EIE4512 - Digital Image Processing
Morphological Filters

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Zhen Li
lizhen@cuhk.edu.cn
School of Science and Engineering
The Chinese University of Hong Kong, Shen Zhen
March 21, 2019

Mathematical Morphology

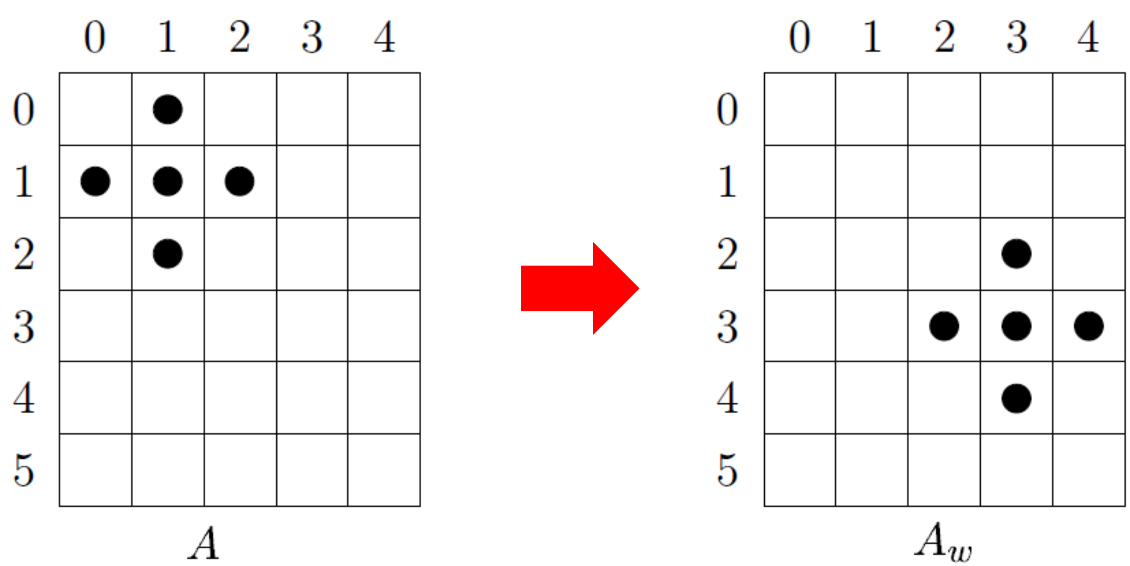
- Originally operated on Binary (black and white) images
- Binary images?
  - Faxes, digitally printed images
  - Obtained from thresholding grayscale images
- Morphological filters alter local structures in an image

Translation

- *A* is set of pixels in binary image
- *w* = (*x*,*y*) is a particular coordinate point
- *A* is set *A* “translated” in direction (*x*,*y*). i.e

$$A_x = \{(a, b) + (x, y) : (a, b) \in A\}.$$

- Example: If *A* is the cross-shaped set, and *w* = (2,2)

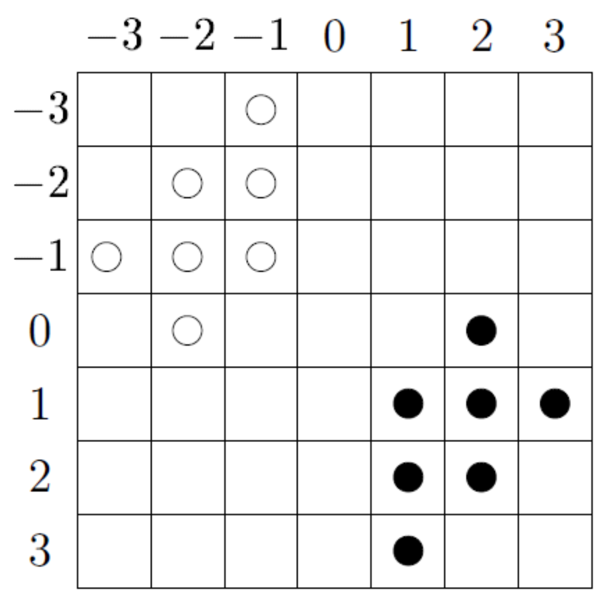


Reflection

- *A* is set of pixels
- Reflection of *A* is given by

$$\hat{A} = \{(-x, -y) : (x, y) \in A\}.$$

- An example of a reflection



Mathematical Morphology

- 2 basic mathematical morphology operations, (built from translations and reflections)
  - **Dilation**
  - **Erosion**

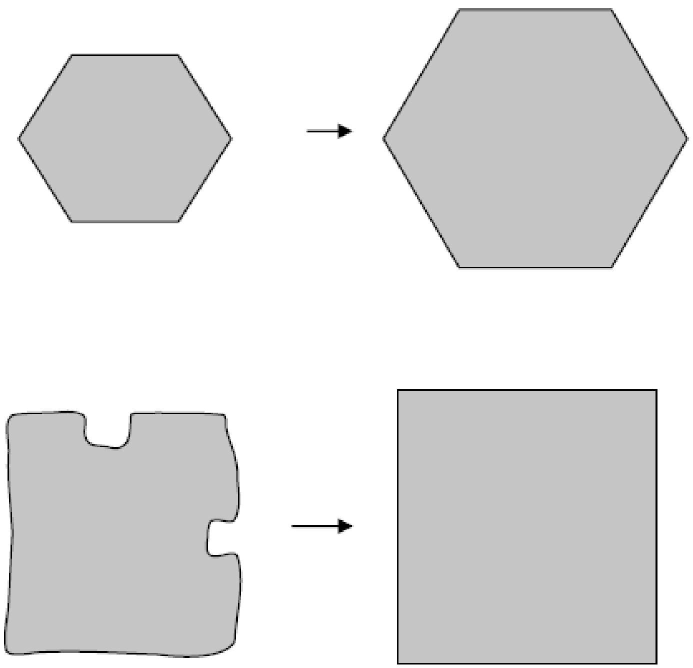
Also several composite relations

- **Closing and Opening**
- **Conditional Dilation**
- **. . .**

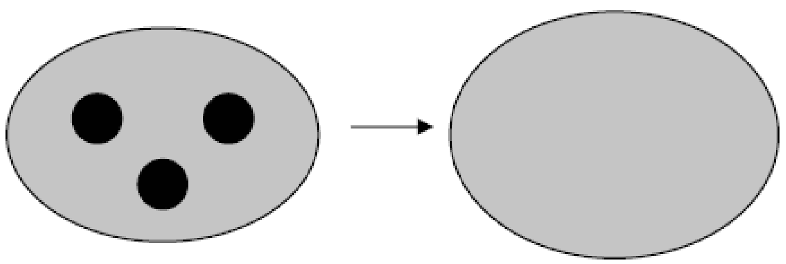
Dilation

- Dilation **expands** connected sets of 1s of a binary image. It can be used for

1. Growing features



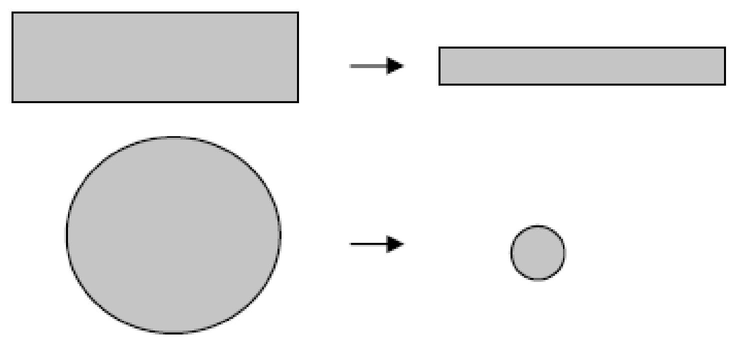
2. Filling holes and gaps



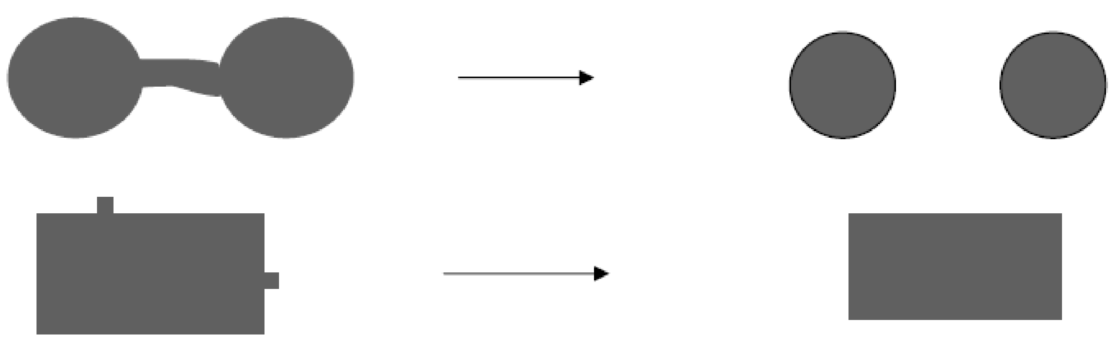
Erosion

- Erosion **shrinks** connected sets of 1s in binary image.
- Can be used for

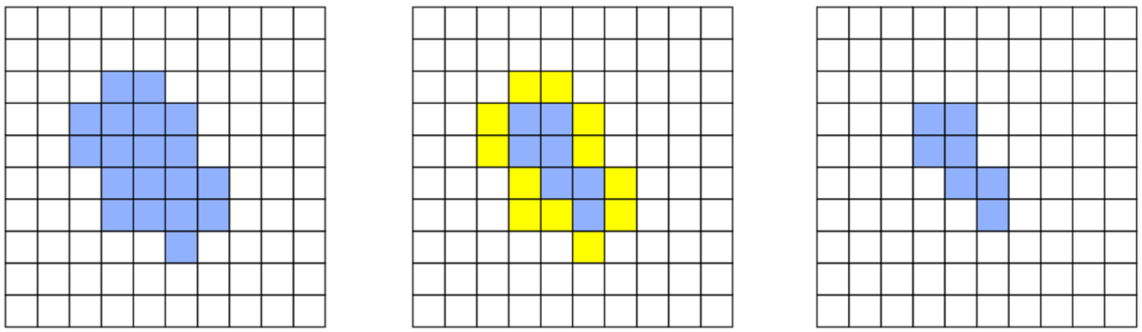
1. shrinking features



2. Removing bridges, branches and small protrusions

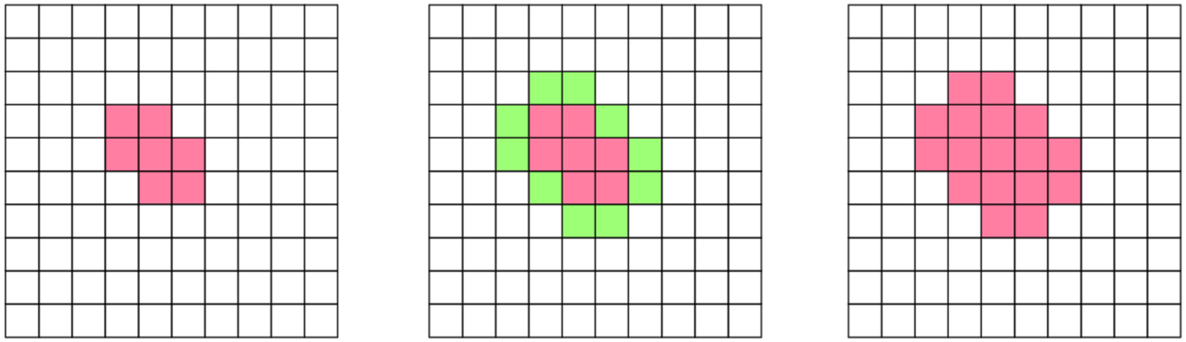


Shrink and Let Grow



(a) (b) (c)

Shrinking: remove border pixels

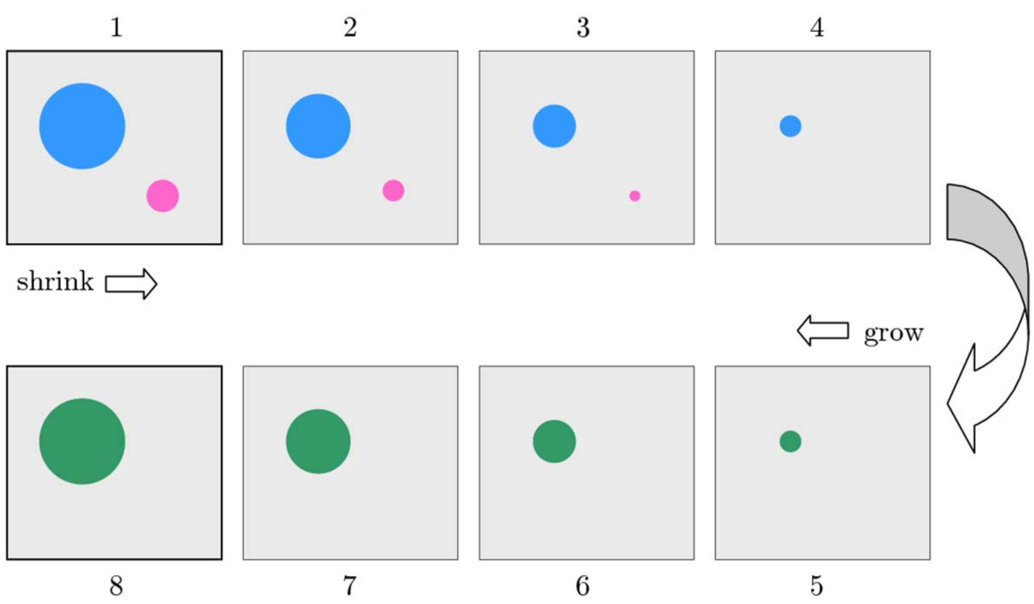


(a) (b) (c)

Growing: add layer of pixels at border

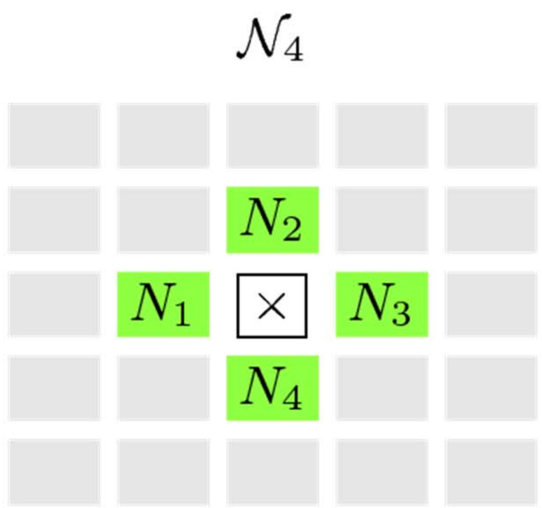
Shrinking and Let Grow

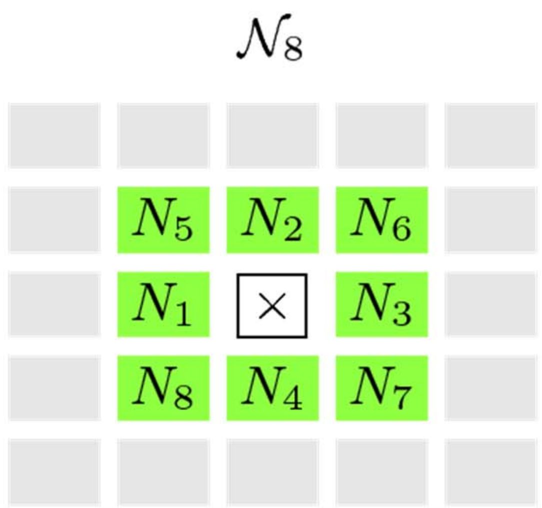
- Image structures are iteratively shrunk by peeling off a layer of thickness (layer of pixel) at boundaries
- Shrinking removes smaller structures, leaving only large structures
- Remaining structures are then grown back by same amount
- Eventually, large structures back to original size while smaller regions have disappeared



