

# Image Processing: Background removal

Johannes Müller

With material from

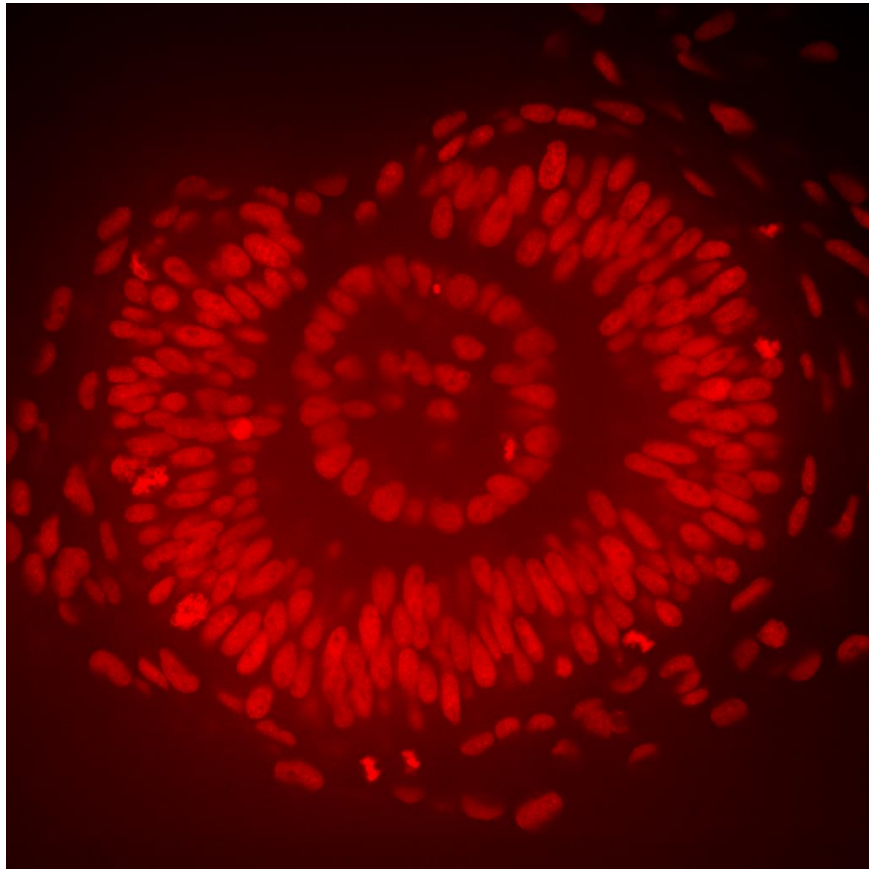
Marcelo Leomil Zoccoloer

Robert Haase, PoL

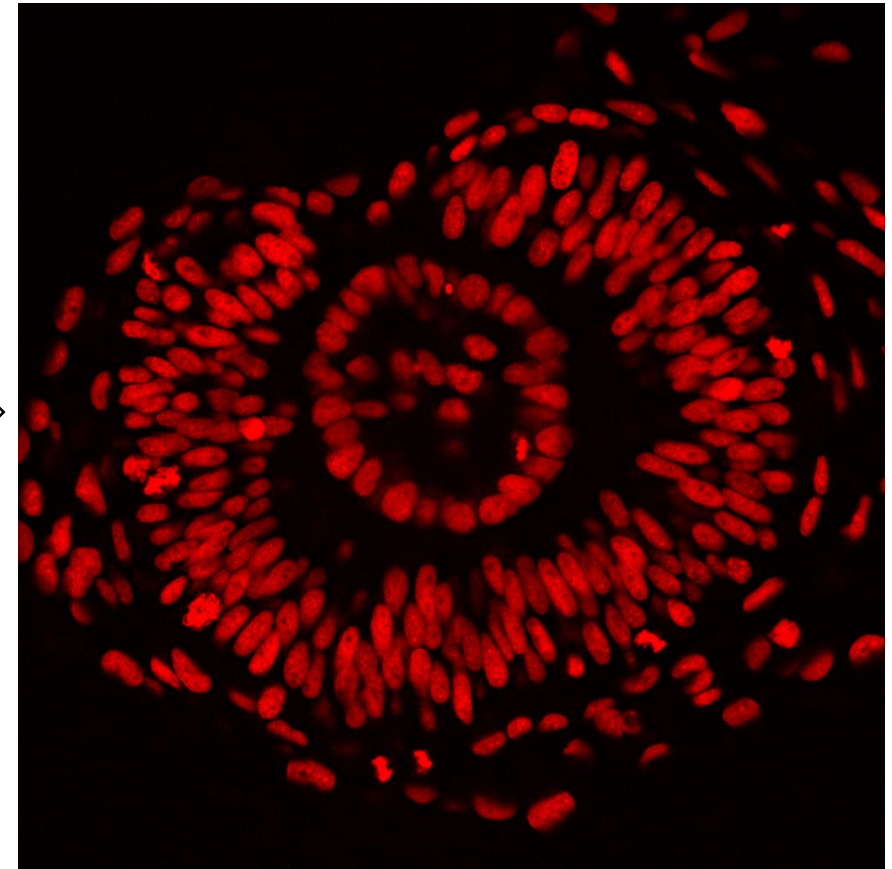
Mauricio Rocha Martins, Norden lab, MPI CBG

