GG
GRAPHICS GROUP

IVS
Intelligent
Vision Systems
New Zealand

Computer Graphics and Image Processing

Part 3: Image Processing

6 – Morphological Operations

Martin Urschler, PhD

Morphological operations

- Main concepts
- Structuring element
- Erosion
- Dilation
- Morphological gradients
- Opening
- Closing

Additional resources and sources of some images:
http://www.inf.u-szeged.hu/ssip/1996/morpho/morphology.html
http://www.cs.ru.nl/~ths/rt2/col/h11/11morphENG.html

Basic concepts

Morphology: a study of structure or form.

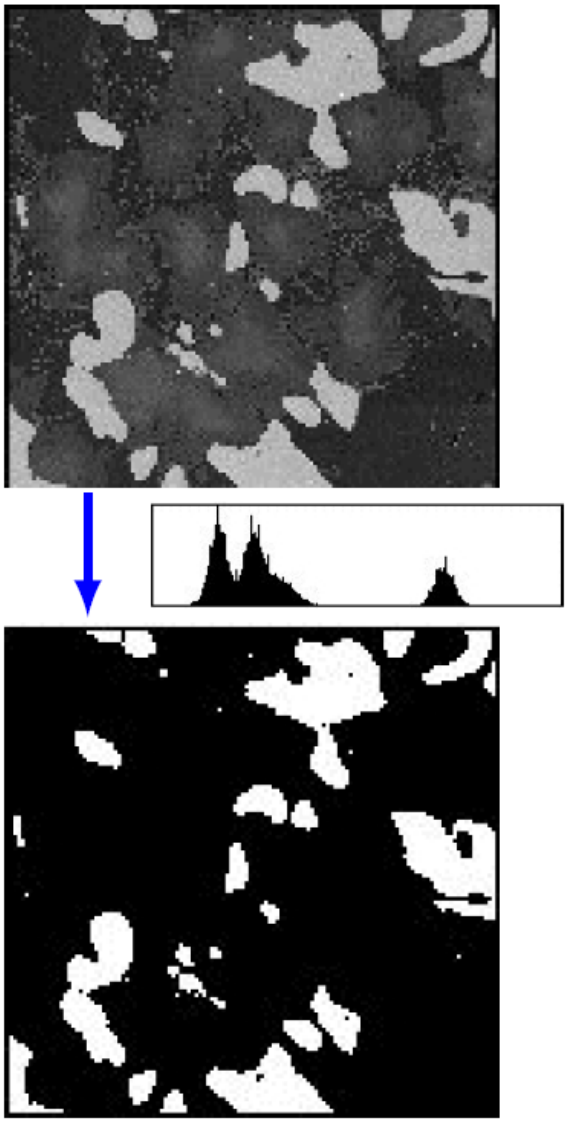
(http://www.merriam-webster.com/dictionary/morphology)

Morphological image processing may remove imperfections of a binary image.

- Regions in binary images produced by simple thresholding are typically distorted by noise.

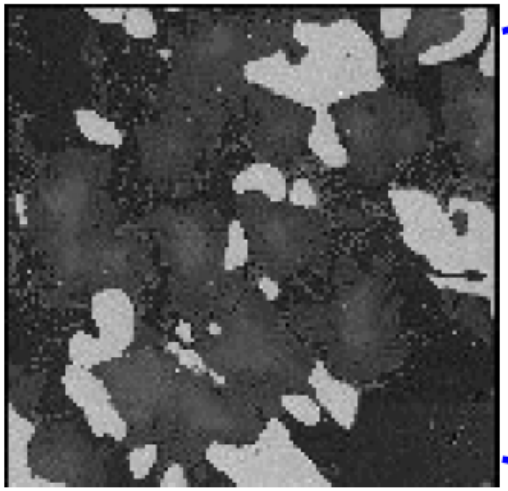
Morphological operations are non-linear and account for structure and forms of regions (objects) to improve an image.

- These operations can be extended to greyscale images.

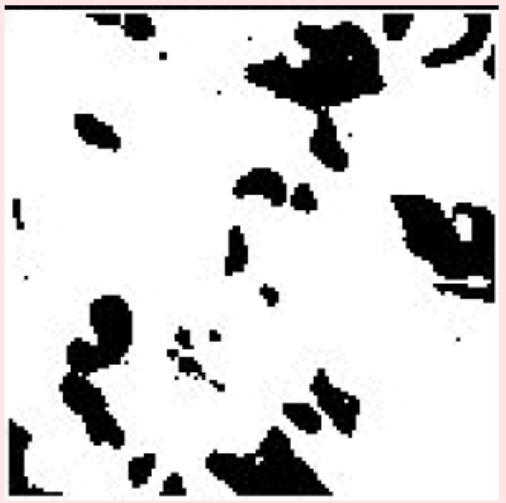


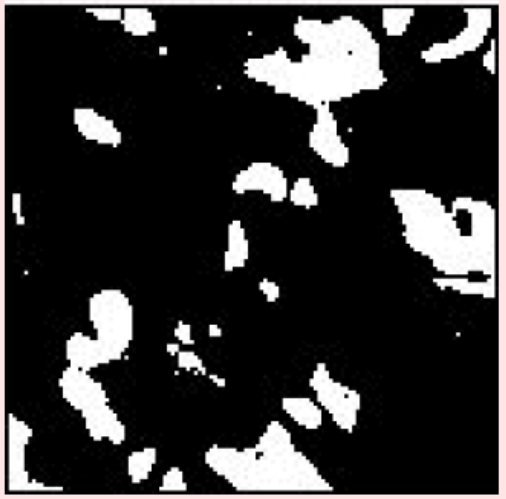
Binary images: object/background

White $\underbrace{\text{background}}_{0}$ – black foreground $\underbrace{\text{objects}}_{1}$ $\Rightarrow$



For morphological processing, the foreground (i.e. object) and background pixels are treated as 1s and 0s, respectively, independently of their visual black-white coding.



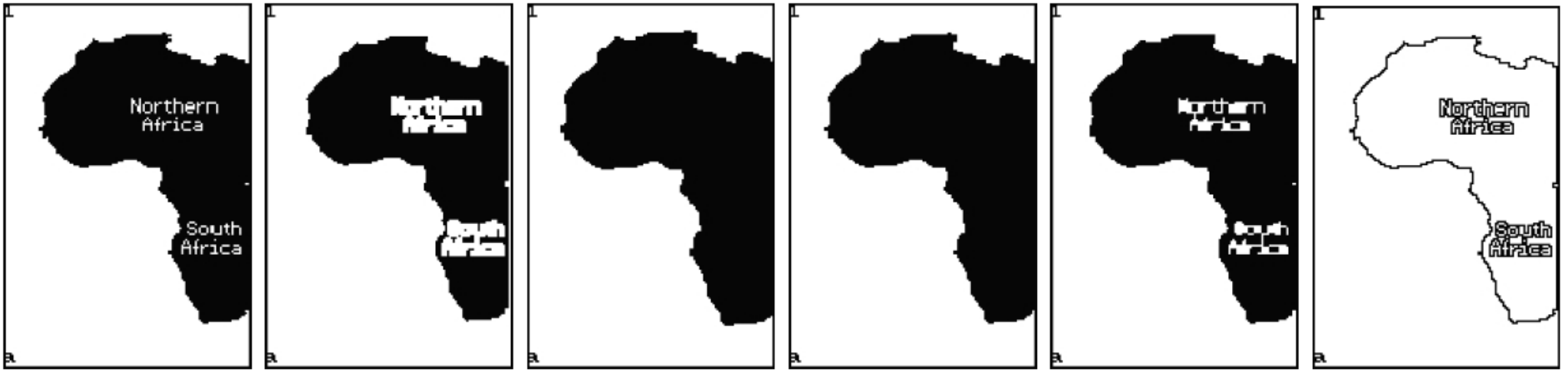


Black $\overbrace{\text{background}}^{0}$ – white foreground $\overbrace{\text{objects}}^{1}$ $\Rightarrow$

Morphological operations: examples

White objects on black background: "1" / "0" as white / black

Image | Erosion | Dilation | Closing | Opening | Gradient



Black objects on white background: "1" / "0" as black / white

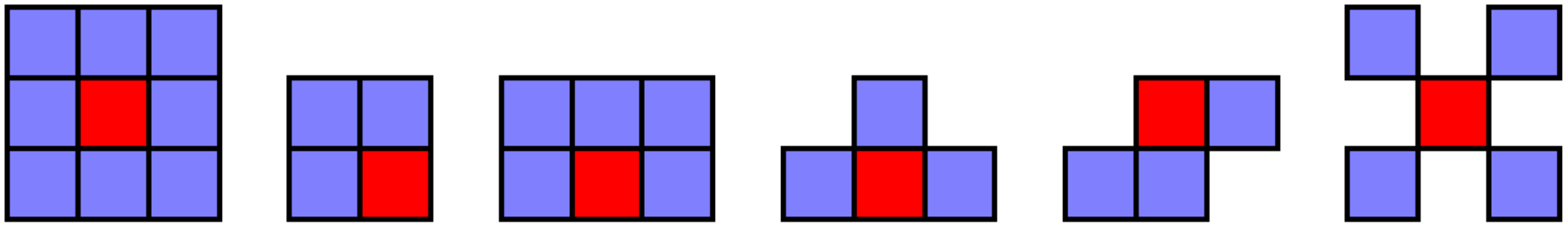
Basic properties

Morphological operations rely on relative ordering of pixel values, not on their numerical values.

- Thus the operations are especially suited to binary image processing.
- These operations can be applied also to greyscale images such that their absolute pixel values are of no or minor interest.

Morphological operations probe an image with a small shape or template called a **structuring**, or structure **element** (SE).

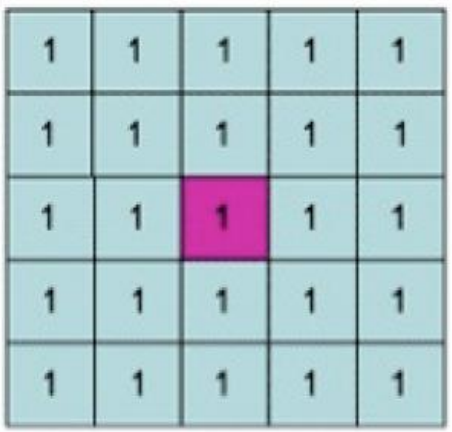
- SE resembles a convolution kernel in linear filtering.



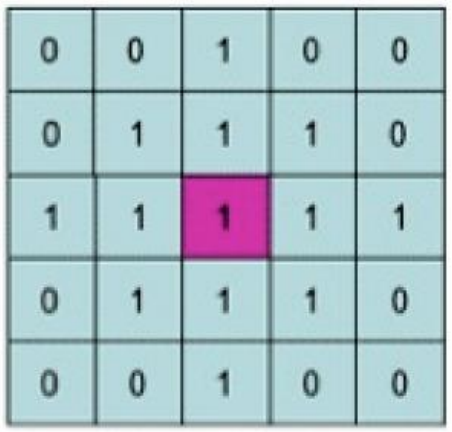
Structuring element

A small binary image, i.e. a small matrix of pixels, each with a value of zero (0) or one (1).

