PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Image Segmentation

Robert Haase

With material from

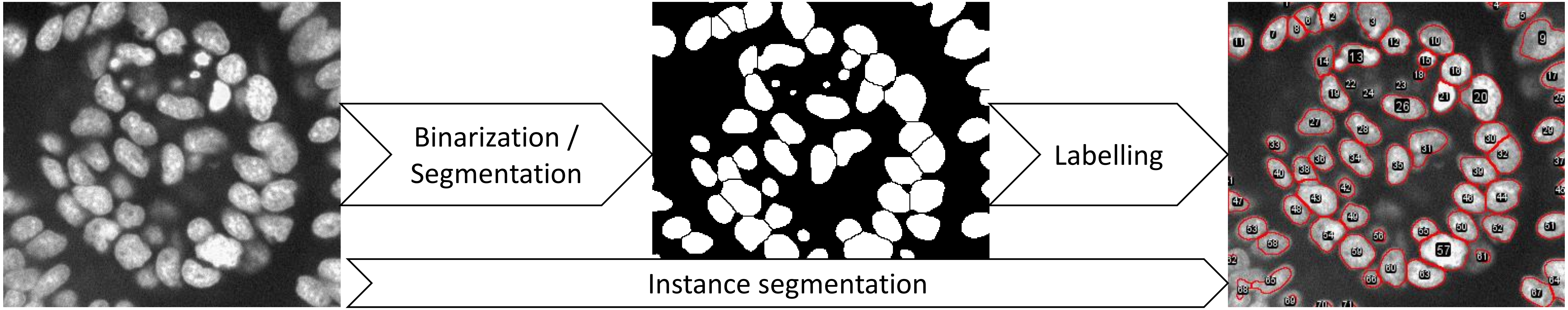
Benoit Lombardot, Scientific Computing Facility, MPI CBG

May 2021

@haesleinhuepf

Segmentation and labelling

- In order to allow the computer analyzing individual objects in images, we need to segment and label them first.
- Methods
  - Thresholding + binary watershed
  - Spot detection + seeded watershed
  - Edge detection based
  - Machine learning



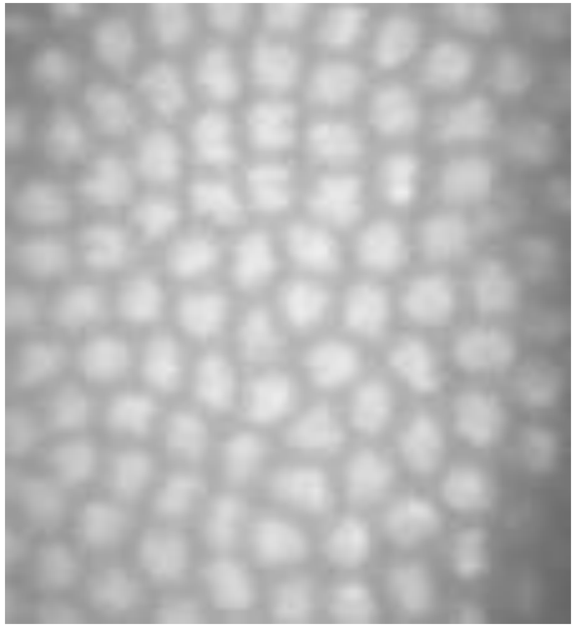
Binarization / Segmentation

Labelling

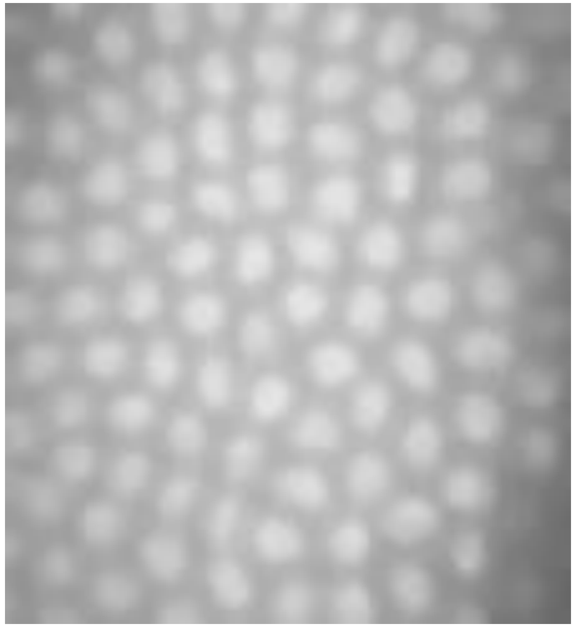
Instance segmentation

Pre-processing

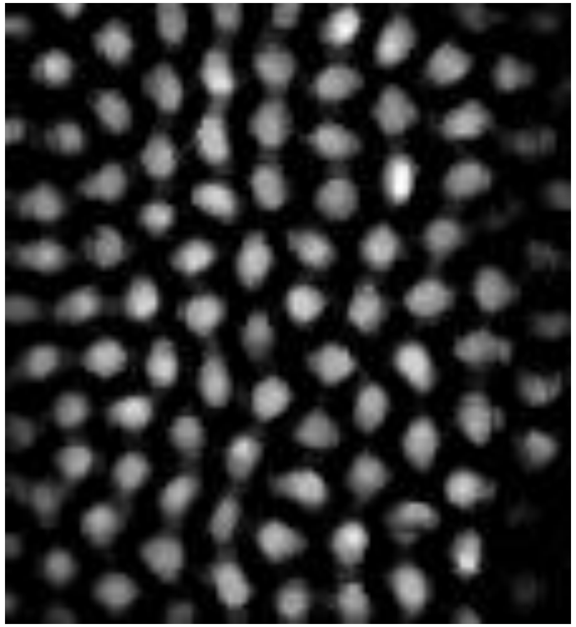
- Before we can create masks, we need to pre-process images.
  - Noise removal
  - Background subtraction
  - Contrast equivalisation



Original image



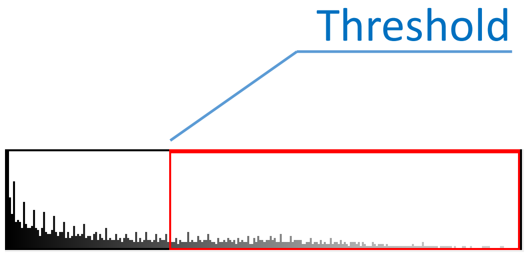
Median filtered



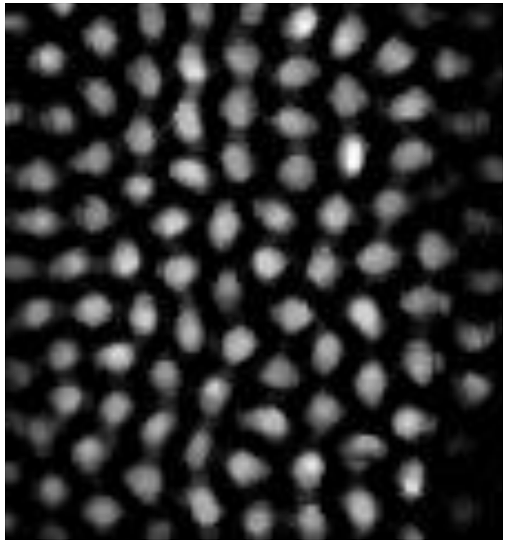
Background subtracted

Thresholding

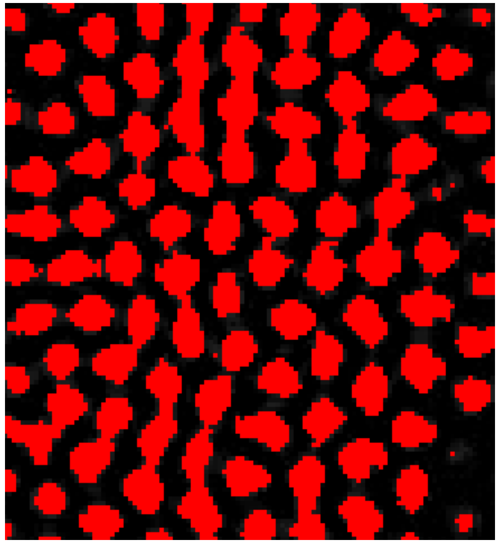
- Thresholding is the process of separating background from regions of interest by setting a minimum and/or maximum signal intensity as *threshold*.



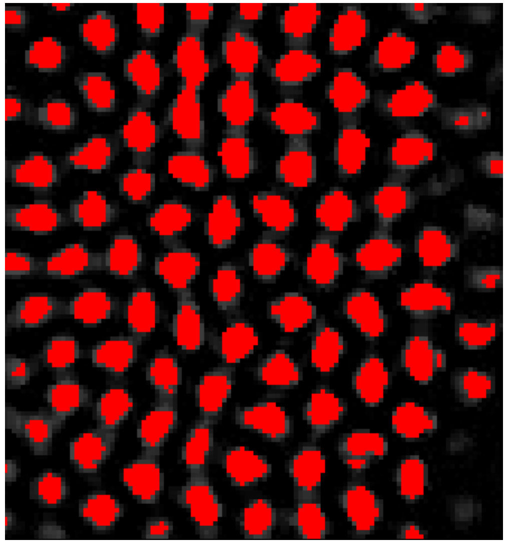
- When thresholding in a scientific context
  - Apply the same algorithm to all images in your project -> objectivity
  - Check carefully with many images when choosing the algorithm -> reliability
  - Document and cite which algorithm you used -> reproducibility



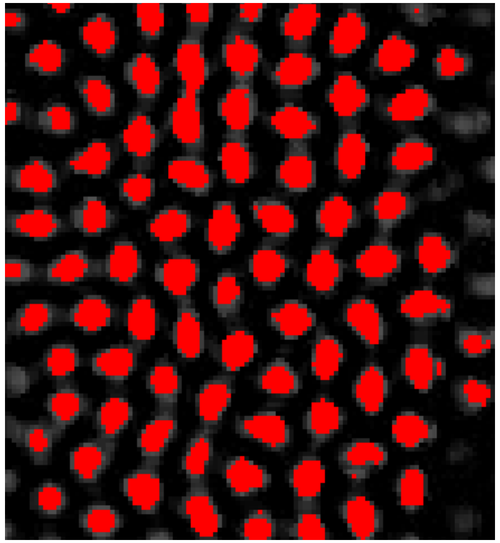
Original image



IsoData-threshold



Huang-threshold



Otsu-threshold

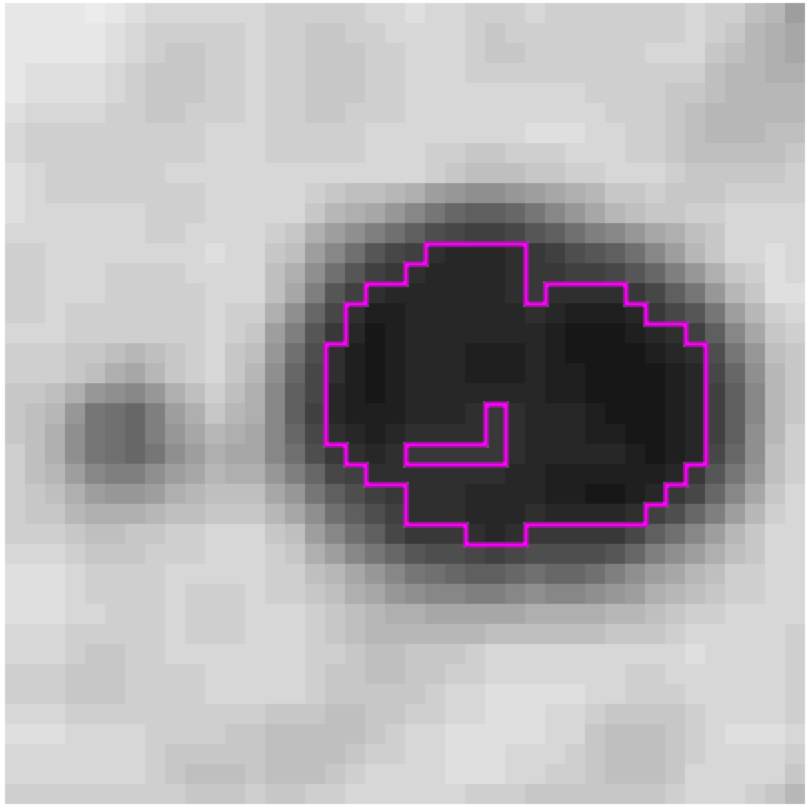
Otsu’s method

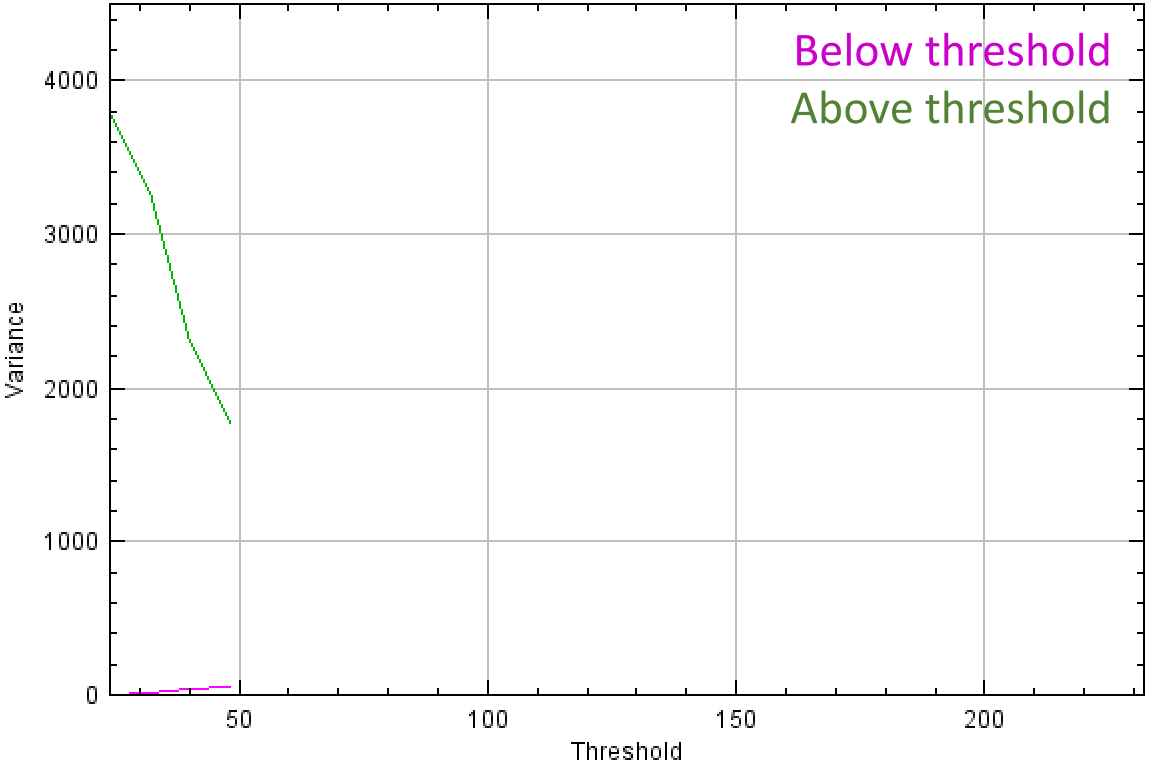
- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.

