ScaDS.AI
DRESDEN LEIPZIG
CENTER FOR SCALABLE DATA ANALYTICS
AND ARTIFICIAL INTELLIGENCE

Surface reconstruction

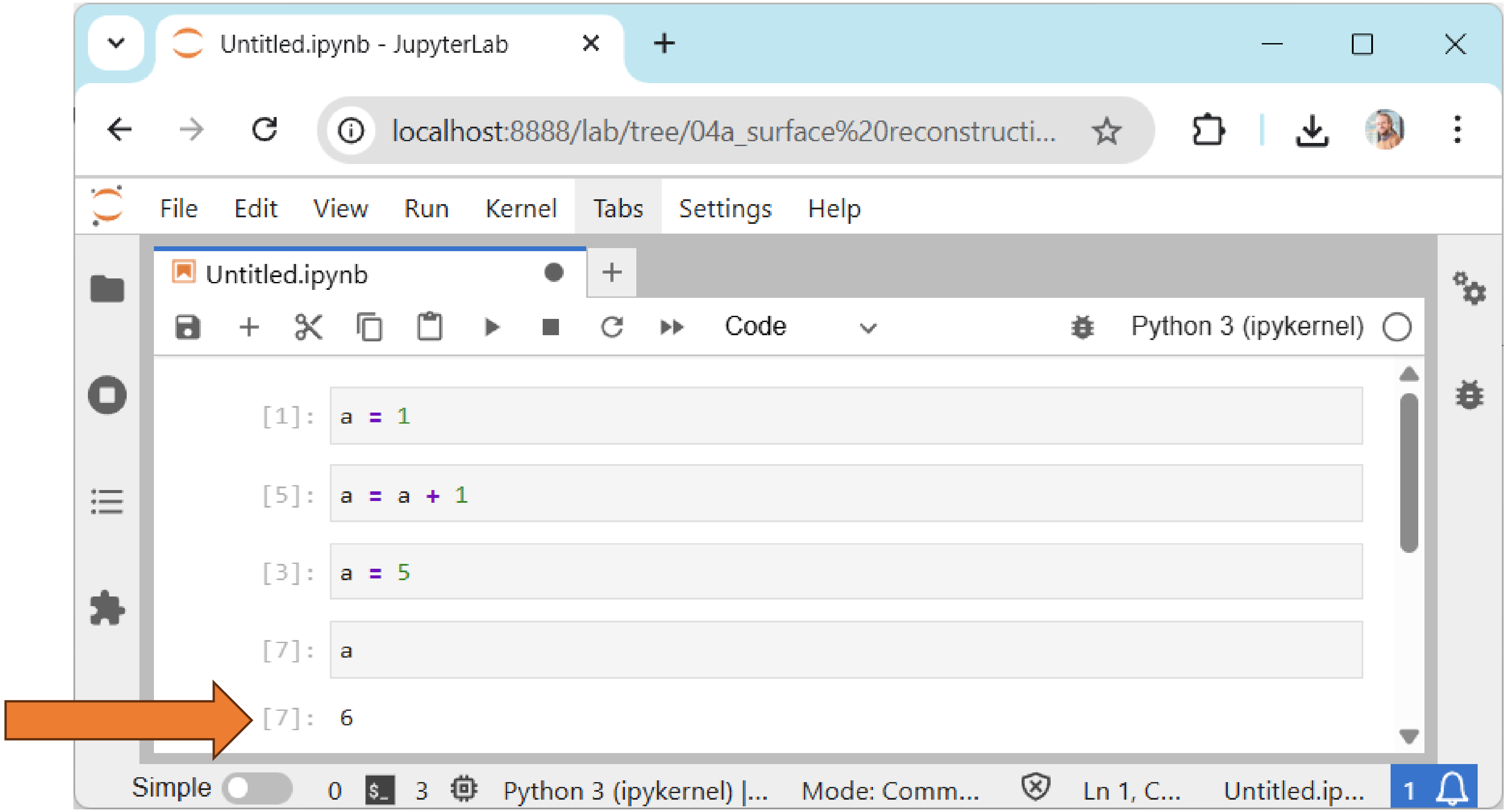
Robert Haase

Using materials from Alba Villaronga Luque and Jesse Veenvliet (MPI CBG Dresden), Marcelo Leomil Zoccoler, Johannes Soltwedel and Mara Lampert, PoL, TU Dresden

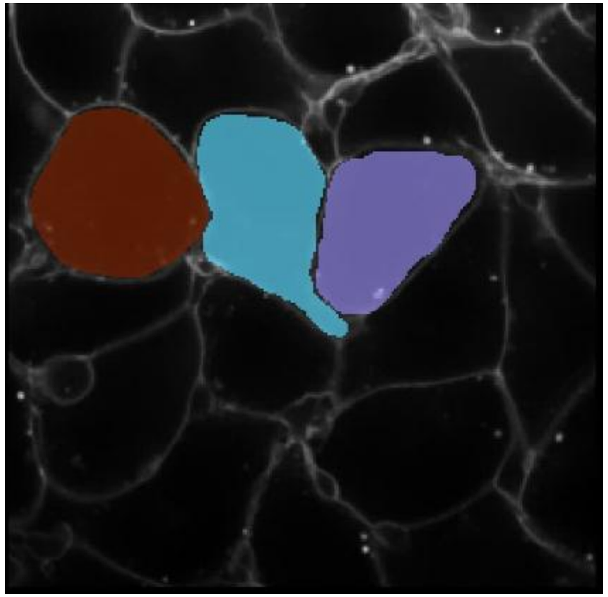
GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Quiz: What is wrong with this result?



Sparse Jaccard Index



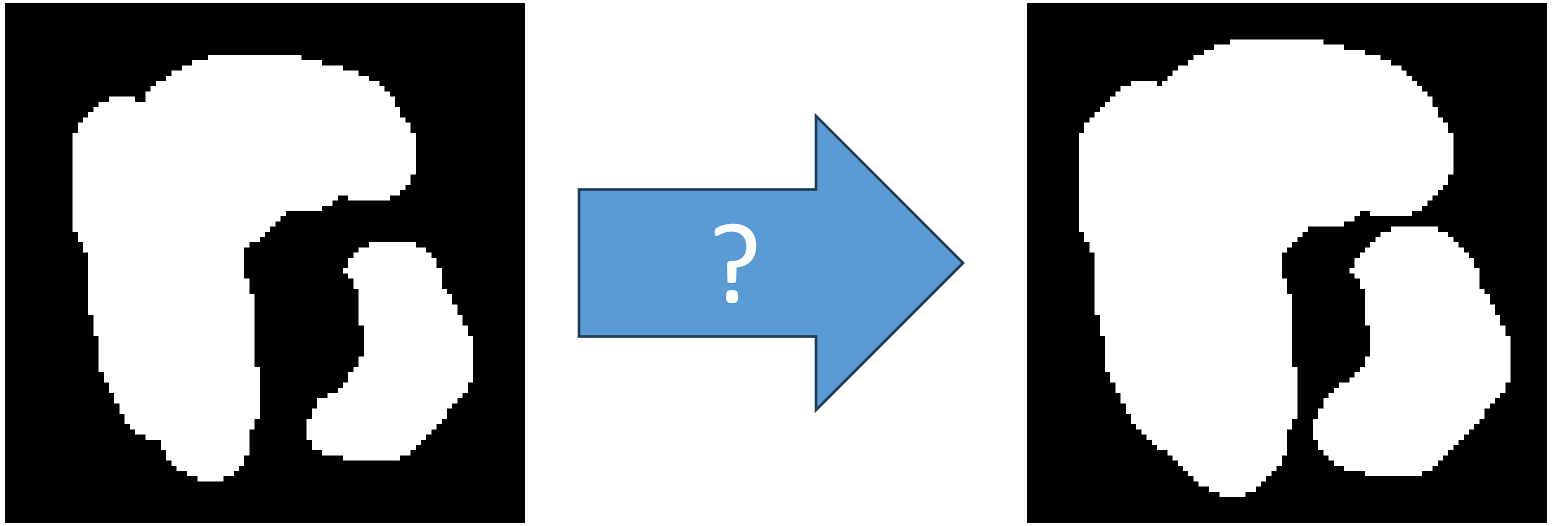
This is a ...

Sparse instance segmentation

Sparse semantic segmentation

Quiz: Recap

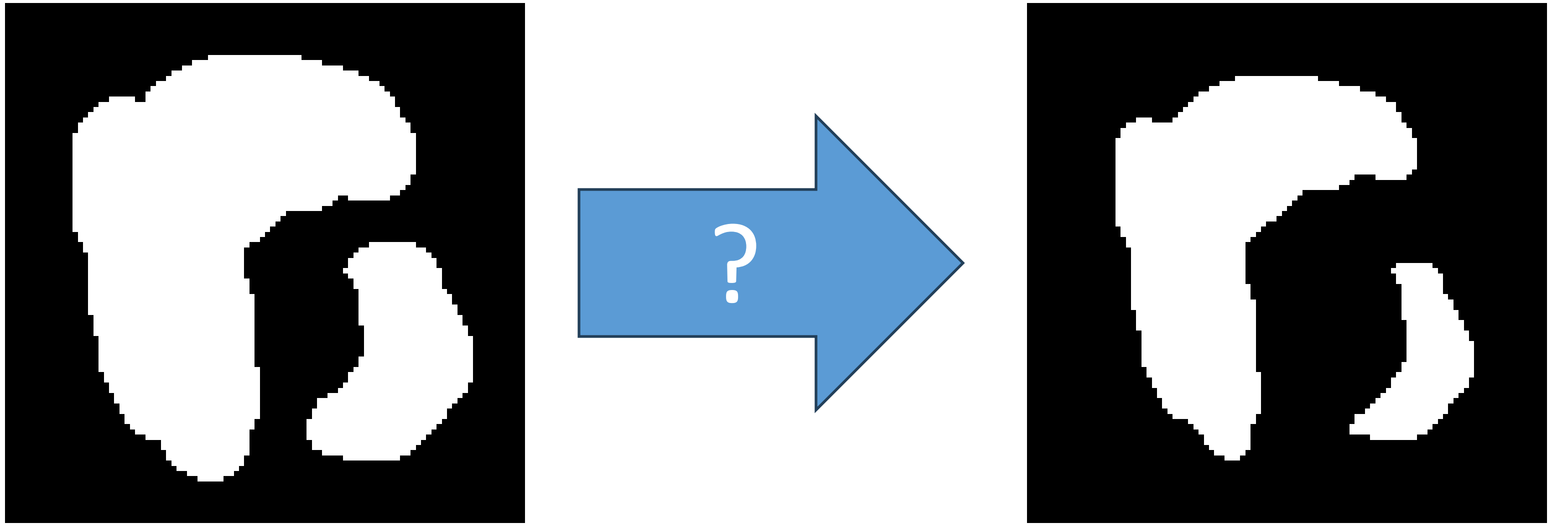
- How is this operation called?



Dilation Erosion Opening Closing

Quiz: Recap

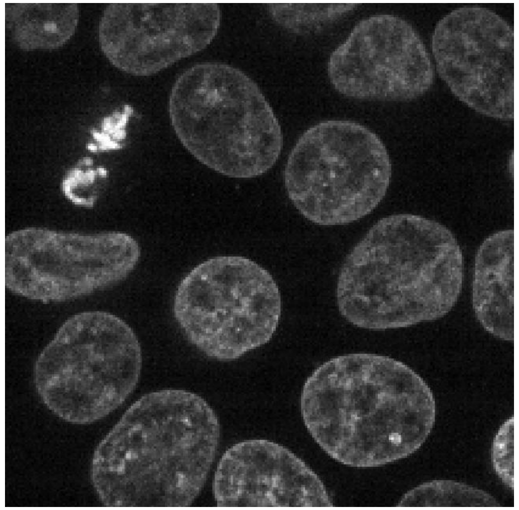
- How is this operation called?



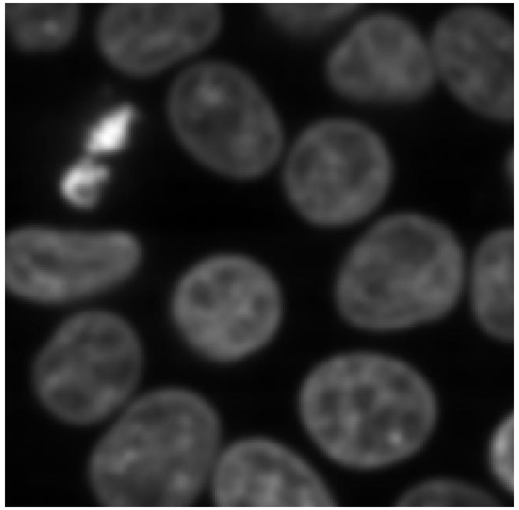
Dilation Erosion Opening Closing

Motivation: Surface reconstruction

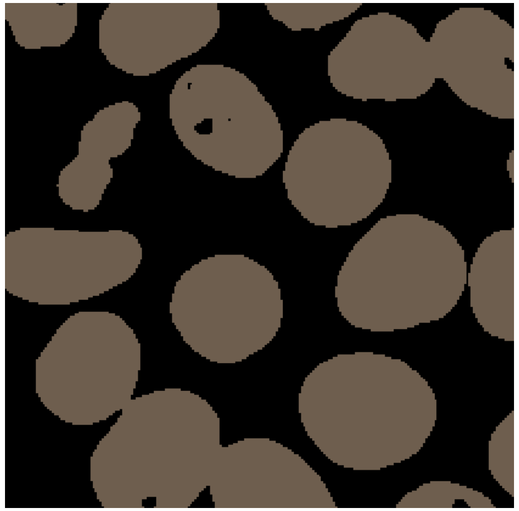
- Pixel and voxel arrays can be huge in memory
- Processing 3D arrays is time-consuming



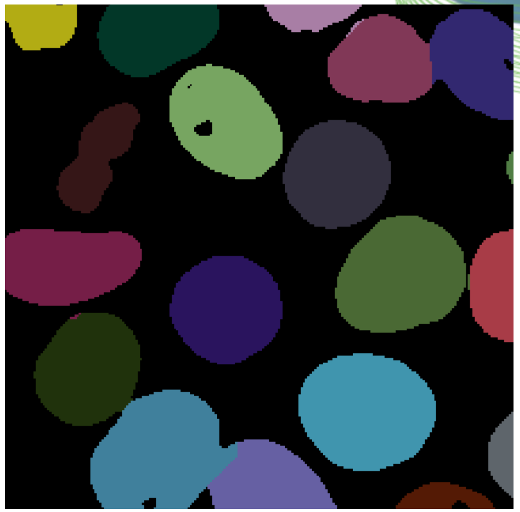
1024 x1024 x 100
16-bit image



1024 x1024 x 100
16-bit image



1024 x1024 x 100
8-bit image



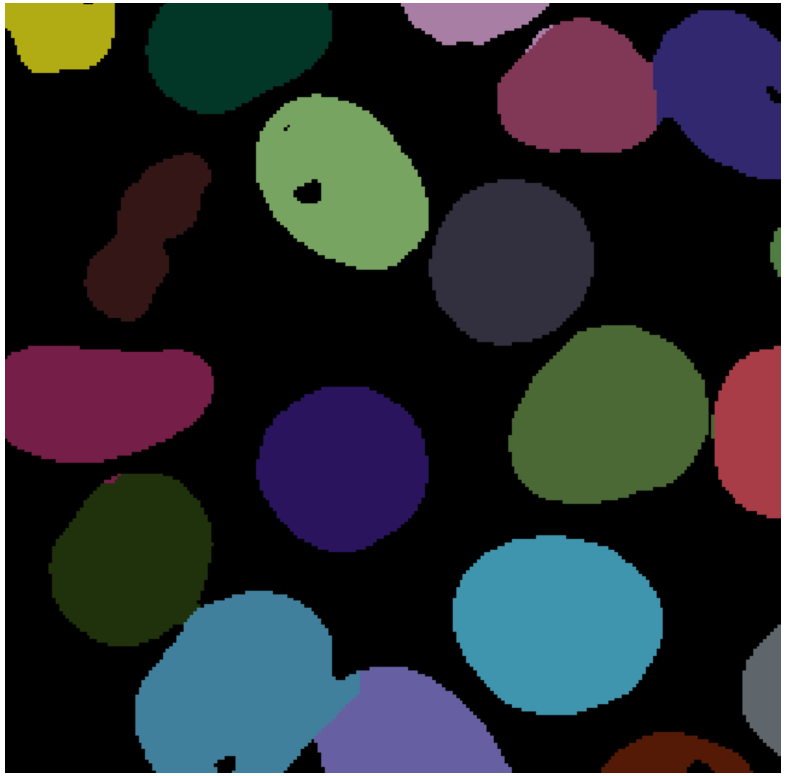
1024 x1024 x 100
16-bit image

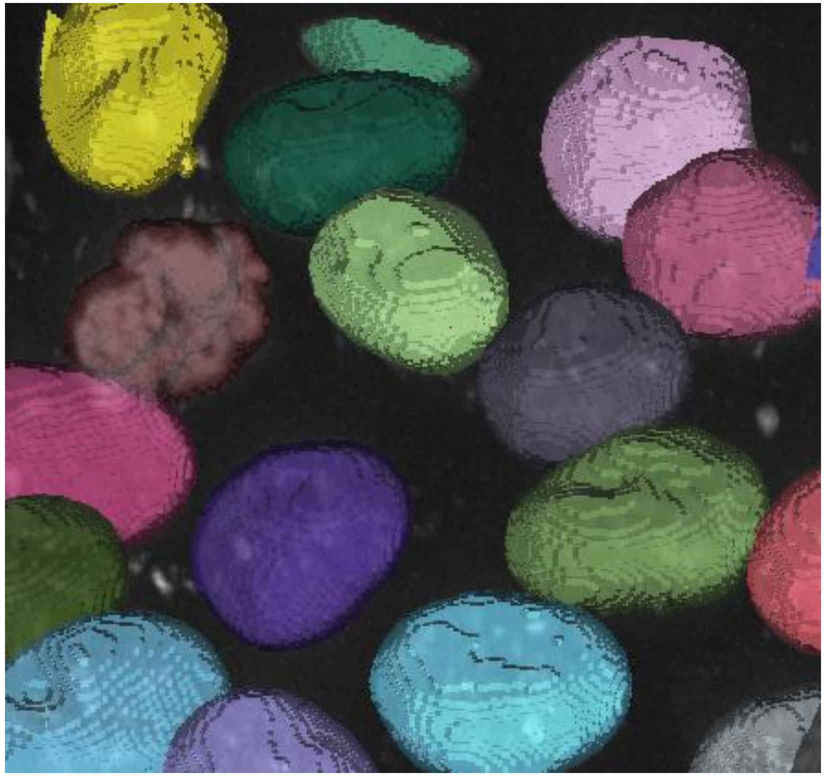
How much memory does this workflow cost?

