

# Bioimage Analysis with Fiji / ImageJ

**Wei-Chen CHU (朱韋臣)**

Project Scientist @Imaging Core Facility  
Institute of Cellular and Organismic Biology (ICOB),  
Academia Sinica

PowerPoint Slide:



This material is licensed by Wei-Chen CHU under CC BY 4.0 license  
<https://creativecommons.org/licenses/by/4.0/>



@WeiChenCHU1

2023/07/19



ICOB Imaging Core

# Outline

## Part I: Basic of Basic

- Terminology
- FIJI / ImageJ introduction
- Image format and visualization
- Region of Interest (ROI), Measurement

## Part II: Typical bioimage analysis workflow

- Segmentation
- Selection, ROI List, Mask, Label
- ImageJ Macro
- CLIJ (GPU accelerated image processing)
- Interactive bioimage analysis and automatic code generation
- Future of bioimage analysis (AI)



# Part of this document is adapted from the following material under CC BY 4.0



**Dr. Robert Haase**

Clusters Excellence “Physic of Life” (PoL),  
TU Dresden, Germany

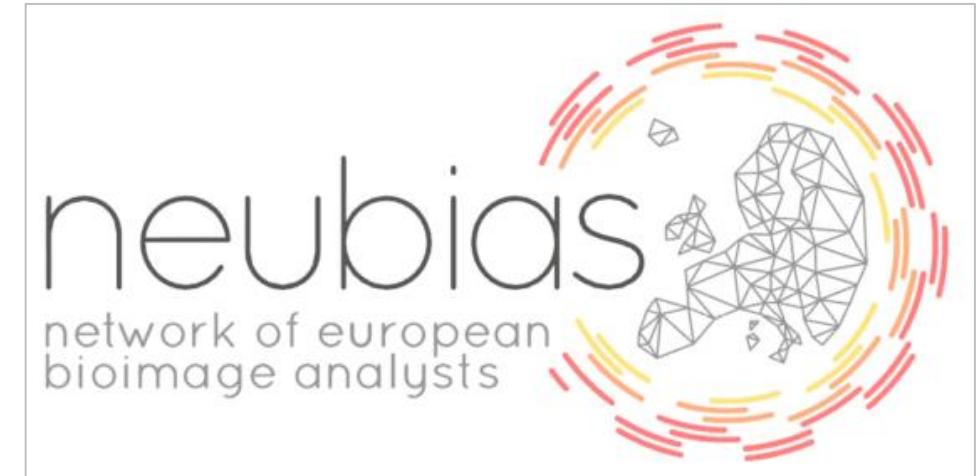
<https://haesleinhuepf.github.io/>



**Dr. Peter Bankhead**

University of Edinburgh, UK

Introduction to Bioimage Analysis (2022)  
<https://bioimagebook.github.io/>

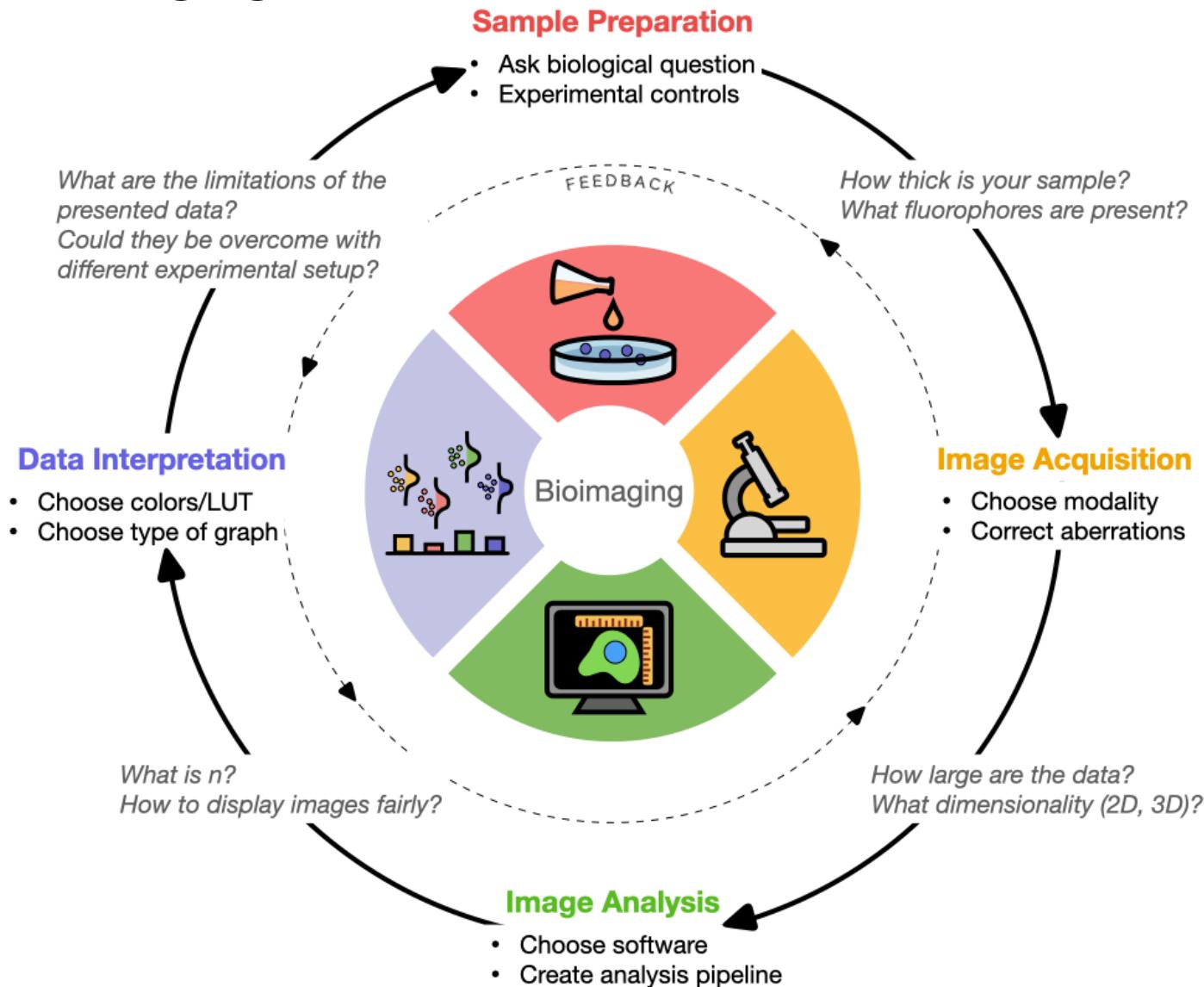


<https://neubias.github.io/training-resources/index.html>

<https://www.youtube.com/@NEUBIAS>



# Quantitative bioimaging



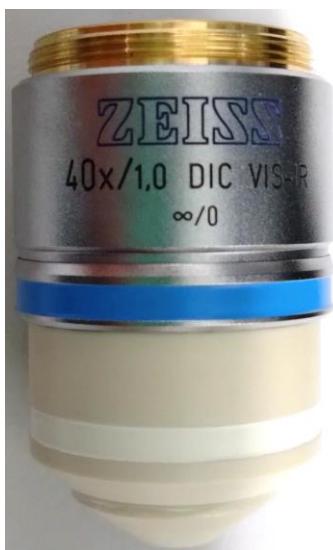
# Coverslip thickness is important!

- Most objectives are designed to use #1.5 coverslips!
- Better to use #1.5H for high resolution image

Most



Exception



Dipping objective

Objective with  
correction ring

| Coverslip No. | Ideal thickness | Range                   |
|---------------|-----------------|-------------------------|
| #0            | 0.10 mm         | 0.080 - 0.130 mm        |
| #1            | 0.15 mm         | 0.130 - 0.170 mm        |
| <b>#1.5</b>   | <b>0.17 mm</b>  | <b>0.160 - 0.190 mm</b> |
| <b>#1.5H</b>  | <b>0.17 mm</b>  | <b>0.165 - 0.175 mm</b> |
| #2.0          | 0.22 mm         | 0.190 – 0.250 mm        |

## Performance Reduction

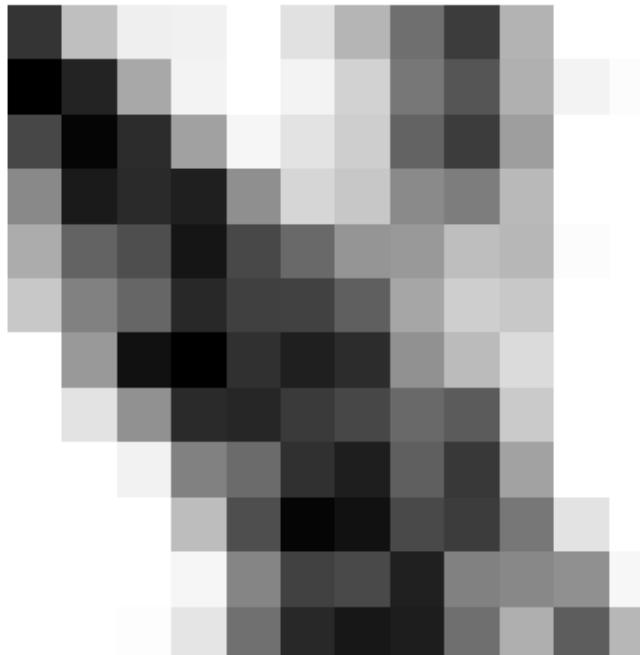
| Objective NA. | 0.01 mm Deviation | 0.02 mm Deviation |
|---------------|-------------------|-------------------|
| 0.30          | -                 | -                 |
| 0.45          | -                 | -                 |
| 0.70          | -2%               | -8%               |
| 0.85          | -19%              | -57%              |
| 0.95          | -55%              | -71%              |

# Image are composed of pixels (picture element)

(A) Original image



(B) Enlarged view from (A)



(C) Pixel values from (B)

|     |     |     |     |     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 53  | 191 | 239 | 241 | 255 | 225 | 181 | 111 | 61  | 180 | 255 | 255 |
|     | 35  | 168 | 244 | 255 | 243 | 210 | 119 | 85  | 176 | 244 | 252 |
| 71  | 5   | 45  | 161 | 246 | 227 | 206 | 99  | 60  | 158 | 255 | 255 |
| 137 | 26  | 42  | 31  | 143 | 214 | 199 | 138 | 125 | 185 | 255 | 255 |
| 172 | 99  | 78  | 21  | 72  | 106 | 149 | 153 | 190 | 183 | 252 | 255 |
| 200 | 129 | 102 | 41  | 64  | 65  | 95  | 166 | 206 | 200 | 255 | 255 |
| 255 | 153 | 17  |     | 49  | 31  | 44  | 145 | 187 | 219 | 255 | 255 |
| 255 | 227 | 145 | 42  | 38  | 58  | 71  | 106 | 91  | 202 | 255 | 255 |
| 255 | 255 | 242 | 129 | 107 | 48  | 30  | 95  | 57  | 162 | 255 | 255 |
| 255 | 255 | 255 | 189 | 78  | 5   | 17  | 74  | 60  | 119 | 228 | 255 |
| 255 | 255 | 255 | 246 | 133 | 65  | 73  | 32  | 129 | 136 | 144 | 247 |
| 255 | 255 | 253 | 229 | 112 | 40  | 23  | 29  | 111 | 175 | 93  | 183 |



0

255

# Bit-depth

- A bit is the smallest memory unit in computers.
- The bit-depth  $n$  enumerates how many different intensity values are present in an image:
  - $2^n$  grey values
- In microscopy, images are usually stored as 8, 12 or 16-bit images.

Bit = "0" or "1"

1 Bytes = 8 bits -> basic store unit

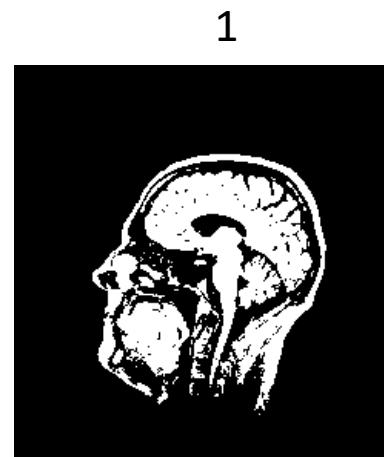
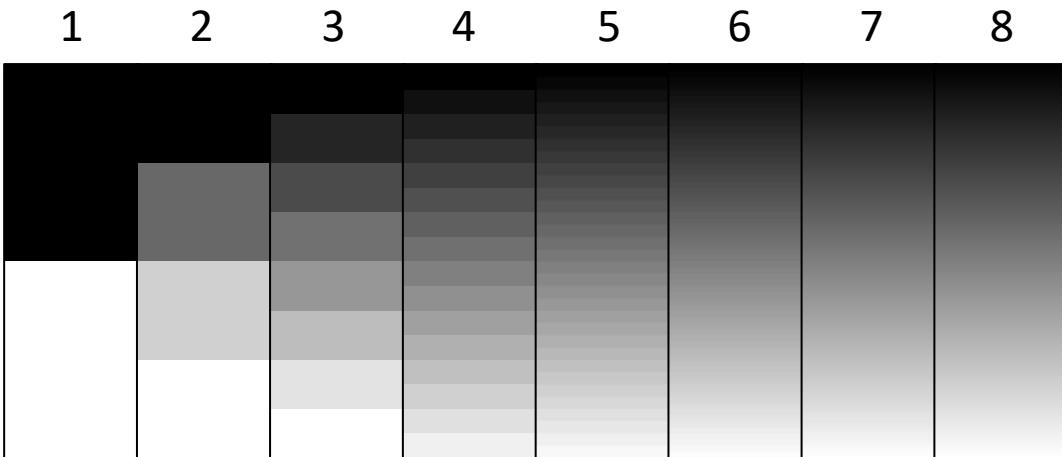
8 bits (1 byte) = 0 ~ 255

16 bits (2 bytes) = 0 ~ 65535

12 bits (2 bytes!) = 0 ~ 4095

14 bits (2 bytes!) = 0 ~ 16383

32 bits (4 bytes, floating-point)



A ‘*Lookup table*’ (*LUT*) determines which color will be used to represent each value

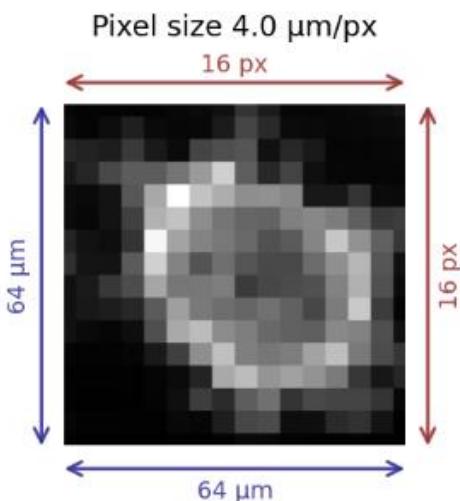
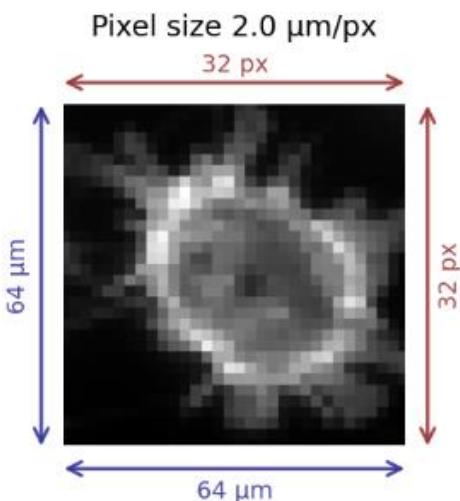
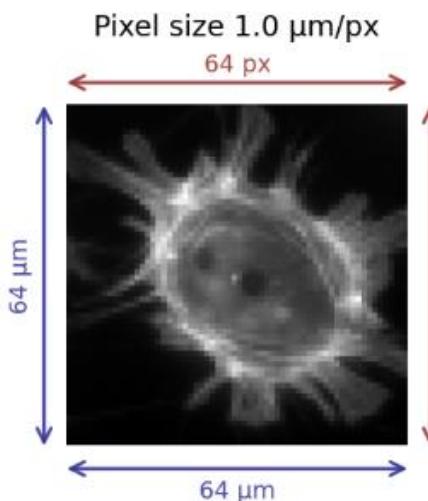
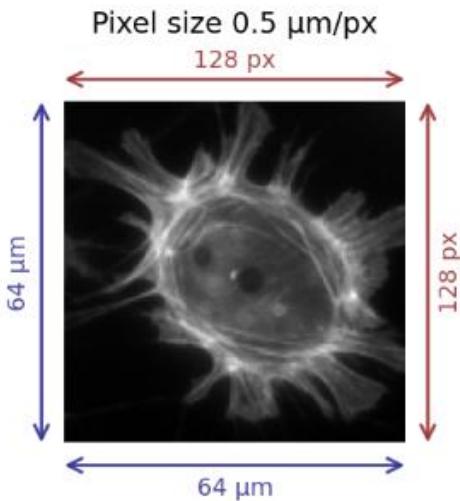
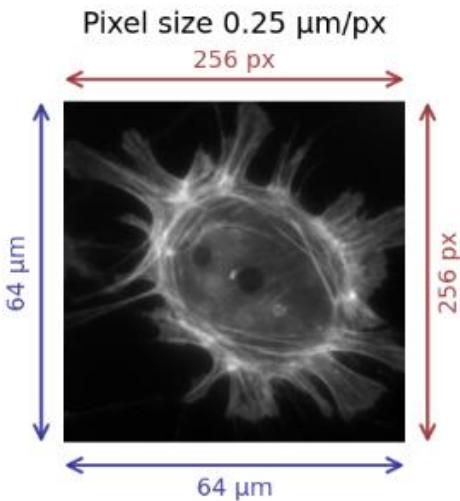
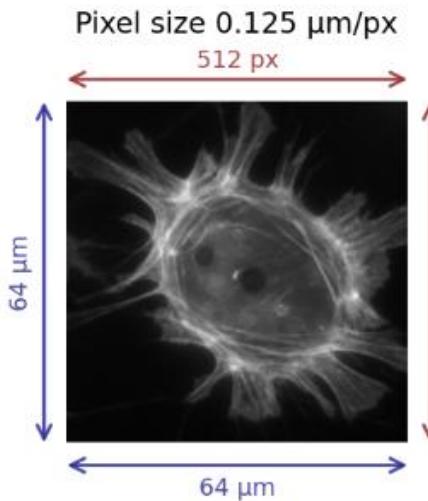
|    |     |     |     |     |     |     |    |
|----|-----|-----|-----|-----|-----|-----|----|
| 4  | 6   | 58  | 144 | 183 | 53  | 7   | 3  |
| 2  | 34  | 208 | 239 | 248 | 212 | 74  | 5  |
| 83 | 138 | 232 | 242 | 235 | 249 | 171 | 3  |
| 68 | 223 | 219 | 174 | 202 | 229 | 87  | 4  |
| 5  | 152 | 146 | 47  | 150 | 223 | 89  | 4  |
| 0  | 99  | 213 | 121 | 136 | 110 | 180 | 4  |
| 1  | 115 | 209 | 252 | 199 | 216 | 255 | 38 |
| 3  | 161 | 148 | 231 | 213 | 213 | 201 | 51 |

*Pixel values*



*Lookup table (LUT)*

# Pixel Size



## Quiz:

Pixel size: 4  $\mu\text{m}$

10 px

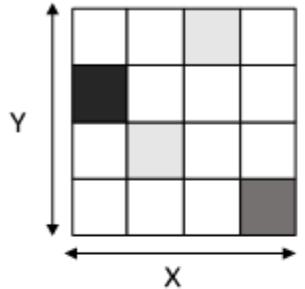


5 px

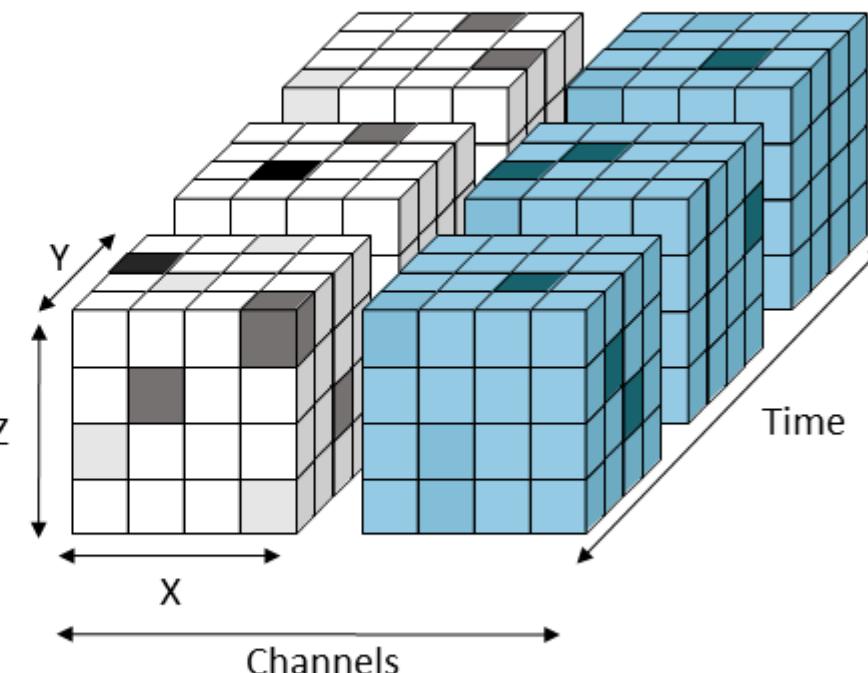
What is the area of the structure in  $\mu\text{m}^2$  ?

# N-dimensional images

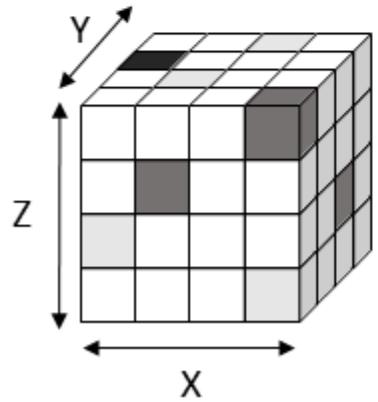
2D image



5D image



3D image



- Most fluorescence image: (n color -> n Channels)
- Data from color camera: RGB format (n color -> R/G/B Value, 24 bits)

# (Fiji is just) imageJ



# Fiji

Fiji is an image processing package — a "batteries-included" distribution of [ImageJ](#), bundling many plugins which facilitate scientific image analysis.

 Download for Windows (64-bit) ▾

---

 More Downloads  Cite  Contribute

## Why Fiji?



### Easy to Use

Fiji is easy to use and install - in one-click, Fiji installs all of its plugins, features an automatic updater, and offers comprehensive documentation.



### Powerful

Fiji bundles together many popular and useful ImageJ plugins for image analysis into one installation, and automatically manages their dependencies and updating.



### Free & Open Source

Like ImageJ itself, Fiji is an [open source](#) project hosted on [GitHub](#), developed and written by the community.

# ImageJ, ImageJ2 and FIJI



## ImageJ Citation:

Schneider, C. A., Rasband, W. S., & Eliceiri, K. W. (2012). NIH Image to ImageJ: 25 years of image analysis. *Nature Methods*, 9(7), 671–675. [doi:10.1038/nmeth.2089](https://doi.org/10.1038/nmeth.2089)



## ImageJ2 Citation:

Rueden, C. T., Schindelin, J., Hiner, M. C., DeZonia, B. E., Walter, A. E., Arena, E. T., & Eliceiri, K. W. (2017). ImageJ2: ImageJ for the next generation of scientific image data. *BMC Bioinformatics*, 18(1). [doi:10.1186/s12859-017-1934-z](https://doi.org/10.1186/s12859-017-1934-z)

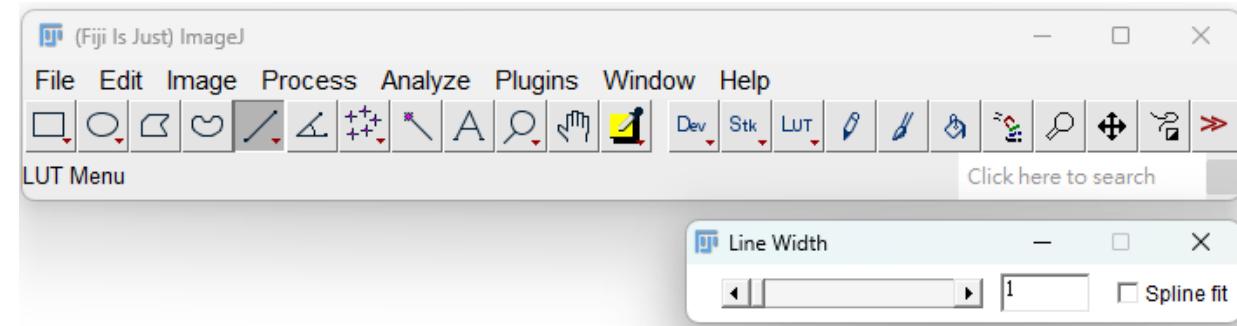
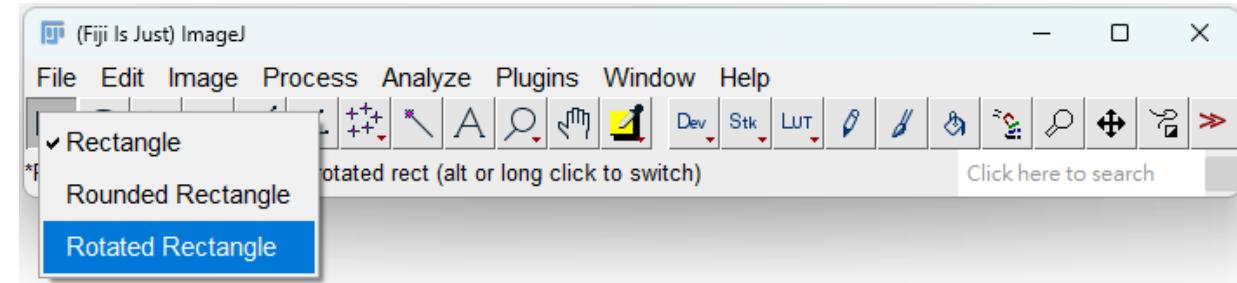


## FIJI Citation:

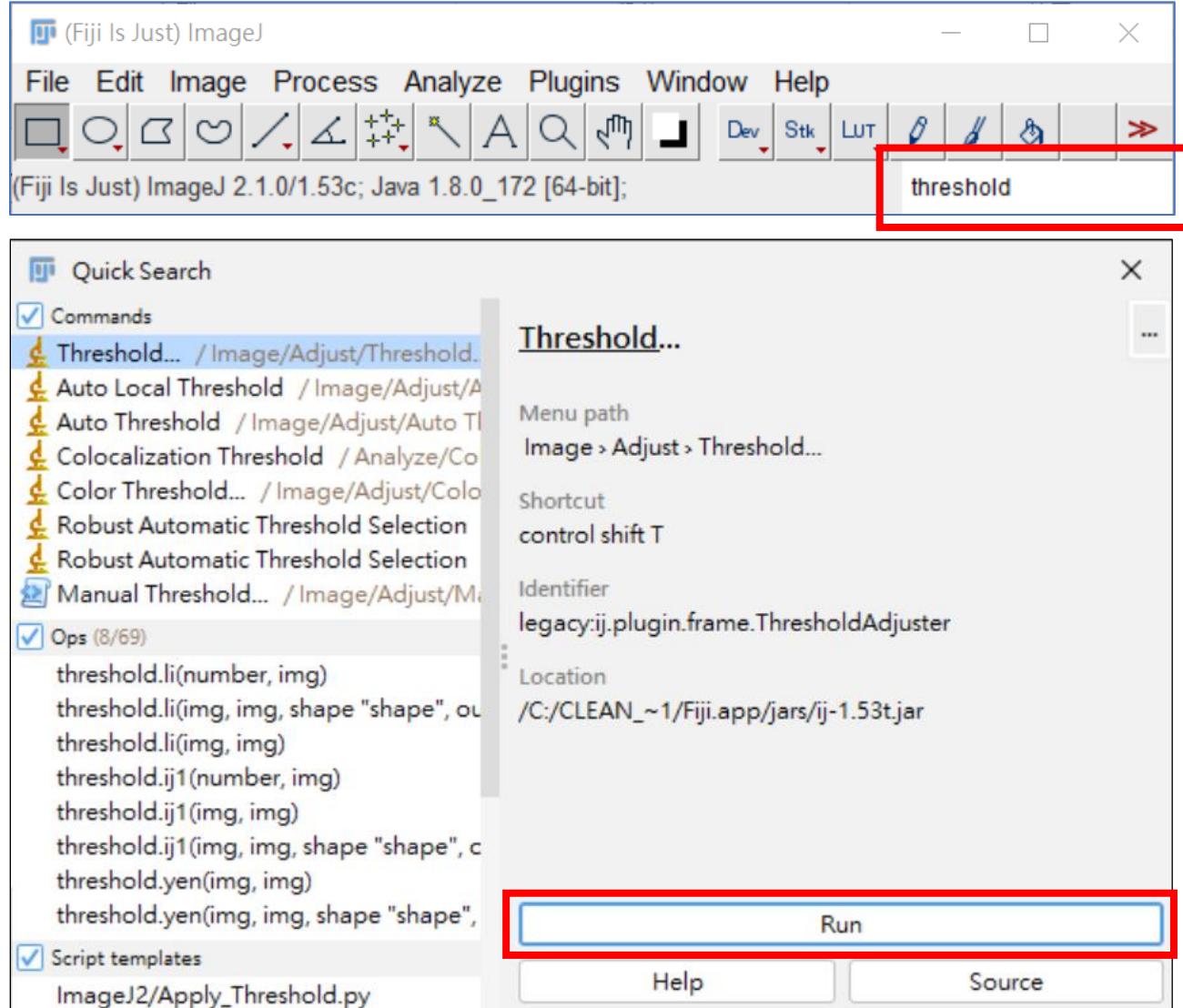
Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., ... Cardona, A. (2012). Fiji: an open-source platform for biological-image analysis. *Nature Methods*, 9(7), 676–682. [doi:10.1038/nmeth.2019](https://doi.org/10.1038/nmeth.2019)

# Fiji's user interface

- There are more tools in the toolbar than expected...
- Use the right click or double click to discover them!



# Search Bar



- It shows you where the plugin is located
- You can run it from here (press enter)

ImageJ : command finder (Hotkey “L”)

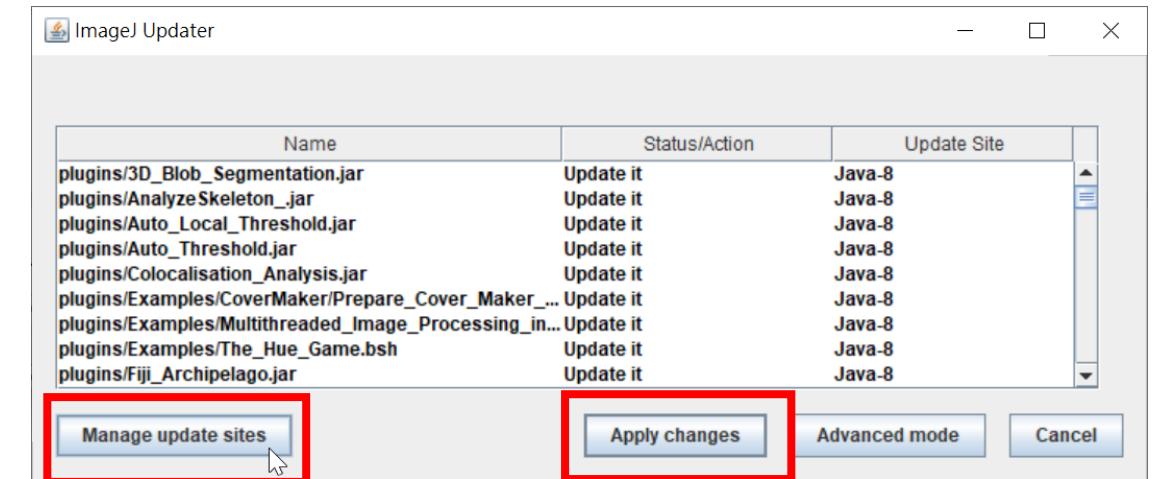
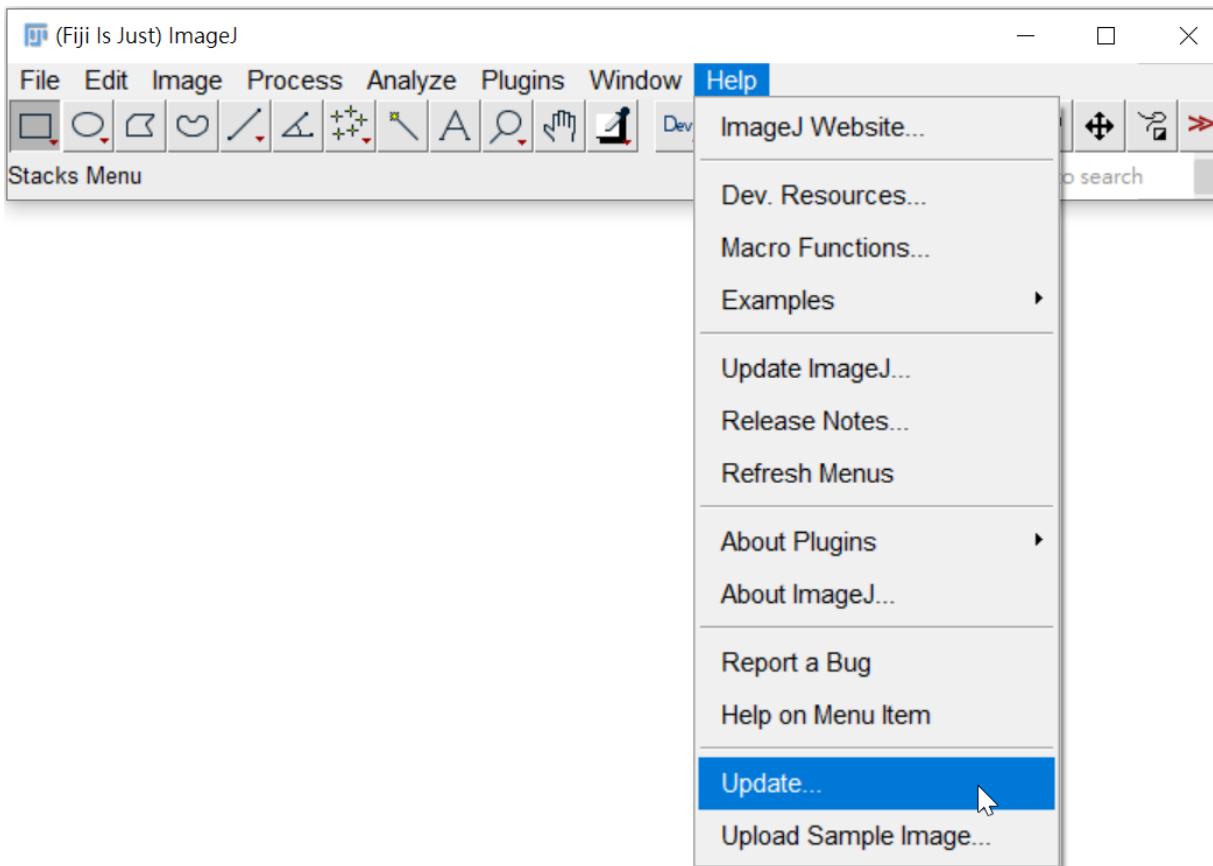
| Command                 | Menu Path                 | Class                                | File |
|-------------------------|---------------------------|--------------------------------------|------|
| 16 Colors               | Image>Lookup Tables       |                                      |      |
| 16-bit                  | Image>Type                | ij.plugin.Converter("16-bit")        |      |
| 3-2 RGB                 | Image>Lookup Tables       | ij.plugin.LutLoader("3-2 RGB")       |      |
| 32-bit                  | Image>Type                | ij.plugin.Converter("32-bit")        |      |
| 3D Project...           | Image>Stacks              | ij.plugin.Projector                  |      |
| 5 Ramps                 | Image>Lookup Tables       |                                      |      |
| 6 Shades                | Image>Lookup Tables       |                                      |      |
| 8-bit                   | Image>Type                | ij.plugin.Converter("8-bit")         |      |
| 8-bit Color             | Image>Type                | ij.plugin.Converter("8-bit Color")   |      |
| AND...                  | Process>Math              | ij.plugin.filter.ImageMath("and")    |      |
| AVI...                  | File>Import               | ij.plugin.AVI_Reader                 |      |
| AVI...                  | File>Save As              | ij.plugin.filter.AVI_Writer          |      |
| About ImageJ...         | Help                      | ij.plugin.AboutBox                   |      |
| About Startup Macros... | Plugins>Macros            |                                      |      |
| About These Macros      | Plugins>Examples>_Macros  | ij.plugin.Macro_Runner("Example...") |      |
| About These Scripts     | Plugins>Examples>_Scripts | ij.plugin.Macro_Runner("Example...") |      |
| About These Tools       | Plugins>Tools             | ij.plugin.Macro_Runner("Tools")      |      |
| About This Submenu...   | Help>About Plugins        | ij.plugin.SimpleCommands("abou...")  |      |

Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

# FIJI updater



| ...                                 | Name                 | URL   | Host | Directory on Host |
|-------------------------------------|----------------------|---|------|-------------------|
| <input checked="" type="checkbox"/> | ImageJ               | https://update.imagej.net/                        |      |                   |
| <input checked="" type="checkbox"/> | Fiji                 | https://update.fiji.sc/                           |      |                   |
| <input type="checkbox"/>            | Fiji-Legacy          | https://sites.imagej.net/Fiji-Legacy/             |      |                   |
| <input checked="" type="checkbox"/> | Java-8               | https://sites.imagej.net/Java-8/                  |      |                   |
| <input type="checkbox"/>            | 2015-Conference      | https://sites.imagej.net/2015-Conference/         |      |                   |
| <input type="checkbox"/>            | 3D ImageJ Suite      | https://sites.imagej.net/Tboudier/                |      |                   |
| <input type="checkbox"/>            | 3Dscript             | https://romulus.oice.uni-erlangen.de/updatesit... |      |                   |
| <input type="checkbox"/>            | ActogramJ            | https://romulus.oice.uni-erlangen.de/imagej/u...  |      |                   |
| <input type="checkbox"/>            | AdipoQ               | https://sites.imagej.net/AdipoQ/                  |      |                   |
| <input type="checkbox"/>            | AIC Janelia - Course | https://sites.imagej.net/AICJanelia-course/       |      |                   |
| <input type="checkbox"/>            | AMPT                 | https://sites.imagej.net/AMPT/                    |      |                   |
| <input type="checkbox"/>            | Angiogenesis         | https://sites.imagej.net/Angiogenesis/            |      |                   |

Buttons at the bottom: Add update site, Remove, Update URLs, Close.

- Keep a backup of working version before you update.



ICOB Imaging Core

# Loading Data

## Drag to FIJI and release

- Generic image format
- Folder (Import Image Sequence)
- ROI (.roi, .zip)
- Macro files (.ijm)

## Bio-Format Importer

- Vendor specific image format(.czi, .lif, .oir...etc)
- Huygens deconvoluted image ICS2 format (.ics)
- Big Data (Use Virtual Stack)
- Generic image format
- More details in the measurement “label”  
(Channel, Z, T, ROI)

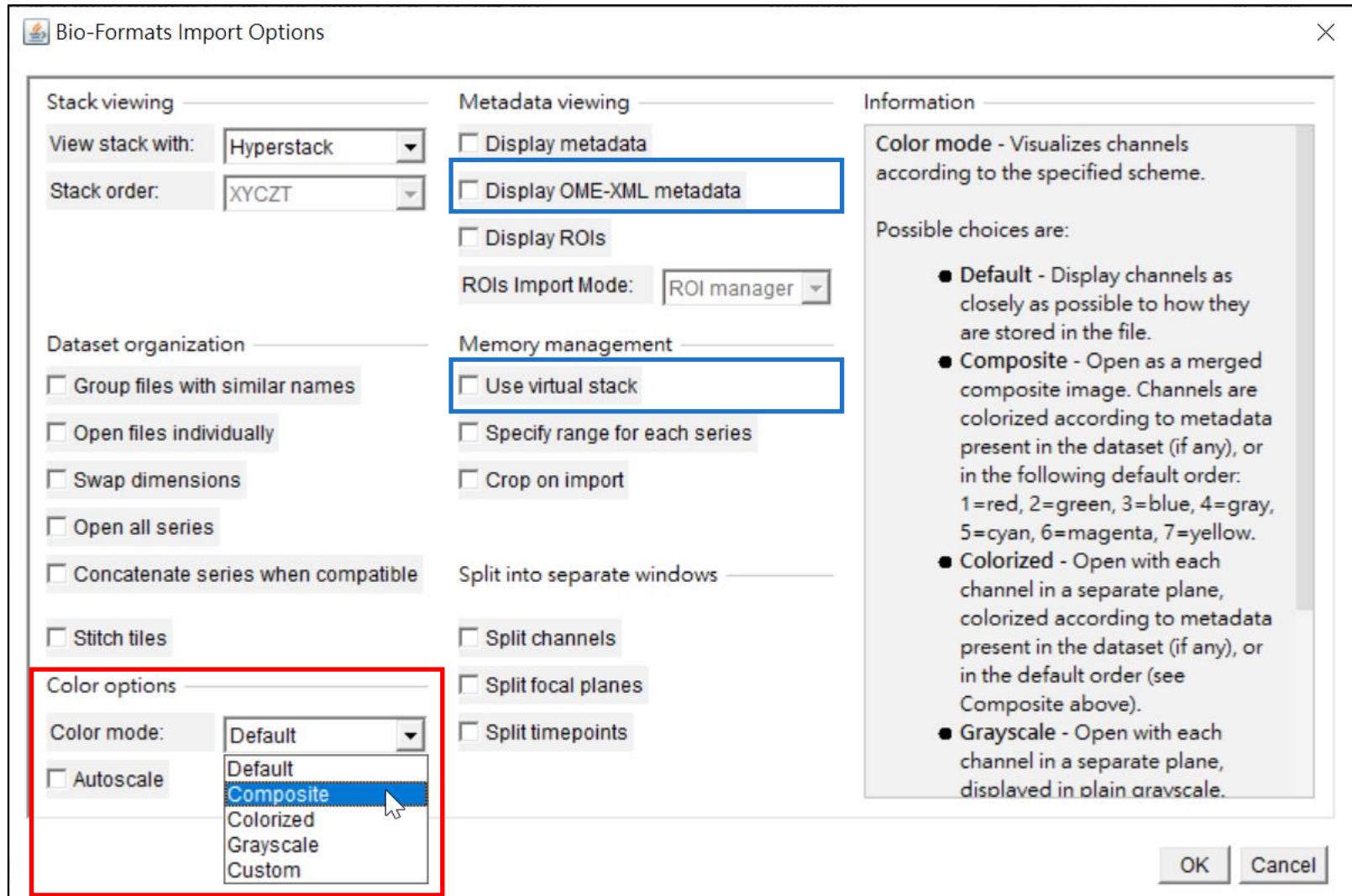
## Generic image format

- Tiff: 8 bits, 16 bits, 32 bits per channel  
24 bits RGB  
Can keep part of metadata (e.g. Pixel Size)
- Png: RGB mode only, metadata loss
- Jpg: lossy compression
- Zip: compressed Tiff (lossless)