$$Var(I) = \sum_{i \in I} g_i - \bar{g}_I$$

$$\bar{g}_I = \sum_{i \in I} \frac{g_i}{n_I}$$

$Var(I)$ … Variance in image I
g_i … grey value of a pixel i
$\bar{g}_I$ … mean grey value of the whole image I
n_I … number of pixels in Image I



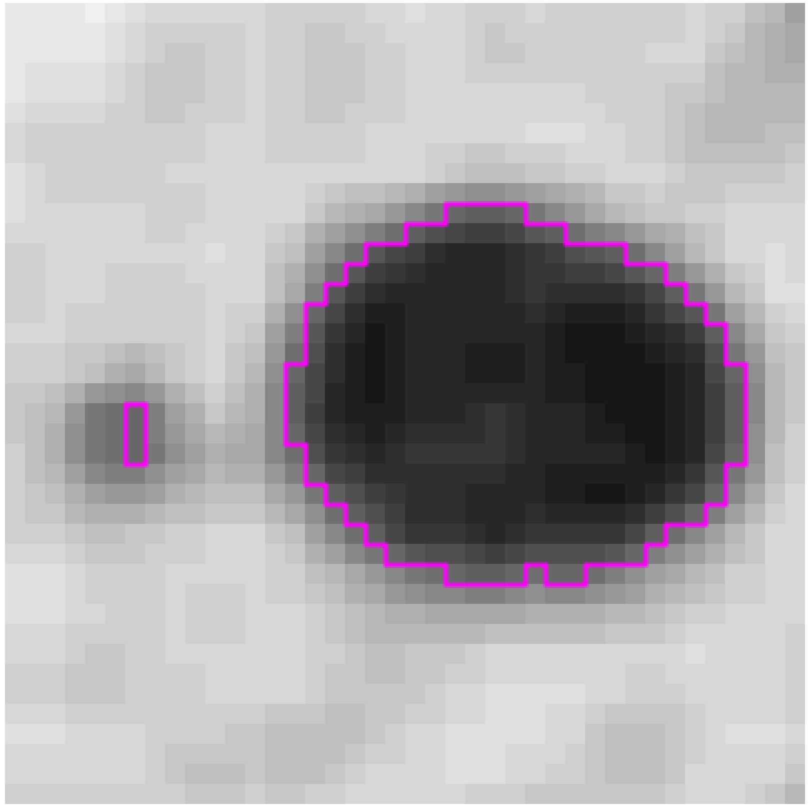


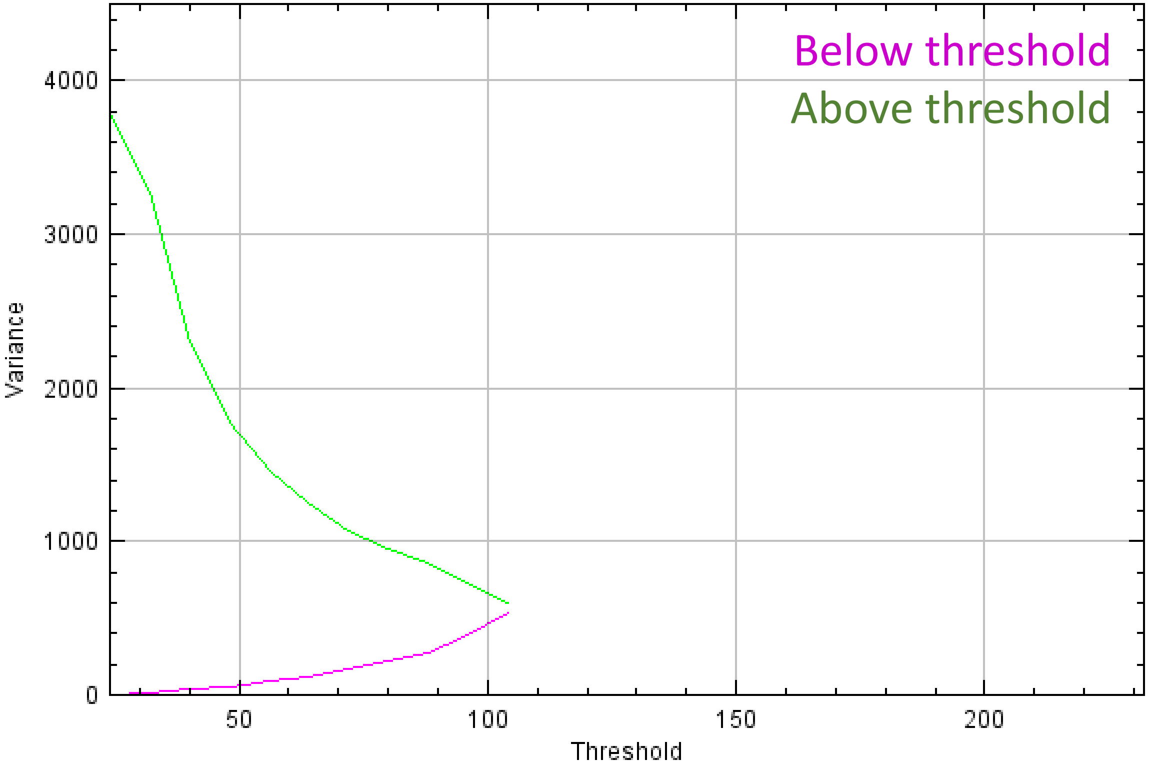
Otsu’s method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.

$$Var(I) = \sum_{i \in I} g_i - \bar{g}_I \qquad \bar{g}_I = \sum_{i \in I} \frac{g_i}{n_I}$$

$Var(I)$ … Variance in image I
g_i … grey value of a pixel i
$\bar{g}_I$ … mean grey value of the whole image I
n_I … number of pixels in Image I





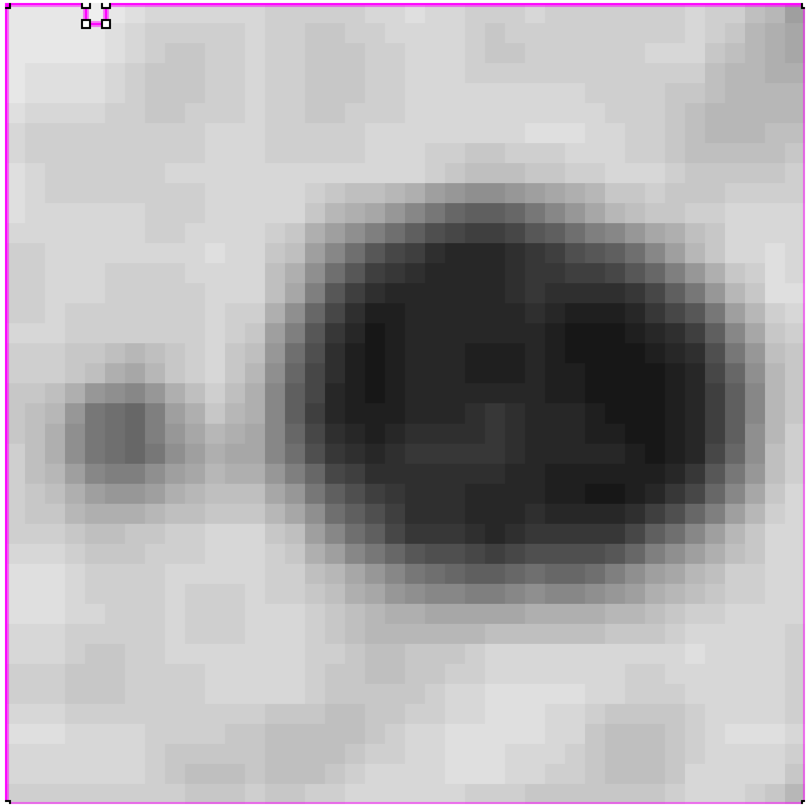
Otsu’s method

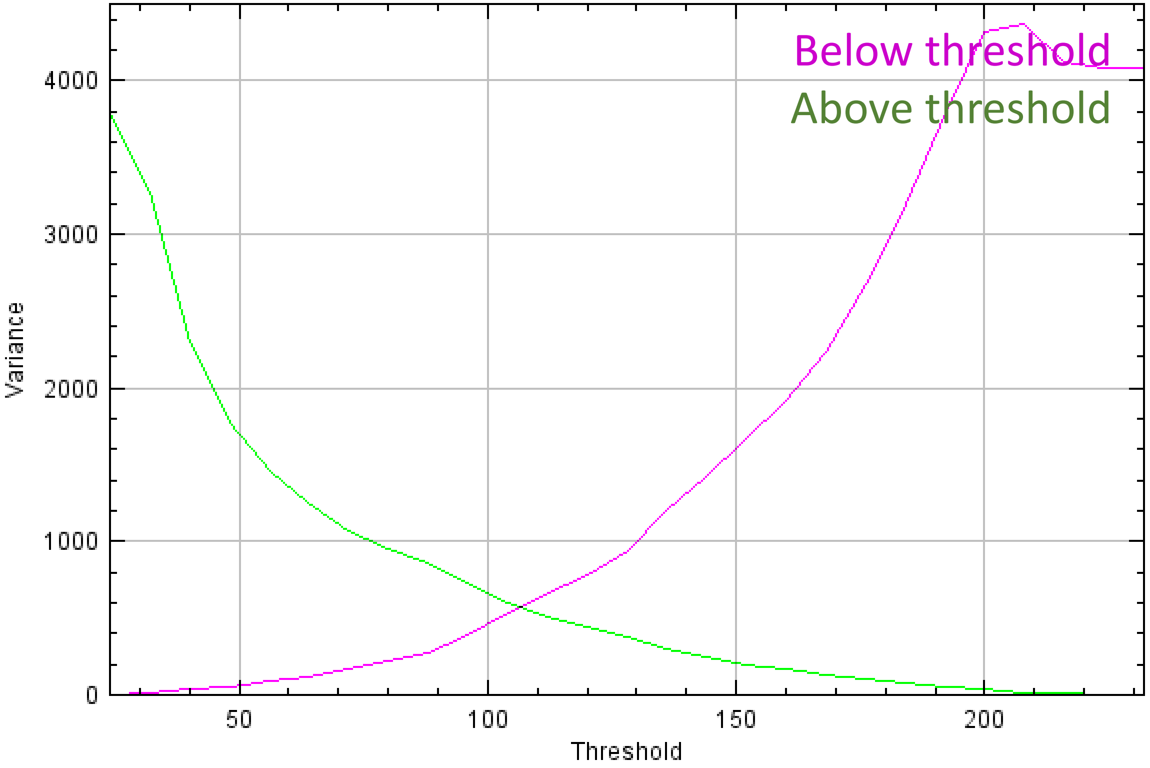
- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.

$$Var(I) = \sum_{i \in I} g_i - \bar{g}_I$$

$$\bar{g}_I = \sum_{i \in I} \frac{g_i}{n_I}$$

$Var(I)$ … Variance in image I
g_i … grey value of a pixel i
$\bar{g}_I$ … mean grey value of the whole image I
n_I … number of pixels in Image I

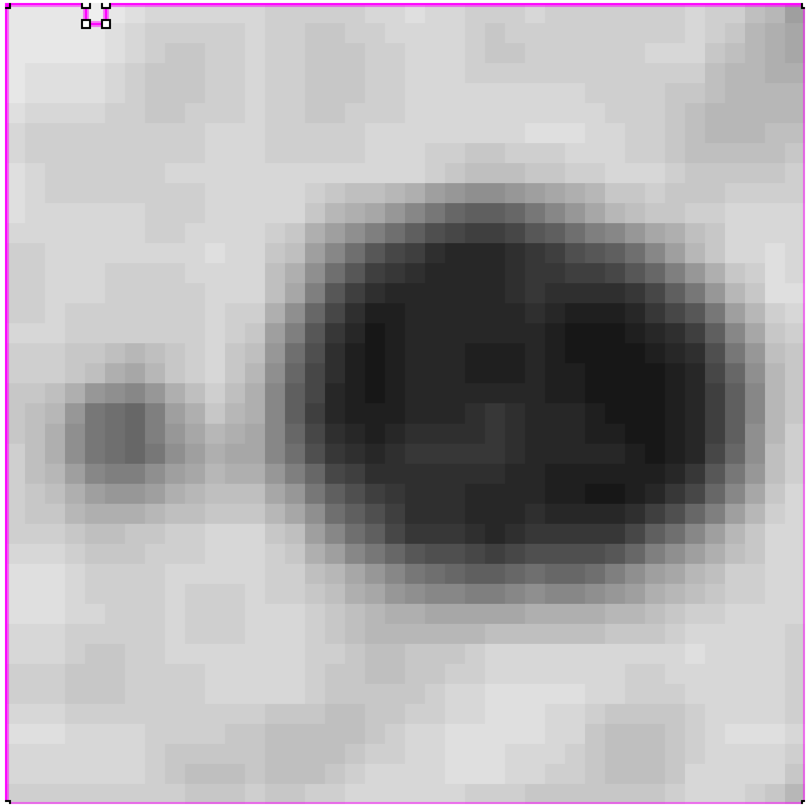


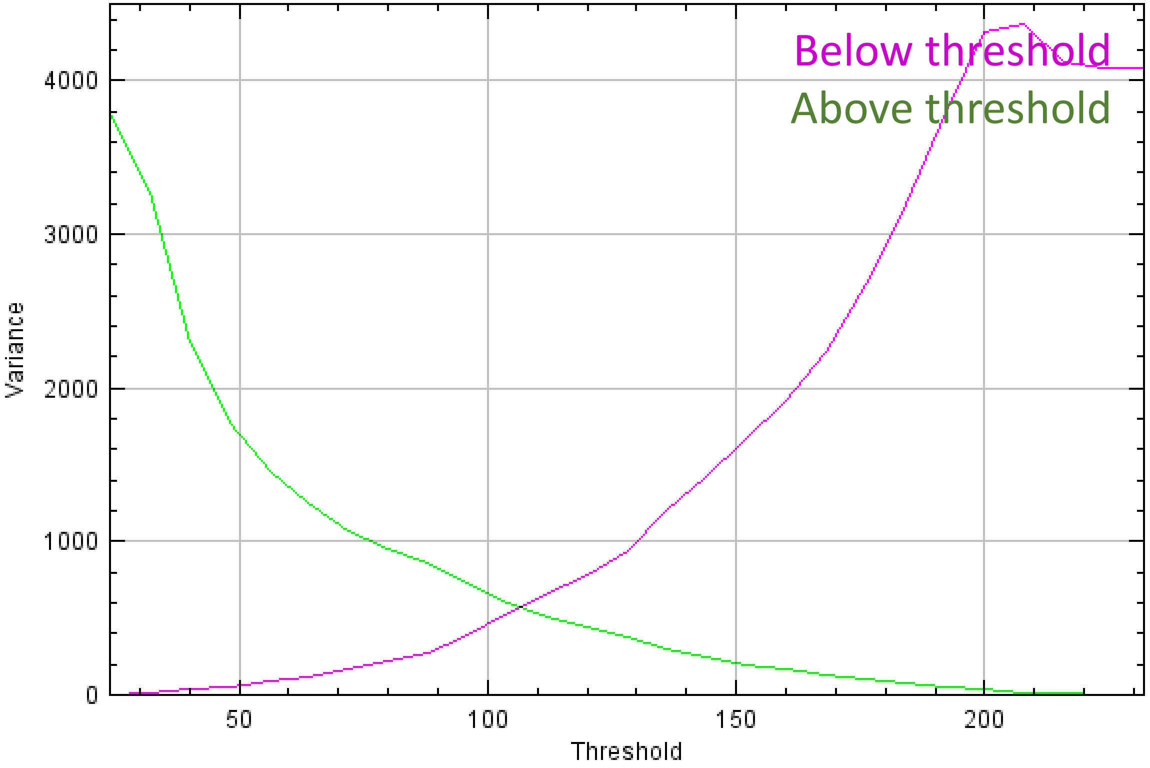


Otsu's method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.
- Weighted (!) sum variance

$$Var'(I) = \frac{n_A}{n_I} Var(A) + \frac{n_B}{n_I} Var(B) \qquad I = A \cup B$$

