XỬ LÝ ẢNH SỐ

Chương 4. Xử lý ảnh hình thái

- 4.1 Thành phần cơ bản trong xử lý ảnh hình thái
- 4.2 Phép co và phép giãn
- 4.3 Phép đóng và phép mở
- 4.4 Biến đổi hit-or-miss
- 4.5 Một vài giải thuật hình thái cơ bản
- - Trích xuất hình bao
- - Làm đầy hố trong ảnh
- ....

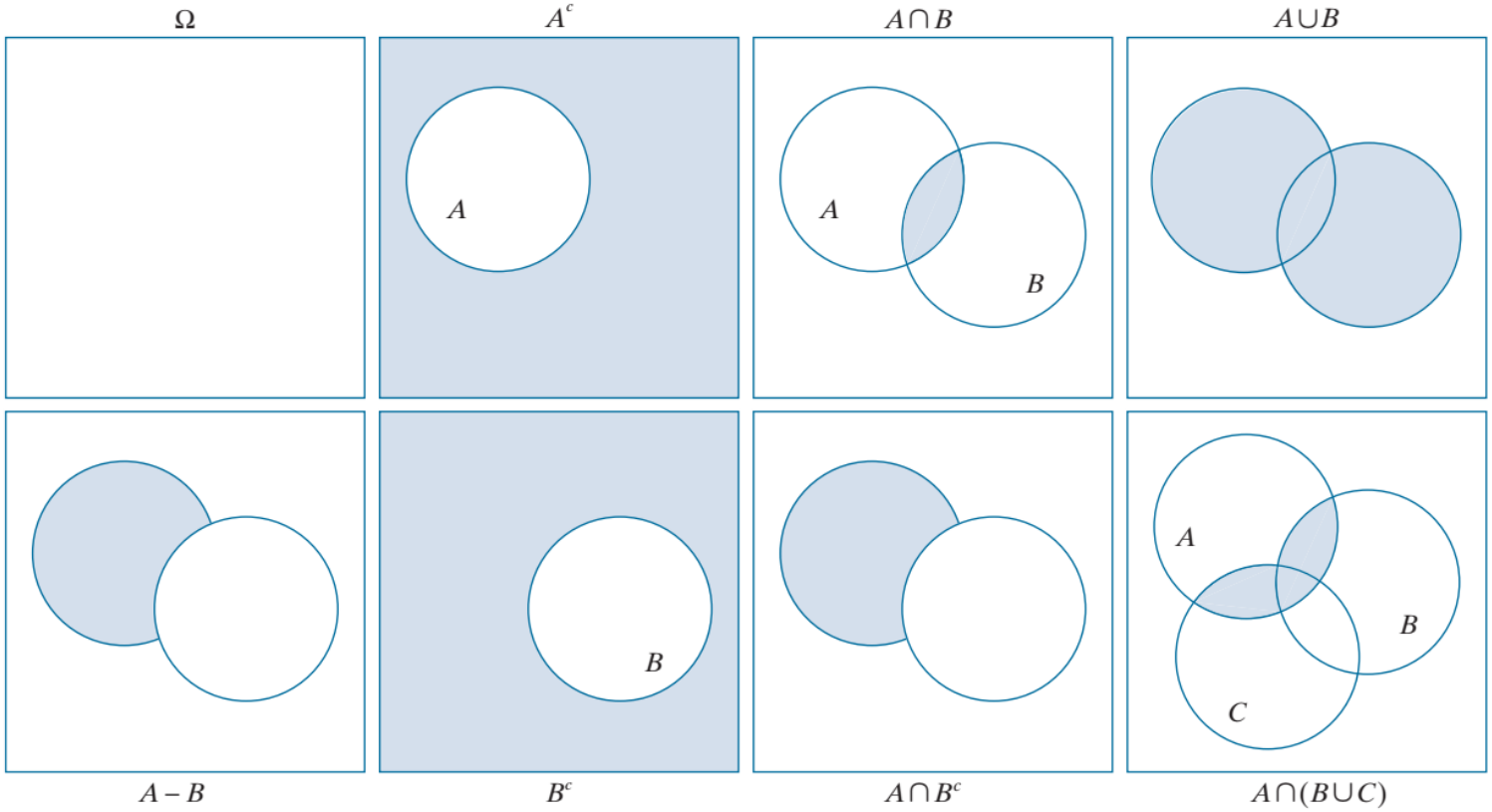
1

GIỚI THIỆU

Hình thái toán học

- "Hình thái – Morphology":
  - Thường để chỉ một nhánh sinh học liên quan đến hình dạng và cấu trúc của động vật và thực vật.
- "Hình thái toán học":
  - Sử dụng lý thuyết tập hợp
  - Trích xuất thành phần ảnh
  - Biểu diễn và mô tả hình dạng của vùng

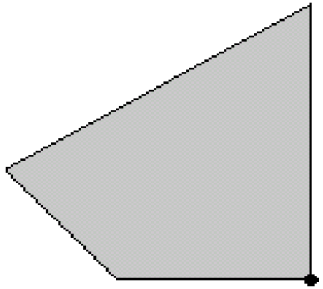
Lý thuyết tập hợp



Phản chiếu và tịnh tiến

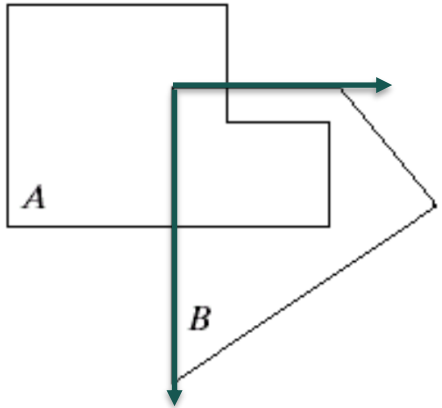
Phản chiếu

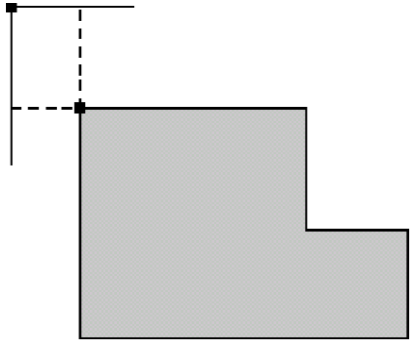
$$\hat{B} = \{w | w = -b, b \in B\}$$



Tịnh tiến

$$(A)_z = \{c | c = a + z, a \in A\}$$

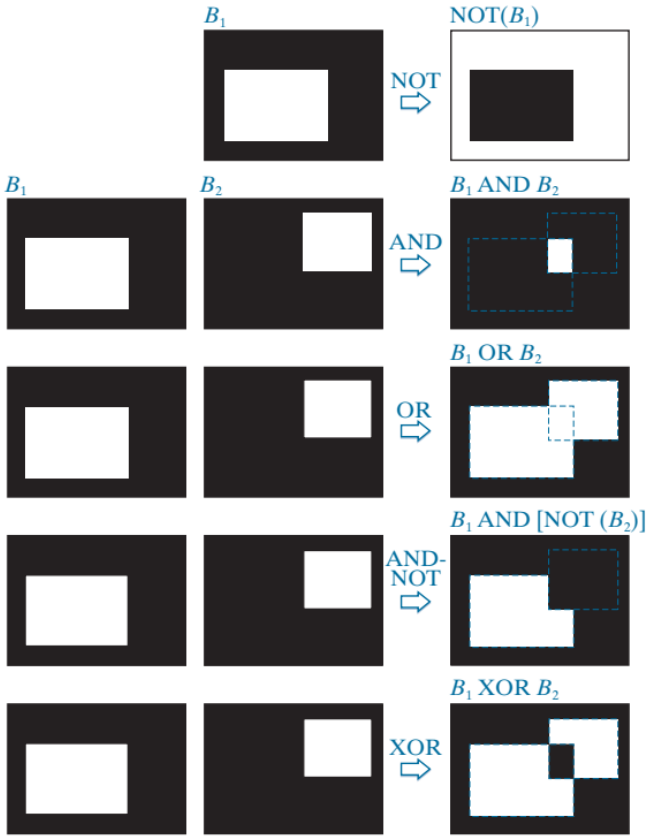




B là tập các điểm tọa độ (x,y) thì $\hat{B}$ chứa các điểm (-x,-y)

$$z = (z_1, z_2)$$

Phép toán logic



Phần tử cấu trúc – SE (Structuring element)

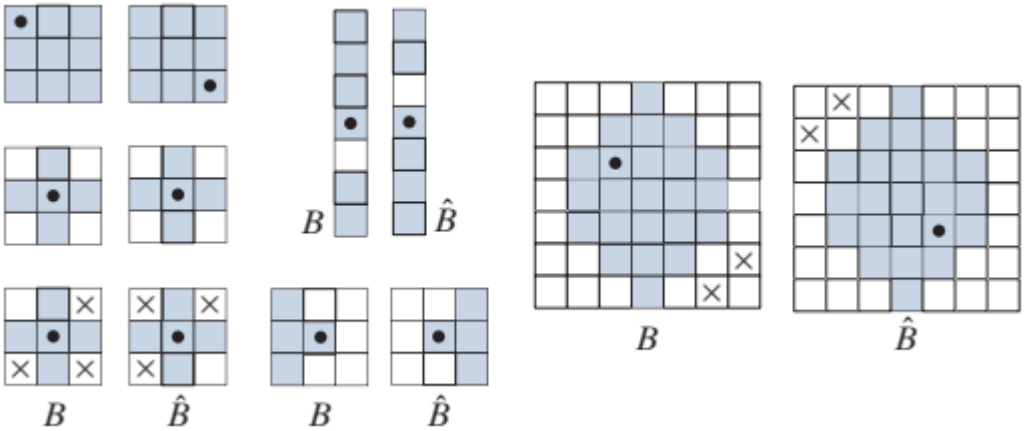
- Các SE - phần tử cấu trúc- là các mảng (ma trận) có cấu trúc được tạo bởi 2 mức xám 0 và 1.
- Thông thường SEs có dạng đối xứng với trọng tâm ở giữa.
- Đôi khi trong SE có thành phần “không quan tâm”, ký hiệu là ×

