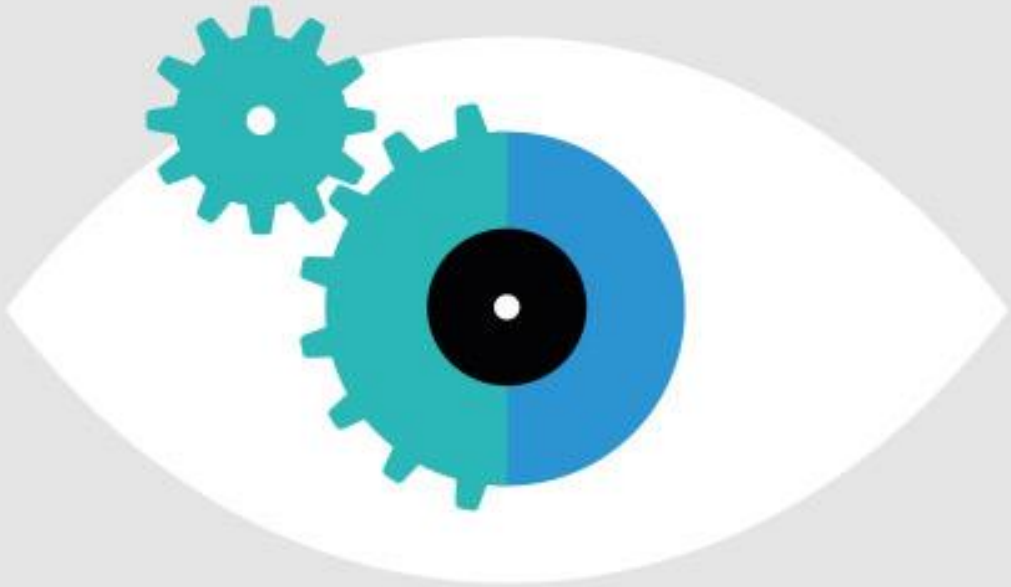


Computer Vision

THỊ GIÁC MÁY TÍNH

ThS. Huỳnh Minh Vũ
Khoa Kỹ thuật cơ khí
Trường Đại học Kỹ thuật – Công nghệ Cần Thơ
Email: hmvu@ctuet.edu.vn

Chương 4: Xử lý hình thái ảnh

4.1 Giới thiệu

4.2 Phần tử cấu trúc

4.3 Các phép toán cơ bản

4.4 Các phép toán kết hợp

4.1 Giới thiệu

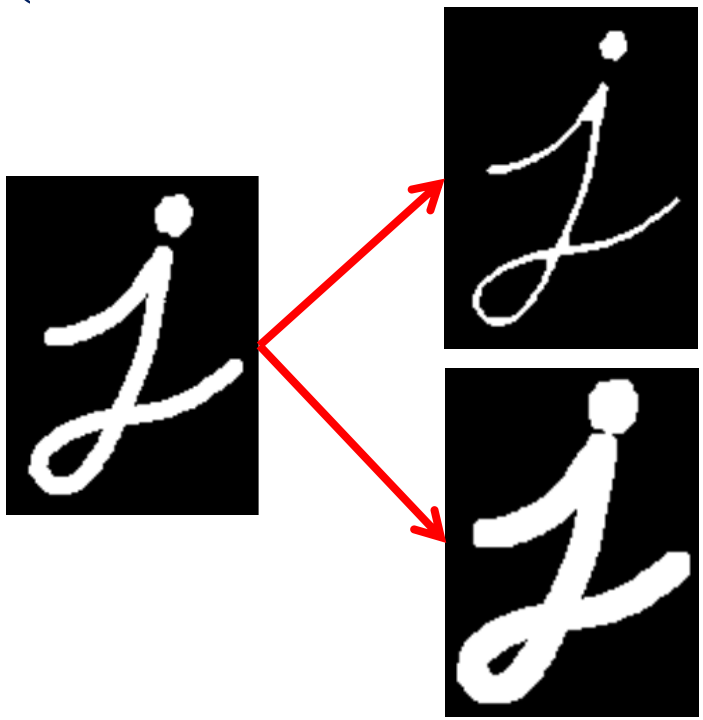
- Trong lĩnh vực xử lý và phân tích ảnh, một trong những công việc quan trọng là trích lọc được những đặc trưng của đối tượng, mô tả hình dáng và nhận dạng mẫu.
- Xử lý hình thái là những phép toán liên quan đến cấu trúc hình học của các đối tượng trong ảnh.
- Một trong những khái niệm thường đề cập liên quan đến hình học của đối tượng như: kích thước, hình dáng và hướng của đối tượng trong ảnh.

4.1 Giới thiệu

- Trong quá trình xử lý ảnh, các kỹ thuật xử lý hình thái ảnh có thể được sử dụng để:
  - Loại bỏ những phần không hoàn chỉnh (làm sạch) ảnh kết quả sau khi phân ngưỡng bằng cách xóa nhiễu hoặc lấp chỗ trống của một vùng.
  - Làm trơn ảnh kết quả sau khi phân ngưỡng.
  - Cung cấp thông tin dựa trên cấu trúc và hình dạng của ảnh sau khi phân ngưỡng.
  - Thao tác hậu xử lý của phân đoạn ngưỡng.
- Biến đổi hình thái học là các phép biến đổi thường được thực hiện trên ảnh nhị phân. Để thực hiện các phép biến đổi hình thái ta cần hai đầu vào chính bao gồm bức ảnh gốc và một thành phần khác gọi là **structuring element** hoặc **kernel**.

4.1 Giới thiệu

- Các phép toán xử lý hình thái học bao gồm:
  - Phép co (Erosion)
  - Phép dãn nở (Dilation)
  - Phép toán mở (Open = Erosion + Dilation)
  - Phép toán đóng (Close = Dilation + Erosion)

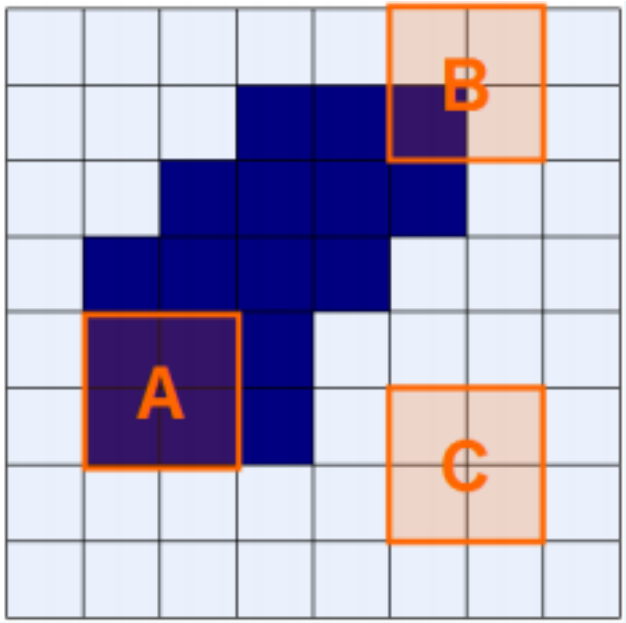


4.2 Phần tử cấu trúc (structuring element)

- Phần tử cấu trúc là một ảnh có kích thước nhỏ gồm có hai giá trị 0 và 1, các giá trị bằng 0 được bỏ qua trong quá trình tính toán.
- Gọi *H(i, j)* là phần tử cấu trúc của ảnh nhị phân và sẽ được biểu diễn như sau:

$$H(i, j) \in \{0,1\}$$

- Phần tử cấu trúc sẽ trượt trên ảnh theo một trong 2 cách:
  - Fit: Tất cả các pixels của phần tử cấu trúc phủ trên các pixels của ảnh.
  - Hit: Một phần (một số pixels) của phần tử cấu trúc sẽ phủ lên các pixels của ảnh.