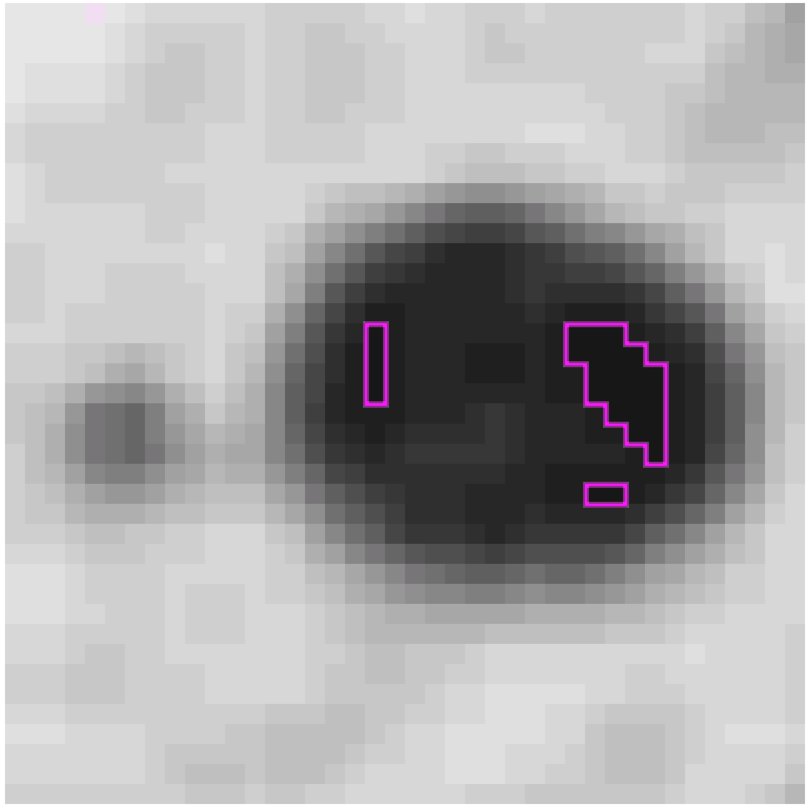


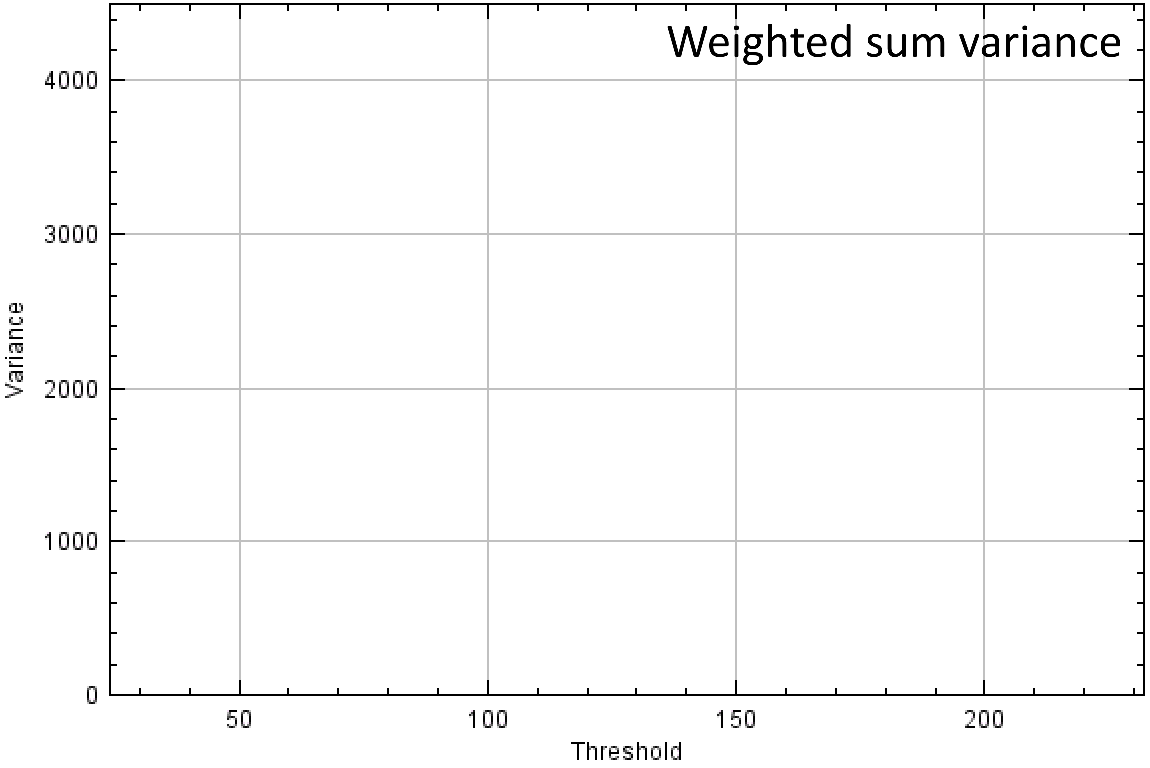


Otsu’s method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.
- Weighted (!) sum variance

$$Var'(I) = \frac{n_A}{n_I} Var(A) + \frac{n_B}{n_I} Var(B) \qquad I = A \cup B$$

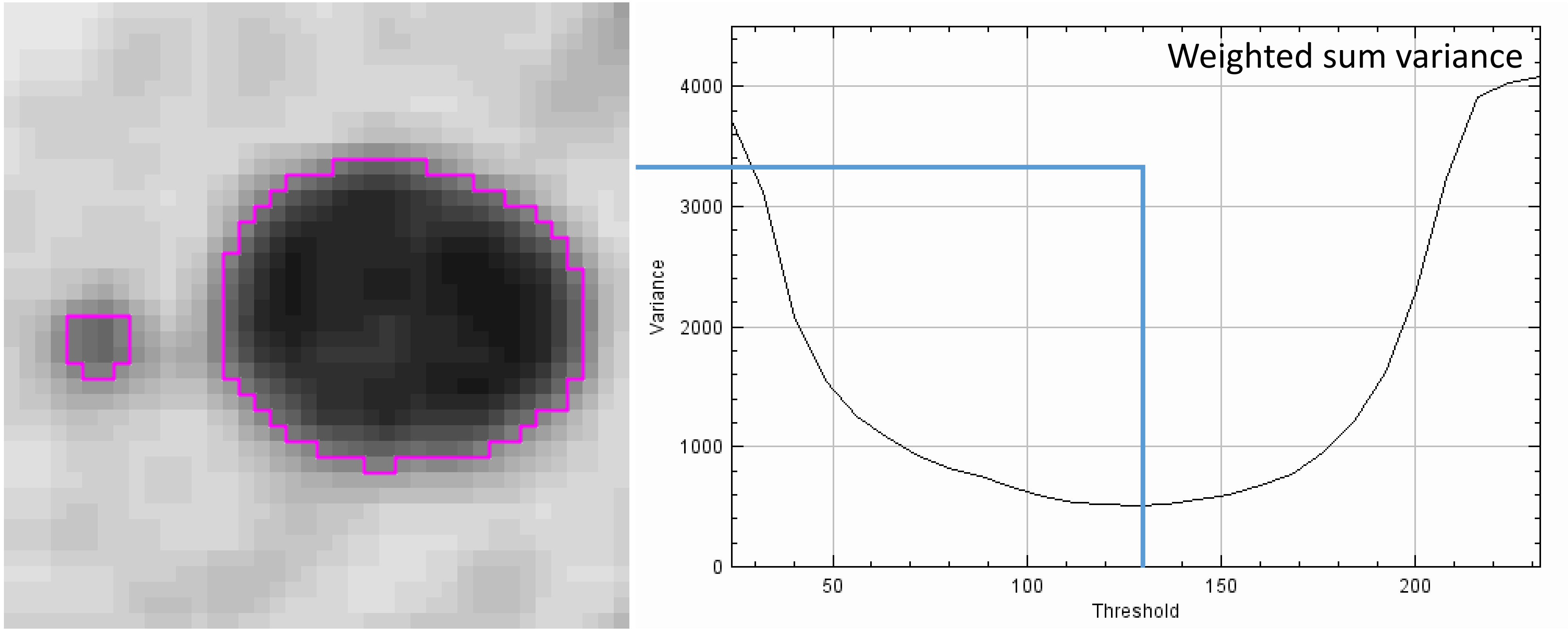




Otsu’s method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.
- Weighted (!) sum variance

$$Var'(I) = \frac{n_A}{n_I} Var(A) + \frac{n_B}{n_I} Var(B) \qquad I = A \cup B$$

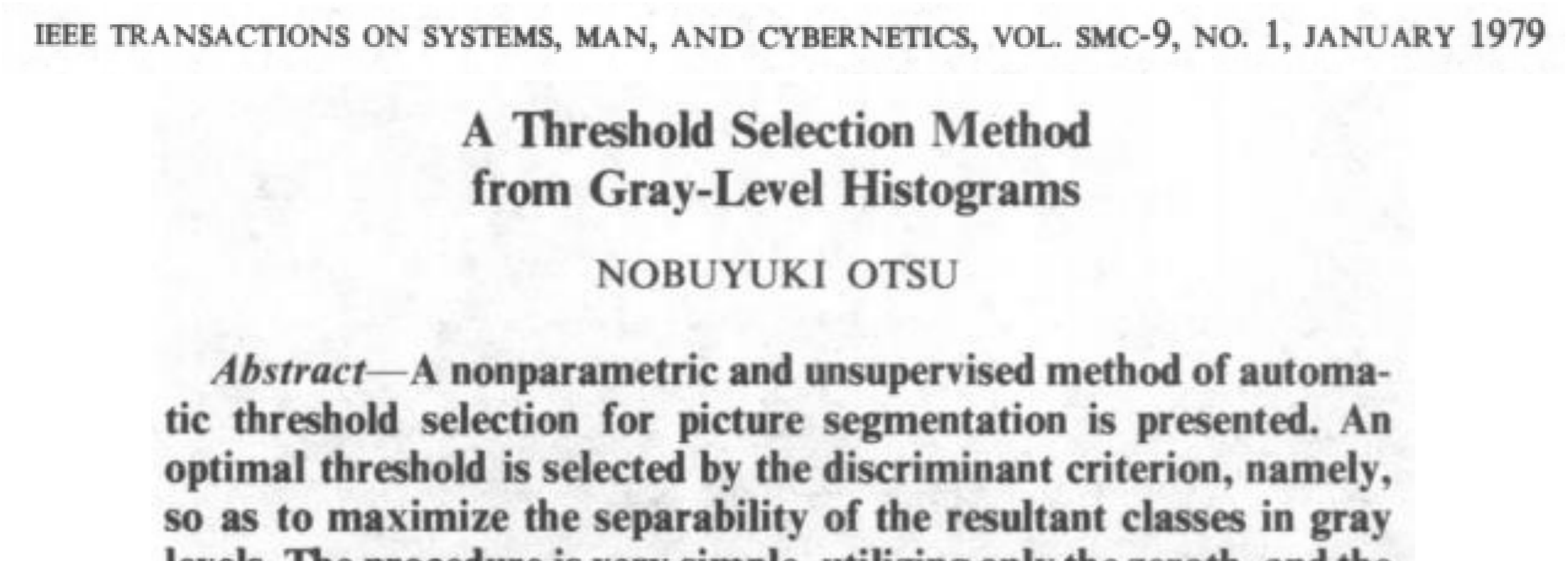


See also: http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html

Thresholding: Citing

- Cite the thresholding method of your choice properly

We segmented the cell nuclei in the images using the Otsu thresholding method (Otsu et Al. 1979) implemented in Fiji (Schindelin et Al. 2012).



IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, VOL. SMC-9, NO. 1, JANUARY 1979

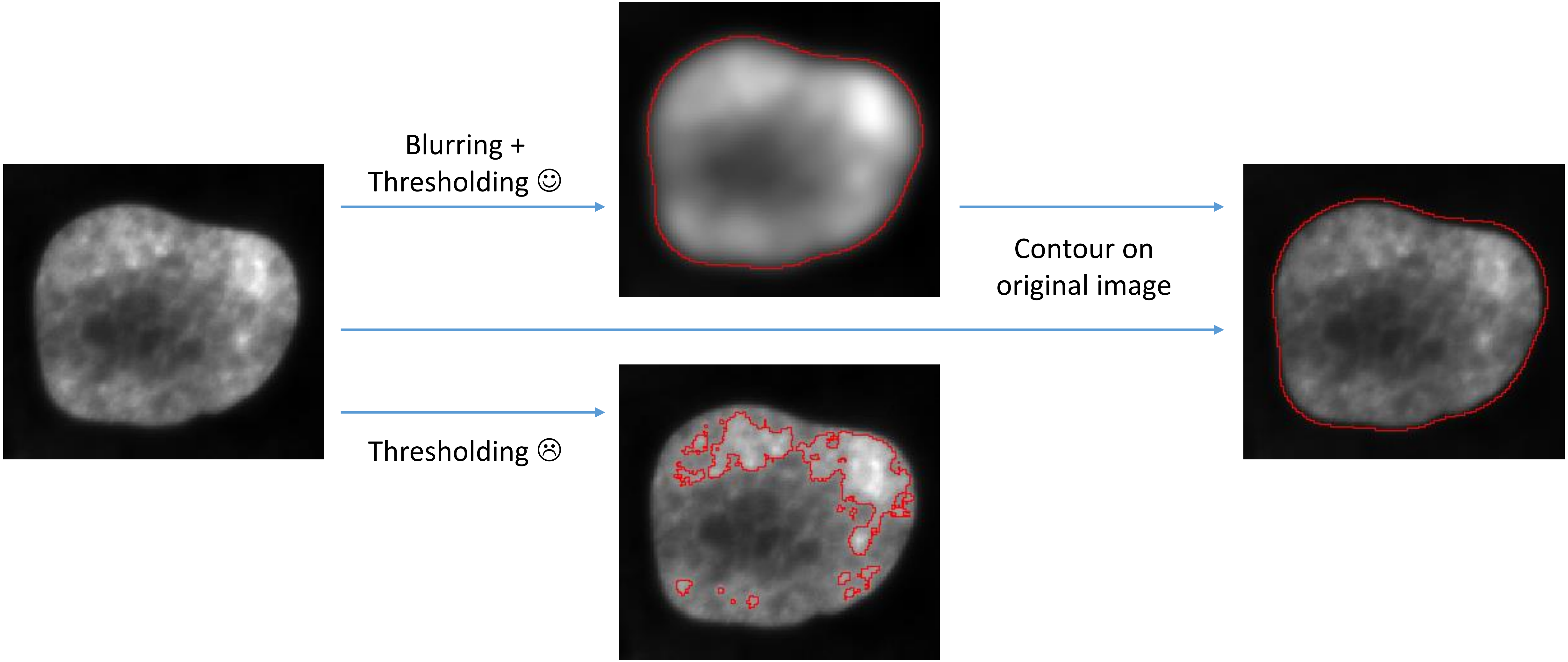
A Threshold Selection Method from Gray-Level Histograms

NOBUYUKI OTSU

Abstract—A nonparametric and unsupervised method of automatic threshold selection for picture segmentation is presented. An optimal threshold is selected by the discriminant criterion, namely, so as to maximize the separability of the resultant classes in gray levels. The procedure is very simple, utilizing only the zeroth- and the

Low-pass filtering to improve thresholding results

- In case thresholding algorithms outline the wrong structure, blurring in advance may help.
- However: **Do not** continue processing the blurred image, continue with the original!