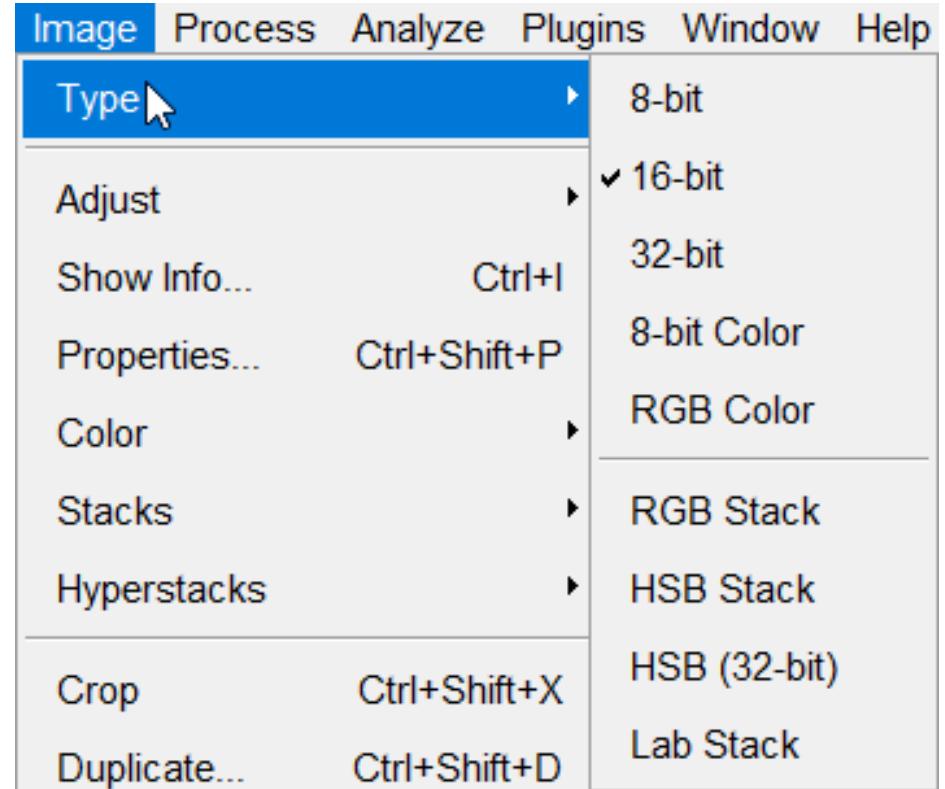
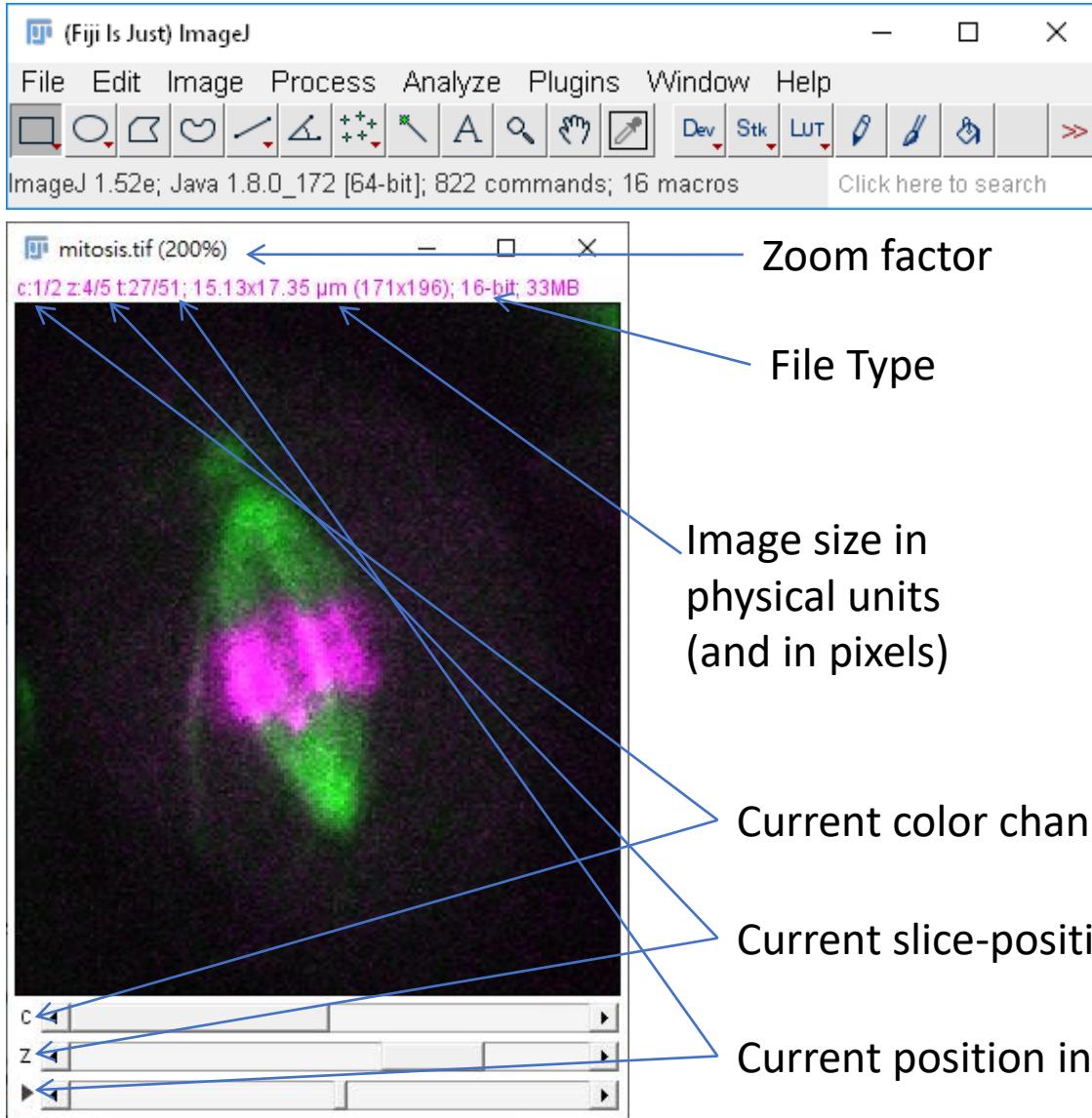
# Bioformats importer

- **Composite** is useful for multi-color imaging
- **Autoscale** -> similar to auto contrast
- **Display OME-XML metadata**: easier to read image acquisition settings.
- **Virtual Stack** -> useful for large image (Lazy loading)



# Fijis user interface

Only 1 active window, 1 activate channel



Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

# Useful tools when you open an image

Image -> Channels Tool

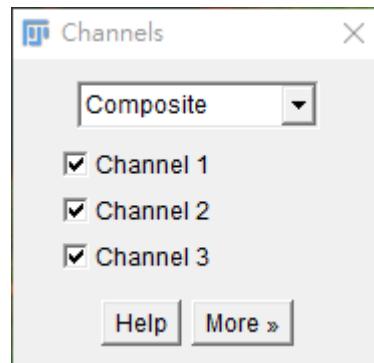
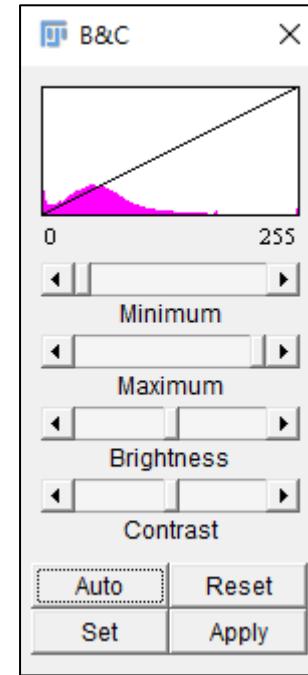
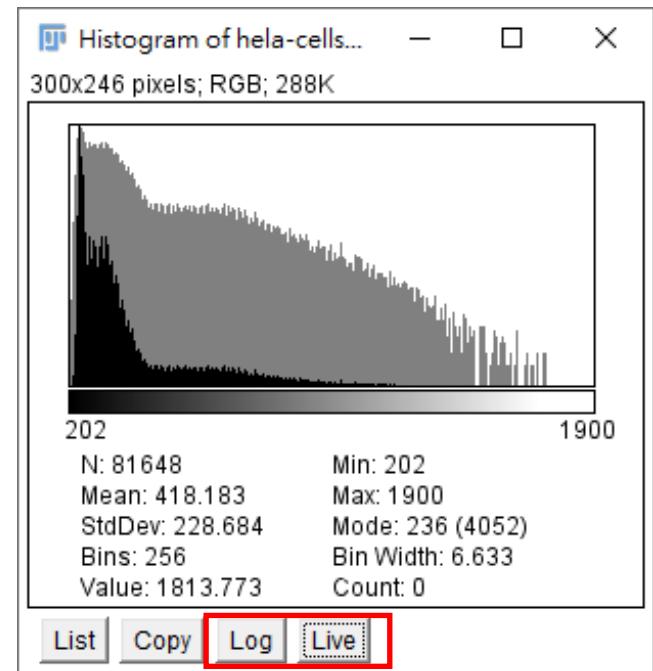


Image -> Adjust -> Brightness & Contrast



Analyze -> Histogram



# Explore your image

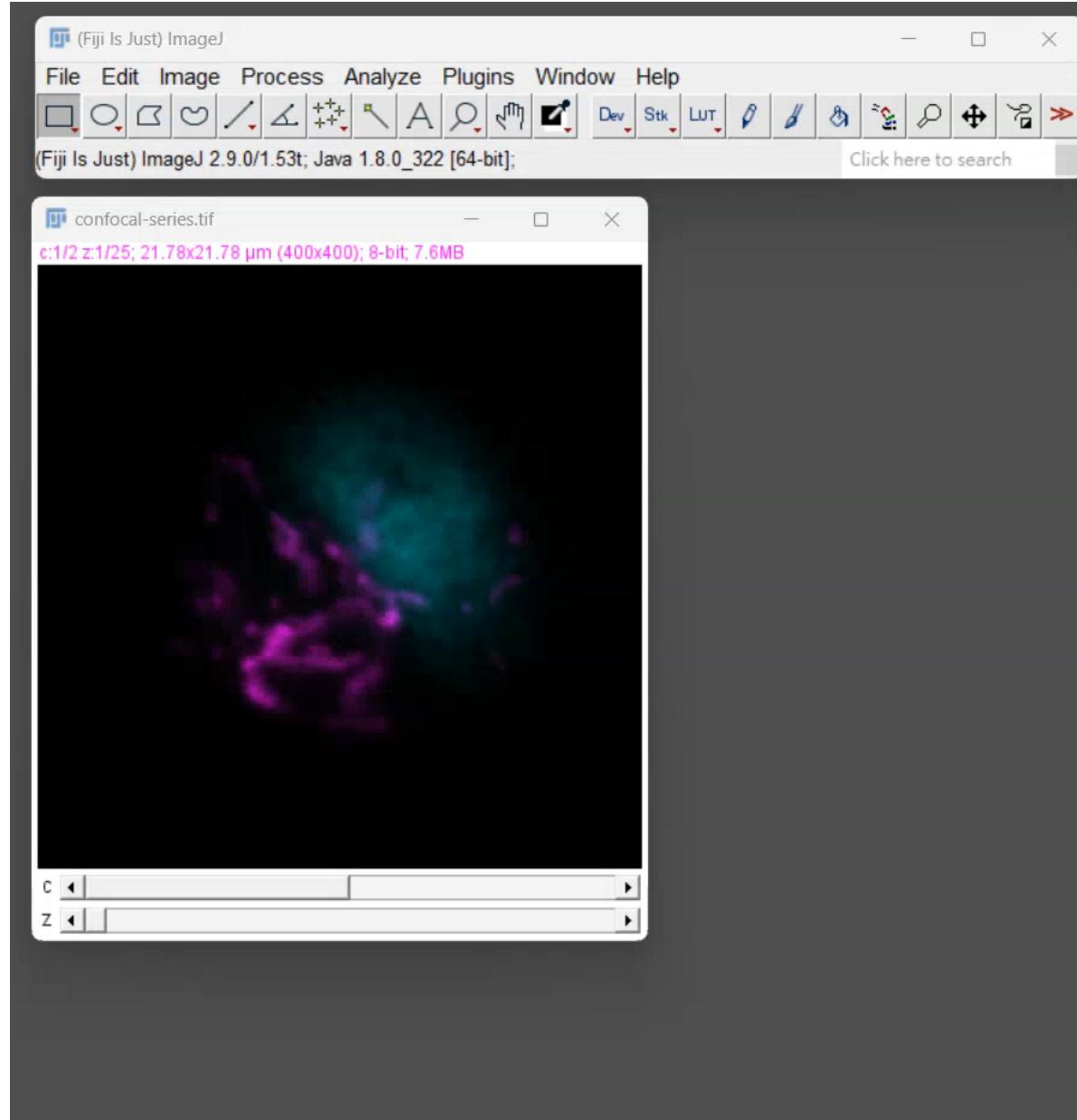
## Channels Tool



## Brightness & Contrast



## Histogram

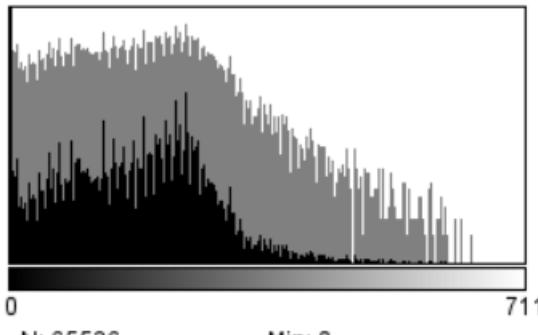


Video

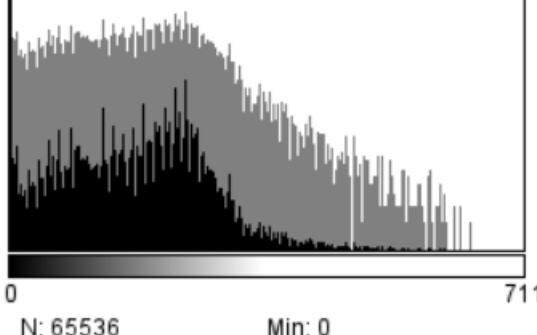


ICOB Imaging Core

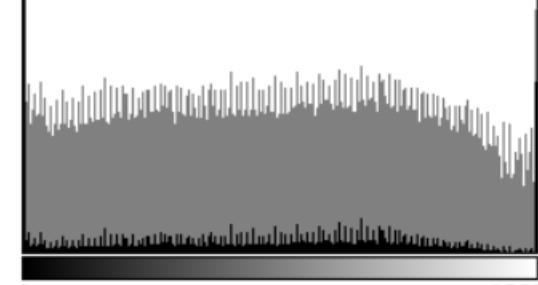
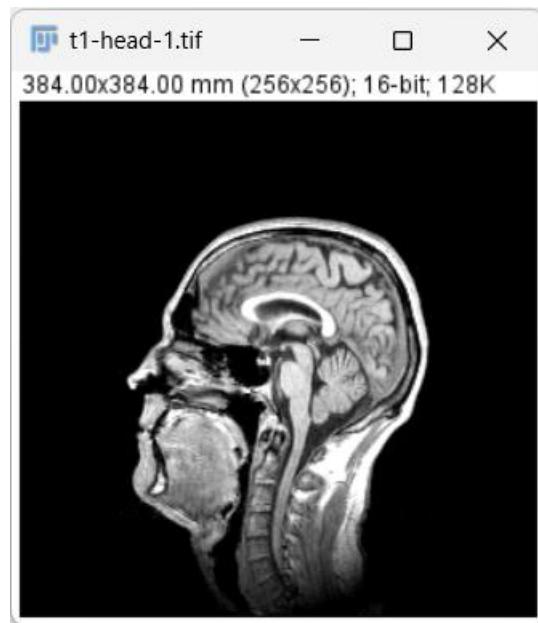
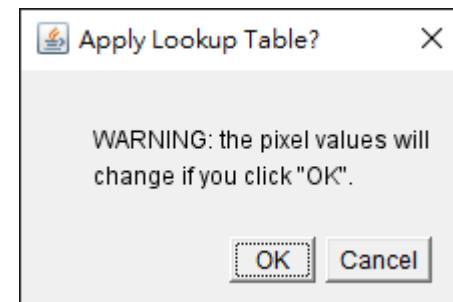
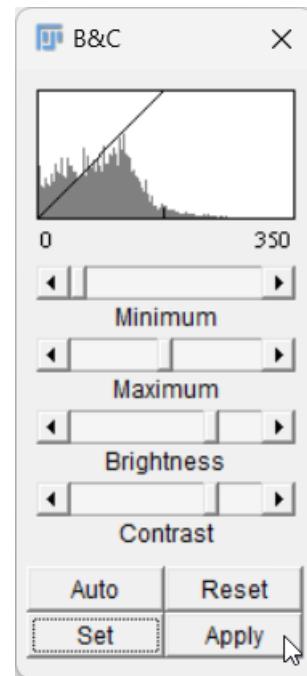
# Don't “Apply” when you adjust the Brightness & Contrast



N: 65536      Min: 0  
Mean: 56.938      Max: 711  
StdDev: 102.475      Mode: 0 (44916)

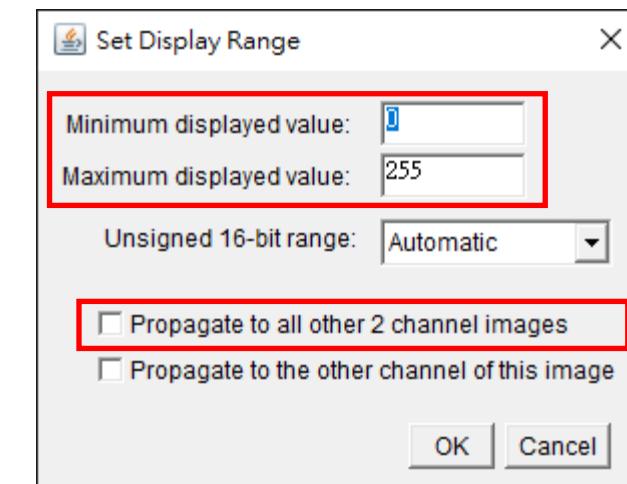
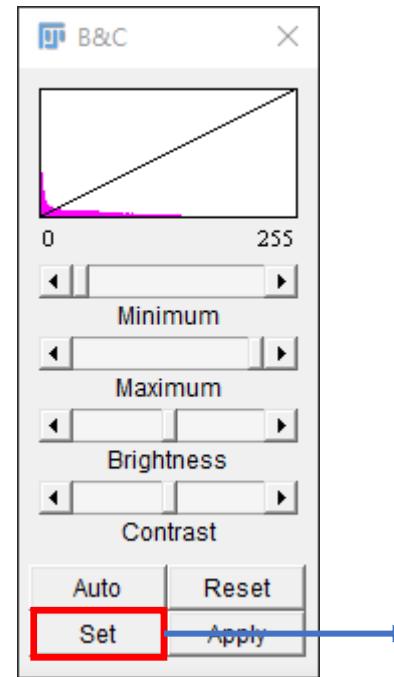
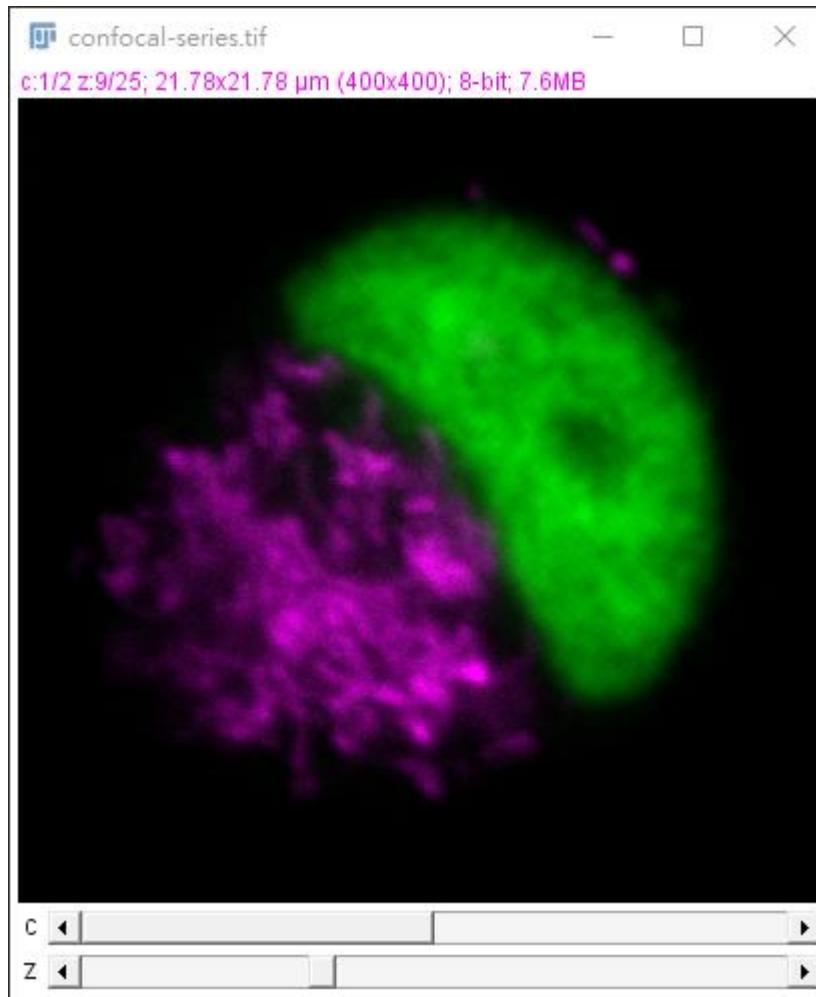


N: 65536      Min: 0  
Mean: 56.938      Max: 711  
StdDev: 102.475      Mode: 0 (44916)



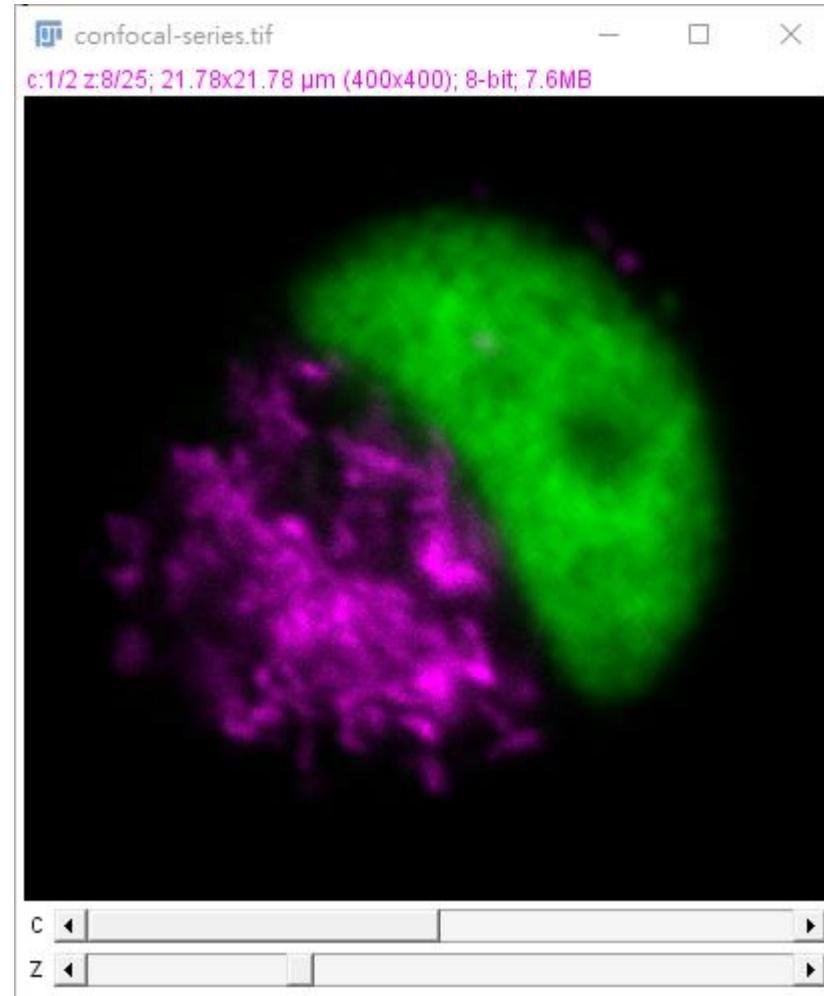
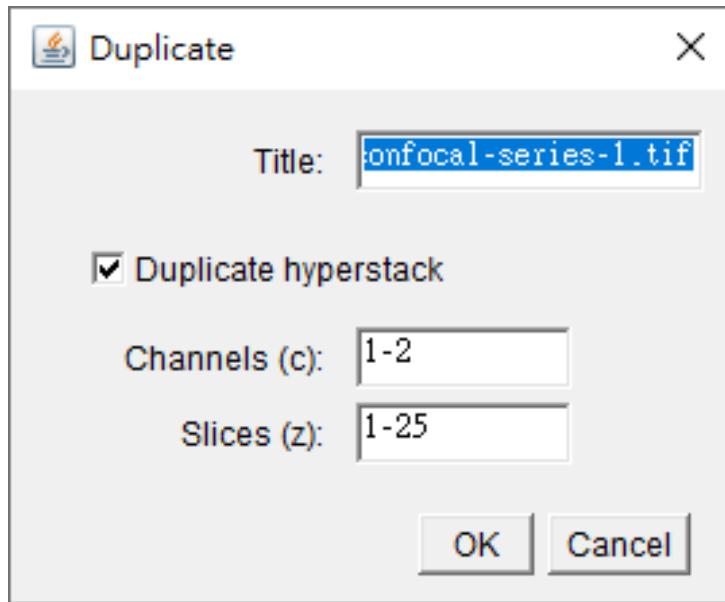
N: 65536      Min: 0  
Mean: 10415.383      Max: 65535  
StdDev: 18293.022      Mode: 0 (44837)

# Use the same display range for the image you want to compare



# Duplicate

- Image -> Duplicate ..

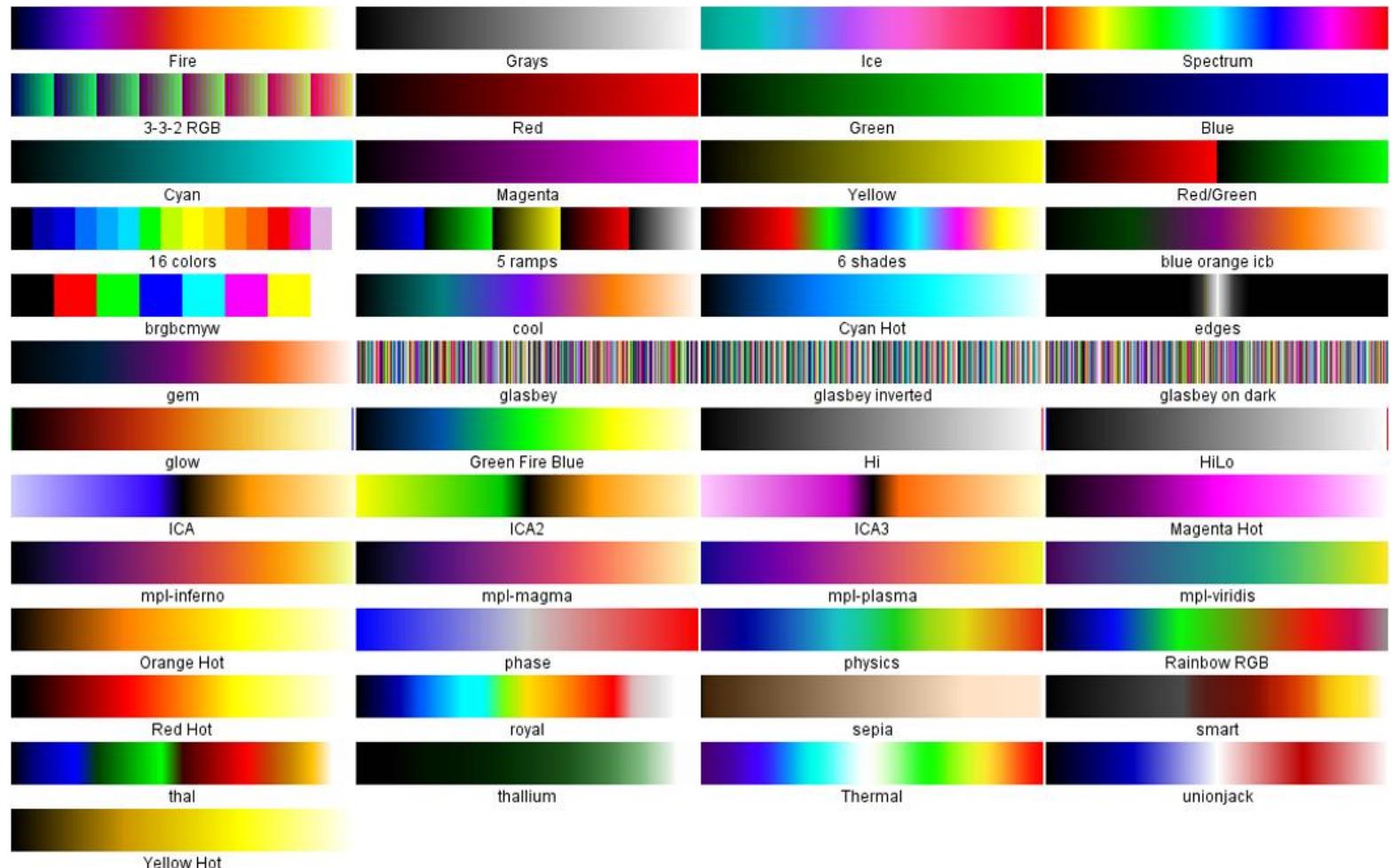


# LUT

- Image -> Lookup tables



- Image -> Display LUT



# Useful LUT

“Classic”



Red



Green



Blue



Grays

Color Blindness friendly



Magenta



Yellow



Cyan

Range indicator



HiLo



glow

Good for weak signal visualization  
(nonlinear, calibration bar is necessary)



Magenta Hot



Yellow Hot



Cyan Hot



Fire

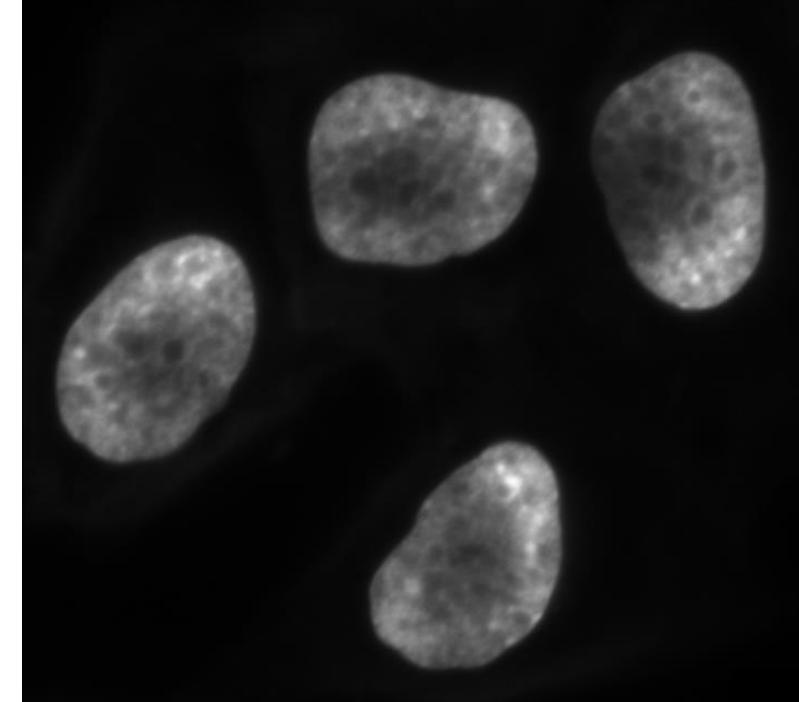
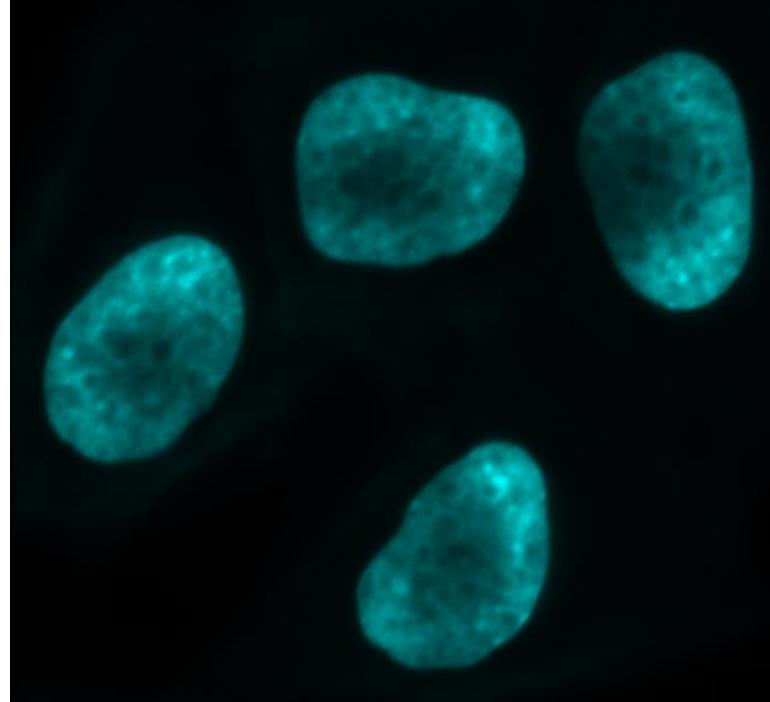
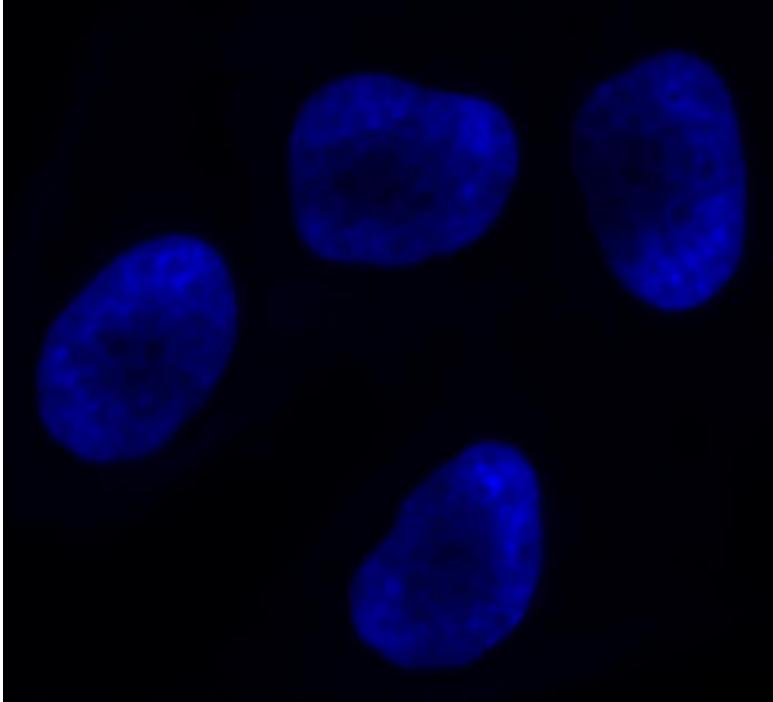
For labeling/ count mask



glasbey on dark



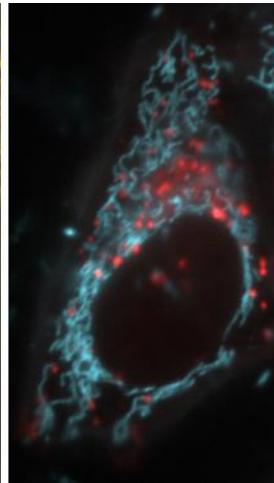
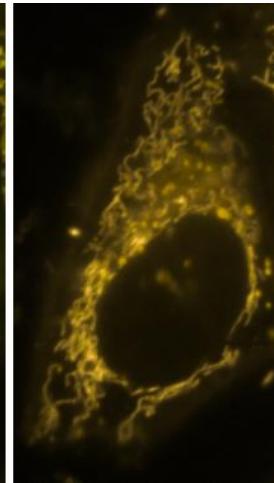
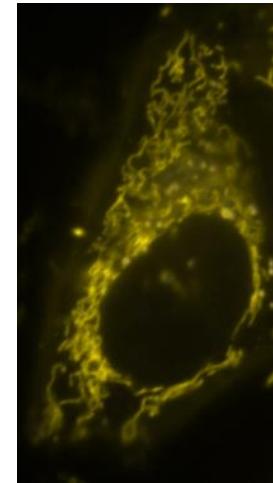
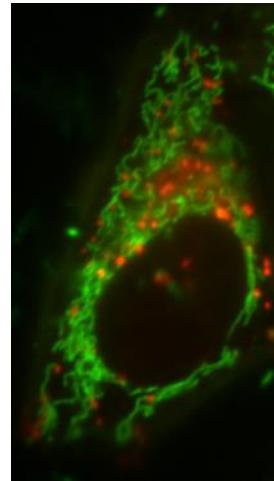
# Our eye has different sensitivity to different colors



# Be friendly to colorblindness

**Don't use red and green in  
the same image!**

Red/Green only:  
→ Replace Red with Magenta

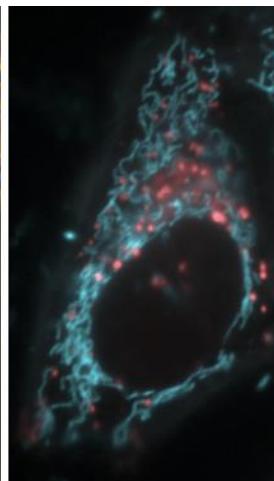
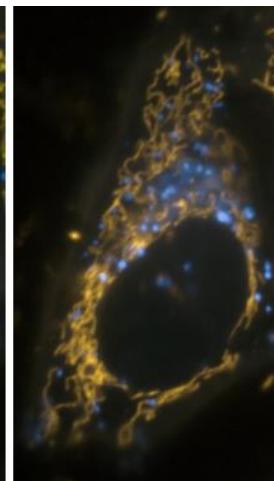
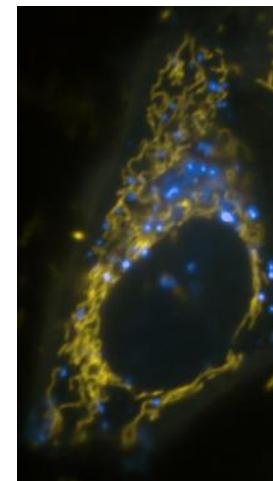
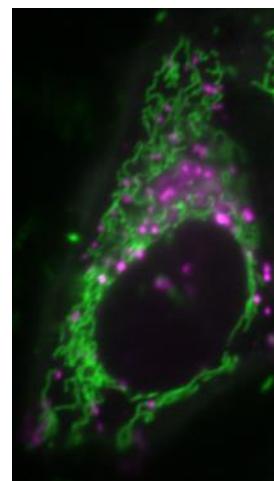


Color blindness simulation

Red

Green

Blue



# Be friendly to colorblindness (3 Color)

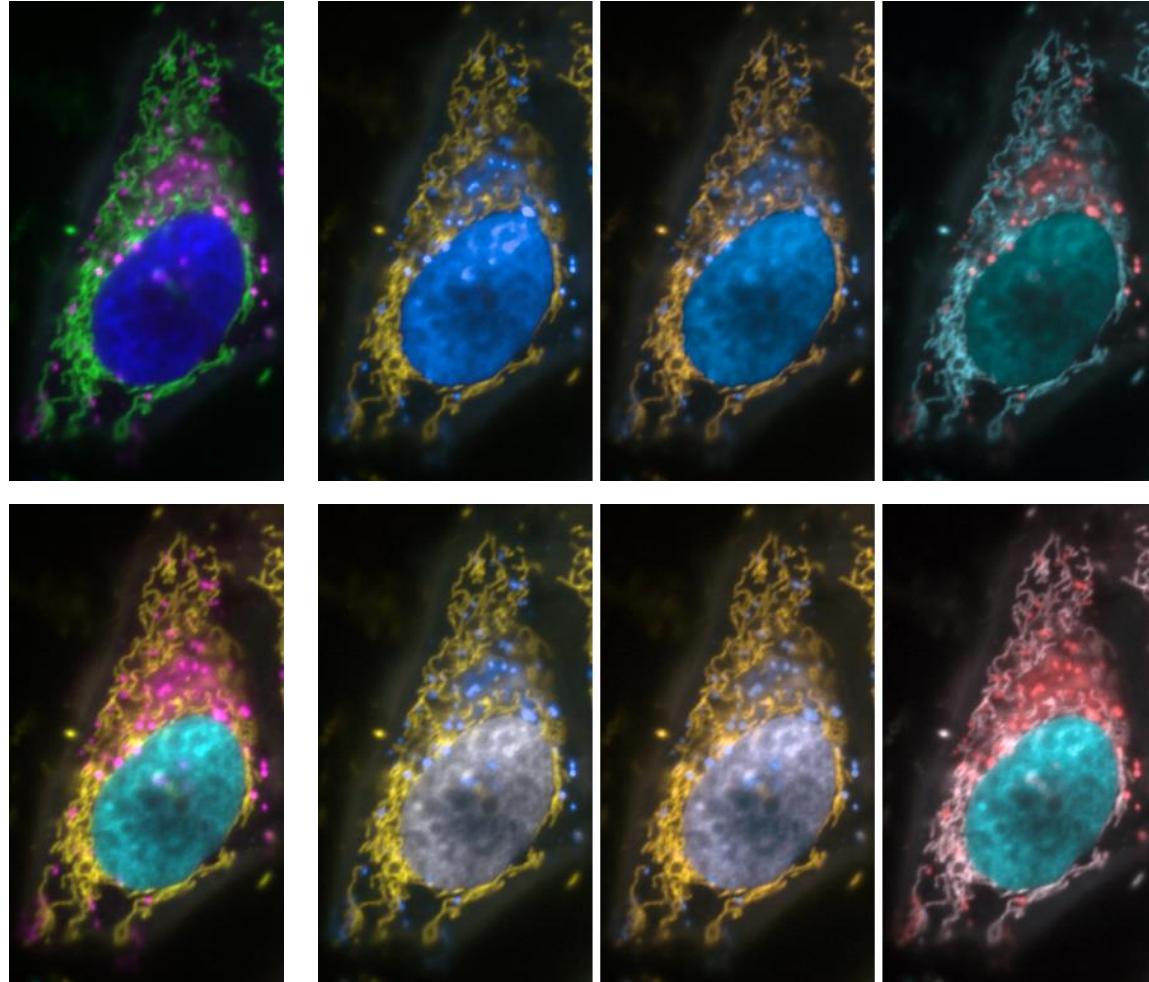
- Magenta/Yellow/Cyan
- Show grayscale pictures of each channel. (when distribution of each signal is interesting)
- Show combinations of two channels in magenta and green. (when spatial correlation between channels is interesting)

Color blindness simulation

Red

Green

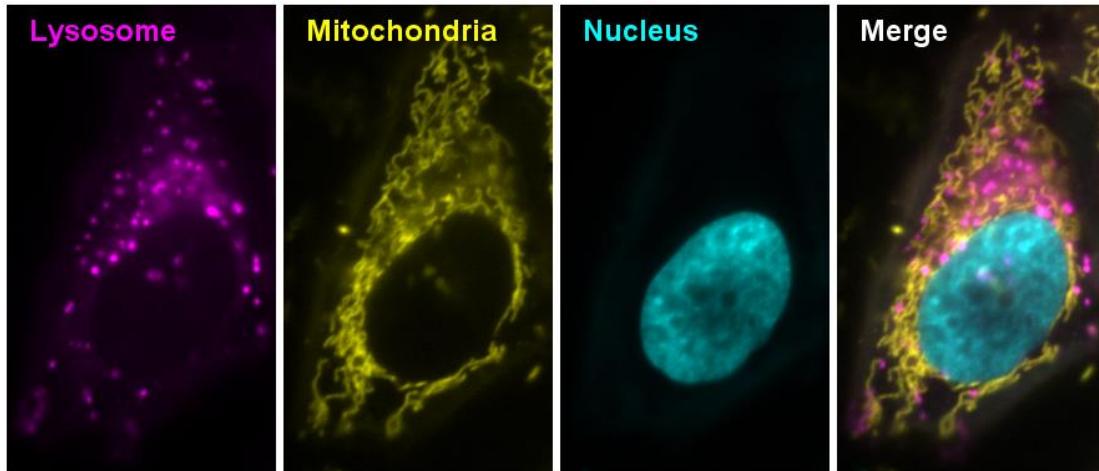
Blue



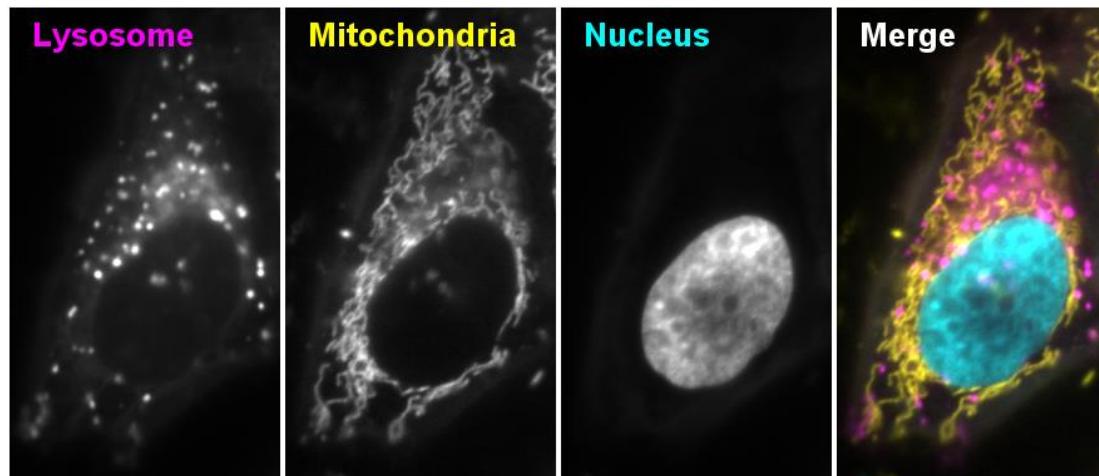
# Colorblindness friendly presentation

- For individual channel presentation, the grey scale LUT is recommended!

