Image segmentation

Robert Haase

Using materials from

Alba Villaronga Luque and Jesse Veenvliet, MPI-CBG Dresden

Ryan Savill George, MPI CBG Dresden

Johannes Soltwedel, PoL, TU Dresden

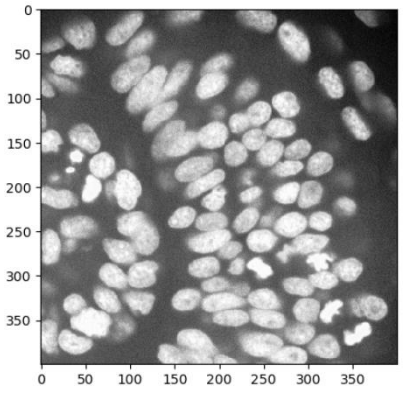
April 2023

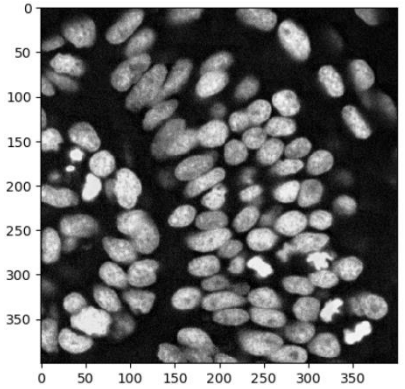
@haesleinhuepf

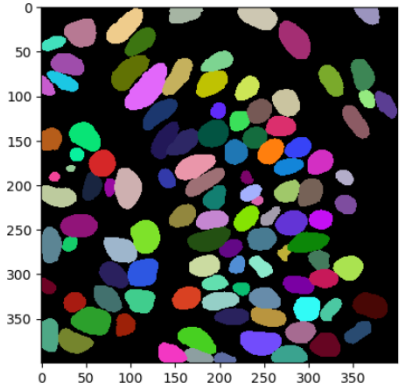
Lecture overview: Bio-image Analysis

PoL Physics of Life TU Dresden

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

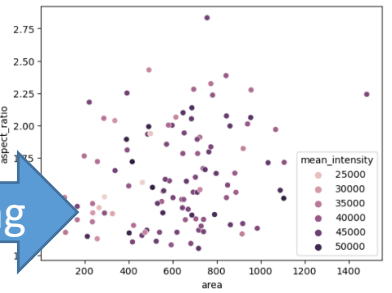






	area	mean_intensity	major_axis_length	minor_axis_length
0	594.0	40572.809764	28.611591	26.537947
1	645.0	43764.872868	33.511511	24.566916
2	1105.0	51970.561991	45.232031	31.456308
3	718.0	47015.487465	31.023274	29.5
4	791.0	49132.515803	36.382253	27.7
...	...	...	...	
105	238.0	30477.126050	20.252197	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		

Plotting



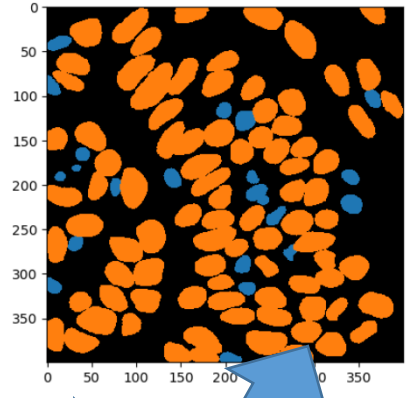


Image filtering

Image segmentation

Feature extraction

Object classification

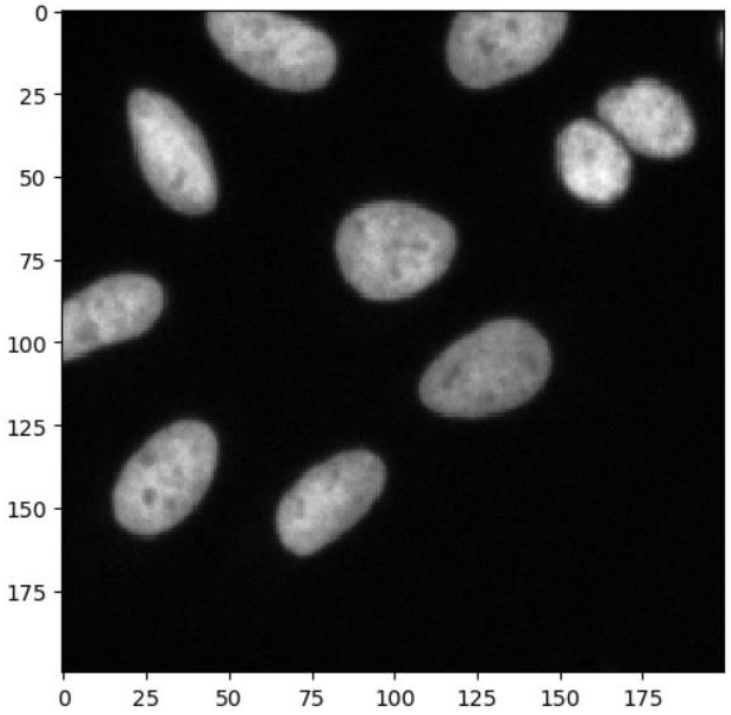
Visualization

Bio-image analysis

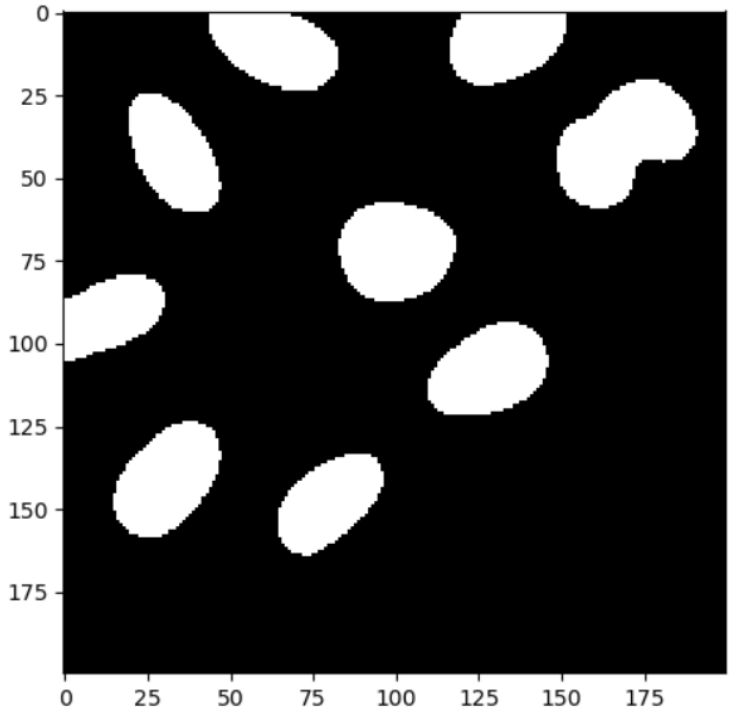
Bio-statistics

Terminology

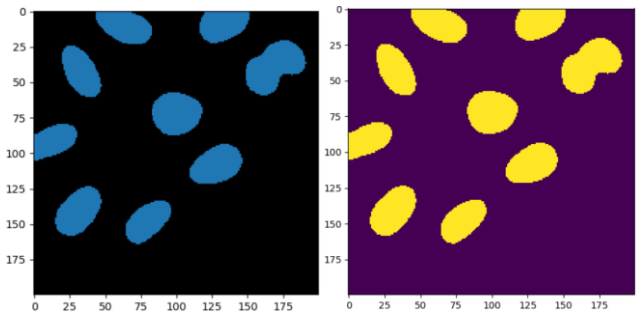
Intensity image



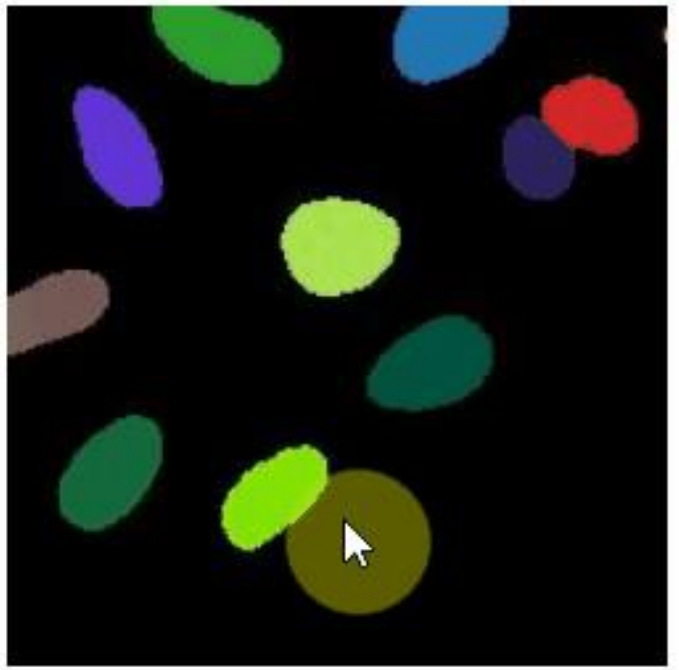
Binary image



No matter how they are displayed



Label image



[y=152, x=92] = 0

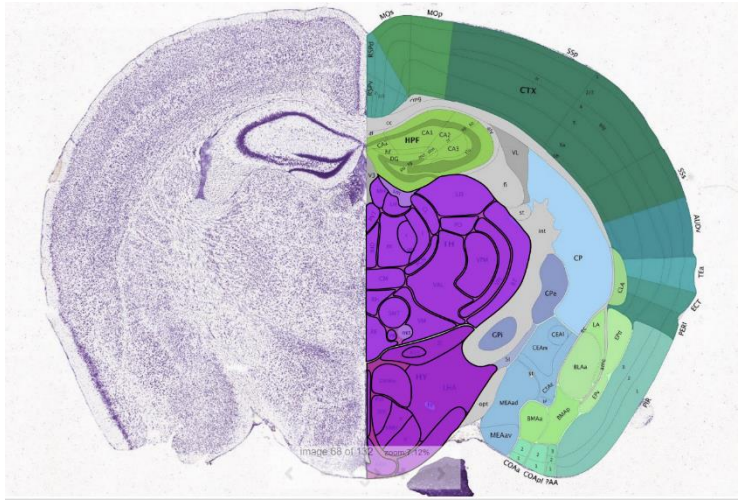
Terminology

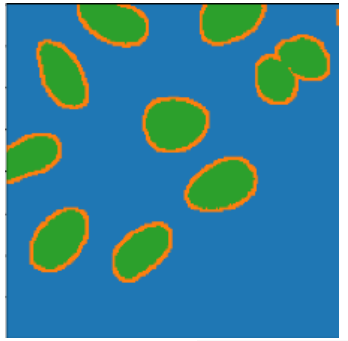
Instance segmentation

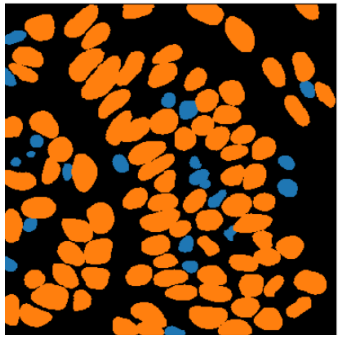
Instances:

- Cells, nuclei, cats, dogs, cars, trees

Semantic segmentation







Regions:

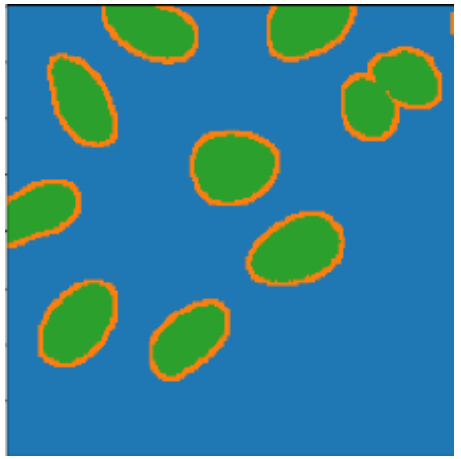
- Anatomical, geographical
- All pixels belonging to the same type of object have the same value

Source: Allen Mouse Brain Atlas [dataset]. Available from mouse.brain-map.org. Allen Institute for Brain Science (2011).

Terminology

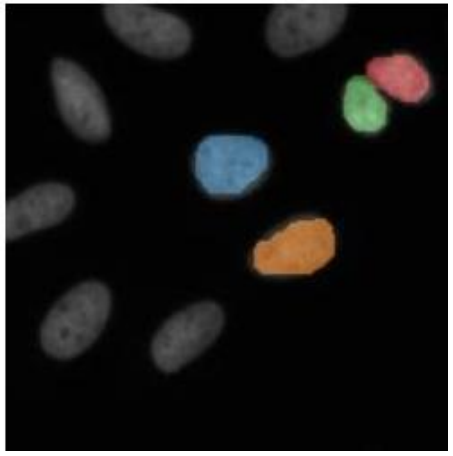
- Annotations are typically drawn by humans (e.g. to train machine learning models)

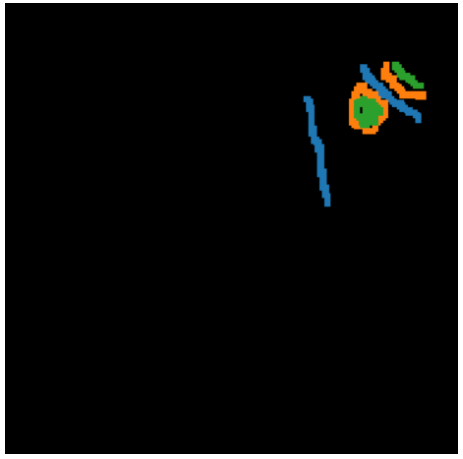
Instance segmentation



Semantic segmentation

Sparse instance annotation

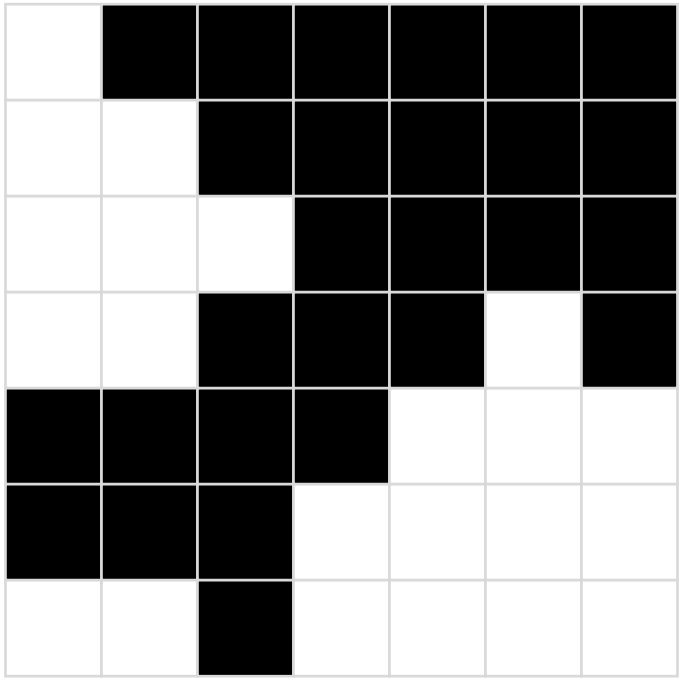




Sparse semantic annotation

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.



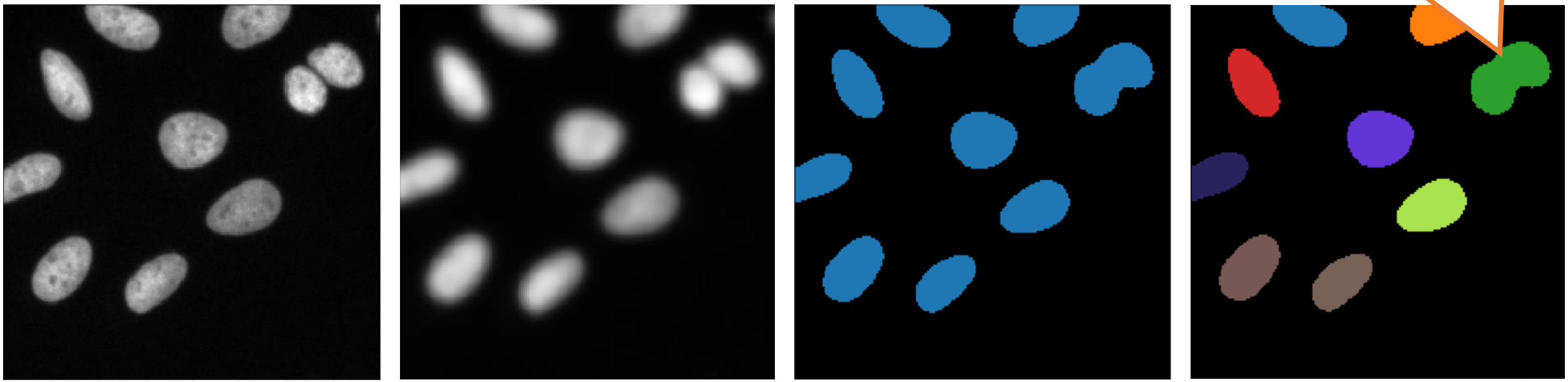
CCA

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

Common image segmentation workflows

- Presumably the most common segmentation algorithm used for fluorescent microscopy images:
  - Gaussian blur, Otsu's Threshold, Connected Component Labeling

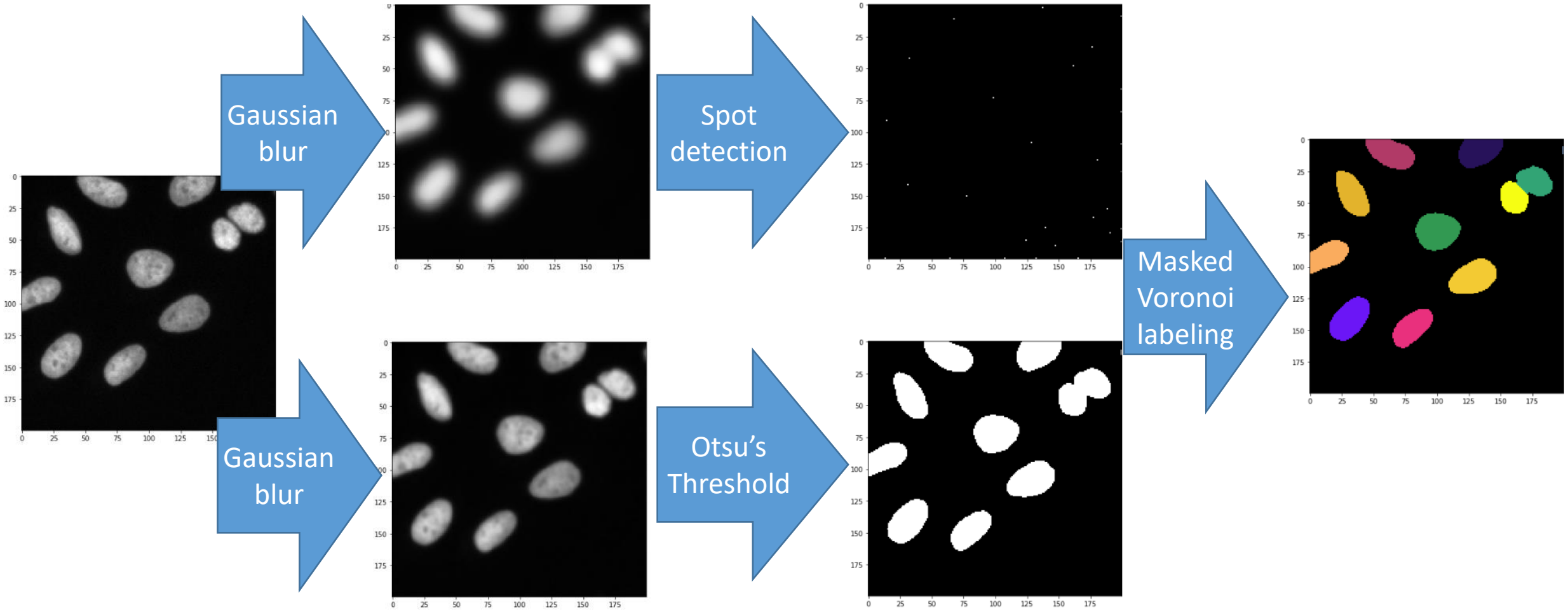
Limitation: Dense objects



Denoising → Binarization → Labeling

Common image segmentation workflows

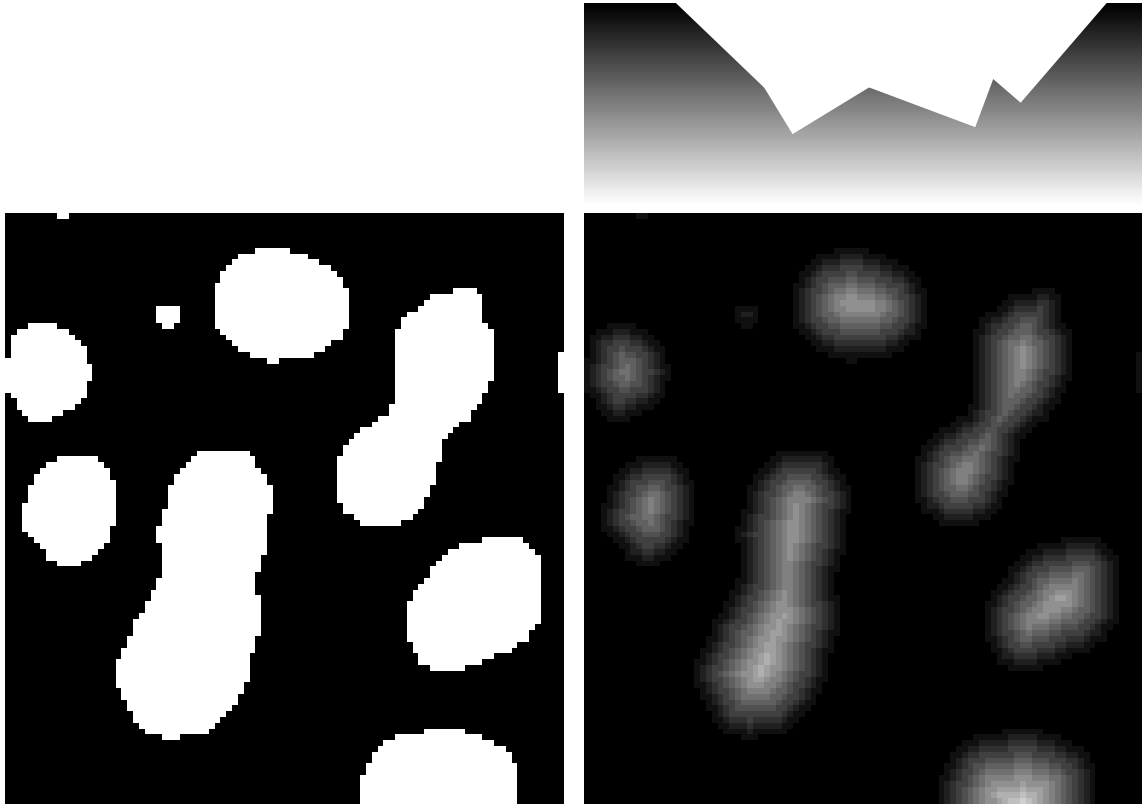
- Combination of Gaussian blur, Otsu's Threshold and Voronoi-labeling



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

Watershed

- The watershed algorithm for binary images allows cutting one object into tow where it’s reasonable.

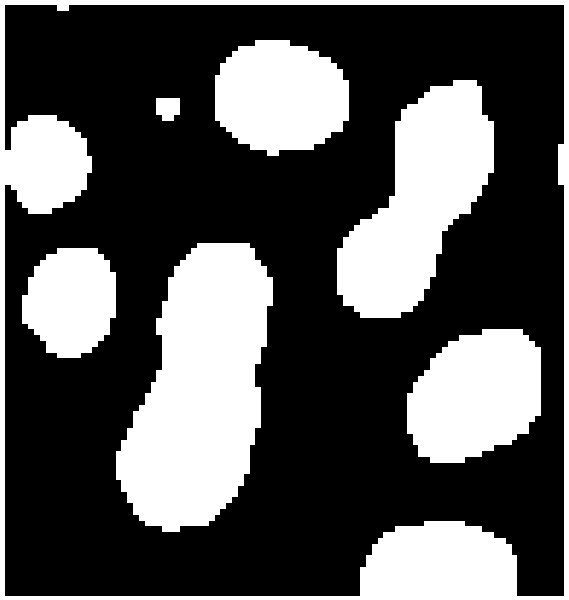


Binary segmentation

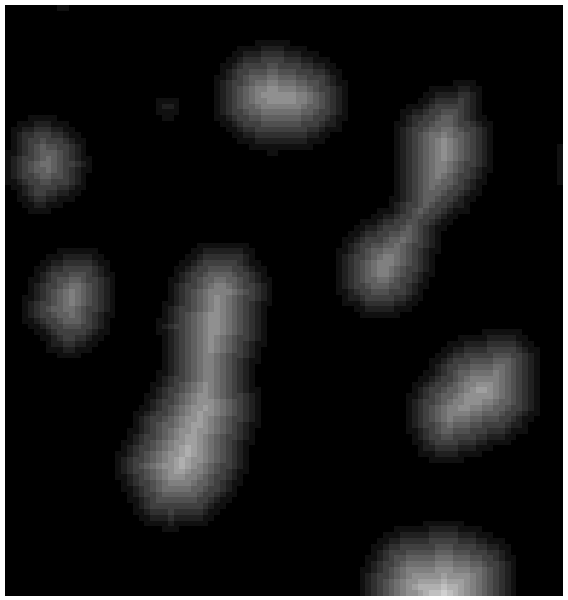
Distance map

Watershed

- The watershed algorithm for binary images allows cutting one object into tow where it’s reasonable.



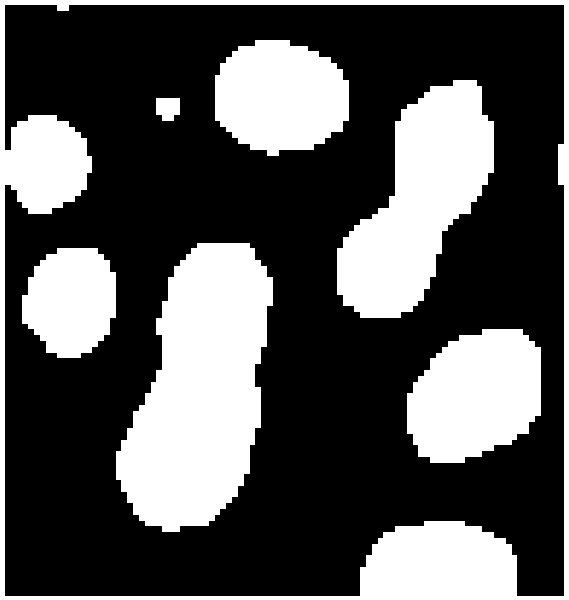
Binary segmentation



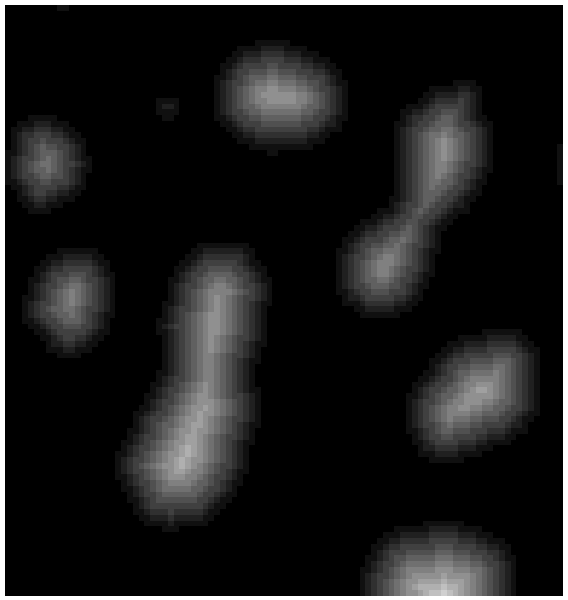
Distance map

Watershed

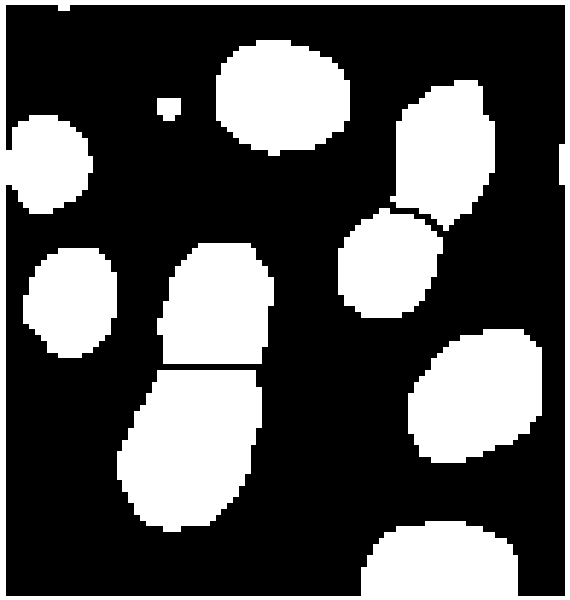
- The watershed algorithm for binary images allows cutting one object into two where it’s reasonable.
- The watersheds are made from binary images. The algorithm does not take the original image into account!