4.2 Phần tử cấu trúc (structuring element)

- Về cơ bản cơ chế thực hiện của phần tử cấu trúc gần giống như bộ lọc trong miền không gian.
- Phần tử cấu trúc sẽ di chuyển trên tất các các pixel của ảnh gốc và cho một giá trị pixels mới trên ảnh kết quả tùy theo phép toán hình thái được áp dụng:
  - Erosion: Pixel vật thể (1) thành pixel nền (0).
  - Dilation: Pixel nền (0) thành pixel vật thể (1)
- Một số hình dáng của phần tử cấu trúc phẳng:

Hình vuông

Hình kim cương

Hình chéo

Hình đường ngang

Hình đường dọc

4.3 Các phép toán cơ bản

Phép co (Erosion)

- Erosion là phép biến đổi làm co kích thước của đổi tượng, giúp tách rời các đối tượng gần nhau, làm mảnh và tìm xương của đổi tượng.
- Erosion của một ảnh f bởi một phần tử cấu trúc s được ký hiệu là:

$$f \ominus s$$

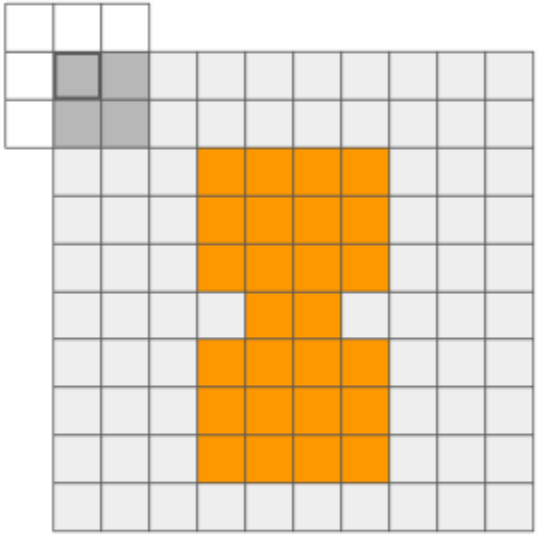
- Gọi (x, y) là vị trí tâm của phần tử cấu trúc s trong ảnh gốc f. Khi đó, giá trị của pixel $f(x,y)$ sẽ được cập nhật thành $g(x,y)$ theo quy tắc:

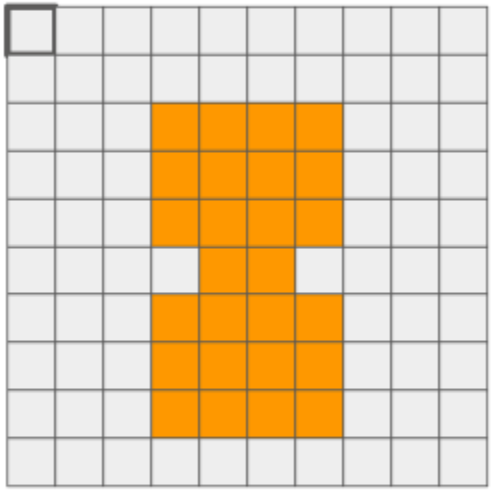
$$g(x,y) = \begin{cases} 0, & \textit{nếu s fits f} \\ 1, & \textit{otherwise} \end{cases}$$

- Quy tắc này có thể diễn giải đơn giản: Nếu một trong các pixels của mặt nạ s là pixel nền (0) thì pixel tại vị trí trung tâm của mặt nạ (phủ lên ảnh f) cũng là nền (0).

4.3 Các phép toán cơ bản

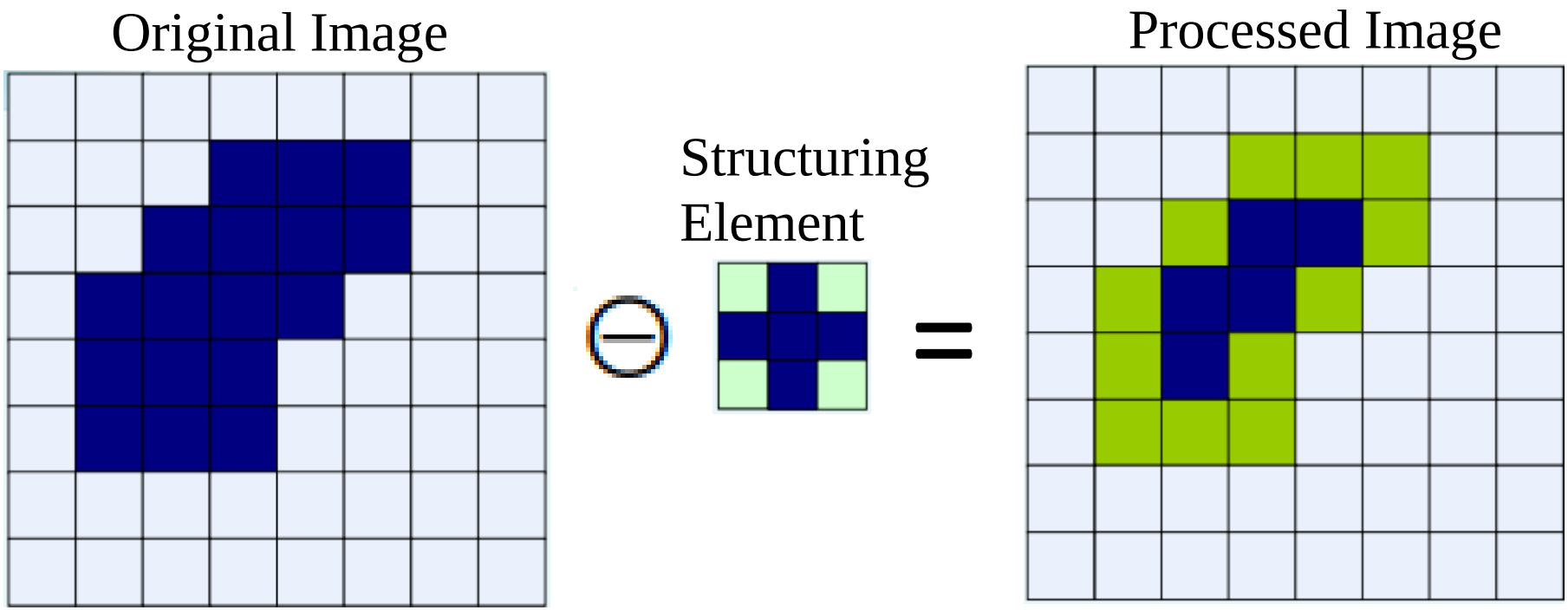
Phép co (Erosion)





