Introduction to Medical Image Analysis

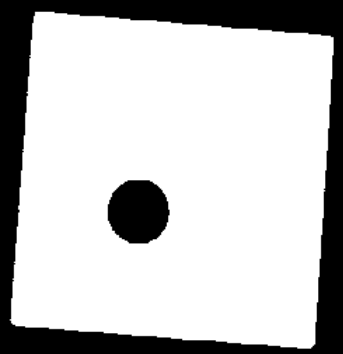
Rasmus R. Paulsen
DTU Compute

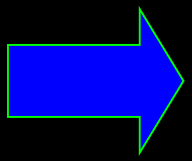
rapa@dtu.dk

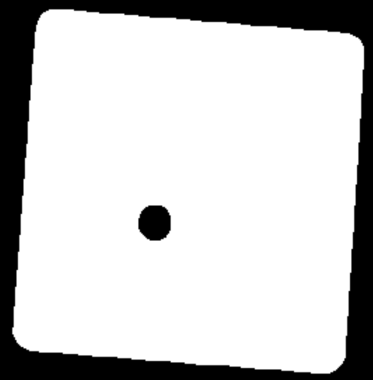
http://www.compute.dtu.dk/courses/02512

Plenty of slides adapted from Thomas Moeslunds lectures

Lecture 5 – Morphology







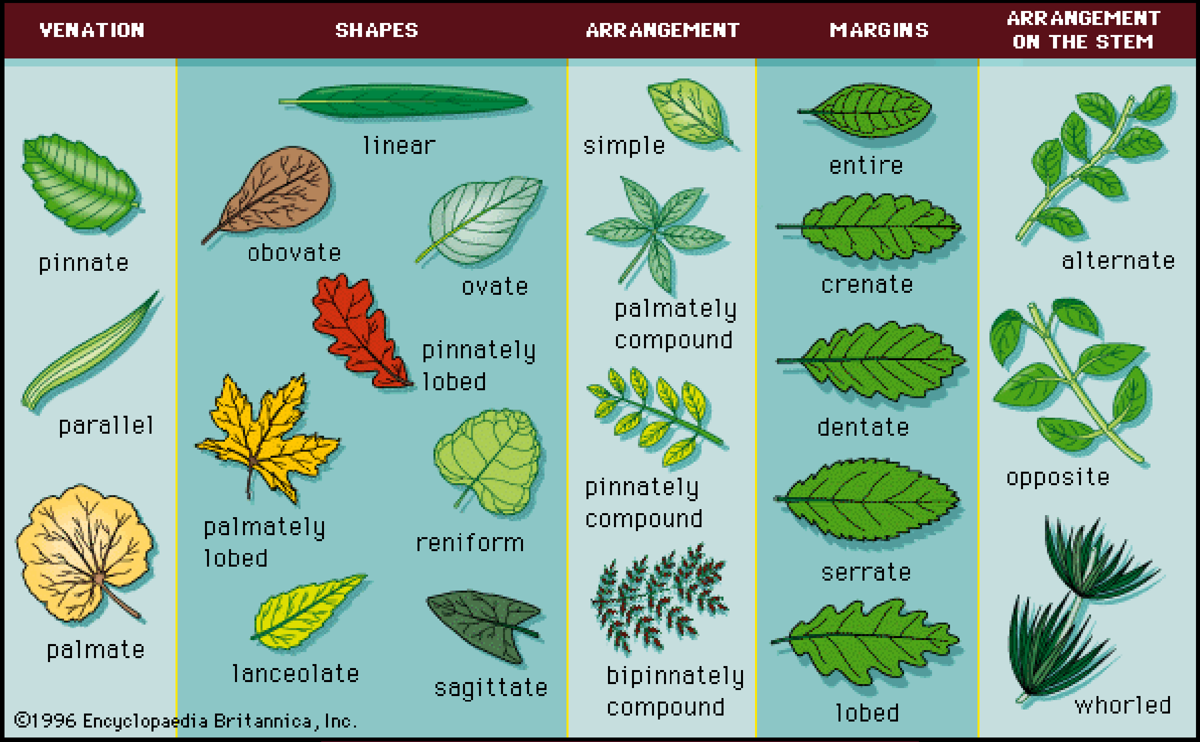
0	0	1	1	1	0	0
0	1	1	1	1	1	0
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
0	1	1	1	1	1	0
0	0	1	1	1	0	0

What can you do after today?

- Describe the similarity between filtering and morphology
- Describe a structuring element
- Compute the dilation of a binary image
- Compute the erosion of a binary image
- Compute the opening of a binary image
- Compute the closing of a binary image
- Apply compound morphological operations to binary images
- Describe typical examples where morphology is suitable
- Remove unwanted elements from binary images using morphology
- Choose appropriate structuring elements and morphological operations based on image content

Morphology

- The science of *form, shape* and *structure*
- In biology: The form and structure of animals and plants



Common leaf morphologies

Mathematical morphology

- Developed in 1964
- Theoretical work done in Paris
- Used for classification of minerals in cut stone
- Initially used for binary images

Theorem 4.10

$$\begin{cases} \psi_m = \widetilde{\varphi}\,\widetilde{\gamma} = \widetilde{\gamma}\,\widetilde{\varphi}\,\widetilde{\gamma} = \psi\widetilde{\gamma} & , \\ \psi_M = \widetilde{\gamma}\widetilde{\varphi} = \widetilde{\varphi}\,\widetilde{\gamma}\,\widetilde{\varphi} = \psi\widetilde{\varphi} & , \\ \psi = \widetilde{\gamma}\,\psi = \widetilde{\varphi}\,\psi, & \\ \widetilde{\gamma} \leq \psi_m \leq \psi \leq \psi_M \leq \widetilde{\varphi} & . \end{cases}$$

The same theorem may be restated in another way. If $\mathcal{J}\mathrm{d}(\mathcal{B}) \neq \varnothing$ then let B_i be a family of elements of $\mathcal{B}$. We have $\vee\, B_i \in \sim B$, and thus $\widetilde{\gamma}(\vee\, B_i) = \vee B_i$. From the first relation above, it follows for any $\psi \in \mathcal{J}\mathrm{d}(\mathcal{B})$, that

$$\psi\,(\vee\, B_i) = \psi\widetilde{\gamma}\,(\vee\, B_i) = \widetilde{\varphi}\,\widetilde{\gamma}\,(\vee\, B_i)\,.$$

But $\widetilde{\gamma}\,(\vee\, B_i) = \vee\, B_i$, so that

$$\widetilde{\varphi}\,(\vee B_i) = \psi\,(\vee B_i) \in \mathcal{B}.$$

In the same way, we also obtain

$$\widetilde{\gamma}\widetilde{\varphi}\,(\wedge\, B_i) = \widetilde{\gamma}\,(\wedge\, B_i) = \psi\,(\wedge\, B_i) \in \mathcal{B}.$$

In other words, $\mathcal{B}$ is a *complete lattice* with respect to the ordering on $\mathcal{B}$ induced by $\leq$, i.e. any family B_i in $\mathcal{B}$ has a smallest upper bound $\widetilde{\varphi}\,(\vee\, B_i)\,\mathcal{B}$ and a greatest lower bound $\widetilde{\gamma}\,(\wedge\, B_i) \in \mathcal{B}$.

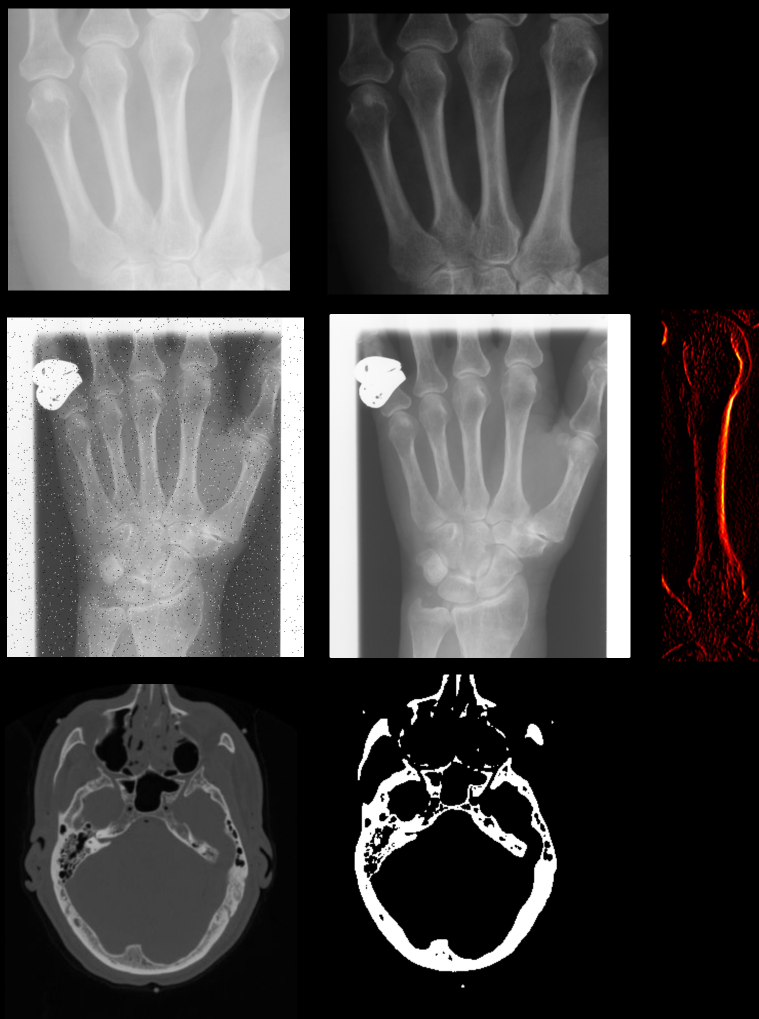
Conversely, let us assume that $\mathcal{B}$ is a complete lattice. Thus, for any $A \in \mathcal{L}$, the family $\{B : B \in \mathcal{B}, B \geq A\}$ has in $\mathcal{B}$ a greatest lower bound, which is

$$\widetilde{\gamma}\,(\wedge\,\{B : B \in \mathcal{B}, B \geq A\}) = \widetilde{\gamma}\,\widetilde{\varphi}\,(A) \in \mathcal{B}.$$

But this implies $\mathcal{B}_{\psi_M} \subseteq \mathcal{B}$ for the filter $\psi_M = \widetilde{\gamma}\,\widetilde{\varphi}$. Conversely, for any

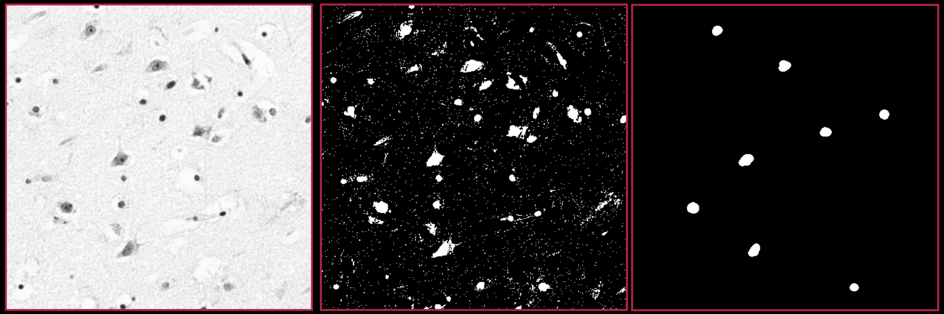
Do not worry! We use a much less theoretical approach!

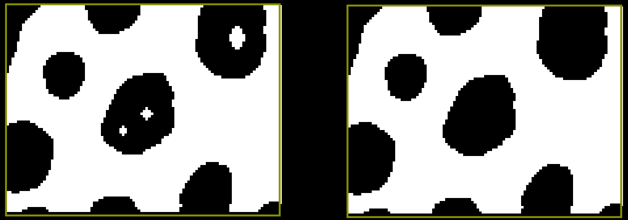
Relevance?



