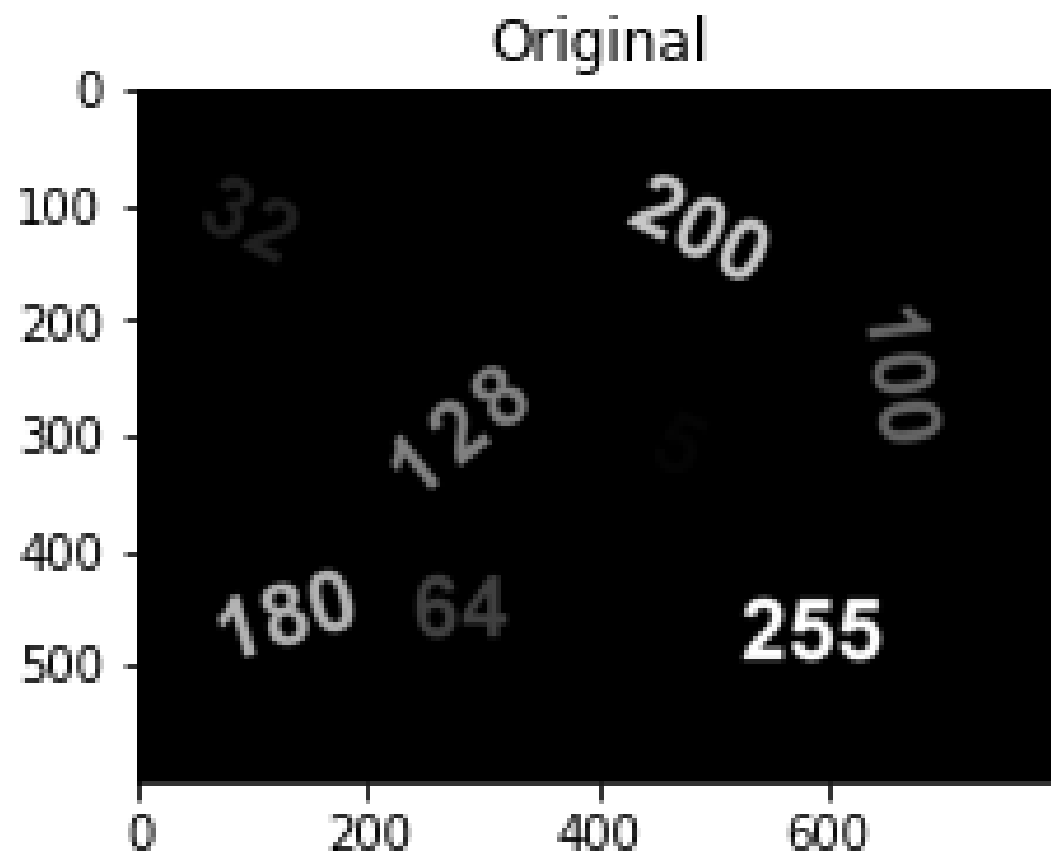
Bài toán 2



```
11.plt.figure(figsize=(12, 12))
12.plt.subplot(1,3,1),plt.imshow(img, cmap='gray')
13.plt.title('Original')
14.plt.subplot(1,3,2),plt.imshow(img1, cmap='gray')
15.plt.title('THRESH_TRUNC')
```



Bài toán 2



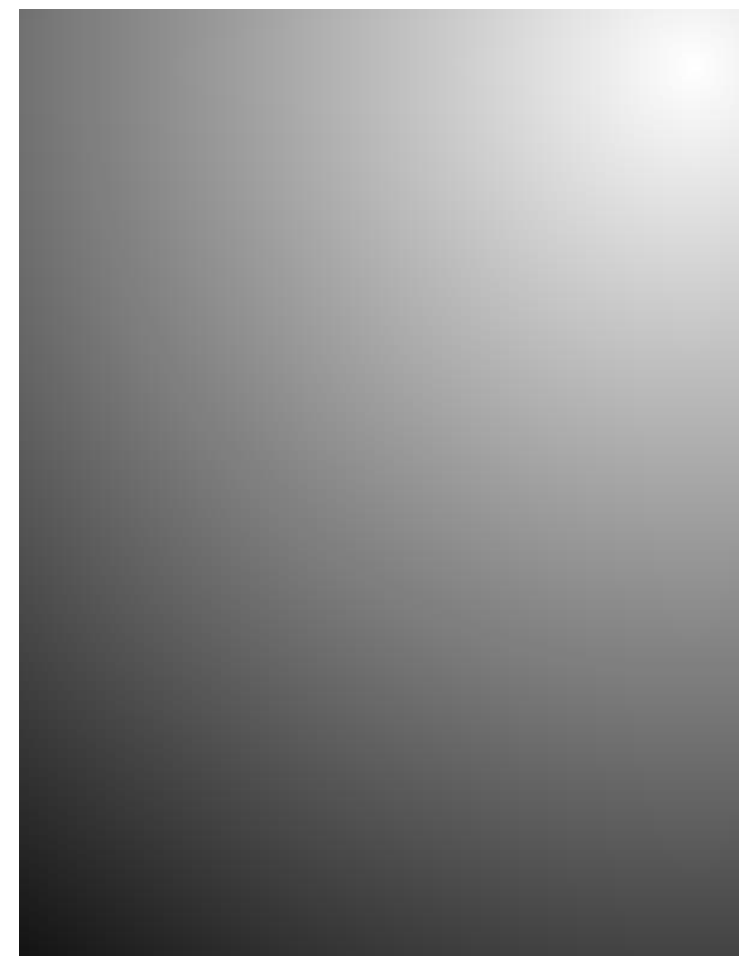
Bài toán 3

— Bài toán: viết chương trình đọc và hiển thị ảnh mức xám có tên **gradient.tif**.

+ Tạo ảnh mới từ ảnh ban đầu với ngưỡng ảnh là 127. Nếu giá trị pixel lớn hơn ngưỡng thì giữ nguyên giá trị. Ngược lại gán bằng 0.

+ Tạo ảnh mới từ ảnh ban đầu bằng cách ...INV cách trên

+ Hiển thị kết quả.



Bài toán 3



```
11.import matplotlib.pyplot as plt
```

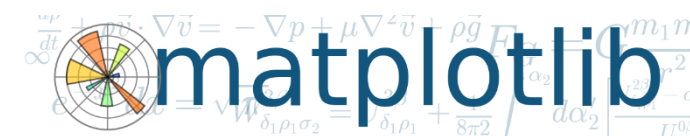
```
12.import numpy as np
```

```
13.import cv2
```

```
14.img = cv2.imread('gradient.tif', 0)
```

```
15.ret1, img1 = cv2.threshold(img,127,255,  
cv2.THRESH_TOZERO)
```

```
16.ret2, img2 = cv2.threshold(img,127,255,  
cv2.THRESH_TOZERO_INV)
```





Bài toán 3

```

11.plt.figure(figsize=(12, 12))
12.plt.subplot(1,3,1),plt.imshow(img, cmap='gray')
13.plt.title('Original')
14.plt.subplot(1,3,2),plt.imshow(img1, cmap='gray')
15.plt.title('THRESH_TOZERO')
16.plt.subplot(1,3,3),plt.imshow(img2, cmap='gray')
17.plt.title('THRESH_TOZERO_INV')