Basic Morphological Operations

- Definitions:
  - **4-Neighborhood (*N_4*):** 4 pixels adjacent to given pixel in horizontal and vertical directions
  - **8-Neighborhood (*N_8*): :** 4 pixels in N_4 + 4 pixels adjacent along diagonals





Formal Specification as Point Sets

- Morphological operations can be expressed by describing images as **2D point sets**
- For example, for a binary image ($I(u,v) \in \{0,1\}$)

$$\mathcal{Q}_I = \{\boldsymbol{p} \mid I(\boldsymbol{p}) = 1\}$$

- **Example:** OR operation union of individual sets

$$\mathcal{Q}_{I_1 \vee I_2} = \mathcal{Q}_{I_1} \cup \mathcal{Q}_{I_2}$$

Dilation

- Suppose *A* and *B* are sets of pixels, **dilation of A by B**

$$A \oplus B = \bigcup_{x \in B} A_x.$$

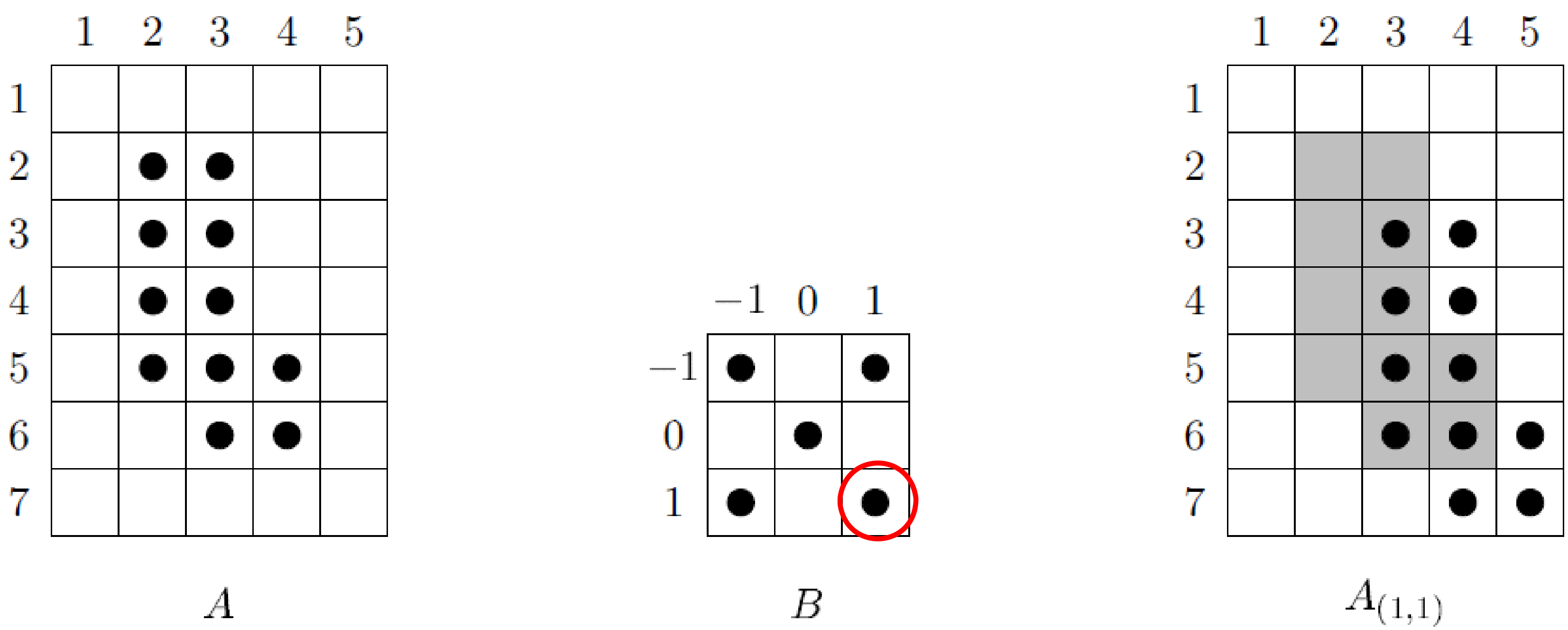
- Also called **Minkowski addition**. **Meaning?**
- Replace every pixel in *A* with copy of *B* (or vice versa)
- For every pixel *x* in *B,*
  - Translate *A* by *x*
  - Take union of all these translations

$$A \oplus B = \{(x, y) + (u, v) : (x, y) \in A, (u, v) \in B\}.$$

Dilation Example

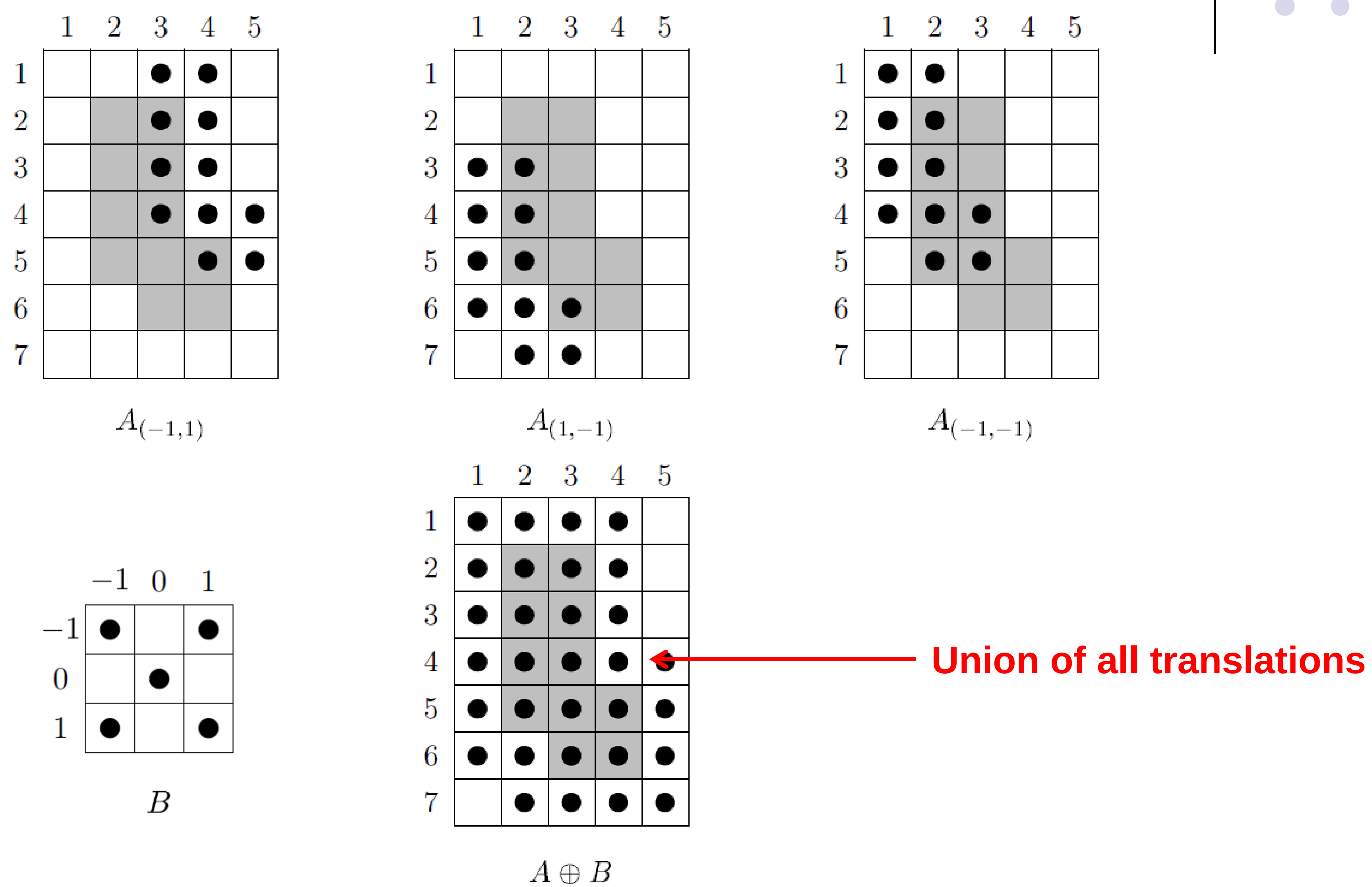
- For *A* and *B* shown below

$$B = \{(0,0),(1,1),(-1,1),(1,-1),(-1,-1)\}$$



Translation of *A* by (1,1)

Dilation Example



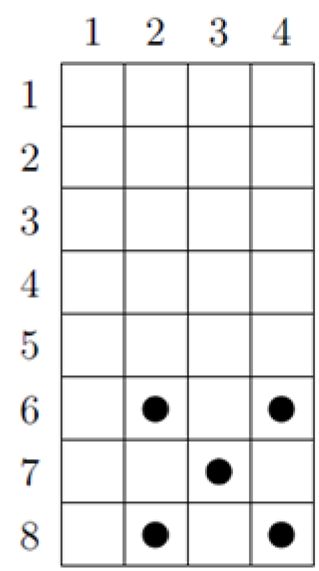
Another Dilation Example

- Dilation increases size of structure
- *A* and *B* do not have to overlap
- **Example:** For the same *A*, if we change *B* to

$$B = \{(7,3), (6,2), (6,4), (8,2), (8,4)\}$$

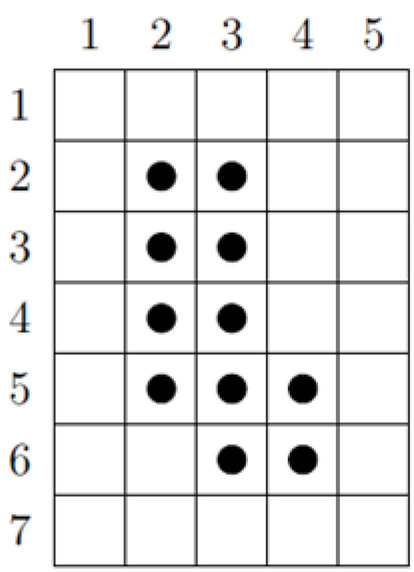
so that

$$A \oplus B = A_{(7,3)} \cup A_{(6,2)} \cup A_{(6,4)} \cup A_{(8,2)} \cup A_{(8,4)}$$