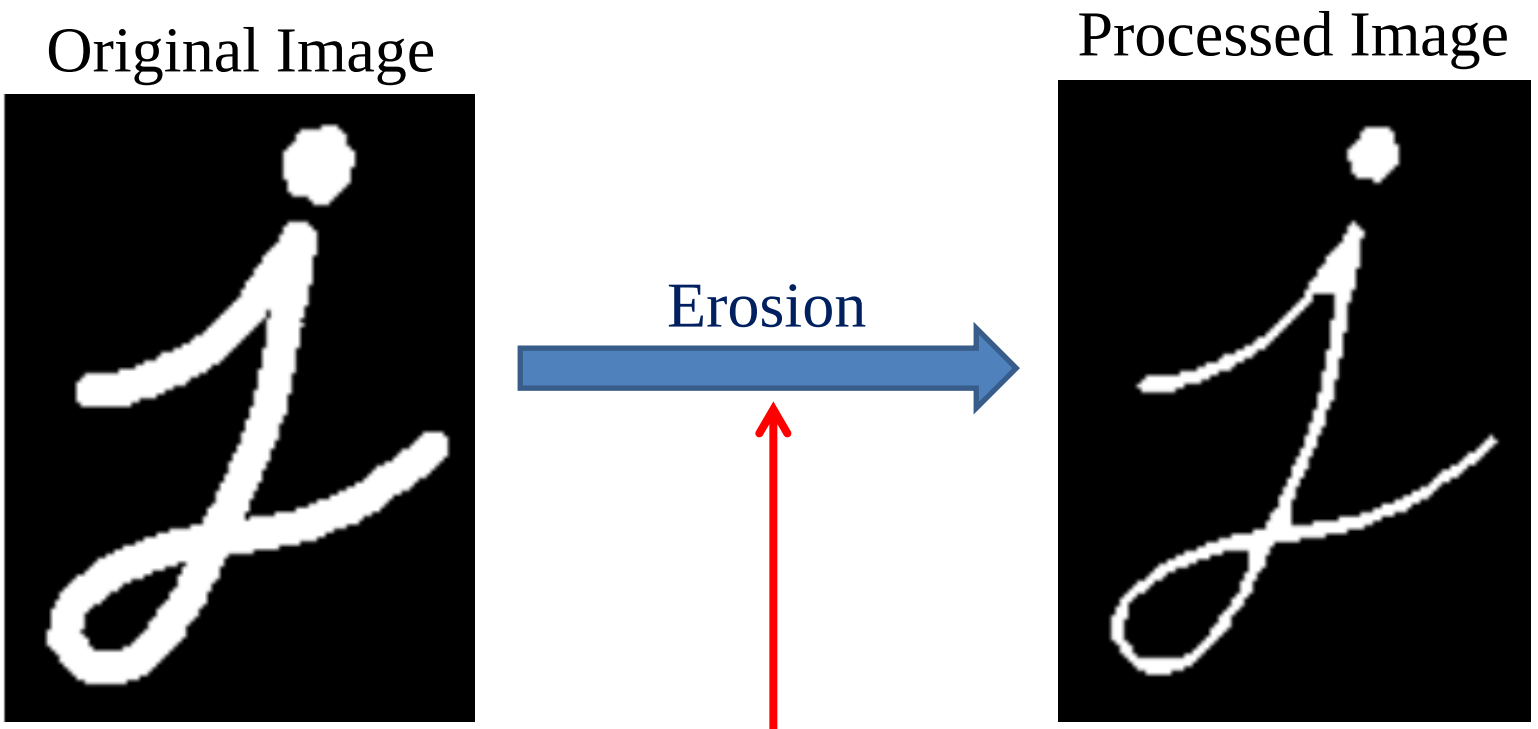
4.3 Các phép toán cơ bản

Phép co (Erosion)



4.3 Các phép toán cơ bản

Phép co (Erosion)




```
kernel = np.ones((5, 5), np.uint8)
erosion = cv2.erode(img, kernel, iterations=1)
```

4.3 Các phép toán cơ bản

Phép dãn nở (Dilation)

- Dilation là phép biến đổi làm giãn kích thước của đối tượng.
- Dilation của một ảnh f bởi một phần tử cấu trúc s được ký hiệu là:

$$f \oplus s$$

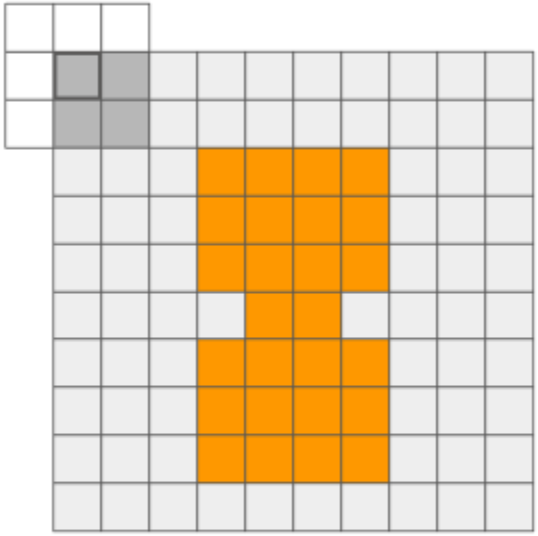
- Gọi (x, y) là vị trí tâm của phần tử cấu trúc s trong ảnh gốc f. Khi đó, giá trị của pixel $f(x,y)$ sẽ được cập nhật thành $g(x,y)$ theo quy tắc:

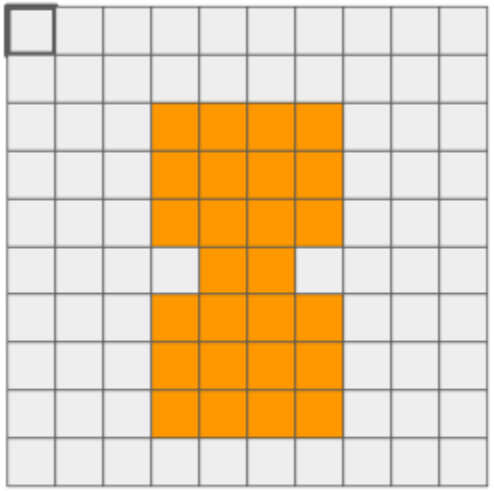
$$g(x,y) = \begin{cases} 1, & \textit{nếu s fits f} \\ 0, & \textit{otherwise} \end{cases}$$

- Quy tắc này có thể diễn giải đơn giản: Nếu một trong các pixels của mặt nạ s là pixel vật thể (1) thì pixel tại vị trí trung tâm của mặt nạ (phủ lên ảnh f) cũng là vật thể (1).