1	1	1
1	**1**	1
1	1	1

0	1	0
1	**1**	1
0	1	0

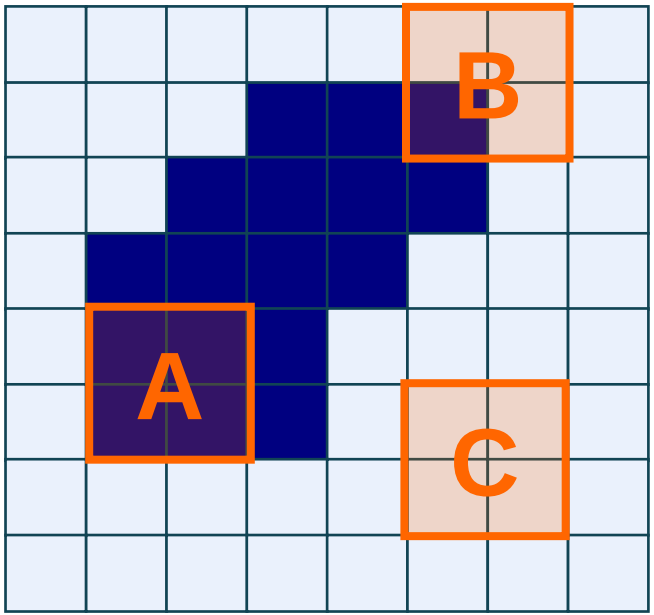
0	0	1	0	0
0	1	1	1	0
1	1	**1**	1	1
0	1	1	1	0
0	0	1	0	0

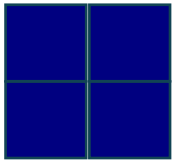
Phần tử cấu trúc – SE (Structuring element)



- Phần tử cấu trúc và phản chiếu của chúng qua tâm (× là thành phần không quan tâm và (.) là tâm. Phản chiếu là phép quay 180^0 của SE quanh tâm.

Hits & Fits





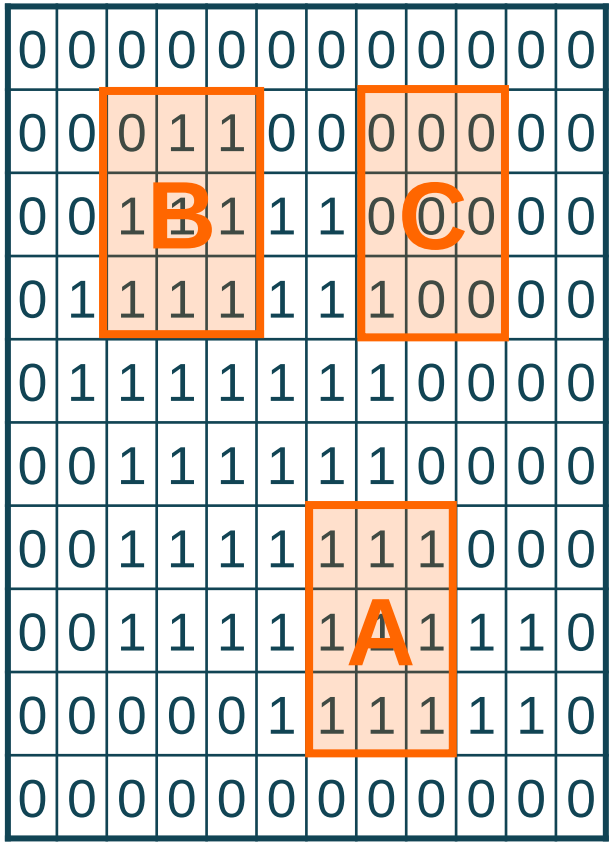
Structuring Element

Fit: Fit xảy ra khi tất cả pixel của SE trùng với các pixel của bức ảnh

Hit: Hit xảy ra khi bất kỳ pixel nào của SE trùng với pixel của ảnh

Tất cả các phép toán xử lý hình thái đều được dựa trên 2 hiện tượng trên!

Hitting & Fitting



1	1	1
1	1	1
1	1	1

Structuring Element 1

0	1	0
1	1	1
0	1	0

Structuring Element 2

2

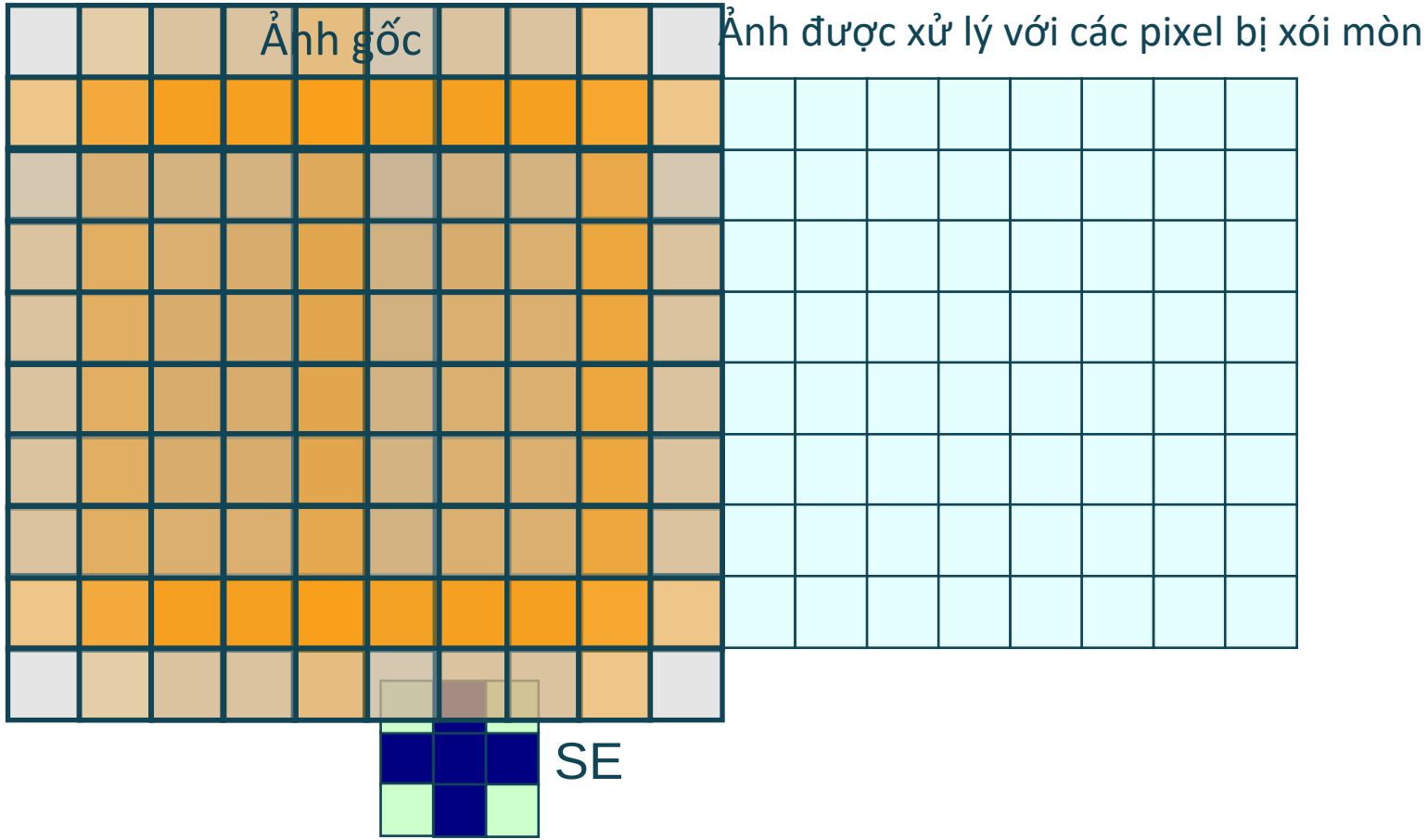
PHÉP CO VÀ PHÉP GIÃN

Các phép xử lý hình thái cơ bản

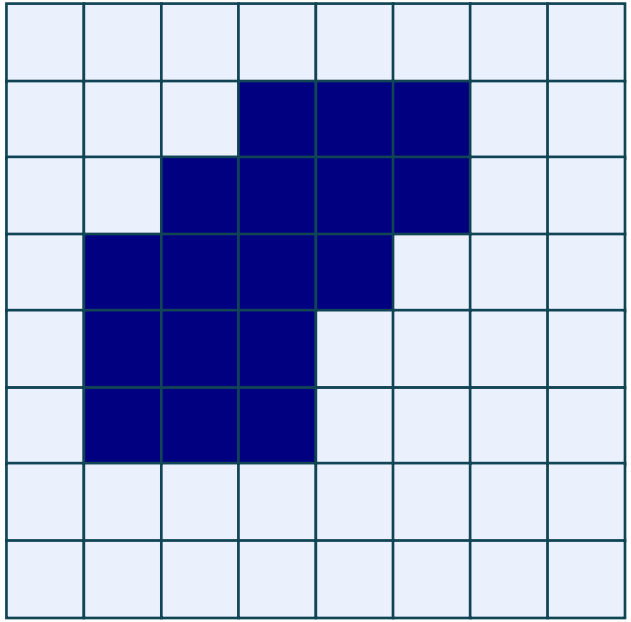
- Phép xử lý ảnh hình thái về cơ bản tương tự như phép lọc không gian.
- SE được dịch chuyển qua tất cả các pixel của ảnh gốc để tạo ra bức ảnh mới.
- Giá trị của các pixel mới phụ thuộc vào phép toán xử lý hình thái.
- Hai phép toán xử lý hình thái cơ bản là: co (erosion, shrink, reduce) và giãn (dilation, grow, expand)