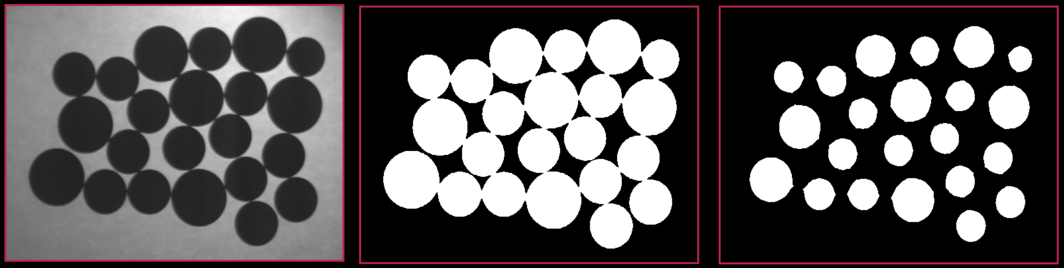
- Point wise operations
- Filtering
- Thresholding
  - Gives us objects that are separated by the background
- Morphology
  - Manipulate and enhance binary objects

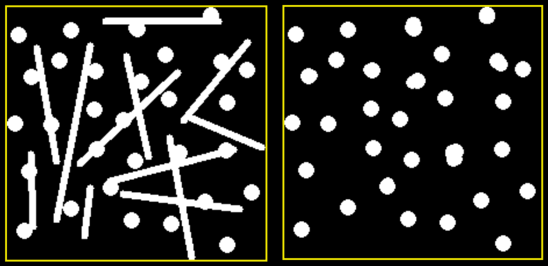
What can it be used for?



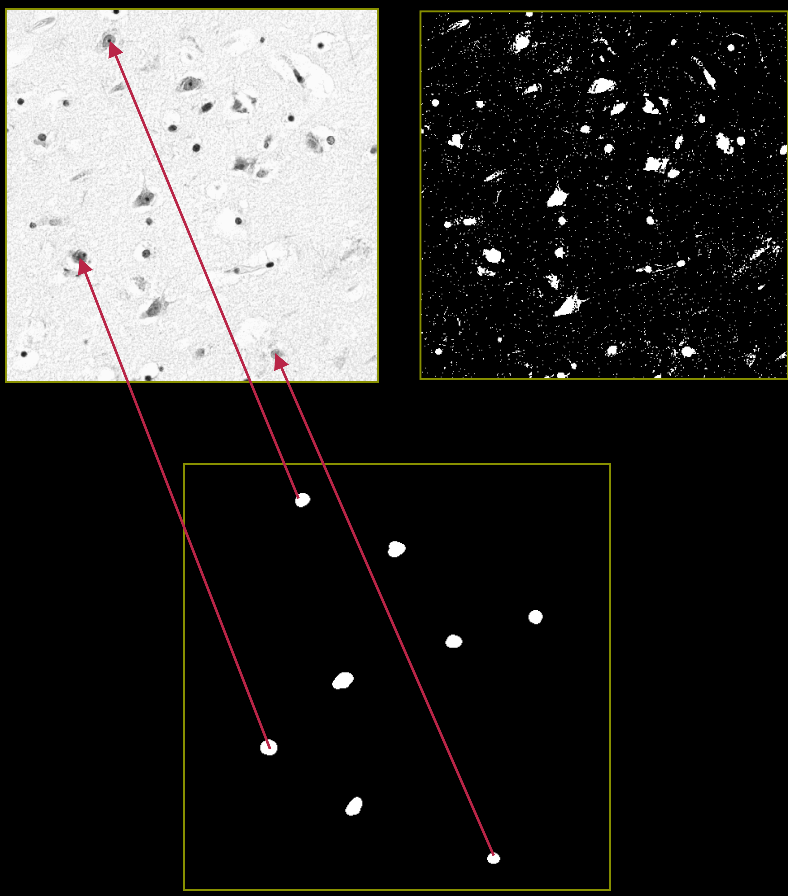


- Remove noise
  - Small objects
  - Fill holes
- Isolate objects
- Customized to specific shapes



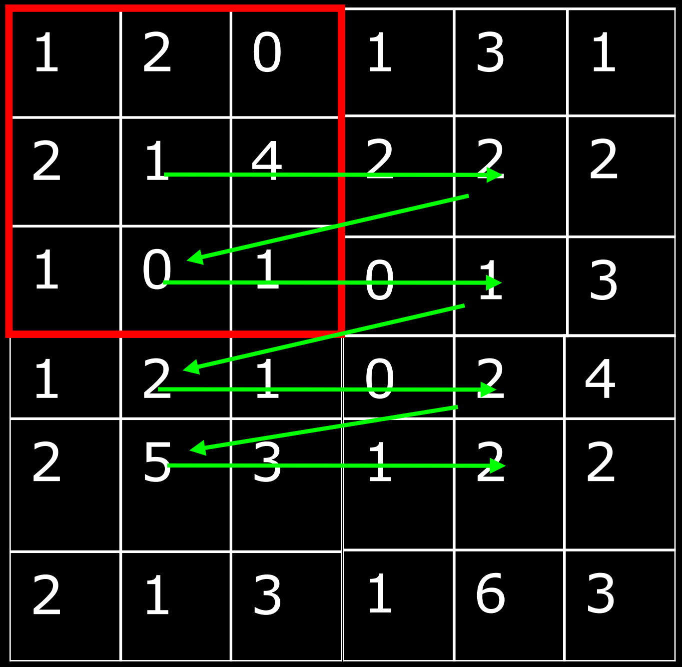


How does it work?



- Grayscale image
- Preprocessing
  - Inversion
- Threshold => Binary image
- Morphology

Filtering and morphology



- Filtering
  - Gray level images
  - Kernel
  - Moves it over the input image
  - Creates a new output image

Filtering and morphology

0	1	0
1	1	1
0	1	0

Disk

1	1	1
1	1	1
1	1	1

Box

- Filtering
  - Gray level images
  - Kernel
  - Moves it over the input image
  - Creates a new output image
- Morphology
  - Binary images
  - Structuring element (SE)
  - Moves the SE over the input image
  - Creates a new binary output image

1D Morphology

Input image

1	0	0	0	1	1	1	0	1	1

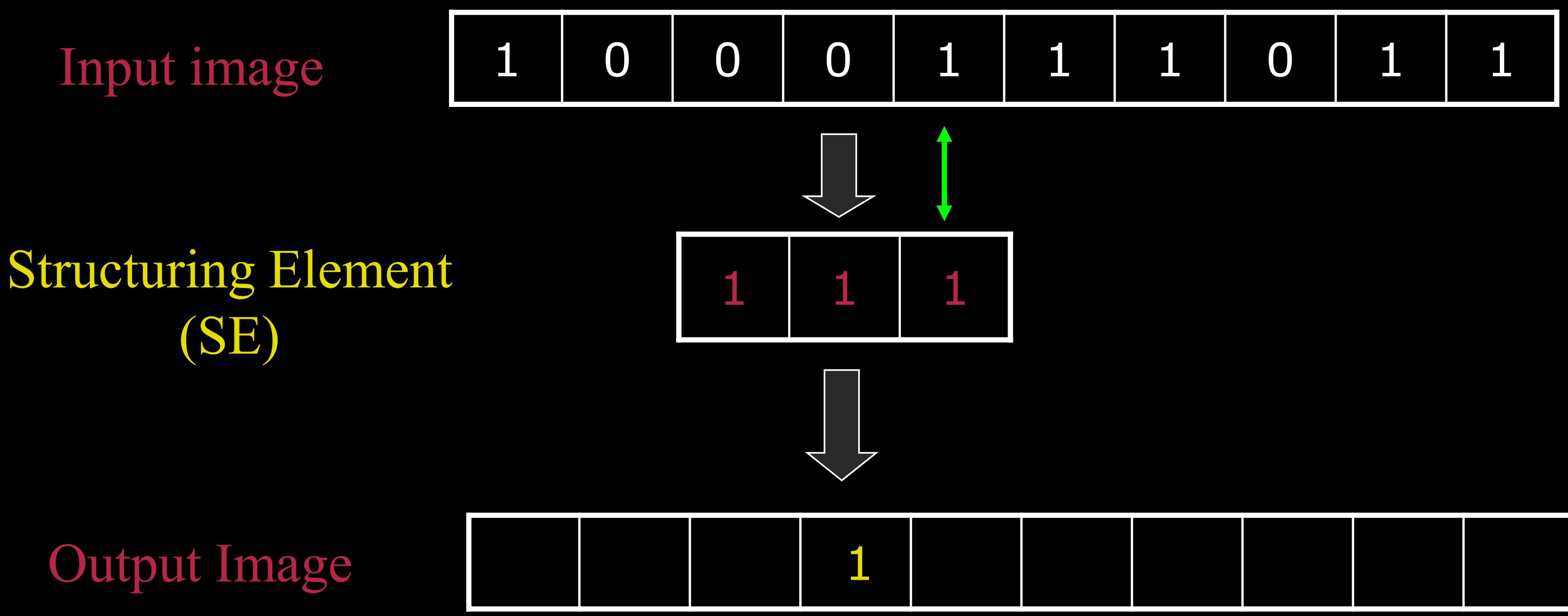
Structuring Element (SE)

1	1	1

Output Image

			?						

1D Morphology : The hit operation



- If just one 1 in the SE match with the input
    - output 1
- else
    - output 0

1D Morphology : The fit operation

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element (SE)

1	1	1

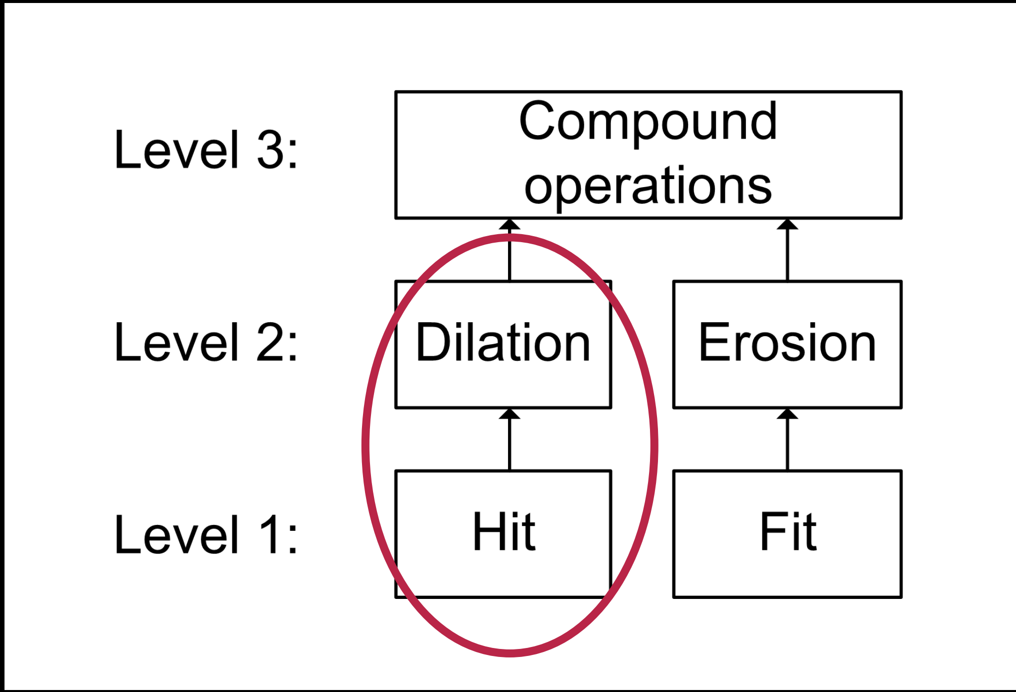
Output Image

			0						

- If all 1 in the SE match with the input
  - output 1
- else
  - output 0

1D Morphology : Dilation

- Dilate : To make wider or larger
  - Dansk : udvide
- Based on the *hit* operation



