01.10.2021

Digital Image Processing (CSE/ECE 478)
Lecture-12: Morphological Operations

Ravi Kiran and Sudipta Banerjee

Image – Set of Pixels

Morphological Processing: Set of non-linear operations related to the shape or morphology of features in an image

- Basic idea:
  - Object/Region = set of pixels (or coordinates of pixels)

- 0 = background
- 1 = foreground

Object = set of pixels (or coordinates of pixels)

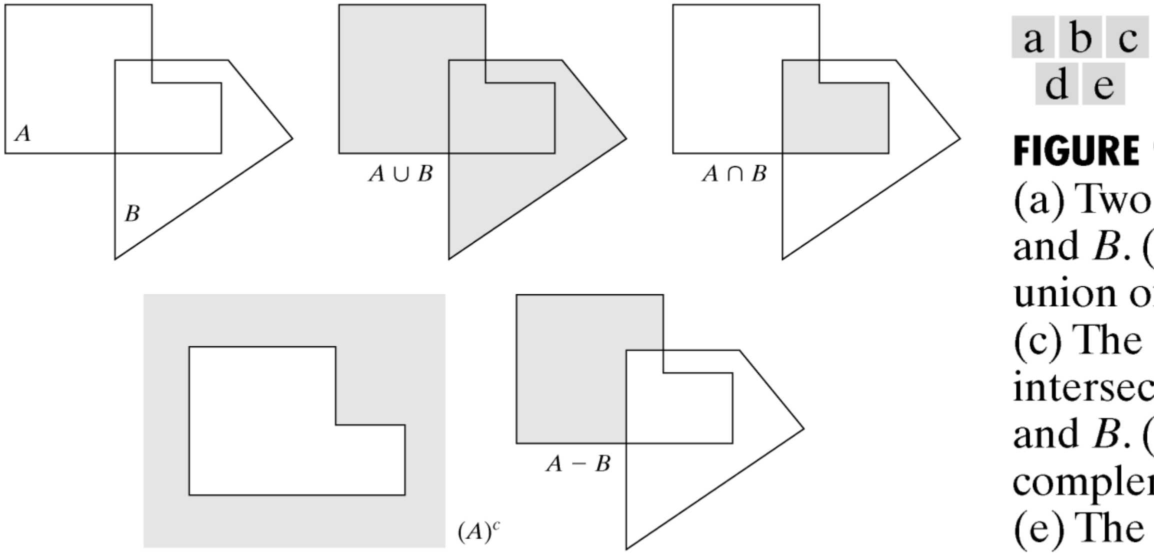


FIGURE 9.1 (a) Two sets A and B. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B.

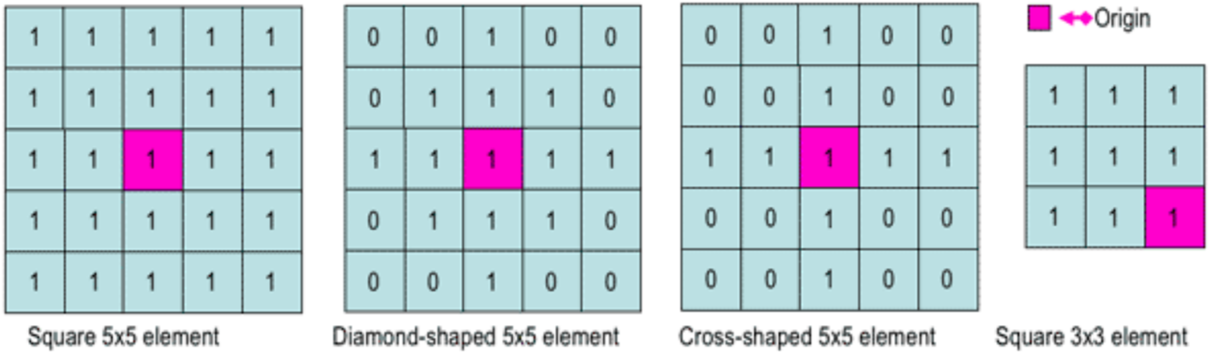
Basic operations on shapes

From: Digital Image Processing, Gonzalez,Woods And Eddins

Structuring Element

The **structuring element** is a small binary image, i.e. a small matrix of pixels, each with a value of zero or one:

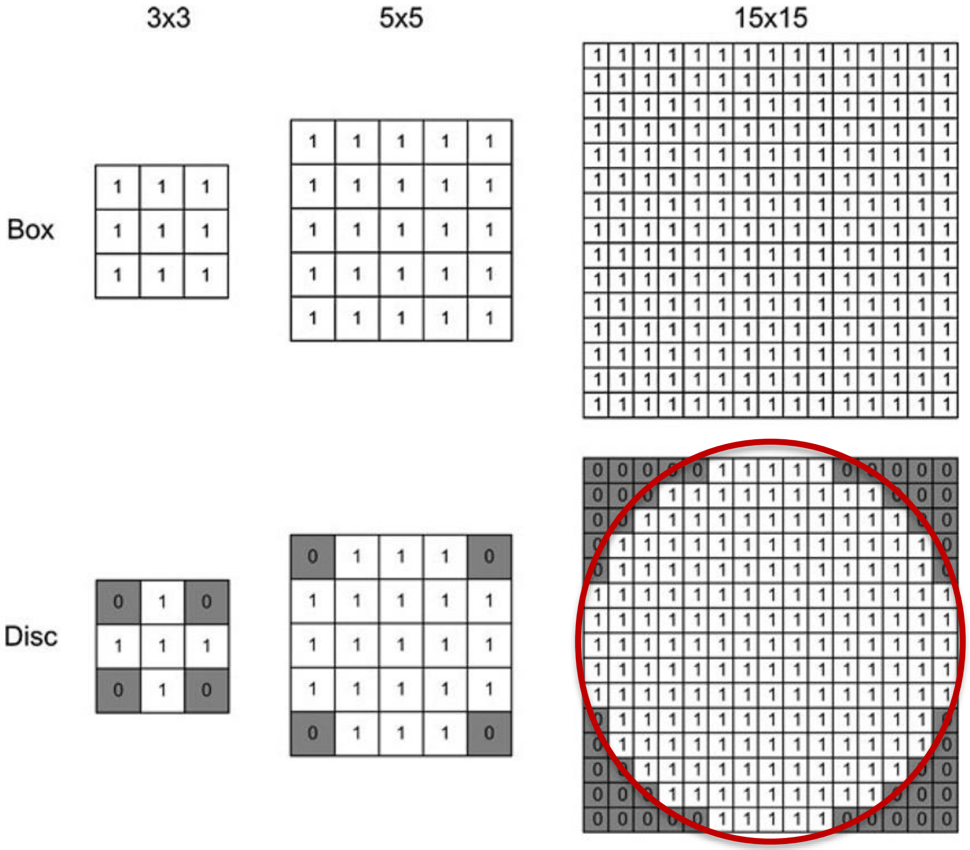
- The matrix dimensions specify the *size* of the structuring element.
- The pattern of ones and zeros specifies the *shape* of the structuring element.
- An *origin* of the structuring element is usually one of its pixels, although generally the origin can be outside the structuring element.



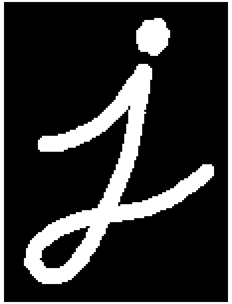
Examples of simple structuring elements.

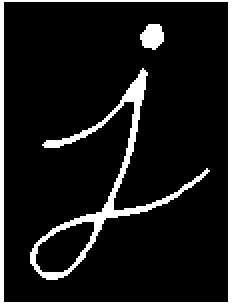
Image Courtesy: https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html/topic4.htm