Phép co nhị phân

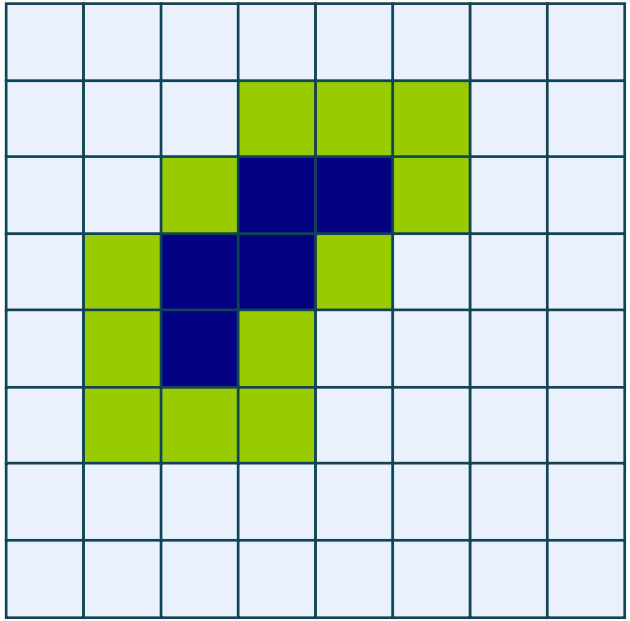
- Co tập A bởi phần tử cấu trúc B, ký hiệu là $A \ominus B$ được định nghĩa như sau:
- $A \ominus B = \{z|(B)_z \subseteq A\}$
- Giả sử SE B đang ở vị trí (x, y). Pixel mới sau khi thực hiện phép toán có giá trị như sau:
- $$g(x,y) = \begin{cases} 1 \ \ nếu\ B\ fit\ A \\ 0 \quad nếu\ khác \end{cases}$$

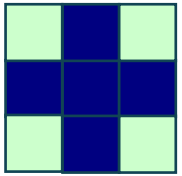

Ảnh gốc
Ảnh được xử lý với các pixel bị xói mòn
SE

Ảnh gốc



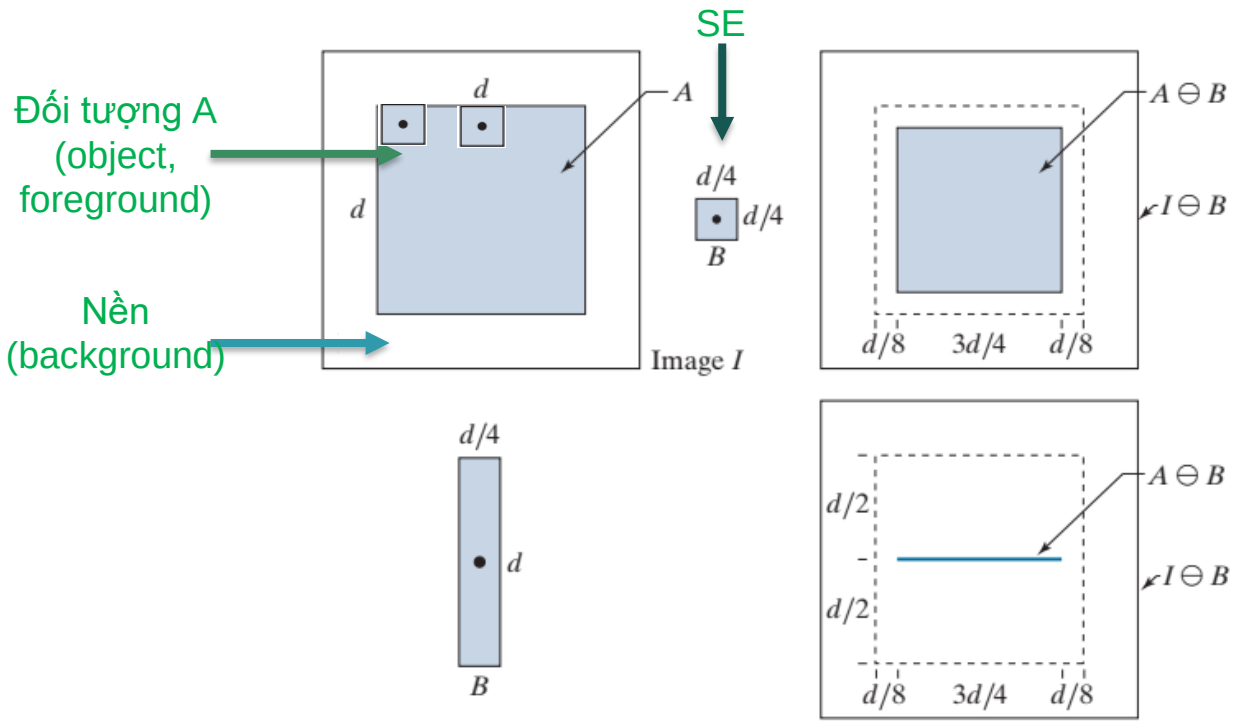
Ảnh sau xử lý





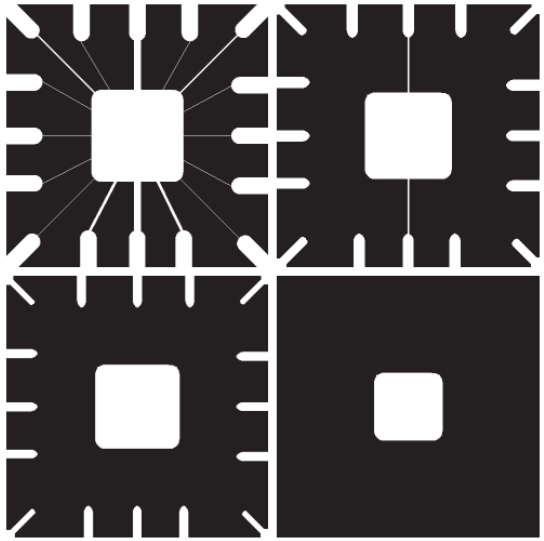
SE

Ví dụ phép co nhị phân



Ví dụ phép co nhị phân

Ảnh nhị phân
486 x 486



SE 11 x 11

SE có tất cả là
giá trị 1

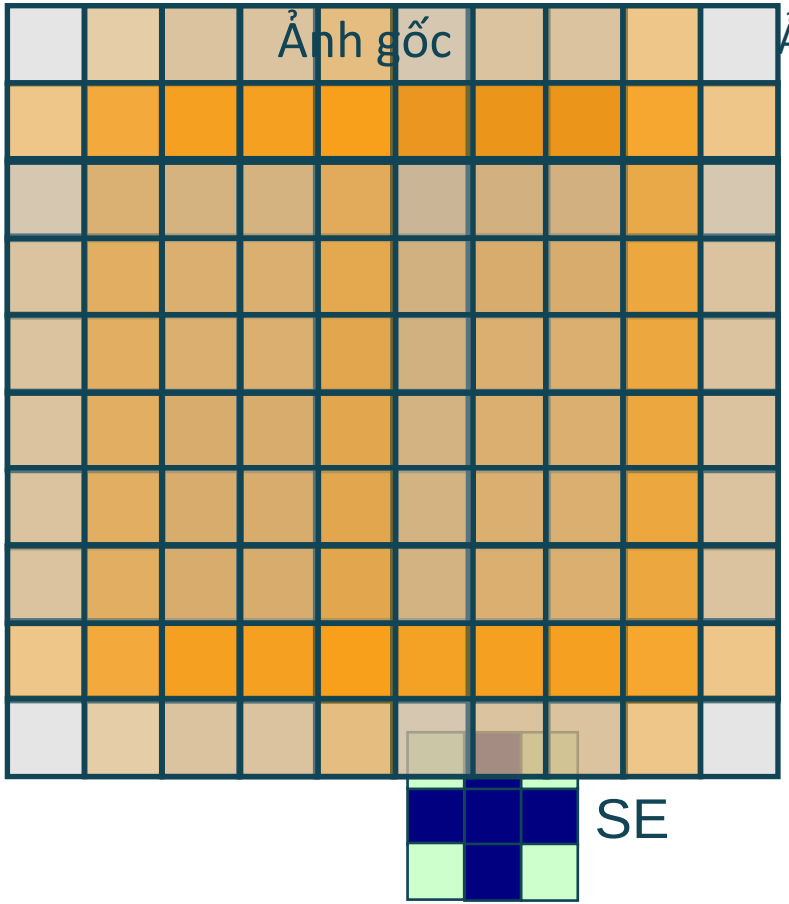
SE 15 x 15

SE 45 x 45

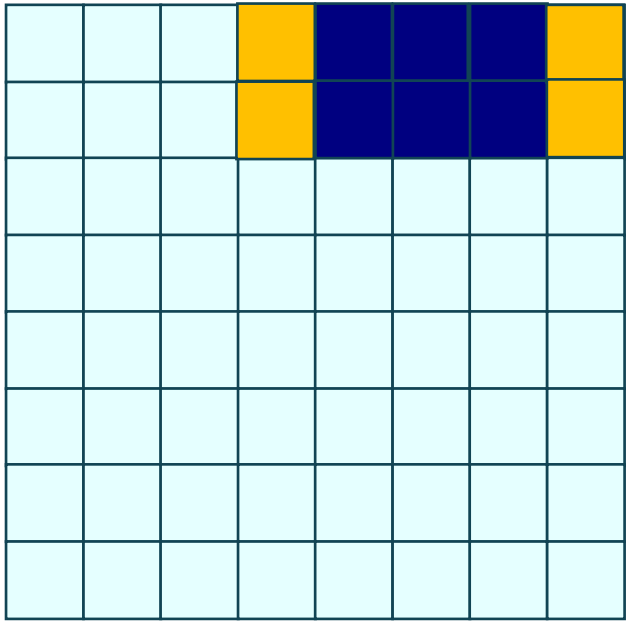
Phép giãn nhị phân

- Giãn tập A bởi phần tử cấu trúc B, ký hiệu là $A \oplus B$ được định nghĩa như sau:
- $A \oplus B = \left\{ z | \left(\hat{B} \right)_z \cap A \neq \emptyset \right\}$
- Giả sử SE B đang ở vị trí (x, y). Pixel mới sau khi thực hiện phép toán có giá trị như sau:
- $$g(x,y) = \begin{cases} 1 & nếu\ B\ hit\ A \\ 0 & nếu\ khác \end{cases}$$