1D Dilation example

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

$$g(x) = f(x) \oplus SE$$

to make bigger

Output Image

	1								

Hit

- If just one 1 in the SE match with the input
  - output 1
- else
  - output 0

Example for Dilation

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	1	0							

Example for Dilation

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	1	0	1						

Example for Dilation

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	1	0	1	1					

Example for Dilation

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	1	0	1	1	1				

Example for Dilation

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	1	0	1	1	1	1			

Example for Dilation

Input image

1	0	0	0	1	1	1	0	1	1

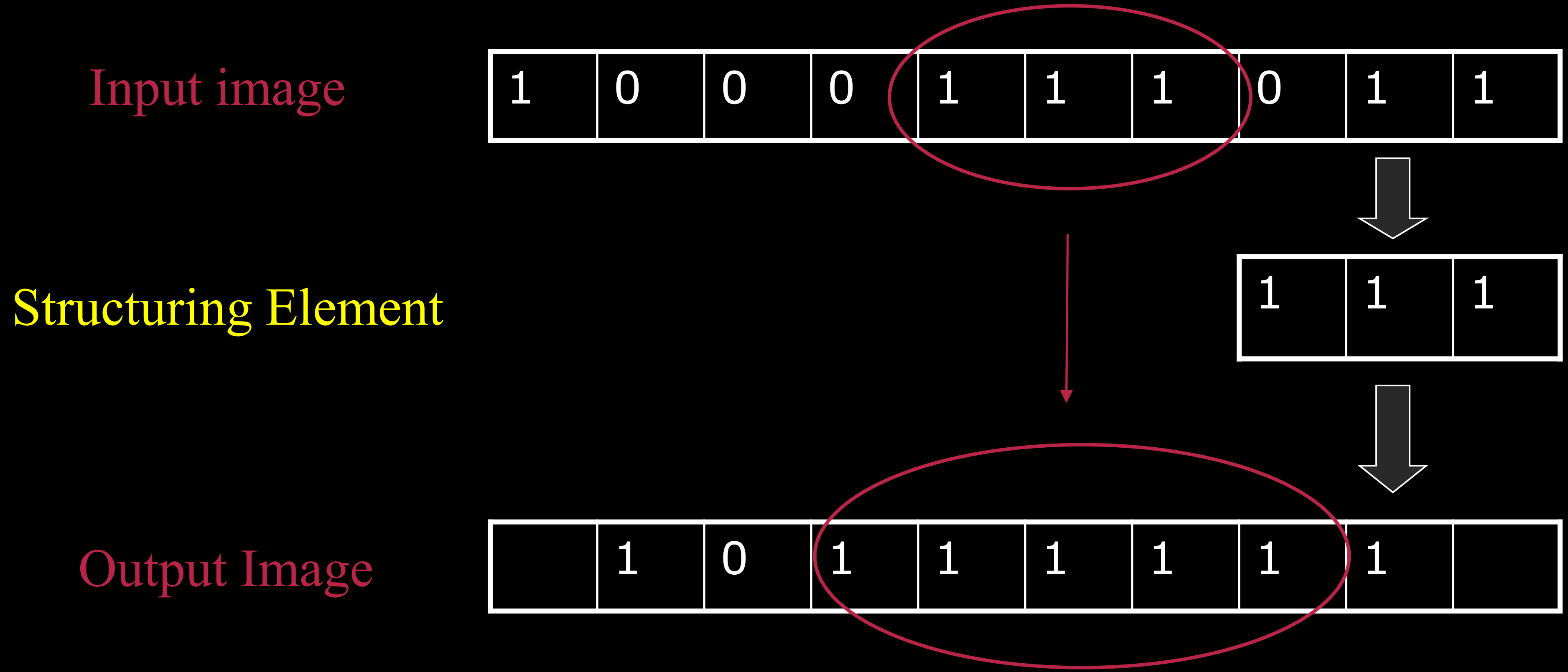
Structuring Element

1	1	1

Output Image

	1	0	1	1	1	1	1		

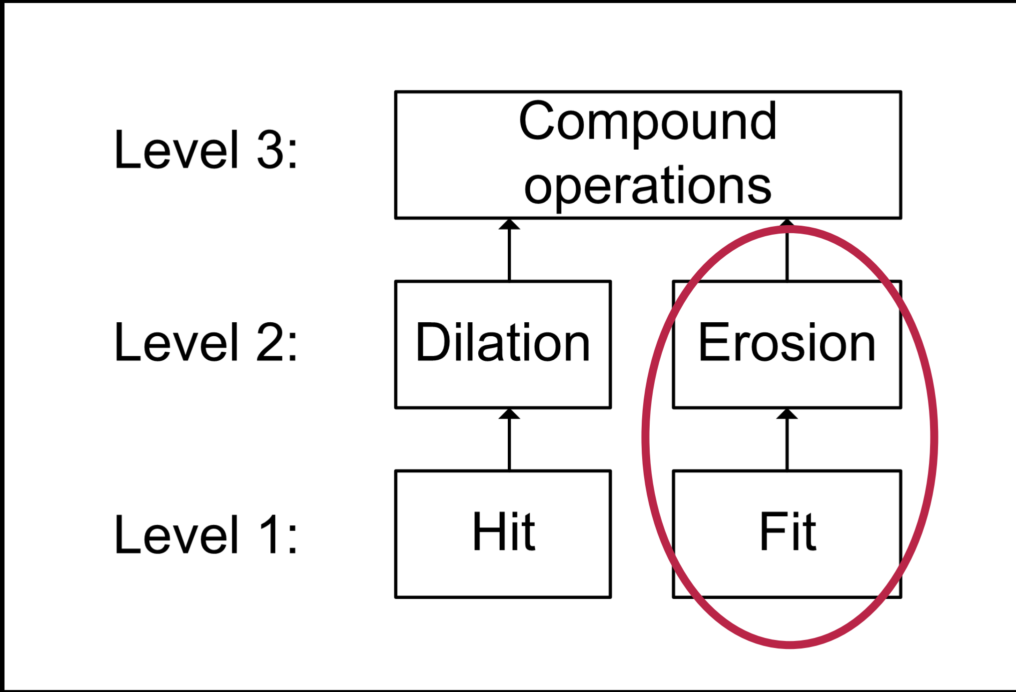
Example for Dilation



The object gets bigger and holes are filled!

1D Morphology : Erosion

- Erode : To wear down (*Waves eroded the shore*)
  - Dansk : tære, gnave
- Based on the *fit* operation



Example for Erosion

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

$$g(x) = f(x) \ominus SE$$

to make smaller

Output Image

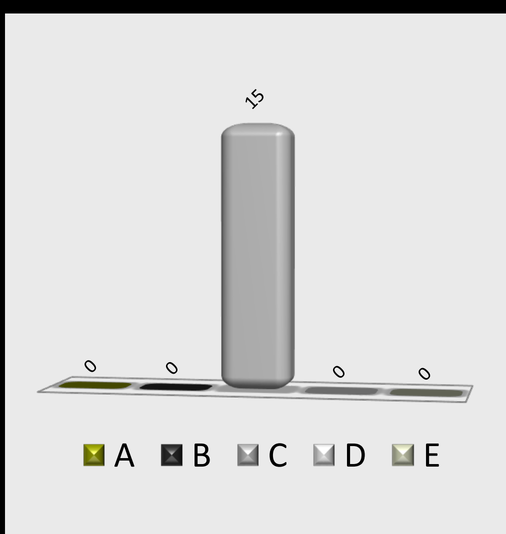
	0								

Fit

- If all 1 in the SE match with the input
  - output 1
- else
  - output 0

Erosion

A) 0 1 0 0 1 1 0 0
B) 0 0 1 0 1 0 0 0
C) 0 0 0 0 1 0 0 0
D) 0 0 1 0 0 0 0 1
E) 0 1 0 0 0 1 0 0



Input image

1	0	0	0	1	1	1	0	1	1

1	1	1

	0	0							

Output Image

Example for Erosion

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	0	0	0						

Example for Erosion

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	0	0	0	0					

Example for Erosion

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	0	0	0	0	1				

Example for Erosion

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	1	1

Output Image

	0	0	0	0	1	0			

Example for Erosion

Input image

1	0	0	0	1	1	1	0	1	1

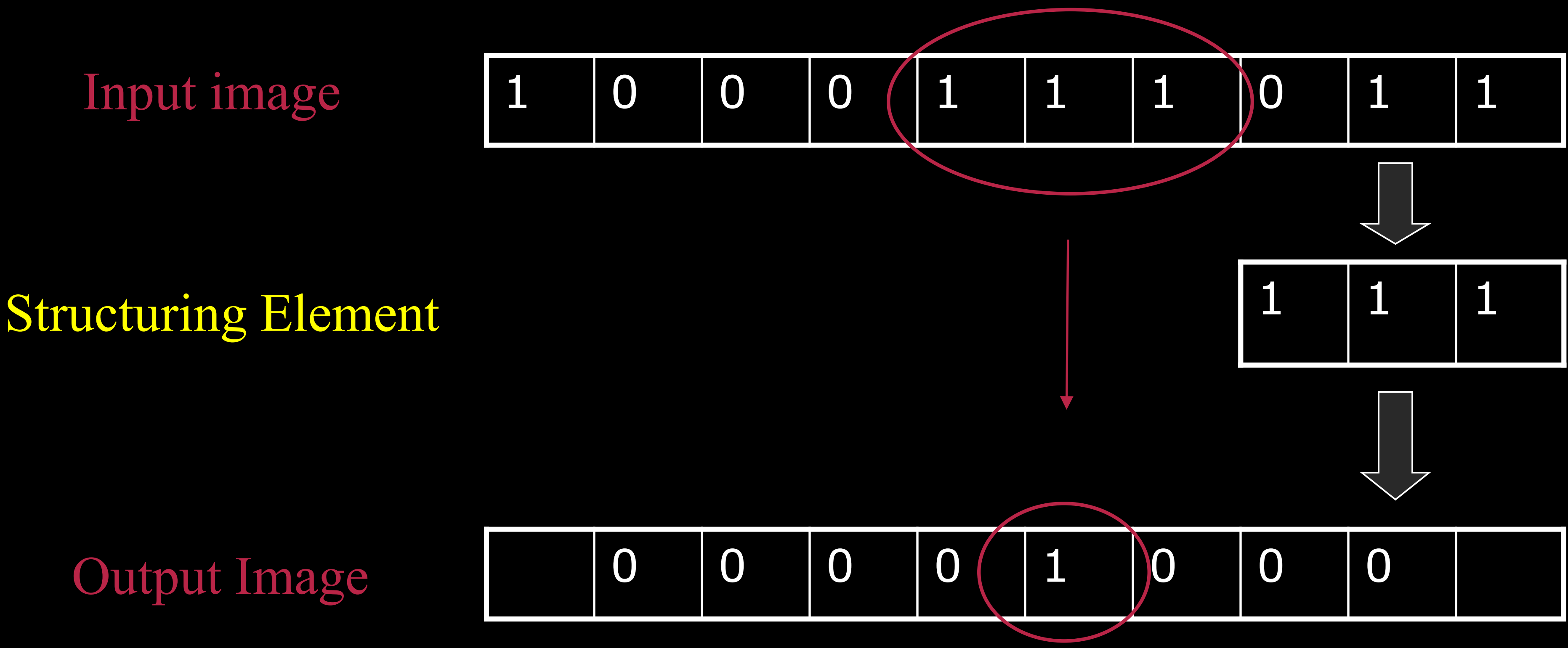
Structuring Element

1	1	1

Output Image

	0	0	0	0	1	0	0		

Example for Erosion



The object gets smaller

Structuring Element (Kernel)

0	1	0
1	1	1
0	1	0

Disk

1	1	1
1	1	1
1	1	1

Box

		1	1	1		
	1	1	1	1	1	
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
	1	1	1	1	1	
		1	1	1		

- Structuring Elements can have varying sizes
- Usually, element values are 0 or 1, but other values are possible (including none!)
- Structural Elements have an origin
- Empty spots in the Structuring Elements are *don't cares*!