700 MB	400 MB	4 GB	7 GB

Motivation: Surface reconstruction

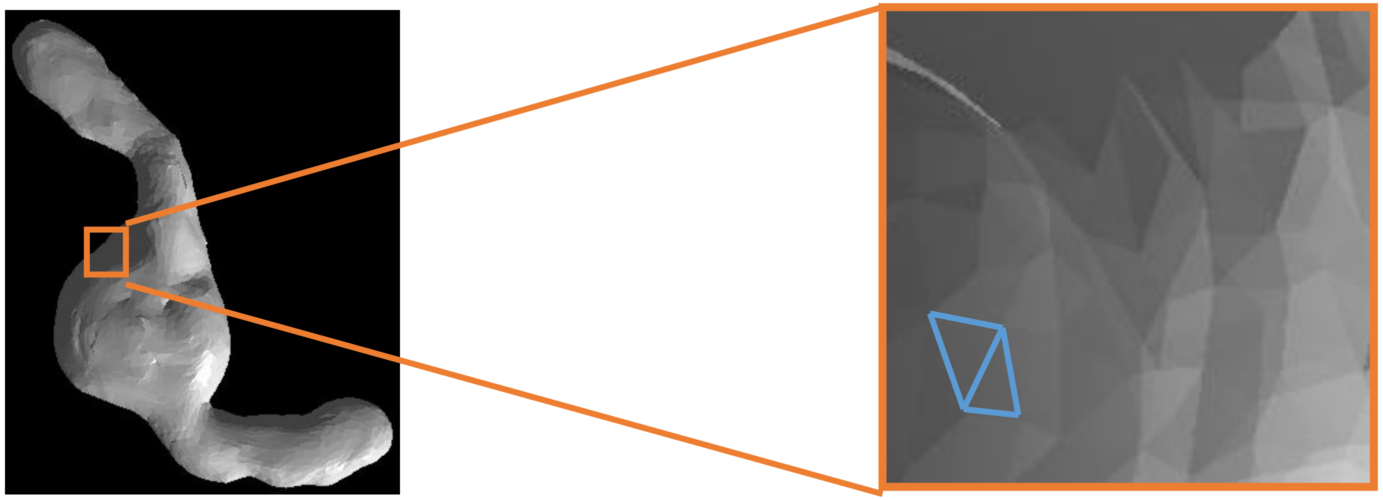
- Pixel and voxel borders introduce artifacts, potentially problematic for measurements, e.g. surface area

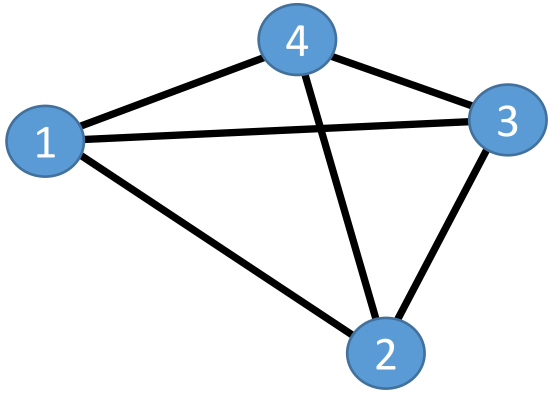




Surface meshes

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





“Vertices” / points

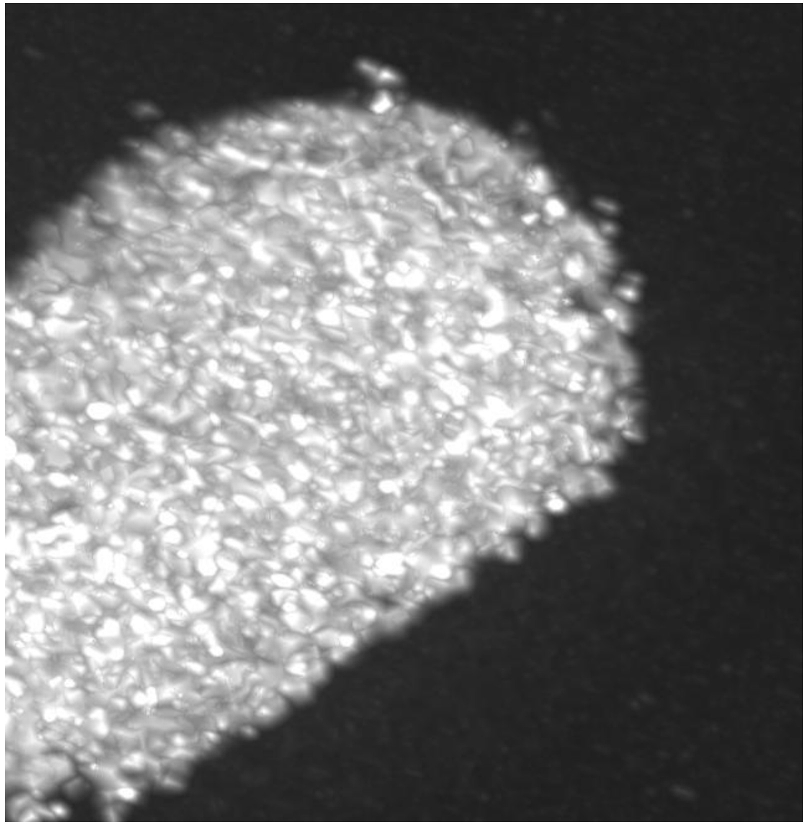
Point x	Point y	Point z
x_1	y_1	z_1
X_2	Y_2	Z_2
X_3	Y_3	Z_3
X_4	Y_4	Z_4
…	…	…

\+

“Faces” / Triangles

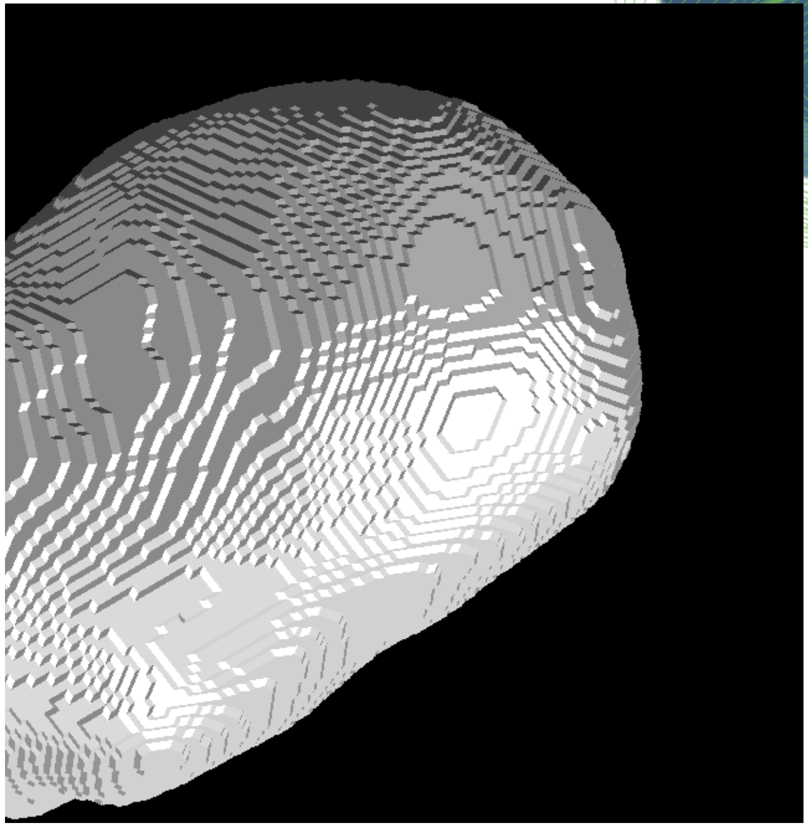
Point 1	Point 2	Point 3
1	2	3
1	2	4
2	3	4
1	3	4

Surface reconstruction



3D image of nuclei

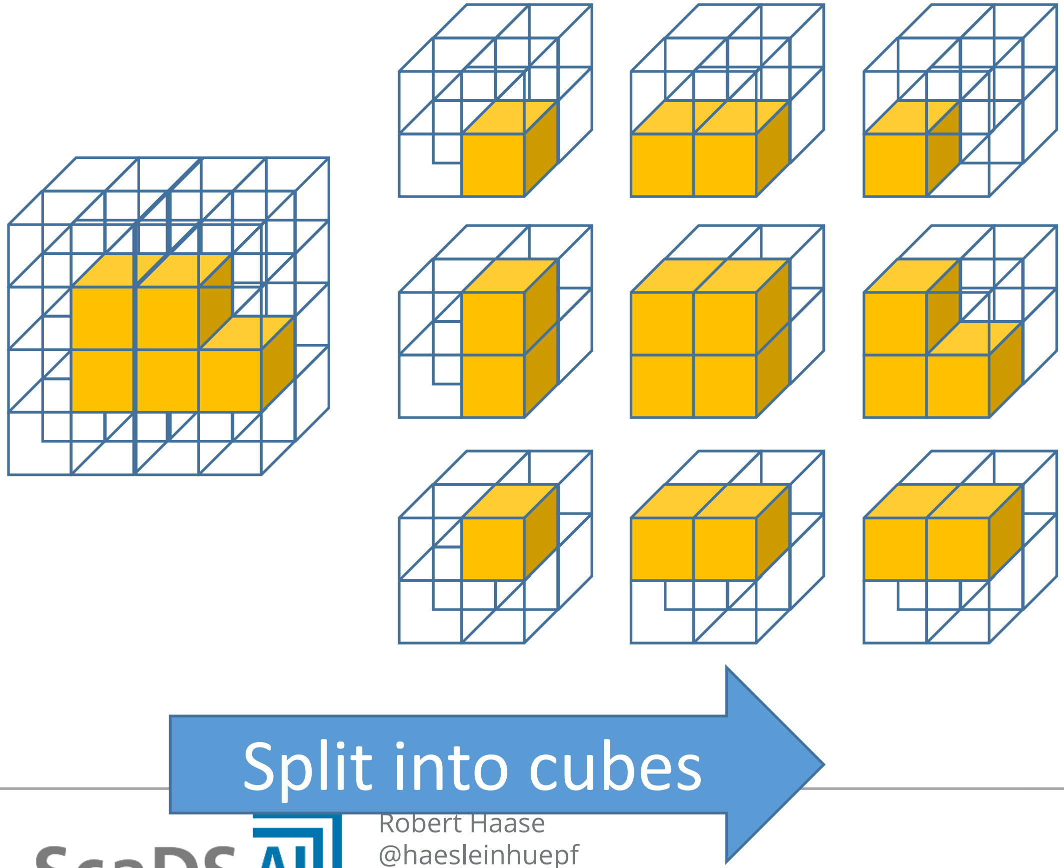
Gaussian filtered



Binary 3D image
(visualized as surface mesh)

Marching cubes algorithm

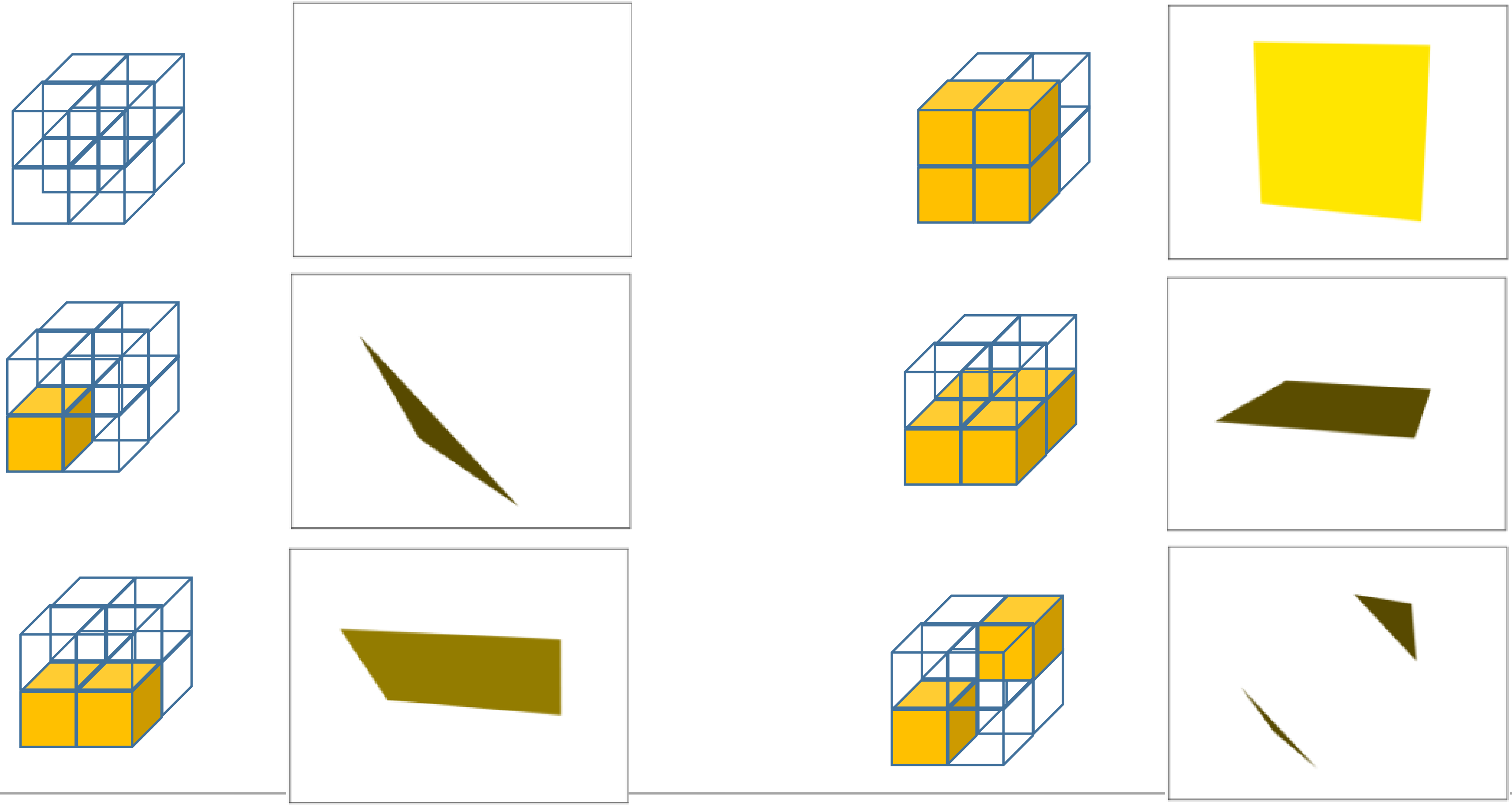
- Starting point: 3D binary image
- Cuts the image in small cubes and iterates over them



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

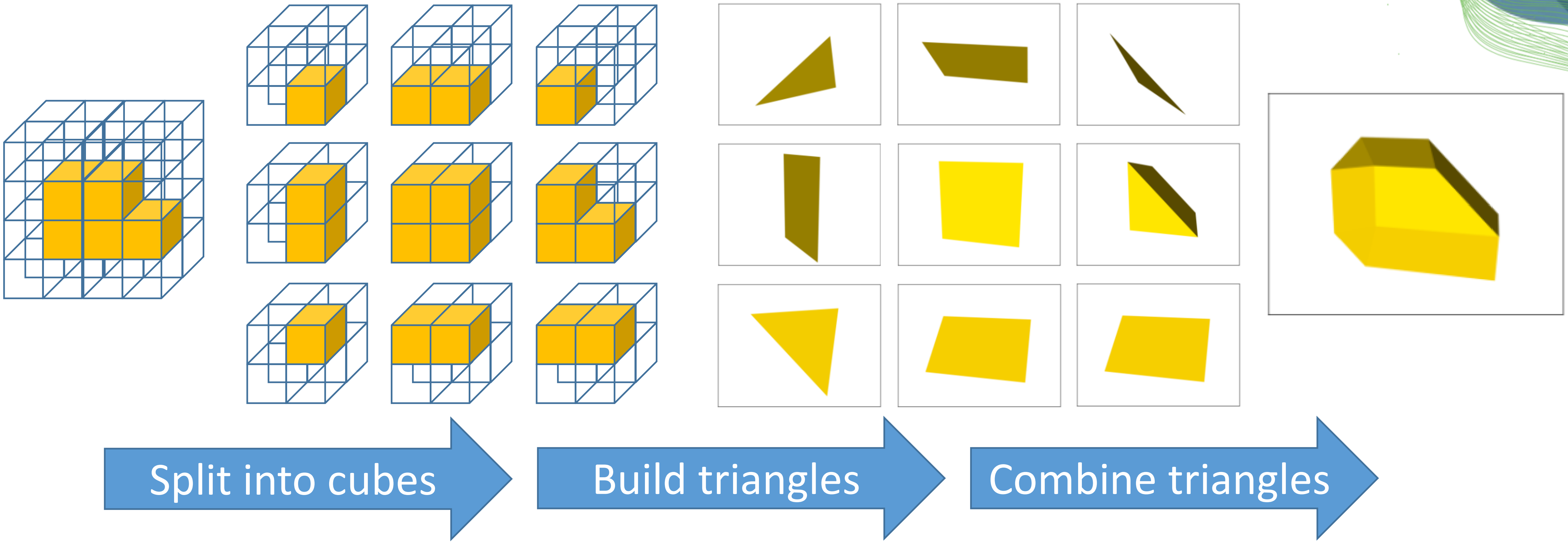
- Starting point: 3D binary image
- Cuts the image in small cubes and iterates over them



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

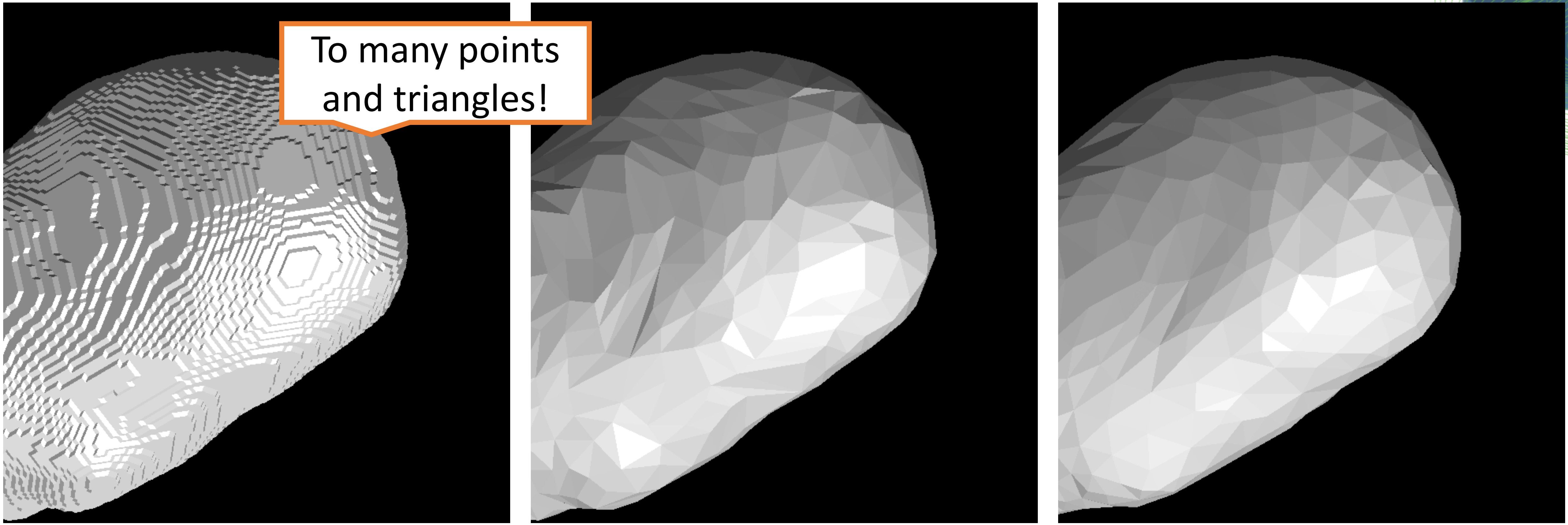
- Starting point: 3D binary image
- Cuts the image in small cubes and iterates over them



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)