4.3 Các phép toán cơ bản

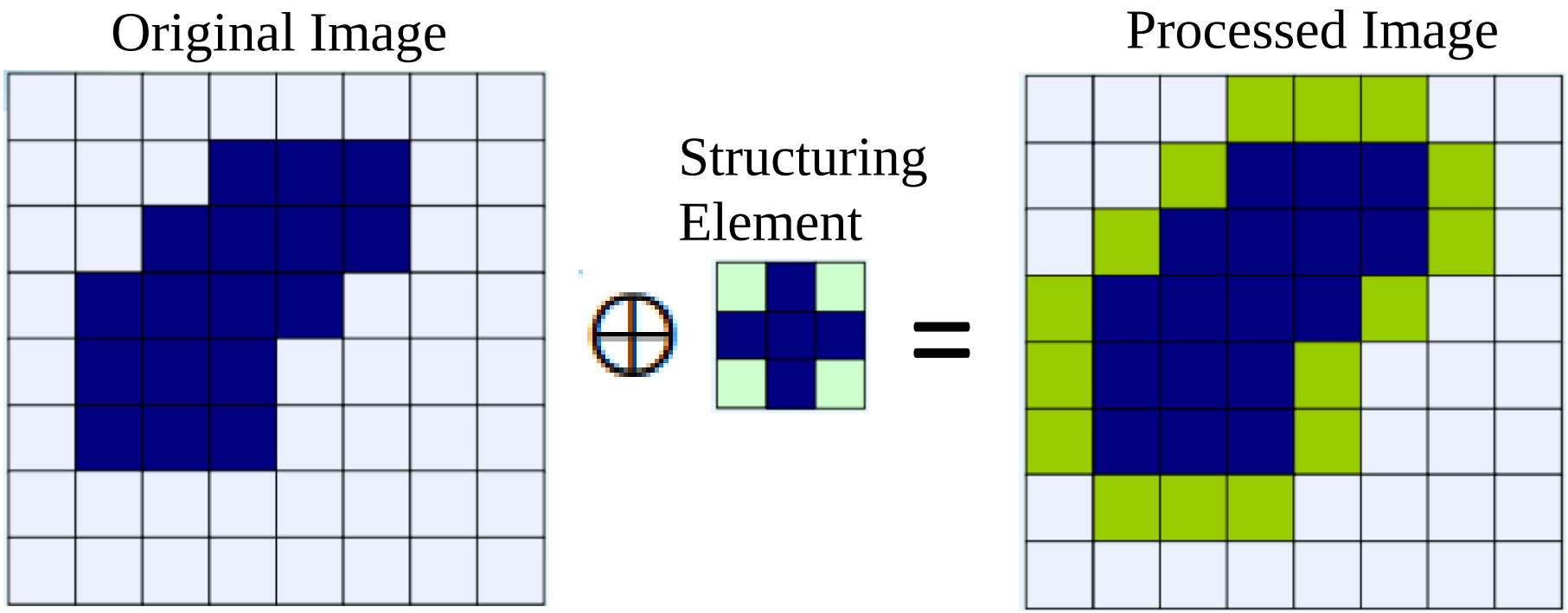
Phép dãn nở (Dilation)





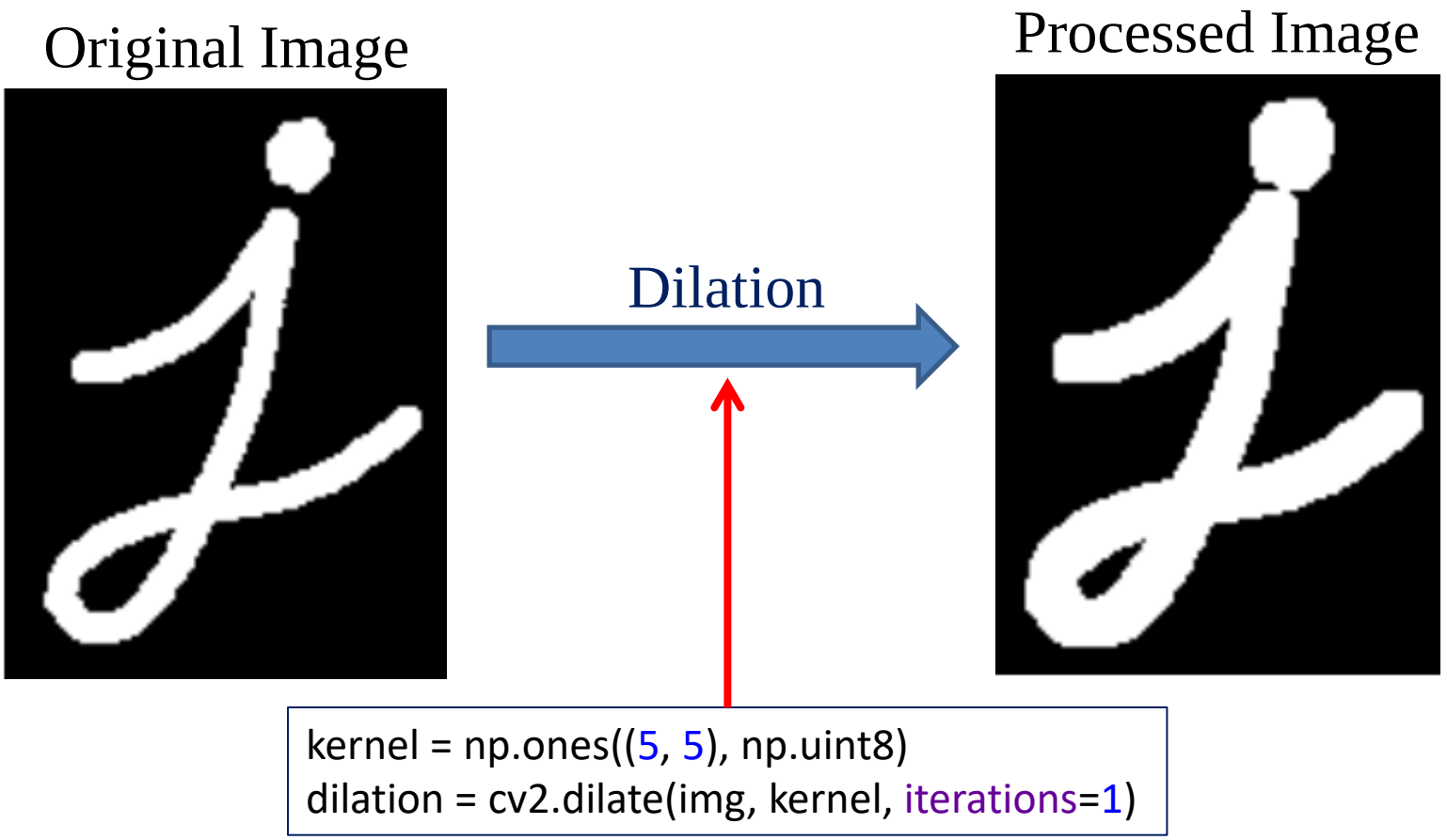
4.3 Các phép toán cơ bản

Phép dãn nở (Dilation)