```

Bài toán 3

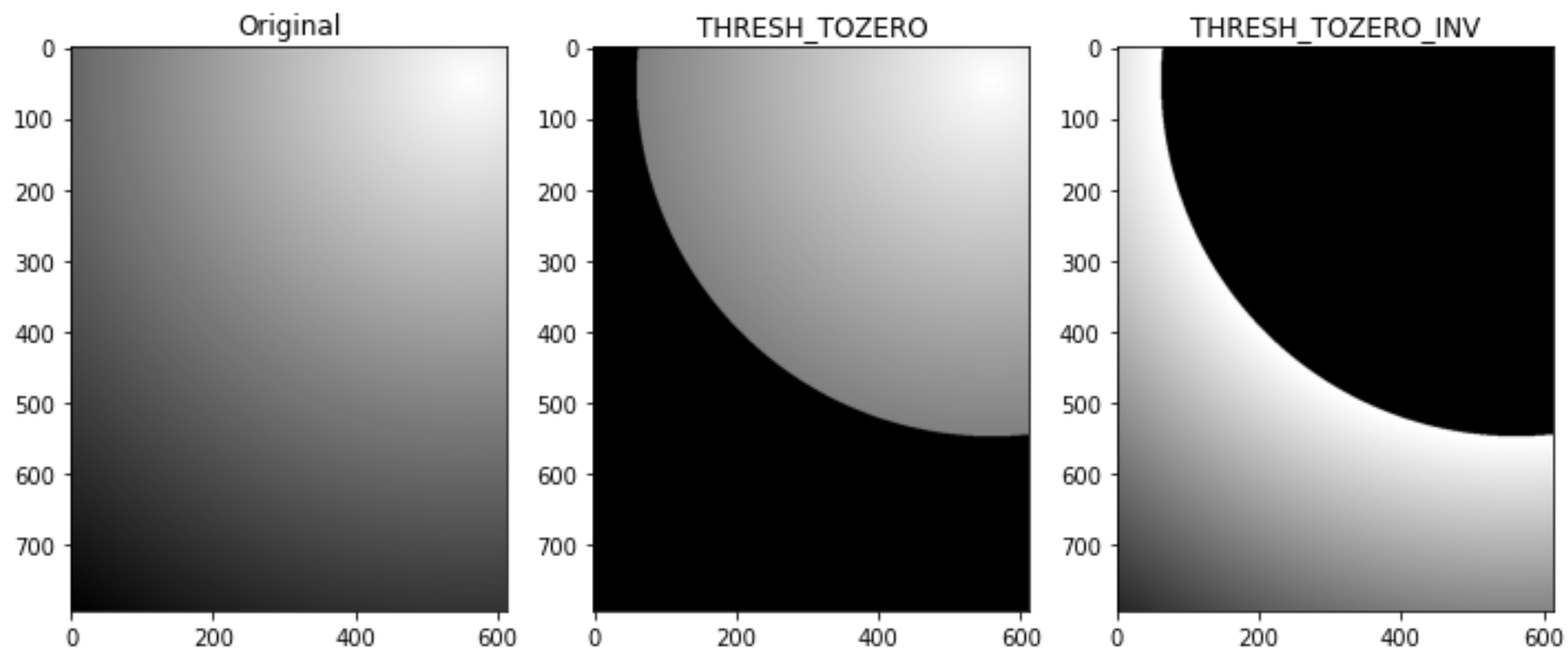


Image thresholding

ADAPTIVE THRESHOLDING



Phân ngưỡng thích nghi

- Phương pháp Adaptive Thresholding sẽ chia bức ảnh thành nhiều vùng và mỗi vùng sẽ có giá trị ngưỡng riêng.
- Phương pháp Simple Thresholding, chúng ta sử dụng một giá trị ngưỡng duy nhất cho toàn bộ bức ảnh.
- Nhược điểm của phương pháp này có thể sẽ cho kết quả không tốt nếu các vùng trong bức ảnh có nguồn sáng không đồng đều.

Phân ngưỡng thích nghi

- Sử dụng hàm `cv2.adaptiveThreshold()` cho phương pháp Adaptive Thresholding. Hàm này có 3 tham số đặc biệt và một đầu ra.
 - + Tham số Adaptive Method: cho biết cách xác định giá trị ngưỡng cho mỗi vùng
 - `ADAPTIVE_THRESH_MEAN_C`: ngưỡng là giá trị trung bình của các vùng lân cận.
 - `ADAPTIVE_THRESH_GAUSSIAN_C`: ngưỡng là giá trị trung bình nhưng có trọng số của các vùng lân cận.
 - + Block Size: nó cho biết kích thước của vùng lân cận.
 - + C: một hằng số cho biết lượng khấu trừ từ giá trị trung bình.

Bài toán 4



— Bài toán: viết chương trình đọc và hiển thị ảnh mức xám có tên `gradient_with_text.tif`.

- + Tạo ảnh mới từ ảnh ban đầu với
- + Tạo ảnh mới từ ảnh ban đầu bằng cách...
- + Tạo ảnh mới từ ảnh ban đầu bằng cách...
- + Hiển thị kết quả.

What is MATLAB?

MATLAB® is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include Math and computation, Algorithm development, Data acquisition, Modeling, simulation, and prototyping, Data analysis, exploration, and visualization, Scientific and engineering graphics, Application development, including graphical user interface building.

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non-interactive language such as C or Fortran.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB engines incorporate the LAPACK and BLAS libraries, embedding the state of the art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of add-on application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

The MATLAB System

The MATLAB system consists of five main parts:

Development Environment. This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the search path.

The MATLAB Mathematical Function Library. This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

The MATLAB Language. This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create large and complex application programs.

Graphics. MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level functions that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

The MATLAB External Interfaces (API). This is a library that allows you to write C and Fortran programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

Bài toán 4



```
11.import matplotlib.pyplot as plt
```

```
12.import numpy as np
```

```
13.import cv2
```

```
14.img = cv2.imread('gradient_with_text.tif', 0)
```

```
15.ret1, img1 = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)
```

```
16.img2 = cv2.adaptiveThreshold(img, 255,  
                                cv2.ADAPTIVE_THRESH_MEAN_C, cv2.THRESH_BINARY, 9, 2)
```

```
17.img3 = cv2.adaptiveThreshold(img, 255,  
                                cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY, 9, 2)
```



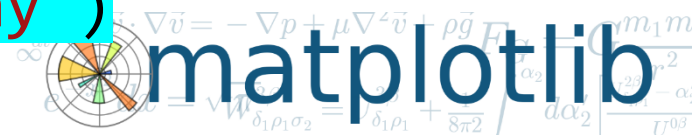
Bài toán 4



```

11. plt.figure(figsize=(12, 12))
12. plt.subplot(2,2,1), plt.imshow(img, cmap='gray')
13. plt.title('Original')
14. plt.subplot(2,2,2), plt.imshow(img1, cmap='gray')
15. plt.title('THRESH_BINARY')
16. plt.subplot(2,2,3), plt.imshow(img2, cmap='gray')
17. plt.title('ADAPTIVE_THRESH_MEAN_C')
18. plt.subplot(2,2,4), plt.imshow(img3, cmap='gray')
19. plt.title('ADAPTIVE_THRESH_GAUSSIAN_C')