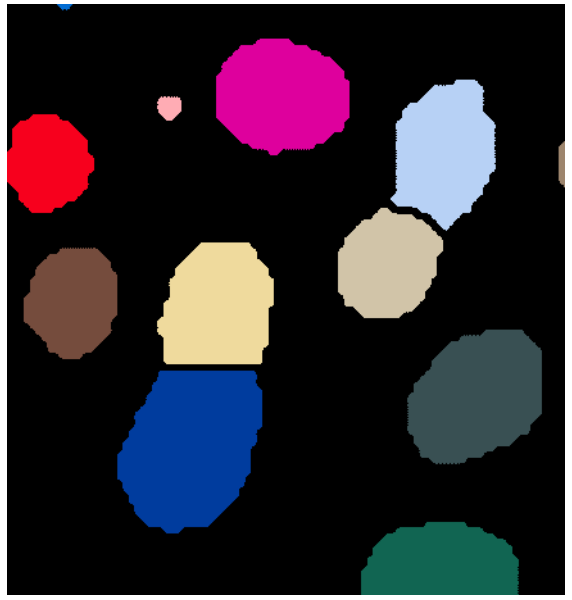
Binary segmentation



Distance map



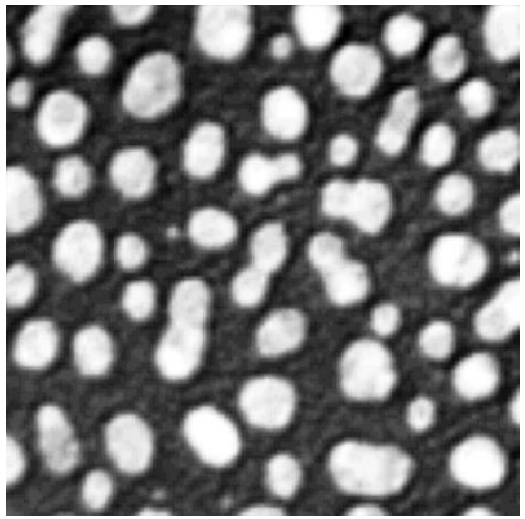
Binary watershed

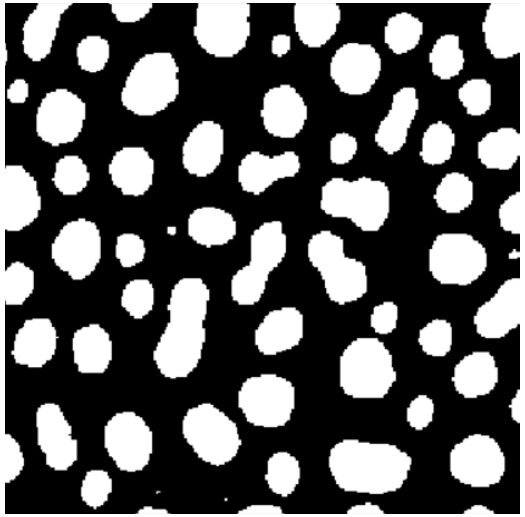


Labeled watershed

Watershed use-cases

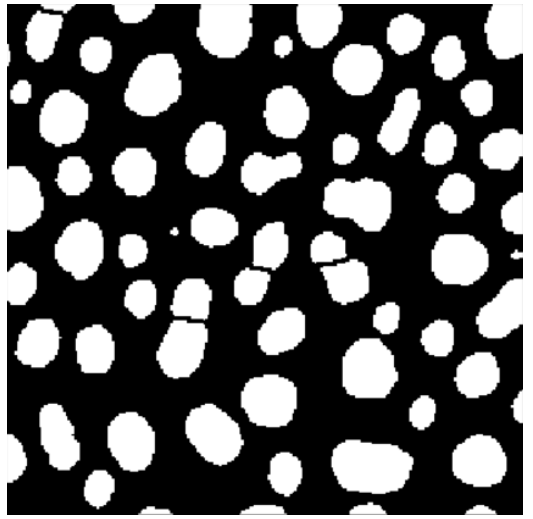
- Split dense objects

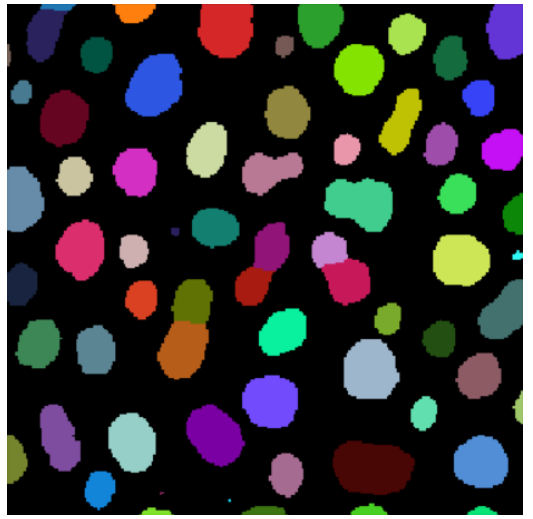




Binarization

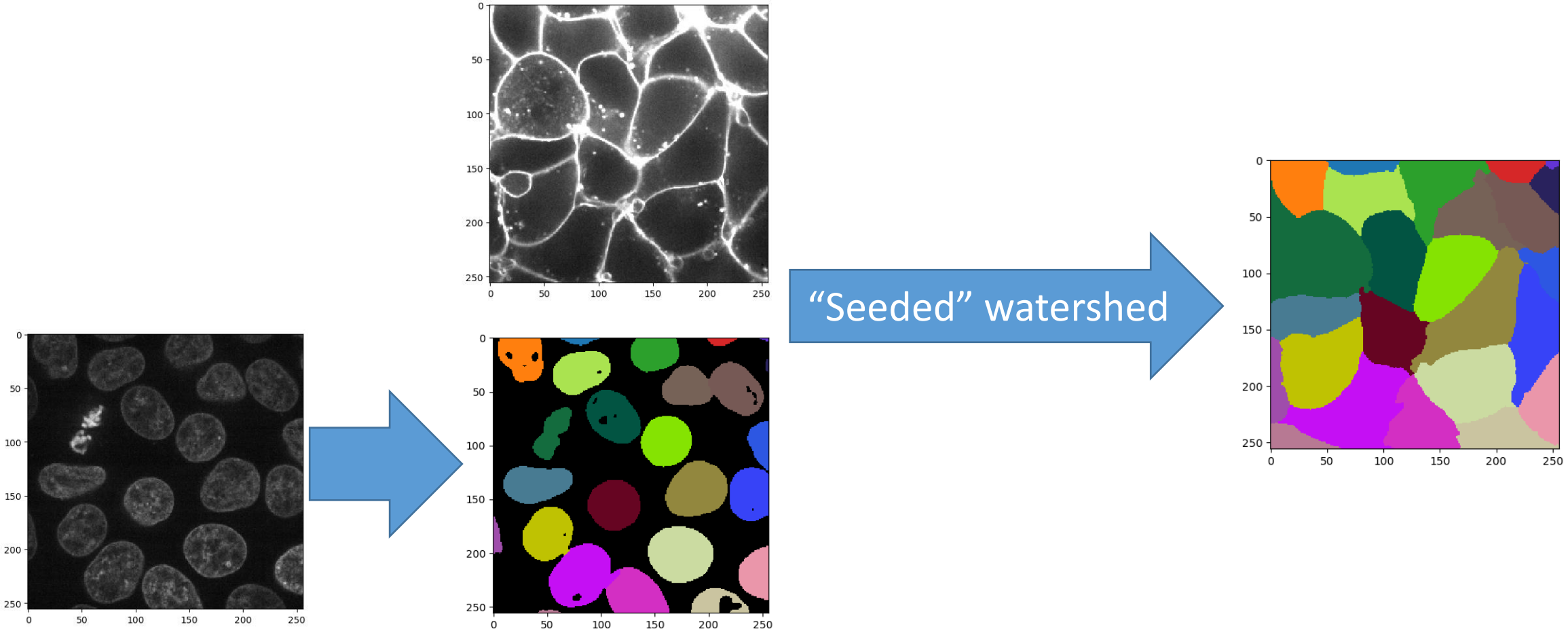
Watershed





Watershed use-cases

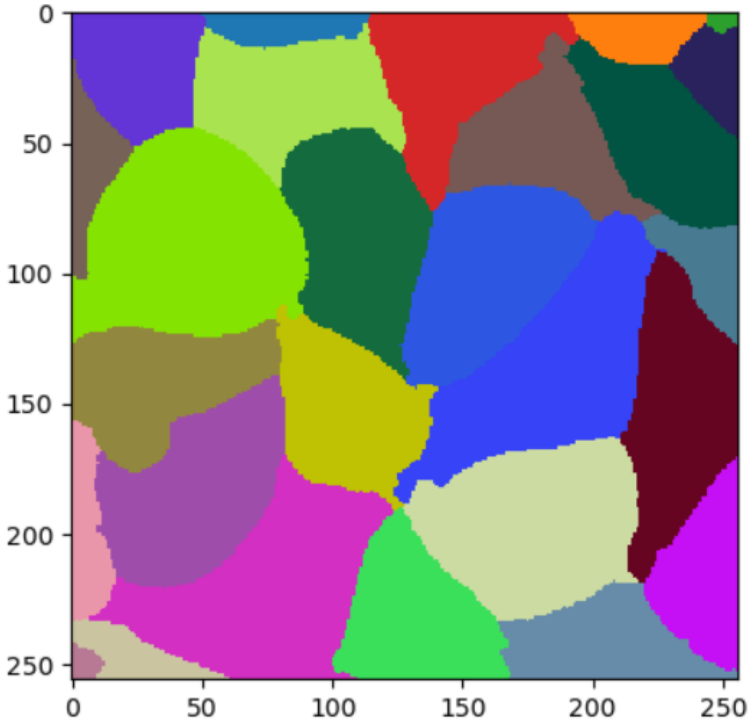
- Seeded watershed: Flood regions from pre-defined seeds
- Example: Flood cells from nuclei positions



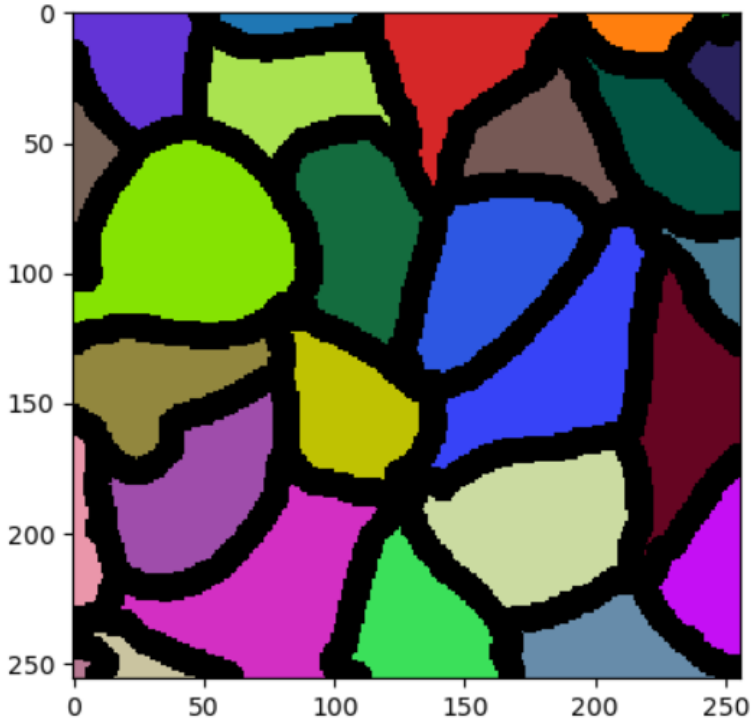
Label post-processing / morphological operations

PoL Physics of Life TU Dresden

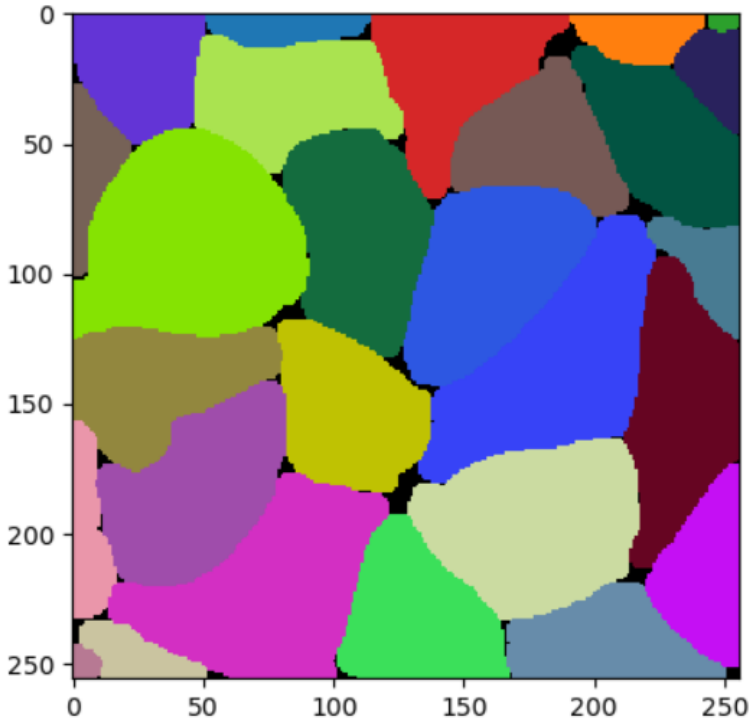
- … similar to morphological operations on binary images



Original



Eroding labels

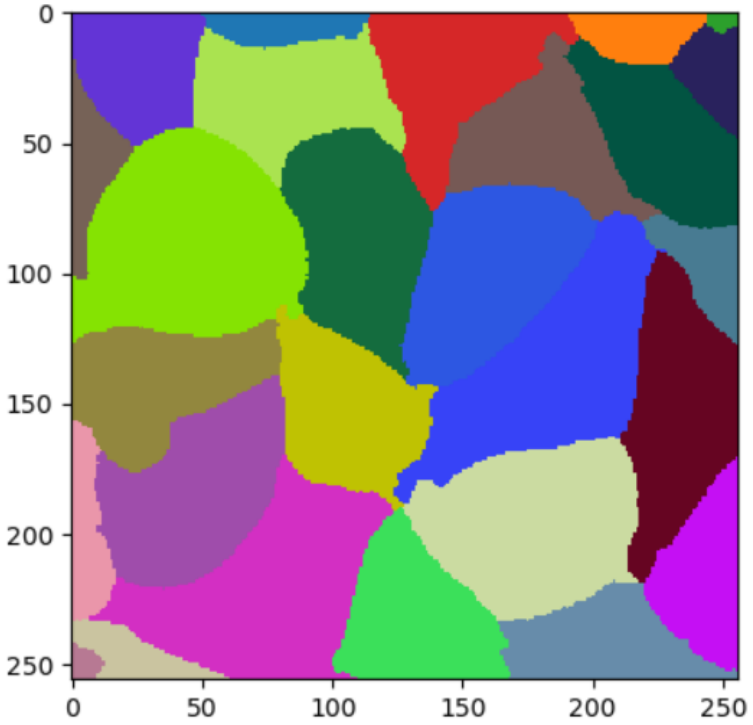


Dilating Labels

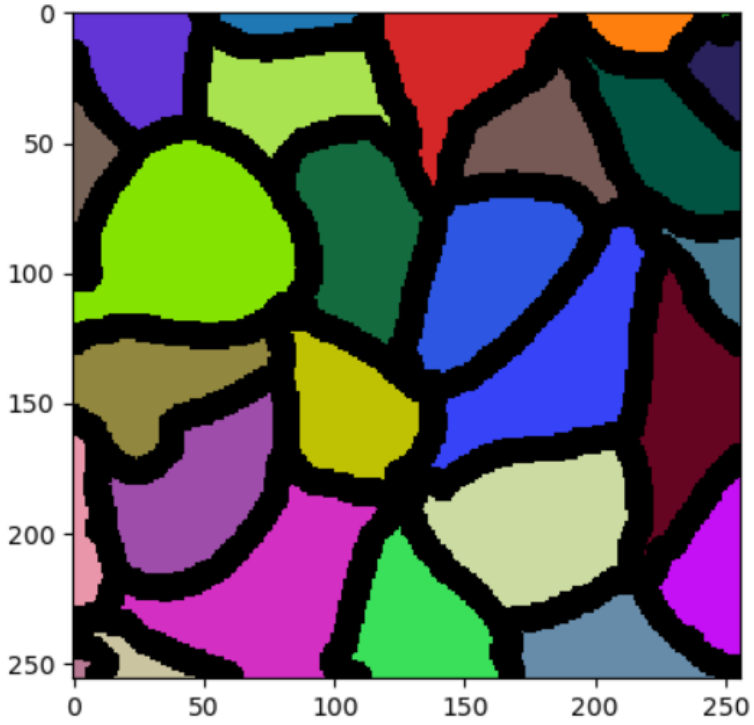
Label post-processing / morphological operations

PoL Physics of Life TU Dresden

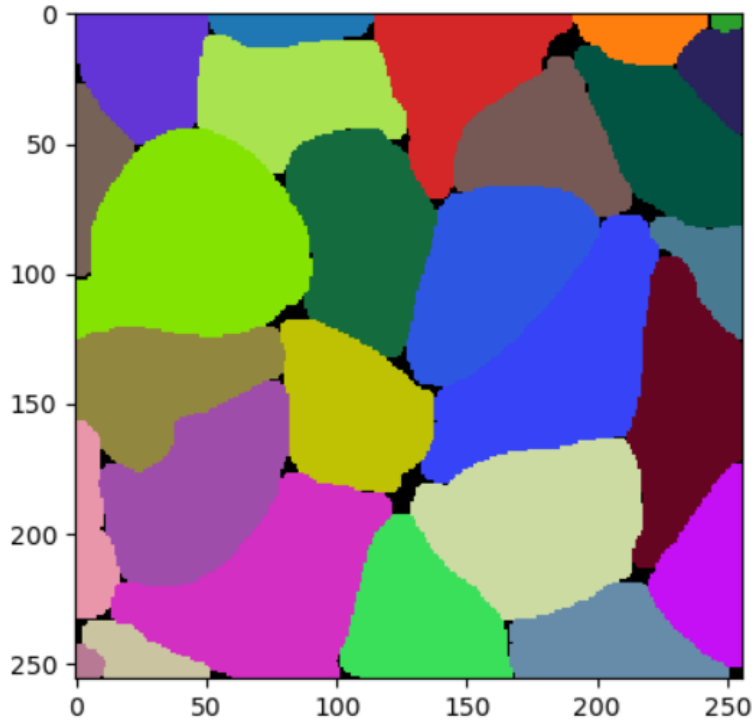
- … similar to morphological operations on binary images



Original



Eroding labels



Dilating Labels

This combination is called

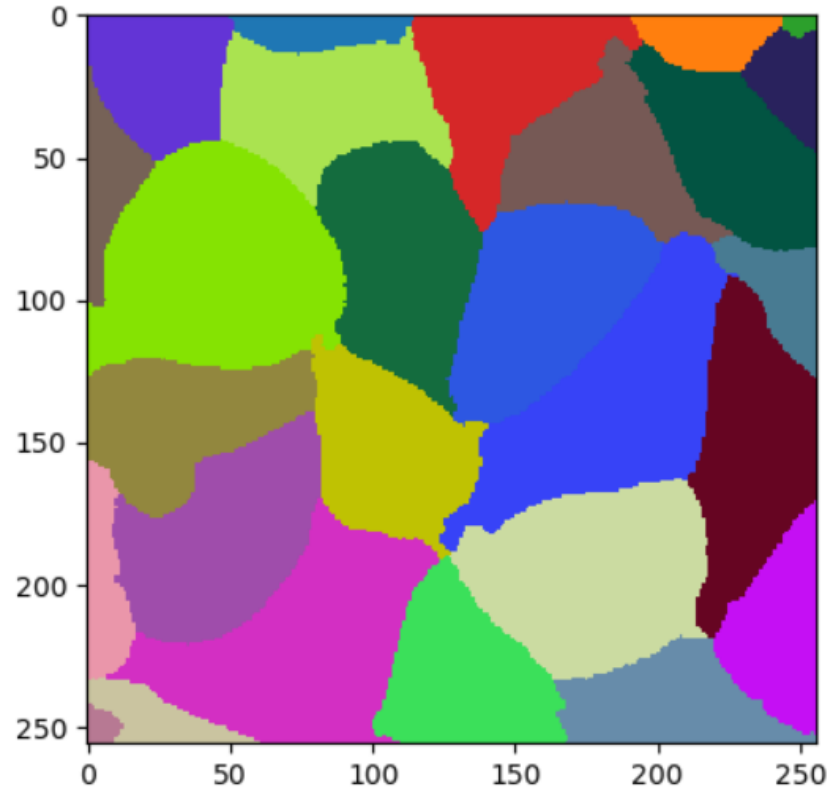
Opening **Closing**

Epilepsy warning

Label post-processing / morphological operations

PoL Physics of Life TU Dresden

- … similar to morphological operations on binary images



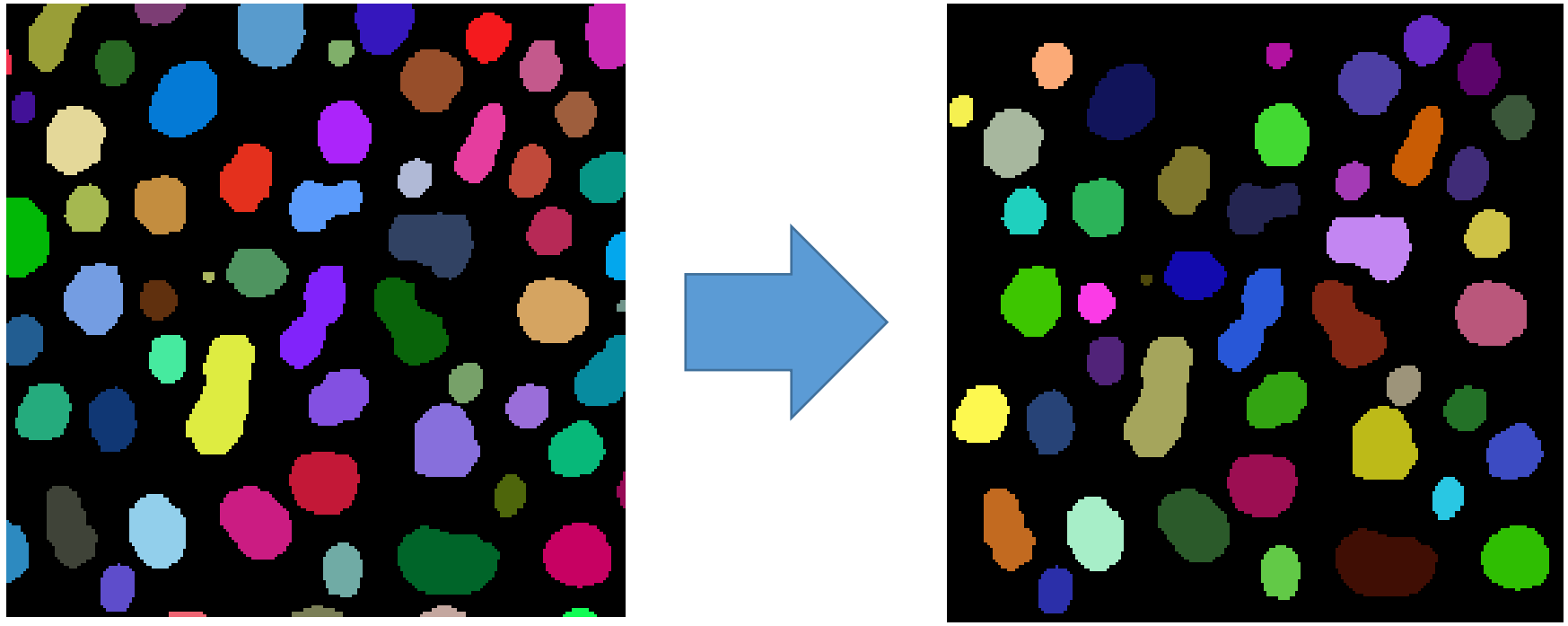
Original

Opening labels

Smoothing Labels

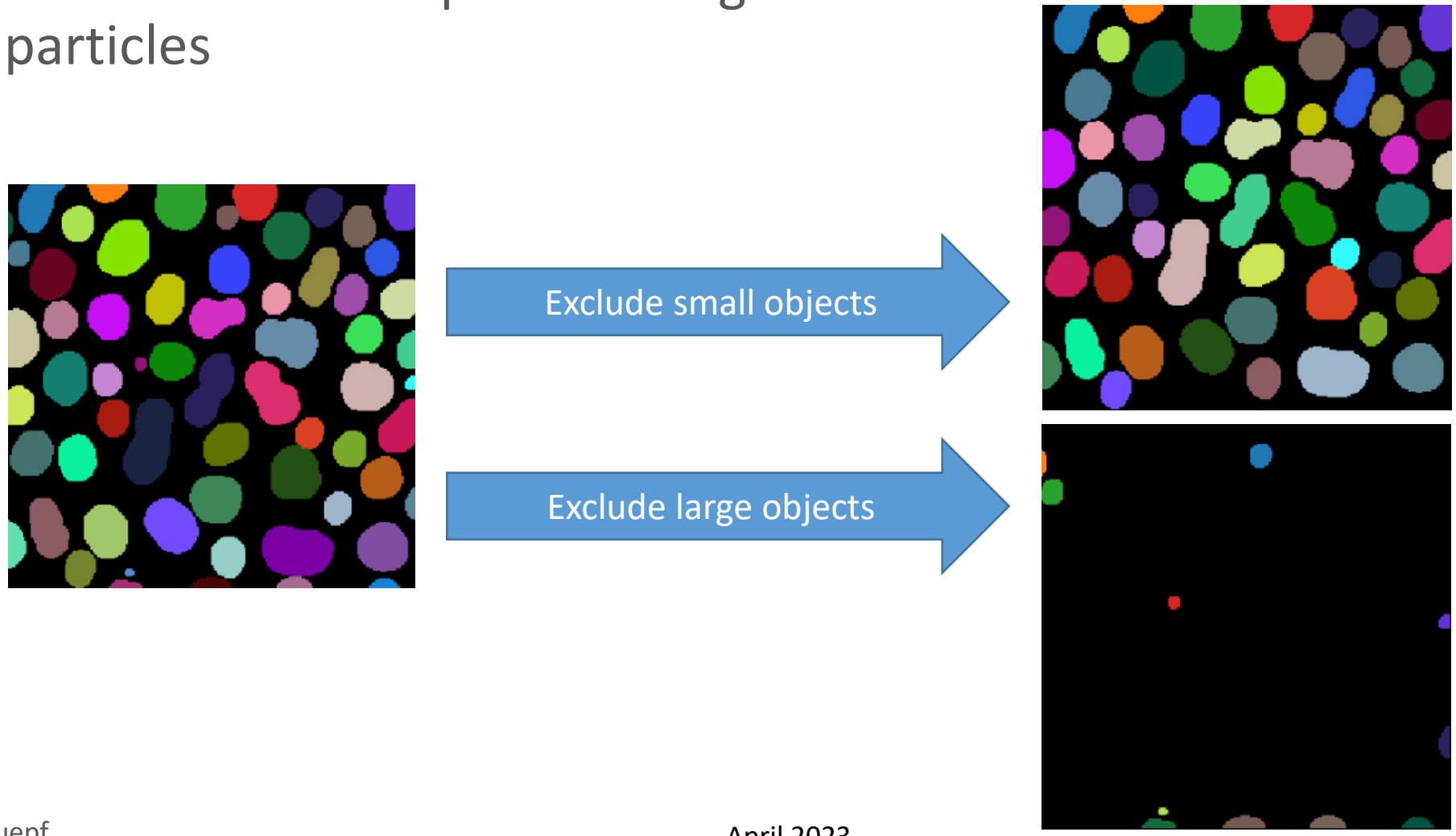
Label post-processing

- Remove objects at the image border
- Their measurements (shape, size) would be misleading anyway



Label post-processing

- Excluding small / large objects
- Common correction-step in case segmentations contain noise-related small particles



PoL Physics of Life TU Dresden

CENTER FOR SYSTEMS BIOLOGY DRESDEN

Surface reconstruction

Robert Haase

Using materials from

Alba Villaronga Luque and Jesse Veenvliet, MPI-CBG Dresden

Johannes Soltwedel, PoL, TU Dresden

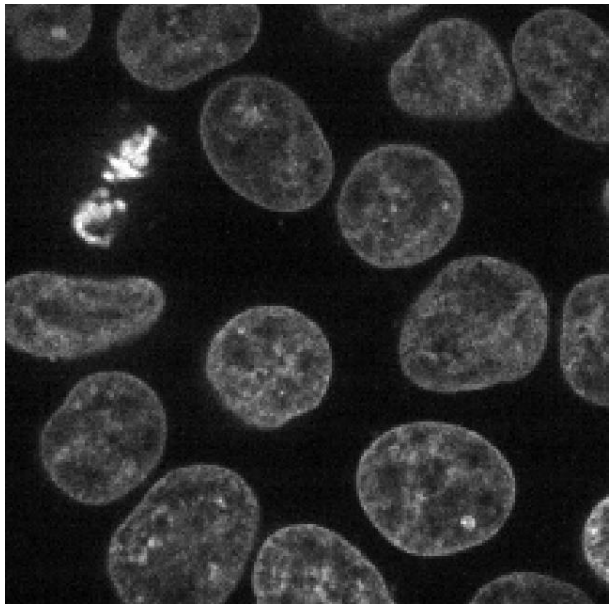
April 2023

@haesleinhuepf

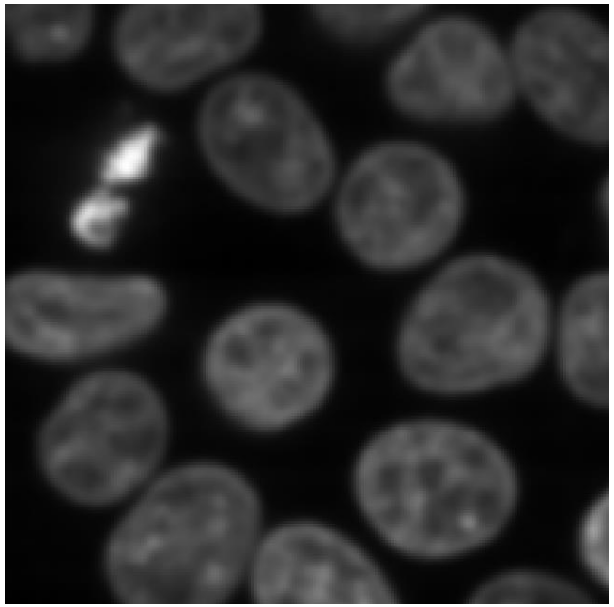
Motivation: Surface reconstruction

PoL Physics of Life TU Dresden

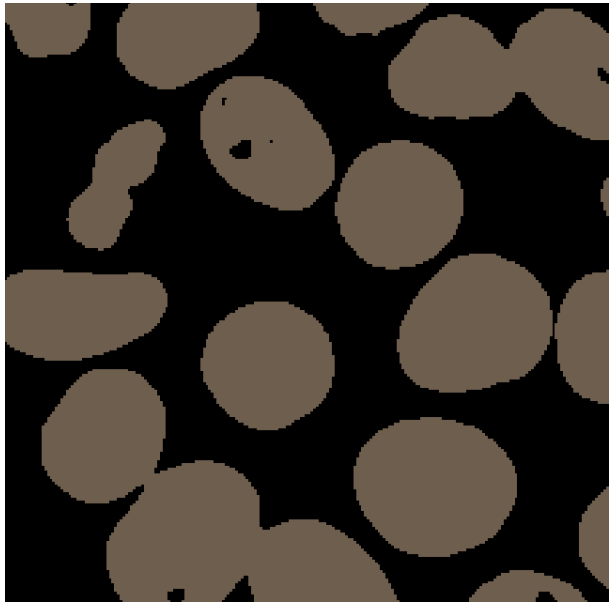
- Pixel and voxel arrays can be huge in memory
- Processing 3D arrays is time-consuming