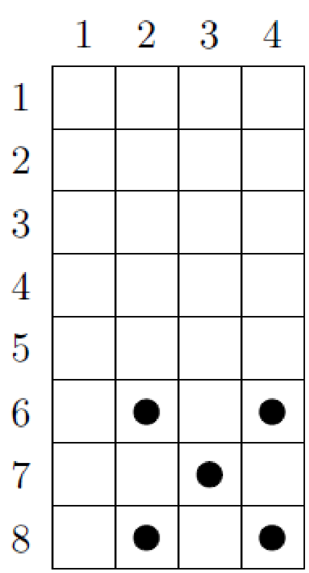


B

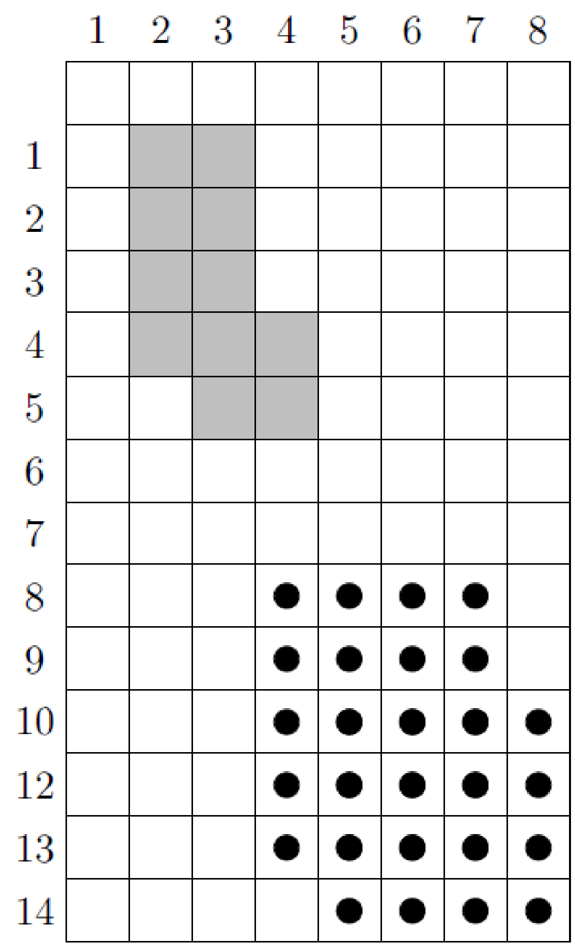
Another Dilation Example



A

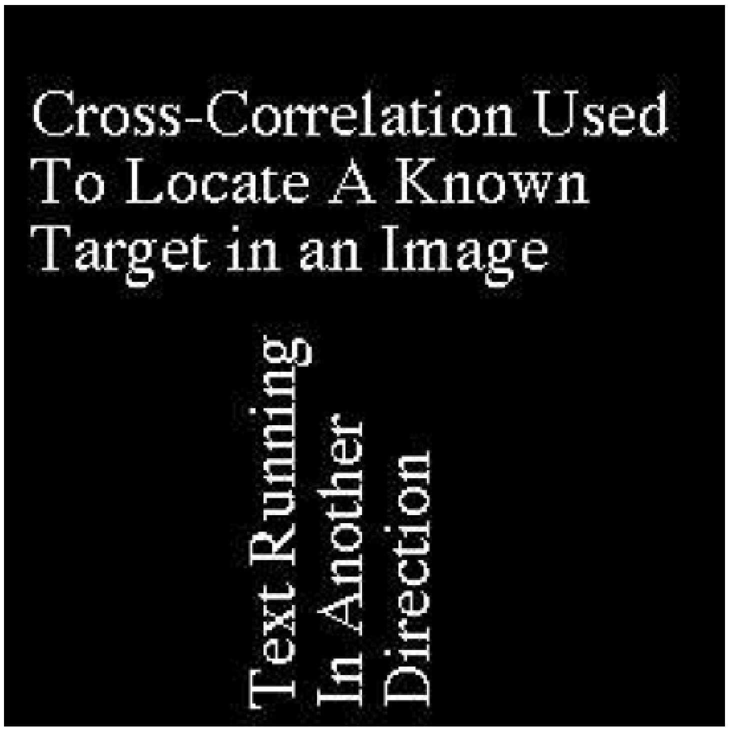


B



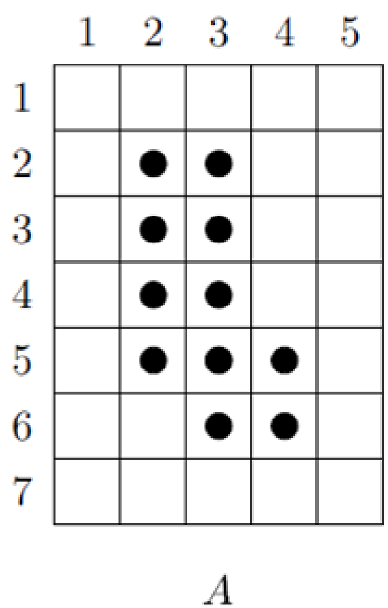
$A \oplus B$

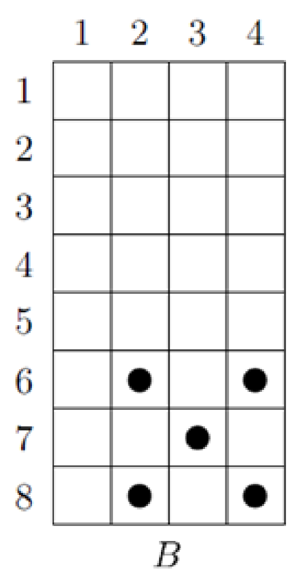
Example: Dilation of a Binary Image



Dilation

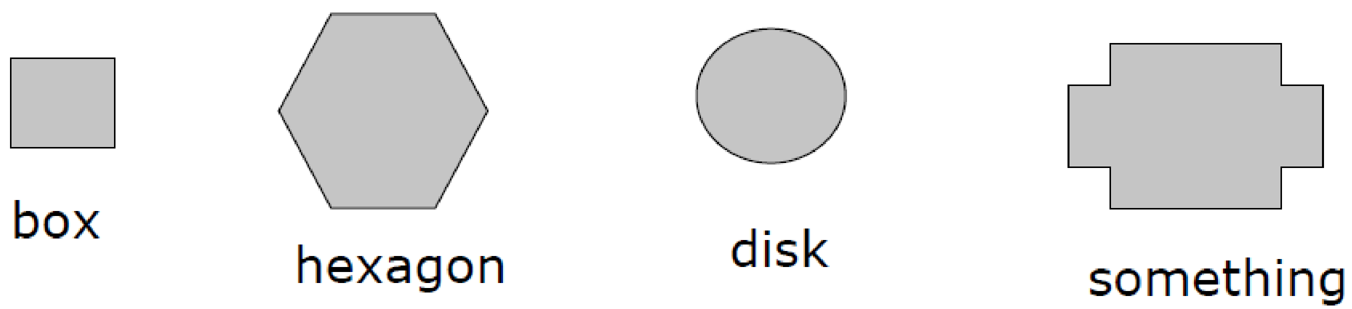
- We usually assume
  - *A* is being processed
  - *B* is a smaller set of pixels, called the **structuring element**





The Structuring Element

- A structuring element is a shape mask used in the basic morphological operations
- They can be any shape and size that is digitally representable, and each has an origin.



The Structuring Element

- Structuring element somewhat similar to a filter
- Contains only 0 and 1 values
- **Hot spot** marks origin of coordinate system of ***H***
- **Example of structuring element:** 1-elements marked with •, 0-cells are empty

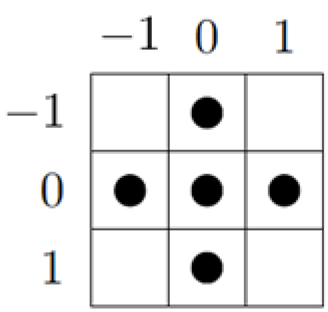
$$H(i,j) \in \{0,1\}$$

Erosion

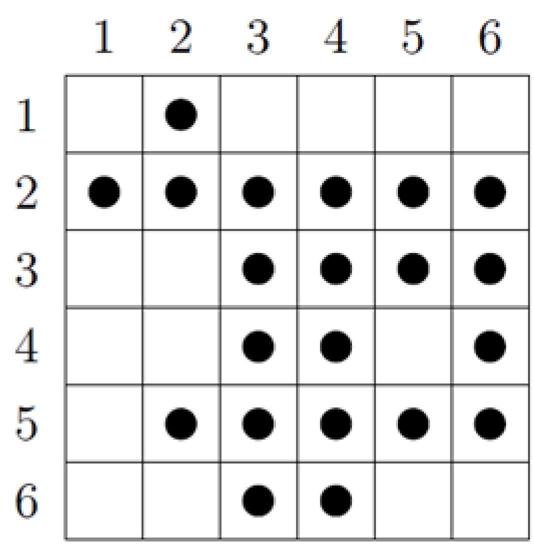
- Given sets A and B, the **erosion of A by B**

$$A \ominus B = \{w : B_w \subseteq A\}.$$

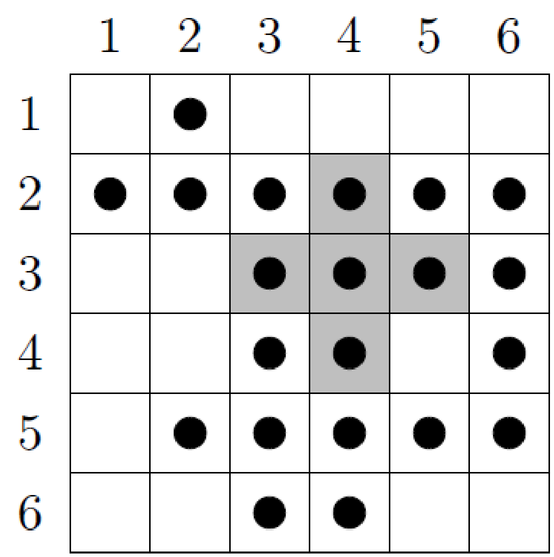
- Find all occurrences of B in A



B



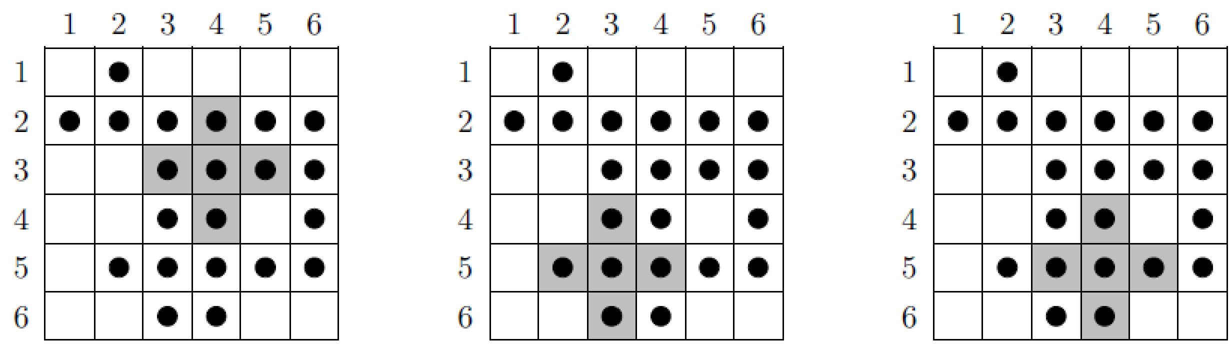
A



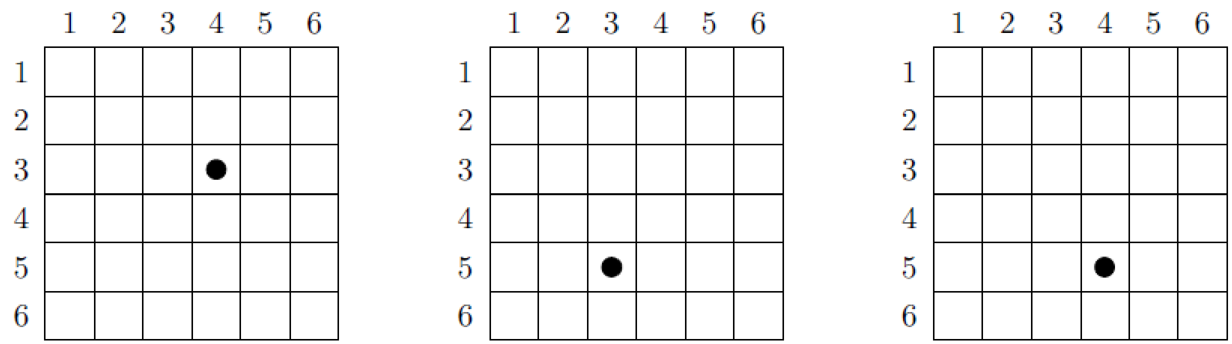
Example: 1 occurrence of B in A

Erosion

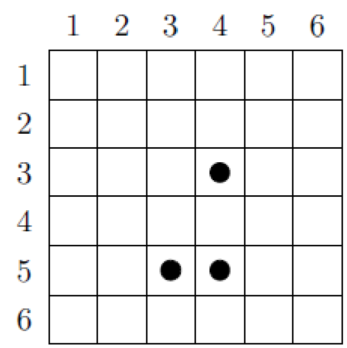
All occurrences of *B* in *A*



For each occurrences Mark center of *B*



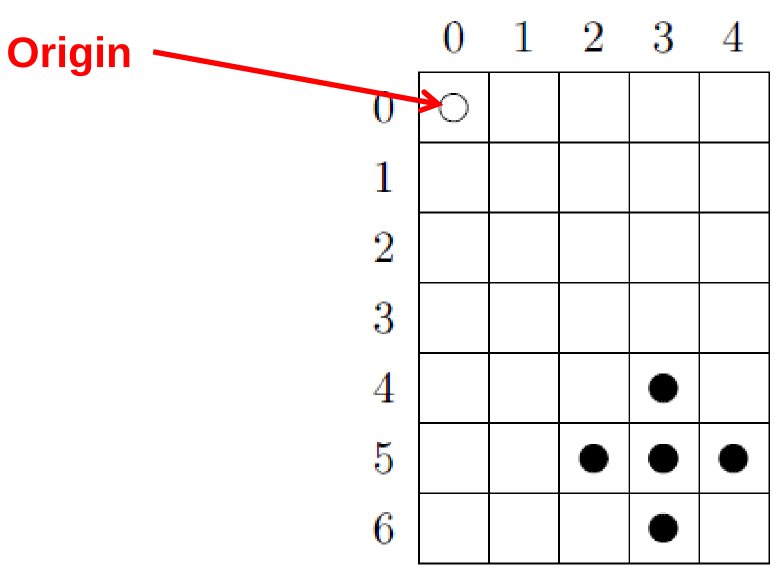
Erosion: union of center of all occurrences of *B* in *A*



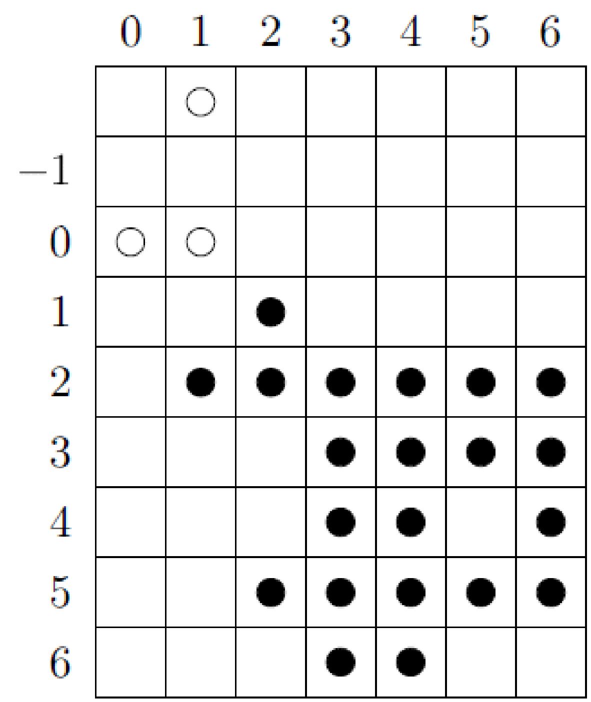
$A \ominus B$

Another Erosion Example

- The structuring element (*B*) does not have to contain the origin
- Another example where ***B*** does not contain the origin

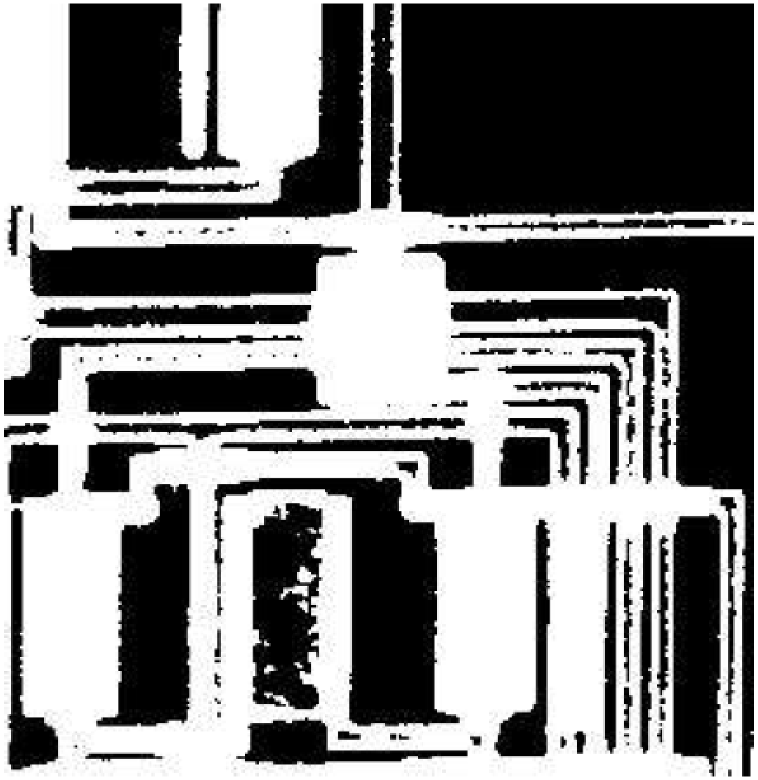


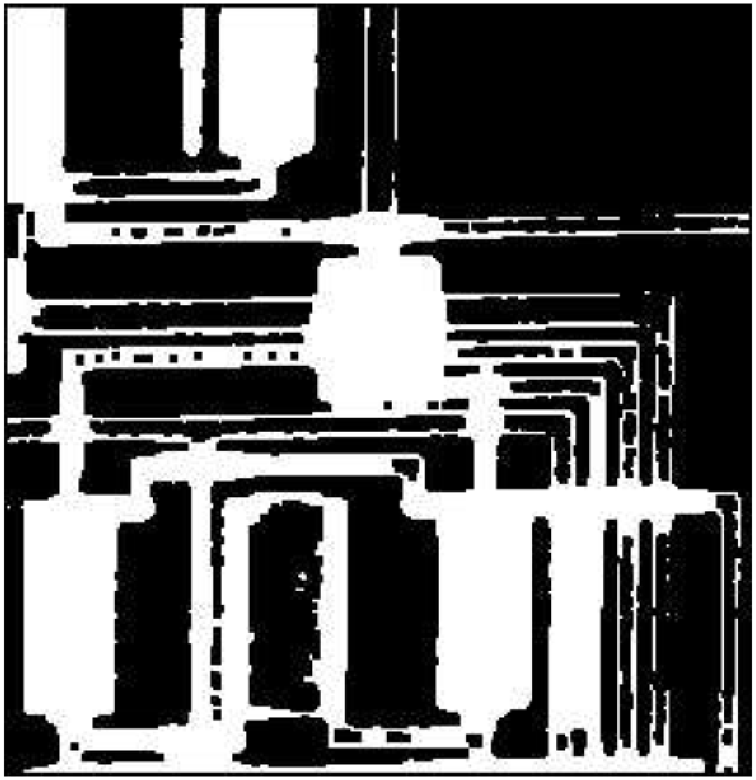
B



$A \ominus B$

Example: Erosion of Binary Image





Erosion

- Erosion related to **minkowski subtraction**

$$A - B = \bigcap_{b \in B} A_b.$$

- Erosion and dilation are **inverses** of each other
- It can be shown that

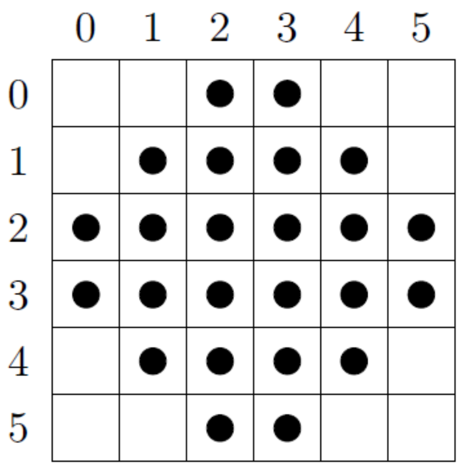
$$\overline{A \ominus B} = \overline{A} \oplus \hat{B}. \quad \text{reflection}$$