April 2022

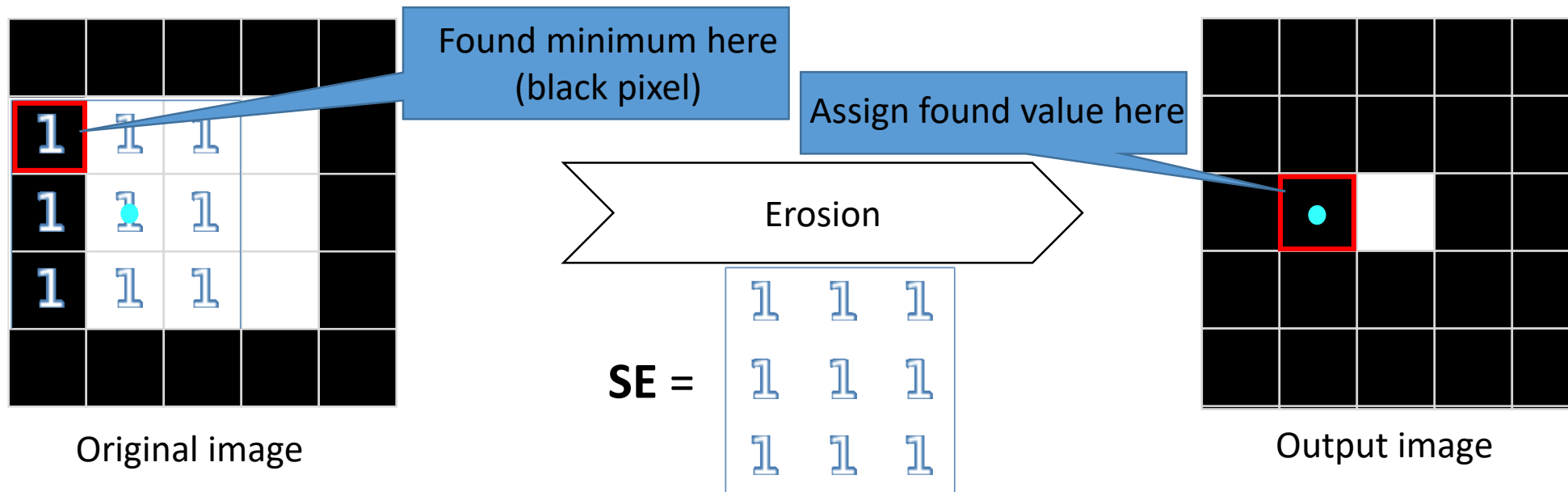
- Differentiating objects is easier if their background intensity is equal.



Subtract  
background



- Erosion: Every pixel with at least one black neighbor becomes black.