Structuring Element



Erosion





Erosion : Effect

1	1	1
1	1	1
1	1	1

Set of coordinate points =

{ (-1, -1), (0, -1), (1, -1),

(-1, 0), (0, 0), (1, 0),

(-1, 1), (0, 1), (1, 1) }

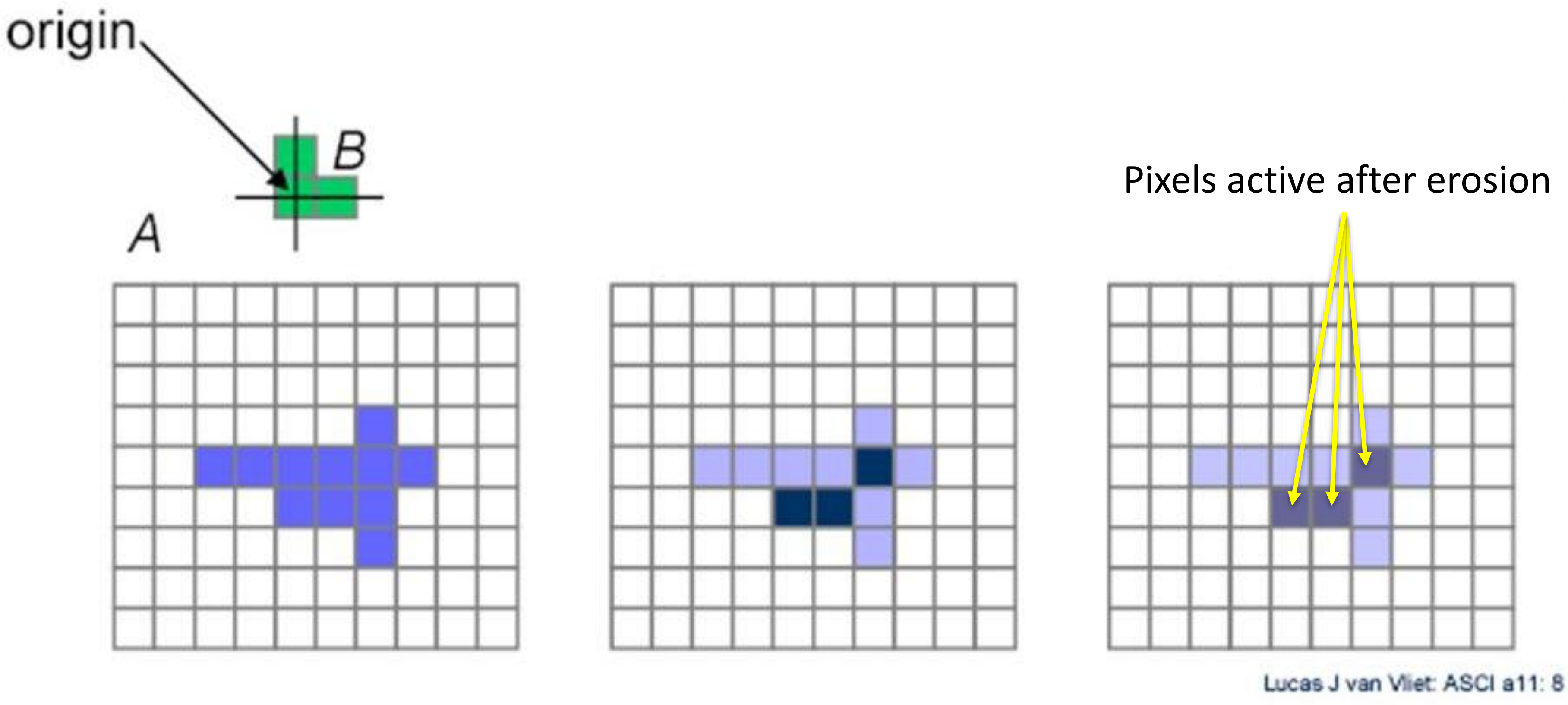
If, for a particular location of Structuring Element (SE) origin, SE lies **fully within the region**, retain the location, else set to 0

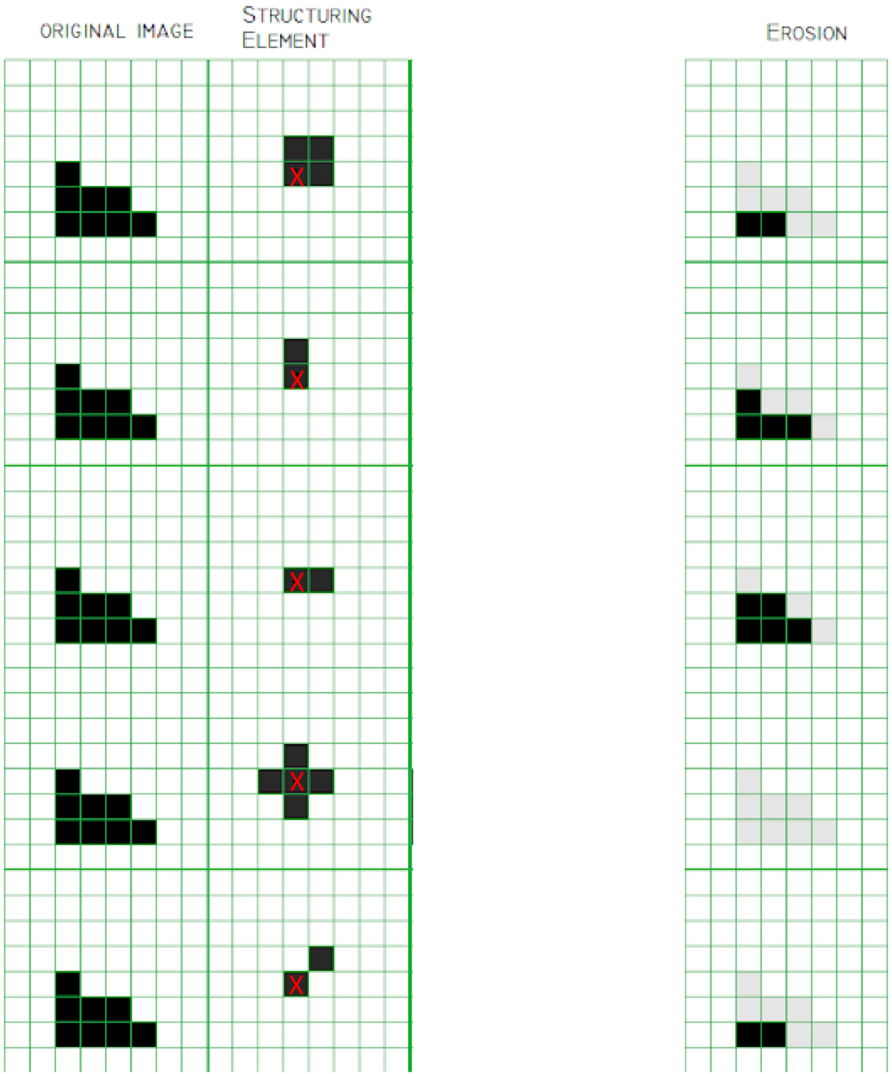
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$\longrightarrow$

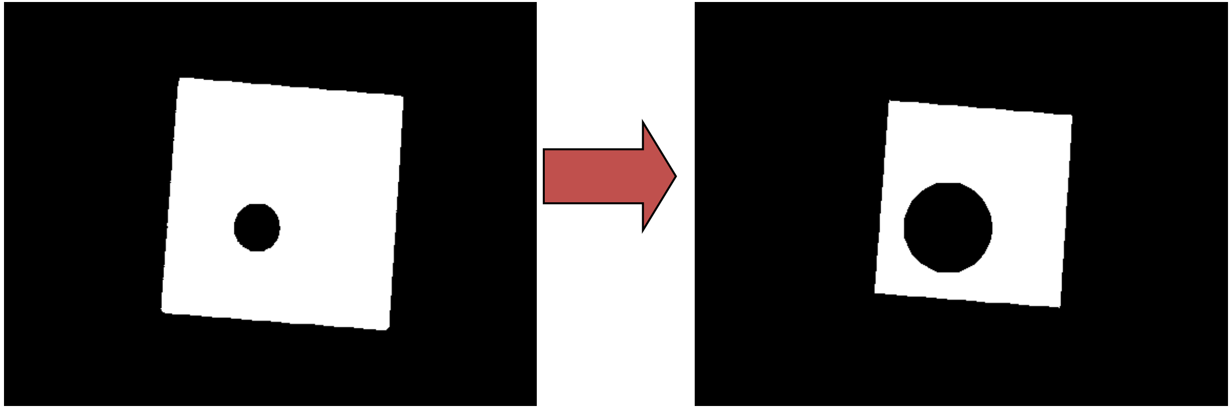
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SEs operate wrt an origin




ORIGINAL IMAGE
STRUCTURING ELEMENT
EROSION

Another example of erosion



Erosion ➔ Image gets darker

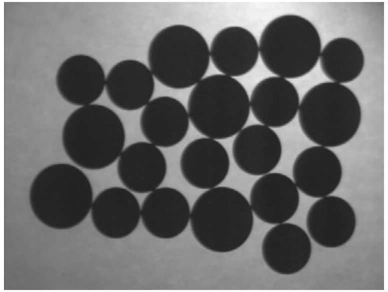
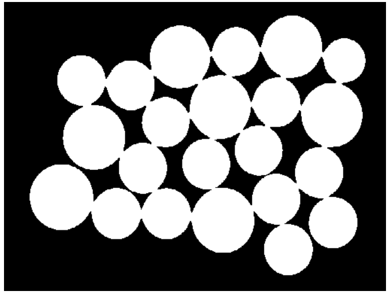
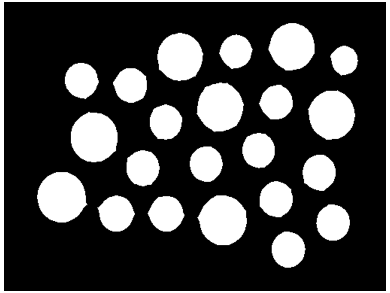
SE: disk of 11 pixels in diameter

Erosion performed 4 times

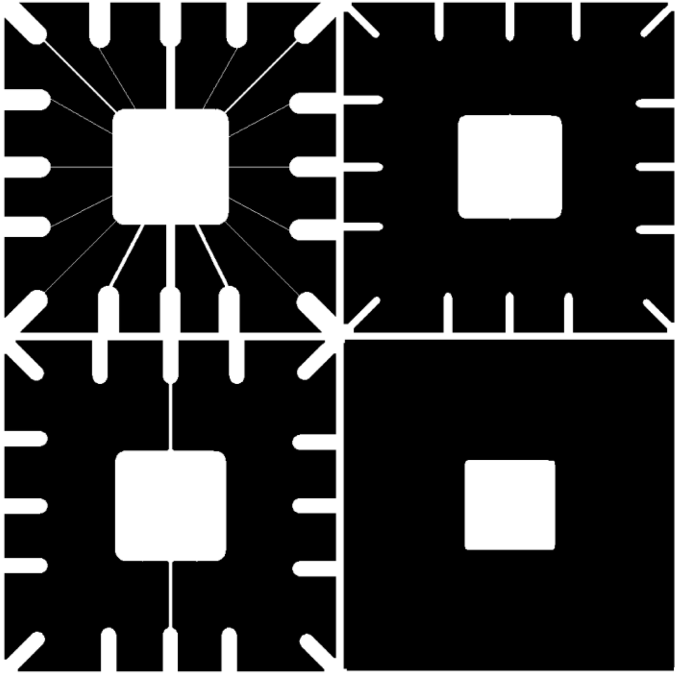
https://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm

Example: Counting coins

- Difficult because they touch each other!
- Solution: Binarization and Erosion separates them!