- And also that

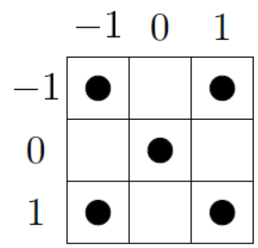
$$\overline{A \oplus B} = \overline{A} \ominus \hat{B}.$$

An Application: Boundary Detection

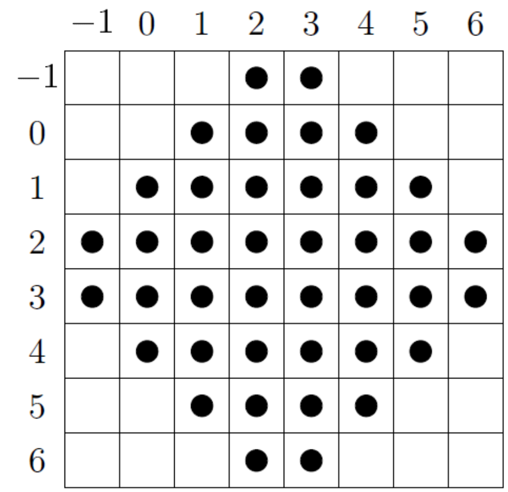
- Given an image *A* and structuring element *B*



A

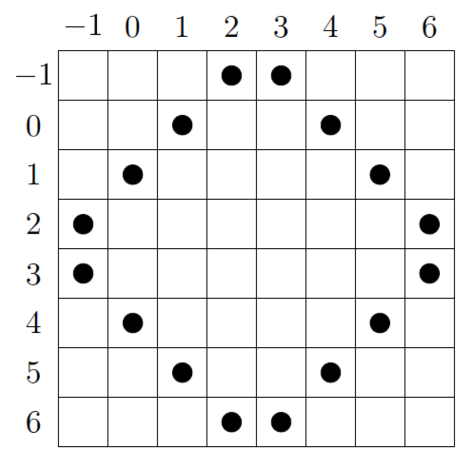


B



$A \oplus B$

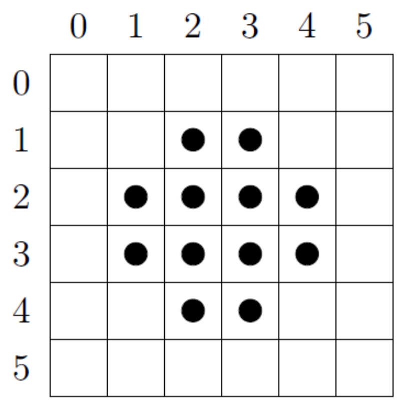
We can define **external boundary**



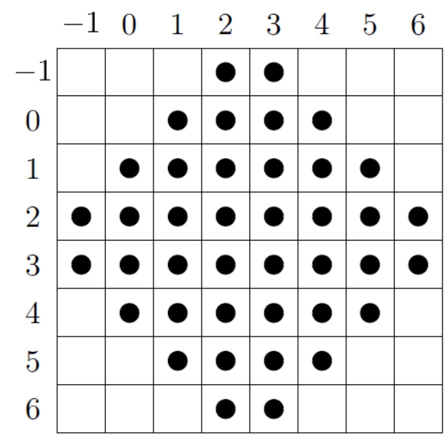
$(A \oplus B) - A)$

An Application: Boundary Detection

- **Dilation of image *A* - erosion image *A*** (by structuring element *B*)



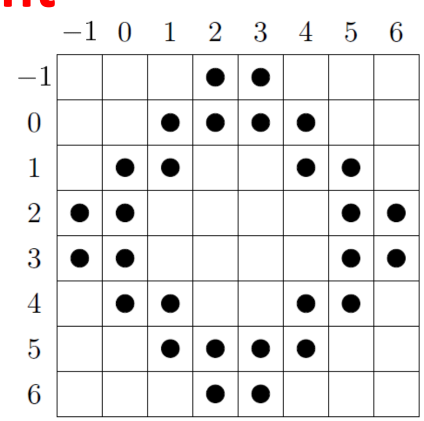
$A \ominus B$



$A \oplus B$

- We can define **morphological gradient**

$$(A \oplus B) - (A \ominus B)$$



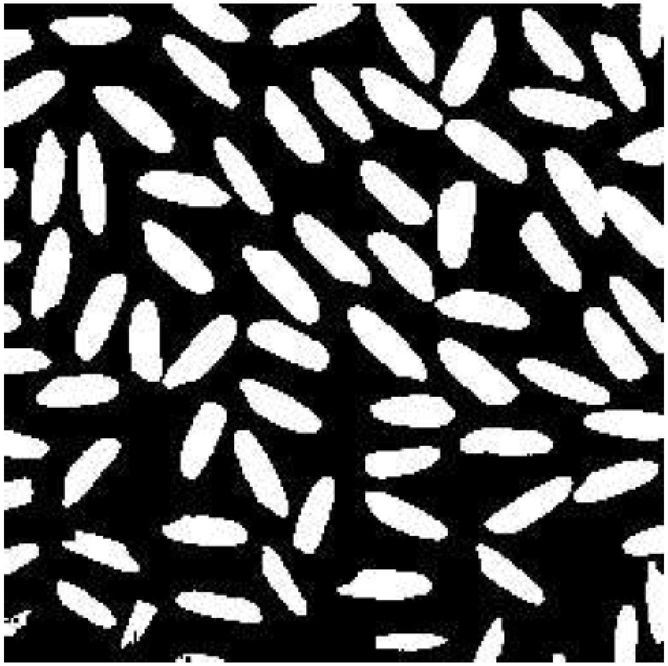
$(A \oplus B) - (A \ominus B)$

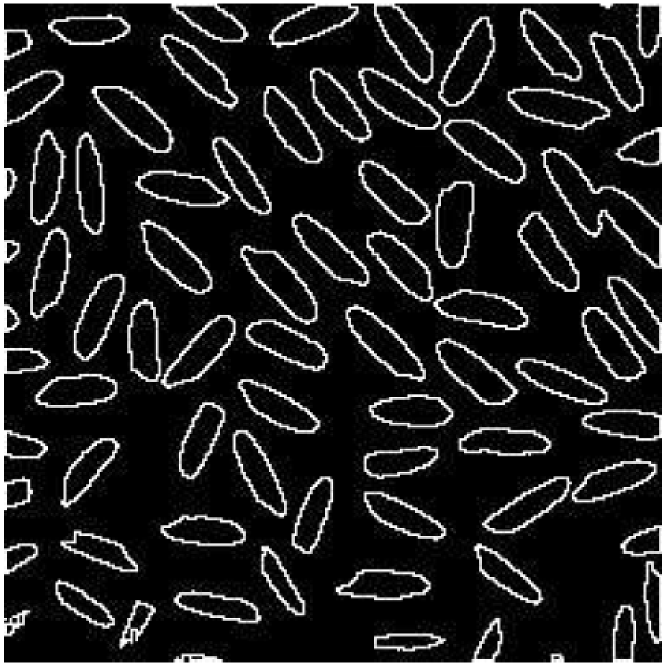
Morphological gradient = Dilation - erosion

Example: Internal Boundary of Binary Image

- We can also define **internal boundary** as

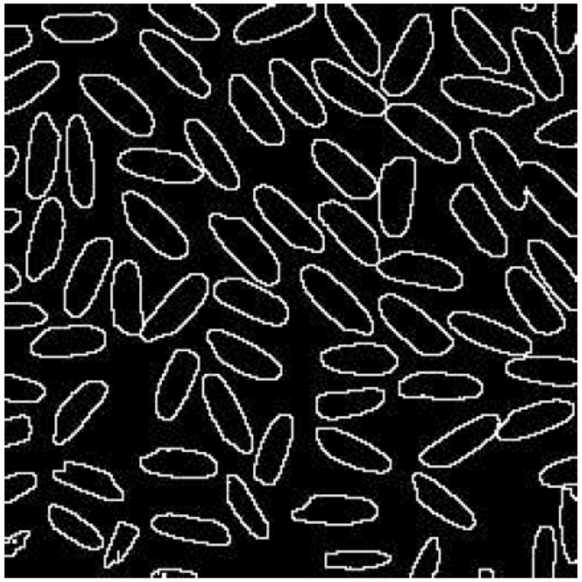
$$A - (A \ominus B)$$





Example: External Boundary and Morphological Gradient

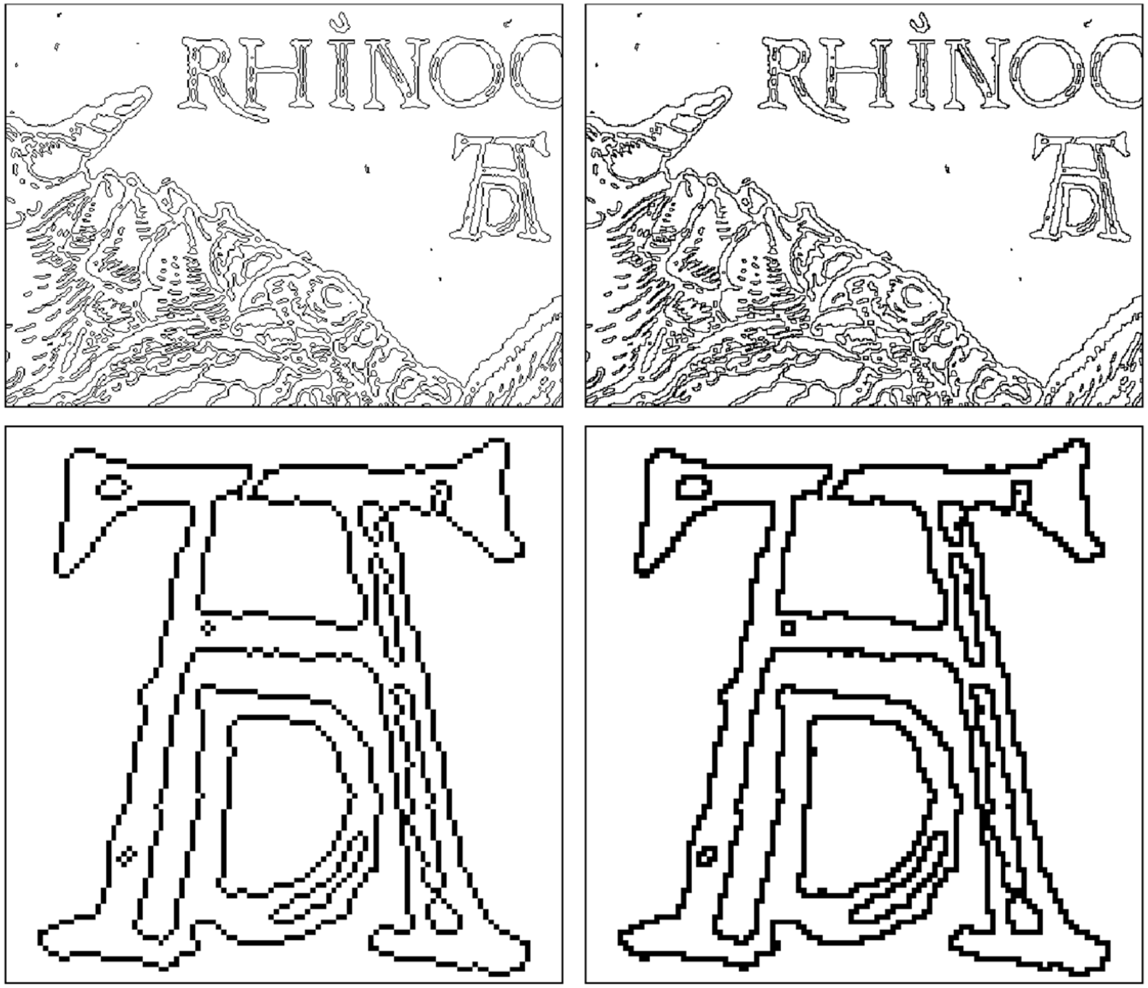
Image



External Boundary

Morphological Gradient

Example: Extraction of Boundary Pixels using Morphological Operations



(a) (b)

Properties of Dilation

- Dilation operation is **commutative**

$$I \oplus H = H \oplus I$$

- Dilation is **associative** (ordering of applying it not important)

$$(I_1 \oplus I_2) \oplus I_3 = I_1 \oplus (I_2 \oplus I_3)$$

- Thus as with separable filters, more efficient to apply large structuring element as sequence of smaller structuring elements

$$I \oplus H_{\text{big}} \;=\; (\ldots((I \oplus H_1) \oplus H_2) \oplus \ldots \oplus H_K)$$

Properties of Erosion

- Erosion is **not commutative**

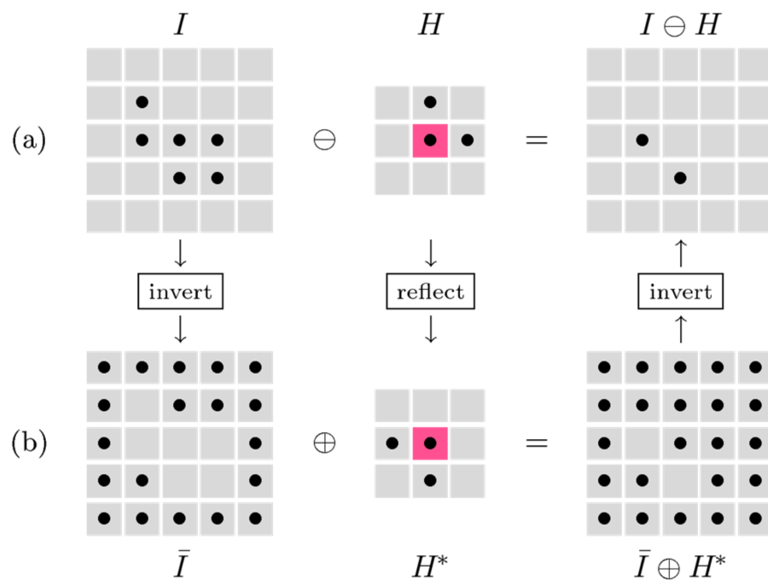
$$I \ominus H \neq H \ominus I$$

- If erosion and dilation are combined, this chain rule holds

$$(I_1 \ominus I_2) \ominus I_3 \; = \; I_1 \ominus (I_2 \oplus I_3)$$

- Dilation of **foreground** = inverting (erosion of **background**)

$$I \oplus H \; \equiv \; \overline{(\bar{I} \ominus H^*)}$$



Dilation and Erosion Algorithm

DILATE (I, H)