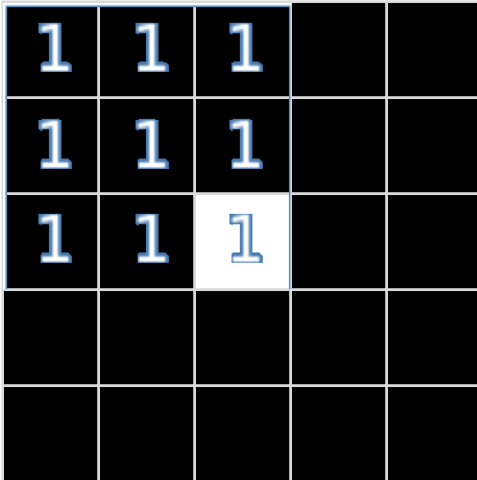


Erosion is essentially a minimum filter whose extent is defined by the kernel (structural element or **SE**) size and shape.

Dilation is essentially a maximum filter whose extent is defined by the kernel (structural element or **SE**) size and shape.

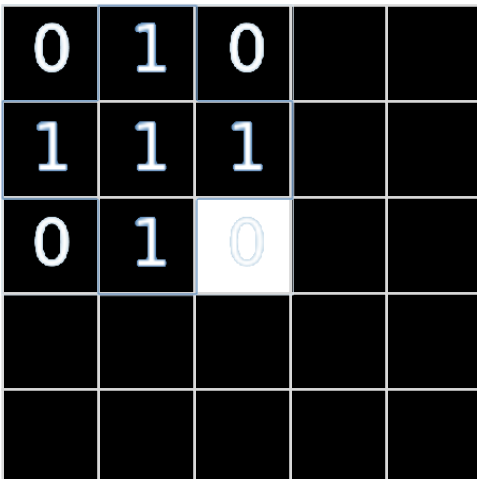
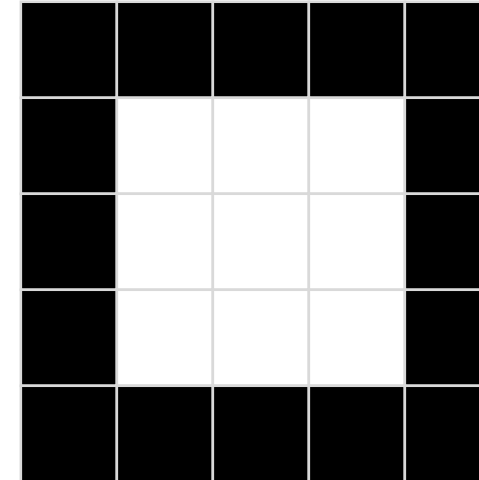
For an example with a grayscale image, check out this gif:  
[https://en.wikipedia.org/wiki/Erosion\\_\(morphology\)#/media/File:Grayscale\\_Morphological\\_Erosion.gif](https://en.wikipedia.org/wiki/Erosion_(morphology)#/media/File:Grayscale_Morphological_Erosion.gif)

- Dilation: Every pixel with at least one white neighbor becomes white.