Erosion - example



a b
c d

FIGURE 9.8 An illustration of erosion.
(a) Original image.
(b) Erosion with a disk of radius 10.
(c) Erosion with a disk of radius 5.
(d) Erosion with a disk of radius 20.

From: Digital Image Processing, Gonzalez,Woods And Eddins

Erosion : Operation (min filter)

1	1	1
1	1	1
1	1	1

Set of coordinate points =

{ (-1, -1), (0, -1), (1, -1),

(-1, 0), (0, 0), (1, 0),

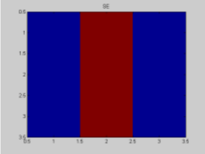
(-1, 1), (0, 1), (1, 1) }

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

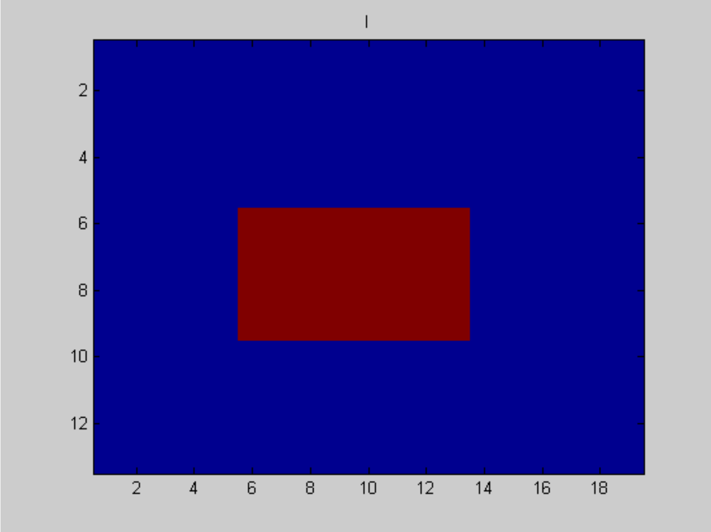
$\longrightarrow$

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

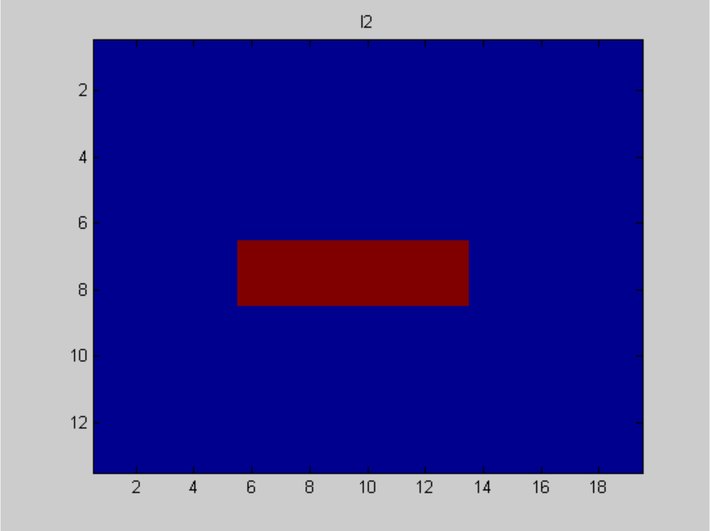
MATLAB code



SE = 3x3



I2



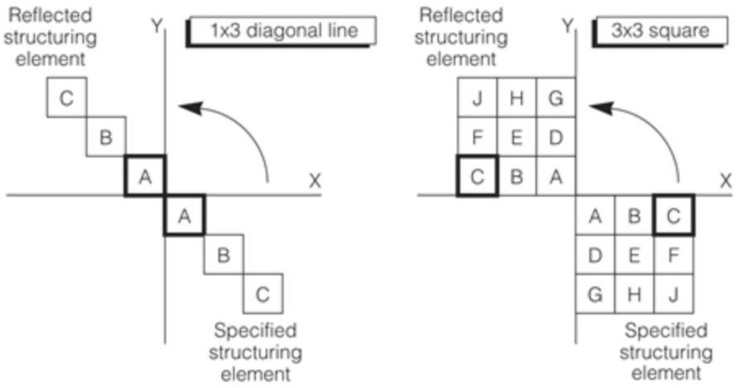
I3=imerode(I2,SE);

Erosion

- Shrinks foreground objects
- Foreground holes are enlarged
- Small (relative to structuring element size) foreground objects are removed.
- Representation: $f \ominus s$ (f: binary image, s: SE)

Dilation

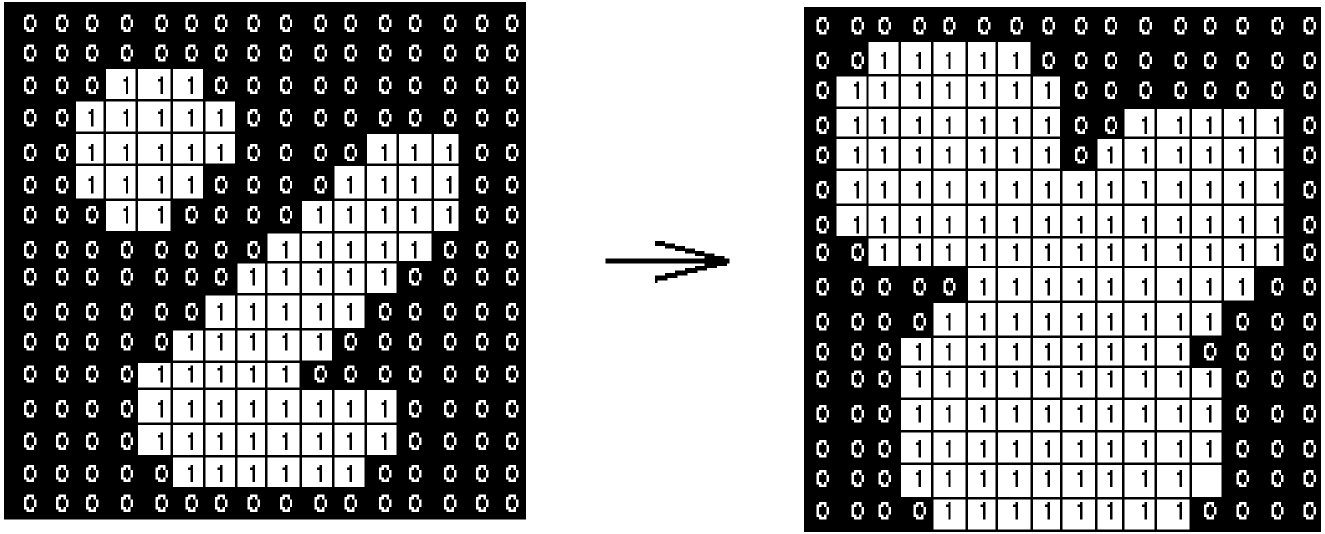
- Expands foreground objects
- Foreground holes are shrunk
- Representation: $f \oplus \hat{s}$ (f: binary image, $\hat{s}$: Reflected version of SE about its origin)



Dilation (max filter)

1	1	1
1	1 X	1
1	1	1

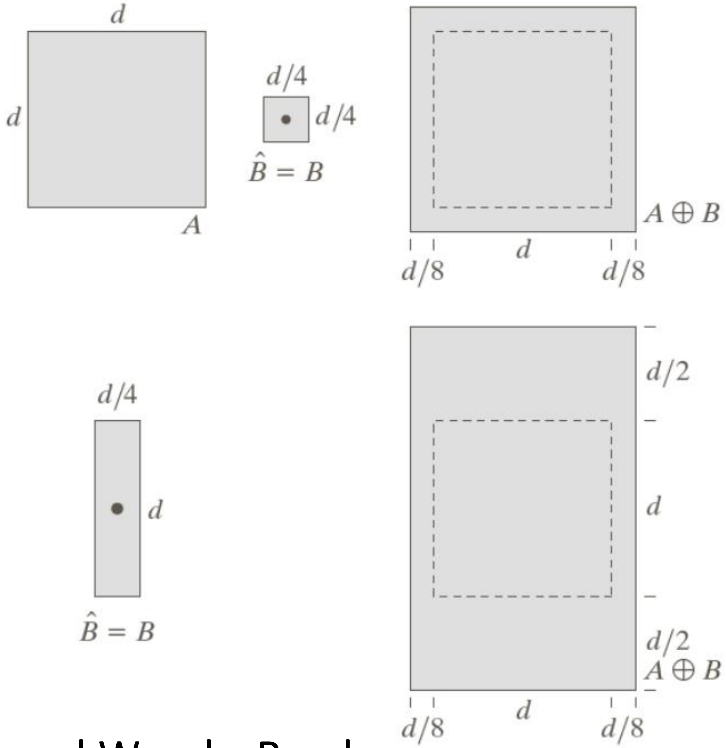
1. First reflect the SE about its origin
2. Move the SE within foreground
3. Check if the SE intersects with foreground of the binary image:
4. If the origin intersects, add the remaining background pixels as foreground