Good

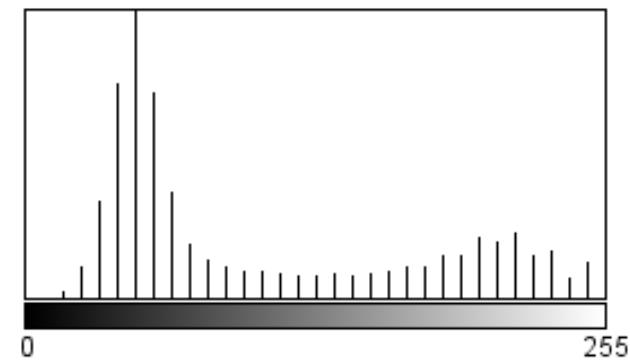
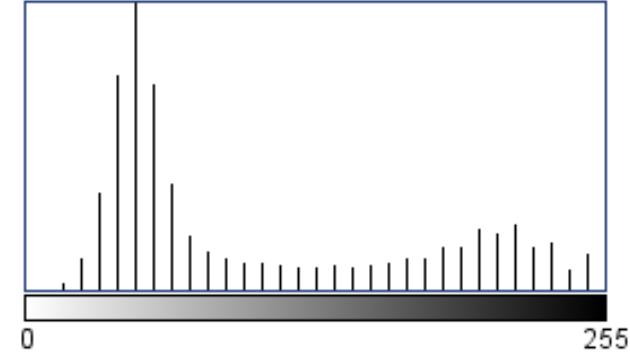
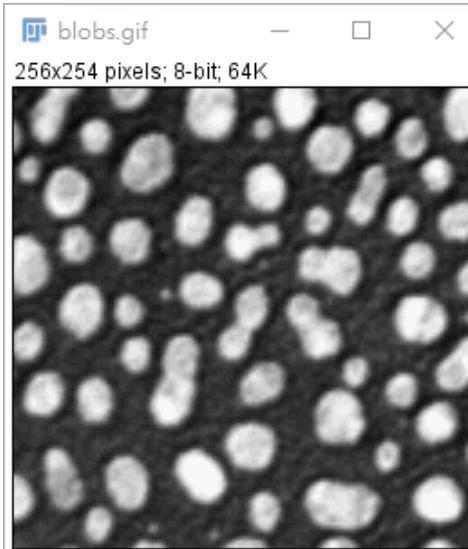
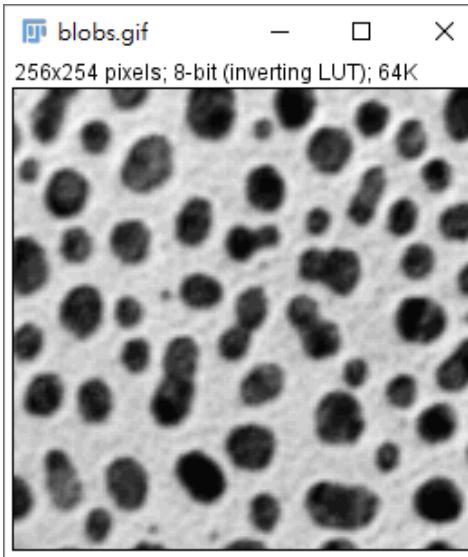
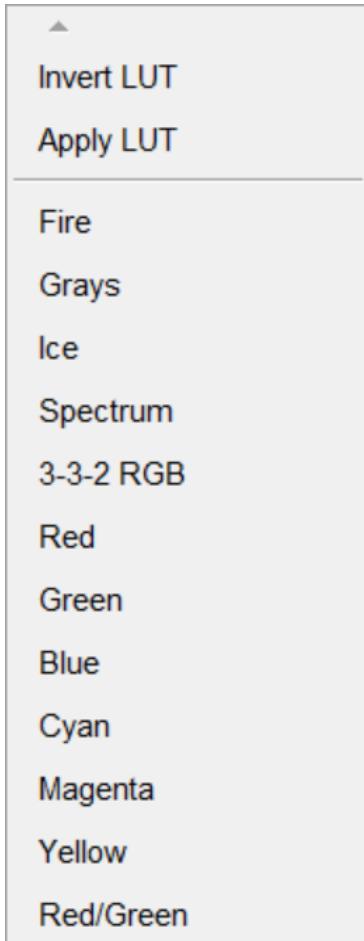


Best



# Invert LUT

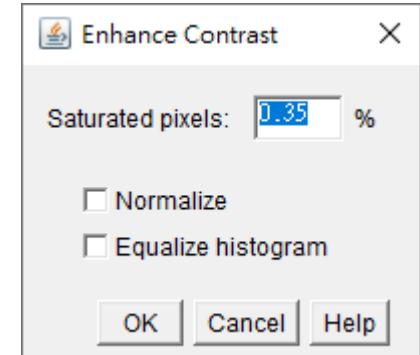
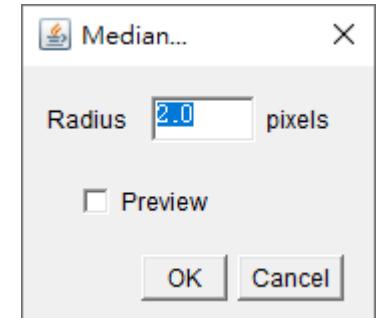
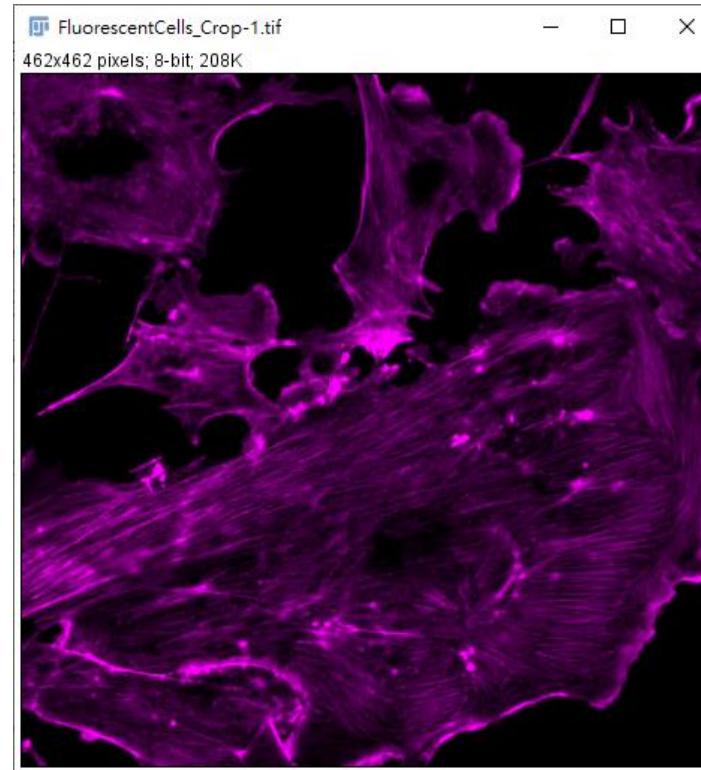
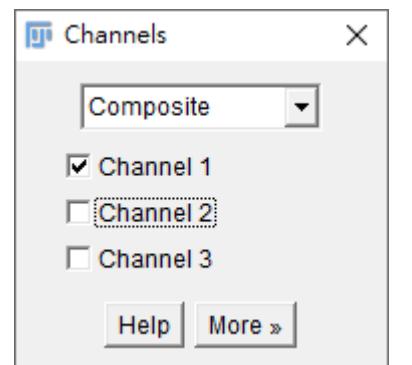
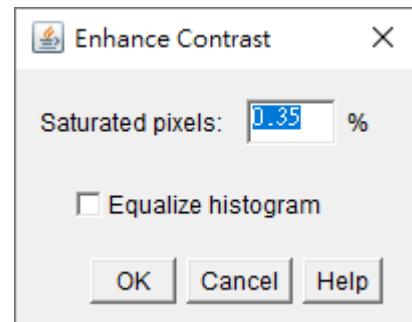
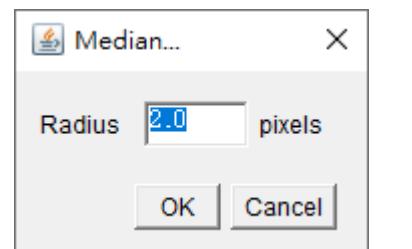
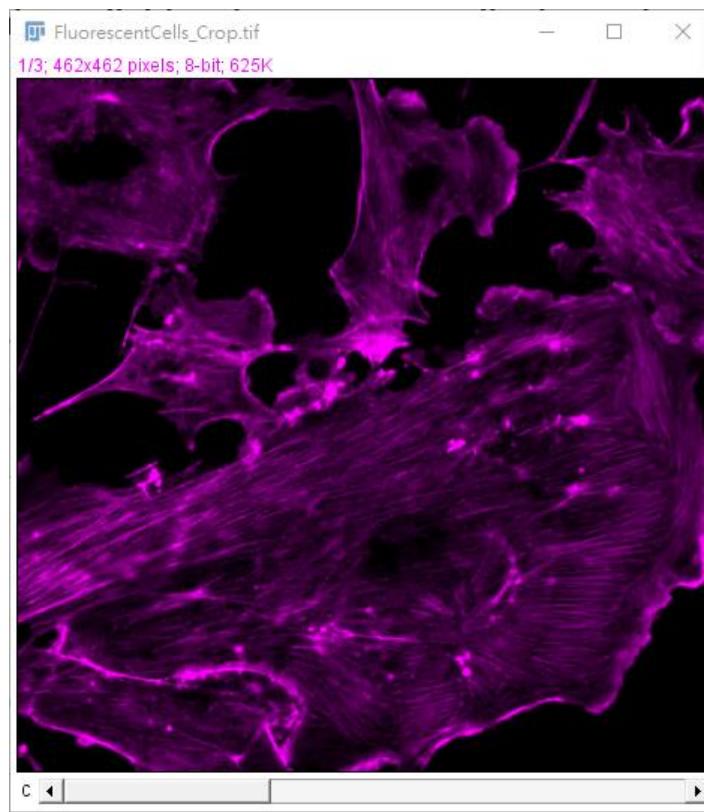
- Image -> Lookup tables



Images that look different may contain the same pixel values!

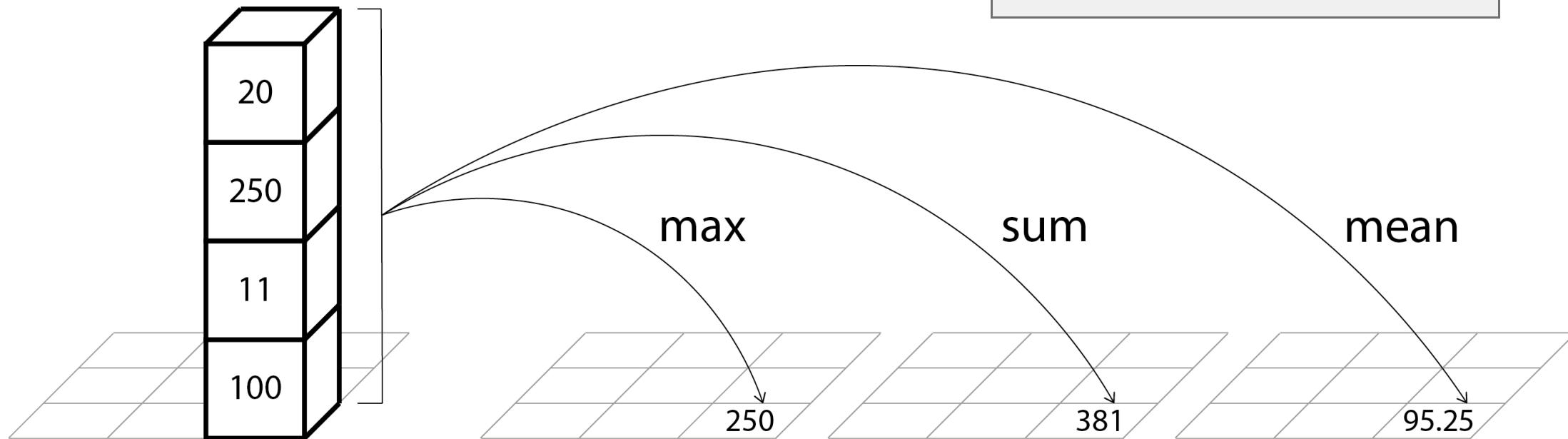
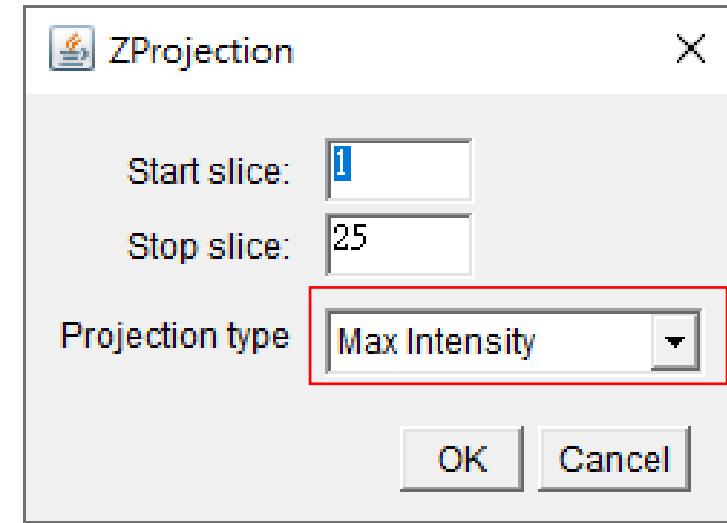


# Some functions are only available when you use a single channel window

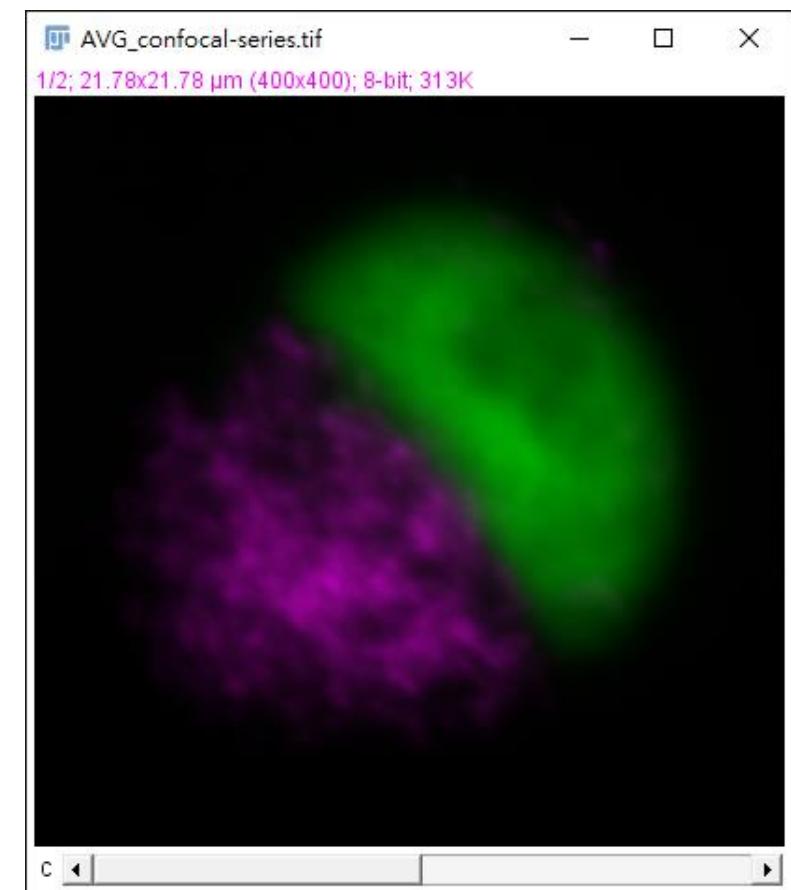
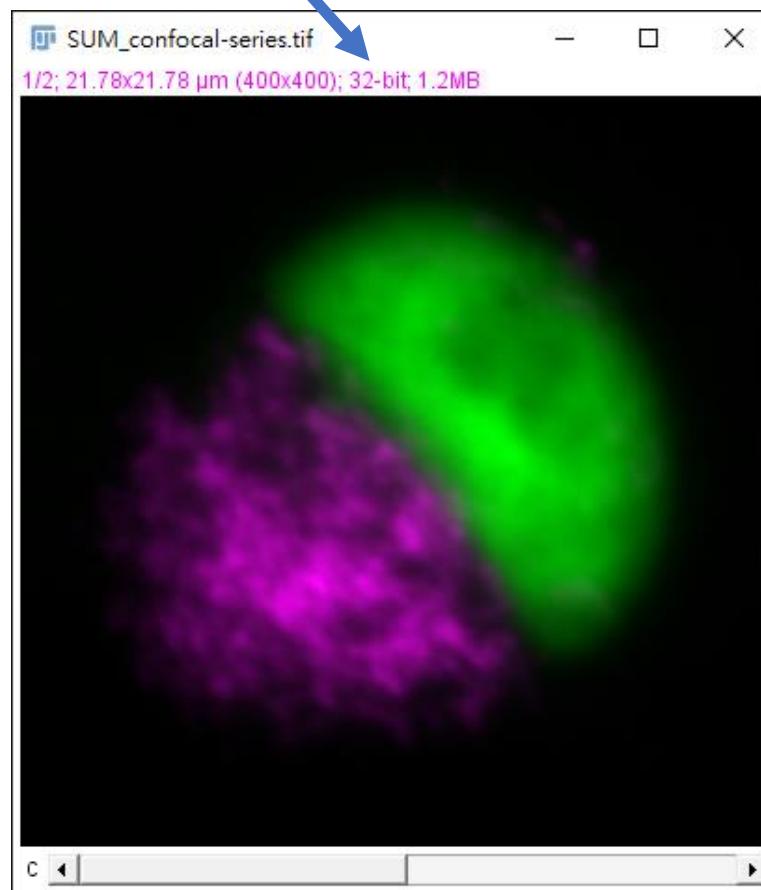
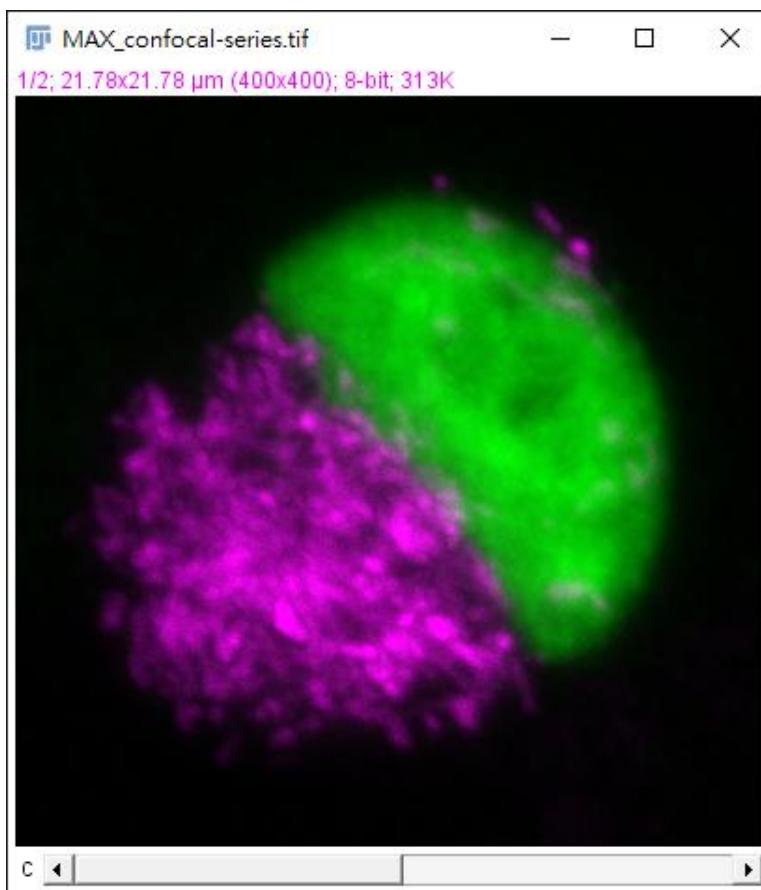


# Z Projection

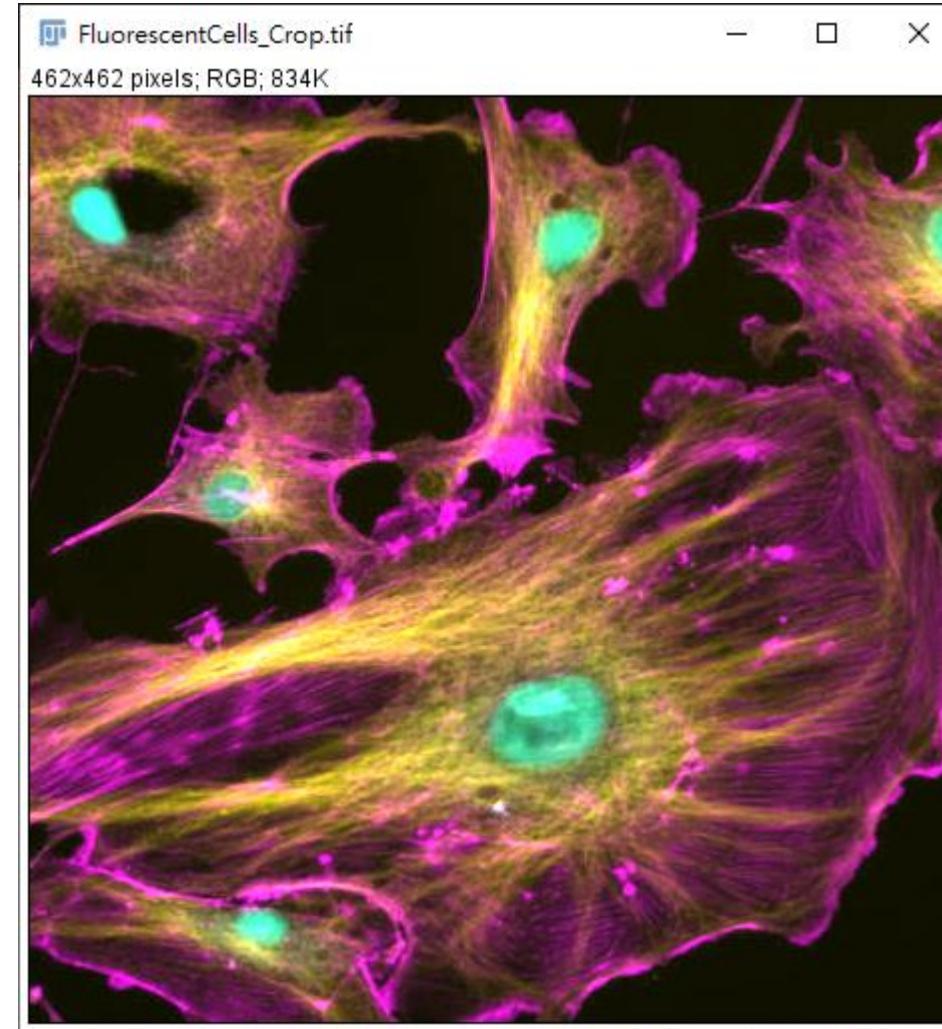
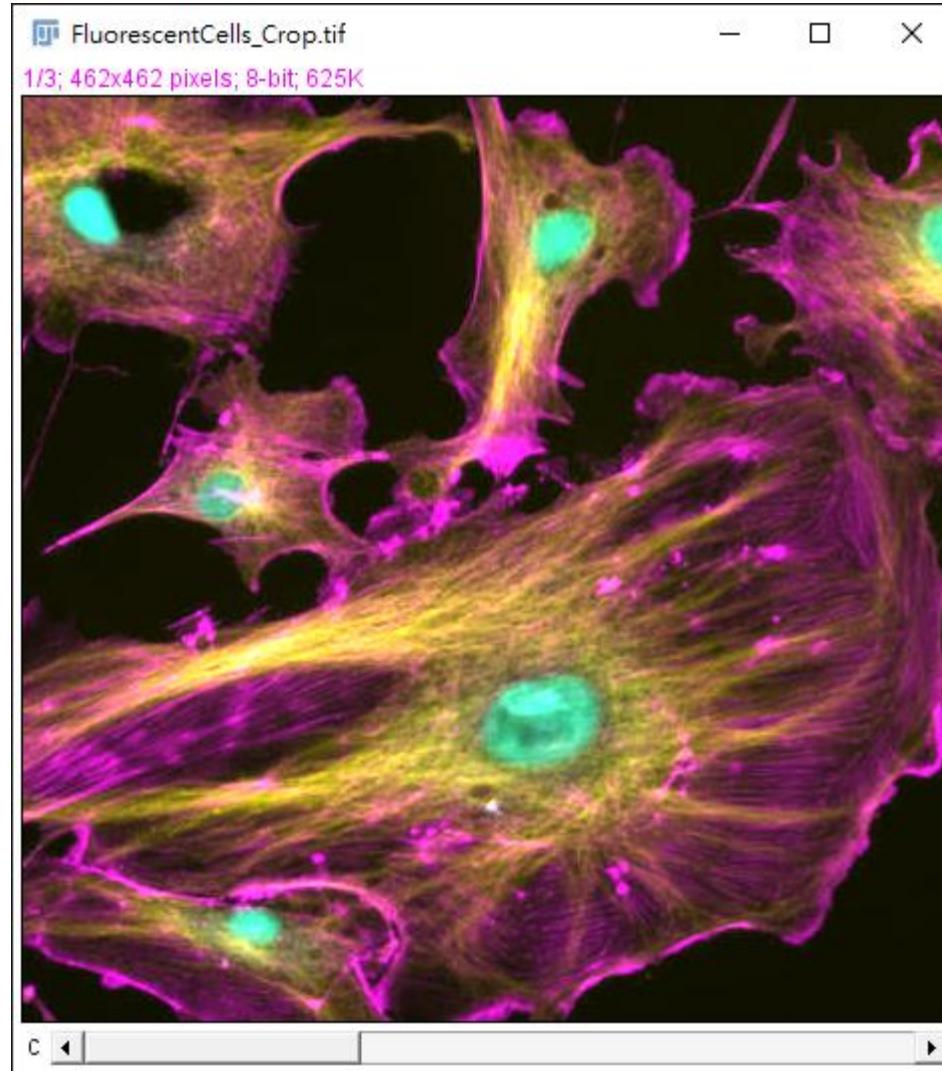
- Image -> Stacks -> Z Project...
- Max intension projection-> better for visualization
- Sum slices projection -> better for signal quantification  
**(for comparison, use the same number of slices)**
- Since Z projects lose 3D information, two signal colocalization may be false positive (e.g. two signals come from two different focal planes).



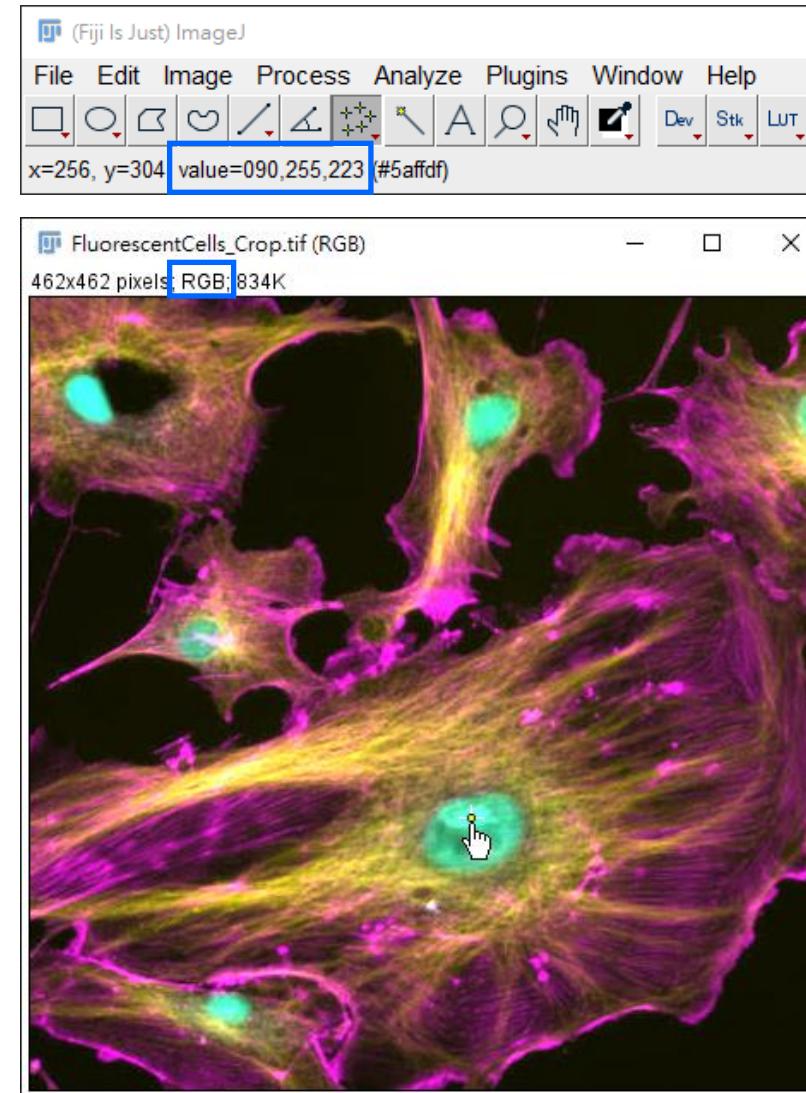
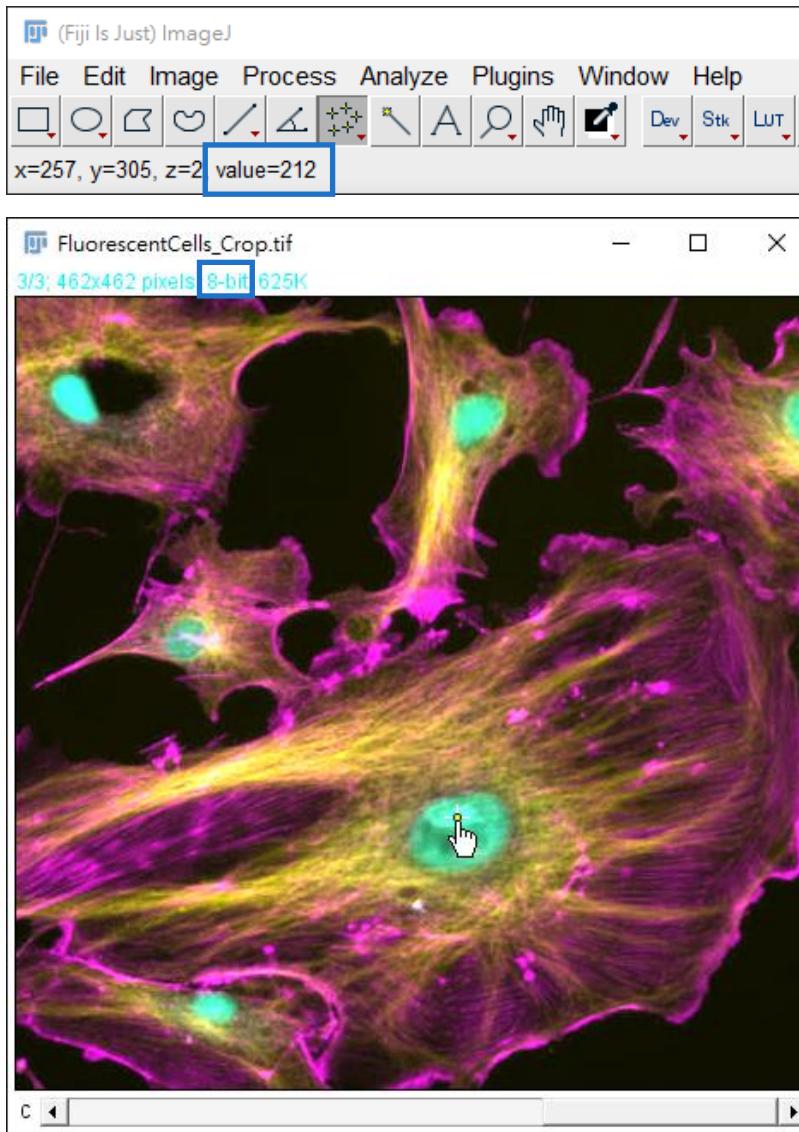
# Quiz: Why Sum projection use 32-bit format?



# Quiz: Which one is suitable for signal intensity quantification?

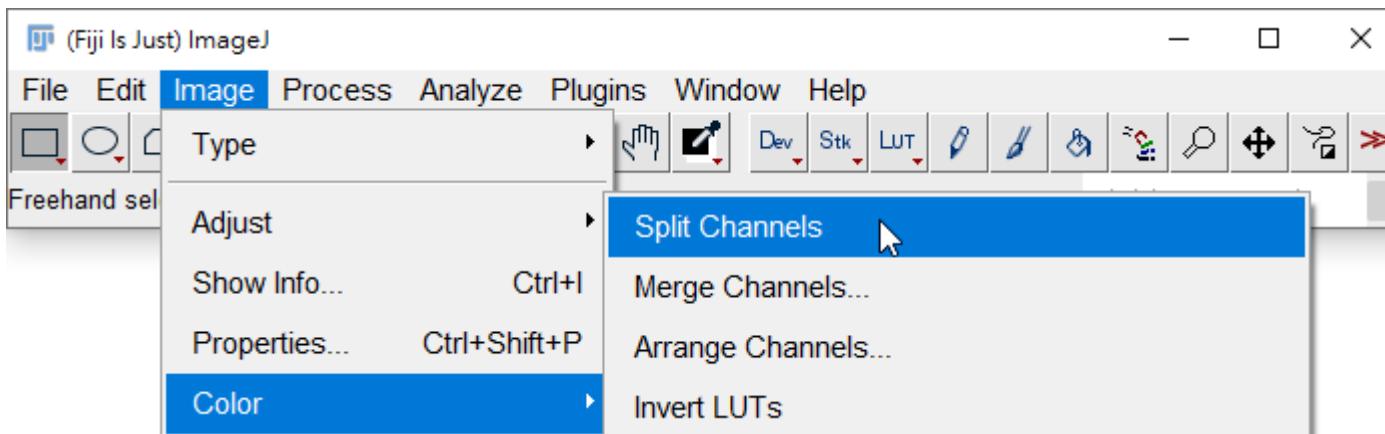
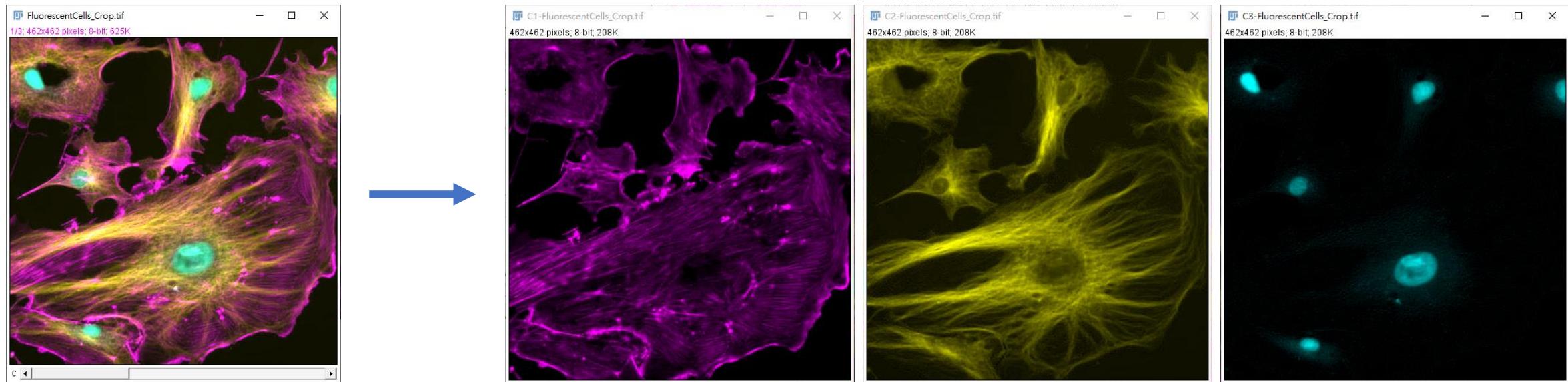


# In general, don't use RGB format for quantification



Images that look the same may give you different value!

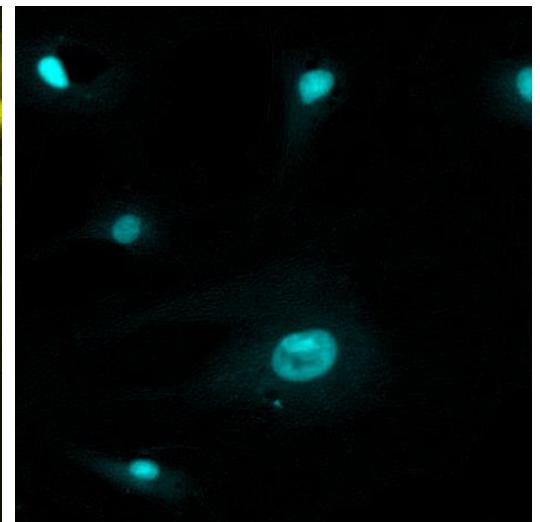
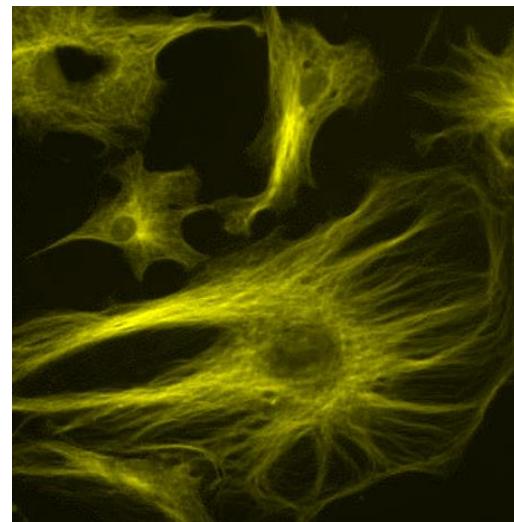
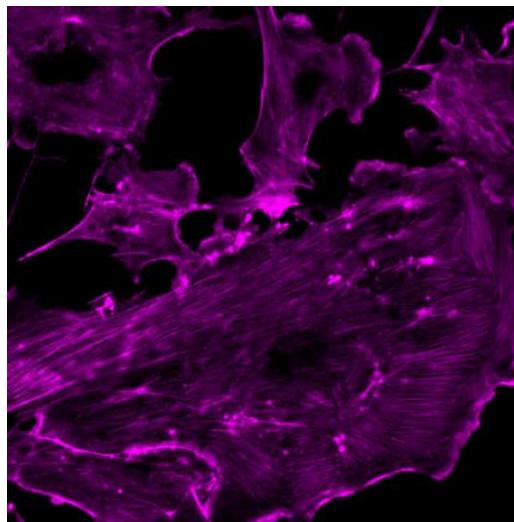
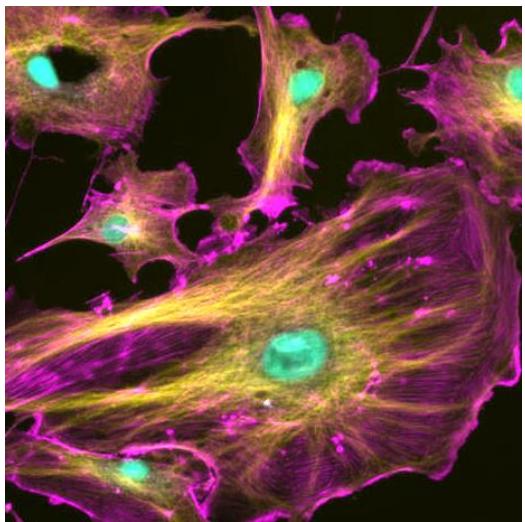
# Split channel



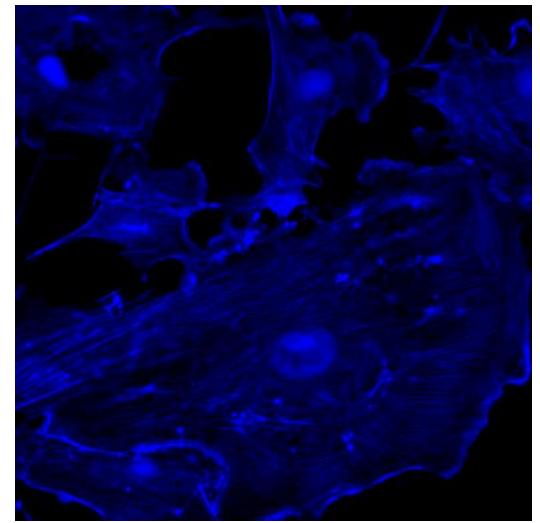
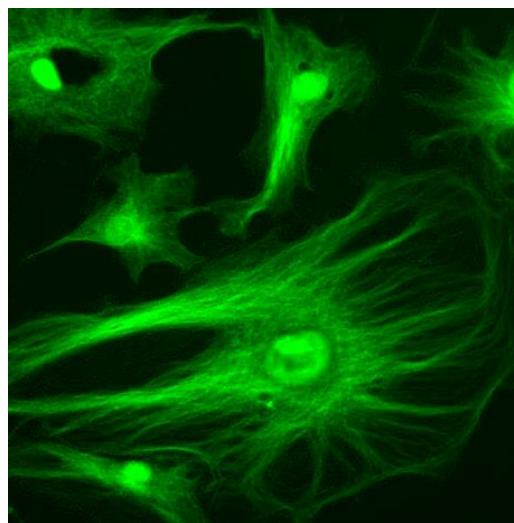
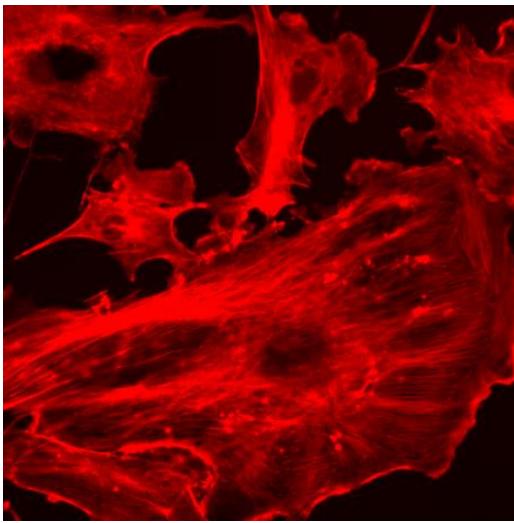
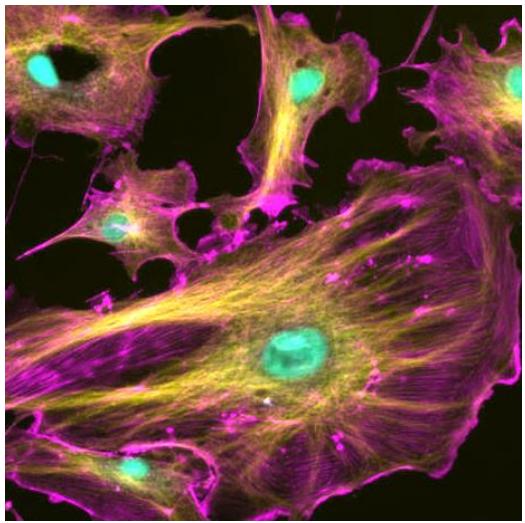
# Original data vs RGB format

Split Channels

3 Channel  
Composite



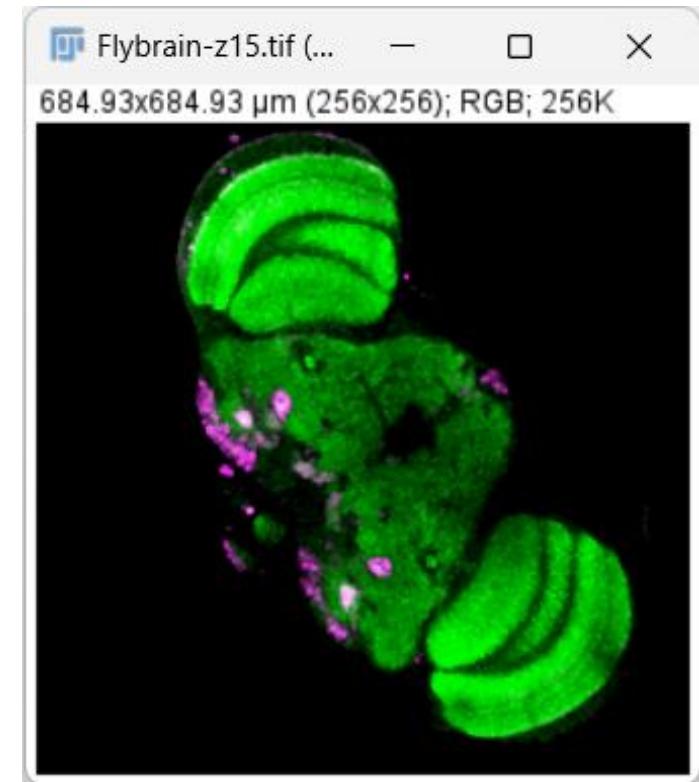
RGB



# Convert your image data to RGB color for presentation



- Image -> Overlay -> Flatten
- Image -> Type -> RGB color
- SaveAs -> Png... (metadata loss)



# Scale Bar

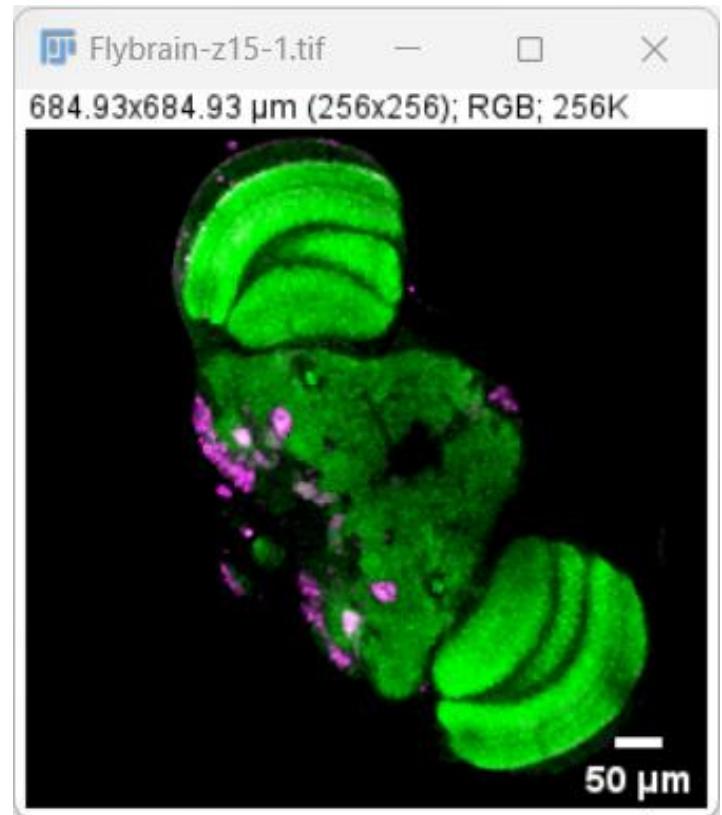
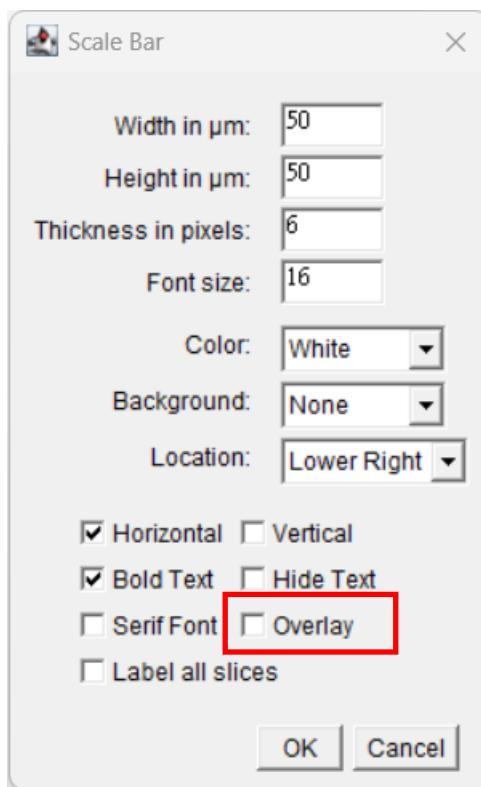
- Analyze -> Tools -> Scale Bar
- Confirm the pixel size first (“Ctrl + Shift + P”)
- Recommended to add a scale bar on the image files for presentation (RGB format)

To allow other software can read the scale bar:

- Uncheck “Overlay”.
- If “Overlay” is checked -> Please “Flatten” it!