Structuring Element Origin

0	(1)	0
1	1	1
0	1	0

(1)	1	1
1	1	1
1	1	1

- The origin is not always the center of the SE

Special structuring elements

- Structuring elements can be customized to a specific problem

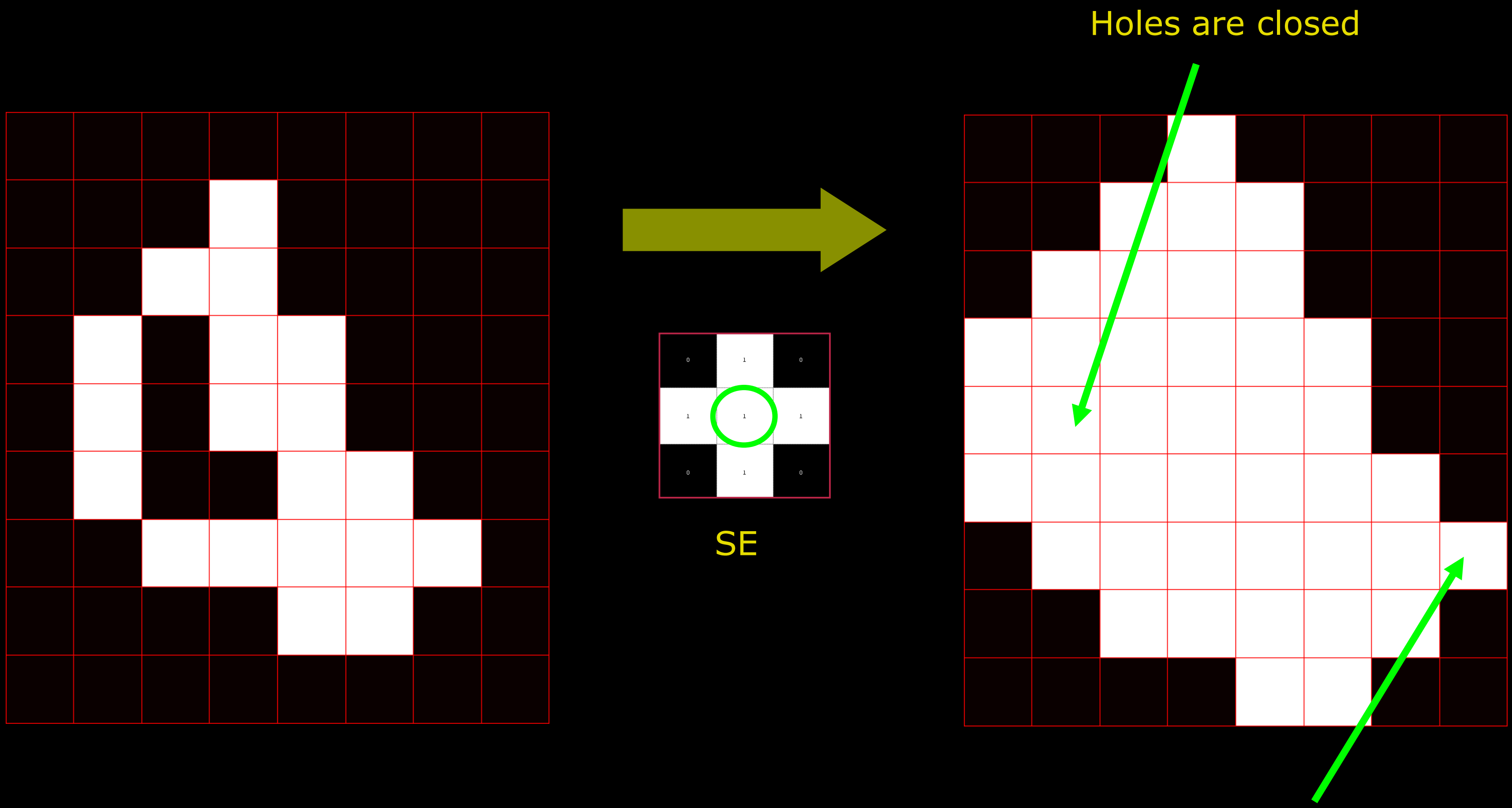
0	0	0	1	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
1	1	1	1	1	1	1
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	1	0	0	0