4.3 Các phép toán cơ bản

Phép dãn nở (Dilation)



4.4 Các phép toán kết hợp

Phép toán mở (Opening)

- Phép mở ảnh giúp loại bỏ các phần lồi lõm, làm cho đường bao đối tượng trong ảnh trở lên mượt hơn, loại bỏ nhiễu hạt tiêu của ảnh nhị phân. Cụ thể như, có thể giúp loại bỏ các nét thừa của ký tự trong ảnh chụp văn bản.
- Opening của một ảnh f bởi một phần tử cấu trúc s được ký hiệu và tính theo công thức sau:

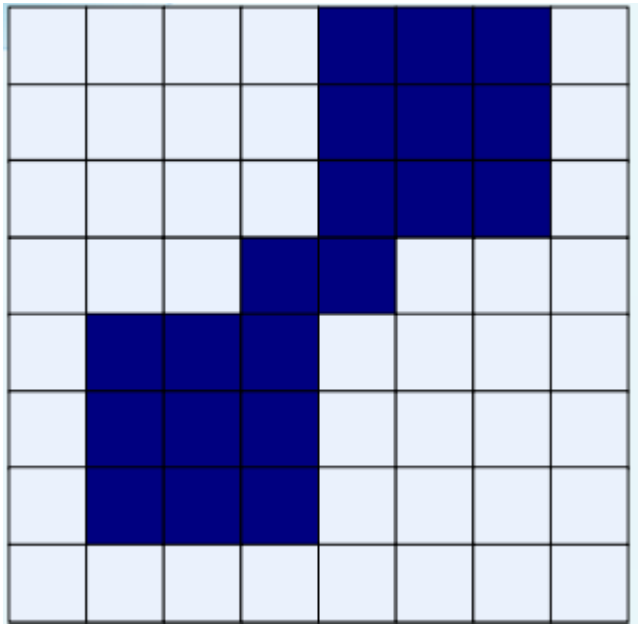
$$f \circ s = (f \ominus s) \oplus s$$

- Opening thực ra là việc thực hiện phép Erosion, rồi sau đó thực hiện phép Dilation. Để thực hiện phép Opening ta sử dụng hàm **cv2.morphologyEx().**

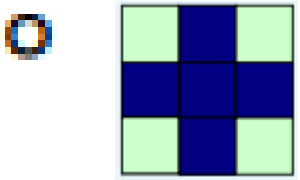
4.4 Các phép toán kết hợp

Phép toán mở (Opening)