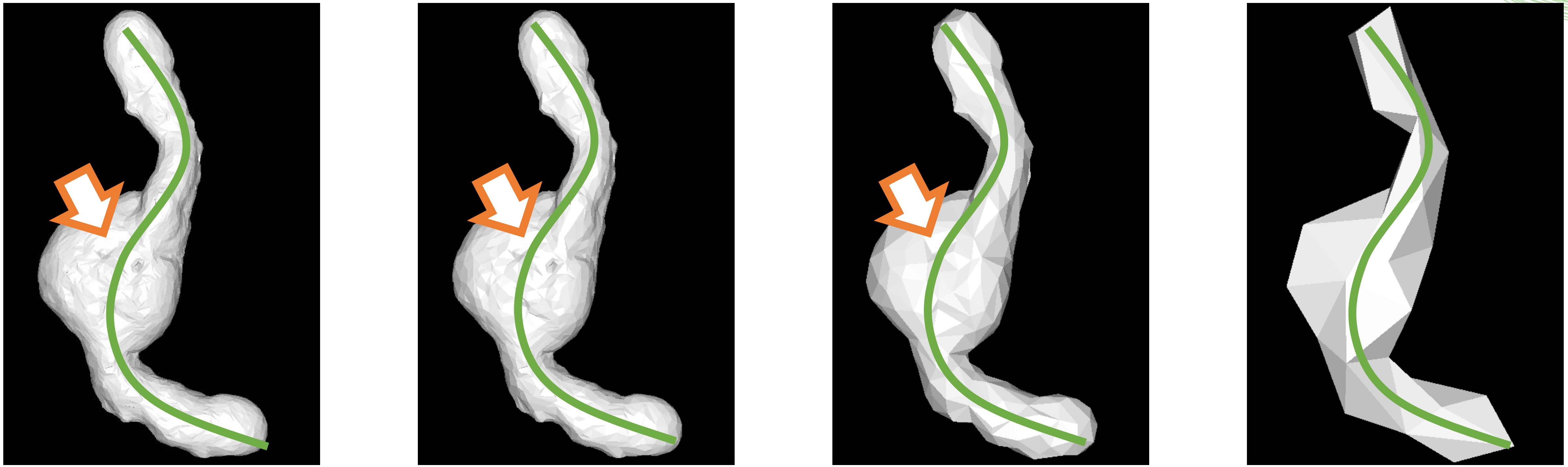
Smoothed mesh
(position locally planarized)

Surface post-processing

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

Number of small concave regions

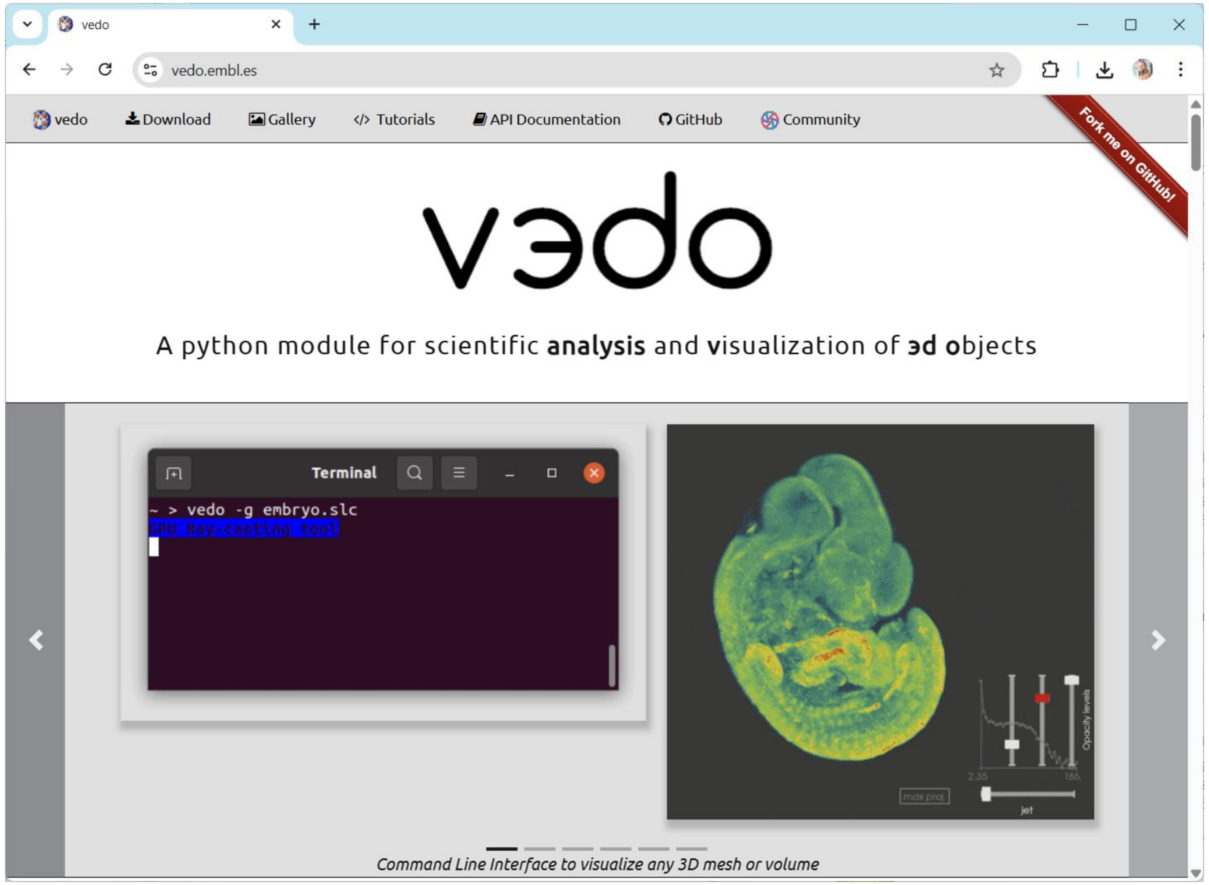
Total length

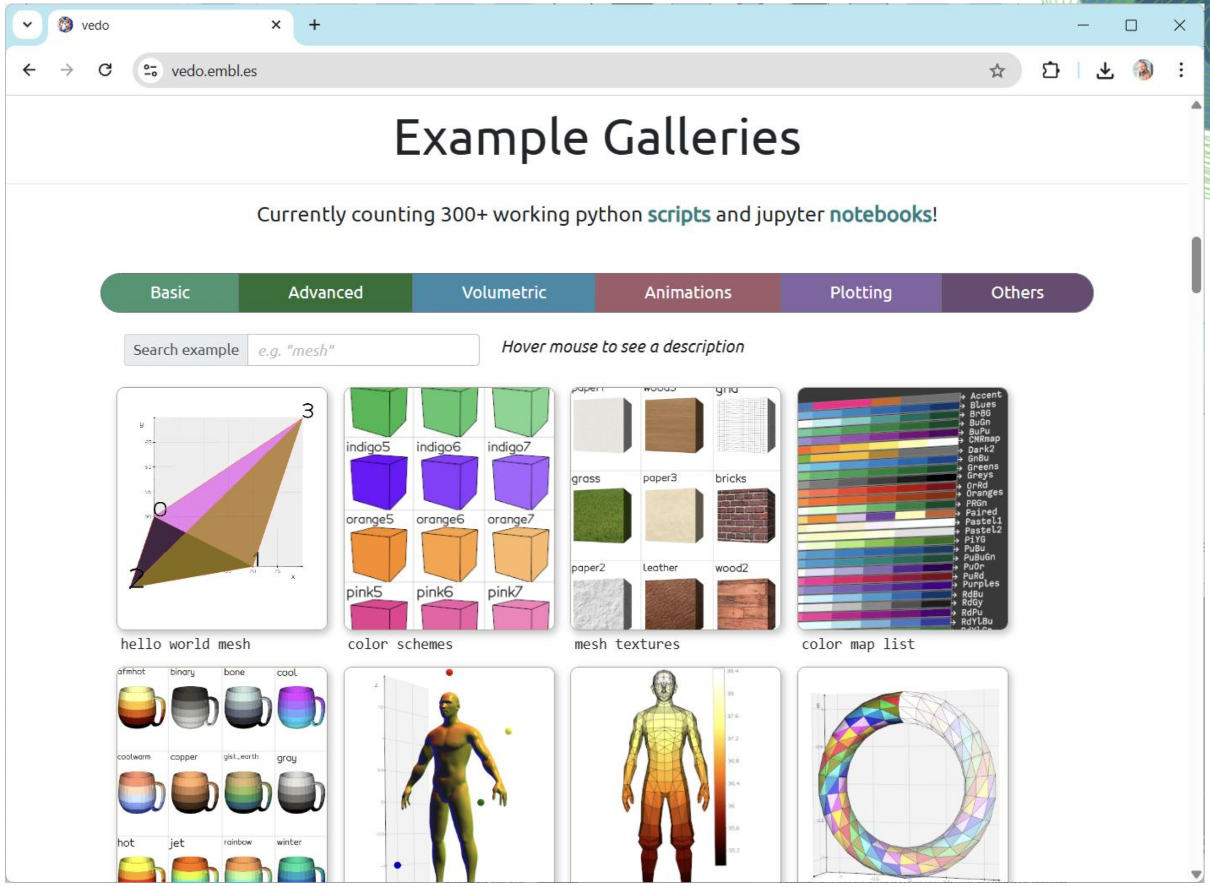


Surface mesh | Simplified by factor 0.5 | Simplified by factor 0.05 | Simplified by factor 0.01

Surface processing: vedo

- Open source mesh + point cloud processing library (MIT licensed)





Surface post-processing

- Meshes are lists of points [vertices] and triangles [faces]

```
[8]: mesh.points
```

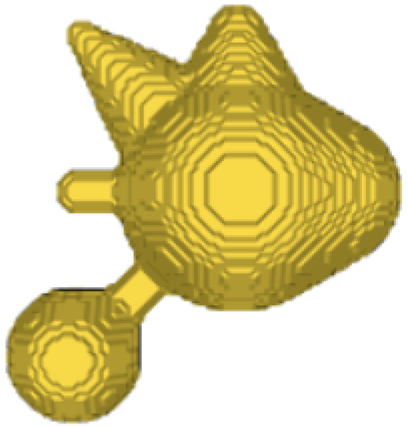
```
[8]: array([[ 47. ,  44. , -25.5],
            [ 46.5,  44. , -26. ],
            [ 47. ,  43.5, -26. ],
            ...,
            [ 51. ,  56. , -74.5],
            [ 52. ,  56. , -74.5],
            [ 53. ,  56. , -74.5]], dtype=float32)
```

```
[9]: mesh.cells[:10]
```

```
[9]: [[2, 1, 0],
      [4, 3, 0],
      [0, 3, 2],
      [6, 5, 4],
      [4, 5, 3],
```

```
[5]: mesh
```

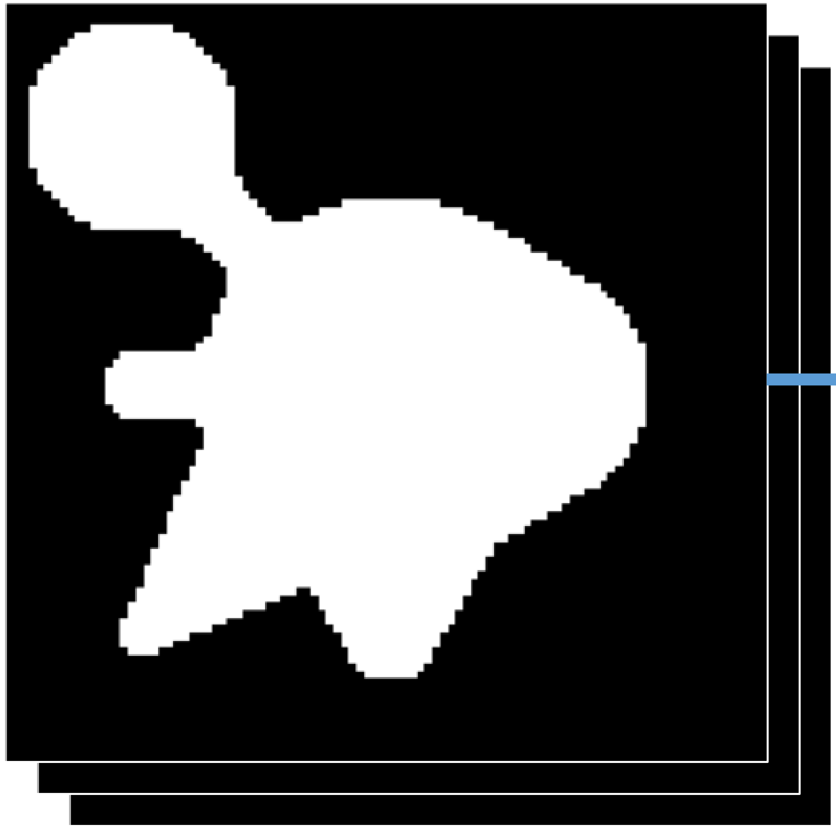
[5]:



Mesh:	**vedo.mesh.Mesh**
bounds (x/y/z)	2.500 ... 83.50 2.500 ... 88.50 -74.50 ... -25.50
center of mass	(42.6, 46.6, -50.0)
average size	31.277
nr. points / faces	19040 / 38076

Surface reconstruction with vedo

- Turn binary and/or label images into surface meshes



[3]:

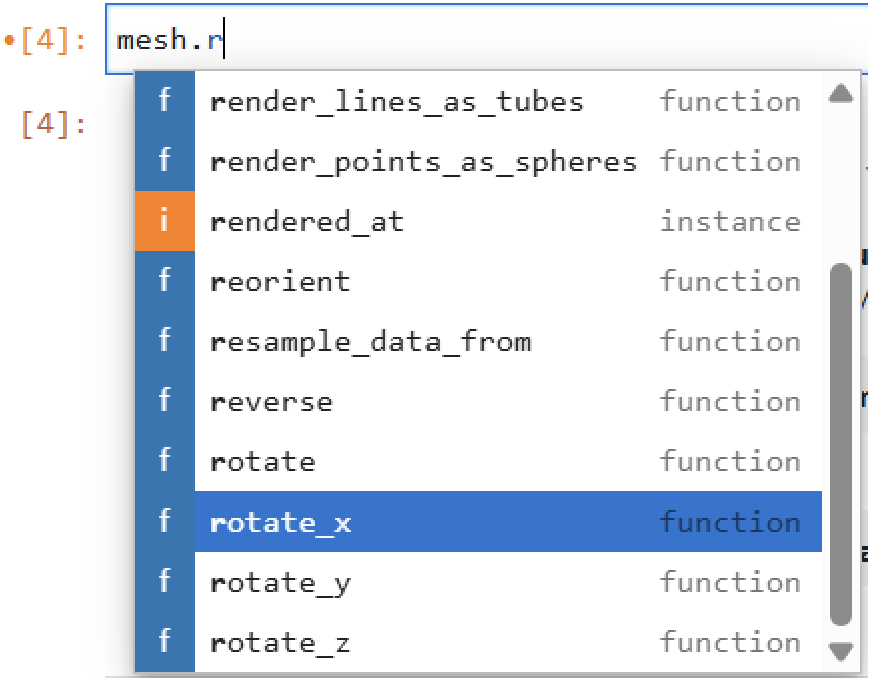
```
verts, faces, normals, values = marching_cubes(binary_image)

mesh = vedo.mesh.Mesh((verts, faces))
mesh
```

[3]:

Mesh:	vedo.mesh.Mesh
bounds (x/y/z)	25.50 ... 74.50 2.500 ... 88.50 2.500 ... 83.50
center of mass	(50.0, 46.6, 42.6)
average size	31.277
nr. points / faces	19040 / 38076

Processing surface meshes with vedo

- Object oriented: mesh... [hit Shift-Tab, to learn more!]

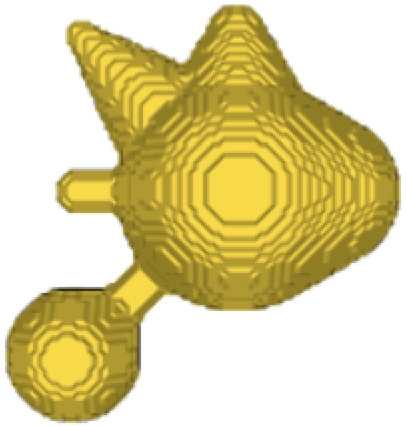


Processing surface meshes with vedo

- Object oriented: mesh...

```
[4]: mesh.rotate_y(90)
```

[4]:



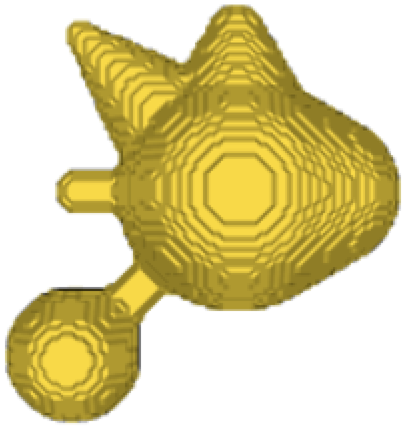
Mesh: vedo.mesh.Mesh

bounds (x/y/z)	2.500 ... 83.50 2.500 ... 88.50 -74.50 ... -25.50
center of mass	(42.6, 46.6, -50.0)
average size	31.277
nr. points / faces	19040 / 38076

Pitfall: vedo uses in-place operations. Calling a function modifies data!

```
[5]: mesh
```

[5]:



Mesh: vedo.mesh.Mesh

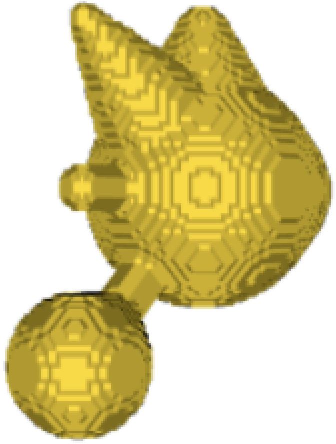
bounds (x/y/z)	2.500 ... 83.50 2.500 ... 88.50 -74.50 ... -25.50
center of mass	(42.6, 46.6, -50.0)
average size	31.277
nr. points / faces	19040 / 38076

Processing surface meshes with vedo

- Copy objects to prevent changing the original data

[6]:
```
rotated = mesh.copy().rotate_y(angle=45)
rotated
```

[6]:

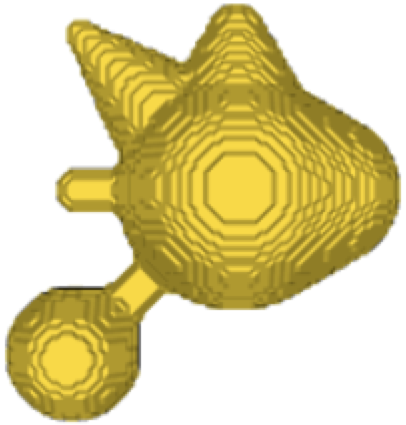


Mesh: vedo.mesh.Mesh

bounds (x/y/z)	-37.83 ... 28.64 2.500 ... 88.50 -99.35 ... -32.88
center of mass	(-5.24, 46.6, -65.5)
average size	31.277
nr. points / faces	19040 / 38076

[7]:
```
mesh
```

[7]:



Mesh: vedo.mesh.Mesh

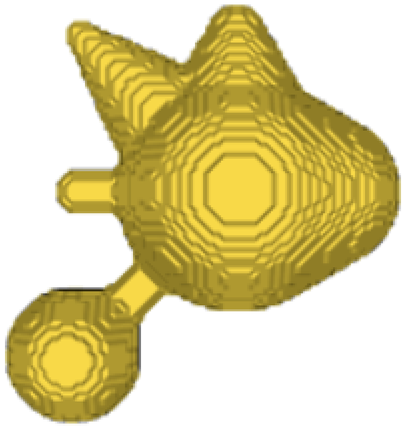
bounds (x/y/z)	2.500 ... 83.50 2.500 ... 88.50 -74.50 ... -25.50
center of mass	(42.6, 46.6, -50.0)
average size	31.277
nr. points / faces	19040 / 38076

Surface mesh processing

- Surface mesh simplification
- To prevent the computer freezing

[7]:
```
mesh
```

[7]:

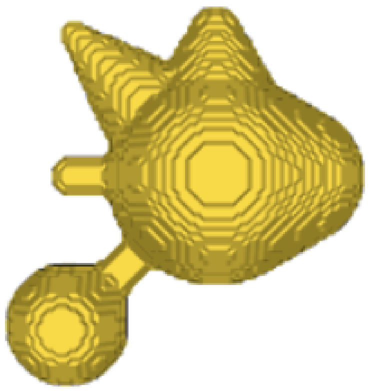


Mesh: vedo.mesh.Mesh

bounds (x/y/z)	2.500 ... 83.50 2.500 ... 88.50 -74.50 ... -25.50
center of mass	(42.6, 46.6, -50.0)
average size	31.277
nr. points / faces	19040 / 38076

[13]:
```
decimated_mesh = mesh.copy().decimate(fraction=0.5)
decimated_mesh
```

[13]:



Mesh: vedo.mesh.Mesh

bounds (x/y/z)	2.500 ... 83.50 2.500 ... 88.50 -74.50 ... -25.50
center of mass	(39.9, 43.6, -48.5)
average size	31.100
nr. points / faces	9521 / 19038

[16]:

Mesh: vedo.mesh.Mesh

bounds (x/y/z)	4.806 ... 81.54 5.490 ... 89.67 -74.54 ... -25.58
center of mass	(40.2, 46.8, -50.3)
average size	33.785
nr. points / faces	22 / 37

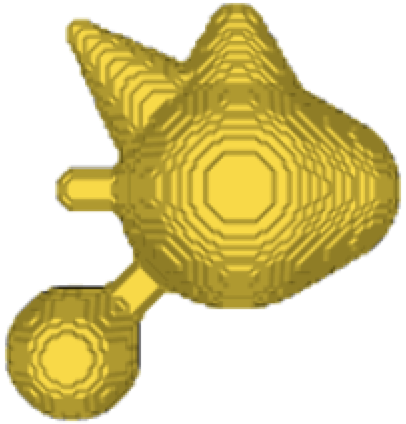
Quiz: What fraction created this surface mesh?