Diamond

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

Line

Dilation on images - disk



$$g(x,y) = f(x,y) \oplus SE$$

Dilation on images - box

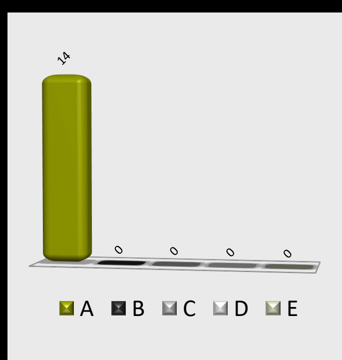
A) 1 0 1 1

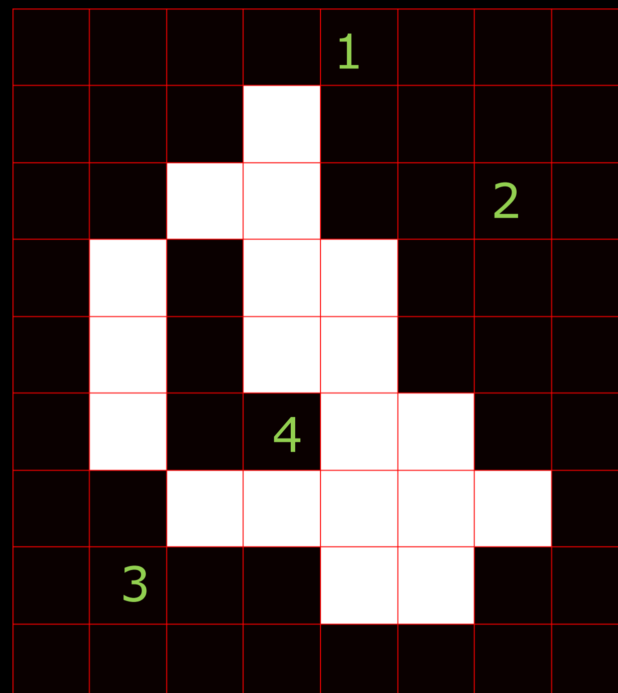
B) 0 1 0 0

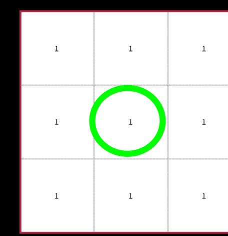
C) 0 1 1 1

D) 0 1 1 1

E) 1 1 0 1



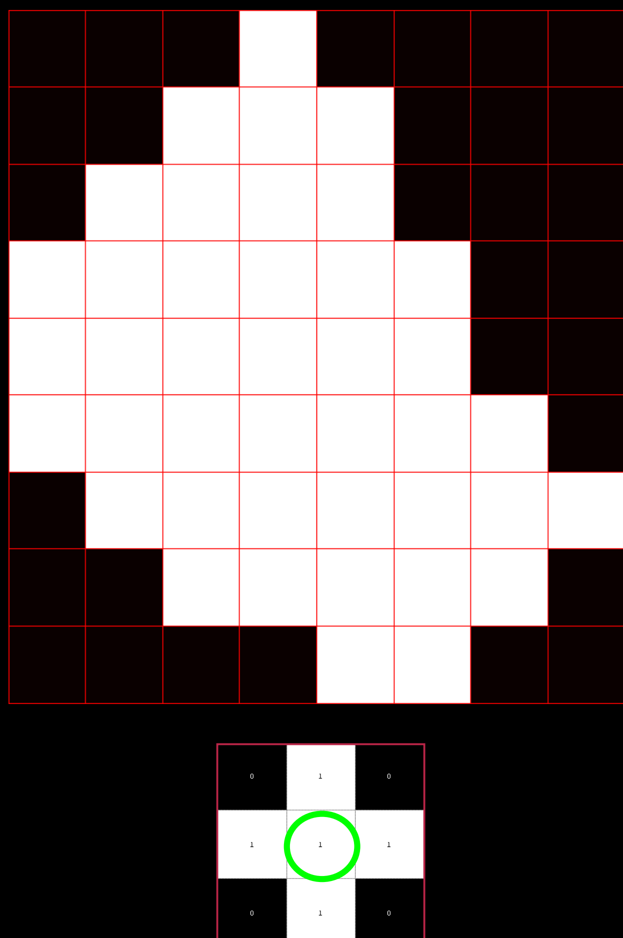


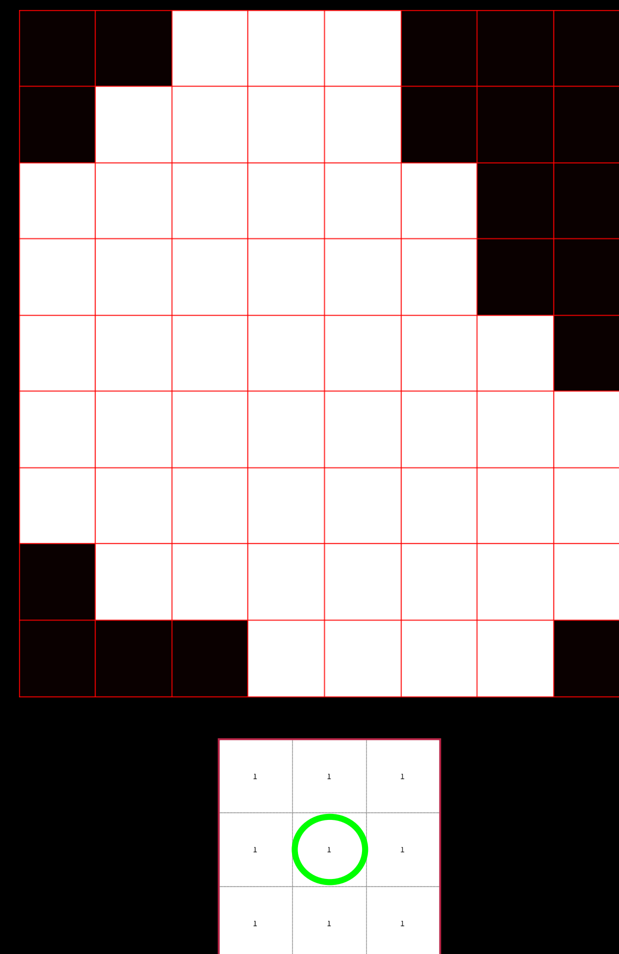


SE

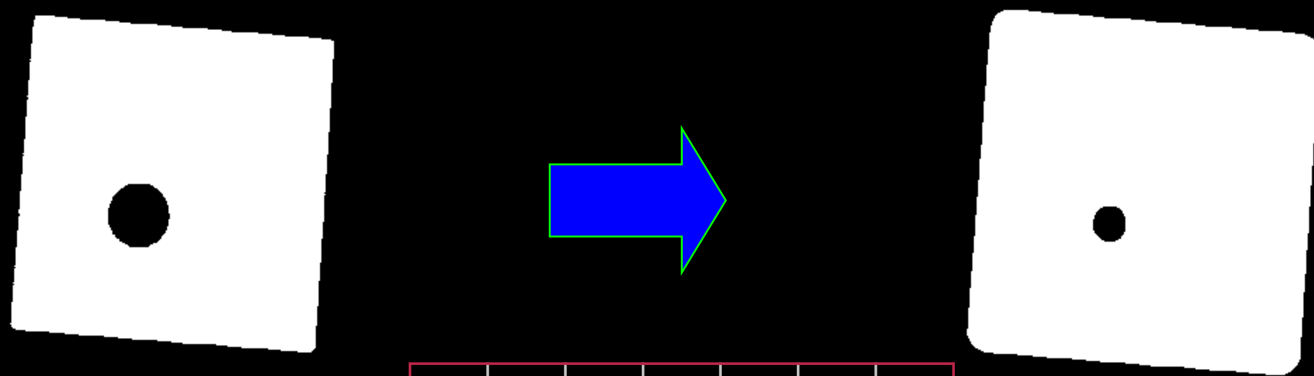
$$g(x,y) = f(x,y) \oplus SE$$

Dilation – the effect of the SE





Dilation Example



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

- Round structuring element (disk)
- Creates round corners

Threshold and dilation

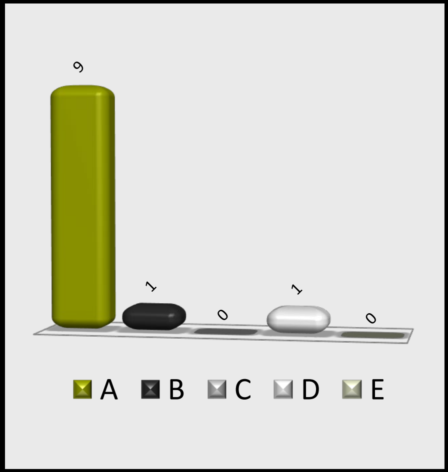
A) 14

B) 17

C) 6

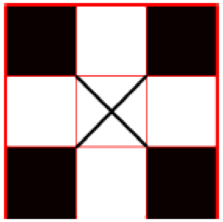
D) 3

E) 12



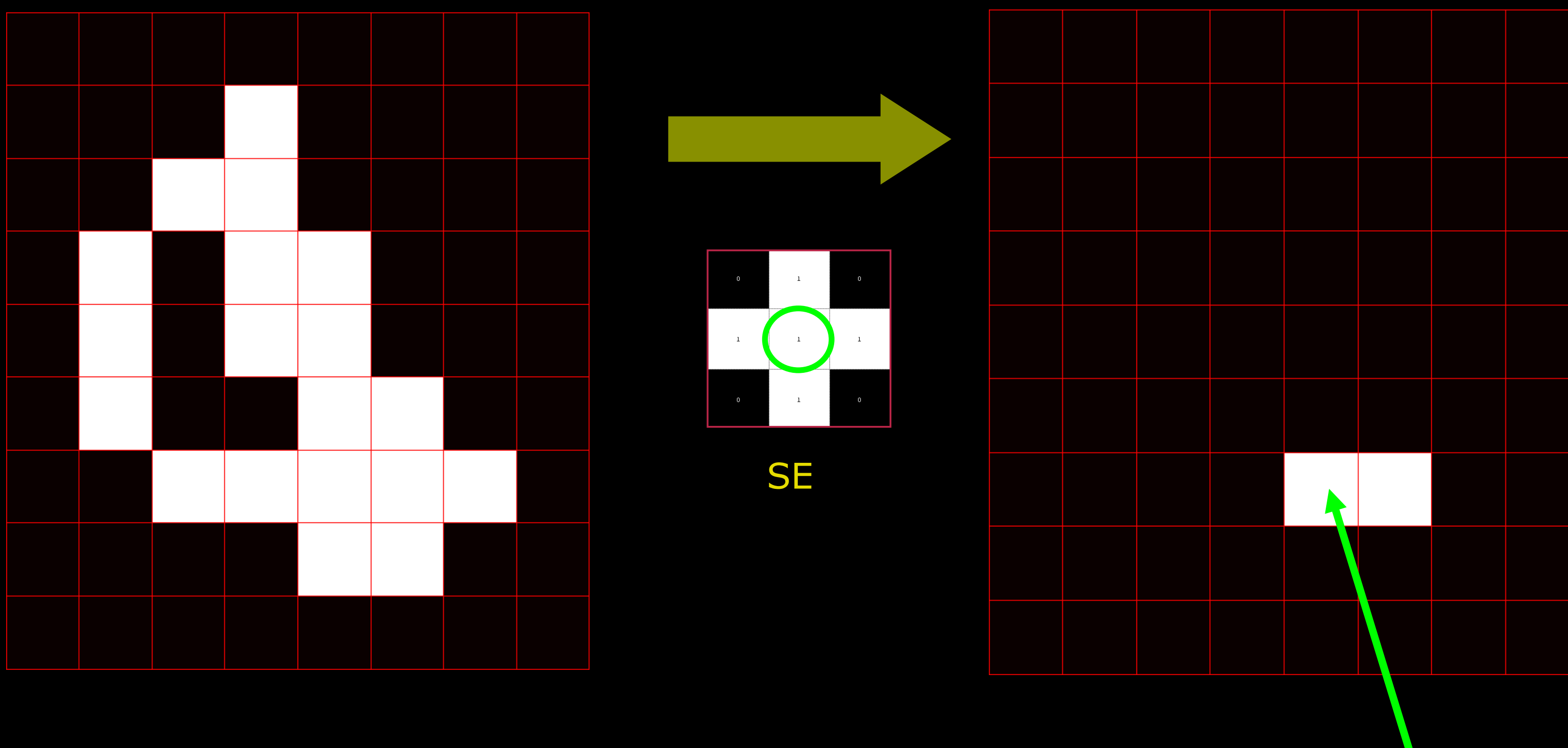
A threshold of 200 is applied to the image and the result is a binary image. Now a dilation is performed with the structuring element below. How many foreground pixels are there in the resulting image?

145	56	86	42	191
19	33	41	255	115
14	240	203	234	21
135	120	209	167	58
199	3	135	176	116