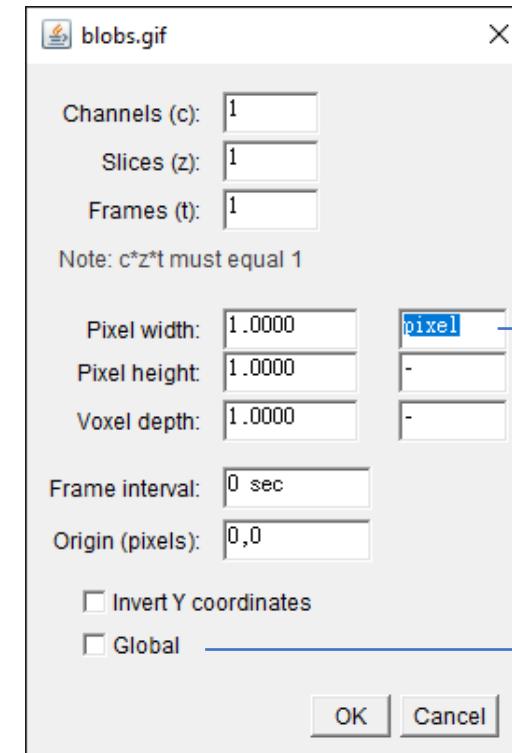
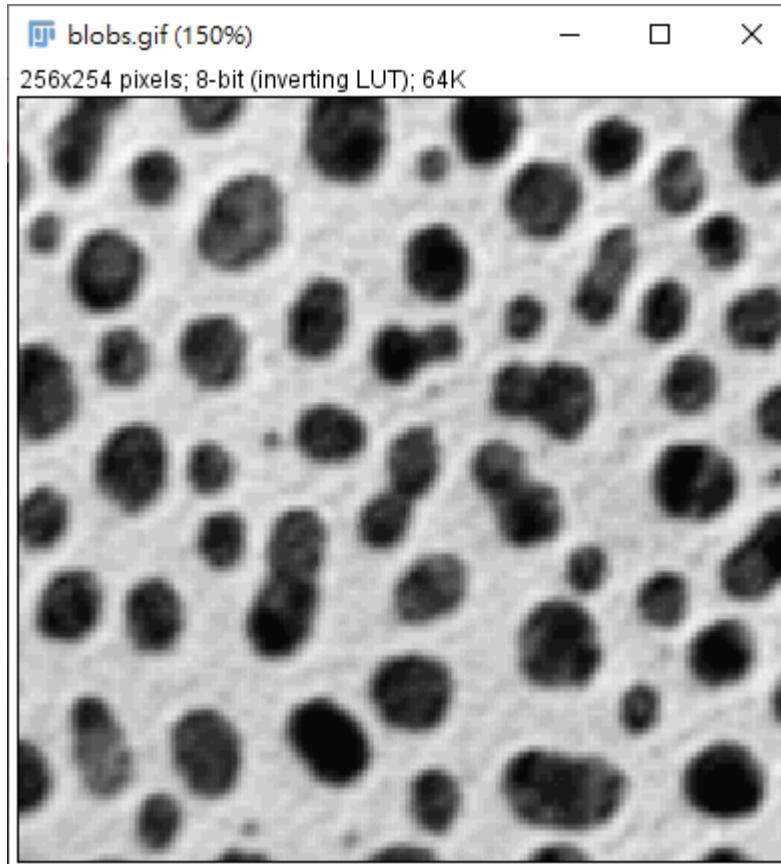
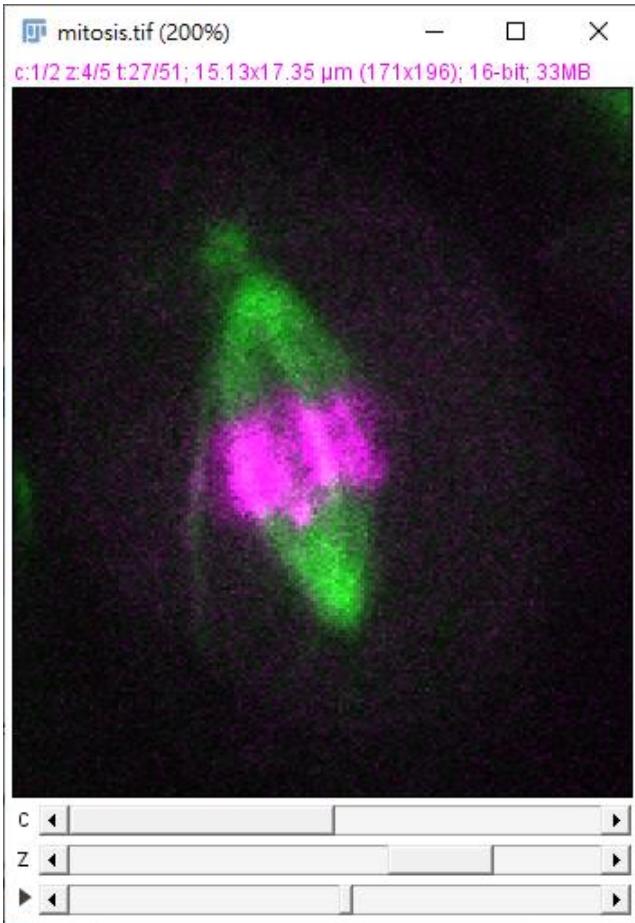


Image -> Overlay -> Flatten



# Pixel size calibration

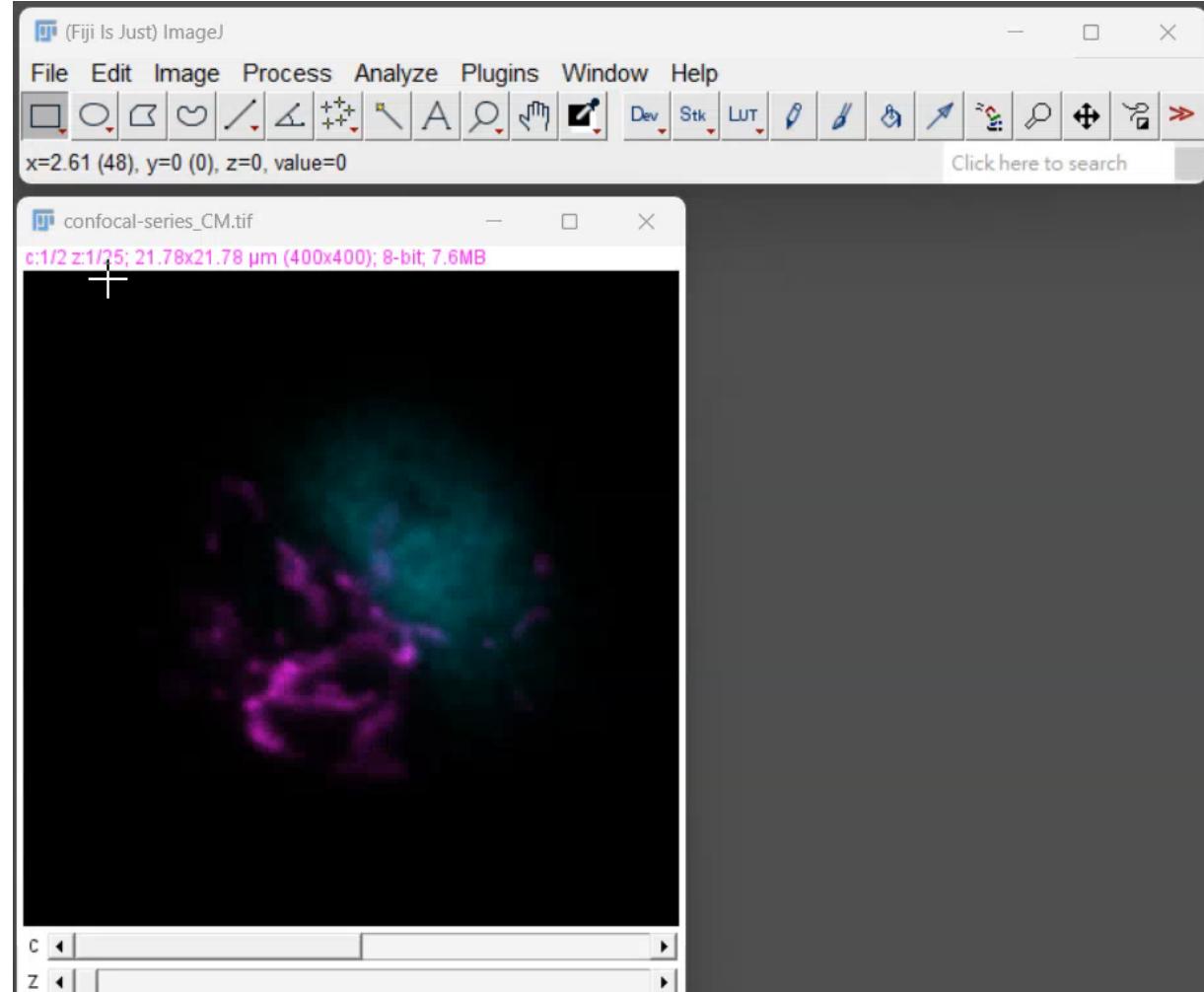
- Image -> Properties



um  
micron

Apply to  
other image  
windows

# Visualization Toolset (super useful for presentation figure prepare)



**Install through the Fiji updater**

- Visualization Toolset
- IBMP-CNRS
- BioVoxcel

Video

[https://github.com/kwolbachia/Visualization\\_toolset](https://github.com/kwolbachia/Visualization_toolset)



ICOB Imaging Core

41

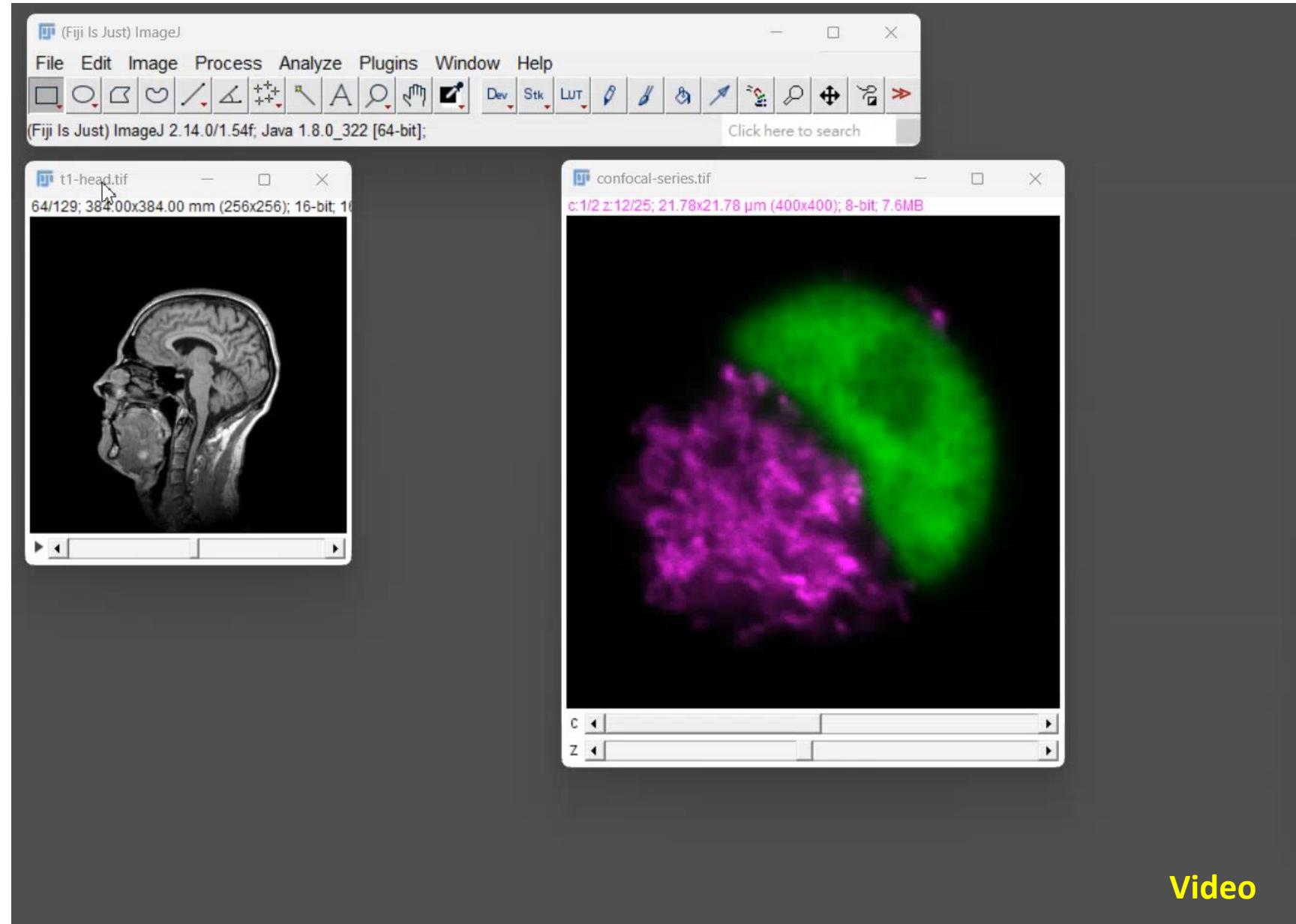
# Summary

- **Always use the original image for analysis (especially for signal intensity)**
  - If the raw data isn't RGB, then don't convert it before analysis!
- **Create an RGB copy of your image for display**
  - Keep the RGB copy separate, so you always retain it *and* the raw image

Exception : Brainbow, Skinbow .....

# Orthogonal View

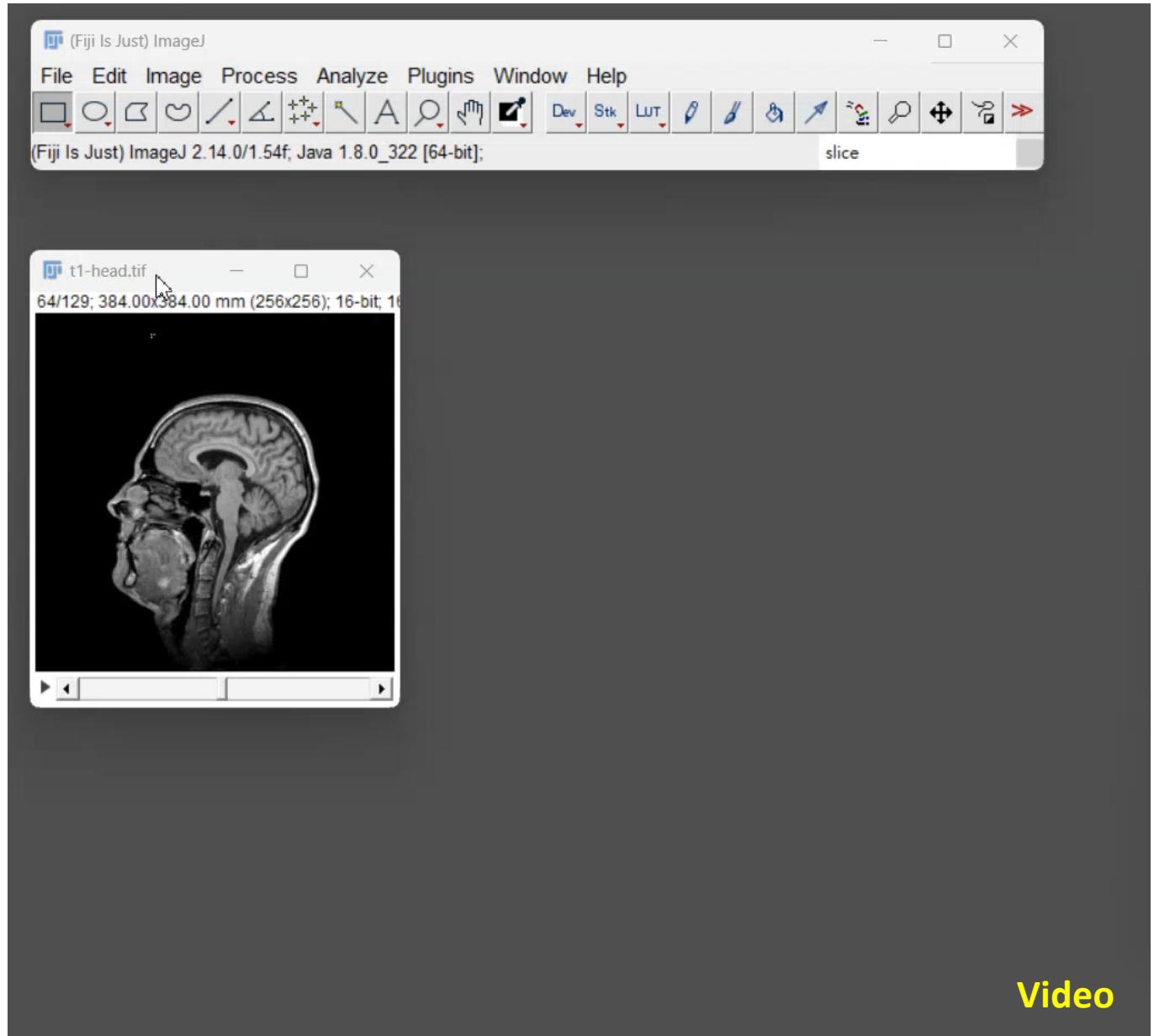
- Image -> Stacks -> orthogonal view
- Multiple Channel:  
XZ, YZ -> RGB Format



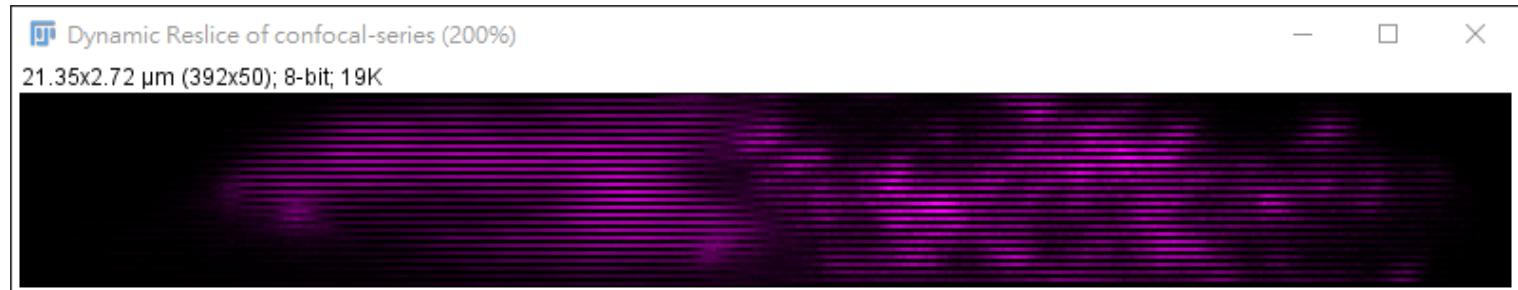
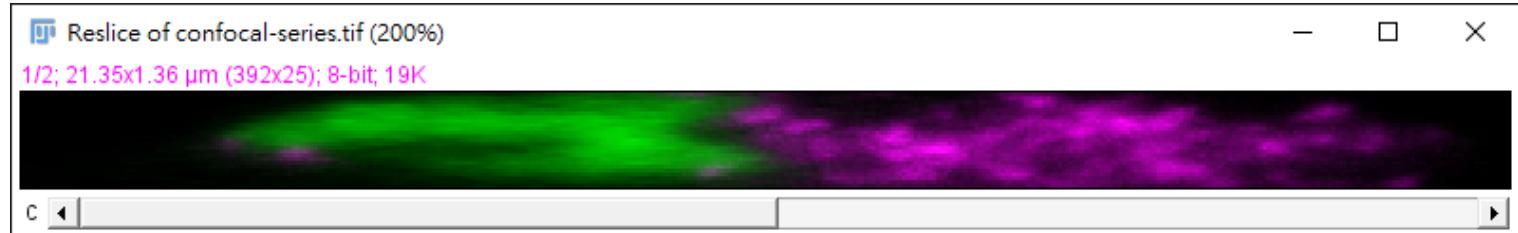
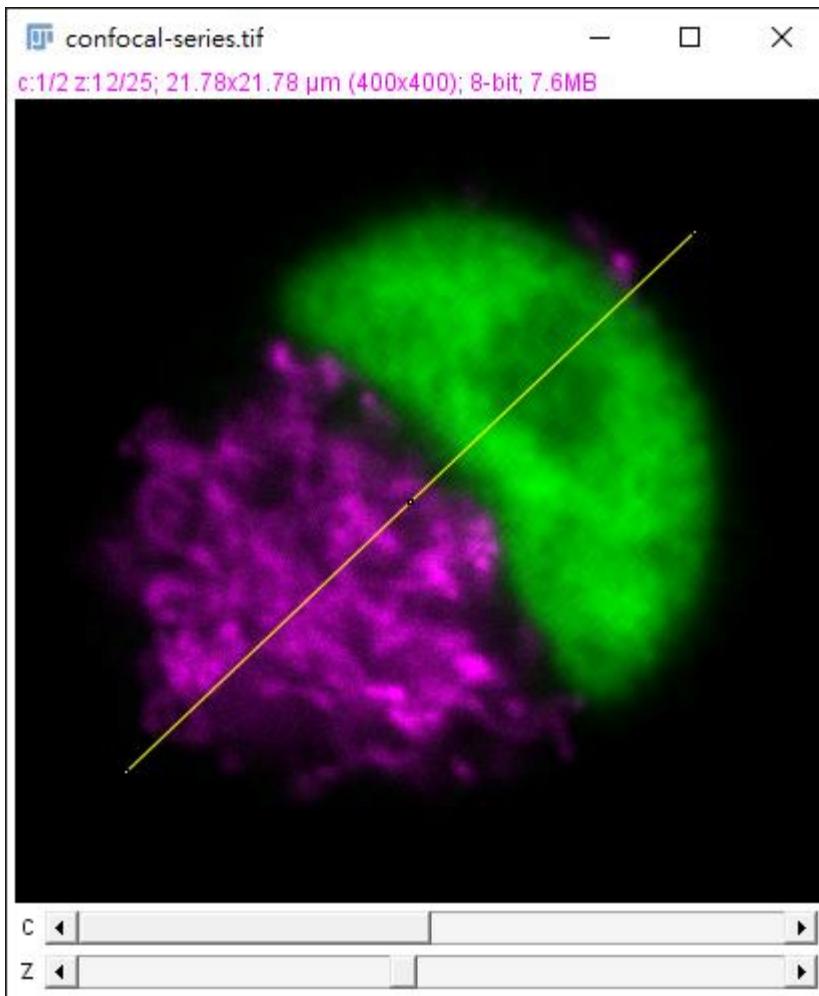
Video

# Dynamic slice and Reslice

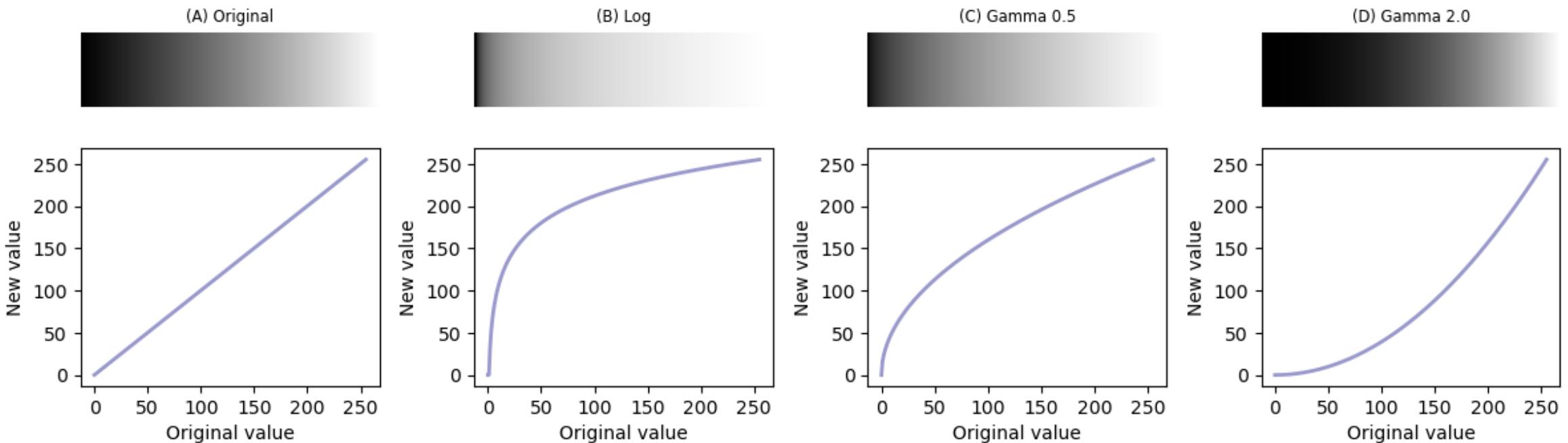
- Image -> Stacks -> Dynamic Slice
- Image -> Stacks -> Reslice [/]



# Reslice is more useful for multiple channel images



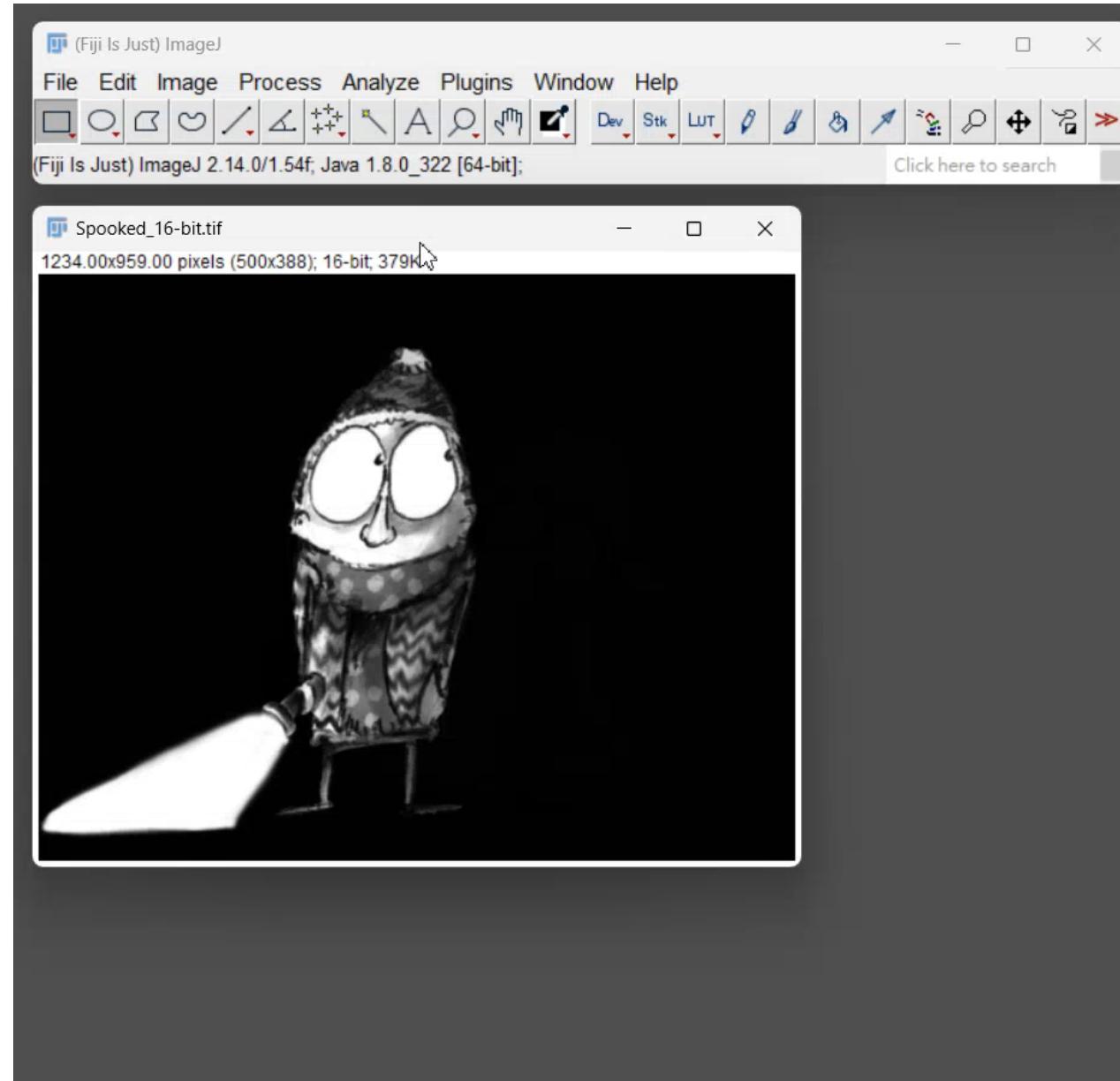
# Gamma adjustment



- In FIJI/ImageJ built-in function, Gamma adjustment change the pixel value!  
(in Zeiss Zen: change the display only!)
- If any nonlinear operations are used, these should always be noted in the figure legend!

# Gamma adjustment

- Use Visualization Toolset
- Adjust gamma without changing pixel value
- Reset to linear LUT :  
Change to another LUT and  
then reselect the LUT you  
need!



Video

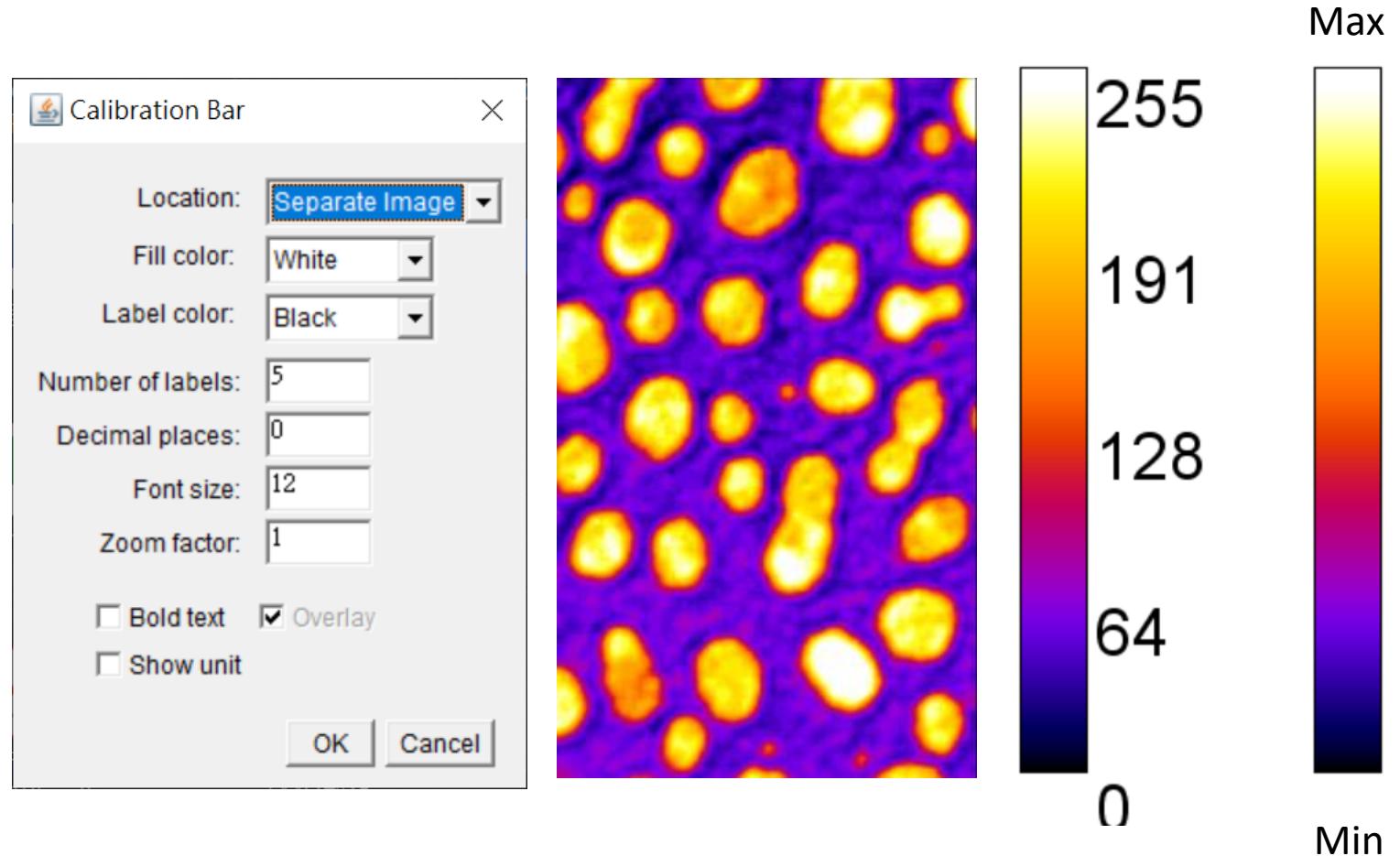
Example image from: <https://bioimagebook.github.io/>



ICOB Imaging Core

# Calibration Bar (Color scale)

- Analysis -> Tools -> Calibration Bar
- Only work for single channel image
- Brightness and Contrast affect the Calibration



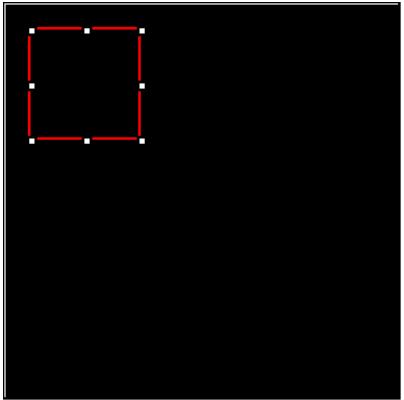
# Community-developed checklists for publishing images and image analysis

## Image colors and channels

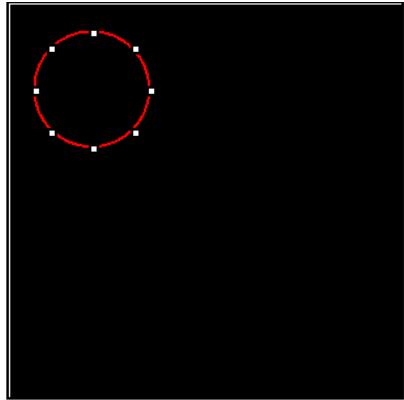
|   |   |
|---|---|
|    | Annotation of channels (staining, marker etc.) visible                        |
|    | Adjust brightness/contrast, report adjustments, use uniform color-scales      |
|    | Image comparison: use same adjustments  |
|    | Multi-color images: accessible to color blind                                 |
|    | Channel color high visibility on background                                   |
|    | Provide grey-scale for each color channel                                     |
|    | Provide color scales for intensity values (greyscale, color, pseudo color...) |
|  | Pseudo-colored images: additionally provide greyscale version for comparison. |
|  | Gamma adjustments: additionally provide linear-adjusted image for comparison  |



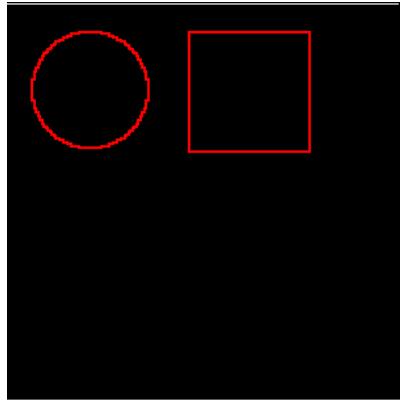
# Regions of interest (ROI)



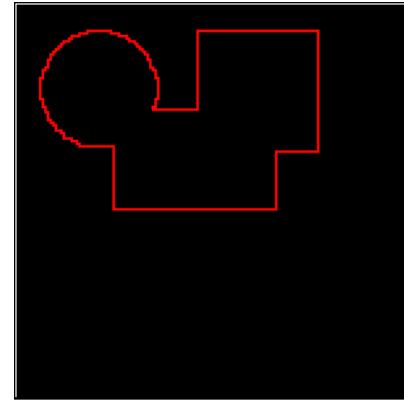
draw a rectangle  
using the rectangle tool



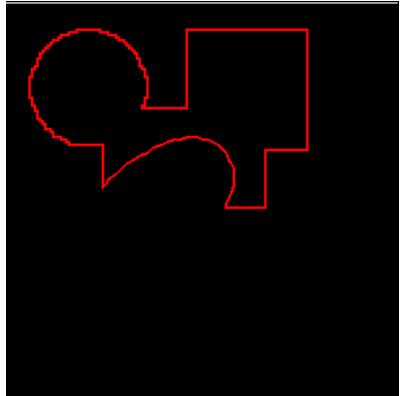
draw a circle  
using the circle tool



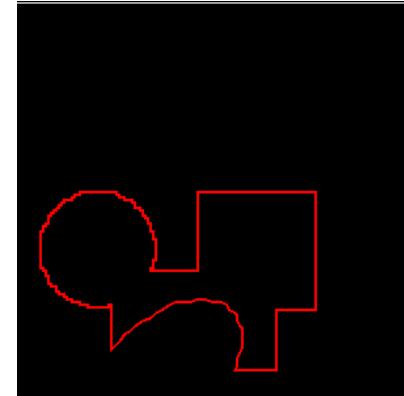
add objects to the ROI  
by holding SHIFT



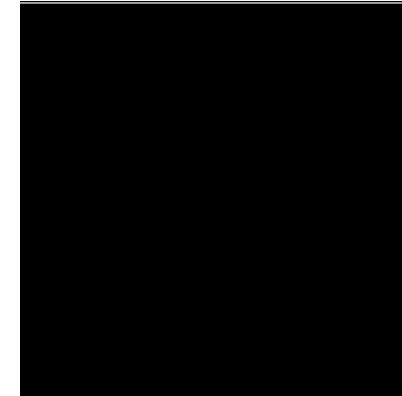
unite objects to the ROI  
by holding SHIFT



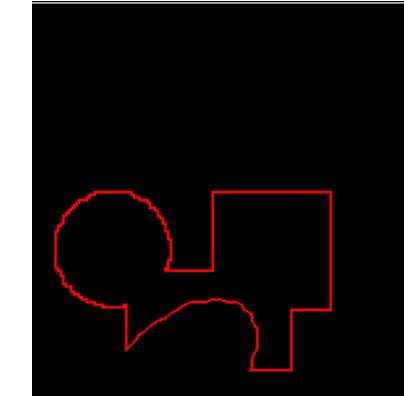
subtract objects to the  
ROI by holding ALT



move the ROI  
by clicking inside and  
move the mouse



delete the ROI  
by clicking outside  
(Ctrl + Shift + A)



Recover the last ROI by using menu  
Edit > Selection > Restore Selection  
(Ctrl + Shift + E)

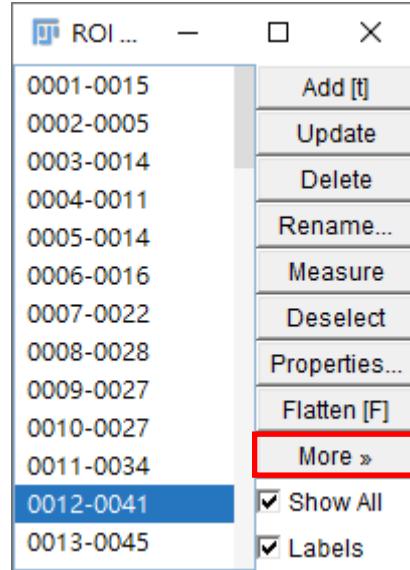
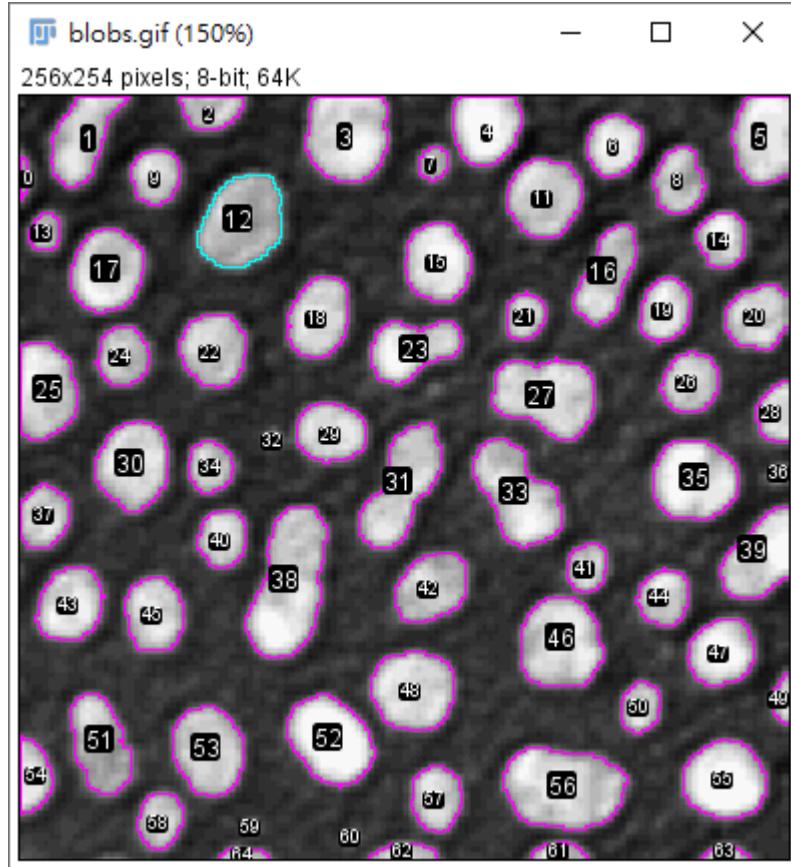
Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

# ROI manager

- Analysis -> Tools -> ROI Manager
- Save the ROI, so you can repeat the measurement!