Dilation Example

Dilation by a small square structuring element extends the set A by 1/2 the width of the structuring element

Dilation by a rectangular structuring element also extends the set A by 1/2 the width of the structuring element but not uniformly in x and y

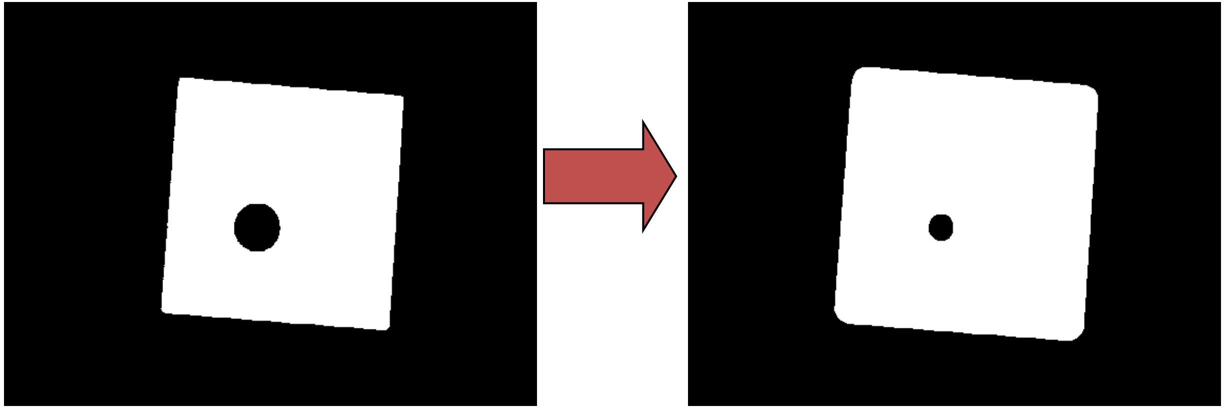


a b c
d e

FIGURE 9.6
(a) Set A.
(b) Square structuring element (the dot denotes the origin).
(c) Dilation of A by B, shown shaded.
(d) Elongated structuring element. (e) Dilation of A using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference

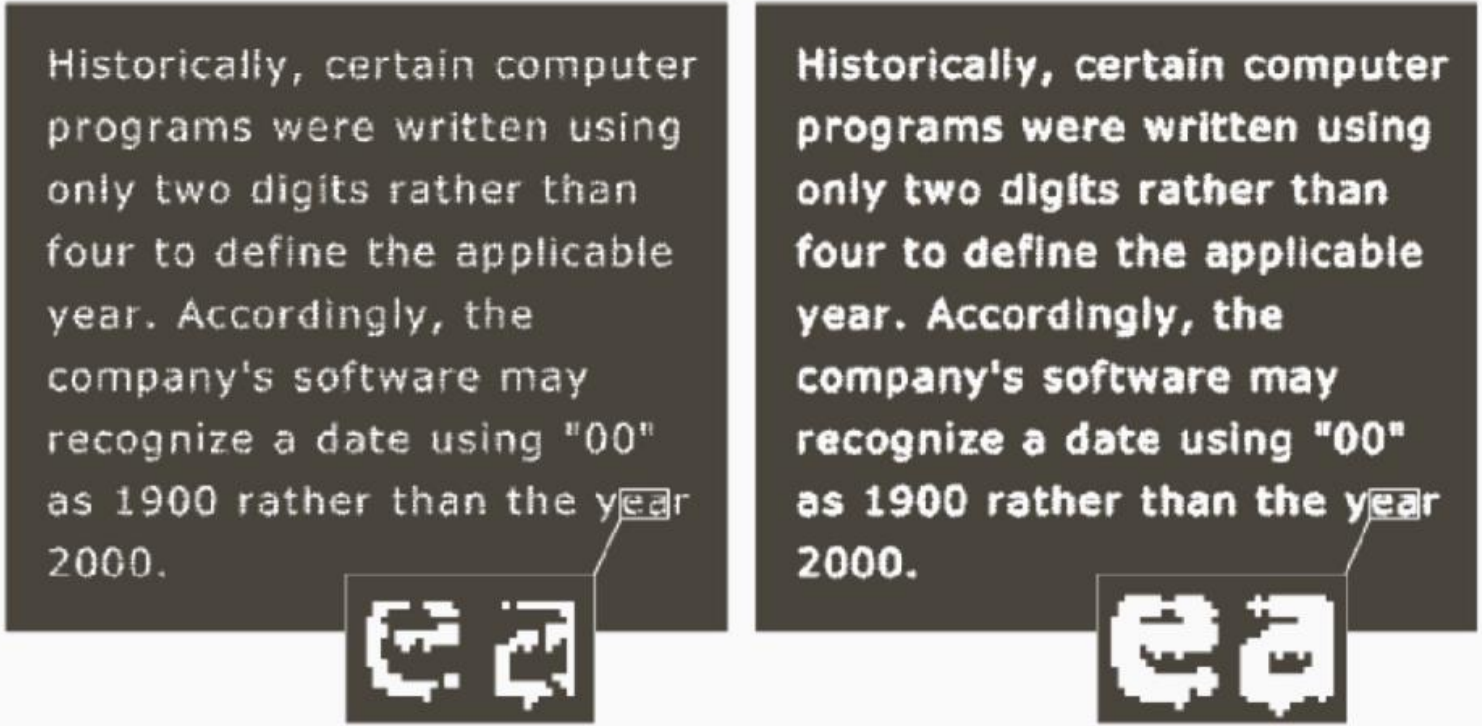
Images taken from Gonzalez and Woods. Read:
http://engr.case.edu/merat_francis/eecs490f07/lectures/lecture17.pdf

Dilation Example



- Image gets lighter, more uniform intensity
- NOTE-1: SE = disk
- NOTE-2: Multiple iterations of dilation (2 passes)

https://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm

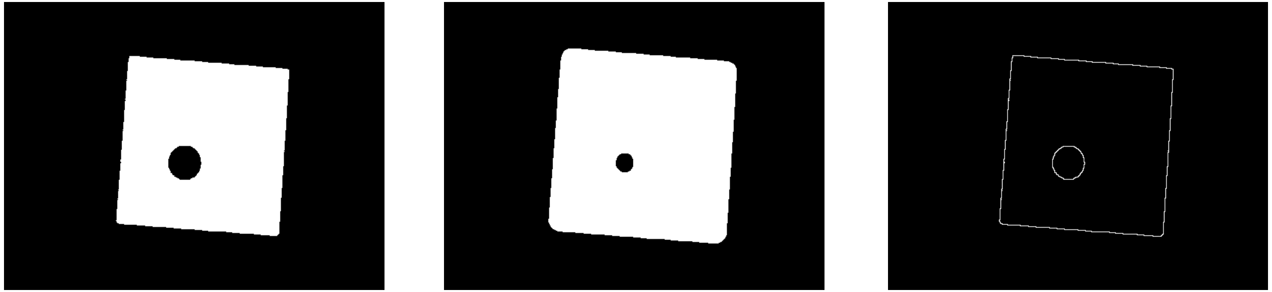


0	1	0
1	1	1
0	1	0

Figure 1: Image before (left) and after (right) dilation with the structuring element shown at the bottom

Boundary Detection

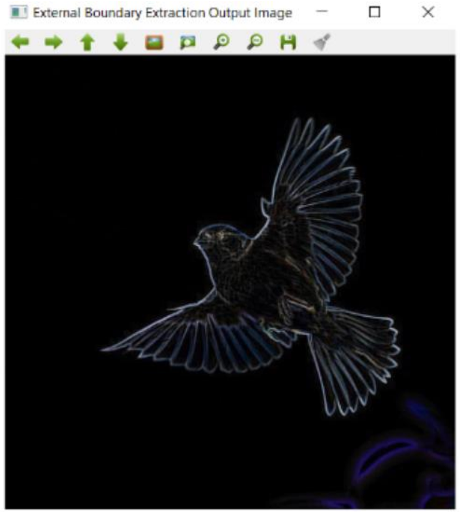
1. Dilate input image
2. Subtract input image from dilated image
3. Boundaries remain!



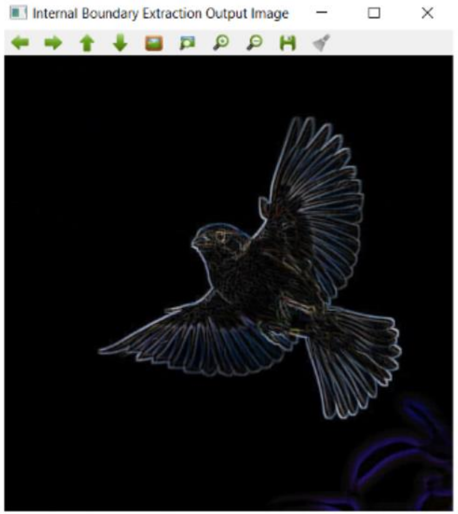
Can use erosion also ..