  I: binary image of size $w \times h$
  H: binary structuring element defined over region $\mathcal{R}_H$
  Returns the dilated image $I' = I \oplus H$

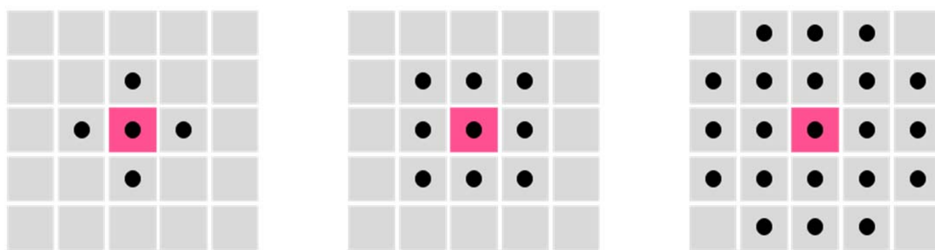
$I' \leftarrow$ new binary image of size $w \times h$
$I'(u,v) \leftarrow 0$, for all (u,v) $\quad \triangleright\ I' \leftarrow \varnothing$
for all $(i,j) \in \mathcal{R}_H$ **do** $\quad \triangleright\ (i,j) = \boldsymbol{q}$
  if $H(i,j) = 1$ **then** $\quad \triangleright\ \boldsymbol{q} \in H$
    MERGE THE SHIFTED $I_{\boldsymbol{q}}$ WITH I': $\quad \triangleright\ I' \leftarrow I' \cup I_{\boldsymbol{q}}$
    for $u \leftarrow 0 \ldots (w-1)$ **do**
      for $v \leftarrow 0 \ldots (h-1)$ **do** $\quad \triangleright\ (u,v) = \boldsymbol{p}$
        if $I(u,v) = 1$ **then** $\quad \triangleright\ \boldsymbol{p} \in I$
          $I'(u+i, v+j) \leftarrow 1$ $\quad \triangleright\ I' \leftarrow I' \cup (\boldsymbol{p}+\boldsymbol{q})$
return I'.

ERODE (I, H)

$\bar{I} \leftarrow$ INVERT(I) $\quad \triangleright\ \bar{I} \leftarrow \neg I$
$H^* \leftarrow$ REFLECT(H)
return INVERT(DILATE$(\bar{I}, H^*)$). $\quad \triangleright\ I \ominus H = \overline{(\bar{I} \oplus H^*)}$

Designing Morphological Filters

- A morphological filter is specified by:
  - Type of operation (e.g. dilation, erosion)
  - Contents of structuring element

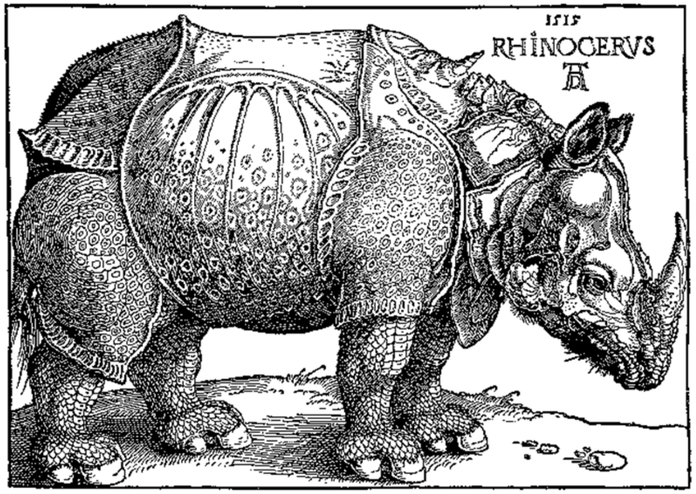


(a) 4-neighborhood (b) 8-neighborhood (c) Small Disk (circular)

- In practice, quasi-circular shaped structuring elements used
- Dilation with circular structuring of radius ***r*** adds thickness ***r***
- Erosion with circular structuring of radius ***r*** removes thickness ***r***

Example: Dilation and Erosion

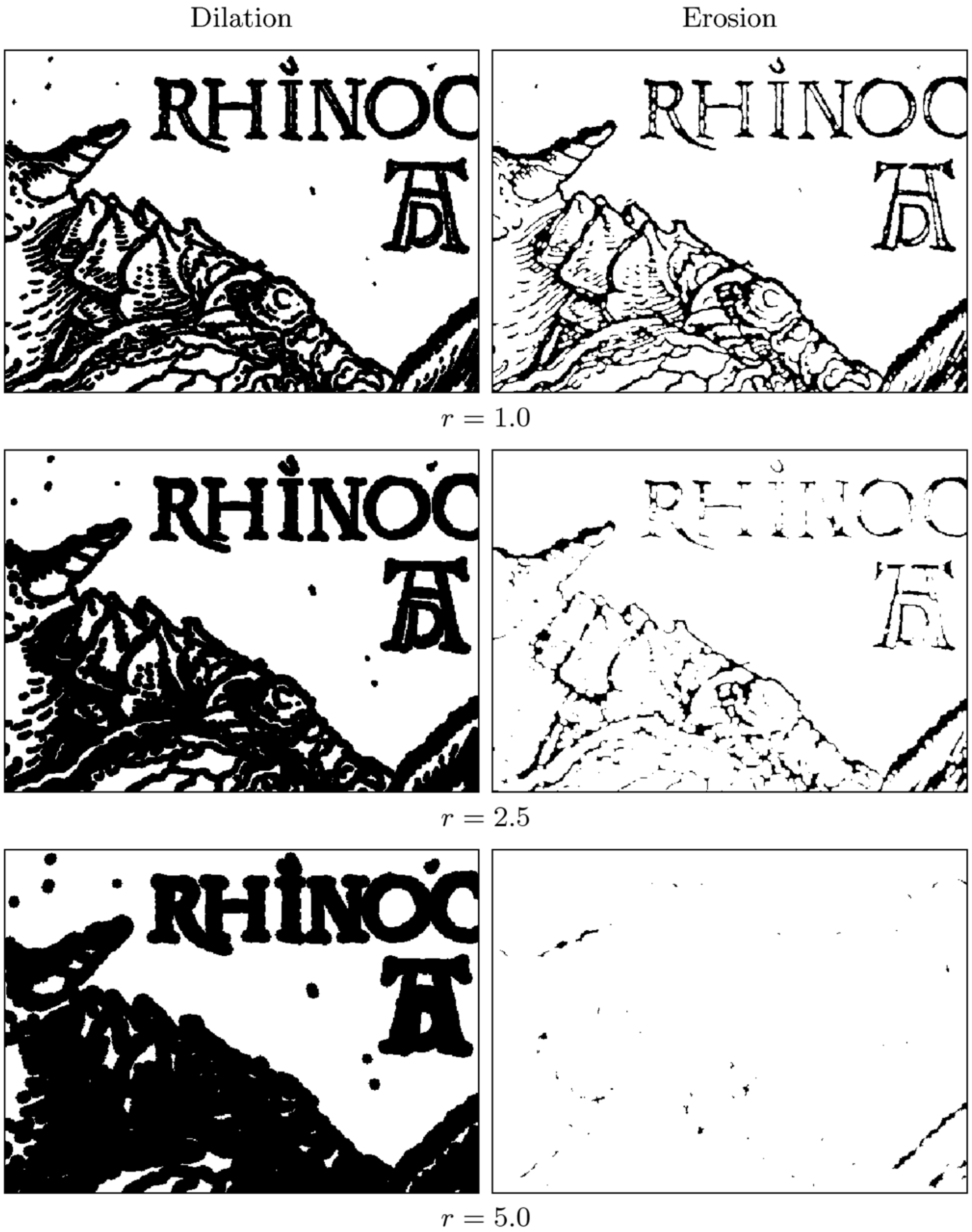
- What if we erode and dilate the following image with disk-shaped structuring element?