- Single ROI: \*.roi
- Multiple ROI: \*.zip
- Deselect
  - > Not select a specific ROI
  - > “All”

# Useful Selection Tool

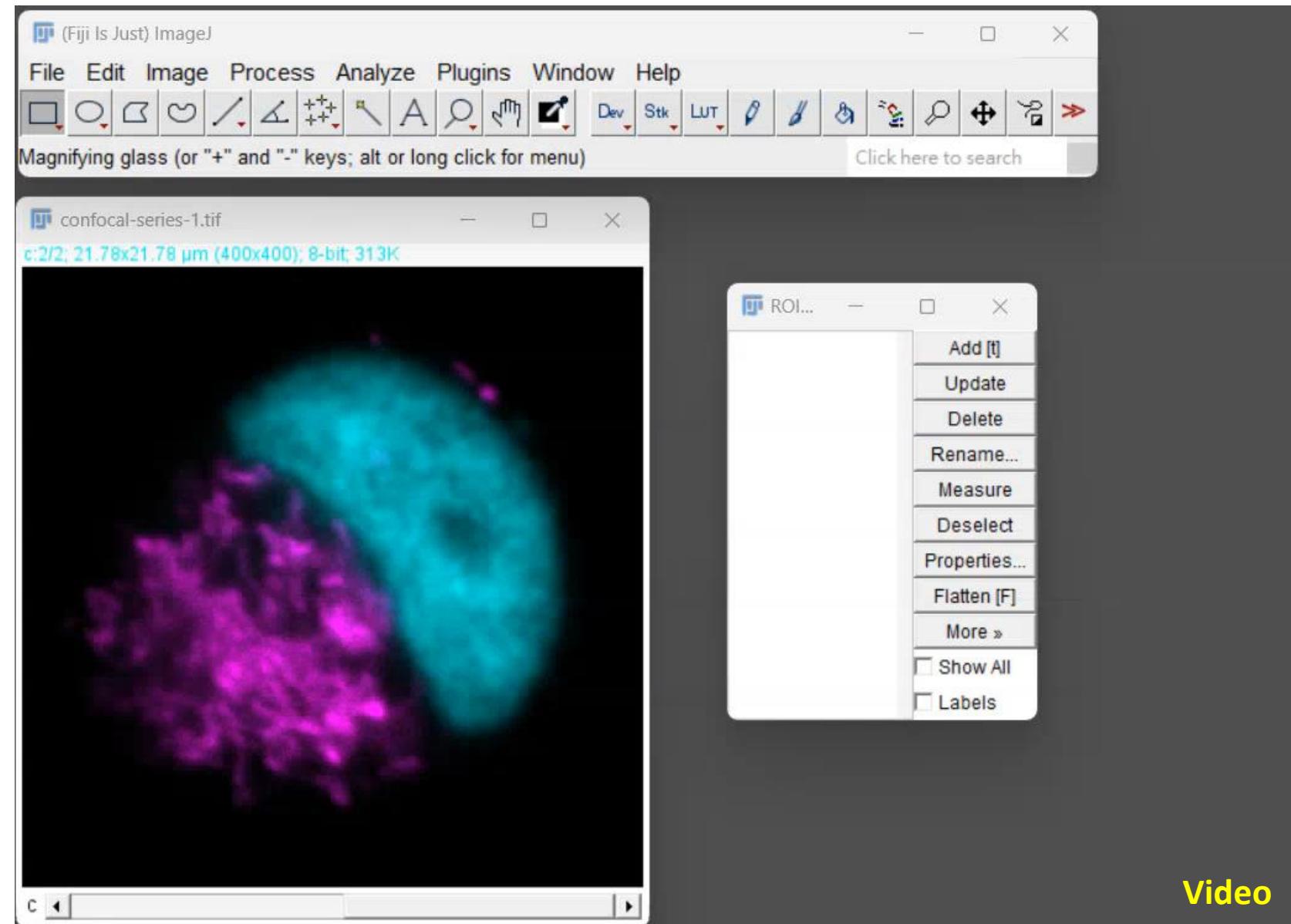
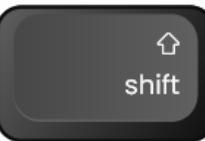
- **Selection Brush Tools**
- **Wand Tools**

## Multiple ROI

- Add to ROI manager

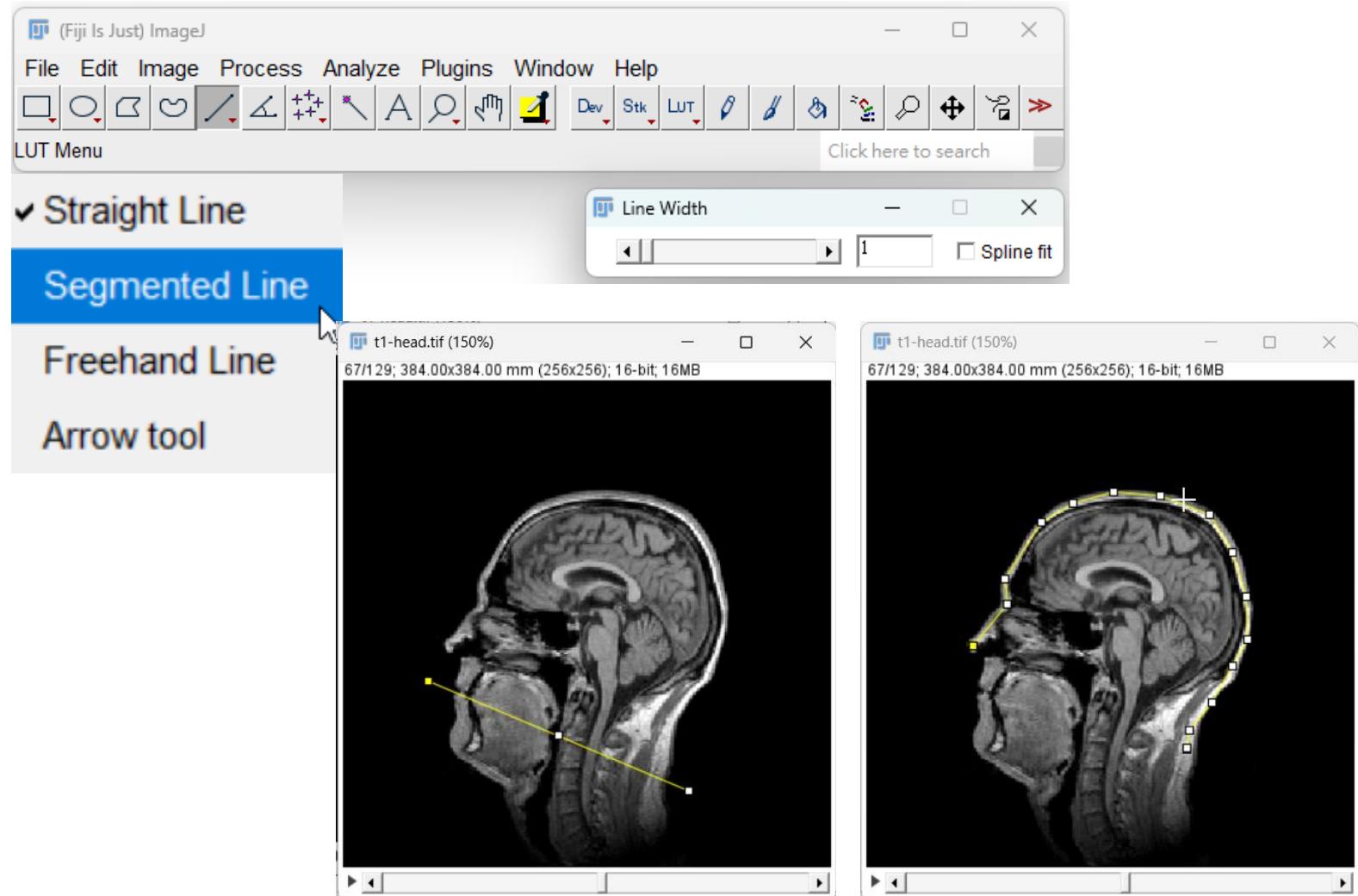


- Select None



# Line tool

- Measure length
  - Add to ROI manager("T")
- Measure
- Plot profile

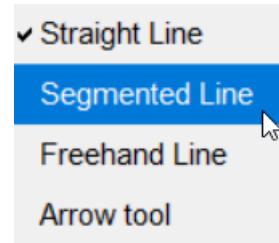


# Plot Profile

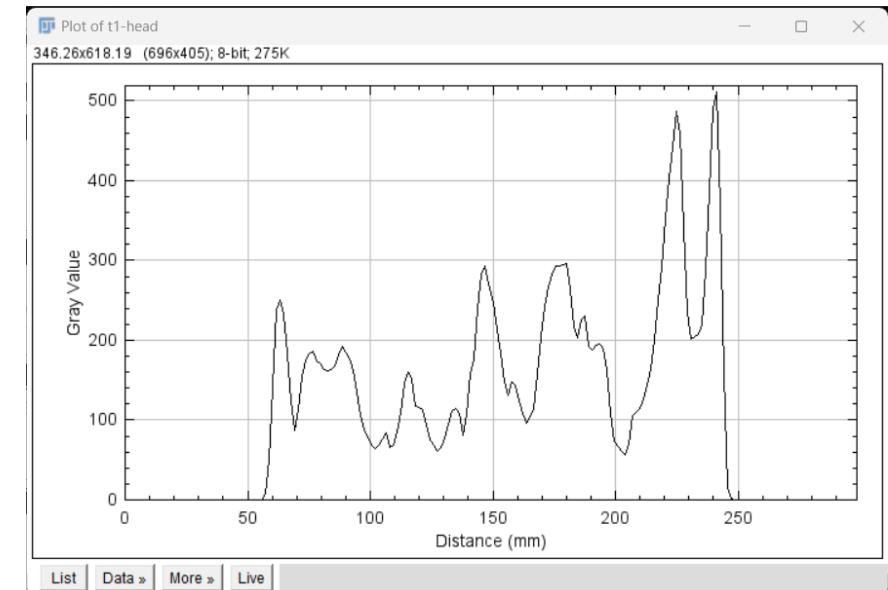
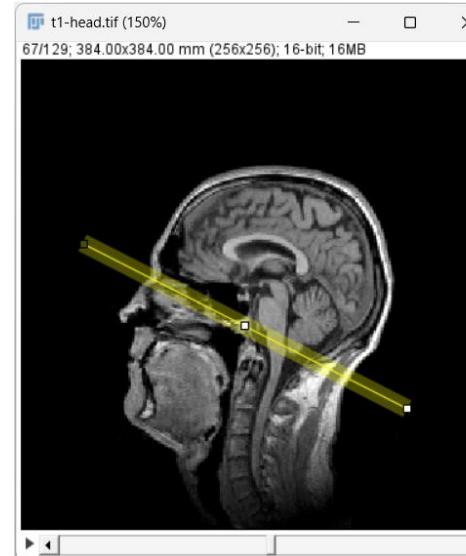
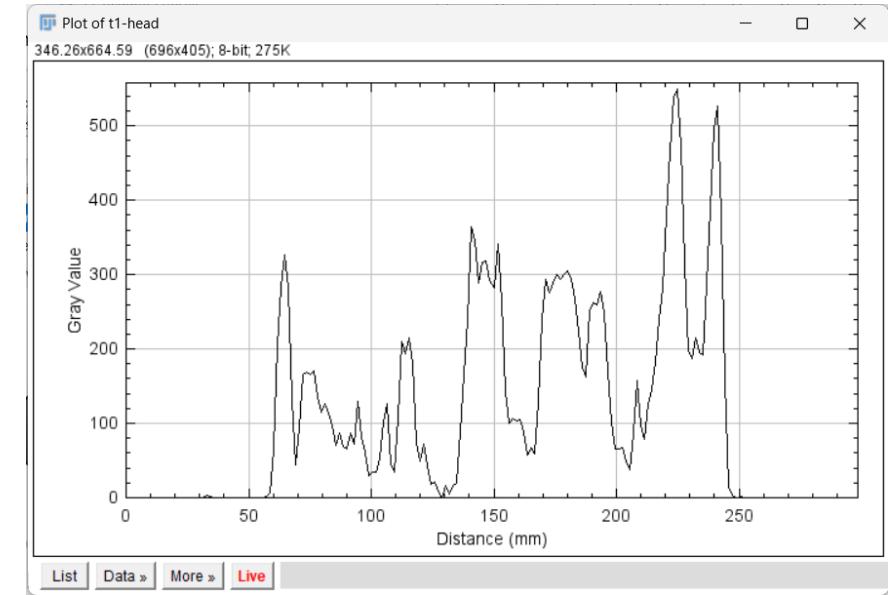
- Analyze -> Plot profile



- Can apply to different Line type, width

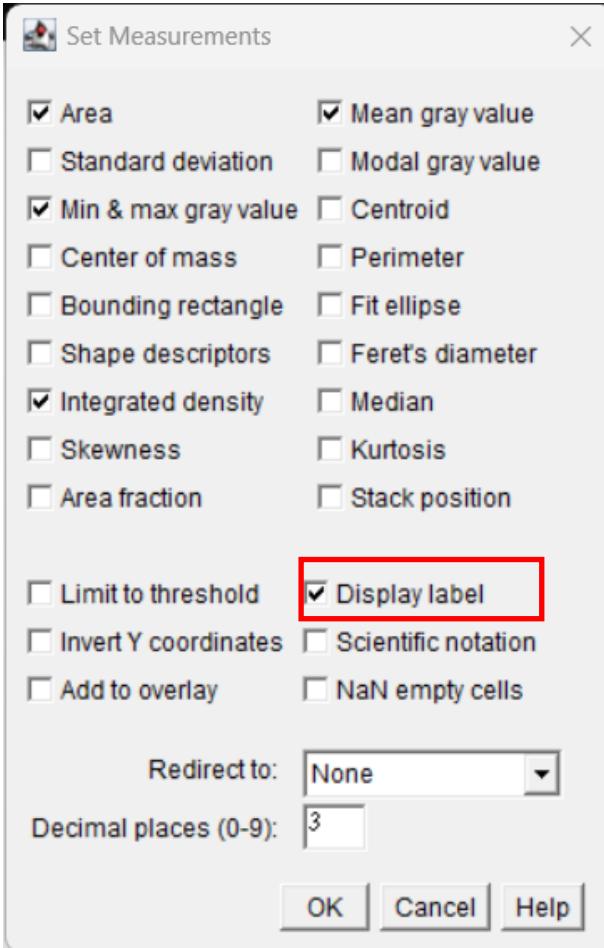


- Note the scale information, unit!



# Measurement

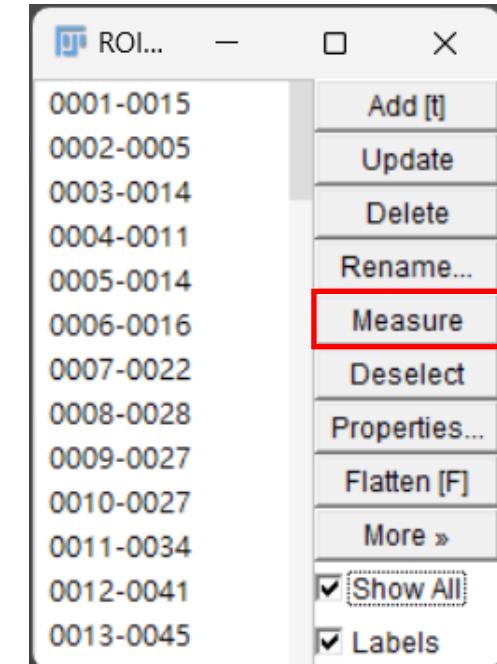
- Analyze > Set measurements



- Analyze > Measure



- ROI Manager > Measure
- Unit: Image -> Properties..  
(Ctrl + Shift + P)



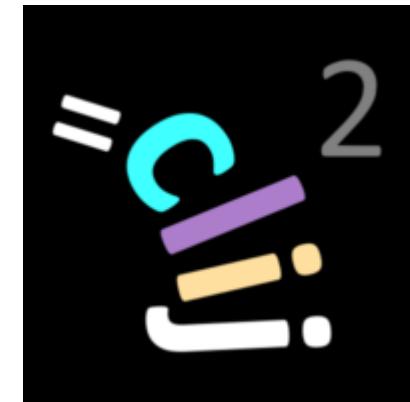
| Label  | Area   | Mean    | Min | Max | IntDen   | RawIntDen   |
|--|--------|---------|-----|-----|----------|-------------|
| 1 confocal-series.tif:cyan_ROI:15                                      | 82.285 | 2.167   | 0   | 93  | 178.345  | 60143.000   |
| 2 confocal-series.tif:magenta_ROI:15                                   | 35.952 | 119.766 | 37  | 253 | 4305.806 | 1452041.000 |
| 3 confocal-series_CM.tif:cyan_ROI:c:2/2 z:8/25 - confocal-series_CM    | 82.285 | 112.240 | 45  | 187 | 9235.684 | 3114537.000 |
| 4 confocal-series_CM.tif:magenta_ROI:c:2/2 z:8/25 - confocal-series_CM | 35.952 | 5.128   | 0   | 56  | 184.365  | 62173.000   |

- Use Bio-Format importer to open the image can reveal more details on the measurement label!

# Part II: Bioimage Analysis Workflow

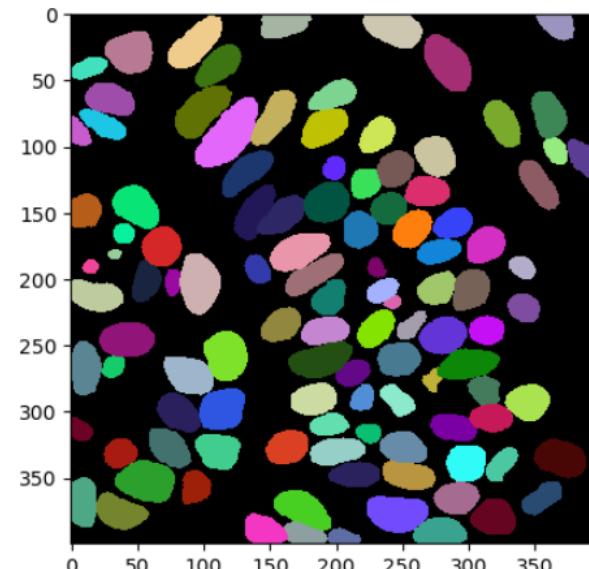
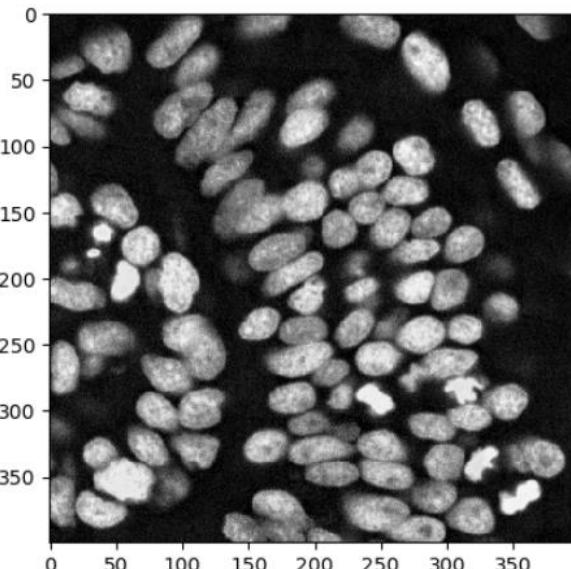
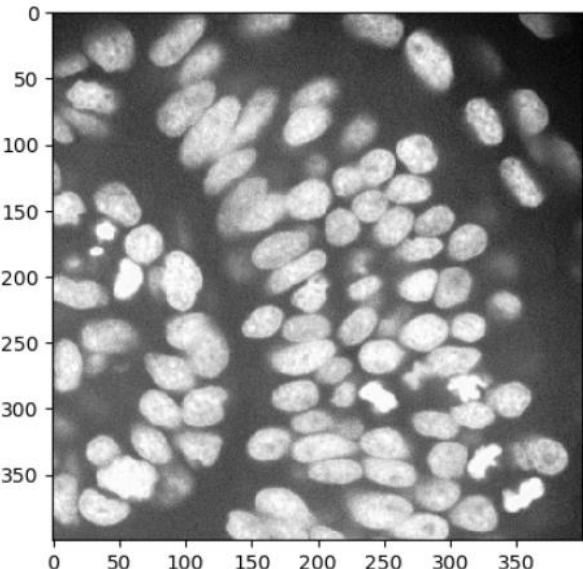
## Part II: Typical bioimage analysis workflow

- Segmentation
- Selection, ROI List, Mask, Label
- ImageJ Macro
- CLIJ (GPU-accelerated image processing )
- Interactive bioimage analysis and automatic code generation
- Future of bioimage analysis (AI)



# 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         | 29.520883         |
| 4   | 791.0  | 49132.515803   | 36.382253         | 27.718301         |
| ... | ...    | ...            | ...               | ...               |
| 105 | 238.0  | 30477.126050   | 20.252197         | 15.276536         |
| 106 | 615.0  | 32886.154472   | 41.030760         | 19.874280         |
| 107 | 110.0  | 33042.445455   | 14.366347         | 9.945911          |
| 108 | 222.0  | 43304.180180   | 25.370549         | 11.637599         |
| 109 | 167.0  | 43378.808383   | 17.895110         | 13.015369         |

Image filtering

Image segmentation

Feature extraction

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

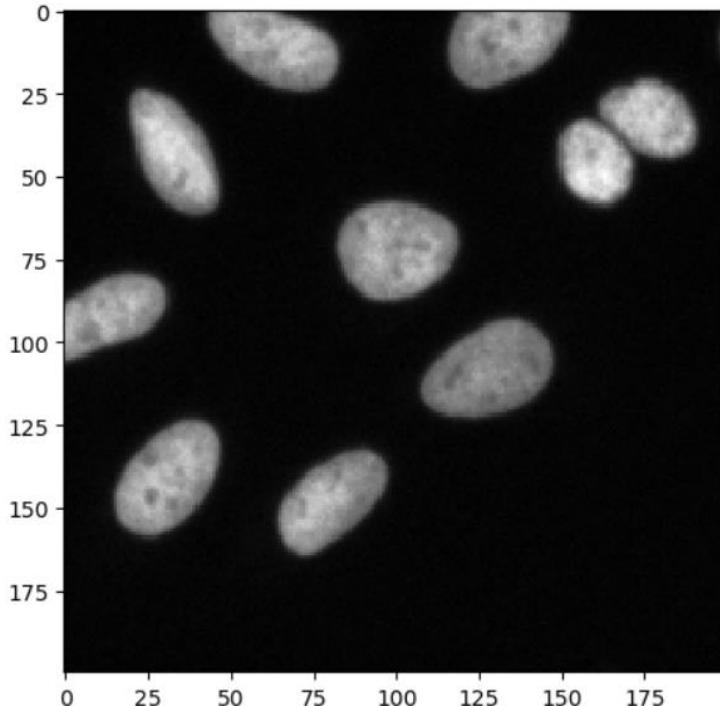
Adapted from the learning material of Dr. Robert Haase



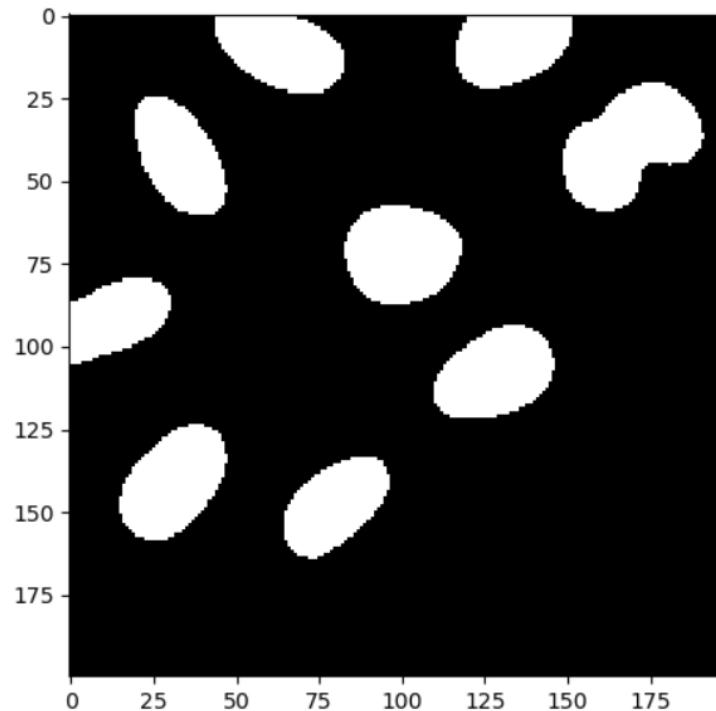
**ICOB Imaging Core**

# Terminology

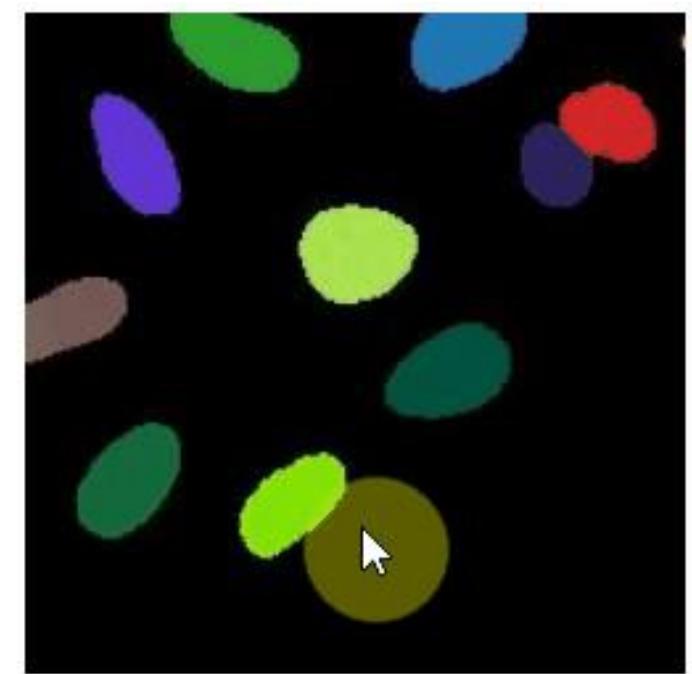
Intensity image



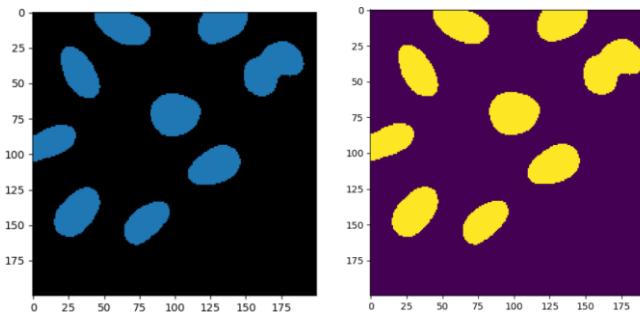
Binary image (Mask)



Label image



[y=152, x=92] = 0



Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

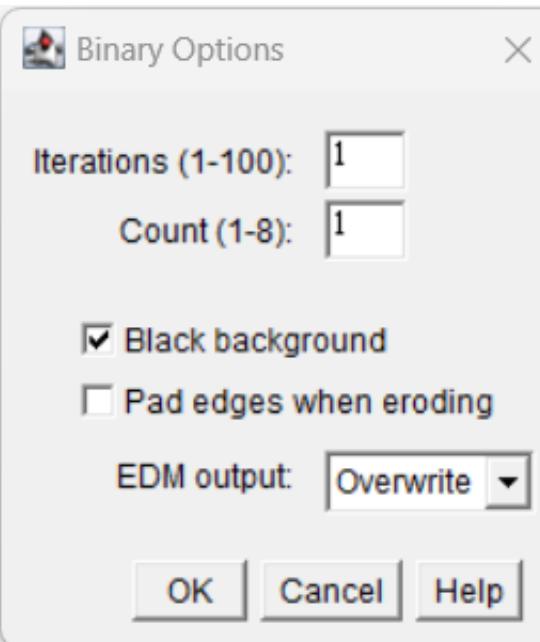
# Mask: different display mode

“Default”

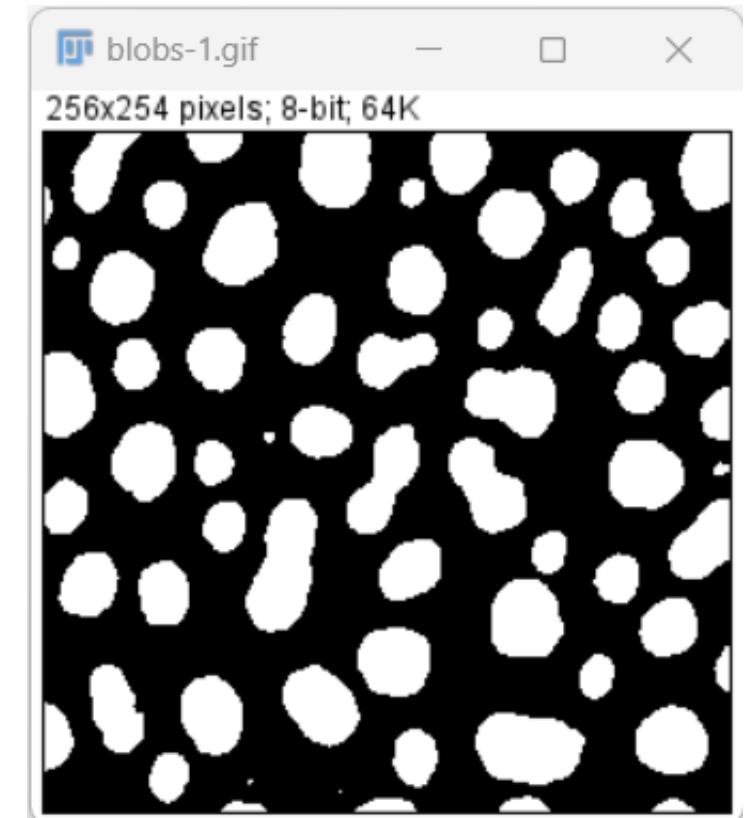


0 255

- Process -> Binary -> Options...



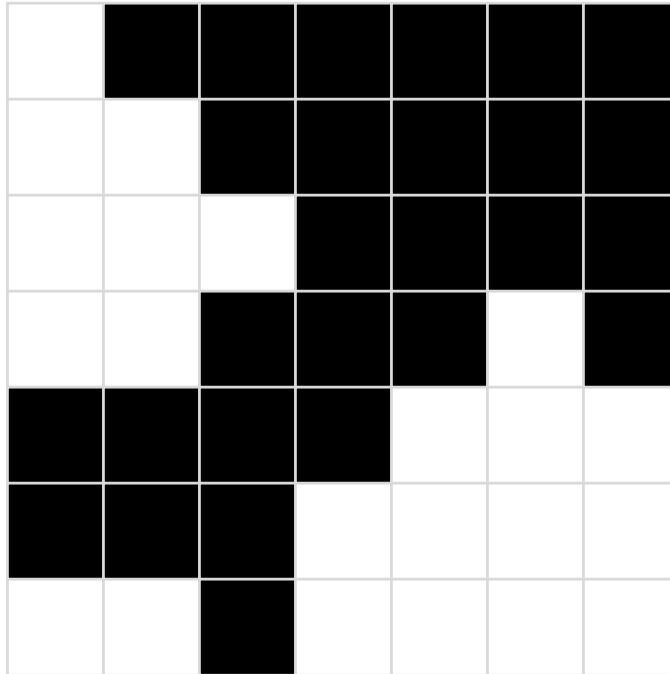
“Black background”



0 255

# 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.

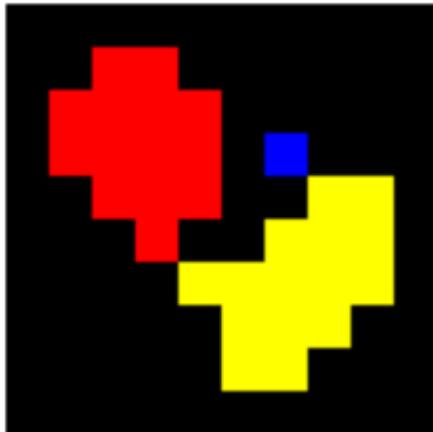


CCA

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 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 |

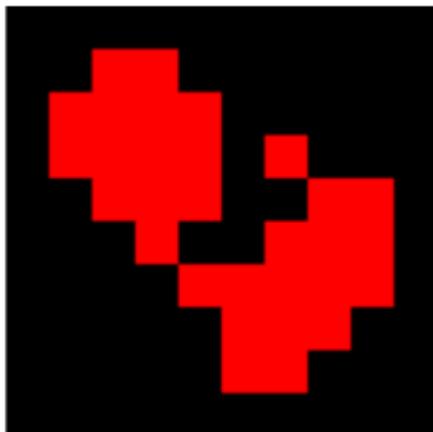
# 4-connectivity vs 8-connectivity

4-connectivity



- Analyze Particle in FIJI/ ImageJ: 8-connectivity
- 6-connectivity is similar to 4-connectivity, but in 3D.
- 26-connectivity is similar to 8-connectivity, but in 3D

8-connectivity

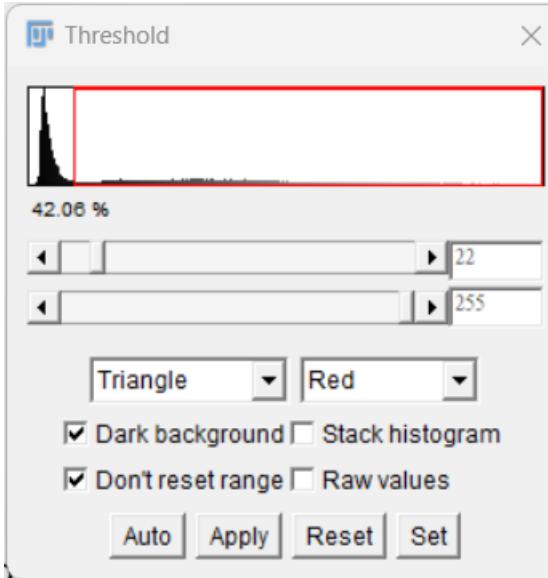


# Thresholding

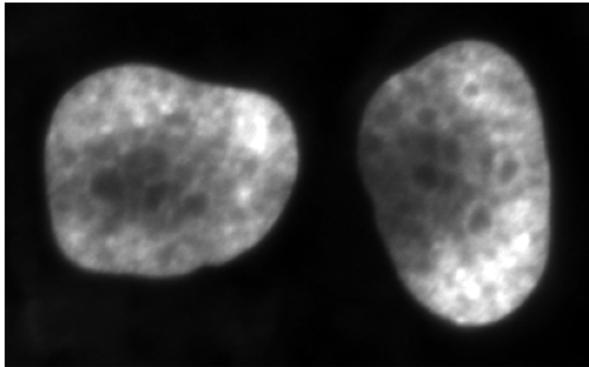
- Image -> Adjust -> Threshold...



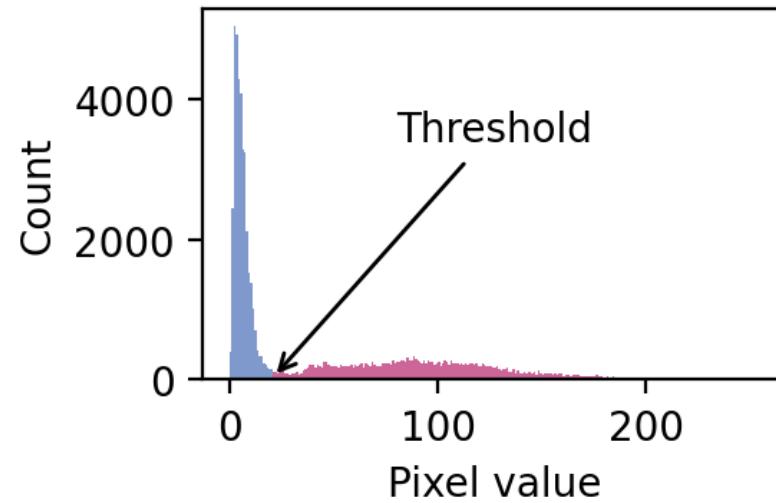
- Find an algorithm that fit to your data set!



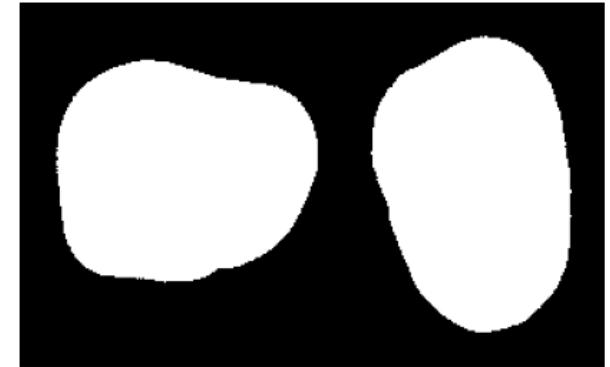
(A) Image



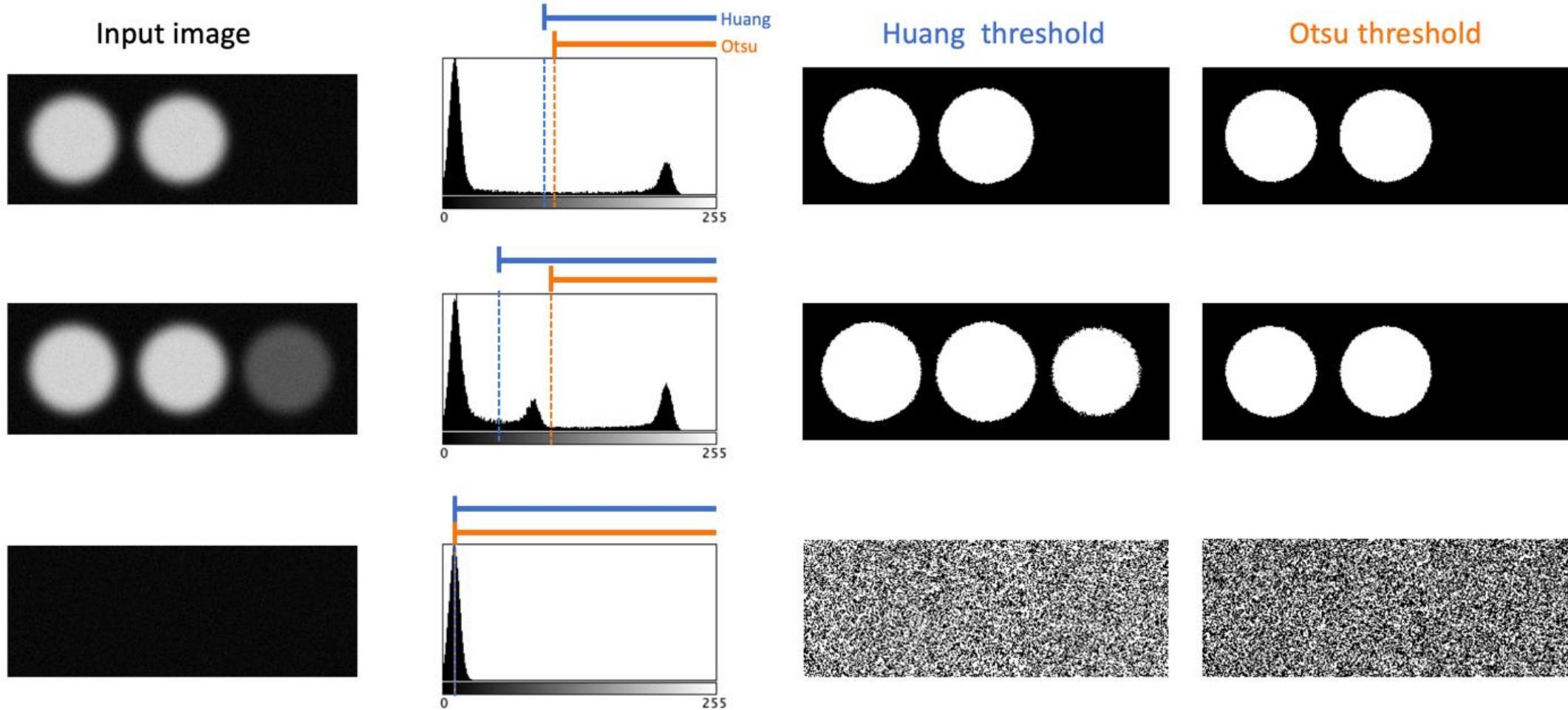
(B) Histogram



(C) Thresholded

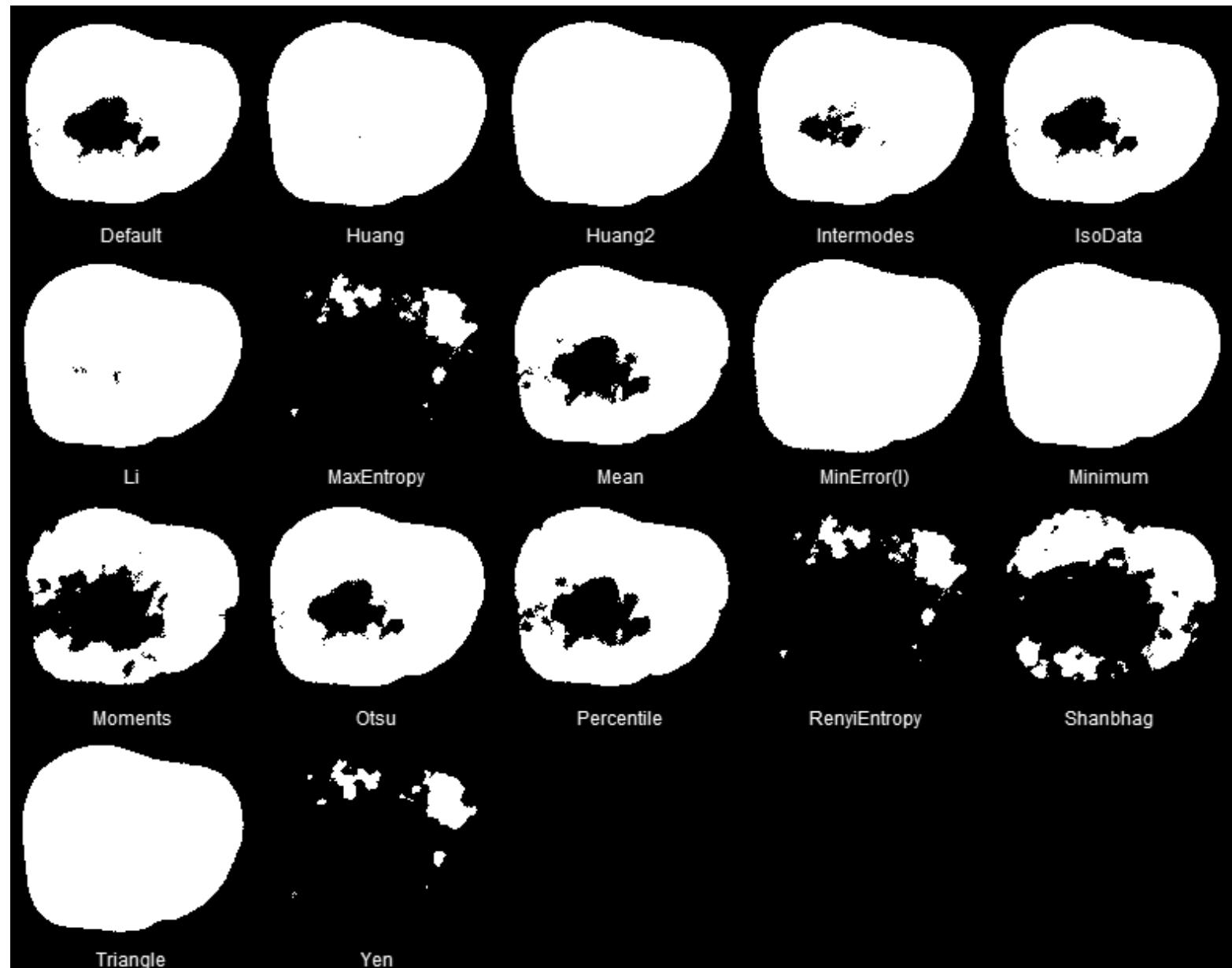
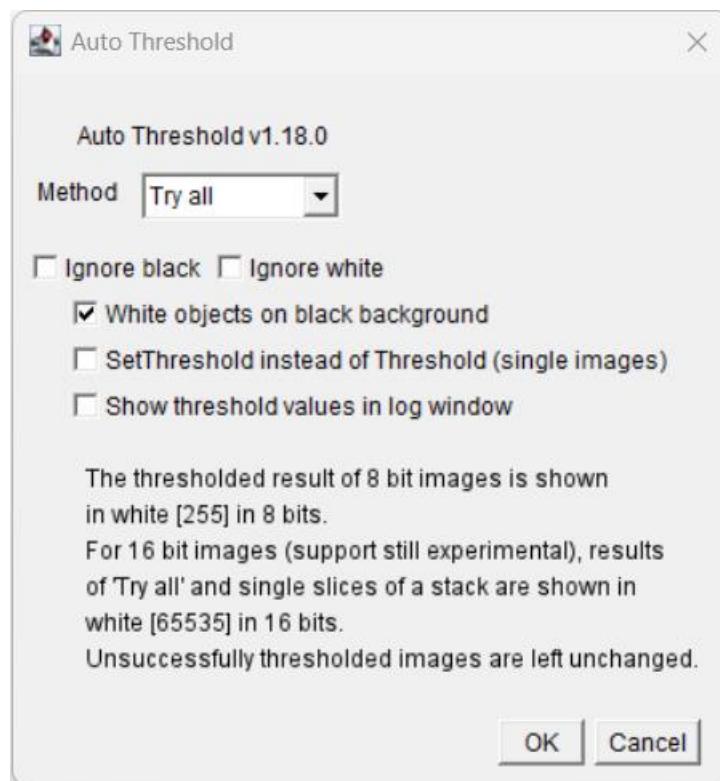


# Automatic thresholding based on histogram



# Try all

- Image -> Adjust -> Auto Threshold  
Method: Try all



# 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 Fiji (Schindelin et al. 2012)."*

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

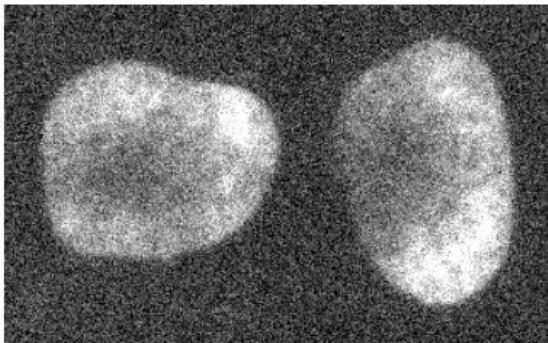
## A Threshold Selection Method from Gray-Level Histograms

NOBUYUKI OTSU

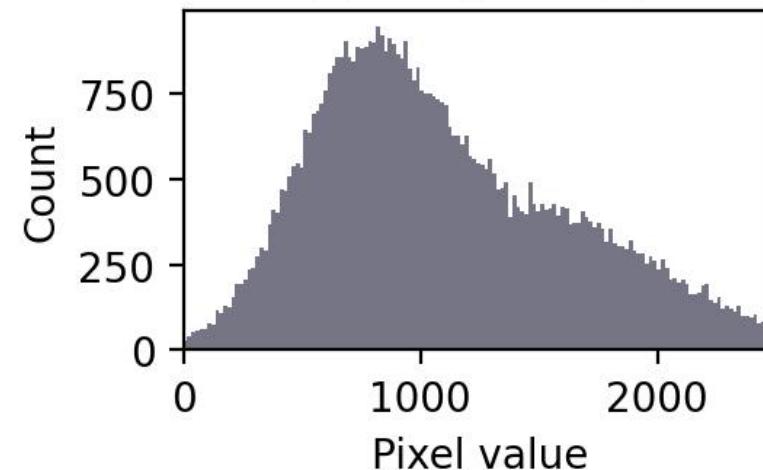
*Abstract*—A nonparametric and unsupervised method of automatic threshold selection for picture segmentation is presented. An optimal threshold is selected by the discriminant criterion, namely, so as to maximize the separability of the resultant classes in gray levels. The procedure is very simple,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