Fig 3: (a) Original Image (linkon.tif) (B) After erosion operation (C) Boundary Extraction with the help of Erosion.

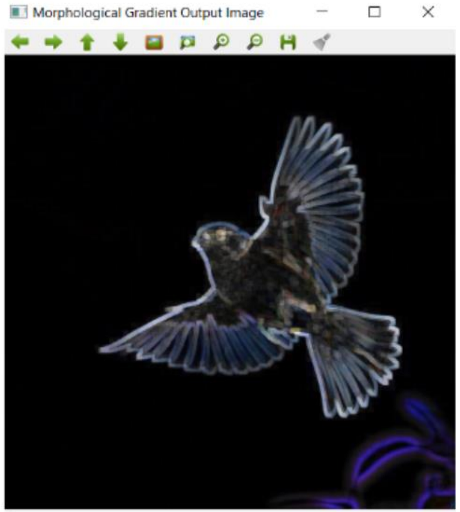
Boundary extraction



$(A \oplus B)-A$



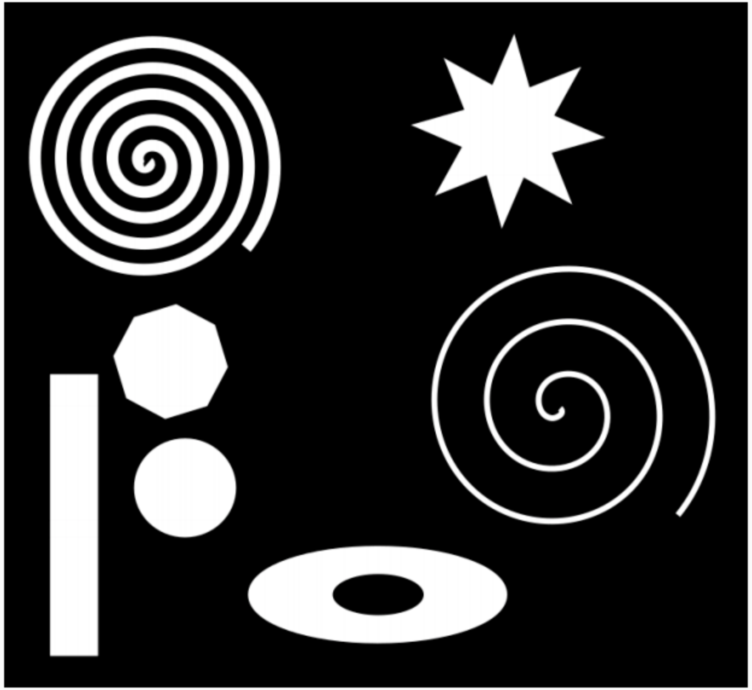
$A- (A \ominus B)$



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

A: Original image; B: SE square of size 7x7

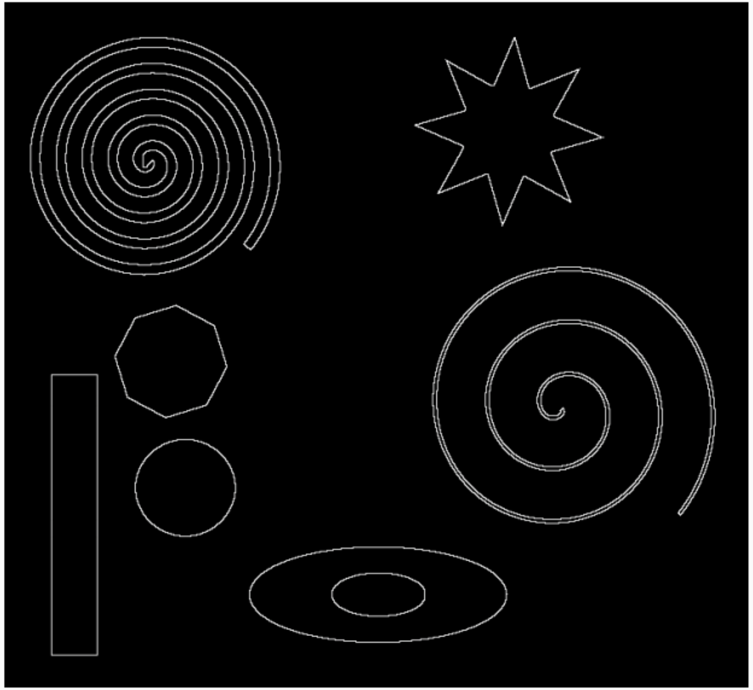
https://towardsdatascience.com/image-processing-part-3-dbf103622909



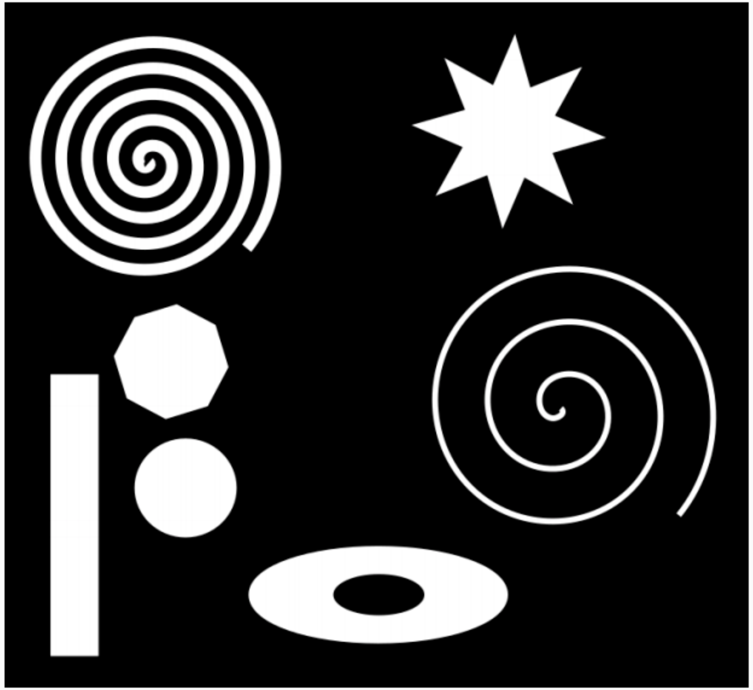
(a) f

1	1	1
1	1	1
1	1	1

(b) s



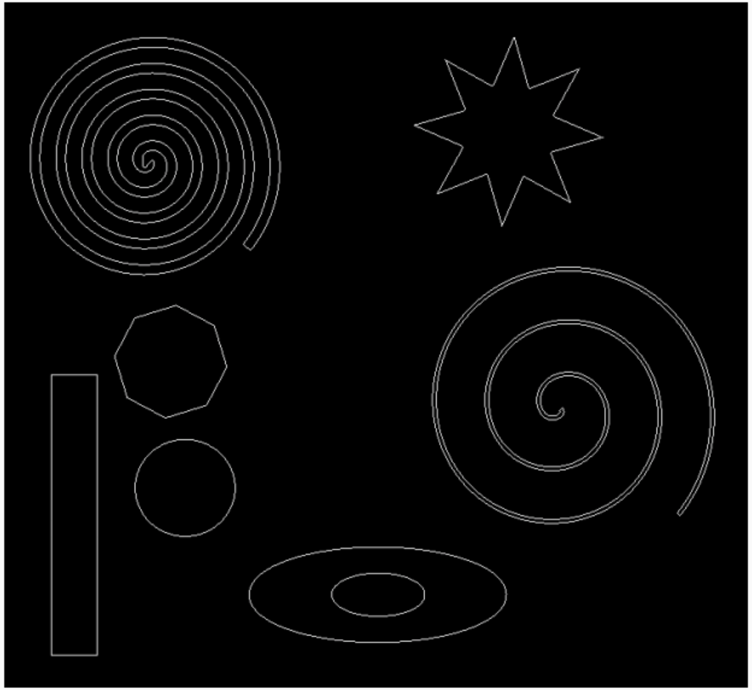
(c) $f - (f \ominus s)$



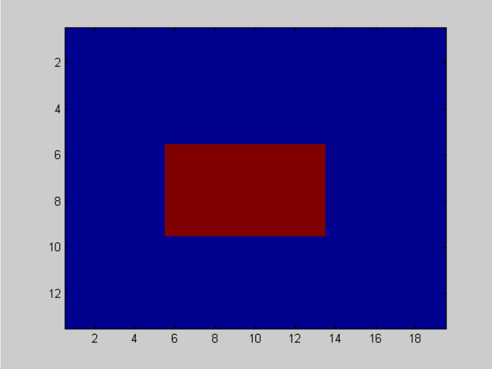
(a) f

0	1	0
1	1	1
0	1	0

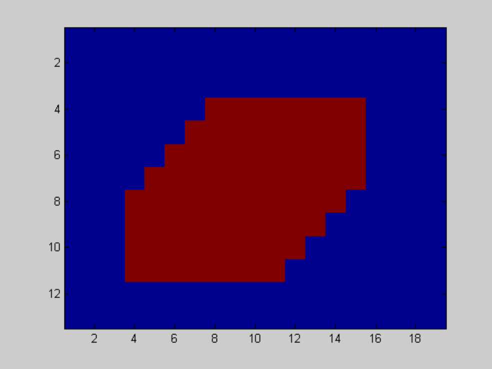
(b) s



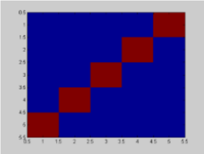
(c) $f - (f \ominus s)$



I



I2



SE

```
>> I(6:9,6:13)=1;
>> figure, imagesc(I)
>> I2=imdilate(I,SE);
>> figure, imagesc(I2)
```