1. 14
2. 17
3. 6
4. 3
5. 12

Erosion on images - disk



$$g(x, y) = f(x, y) \ominus SE$$

Object is smaller

Erosion on images - box

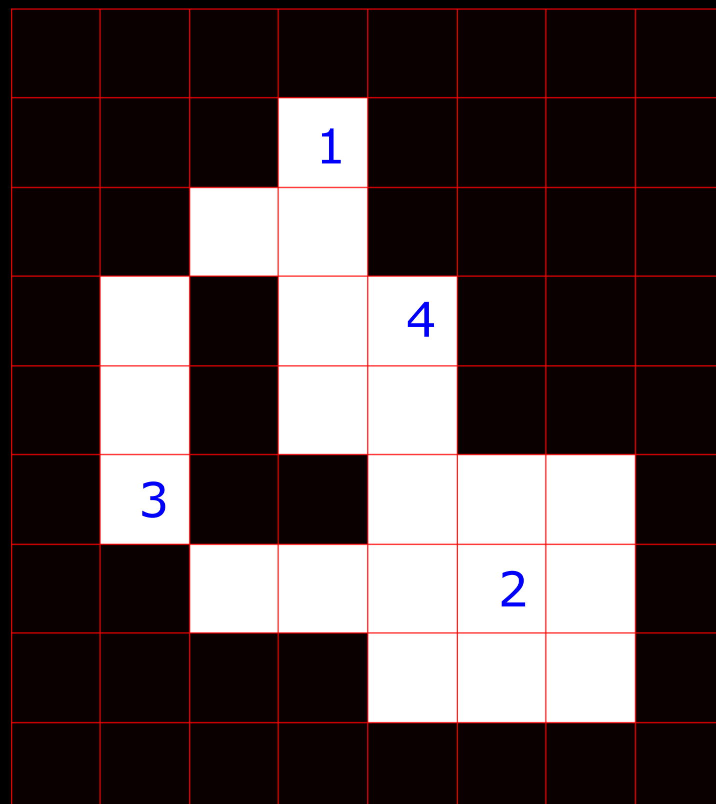
A) 0 0 1 0

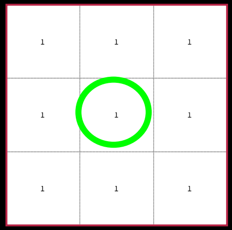
B) 1 0 1 0

C) 0 1 1 0

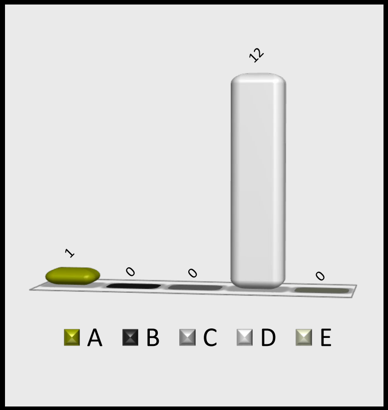
D) 0 1 0 0

E) 1 0 0 0



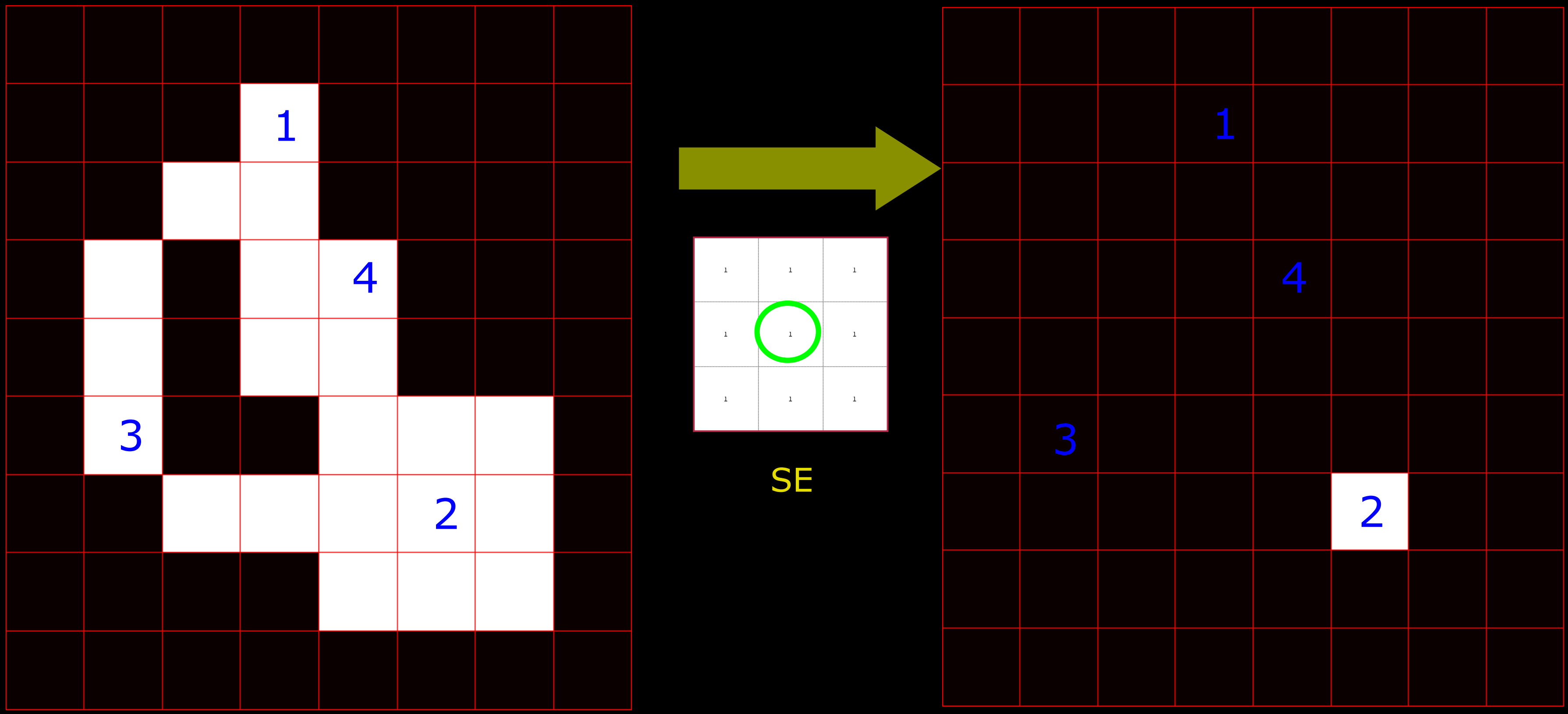


SE



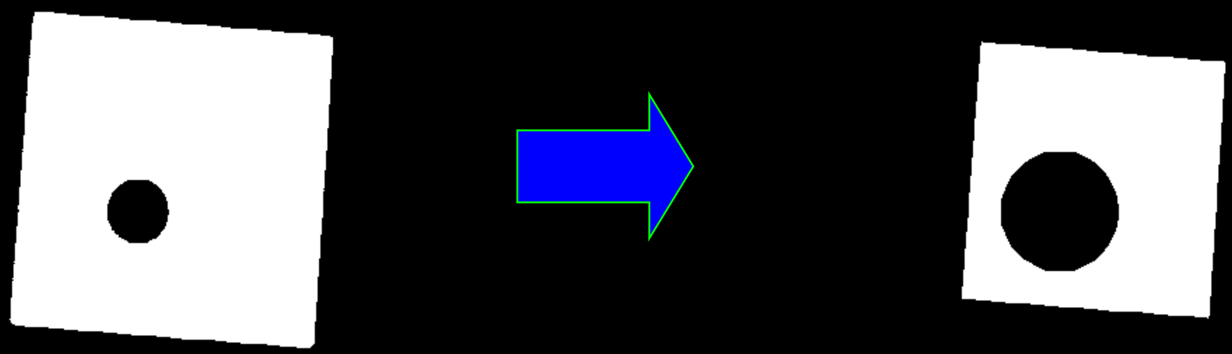
$$g(x, y) = f(x, y) \ominus SE$$

Erosion on images – box (square)



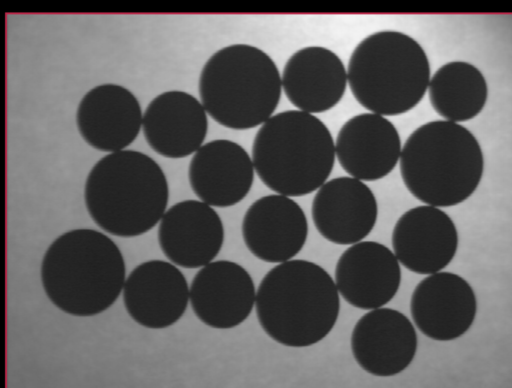
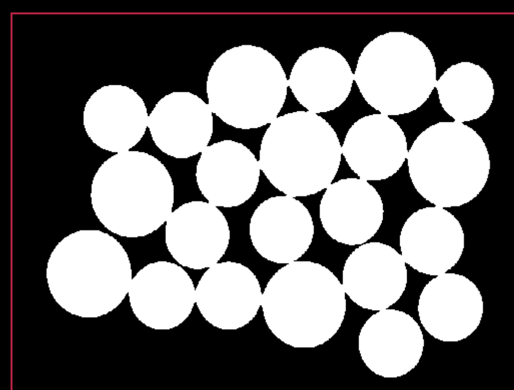
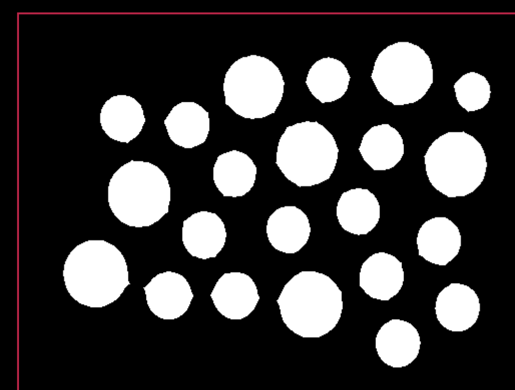
$$g(x,y) = f(x,y) \ominus SE$$

Erosion example

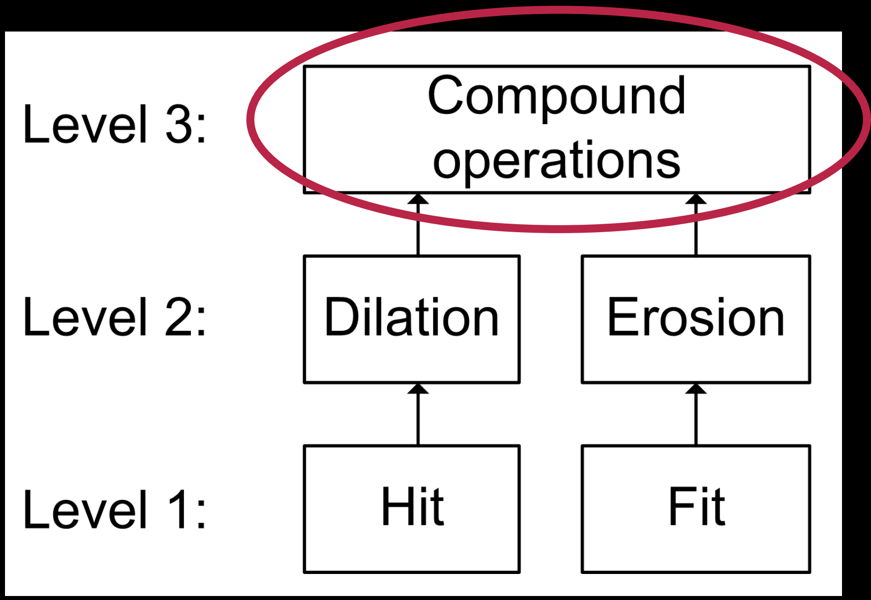


Counting Coins

- Counting these coins is difficult because they touch each other!
- Solution: Threshold and Erosion separates them!
- More on counting next time!

Compound operations

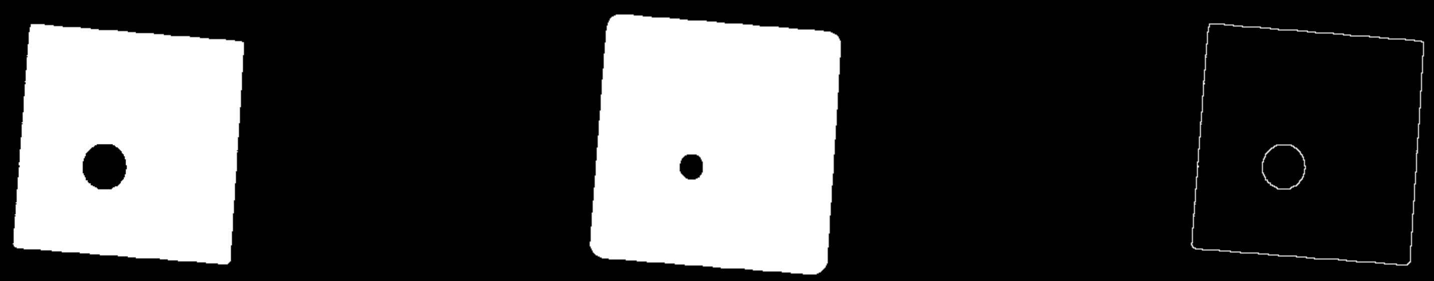


- Compound
  - *made of two or more separate parts or elements*
- Combining Erosion and Dilation into more advanced operations
  - Finding the outline
  - Opening
    - Isolate objects and remove small objects (better than Erosion)
  - Closing
    - Fill holes (better than Dilation)

Finding the outline

1. Dilate input image (object gets bigger)
2. Subtract input image from dilated image
3. The outline remains!

$$g(x,y) = (f(x,y) \oplus SE) - f(x,y)$$



Opening

- Motivation: Remove small objects BUT keep original size (and shape)
- Opening = Erosion + Dilation
  - Use the same structuring element!
  - Similar to erosion but less destructive
- Math:

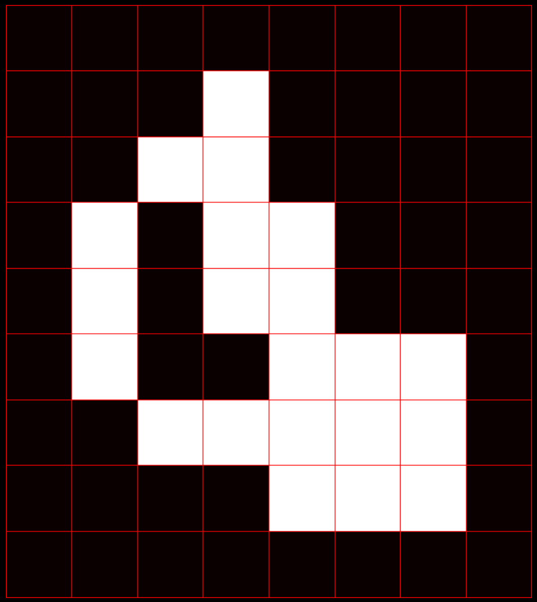
$$g(x,y) = f(x,y) \circ SE = (f(x,y) \ominus SE) \oplus SE$$

- Opening is **idempotent:** Repeated operations has no further effects!

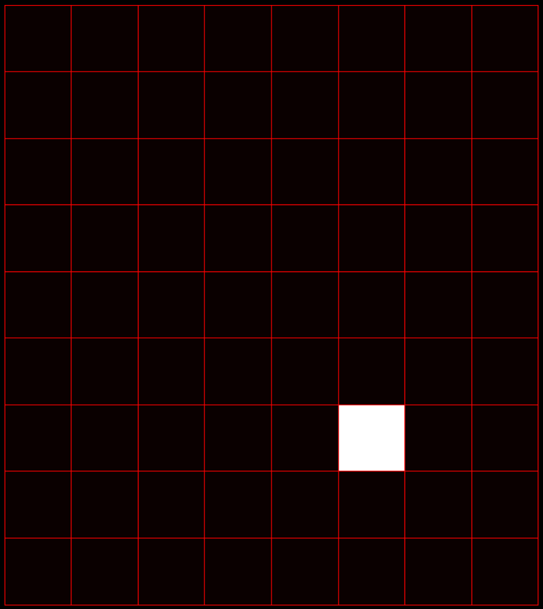
$$f(x,y) \circ SE = (f(x,y) \circ SE) \circ SE$$

Opening

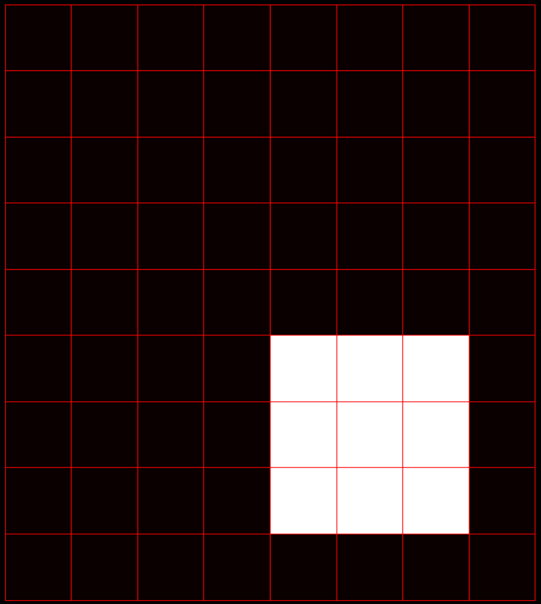
$$g(x,y) = (f(x,y) \ominus SE) \oplus SE$$