Surface mesh processing

- Surface mesh smoothing

```
[7]: mesh
```

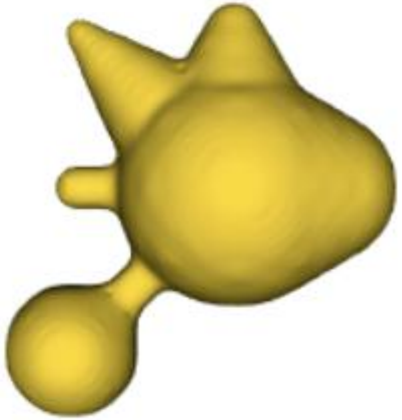
[7]:



Mesh:	vedo.mesh.Mesh
bounds (x/y/z)	2.500 ... 83.50 2.500 ... 88.50 -74.50 ... -25.50
center of mass	(42.6, 46.6, -50.0)
average size	31.277
nr. points / faces	19040 / 38076

```
[27]: smoothed_mesh = mesh.copy().smooth(niter=15,
                                       pass_band=0.0001,
                                       edge_angle=15,
                                       feature_angle=60,
                                       boundary=False)

      smoothed_mesh
```

[27]:



Mesh:	vedo.mesh.Mesh
bounds (x/y/z)	2.386 ... 83.61 2.383 ... 88.59 -74.58 ... -25.42
center of mass	(42.6, 46.6, -50.0)
average size	31.277
nr. points / faces	19040 / 38076

Surface mesh processing

```
[28]: mesh.smooth?
```

```
Signature:
mesh.smooth(
    niter=15,
    pass_band=0.1,
    edge_angle=15,
    feature_angle=60,
    boundary=False,
) -> Self
Docstring:
Adjust mesh point positions using the so-called "Windowed Sinc" method.

Arguments:
    niter : (int)
        number of iterations.
    pass_band : (float)
        set the pass_band value for the windowed sinc filter.
    edge_angle : (float)
        edge angle to control smoothing along edges (either interior or boundary).
    feature_angle : (float)
        specifies the feature angle for sharp edge identification.
    boundary : (bool)
        specify if boundary should also be smoothed or kept unmodified
```

View surface meshes in Napari

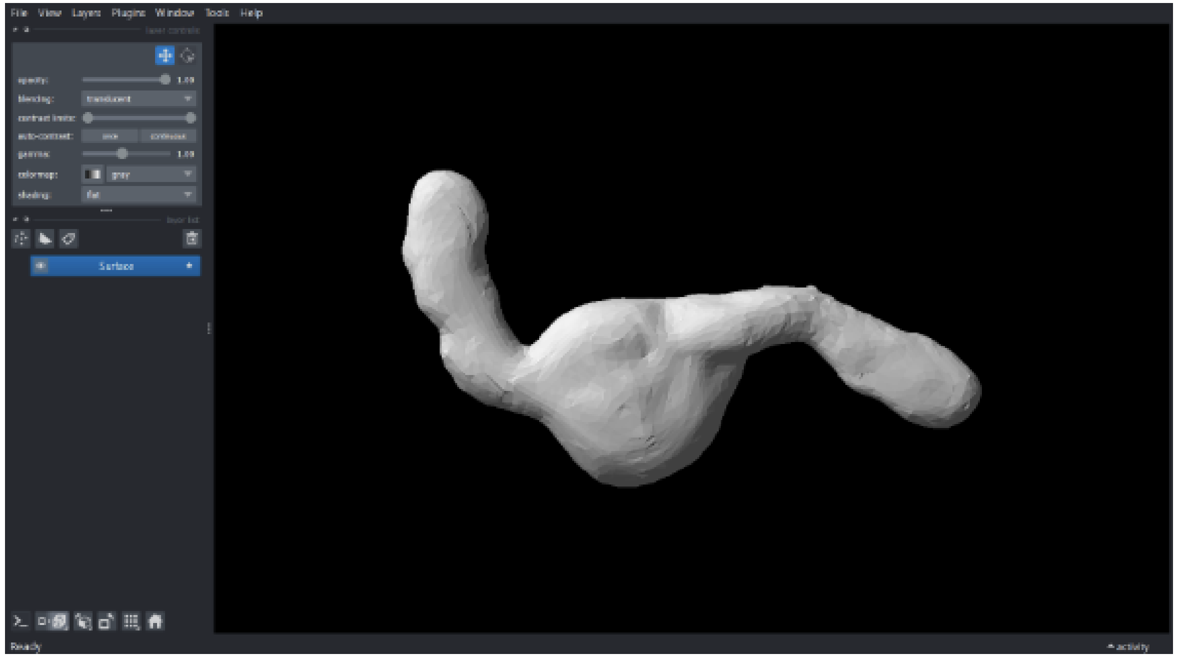
```
[5]: viewer = napari.Viewer(ndisplay=3)
```

Start Napari in 3D-mode

```
[6]: def to_napari_surface_tuple(vedo_mesh):
         import numpy as np
         return (vedo_mesh.points, np.asarray(vedo_mesh.cells))

     viewer.add_surface(to_napari_surface_tuple(surface))

     napari.utils.nbscreenshot(viewer)
```

Surface meshes in Napari are tuples of (vertices, faces)

[6]:



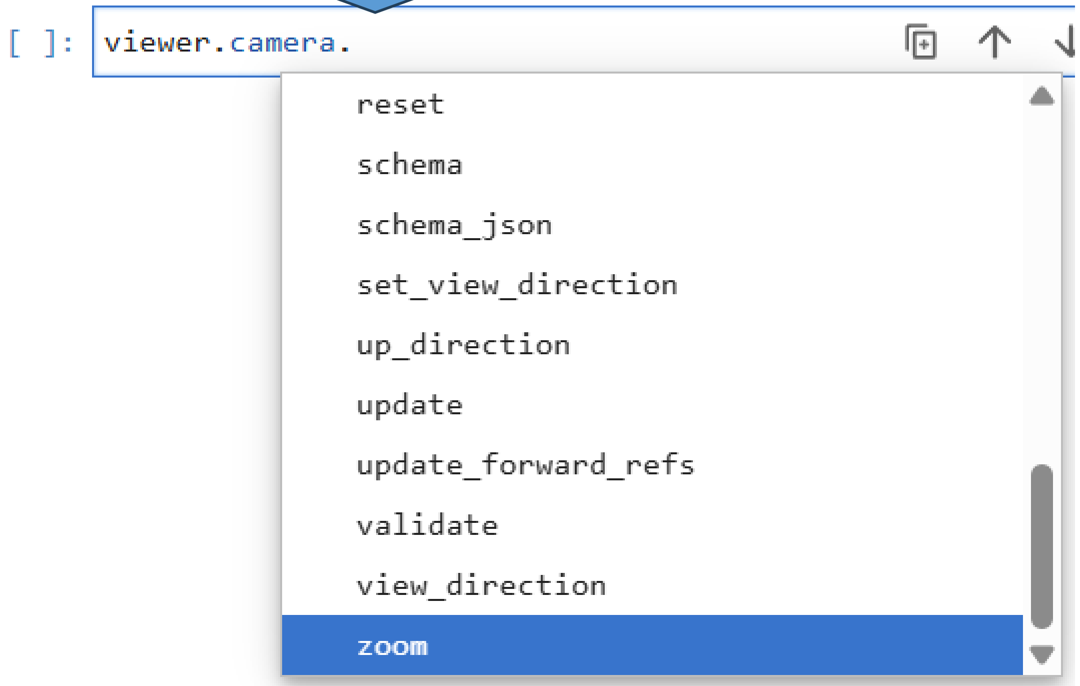
View surface meshes in Napari

- You can modify the view in napari, by changing camera parameters.

```
[7]: viewer.camera.angles = [0,0,0]

     napari.utils.nbscreenshot(viewer)
```

[7]:

Use SHIFT-Tab for more options



CENTER FOR SCALABLE DATA ANALYTICS
AND ARTIFICIAL INTELLIGENCE

Feature extraction

Robert Haase

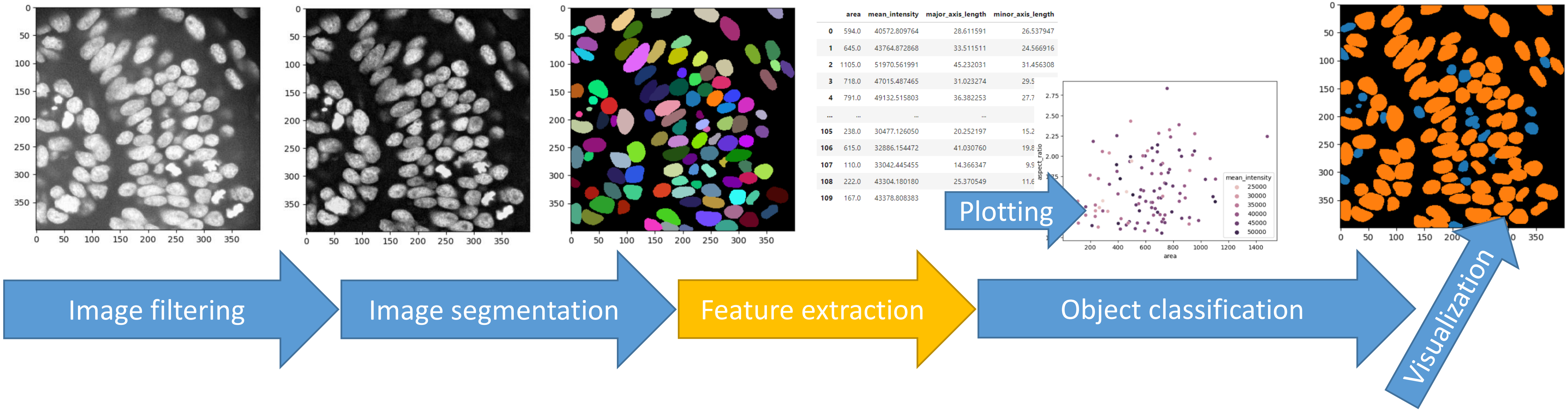
GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Lecture overview: Bio-image Analysis

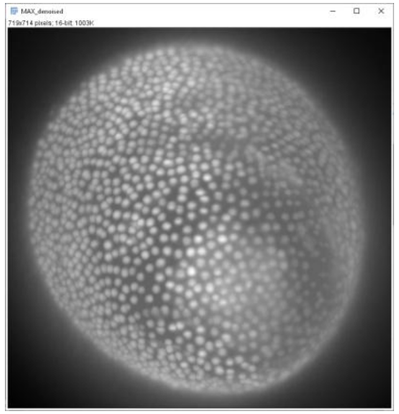
- Image Data Analysis workflows
- Goal: **Quantify observations, substantiate conclusions with numbers**



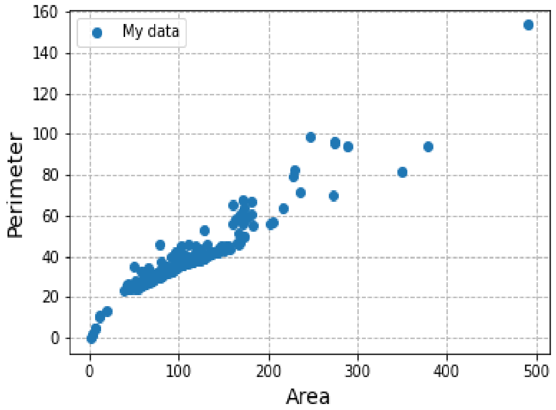
Robert Haase
@haesleinhuepf

Feature extraction

- Feature extraction is a *late* processing step in image analysis.
- It can be used for images or

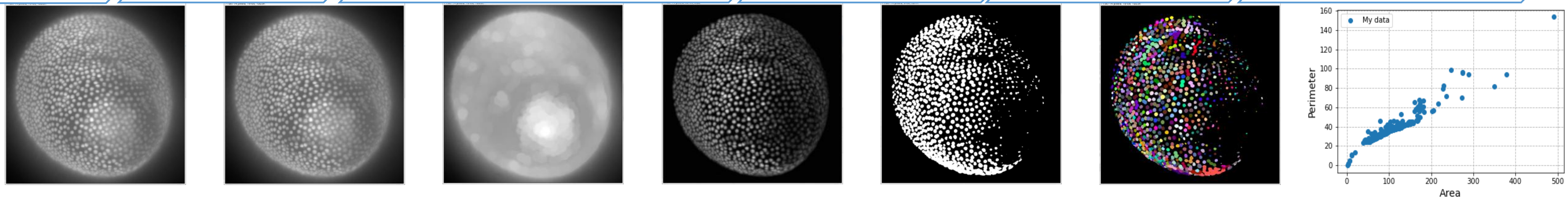


Feature Extraction



- or segmented/labelled images

Acquisition | Denoising | Background subtraction | Segmentation | Labeling | Feature Extraction



Feature extraction

- A *feature* is a countable or measurable property of an image or object.
- Goal of feature extraction is finding a minimal set of features to describe an object well enough to differentiate it from other objects.

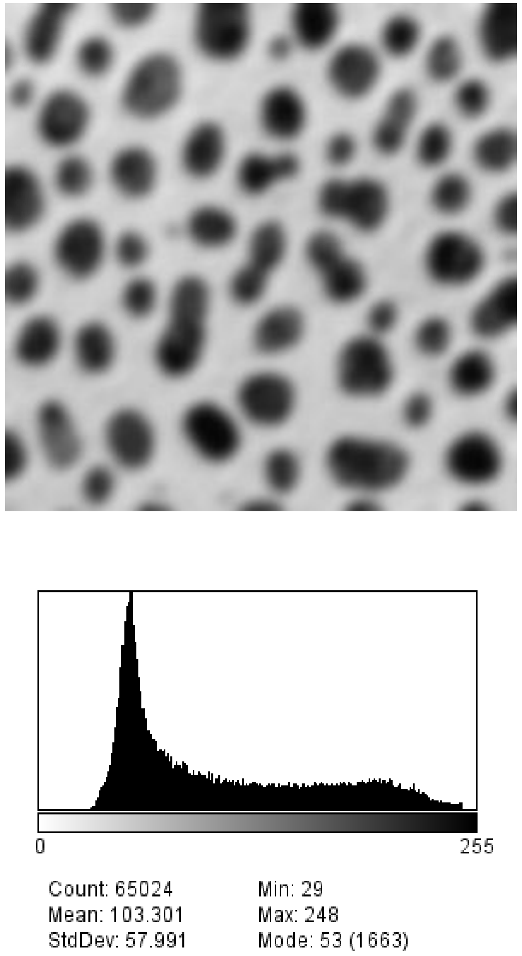
- **Intensity based**
  - Mean intensity
  - Standard deviation
  - Total intensity
  - Textures
- **Mixed features**
  - Center of mass
  - Local minima / maxima
  - Distance to neighbors
  - Average intensity in neighborhood
- **Shape based /spatial**
  - Area / Volume
  - Roundness
  - Solidity
  - Circularity / Sphericity
  - Elongation
  - Centroid
  - Bounding box
- **Spatio-temporal**
  - Displacement,
  - Speed,
  - Acceleration
- **Topological**
  - Number of neighbors
- **Others**
  - Overlap
  - Colocalization

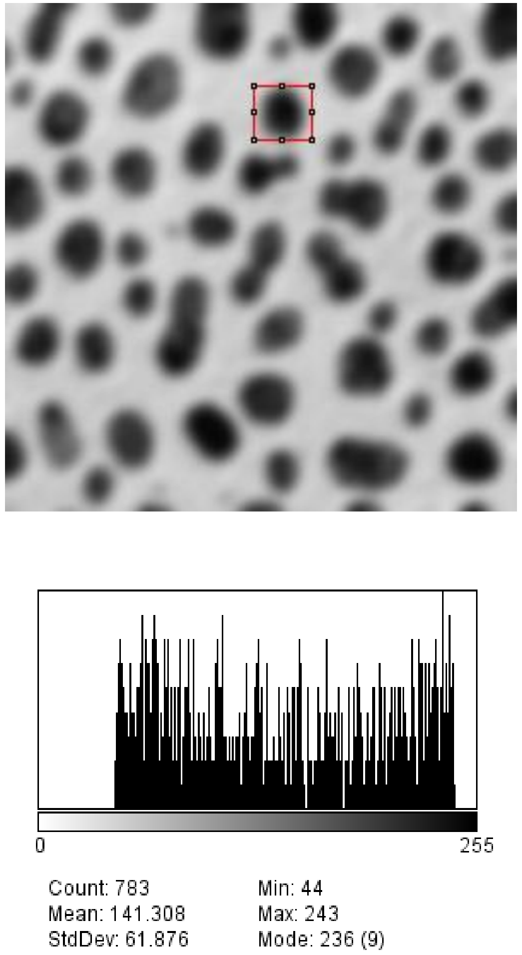
Intensity based features

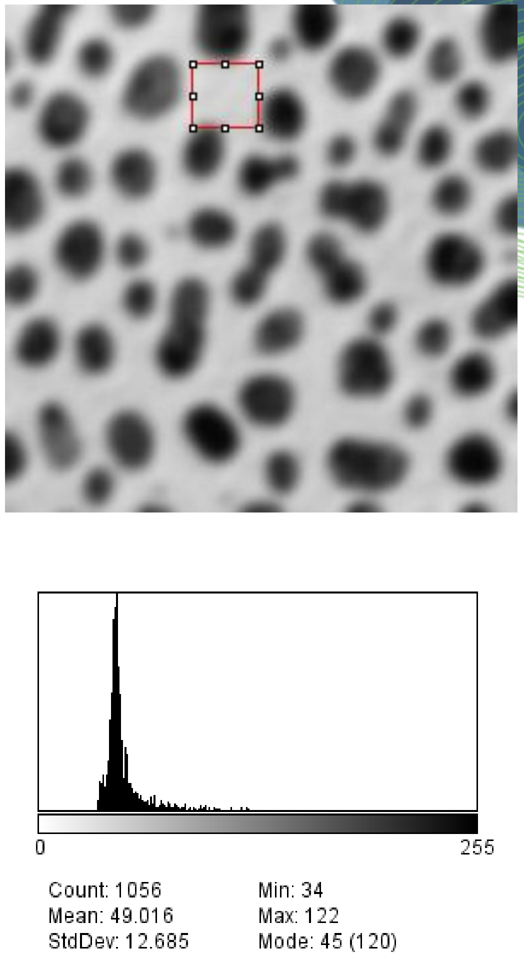
- Min / max
- Median
- Mean
- Mode
- Variance
- Standard deviation