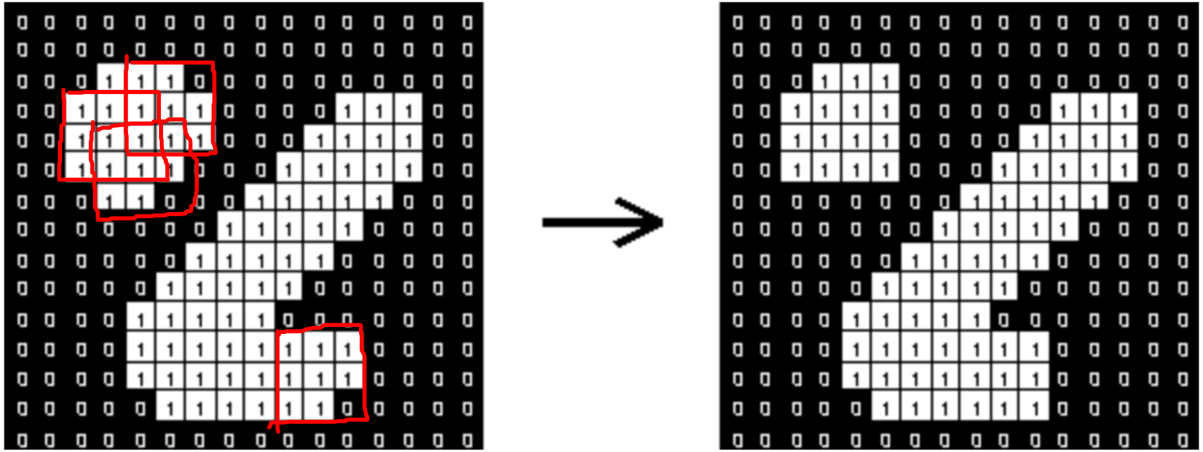
Opening and Closing

- Important operations
- Derived from the fundamental operations
  - Dilation
  - Erosion

Opening (Erosion then Dilation)

- Take the structuring element (SE) and slide it around *inside* each foreground region.
  - All foreground pixels which can be covered by the SE with the SE being entirely within the foreground region will be preserved.
  - All foreground pixels which can *not* be reached by the structuring element without lapping over the edge of the foreground object will be eroded away!