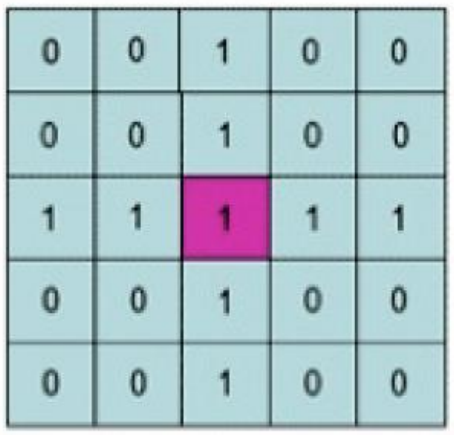
- Zero-valued pixels of the SE are ignored.
- **Size** of the SE: the matrix dimensions.
- **Shape** of the SE: the pattern of ones and zeros.
- **Origin** of the SE: usually, one of its pixels.
  - Generally, the origin can be also outside the matrix.



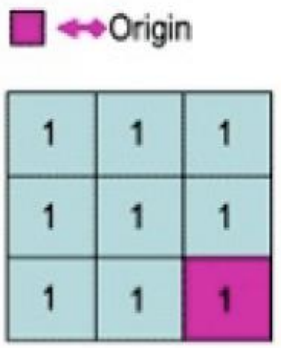
Square 5x5 element



Diamond-shaped 5x5 element



Cross-shaped 5x5 element

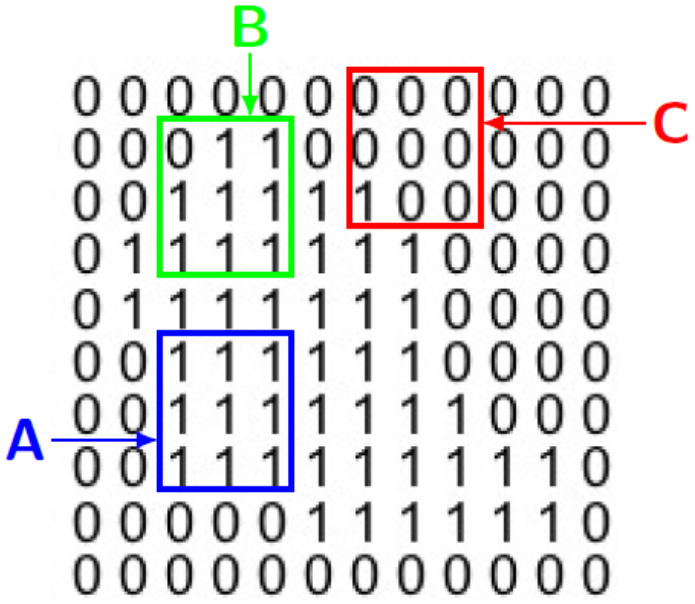


Square 3x3 element

Structuring element

When an SE is placed in a binary image, each its pixel is associated with the pixel of the area under the SE:

- The SE **fits** the image if **for each** of its pixels set to 1 the corresponding image pixel is also 1.
- The SE **hits** (intersects) the image if **at least for one** of its pixels set to 1 the corresponding image pixel is also 1.



s_1

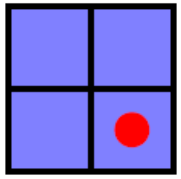
1 1 1
1 1 1
1 1 1

s_2

0 1 0
1 1 1
0 1 0

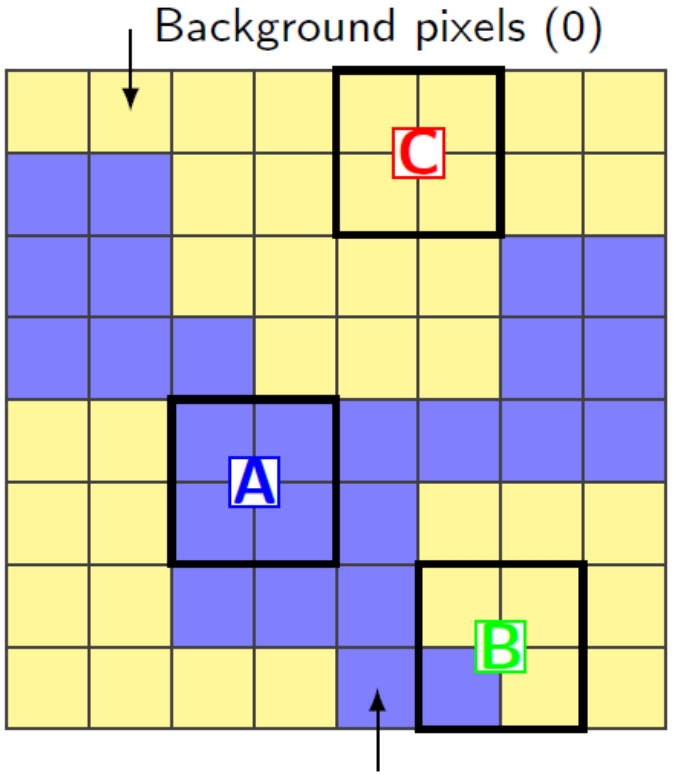
		A	B	C
fit	s_1	yes	no	no
	s_2	yes	yes	no
hit	s_1	yes	yes	yes
	s_2	yes	yes	no

Probing with an SE



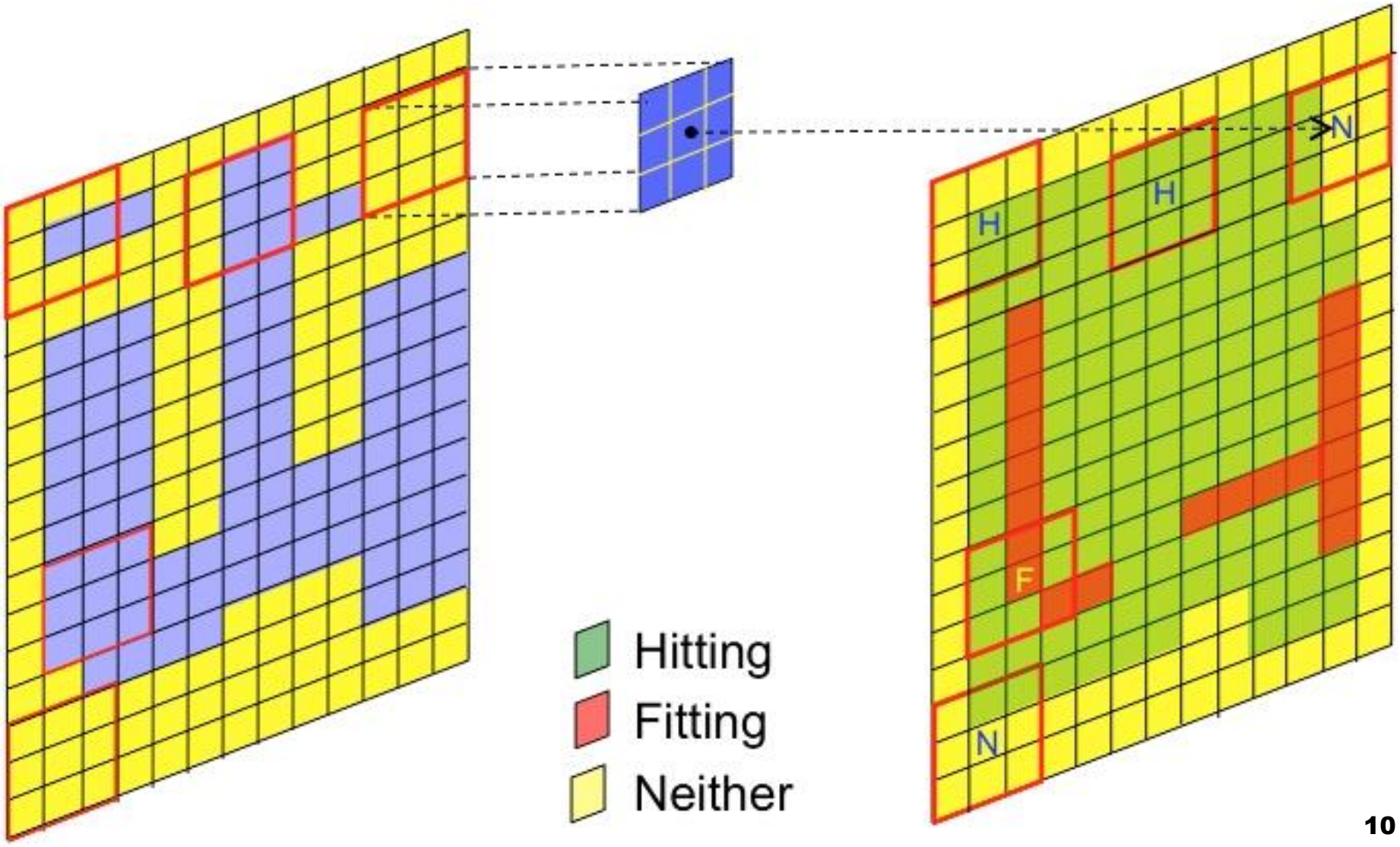
SE

A: SE **fits** the foreground (object);
B: SE **hits** (intersects with) the object;
C: SE neither fits, nor hits the object.



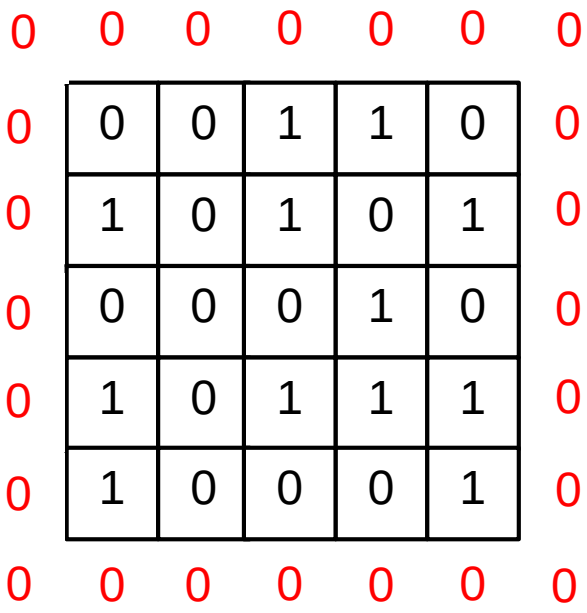
- SE is positioned at all possible locations in an image and compared with the pixel neighbourhood at each location.
- Operations test whether the SE "fits" within the neighbourhood or "hits" the neighbourhood.
- An operation on a binary image creates a new binary image in which the pixel has a foreground value only if the test at that place in the input image is successful.