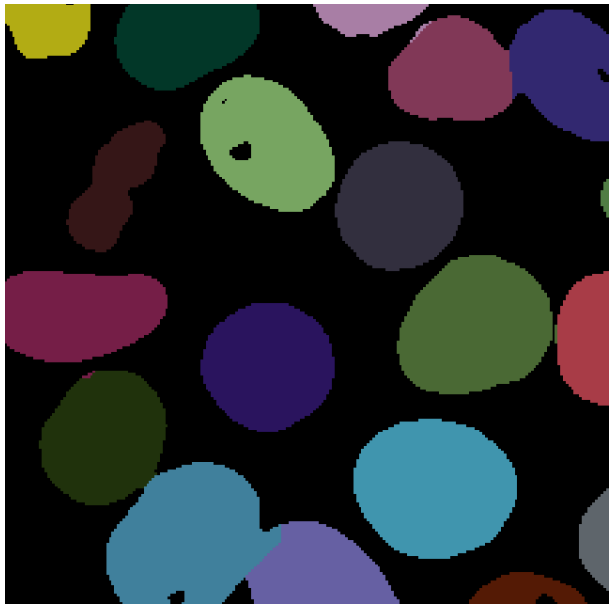
1024 x1024 x 100
16-bit image



1024 x1024 x 100
16-bit image



1024 x1024 x 100
8-bit image



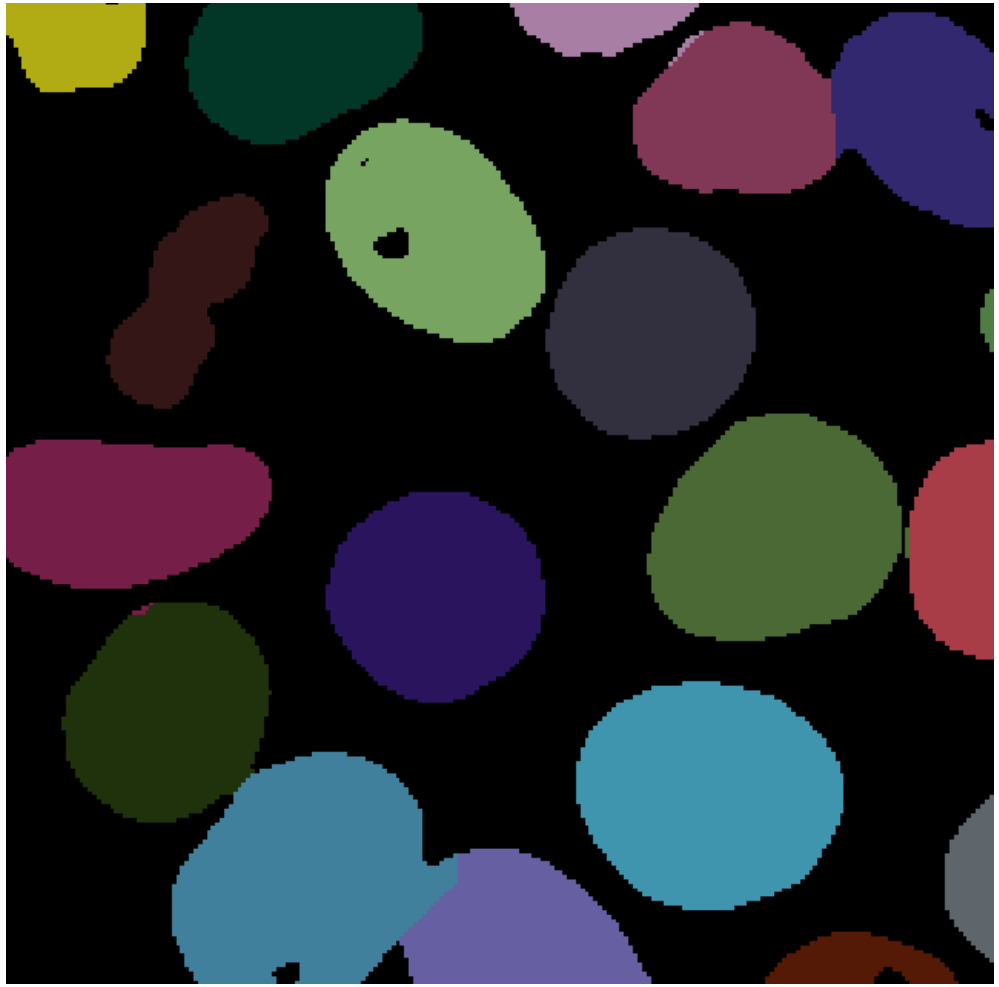
1024 x1024 x 100
16-bit image

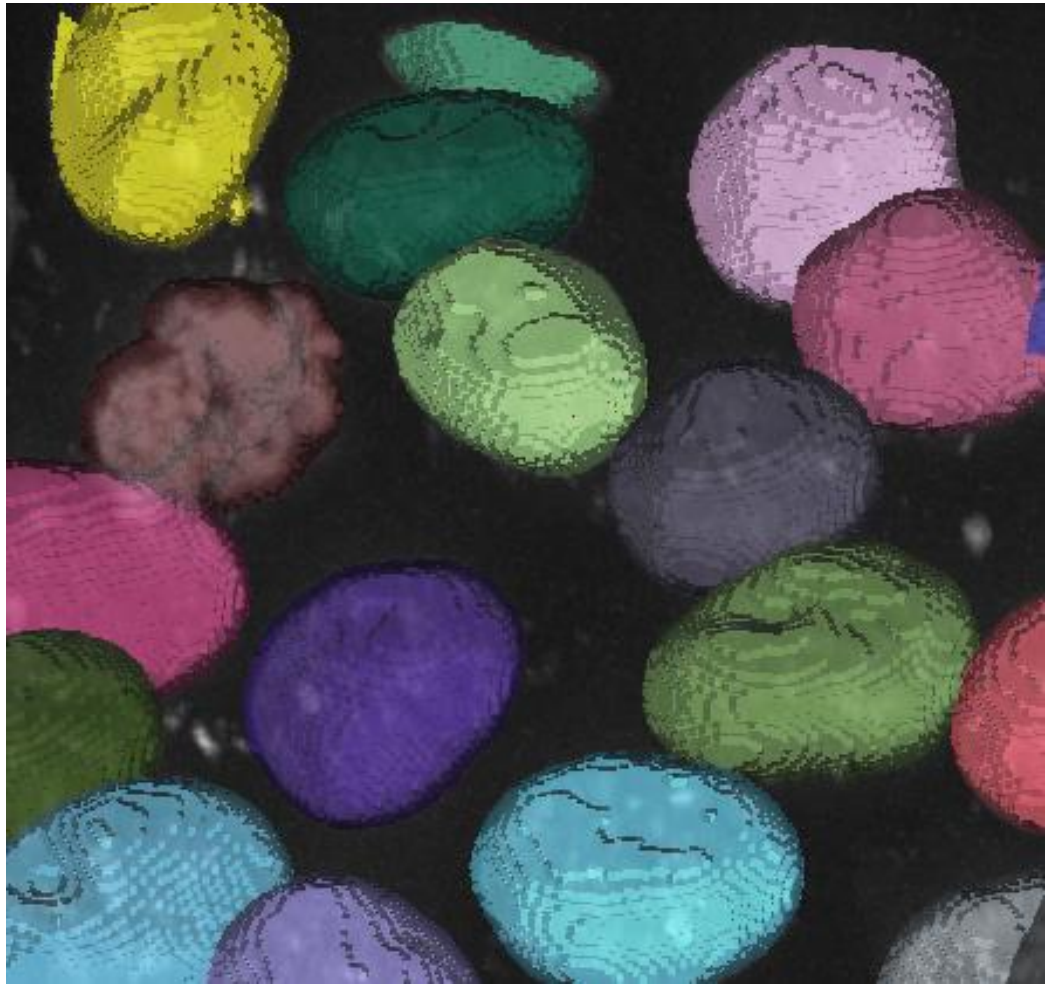
How much memory does this workflow cost?

700 MB	400 MB	4 GB	7 GB

Motivation: Surface reconstruction

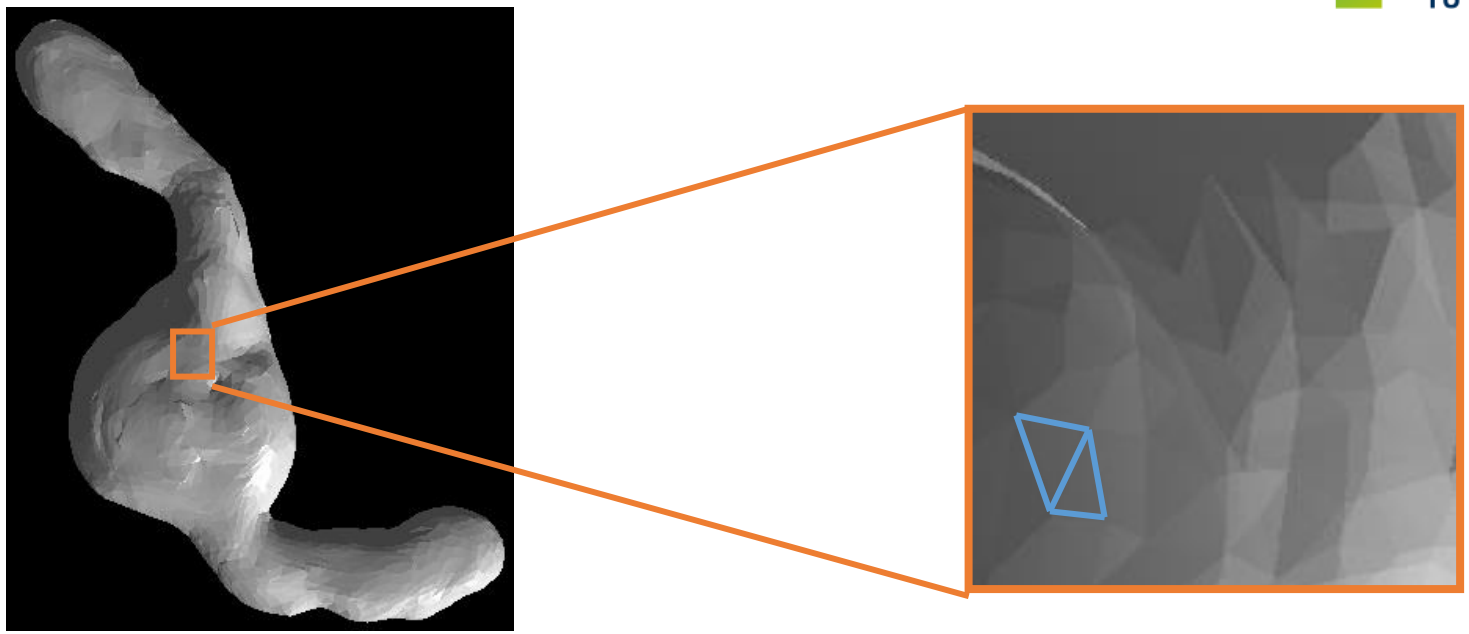
- Pixel and voxel borders introduce artifacts, potentially problematic for measurements, e.g. surface area

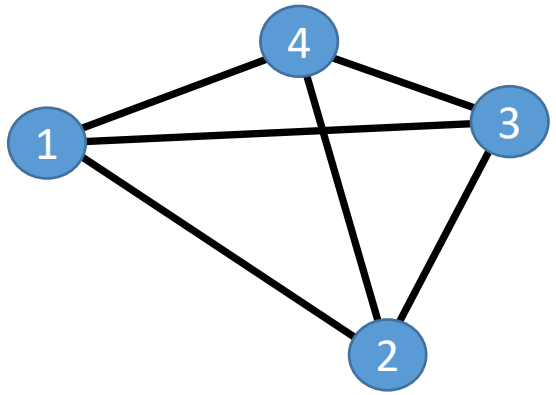




Surface meshes

- Points on a surfaces connected by triangles forma a surface mesh





“Vertices” / points

Point x	Point y	Point z
x_1	y_1	z_1
X_2	Y_2	Z_2
X_3	Y_3	Z_3
X_4	Y_4	Z_4
…	…	…

+

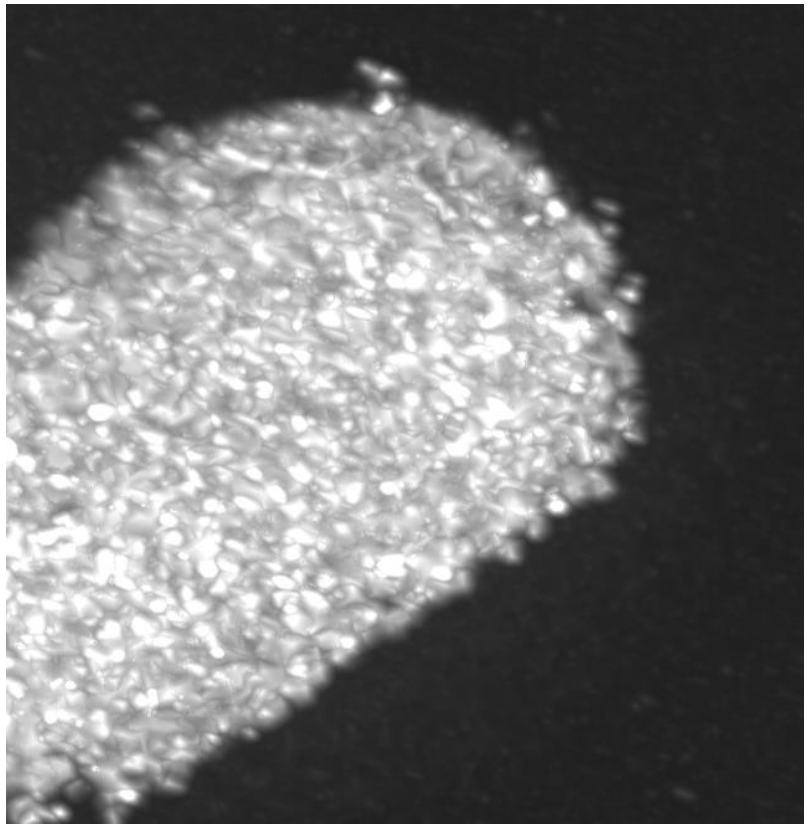
“Faces” / Triangles

Point 1	Point 2	Point 3
1	2	3
1	2	4
2	3	4
1	3	4

@haesleinhuepf

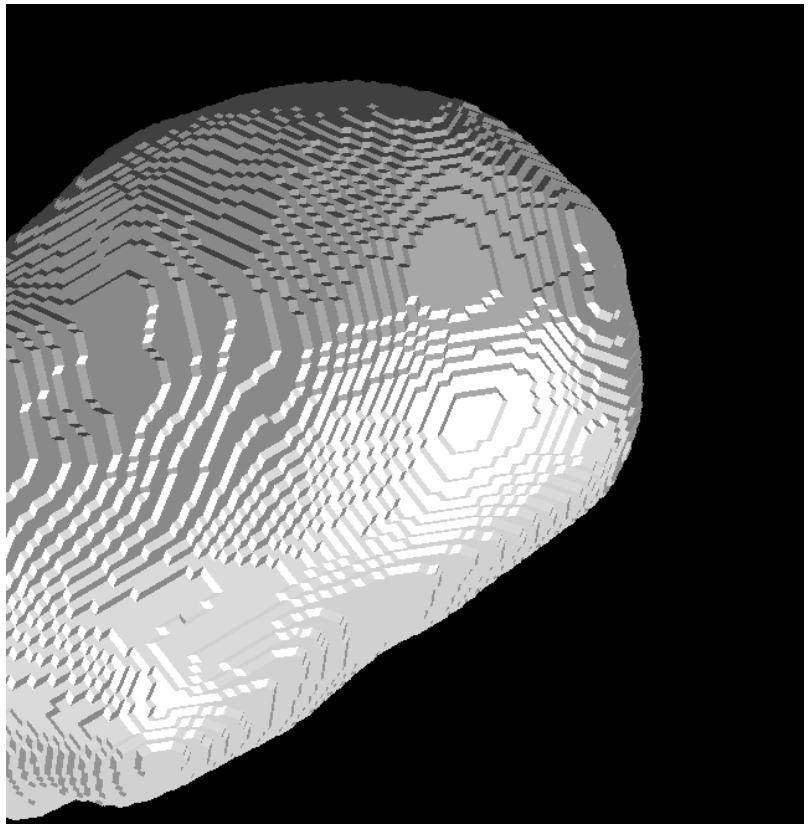
April 2023

Surface reconstruction



3D image of nuclei

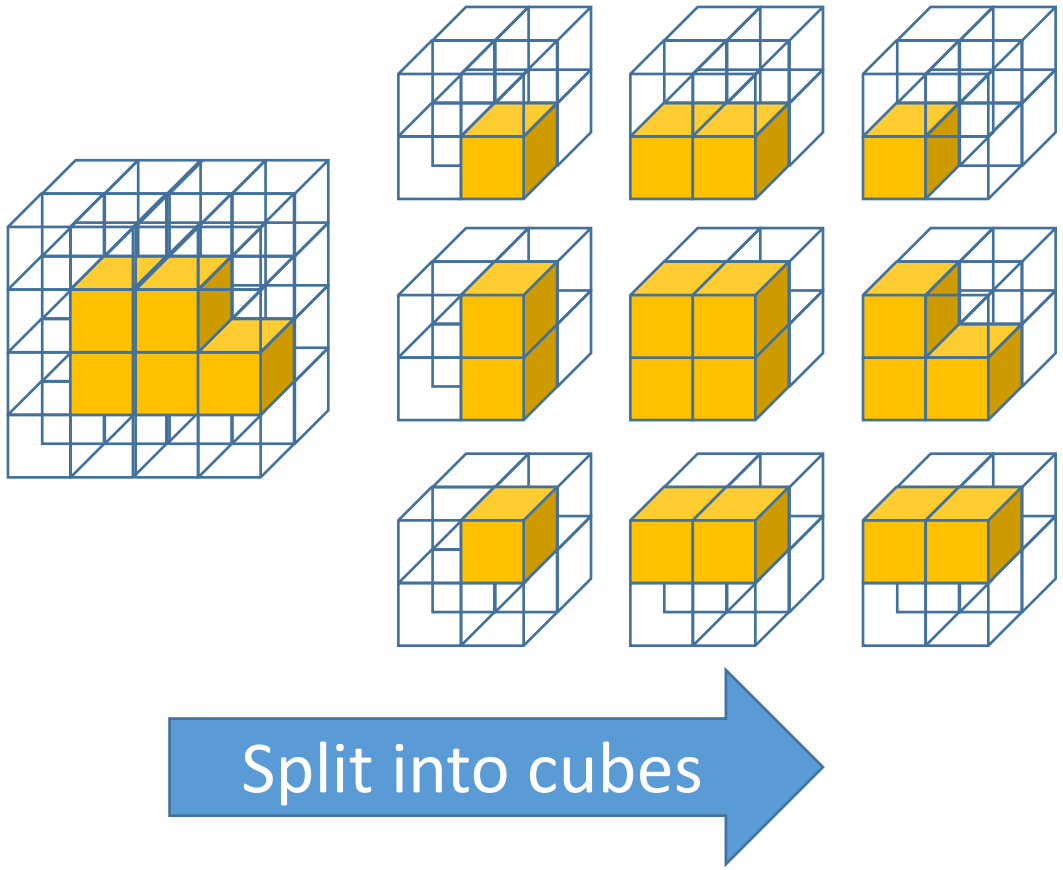
Gaussian filtered



Binary 3D image
(visualized as surface mesh)

Marching cubes algorithm

- Starting point: 3D binary image
- Cuts the image in small cubes and iterates over them

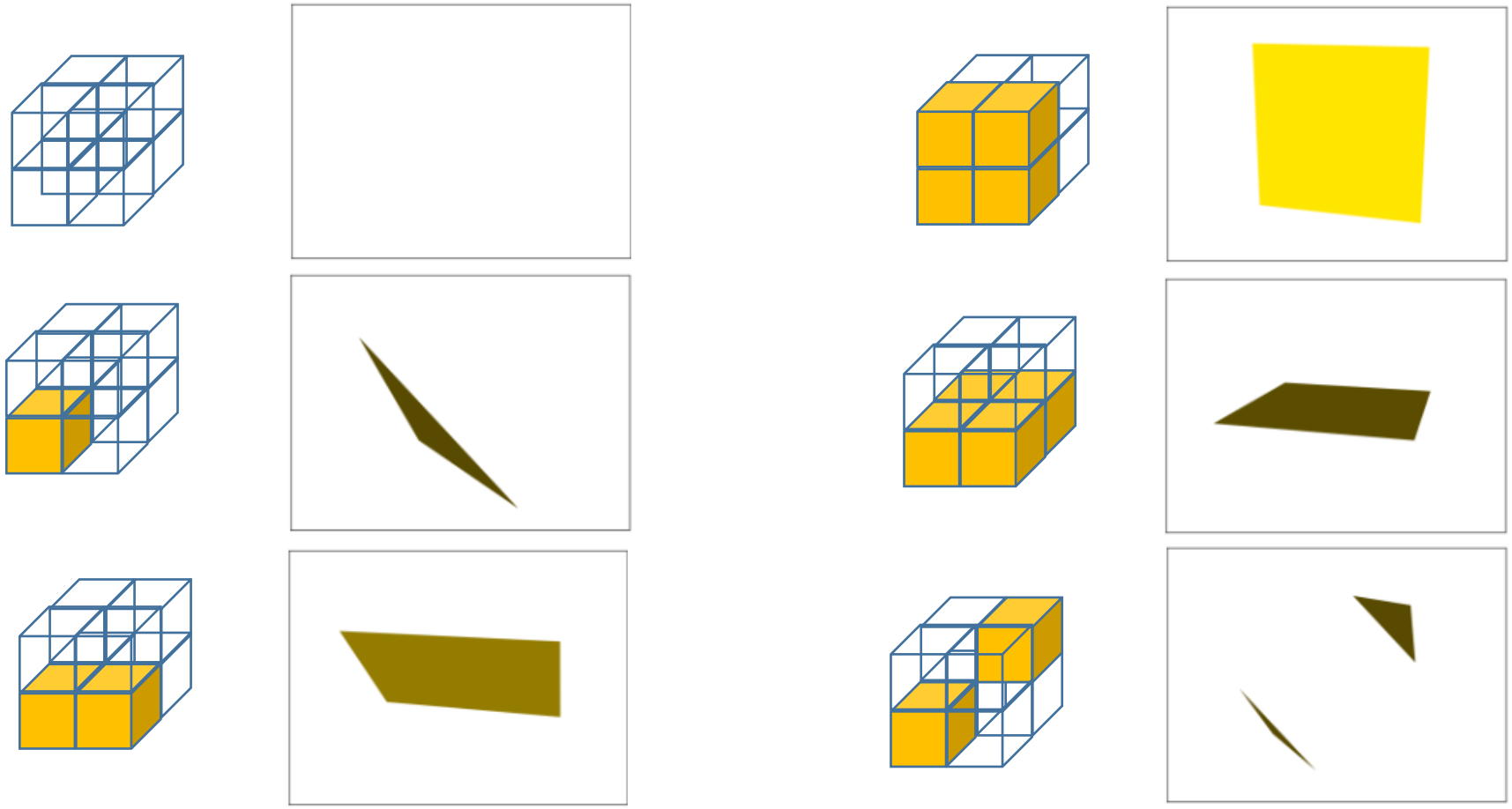


@haesleinhuepf

Lorensen, William E.; Cline, Harvey E. (1 August 1987). "Marching cubes: A high resolution 3D surface construction algorithm". *ACM SIGGRAPH Computer Graphics*. **21** (4): 163-169. CiteSeerX 10.1.1.545.613. doi:10.1145/37402.37422.

Marching cubes algorithm

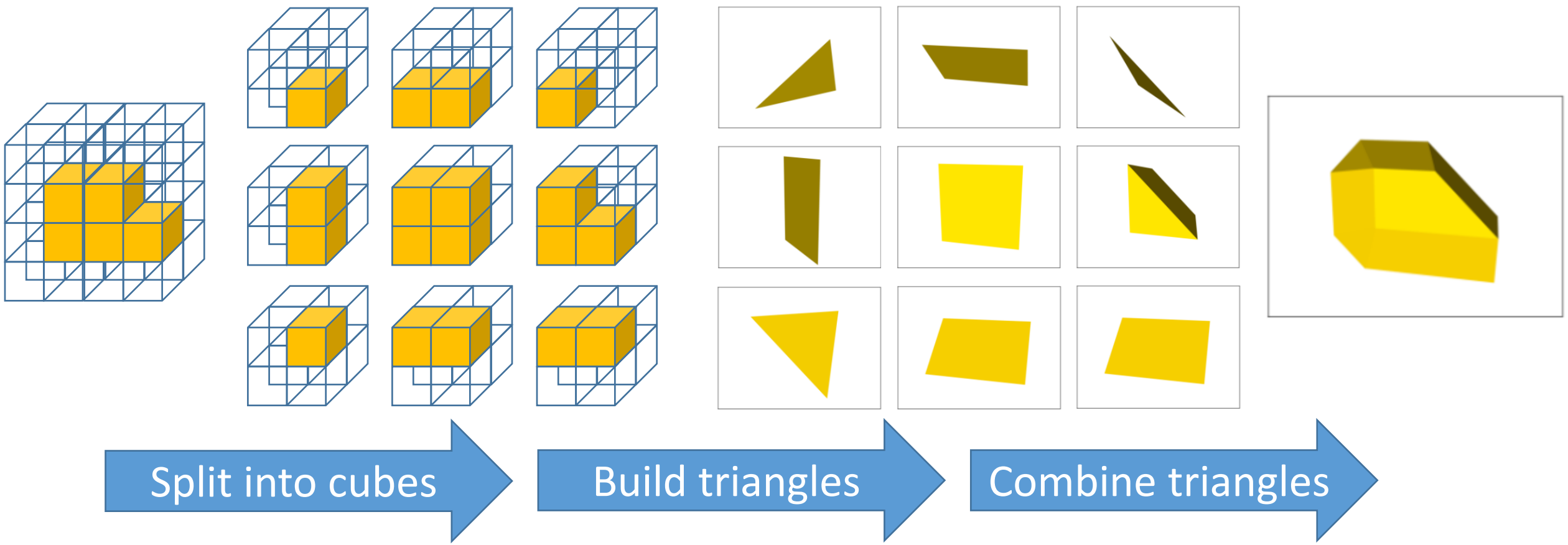
- Starting point: 3D binary image
- Cuts the image in small cubes and iterates over them



Lorensen, William E.; Cline, Harvey E. (1 August 1987). "Marching cubes: A high resolution 3D surface construction algorithm". *ACM SIGGRAPH Computer Graphics*. **21** (4): 163-169. CiteSeerX 10.1.1.545.613. doi:10.1145/37402.37422.

Marching cubes algorithm

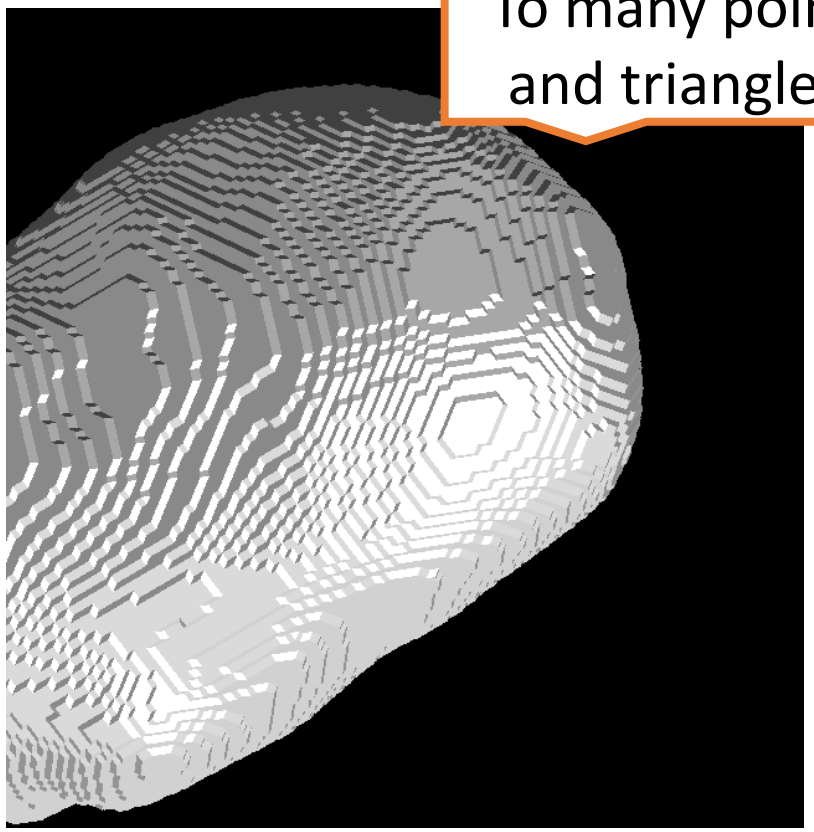
- Starting point: 3D binary image
- Cuts the image in small cubes and iterates over them



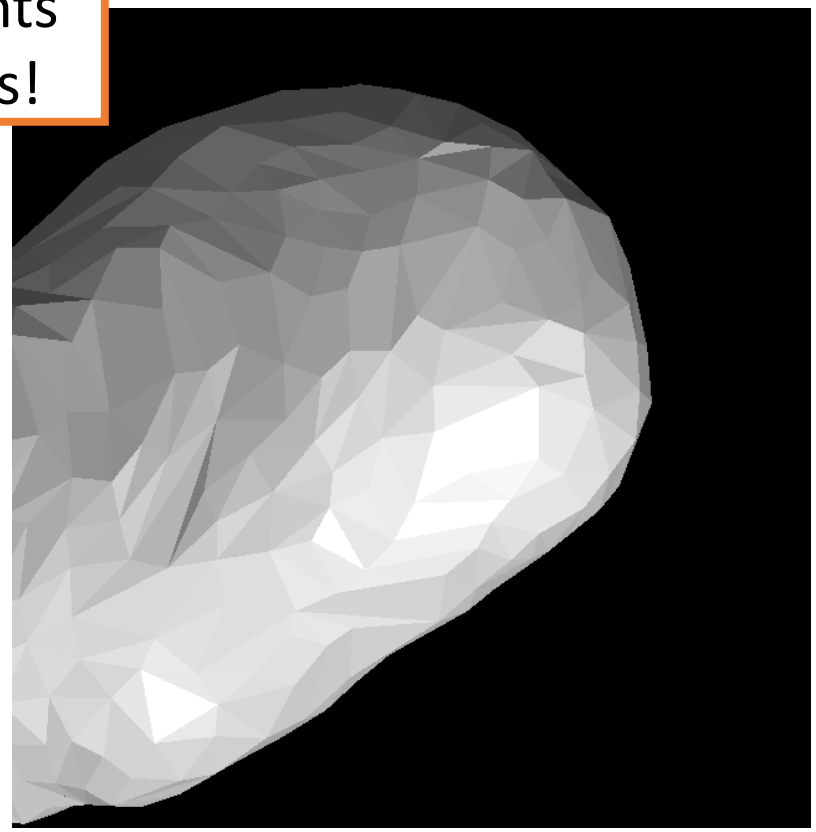
Lorensen, William E.; Cline, Harvey E. (1 August 1987). "Marching cubes: A high resolution 3D surface construction algorithm". *ACM SIGGRAPH Computer Graphics*. **21** (4): 163-169. CiteSeerX 10.1.1.545.613. doi:10.1145/37402.37422.

Surface post-processing

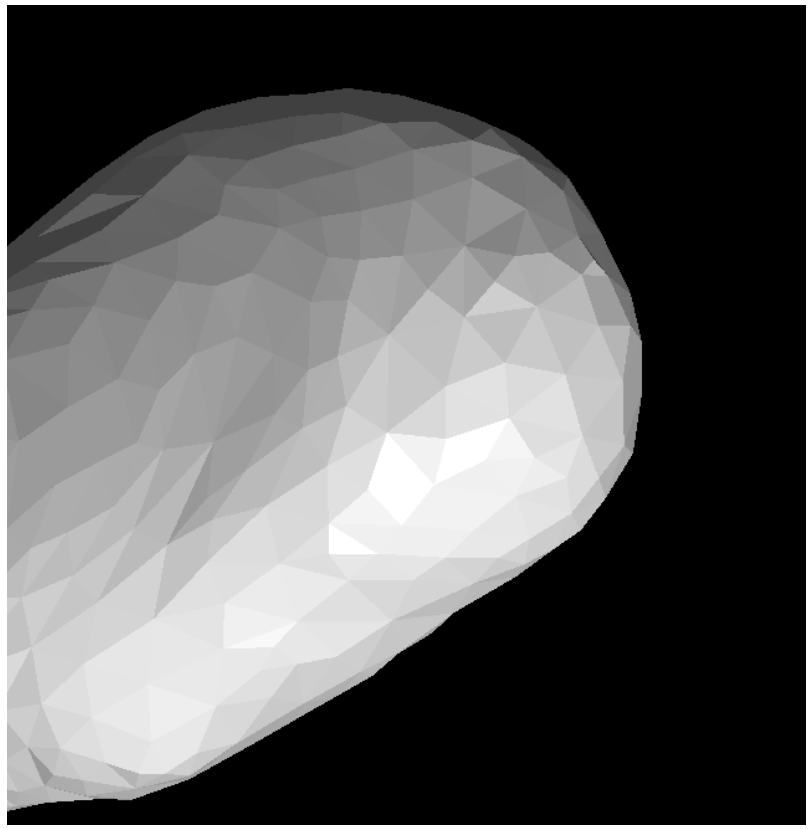
- Necessary to better match biological reality.