Blurring + Thresholding ☺

Contour on original image

Thresholding ☹

May 2021

Thresholding: Pitfalls

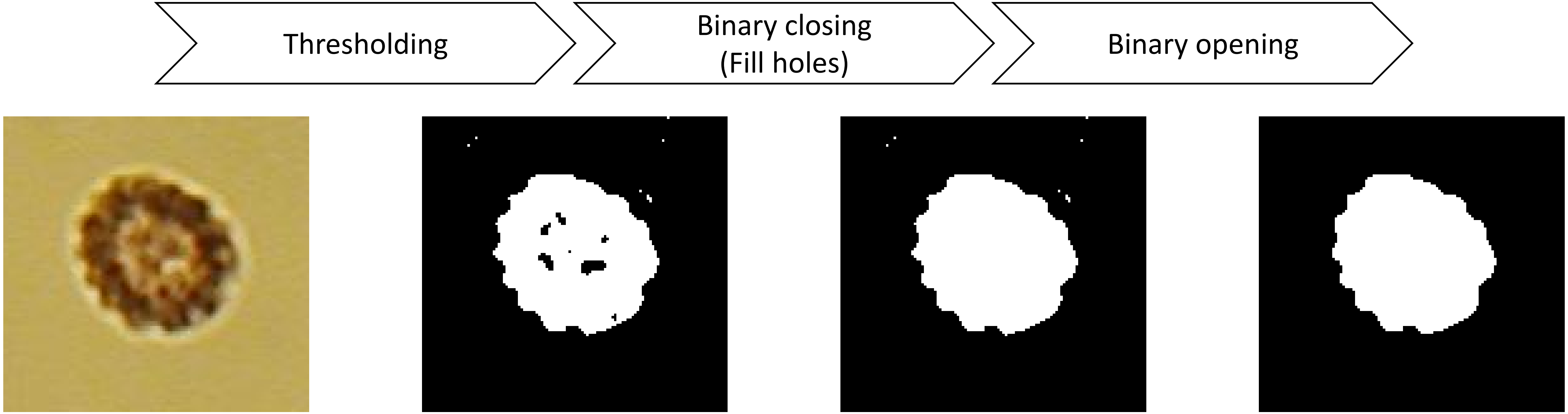
- **Do not enter a manual threshold** and afterwards measure signal intensity of segmented objects!
  - You would measure the threshold you entered.
- Hint: Never use manual thresholding!
  - It’s in general not reproducible and not objective.

- Chose a threshold algorithm and stick to it for the whole study. If you select a new method for every image, the procedure is not reproducible and not objective!

- There is no perfect method. There will be always this one image where thresholding fails.

Refining masks

- Binary mask images may not be perfect immediately after thresholding.
- There are ways of refining them



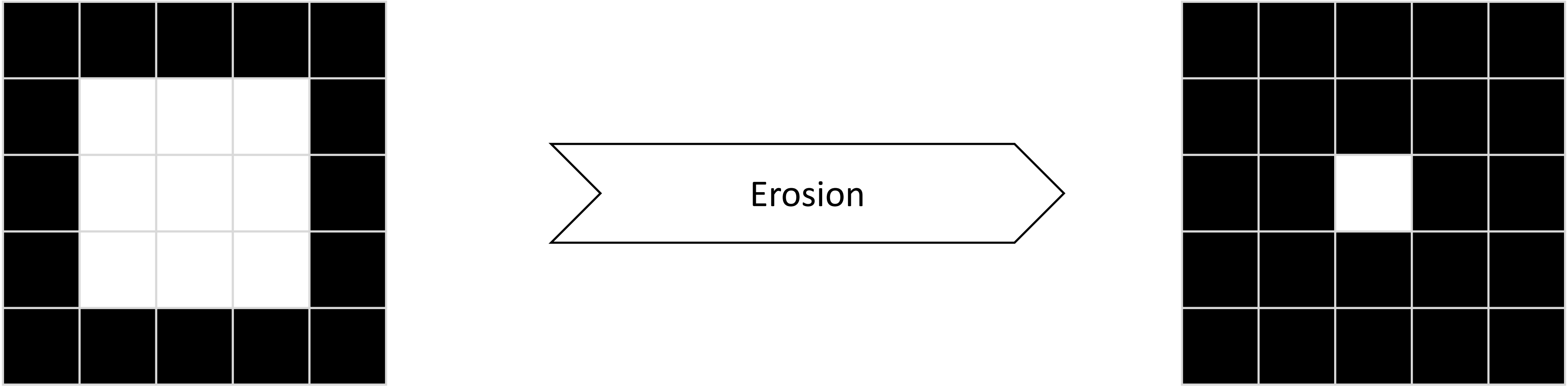
Thresholding

Binary closing
(Fill holes)

Binary opening

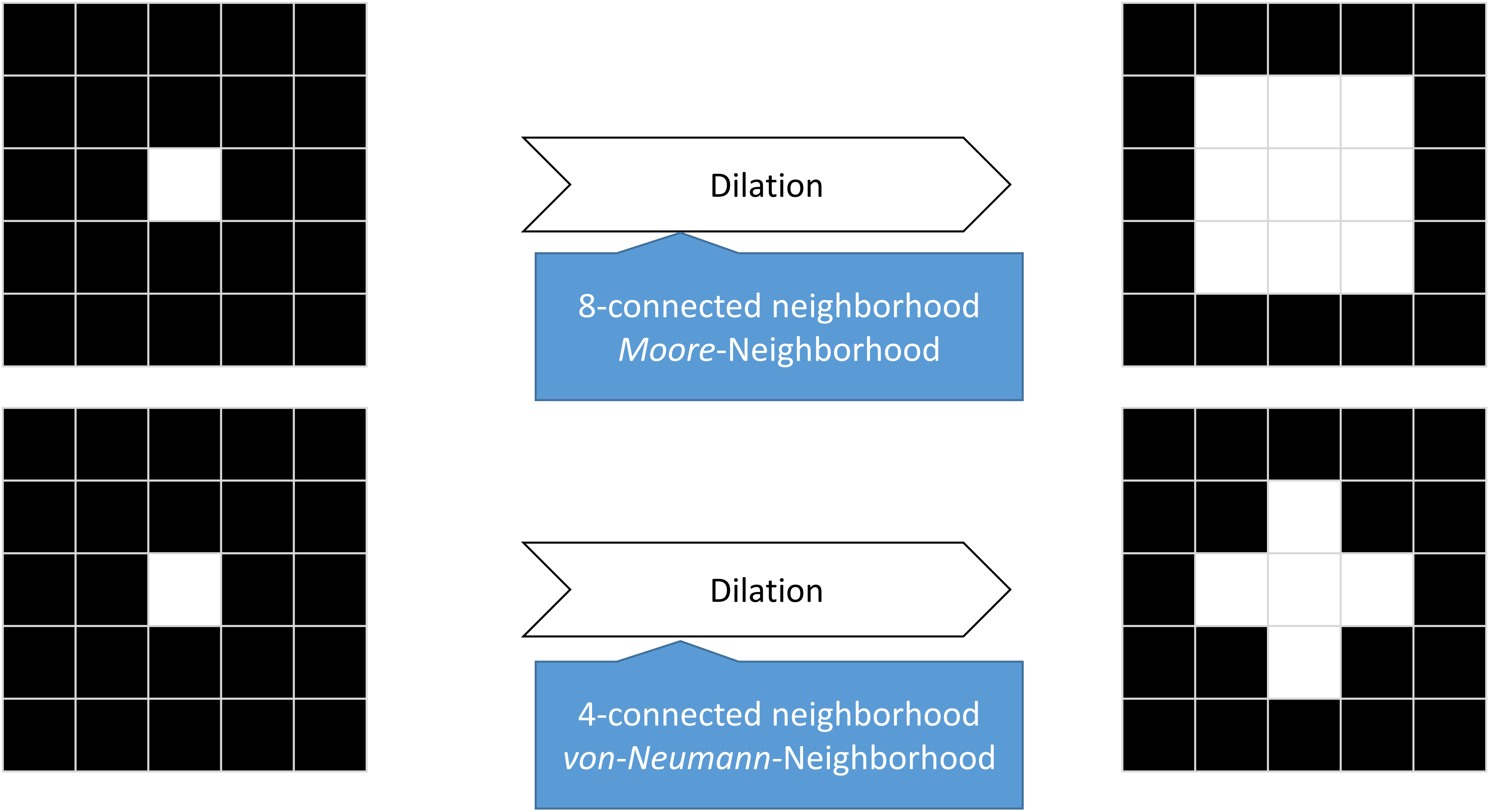
Refining masks: Erosion

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



Refining masks: Dilation

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



Dilation

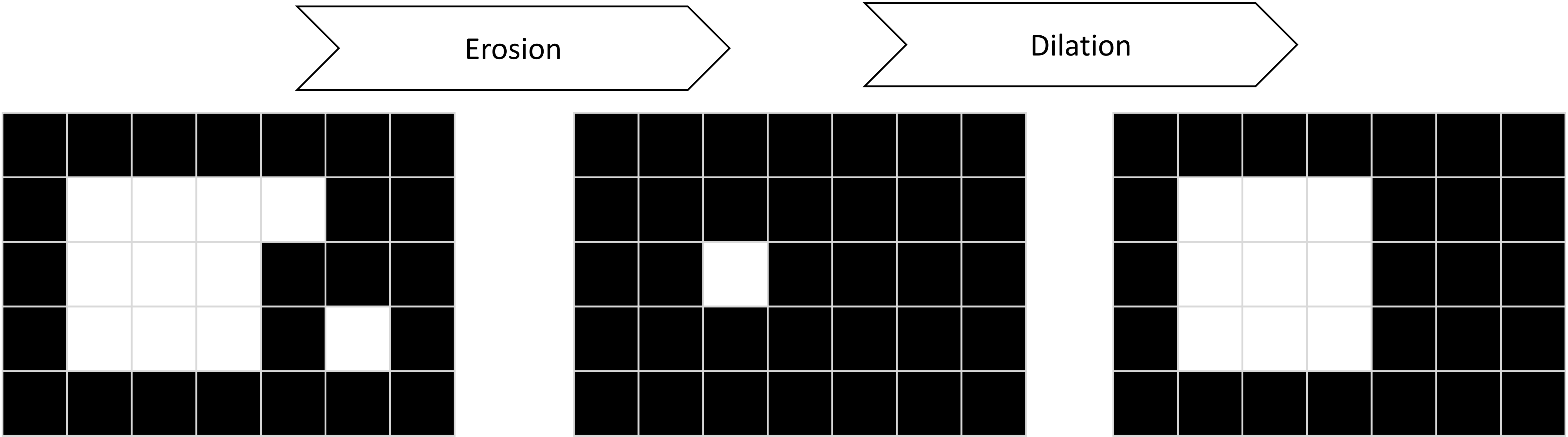
8-connected neighborhood
Moore-Neighborhood

Dilation

4-connected neighborhood
von-Neumann-Neighborhood

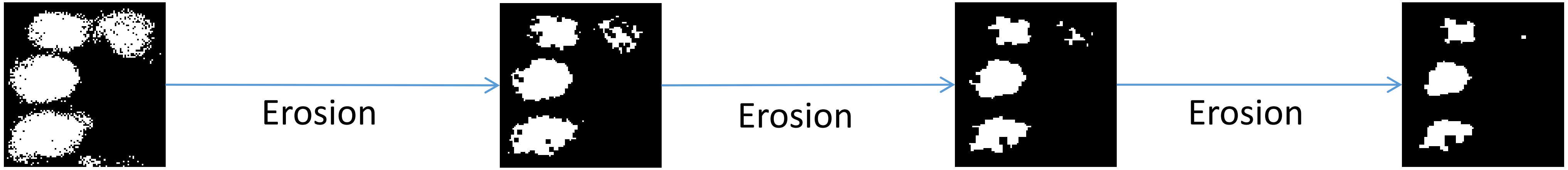
Refining masks: Erosion & Dilation

- Erosion and dilation combined allow correcting outlines.

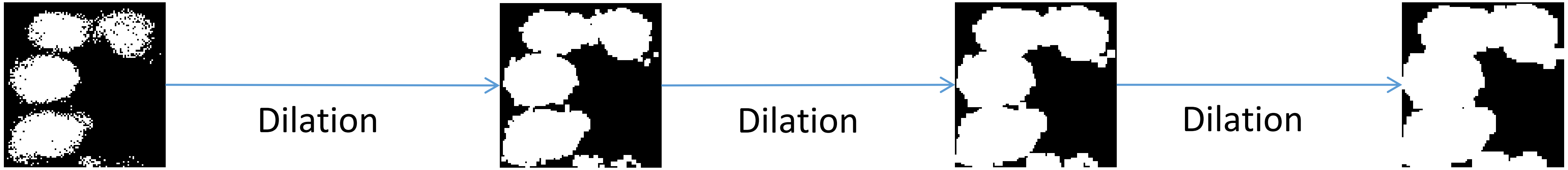


Refining masks: Erosion / dilation

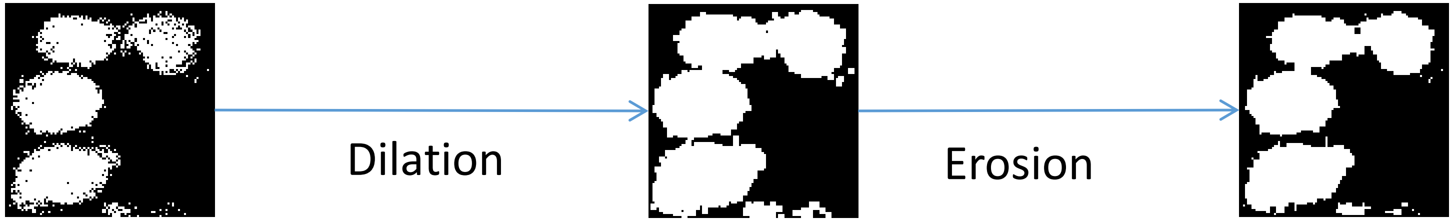
- Erosion: Set all pixels to black which have at least one black neighbor.



