Image Processing and Filtering

Till Korten

With material from

Marcelo Leomil Zoccoler and Robert Haase, PoL TU Dresden

Mauricio Rocha Martins, Norden lab, MPI CBG

Dominic Waithe, Oxford University

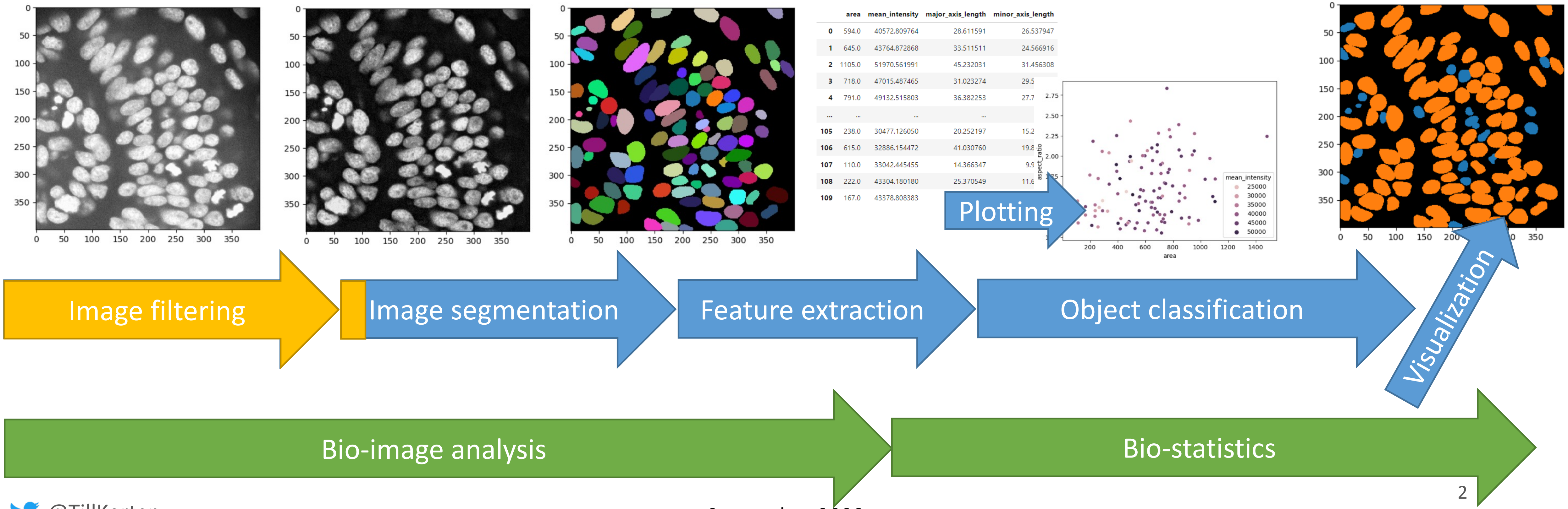
Alex Bird, Dan White, MPI CBG

April 2023

@TillKorten

Lecture overview: Bio-image Analysis

- Image Data Analysis workflows
- Goal: **Quantify observations, substantiate conclusions with numbers**



	area	mean_intensity	major_axis_length	minor_axis_length
0	594.0	40572.809764	28.611591	26.537947
1	645.0	43764.872868	33.511511	24.566916
2	1105.0	51970.561991	45.232031	31.456308
3	718.0	47015.487465	31.023274	29.5
4	791.0	49132.515803	36.382253	27.7
...	...	...	...	
105	238.0	30477.126050	20.252197	15.2
106	615.0	32886.154472	41.030760	19.8
107	110.0	33042.445455	14.366347	9.9
108	222.0	43304.180180	25.370549	11.6
109	167.0	43378.808383		

Image Filtering

Till Korten

With material from

Robert Haase and Marcelo Leomil Zoccoler, PoL, TU Dresden

September 2023

@TillKorten

Filters

- An image processing filter is an operation on an image.
- It takes an image and produces a new image out of it.
- Filters change pixel values.

- There is no “best” filter. Which filter fits your needs, depends on the context.
- Filters do not do magic. They can not make things visible which are not in the image.

- Application examples
  - Noise-reduction
  - Artefact-removal
  - Contrast enhancement
  - Correct uneven illumination

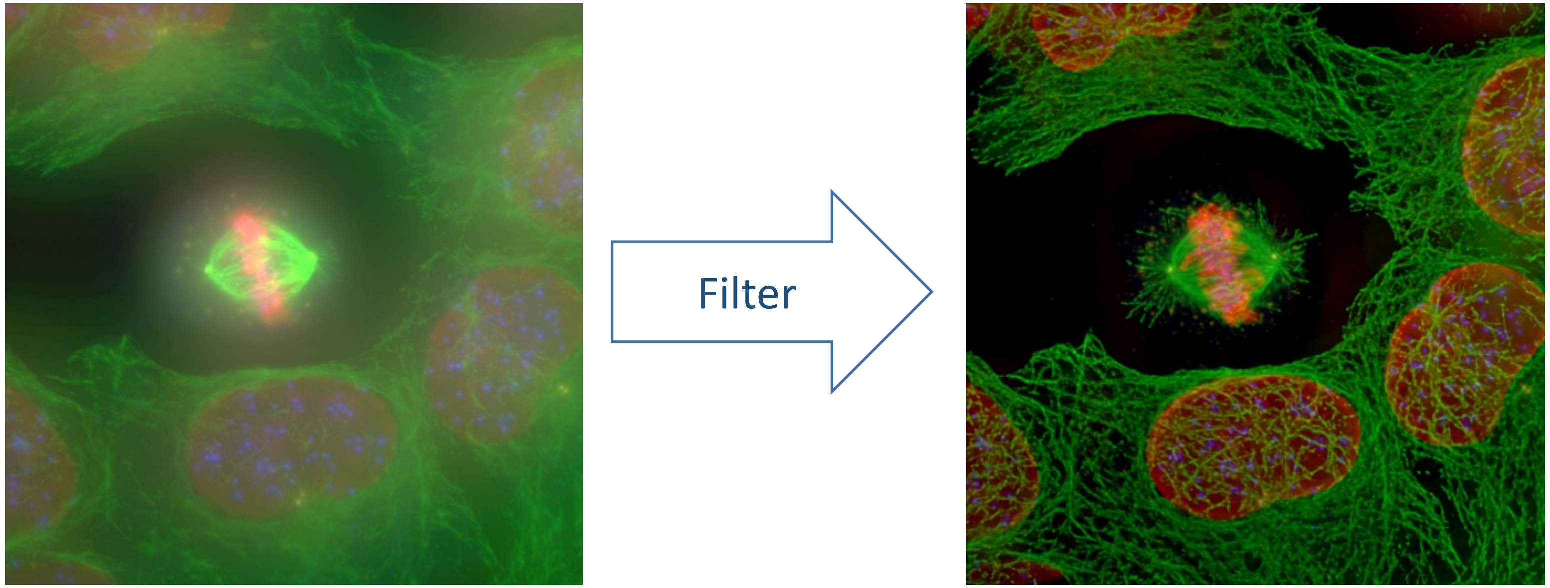
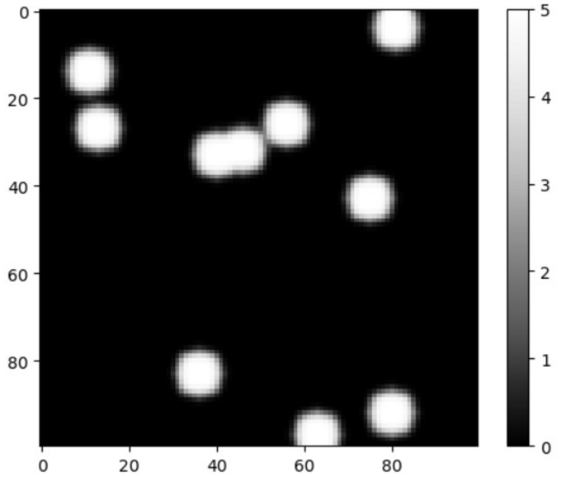


Image source: Alex Bird / Dan White MPI CBG

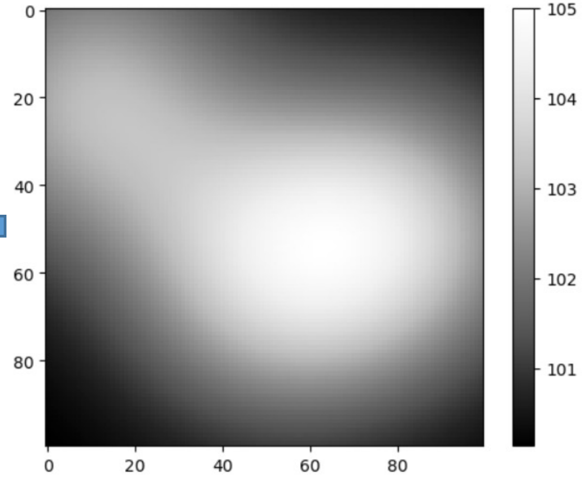
Effects harming image quality

- Image formation (simulated)

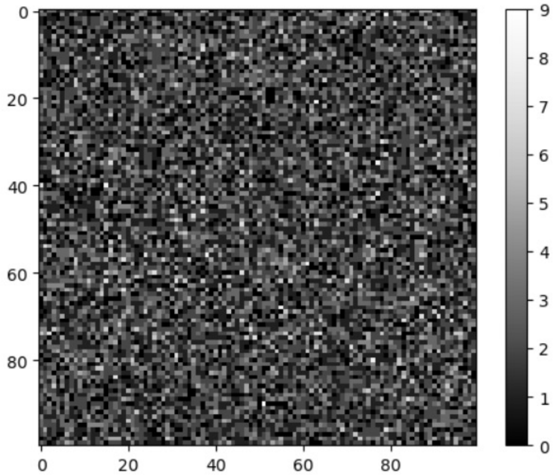
"nuclei"



"background"



"noise"



- Aberrations, defocus
- Motion blur

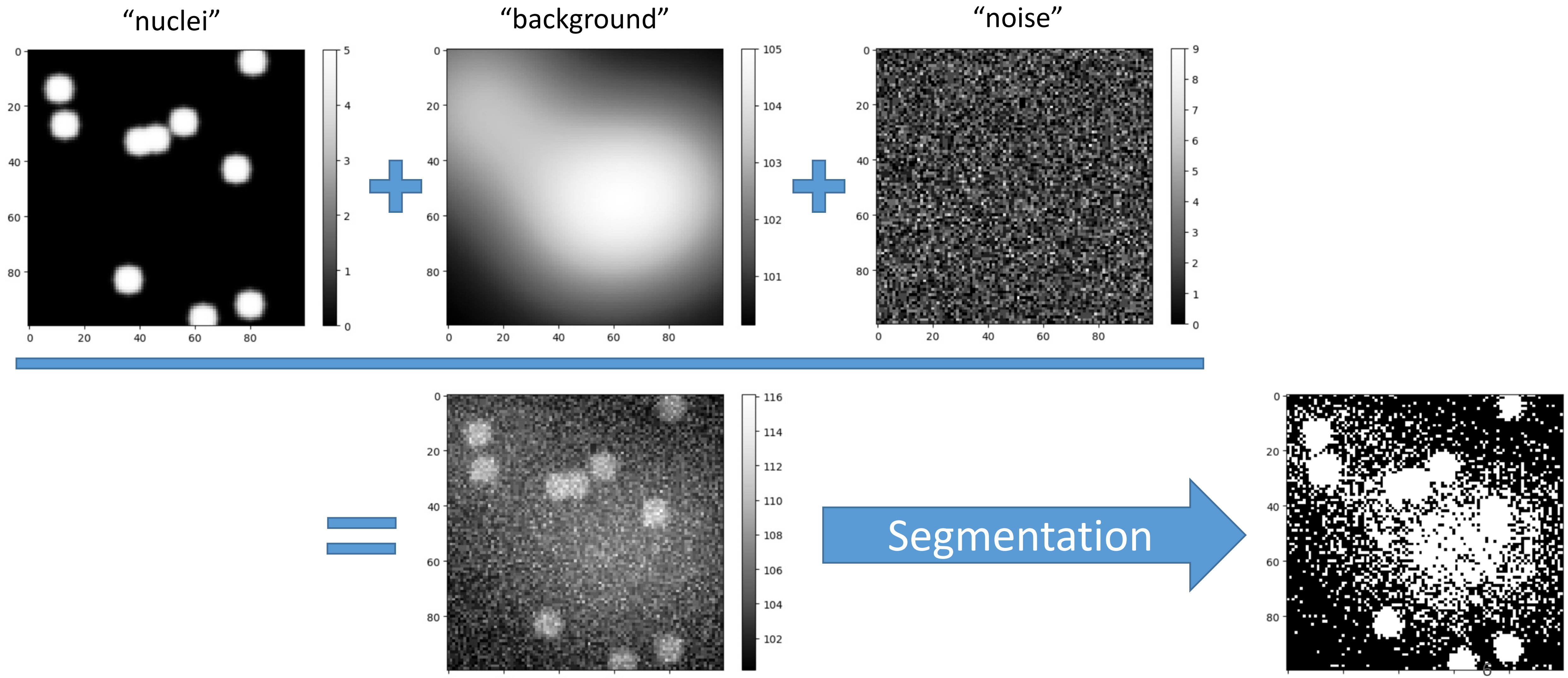
- Light from objects behind and in front of the scene (out-of-focus light)
- Dirt on the object slide
- Camera offset