Original

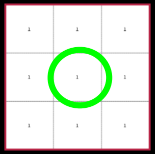


Eroded



Dilated

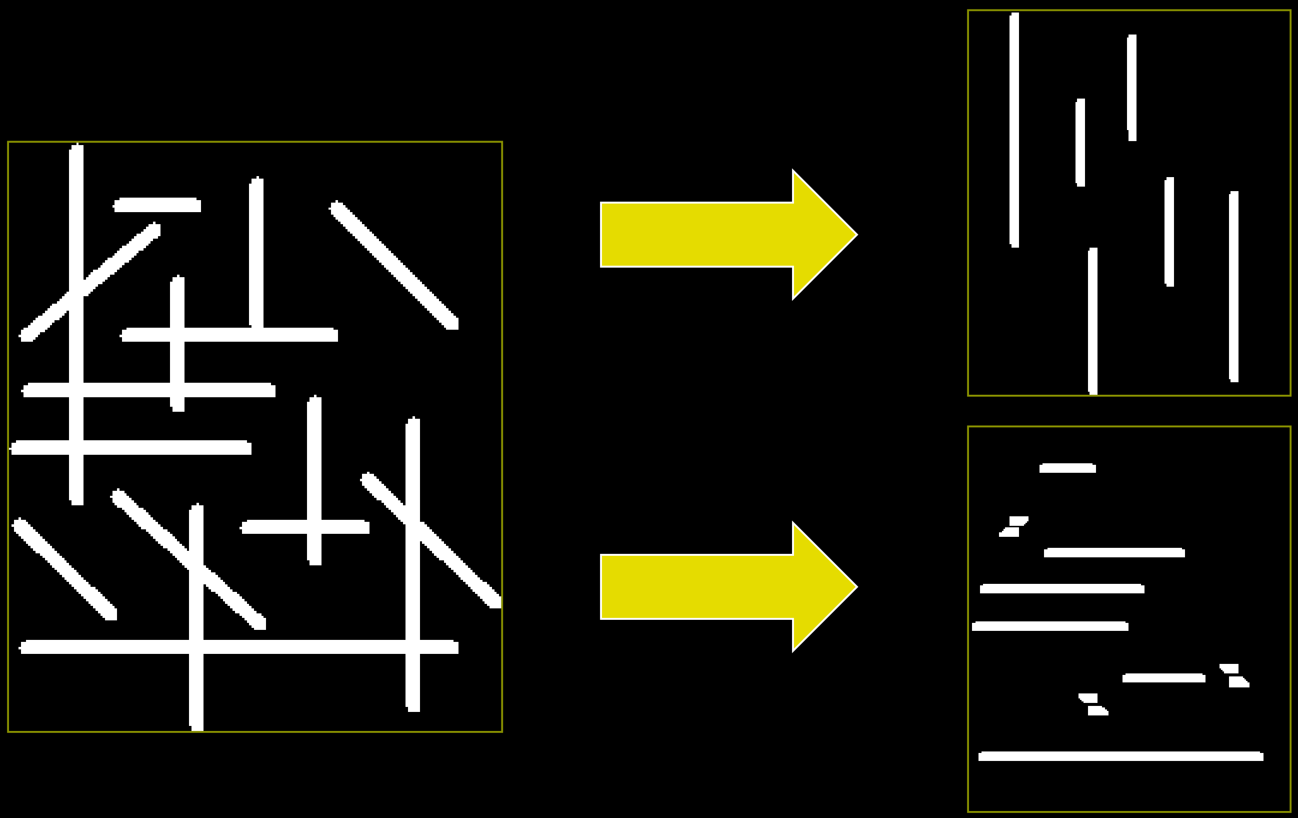
Opening = erosion+dilation



SE

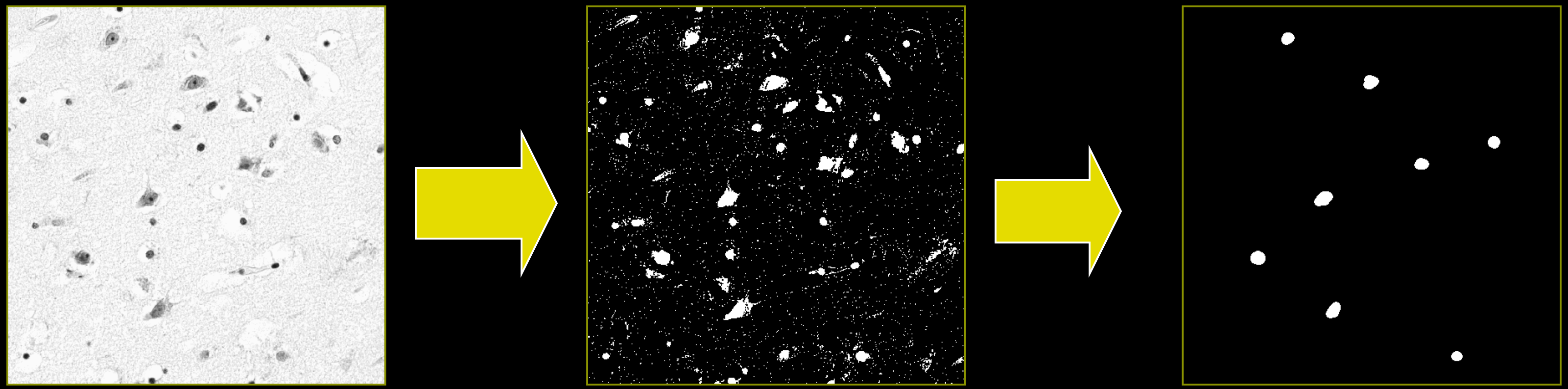
Opening Example

- 9x3 and 3x9 Structuring Elements



Opening example

- Size of structuring element should fit into the smallest object to keep
- Structuring Element: 11 pixel disc



Compound operations

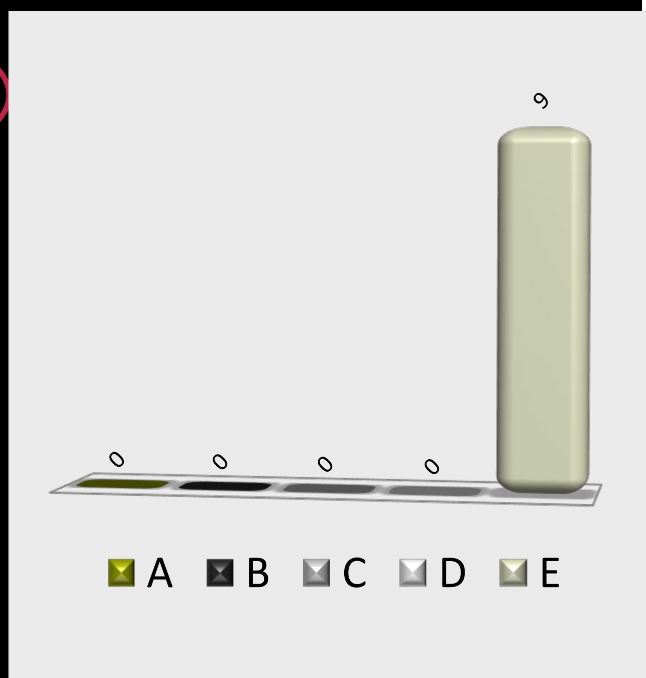
A) 3

B) 23

C) 11

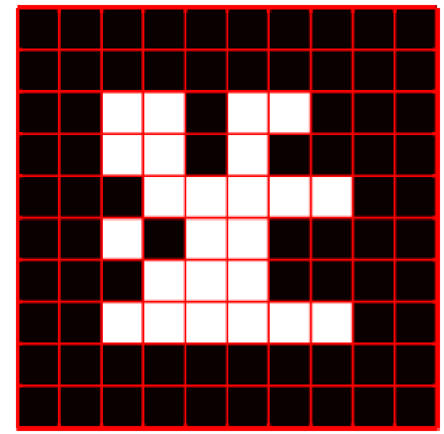
D) 36

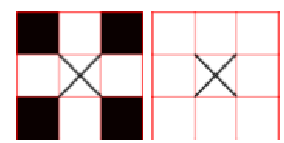
E) 16



The compound morphological operation seen below is applied to the image. How many foreground pixels are there in the resulting image?

$$(\mathrm{I} \ominus \mathrm{SE1}) \oplus \mathrm{SE2},$$





SE1 SE2

1. 3
2. 23
3. 11
4. 36
5. 16

Closing

- Motivation: Fill holes BUT keep original size (and shape)
- Closing = Dilation + Erosion
  - Use the same structuring element!
  - Similar to Dilation but less destructive
- Math:

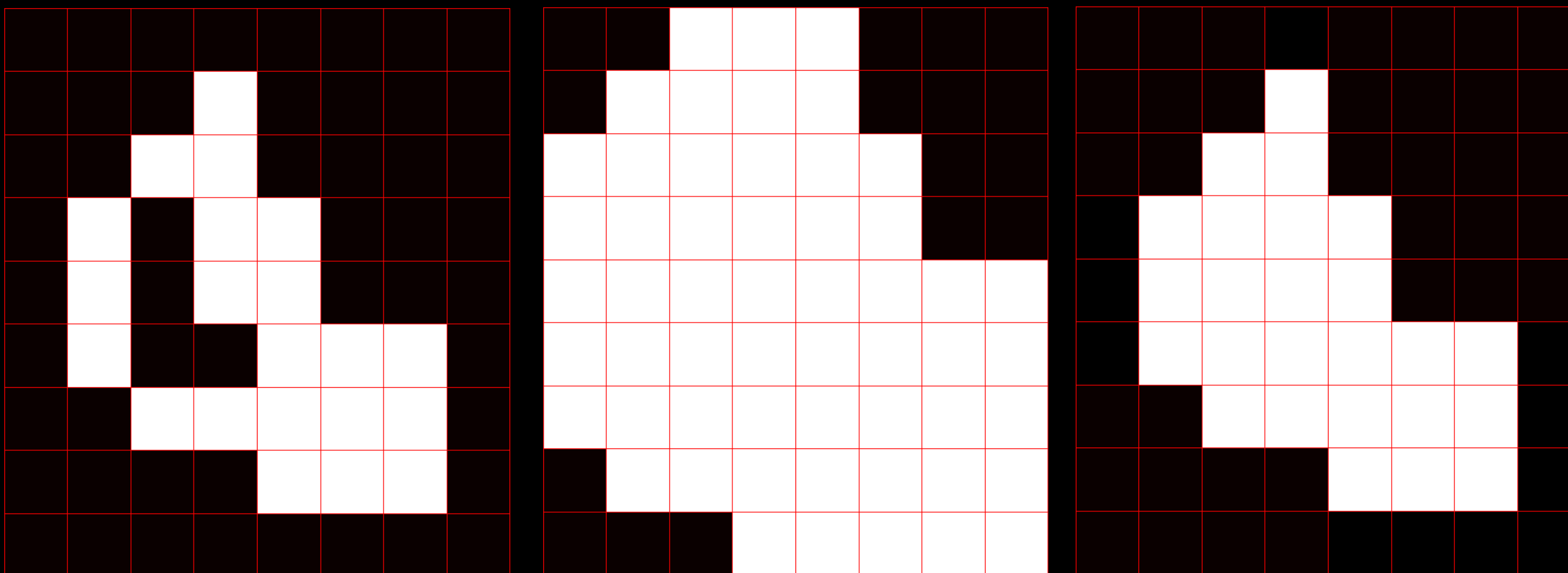
$$g(x,y) = f(x,y) \bullet SE = (f(x,y) \oplus SE) \ominus SE$$

- Closing is **idempotent:** Repeated operations has no further effects!

$$f(x,y) \bullet SE = (f(x,y) \bullet SE) \bullet SE$$

Closing

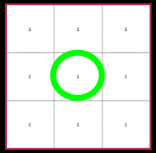
$$g(x,y) = (f(x,y) \oplus SE) \ominus SE$$