- Can be derived from pixel values
- Don't take spatial relationship of pixels into account

- See also:
  - descriptive statistics
  - histogram







Bounding rectangle / bounding box

- Position and size of the smallest rectangle containing all pixels of an object
  - x_b, y_b … position of the bounding box
  - w_b … width of the bounding box
  - h_b … height of the bounding box

variable	value
x_b	0
y_b	2
w_b	3
h_b	2

y \ x	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	1	1	1	0	0
3	0	1	1	0	0
4	0	0	0	0	0

Center of mass

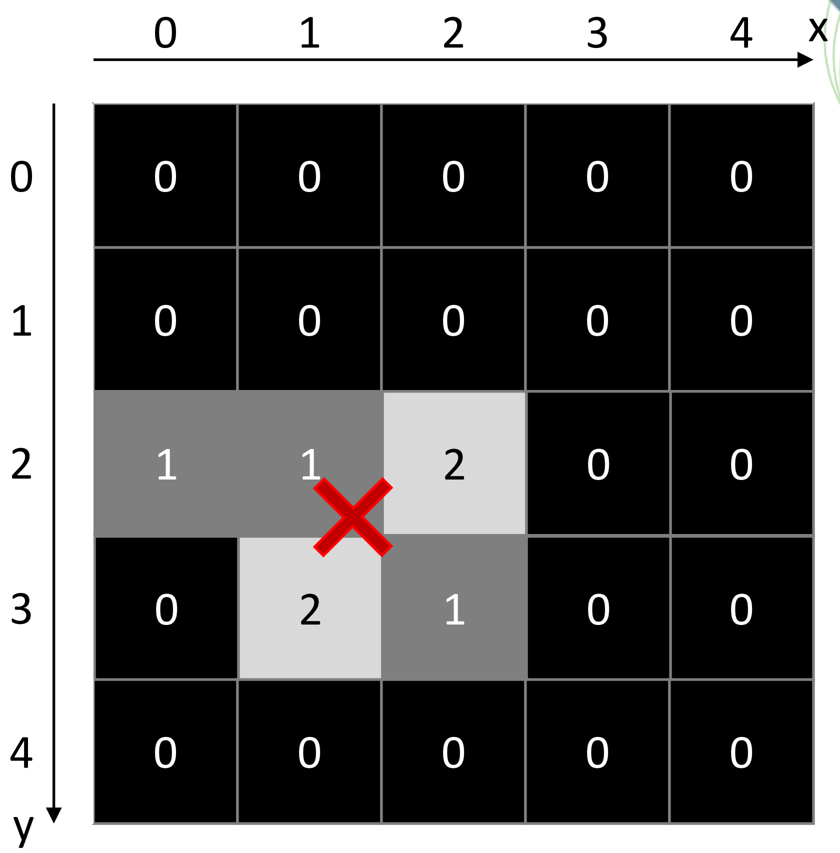
- Relative position in an image weighted by pixel intensities
  - x, y … pixel coordinates
  - w … image width
  - h … image height
  - µ … mean intensity
  - $g_{x,y}$ … pixel grey value
  - x_m, y_m … center of mass coordinates

$$\mu = \frac{1}{wh}\sum_{y=0}^{h-1}\sum_{x=0}^{w-1} g_{x,y}$$

$$x_m = \frac{1}{wh\mu}\sum_{y=0}^{h-1}\sum_{x=0}^{w-1} x\, g_{x,y}$$

“sum intensity”
“total intensity”

$$y_m = \frac{1}{wh\mu}\sum_{y=0}^{h-1}\sum_{x=0}^{w-1} y\, g_{x,y}$$



x_m = 1/7 (1·0 + 1·1 + 2·2 + 2·1 + 1·2) = 1.3

y_m = 1/7 (1·2 + 1·2 + 2·3 + 2·2 + 1·3) = 2.4

Center of geometry / centroid

- Relative position in an image weighted by pixel intensities
- Special case of center of mass for binary images
  - x, y … pixel coordinates
  - w … image width
  - h … image height
  - µ … mean intensity
  - $g_{x,y}$ … pixel grey value, integer in range [0;1]
  - x_m, y_m … center of mass coordinates

$$\mu = \frac{1}{wh} \sum_{y=0}^{h-1} \sum_{x=0}^{w-1} g_{x,y}$$

$$x_m = \frac{1}{wh\mu} \sum_{y=0}^{h-1} \sum_{x=0}^{w-1} x\, g_{x,y}$$

$$y_m = \frac{1}{wh\mu} \sum_{y=0}^{h-1} \sum_{x=0}^{w-1} y\, g_{x,y}$$

Number of white pixels

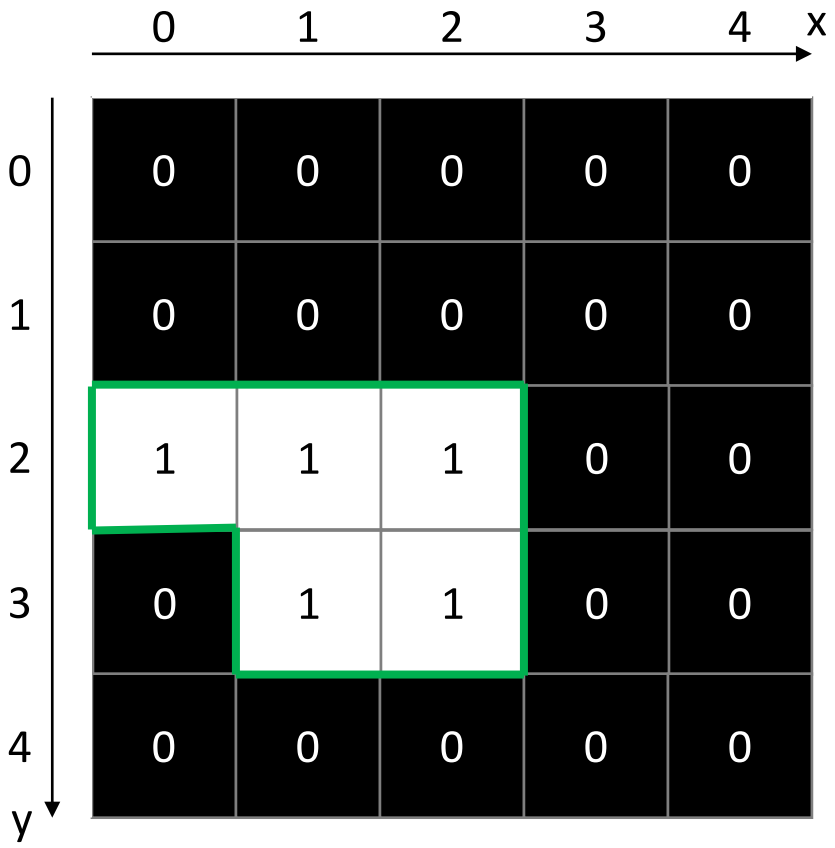
y \ x	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	1	1	1	0	0
3	0	1	1	0	0
4	0	0	0	0	0

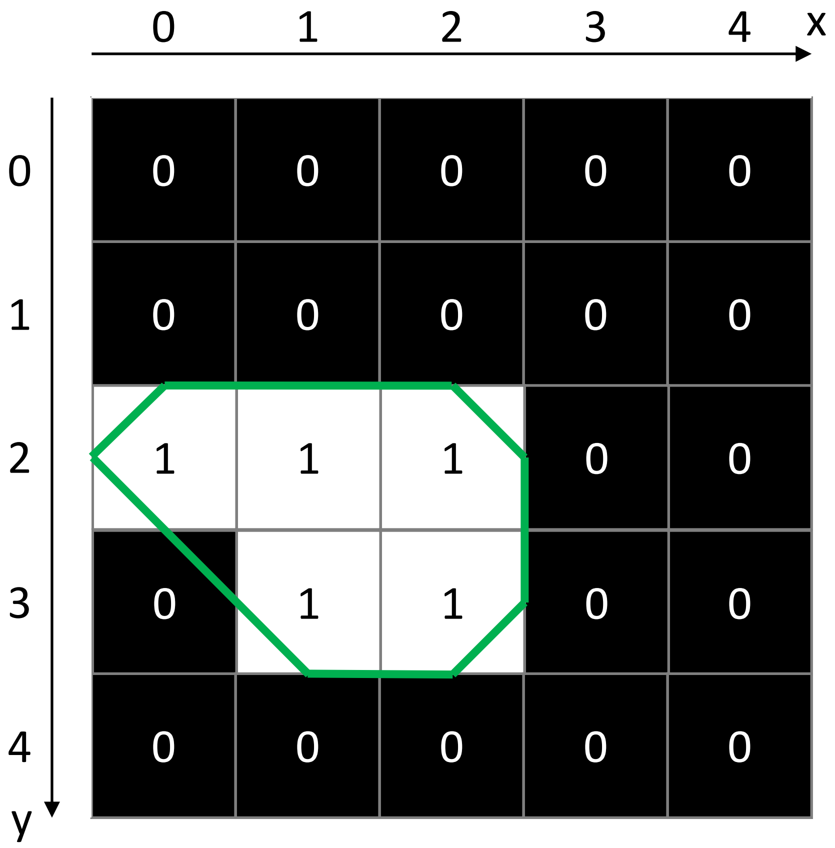
x_m = 1/5 (1·0 + 1·1 + 1·2 + 1·1 + 1·2) = 1.2

y_m = 1/5 (1·2 + 1·2 + 1·3 + 1·2 + 1·3) = 2.4

Perimeter

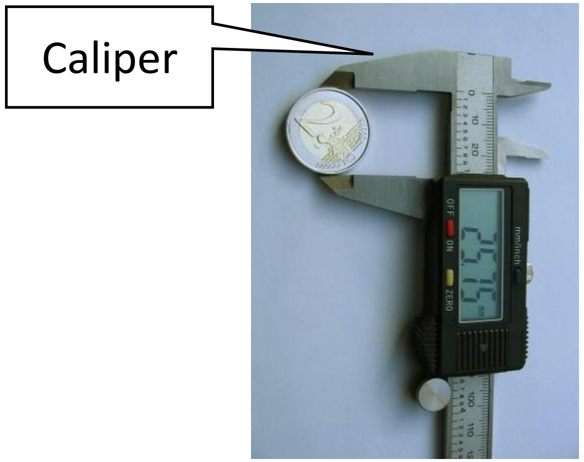
- Length of the outline around an object
- Depends on the actual implementation

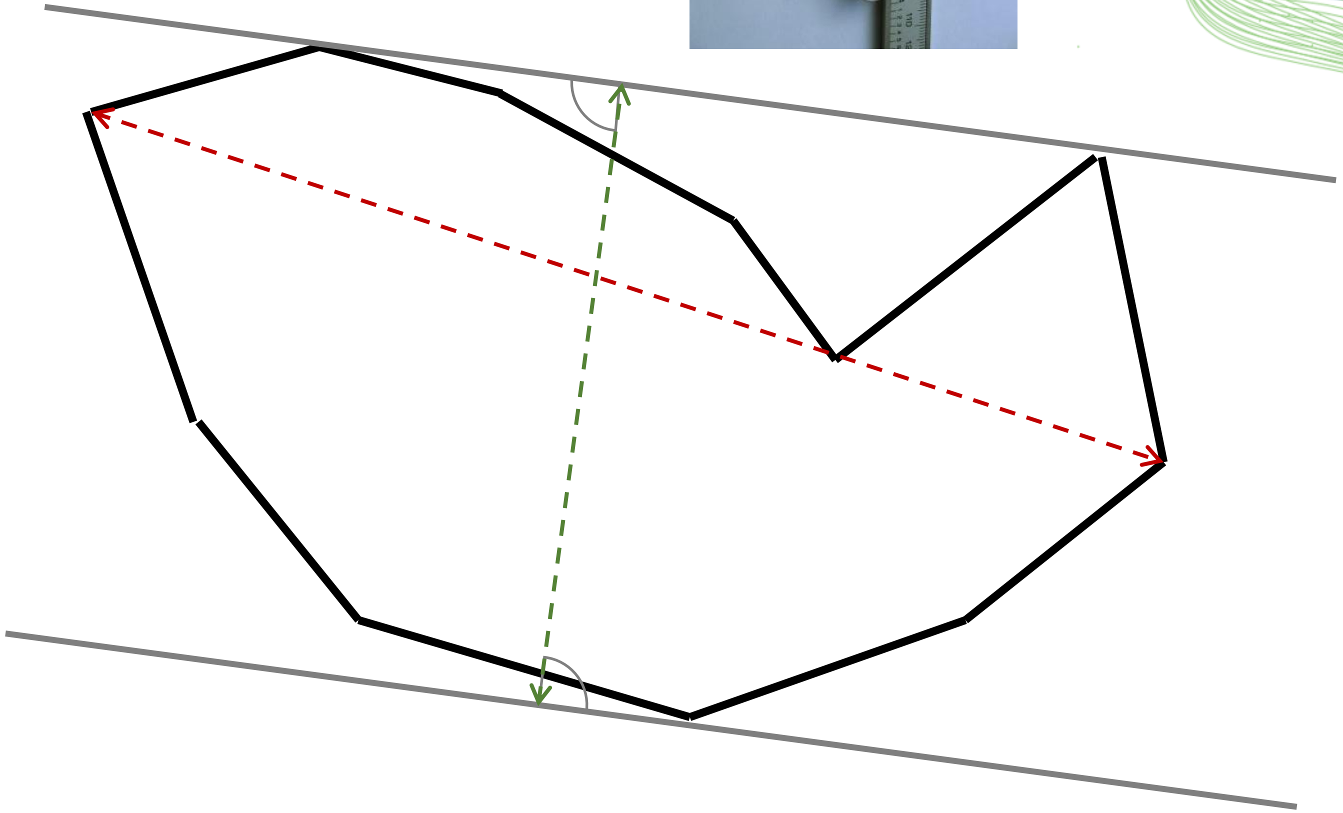




Feret's diameter

- Feret's diameter describes the maximum distance between any two points of an outline.
- The minimum caliper ("Minimum Feret") describes the shortest distance, the object would fit through.
- Feret and Minimum Feret do not need to be perpendicular to each other!

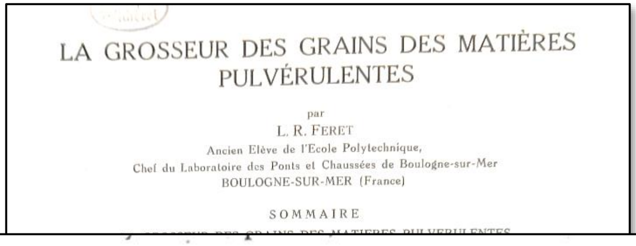




https://en.wikipedia.org/wiki/Feret_diameter#/media/File:DigitalCaliperEuro.jpg Image license: Public domain

Feret’s diameter

- Feret’s diameter (L.R. Feret, 1931) is often cited, but impossible to read online …
- The term “Feret’s Diameter” was established in the 1970s


LA GROSSEUR DES GRAINS DES MATIÈRES PULVÉRULENTES

par

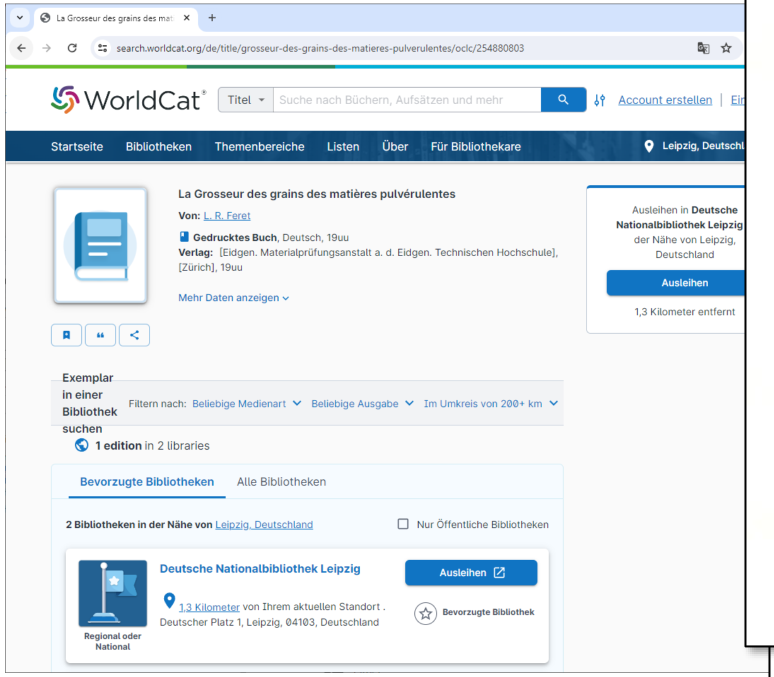
L. R. FERET

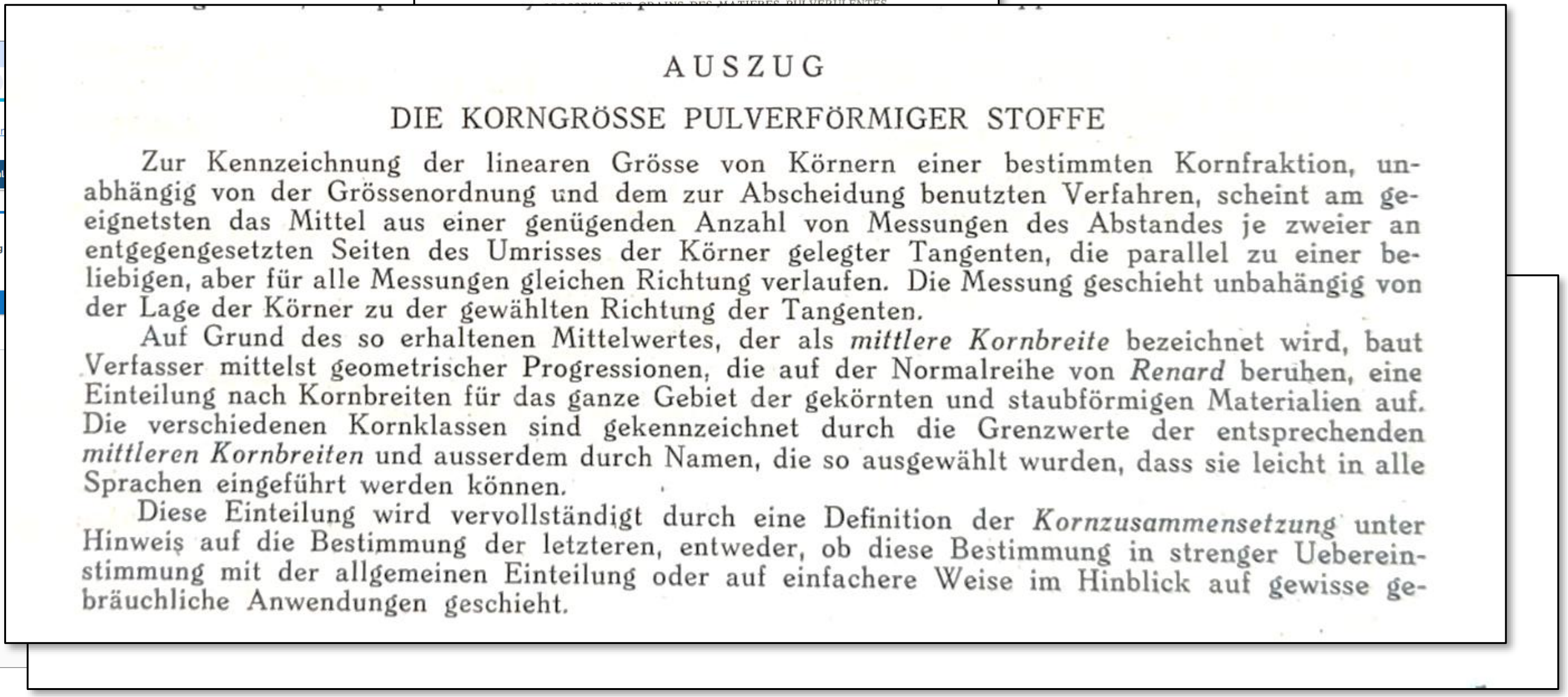
Ancien Elève de l'Ecole Polytechnique,

Chef du Laboratoire des Ponts et Chaussées de Boulogne-sur-Mer

BOULOGNE-SUR-MER (France)

SOMMAIRE




AUSZUG

DIE KORNGRÖSSE PULVERFÖRMIGER STOFFE

Zur Kennzeichnung der linearen Grösse von Körnern einer bestimmten Kornfraktion, unabhängig von der Grössenordnung und dem zur Abscheidung benutzten Verfahren, scheint am geeignetsten das Mittel aus einer genügenden Anzahl von Messungen des Abstandes je zweier an entgegengesetzten Seiten des Umrisses der Körner gelegter Tangenten, die parallel zu einer beliebigen, aber für alle Messungen gleichen Richtung verlaufen. Die Messung geschieht unbahängig von der Lage der Körner zu der gewählten Richtung der Tangenten.