Marching cubes result



Simplified mesh
(less points, locally averaged)



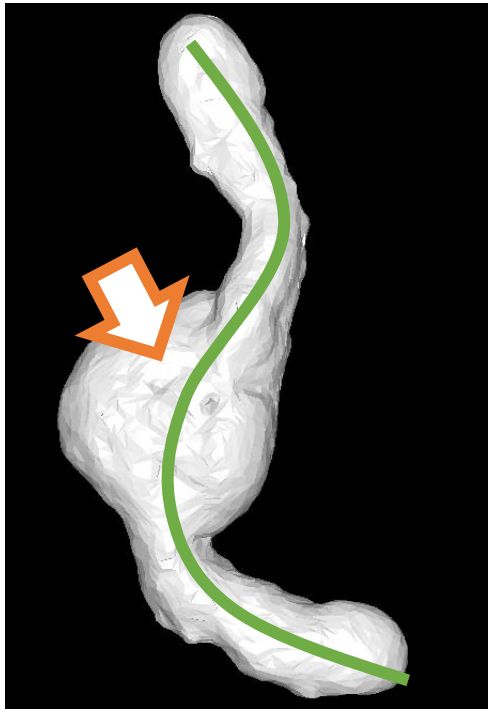
Smoothed mesh
(position locally planarized)

Surface post-processing

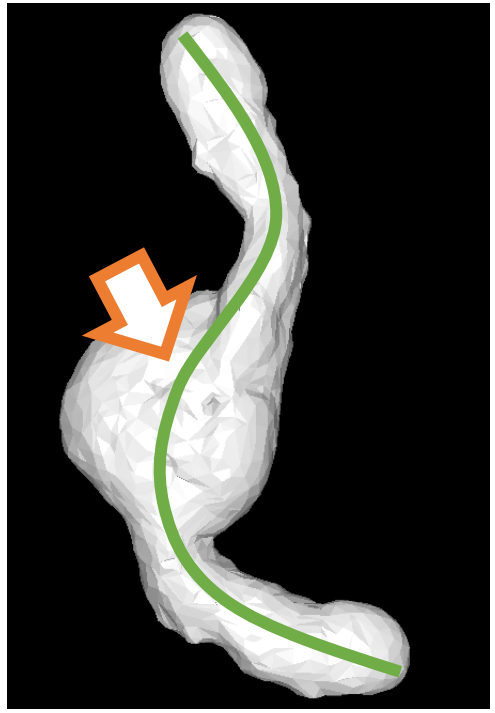
- Every processing step has consequences errors of later measurements
- Depends on desired measurement

Number of small concave regions

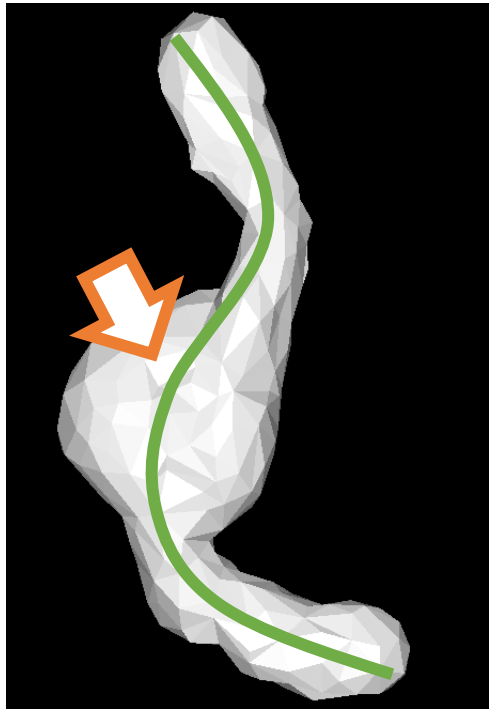
Total length



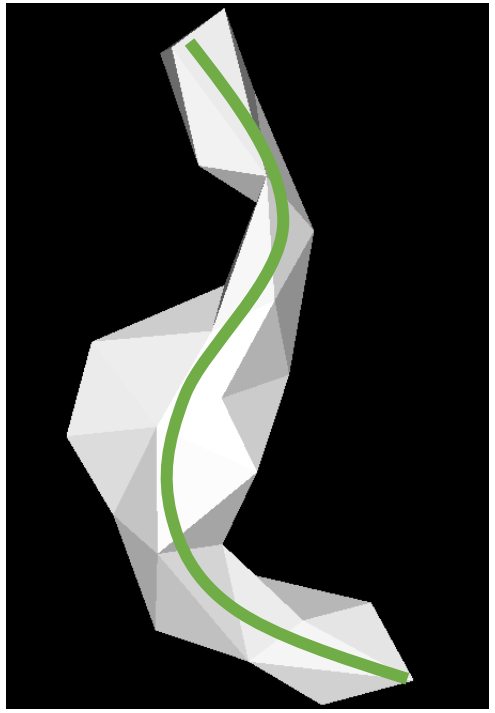
Surface mesh



Simplified by factor 0.5



Simplified by factor 0.05



Simplified by factor 0.01

PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Segmentation quality estimation

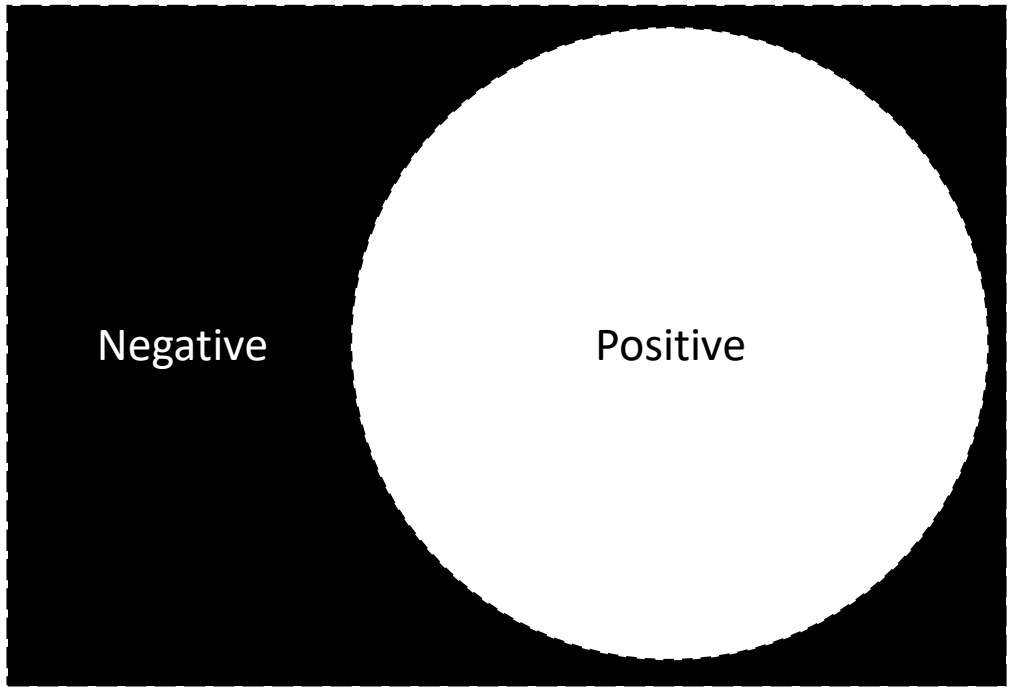
Robert Haase

@haesleinhuepf

December 2022

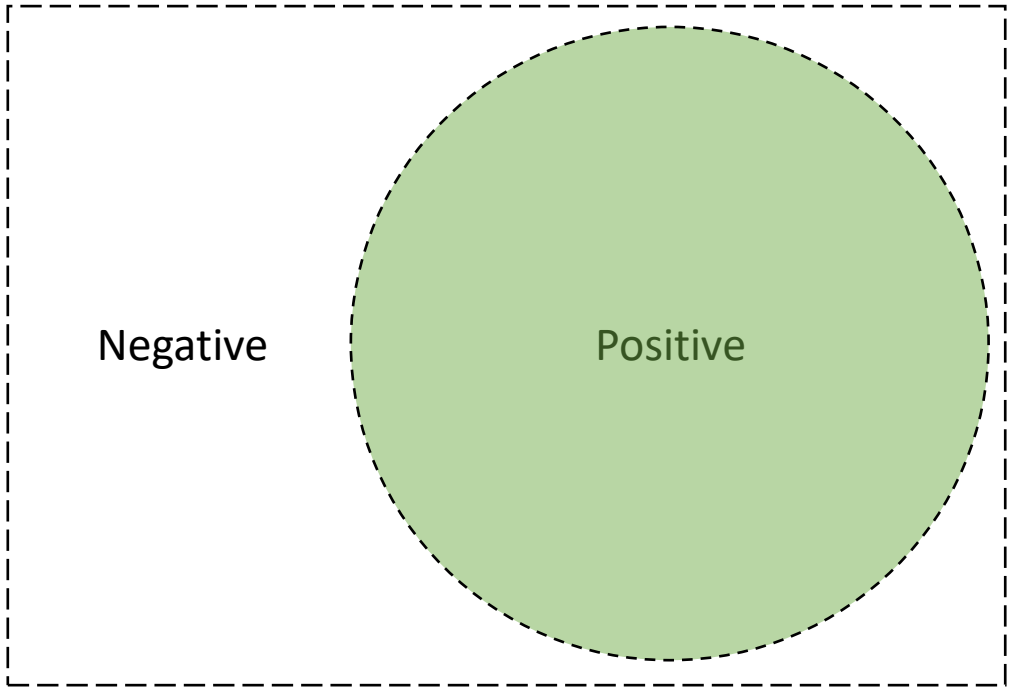
Segmentation quality estimation

- In general
  - Define what's positive and what's negative.



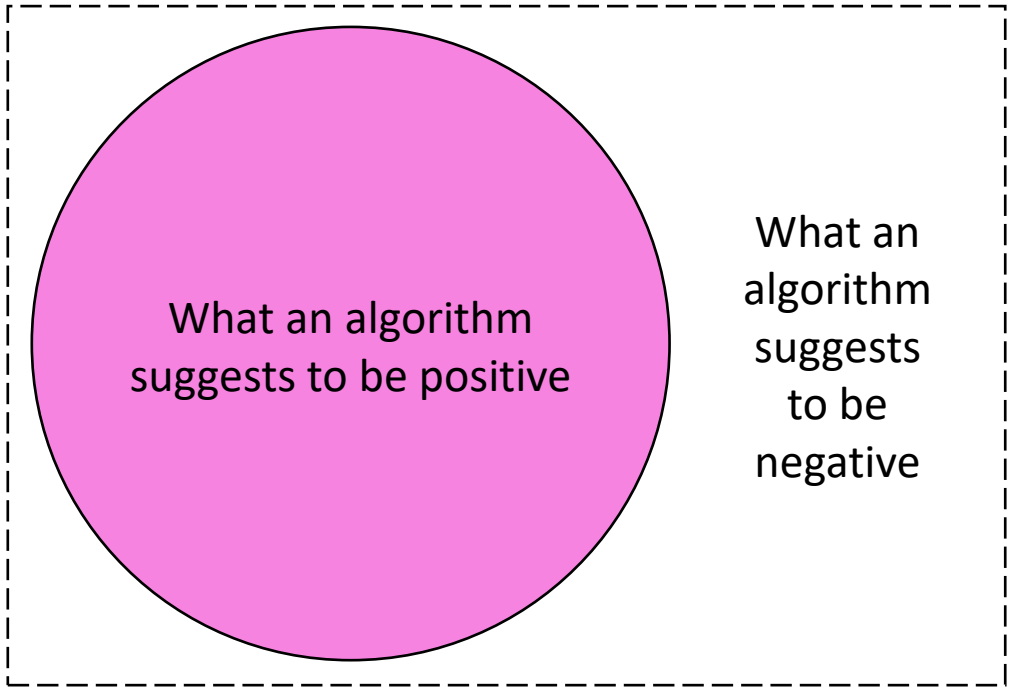
Segmentation quality estimation

- In general
  - Define what's positive and what's negative.



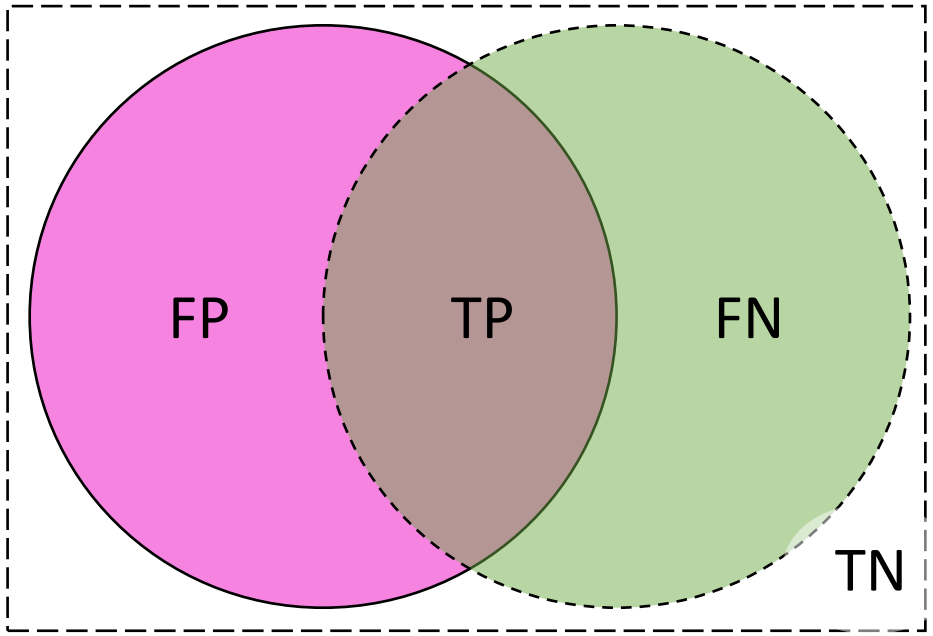
Segmentation quality estimation

- In general
  - Define what's positive and what's negative.



Segmentation quality estimation

- In general
  - Define what's positive and what's negative.
  - Compare with a reference to figure out what was true and false
  - Welcome to the Theory of Sets



A — Prediction A

B — Reference B (ground truth)

ROI — Region of interest

TP — True-positive

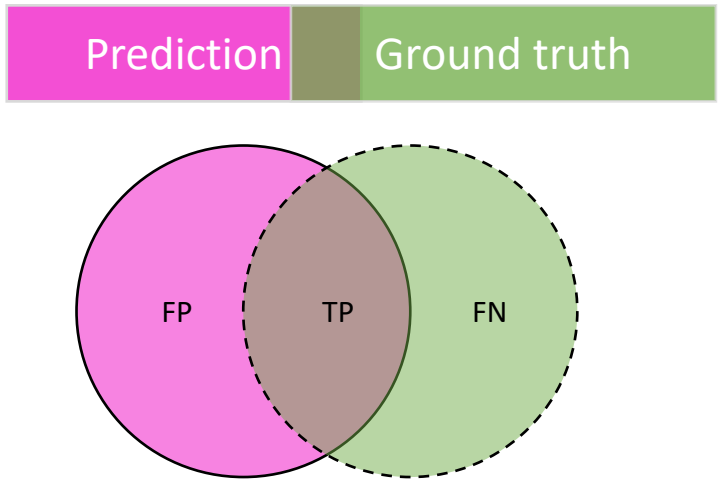
FN — False-negative

FP — False-positive

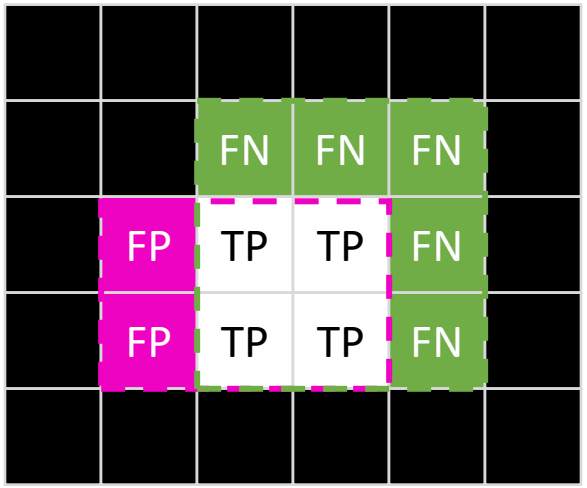
TN — True-negative

Overlap (a.k.a. Jaccard index)	$\frac{TP}{TP+FN+FP}$	How much do A and B overlap?
Precision	$\frac{TP}{TP+FP}$	What fraction of points that were predicted as positives were really positive?
Recall (a.k.a. sensitivity)	$\frac{TP}{TP+FN}$	What fraction of positives points were predicted as positives?

Pixel-wise versus Object-wise evaluation



- Pixel wise: Segmentation quality



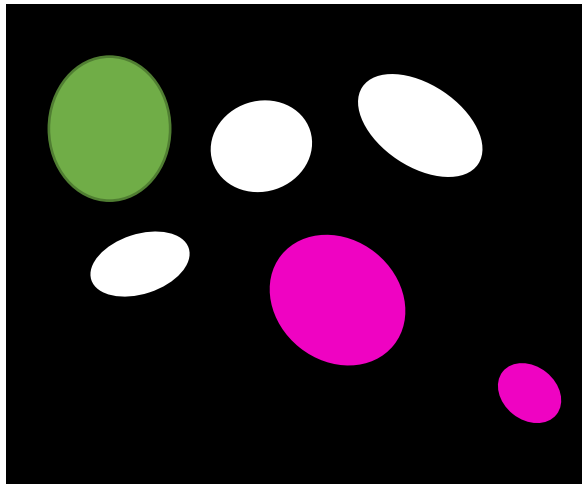
True-positive: 4

False-negative: 5

False-positive: 2

Precision: 4/6 = 66%
Recall: 4/9 = 44%

- Object wise: Detection quality



True-positive: 3

False-negative: 1

False-positive: 2

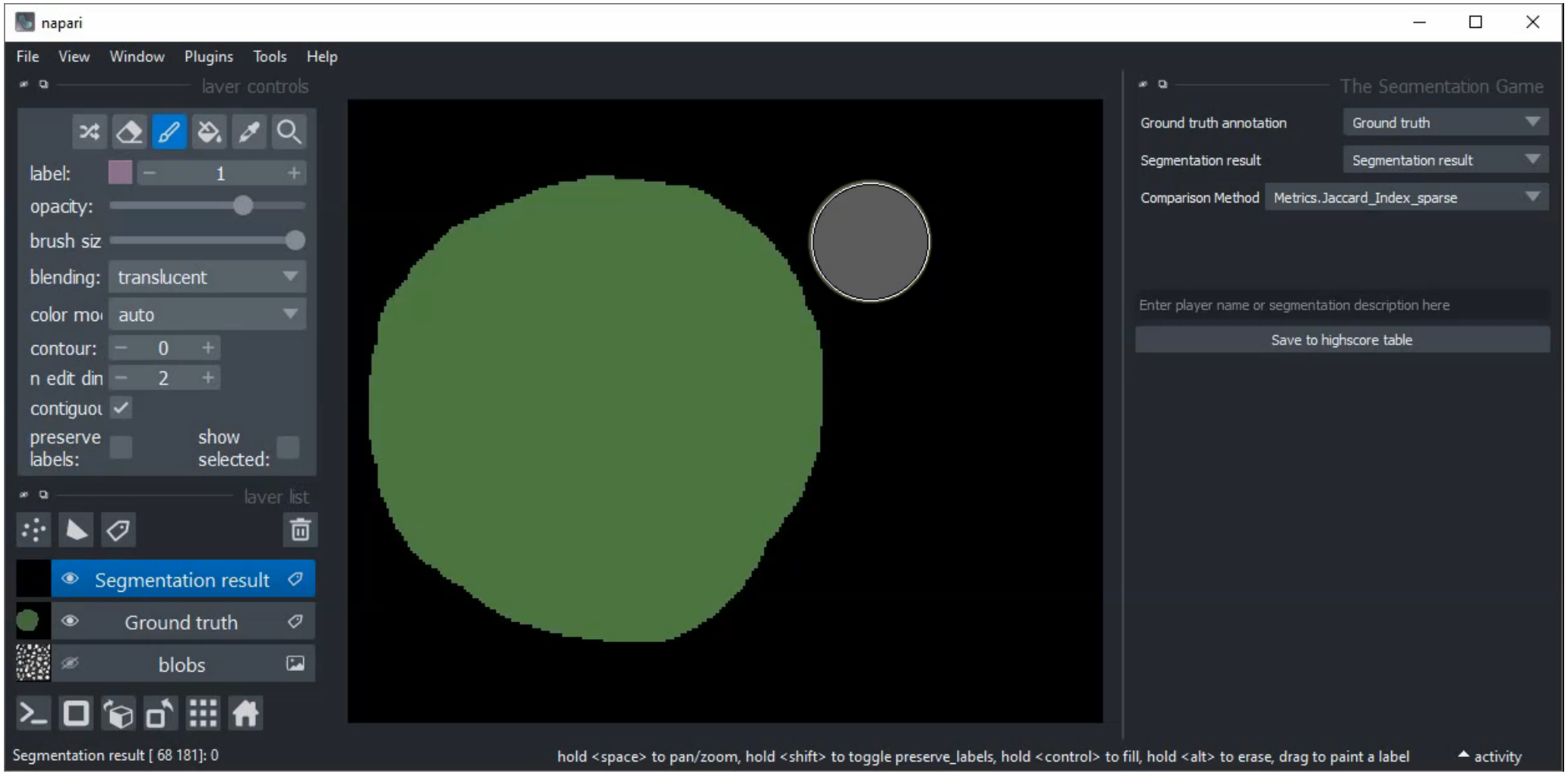
Precision: 3/4 = 75%
Recall: 3/5 = 60%

Precision $\frac{TP}{TP+FP}$

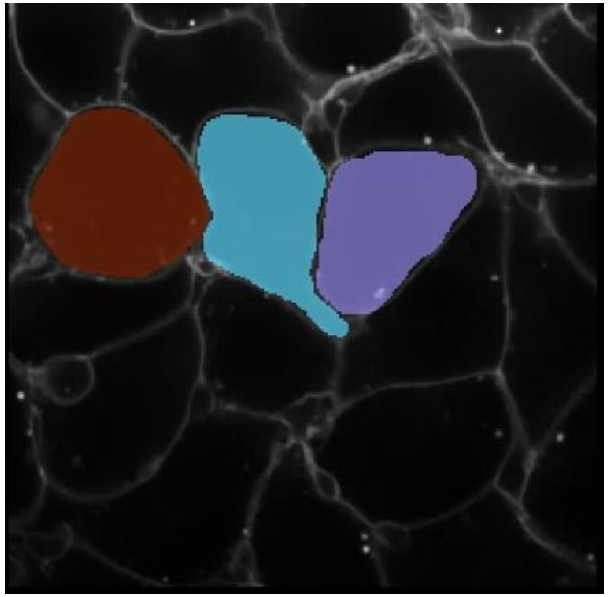
Recall (a.k.a. sensitivity) $\frac{TP}{TP+FN}$

Pixel-wise versus Object-wise evaluation

- Average Overlap for all ground-truth objects
- https://github.com/haesleinhuepf/the-segmentation-game



Sparse Jaccard Index



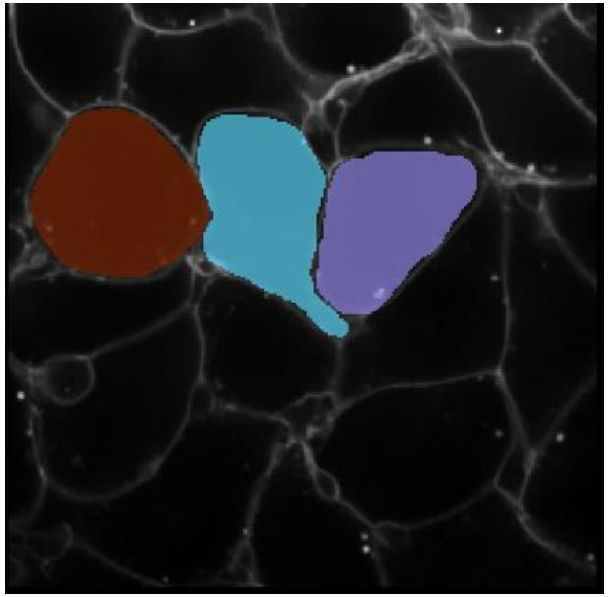
This is a …

Sparse instance segmentation

Sparse semantic segmentation

Sparse Jaccard Index

- From some sparsely labels objects we can estimate segmentation quality



Sparse instance annotation

J = 0.35

J = 0.66

J = 0.69

Source: https://github.com/haesleinhuepf/napari-workflow-optimizer

Segmentation quality estimation

- Voxel-wise Youden-Index

$$YI = p_{TP} + p_{TN} - 1$$

- Volume error

$$\Delta_V = V_A - V_B$$

$$\delta_V = \frac{\Delta_V}{V_B}$$

- Dice Index

$$DI(A,B) = \frac{2\,|A \cap B|}{|A| + |B|}$$

- Jaccard Index

$$JI(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{DI}{2 - DI}$$

- Contour distance

$$d_{e,min}(a,B) = \min(d_e(a,b)|b \in B)$$

$$\bar{d}_c(A,B) = \frac{\sum_{\forall a \in C(A)} d_{e,min}(a, C(B))}{|C(A)|}$$

$$\bar{d}_{bil,c}(A,B) = \frac{\bar{d}_c(A,B) + \bar{d}_c(B,A)}{2}$$

- Hausdorff distance

$$d_H(A,B) = \max(d_{e,min}(a,B)|a \in A)$$

$$d_{bil,H}(A,B) = \max(d_H(A,B), d_H(B,A))$$

- Simplified Hausdorff distance

$$d_H(A,B) = \max(d_{e,min}(a, C(B))|a \in C(A))$$

- Volume standard deviation

$$\delta_{\bar{V}} = 2\frac{|V_A - V_B|}{|V_A + V_B|}$$

- Classification error

$$e_{Class} = \frac{H}{|TP| + |FN|}$$

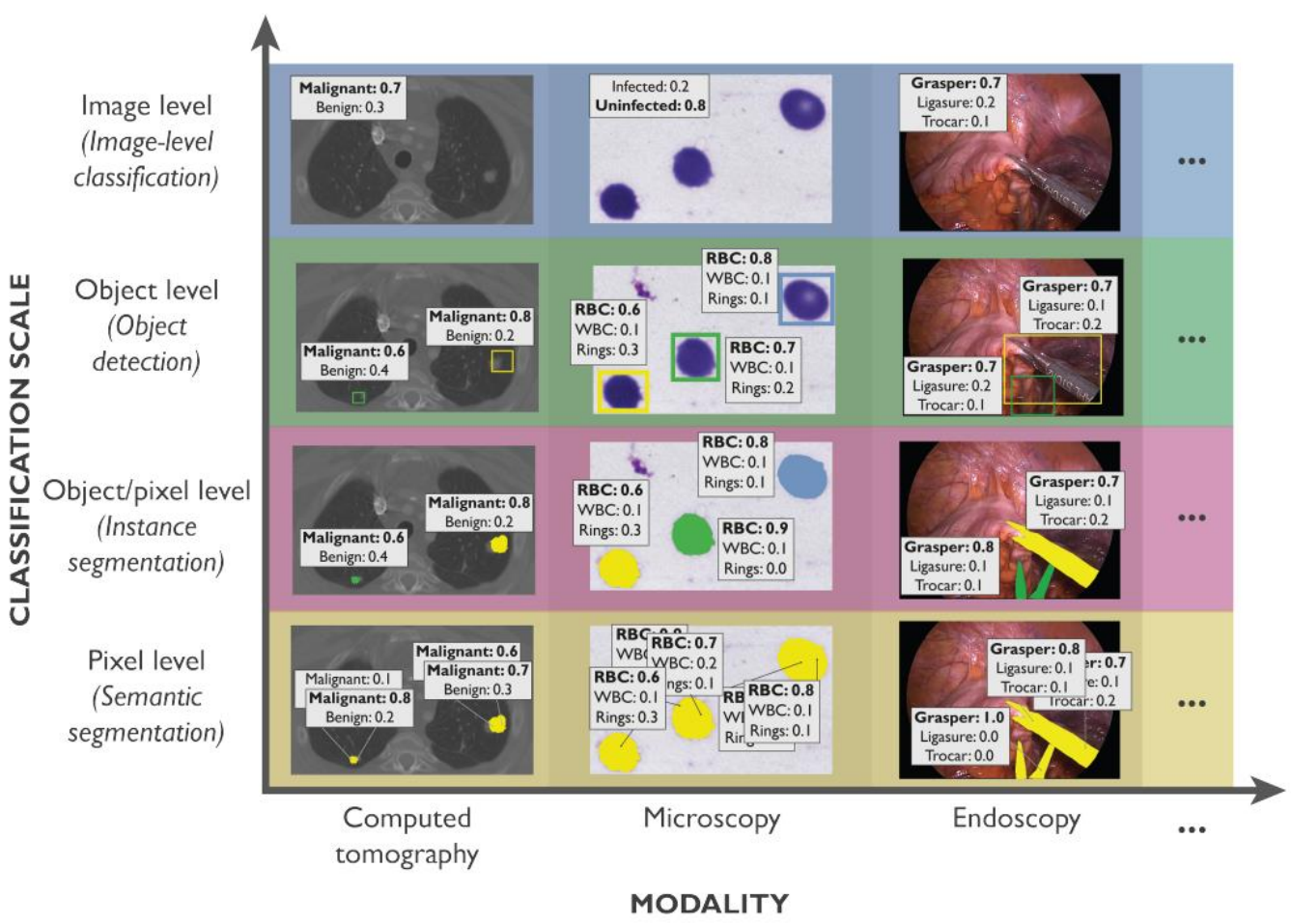
- Hamming distance

$$d_h = |A \cup B| - |A \cap B| = |FP| + |FN|$$

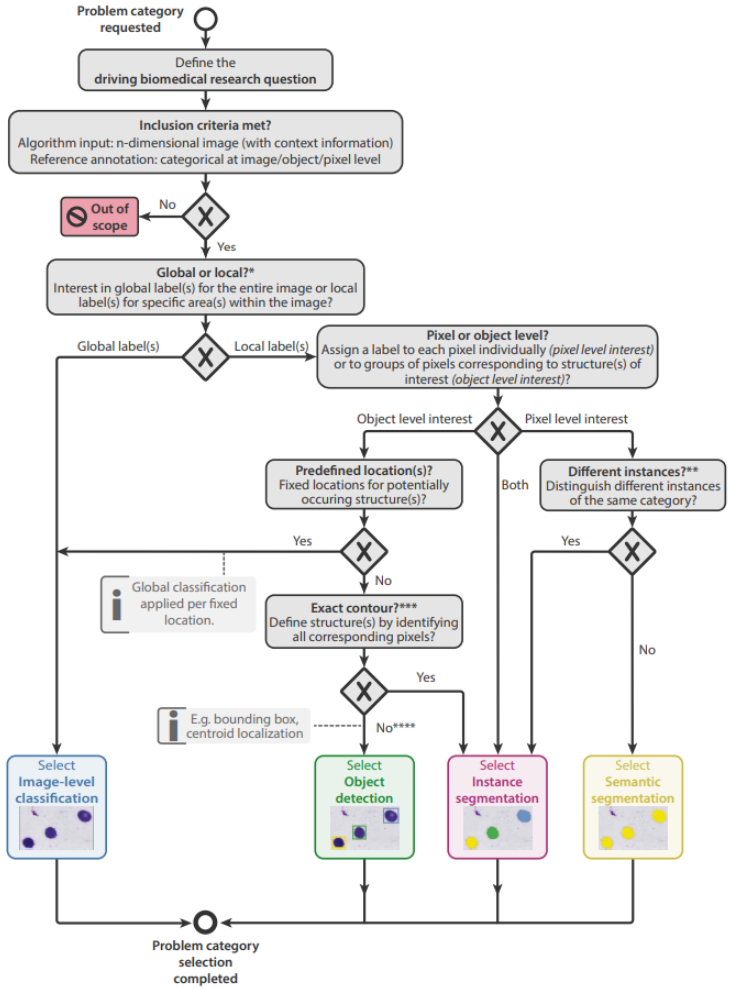
What metric to use when?