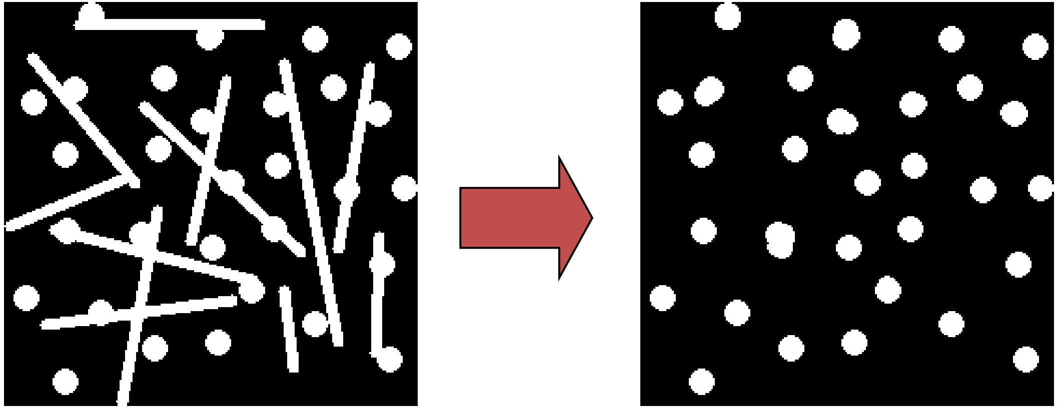
SE: 3x3 square



https://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm

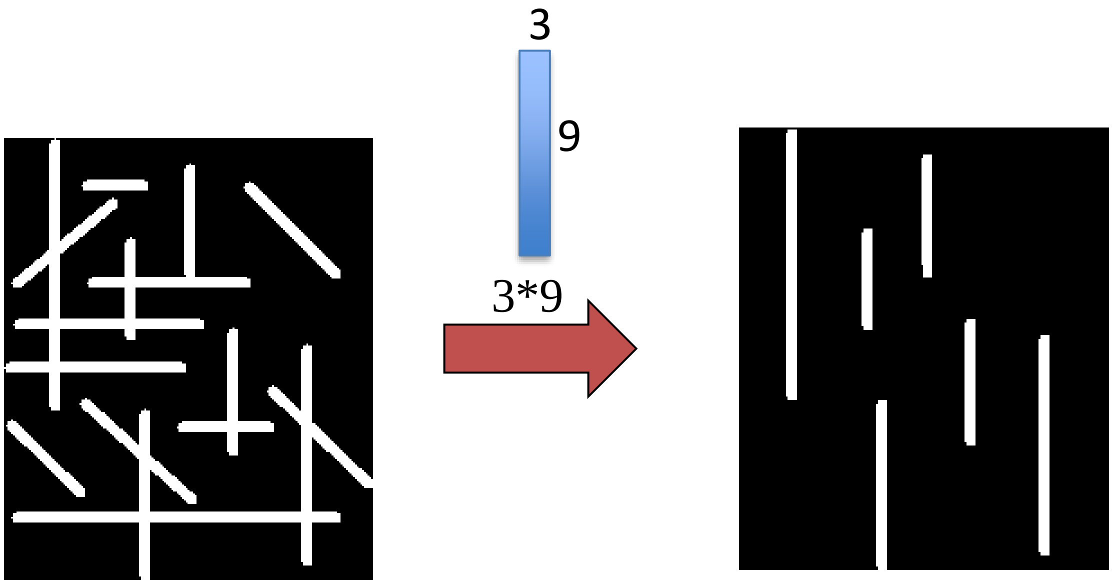
Opening: Example

- Opening with a 11 pixel diameter disc



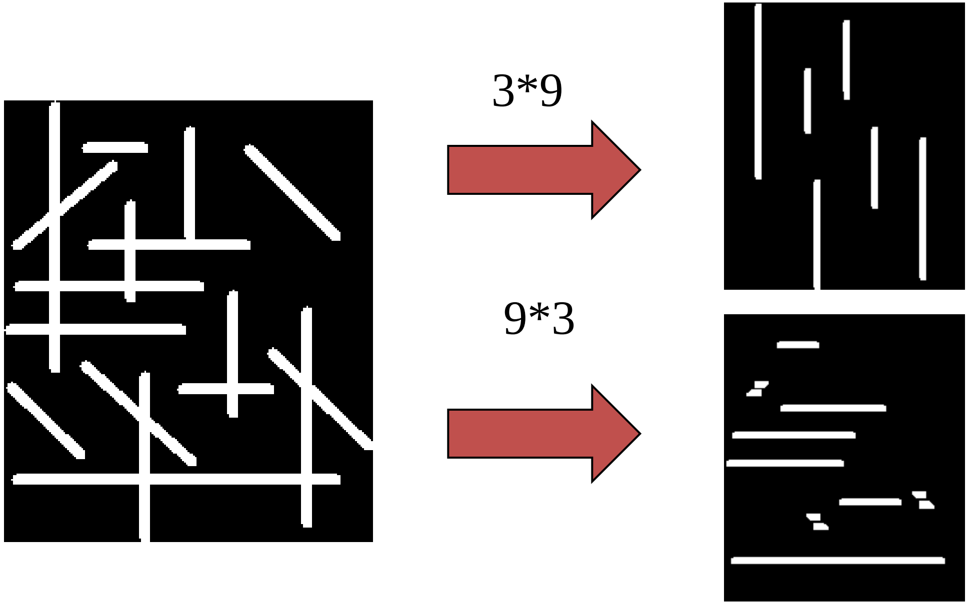
Opening: Another Example

- 3x9 Structuring Element



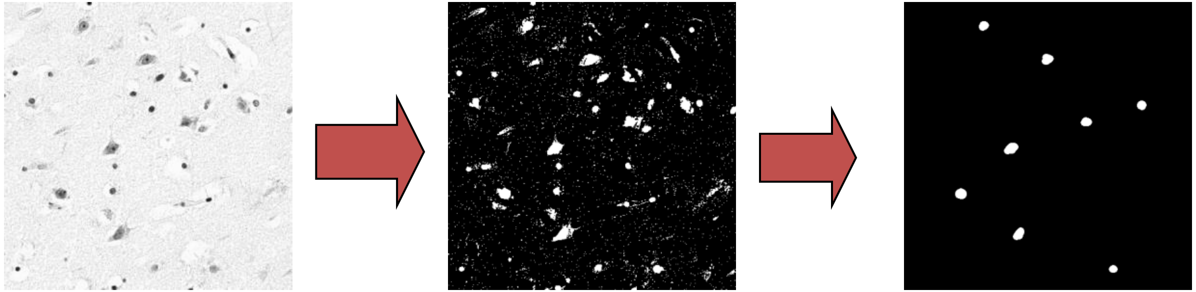
Opening: Another Example

- 3x9 and 9x3 Structuring Element



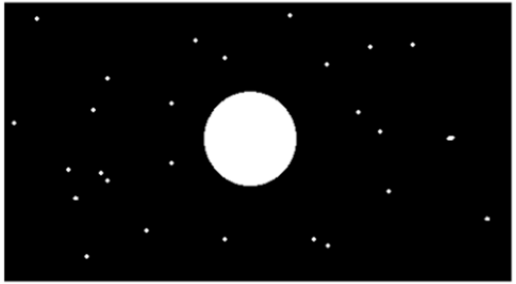
Use Opening for Separating Blobs

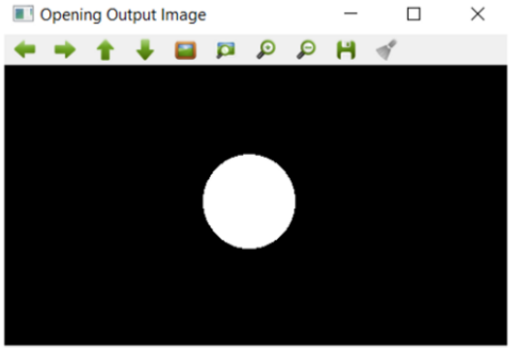
- Use large structuring element that fits into the big blobs
- Structuring Element: 11 pixel disc



Opening

- Opening removes small objects from the foreground (removes morphological noise)
- Erosion followed by Dilation
  - Using the *same structuring element for both operations.*
- Opening is **idempotent:** Repeated application has no further effects!
- Representation: $f \circ s = (f \ominus s) \oplus s$



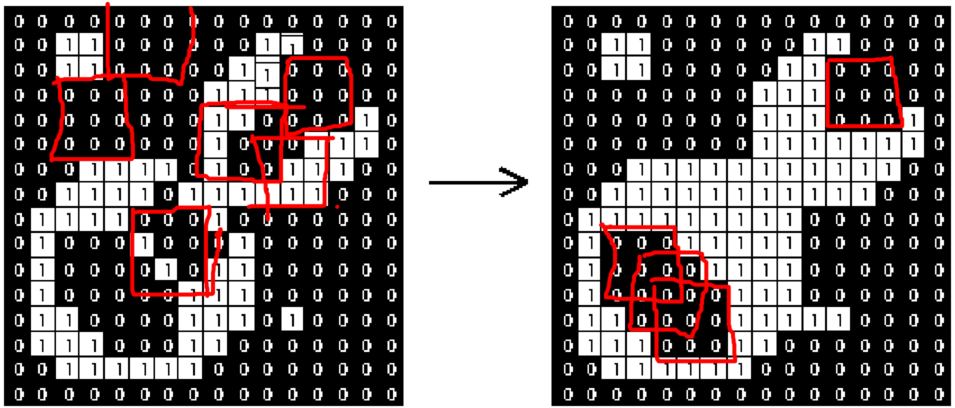


https://towardsdatascience.com/image-processing-part-3-dbf103622909

Closing (Dilation then Erosion)

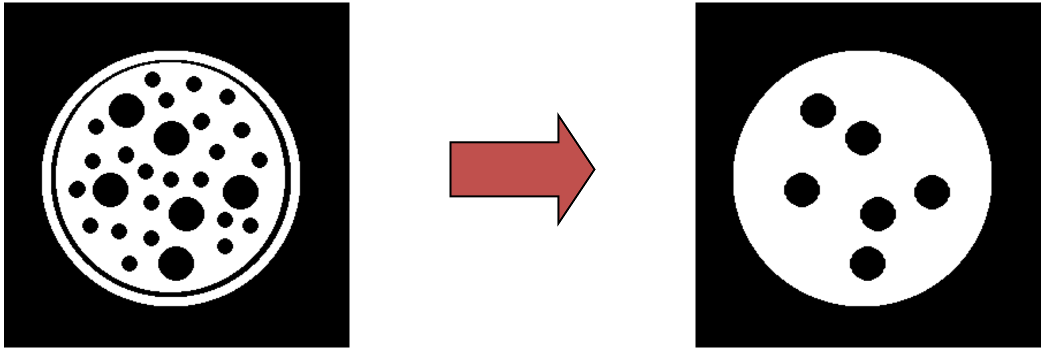
- Take the structuring element (SE) and <u>slide it around *outside* each foreground region</u>.
  - For any background pixel, if the SE *can touch* it without any part of the SE being inside the foreground region, the pixel stays as background
  - Any background pixel that *cannot* be touched by SE without coming inside the foreground region is changed to become a foreground pixel

SE: 3x3 square



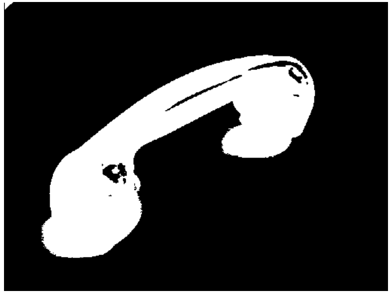
Closing: Example

- Closing operation with a 22 pixel disc
- Closes small holes in the foreground while keeping initial region sizes
- Closing is idempotent
- Representation: $f \cdot s = (f \oplus s) \ominus s$

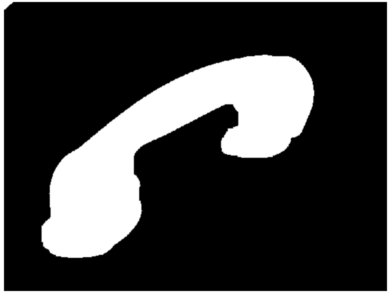


Closing Example 1

1. Threshold
2. Closing with disc of size 20



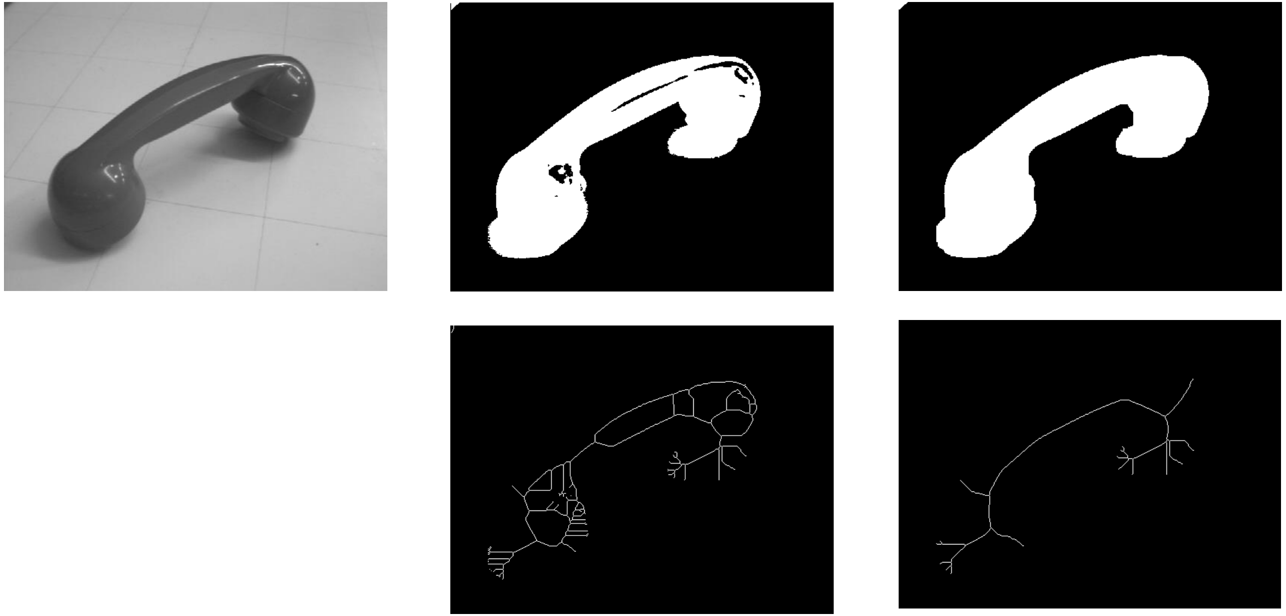
Thresholded



closed

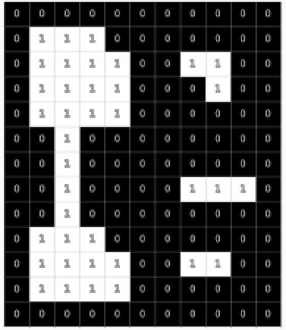
Application of Closing

- Good for further processing: E.g. Skeleton operation looks better for closed image!