Auf Grund des so erhaltenen Mittelwertes, der als *mittlere Kornbreite* bezeichnet wird, baut Verfasser mittelst geometrischer Progressionen, die auf der Normalreihe von *Renard* beruhen, eine Einteilung nach Kornbreiten für das ganze Gebiet der gekörnten und staubförmigen Materialien auf. Die verschiedenen Kornklassen sind gekennzeichnet durch die Grenzwerte der entsprechenden *mittleren Kornbreiten* und ausserdem durch Namen, die so ausgewählt wurden, dass sie leicht in alle Sprachen eingeführt werden können.

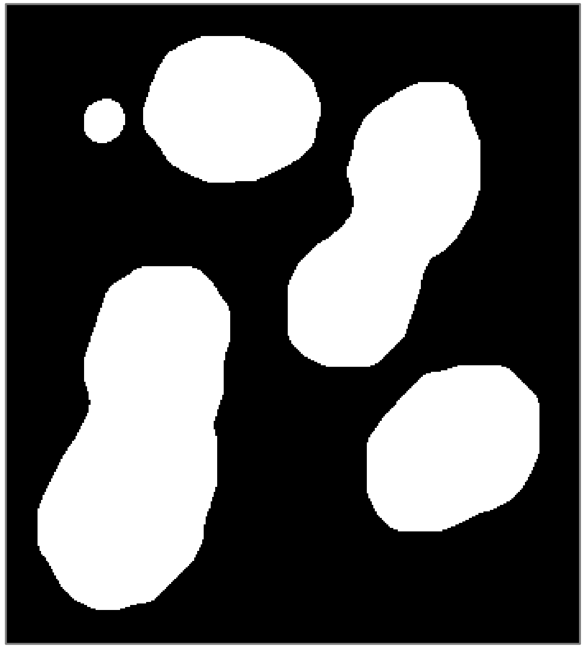
Diese Einteilung wird vervollständigt durch eine Definition der *Kornzusammensetzung* unter Hinweis auf die Bestimmung der letzteren, entweder, ob diese Bestimmung in strenger Uebereinstimmung mit der allgemeinen Einteilung oder auf einfachere Weise im Hinblick auf gewisse gebräuchliche Anwendungen geschieht.

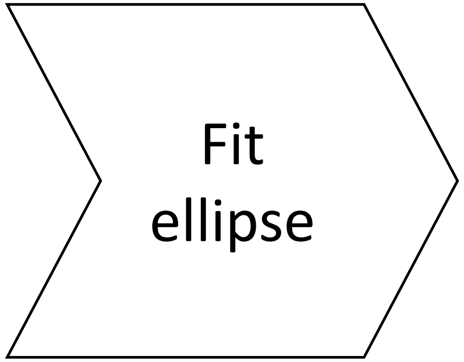
https://search.worldcat.org/de/title/grosseur-des-grains-des-matieres-pulverulentes/oclc/254880803

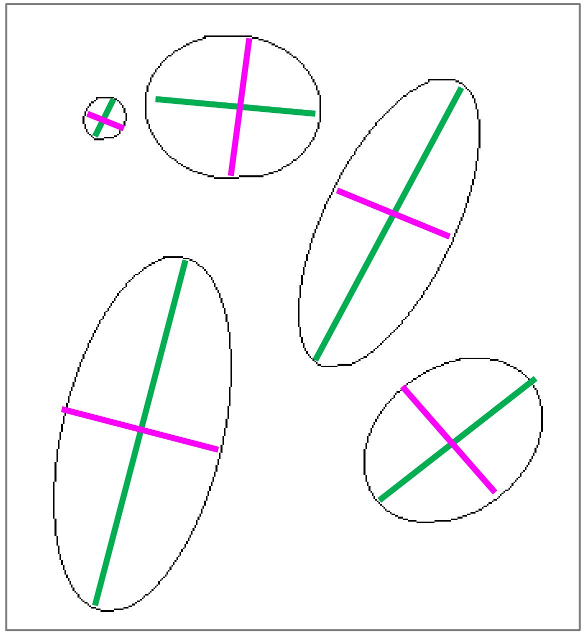
Minor / major axis

- For every object, find the optimal ellipse simplifying the object.
- Major axis … long diameter
- Minor axis … short diameter

- Major and minor axis are perpendicular to each other



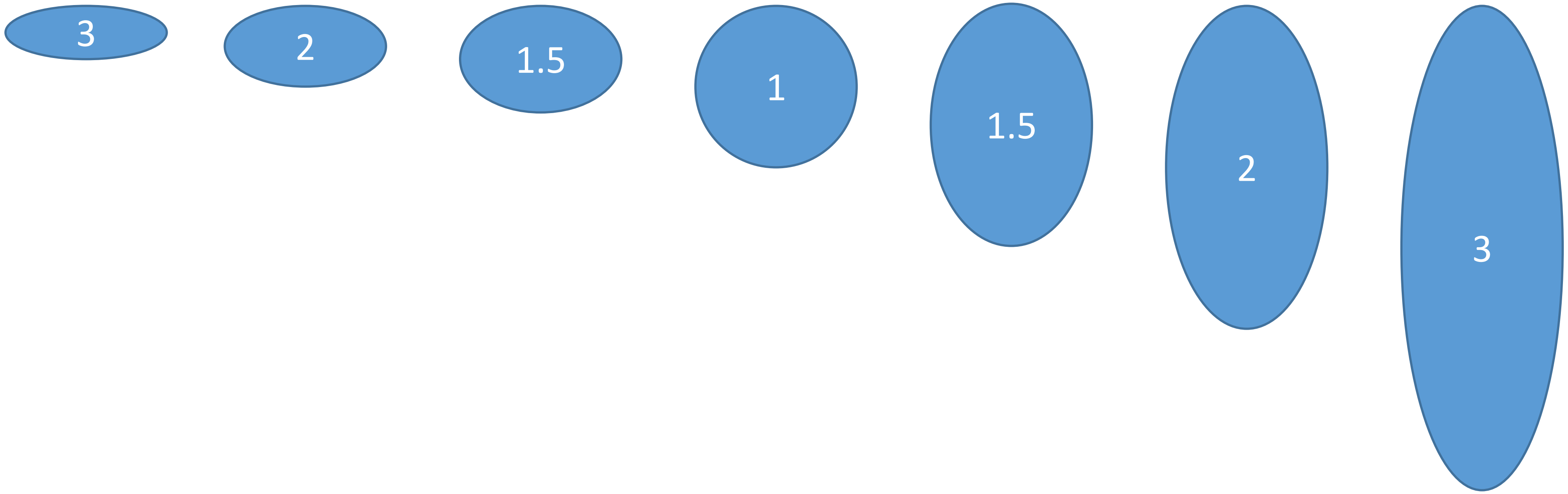




Aspect ratio

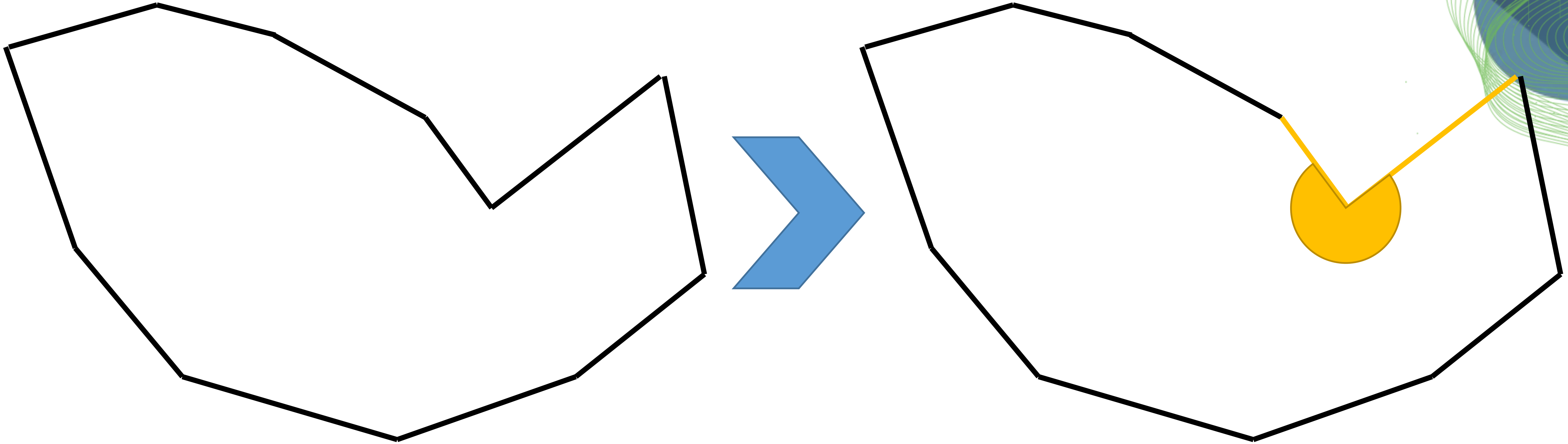
- The aspect ratio describes the elongation of an object.

AR = major / minor



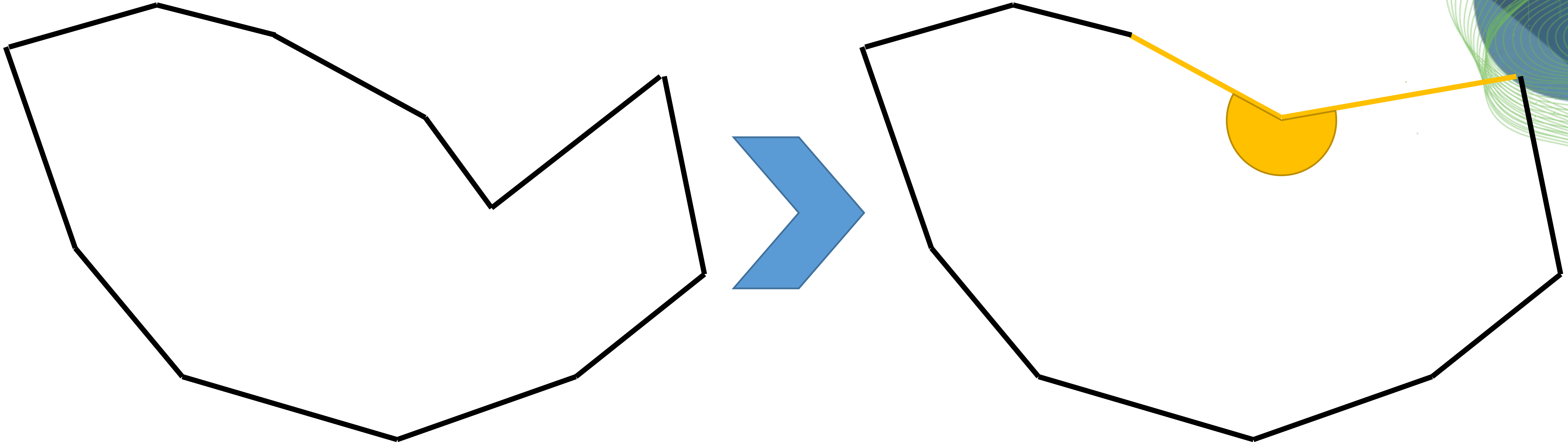
Convex hull

- By removing all concave corners of an object, we retrieve its convex hull.



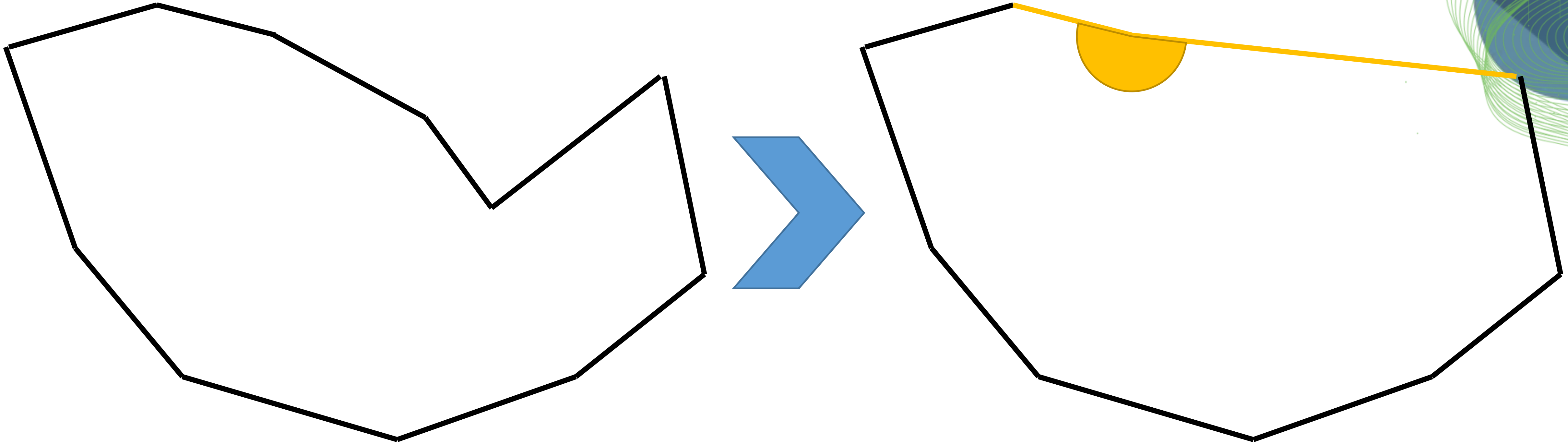
Convex hull

- By removing all concave corners of an object, we retrieve its convex hull.



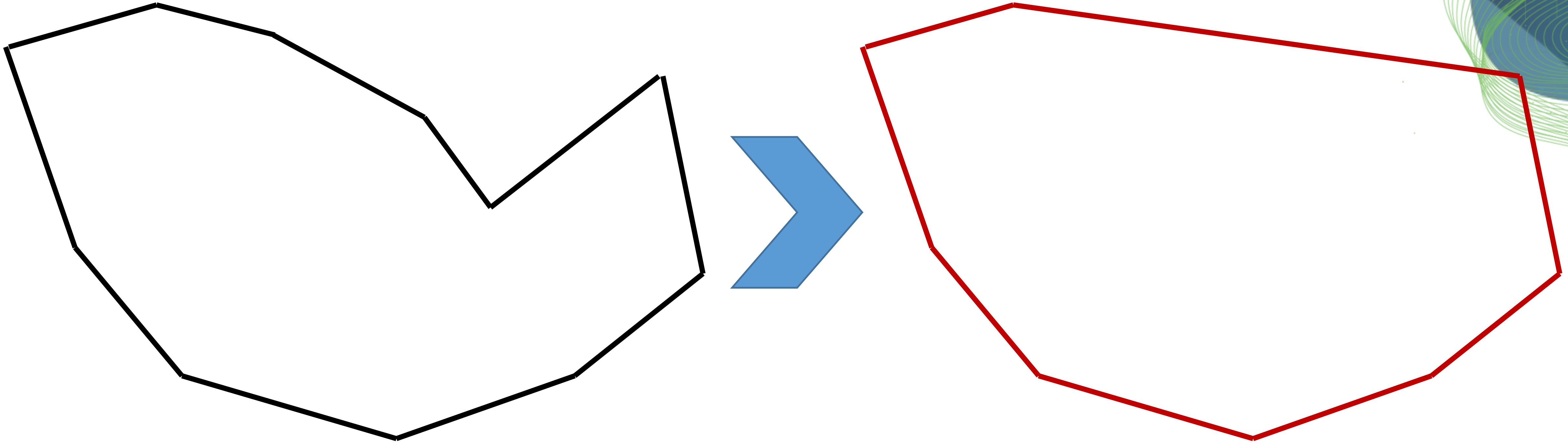
Convex hull

- By removing all concave corners of an object, we retrieve its convex hull.



Convex hull

- By removing all concave corners of an object, we retrieve its convex hull.