Probing with an SE

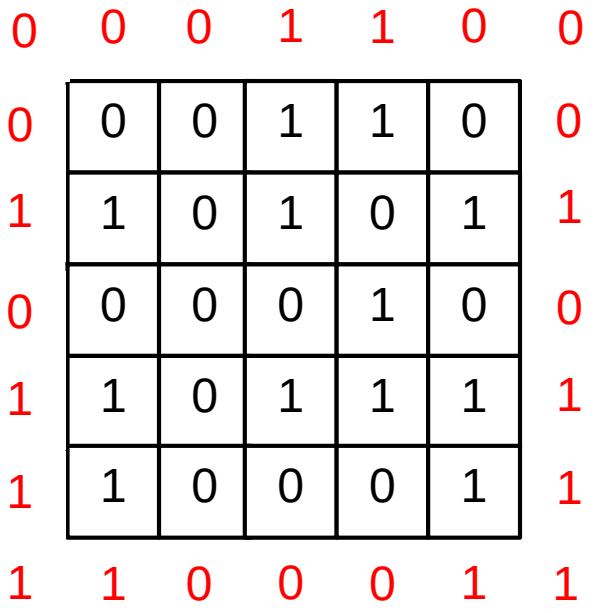


Boundary handling

- Mainly two strategies for padding



Extend image boundary with zeros
(Coderunner quiz *strategy 0*)



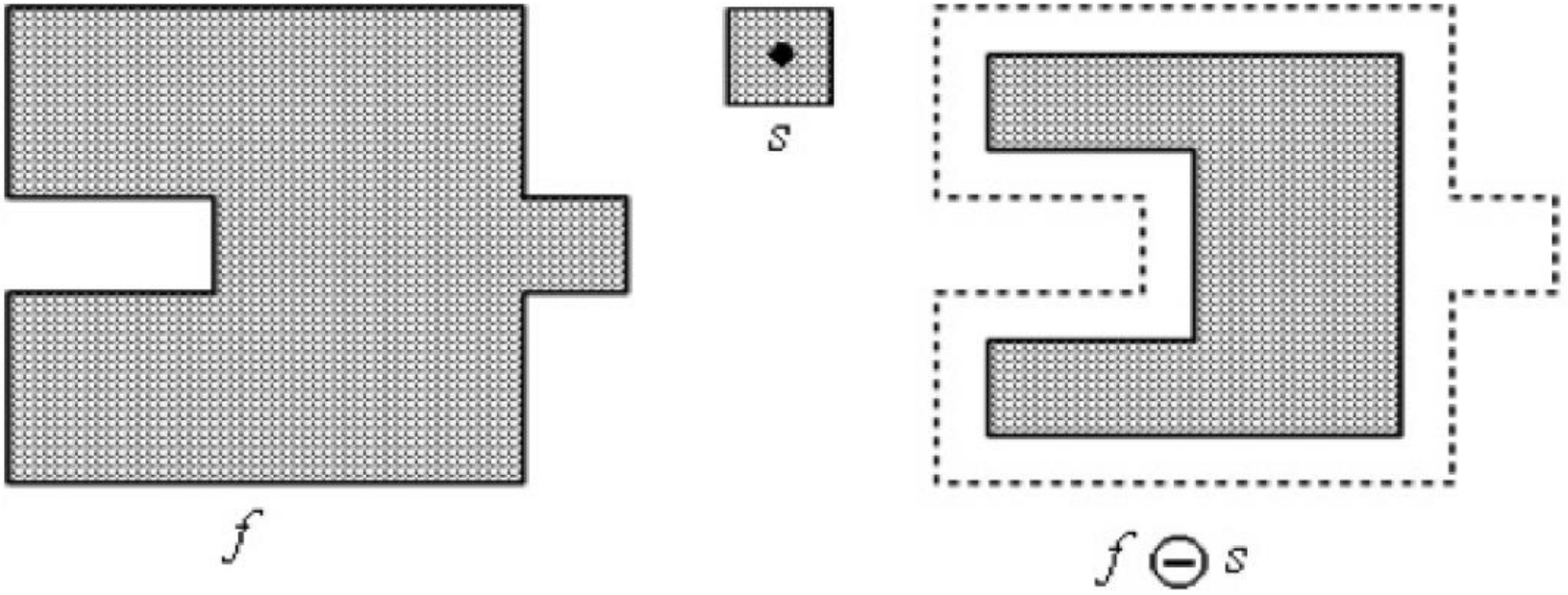
Extend image boundary with boundary values
(Coderunner quiz *strategy 1*)

Fundamental operation 1: Erosion

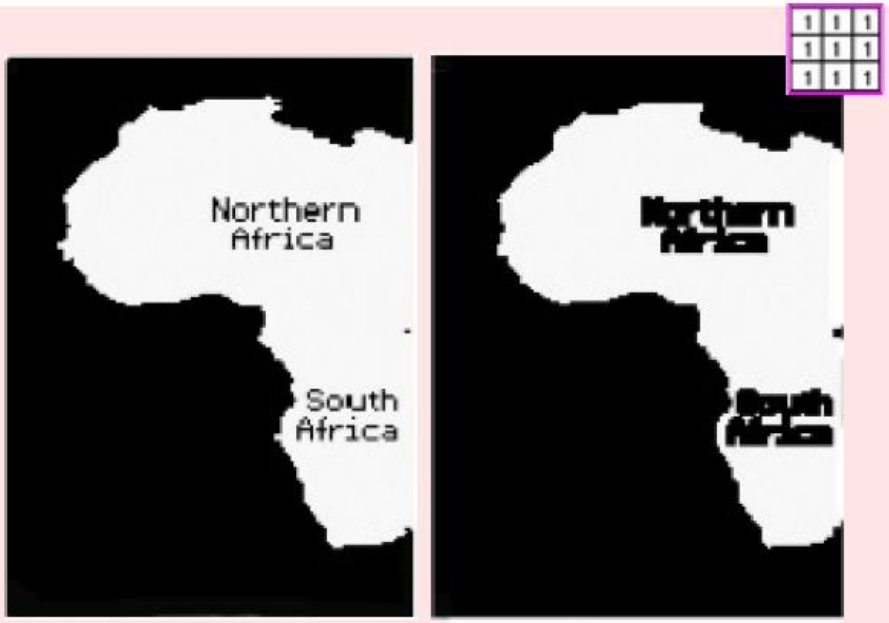
Erosion $f \ominus s$ of a binary image f by an SE s produces a new binary image $g = f \ominus s$.

Eroded image g has ones in all locations (x, y) of an origin of the SE s at which s fits the input image f.

For all pixel coordinates (x, y), $g(x, y) = 1$ if s fits f and 0 otherwise.

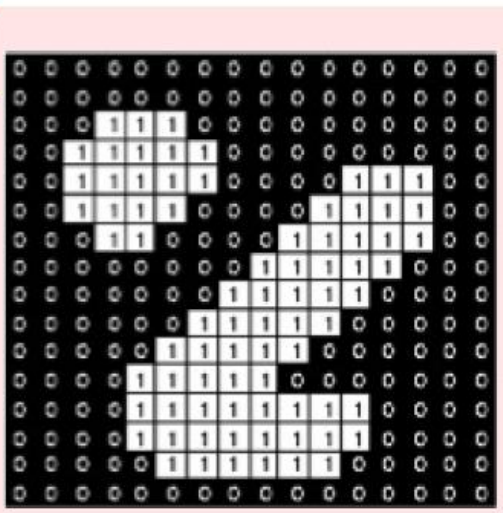


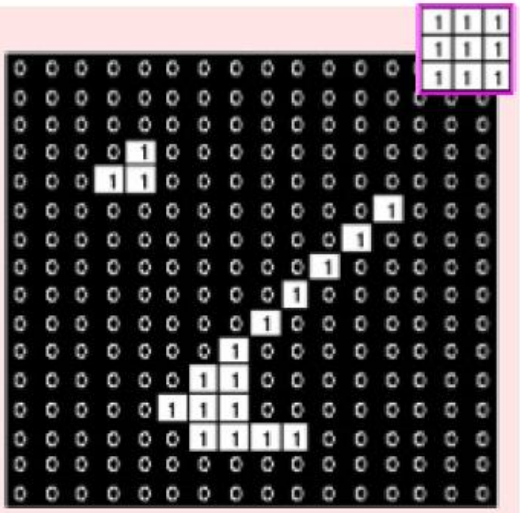
Erosion examples



Binary image

Eroded image





- Erosion with small (e.g. $2 \times 2 - 5 \times 5$) square SEs shrinks an image by stripping away a layer of pixels from both inner and outer region boundaries.
- Holes in and gaps between foreground objects become larger, and small details are eliminated.

Erosion of a binary image

Larger SEs have a more pronounced effect.

- Erosion with a large SE is similar to an iterated erosion with a smaller SE of the same shape.
- If a pair of SEs s_1 and s_2 are identical in shape, with s_2 twice the size of s_1, then $f \ominus s2 \approx (f \ominus s_1) \ominus s_1$.

Erosion removes fine (small-scale) details, as well as noise from a binary image.

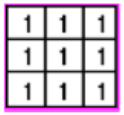
- Simultaneously, erosion reduces the size of regions of interest (objects), too.
- Background areas are growing, i.e. an image with the black or white background becomes blacker or whiter, respectively.

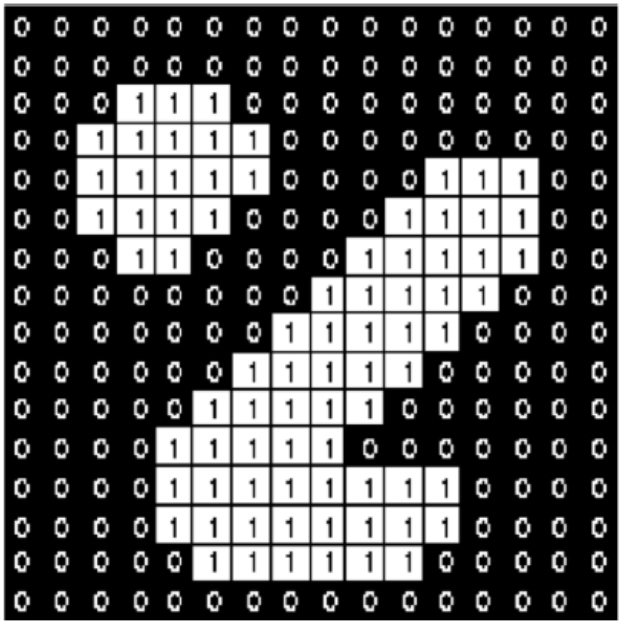
Boundary detection by erosion

Internal gradient $b = f - (f \ominus s)$ of each region:

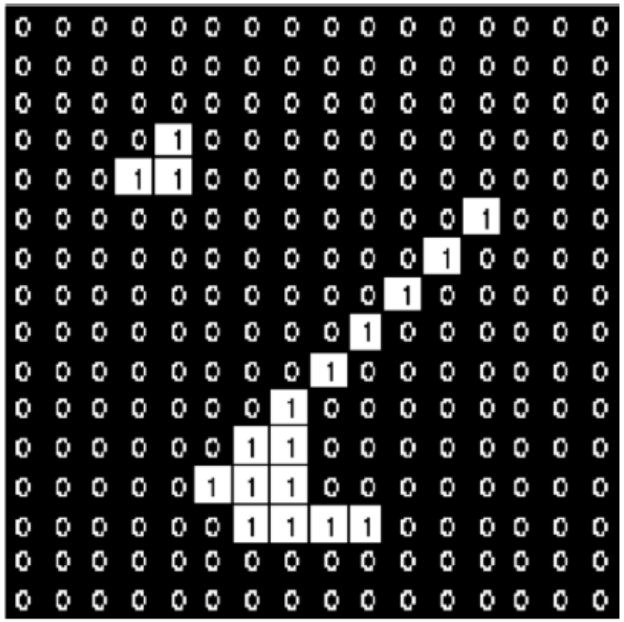
by subtracting the eroded image from an original image.