```



Adaptive Thresholding

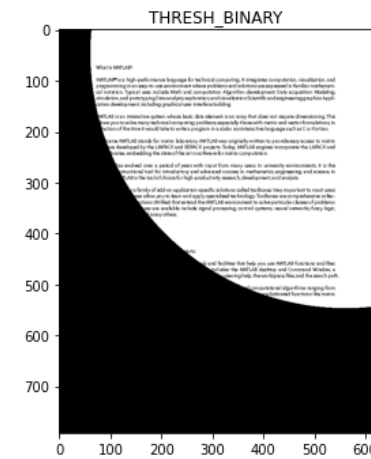
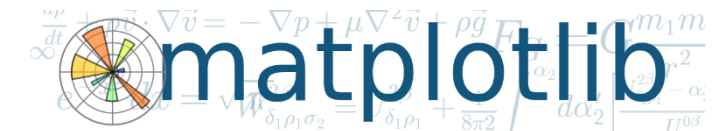


Image thresholding

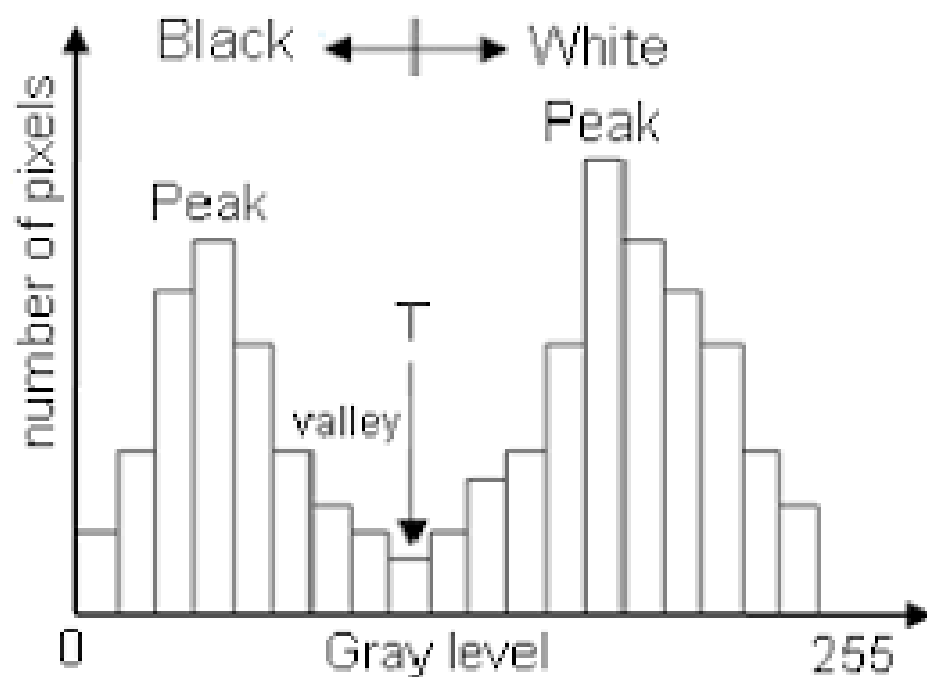
OTSU'S BINARIZATION



Bimodal images

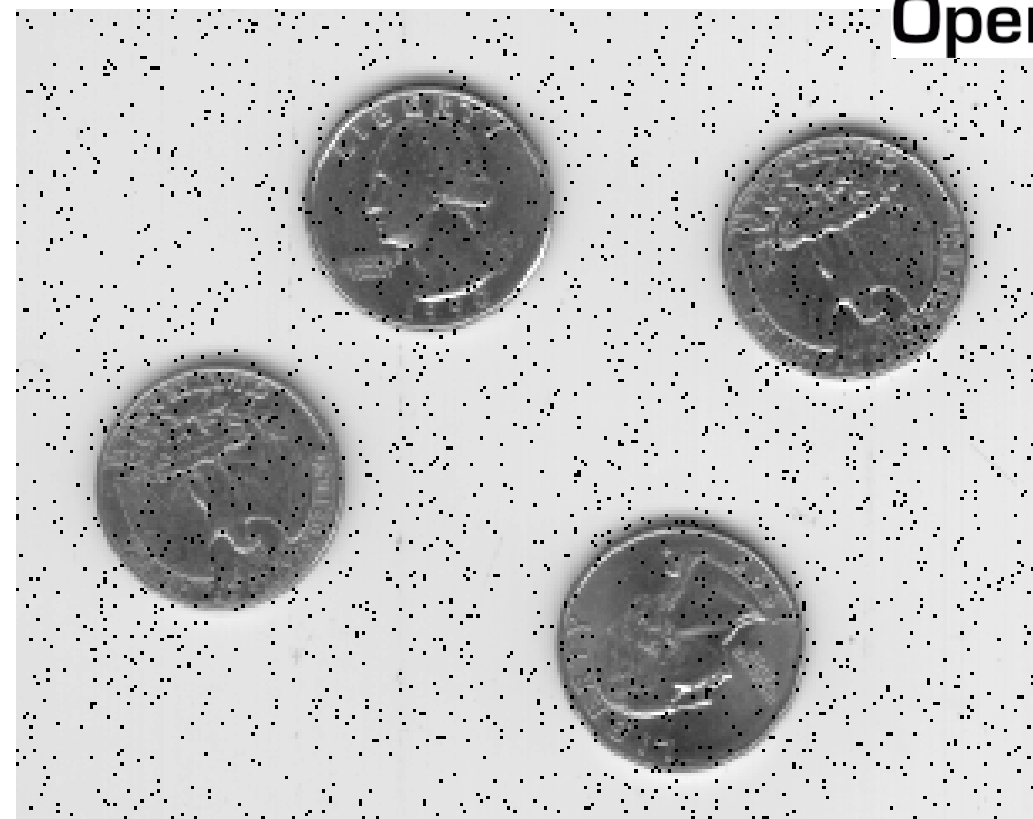


- Ảnh bimodal là ảnh mà biểu đồ tần xuất thống kê số lần xuất hiện các mức sáng trong ảnh (histogram) có hai đỉnh.



Bài toán 5

- Bài toán: viết chương trình đọc và hiển thị ảnh mức xám có tên `eight_pepper.tif`. và vẽ biểu đồ tần xuất thống kê số lần xuất hiện các mức sáng trong ảnh (histogram).