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



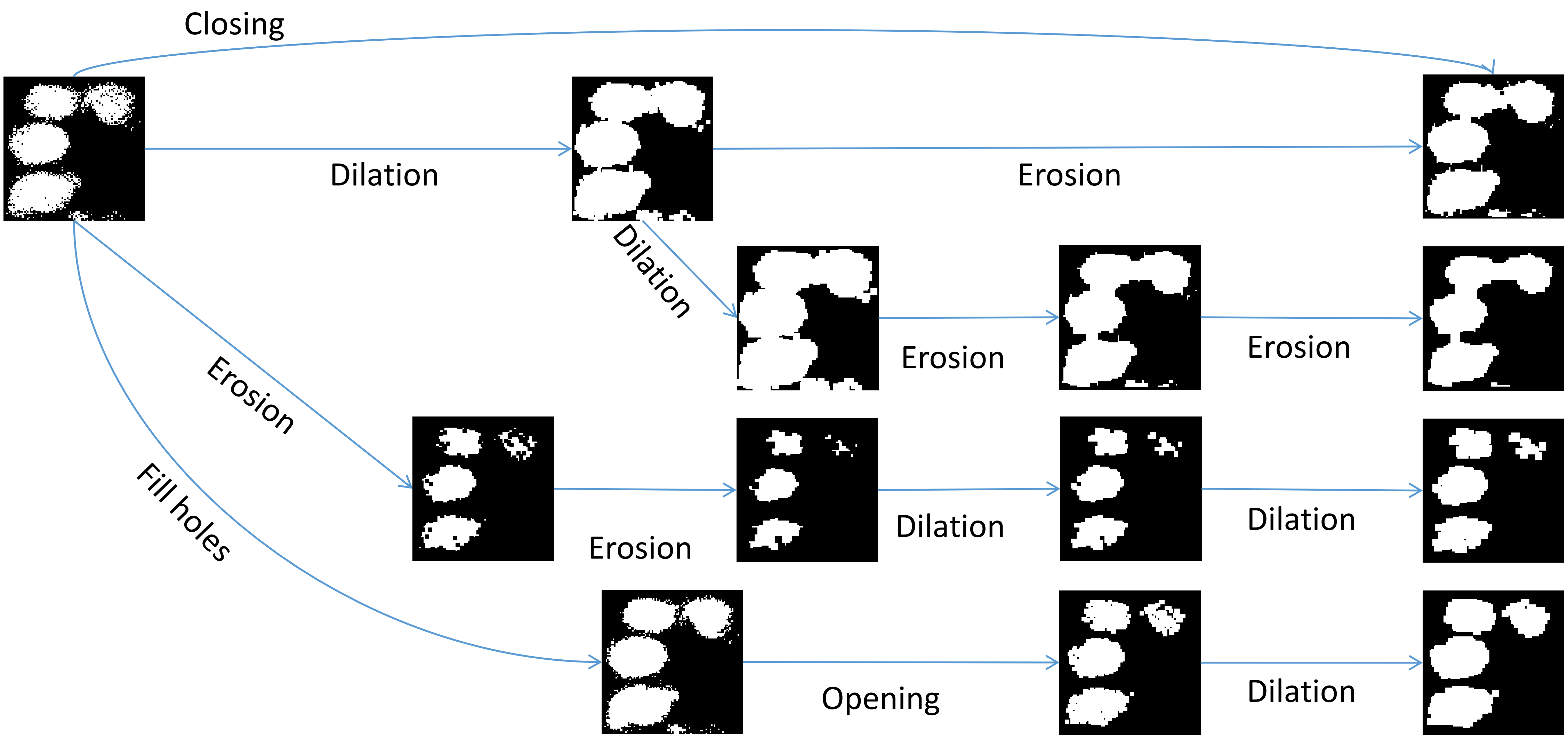
- Closing: Dilation + Erosion



- Opening: Erosion + Dilation

Slide adapted from Haase, Lombardot, Scientific Computing Facility, MPI CBG, licensed CC BY-NC 4.0

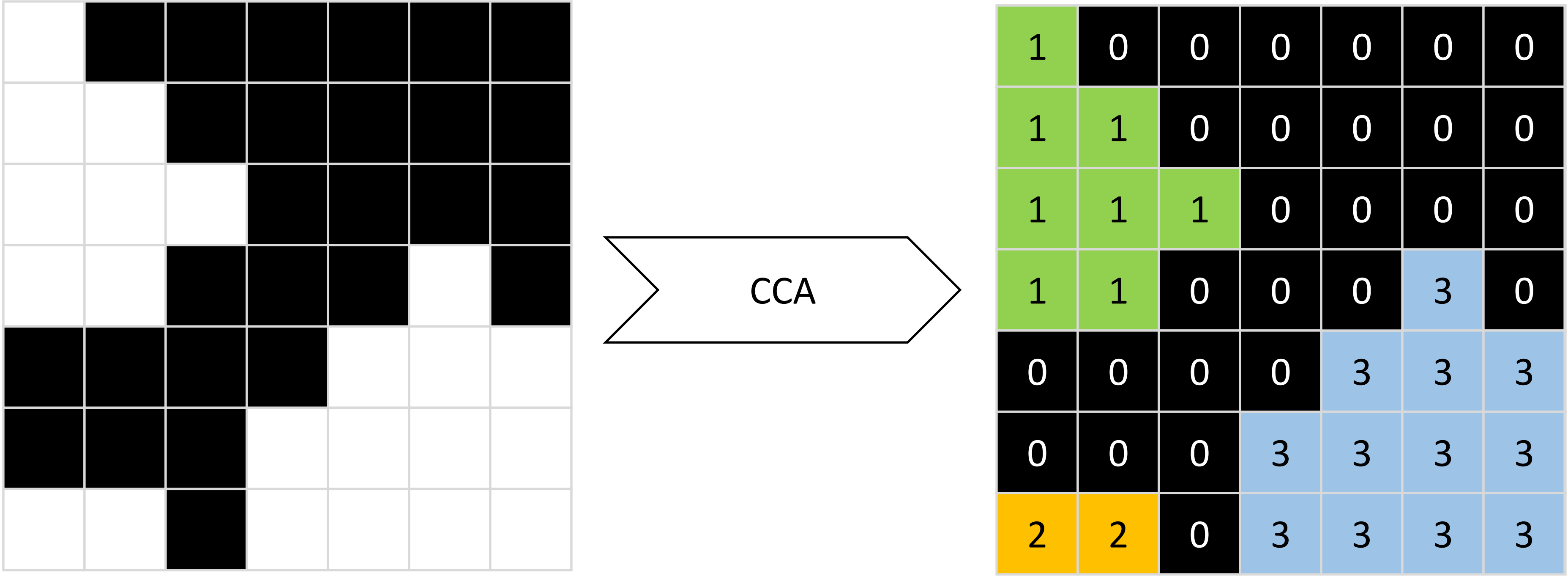
Refining masks: opening / closing



Slide adapted from Haase, Lombardot, Scientific Computing Facility, MPI CBG, licensed CC BY-NC 4.0

Connected components labelling

- In order to allow the computer differentiating objects, connected components analysis (CCA) is used to mark pixels belonging to different objects with different numbers
- Background pixels are marked with 0.
- The maximum intensity of a labelled map corresponds to the number of objects.



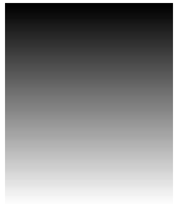
CCA

1	0	0	0	0	0	0
1	1	0	0	0	0	0
1	1	1	0	0	0	0
1	1	0	0	0	3	0
0	0	0	0	3	3	3
0	0	0	3	3	3	3
2	2	0	3	3	3	3

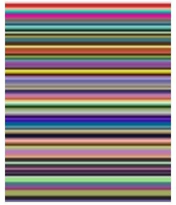
Label map visualisation

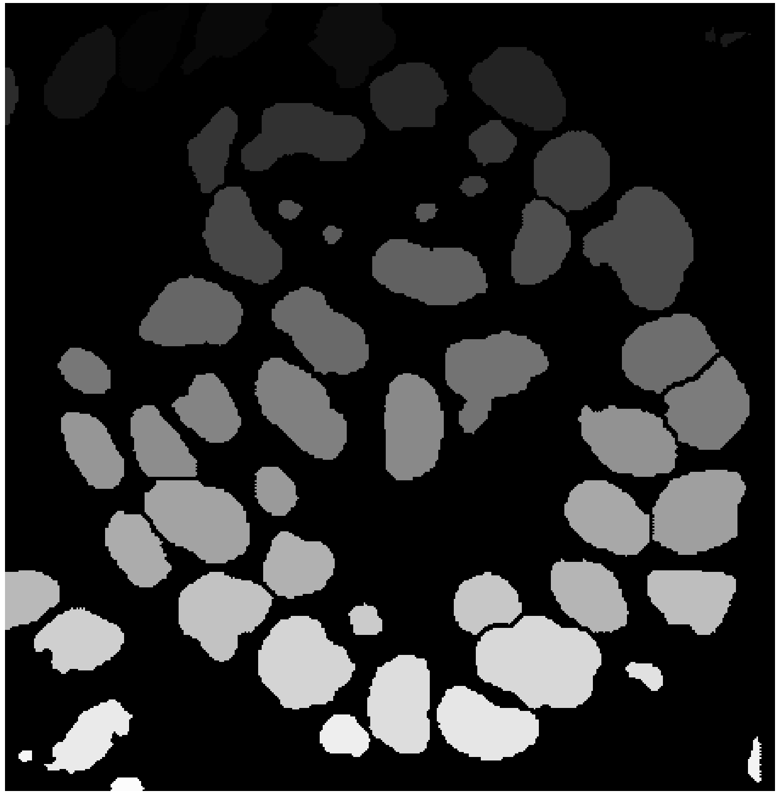
- Label maps can be nicely visualized with the right lookup table

Grey

Pixel value	Display color
0 1 2 ... 255	

Glasbey

Pixel value	Display color
0 1 2 ... 255	



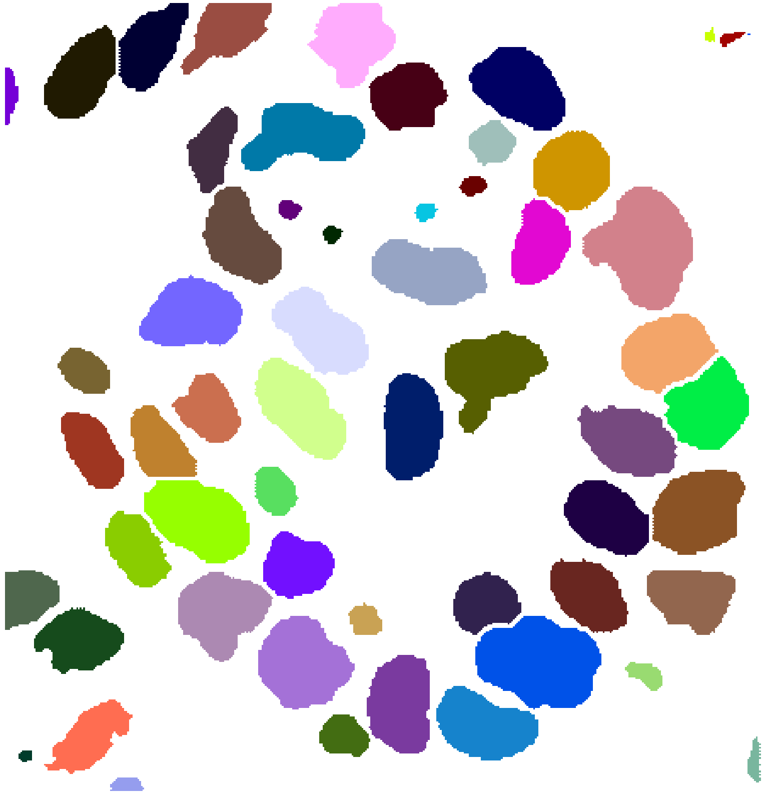


Image segmentation in Fiji

Robert Haase

With material from

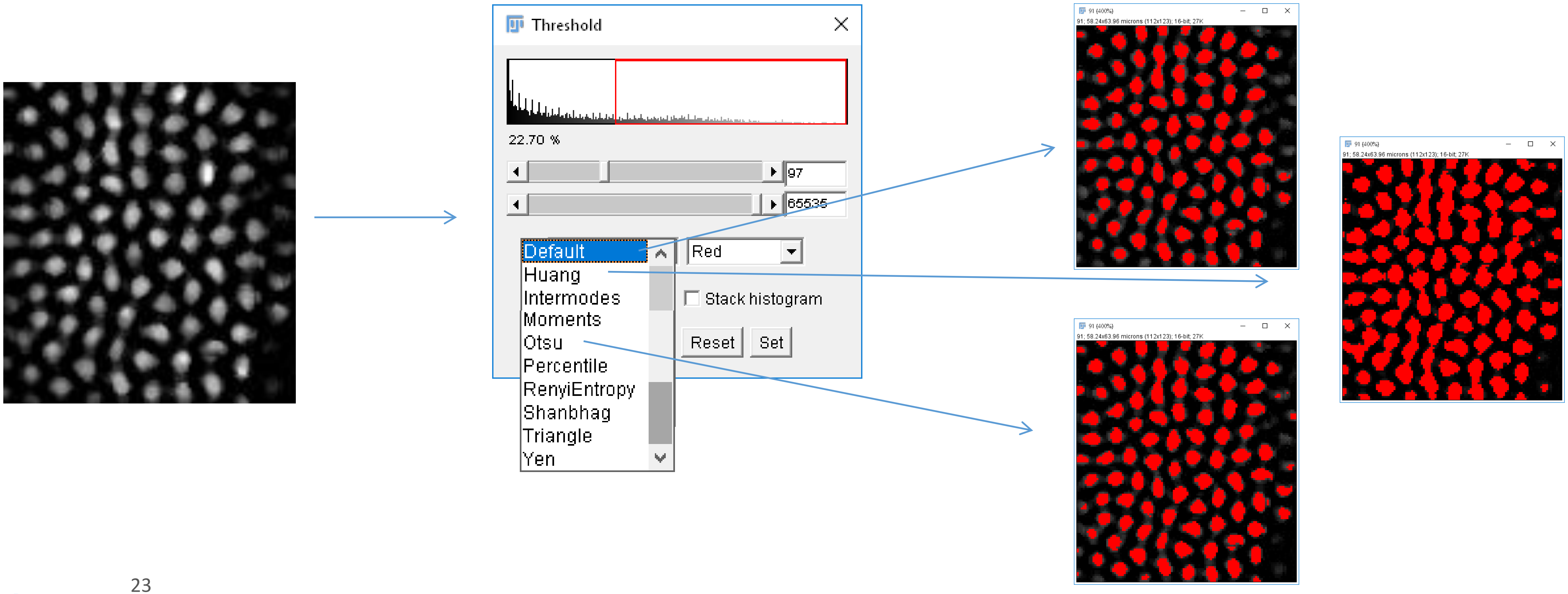
Mauricio Rocha Martins, Norden lab, MPI CBG

May 2021

@haesleinhuepf

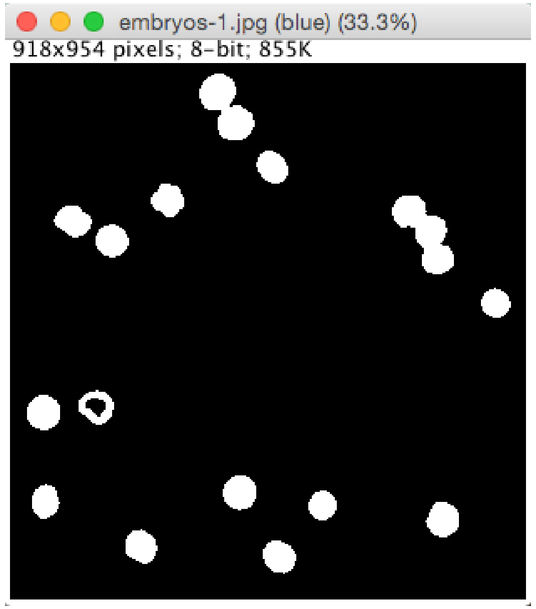
Thresholding

- Image > Adjust > Threshold…
- Remember: Don’t do manual thresholding whenever possible. Choose a reliable method.