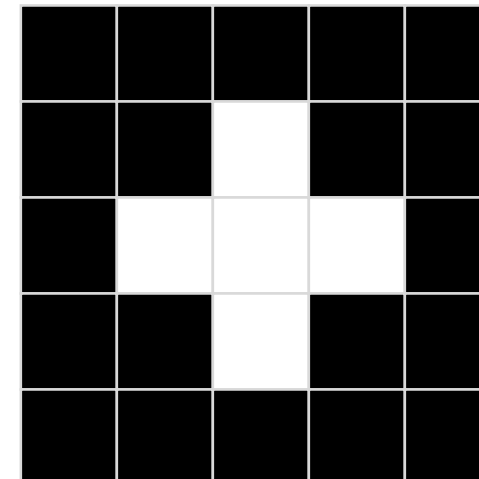
Dilation

8-connected neighborhood  
*Moore-Neighborhood*



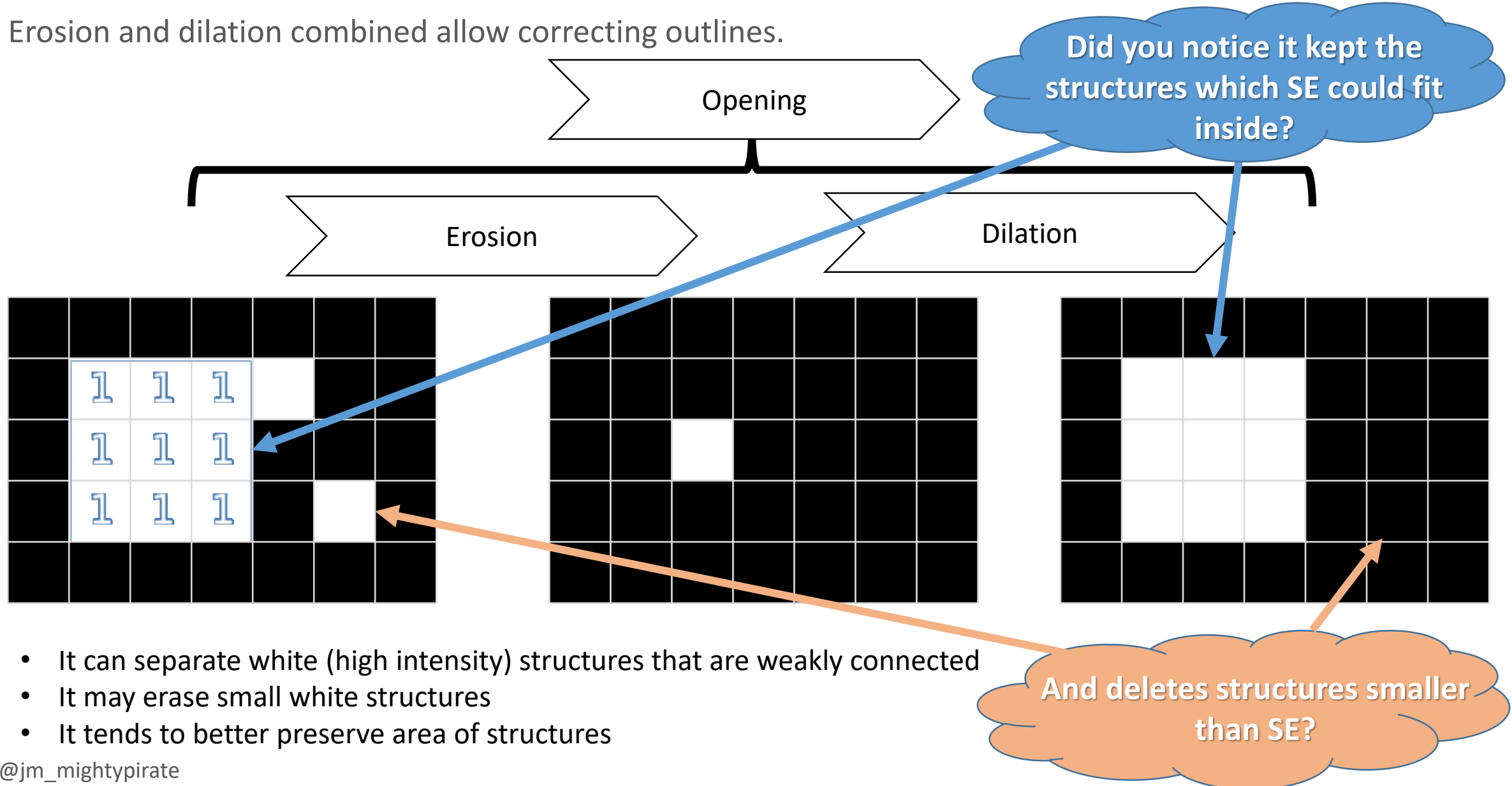
Dilation

4-connected neighborhood  
*von-Neumann-Neighborhood*

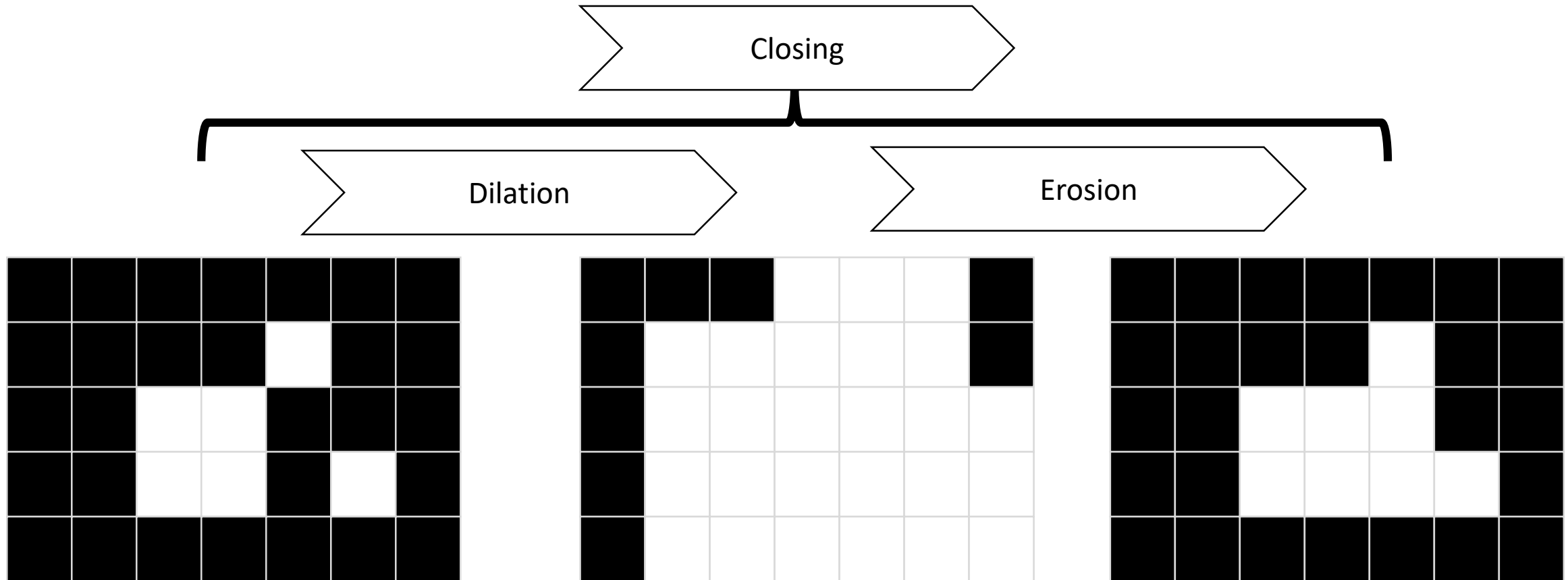


# Refining masks: Erosion & Dilation

- Erosion and dilation combined allow correcting outlines.

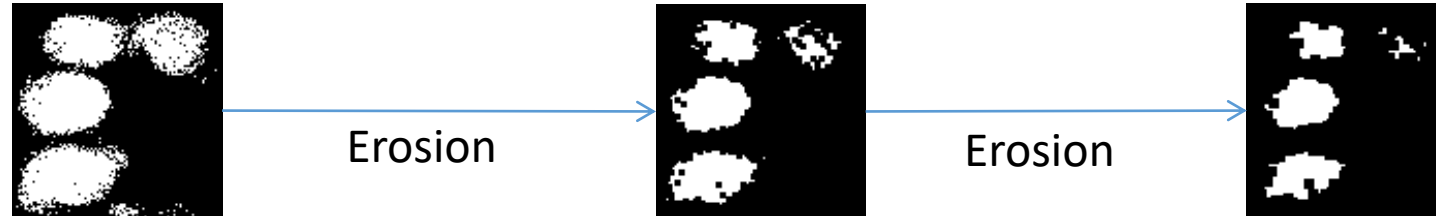


- It can separate white (high intensity) structures that are weakly connected
- It may erase small white structures
- It tends to better preserve area of structures



- It can connect white (high intensity) structures that are nearby
- It may close small holes inside structures
- It tends to better preserve area of structures

- Erosion: Set all pixels to black which have at least one black neighbor.