- Shot noise (arriving photons)
- Dark noise (electrons made from photons)
- Read-out-noise (electronics)

https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/49a787514a367829c3e0e1832f6cc533e96d549f/03_image_processing/simulated_dataset.ipynb

Effects harming image quality

- Image formation (simulated)



https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/49a787514a367829c3e0e1832f6cc533e96d549f/03_image_processing/simulated_dataset.ipynb

Image filtering

- We need to remove the noise to help the computer *interpreting* the image

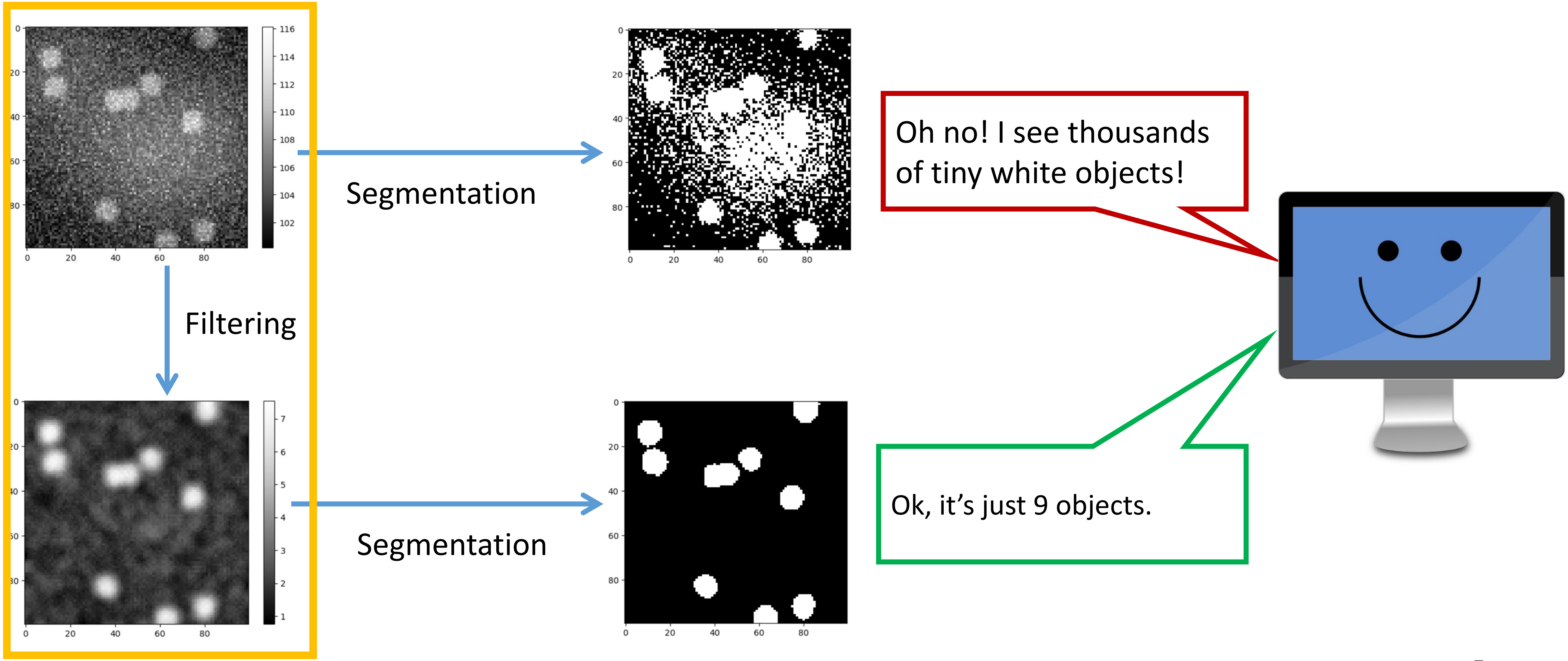
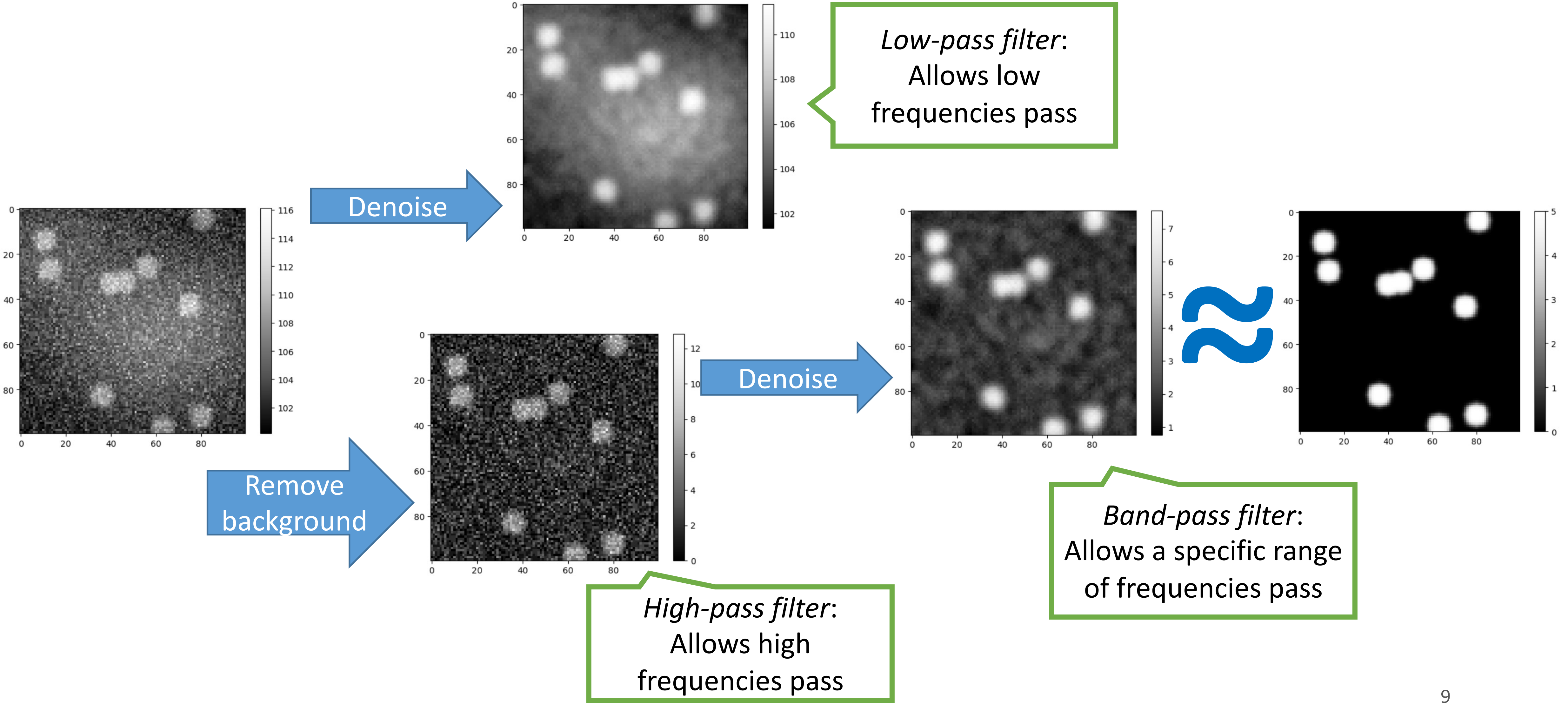


Image filtering

- Attempt to invert / “undo” processes disturbing image quality

Image filtering

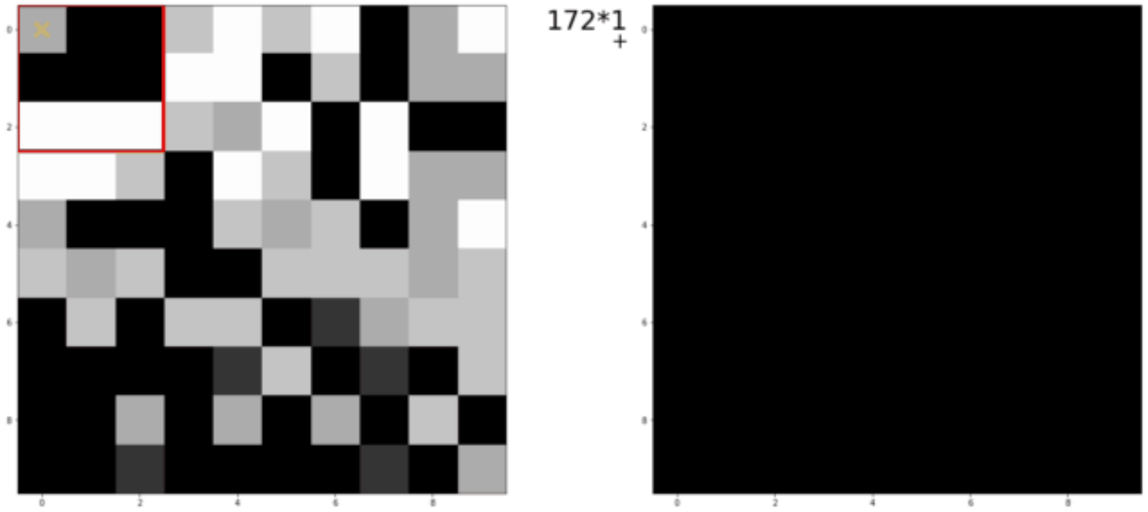
- Attempt to invert / "undo" processes disturbing image quality



https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/49a787514a367829c3e0e1832f6cc533e96d549f/03_image_processing/simulated_dataset.ipynb

Linear Filters

- *Linear filters* replace each pixel value with a weighted linear combination of surrounding pixels
- Filter *kernels* are matrices describing a linear filter
- This multiplication of surrounding pixels according to a matrix is called *convolution*



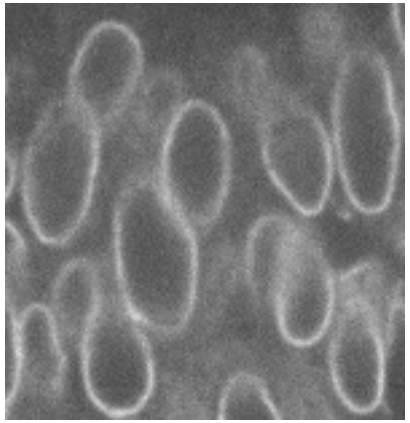
Mean filter, 3x3 kernel

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

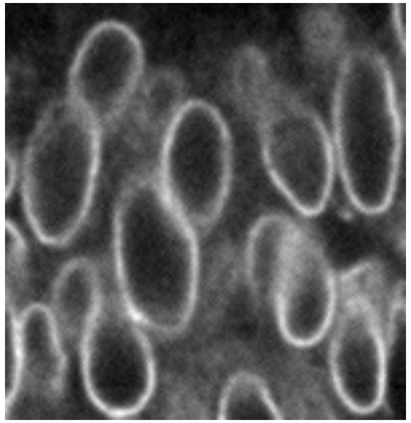
Animation source: Dominic Waithe, Oxford University
https://github.com/dwaithe/generalMacros/tree/master/convolution_ani

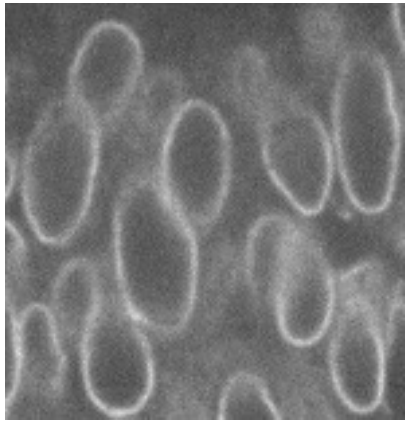
Linear filters

- Terminology:
  - “We convolve an image with a kernel.”
  - Convolution operator: *
  - Examples
    - Mean
    - Gaussian blur
    - Sobel-operator
    - Laplace-filter

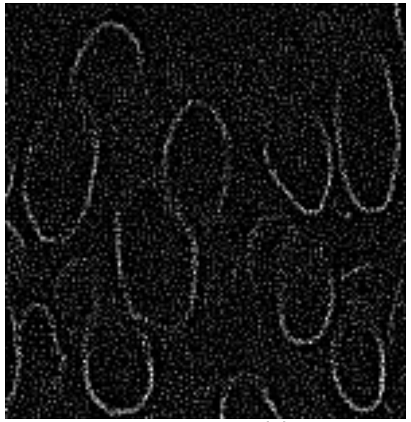


$$* \begin{bmatrix} 1 & 1 & 1 \\ 1 & 8 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$





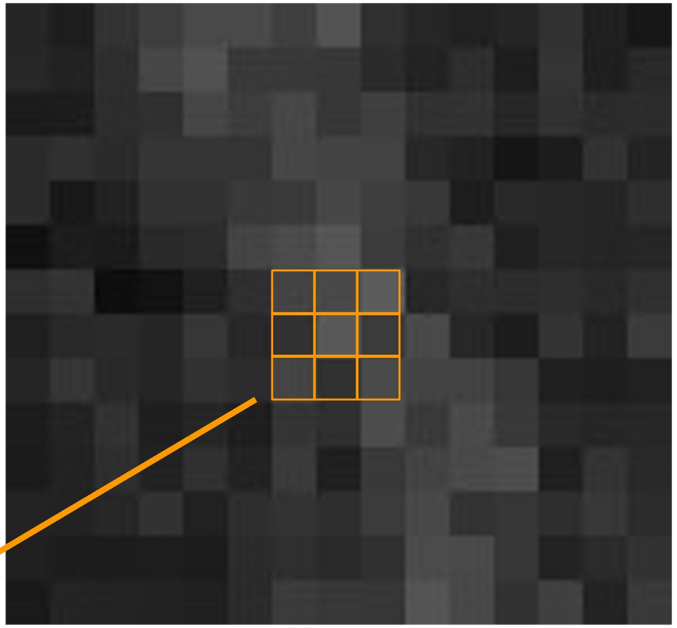
$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} =$$



Nonlinear Filters

- Non linear filters also replace pixel value inside as rolling window but using a non-linear function.