Refining masks

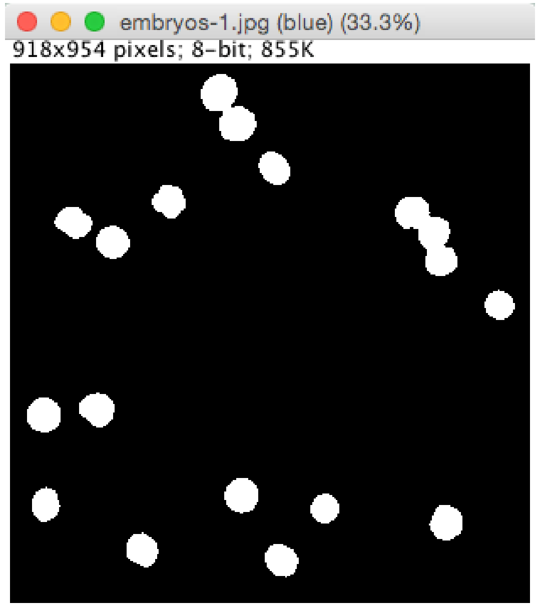
- For further processing, we need a binary image with clearly separated, circular and filled objects.



embryos-1.jpg (blue) (33.3%)

918x954 pixels; 8-bit; 855K

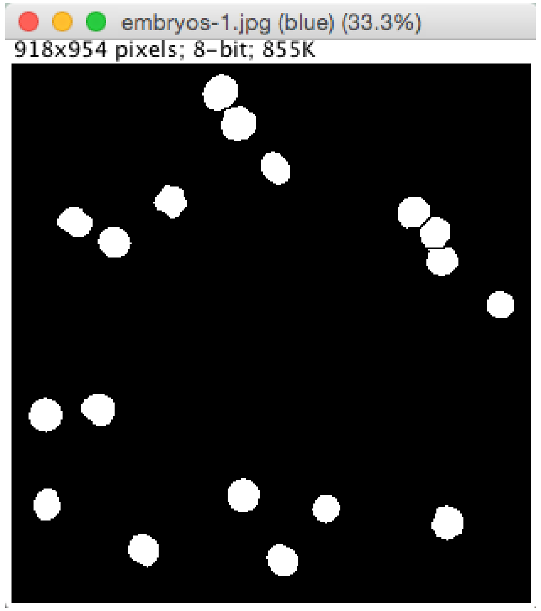
Mask



embryos-1.jpg (blue) (33.3%)

918x954 pixels; 8-bit; 855K

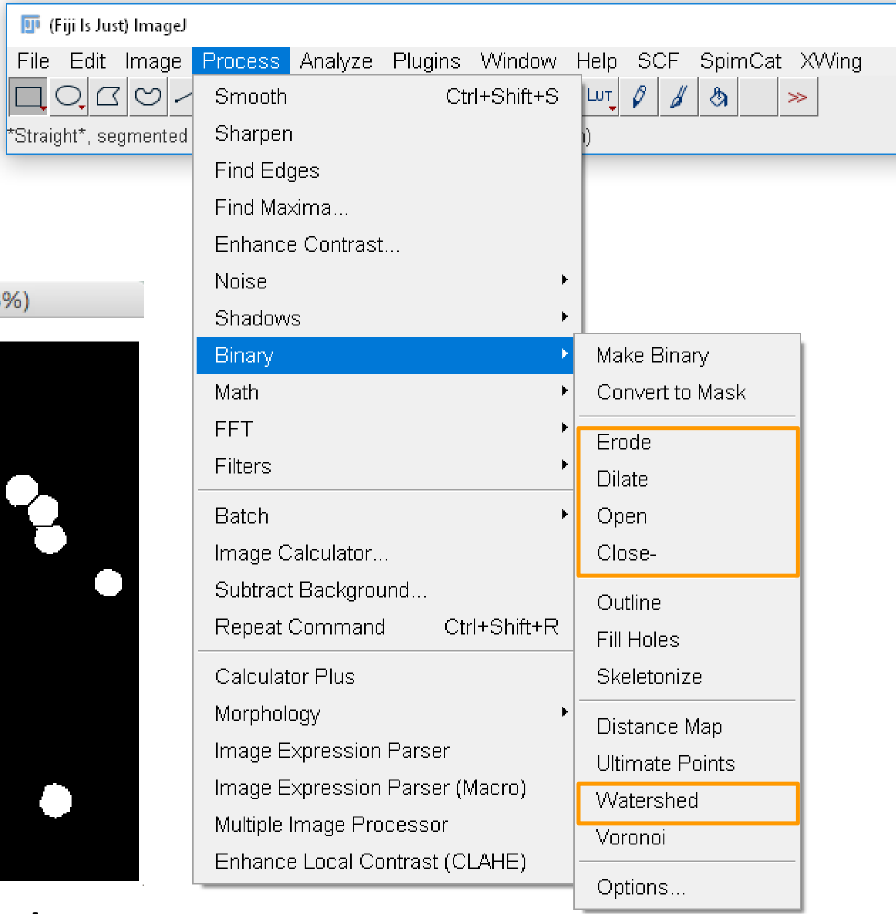
+ filled holes



embryos-1.jpg (blue) (33.3%)

918x954 pixels; 8-bit; 855K

+ Watershed

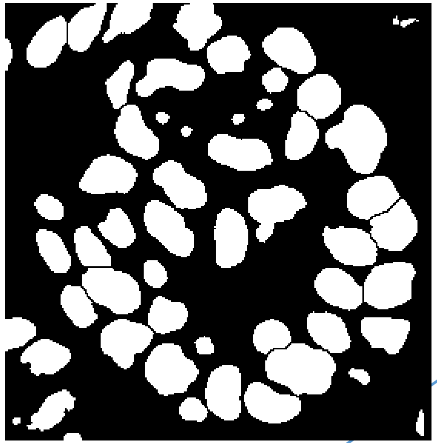


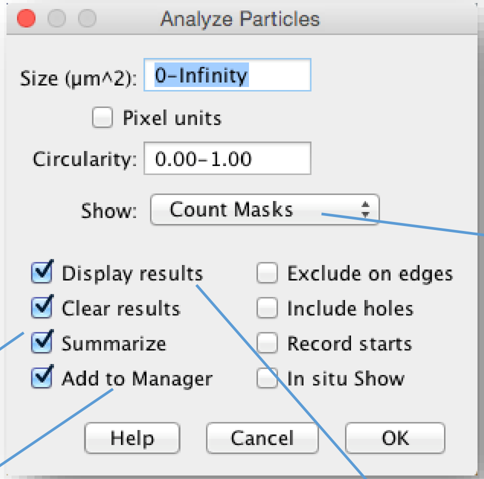
Slide adapted from Haase, Lombardot, Scientific Computing Facility, MPI CBG, licensed CC BY-NC 4.0

The particle analyser

- The particle analyser
  - performs connected components labelling,
  - can make selections and
  - Measure properties of particles (feature extraction)

- Analyze > Analyse particles…





Analyze Particles

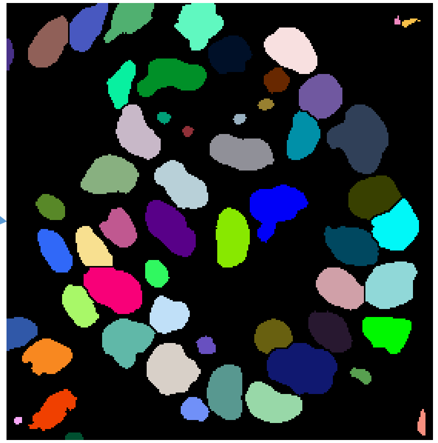
Size (µm^2): 0-Infinity

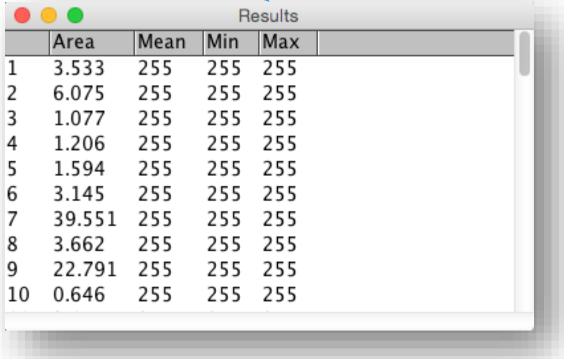
□ Pixel units

Circularity: 0.00-1.00

Show: Count Masks

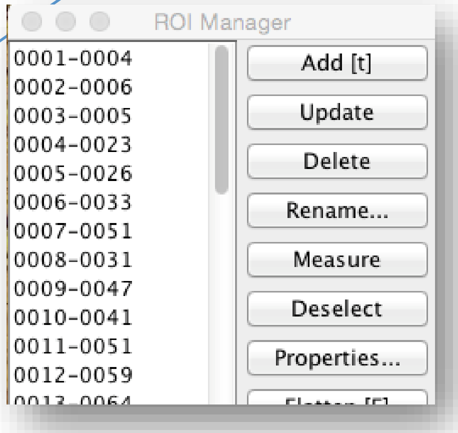
☑ Display results
☑ Clear results
☑ Summarize
☑ Add to Manager
□ Exclude on edges
□ Include holes
□ Record starts
□ In situ Show





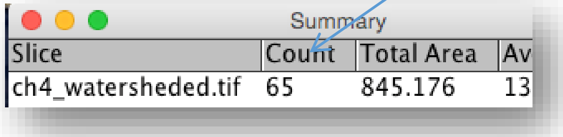
Results

	Area	Mean	Min	Max
1	3.533	255	255	255
2	6.075	255	255	255
3	1.077	255	255	255
4	1.206	255	255	255
5	1.594	255	255	255
6	3.145	255	255	255
7	39.551	255	255	255
8	3.662	255	255	255
9	22.791	255	255	255
10	0.646	255	255	255



ROI Manager

0001-0004
0002-0006
0003-0005
0004-0023
0005-0026
0006-0033
0007-0051
0008-0031
0009-0047
0010-0041
0011-0051
0012-0059
0013-0064



Summary

Slice	Count	Total Area	Av
ch4_watersheded.tif	65	845.176	13

PoL Physics of Life TU Dresden

CENTER FOR SYSTEMS BIOLOGY DRESDEN

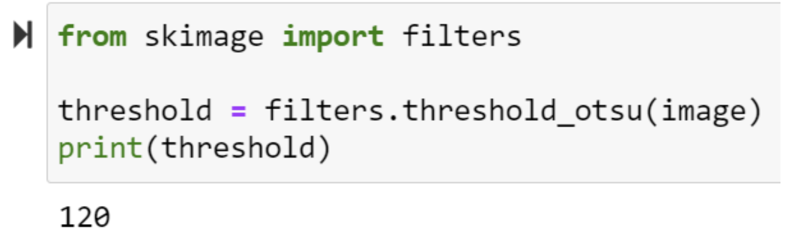
Image Segmentation in Python using scikit-image

Robert Haase

May 2021

Image segmentation: thresholding

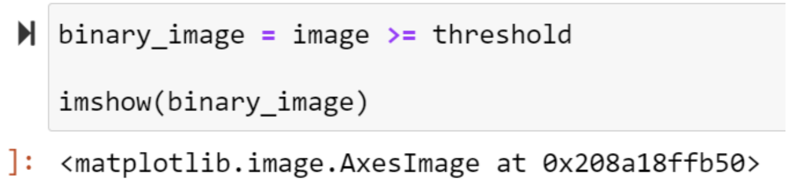
- Threshold algorithms are also filters in scikit-image and result in a number!



```
from skimage import filters

threshold = filters.threshold_otsu(image)
print(threshold)
```