- Dilation: Set all pixels to white which have at least one white neighbor.



- Closing: Dilation + Erosion



- Opening: Erosion + Dilation

Are erosion/dilation/closing linear operations?  
(i.e., can they be written as a multiplication with a kernel?)

Yes

No

Scikit-image has a sub-package called morphology

```
from skimage import morphology
```

You must define a SE first (also called footprint):

```
SE = morphology.square(3)
SE
array([[1, 1, 1],
       [1, 1, 1],
       [1, 1, 1]], dtype=uint8)
```

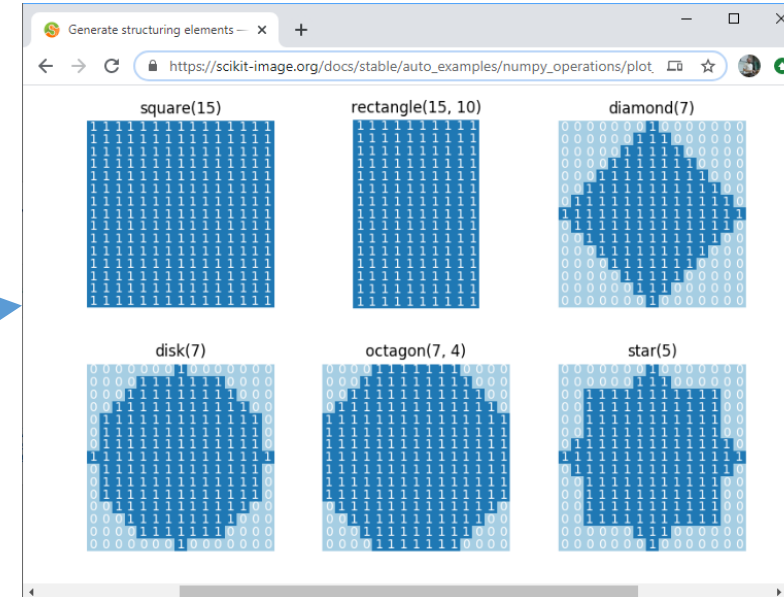
For a binary image, you apply it like this:

```
output = morphology.binary_dilation(binary_image, SE)
```

For a grayscale image, you apply it like this:

```
output = morphology.dilation(image, SE)
```

It can have other shapes/sizes:



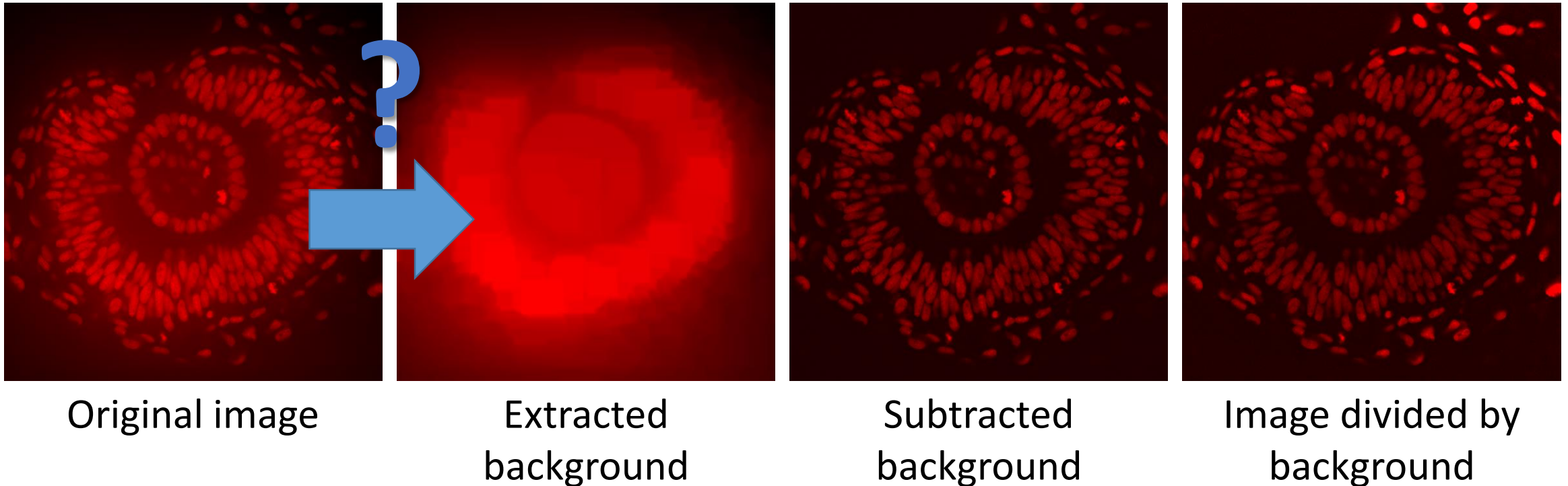
[https://scikit-image.org/docs/stable/auto\\_examples/numpy\\_operations/plot\\_structuring\\_elements.html#sphx-glr-auto-examples-numpy-operations-plot-structuring-elements-py](https://scikit-image.org/docs/stable/auto_examples/numpy_operations/plot_structuring_elements.html#sphx-glr-auto-examples-numpy-operations-plot-structuring-elements-py)