Gradient is computed with the 3×3 square SE:

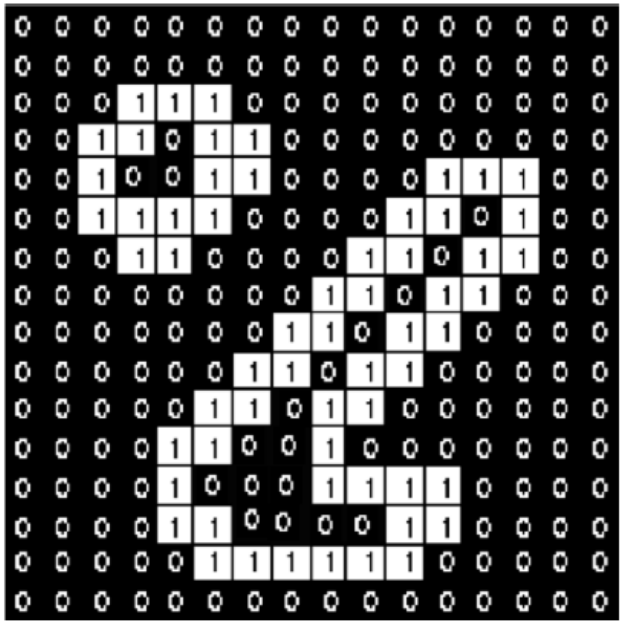




Binary image f



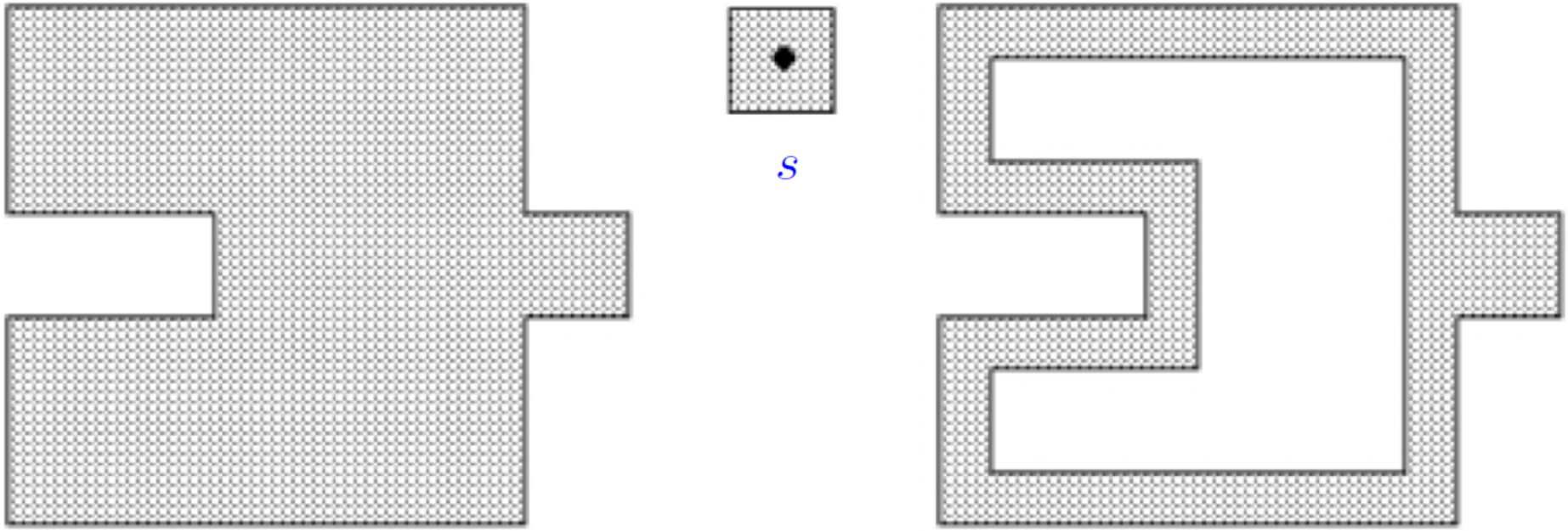
Eroded image $f \ominus s$



Boundary $b = f - (f \ominus s)$

Boundary detection by erosion

Internal gradient of each region:

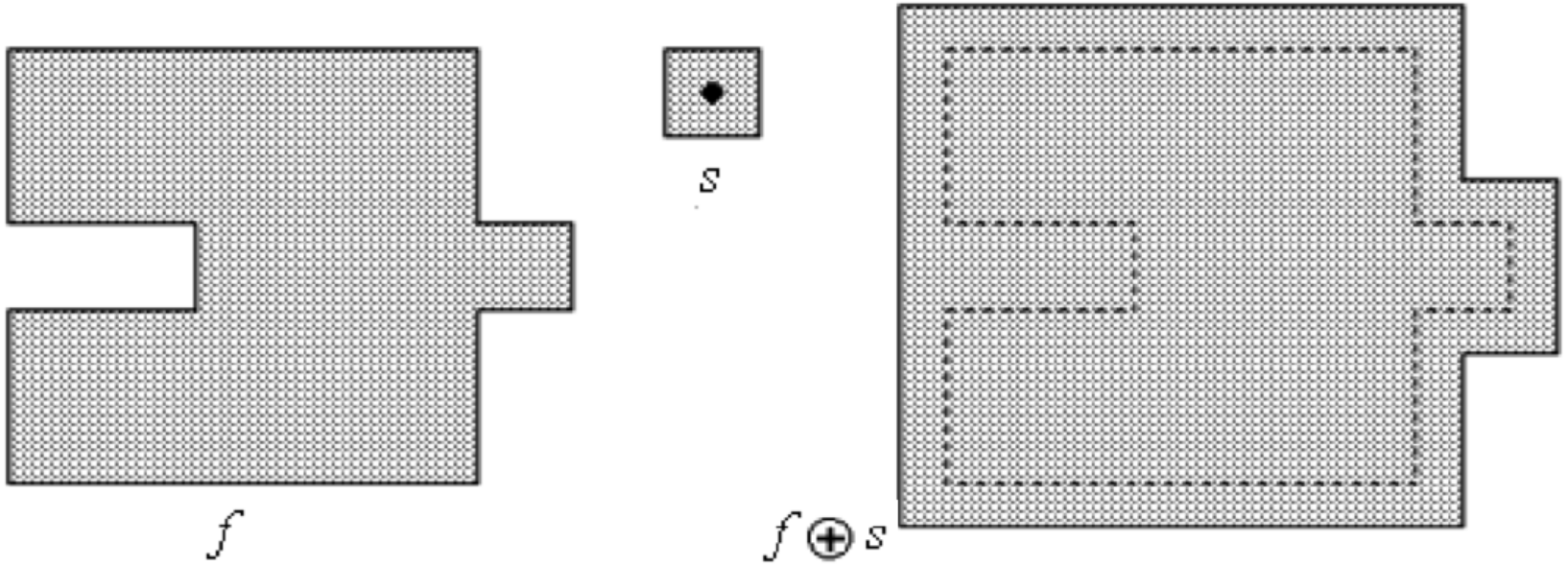


Fundamental operation 2: Dilation

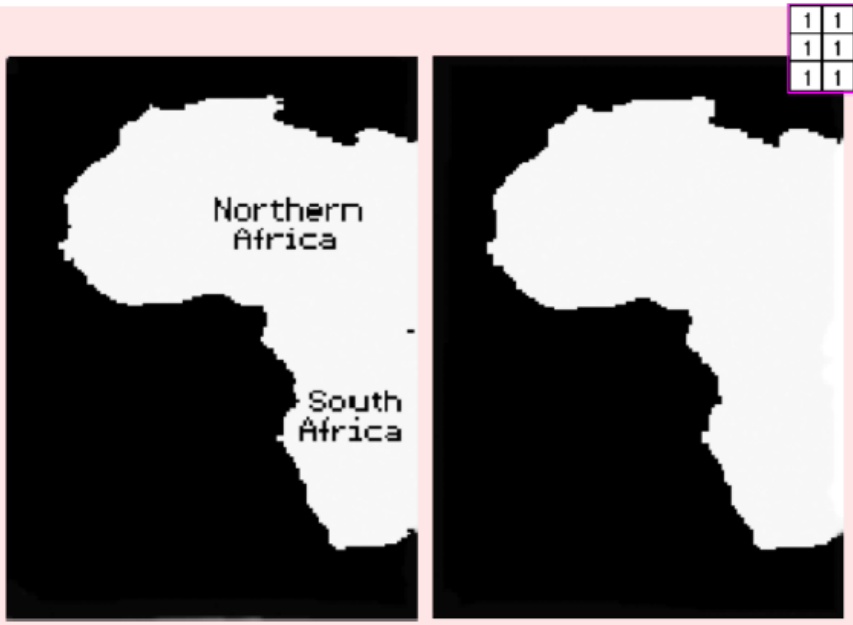
Dilation $f \oplus s$ of a binary image f by an SE s produces a new binary image $g = f \oplus s$.

Dilated image g has ones in all locations (x, y) of an origin of the SE s at which s <u>hits</u> the input image f.

For all pixel coordinates (x, y), $g(x, y) = 1$ if s hits f and 0 otherwise.

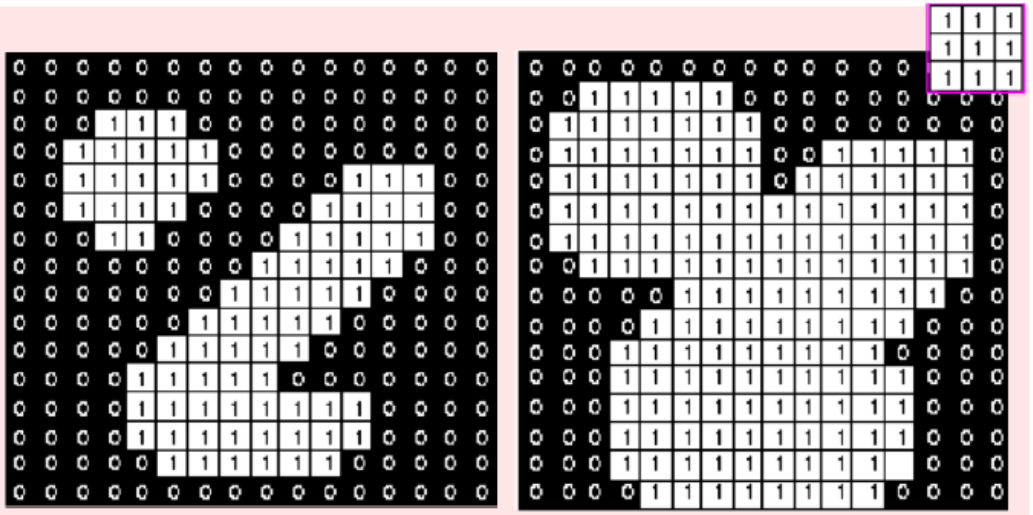


Dilation examples



Binary image

Dilated image

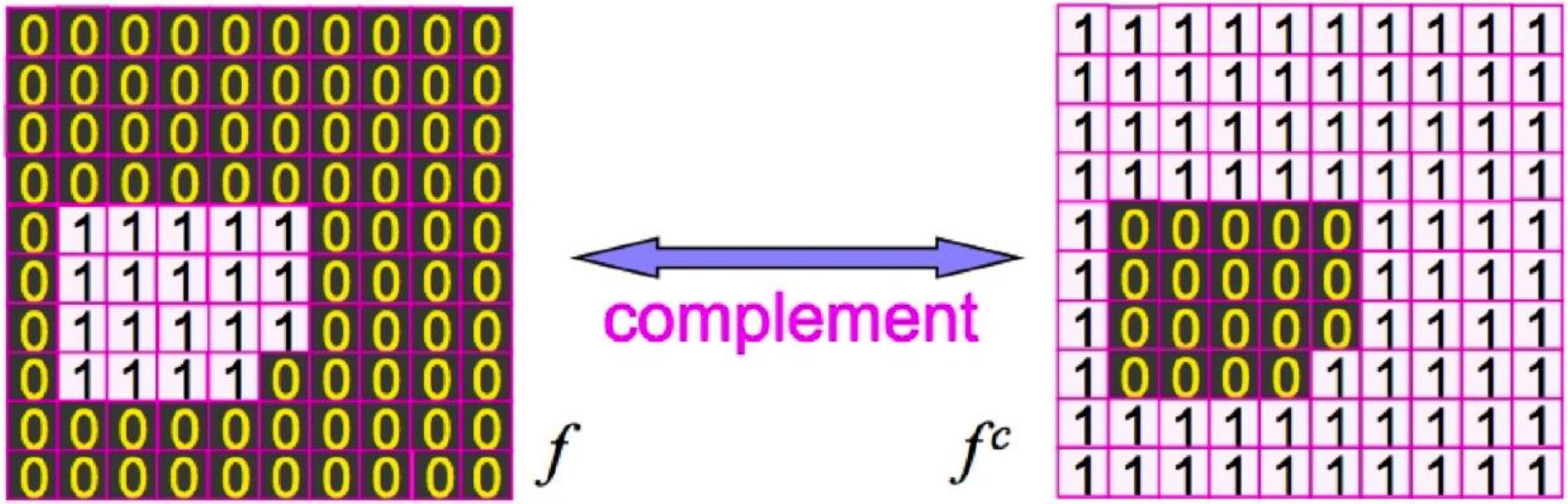


- Dilation is opposite to erosion: it adds a layer of pixels to the inner and outer boundaries of regions.
- Holes enclosed by a single region and gaps between different regions become smaller; small intrusions into boundaries of a region are filled in.