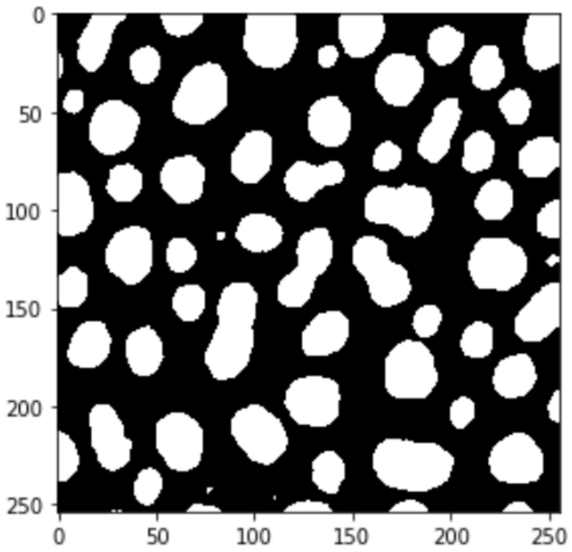
```
120
```



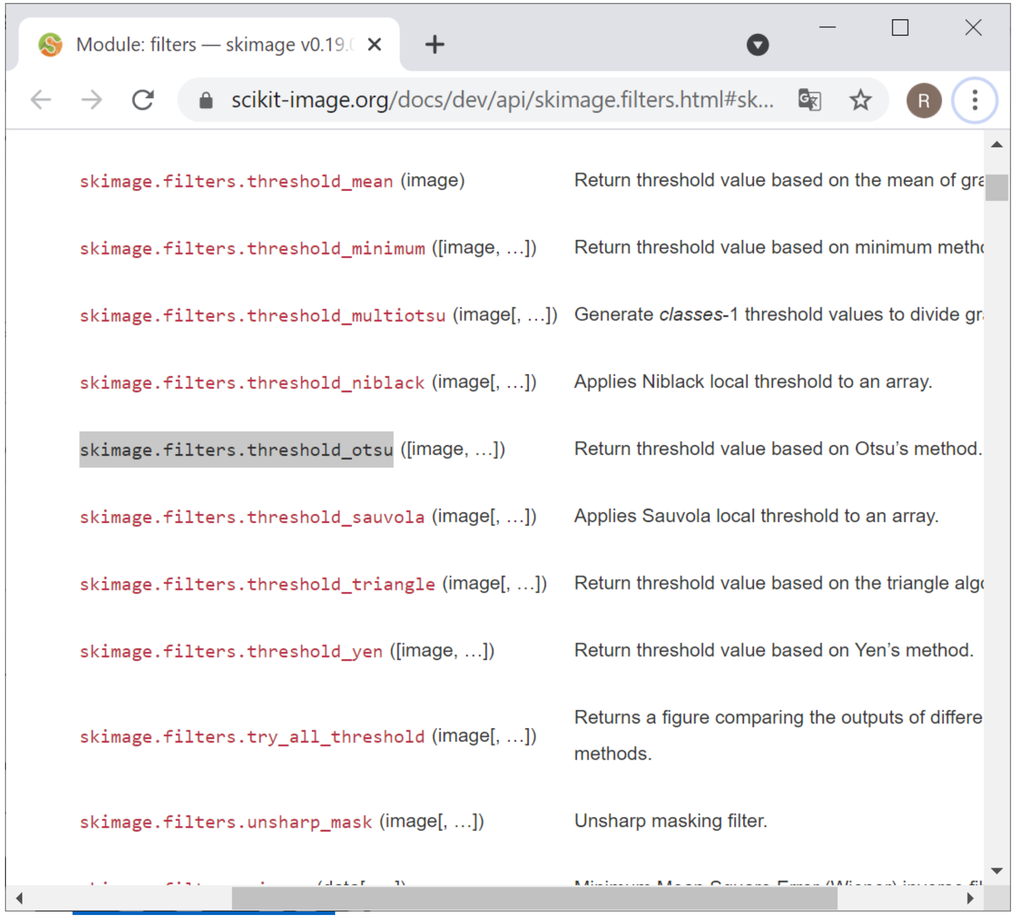
```
binary_image = image >= threshold

imshow(binary_image)
```

```
]: <matplotlib.image.AxesImage at 0x208a18ffb50>
```



Module: filters



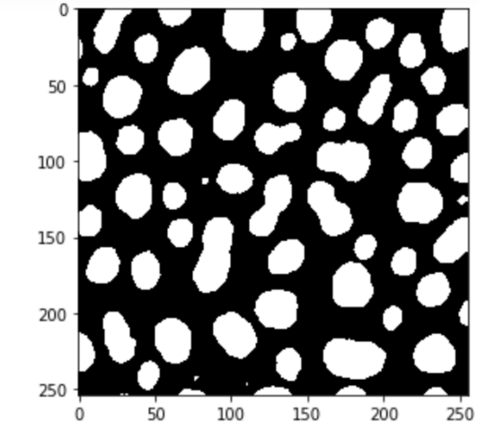
Module: filters — skimage v0.19.
scikit-image.org/docs/dev/api/skimage.filters.html#sk...

skimage.filters.threshold_mean (image)	Return threshold value based on the mean of gra
skimage.filters.threshold_minimum ([image, …])	Return threshold value based on minimum metho
skimage.filters.threshold_multiotsu (image[, …])	Generate *classes*-1 threshold values to divide gr
skimage.filters.threshold_niblack (image[, …])	Applies Niblack local threshold to an array.
skimage.filters.threshold_otsu ([image, …])	Return threshold value based on Otsu's method.
skimage.filters.threshold_sauvola (image[, …])	Applies Sauvola local threshold to an array.
skimage.filters.threshold_triangle (image[, …])	Return threshold value based on the triangle algo
skimage.filters.threshold_yen ([image, …])	Return threshold value based on Yen's method.
skimage.filters.try_all_threshold (image[, …])	Returns a figure comparing the outputs of differe methods.
skimage.filters.unsharp_mask (image[, …])	Unsharp masking filter.

image.org/docs/dev/api/skimage.filters.html#skimage.filters.threshold_otsu

Post-processing binary images

- Some filters are collected in skimage morphology module. Take care, the naming is a bit different to what you learned earlier.

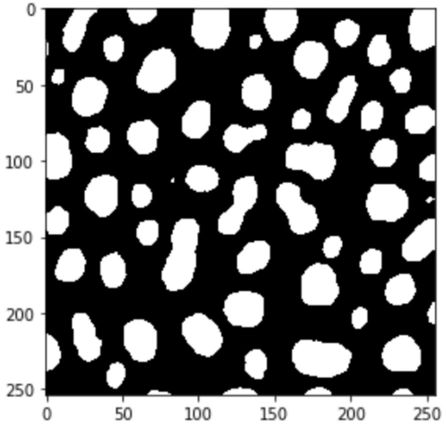


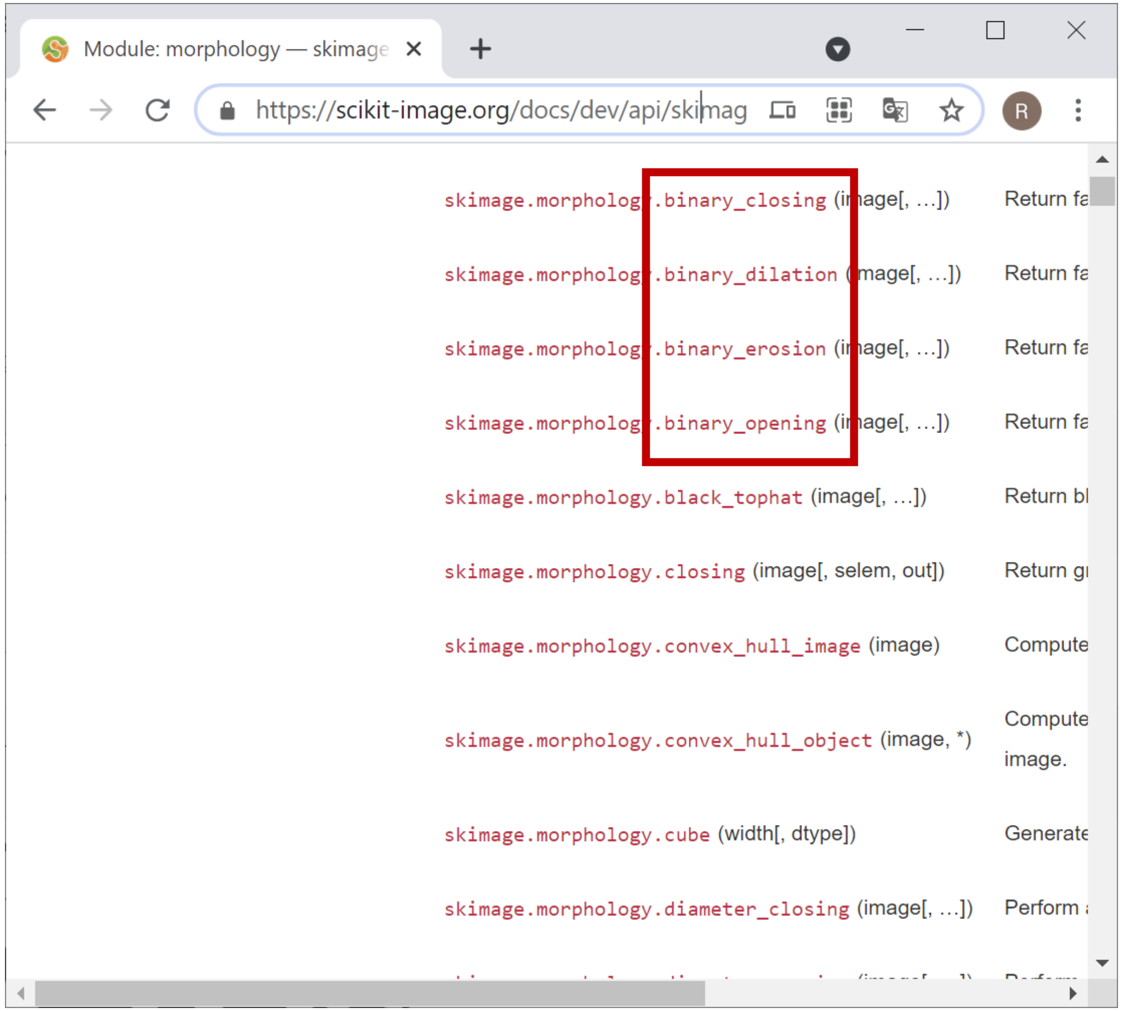
```
from skimage.morphology import binary_erosion

shrinked_objects = binary_erosion(binary_image)

imshow(shrinked_objects)
```

```
<matplotlib.image.AxesImage at 0x1b68958a7c0>
```




Module: morphology — skimage
https://scikit-image.org/docs/dev/api/skimag

skimage.morphology.binary_closing (image[, …]) Return fa

skimage.morphology.binary_dilation (image[, …]) Return fa

skimage.morphology.binary_erosion (image[, …]) Return fa

skimage.morphology.binary_opening (image[, …]) Return fa

skimage.morphology.black_tophat (image[, …]) Return bl

skimage.morphology.closing (image[, selem, out]) Return gi

skimage.morphology.convex_hull_image (image) Compute

skimage.morphology.convex_hull_object (image, *) Compute image.

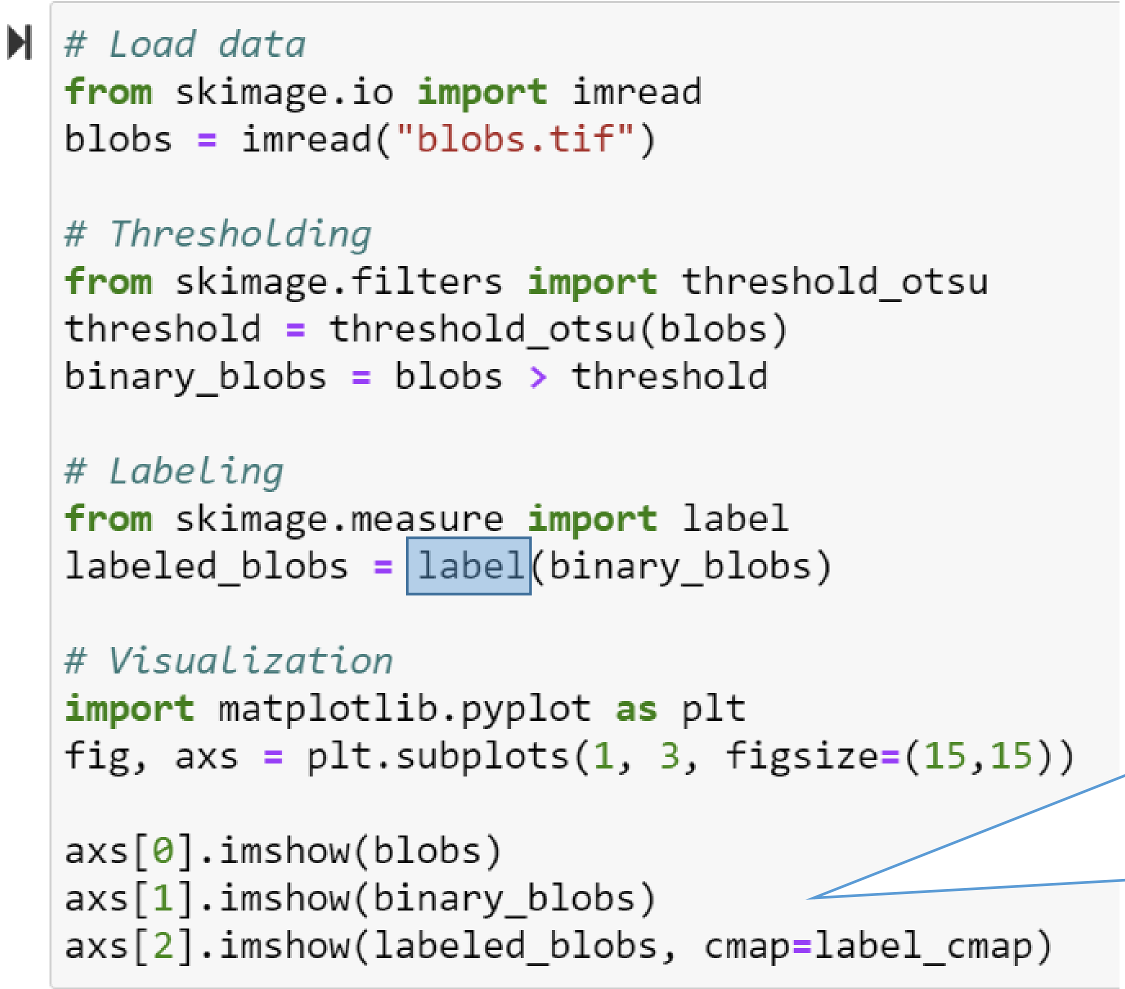
skimage.morphology.cube (width[, dtype]) Generate

skimage.morphology.diameter_closing (image[, …]) Perform

https://scikit-image.org/docs/dev/api/skimage.morphology.html

Connected components analysis

- The label function applies connected component analysis.