Bài toán 5

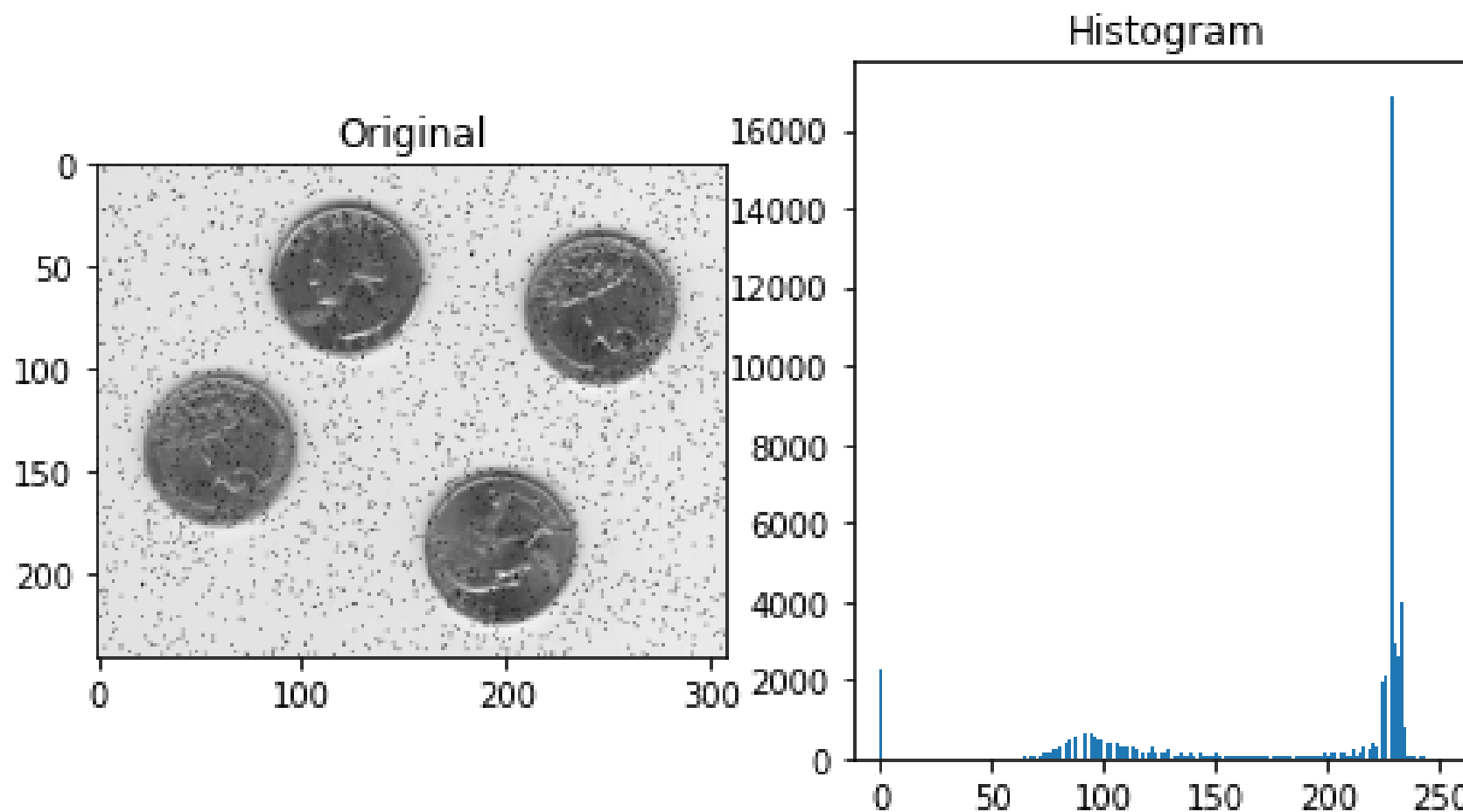
```

11.import matplotlib.pyplot as plt
12.import numpy as np
13.import cv2

14.img = cv2.imread('eight_pepper.tif', 0)
15.plt.figure(figsize=(12, 4))
16.plt.subplot(1,3,1),plt.imshow(img, cmap='gray')
17.plt.title('Original')
18.plt.subplot(1,3,2),plt.hist(img.ravel(),256,[0,256])
19.plt.title('Histogram')

```

Bài toán 5

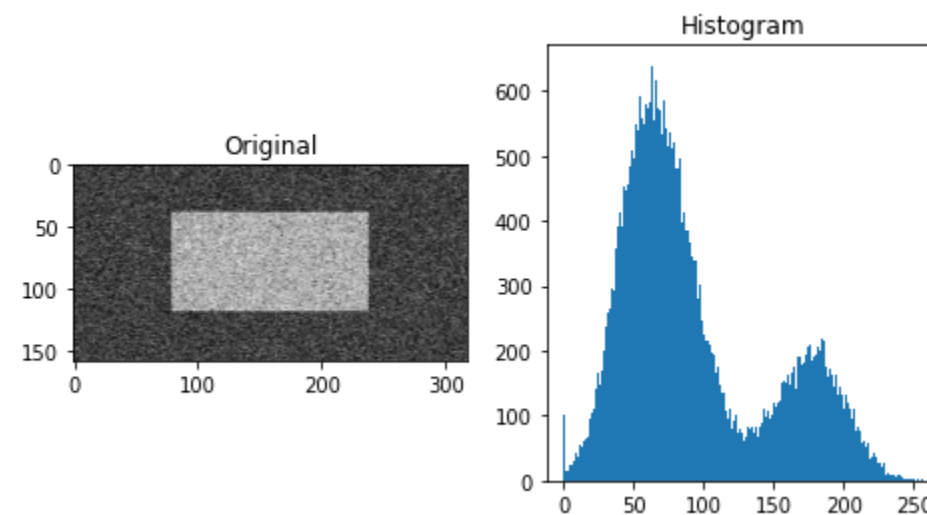


Nhị phân hóa với thuật toán Otsu



- Trong kỹ thuật phân ngưỡng ảnh đơn giản, ta lấy ngưỡng
 - + tùy ý hoặc
 - + dựa vào kinh nghiệm,nhưng giá trị ngưỡng đó cho kết quả tốt hay ko?
- Câu trả lời là phải chạy thực nghiệm mới có đánh giá được tốt hay không.

Nhị phân hóa với thuật toán Otsu



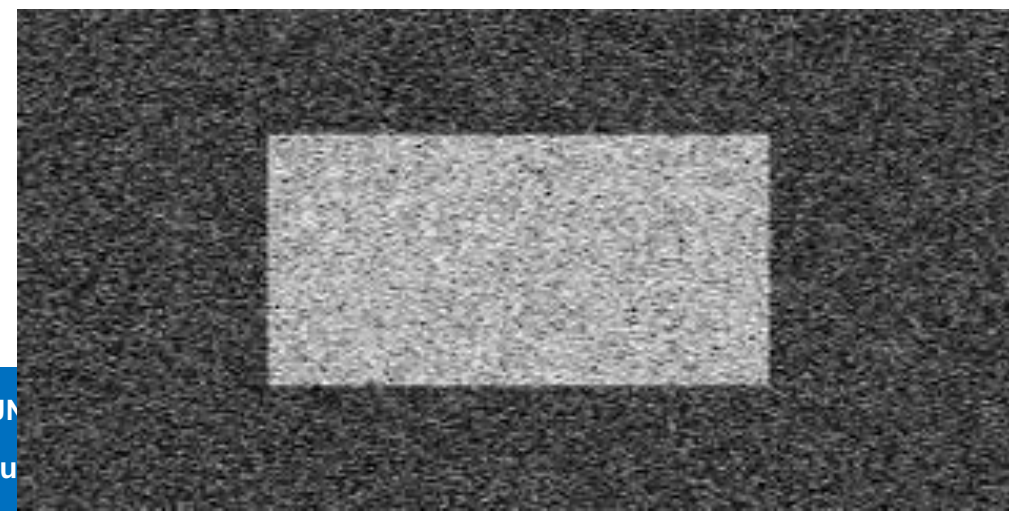
Bài toán 6

- Bài toán: viết chương trình đọc và hiển thị ảnh mức xám có tên `eight_pepper.tif` thực hiện các yêu cầu sau:



Bài toán 7

- Bài toán: viết chương trình đọc và hiển thị ảnh mức xám có tên **noisy2.png** thực hiện các yêu cầu sau:



Bài toán 7

```

11.import matplotlib.pyplot as plt
12.import numpy as np
13.import cv2
14.img = cv2.imread('noisy2.png', 0)
15.ret1, th1 = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)
16.ret2, th2 = cv2.threshold(img, 0, 255,
                             cv2.THRESH_BINARY + cv2.THRESH_OTSU)
17.blur = cv2.GaussianBlur(img, (5, 5), 0)
18.ret3, th3 = cv2.threshold(blur, 0, 255,
                             cv2.THRESH_BINARY + cv2.THRESH_OTSU)