Set-theoretic binary operations

- Many morphological operations are combinations of erosion, dilation, and simple set-theoretic operations.
- Set-theoretic **complement** of a binary image:

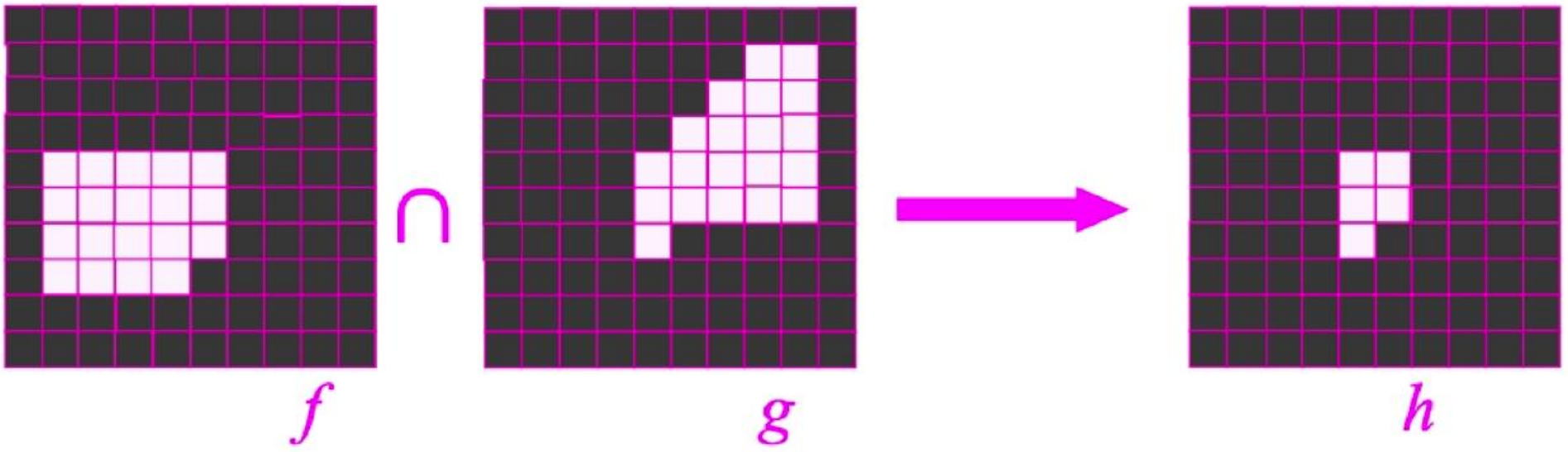
$$f^{c}(x,y) = \begin{cases} 1 & \text{if} \quad f(x,y) = 0 \\ 0 & \text{if} \quad f(x,y) = 1 \end{cases}$$



Set-theoretic operations: intersection

Intersection $h = f \bigcap g$ of two binary images f and g:

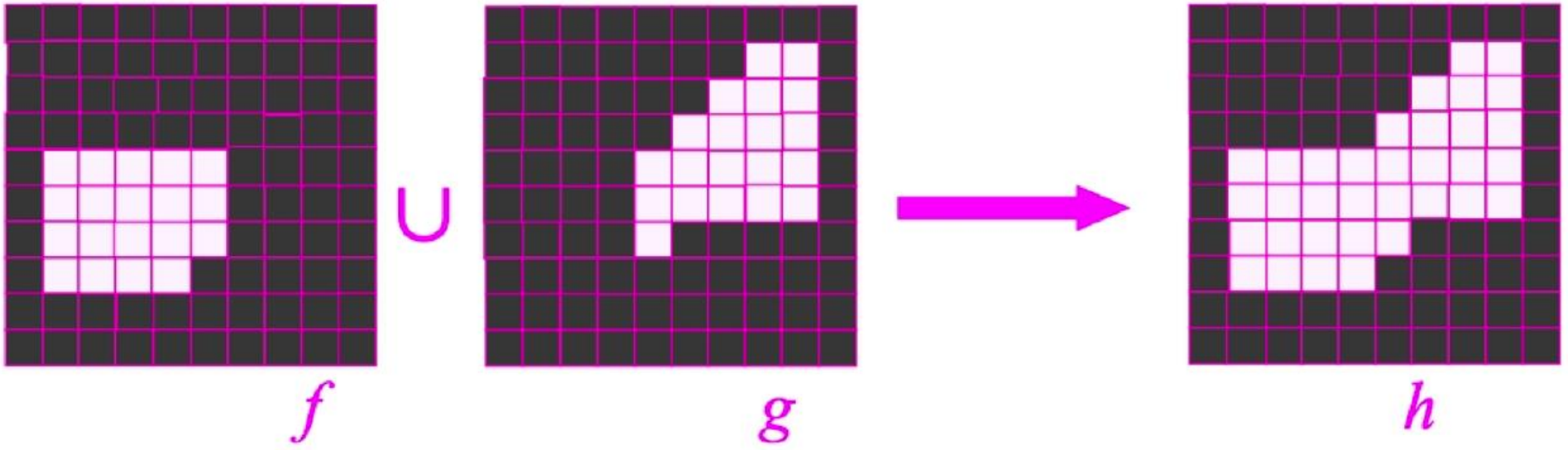
$$h(x,y) = \begin{cases} 1 & \text{if} \quad f(x,y) = 1 \ \ \textbf{AND} \ \ g(x,y) = 1 \\ 0 & \text{otherwise} \end{cases}$$



Set-theoretic operations: union

Union $h = f \bigcup g$ of two binary images f and g:

$$h(x,y) = \begin{cases} 1 & \text{if} \quad f(x,y) = 1 \ \ \mathbf{OR} \ \ g(x,y) = 1 \\ 0 & \text{otherwise} \end{cases}$$



Duality of dilation and erosion

Dilation and erosion are **dual operations** in that they have opposite effects: $f \oplus s = f^{\mathrm{c}} \ominus s_{\mathrm{rot}}$.

- f^{c} – the complement of f produced by replacing "1"s / "0"s with "0"s / "1"s.
- s_{rot} – the SE s rotated by $180°$.
  - If a SE is symmetric with respect to rotation, then s_{rot} does not differ from s.

If a binary image is considered as a collection of connected regions of "1"s on background of "0"s:

- **Erosion** is the fitting of the SE to these regions.
- **Dilation** is the fitting of the SE, rotated if necessary, into the background, followed by inversion of the result.

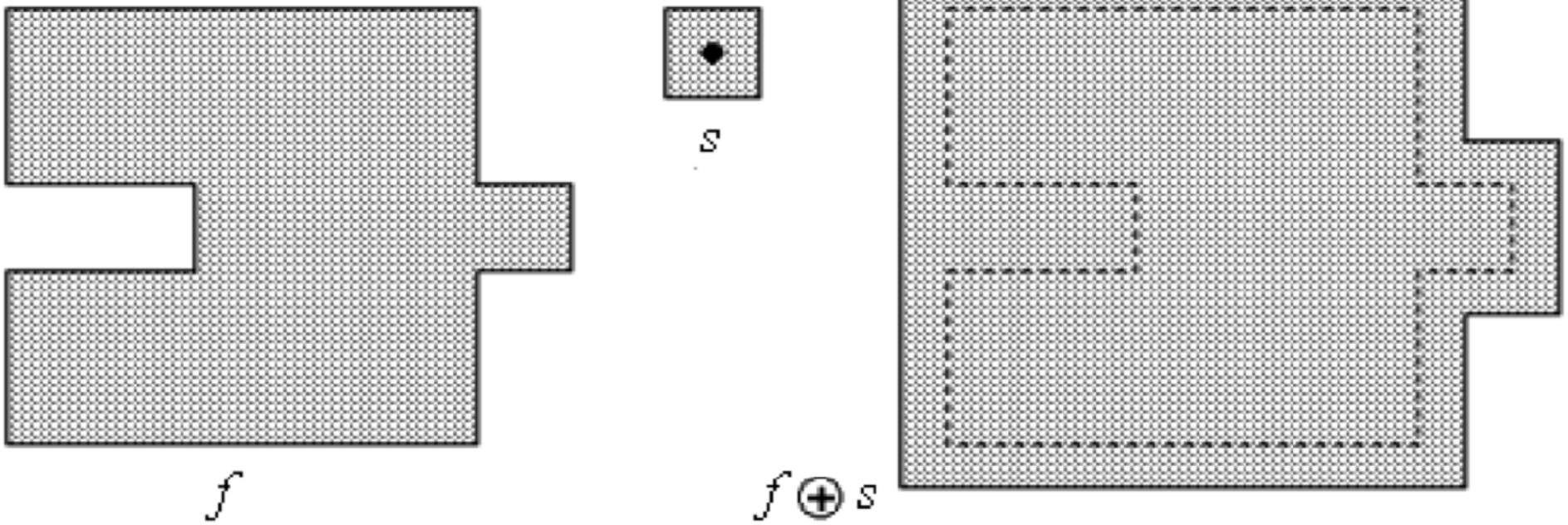
Boundary detection by dilation

External gradient $b = (f \oplus s) - f$ of each region:

by subtracting an original image from the dilated image.