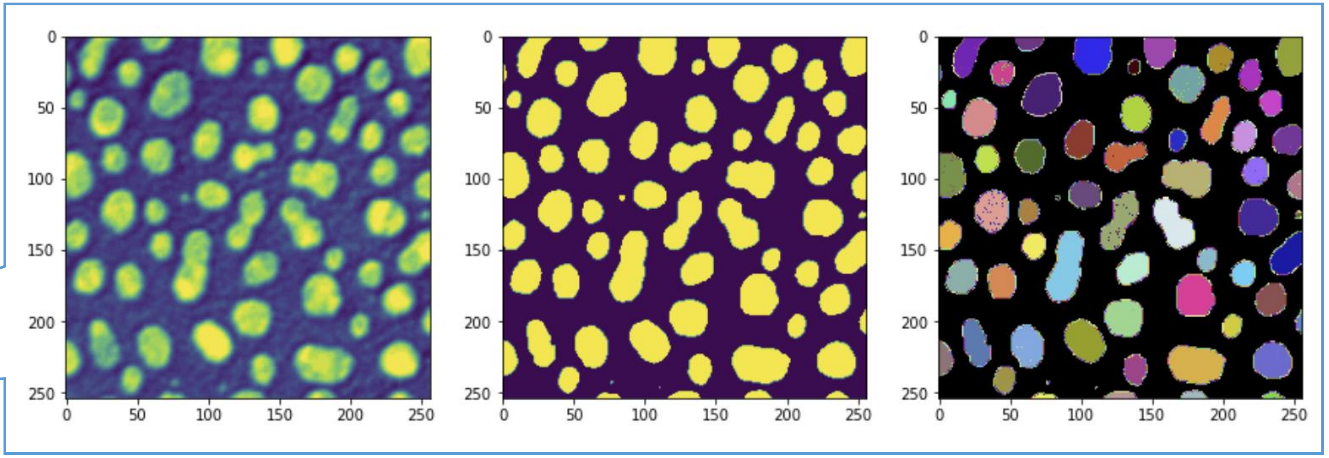
```
# Load data
from skimage.io import imread
blobs = imread("blobs.tif")

# Thresholding
from skimage.filters import threshold_otsu
threshold = threshold_otsu(blobs)
binary_blobs = blobs > threshold

# Labeling
from skimage.measure import label
labeled_blobs = label(binary_blobs)

# Visualization
import matplotlib.pyplot as plt
fig, axs = plt.subplots(1, 3, figsize=(15,15))

axs[0].imshow(blobs)
axs[1].imshow(binary_blobs)
axs[2].imshow(labeled_blobs, cmap=label_cmap)
```



https://scikit-image.org/docs/dev/api/skimage.measure.html#skimage.measure.label

Working with napari

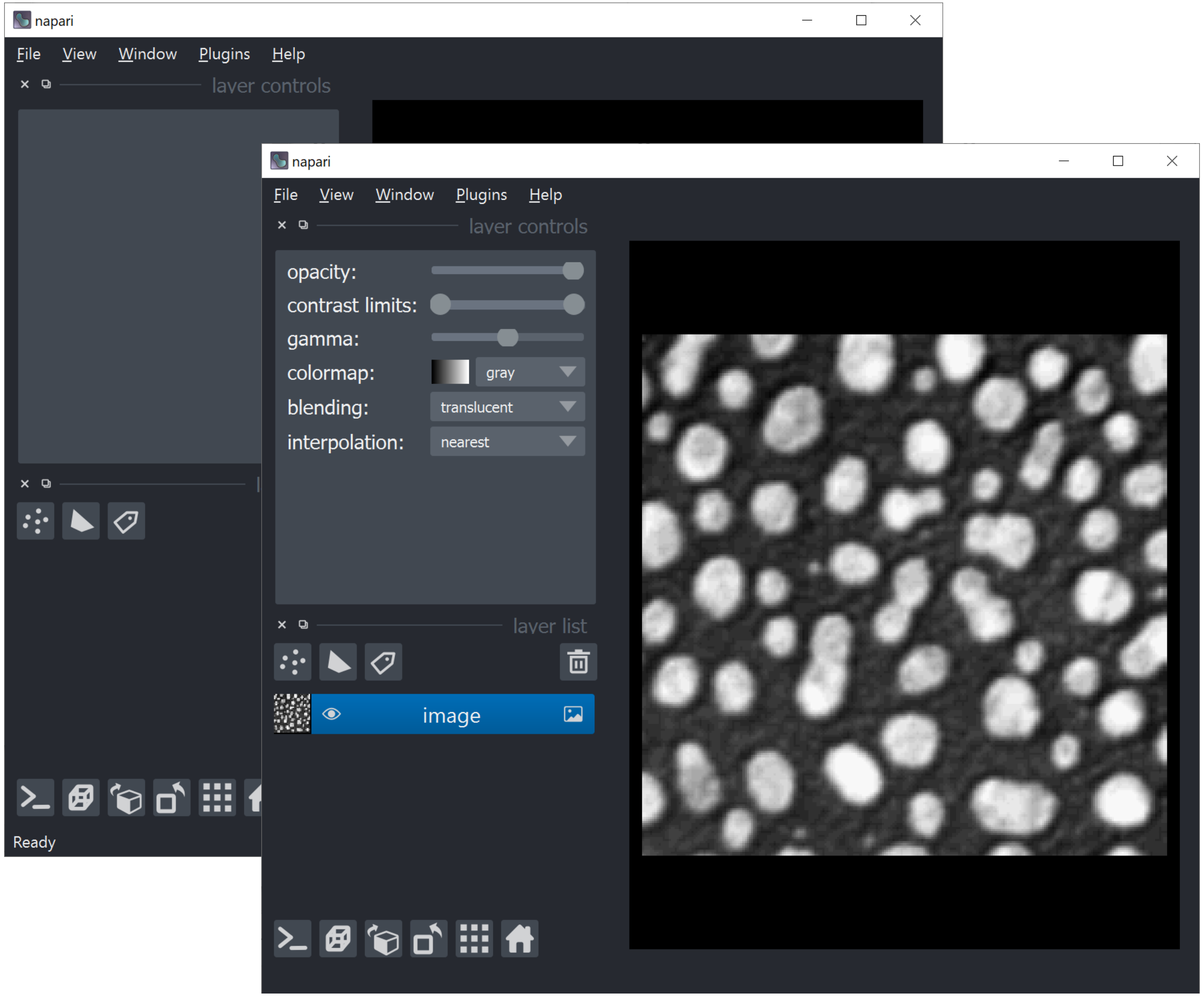
- Initialization

```
import napari

# Create an empty viewer
viewer = napari.Viewer()
```

- Adding images

```
viewer.add_image(image)
```



opacity:

contrast limits:

gamma:

colormap: gray

blending: translucent

interpolation: nearest

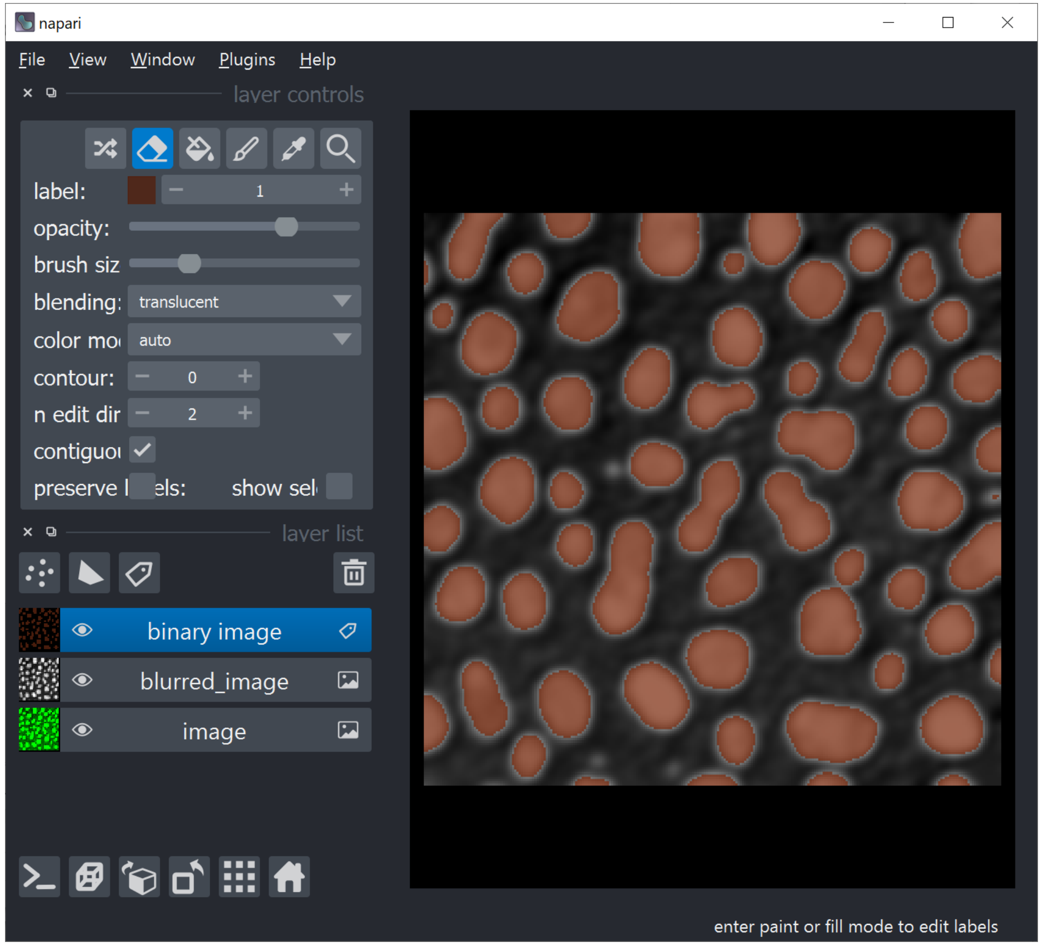
http://nbviewer.jupyter.org/github/BiAPoL/on_the_fly_image_processing_napari/blob/master/01_napari.ipynb

Visualizing image segmentation

- Binary images and label images visualized as label layers

```
# Add a new labels layer containing an image
viewer.add_labels(binary_image,
                  name="binary image")
```

Name your layers to keep track of what they contain



label: 1
opacity:
brush siz
blending: translucent
color mo: auto
contour: 0
n edit dir 2
contiguo ☑
preserve l□els: show sel □

binary image
blurred_image
image

enter paint or fill mode to edit labels

http://nbviewer.jupyter.org/github/BiAPoL/on_the_fly_image_processing_napari/blob/master/01_napari.ipynb

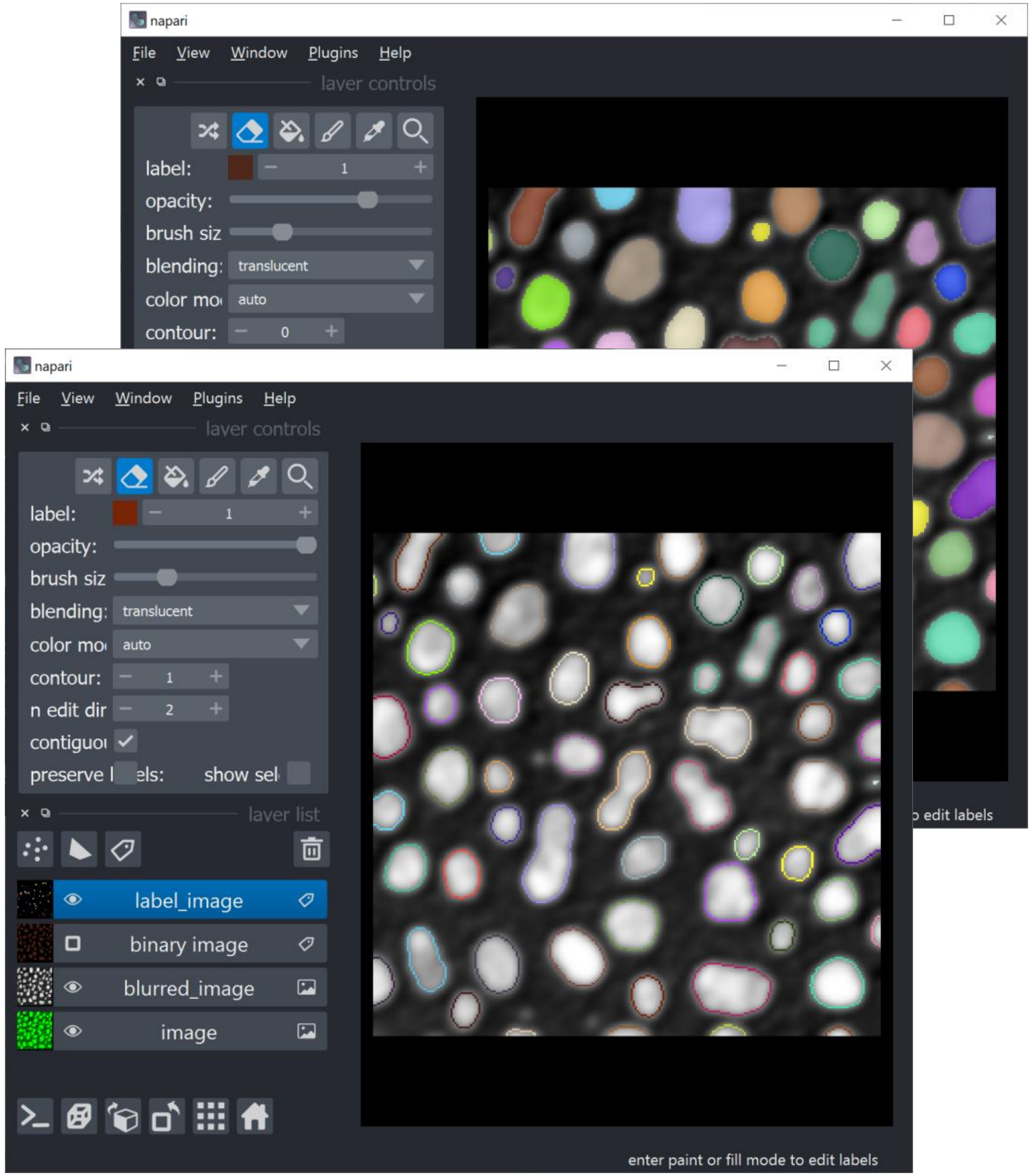
Visualizing image segmentation

- Binary images and label images visualized as label layers

```
# add labels to viewer
label_layer = viewer.add_labels(label_image)
```

- Visualize contours instead of the overlay

```
label_layer.contour = 1
label_layer.opacity = 1
```



label: 1
opacity:
brush siz
blending: translucent
color mo auto
contour: 0

label: 1
opacity:
brush siz
blending: translucent
color mo auto
contour: 1
n edit dir 2
contiguo ✓
preserve l els: □ show sel □

label_image
binary image
blurred_image
image

enter paint or fill mode to edit labels

http://nbviewer.jupyter.org/github/BiAPoL/on_the_fly_image_processing_napari/blob/master/01_napari.ipynb

Summary

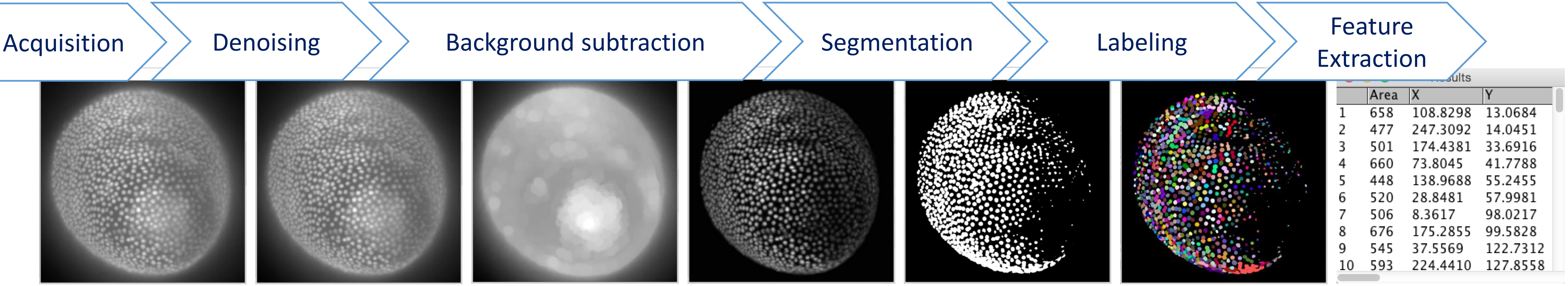
Today, you learned

- Thresholding / segmentation
- Morphological operations
- Connected components labelling

- Thresholding + labeling in Fiji and napari

Next week:

- Feature extraction / measurements
- Batch processing
- 3D image analysis



	Area	X	Y
1	658	108.8298	13.0684
2	477	247.3092	14.0451
3	501	174.4381	33.6916
4	660	73.8045	41.7788
5	448	138.9688	55.2455
6	520	28.8481	57.9981
7	506	8.3617	98.0217
8	676	175.2855	99.5828
9	545	37.5569	122.7312
10	593	224.4410	127.8558