PoL Physics of Life TU Dresden

- "Metrics reloaded: Pitfalls and recommendations for image analysis validation" Maier-Hein, Reinke et al. https://arxiv.org/abs/2206.01653

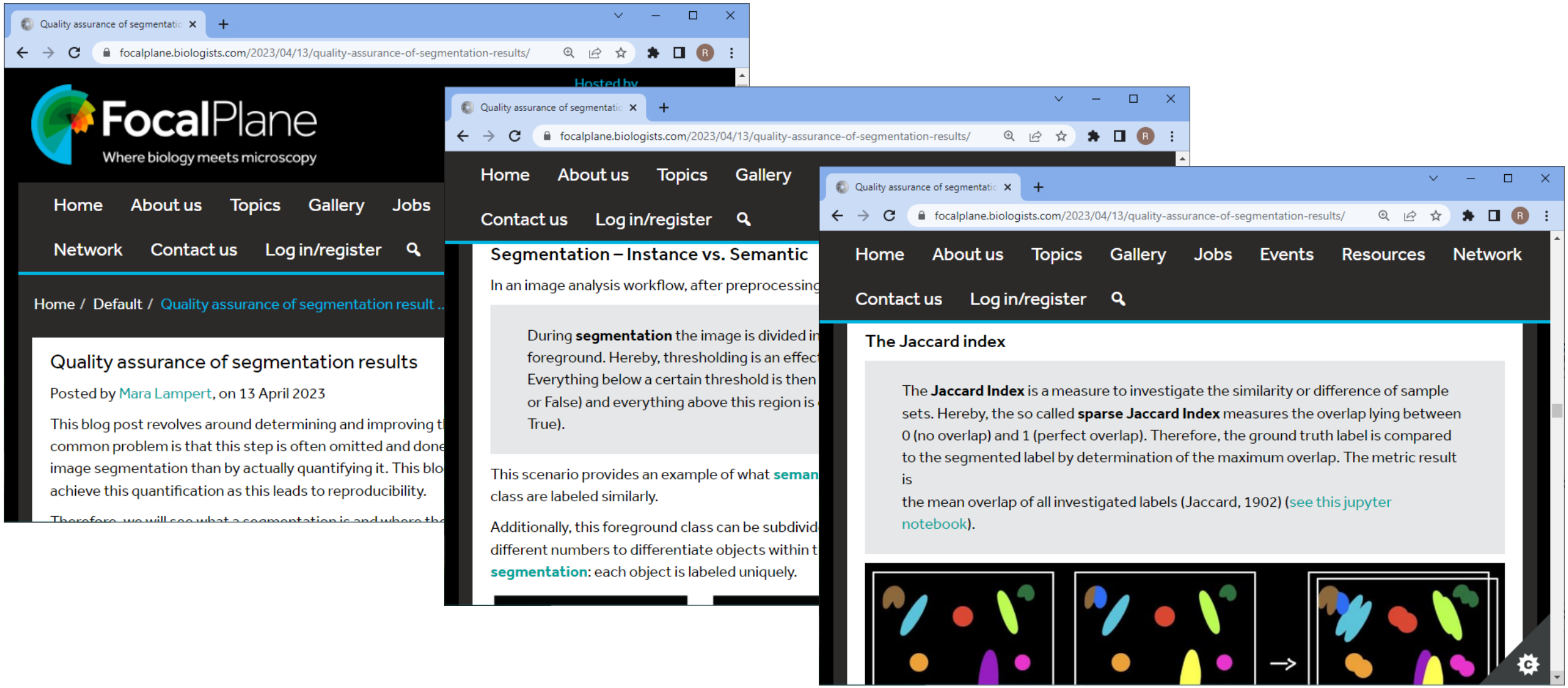


+S1



Further reading

PoL Physics of Life TU Dresden



https://focalplane.biologists.com/2023/04/13/quality-assurance-of-segmentation-results/

Image segmentation in Napari

Robert Haase

Using materials from

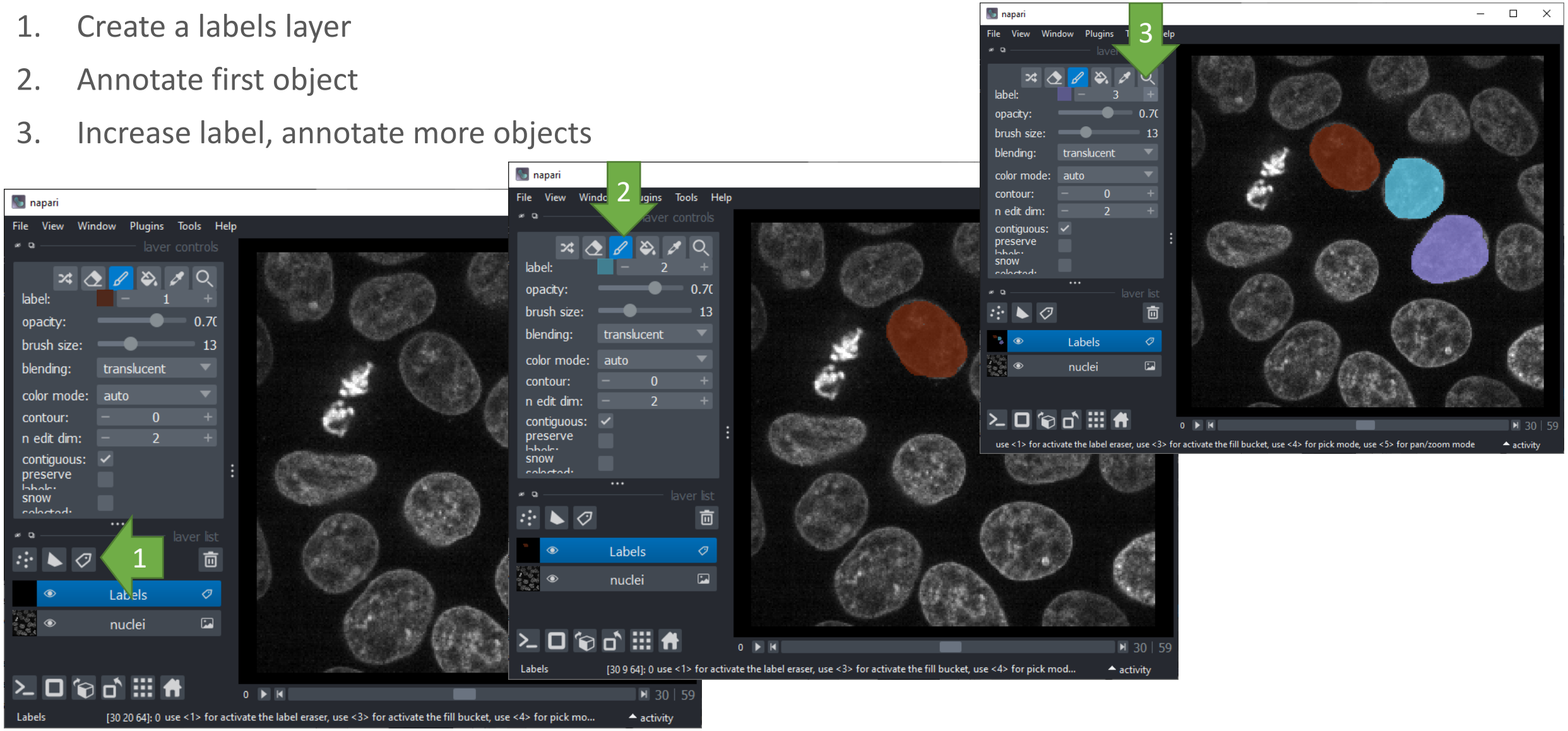
Ryan Savill George, MPI CBG Dresden

April 2023

@haesleinhuepf

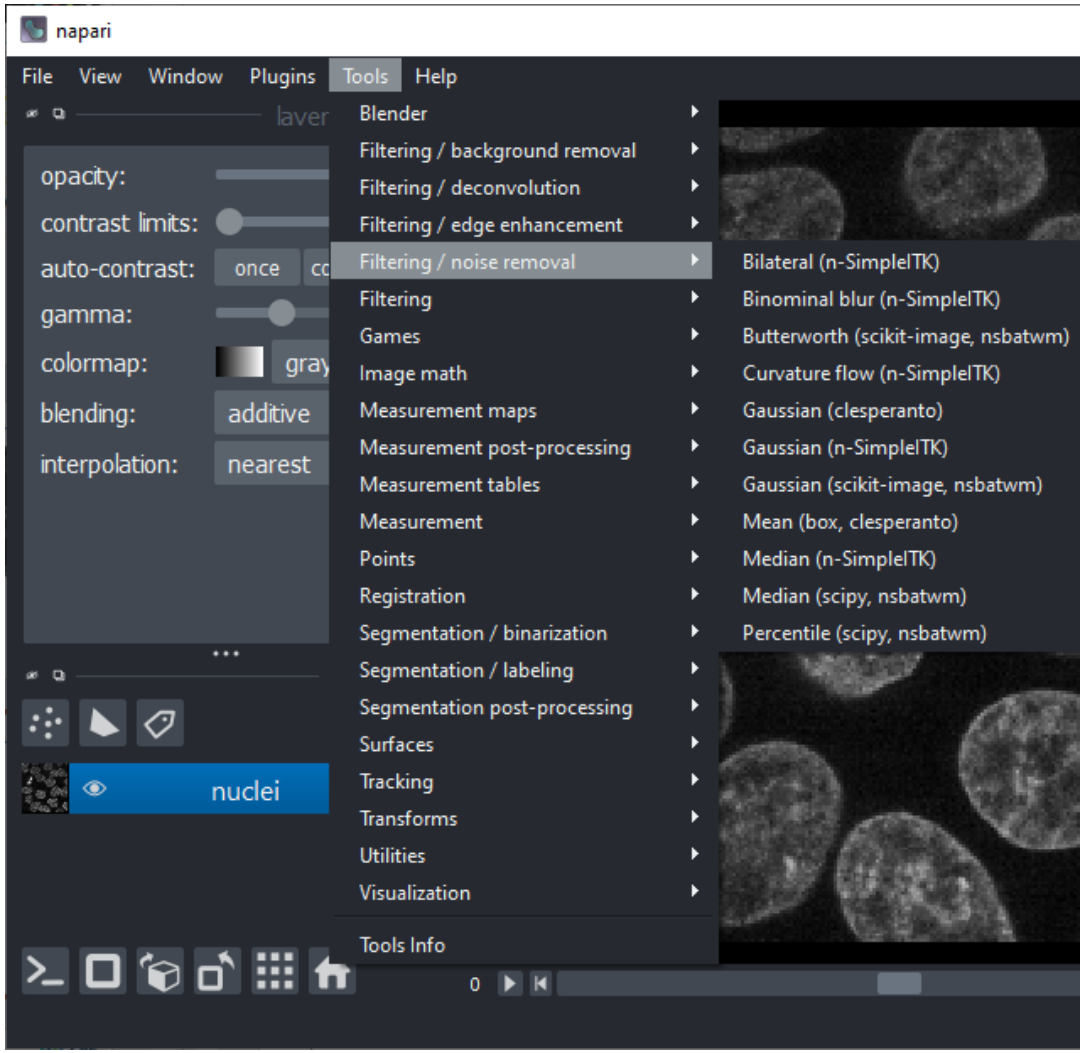
Manual annotation

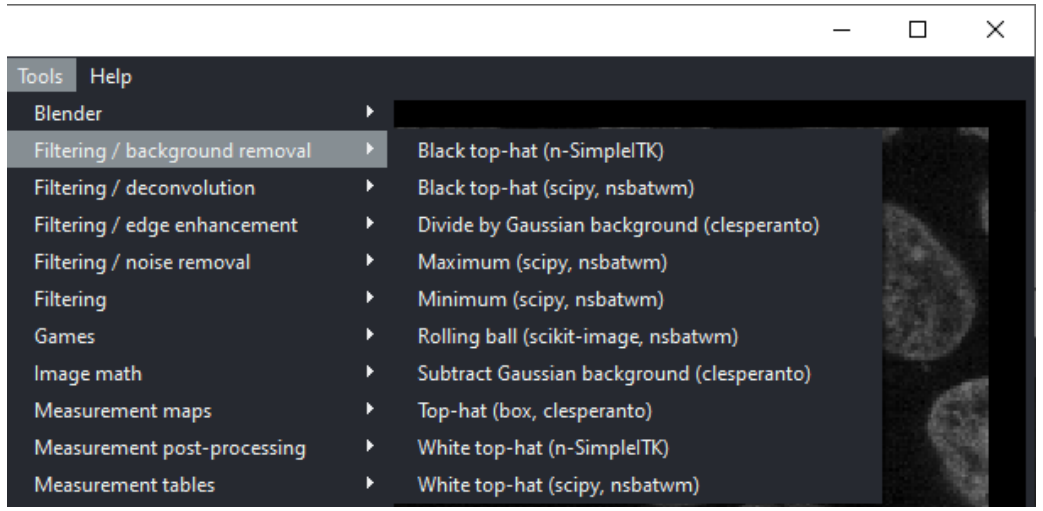
1. Create a labels layer
2. Annotate first object
3. Increase label, annotate more objects

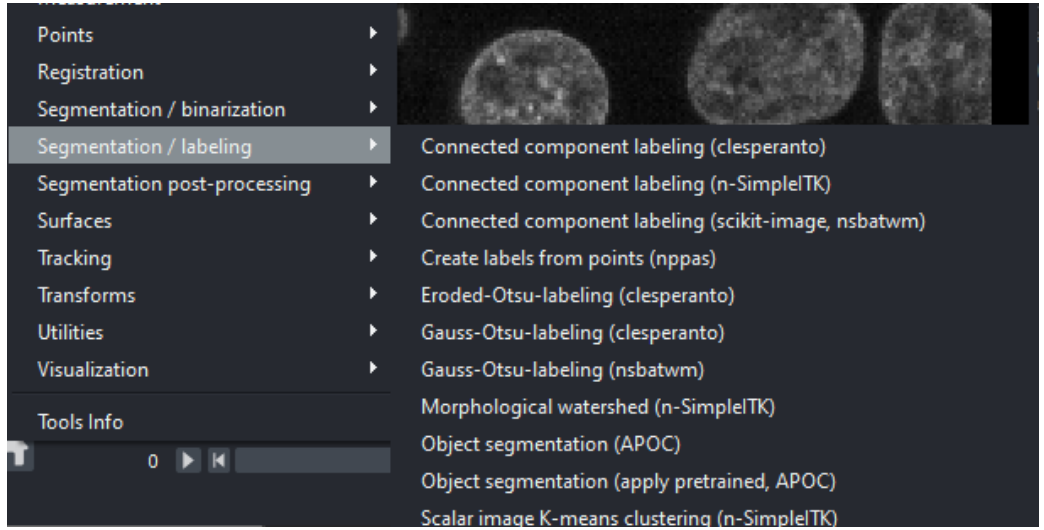


The Tools menu

- Organized in categories similar to what you learned last week







The Napari Assistant

- Tools > Utilities > Assistant (na)

Viewer controls

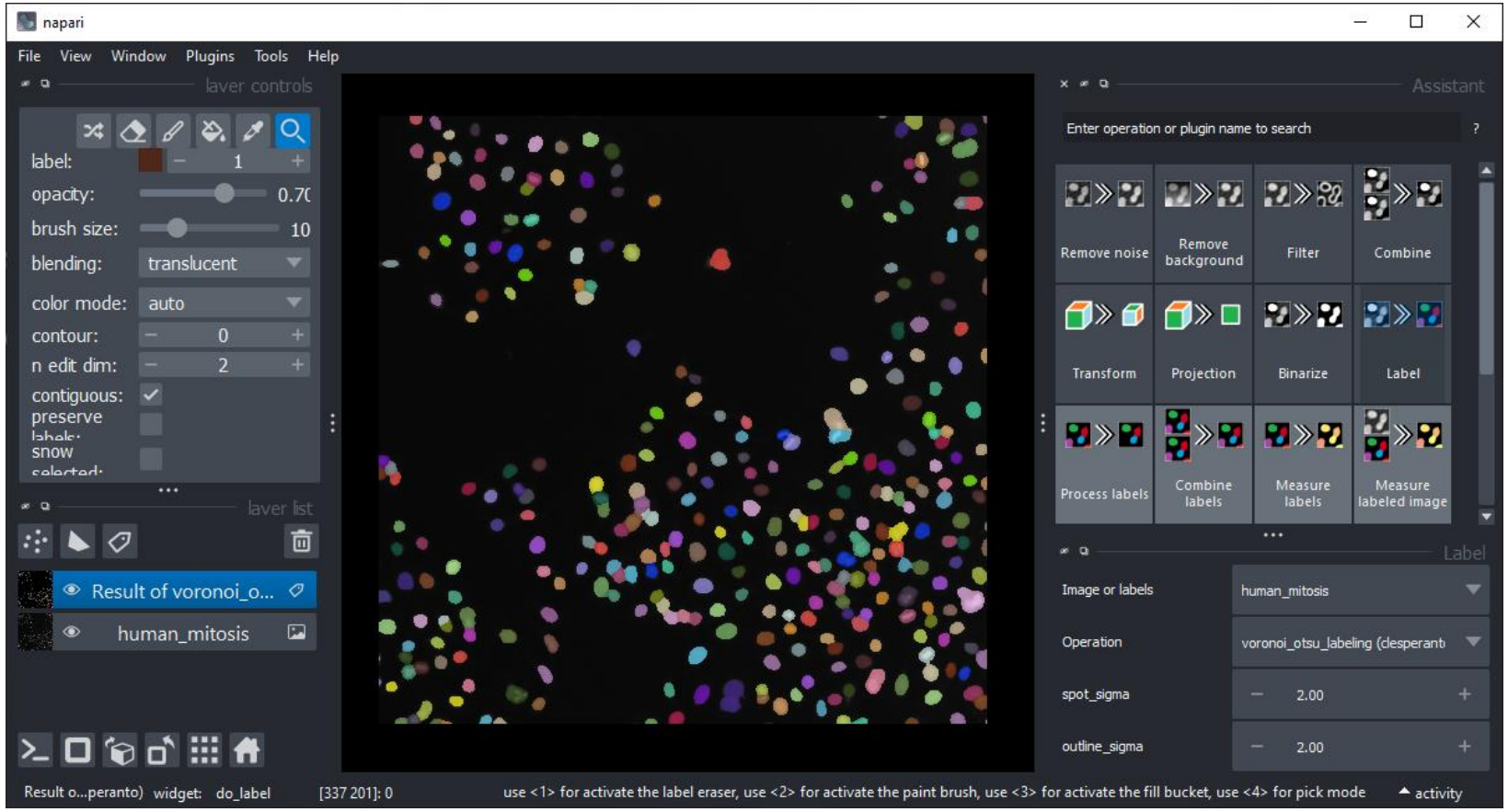
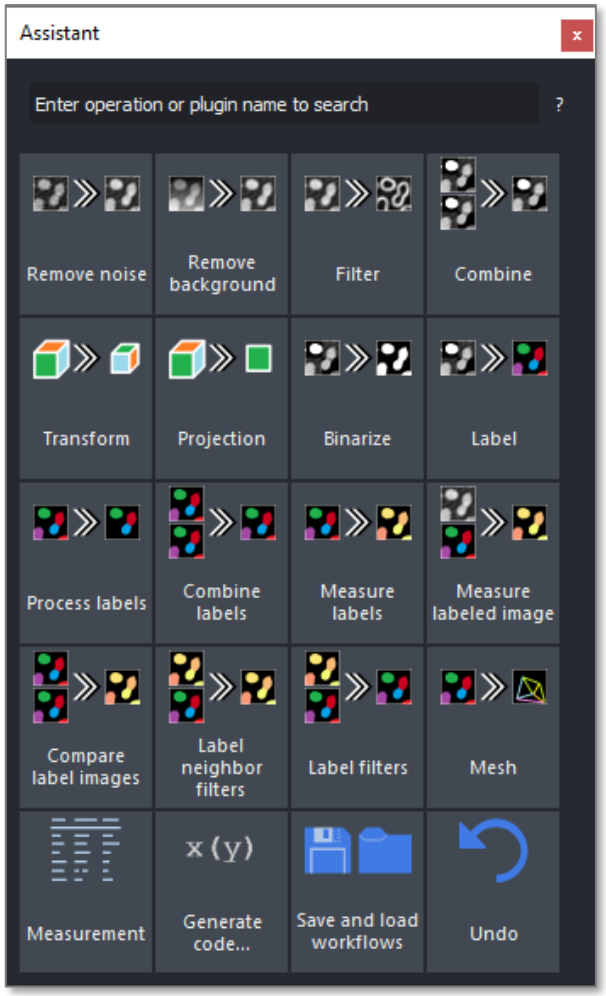


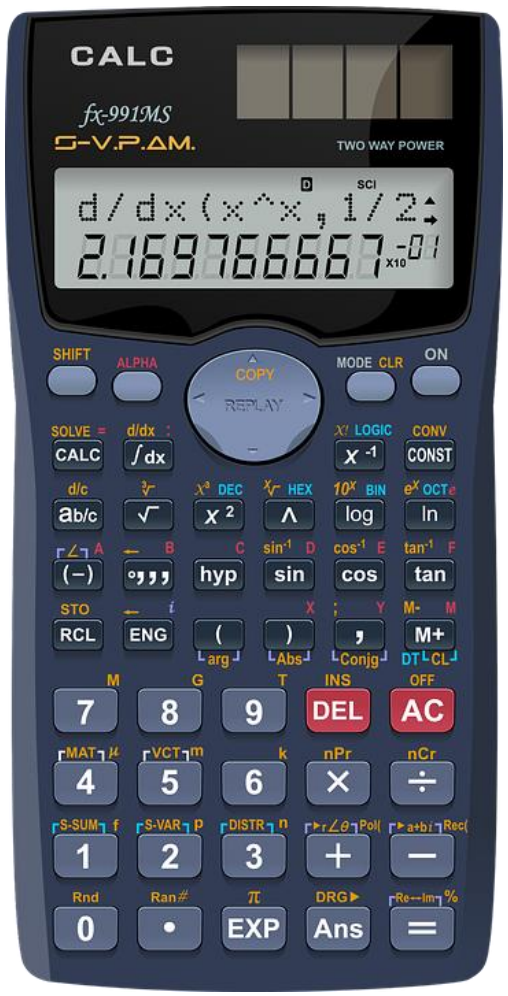
Image Processing

https://www.napari-hub.org/plugins/napari-assistant

The Napari Assistant

- A pocket-calculator-like interface to build image analysis workflows



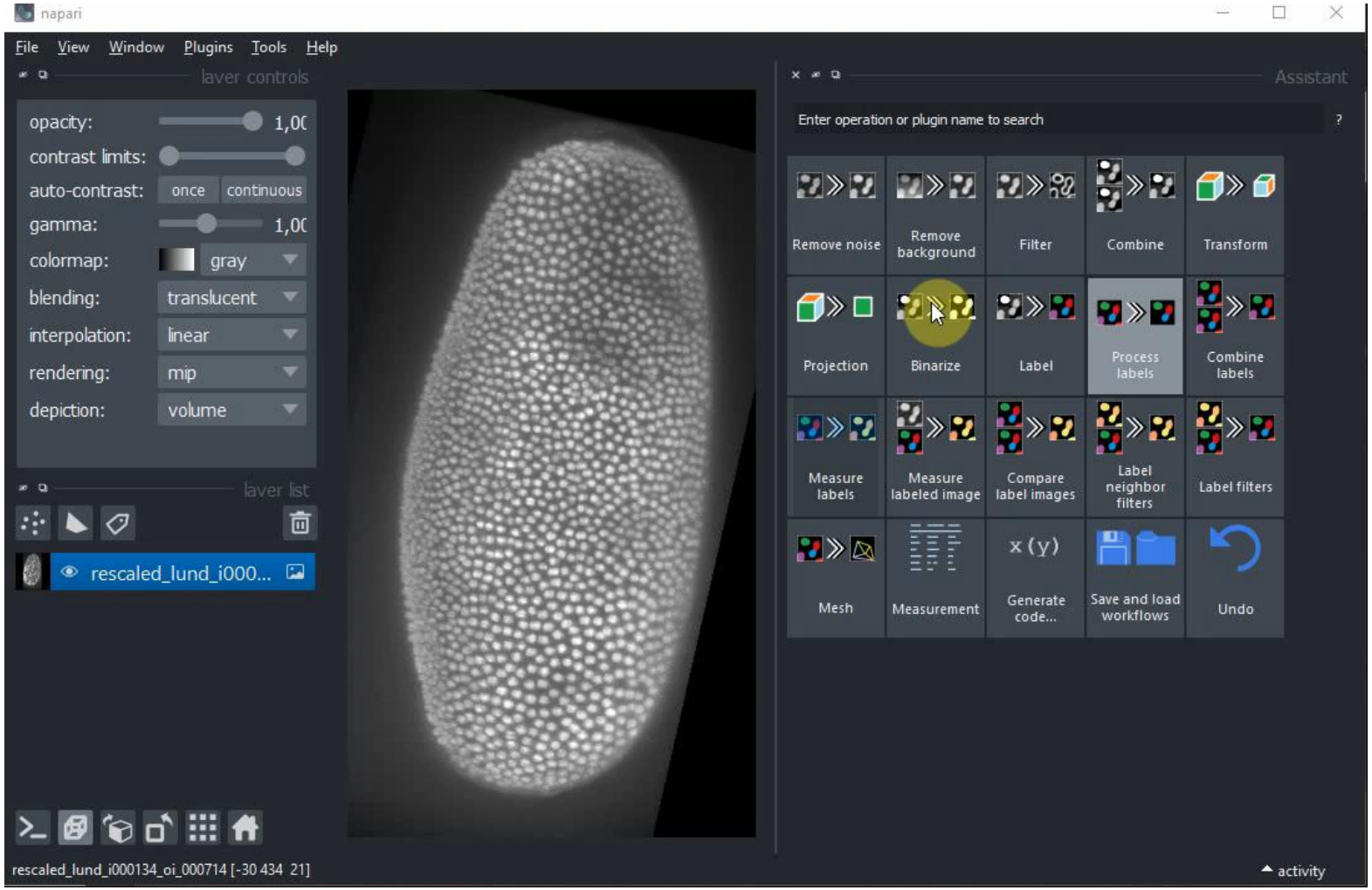


The Napari Assistant

- Classical image processing operations + advanced tools
- Saving&loading supported
- Undo [redo]
- Hints for next steps
- …

Big thanks to:

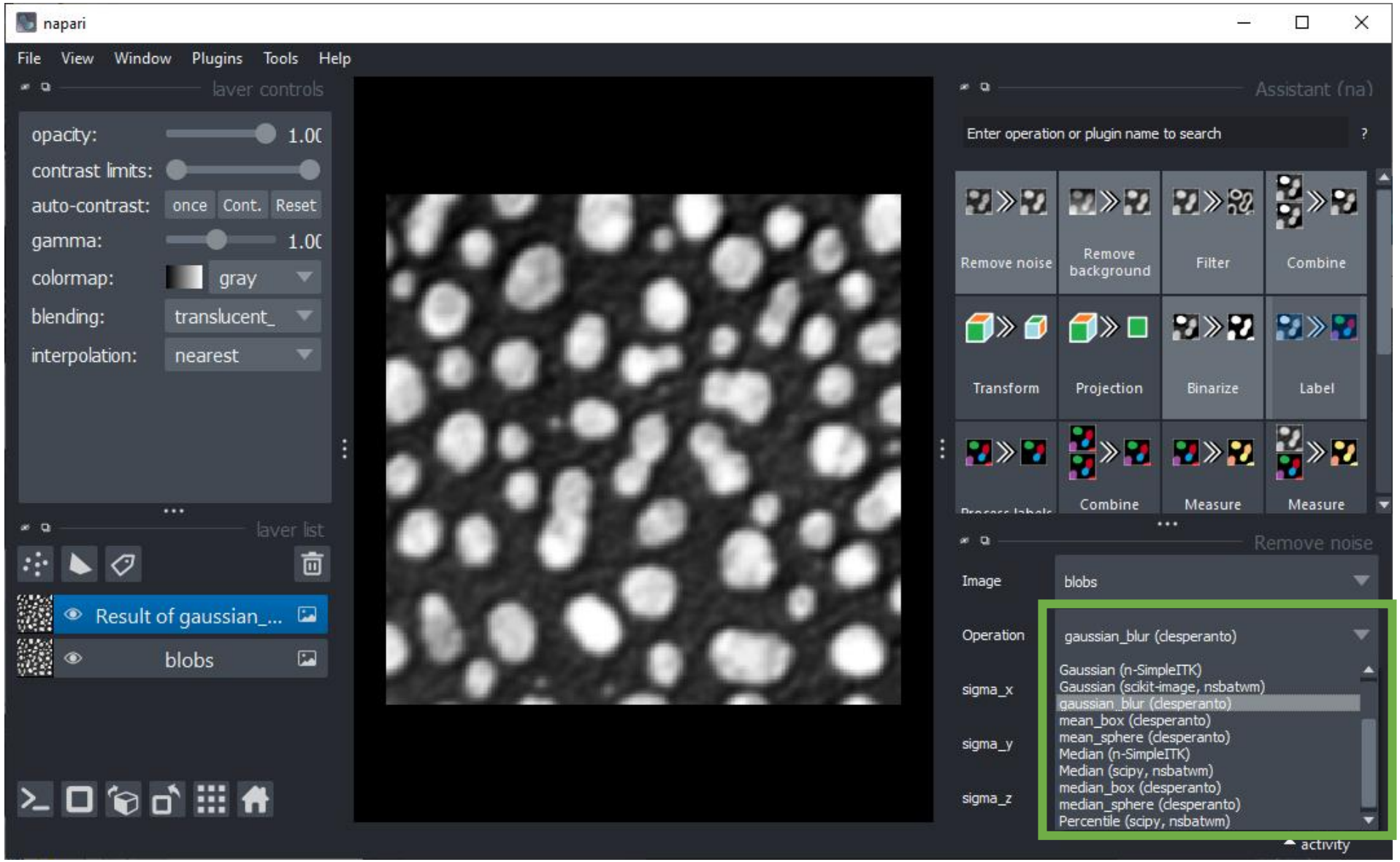
Ryan Savill
@RyanSavill4



https://www.napari-hub.org/plugins/napari-assistant

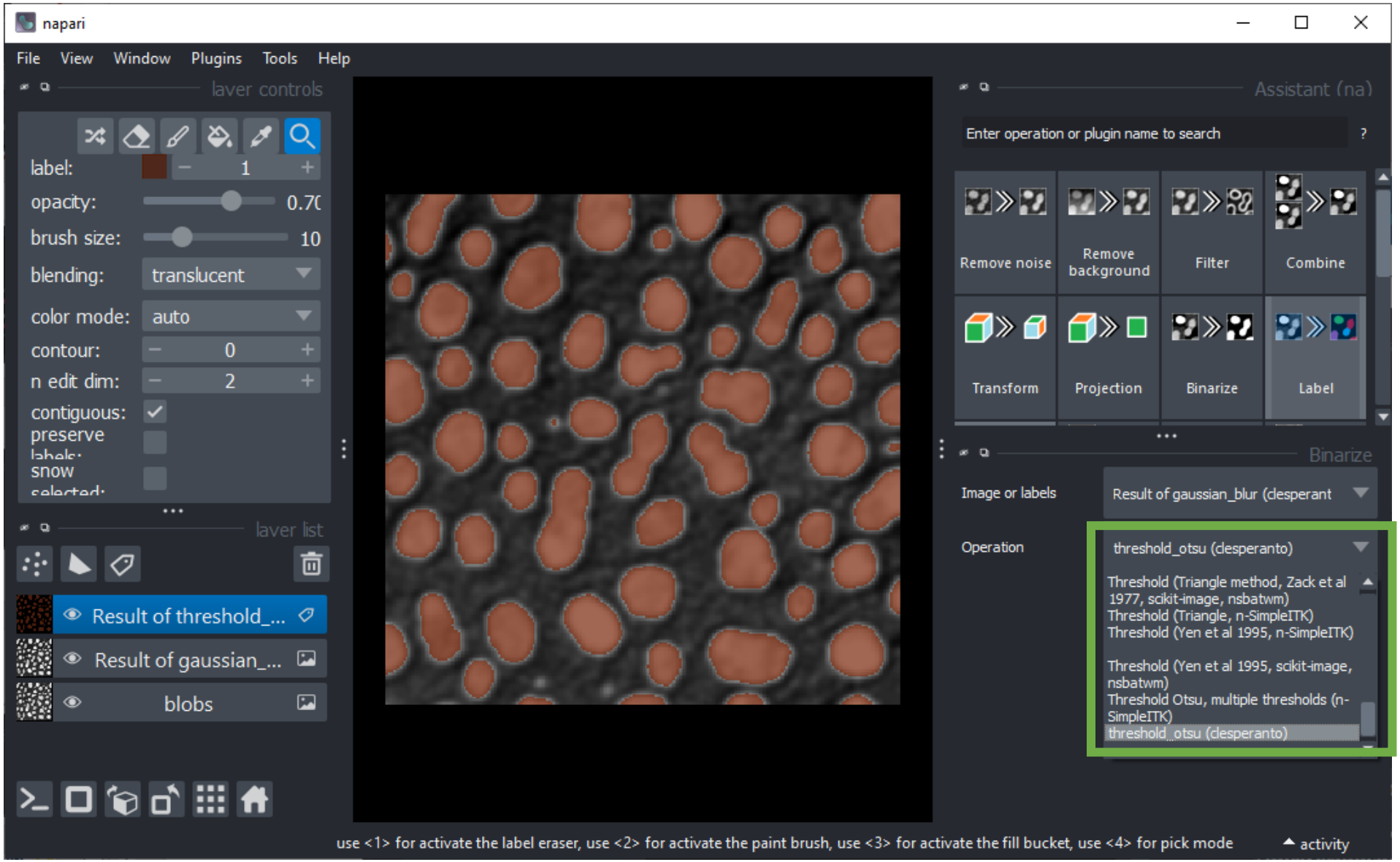
Workflow building

- Try different algorithms, e.g. for removing noise
- Find them in the pulldown



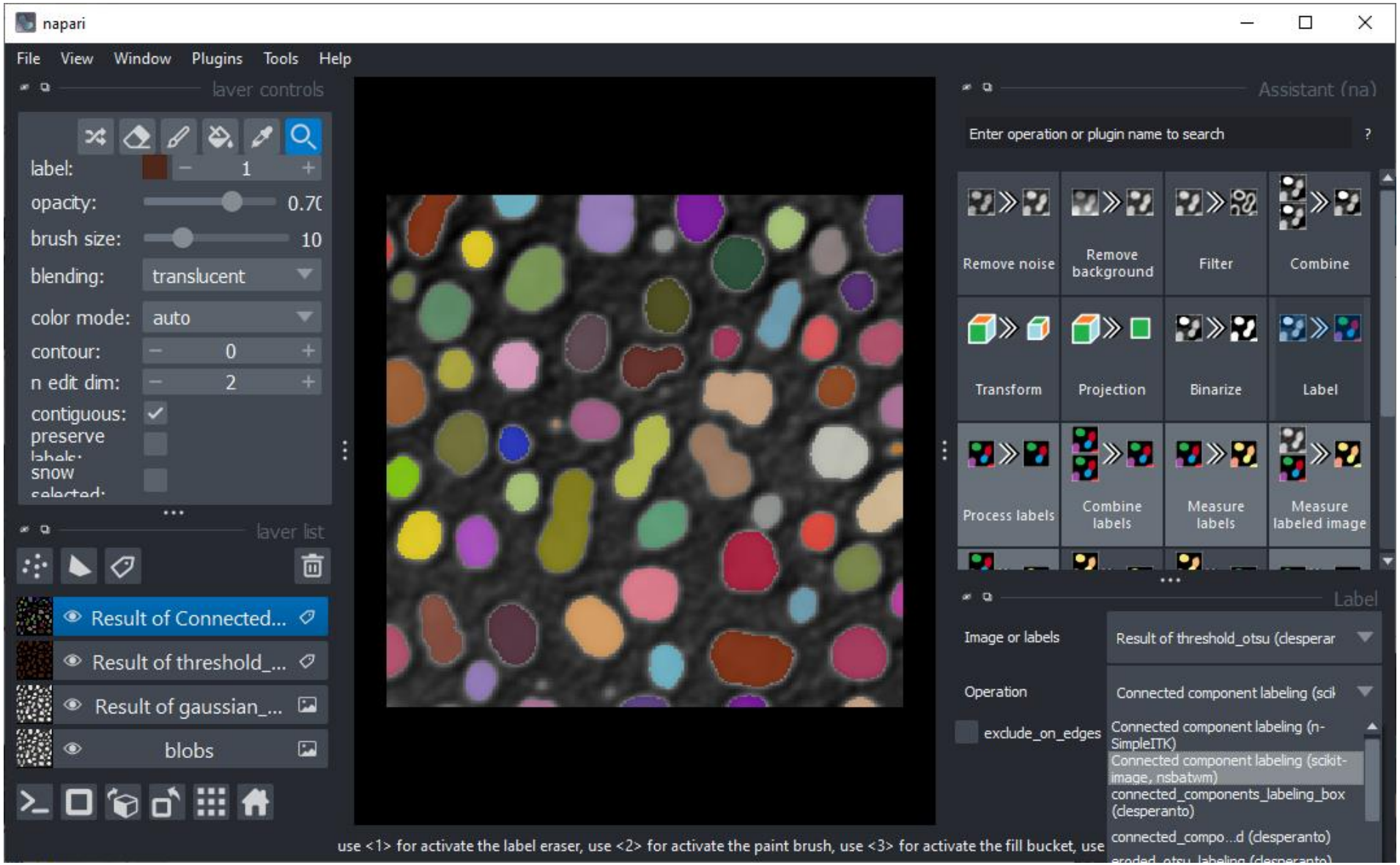
Workflow building

- Try different binarization algorithms



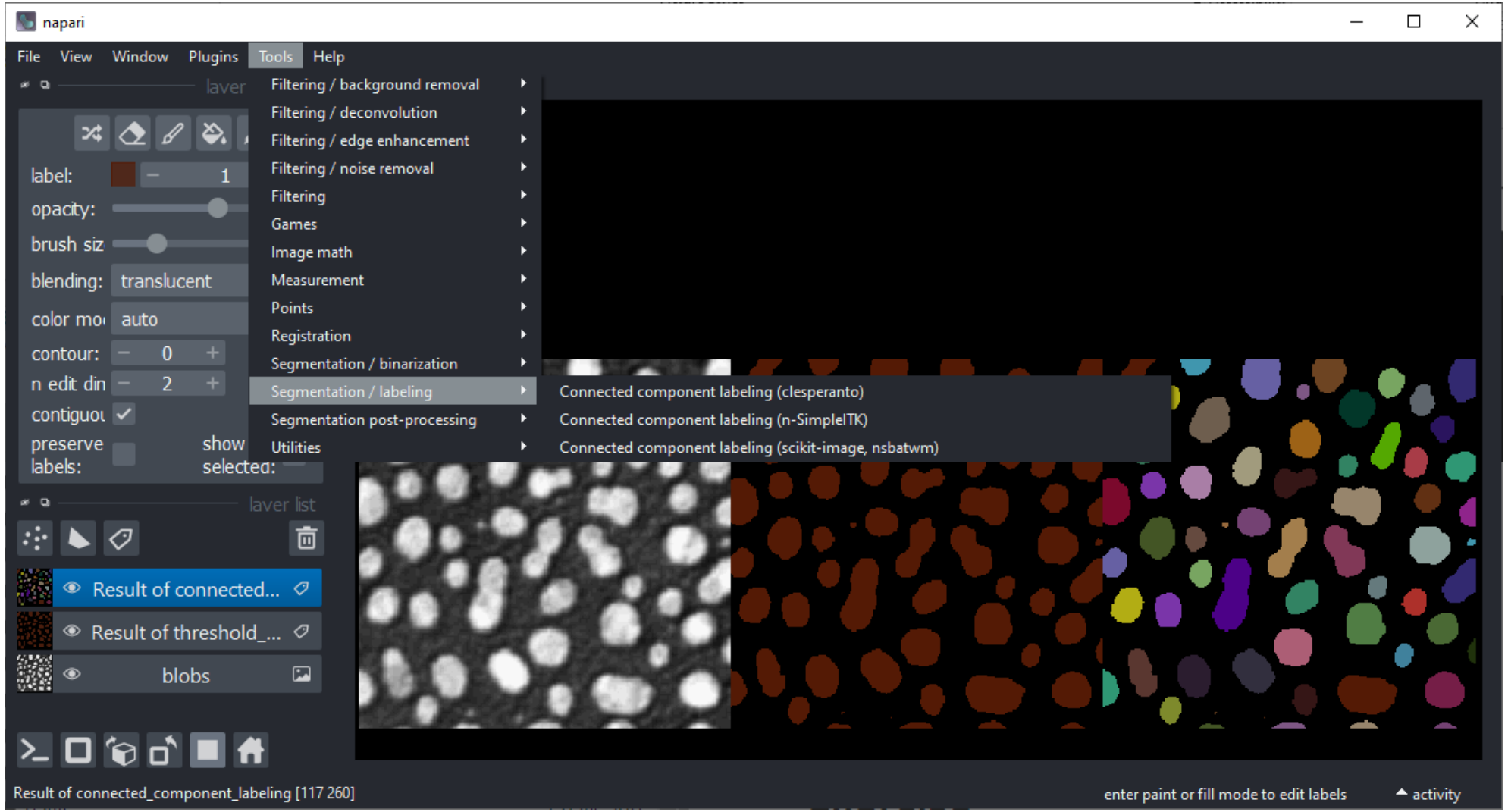
Workflow building

- Try different labeling algorithms



Workflow building

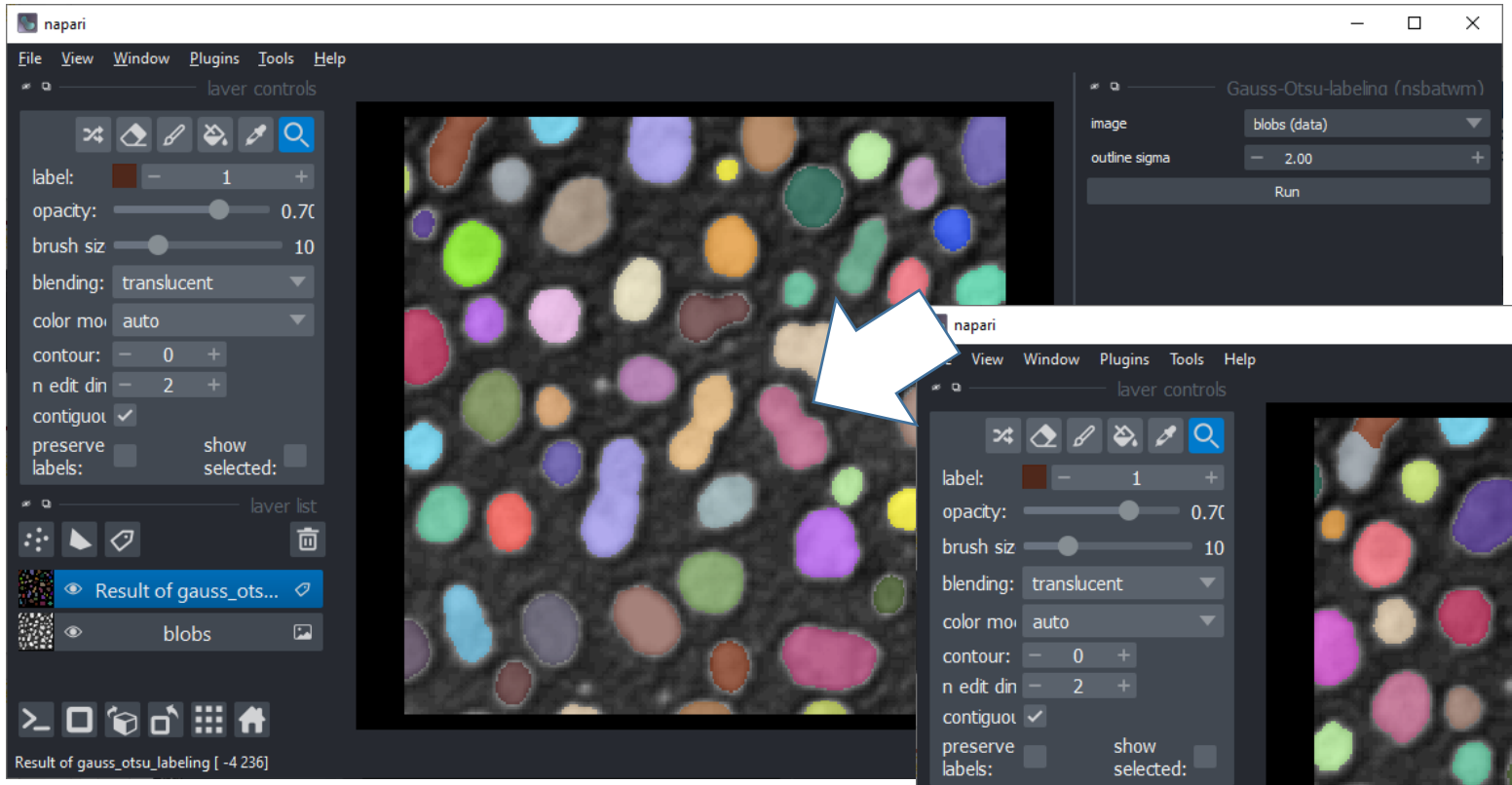
Also check out the Tools > Segmentation / labeling menu



Short-cuts: Voronoi-Otsu-Labeling

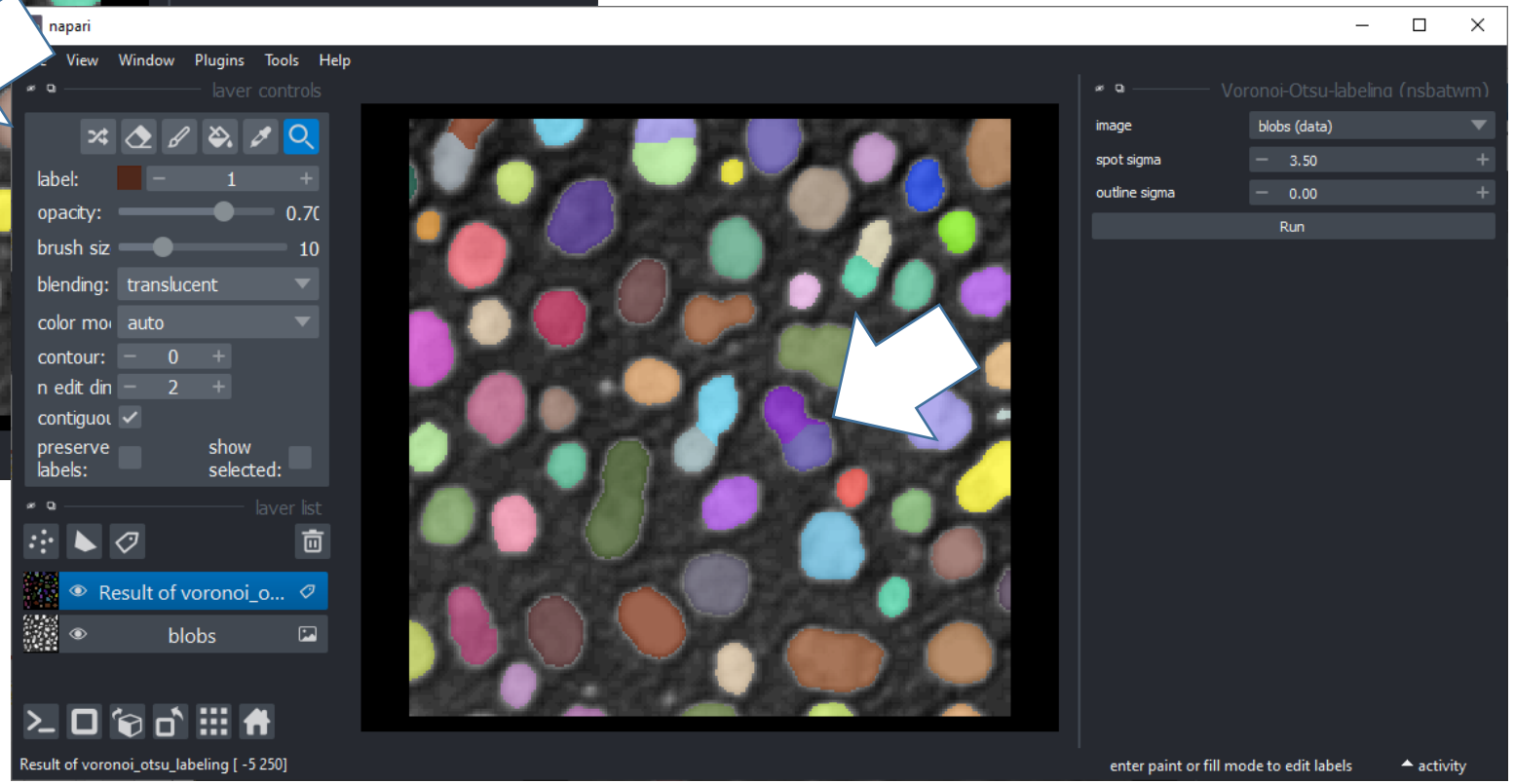
PoL Physics of Life TU Dresden

Also check out the Tools > Segmentation / labeling menu



Gauss-Otsu-Labeling

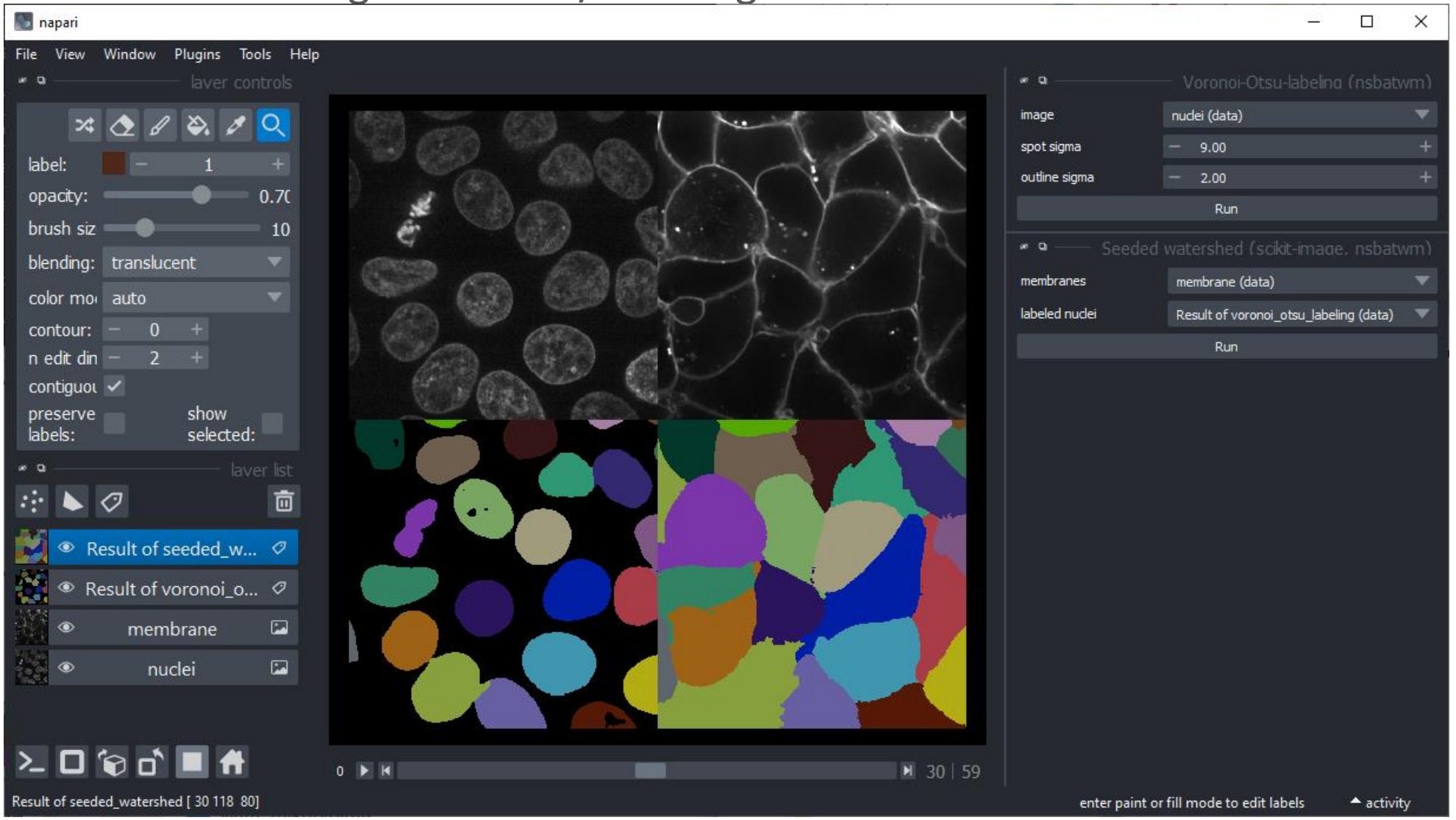
Voronoi-Otsu-Labeling



Watershed

PoL Physics of Life TU Dresden

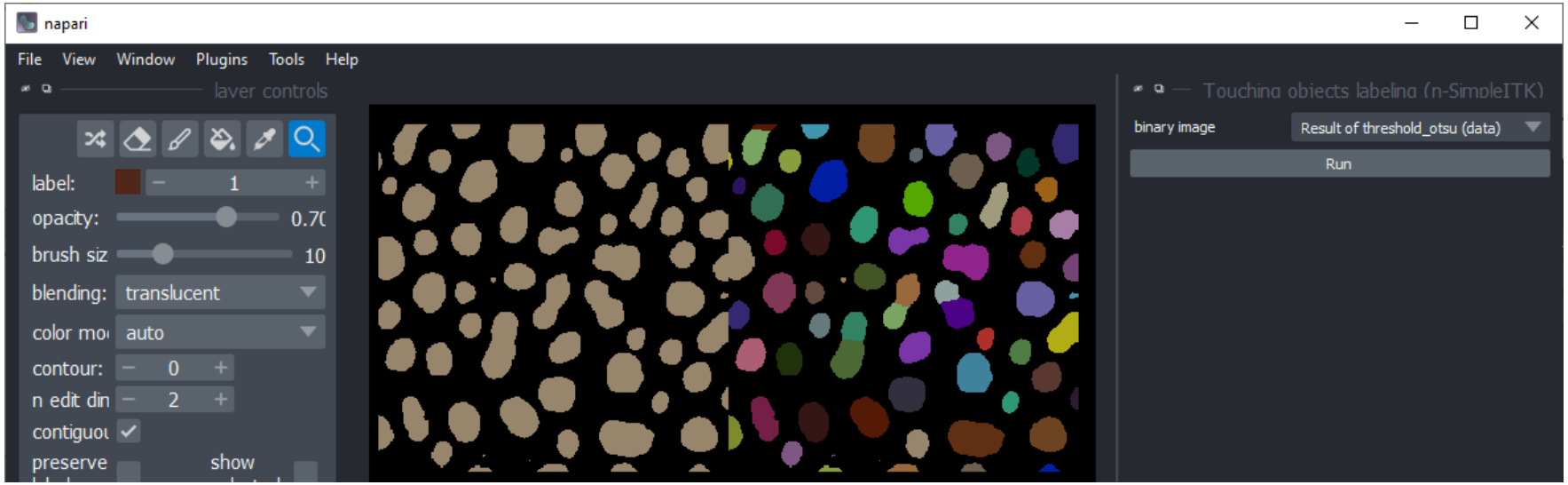
Also check out the Tools > Segmentation / labeling menu



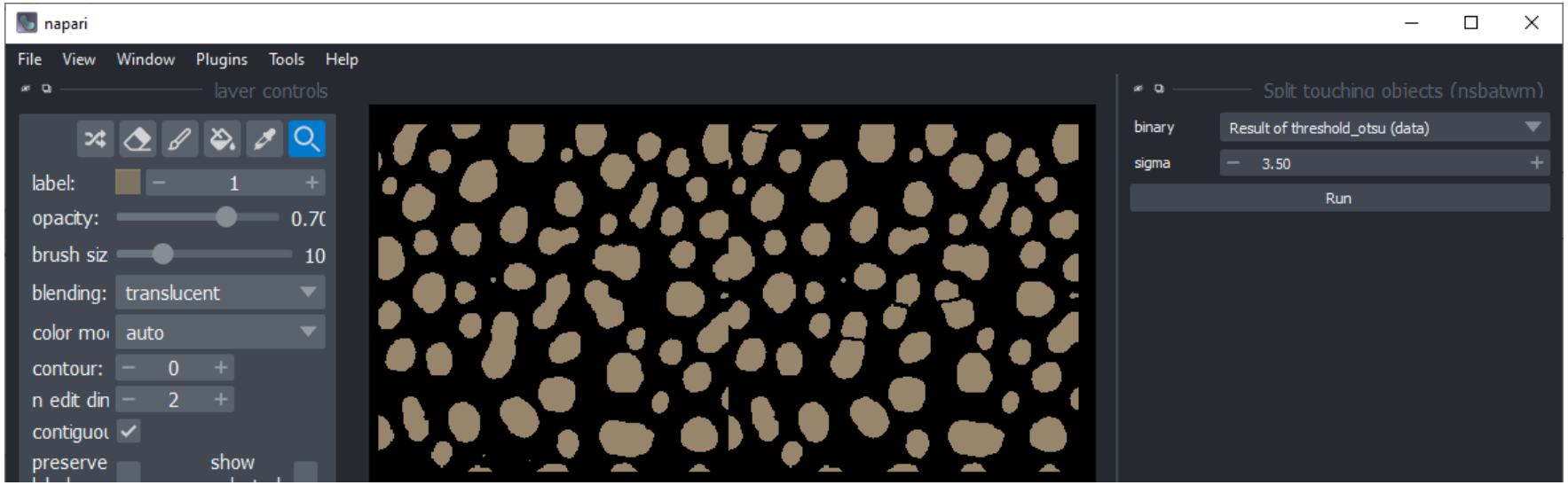
Watershed

- From binary images

Tools > Segmentation / labeling >
Label touching objects

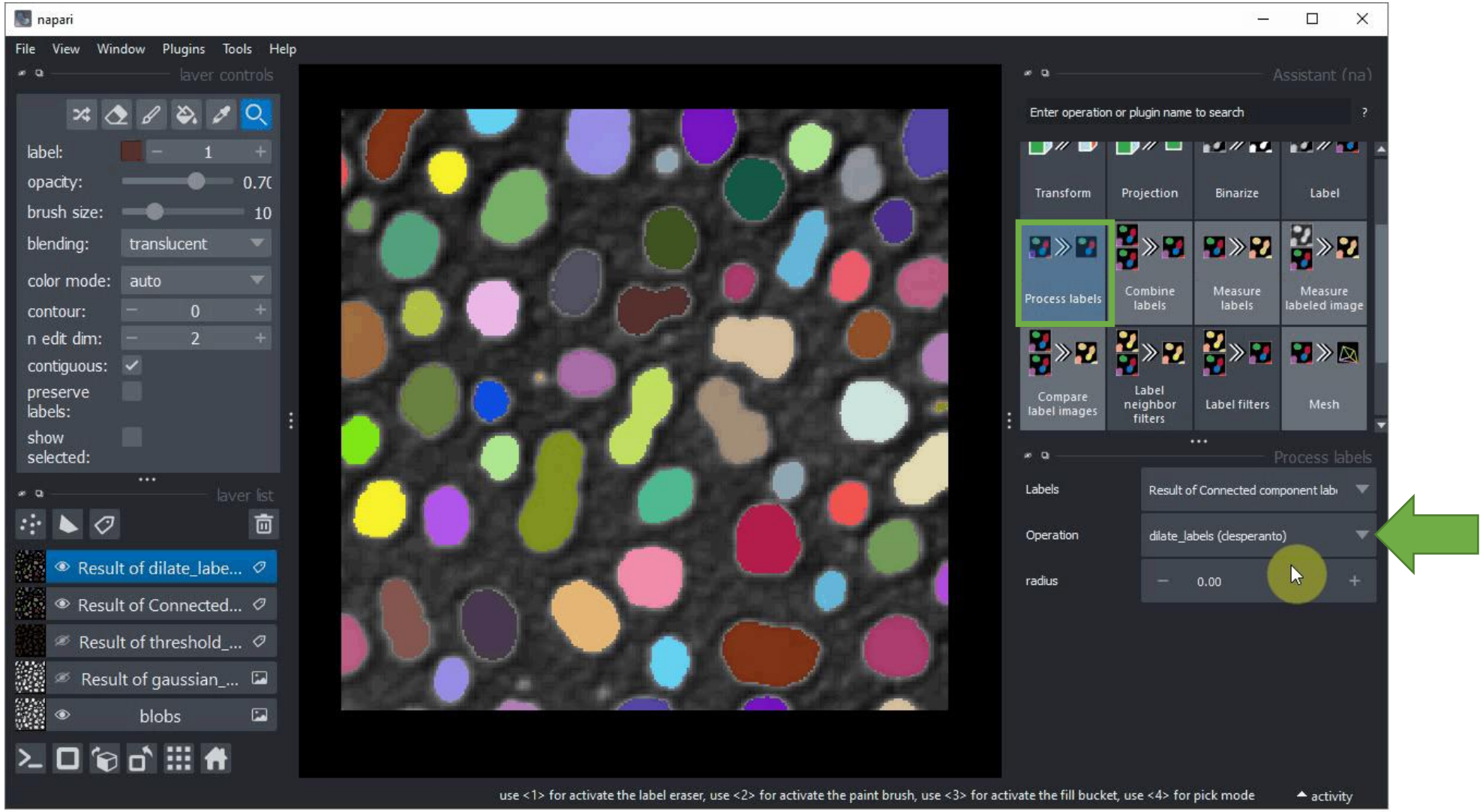


Tools > Segmentation post-processing >
Split touching objects
(Similar to ImageJ's Watershed)