Original Image

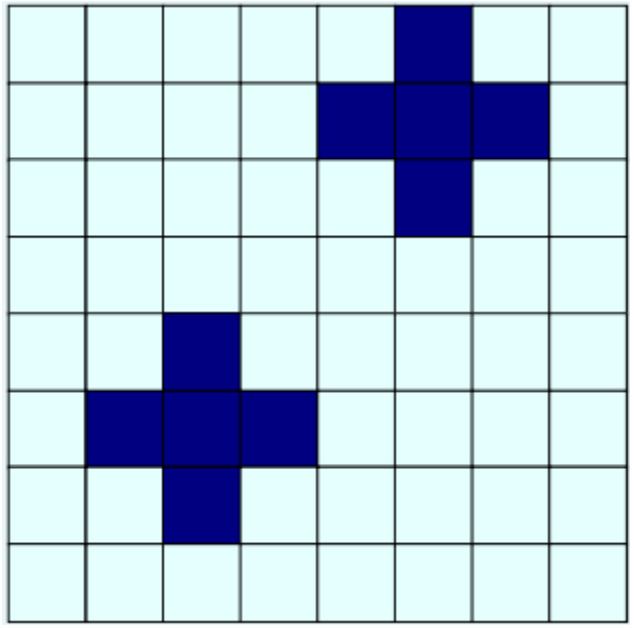


Structuring Element



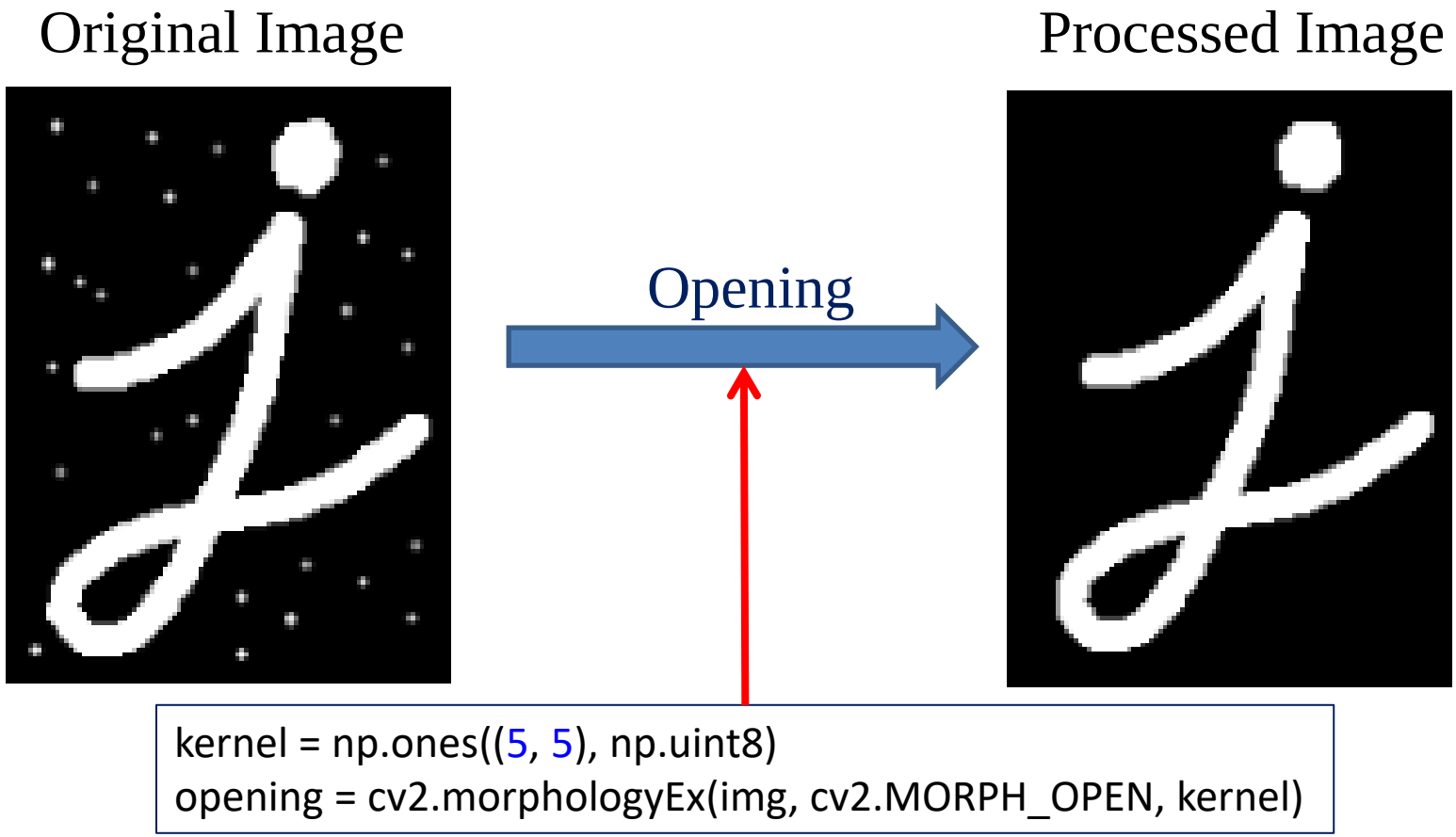
=

Processed Image



4.4 Các phép toán kết hợp

Phép toán mở (Opening)




```
kernel = np.ones((5, 5), np.uint8)
opening = cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)
```

4.4 Các phép toán kết hợp

Phép toán đóng (Closing)

- Phép đóng ảnh giúp làm trơn đường bao đối tượng, lấp đầy các khoảng trống trong biên, loại bỏ những hố nhỏ. Cụ thể như, có thể giúp khôi phục các nét đứt của ký tự trong ảnh chụp văn bản.
- Closing của một ảnh f bởi một phần tử cấu trúc s được ký hiệu và tính theo công thức sau:

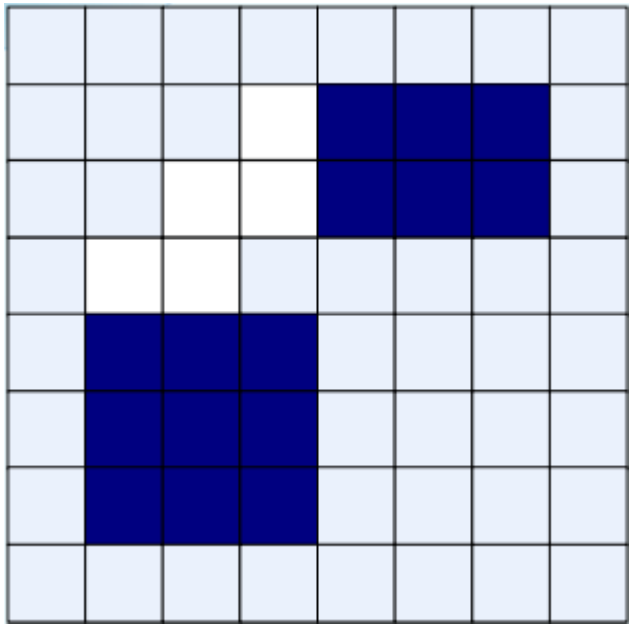
$$f \cdot s = (f \oplus s) \ominus s$$

- Closing ngược lại thực hiện phép Dilation trước sau đó thực hiện phép Erosion. Nó giúp loại bỏ nhiễu ở trong đối tượng thay vì ở ngoài như phép Opening.

4.4 Các phép toán kết hợp

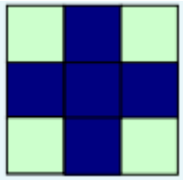
Phép toán đóng (Closing)