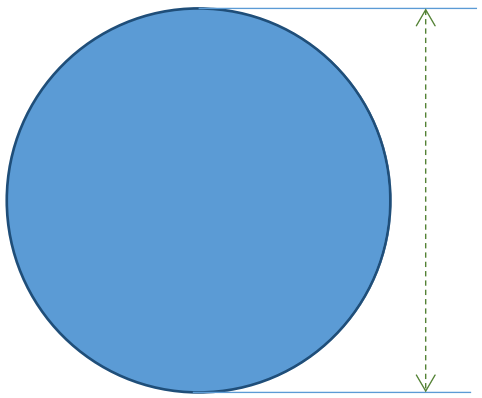


$$solidity = \frac{A}{A_{convexHull}}$$

Roundness and circularity

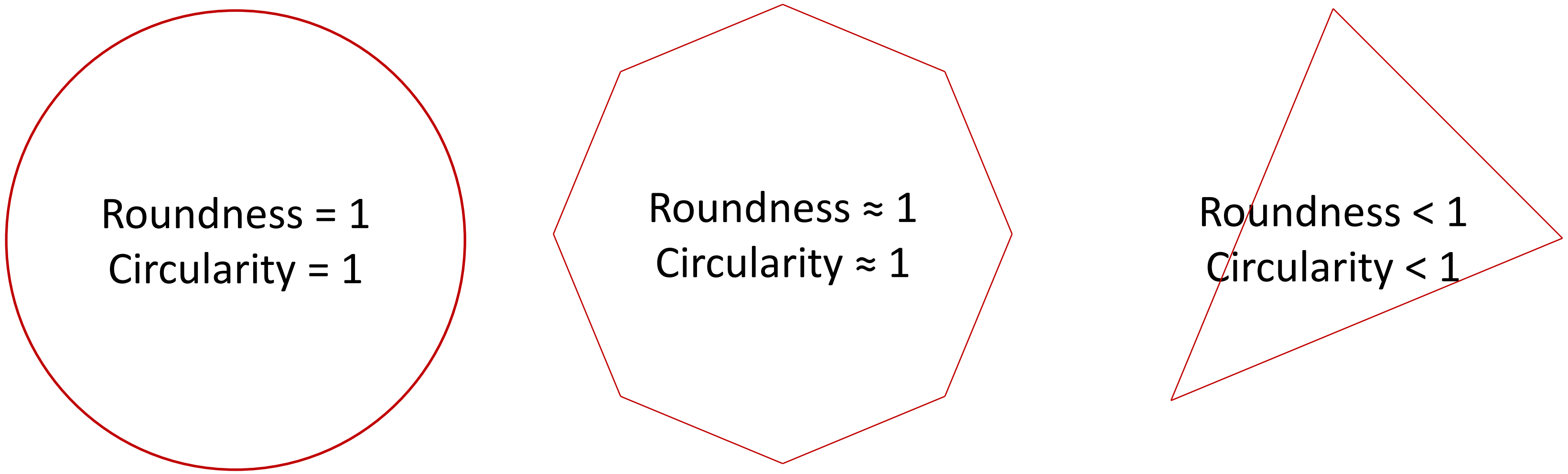
- The definition of a circle leads us to measurements of circularity and roundness.
- In case you use these measures, define them correctly. They are not standardized!

Diameter	d
Circumference	$C = \pi d$
Area	$A = \dfrac{\pi d^2}{4}$



$$roundness = \frac{4 * A}{\pi\ major^2}$$

$$circularity = \frac{4\pi * A}{perimeter^2}$$



Feature extraction in Python

- **In Python:** `from skimage import measure`

  https://scikit-image.org/docs/stable/api/skimage.measure.html

`skimage.measure.regionprops` (label_image[, ...])	Measure properties of labeled image regions.
`skimage.measure.regionprops_table` (label_image)	Compute image properties and return them as a pandas-compatible table.

area : int

Number of pixels of the region.

area_bbox : int

Number of pixels of bounding box.

area_convex : int

Number of pixels of convex hull image, which is the smallest convex polygon that encloses the region.

area_filled : int

Number of pixels of the region will all the holes filled in. Describes the area of the image_filled.

axis_major_length : float

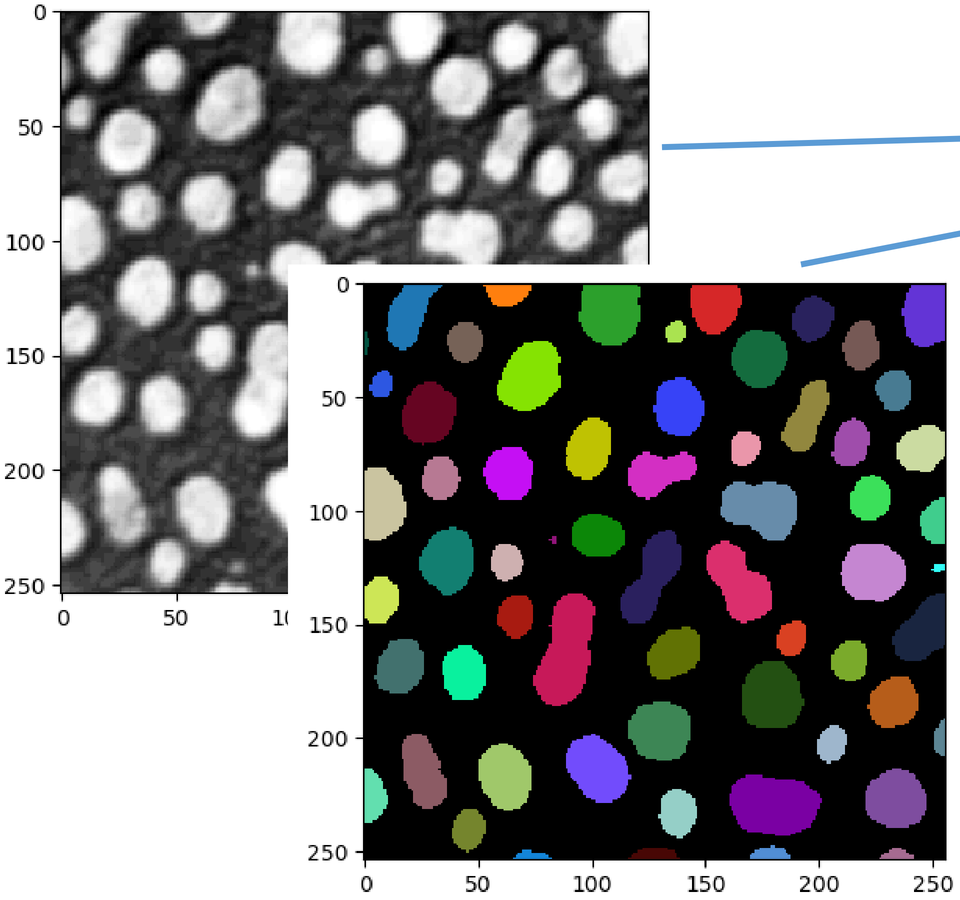
The length of the major axis of the ellipse that has the same normalized second central moments as the region.

axis_minor_length : float

The length of the minor axis of the ellipse that has the same normalized second central moments as the region.

Feature extraction in Python

- The transition from image data to tabular data / pandas DataFrames



[4]:
```python
df = pd.DataFrame(regionprops_table(label_image,
                                     intensity_image=image,
                                     properties=["label", "mean_intensity", "area
                                                 "major_axis_length", "minor_axis

df
```

[4]:

	label	mean_intensity	area	perimeter	major_axis_length	minor_axis_length
0	1	191.440559	429.0	89.012193	34.779230	16.654732
1	2	179.846995	183.0	53.556349	20.950530	11.755645
2	3	205.604863	658.0	95.698485	30.198484	28.282790
3	4	217.515012	433.0	77.455844	24.508791	23.079220
4	5	213.033898	472.0	83.798990	31.084766	19.681190
...	...	...	...	...	...	...

Feature extraction in Python

- The transition from image data to tabular data / pandas DataFrames

[4]:
```
df = pd.DataFrame(regionprops_table(label_image,
                                    intensity_image=image,
                                    properties=["label", "mean_intensity", "area", "perimeter",
                                                "major_axis_length", "minor_axis_length"]))
df
```

[4]:

	label	mean_intensity	area	perimeter	major_axis_length	minor_axis_length
0	1	191.440559	429.0	89.012193	34.779230	16.654732
1	2	179.846995	183.0	53.556349	20.950530	11.755645
2	3	205.604863	658.0	95.698485	30.198484	28.282790
3	4	217.515012	433.0	77.455844	24.508791	23.079220
4	5	213.033898	472.0	83.798990	31.084766	19.681190
...	...	...	...	...	...	...

Feature extraction in Python

- Customized features passed as function(s).

[5]:
```python
def standard_deviation_intensity(region, intensities):
    return np.std(intensities[region])

df = pd.DataFrame(regionprops_table(label_image,
                                    intensity_image=image,
                                    properties=["label", "mean_intensity", "area", "perimeter",
                                                "major_axis_length", "minor_axis_length"],
                                    extra_properties=[standard_deviation_intensity]))
df
```

[5]:

	label	mean_intensity	area	perimeter	major_axis_length	minor_axis_length	standard_deviation_intensi
0	1	191.440559	429.0	89.012193	34.779230	16.654732	29.7931
1	2	179.846995	183.0	53.556349	20.950530	11.755645	21.2705
2	3	205.604863	658.0	95.698485	30.198484	28.282790	29.3922
3	4	217.515012	433.0	77.455844	24.508791	23.079220	35.8523
4	5	213.033898	472.0	83.798990	31.084766	19.681190	28.7410
...	...	...	...	...	...	...	

Feature extraction in Python

- Customized features computed afterwards.

Quiz: Why didn't we compute standard deviation of the intensity like this?

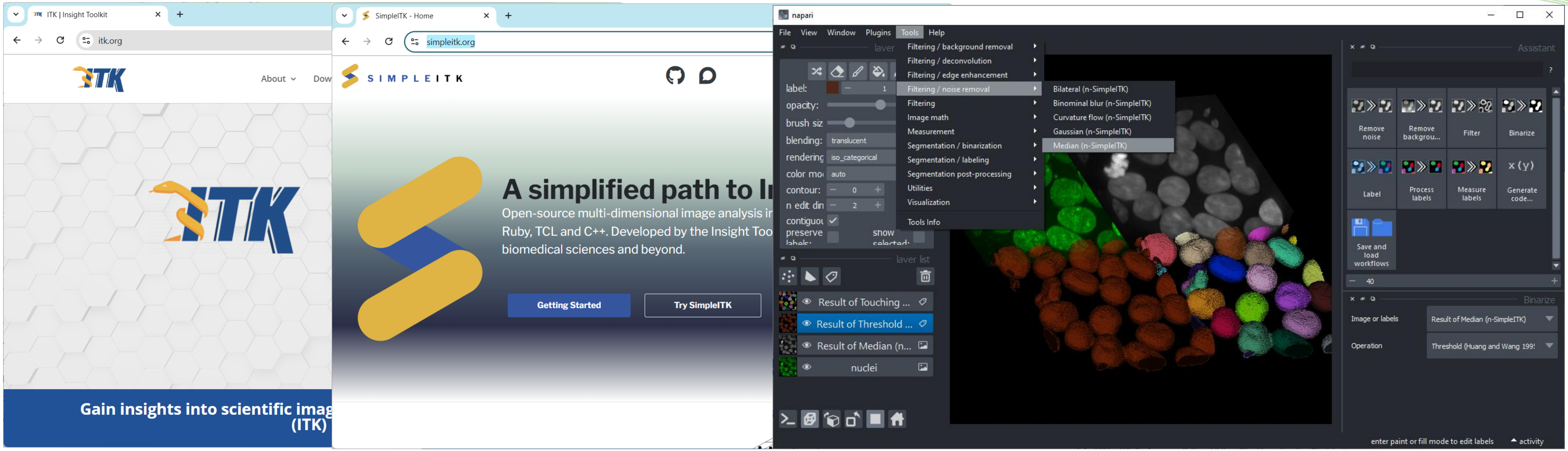
[6]:
```
df['roundness'] = 4 * df['area'] / np.pi / pow(df['major_axis_length'], 2)
df['circularity'] = 4 * np.pi * df['area'] / pow(df['perimeter'], 2)

df
```

[6]:

	label	mean_intensity	area	perimeter	major_axis_length	minor_axis_length	standard_deviation_intensity	roundness	circularity
0	1	191.440559	429.0	89.012193	34.779230	16.654732	29.793138	0.451572	0.680406
1	2	179.846995	183.0	53.556349	20.950530	11.755645	21.270534	0.530849	0.801750
2	3	205.604863	658.0	95.698485	30.198484	28.282790	29.392255	0.918683	0.902871
3	4	217.515012	433.0	77.455844	24.508791	23.079220	35.852345	0.917813	0.906963
4	5	213.033898	472.0	83.798990	31.084766	19.681190	28.741080	0.621952	0.844645
...	...	...	...	...	...	...	...	...	...

SimpleITK

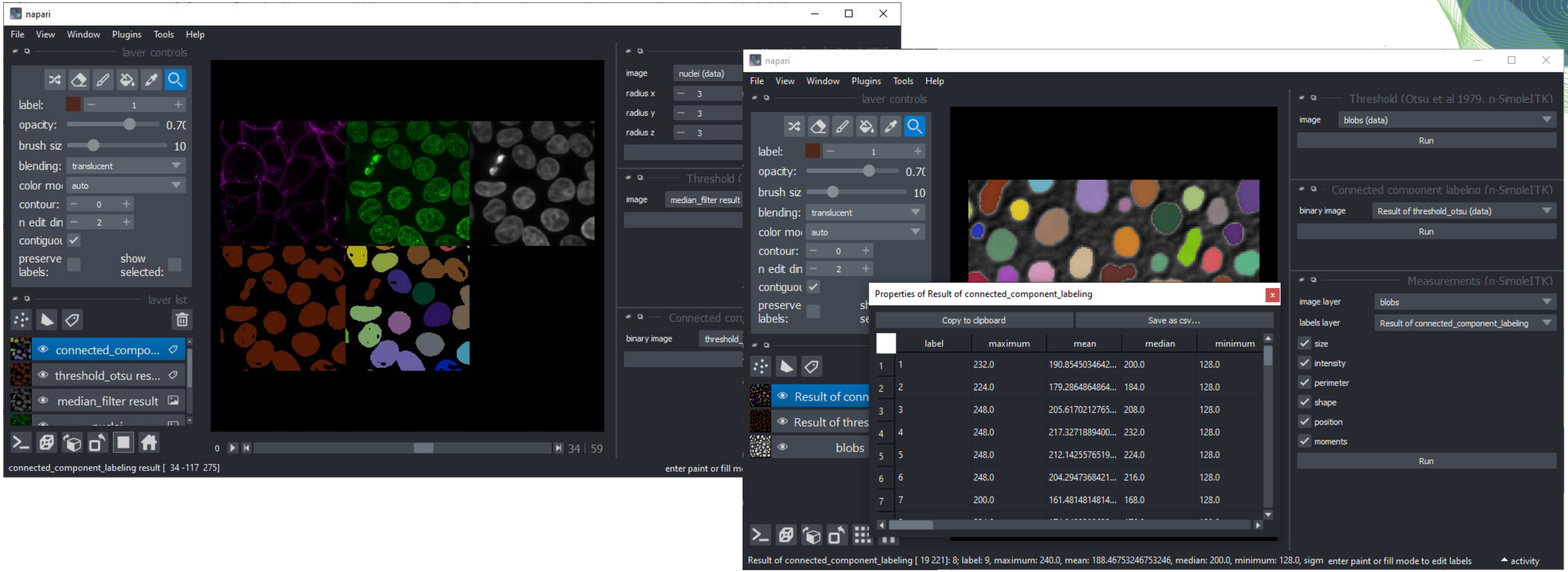
- ITK: Insight Toolkit, a [medical] image processing library, written in C++, originating in the 80s.
- SimpleITK: A Python wrapper around ITK
- Napari-simpleitk-image-processing: A Napari Plugin and simplifaction wrapper around simple-itk.



Robert Haase
@haesleinhuepf
BIDS Lecture 4/14
May 9th 2025

SimpleITK in Napari

- Menu Tools > Measurement Tables > Measurements (n-SimpleITK)



	label	maximum	mean	median	minimum
1	1	232.0	190.8545034642...	200.0	128.0
2	2	224.0	179.2864864864...	184.0	128.0
3	3	248.0	205.6170212765...	208.0	128.0
4	4	248.0	217.3271889400...	232.0	128.0
5	5	248.0	212.1425576519...	224.0	128.0
6	6	248.0	204.2947368421...	216.0	128.0
7	7	200.0	161.4814814814...	168.0	128.0

SimpleITK

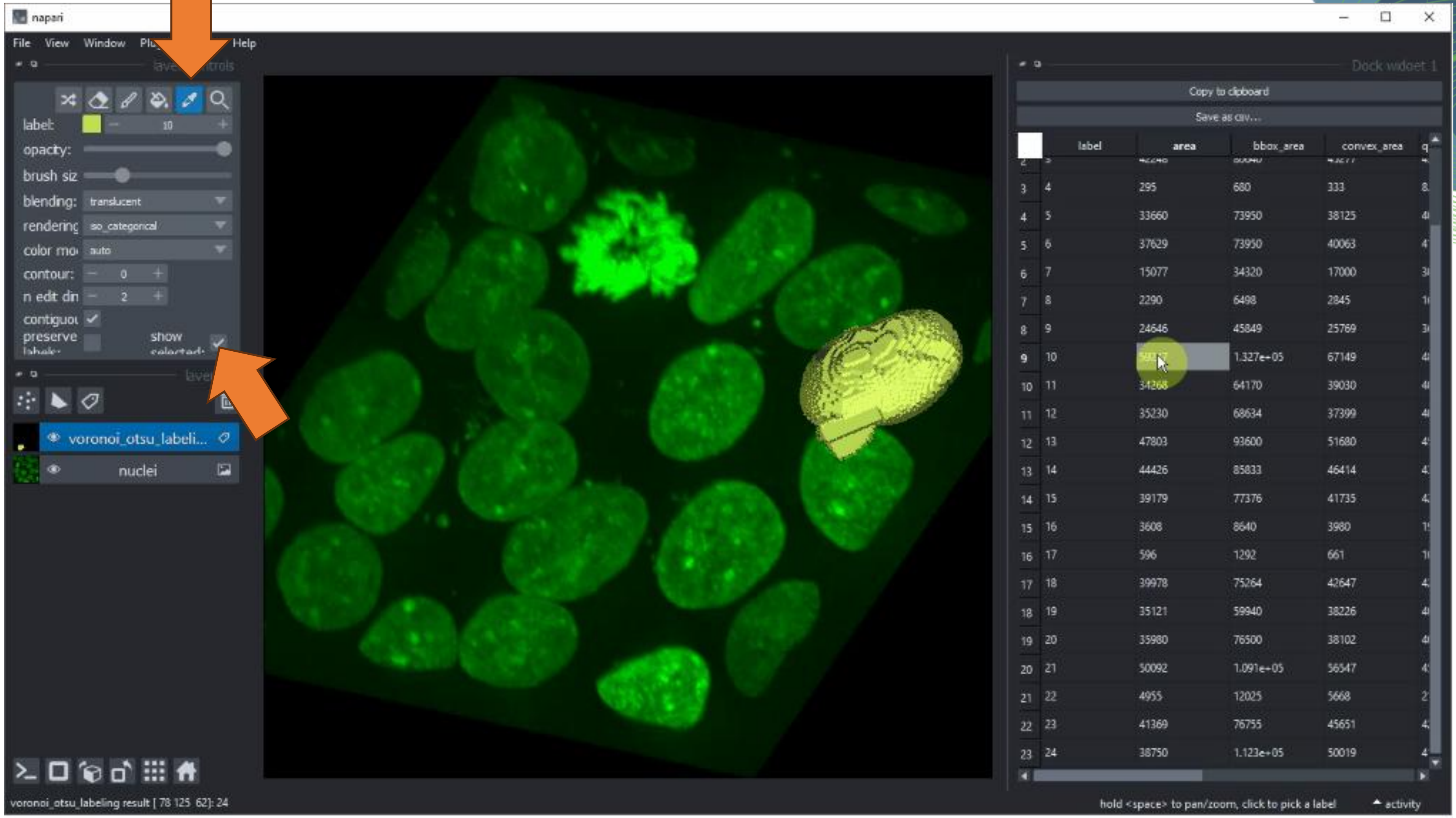
- The napari-plugin for creating tables can also be called from Python.
- **Recommended for working with 3D data.**

```
statistics = label_statistics(blobs, labels,
                              intensity=True,
                              size=True,
                              shape=True,
                              perimeter=True,
                              position=True,
                              moments=True)

df = pd.DataFrame(statistics)
df
```

	label	maximum	mean	median	minimum	sigma	sum	variance	bbox_0	bbox_1
0	1	224.0	137.526132	136.0	112.0	13.360739	157880.0	178.509343	0	0
1	2	232.0	193.014354	200.0	128.0	28.559077	80680.0	815.620897	11	0
2	3	224.0	179.846995	184.0	128.0	21.328889	32912.0	454.921516	53	0