Label erosion, dilation, opening, closing, …

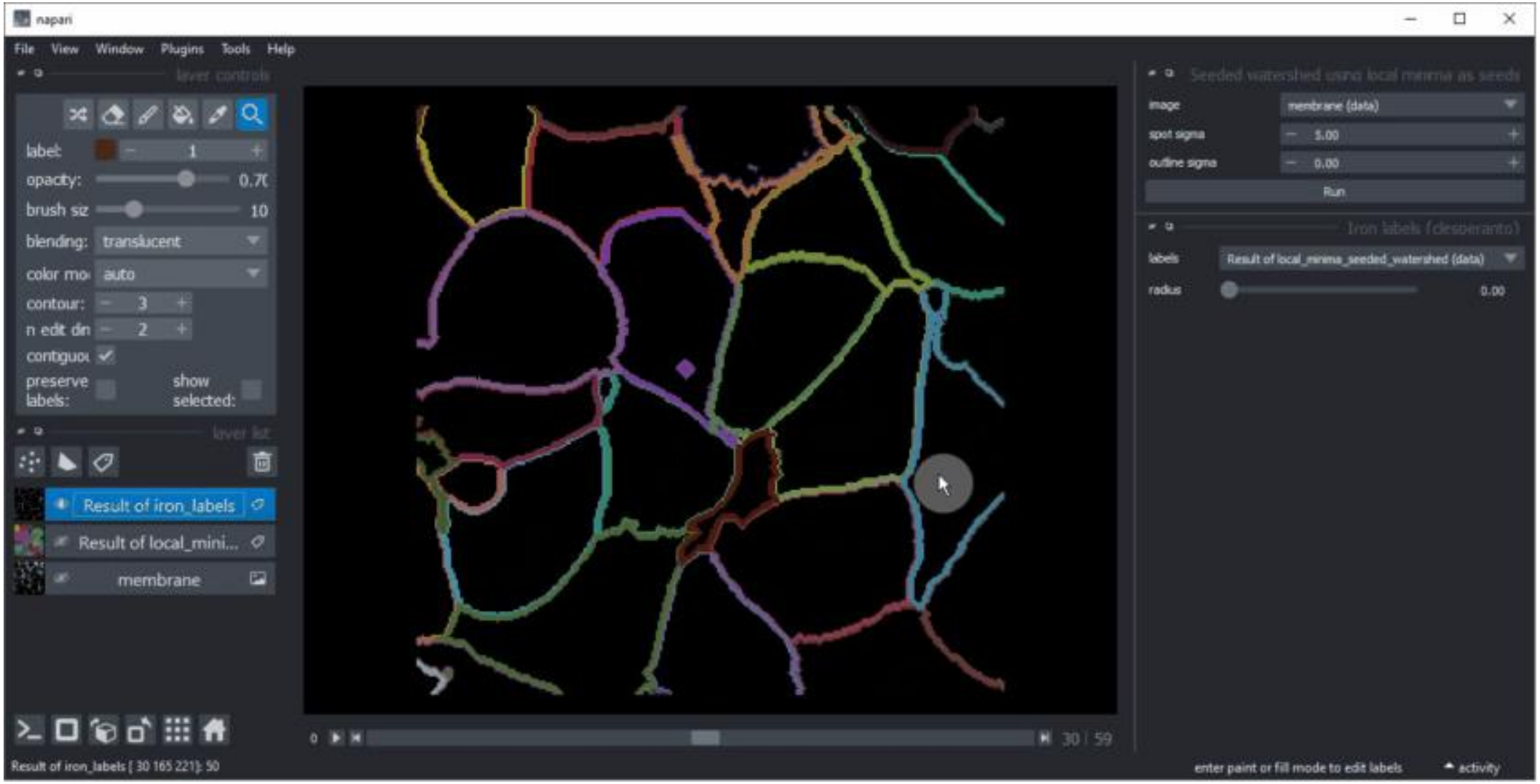
- In Napari Assistant: Process labels



Label post-processing / morphological operations

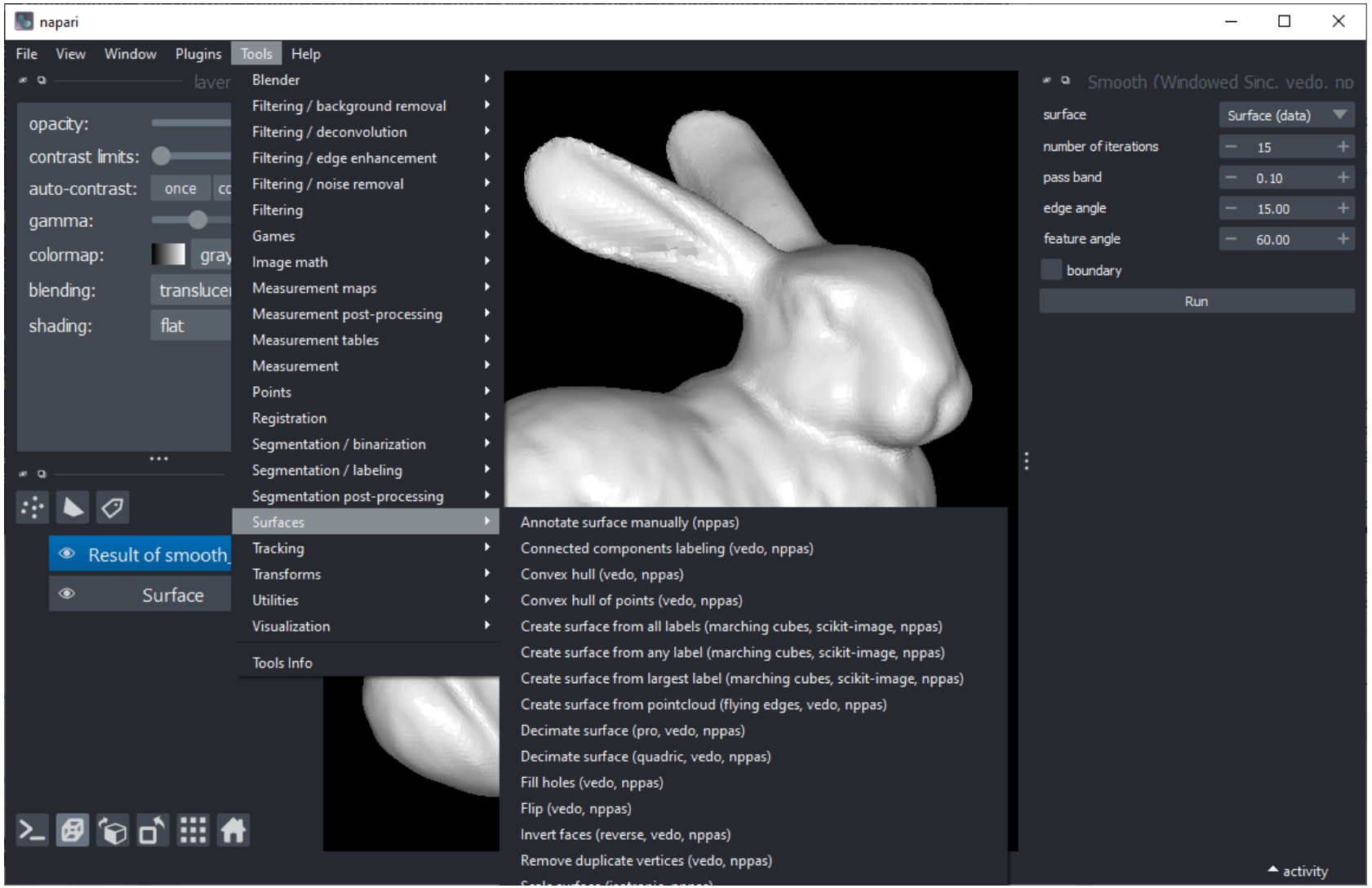
PoL Physics of Life TU Dresden

- In Napari menu Tools > Segmentation post-processing > Smooth labels (clEsperanto)



Surface reconstruction / Processing

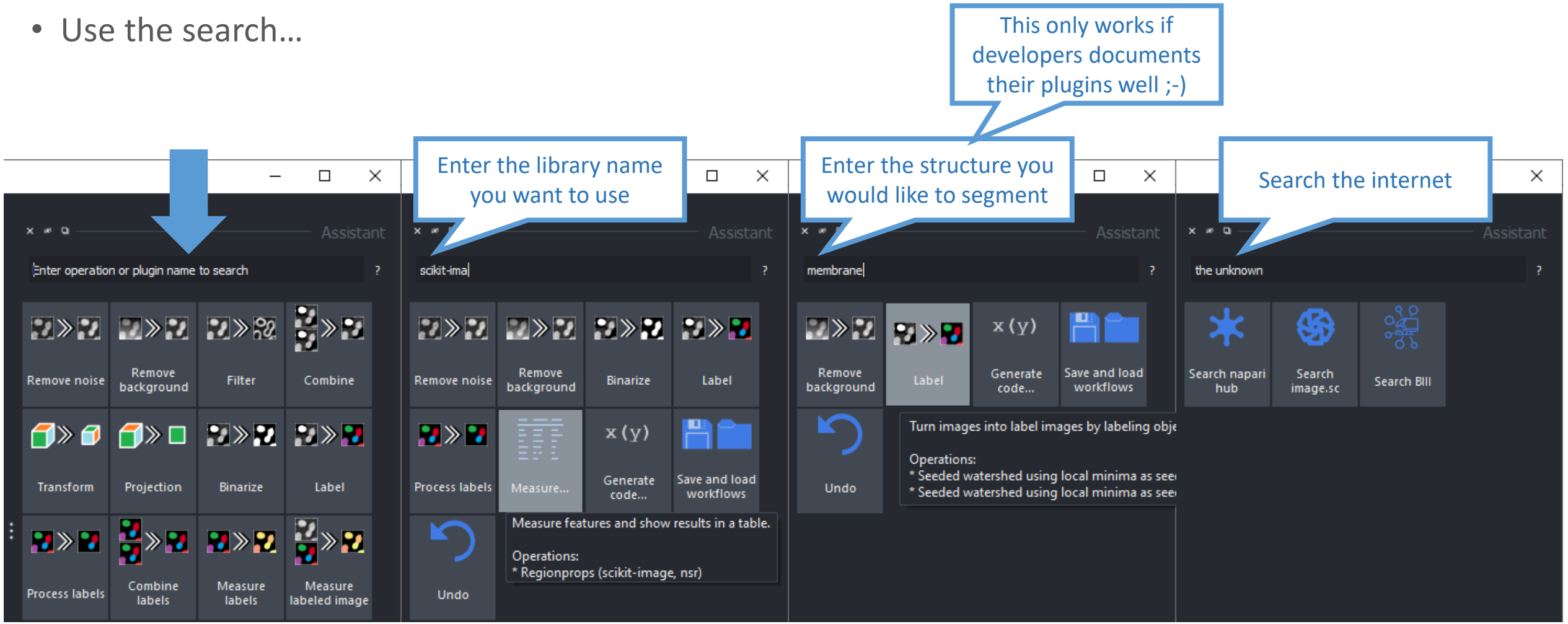
- Tools > Surfaces > Create surface …



You need to install an extra napari-plugin: https://github.com/haesleinhuepf/napari-process-points-and-surfaces

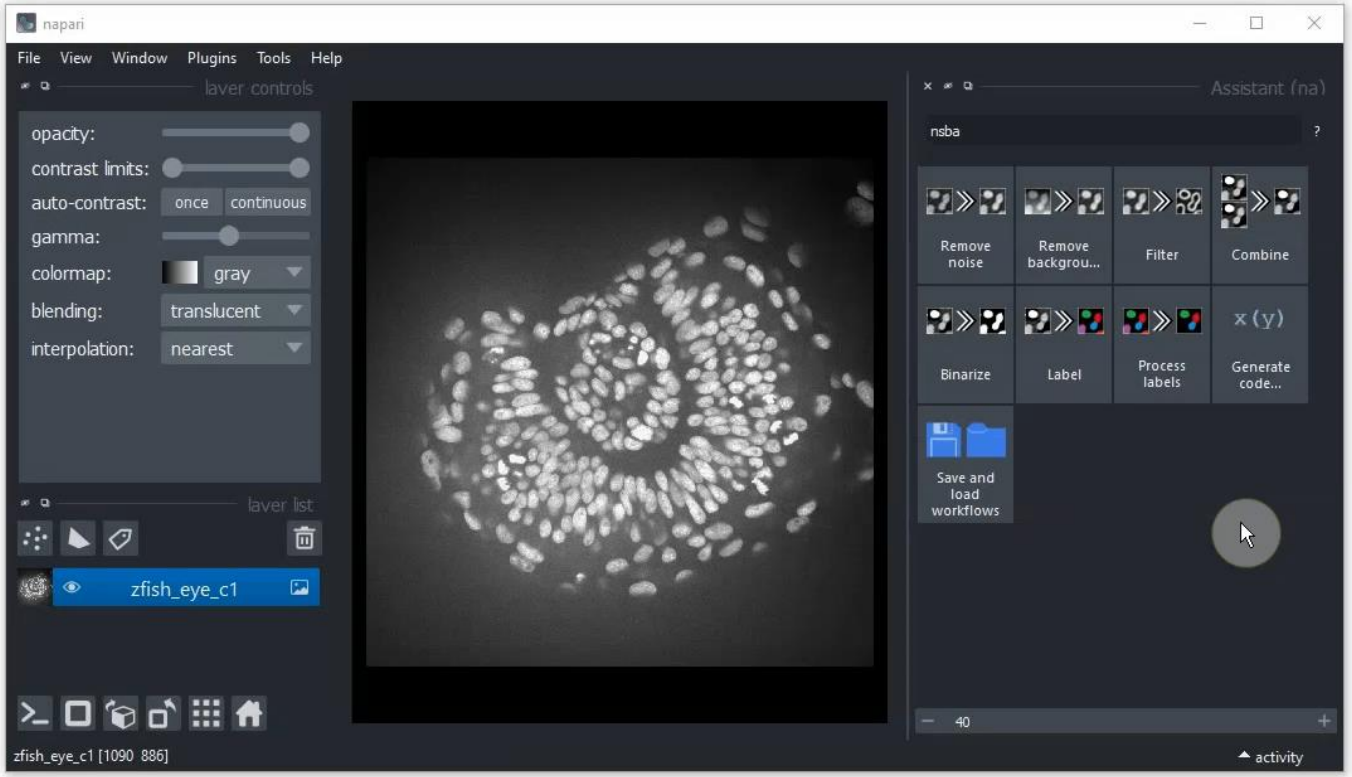
Browse operations

- Use the search…



https://www.napari-hub.org/plugins/napari-assistant

Export code to Jupyter Notebooks

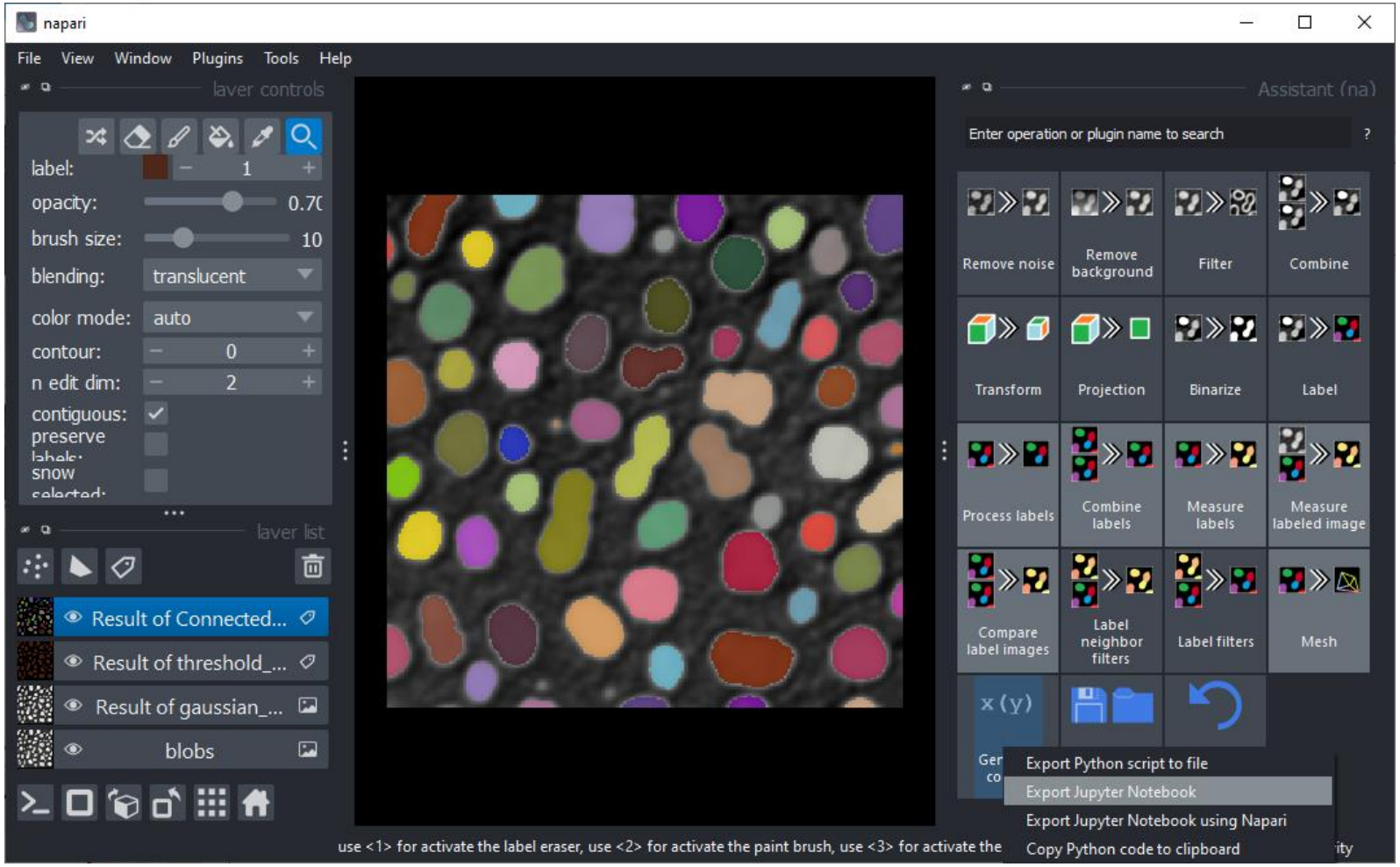


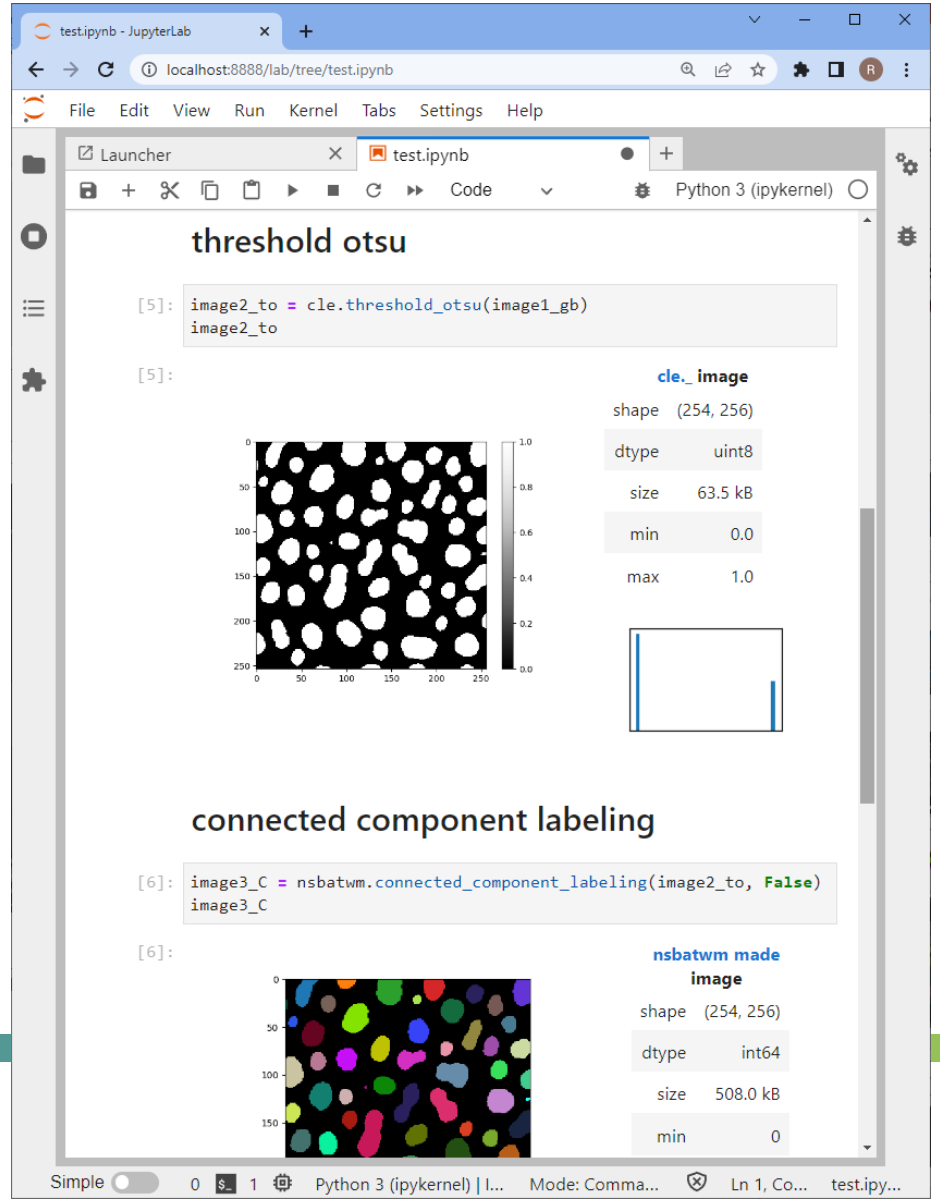
https://github.com/haesleinhuepf/napari-assistant

Image data source: Mauricio Rocha Martins, Norden lab, MPI CBG (now at IGC Oeiras)

Export code to Jupyter Notebooks

PoL Physics of Life TU Dresden





TECHNISCHE UNIVERSITÄT DRESDEN

https://github.com/haesleinhuepf/napari-assistant#code-generation

PoL Physics of Life TU Dresden

CENTER FOR SYSTEMS BIOLOGY DRESDEN

Image segmentation in Python

Robert Haase

April 2023

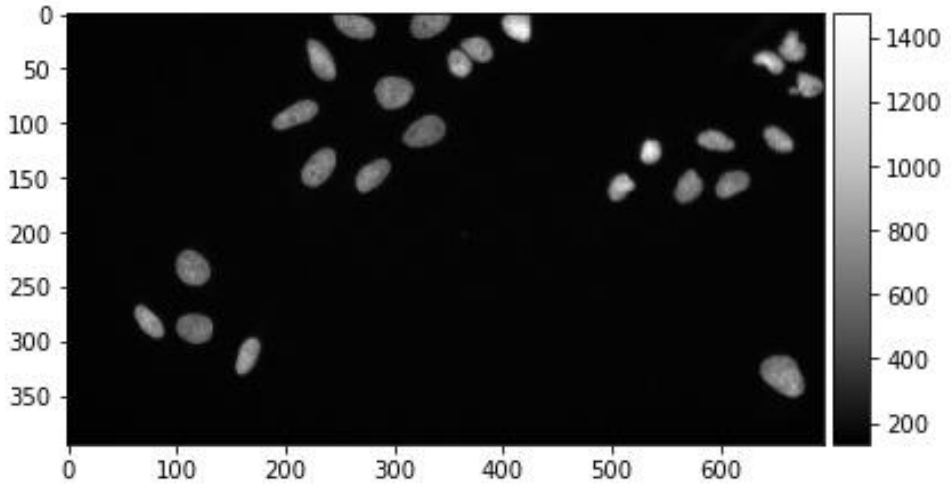
@haesleinhuepf

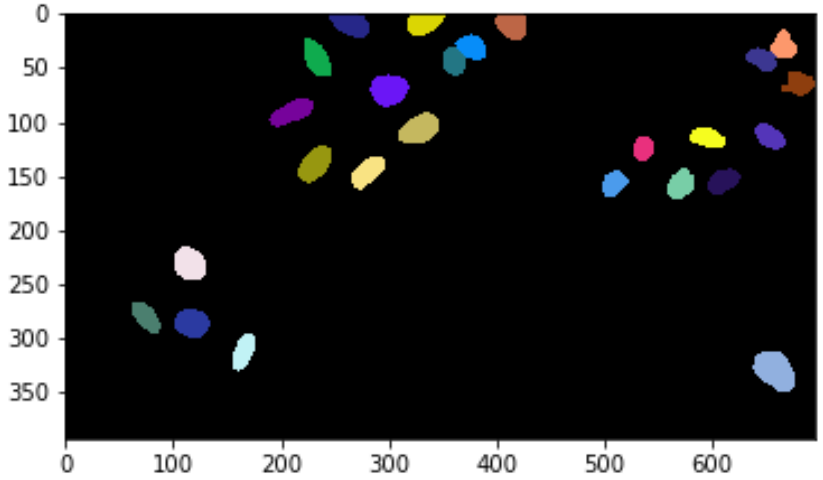
Voronoi-Otsu-Labeling

- 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

https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/20_image_segmentation/11_voronoi_otsu_labeling.html
https://www.youtube.com/watch?v=evgRgDfVuEc

Anisotropy

- Some [segmentation] algorithms have prerequisites…

```
[1]: import pyclesperanto_prototype as cle

[ ]: cle.voronoi_otsu_labeling(

[ ]:
```

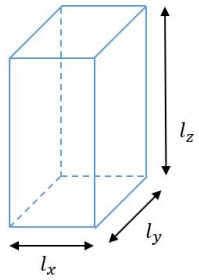
```
Docstring:
Labels objects directly from grey-value images.

The two sigma parameters allow tuning the segmentation result. Under the hood,
this filter applies two Gaussian blurs, spot detection, Otsu-thresholding [2] and Voronoi-labeling
[3]. The
thresholded binary image is flooded using the Voronoi tesselation approach starting from the found
local maxima.

Notes
-----
* This operation assumes input images are isotropic.

Parameters
----------
source : Image
    Input grey-value image
label_image_destination : Image, optional
    Output image
spot_sigma : float, optional
```

Anisotropy

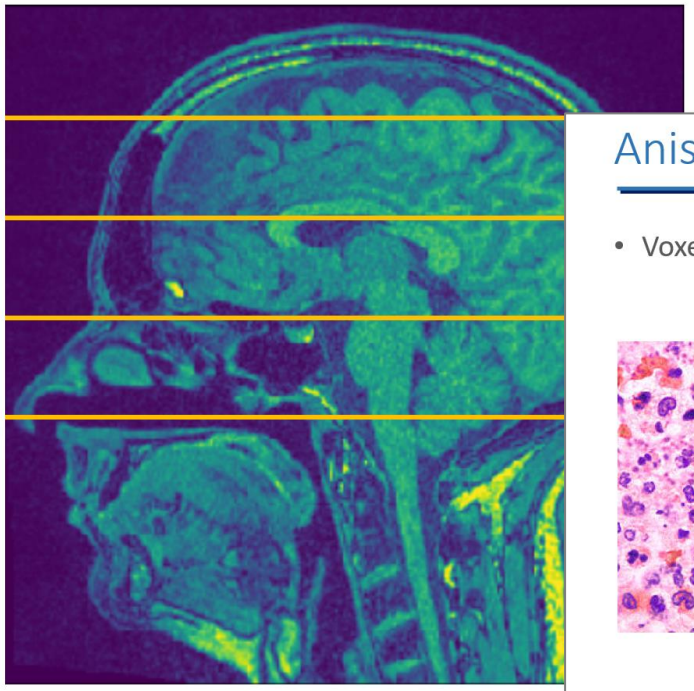
- Reminder: Anisotropic images might be tricky to process properly

Image stacks and voxels

- 3-dimensional images consisting of voxels
- "Image stack"
- Often anisotropic (not equally large in all directions)



$$l_x = l_y \neq l_z$$

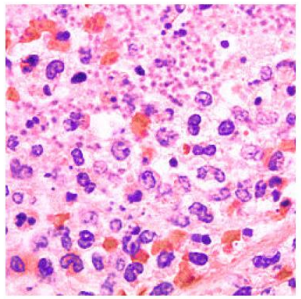


@haesleinhuepf

April 2023

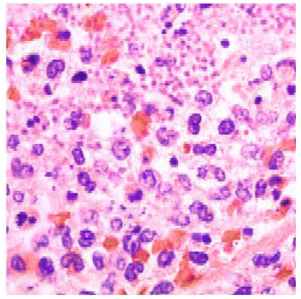
Anisotropy

- Voxel size has immediate impact on image quality and thus, on processing / analysis results.