```




Bài toán 7

```

11.import matplotlib.pyplot as plt
12.import numpy as np
13.import cv2
14.img = cv2.imread('noisy2.jpg', 0)
15.ret1, th1 = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)
16.ret2, th2 = cv2.threshold(img, 0, 255,
                             cv2.THRESH_BINARY + cv2.THRESH_OTSU)
17.blur = cv2.GaussianBlur(img, (5, 5), 0)
18.ret3, th3 = cv2.threshold(blur, 0, 255,
                             cv2.THRESH_BINARY + cv2.THRESH_OTSU)

```



Bài toán 7

```

— # plot all the images and their histograms
— images = [img,0,th1,img,0,th2,blur,0,th3]
— titles = [
    'Original Noisy Image', 'Histogram',
    'Global Thresholding (v=127)',
    'Original Noisy Image', '', "Otsu's Thresholding",
    'Gaussian filtered Image', '', "Otsu's Thresholding"]

```



Bài toán 7

```

11. for i in range(3):
12.     plt.subplot(3,3,i*3+1),plt.imshow(images[i*3], 'gray')
13.     plt.title(titles[i*3]),plt.xticks([]), plt.yticks([])
14.     plt.subplot(3,3,i*3+2),plt.hist(images[i*3].ravel(),256)
15.     plt.title(titles[i*3+1]),plt.xticks([]),plt.yticks([])
16.     plt.subplot(3,3,i*3+3),plt.imshow(images[i*3+2], 'gray')
17.     plt.title(titles[i*3+2]),plt.xticks([]),plt.yticks([])
18. plt.show()

```

Bài toán 7

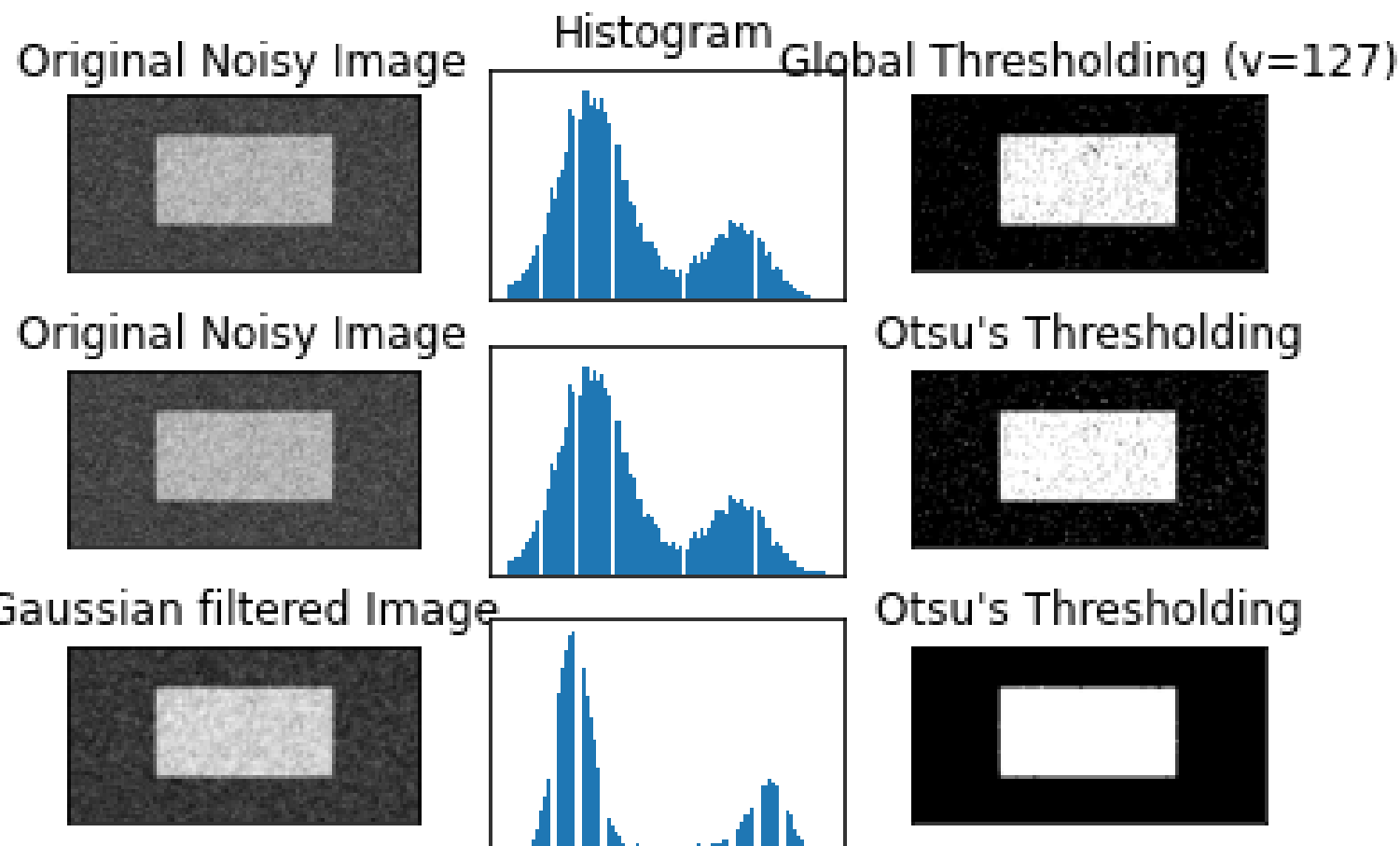


Image thresholding

OTSU'S BINARIZATION

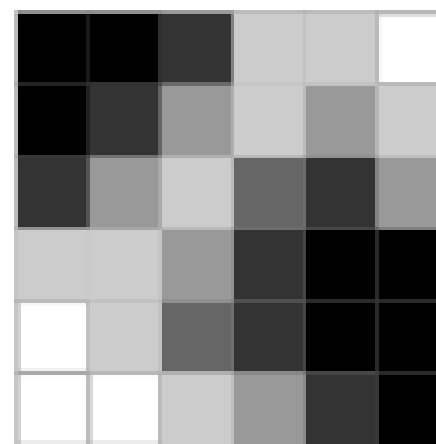
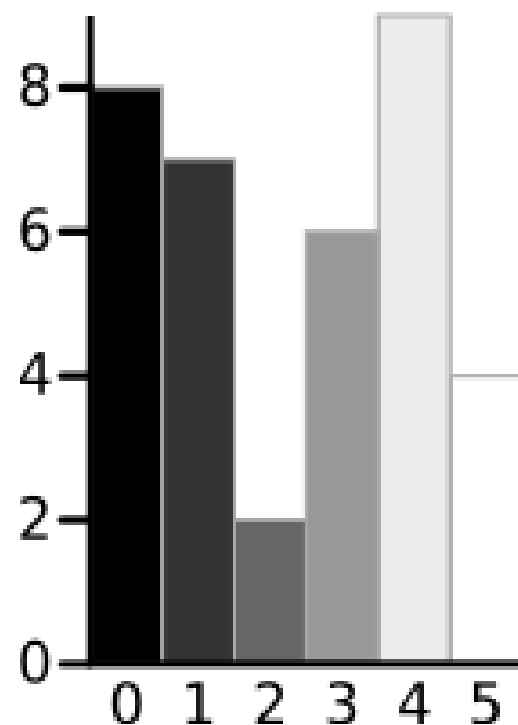