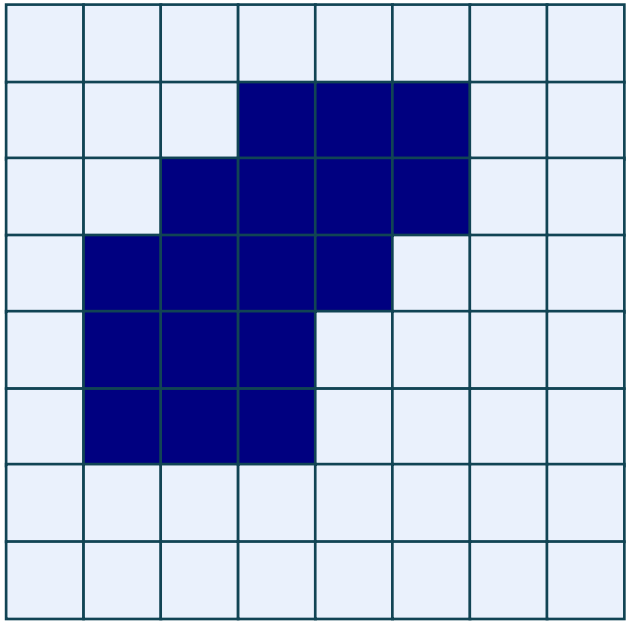
Ảnh gốc



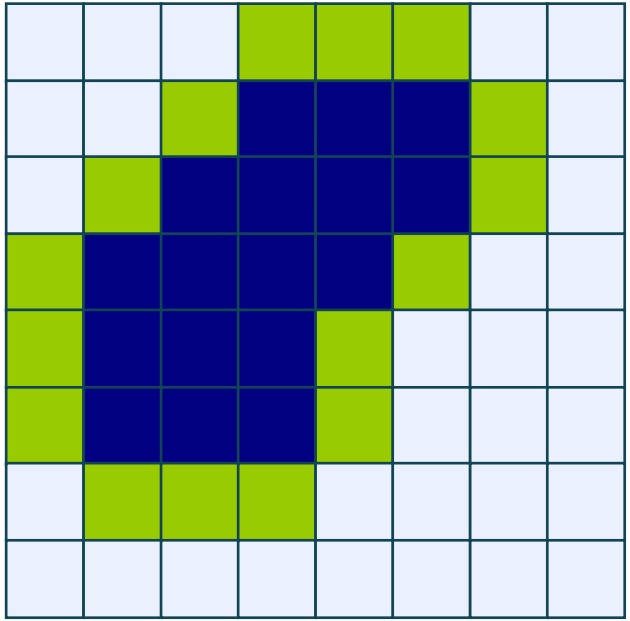
Ảnh được xử lý với các pixel được giãn thêm

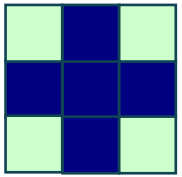


Ảnh gốc



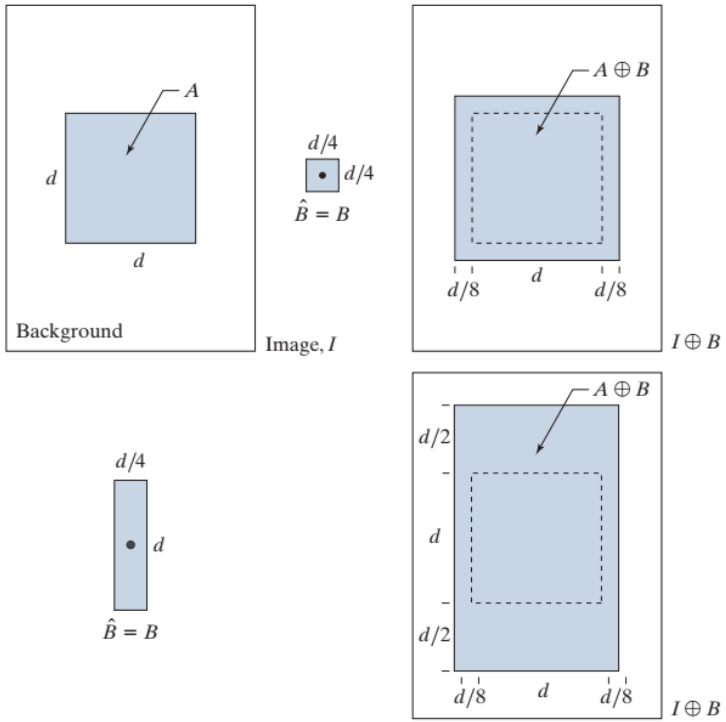
Ảnh được xử lý với các pixel được giãn thêm

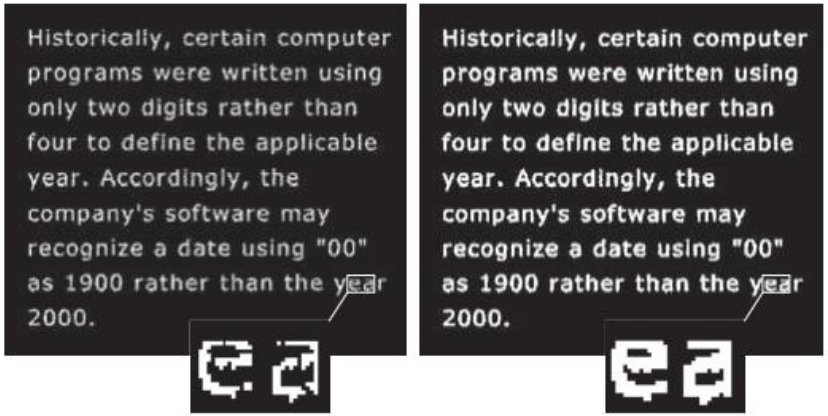




SE

Ví dụ phép giãn nhị phân



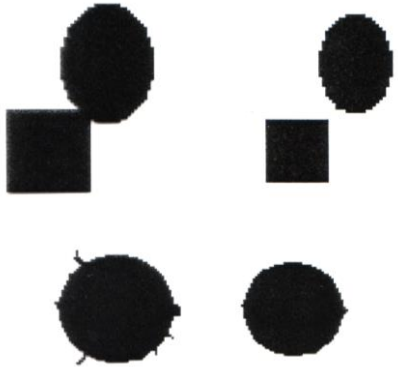


1	1	1
1	1	1
1	1	1

Ứng dụng của phép co và phép giãn

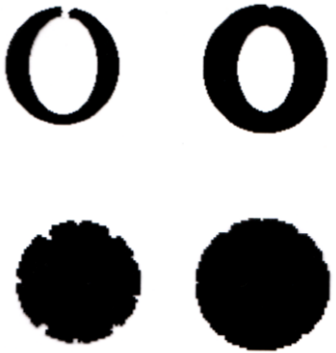
Phép co

- Thu nhỏ kích thước đối tượng.
- Loại bỏ chi tiết không liên quan, cắt bớt phần dư thừa.
- Tách rời các đối tượng liền nhau



Phép giãn:

- Tăng kích thước đối tượng.
- Bổ sung phần thiếu của đối tượng
- Sửa những đối tượng bị đứt gãy



3

Phép đóng và phép mở

Kết hợp phép co và phép giãn

- Phép giãn: mở rộng đối tượng với phần tử cấu trúc.
- Phép co: Thu nhỏ đối tượng
- Mong muốn:
  - Loại bỏ cấu trúc hoặc điền đầy lỗ trống.
  - Không làm thay đổi các phần còn lại.
- Giải pháp:
  - Kết hợp phép co và phép giãn.

Liệu phép co + phép giãn = ảnh ban đầu???