# Filtering for improving thresholding results

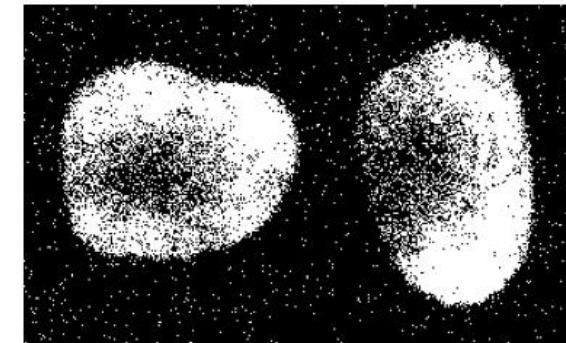
(A) Noisy image



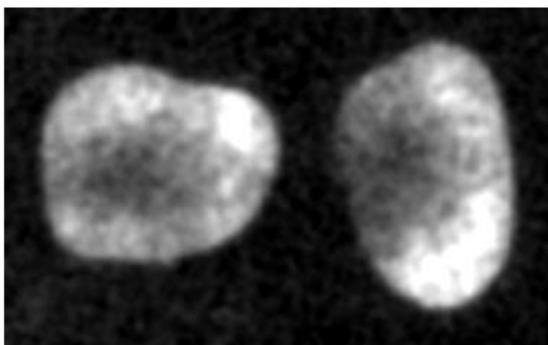
(B) Histogram of (A)



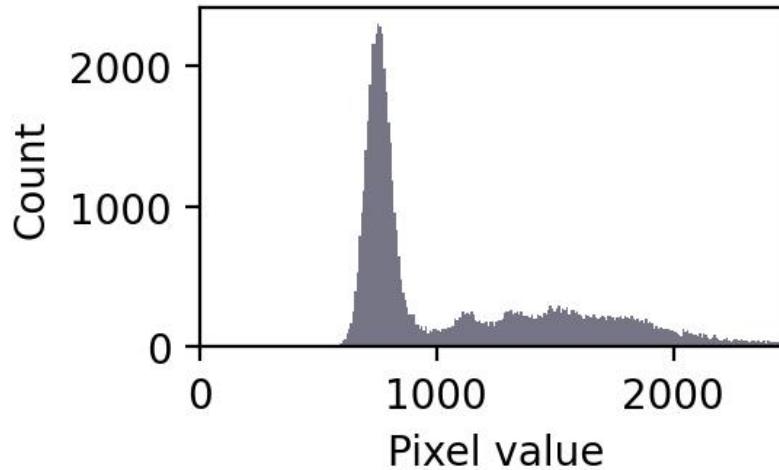
(C) Threshold applied to (A)



(D) Gaussian filtered image



(E) Histogram of (D)

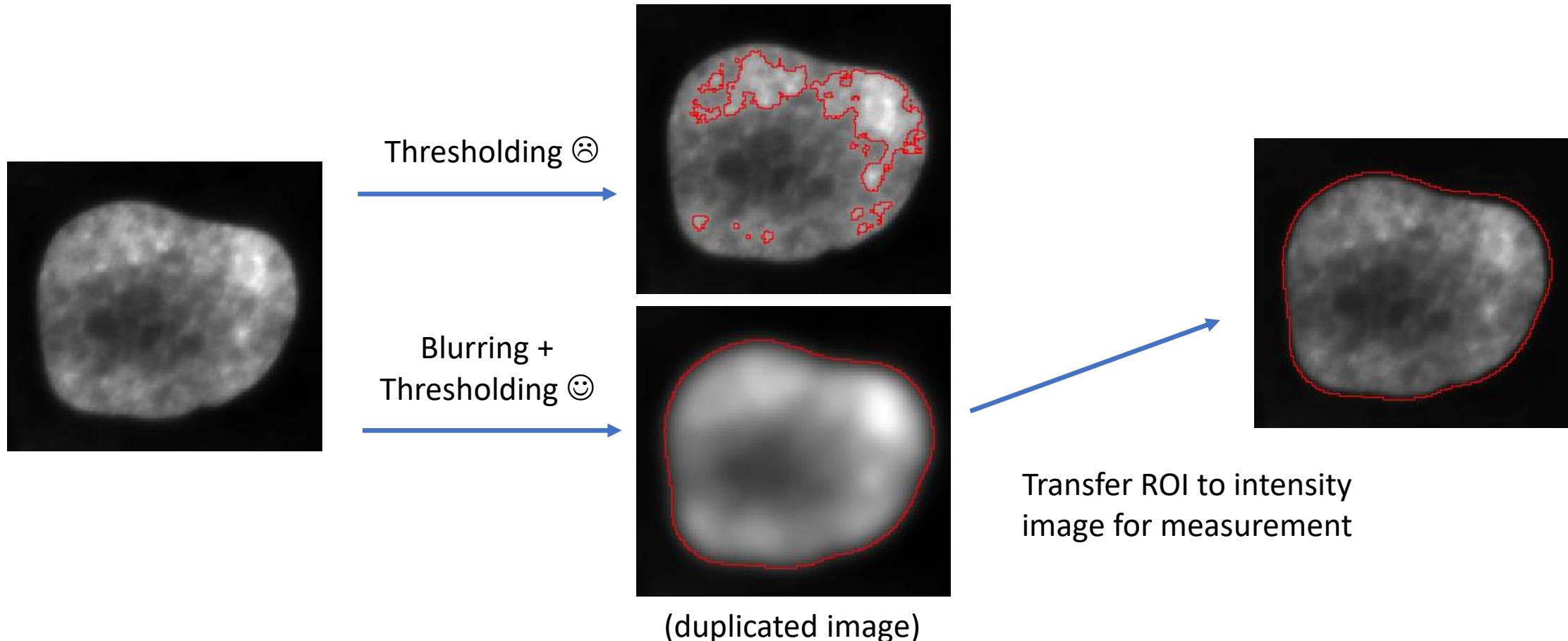


(F) Threshold applied to (D)



# 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!



Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

# Noise removal

- Gaussian filter
- Median filter (computationally expensive)

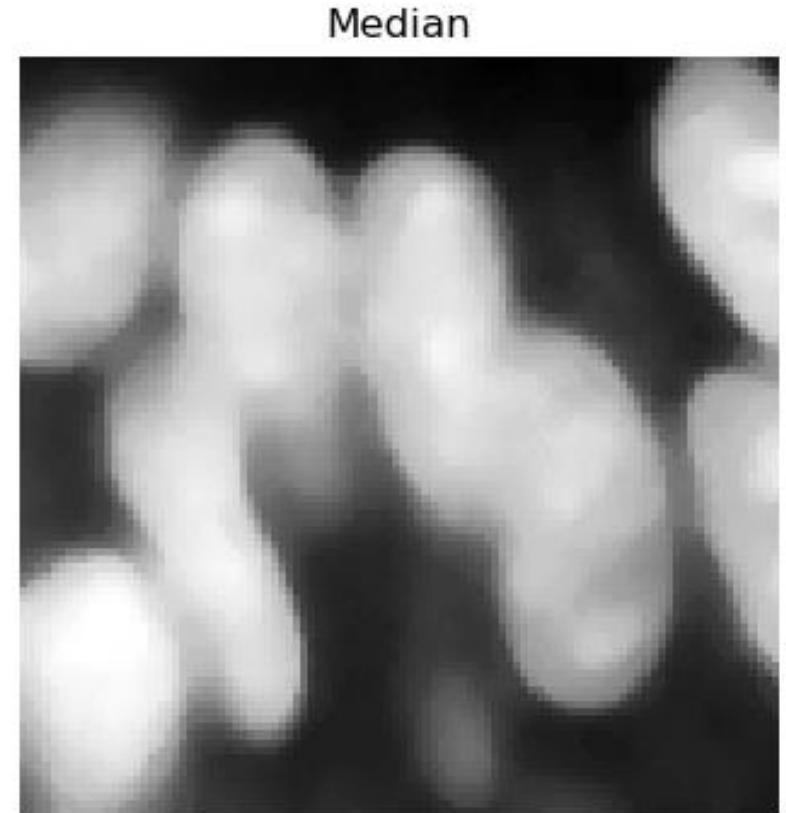
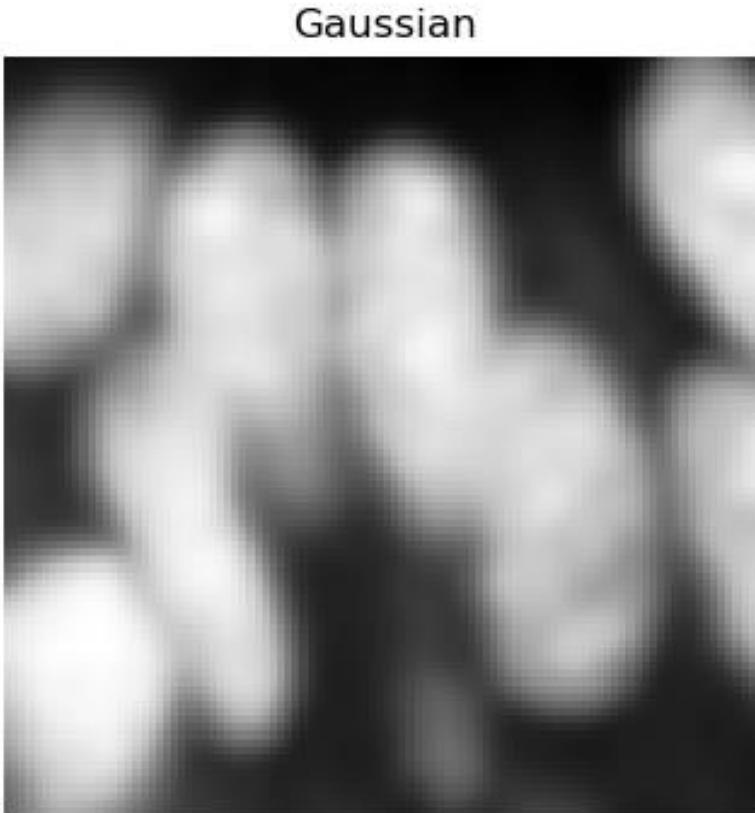
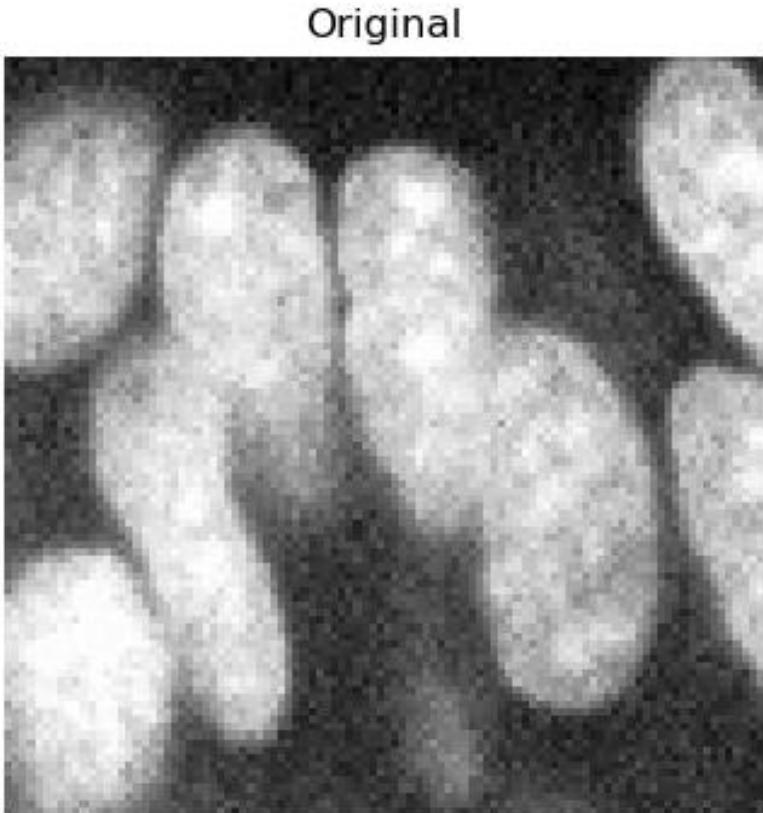


Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

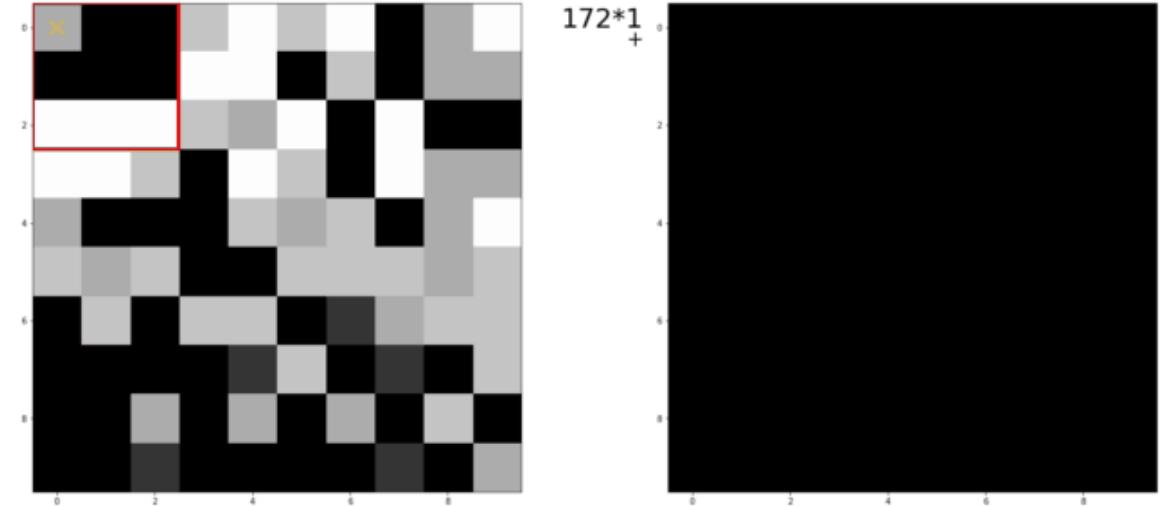
Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

# 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*



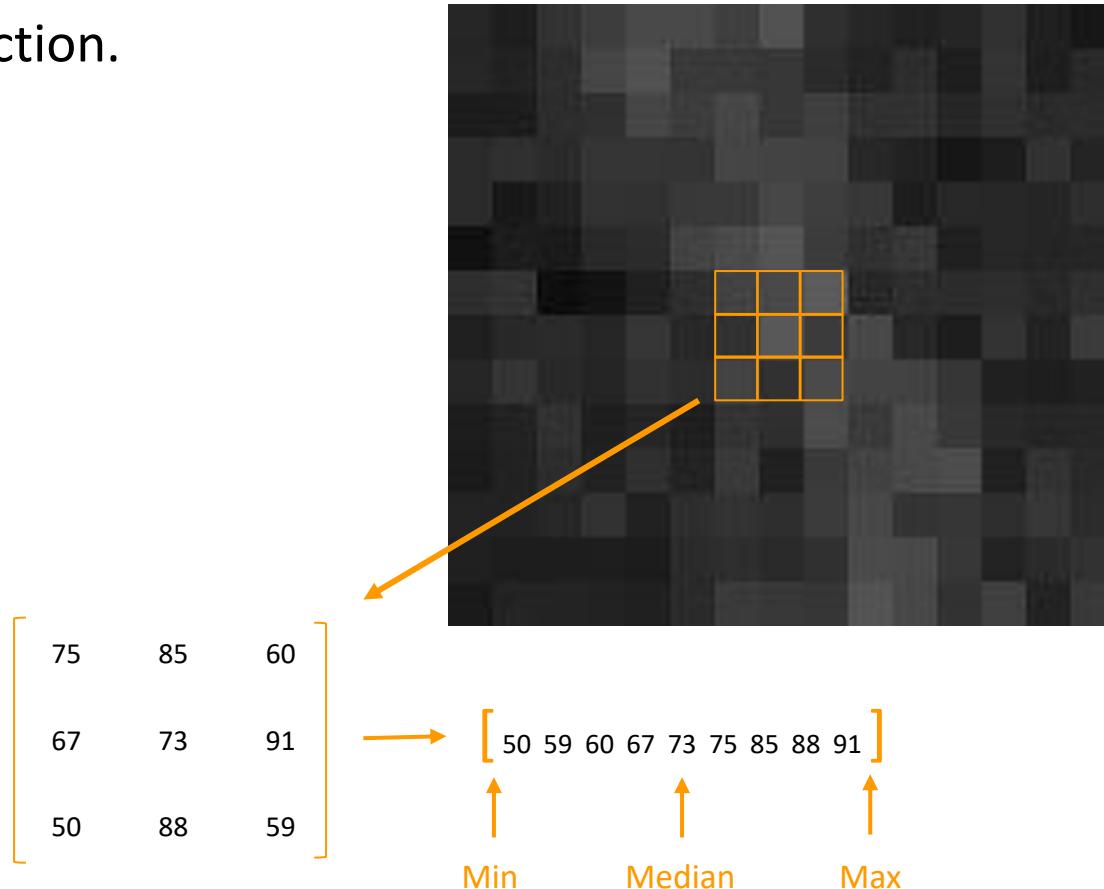
Mean filter, 3x3 kernel

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

Animation source: Dominic Waite, Oxford University  
[https://github.com/dwaithe/generalMacros/tree/master/convolution\\_ani](https://github.com/dwaithe/generalMacros/tree/master/convolution_ani)

# Nonlinear Filters

- Non linear filters also replace pixel value inside a rolling window but using a non-linear function.
- Examples: order statistics filters
  - Min
  - Median
  - Max
  - Variance
  - Standard deviation



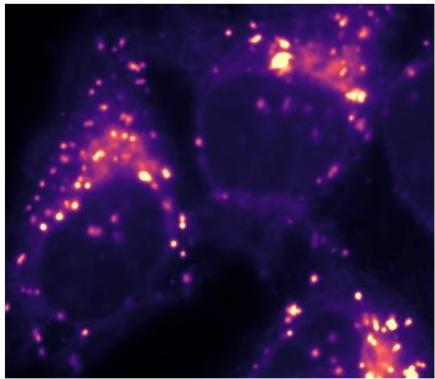
Adapted from the learning material of Dr. Robert Haase



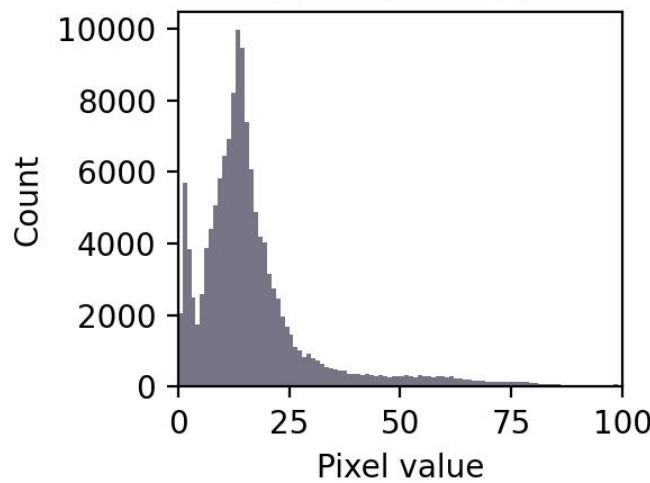
**ICOB Imaging Core**

# Background subtraction improving thresholding results

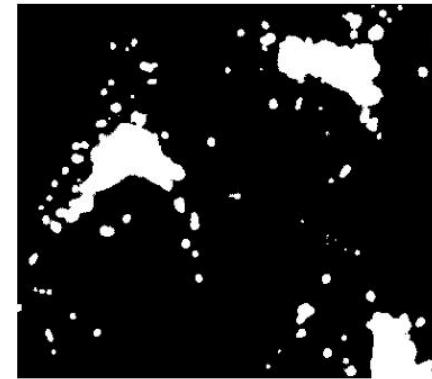
(A) Original image



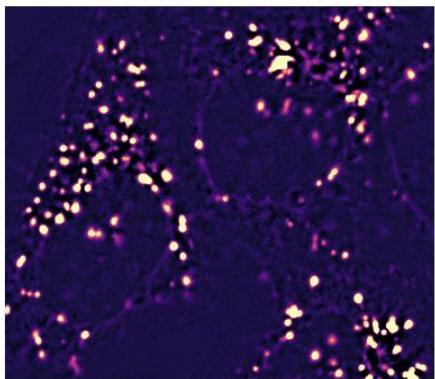
(B) Histogram of (A)



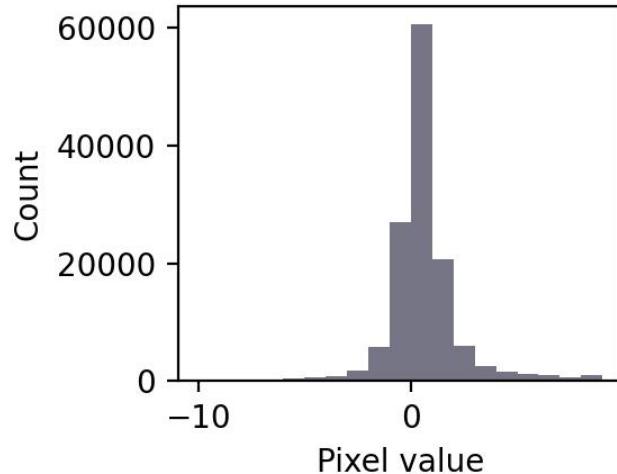
(C) Triangle threshold applied to (A)



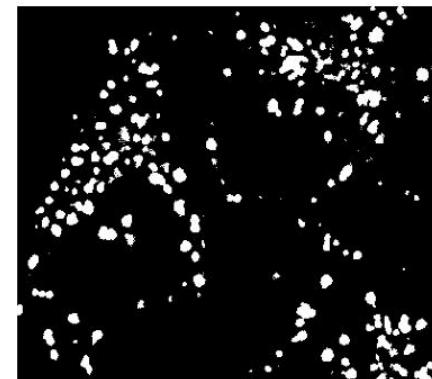
(D) 'Background' subtracted image



(E) Histogram of (D)

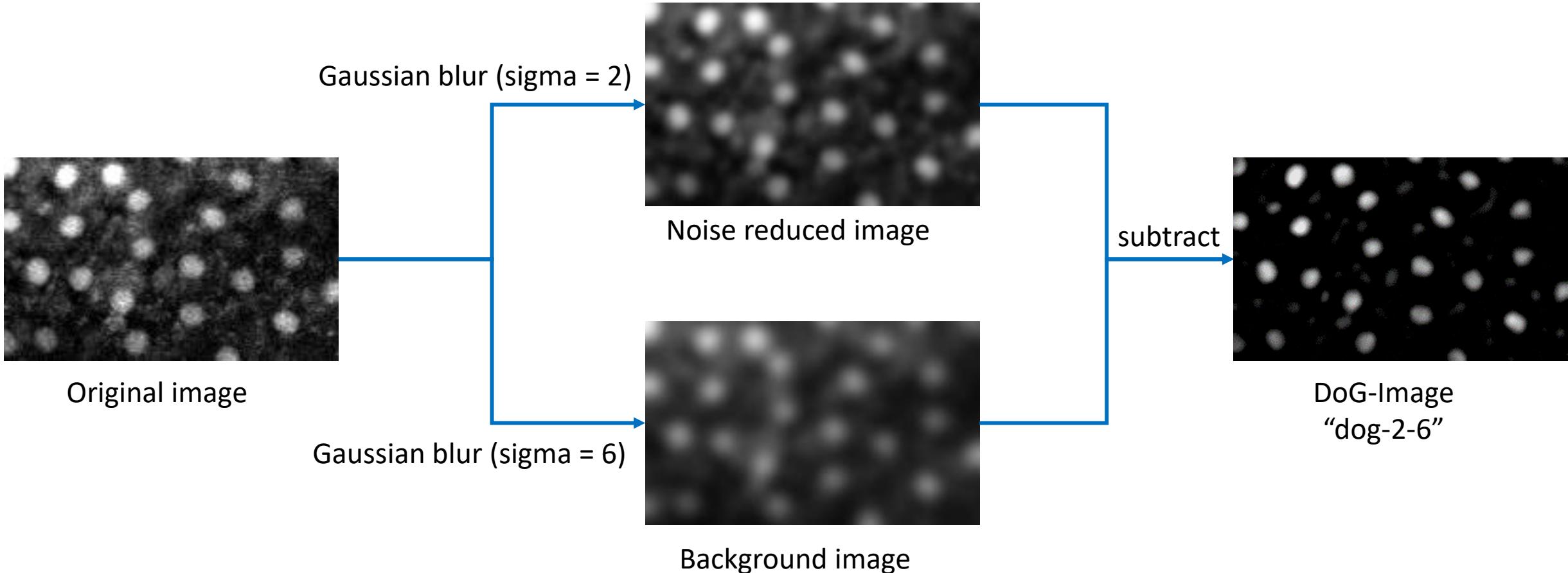


(F) Triangle threshold applied to (D)



# Difference-of-Gaussian (DoG)

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



Adapted from the learning material of Dr. Robert Haase



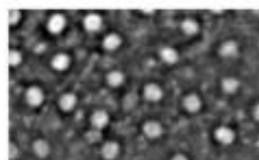
**ICOB Imaging Core**

# Difference-of-Gaussian (DoG)

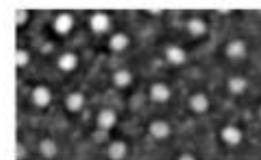
- Example  
DoG  
images



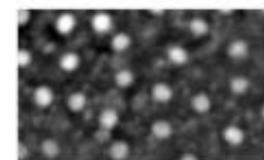
dog-1-1



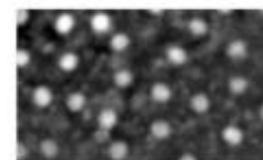
dog-1-4



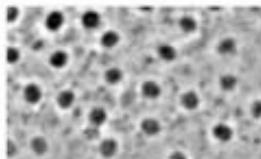
dog-1-7



dog-1-10



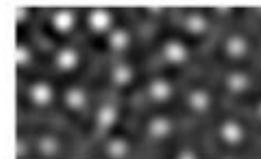
dog-1-13



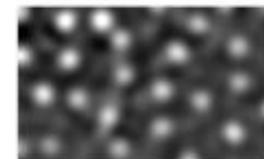
dog-4-1



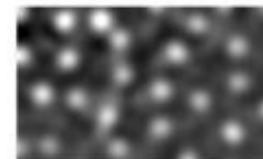
dog-4-4



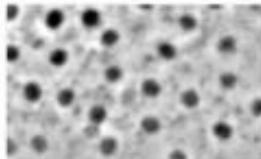
dog-4-7



dog-4-10



dog-4-13



dog-7-1



dog-7-4



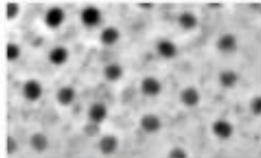
dog-7-7



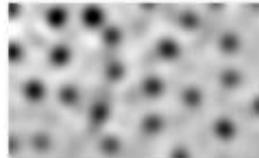
dog-7-10



dog-7-13



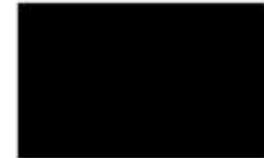
dog-10-1



dog-10-4



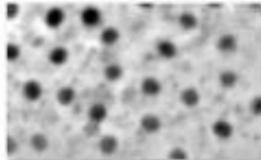
dog-10-7



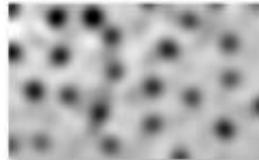
dog-10-10



dog-10-13



dog-13-1



dog-13-4



dog-13-7



dog-13-10



dog-13-13

Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

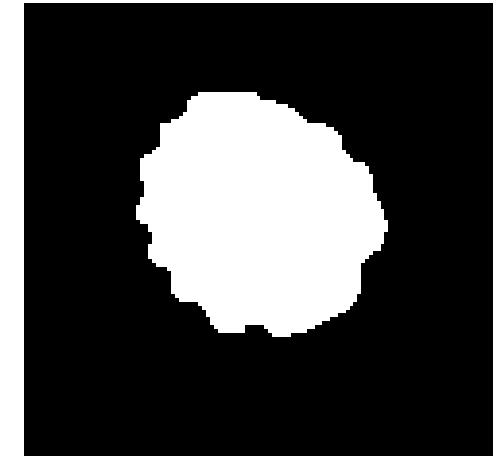
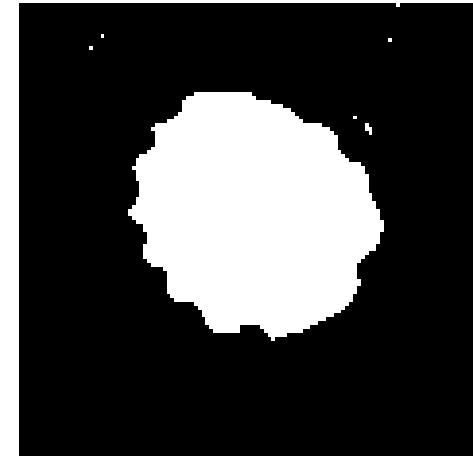
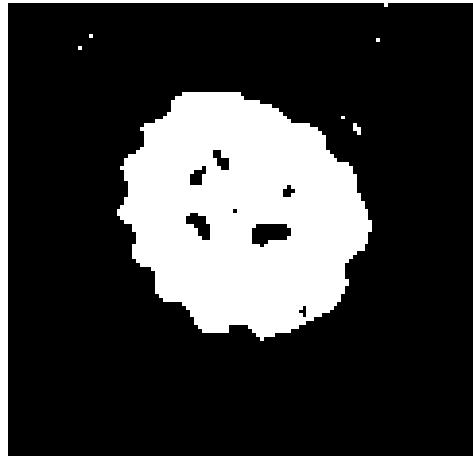
# Refining masks

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

Thresholding

Closing

Opening



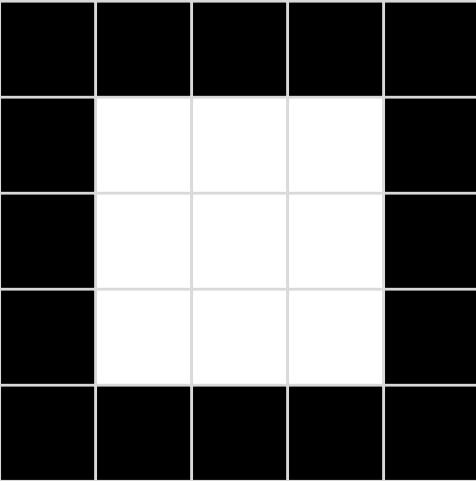
Adapted from the learning material of Dr. Robert Haase



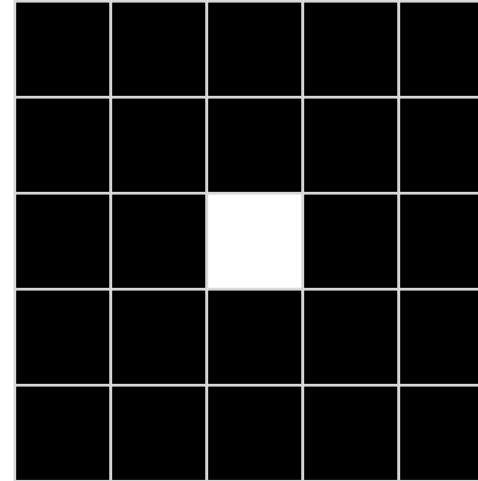
ICOB Imaging Core

# Erosion

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

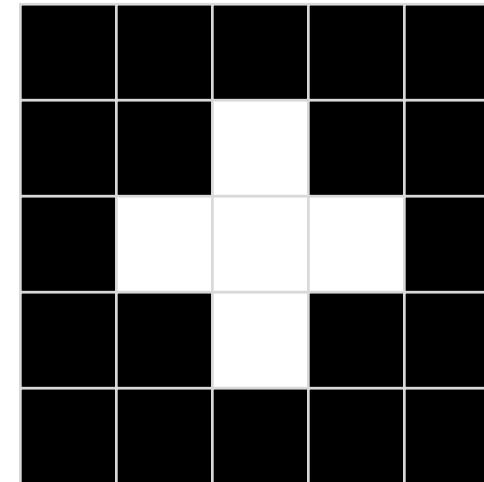
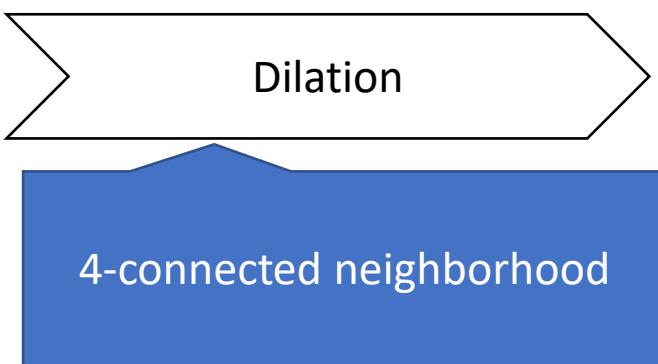
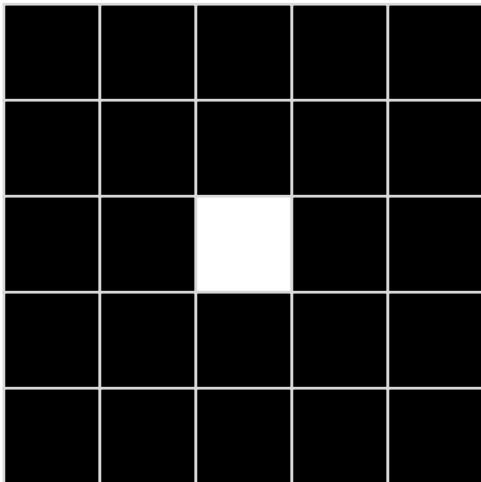
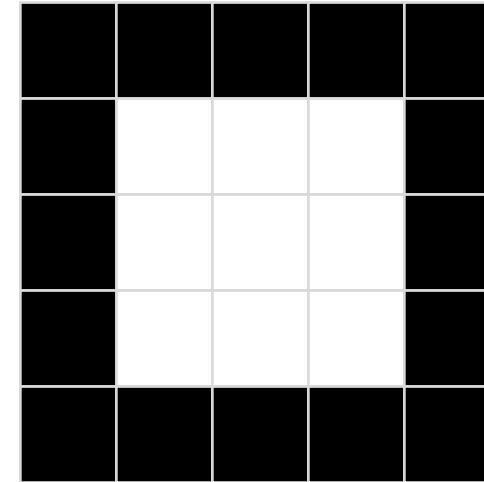
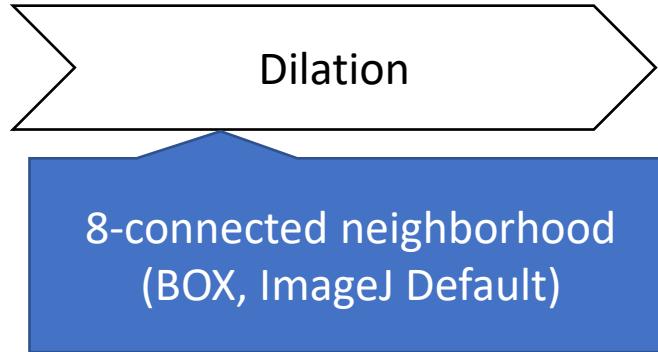
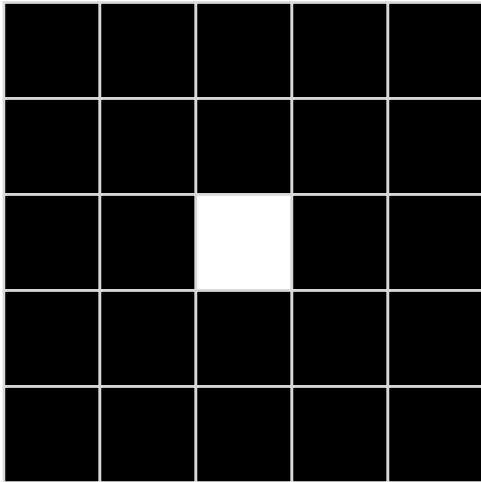


Erosion



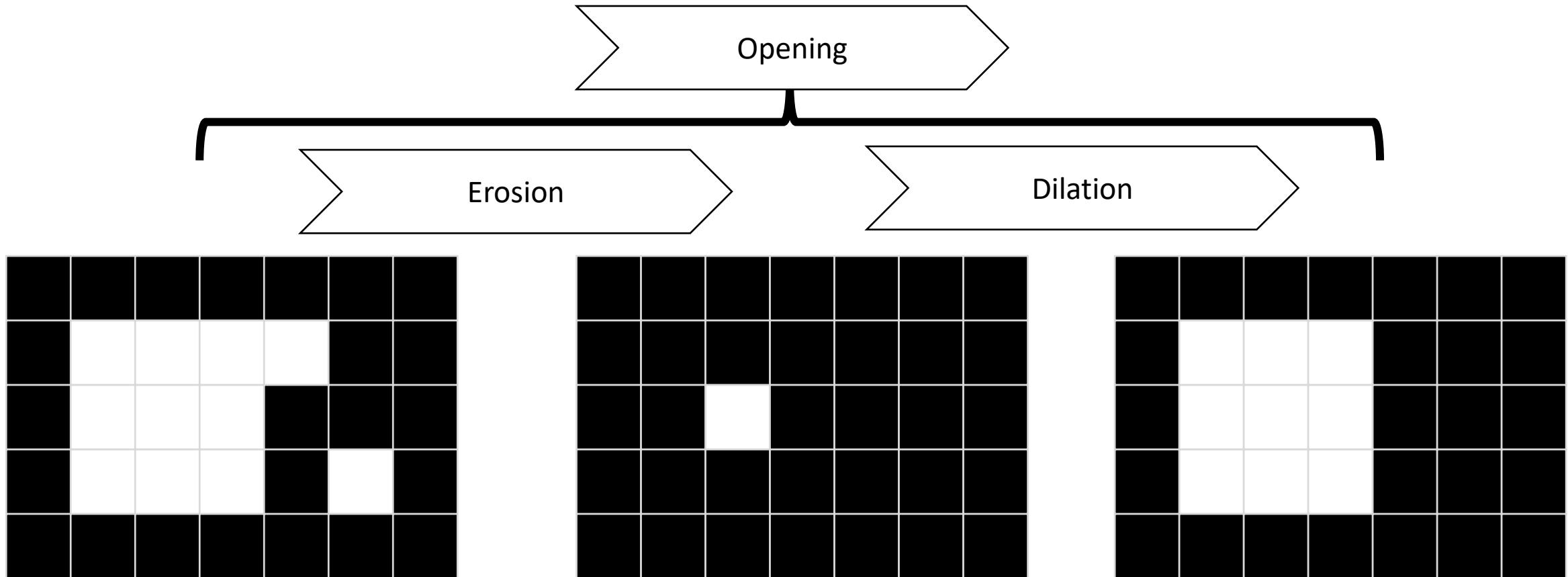
# Dilation

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



# Opening

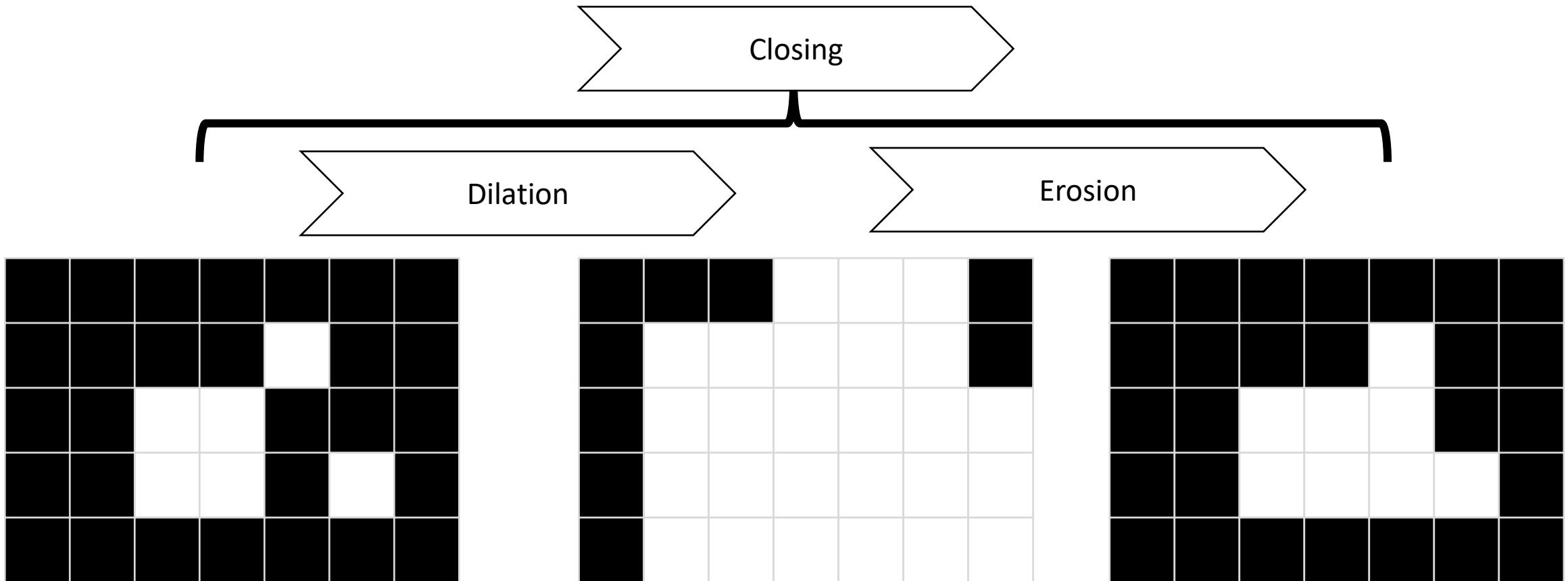
- Erosion and dilation combined allow correcting outlines.



- It can separate white (high intensity) structures that are weakly connected
  - It may erase small white structures

# Closing

- Erosion and dilation combined allow correcting outlines.



- It can connect white (high intensity) structures that are nearby
- It may close small holes inside structures



ICOB Imaging Core

# Chaining erosion and dilation

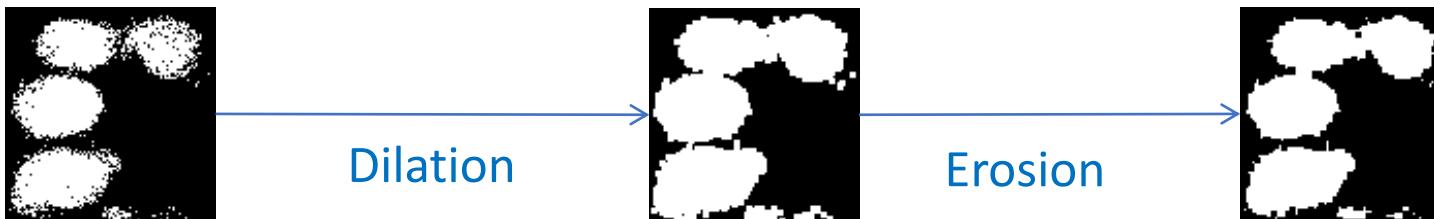
- Erosion: Set all pixels to black which have at least one black neighbor.