- Examples: order statistics filters
  - Min
  - Median
  - Max
  - Variance
  - Standard deviation



$$\begin{bmatrix} 75 & 85 & 60 \\ 67 & 73 & 91 \\ 50 & 88 & 59 \end{bmatrix} \rightarrow [\,50 \;\; 59 \;\; 60 \;\; 67 \;\; 73 \;\; 75 \;\; 85 \;\; 88 \;\; 91\,]$$

Min Median Max

Noise removal

- Gaussian filter
- Median filter (computationally expensive)

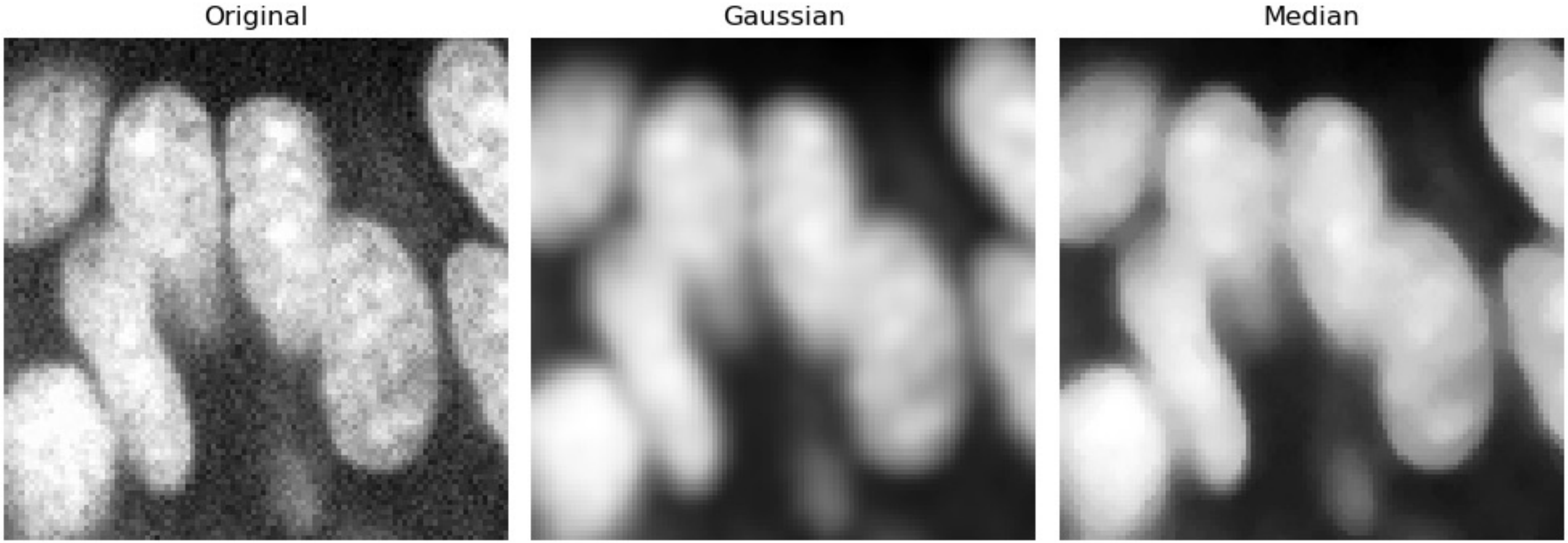
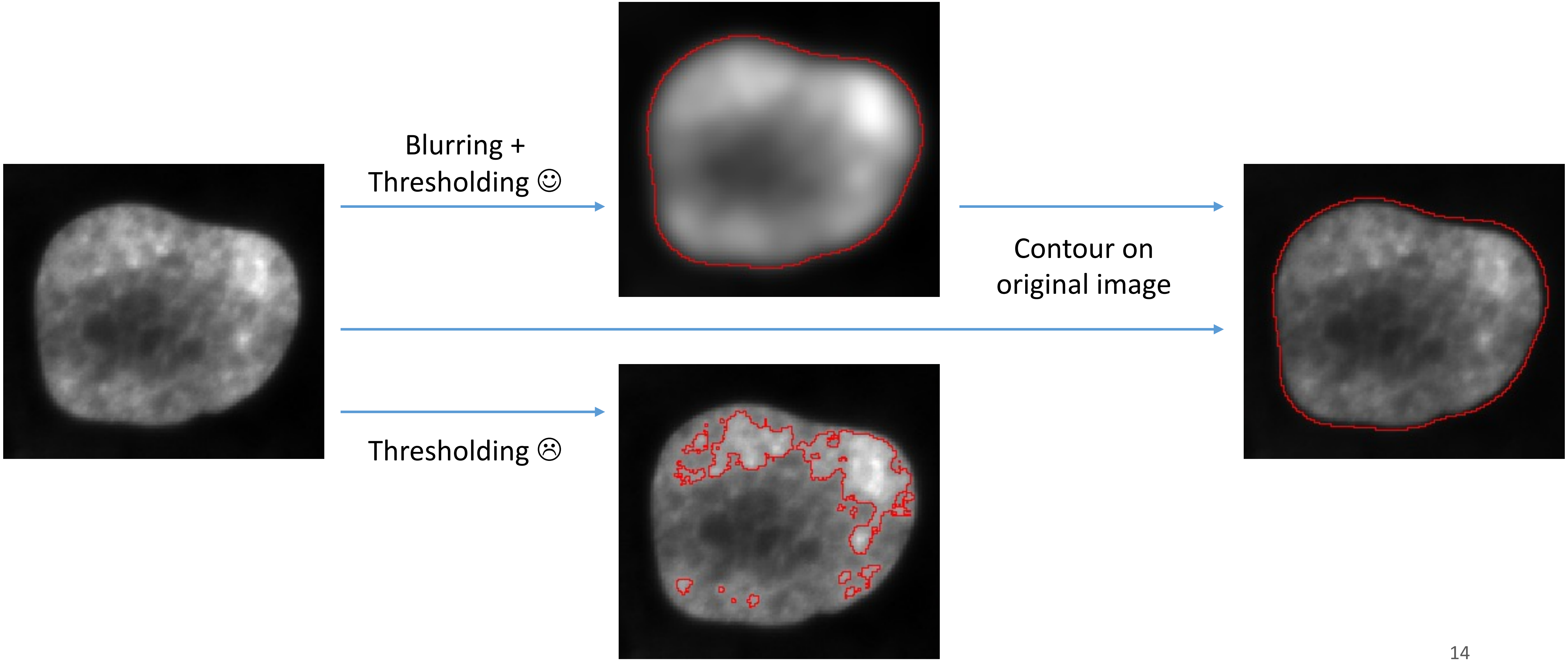


Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

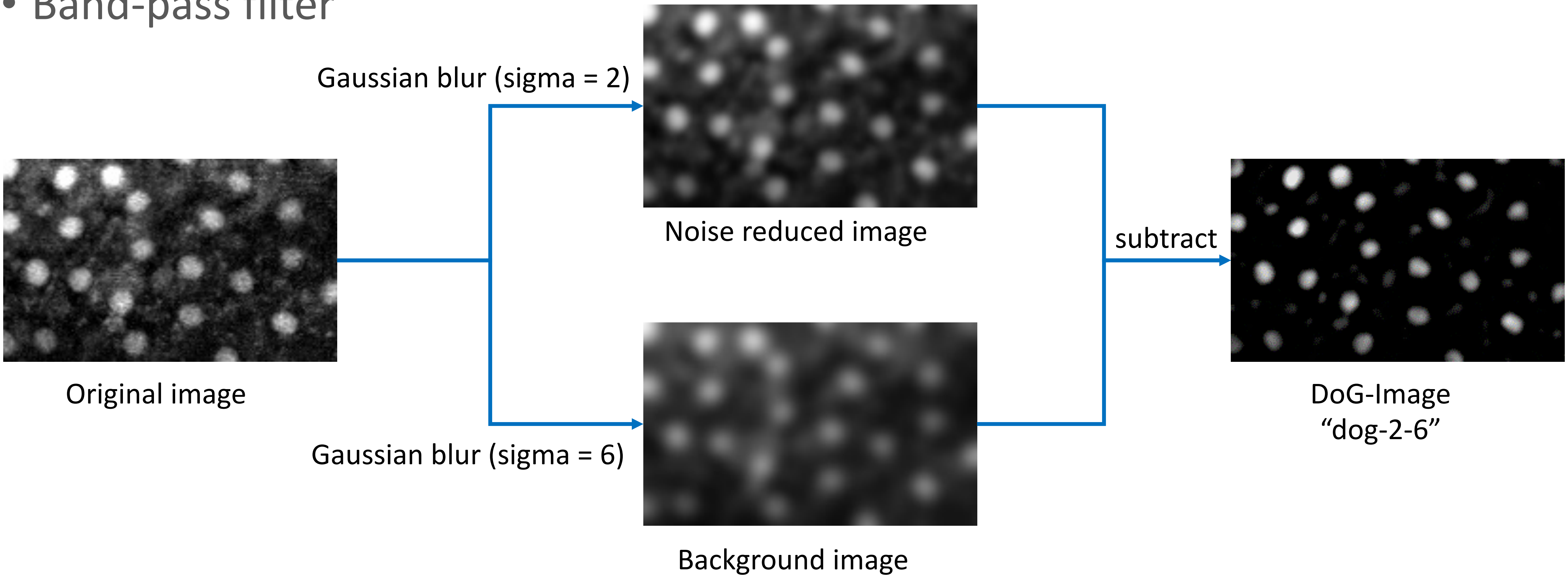
Filtering for improving thresholding results

- 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!



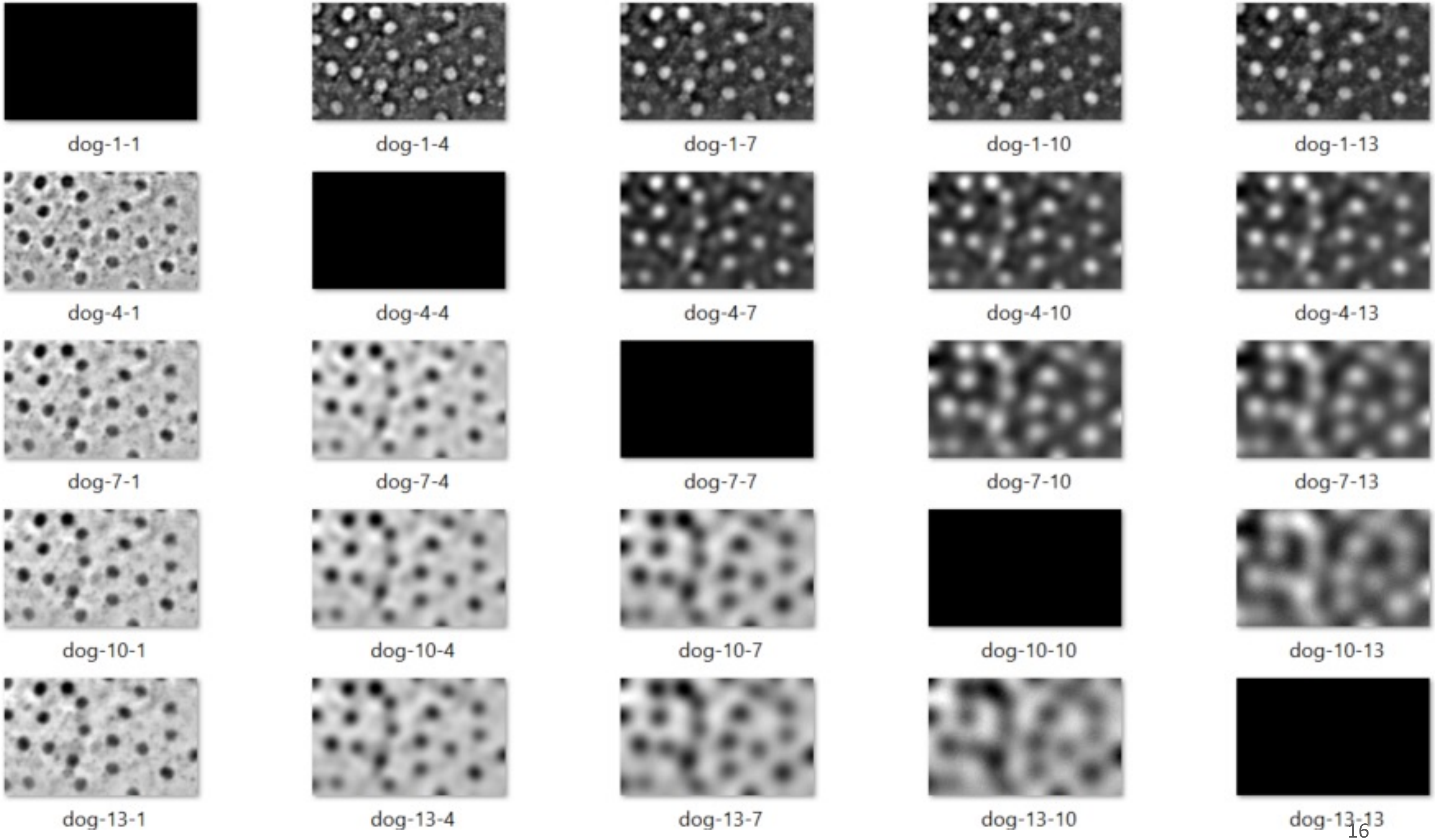
Difference-of-Gaussian (DoG)