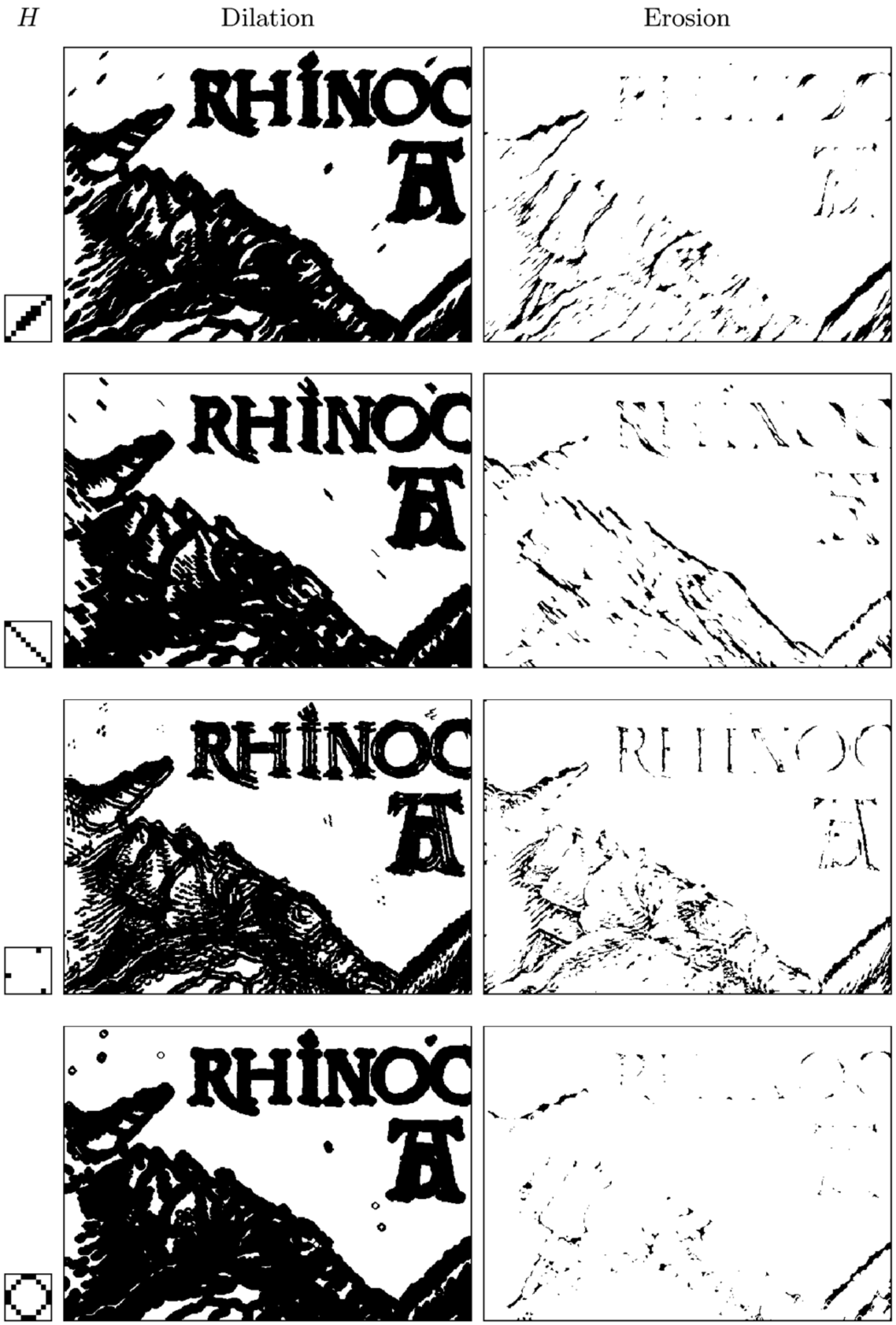


Original image

Apply dilation and erosion to this close up section

Example: Dilation and Erosion



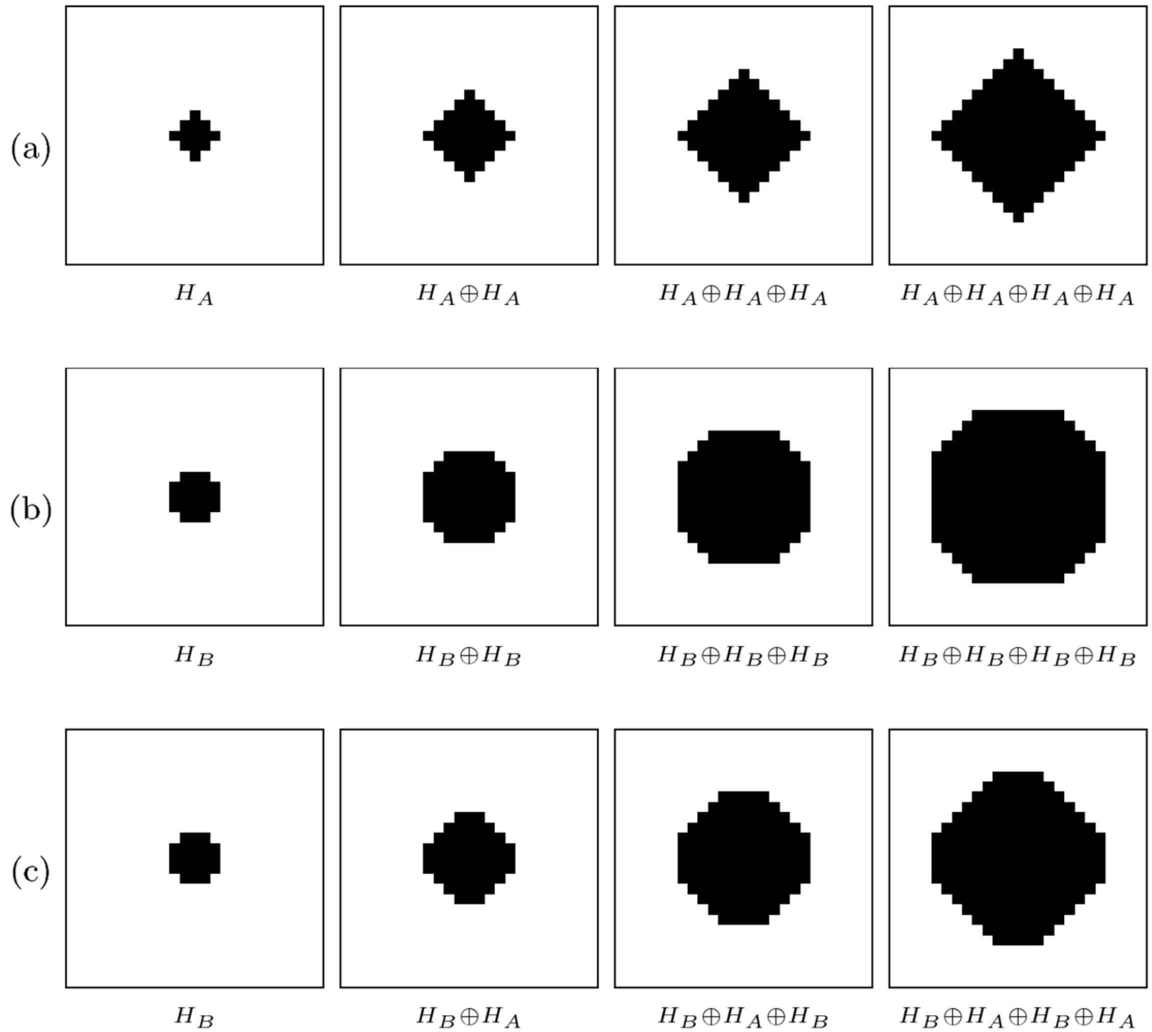


Dilation and Erosion using Different Structuring Elements

Example: Composing Large Filters by Repeatedly Applying Smaller Filters

- More efficient
- E.g. composing

Isotropic filter



Opening

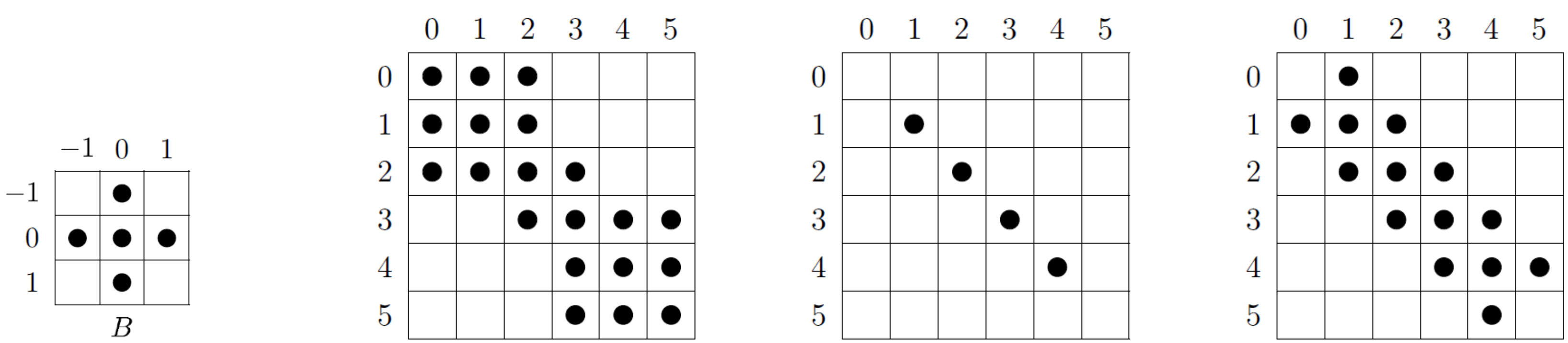
- Opening and closing: operations built on dilation and erosion
- Opening of *A* by structuring element *B*

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

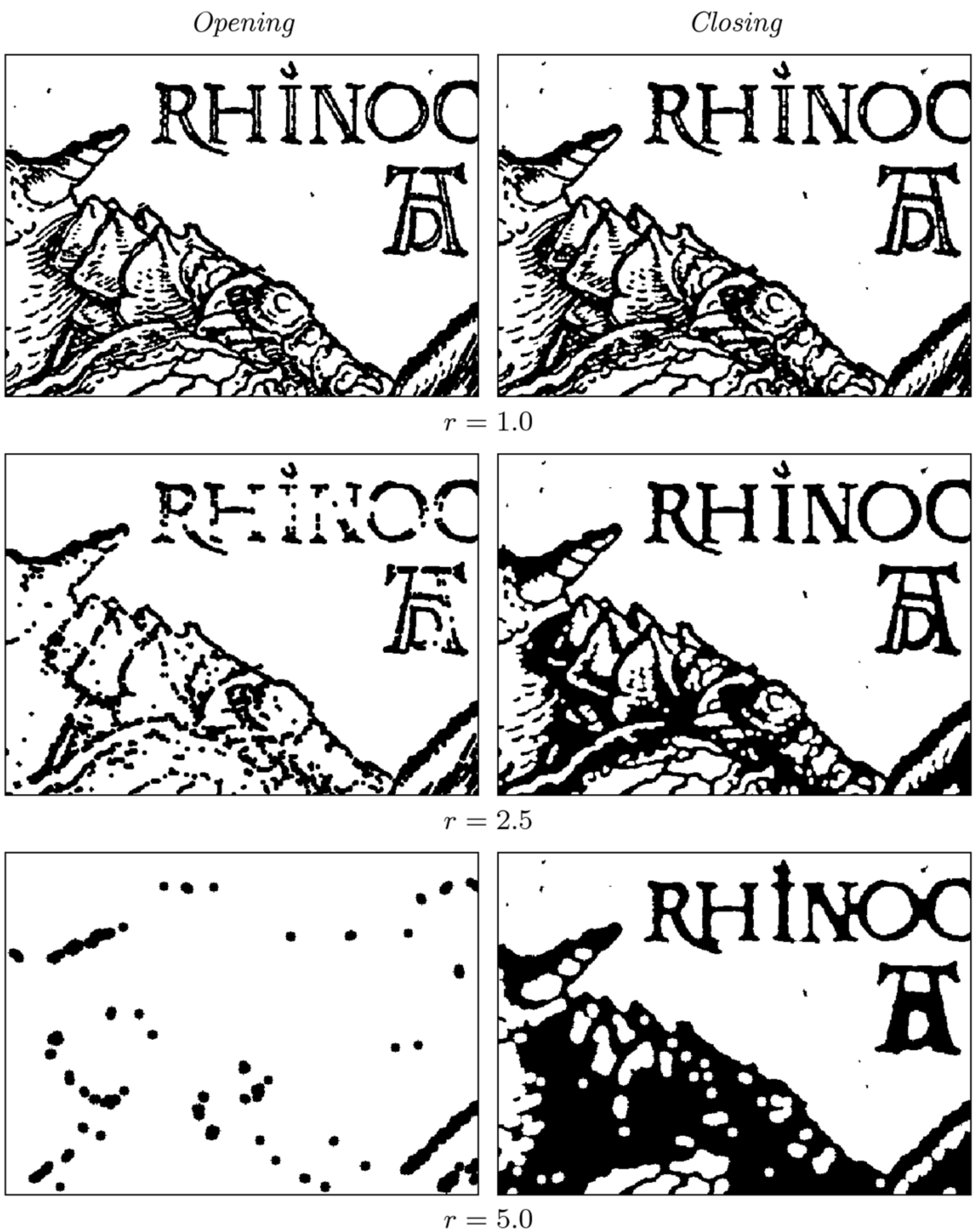
- i.e. opening = erosion followed by dilation. Alternatively

$$A \circ B = \cup\{B_w : B_w \subseteq A\}.$$

- i.e. Opening = union of all translations of *B* that fit in *A*
- **Note:** Opening includes all of B, erosion includes just (0,0) of *B*



B $\quad$ A $\quad$ $A \ominus B$ $\quad$ $A \circ B$

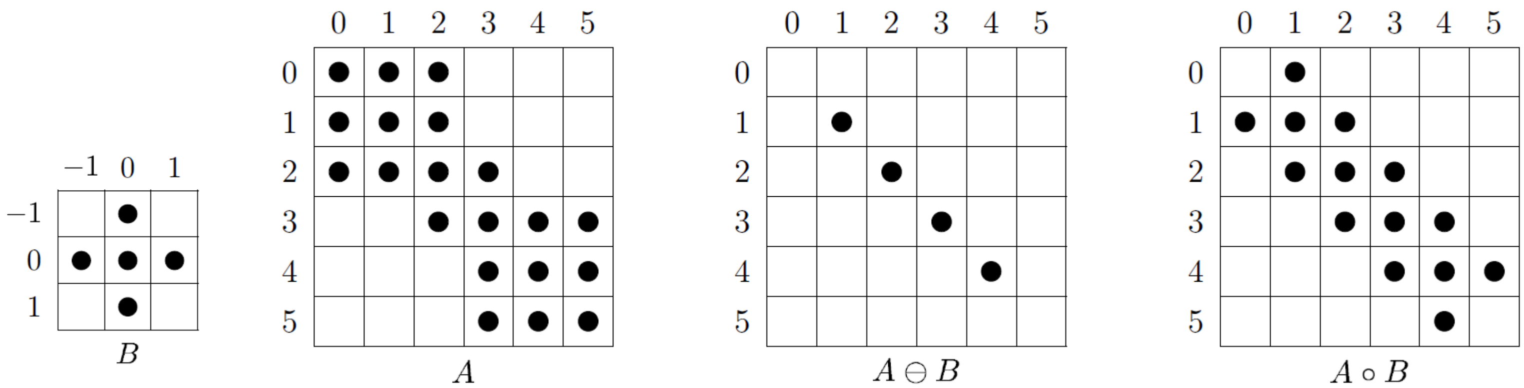


Binary opening and closing with disk-shaped Structuring elements of radius r = 1.0, 2.5, 5.0

Opening

- All foreground structures smaller than structuring element are eliminated by first step (erosion)
- Remaining structures smoothed by next step (dilation) then grown back to their original size

Properties of Opening



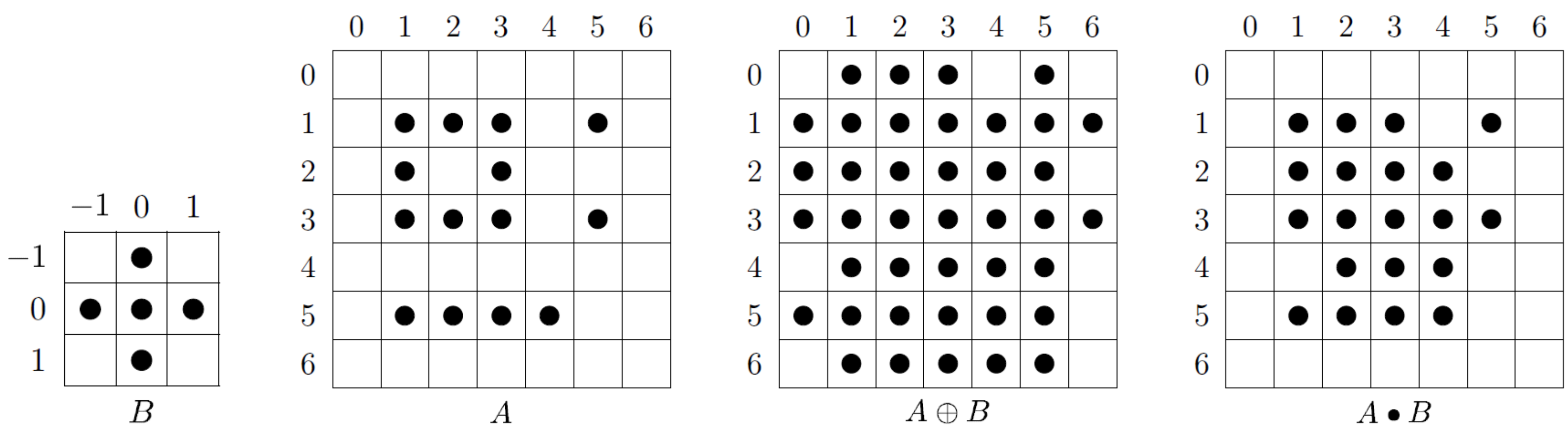
1. $(A \circ B) \subseteq A$. : Opening is subset of *A* (not the case with erosion)
2. $(A \circ B) \circ B = A \circ B$. : Can apply opening only once, also called **idempotence** (not the case with erosion
3. Subsets: If $A \subseteq C$, then $(A \circ B) \subseteq (C \circ B)$.
4. Opening tends to smooth an image, break narrow joins, and remove thin protrusions.

Closing

- Closing of *A* by structuring element *B*

$$A \bullet B = (A \oplus B) \ominus B.$$

- i.e. closing = dilation followed by erosion



Properties of Closing

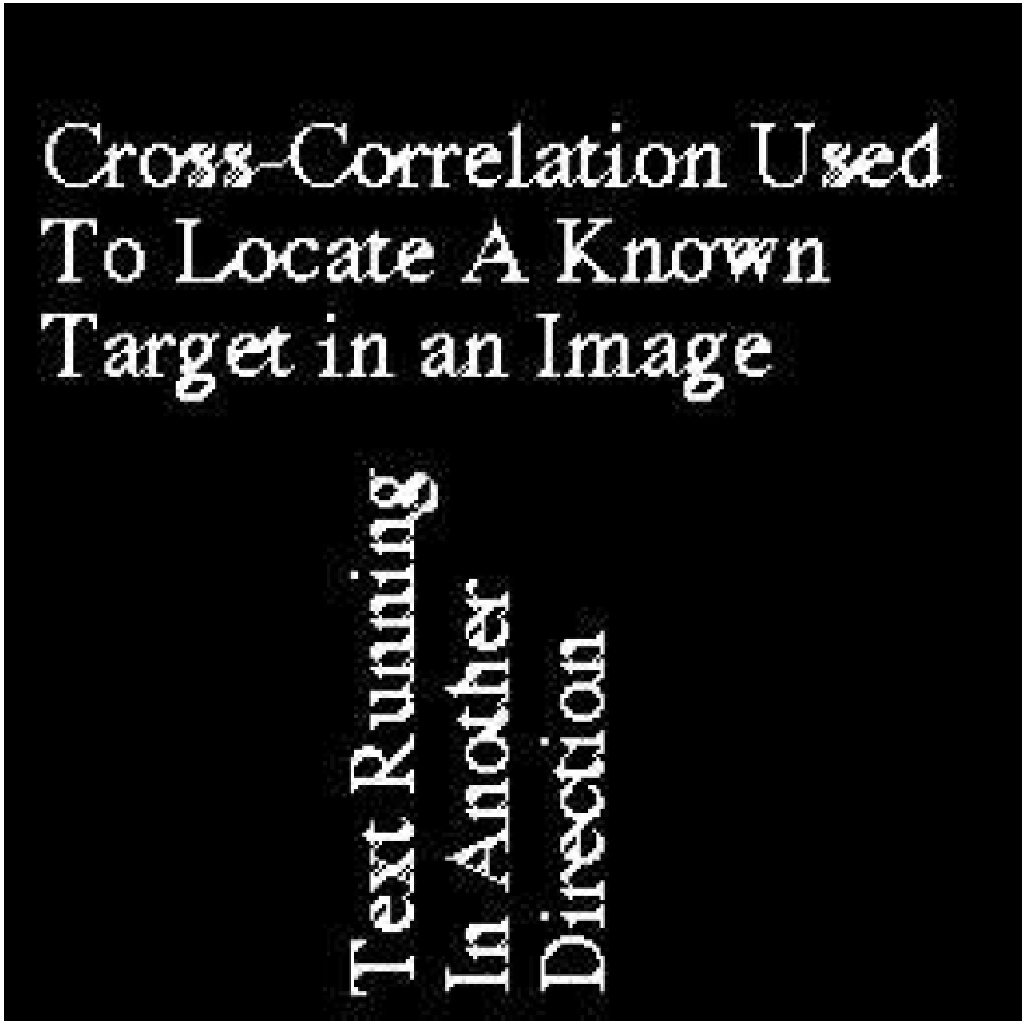
1. Subset: $A \subseteq (A \bullet B)$.