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



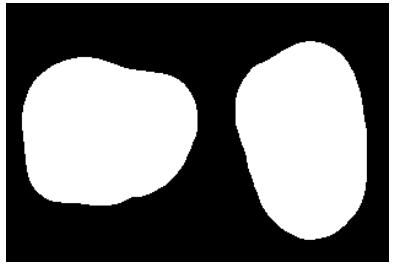
- Closing: Dilation + Erosion



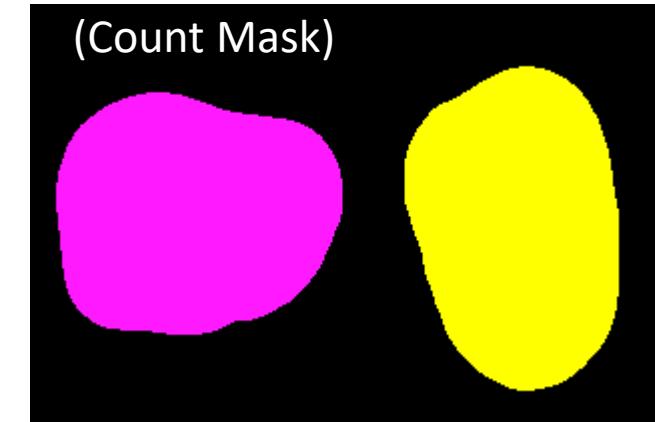
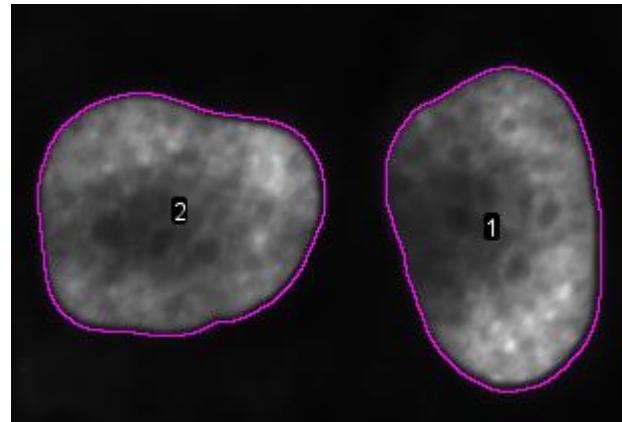
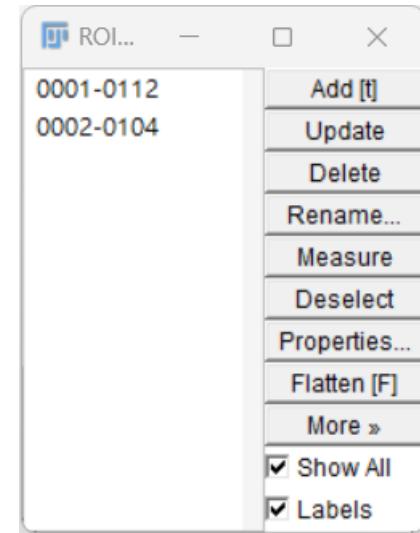
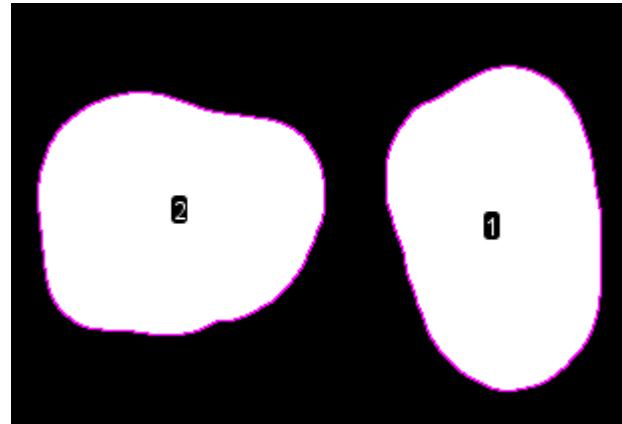
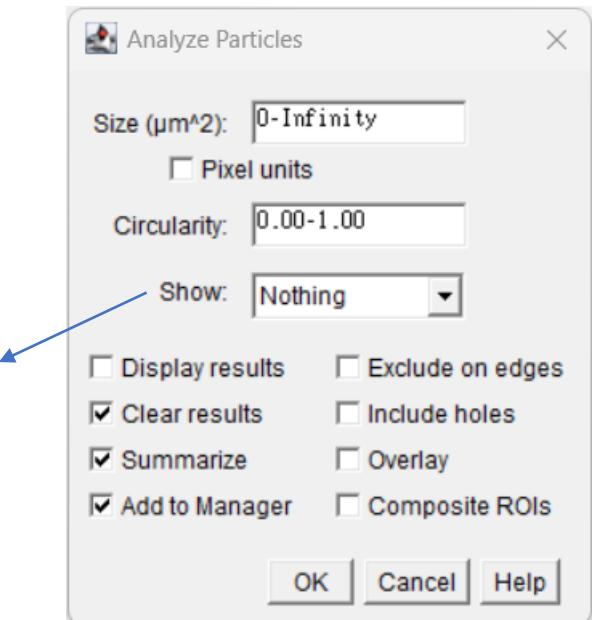
- Opening: Erosion + Dilation

# Analysis Particle: Convert Mask to ROI or Label

- Analyze -> Analyze Particle

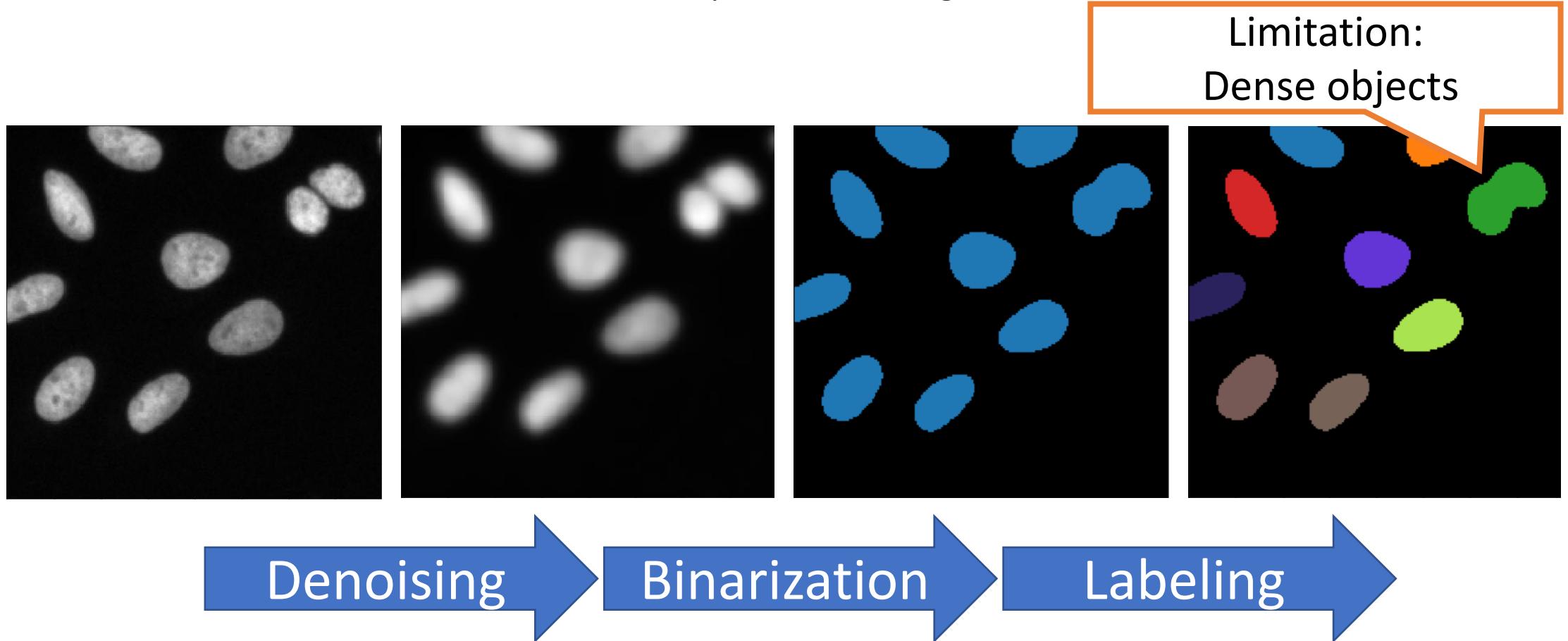


Analyze Particle



# Common image segmentation workflows

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

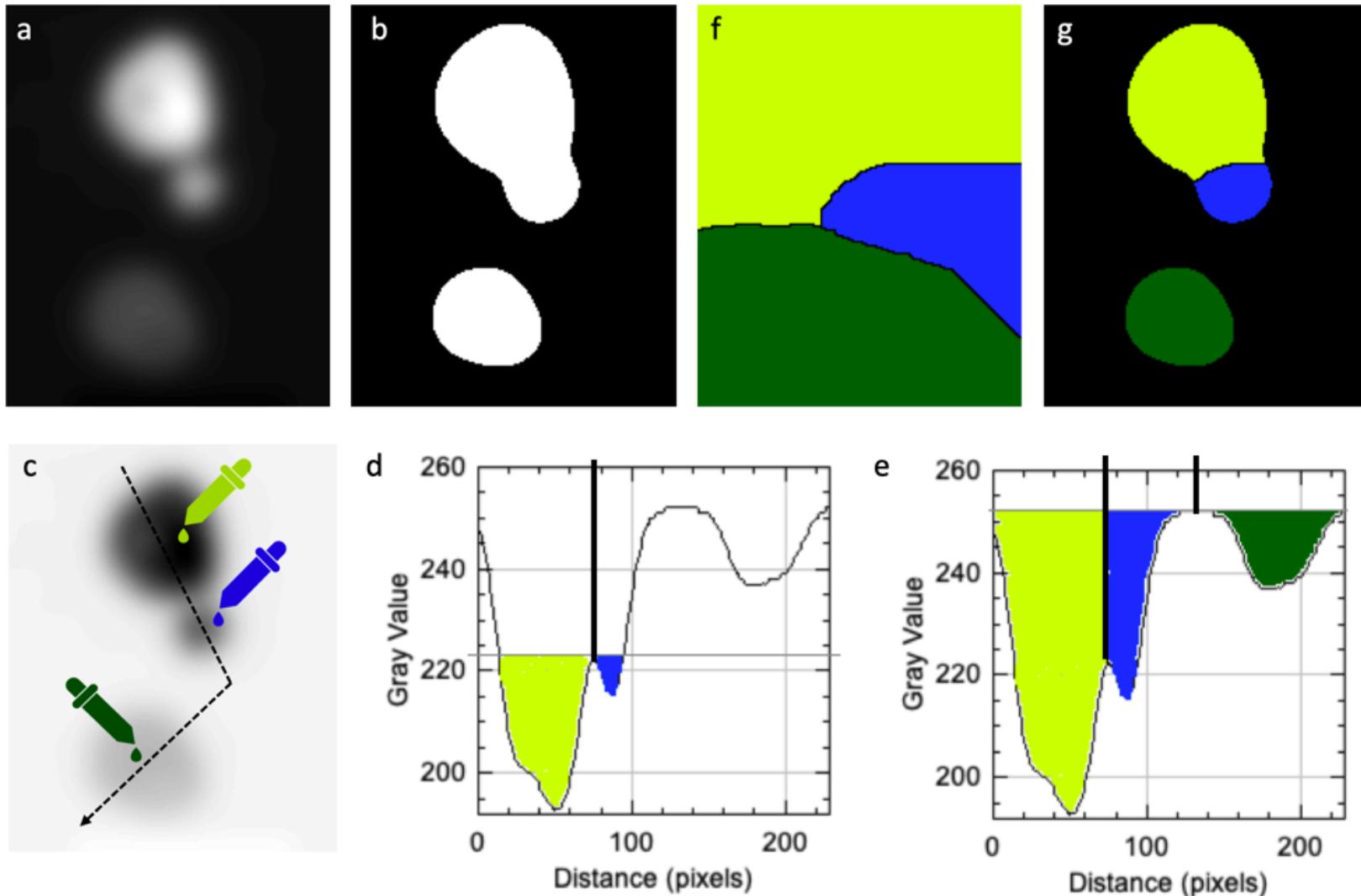


Adapted from the learning material of Dr. Robert Haase



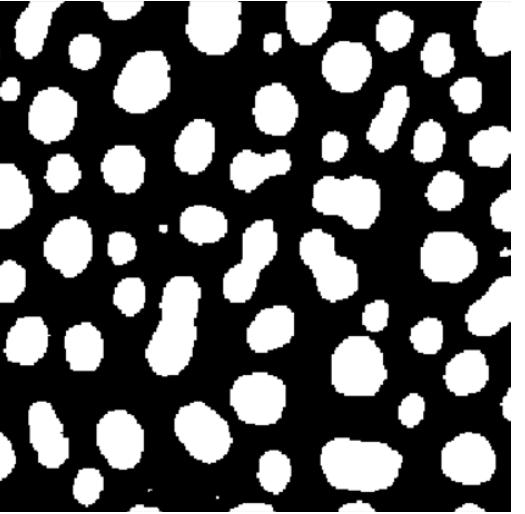
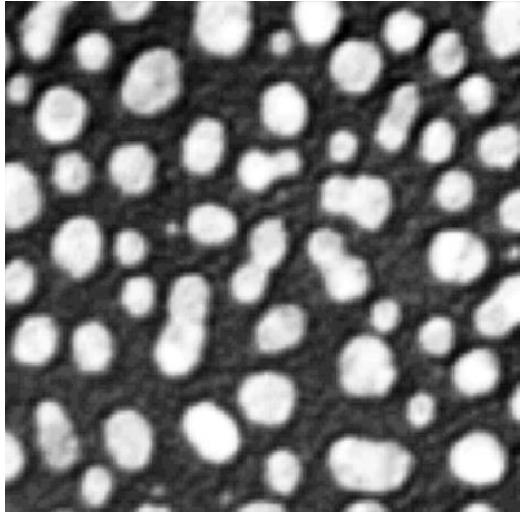
**ICOB Imaging Core**

# Watershed

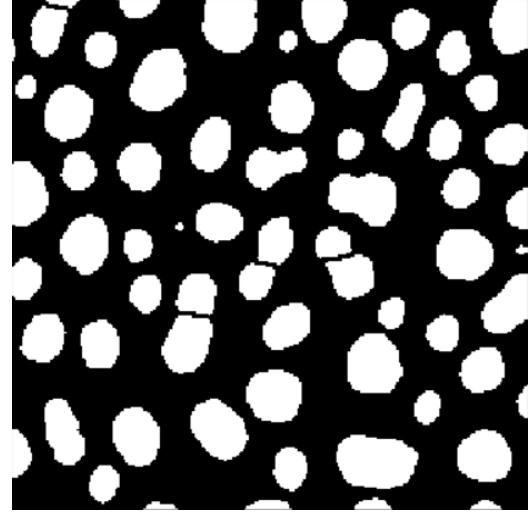


# Watershed use-cases

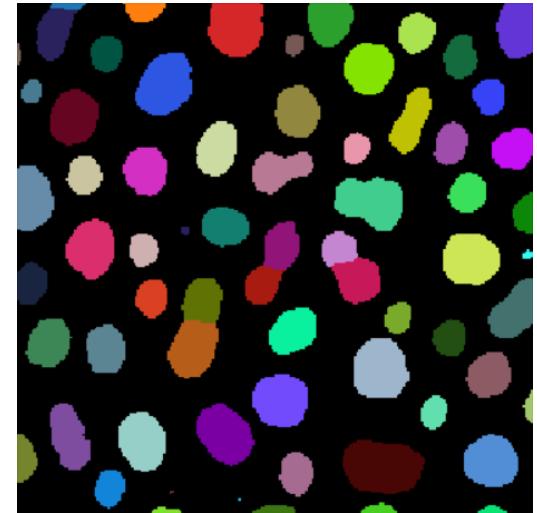
- Split dense objects



*Watershed*



*Binarization*

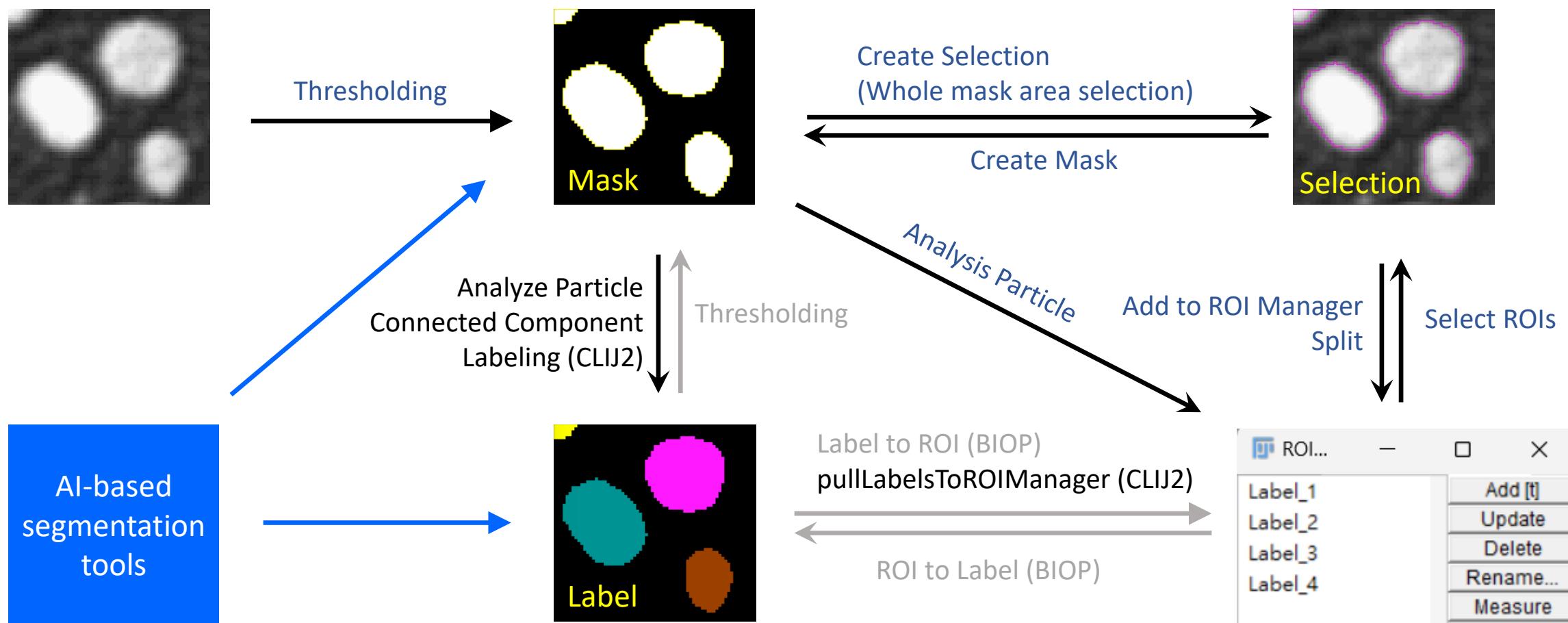


Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

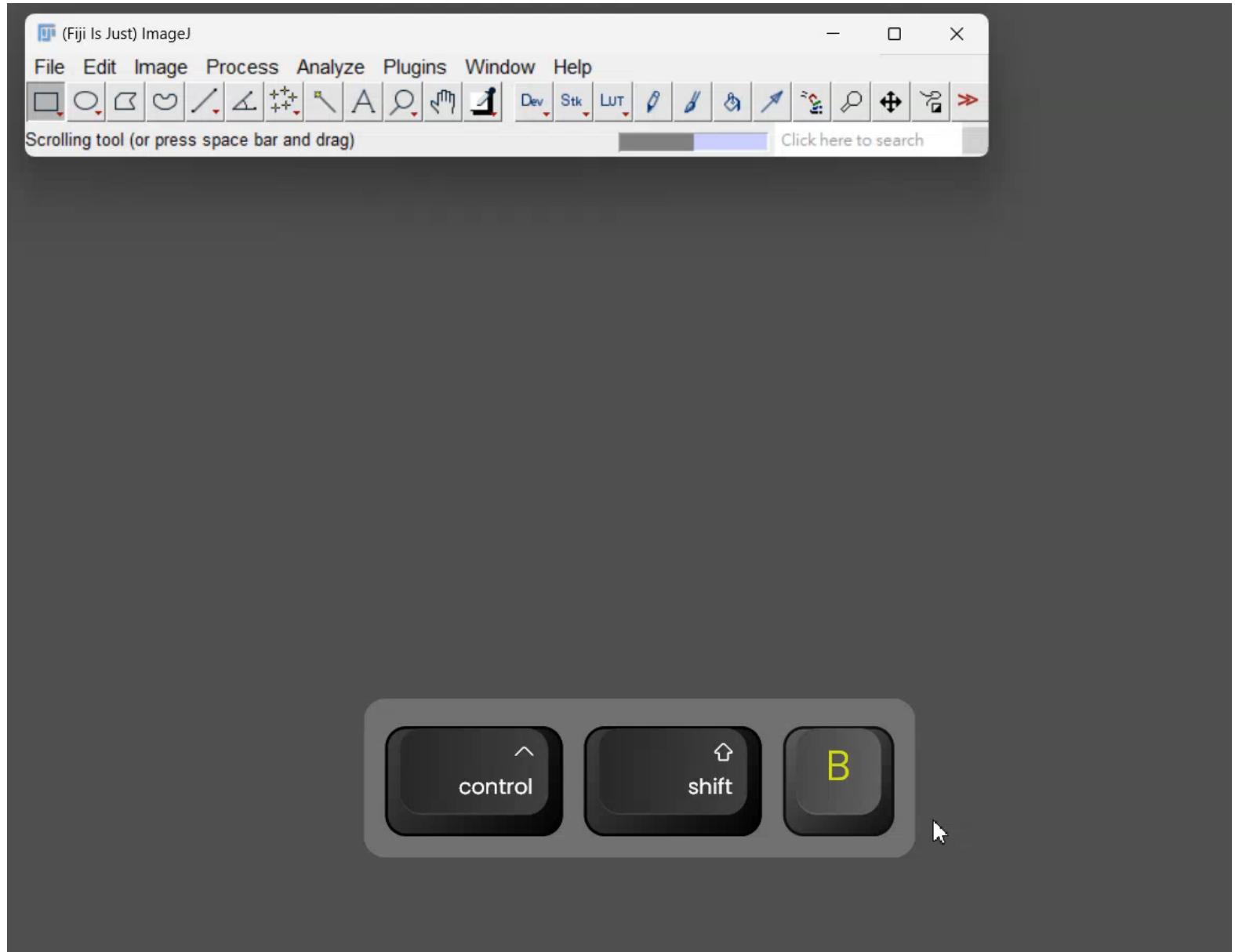
# Segmentation with ROI list, Mask, and Label



This slide is important!

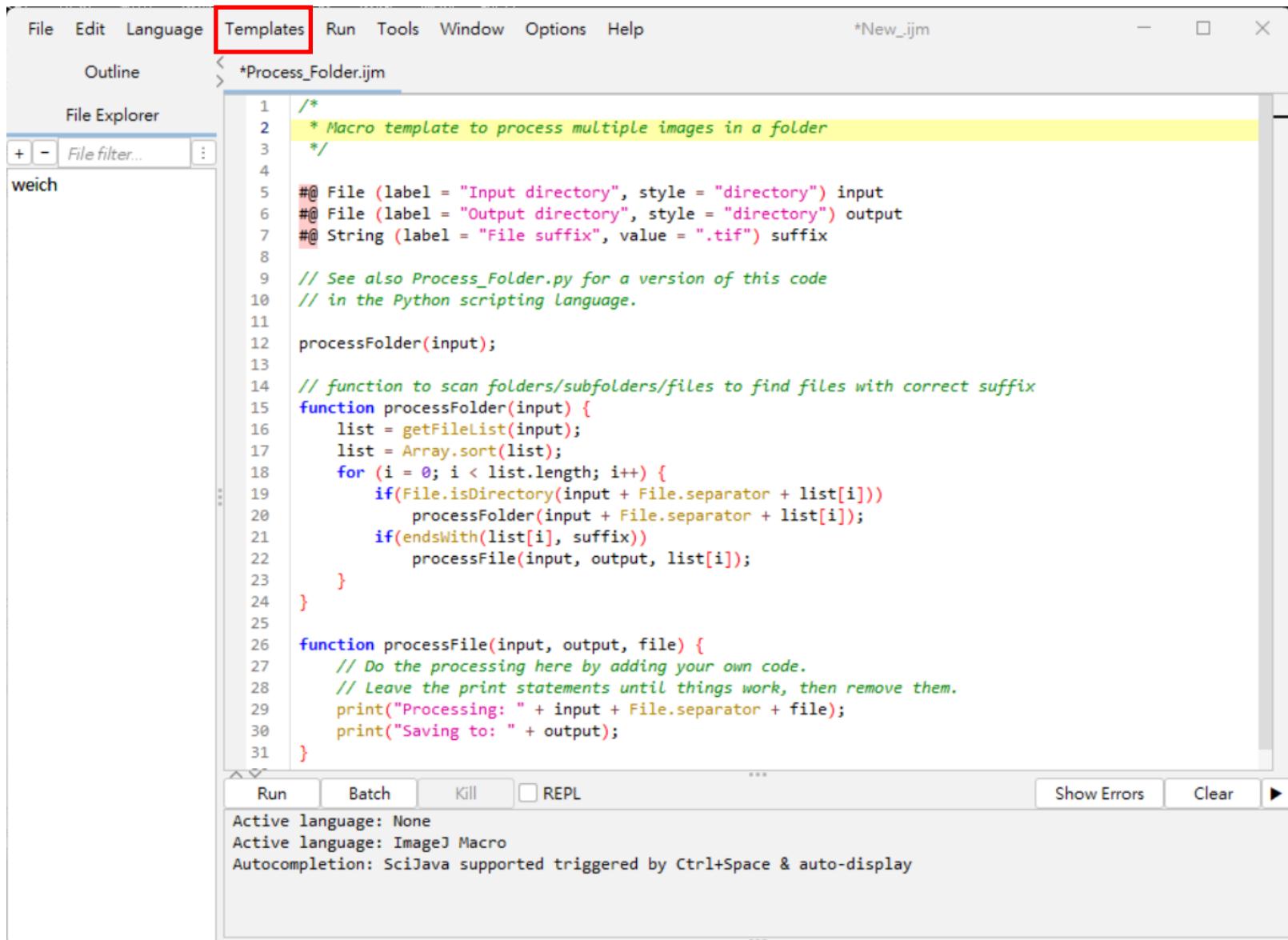


# Typical segmentation workflow in Fiji



# ImageJ macro

- Plugins -> Macros -> Record...
- File -> New -> Script
- Check Templates
- Use “Auto code completion”  
(Thanks to Dr. Robert Haase)
- Print a “Cheat Sheet” as reference
- Try and Error, Debug



The screenshot shows the ImageJ Script Editor window. The menu bar at the top includes File, Edit, Language, **Templates**, Run, Tools, Window, Options, and Help. A red box highlights the 'Templates' option in the menu. The title bar shows the file name \*Process\_Folder.ijm. The main editor area displays the following Java script code:

```
/*
 * Macro template to process multiple images in a folder
 */
#@ File (label = "Input directory", style = "directory") input
#@ File (label = "Output directory", style = "directory") output
#@ String (label = "File suffix", value = ".tif") suffix
// See also Process_Folder.py for a version of this code
// in the Python scripting language.
processFolder(input);

// function to scan folders/subfolders/files to find files with correct suffix
function processFolder(input) {
    list = getFileList(input);
    list = Array.sort(list);
    for (i = 0; i < list.length; i++) {
        if(File.isDirectory(input + File.separator + list[i]))
            processFolder(input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
            processFile(input, output, list[i]);
    }
}

function processFile(input, output, file) {
    // Do the processing here by adding your own code.
    // Leave the print statements until things work, then remove them.
    print("Processing: " + input + File.separator + file);
    print("Saving to: " + output);
}
```

Below the editor, there are buttons for Run, Batch, Kill, and REPL. A status bar at the bottom shows: Active language: None, Active language: ImageJ Macro, and Autocompletion: SciJava supported triggered by Ctrl+Space & auto-display.

# ImageJ macro CHEAT SHEET

## Macro language elements

```
// comments for code documentation  
numericVariable = 5;  
stringVariable = "text value";  
builtInCommand();
```

## Switch between image windows

```
titleOfCurrentImage = getTitle();  
selectWindow(titleOfAnyImage);
```

## Navigation in image stacks

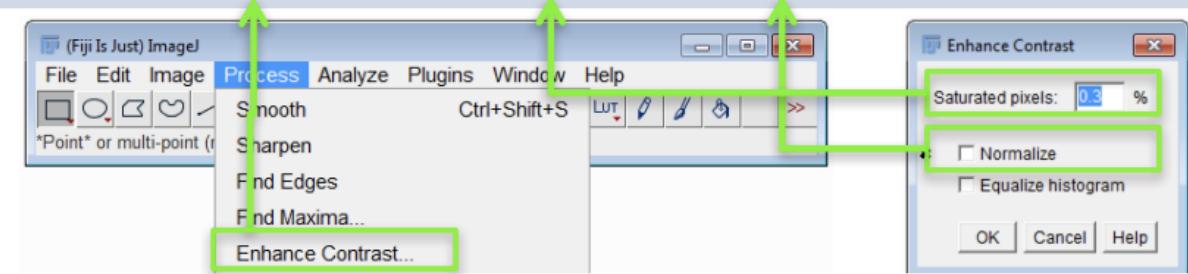
```
Stack.getDimensions(width, height,  
channels, slices, frames);  
  
Stack.setSlice(slice);  
  
Stack.setChannel(channel);  
  
Stack setFrame(frame);  
  
Stack.setDisplayMode("color");  
Stack.setDisplayMode("composite");  
Stack.setDisplayMode("grayscale");
```

## Keep in mind:

- Only one active window!
- One activate channel, slices!
- One ROI list
- One result table!

## Calling any ImageJ/FIJI menu

```
run("Enhance Contrast...", "saturated=0.3 normalize")
```



## Handle image files and folders

```
open(folder+imagefilename);  
close();  
  
fileList = getFileList(folder);  
numFiles = lengthOf(fileList);  
  
for (i=0;i<lengthOf(fileList);i++){  
    file = fileList[i];  
    open(file);  
    // actual image processing...  
    close();  
}
```

## Reading image calibration

```
getPixelSize(unit, pWidth, pHeight);  
getVoxelSize(vWidth, vHeight,  
vDepth, unit);
```

## Result tables

```
run("Set Measurements...", "area  
mean standard min centroid");  
  
corresp.  
to this:  
Area  
Standard deviation  
Min & max gray value  
Mean gray value  
Modal gray value  
Centroid
```

```
run("Analyze Particles...",  
"add clear display");
```



```
roiManager("Measure");
```

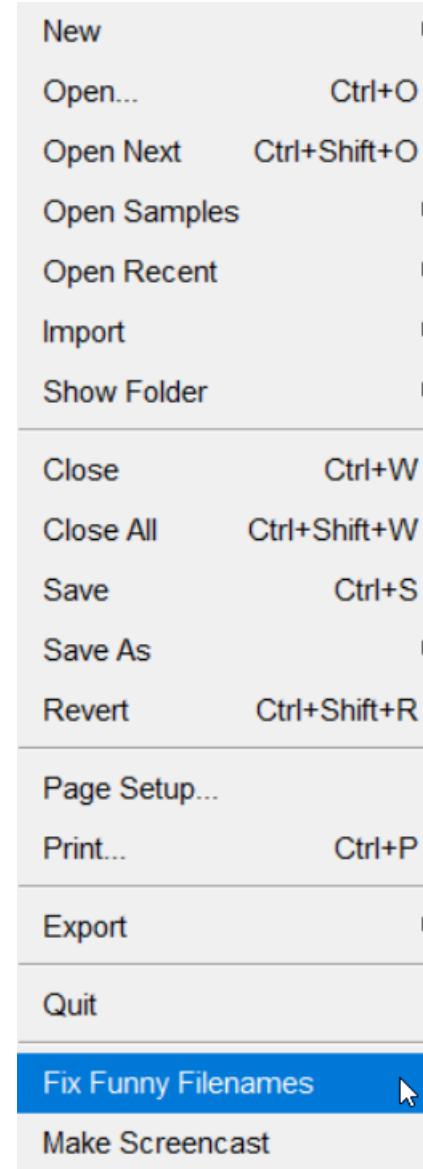
```
rowCount = nResults();  
value = getResult("column title",  
rowNumber);  
setResult("column title",  
rowNumber, newValue);  
saveAs("Results", "myResults.xls");  
run("Clear results");
```

## ROI manager

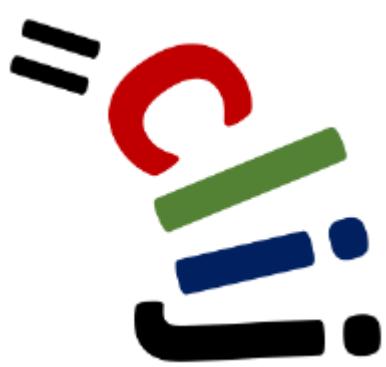
```
roiManager("add");  
roiManager("split");  
roiManager("delete");  
roiManager("reset");  
  
roiManager("measure");  
roiManager("count");  
  
roiManager("open", filename);  
roiManager("save", filename);  
roiManager("save selected", filename);  
  
roiManager("select", index);  
roiManager("select", newArray(index1,  
index2, ...));  
roiManager("deselect");  
  
roiManager("show all");  
roiManager("show all with labels");  
roiManager("show none");  
  
roiManager("and");  
roiManager("combine");
```

# Fix funny filename

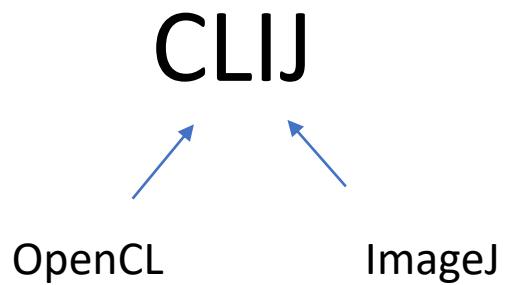
- Rename files with problematic file or folder names automatically  
image 001.tif → image\_001.tif
- Try it if your ImageJ macro can not find you file or folder!



# CLIJ: GPU accelerated image processing for everyone



Integrated GPU-acceleration into ImageJ / Fiji using ImageJ Macro “instead” of OpenCL.



**Dr. Robert Haase**

Clusters Excellence “Physic of Life” (PoL),  
TU Dresden, Germany

<https://haesleinhuepf.github.io/>

# Benchmarking in Fiji

Comparison of processing performance Fiji versus CLIJ: 1-2 orders of magnitude speedup are possible!



vs.



Workstation CPU

2x Intel Xeon Silver  
4110

Laptop CPU

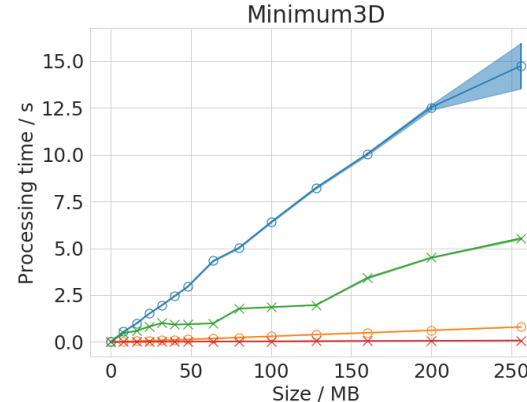
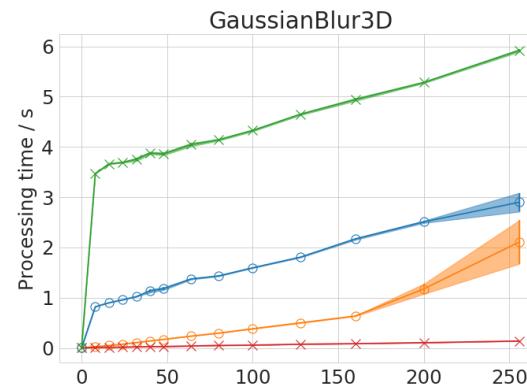
Intel Core i7-8650U

Workstation GPU

Nvidia Quadro  
P6000

Laptop GPU

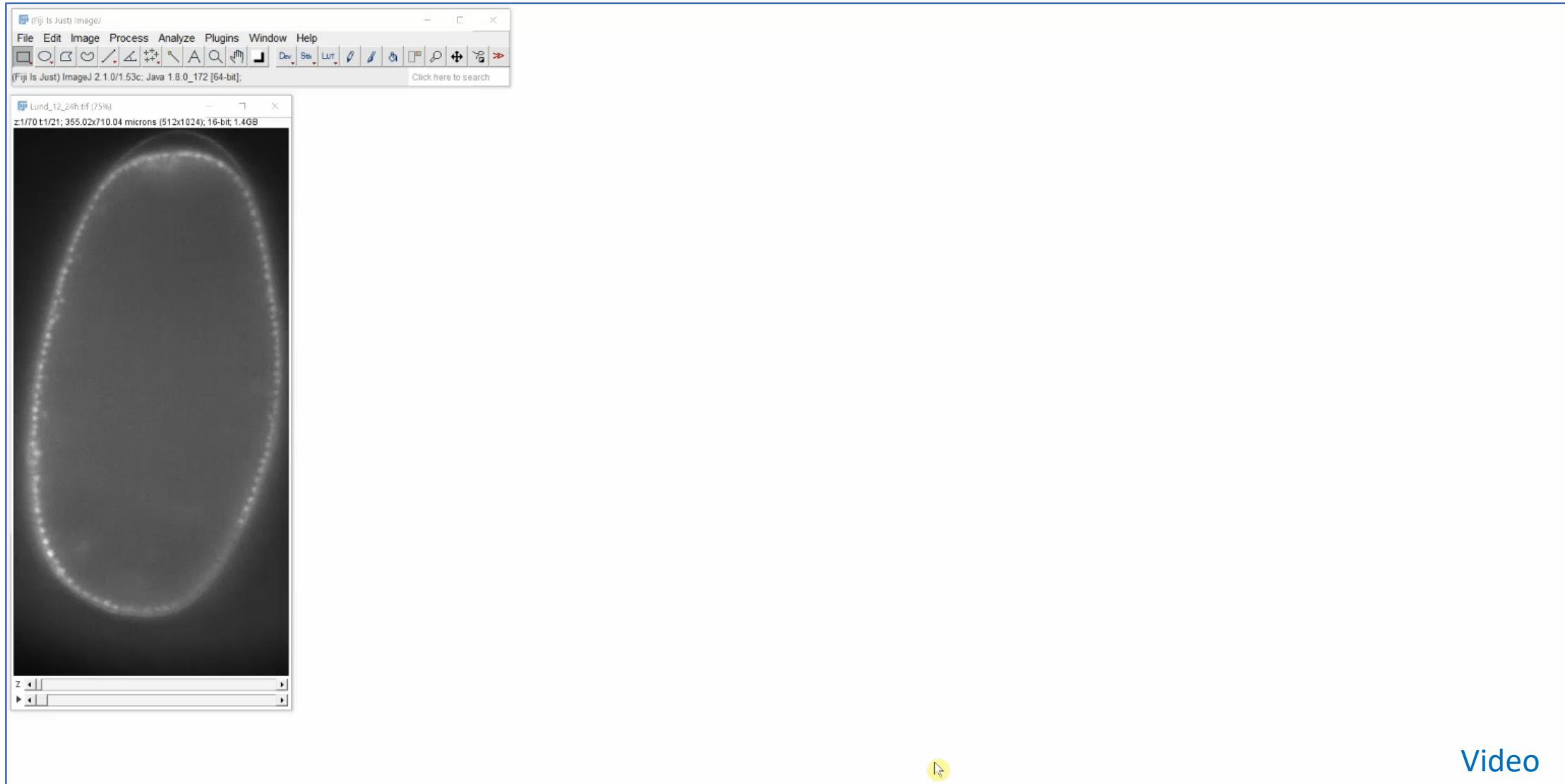
Intel UHD 620 GPU



Speedup factor compared to Laptop CPU

|                     | Laptop GPU | Workstation GPU |
|---------------------|------------|-----------------|
| AddImagesWeighted2D | 3          | 8               |
| AddScalar2D         | 7          | 14              |
| AutoThreshold2D     | 2          | 2               |
| BinaryAnd2D         | 2          | 4               |
| Erode2D             | 11         | 20              |
| FixedThreshold2D    | 2          | 5               |
| Flip2D              | 16         | 37              |
| GaussianBlur2D      | 3          | 9               |
| Mean2D              | 3          | 10              |
| Median2D            | 2          | 35              |
| Minimum2D           | 7          | 22              |
| MultiplyScalar2D    | 10         | 21              |
| Rotate2D            | 3          | 22              |
| AddImagesWeighted3D | 3          | 26              |
| AddScalar3D         | 3          | 23              |
| AutoThreshold3D     | 3          | 5               |
| BinaryAnd3D         | 3          | 24              |
| Erode3D             | 2          | 13              |
| FixedThreshold3D    | 4          | 30              |
| Flip3D              | 15         | 119             |
| GaussianBlur3D      | 6          | 35              |
| MaximumZProjection  | 7          | 46              |
| Mean3D              | 18         | 150             |
| Median3D            | 3          | 43              |
| Minimum3D           | 23         | 188             |
| MultiplyScalar3D    | 4          | 28              |
| RadialReslice       | 14         | 42              |
| Rotate3D            | 0.1        | 2               |

# Image processing in life-sciences (Fiji)



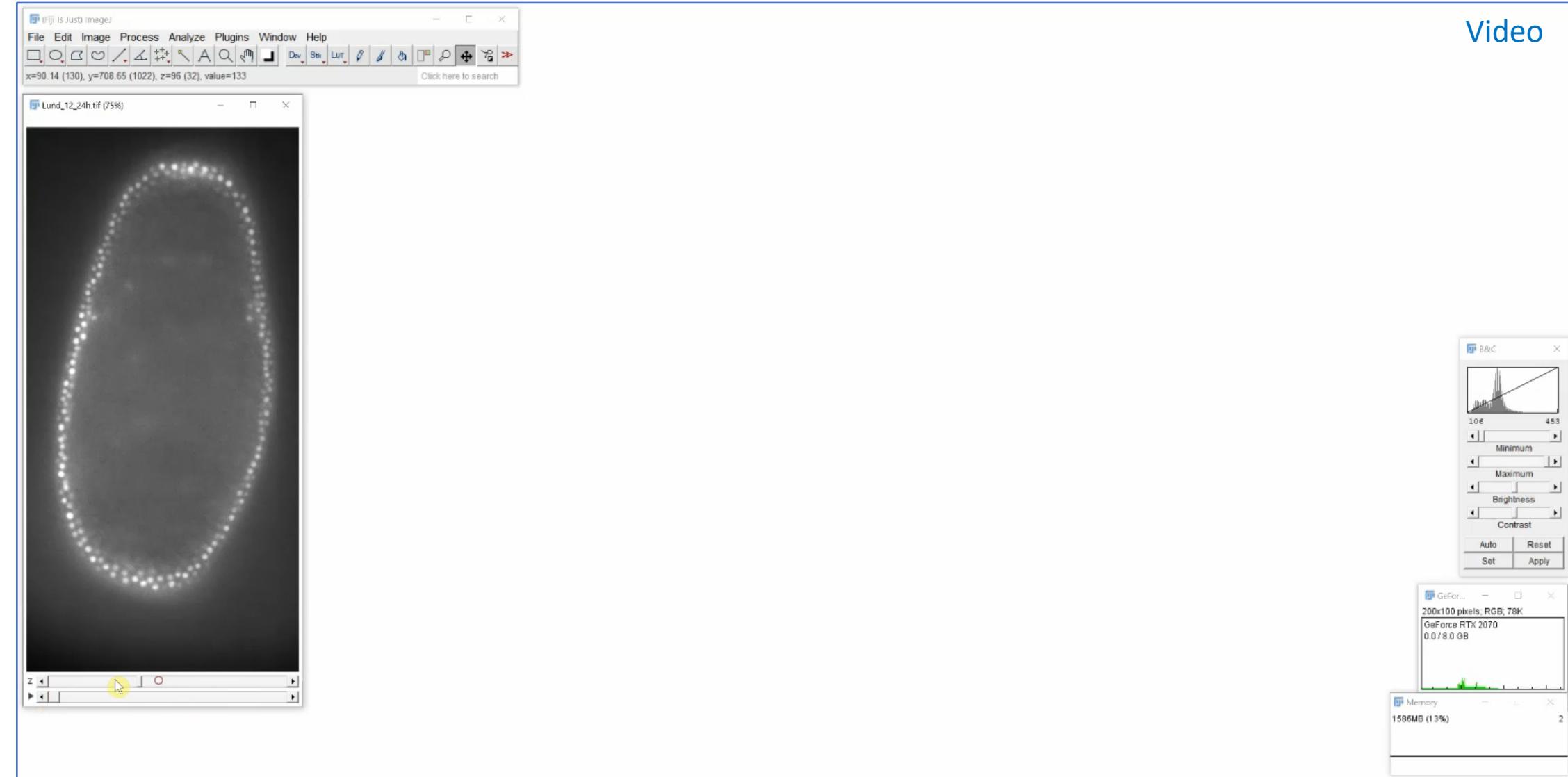
Video

Adapted from the learning material of Dr. Robert Haase  
<https://f1000research.com/slides/10-978>



ICOB Imaging Core

# How image processing is supposed to be



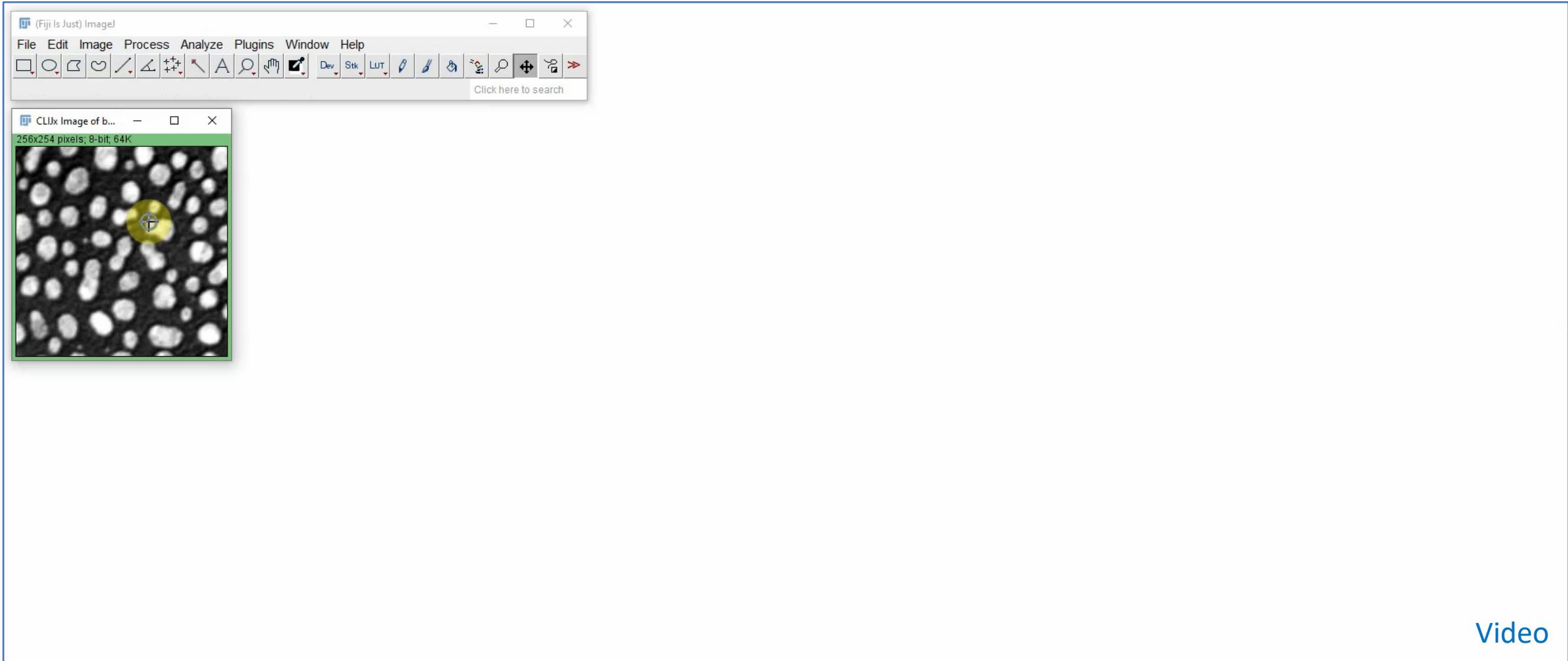
Adapted from the learning material of Dr. Robert Haase  
<https://f1000research.com/slides/10-978>



ICOB Imaging Core

# CLIJ-Assistant

- Interactive workflow design



Video

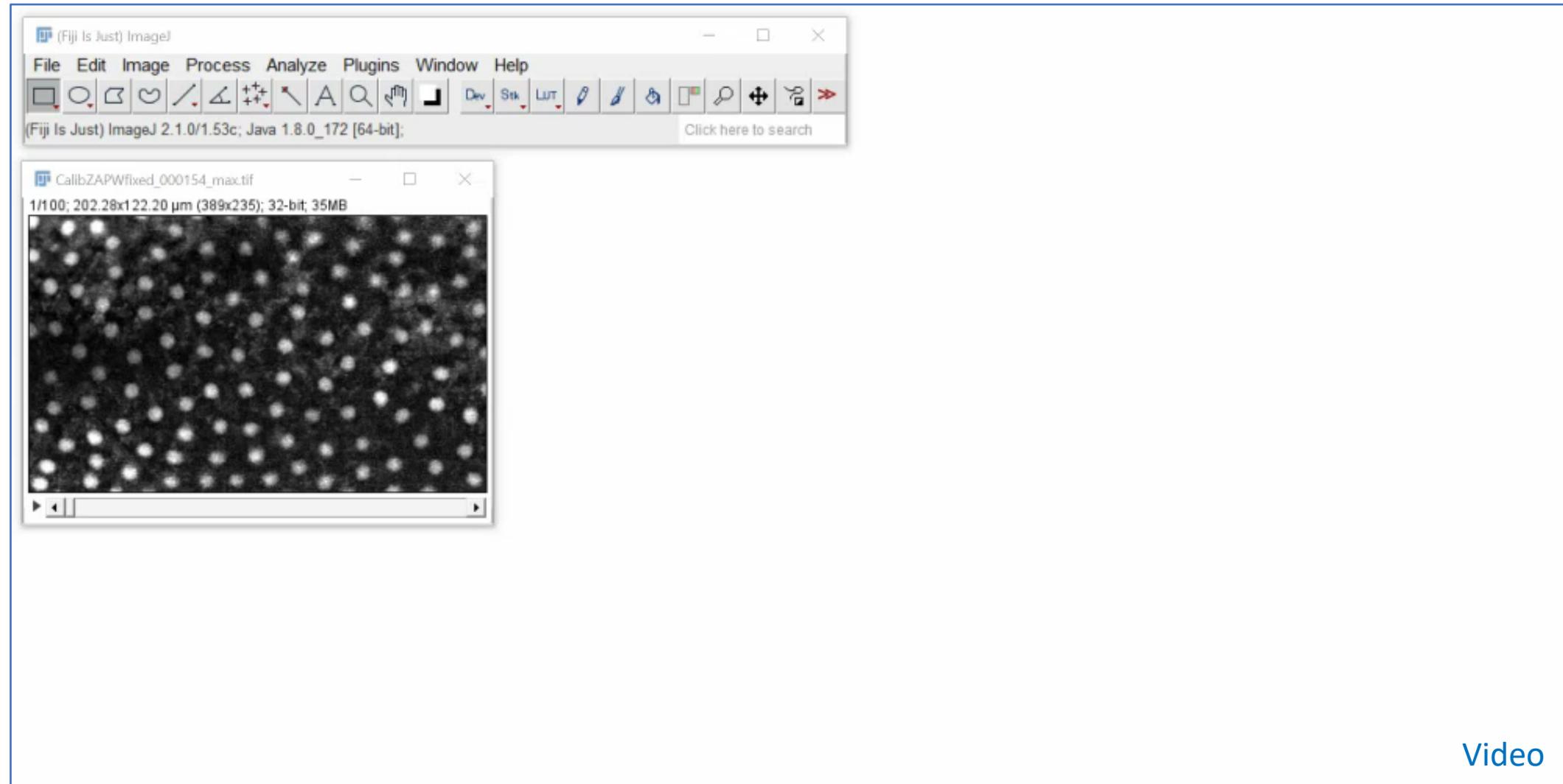
Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

# Timelapse processing

Setup the workflow on one timepoint and check if it's working on other timepoints



Video

Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

# GPU acceleration + code generation

After setting up the workflow, generate code!