Otsu's Binarization

- Otsu được lấy tên theo tên tác giả của phương pháp này Nobuyuki Otsu đã giới thiệu về kĩ thuật này trong tạp chí của IEEE Transactions on Systems, Man, and Cybernetics.
- **A Threshold Selection Method from Gray-Level Histograms, 1979.**
- https://cw.fel.cvut.cz/wiki/_media/courses/a6m33bio/otsu.pdf
- <http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html>

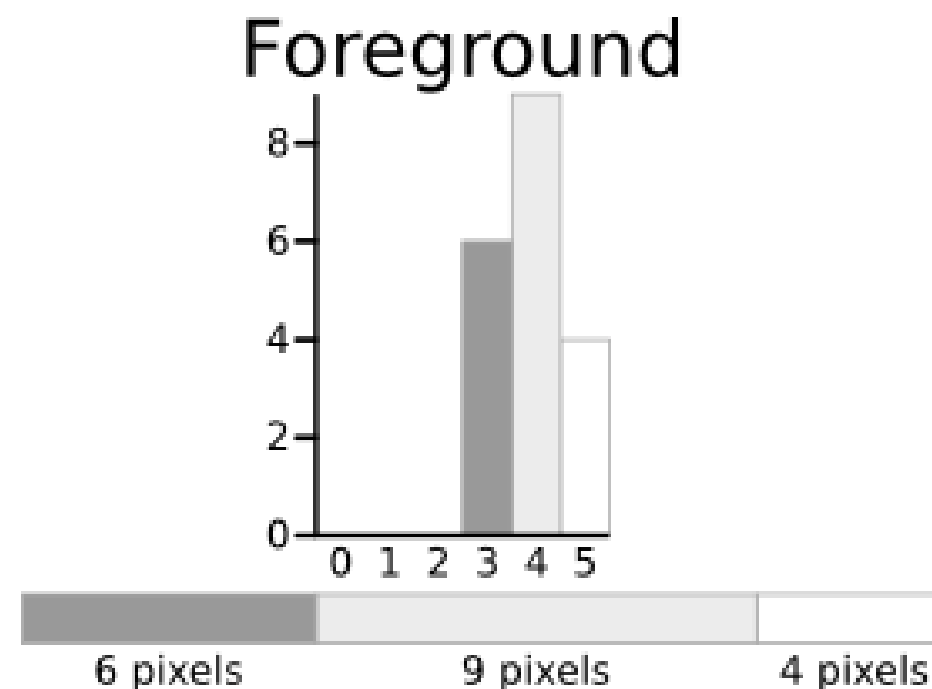
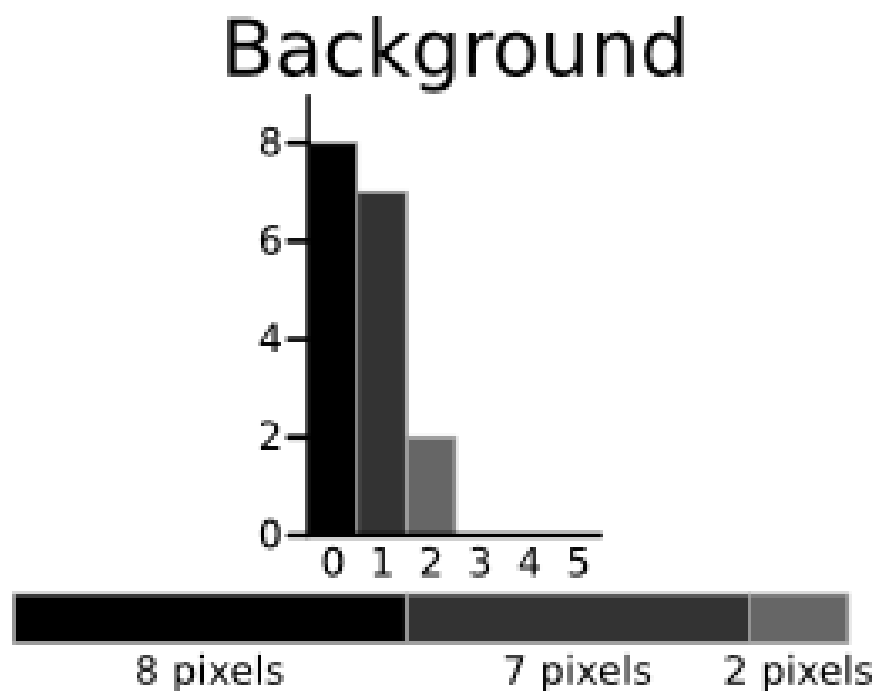
Otsu's Binarization

— Bức ảnh 6×6 với 6 mức xám và có histogram như sau:



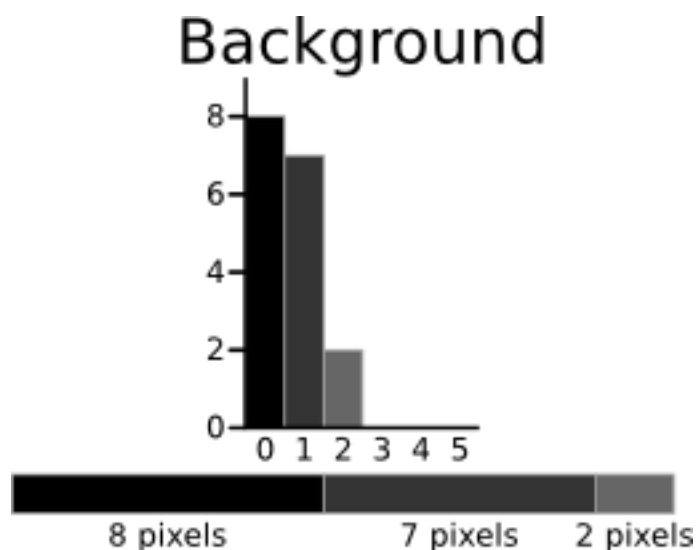
Otsu's Binarization

- Lựa chọn ngưỡng ban đầu bằng $T = 3$, ta tính toán trên 2 tập $Background < 3$ và $Foreground \geq 3$ như sau:



Otsu's Binarization

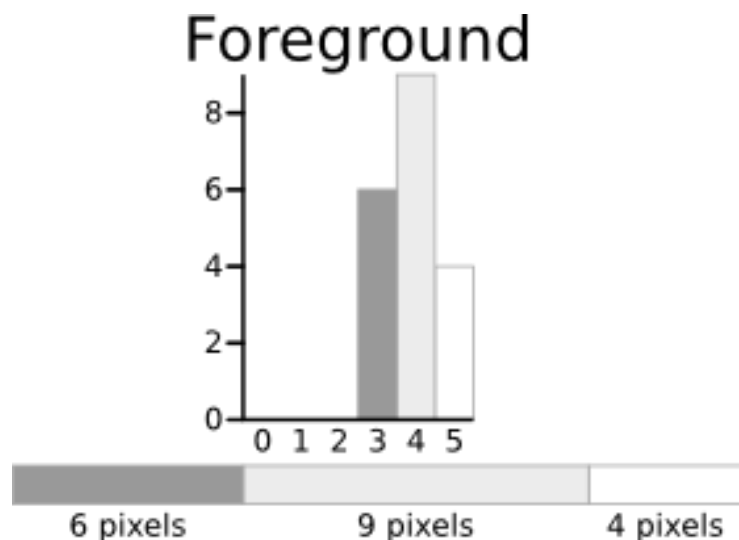
- Tính trọng số (weight – ω), giá trị trung bình (mean – μ), phương sai (variance – σ^2).