- Improve image in order to detect bright objects.
- Band-pass filter



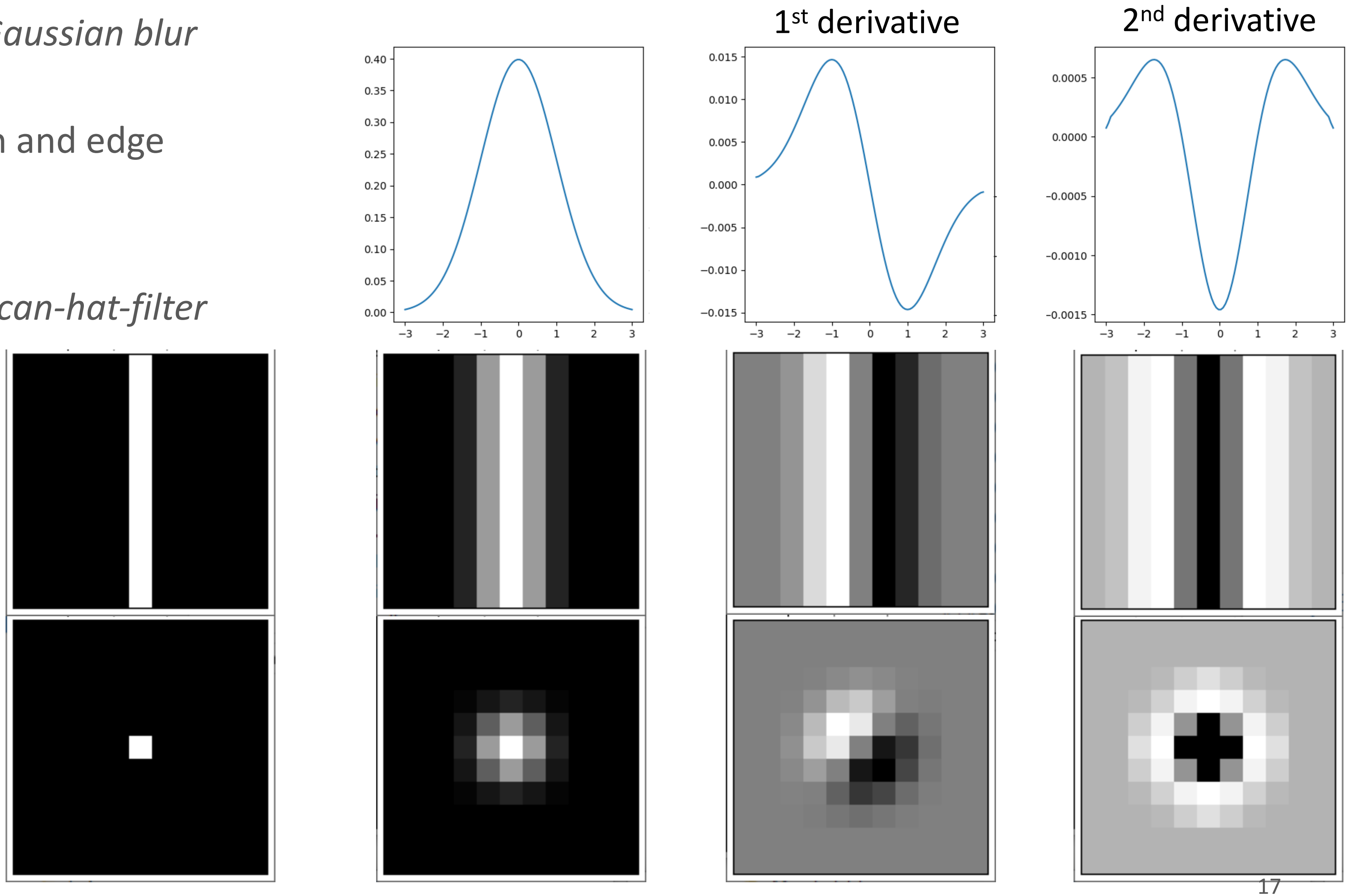
Difference-of-Gaussian (DoG)

- Example DoG images



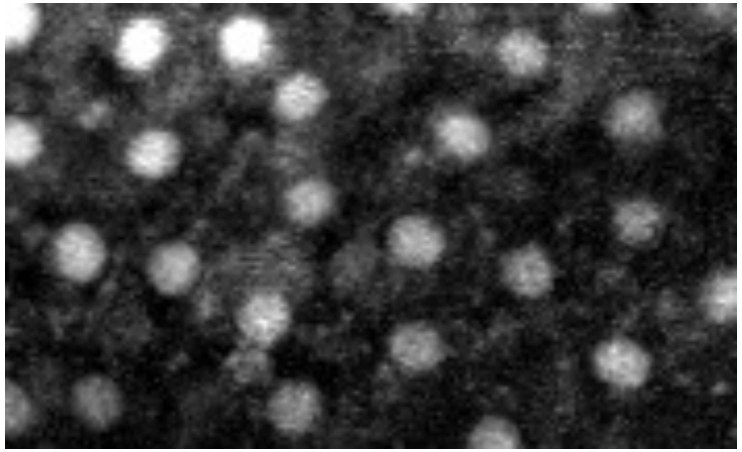
Laplace-filter

- *Second derivative of a Gaussian blur filter*
- Used for edge-detection and edge enhancement
- Also known as the *Mexican-hat-filter*



Laplacian-of-Gaussian (LoG)

Laplace filter

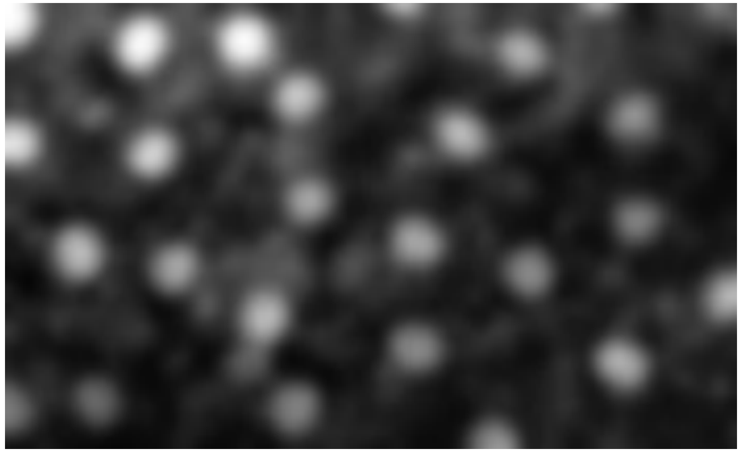


$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} =$$

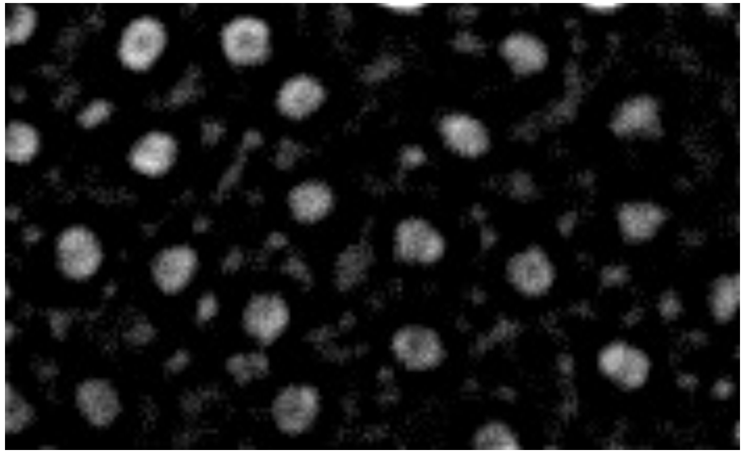
Laplace filtered image

Gaussian filter

Laplacian of Gaussian filter



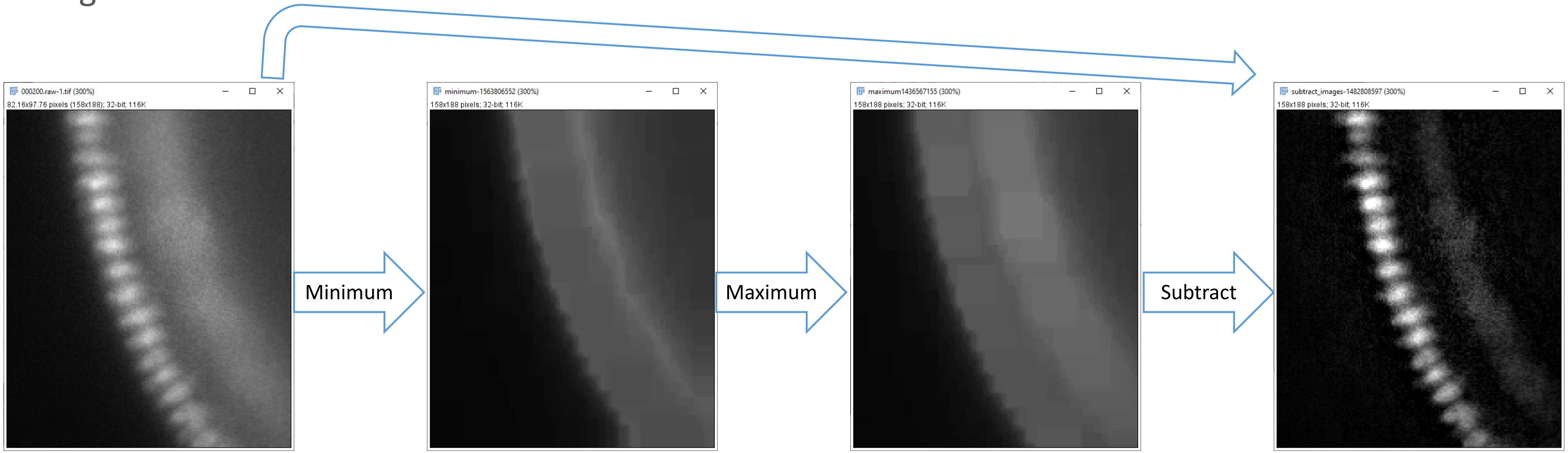
$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} =$$