Phép mở ảnh – Phép đóng ảnh

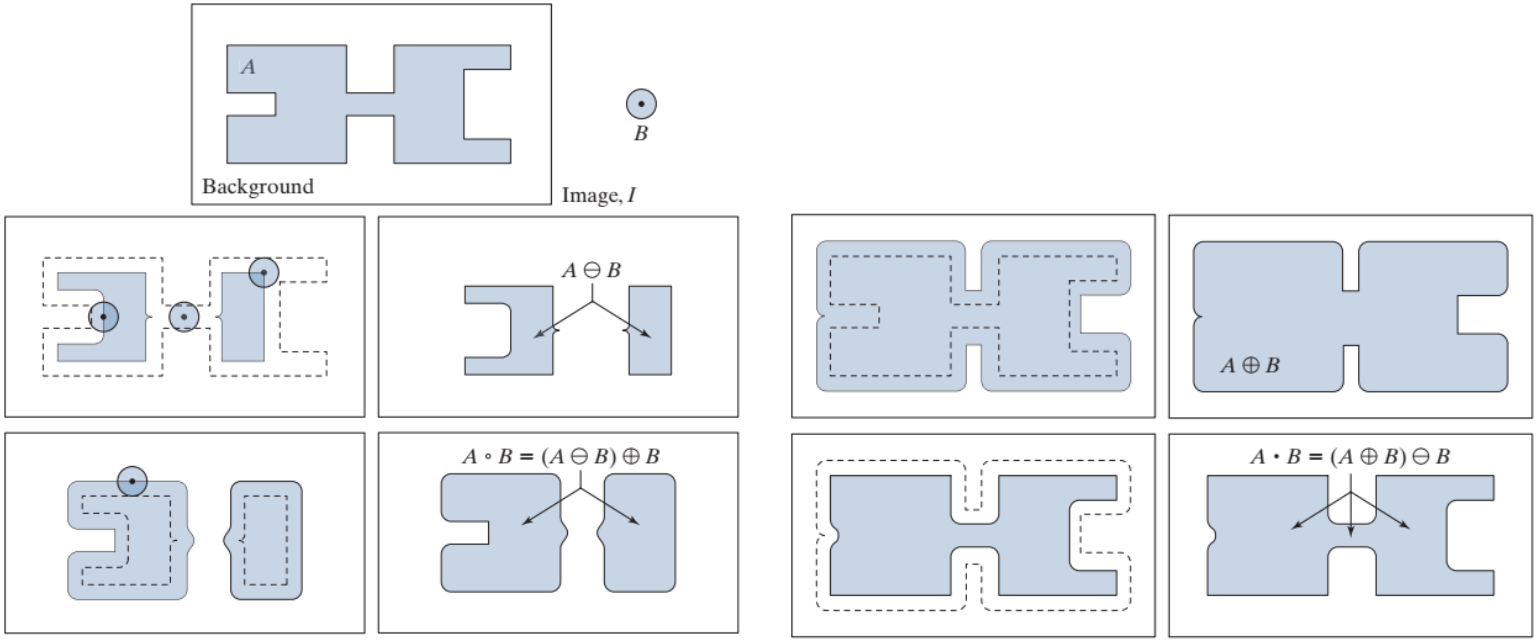
- Phép mở tập A bởi SE B được ký hiệu là $A \circ B$, được định nghĩa như sau:

$$A \circ B = (A \ominus B) \oplus B$$

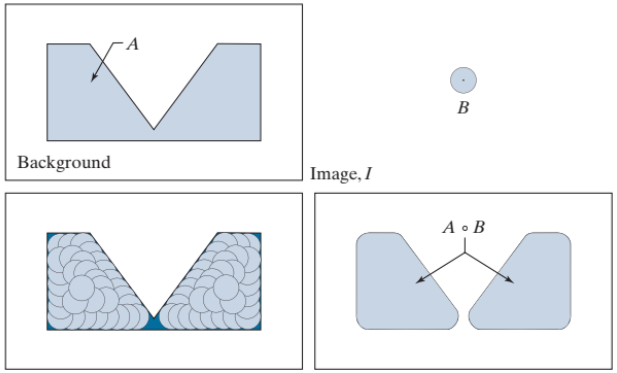
- Phép đóng tập A bởi SE B được ký hiệu là $A \cdot B$, được định nghĩa như sau:

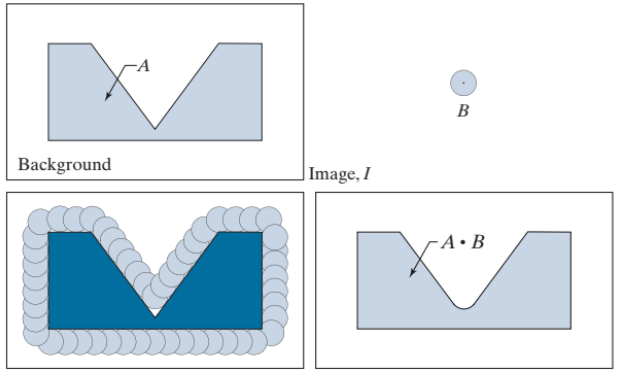
$$A \cdot B = (A \oplus B) \ominus B$$

Phép mở ảnh – Phép đóng ảnh



Phép mở ảnh – Phép đóng ảnh





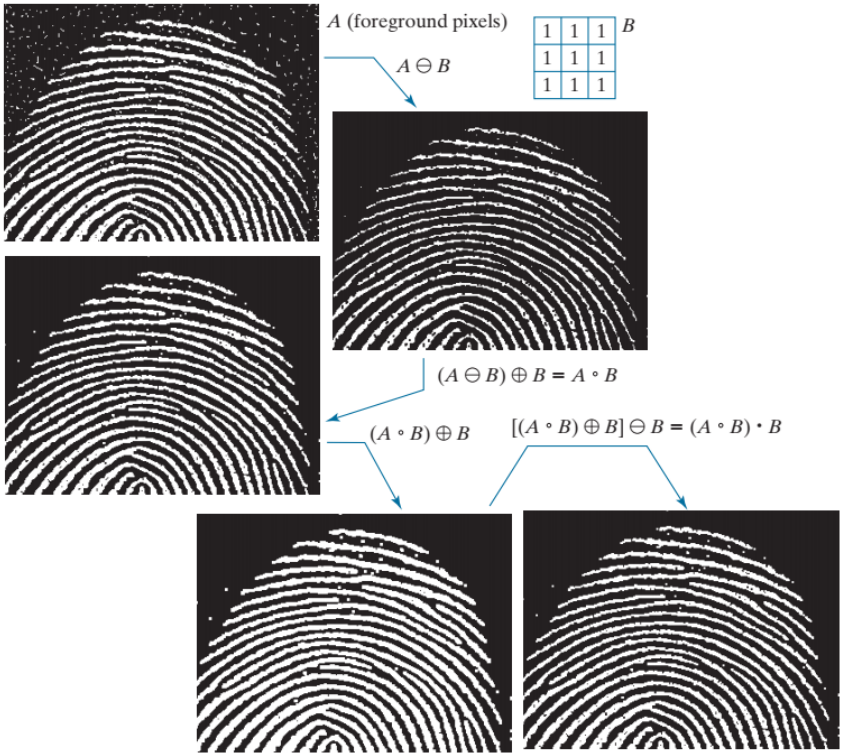
Phép mở

- Làm mượt đường viền của đối tượng.
- Phá bỏ các vùng eo hẹp, loại bỏ phần dư thừa.

Phép đóng

- Làm mượt đường viền của đối tượng.
- Nối liền các vùng eo hẹp, các rãnh dài mỏng, loại bỏ lỗ nhỏ và lấp đầy khoảng trống trên đường viền.

Phép mở ảnh – Phép đóng ảnh



$$X = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \text{ với } B = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$X = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \text{ với } B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

4

Phép biến đổi hit or miss (HMT)

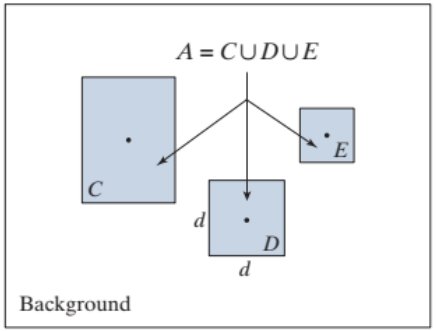
Hit-or-miss transform (HMT)

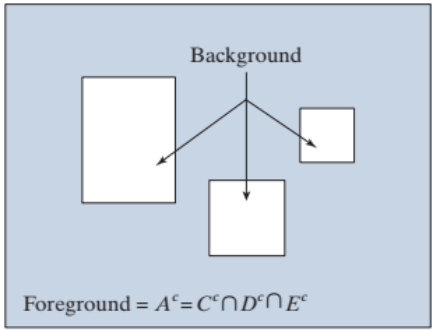
- HMT là công cụ cơ bản để phát hiện đối tượng.
- I là một hình ảnh nhị phân bao gồm các pixel foreground (A) và background tương ứng.
- HMT sử dụng hai phần tử cấu trúc: B_1, để phát hiện các hình dạng ở foreground và B_2, để phát hiện các hình dạng ở background.
- HMT của hình ảnh ***I*** được định nghĩa là

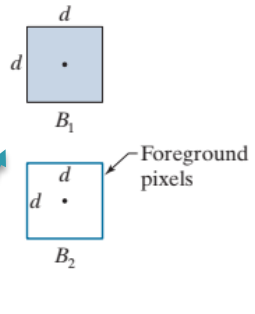
$$I \circledast B_{1,2} = \{z|(B_1)_z \subseteq A \; and \; (B_2)_z \subseteq A^c\}$$

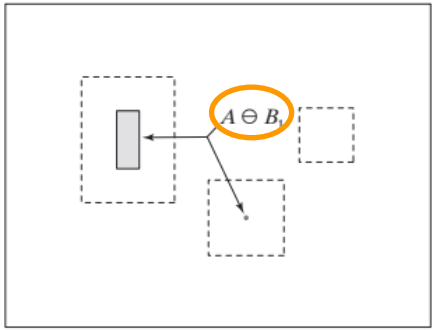
$$= (A \ominus B_1) \cap (A^c \ominus B_2)$$

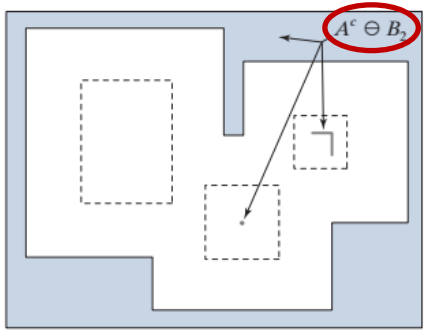
Ví dụ: tìm vị trí của điểm gốc của đối tượng (tập hợp) D trong hình I.