$$\begin{aligned} \text{Weight } W_b &= \frac{8 + 7 + 2}{36} = 0.4722 \\ \text{Mean } \mu_b &= \frac{(0 \times 8) + (1 \times 7) + (2 \times 2)}{17} = 0.6471 \\ \text{Variance } \sigma_b^2 &= \frac{((0 - 0.6471)^2 \times 8) + ((1 - 0.6471)^2 \times 7) + ((2 - 0.6471)^2 \times 2)}{17} \\ &= \frac{(0.4187 \times 8) + (0.1246 \times 7) + (1.8304 \times 2)}{17} \\ &= 0.4637 \end{aligned}$$

Otsu's Binarization

- Tính trọng số (weight – ω), giá trị trung bình (mean – μ), phương sai (variance – σ^2).



$$\text{Weight } W_f = \frac{6 + 9 + 4}{36} = 0.5278$$

$$\text{Mean } \mu_f = \frac{(3 \times 6) + (4 \times 9) + (5 \times 4)}{19} = 3.8947$$

$$\begin{aligned} \text{Variance } \sigma_f^2 &= \frac{((3 - 3.8947)^2 \times 6) + ((4 - 3.8947)^2 \times 9) + ((5 - 3.8947)^2 \times 4)}{19} \\ &= \frac{(4.8033 \times 6) + (0.0997 \times 9) + (4.8864 \times 4)}{19} \\ &= 0.5152 \end{aligned}$$

Otsu's Binarization

— Within-Class Variance – Tính phương sai theo lớp.

$$\sigma_W^2 = W_b \sigma_b^2 + W_f \sigma_f^2.$$

$$-\sigma_W^2 = W_b \sigma_b^2 + W_f \sigma_f^2$$

$$-\sigma_W^2 = 0.4722 * 0.4637 + 0.5278 * 0.5152$$

$$-\sigma_W^2 = 0.4909$$



Otsu's Binarization

– Tính toán tương tự như trên với tất cả các giá trị ngưỡng có thể trong bức ảnh

$$+ T = 0$$

$$+ T = 1$$

$$+ T = 2$$

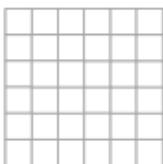
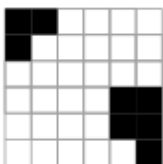
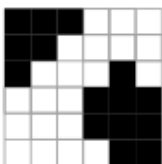
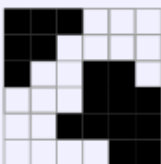
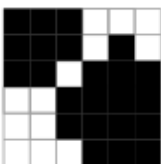
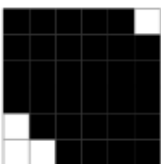
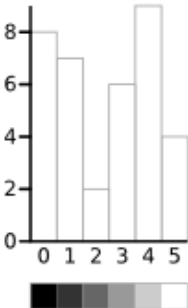
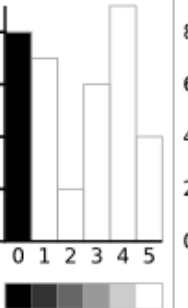
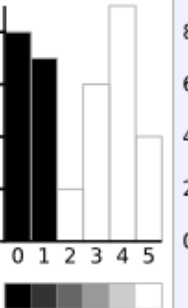
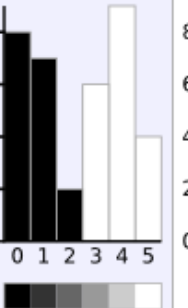
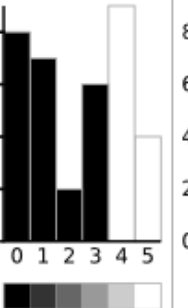
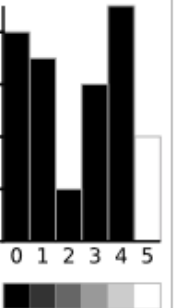
$$+ T = 3$$

$$+ T = 4$$

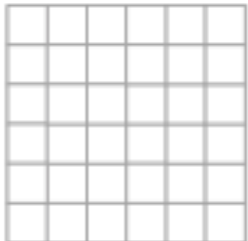
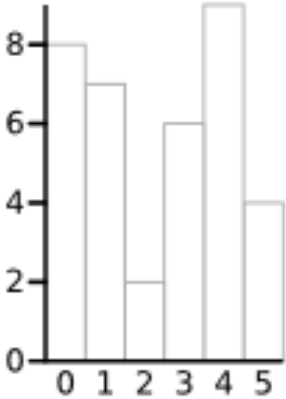

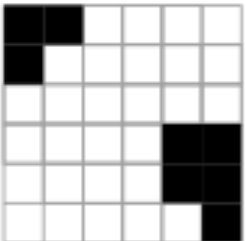
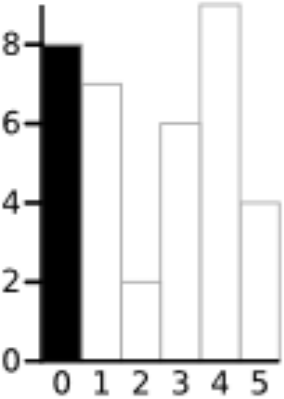

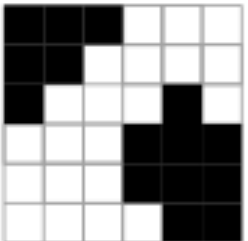
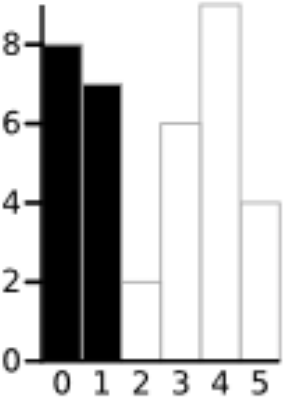

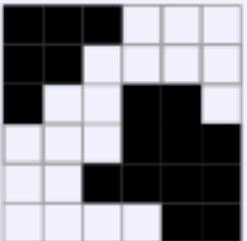
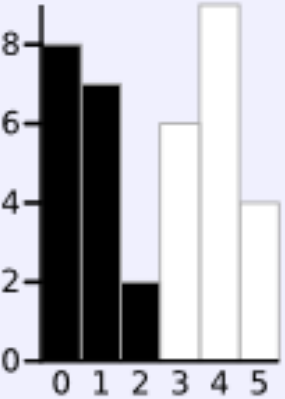

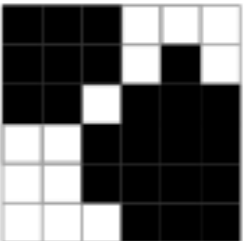
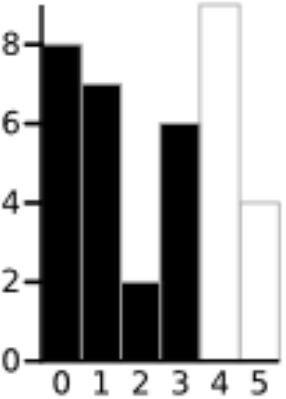