LoG image

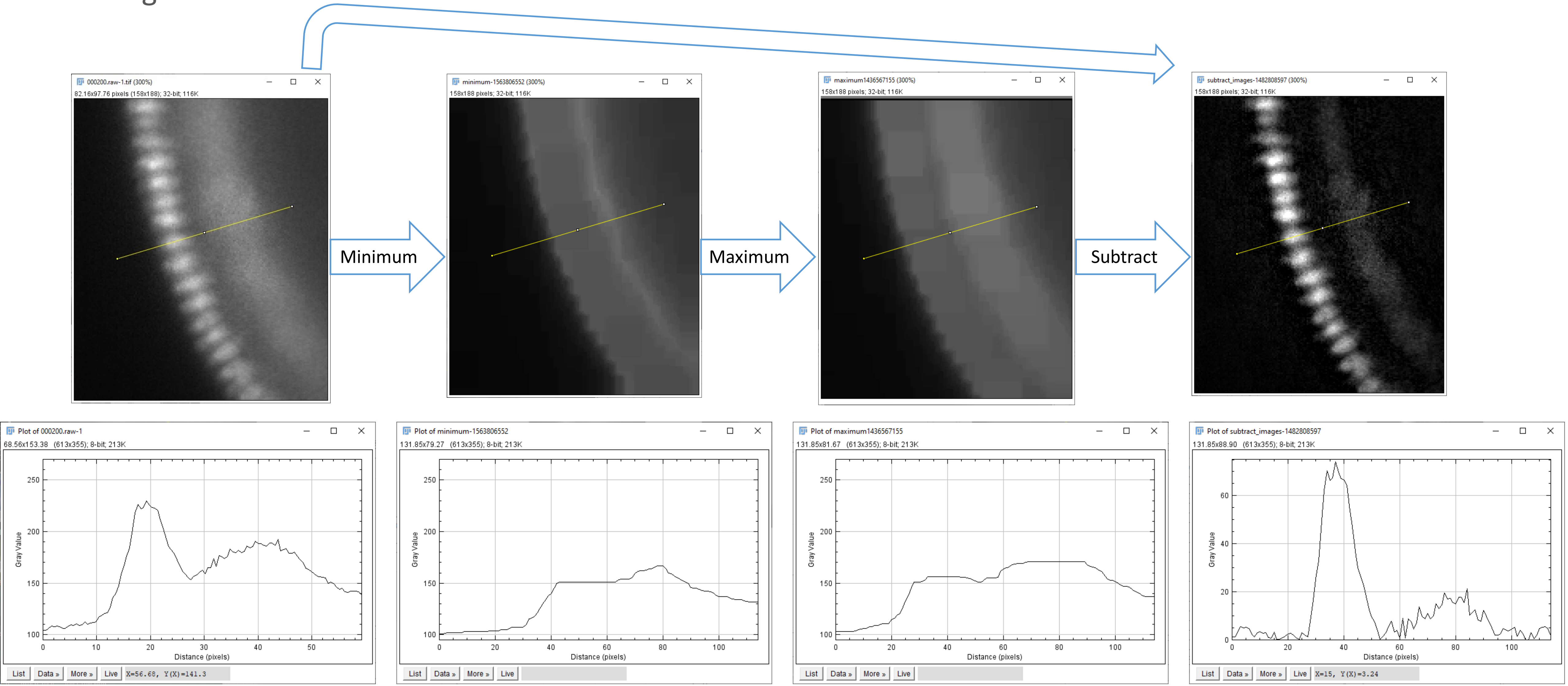
Top-hat filter

- Background subtraction



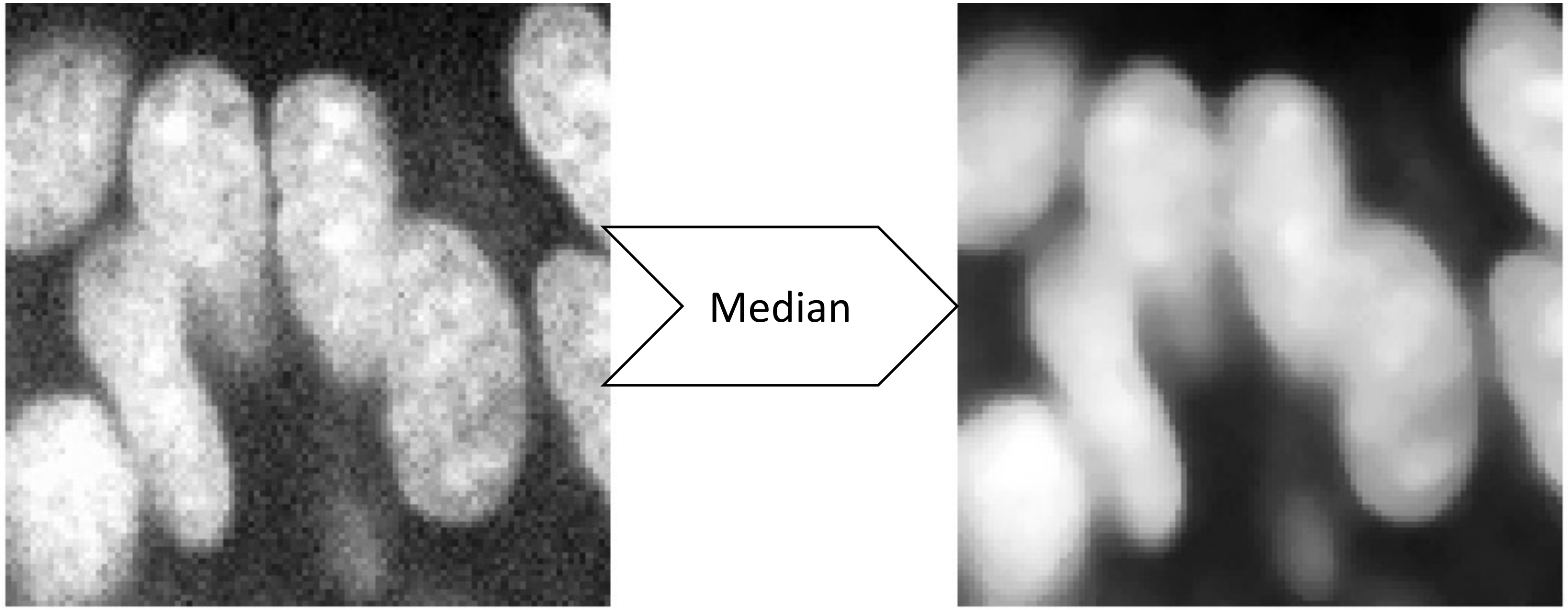
Top-hat filter

- Background subtraction



Quiz: Noise removal

- The median filter is a …

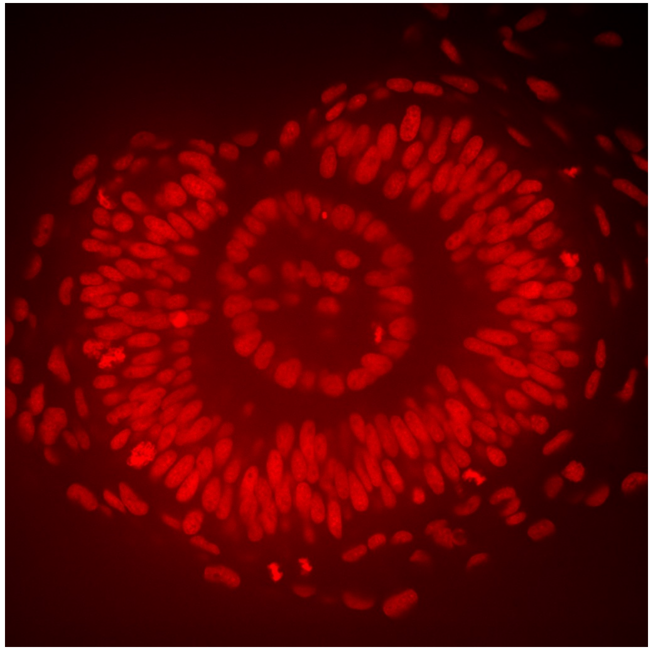


Linear filter

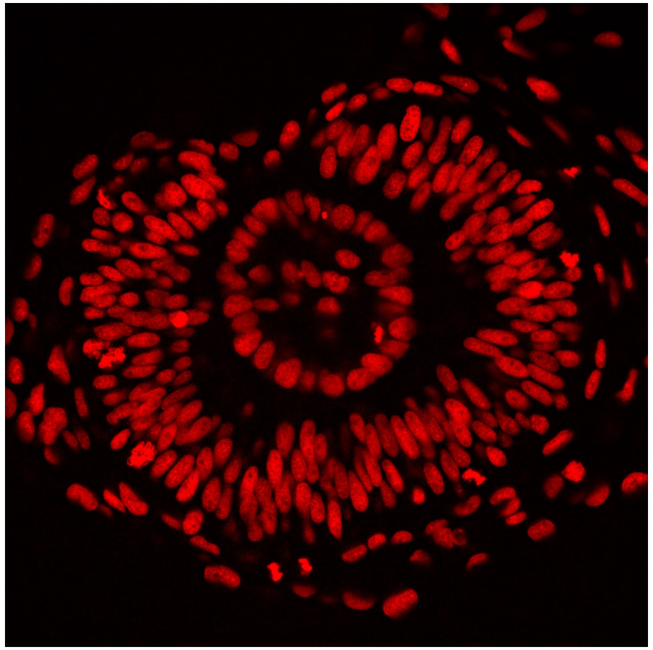
Non-linear filter

Background removal

- Removing background from an image is a … ?



Subtract background



Low-pass filter

High-pass filter

Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

Image segmentation

Till Korten

With material from

Robert Haase, PoL, TU Dresden

September 2023

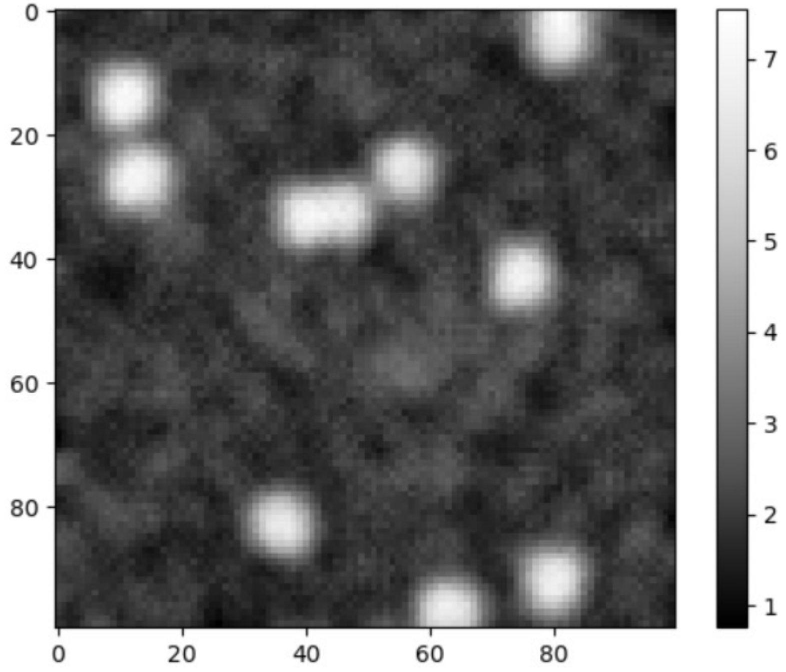
@TillKorten

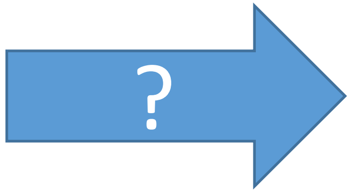
Short detour: Segmentation

Thresholding

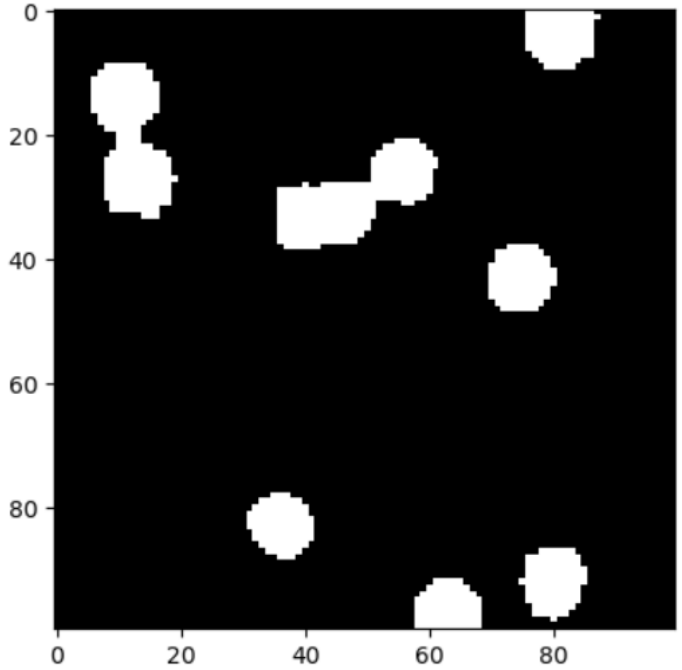
- Very basic and yet efficient segmentation technique
- Histogram based, to determine an intensity threshold
- Not state-of-the-art in many fields (anymore)