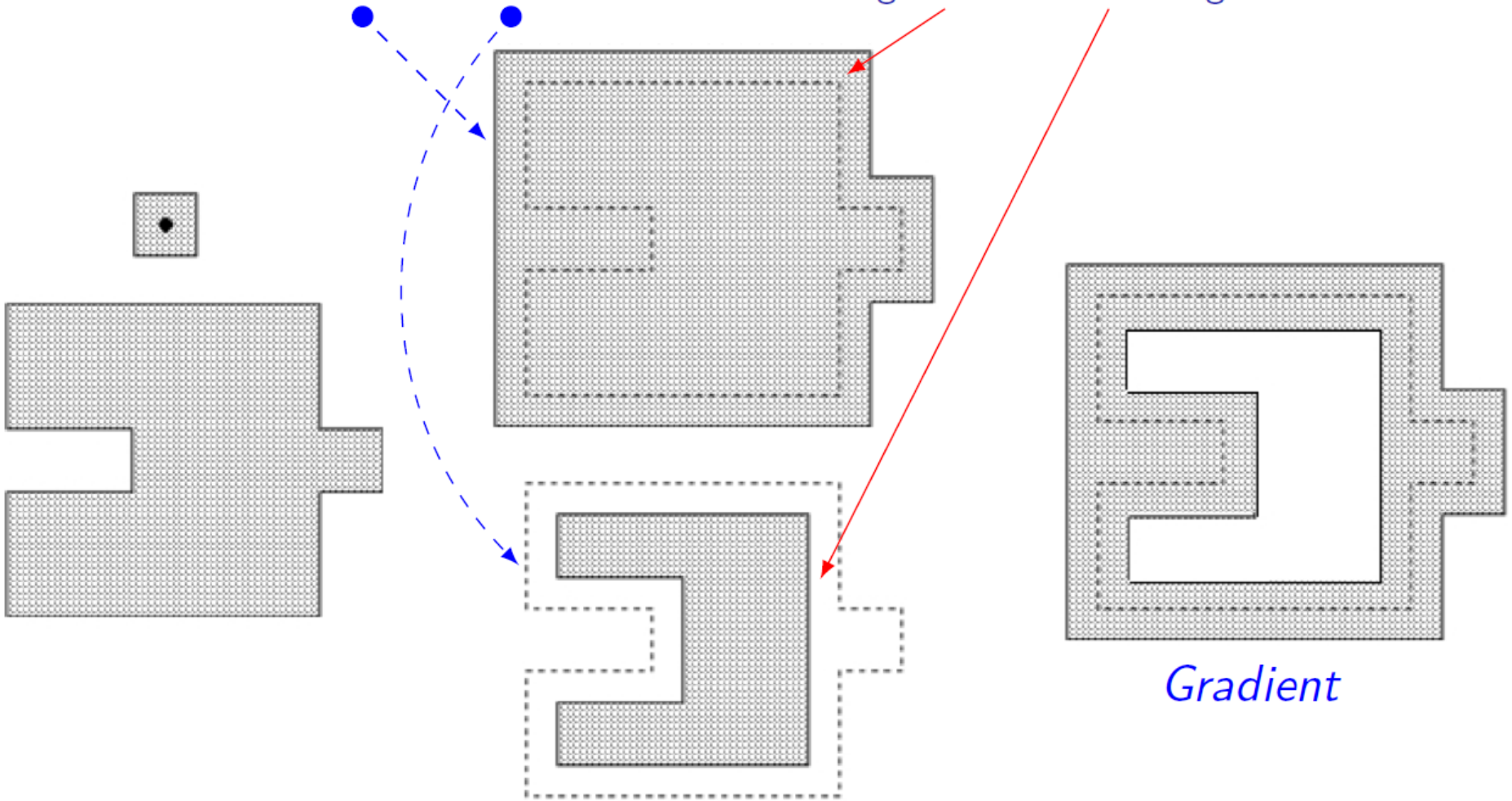
Gradient is computed with the 3×3 square SE:

1	1	1
1	1	1
1	1	1



Morphological gradient = Dilation - Erosion

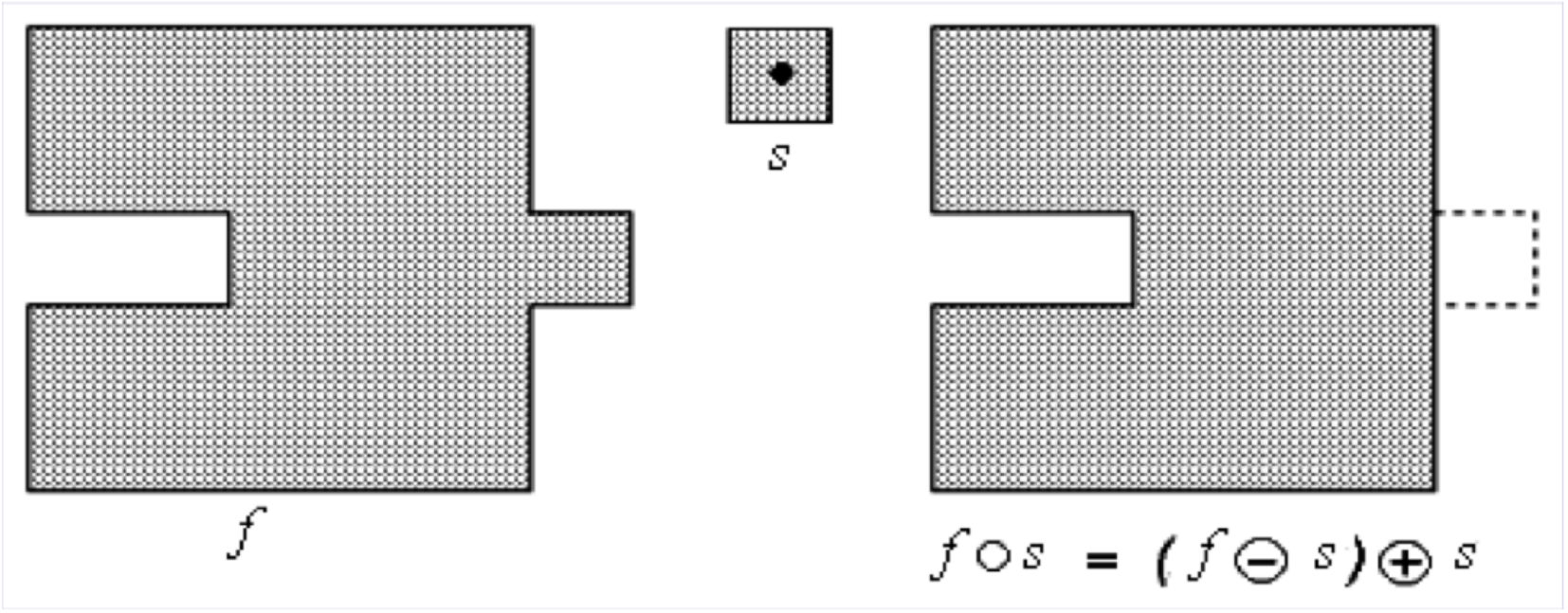
$$\nabla f = \underbrace{(f \oplus s)}_{\text{Dilation}} - \underbrace{(f \ominus s)}_{\text{Erosion}} \equiv \underbrace{[(f \oplus s) - f]}_{\text{External gradient}} + \underbrace{[f - (f \ominus s)]}_{\text{Internal gradient}}$$



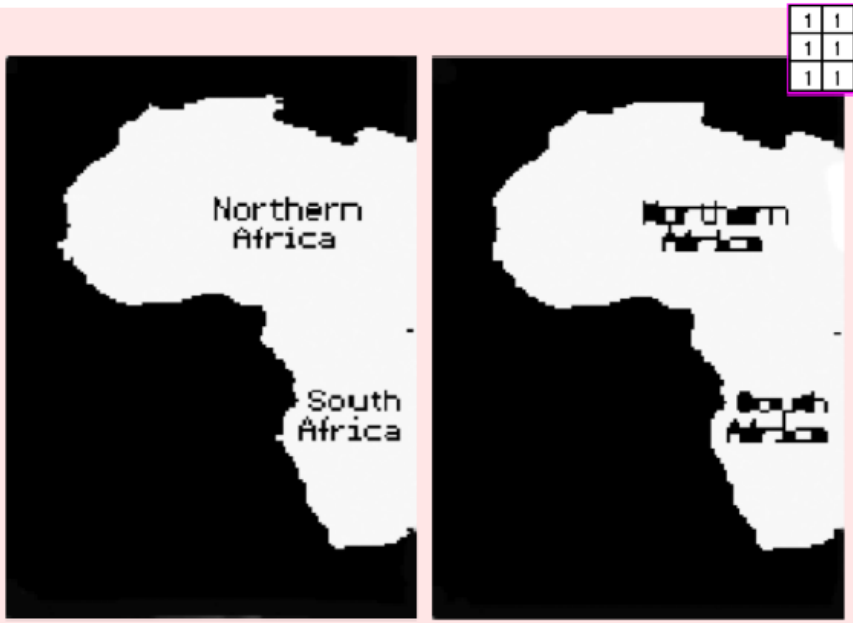
Opening

Opening $f \circ s$ of an image f by a structuring element s is an **erosion** followed by a **dilation**:

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

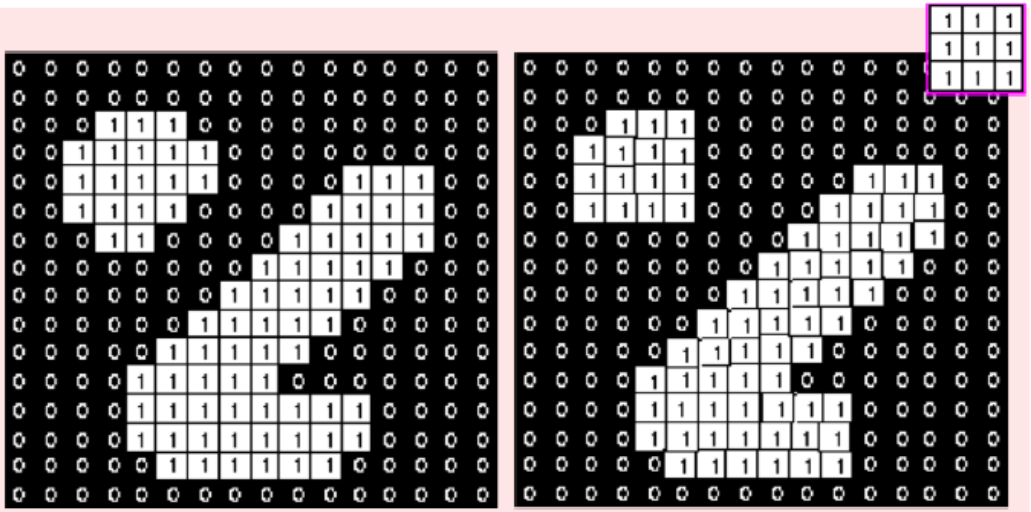


Opening examples



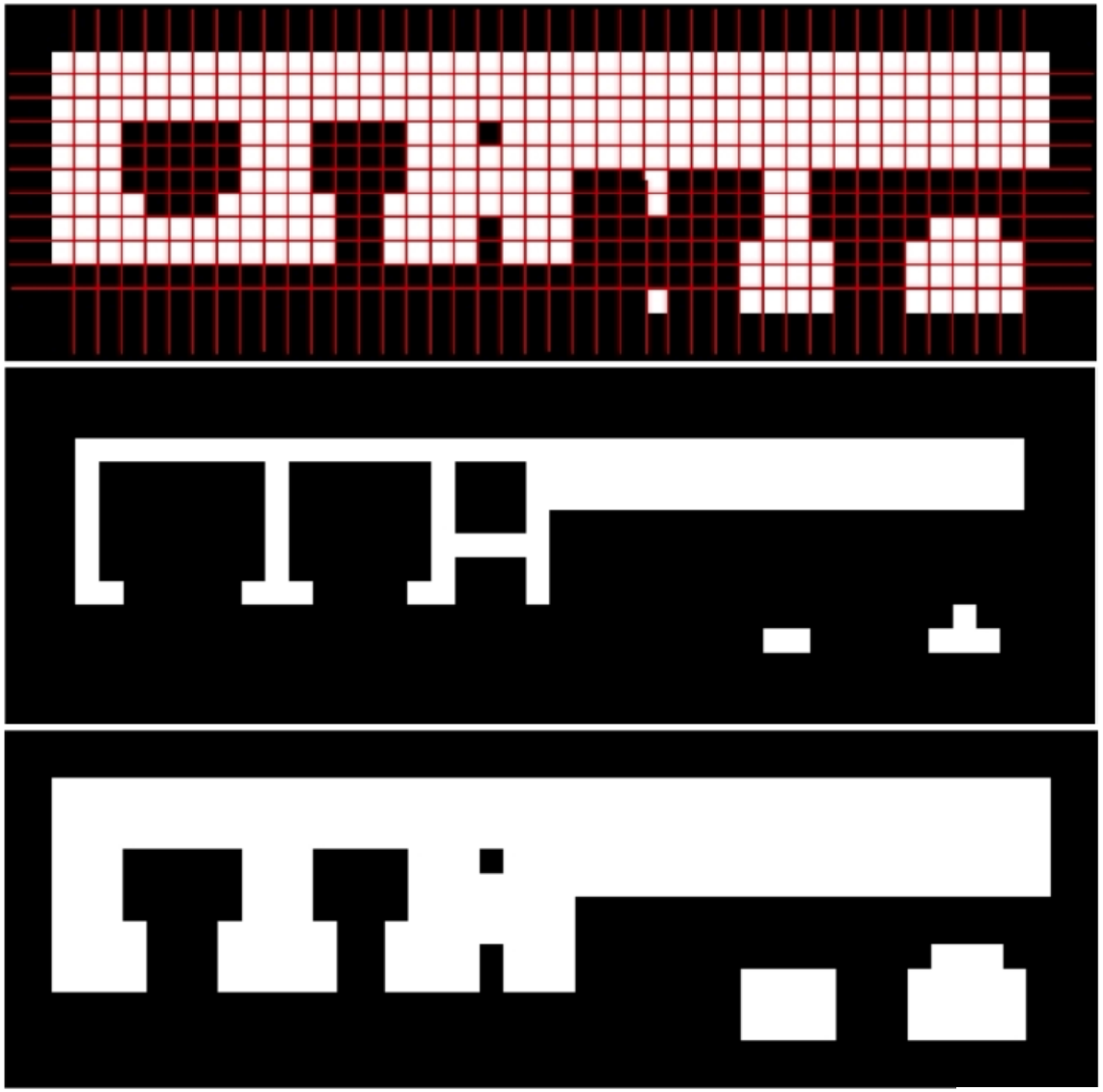
Binary image

Opened image



Opening is so called because it can open up a gap between objects connected by a thin bridge of pixels.

Opening examples



Initial image f:
white "1"s and black "0"s

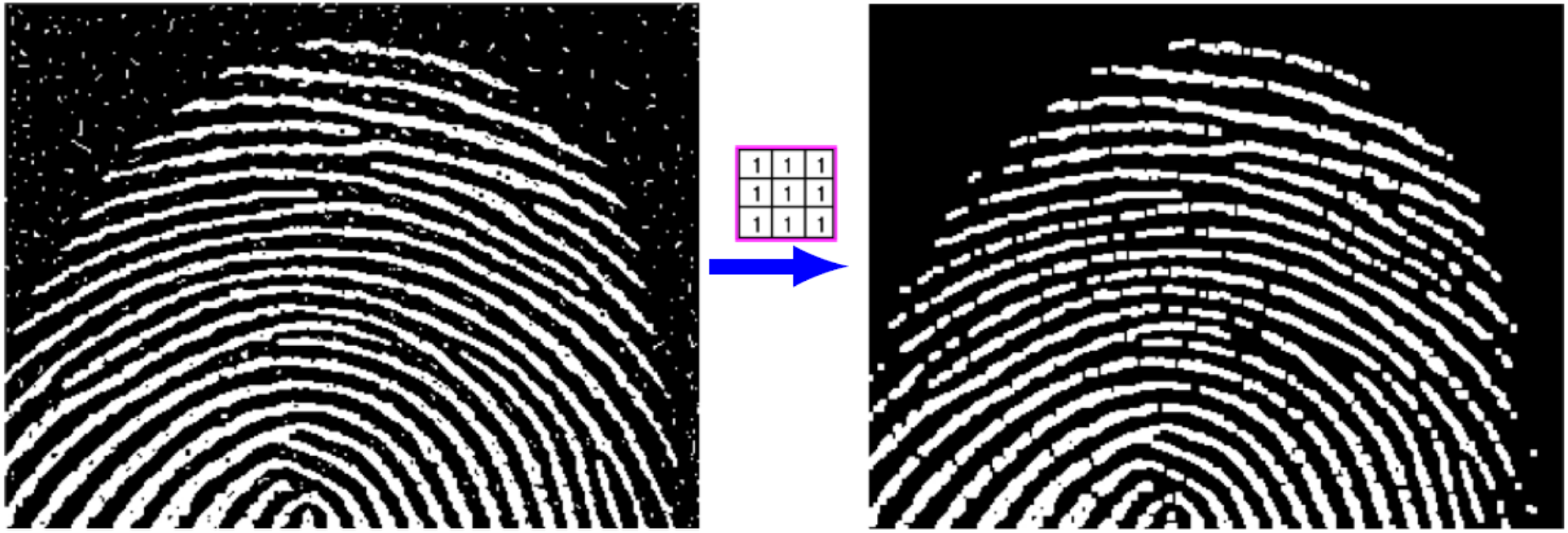
SE s 

Erosion $f \ominus s$

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

Opening a binary image

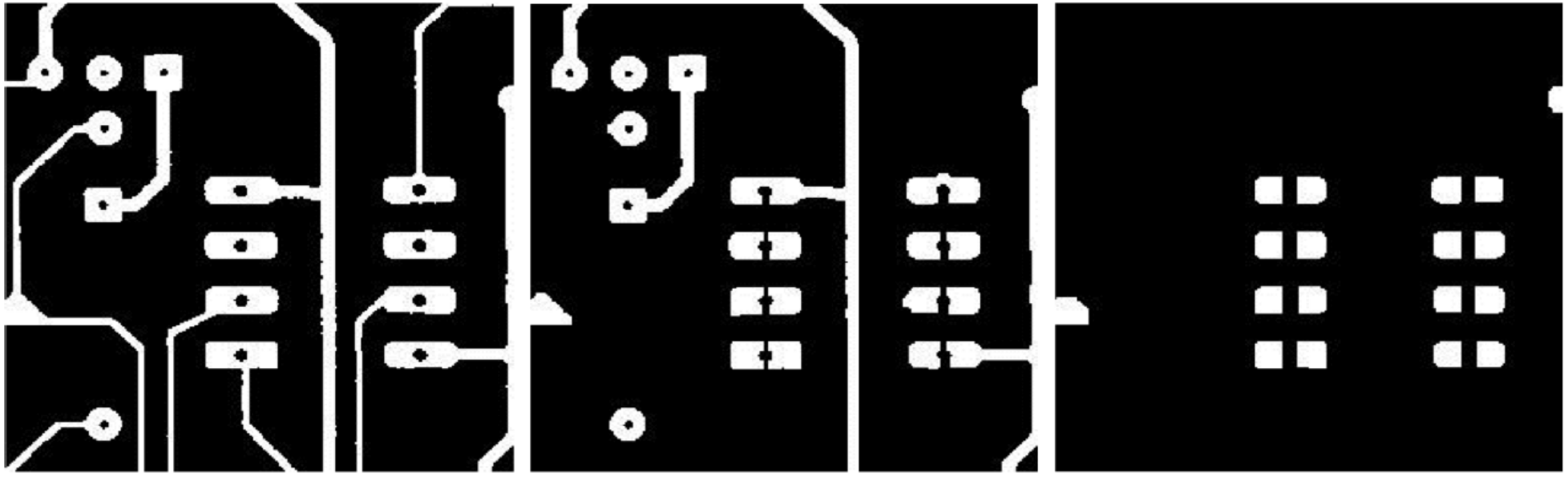
Any regions that survived the erosion are restored to their original size by the dilation:



Idempotent operation: $(f \circ s) \circ s = f \circ s$

- Once an image is opened, next openings with the same structuring element have no further effect.

Opening a binary image



Binary image

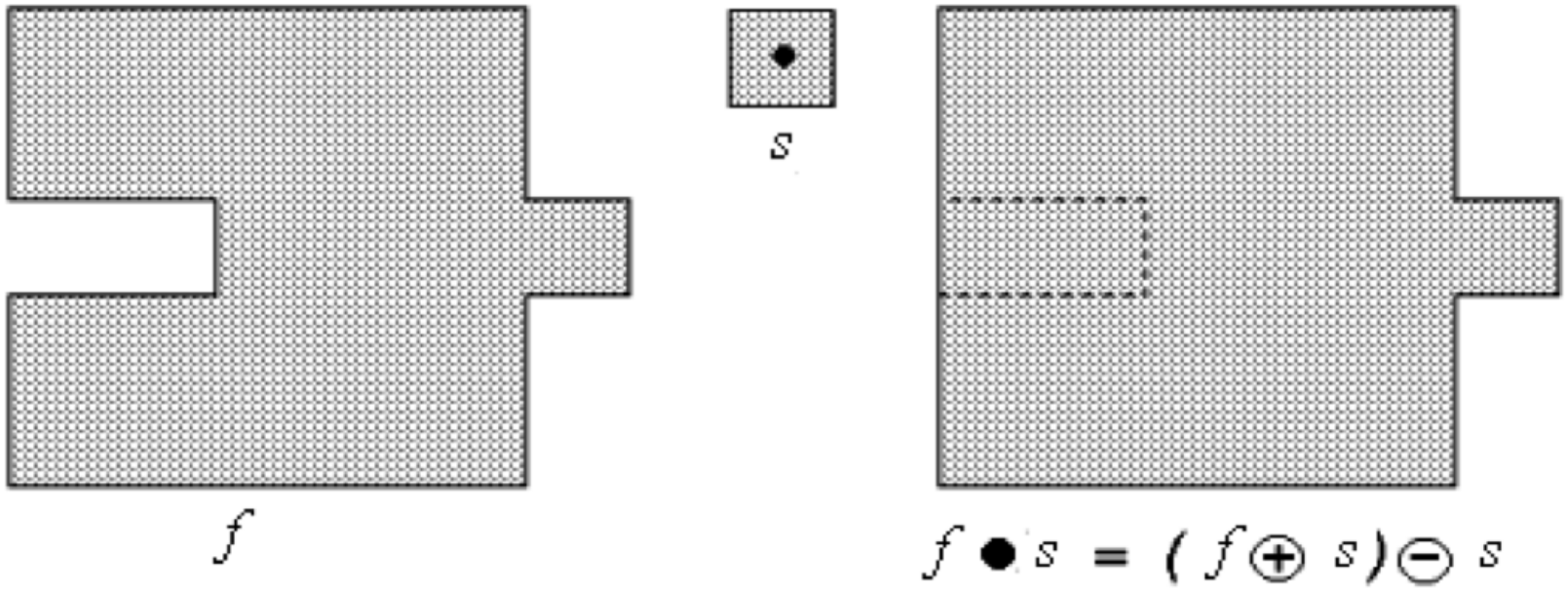
Opening with a 5×5 SE

Opening with a 9×9 SE

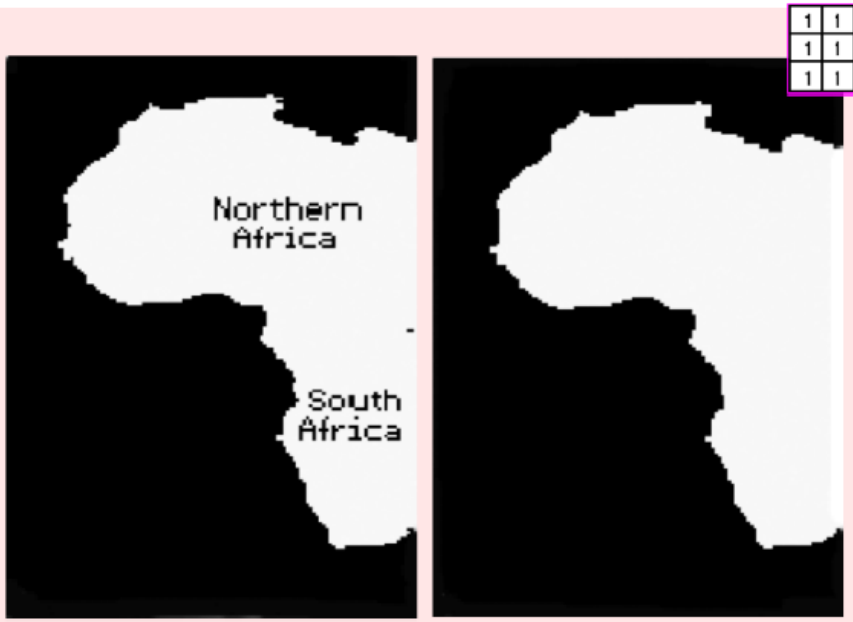
Closing

Closing $f \bullet s$ of an image f by a structuring element s is a **dilation** followed by an **erosion**:

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

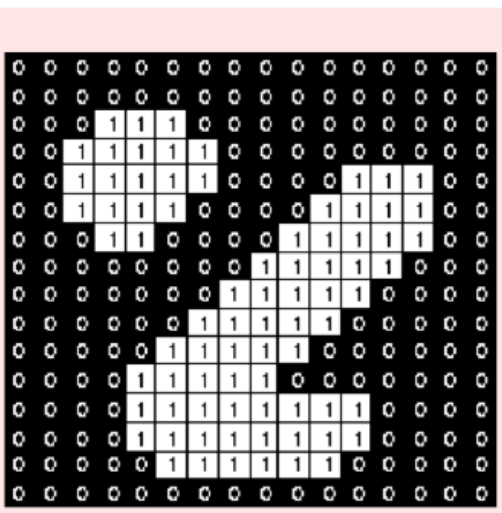


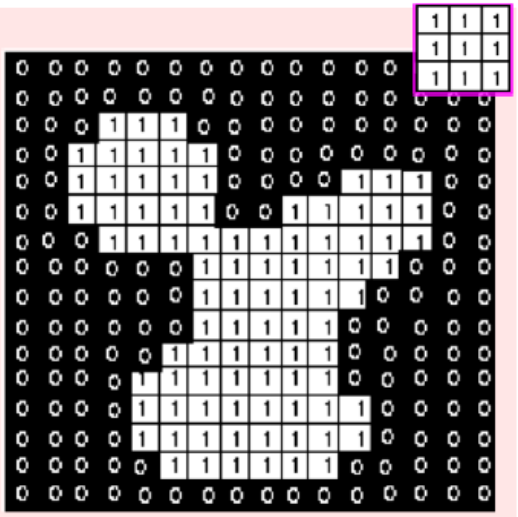
Closing examples



Binary image

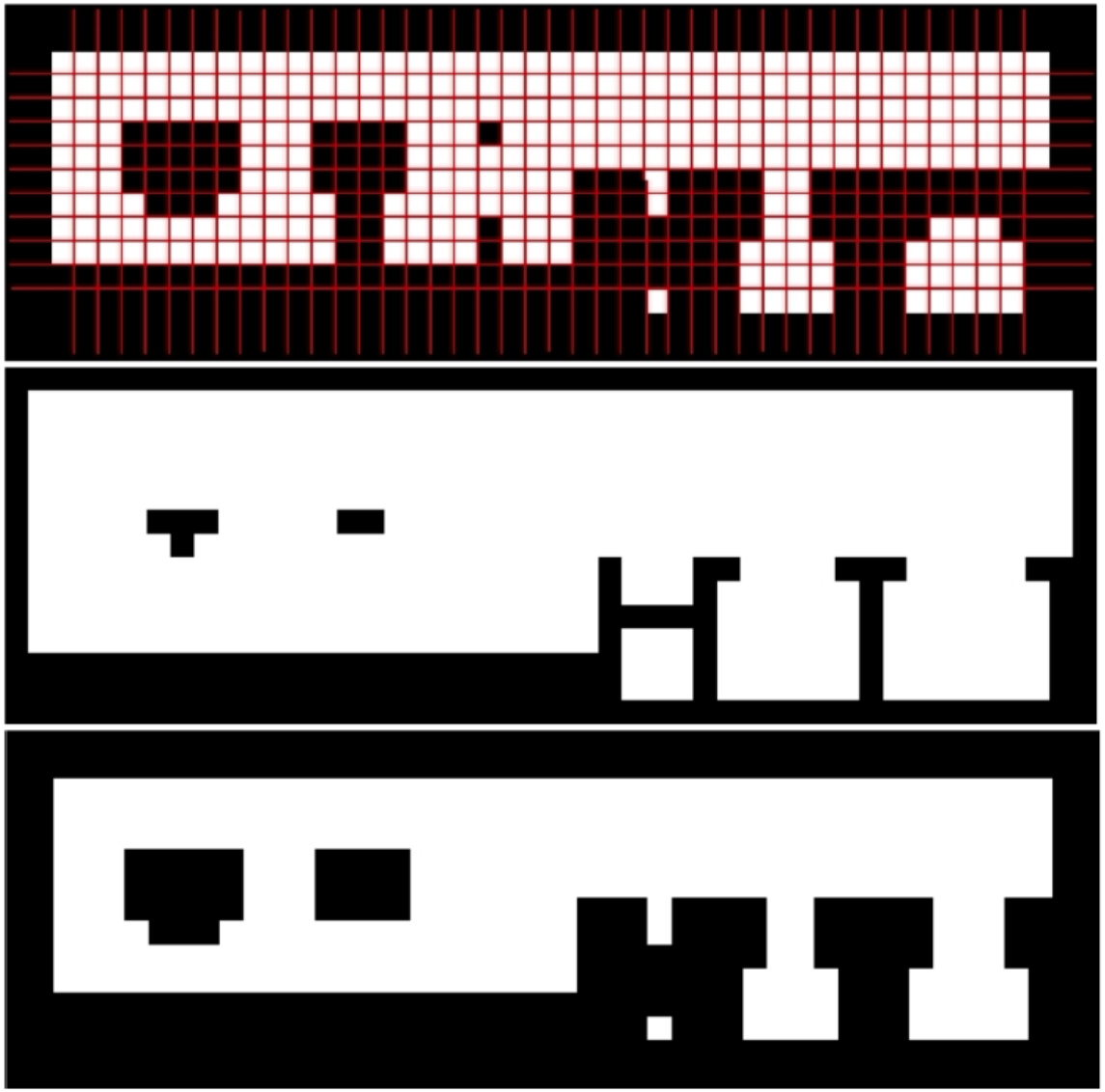
Closed image





Closing is so called because it can fill holes in the regions while keeping the initial region sizes.

Closing examples



Initial image f:
white "1"s and black "0"s

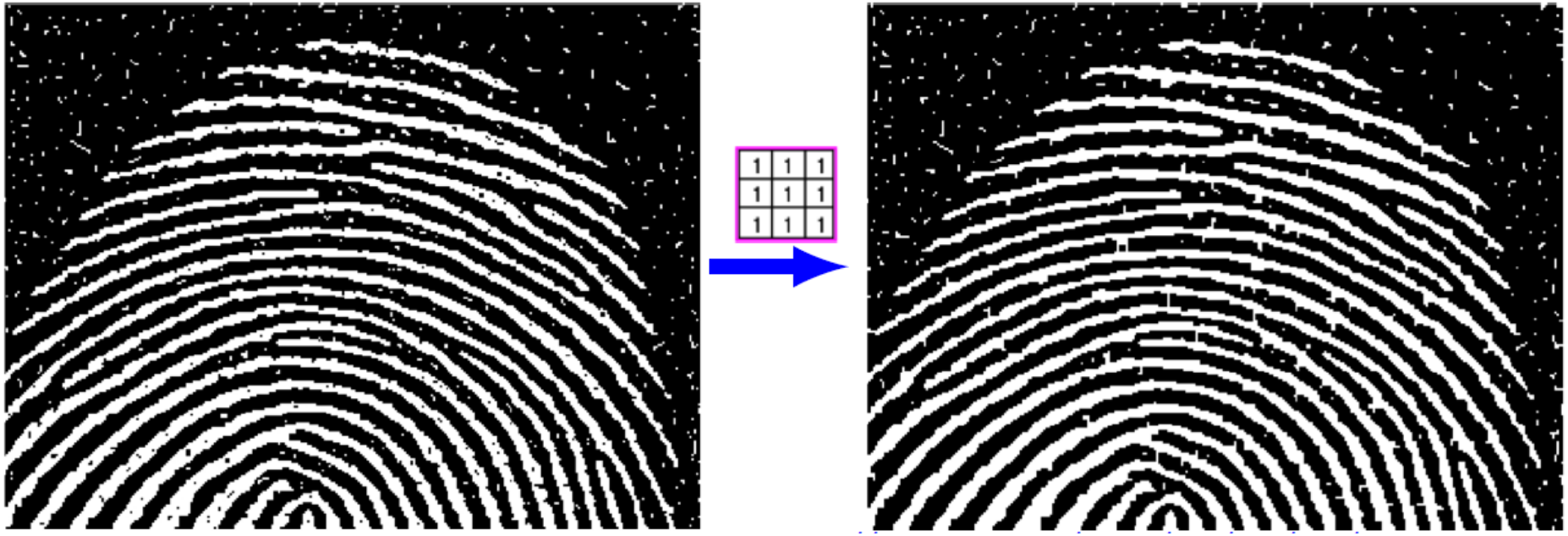
SE s 

Dilation $f \oplus s$

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

Closing a binary image

Any holes or channels smaller than the structuring element are filled:



Idempotent operation: $(f \bullet s) \bullet s = f \bullet s$

- Once an image is closed, next closings with the same structuring element have no further effect.

Closing vs. opening

Closing is the **dual operation** of opening.

- Just as opening is the dual operation of closing.

Closing of a binary image (*dual implementation*):

- Take the **complement** of that image ("1 / 0" $\Rightarrow$ "0 / 1").
- Perform **opening** with the structuring element.
- Take the **complement** of the result.

Opening of a binary image (*dual implementation*):

- Take the **complement** of that image ("1 / 0" $\Rightarrow$ "0 / 1").
- Perform **closing** with the structuring element.
- Take the **complement** of the result.

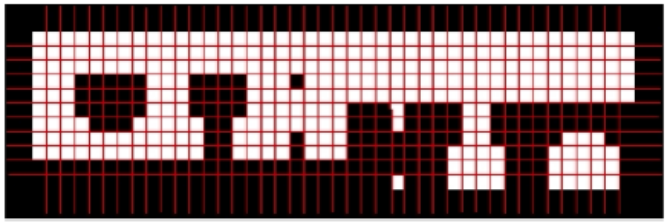
Closing vs. opening

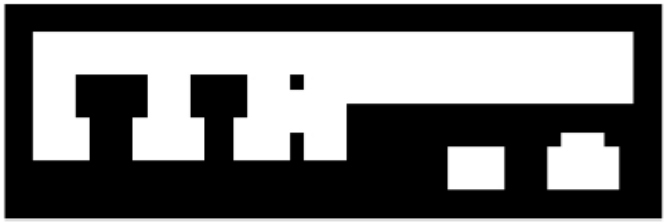
Closing with a square or disk SE:

- Fills thin connections within an object.
- Eliminates small holes and fills dents in contours.
- Fills small gaps in parts of an object.

Opening with a square or disk SE:

- Breaks thin connections within an object.
- Eliminates small islands and sharp protrusions.





Binary image: closing and opening

Initial image f

Closed, then opened image

$g = (f \bullet s) \circ s$

Binary image: opening and closing

Initial image f

Opened, then closed image

$g = (f \circ s) \bullet s$

Opening + closing vs. closing + opening