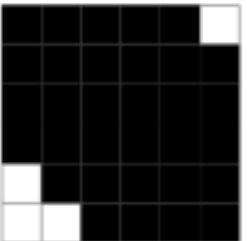
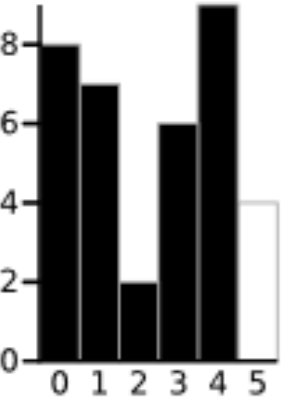

$$+ T = 5$$



Otsu's Binarization

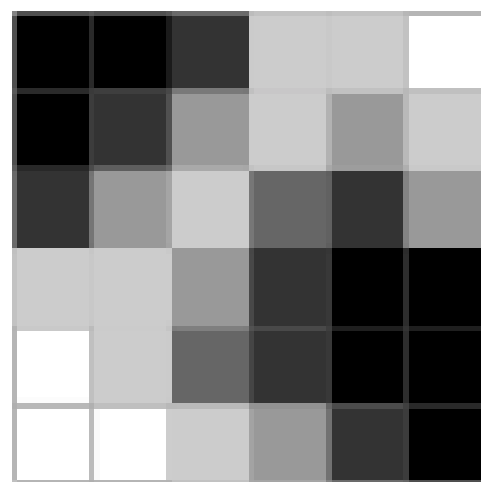
| Threshold | T=0 | T=1 | T=2 | T=3 | T=4 | T=5 |
|-----------------------|---|--|---|---|---|---|
| |  |  |  |  |  |  |
| |  |  |  |  |  |  |
| Weight, Background | $W_b = 0$ | $W_b = 0.222$ | $W_b = 0.4167$ | $W_b = 0.4722$ | $W_b = 0.6389$ | $W_b = 0.8889$ |
| Mean, Background | $M_b = 0$ | $M_b = 0$ | $M_b = 0.4667$ | $M_b = 0.6471$ | $M_b = 1.2609$ | $M_b = 2.0313$ |
| Variance, Background | $\sigma_b^2 = 0$ | $\sigma_b^2 = 0$ | $\sigma_b^2 = 0.2489$ | $\sigma_b^2 = 0.4637$ | $\sigma_b^2 = 1.4102$ | $\sigma_b^2 = 2.5303$ |
| Weight, Foreground | $W_f = 1$ | $W_f = 0.7778$ | $W_f = 0.5833$ | $W_f = 0.5278$ | $W_f = 0.3611$ | $W_f = 0.1111$ |
| Mean, Foreground | $M_f = 2.3611$ | $M_f = 3.0357$ | $M_f = 3.7143$ | $M_f = 3.8947$ | $M_f = 4.3077$ | $M_f = 5.0000$ |
| Variance, Foreground | $\sigma_f^2 = 3.1196$ | $\sigma_f^2 = 1.9639$ | $\sigma_f^2 = 0.7755$ | $\sigma_f^2 = 0.5152$ | $\sigma_f^2 = 0.2130$ | $\sigma_f^2 = 0$ |
| Within Class Variance | $\sigma_W^2 = 3.1196$ | $\sigma_W^2 = 1.5268$ | $\sigma_W^2 = 0.5561$ | $\sigma_W^2 = 0.4909$ | $\sigma_W^2 = 0.9779$ | $\sigma_W^2 = 2.2491$ |



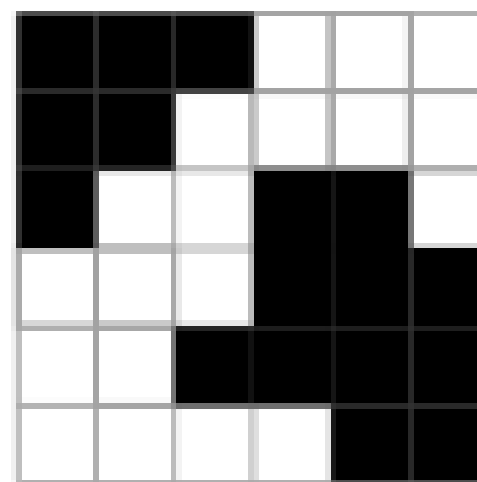
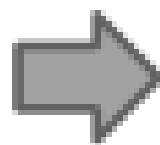
| Threshold | T=0 | T=1 | T=2 | T=3 | T=4 | T=5 |
|-----------------------|--|---|--|--|--|--|
| |    |    |    |    |    |    |
| Weight, Background | $W_b = 0$ | $W_b = 0.222$ | $W_b = 0.4167$ | $W_b = 0.4722$ | $W_b = 0.6389$ | $W_b = 0.8889$ |
| Mean, Background | $M_b = 0$ | $M_b = 0$ | $M_b = 0.4667$ | $M_b = 0.6471$ | $M_b = 1.2609$ | $M_b = 2.0313$ |
| Variance, Background | $\sigma_b^2 = 0$ | $\sigma_b^2 = 0$ | $\sigma_b^2 = 0.2489$ | $\sigma_b^2 = 0.4637$ | $\sigma_b^2 = 1.4102$ | $\sigma_b^2 = 2.5303$ |
| Weight, Foreground | $W_f = 1$ | $W_f = 0.7778$ | $W_f = 0.5833$ | $W_f = 0.5278$ | $W_f = 0.3611$ | $W_f = 0.1111$ |
| Mean, Foreground | $M_f = 2.3611$ | $M_f = 3.0357$ | $M_f = 3.7143$ | $M_f = 3.8947$ | $M_f = 4.3077$ | $M_f = 5.000$ |
| Variance, Foreground | $\sigma_f^2 = 3.1196$ | $\sigma_f^2 = 1.9639$ | $\sigma_f^2 = 0.7755$ | $\sigma_f^2 = 0.5152$ | $\sigma_f^2 = 0.2130$ | $\sigma_f^2 = 0$ |
| Within Class Variance | $\sigma_W^2 = 3.1196$ | $\sigma_W^2 = 1.5268$ | $\sigma_W^2 = 0.5561$ | $\sigma_W^2 = 0.4909$ | $\sigma_W^2 = 0.9779$ | $\sigma_W^2 = 2.2491$ |

Otsu's Binarization

- Chọn giá trị có **Within-Class Variance** nhỏ nhất,
- Suy ra mức ngưỡng thích hợp là $T = 3$.



$T=3$



Otsu's Binarization

Within Class Variance $\sigma_W^2 = W_b \sigma_b^2 + W_f \sigma_f^2$ (as seen above)

Between Class Variance $\sigma_B^2 = \sigma^2 - \sigma_W^2$
 $= W_b(\mu_b - \mu)^2 + W_f(\mu_f - \mu)^2$ (where $\mu = W_b \mu_b + W_f \mu_f$)
 $= W_b W_f (\mu_b - \mu_f)^2$

| Threshold | T=0 | T=1 | T=2 | T=3 | T=4 | T=5 |
|------------------------|-----------------------|-----------------------|-----------------------|---|-----------------------|-----------------------|
| Within Class Variance | $\sigma_W^2 = 3.1196$ | $\sigma_W^2 = 1.5268$ | $\sigma_W^2 = 0.5561$ | $\sigma_W^2 = 0.4909$ | $\sigma_W^2 = 0.9779$ | $\sigma_W^2 = 2.2491$ |
| Between Class Variance | $\sigma_B^2 = 0$ | $\sigma_B^2 = 1.5928$ | $\sigma_B^2 = 2.5635$ | $\sigma_B^2 = 2.6287$ | $\sigma_B^2 = 2.1417$ | $\sigma_B^2 = 0.8705$ |

Chúc các bạn học tốt
Thân ái chào tạm biệt các bạn

ĐẠI HỌC QUỐC GIA TP.HCM
ĐẠI HỌC CÔNG NGHỆ THÔNG TIN TP.HCM
TOÀN DIỆN – SÁNG TẠO – PHỤNG SỰ