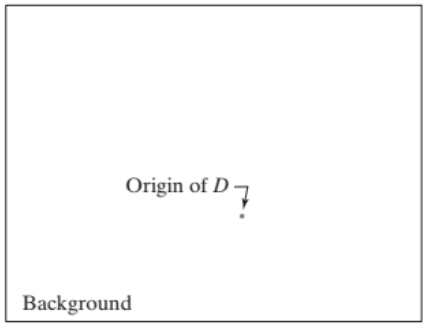












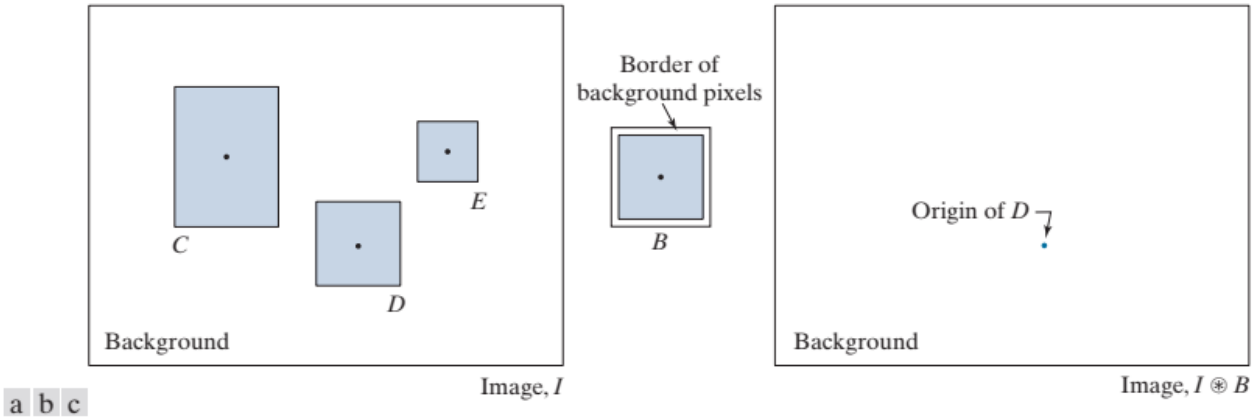
Hai phần tử cấu trúc

$$I \circledast B_{1,2} = \{z | (B_1)_z \subseteq A \; and \; (B_2)_z \subseteq A^c\}$$
$$= (A \ominus B_1) \cap (A^c \ominus B_2)$$

Một cách giải khác...

– Tại sao không cố gắng phát hiện D trực tiếp trong hình ảnh I chỉ sử dụng một phần tử cấu trúc duy nhất?

- Câu trả lời là hoàn toàn có thể làm được, nhưng không phải trong bối cảnh phép co "truyền thống".
- HMT được viết lại là: $I \circledast B = \{z|(B)_z \subseteq I\}$

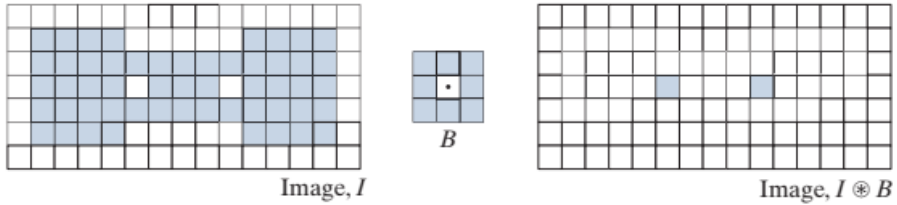


a b c

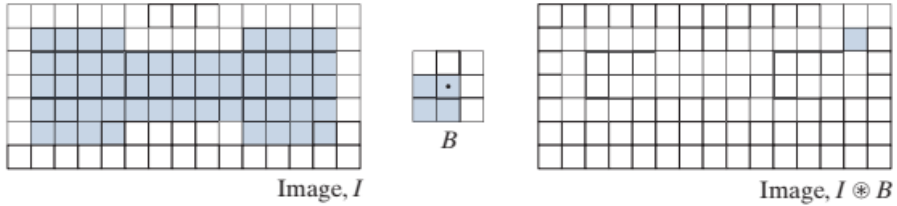
Một số ví dụ khác

– HMT được viết lại là: $I \circledast B = \{z|(B)_z \subseteq I\}$

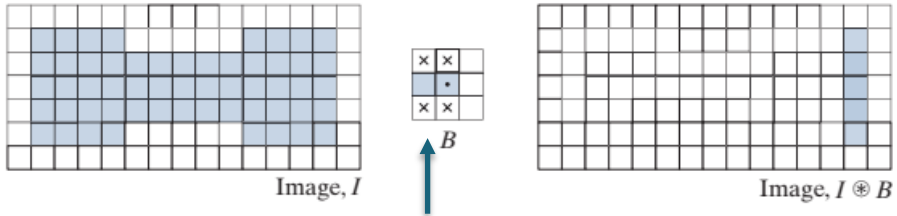
Phát hiện lỗ có độ rộng 1 pixel



Phát hiện góc trên bên phải



Phát hiện nhiều đặc điểm



× -Phần tử "không quan tâm" (don't care)

5

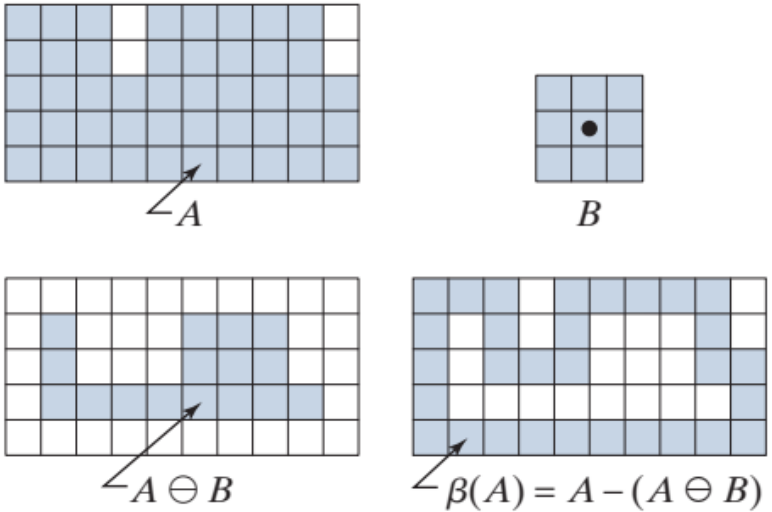
MỘT SỐ THUẬT TOÁN XỬ LÝ HÌNH THÁI CƠ BẢN

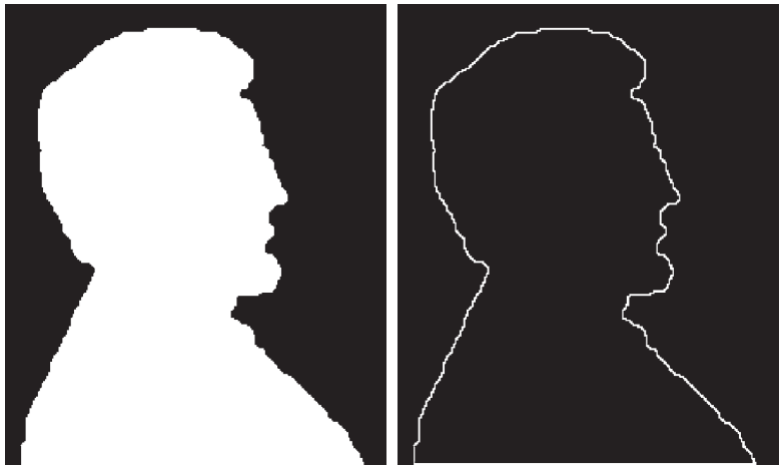
Giới thiệu

- Khi xử lý hình ảnh nhị phân, một trong những ứng dụng chính của hình thái học là trích xuất các thành phần hình ảnh hữu ích trong việc biểu diễn và mô tả hình dạng.
- Cụ thể, chúng ta xem xét các thuật toán hình thái để tách biên, các thành phần kết nối, vỏ lồi của một vùng.
- Chúng ta cũng phát triển một số phương pháp (làm đầy vùng, làm mỏng, làm dày) được sử dụng thường xuyên cho việc tiền xử lý và hậu xử lý.
- Những hình ảnh nhị phân này được hiển thị bằng đồ họa với foreground (các giá trị 1) được tô xám và background (0) màu trắng.

Tách biên

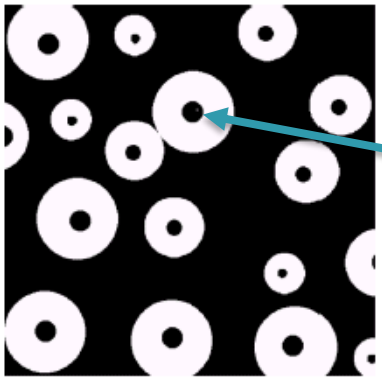
- Biên của một tập hợp A gồm các pixel foreground, ký hiệu là β(A), có thể nhận được bằng cách như sau: trước tiên là co **A** bởi một phần tử có cấu trúc phù hợp **B**, và sau đó tìm tập hợp là sự sai khác giữa **A** và co của **A**.
- $\boldsymbol{\beta(A) = A - (A \ominus B)}$





Làm đầy vùng (Hole filling)

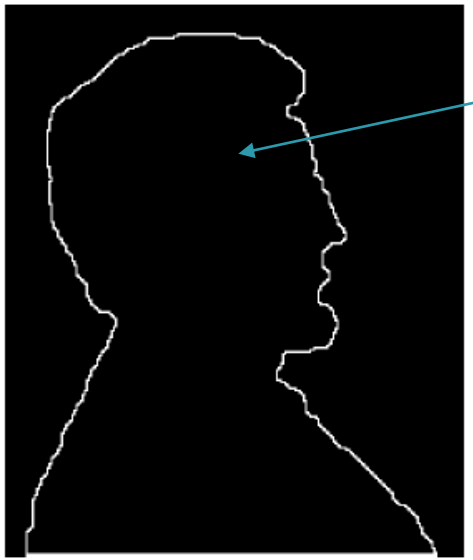
- Một lỗ có thể được định nghĩa là một vùng background được bao quanh bởi một đường viền liên kết của các pixel foreground.
- Mục tiêu là lấp đầy tất cả các lỗ bằng pixel foreground.
- Nói cách khác, thuật toán làm đầy vùng là quá trình tìm tất cả các pixel bên trong của một vùng background và gán các pixel tìm được giá trị bằng 1.