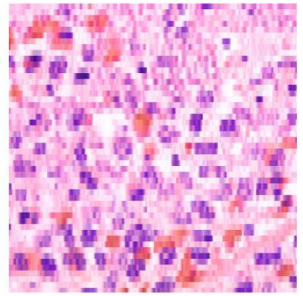
1:1

250 x 250 px



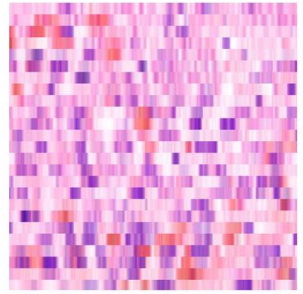
1:2

250 x 125 px



1:5

250 x 50 px



1:10

250 x 25 px

@haesleinhuepf

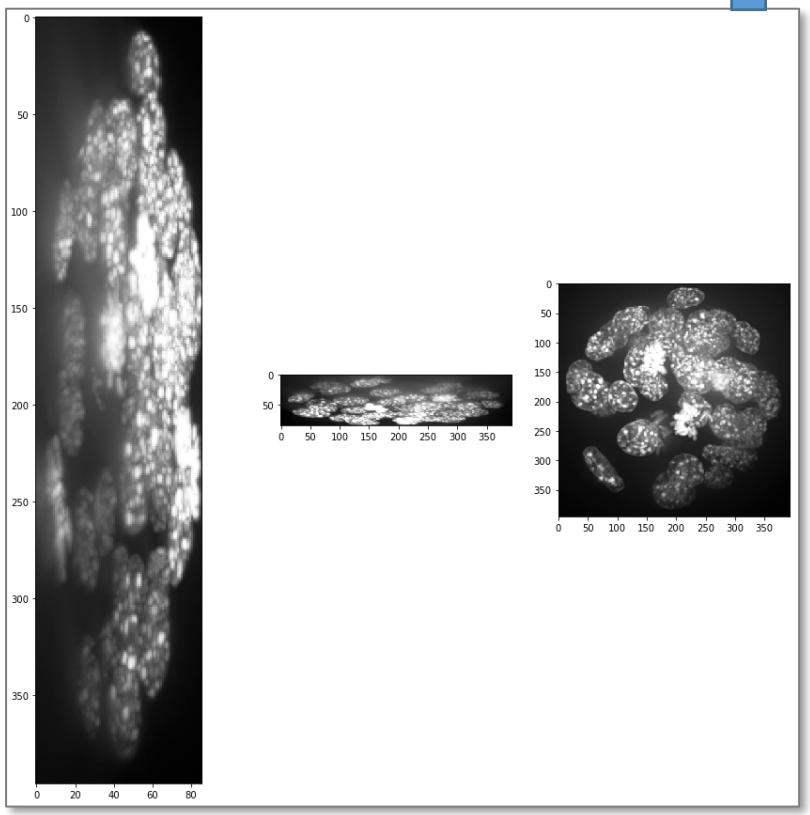
April 2023

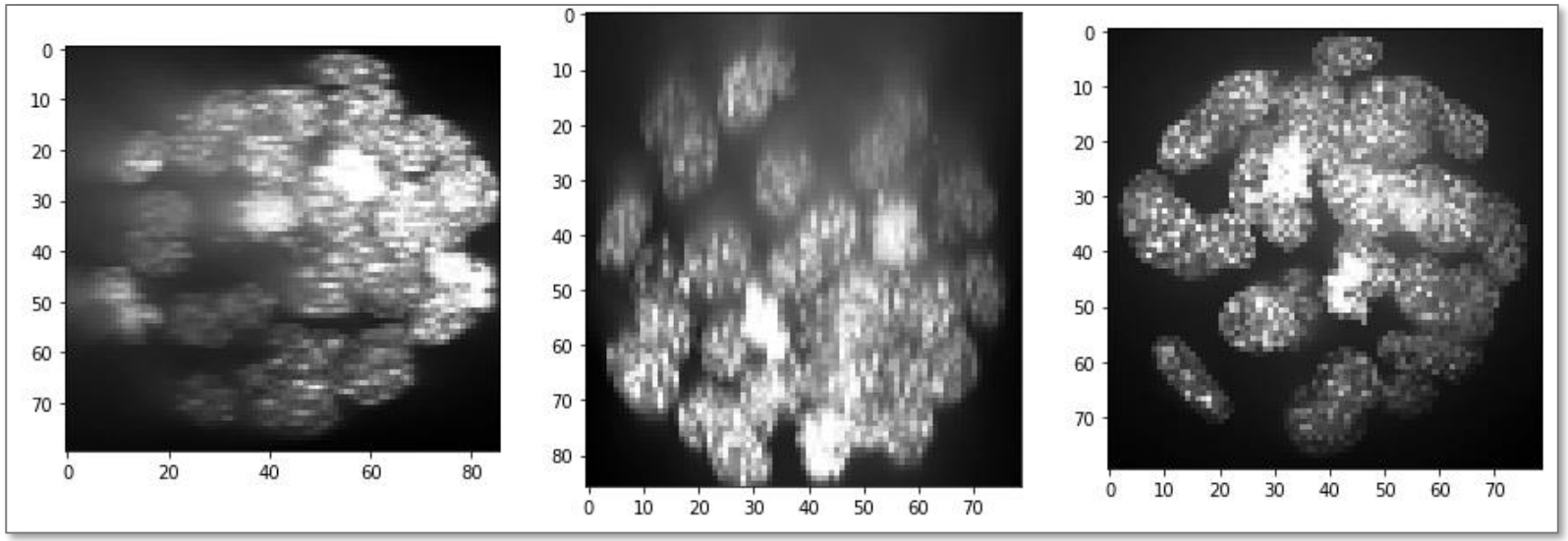
Image source: cropped from https://de.m.wikipedia.org/wiki/Datei:Histo_Lungenpest.jpg

Reslicing / scaling / sampling

- Resample image data to a specific voxel size

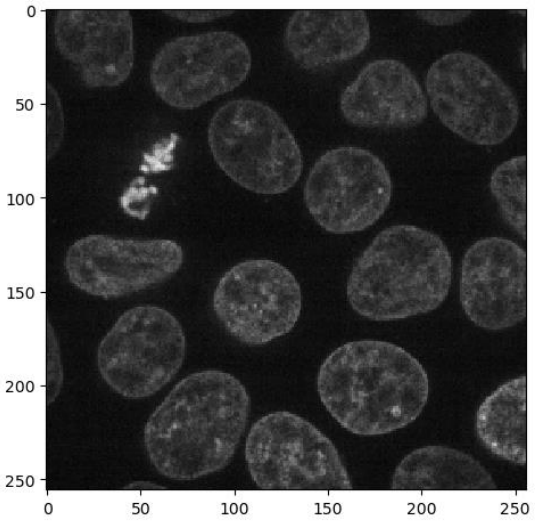
```
resampled = cle.scale(input_image, factor_x=voxel_size_x, factor_y=voxel_size_y, factor_z=voxel_size_z, auto_size=True)

show(resampled)
```

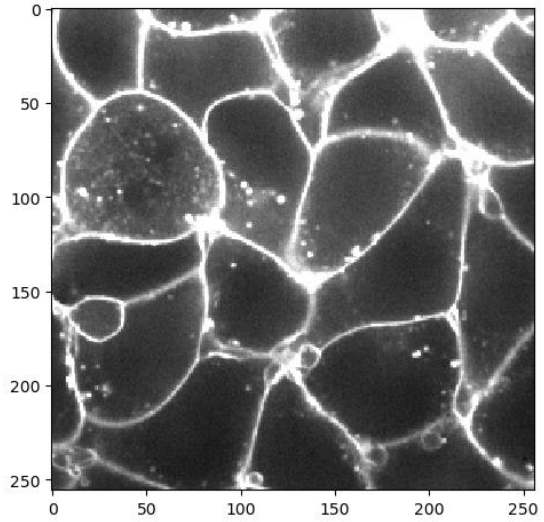


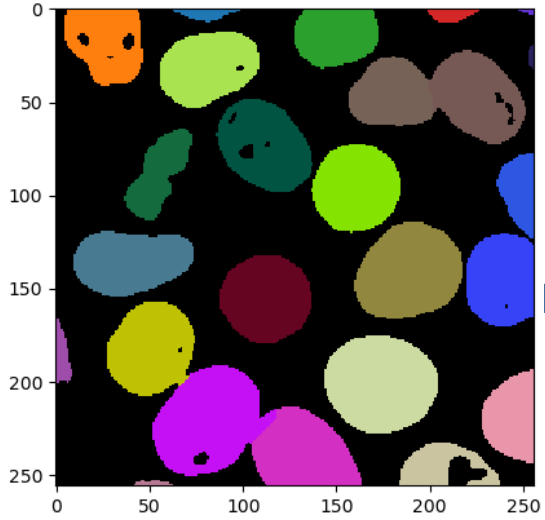


Watershed

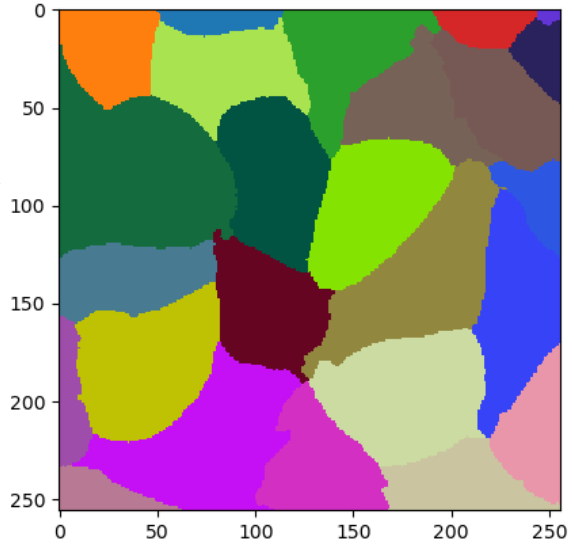
- … in Python practice







```
labeled_cells = seeded_watershed(membrane_channel, labeled_nuclei)

labeled_cells
```

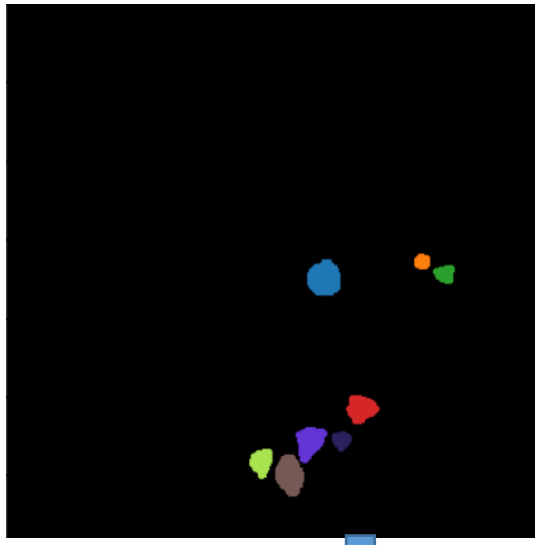


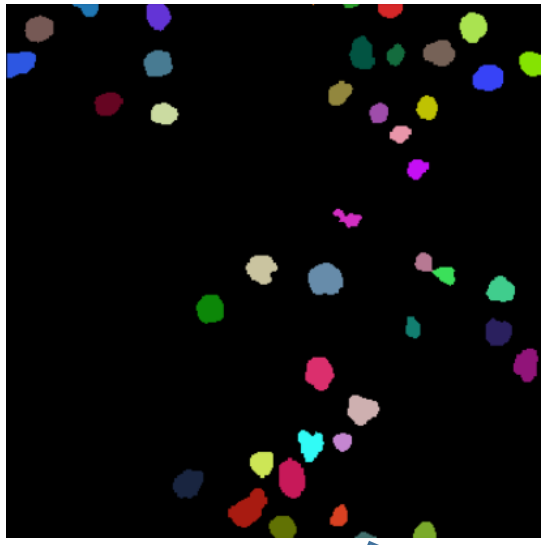
https://www.napari-hub.org/plugins/napari-segment-blobs-and-things-with-membranes#seeded-watershed

Segmentation quality estimation

- Compare annotations with algorithm results

Sparse instance annotation





Instance segmentation

```
from the_segmentation_game import metrics
```

```
[10]: metrics.jaccard_index_sparse(annotation, labels)
```

```
[10]: 0.8357392602053431
```

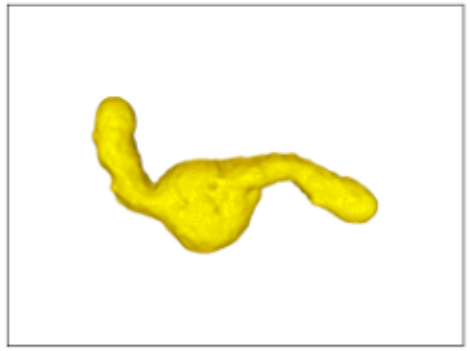
https://github.com/haesleinhuepf/the-segmentation-game#segmentation-algorithm-comparison

Surface reconstruction

- Turn binary and/or label images into surface meshes

```
surface = nppas.all_labels_to_surface(binary_filled)
```

```
surface
```



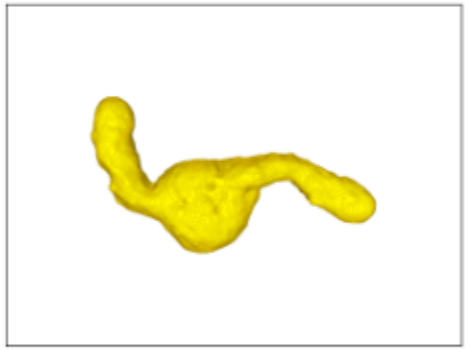
nppas.SurfaceTuple

origin (z/y/x)	[0. 0. 0.]
center of mass(z/y/x)	57.710,309.963,440.042
scale(z/y/x)	1.000,1.000,1.000
bounds (z/y/x)	12.500...113.500 111.500...461.500 169.500...807.500
average size	170.769
number of vertices	330776
number of faces	661548

Surface mesh processing

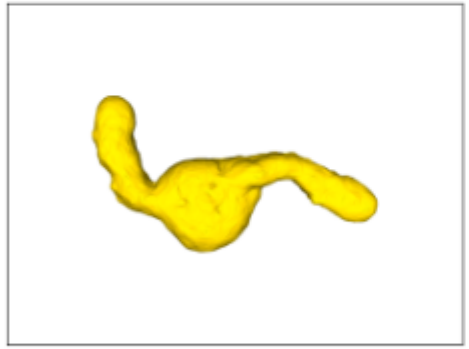
- Surface mesh simplification
- To prevent the computer freezing

```
simplified_surface = nppas.decimate_quadric(surface, fraction=0.01)
simplified_surface
```



nppas.SurfaceTuple

origin (z/y/x)	[0. 0. 0.]
center of mass(z/y/x)	57.710,309.963,440.042
scale(z/y/x)	1.000,1.000,1.000
bounds (z/y/x)	12.500...113.500 111.500...461.500 169.500...807.500
average size	170.769
number of vertices	330776
number of faces	661548



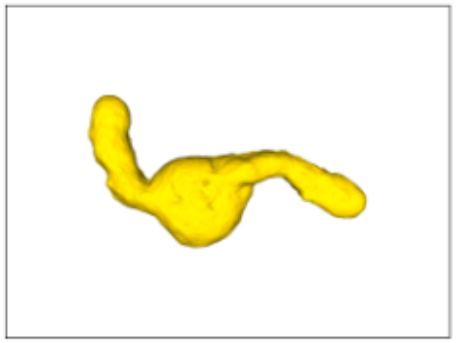
nppas.SurfaceTuple

origin (z/y/x)	[0. 0. 0.]
center of mass(z/y/x)	57.928,308.938,440.985
scale(z/y/x)	1.000,1.000,1.000
bounds (z/y/x)	13.231...113.510 111.642...461.602 170.022...806.468
average size	170.083
number of vertices	3310
number of faces	6615

Surface mesh processing

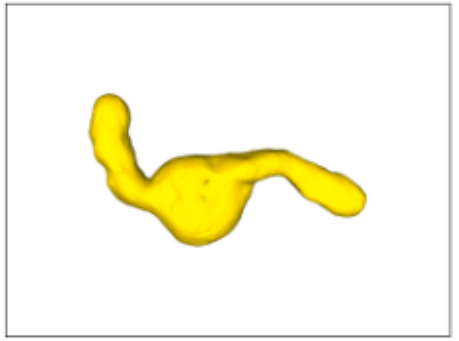
- Surface mesh smoothing

```
smoothed_surface = nppas.smooth_surface(simplified_surface)
smoothed_surface
```



nppas.SurfaceTuple

origin (z/y/x)	[0. 0. 0.]
center of mass(z/y/x)	57.928,308.938,440.985
scale(z/y/x)	1.000,1.000,1.000
bounds (z/y/x)	13.231...113.510 111.642...461.602 170.022...806.468
average size	170.083
number of vertices	3310
number of faces	6615



nppas.SurfaceTuple

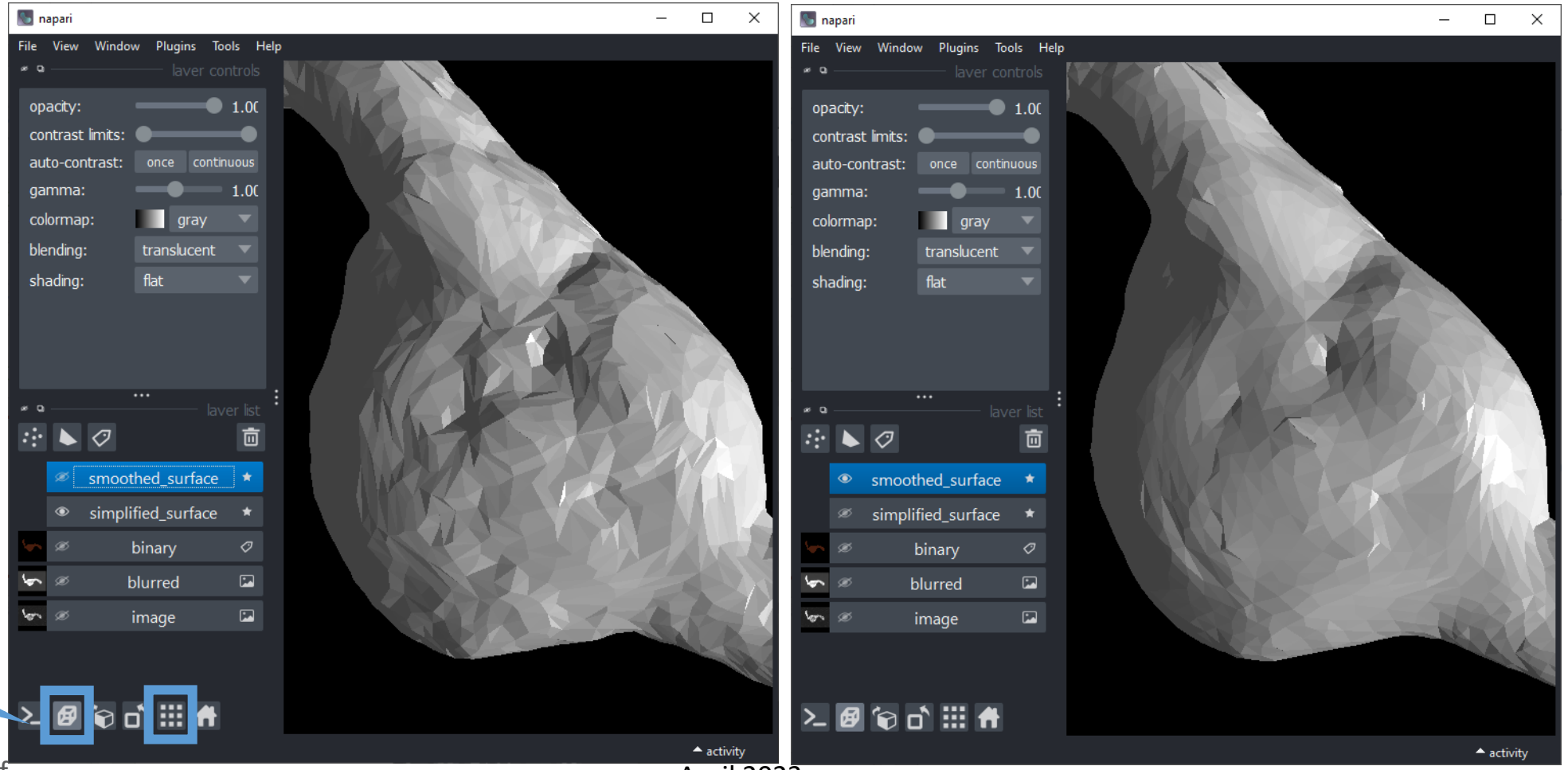
origin (z/y/x)	[0. 0. 0.]
center of mass(z/y/x)	57.913,308.988,440.878
scale(z/y/x)	1.000,1.000,1.000
bounds (z/y/x)	13.901...113.627 110.982...461.191 169.711...807.193
average size	170.378
number of vertices	3310
number of faces	6615

View surface meshes in Napari

PoL Physics of Life TU Dresden

```
viewer.add_surface(surface, scale=[zoom, zoom, zoom])
viewer.add_surface(simplified_surface, scale=[zoom, zoom, zoom])
viewer.add_surface(smoothed_surface, scale=[zoom, zoom, zoom])
```

In case your computer freezes, comment out this line



Toggle 3D view and grid mode

PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Exercises

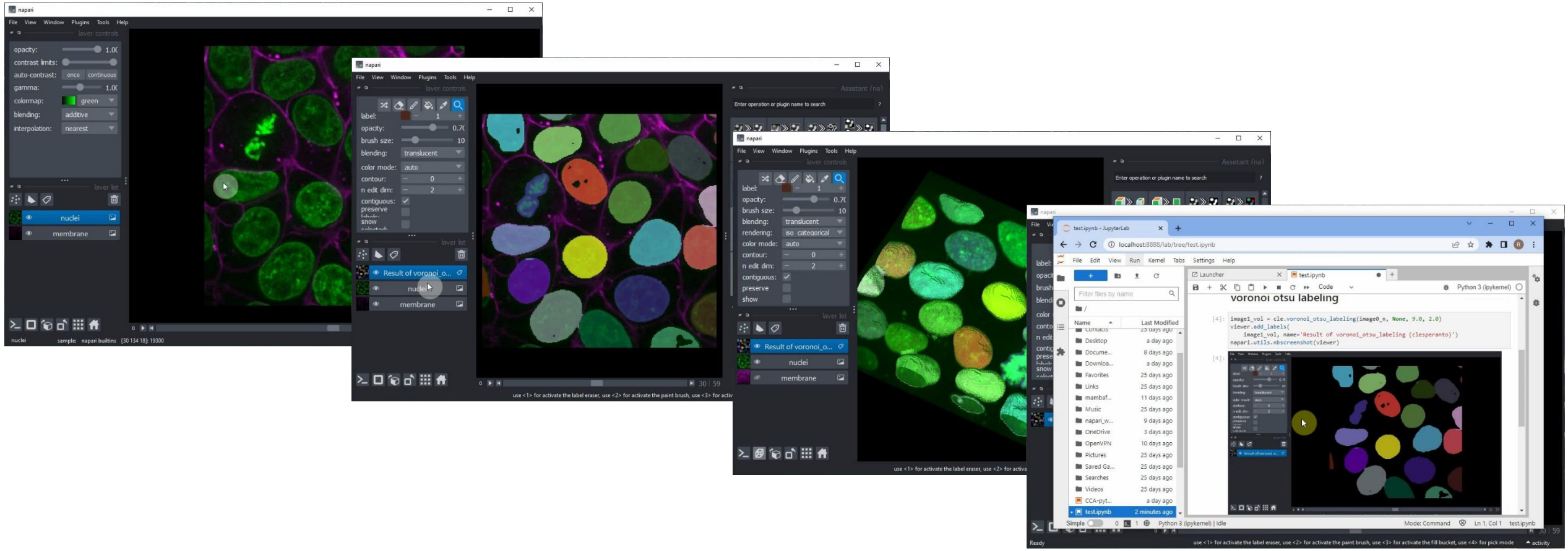
Robert Haase

April 2023

@haesleinhuepf

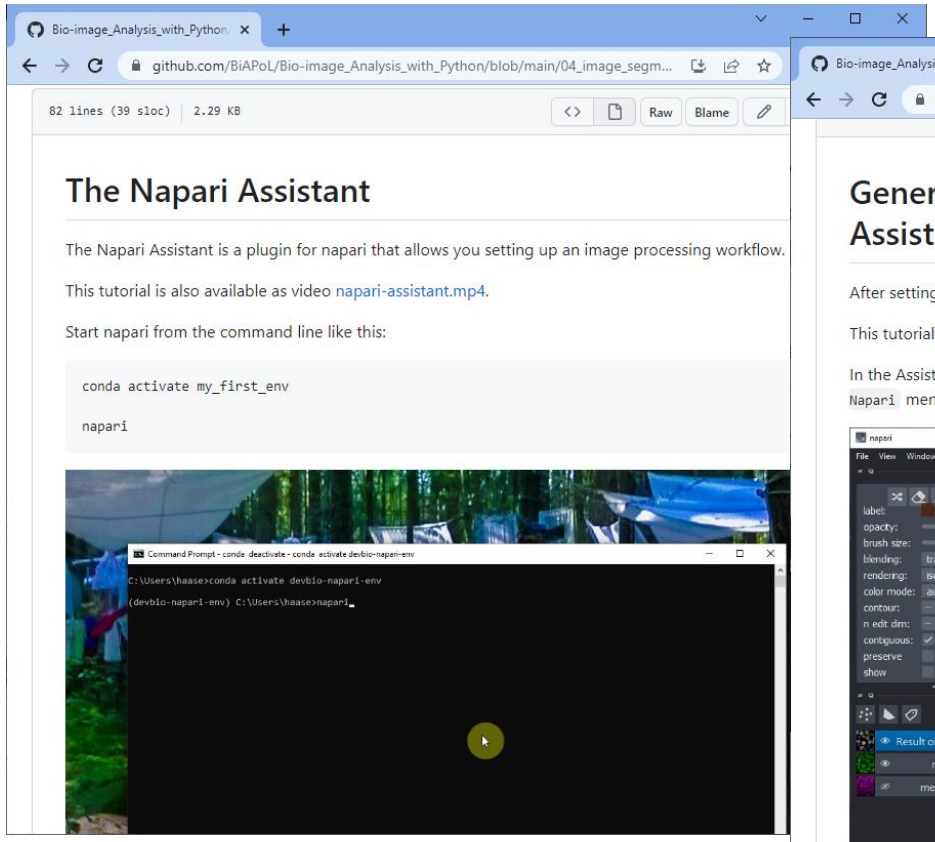
Exercise

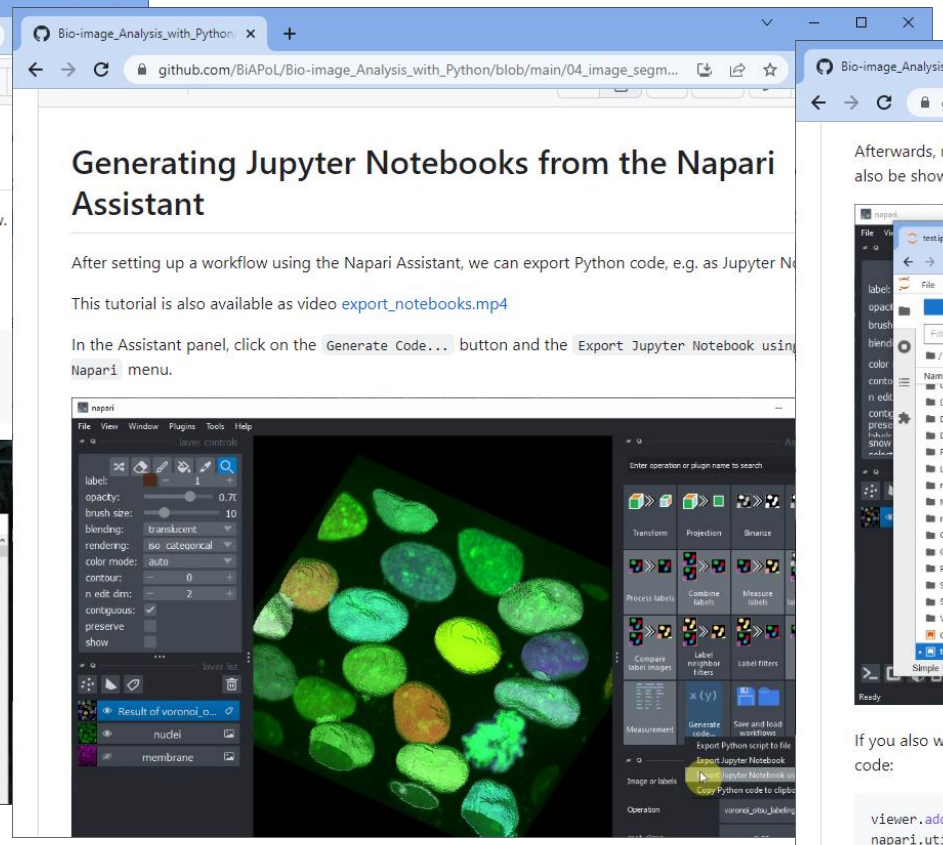
- Use the Napari Assistant to generate a Jupyter Notebook

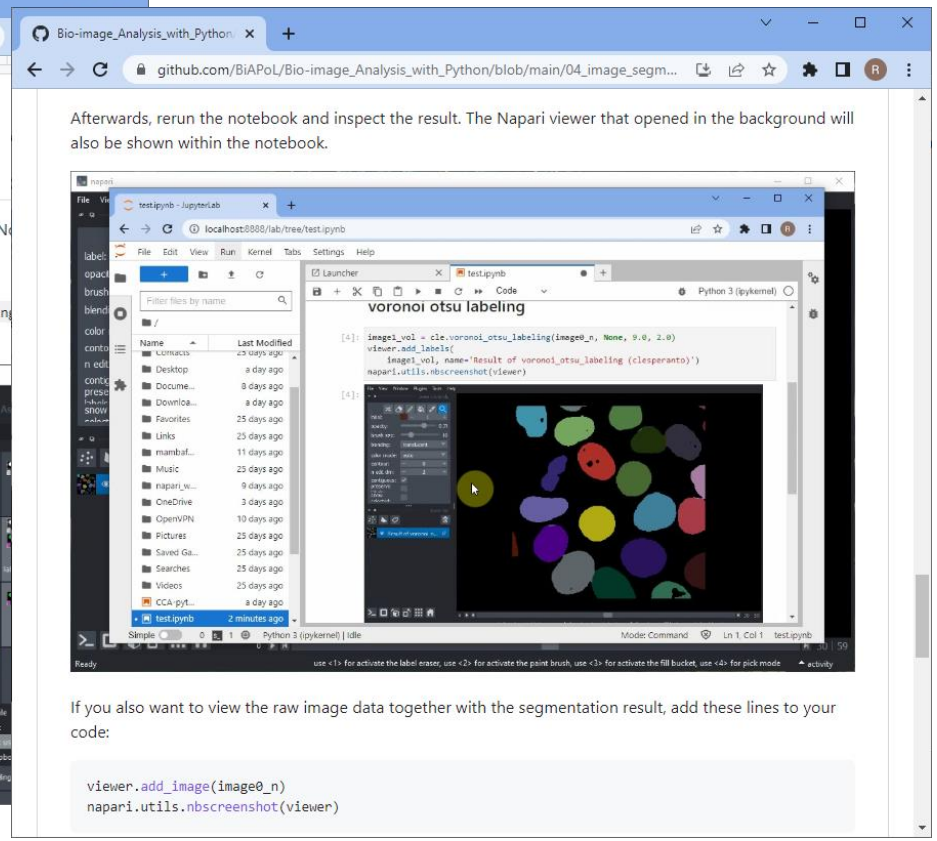


Exercise

- Follow the online instructions
- https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/04_image_segmentation/06_napari-assistant.md
- https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/04_image_segmentation/07_notebook_export.md







Exercise

- Measure the quality of a segmentation algorithm applied to a folder of images.
- https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/04_image_segmentation/17_segmentation_quality_estimation.ipynb

```
metrics.jaccard_index_sparse(sparse_labels, labels)
```

```
0.8357392602053431
```

```
for image_filename in os.listdir(image_folder):
    print(image_folder + image_filename)
```

```
../../data/BBBC007_batch/17P1_POS0013_D_1UL.tif
../../data/BBBC007_batch/20P1_POS0005_D_1UL.tif
../../data/BBBC007_batch/20P1_POS0007_D_1UL.tif
../../data/BBBC007_batch/20P1_POS0010_D_1UL.tif
../../data/BBBC007_batch/A9 p7d.tif
../../data/BBBC007_batch/AS_09125_040701150004_A02f00d0.tif
```