Intensity image





Binary image

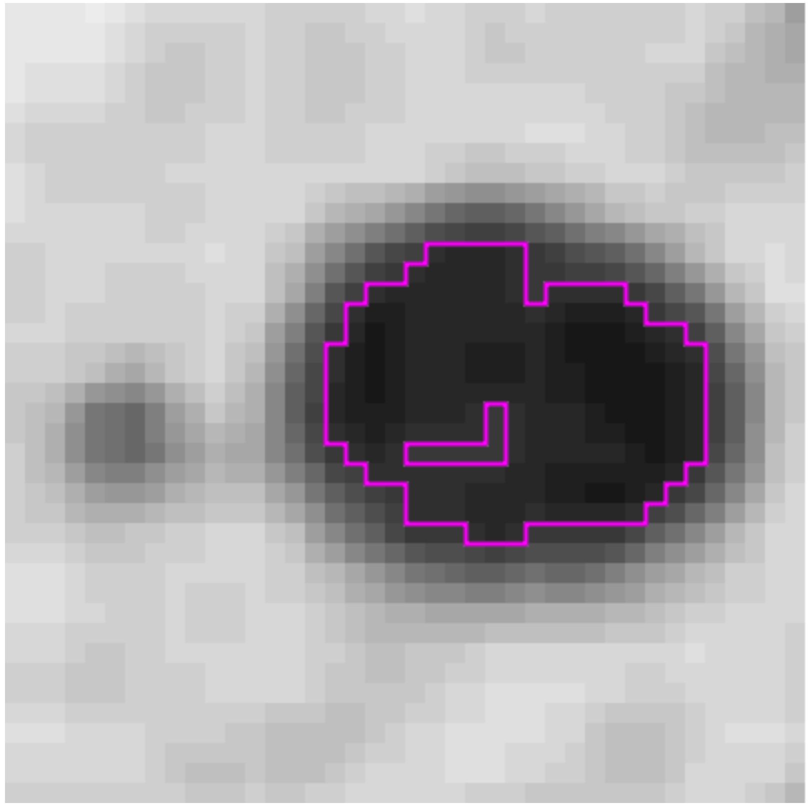


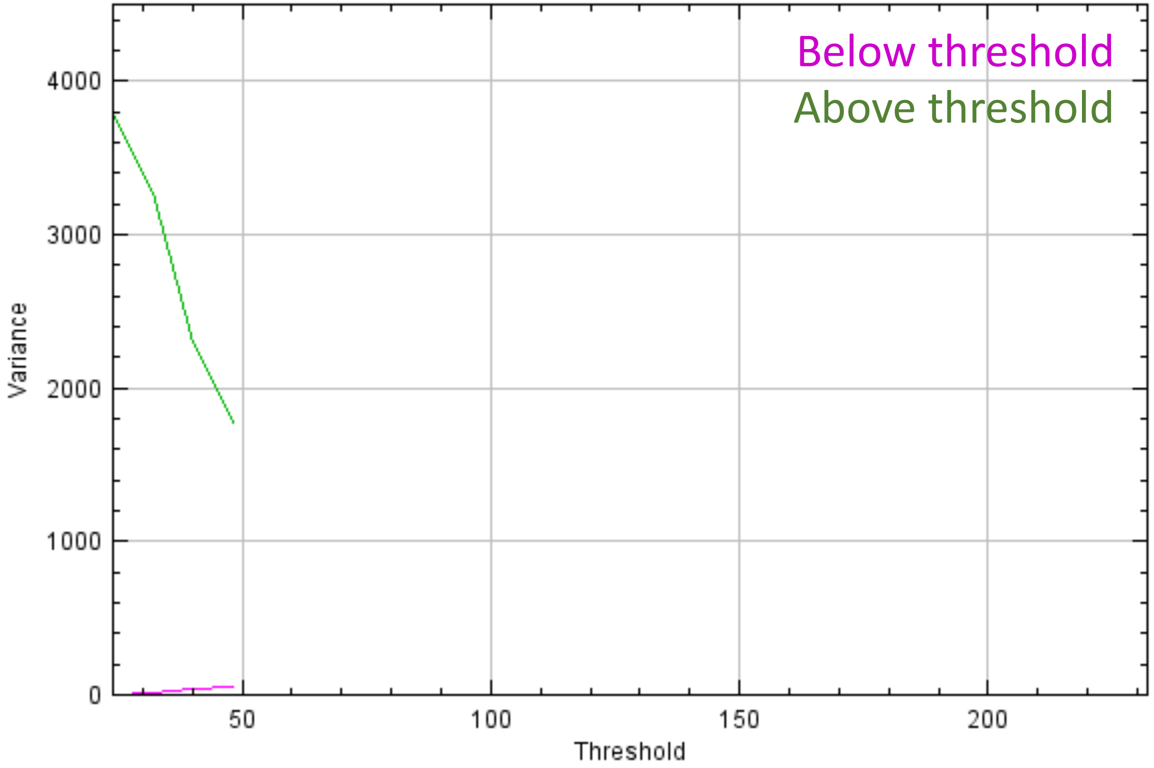
Thresholding: Otsu's method

- 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



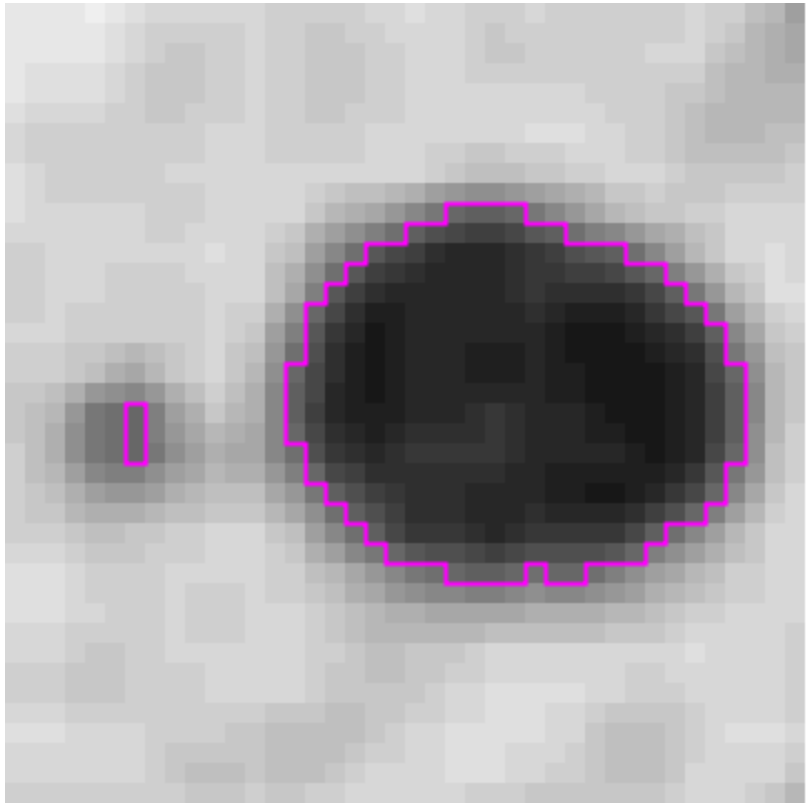


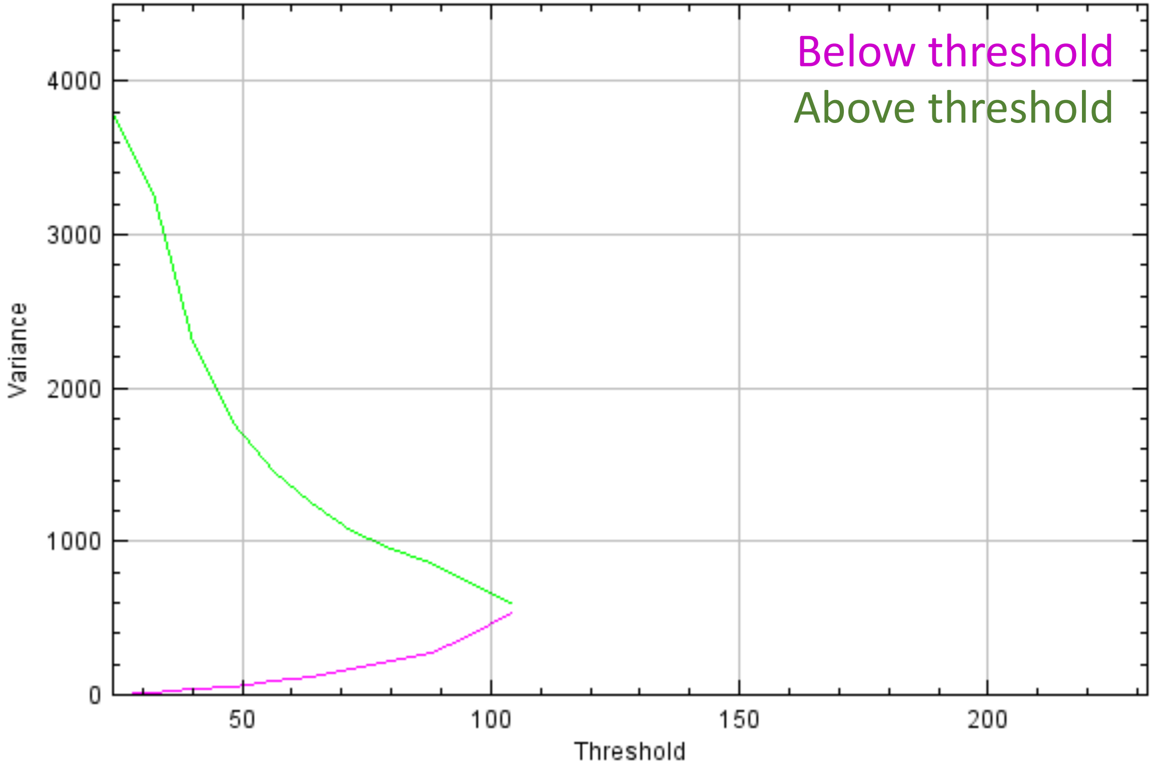
Thresholding: Otsu's method

- 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



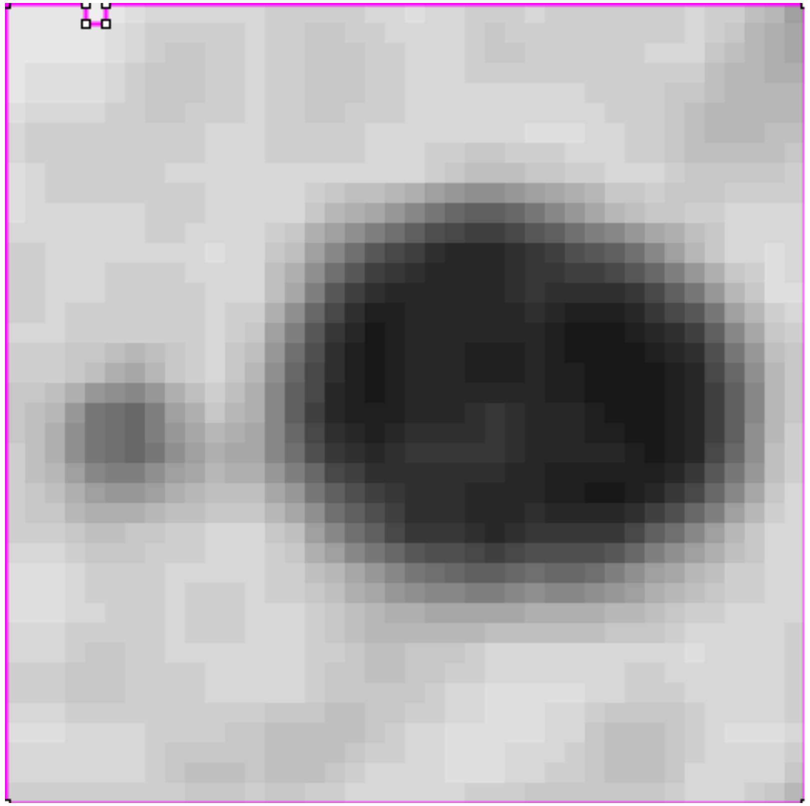


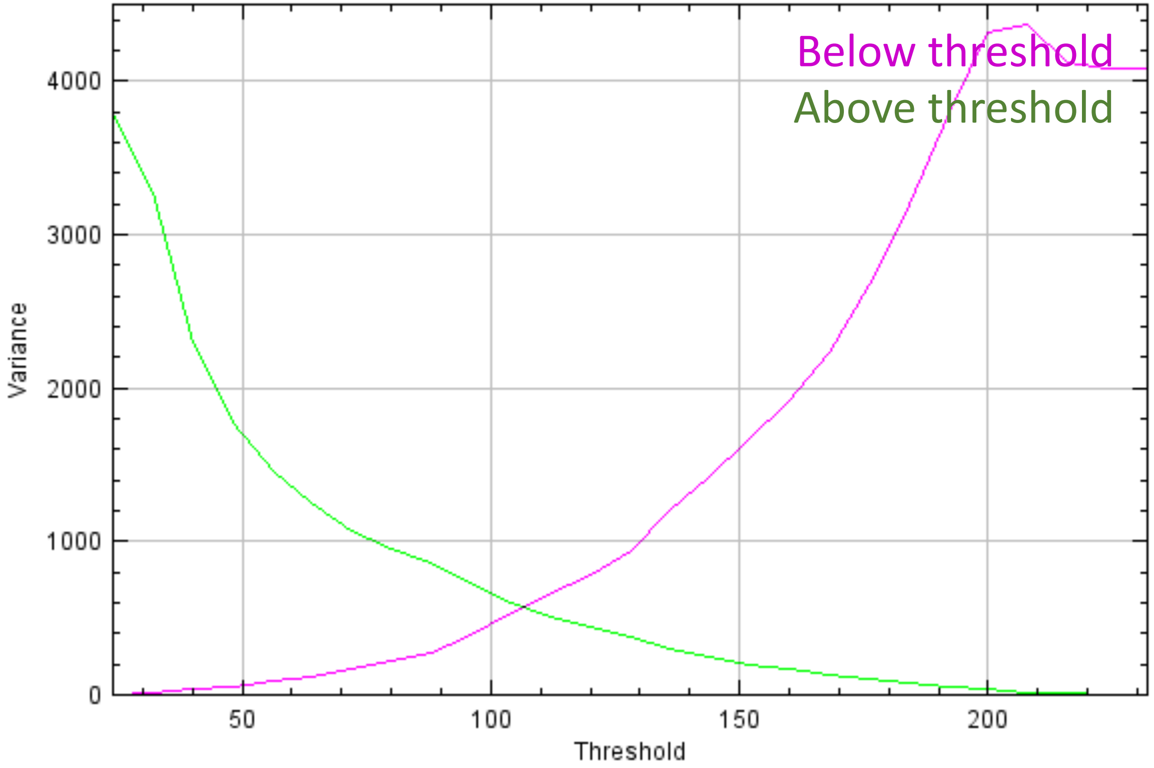
Thresholding: Otsu's method

- 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

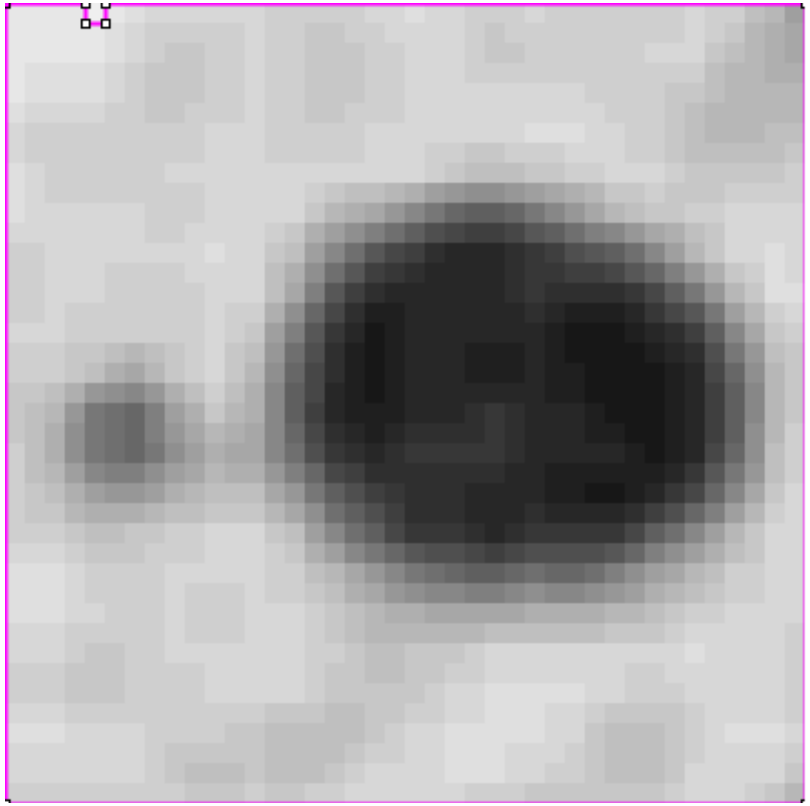


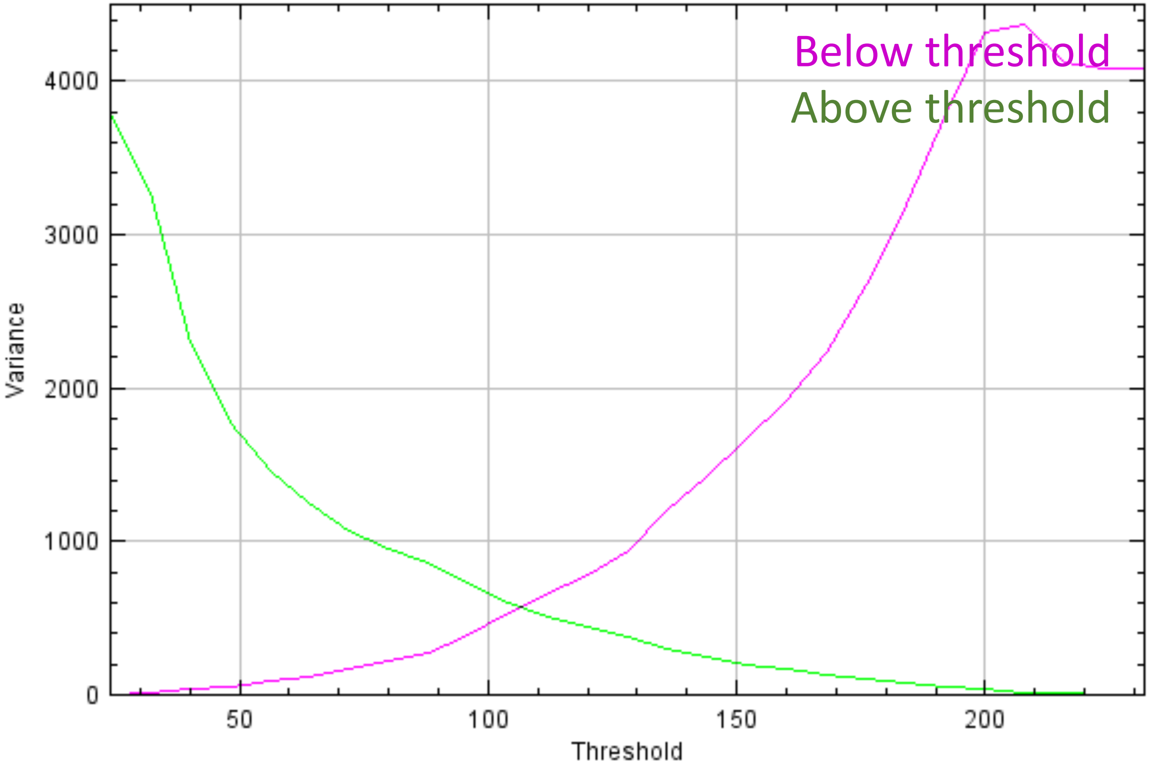