Original

Dilated

Eroded

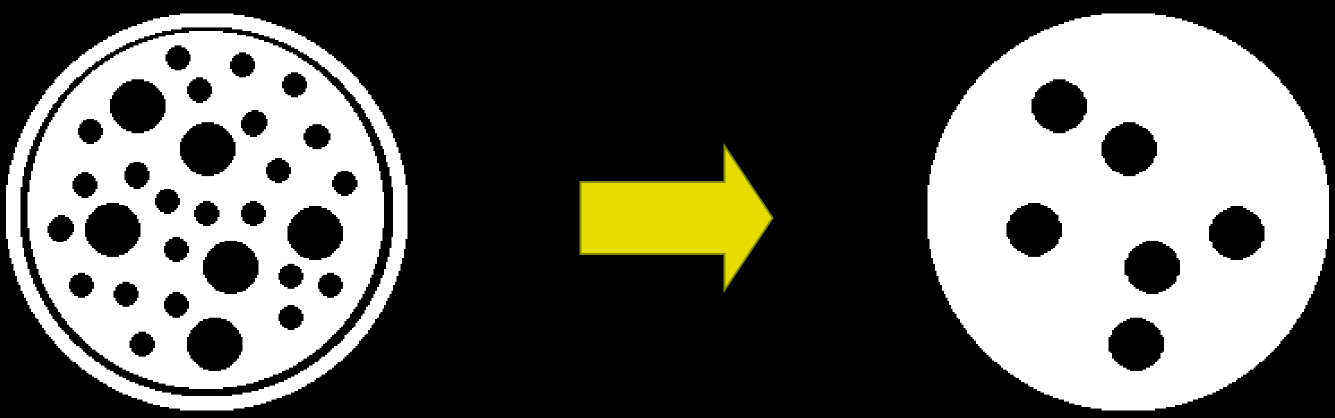


SE

Closing = dilation + erosion

Closing Example

- Closing operation with a 22 pixel disc
- Closes small holes



Closing

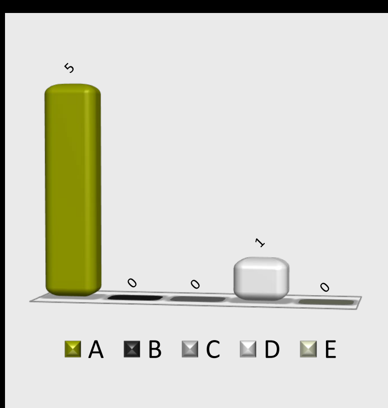
A) 31

B) 18

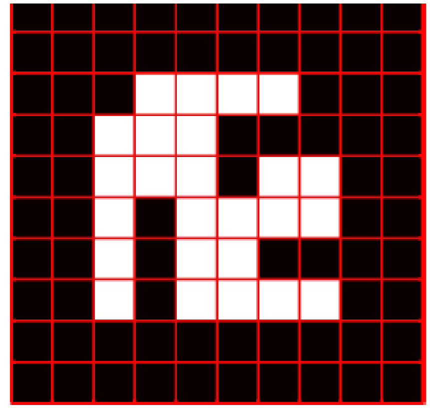
C) 6

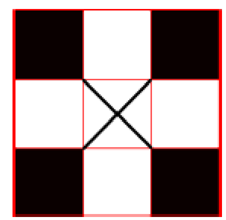
D) 35

E) 21



Morphological closing is applied to the image using the structuring element below. How many foregrounds pixels are there in the resulting image?





1. 31
2. 18
3. 6
4. 35
5. 21

How do you like the book?

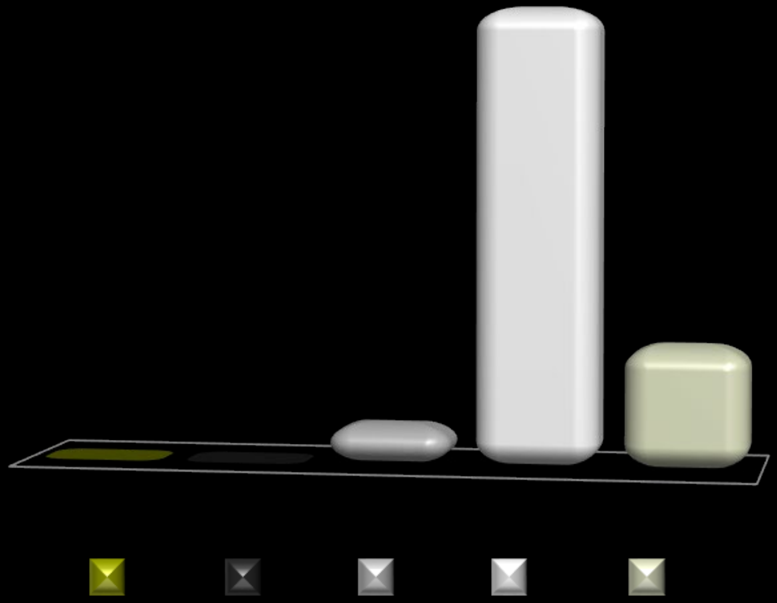
A) Very bad book

B) Bad book

C) Ok book

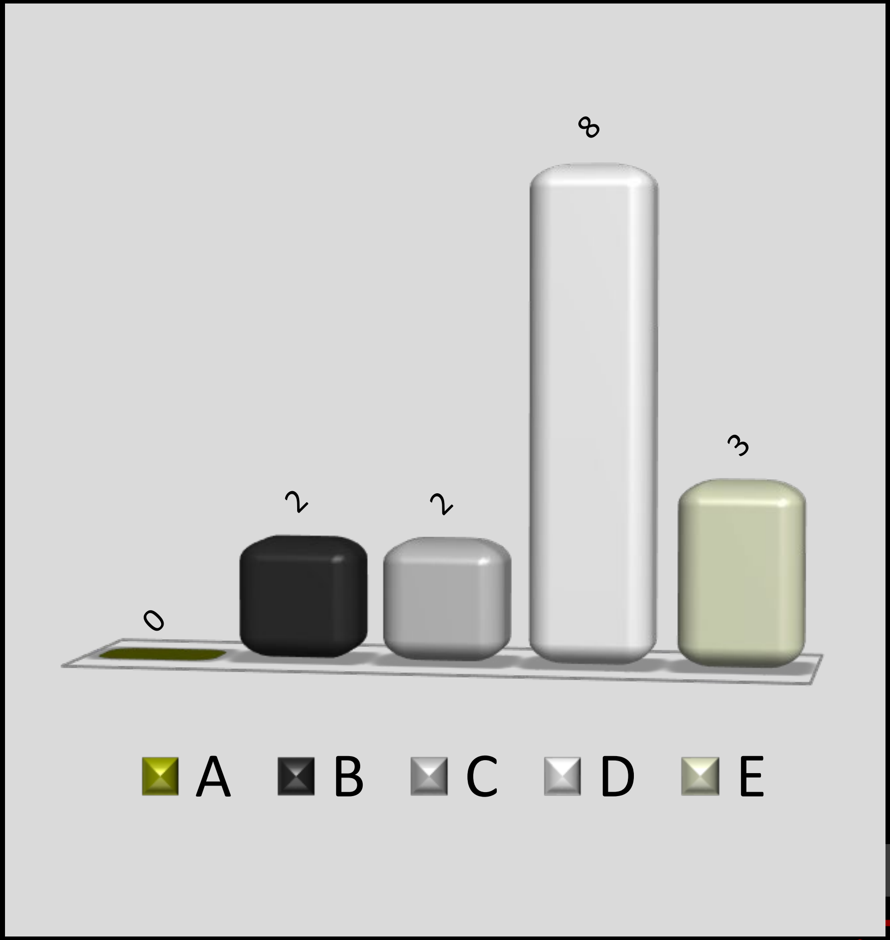
D) Good book

E) Really good book



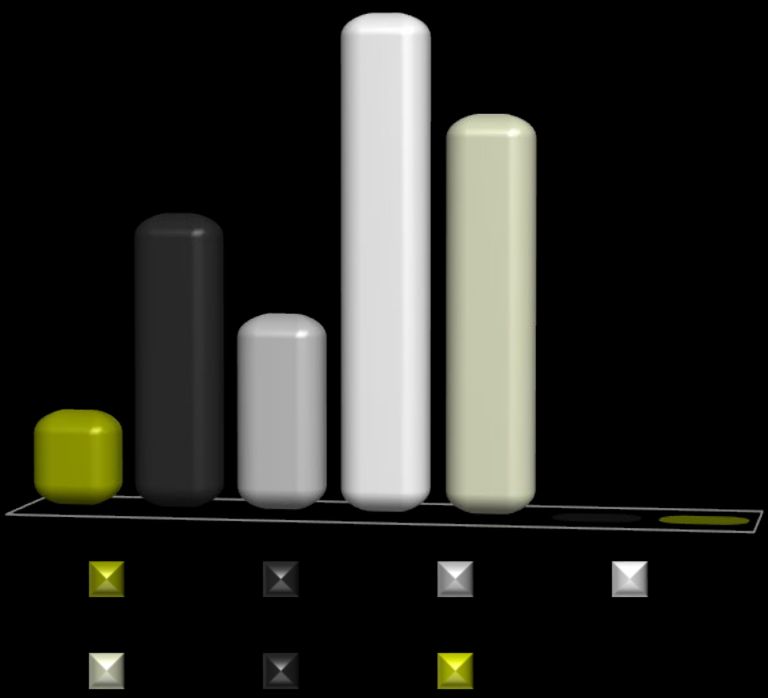
Flipped classroom
TA 8-10, Lecture 10-12

A) It really does not work
B) It is not optimal
C) It is ok
D) It is fine
E) It works very well



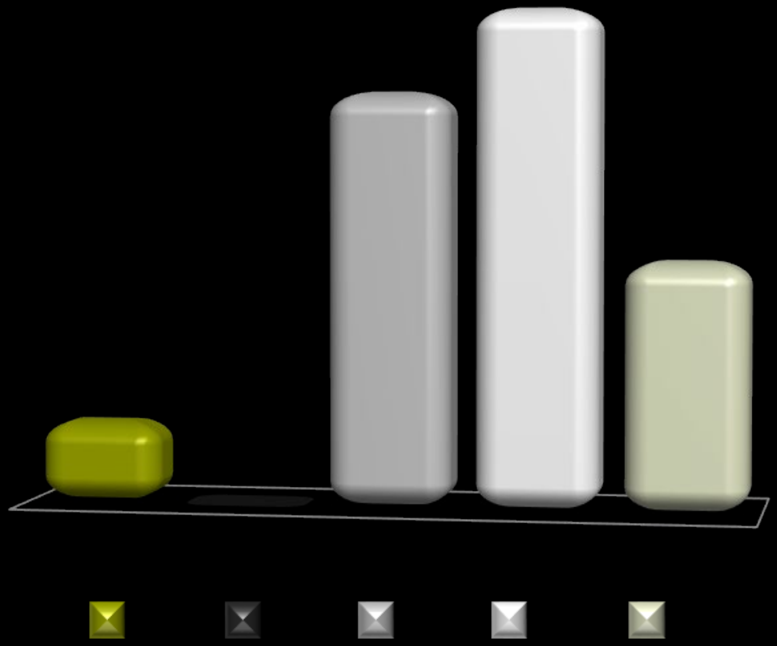
How much time do I spend on preparing every week?

A) 0 minutes
B) 0-15 minutes
C) 15-30 minutes
D) 30-60 minutes
E) 1-2 hours
F) 2-4 hours
G) More than 4 hours



How do I feel about Matlab

A) I simply do not get it
B) I find it hard
C) We are ok friends
D) I feel confident in Matlab
E) I write Matlab scripts even when I sleep



Next week: Blob Analysis