Cho 1 điểm bên trong hình tròn, liệu có thể tô toàn bộ hình tròn này không?

Làm đầy vùng

- Thuật toán : Xác định 1 điểm bên trong vùng, sau đó thực hiện liên tục các phép giãn hình.



$$X_0 = p$$

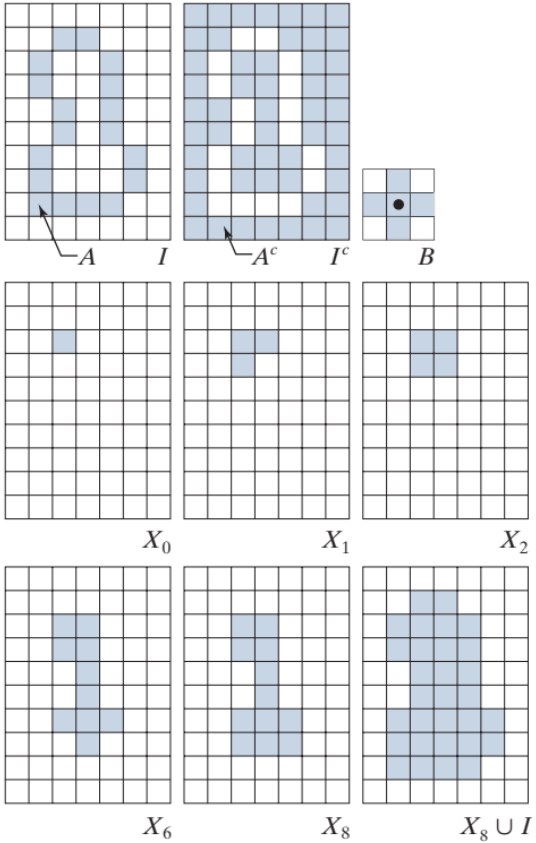
$$X_k = (X_{k-1} \oplus B) \cap A^c, k = 1,2,3,...$$

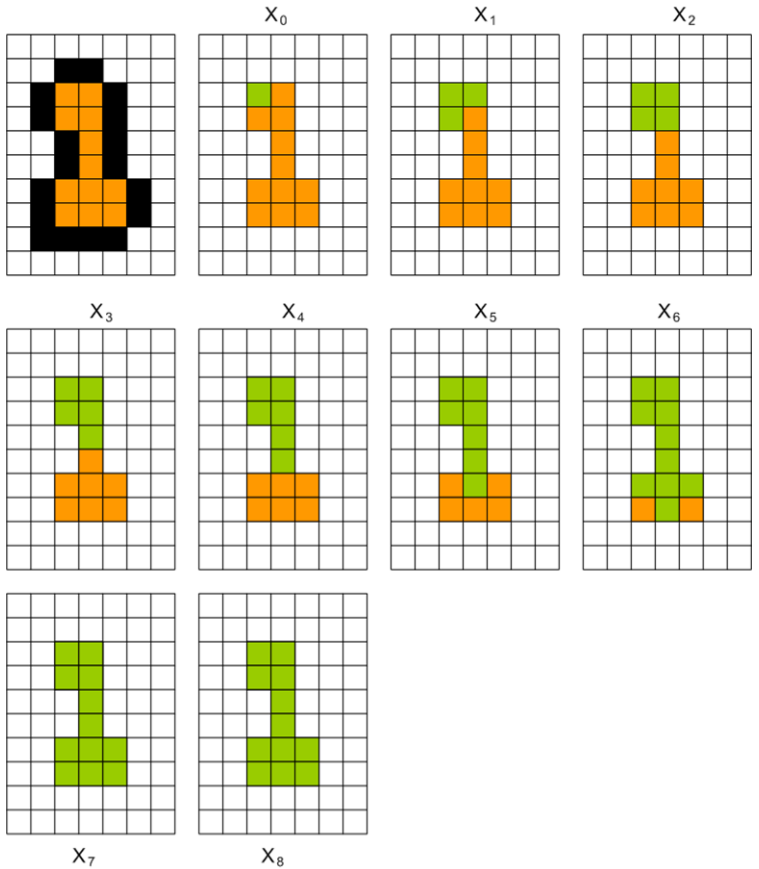
until $X_k = X_{k-1}$

Ràng buộc sự phát triển vùng

Ví dụ làm đầy vùng

$$X_k = (X_{k-1} \oplus B) \cap A^c, k = 1,2,3,...$$

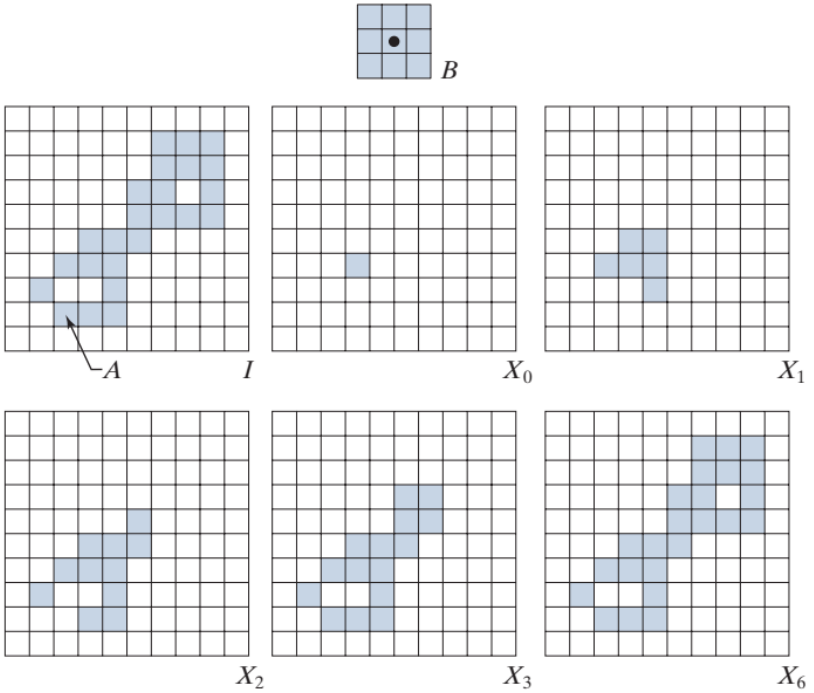




Tách các thành phần kết nối

- Ý tưởng: bắt đầu từ một điểm trong thành phần kết nối và thực hiện phép giãn nhiều lần.
- $X_0 = p$
- $X_k = (X_{k-1} \oplus B) \cap A, k = 1,2,3, \ldots$

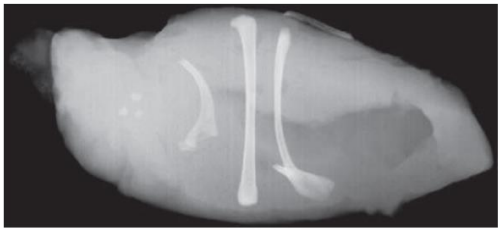
  Cho đến khi $X_k = X_{k-1}$



Ví dụ: Sử dụng các thành phần kết nối để phát hiện các vật thể lạ trong thực phẩm đóng gói.

- Các thành phần kết nối thường được sử dụng để kiểm tra tự động: Ví dụ phát hiện những vật thể lạ trong thực phẩm chế biến trước khi vận chuyển.

Hình ảnh tia X của một ức gà có chứa các mảnh xương →



Ảnh sau phân ngưỡng →

Sau khi co ảnh bằng SE 5 × 5 các giá trị 1 →

Thực hiện tách thành phần kết nối

Connected component	No. of pixels in connected comp
01	11
02	9
03	9
04	39
05	133
06	1
07	1
08	743
09	7
10	11
11	11
12	9
13	9
14	674
15	85

CONVEX HULL

- Tập hợp S các điểm trong mặt phẳng Euclide được cho là lồi khi và chỉ khi một đoạn thẳng nối hai điểm bất kỳ trong S nằm hoàn toàn trong S.
- Vỏ lồi (convex hull) H của S là tập lồi nhỏ nhất chứa S.
- Mở rộng: Một tập hợp kỹ thuật số (digital set), A, được cho là lồi khi và chỉ khi vỏ lồi Euclide của nó chỉ chứa các điểm kỹ thuật số thuộc A

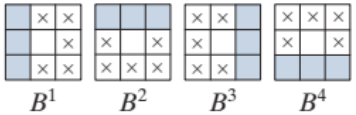
CONVEX HULL

- Gọi B^i, $i = 1,2,3,4$ mô tả bốn phần tử cấu trúc.
- Quy trình bao gồm thực hiện phương trình hình thái học:
- $\boldsymbol{X_k^i = (X_{k-1}^i \circledast B) \cup X_{k-1}^i \quad i = 1,2,3,4}$ **và** $\boldsymbol{k = 1,2,3}$

  với $\boldsymbol{X_k^i = \mathrm{I}}$
- Khi quá trình hội tụ với phần tử cấu trúc thứ i (tức là $X_k^i = X_{k-1}^i$) ta gán $\boldsymbol{D^i = X_k^i}$.
- Vỏ lồi của A là hợp của 4 kết quả:
- $\boldsymbol{C(A) = \cup_{i=1}^{4} D^i}$