Thresholding: Otsu's method

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

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

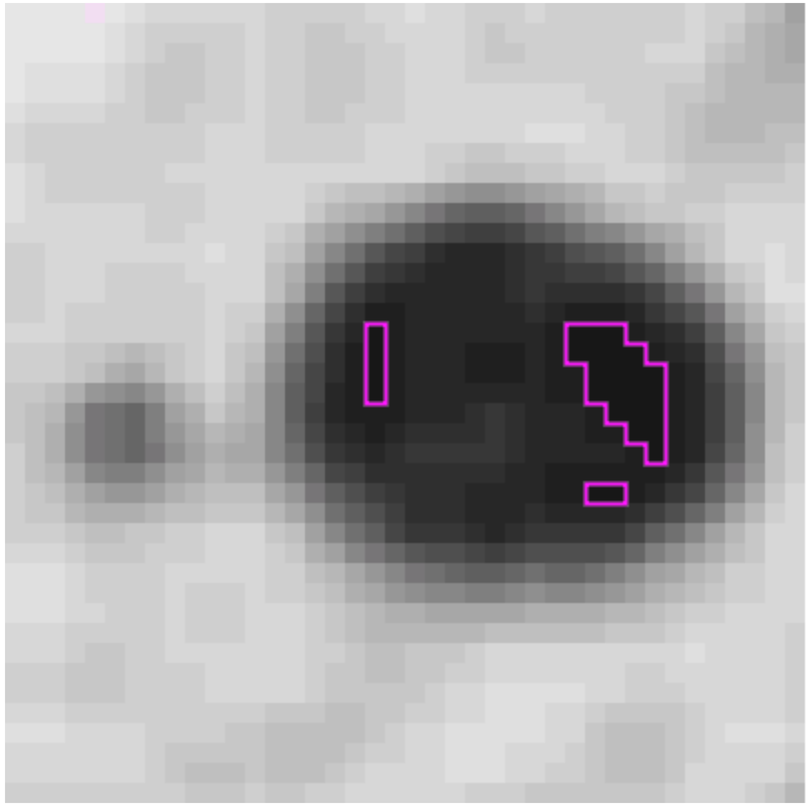


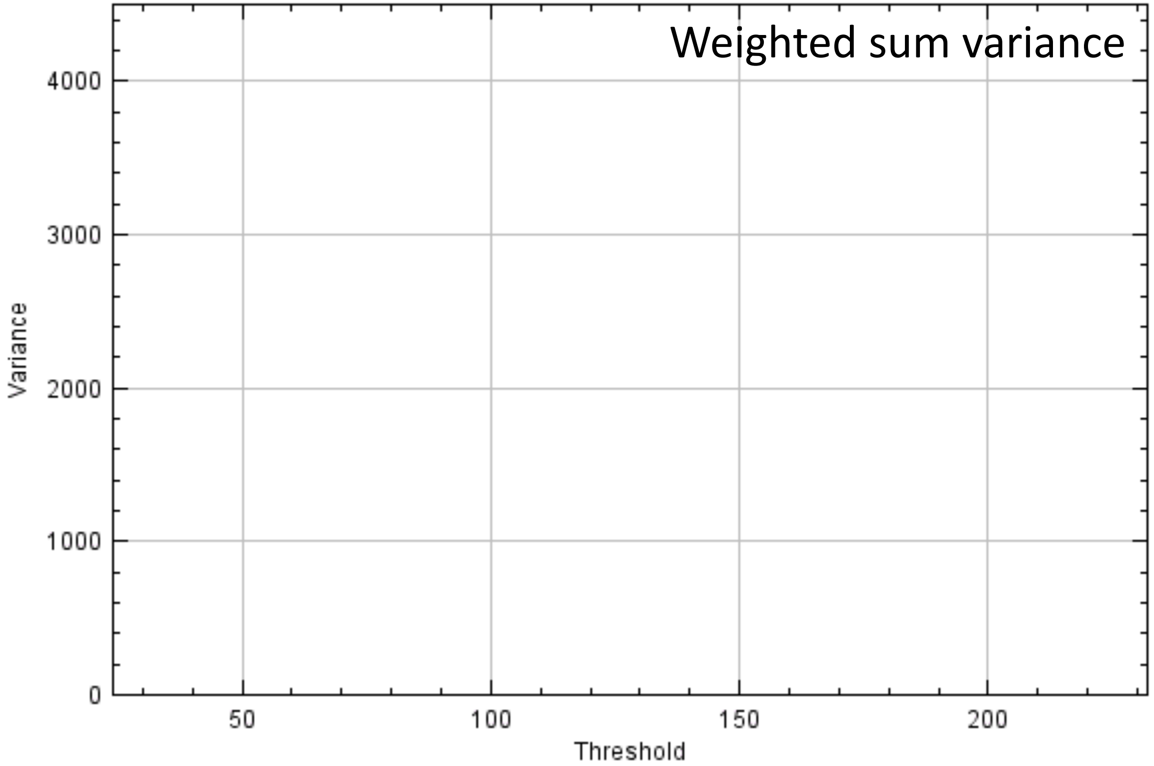


Thresholding: Otsu's method

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

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

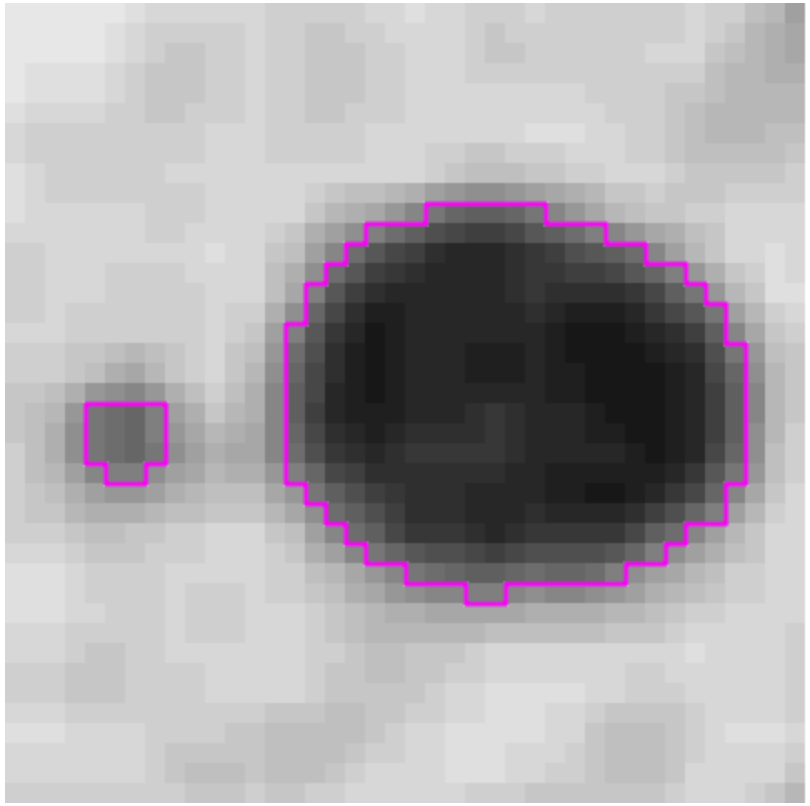


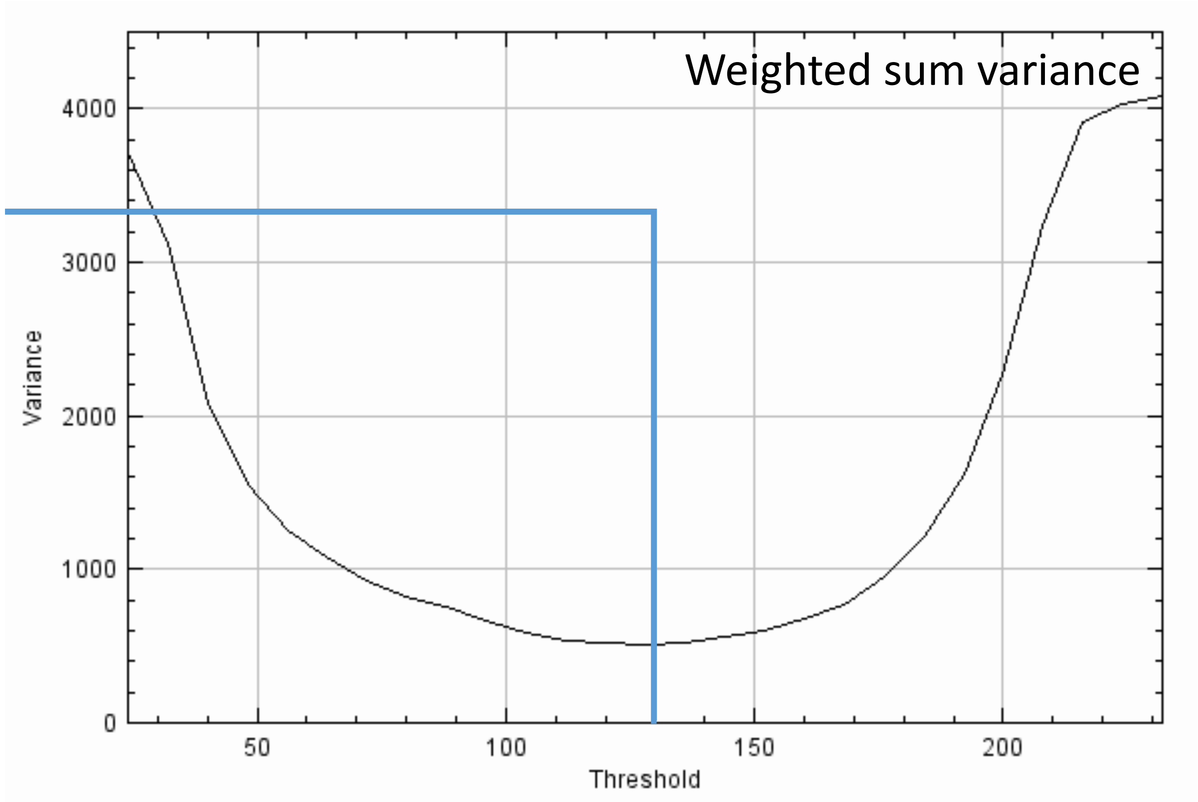


Thresholding: Otsu's method

- 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 Otsu's thresholding method (Otsu et Al. 1979) implemented in scikit-image (van der Walt et al. 2014)."