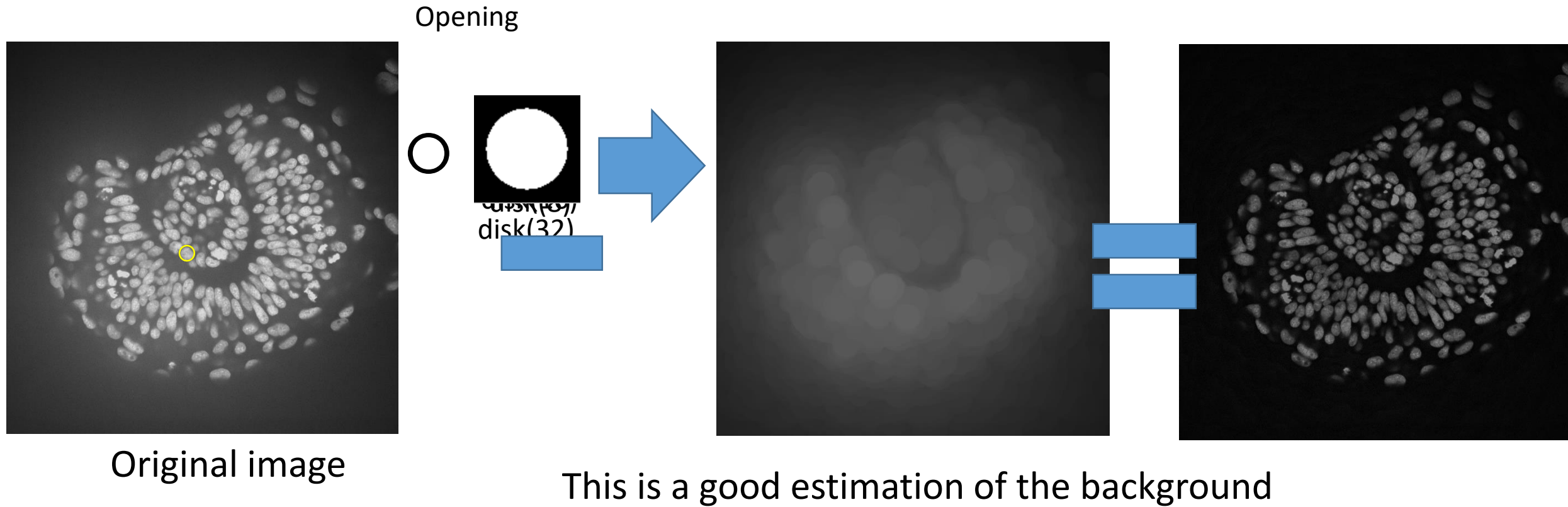


- Depending on the effect we want to correct for, it might make sense to divide an image by its background.

Do you remember that opening kept the structures which the SE could fit inside?

And do you remember that opening deletes structures smaller than SE?





Structures have a radius  $\approx 12$

# Background removal

What happens for a small structuring element (e.g., disk (2))?