2. **Idempotence:** $(A \bullet B) \bullet B = A \bullet B$;

3. Also If $A \subseteq C$, then $(A \bullet B) \subseteq (C \bullet B)$.

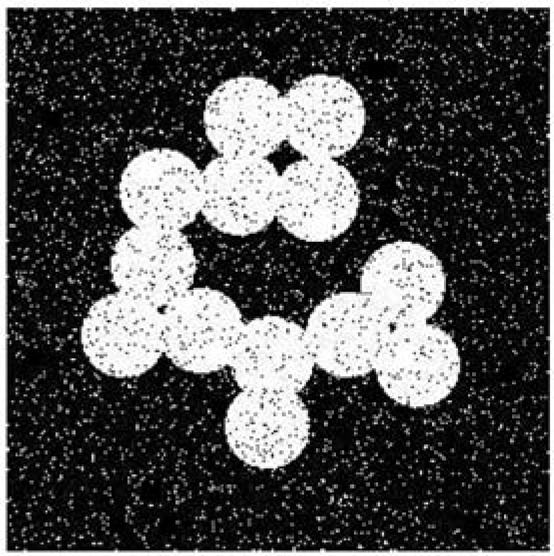
4. Closing tends to:
   a) Smooth an image
   b) Fuse narrow breaks and thin gulfs
   c) Eliminates small holes.

An Example of Closing



Noise Removal: Morphological Filtering

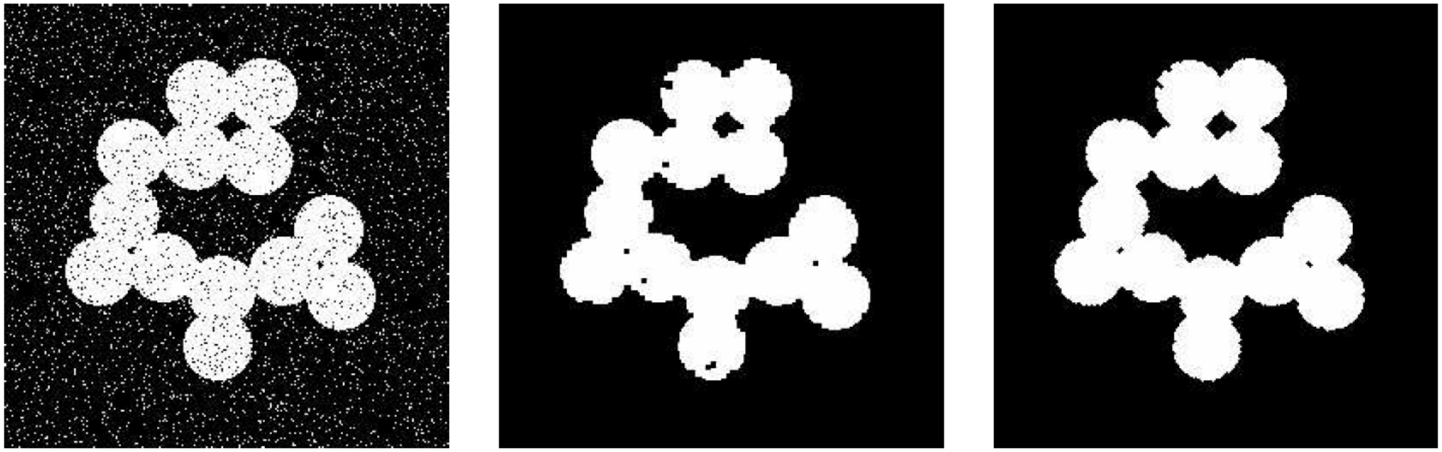
- Suppose *A* is image corrupted by impulse noise (some black, some white pixels, shown in (a) below)



(a)

- $A \ominus B$ removes single black pixels, but enlarges holes
- We can fill holes by dilating twice $((A \ominus B) \oplus B) \oplus B$.

Noise Removal: Morphological Filtering



(a) (b) (c)

(b) Filter once
(c) Filter Twice

- First dilation returns the holes to their original size
- Second dilation removes the holes but enlarges objects in image
- To reduce them to their correct size, perform a final erosion:

$$(((A \ominus B) \oplus B) \oplus B) \ominus B.$$

- Inner 2 operations = opening, Outer 2 operations = closing.
- This noise removal method = opening followed by closing

$$(A \circ B) \bullet B).$$

Relationship Between Opening and Closing

- Opening and closing are duals
  - i.e. Opening foreground = closing background, and vice versa

- Complement of an opening = the closing of a complement

$$\overline{A \bullet B} = \overline{A} \circ \hat{B}$$

- Complement of a closing = the opening of a complement.

$$\overline{A \circ B} = \overline{A} \bullet \hat{B}.$$

Grayscale Morphology

- Morphology operations can also be applied to grayscale images
- Just replace (OR, AND) with (MAX, MIN)
- Consequently, morphology operations defined for grayscale images can also operate on binary images (but not the other way around)
  - Matlab has single implementation of morphological operations that works on binary and grayscale
- For color images, perform grayscale morphology operations on each color channel (RGB)
- For grayscale images, structuring element contains real values

Grayscale Morphology

- Elements in structuring element that have value 0 do contribute to result
- Design of structuring elements for grayscale morphology must distinguish between 0 and empty (don't care)

$$\begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & \boxed{2} & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array} \neq \begin{array}{|c|c|c|} \hline & 1 & \\ \hline 1 & \boxed{2} & 1 \\ \hline & 1 & \\ \hline \end{array}$$

Grayscale Dilation

- **Grayscale dilation:** Max (value in filter H + image region)

$$(I \oplus H)(u,v) = \max_{(i,j)\in H} \{ I(u+i, v+j) + H(i,j) \}$$

1. Place filter ***H*** over region of image ***I***

$$\overset{I}{\begin{array}{|c|c|c|c|}\hline 6 & 7 & 3 & 4 \\ \hline 5 & 6 & 6 & 8 \\ \hline 6 & 4 & 5 & 2 \\ \hline 6 & 4 & 2 & 3 \\ \hline \end{array}} \oplus \overset{H}{\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & \boxed{2} & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}} = \overset{I \oplus H}{\begin{array}{|c|c|c|c|}\hline & & & \\ \hline & 8 & 9 & \\ \hline & 7 & 9 & \\ \hline & & & \\ \hline \end{array}}$$

4. Place max value (8) at current filter origin

2. Add corresponding values (***I*** + ***H***)

$$\overset{I+H}{\begin{array}{|c|c|c|}\hline 7 & 8 & 4 \\ \hline 6 & \boxed{8} & 7 \\ \hline 7 & 5 & 6 \\ \hline \end{array}} \quad \max$$

3. Find max of all values (***I*** + ***H***) = 8

Note: Result may be negative value

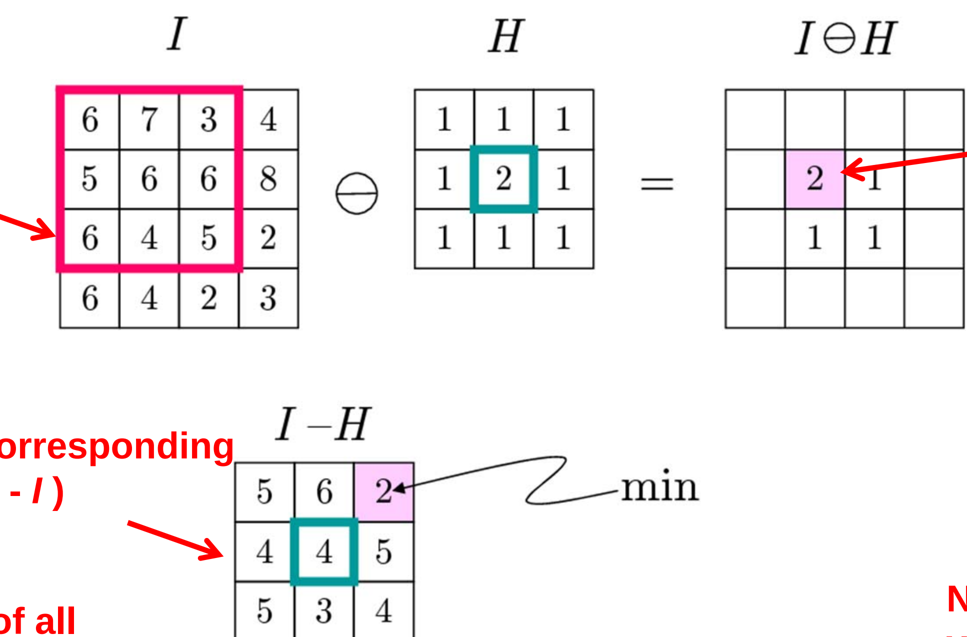
Grayscale Erosion

- **Grayscale erosion:** Min (value in filter *H* + image region)

$$(I \ominus H)(u,v) = \min_{(i,j)\in H} \{ I(u+i, v+j) - H(i,j) \}$$

1. **Place filter *H* over region of image *I***

4. Place min value (2) at current filter origin



2. Subtract corresponding values (*H* - *I*)

3. Find max of all values (*H* - *I*) = 2

Note: Result may be negative value

Grayscale Opening and Closing

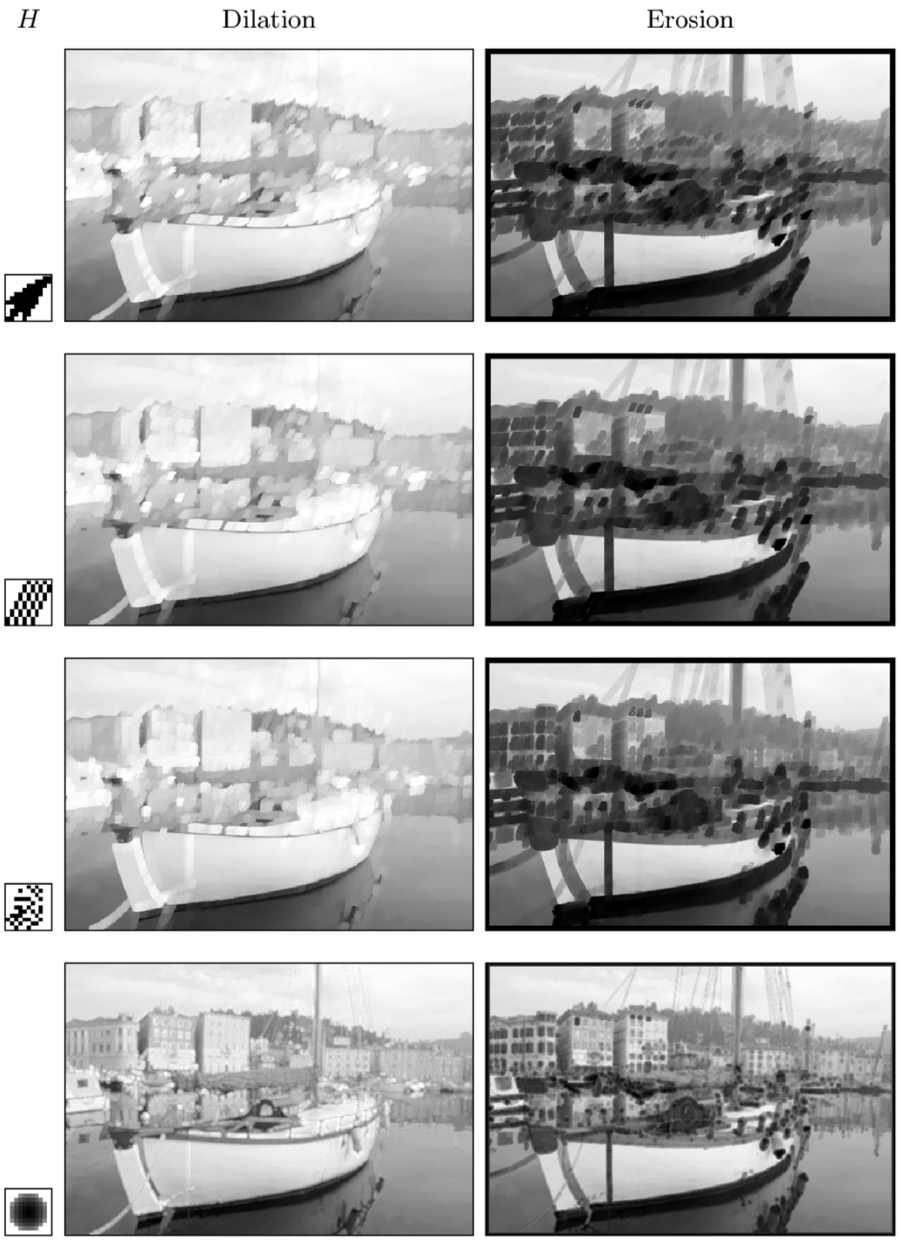
- **Recall:** Opening = erosion then dilation:
- So we can implement grayscale opening as:
  - Grayscale erosion then grayscale dilation

- **Recall:** Closing = dilation then erosion:
- So we can implement grayscale erosion as:
  - Grayscale dilation then grayscale erosion

Grayscale Dilation and Erosion

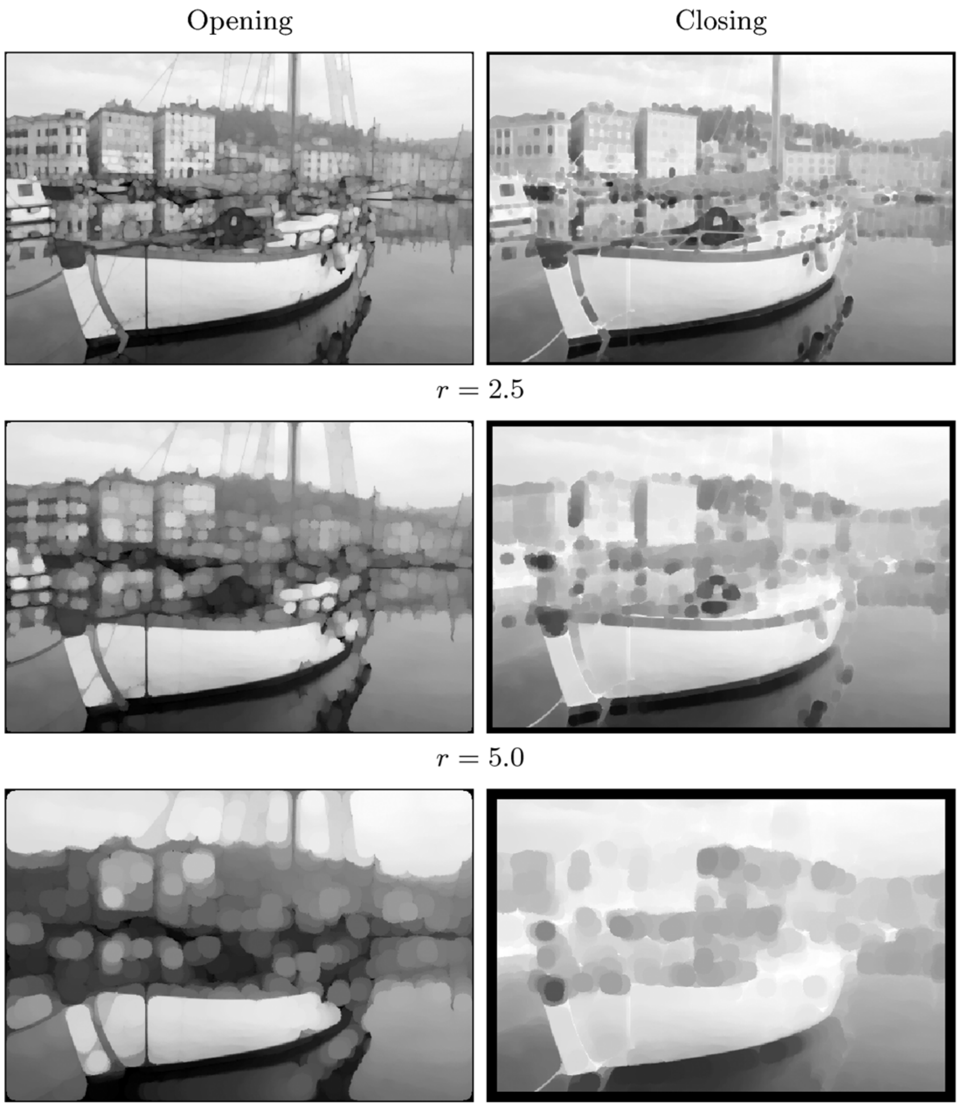
- Grayscale dilation and erosion with disk-shaped structuring elements of radius r = 2.5, 5.0, 10.0