Exploring features in Napari

- Select table rows and view corresponding object in 2D/3D space

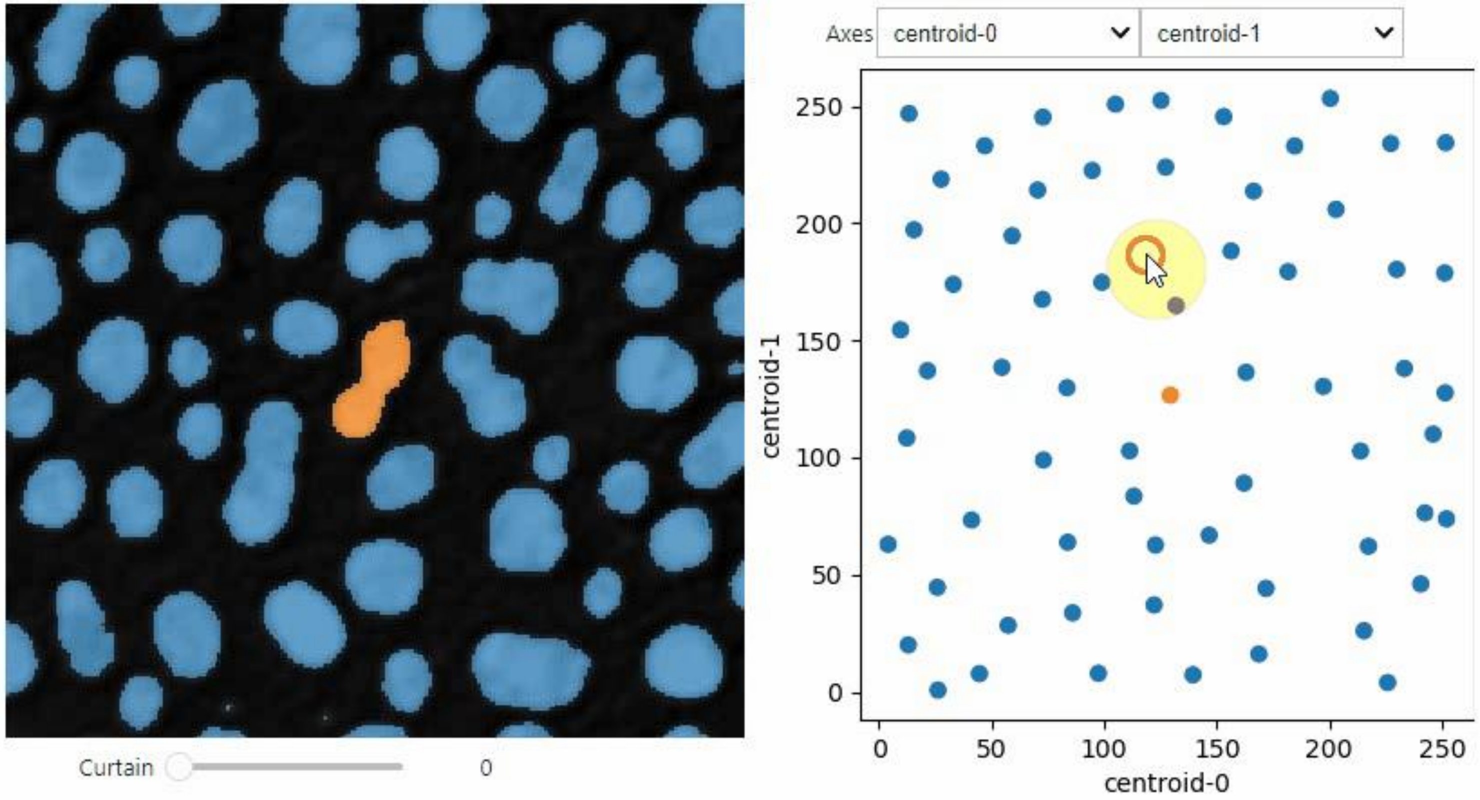


Selecting objects according to their properties

- Understanding what features *mean* may require interactive user interfaces

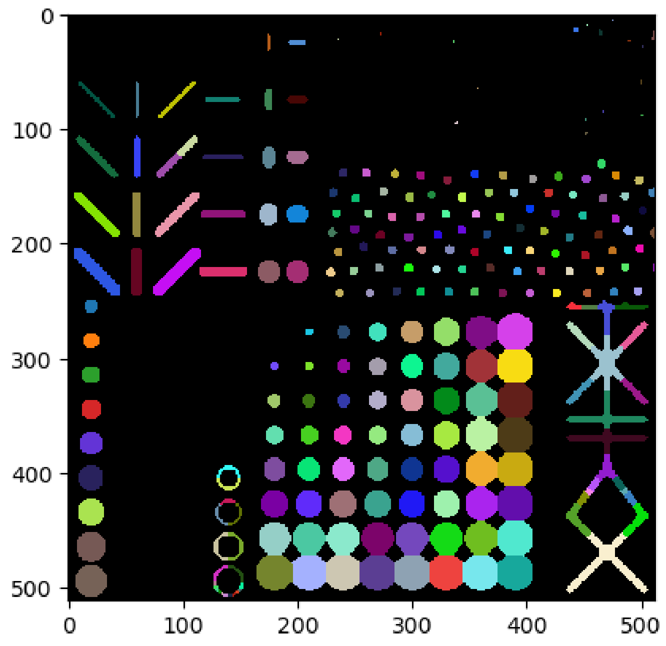
```
[6]: stackview.clusterplot(image=image,
                           labels=labeled_image,
                           df=df,
                           column_x="area",
                           column_y="aspect_ratio",
                           zoom_factor=1.6,
                           alpha=0.7)
```

[6]:

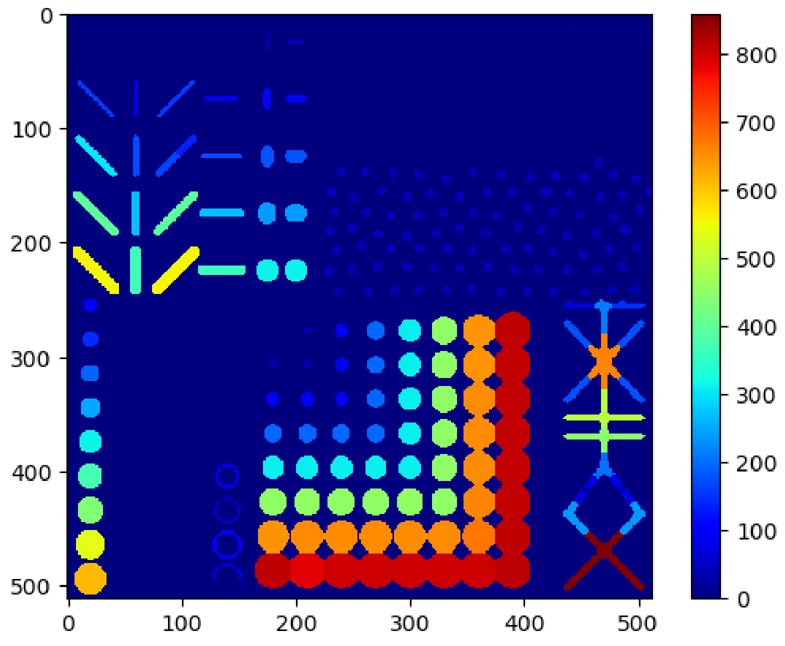


Parametric images

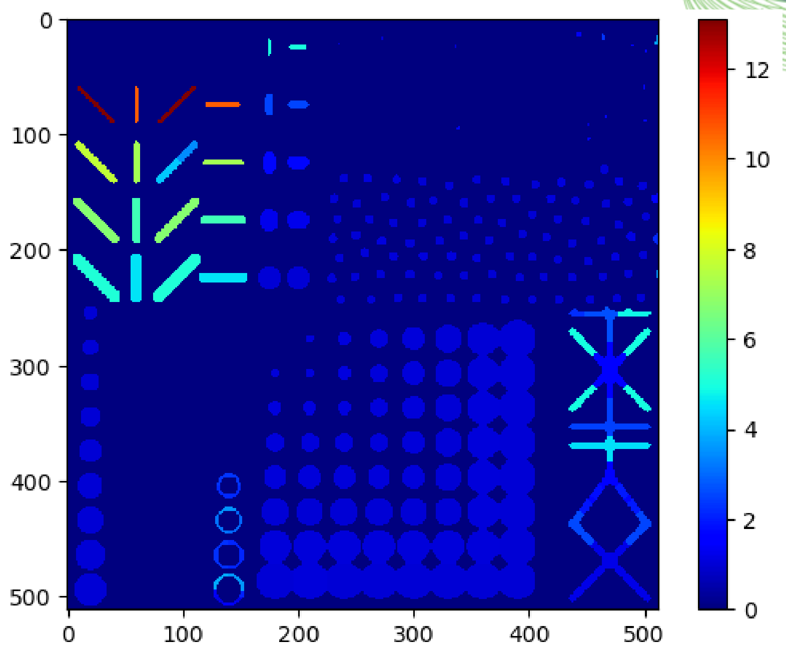
- Visualizing quantitative measurements in image space.



Label image



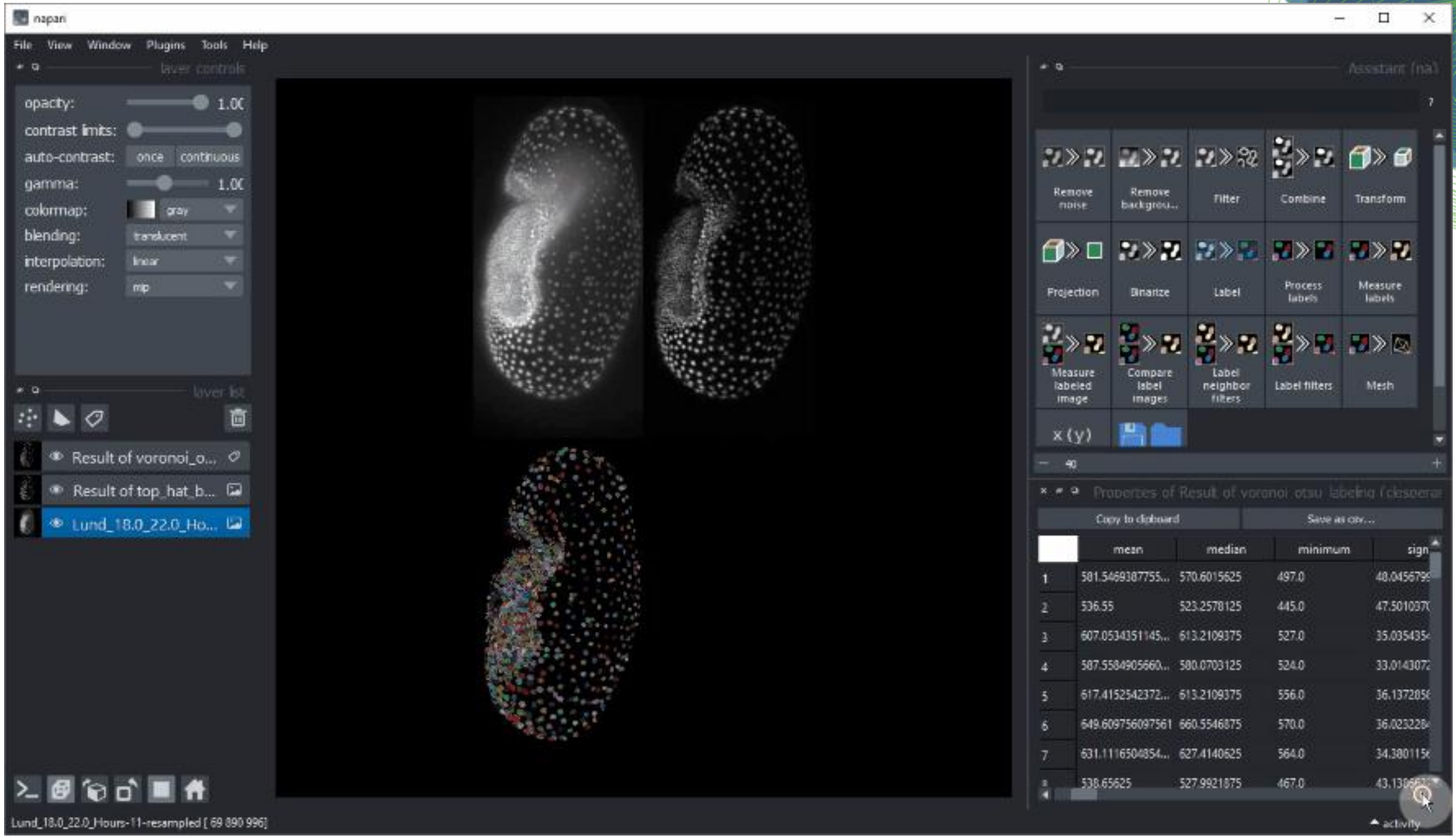
Pixel count image



Aspect ratio image

Exploring features in Napari

- Double-click on table column to retrieve a parametric map image



ScaDS.AI DRESDEN LEIPZIG
CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

Complex exercise

Robert Haase

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

SACHSEN
Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Complex exercise

- Scenario: Imagine a biologist sent you some data (images + maybe corresponding label image). They ask you to write an image-analysis workflow for processing these images + more images of similar kind.
- You will receive a link to data from me
  - You can return the link and exchange it with another link 2 times.
- Scientific tasks
  - Develop an image-segmentation workflow, which produces label images (if given)
  - Extract features from these images, at least area/volume of objects + 1 more feature
  - Plot area [or volume] against another feature.
  - Visualize an area / volume parametric image.

Complex exercise

- Engineering tasks
  - Setup a software environment
  - Setup an image processing workflow
  - Setup a data analysis / visualization workflow
  - Setup a quality assurance procedure
- Documentation tasks
  - Installation instructions
  - User guide
  - Documentation of used data
  - Explanation of the used algorithms

Act as if you would communicate with a biologist, with limited image-analysis, conda, and programming skills.

Complex exercise

- Submission
  - Submit a password-protected ZIP file to robert.haase@uni-leipzig.de (Why password protected: The virus scanner cannot reject python files in encrypted zip-files)
  - Allowed file formats: ipynb, py, docx, pdf, md, csv, yml, json, xml, txt
  - Deadline: July 4th 2025
- Hint
  - Send this ZIP file to a friend and ask them to run the analysis. If they can follow your instructions successfully, without communicating with you, proceed to final submission.

Complex exercise

- Checklist
  - The software environment is reproducible
  - The example data is available in the right directory (note: you cannot submit a 500MB ZIP file via email)
  - The image/data analysis code is executable
  - The code is well documented / commented
  - Segmentation results are visualized
  - Segmentation results are stored to disc as label images
  - The quality of the segmentation result is measured
  - Used algorithms are cited, and well explained
  - Extracted features / measurements are saved as CSV-file in a way that one can associate an entry in the CSV file with the corresponding segmented object
  - Resulting plots and visualizations have reasonable axis labels and are well explained
  - The copyright of re-used data and code are respected

Complex exercise

- Option A: You do the analysis yourself. Minor question-answer interactions with ChatGPT etc are allowed.
- Option B: You use a Large Language Model to do it.
  If you choose this option, you need to submit all prompts you used.

ScaDS.AI DRESDEN LEIPZIG
CENTER FOR SCALABLE DATA ANALYTICS
AND ARTIFICIAL INTELLIGENCE

Exercises

Robert Haase

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

SACHSEN
Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Exercise: Surface meshes

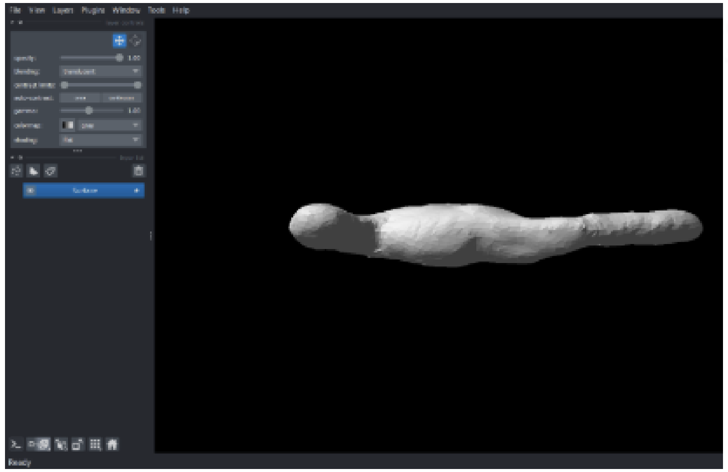
- Creating, storing, processing surface mesh data
- Task: Reproduce a specific view in Napari.
- Challenge: Try to code the view-adaptation in Napari in one minute or less.

```
napari.utils.nbscreenshot(viewer)
```

[6]:

[7]:
```
viewer.camera.angles = [0,0,0]

napari.utils.nbscreenshot(viewer)
```

[7]:



[9]:
```
napari.utils.nbscreenshot(viewer)
```

[9]:

Exercise: Quantitative measurements

- Use the given feature extraction notebook to apply some basic statistics to measurements

[5]:
```
df = pd.DataFrame(regionprops_table(image , label_image,
                                     perimeter = True,
                                     shape = True,
                                     position=True,
                                     moments=True))
df
```

[5]:

	label	area	bbox_area	equivalent_diameter	convex_area	max_intensity	mean_intensity	min_intensity	perimeter	perimete
0	1	429.0	750.0	23.371345	479.0	232.0	191.440559	128.0	89.012193	
1	2	183.0	231.0	15.264430	190.0	224.0	179.846995	128.0	53.556349	
2	3	658.0	756.0	28.944630	673.0	248.0	205.604863	120.0	95.698485	
3	4	433.0	529.0	23.480049	445.0	248.0	217.515012	120.0	77.455844	
4	5	472.0	551.0	24.514670	486.0	248.0	213.033898	128.0	83.798990	
...	...	...	...	...	...	...	...	...	...	
57	58	213.0	285.0	16.468152	221.0	224.0	184.525822	120.0	52.284271	
58	59	79.0	108.0	10.029253	84.0	248.0	184.810127	128.0	39.313708	
59	60	88.0	110.0	10.585135	92.0	216.0	182.727273	128.0	45.692388	
60	61	52.0	75.0	8.136858	56.0	248.0	189.538462	128.0	30.692388	
61	62	48.0	68.0	7.817640	53.0	224.0	173.833333	128.0	33.071068	

62 rows × 86 columns

Exercises

Make a table with only `area`, `mean_intensity`, `standard_deviation_intensity` and `label`.

[ ]:

How many object are in the dataframe?

[ ]:

How large is the largest object?

[ ]:

What is the mean intensity of the brightest object?

[ ]:

What are mean and standard deviation intensity of the image?

[ ]:

Exercise: Parametric maps

- Interpreting parameters: What's the difference between elongation and flatness?
- Interactive tools might help.

```
[5]: stackview.clusterplot(df, label_image, column_x="number_of_pixels", column_y="perimeter")
```