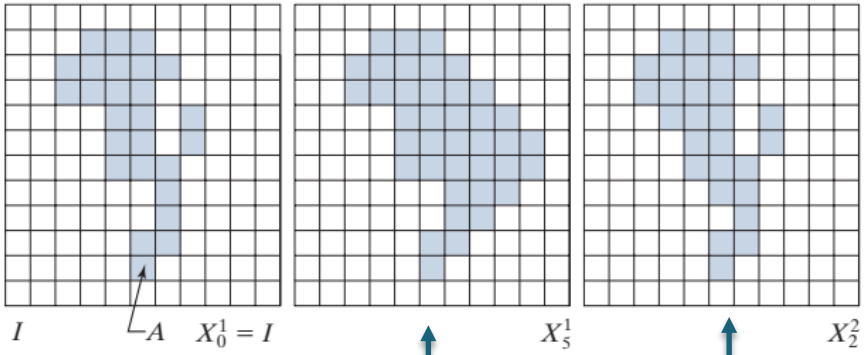
CONVEX HULL

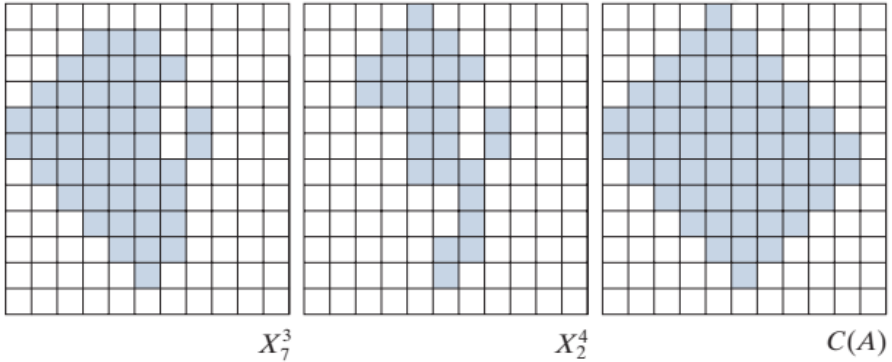
Convex hull (vỏ lồi)



Các SE

$$X_k^i = (X_{k-1}^i \circledast B) \cup X_{k-1}^i$$

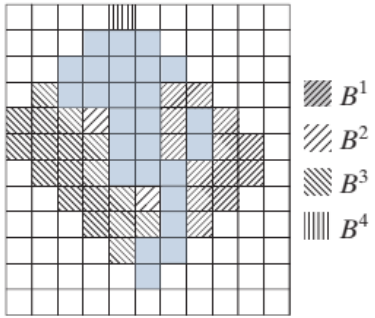




5 lần thực hiện B^1

2 lần thực hiện B^2

Sự đóng góp của mỗi phần tử cấu trúc



LÀM MẢNH

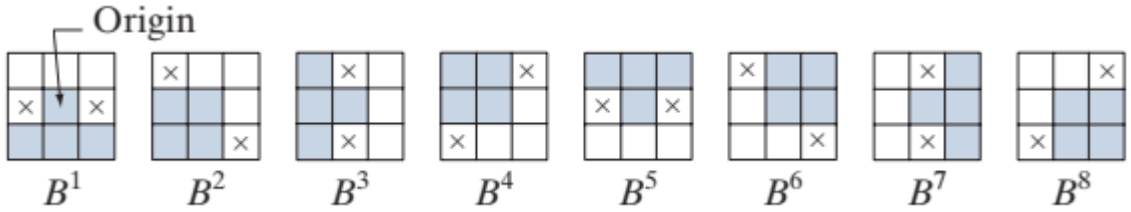
- Làm mảnh tập hợp A gồm các pixel foreground bởi một phần tử có cấu trúc B, được ký hiệu là A⊗B, có thể được định nghĩa theo biến đổi hit-or-miss:

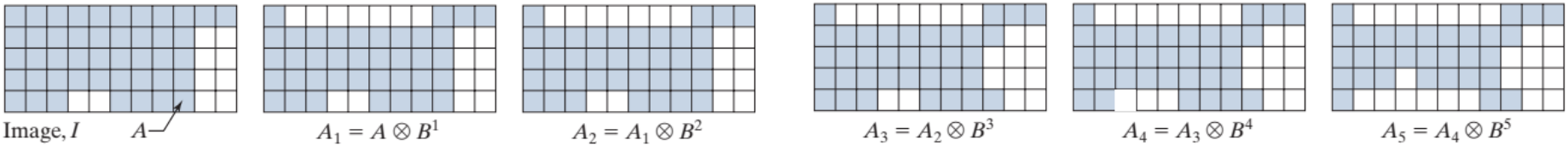
$$A \otimes B = A - (A \circledast B) = A \cap (A \circledast B)^c$$

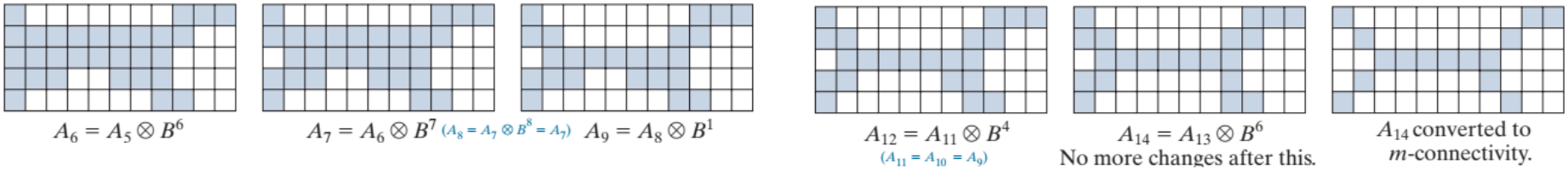
- Một biểu thức hữu ích hơn để làm mỏng A một cách đối xứng dựa trên một chuỗi các phần tử cấu trúc: $B = \{B^1, B^2, \dots, B^n\}$
- Làm mảnh bằng một chuỗi các phần tử cấu trúc:

$$A \otimes \{B\} = ((\dots((A \otimes B^1) \otimes B^2)\dots) \otimes B^n)$$

LÀM MẢNH (ví dụ)







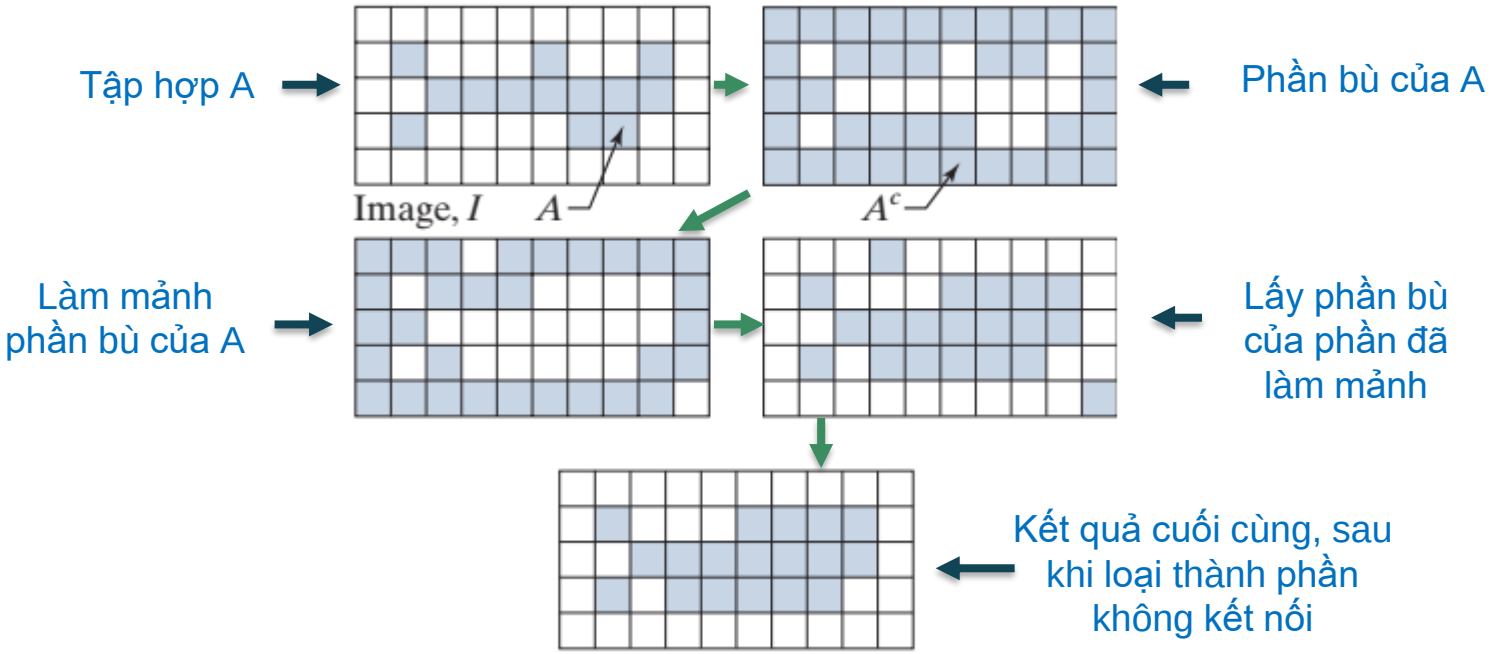
LÀM DÀY

- Làm dày là hình thái đối ngẫu của làm mảnh và được xác định bằng biểu thức: $A \odot B = A \cup (A \circledast B)$
- B là phần tử cấu trúc để làm dày.
- Tương tự như làm mảnh, làm dày có thể được định nghĩa là một tập các phép toán tuần tự:

$$A \odot \{B\} = \left(\left(\ldots \left(\left(A \odot B^1 \right) \odot B^2 \right) \ldots \right) \odot B^n \right)$$

- Các phần tử cấu trúc làm dày có dạng giống như slide trước nhưng giá trị 1 và 0 được hoán đổi cho nhau.
- Thực tế**, thường làm mảnh vùng background và lấy phần bù của kết quả đó**.

LÀM DÀY (ví dụ)


Tập hợp A
Image, I
A
A^c
Phần bù của A
Làm mảnh phần bù của A
Lấy phần bù của phần đã làm mảnh
Kết quả cuối cùng, sau khi loại thành phần không kết nối