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 first-order cumulative moments of the gray-level histogram. It is straightforward to extend the method to multithreshold problems. Several experimental results are also presented to support the validity of the method.

PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Instance segmentation

Till Korten

With material from

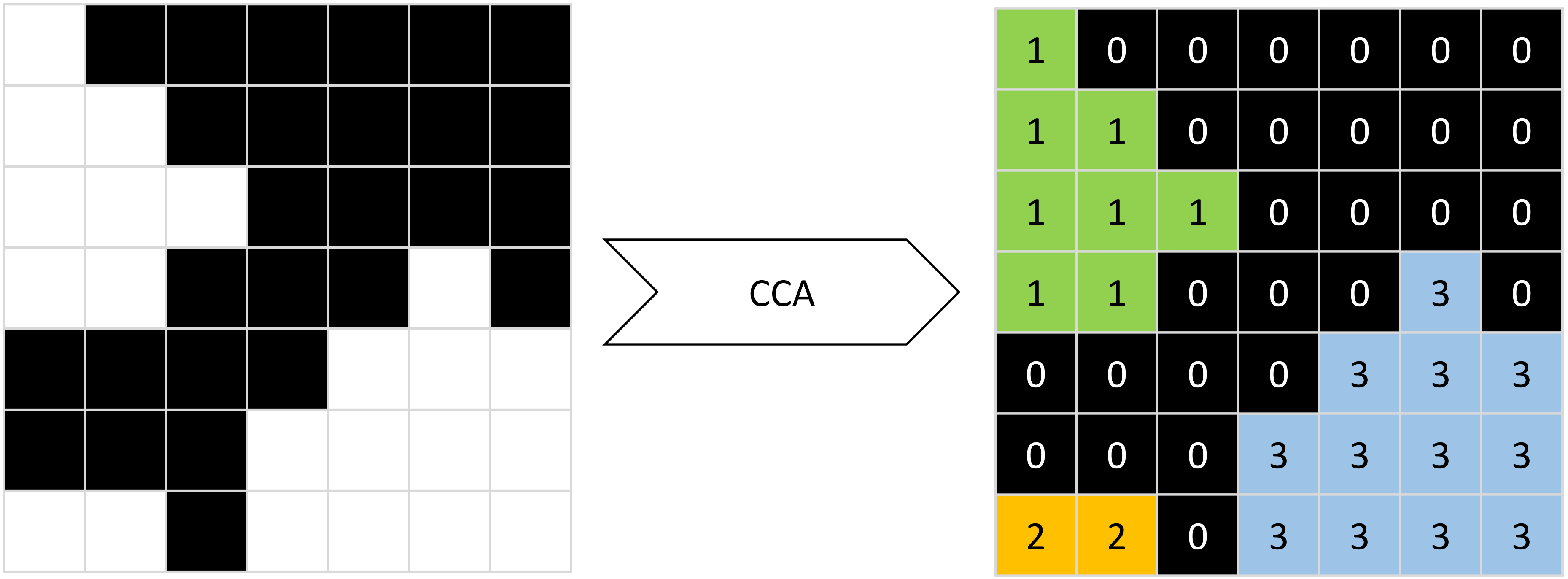
Robert Haase, PoL, TU Dresden

September 2023

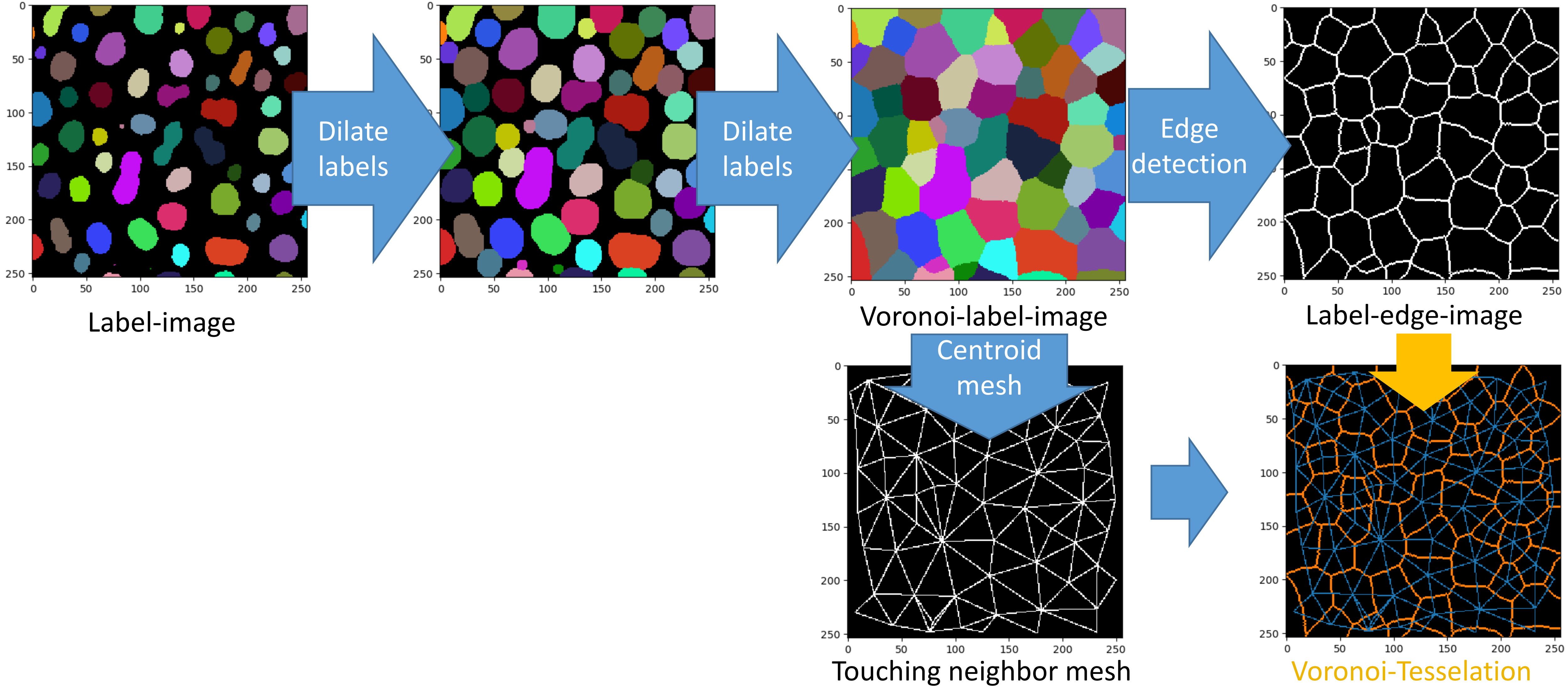
@TillKorten

Connected component labelling

- In order to allow the computer differentiating objects, connected component 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.



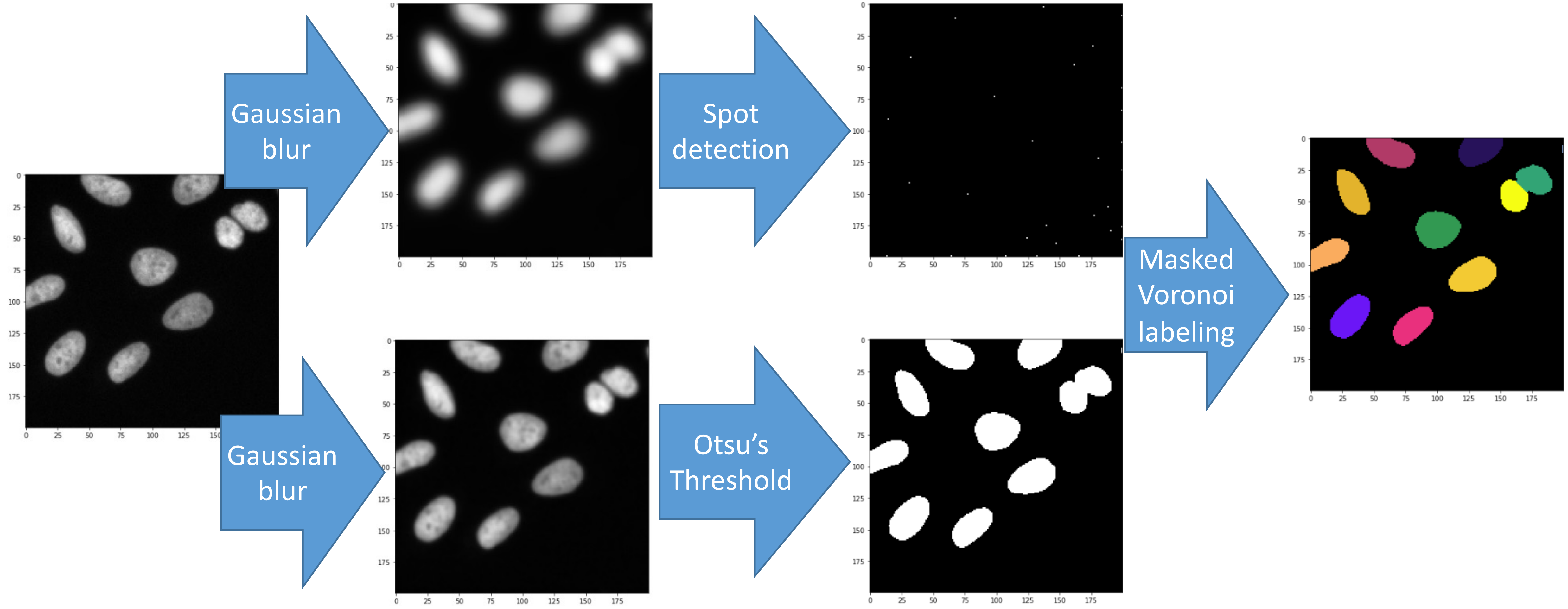
Voronoi-Tesselation



Voronoi-Otsu-Labeling

PoL Physics of Life TU Dresden

- Combination of Gaussian blur, Otsu's Threshold and Voronoi-labeling



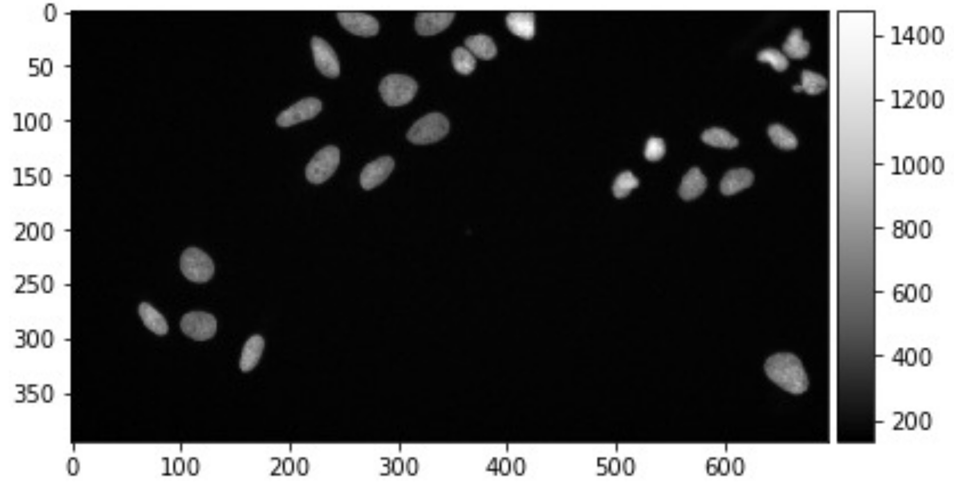
https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/20_image_segmentation/11_voronoi_otsu_labeling.html

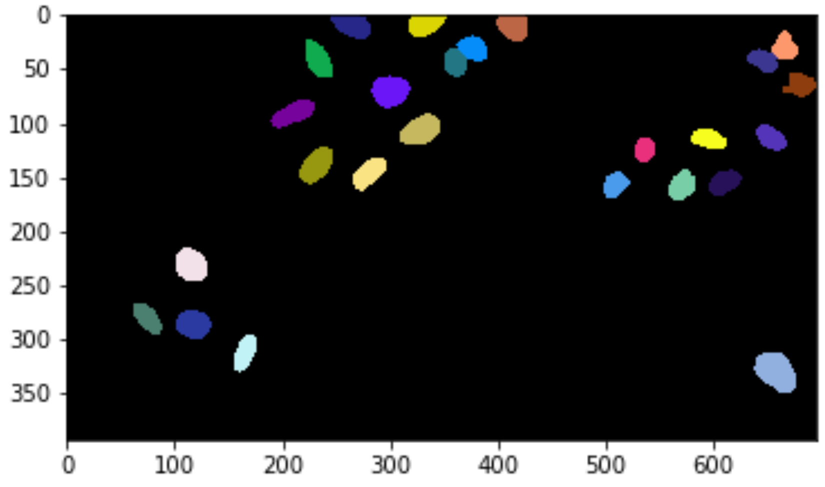
Short-cuts: Voronoi-Otsu-Labeling

- Gaussian-Blur
- Otsu-Thresholding
- Spot-detection
- Watershed on the binary image

… in a single line of code:

```
segmented = nsbatwm.voronoi_otsu_labeling(input_image,
                                           spot_sigma=5,
                                           outline_sigma=1
                                          )
segmented
```





nsbatwm made image

shape	(395, 695)
dtype	int32
size	1.0 MB
min	0
max	25