Original Image



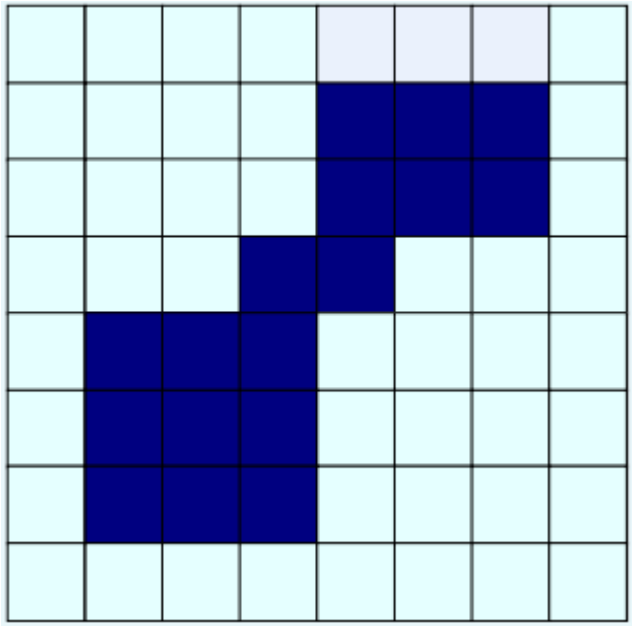
•

Structuring Element



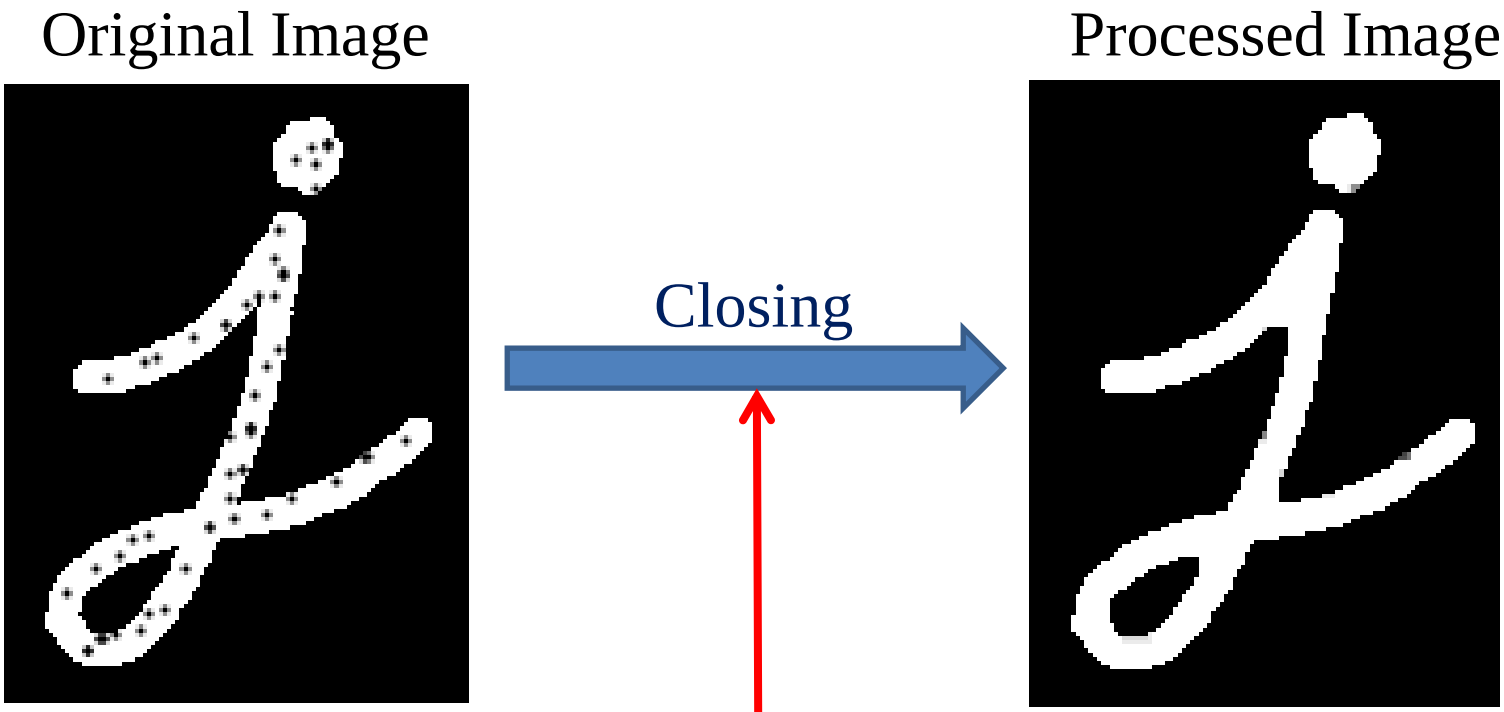
=

Processed Image



4.4 Các phép toán kết hợp

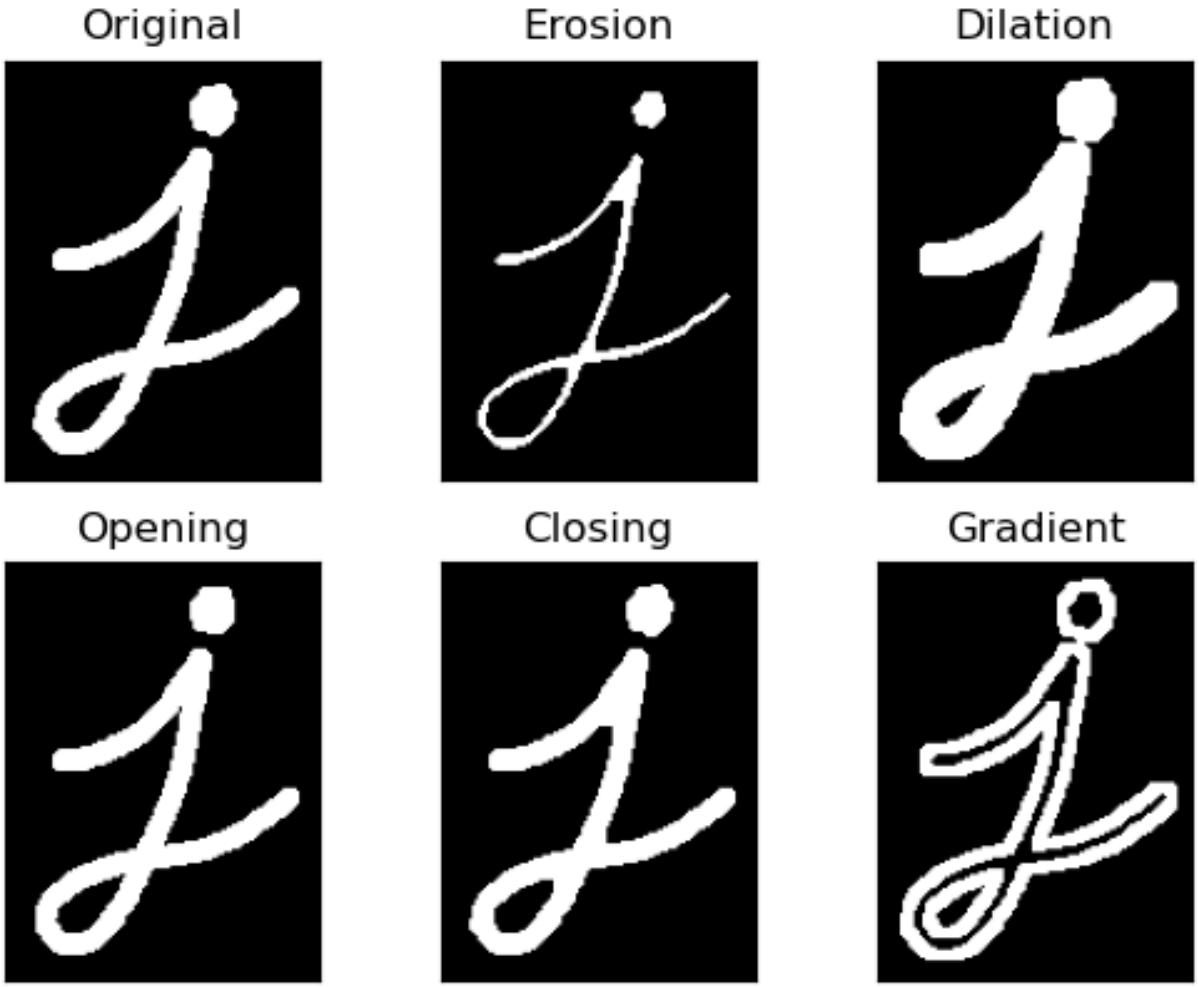
Phép toán đóng (Closing)




```
kernel = np.ones((5, 5), np.uint8)
closing = cv2.morphologyEx(img, cv2.MORPH_CLOSE, kernel)
```

4.4 Các phép toán kết hợp

So sánh các phương pháp xử lý hình thái



4.4 Các phép toán kết hợp

Xử lý hình thái trong OpenCV

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread('Chuj.png', 0)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

kernel = np.ones((5, 5), np.uint8)

erosion = cv2.erode(img, kernel, iterations=1)
dilation = cv2.dilate(img, kernel, iterations=1)
opening = cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)
closing = cv2.morphologyEx(img, cv2.MORPH_CLOSE, kernel)
gradient = cv2.morphologyEx(img, cv2.MORPH_GRADIENT, kernel)
.....
```

4.4 Các phép toán kết hợp

Xử lý hình thái trong OpenCV