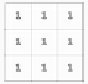


Opening & Closing

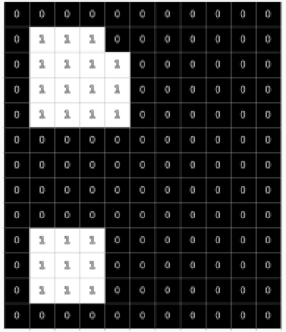
- Opening is the *dual* of closing. Duality in terms of complementation and reflection
- Representation: $(f \cdot s)^c = f^c \circ \hat{s}$; $(f \circ s)^c = f^c \cdot \hat{s}$
  - $\hat{s}$ reflected version of SE (for symmetric SE with origin at center $\hat{s} = s$)



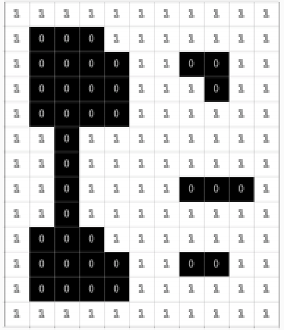
(a) f



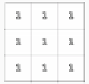
(b) s



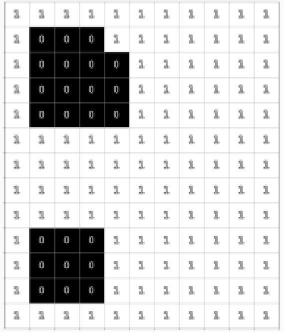
(c) $f \circ s$



(a) f^c

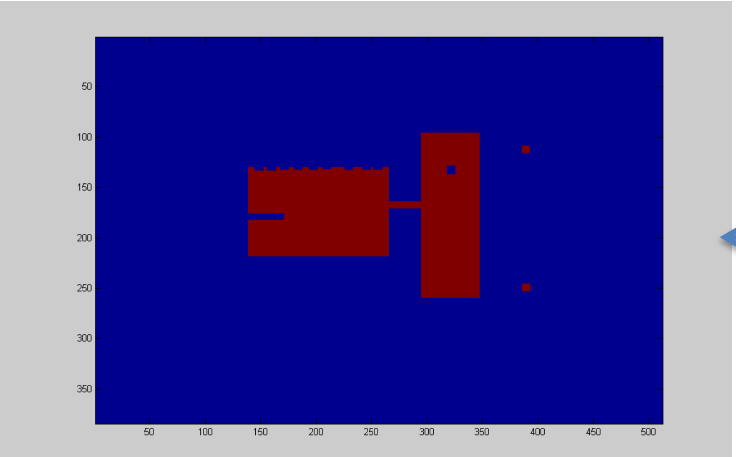


(b) s

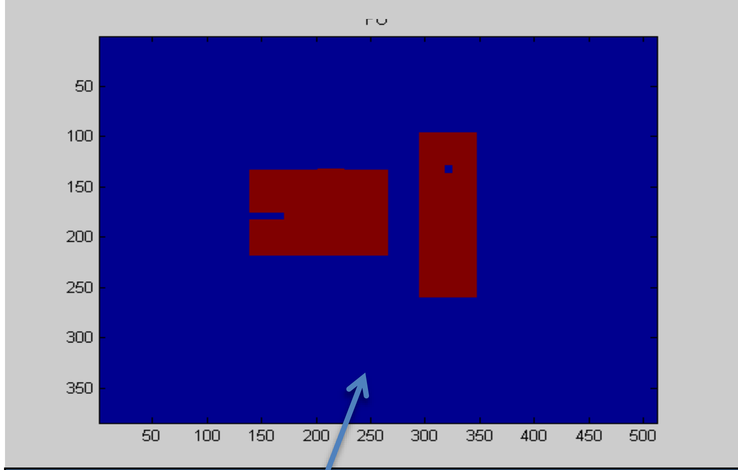


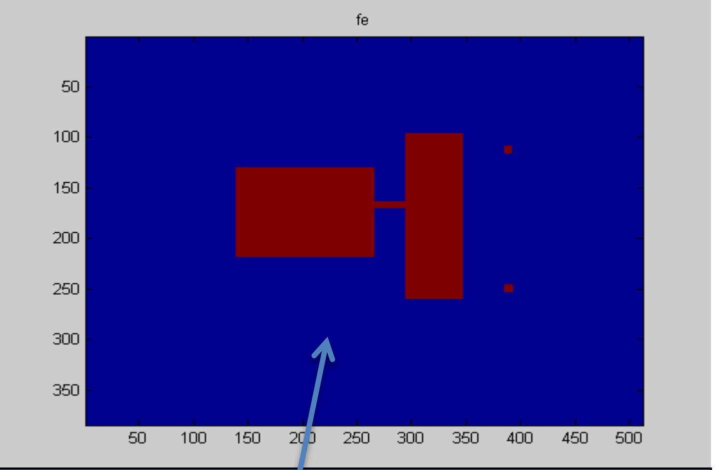
(c) $f^c \bullet s$

Here, $\hat{s} = s$



ORIGINAL





OPENING

CLOSING

J=imopen(I,SE)

J=imclose(I,SE)

Fingerprint problem

FIGURE 9.11 (a) Noisy fingerprint image. (b) Opening of image. (c) Opening followed by closing. (Original image courtesy of the National Institute of Standards and Technology.)

From: Digital Image Processing, Gonzalez,Woods And Eddins

MAGNITUDE RELATIONS

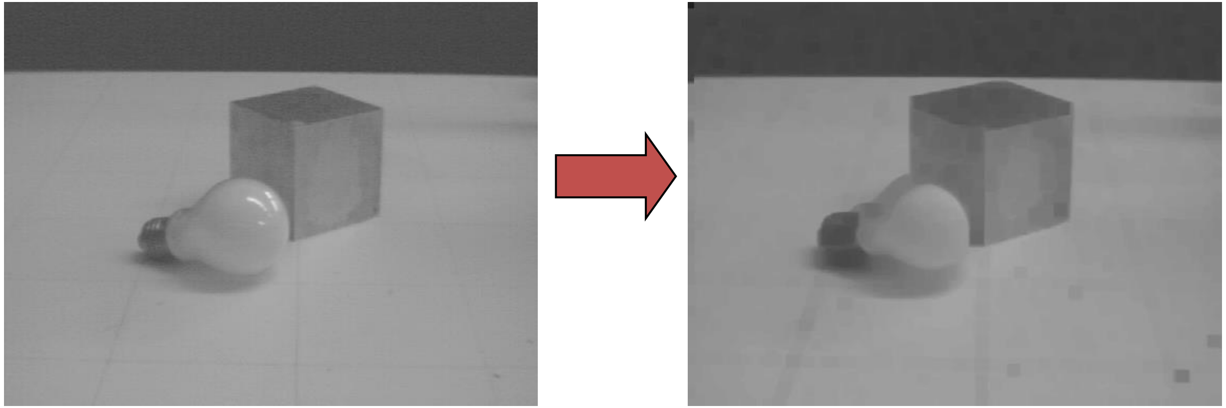
- Dilation and closing are *extending operations*, meaning that foreground pixels are added to the image.
- Erosion and opening are *narrowing operations*, meaning that foreground pixels are removed.
- For a binary image f and a binary structuring element s, we have that

$$(f \ominus s)(x) \leq (f \circ s)(x) \leq f(x) \leq (f \bullet s)(x) \leq (f \oplus s)(x)$$

- On a similar note, if $F(g)$ is the set of foreground pixels in g,

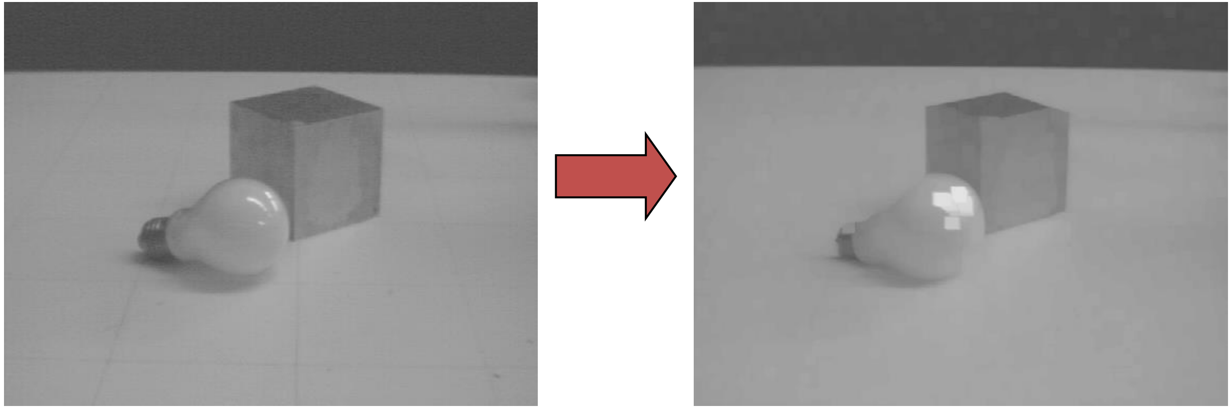
$$F(f \ominus s) \subseteq F(f \circ s) \subseteq F(f) \subseteq F(f \bullet s) \subseteq F(f \oplus s)$$

Erosion on Gray Value Images



- min filter
- Images get darker!

Dilation on Gray Value Images



- max filter
- More uniform intensity

References

- G&W, 3rd Ed., 9.1-9.3, 9.6
- https://in.mathworks.com/help/images/morphological-dilation-and-erosion.html
- https://scikit-image.org/docs/dev/auto_examples/applications/plot_morphology.html#sphx-glr-auto-examples-applications-plot-morphology-py