Grayscale Dilation and Erosion



- Grayscale dilation and erosion with various free-form structuring elements

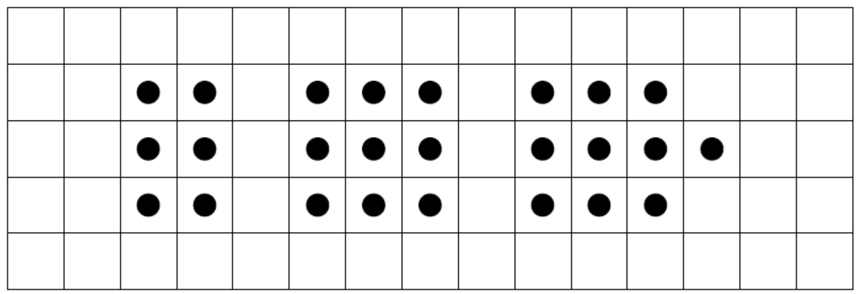
Grayscale Opening and Closing



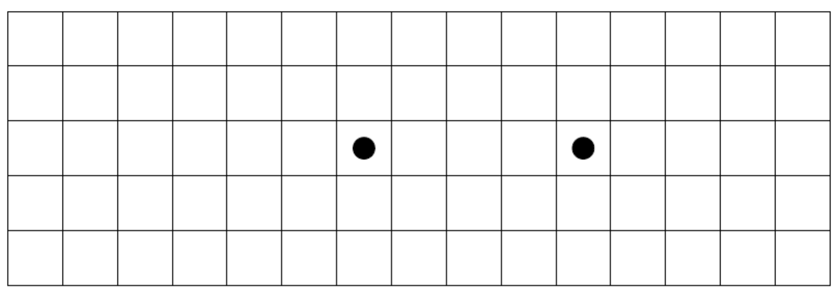
- Grayscale opening and closing with disk-shaped structuring elements of radius r = 2.5, 5.0, 10.0

Hit-or-Miss Transform

- Powerful method for finding shapes in images
- Can be defined in terms of erosion
- Suppose we want to locate 3x3 square shapes (in image center below)



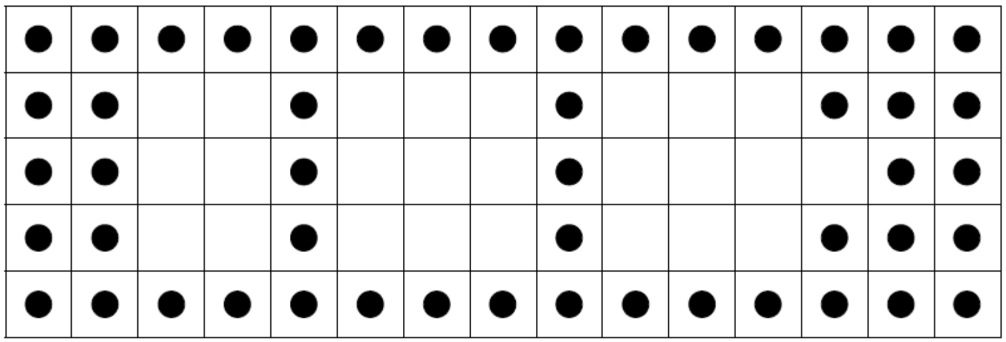
- If we perform an erosion $A \ominus B$ with *B* being the square element, result is:



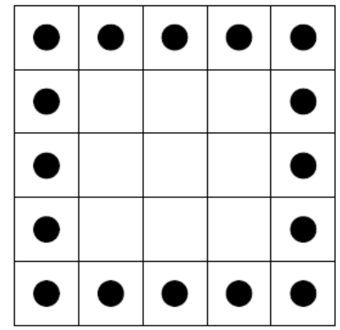
Hit or Miss Transform

- If we erode the complement of *A*, with a structuring element *C* that fits around 3x3 square

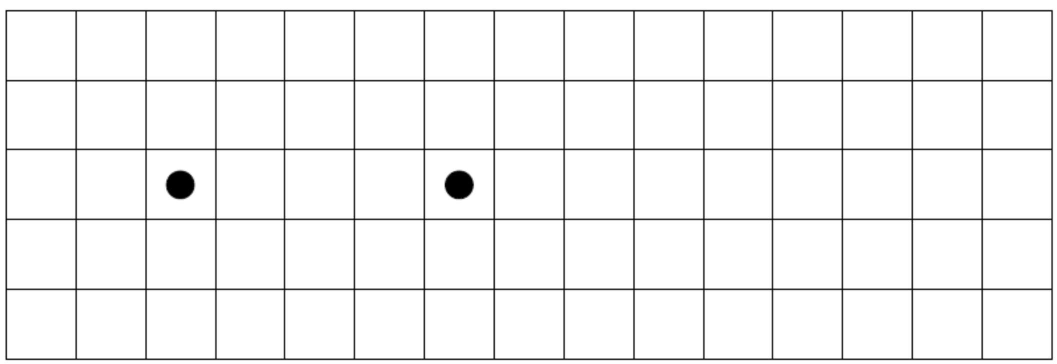
$\overline{A}$:



C:



- Result of $\overline{A} \ominus C$ is



- Intersection of 2 erosion operations produces 1 pixel at center of 3x3 square, which is what we want **(hit or miss transform)**

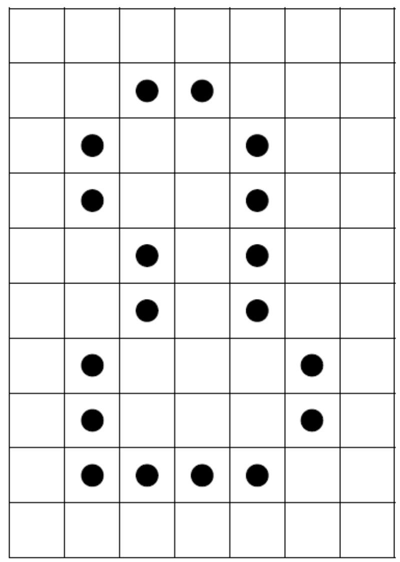
Hit-or-Miss Transform: Generalized

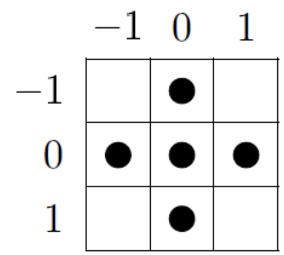
- If we are looking for a particular shape in an image, design 2 structuring elements:
  - B_1 which is same as shape we are looking for, and
  - B_2 which fits around the shape
  - We can then write B = (B_1, B_2)
- The hit-or-miss transform can be written as:

$$A \circledast B = (A \ominus B_1) \cap (\overline{A} \ominus B_2)$$

Morphological Algorithms: Region Filling

- Suppose an image has an 8-connected boundary





B

- Given a pixel **p** within the region, we want to fill region
- To do this, start with **p**, and dilate as many times as necessary with the cross-shaped structuring element **B**

Region Filling

- After each dilation, intersect with $\bar{A}$ before continuing
- We thus create the sequence:

$$\{p\} = X_0, X_1, X_2, \ldots, X_k = X_{k+1}$$

for which

$$X_n = (X_{n-1} \oplus B) \cap \overline{A}.$$

- Finally $X_k \cup A$ is the filled region

$\overline{A}$:

●	●	●	●	●	●	●
●	●			●	●	●
●		●	●		●	●
●		●	●		●	●
●	●		●		●	●
●	●		●		●	●
●		●	●	●		●
●		●	●	●		●
●					●	●
●	●	●	●	●	●	●

		6	5			
		5	4			
			3			
			2			
		2	1	2		
		1	p	1		

Connected Components

- We use similar algorithm for connected components
  - Cross-shaped structuring element for 4-connected components
  - Square-shaped structuring element for 8-connected components
- To fill rest of component by creating sequence of sets

$$X_0 = \{p\}, X_1, X_2, \ldots$$

such that

$$X_n = (X_{n-1} \oplus B) \cap A$$

until $X_k = X_{k-1}$.

- Example:

●	●		●	●	
●	●	●		●	
			●	●	●
●	●	●			
●	●	●			
●	●	●			

2	1	2			
1	p	1			
2	1	2			

Using the cross

5	4		4	4	
5	4	3		3	
			2	3	4
1	1	1			
1	p	1			
1	1	1			

Using the square

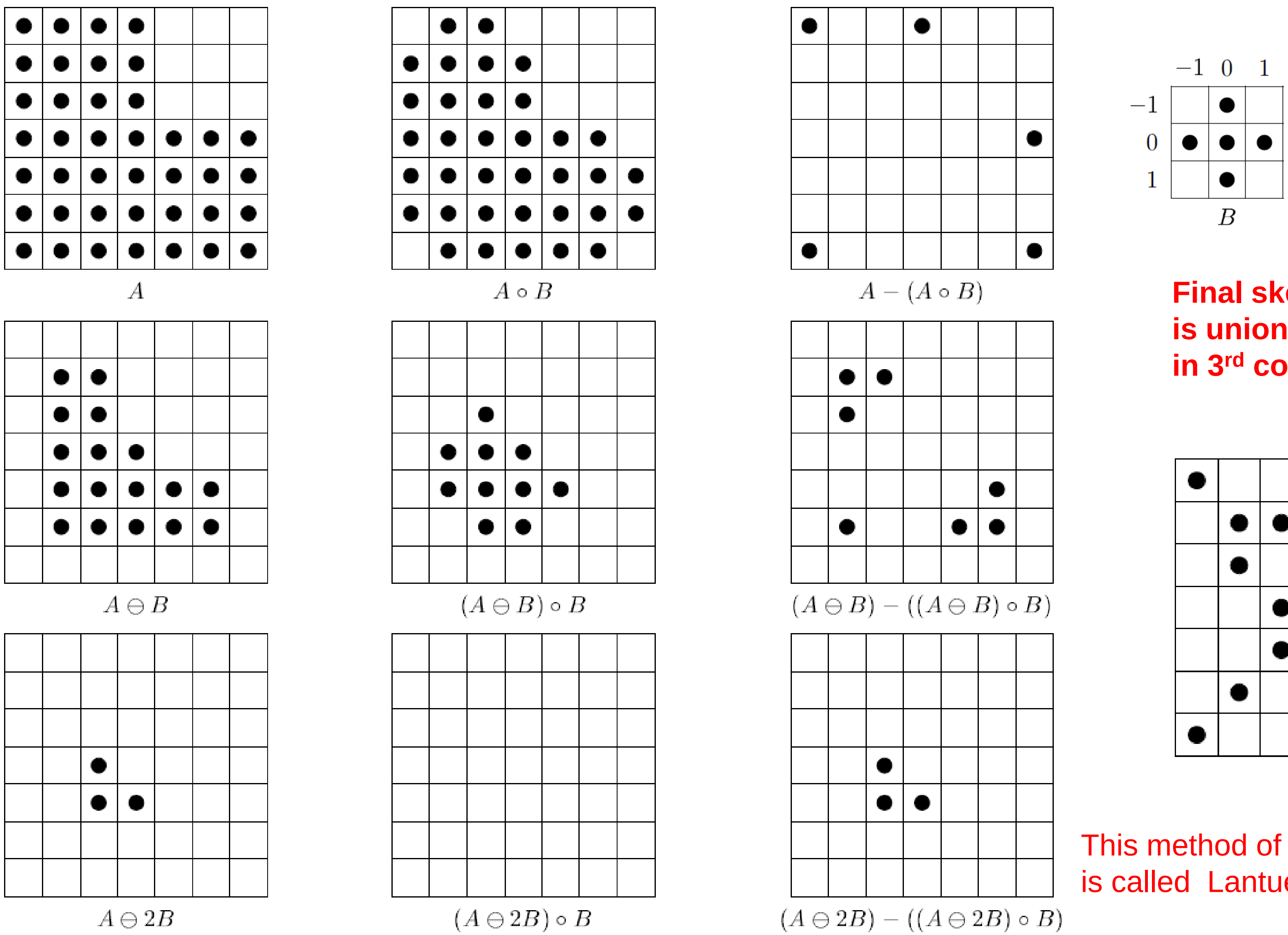
Skeletonization

- Table of operations used to construct skeleton

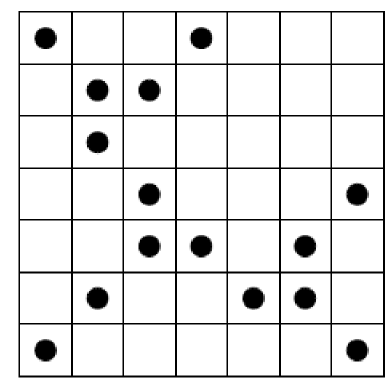
Erosions	Openings	Set differences
A	$A \circ B$	$A - (A \circ B)$
$A \ominus B$	$(A \ominus B) \circ B$	$(A \ominus B) - ((A \ominus B) \circ B)$
$A \ominus 2B$	$(A \ominus 2B) \circ B$	$(A \ominus 2B) - ((A \ominus 2B) \circ B)$
$A \ominus 3B$	$(A \ominus 3B) \circ B$	$(A \ominus 3B) - ((A \ominus 3B) \circ B)$
$\vdots$	$\vdots$	$\vdots$
$A \ominus kB$	$(A \ominus kB) \circ B$	$(A \ominus kB) - ((A \ominus kB) \circ B)$

- Notation, sequence of *k* erosions with same structuring element: $A \ominus kB$
- Continue table until $(A \ominus kB) \circ B$ is empty
- Skeleton is union of all set differences

Skeletonization Example



Final skeletonization is union of all entries in 3rd column



This method of skeletonization is called Lantuéjoul's method

References

- Wilhelm Burger and Mark J. Burge, Digital Image Processing, Springer, 2008
- University of Utah, CS 4640: Image Processing Basics, Spring 2012
- Rutgers University, CS 334, Introduction to Imaging and Multimedia, Fall 2012
- Gonzales and Woods, Digital Image Processing (3rd edition), Prentice Hall