```
.....
plt.figure()
plt.subplot(231), plt.imshow(img), plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(232), plt.imshow(erosion), plt.title('Erosion')
plt.xticks([]), plt.yticks([])
plt.subplot(233), plt.imshow(dilation), plt.title('Dilation')
plt.xticks([]), plt.yticks([])
plt.subplot(234), plt.imshow(opening), plt.title('Opening')
plt.xticks([]), plt.yticks([])
plt.subplot(235), plt.imshow(closing), plt.title('Closing')
plt.xticks([]), plt.yticks([])
plt.subplot(236), plt.imshow(gradient), plt.title('Gradient')
plt.xticks([]), plt.yticks([])
plt.show()
```

TỔNG HỢP MỘT SỐ HÀM QUAN TRỌNG

cv.morphologyEx(src, op, kernel)

Parameters

src	Source image. The number of channels can be arbitrary.
op	Type of a morphological operation, see **MorphTypes**
kernel	Structuring element. It can be created using **getStructuringElement**.
anchor	Anchor position with the kernel. Negative values mean that the anchor is at the kernel center.
iterations	Number of times erosion and dilation are applied.
borderType	Pixel extrapolation method, see **BorderTypes**. **BORDER_WRAP** is not supported.
borderValue	Border value in case of a constant border. The default value has a special meaning.

TỔNG HỢP MỘT SỐ HÀM QUAN TRỌNG

Morphology Types

MORPH_ERODE Python: cv.MORPH_ERODE	see **erode**
MORPH_DILATE Python: cv.MORPH_DILATE	see **dilate**
MORPH_OPEN Python: cv.MORPH_OPEN	an opening operation $$\texttt{dst} = \mathrm{open}(\texttt{src}, \texttt{element}) = \mathrm{dilate}(\mathrm{erode}(\texttt{src}, \texttt{element}))$$
MORPH_CLOSE Python: cv.MORPH_CLOSE	a closing operation $$\texttt{dst} = \mathrm{close}(\texttt{src}, \texttt{element}) = \mathrm{erode}(\mathrm{dilate}(\texttt{src}, \texttt{element}))$$
MORPH_GRADIENT Python: cv.MORPH_GRADIENT	a morphological gradient $$\texttt{dst} = \mathrm{morph\_grad}(\texttt{src}, \texttt{element}) = \mathrm{dilate}(\texttt{src}, \texttt{element}) - \mathrm{erode}(\texttt{src}, \texttt{element})$$

TỔNG HỢP MỘT SỐ HÀM QUAN TRỌNG

cv.getStructuringElement(shape, ksize[anchor])

Parameters

shape Element shape that could be one of **MorphShapes**

ksize Size of the structuring element.

anchor Anchor position within the element.
The default value $(-1, -1)$ means that the anchor is at the center.

Morphology Shapes

MORPH_RECT Python: cv.MORPH_RECT	a rectangular structuring element: $$E_{ij} = 1$$
MORPH_CROSS Python: cv.MORPH_CROSS	a cross-shaped structuring element: $$E_{ij} = \begin{cases} 1 & \texttt{if } i = \texttt{anchor.y } or j = \texttt{anchor.x} \\ 0 & \texttt{otherwise} \end{cases}$$
MORPH_ELLIPSE Python: cv.MORPH_ELLIPSE	an elliptic structuring element, that is, a filled ellipse inscribed into the rectangle Rect(0, 0, esize.width, 0.esize.height)

CASE STUDY 4.1: ĐIỀN ĐẦY LỖ BÊN TRONG

Mô tả: Ứng dụng Morphology để điền đầy khoảng trống cho ảnh nhị phân.

Yêu cầu:

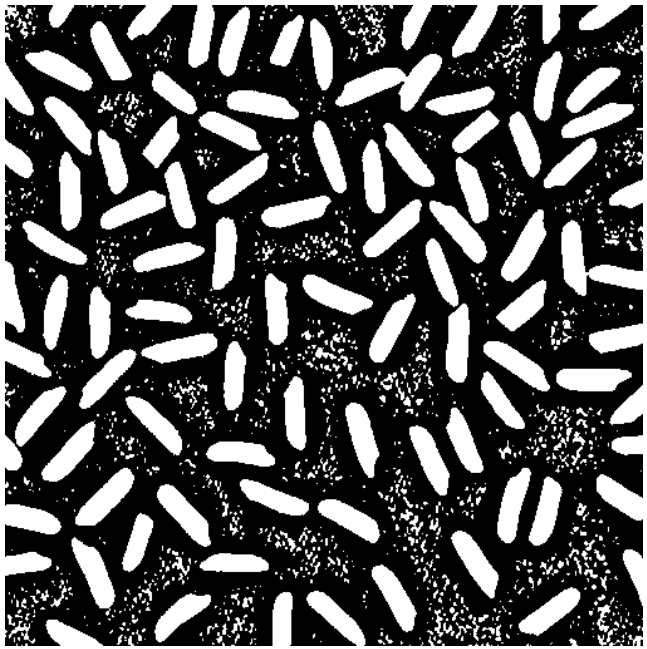
- Nhị phân hóa ảnh;
- Điền đầy lỗ trống bằng phép Morphology Close.

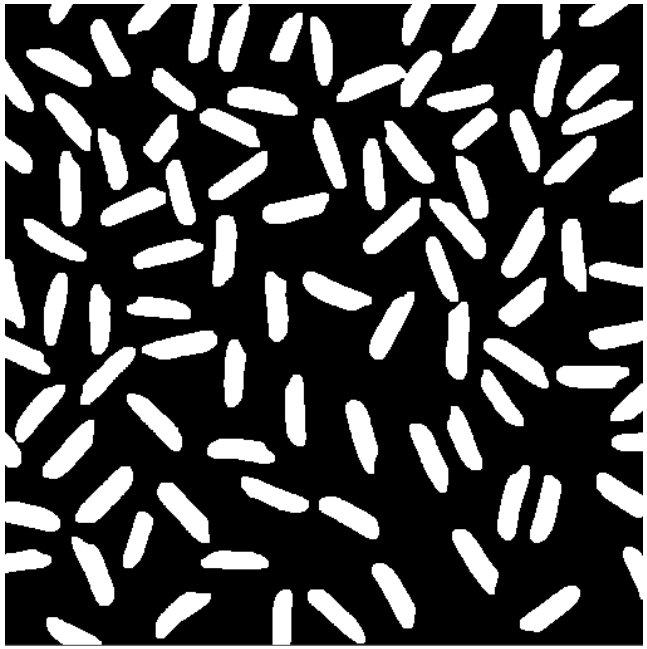
CASE STUDY 4.2: LỌC NHIỄU HẠT BÊN NGOÀI

Mô tả: Ứng dụng Morphology để lọc nhiễu hạt bên ngoài cho ảnh nhị phân.

Yêu cầu:

- Nhị phân hóa ảnh;
- Lọc nhiễu hạt bằng phép Morphology Open.





Thank you !!!