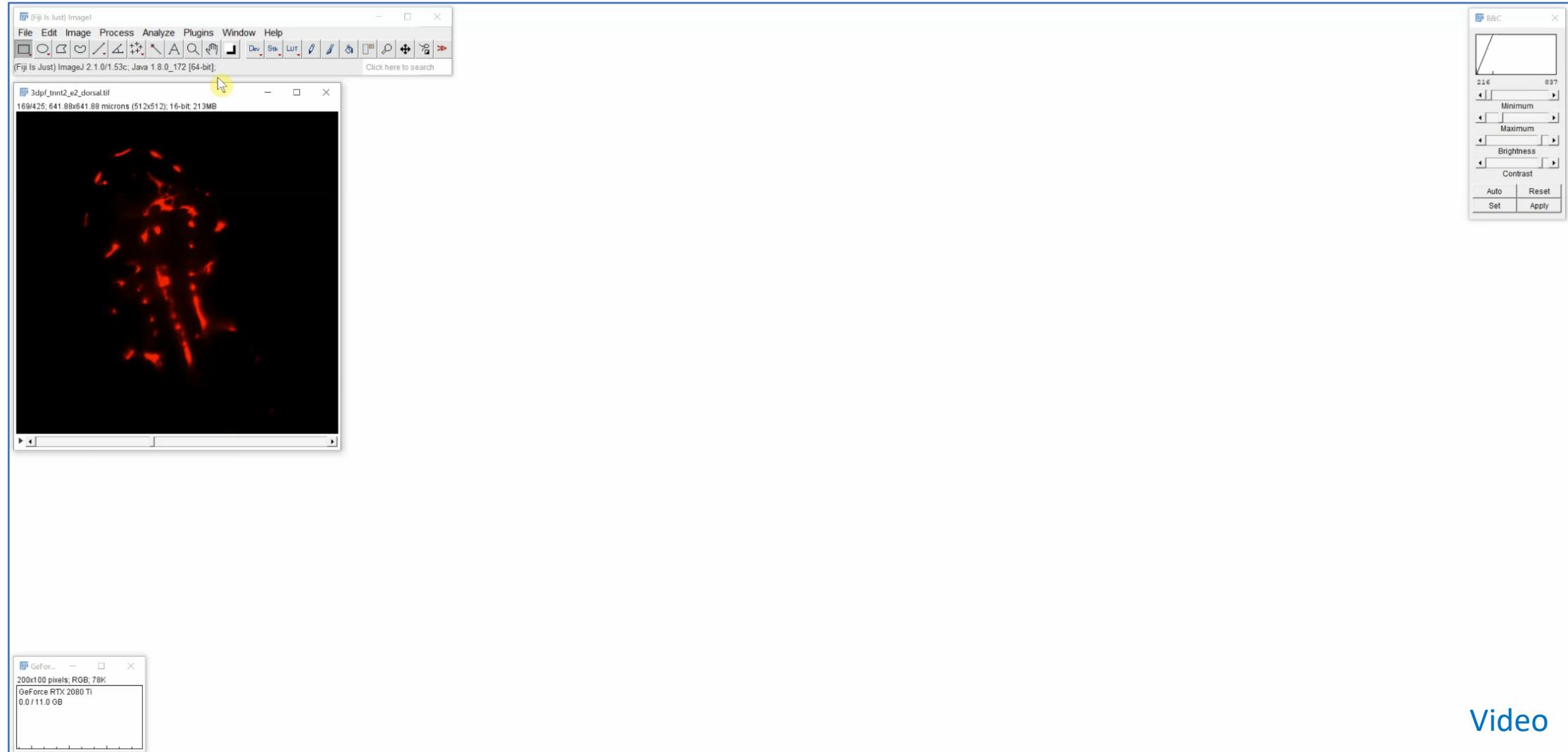


Image data source: Elisabeth Kugler; labs of Tim Chico and Paul Armitage, The University of Sheffield (UK)" <https://zenodo.org/record/4204839#.X8DCRGj7Q2w>  
<https://f1000research.com/slides/10-978>



**ICOB Imaging Core**

# CLIJ2: What every ImageJ Macro script must have

Load data

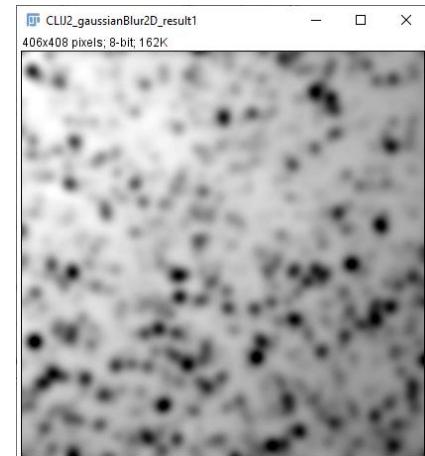
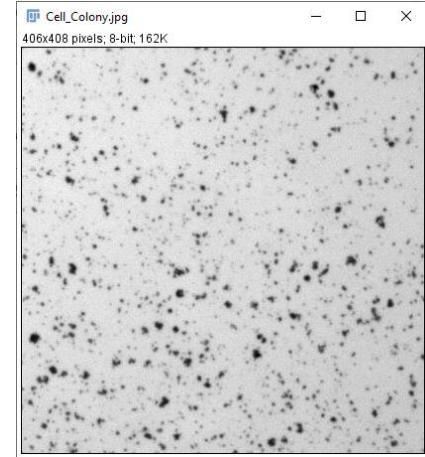
```
1 // Load data
2 run("Cell Colony (31K)");
3
4 // initialize GPU
5 run("CLIJ2 Macro Extensions", "cl_device=");
6 Ext.CLIJ2_clear();
7
8 // push image to GPU
9 input_image = getTitle();
10 Ext.CLIJ2_push(input_image);
11
12 // process image
13 sigma = 5;
14 Ext.CLIJ2_gaussianBlur2D(input_image, result_image, sigma, sigma);
15
16 // optional: release input data
17 Ext.CLIJ2_release(input_image);
18
19 // pull result back from GPU
20 Ext.CLIJ2_pull(result_image);
21
22 // clean up by the end
23 Ext.CLIJ2_clear();
```

Push

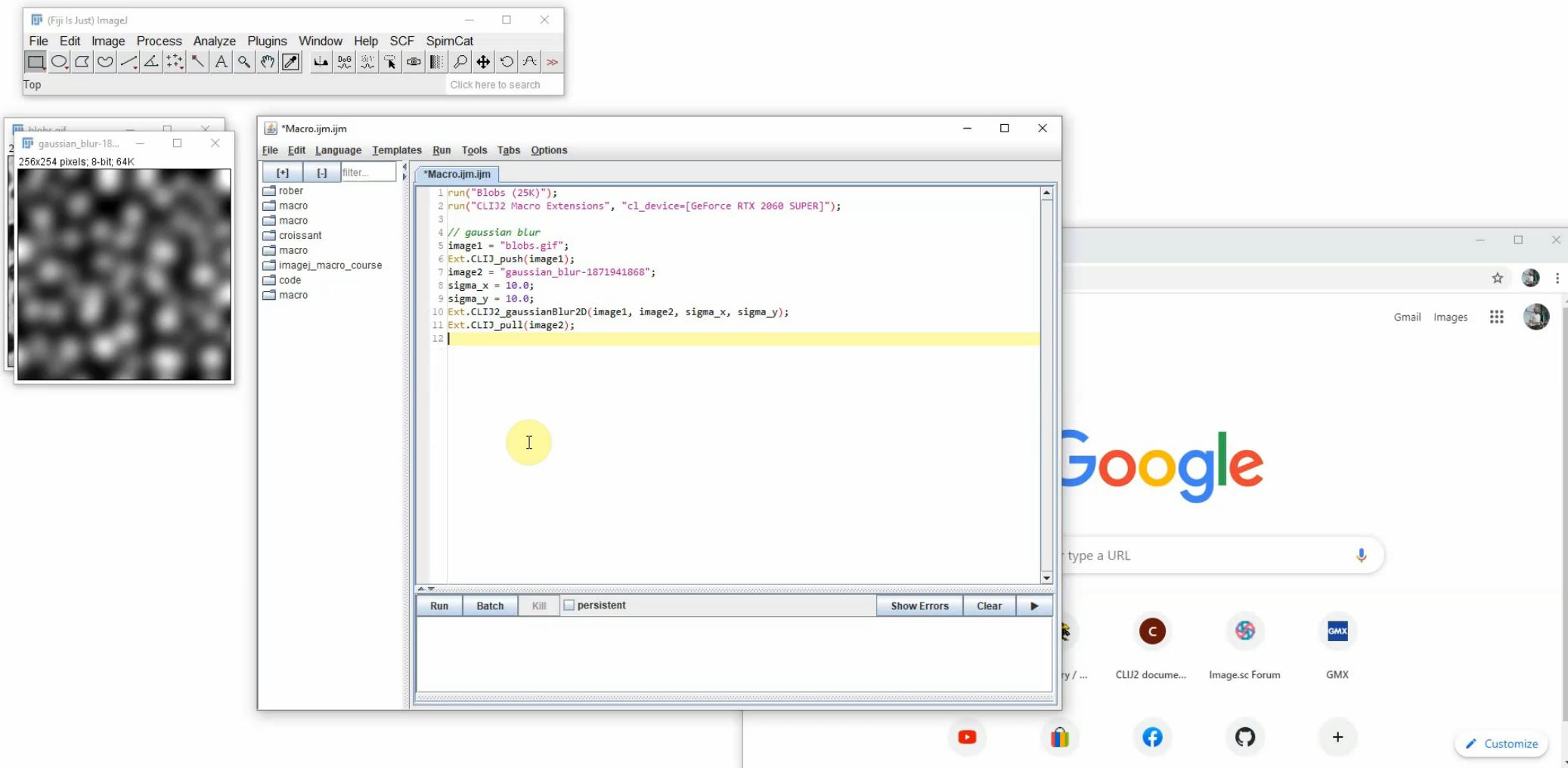
Process images

Pull

Cleanup



# CLIJ2: Macro editing



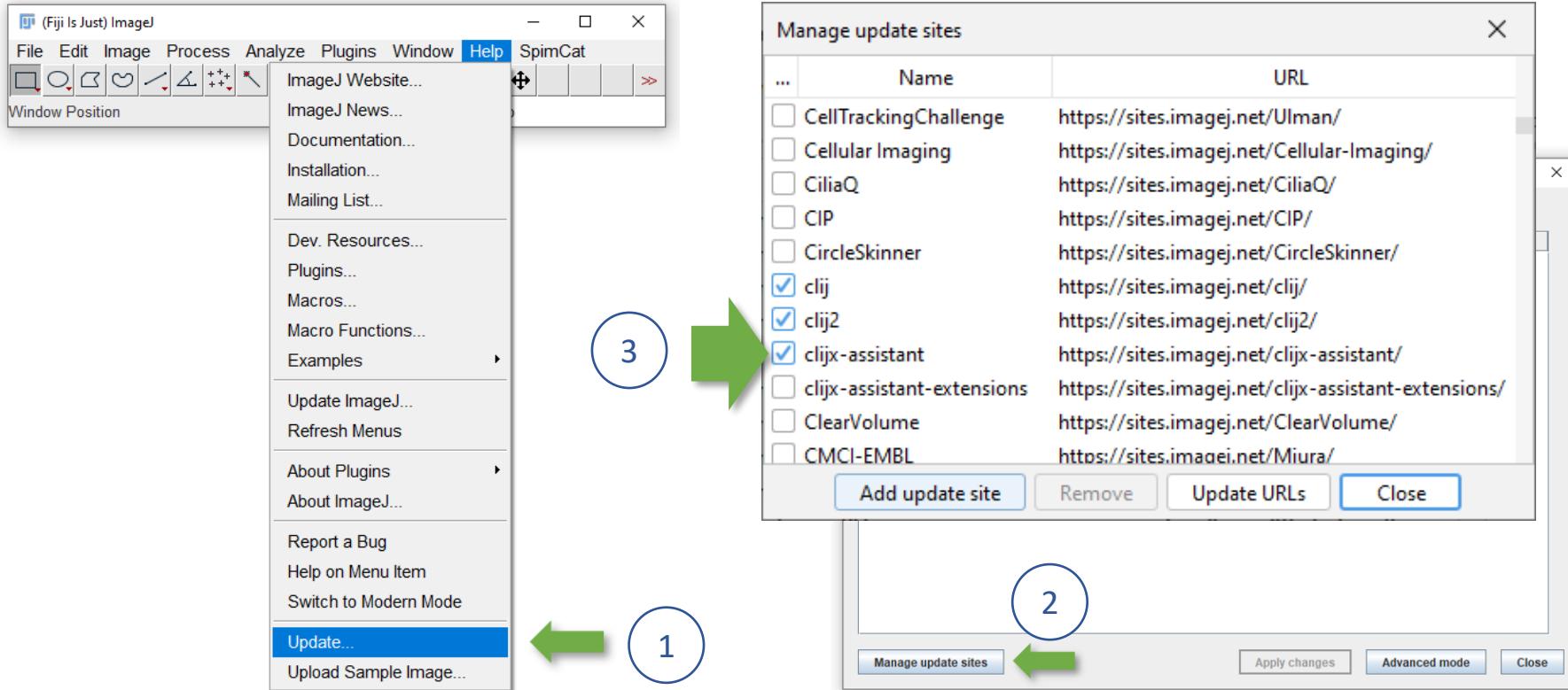
Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

# CLIJ: installation

- Just activate the CLIJ update sites, in menu Help > Update...



installation:

- clij
- clij2
- clijx-assistant

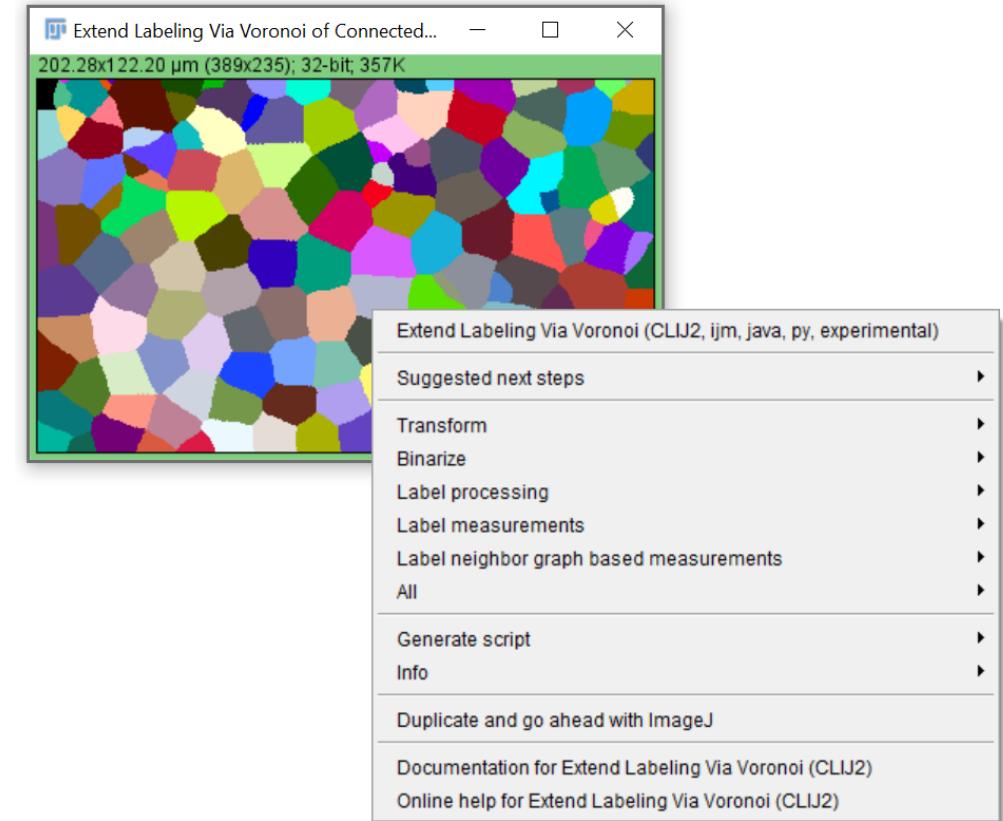
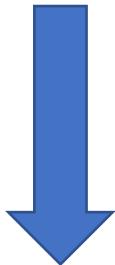
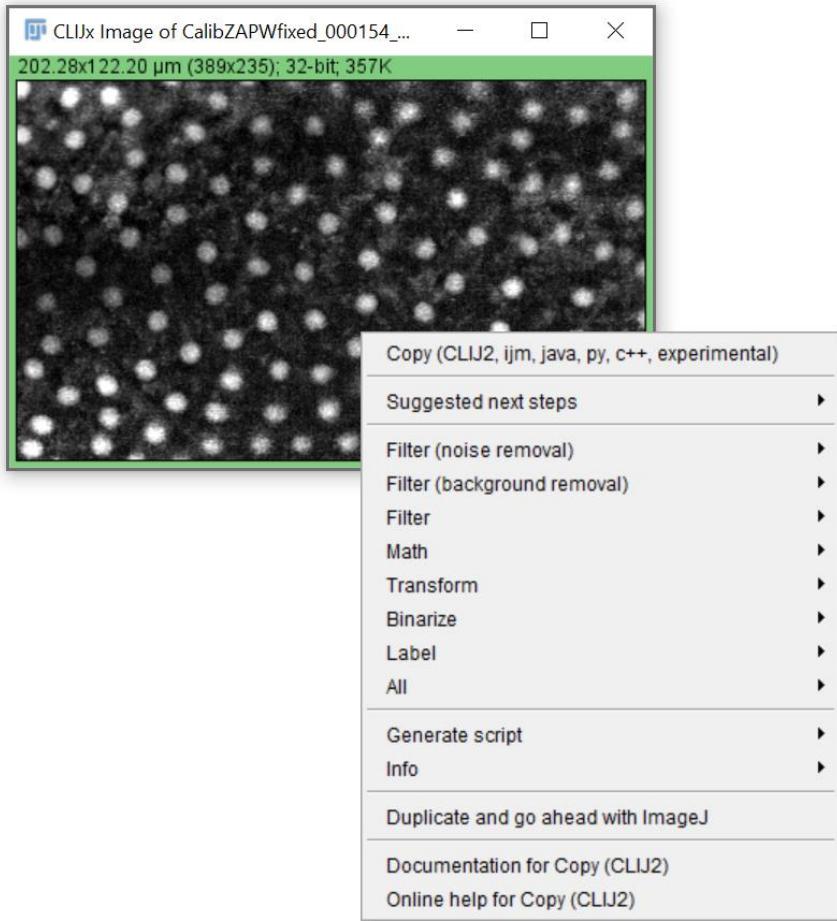
eXperimental

Advanced installation:

- 3D ImageJ Suite
- BoneJ
- IJPB-Plugins
- clijx-assistant-extensions

# Image Data Flow Graph design

The menu order is intentional: From preprocessing to analysis

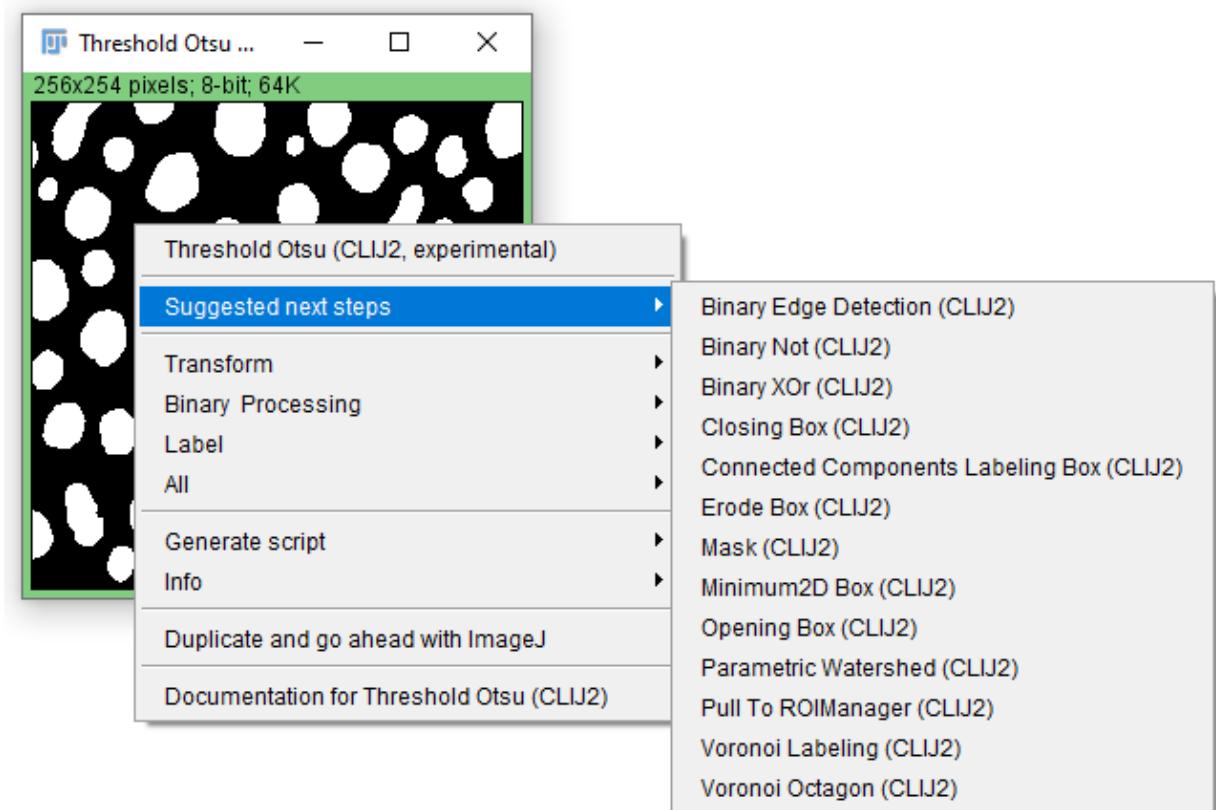
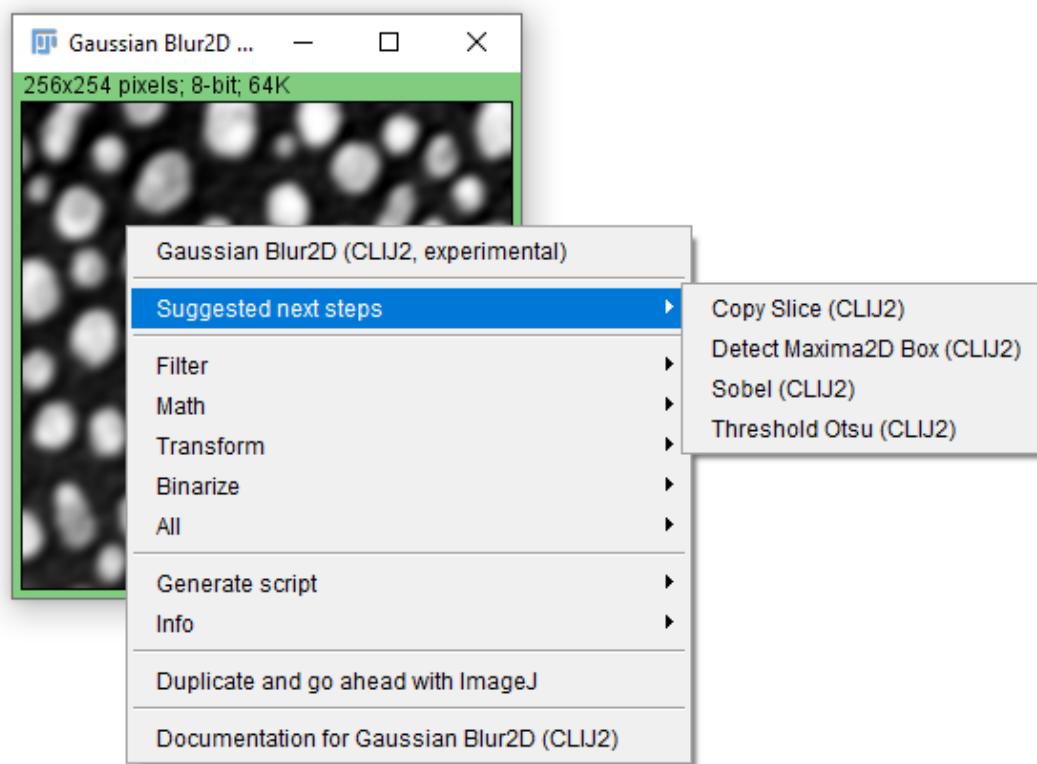


Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

# Expert system: Suggestions



# Fiji search bar

In Fijis search bar result, there is a new category: CLIJ2-assistant which offers operations



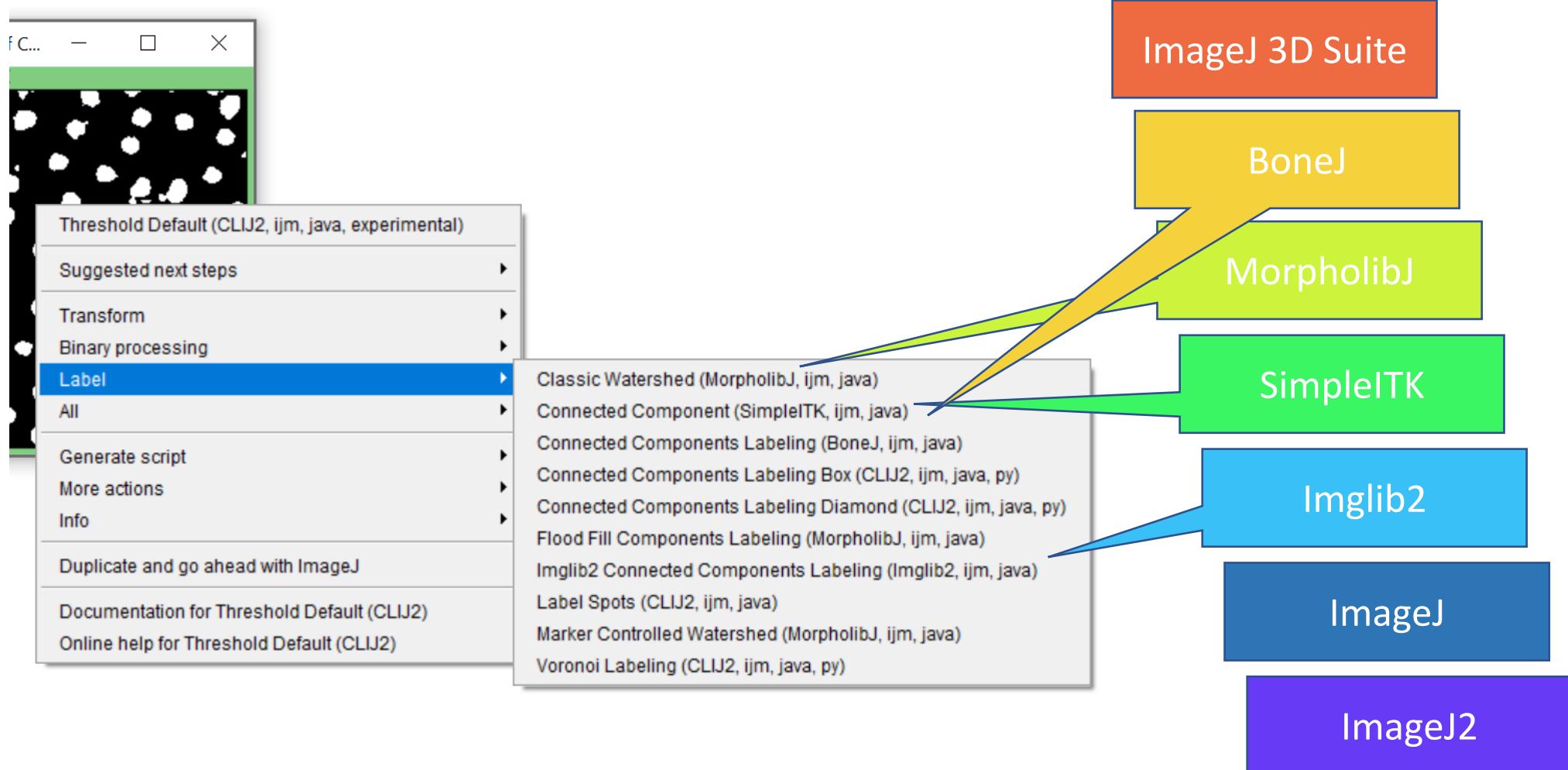
Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

# Extensibility

- Install: <https://clij.github.io/assistant/installation#extensions>



Adapted from the learning material of Dr. Robert Haase

# CLIJ Cheat sheets

Cheat sheets show the most important methods with input and output parameters visually.

| CLIJ2 cheat sheet: ImageJ macro I        |                                       |        |     |  |  |
|--|---------------------------------------|--------|-----|--|--|
| GPU-accelerated image processing in Fiji |                                       |        |     |  |  |
| Operation                                | Parameters                            | Result | Dim | Examples   |  |
| Initialize CLIJ                          | [], HD, GFX or CPU                    |        |     | run("CLIJ Macro Extensions", "-cl_device[]");  |  |
| Push                                     |                                       |        | 2D  | // send current image to GPU<br>input = getTitle();<br>Ext.CLIJ2_push(input);  |  |
| Pull                                     |                                       |        | 2D  | // get result image from GPU back<br>Ext.CLIJ2_pull(output);   |  |
| Create                                   | 1024, 1024, 8                         |        | 3D  | Ext.CLIJ2_create2D("new2D", w, h, bitDepth);<br>Ext.CLIJ2_create3D("new3D", w, h, bitDepth);   |  |
| Convert                                  |                                       |        | 2D  | Ext.CLIJ2_convertToFloat(input, "result_float");<br>Ext.CLIJ2_convertToInt8(input, "result_int8");<br>Ext.CLIJ2_convertToInt16(input, "result_int16"); |  |
| Copy                                     |                                       |        |     | // duplicate source:<br>Ext.CLIJ2_copy(source, result);  |  |
| Copy slice                               | 50                                    |        | 2D  | // put a slice out of a stack<br>Ext.CLIJ2_copySlice(stack, slice, sliceIndex);  |  |
|  |                                       |        | 3D  | // copy a slice out of a stack<br>Ext.CLIJ2_copySlice(slice, stack, sliceIndex);   |  |
| Crop                                     | 20, 20                                |        | 2D  | // crop image<br>Ext.CLIJ2_crop2D("original", "cropped", x, y, width, height);   |  |
| Paste                                    | 9, 9                                  |        | 2D  | // paste image<br>Ext.CLIJ2_paste2D("cropped", "target", x, y);  |  |
| Release                                  |                                       |        |     | // free released memory occupied by an image<br>Ext.CLIJ2_release("image name");   |  |
| Clear                                    |                                       |        |     | Ext.CLIJ2_clear(); // empty GPU memory   |  |
| Spatial Transforms                       |                                       |        |     |  |  |
| Rotate by 90 degrees                     |                                       |        | 2D  | Ext.CLIJ2_rotate90Degrees(input, result);  |  |
| Rotate                                   | 45, true                              |        | 2D  | Ext.CLIJ2_rotate2D(input, result, angle, rotateAroundCenter);  |  |
| Flip                                     | , true, false                         |        | 2D  | Ext.CLIJ2_flipX(input, result, flipX, flipY);<br>Ext.CLIJ2_flipY(input, result, flipX, flipY, flipX);  |  |
| Translate                                | 20, 20                                |        | 2D  | Ext.CLIJ2_translate2D(input, result, shiftX, shiftY);<br>3D  |  |
| Affine transform                         | "center" rotate=45 scale=4.5 "center" |        | 2D  | transf = "center" rotate=45 scale=4.5 "center";<br>Ext.CLIJ2_affineTransform2D(source, result, transf);  |  |
| Deform / warp                            |                                       |        | 2D  | warpImage = Ext.CLIJ2_applyVectorField2D(source, vectorFieldX, vectorFieldY, result);  |  |
| Projections                              |                                       |        | 3D  | Ext.CLIJ2_argmaxProjection(input, result, arg=4);<br>Ext.CLIJ2_stdDeviationProjection(input, result);  |  |

| CLIJ2 cheat sheet: ImageJ macro II       |                |        |     |   |  |
|--|----------------|--------|-----|---|--|
| GPU-accelerated image processing in Fiji |                |        |     |   |  |
| Operation                                | Parameters     | Result | Dim | Examples  |  |
| Gaussian blur                            | , 10, 10       |        | 2D  | Ext.CLIJ2_gaussianBlur2D(input, result, sigmaX, sigmaY);  |  |
| Difference of Gaussian                   | , 2, 2, 20, 20 |        | 2D  | Ext.CLIJ2_differenceOfGaussian2D(input, result, sigmaX, sigmaY, sigmaX2, sigmaY2);  |  |
| Invert                                   |                |        | 2D  | Ext.CLIJ2_invert(input, result);  |  |
| Laplace                                  |                |        | 2D  | Ext.CLIJ2_laplaceBox(input, result);  |  |
| Mean                                     | , 5, 5         |        | 2D  | Ext.CLIJ2_mean2DBox(input, result, radiusX, radiusY);   |  |
| Median                                   | , 5, 5         |        | 2D  | Ext.CLIJ2_median3DBoxBySliceBox(input, result, radiusX, radiusY);   |  |
| Minimum                                  | , 5, 5         |        | 2D  | Ext.CLIJ2_minimum2DBox(input, result, radiusX, radiusY);  |  |
| Maximum                                  | , 5, 5         |        | 2D  | Ext.CLIJ2_maximum2DBox(input, result, radiusX, radiusY);  |  |
| Top-hat                                  | , 25, 25, 0    |        | 2D  | Ext.CLIJ2_topHatBox(input, result, radiusX, radiusY);   |  |
| Logarithm / Exponential                  |                |        | 2D  | Ext.CLIJ2_logarithm(input, result);<br>Ext.CLIJ2_exponential(input, result);  |  |
| Threshold                                | "Otsu", 127 or |        | 2D  | Ext.CLIJ2_threshold(input, binary_result, 127);<br>Ext.CLIJ2_localThreshold(input, binary_result, thresholdImage, binary_result); |  |
| Mask                                     |                |        | 2D  | // mask an image<br>Ext.CLIJ2_mask(input, mask, result);  |  |
| Connected components                     |                |        | 2D  | Ext.CLIJ2_connectedComponentsLabelBox(binary_in, labelmap_out);   |  |
| Label to mask                            | , 4            |        | 2D  | Ext.CLIJ2_labelToMask(labelmap_input, mask_result, label_index);  |  |
| Mask labelled                            | , 4            |        | 2D  | Ext.CLIJ2_maxLabel(input, labelmap, result, label_index);   |  |
| Exclude on edges                         |                |        | 2D  | Ext.CLIJ2_  |  |
| Label spots                              |                |        | 2D  | Ext.CLIJ2_labelSpots();   |  |
| Label Voronoi                            |                |        | 2D  | Ext.CLIJ2_labelVoronoiOctagon(labelled_spots, label_voronoi);   |  |

| CLIJ2 cheat sheet: ImageJ macro III      |                |        |     |  |  |
|--|----------------|--------|-----|--|--|
| GPU-accelerated image processing in Fiji |                |        |     |  |  |
| Operation                                | Parameters     | Result | Dim | Examples   |  |
| Set                                      | , 100          |        | 2D  | Ext.CLIJ2_set(result, pixel_value);<br>Ext.CLIJ2_setColumn(result, column_index, value);   |  |
| Absolute  x                              |                |        | 2D  | Ext.CLIJ2_absolute(input, result);   |  |
| Add / Subtract                           | , or 50        |        | 2D  | Ext.CLIJ2_addImage(input, command, result);<br>Ext.CLIJ2_addImageWithScale(input, result, scale);<br>Ext.CLIJ2_divideImageWithWeighted(in1, in2, result, b);   |  |
| Multiply / Divide                        | , 2            |        | 2D  | Ext.CLIJ2_multiplyImage(input, result);<br>Ext.CLIJ2_multiplyImageWithScale(input, result, n);<br>Ext.CLIJ2_divideImage(dividend, divisor, result);  |  |
| Equal = Not Equal !=                     |                |        | 2D  | Ext.CLIJ2_equal(source1, source2, result);<br>Ext.CLIJ2_notEqual(source1, source2, result);  |  |
| Greater / Smaller                        |                |        | 2D  | Ext.CLIJ2_greater(source1, source2, result);<br>Ext.CLIJ2_smaller(source1, source2, result);<br>Ext.CLIJ2_smallerEqual(source1, source2, result);  |  |
| Equal = Not Equal !=                     |                |        | 2D  | Ext.CLIJ2_equal(source1, source2, result);<br>Ext.CLIJ2_notEqual(source1, source2, result);  |  |
| Binary Images                            |                |        |     |  |  |
| PullBinary                               |                |        | 2D  | Ext.CLIJ2_pullBinary(String image);  |  |
| Draw line / box / sphere                 | 10, 10, 50, 50 |        | 2D  | Ext.CLIJ2_describeLine(x1, y1, x2, y2, thickness, value);<br>Ext.CLIJ2_describeBox(x1, y1, x2, y2, width, height, value);<br>Ext.CLIJ2_describeSphere(x1, y1, z1, r, x2, y2, z2, r, value);<br>// Pixel spent from the line/sphere is untouched; |  |
| Pull regions of interest                 |                |        | 2D  | Ext.CLIJ2_pullingROIbinary(image);<br>Ext.CLIJ2_pullingROIbinary(binary_image);  |  |
| Fill holes                               |                |        | 2D  | Ext.CLIJ2_binaryHoles(result);   |  |
| Not                                      |                |        | 2D  | Ext.CLIJ2_binaryNot(source, result);   |  |
| And / Intersection                       |                |        | 2D  | Ext.CLIJ2_binaryAnd(operand1, operand2, result);<br>Ext.CLIJ2_binaryIntersection(operand1, op2, result);   |  |
| Or / Union                               |                |        | 2D  | Ext.CLIJ2_binaryOr(operand1, operand2, result);<br>Ext.CLIJ2_binaryUnion(operand1, operand2, result);  |  |
| Xor                                      |                |        | 2D  | Ext.CLIJ2_binaryXor(operand1, operand2, result);   |  |
| Dilate/ Erode                            |                |        | 2D  | Ext.CLIJ2_dilateBox2D(result);<br>Ext.CLIJ2_dilateSphere(result);<br>Ext.CLIJ2_erodeSphere(result);  |  |

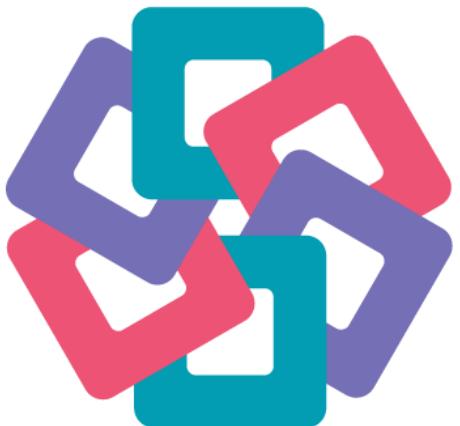
| CLIJ2 cheat sheet: ImageJ macro IV  |                        |        |                           |  |  |  |  |  |
|---|------------------------|--------|---------------------------|--|--|--|--|--|
| GPU-accelerated image processing in Fiji  |                        |        |                           |  |  |  |  |  |
| Operation   | Parameters             | Result | Dim                       | Examples   |  |  |  |  |
| Spots to point lists  |                        |        | 2D                        | Ext.CLIJ2_spotsToPointList(binary_spots, pointlist);<br>Ext.CLIJ2_spotsToPointList(labelled_spots, pointlist);             |  |  |  |  |
| Generate distance matrix  |                        |        | 2D                        | Ext.CLIJ2_generateDistanceMatrix(pointlist1, pointlist2, distance_matrix);   |  |  |  |  |
| Generate touch matrix   |                        |        | 2D                        | Ext.CLIJ2_touchMatrixToMesh(touch_matrix, mesh);   |  |  |  |  |
| Distance matrix to mesh   | , 2,5                  |        | 2D                        | Ext.CLIJ2_distanceMatrixToMesh(pointlist, distance_matrix, mesh, max_distance);  |  |  |  |  |
| Mean of touching neighbors  |                        |        | 2D                        | Ext.CLIJ2_meanOfTouchingNeighbors(values, touch_matrix, mean_value);   |  |  |  |  |
| Count touching neighbors  |                        |        | 2D                        | Ext.countTouchingNeighbors(touch_matrix, count_rects);   |  |  |  |  |
| Working with arrays and tables  |                        |        |                           |  |  |  |  |  |
| Statistics  |                        |        | 2D                        | Ext.CLIJ2_statisticsOfBackgroundLabelledPixels(image, labelmap);<br>Ext.CLIJ2_statisticsOfLabelledPixels(input, labelmap); |  |  |  |  |
| Push Results Table  |                        |        | 2D                        | Ext.CLIJ2_pushResultsTable(image_name);  |  |  |  |  |
| Push Results table column   | ,"Mean"                |        | 2D                        | Ext.CLIJ2_pushResultsTableColumn(image_name, column_name);   |  |  |  |  |
| Pull to Results table   |                        |        | 2D                        | Ext.CLIJ2_pullToResultsTable(image_name);  |  |  |  |  |
| Push Array  | [1,4,0,0,0,2], 3, 2, 1 |        | 2D                        | CLIJ2_pushArray(image_name, array, width, height, depth);  |  |  |  |  |
| Detailed documentation  |                        |        | Installation instructions |  |  |  |  |  |
| CLIJ documentation can be found   |                        |        |                           |  |  |  |  |  |
| • In CLIJ dialogs under the menu Plugins > ImageJ on GPU (CLIJ2)                            |                        |        |                           |  |  |  |  |  |
| • Embedded in Fiji script editor - just start typing  |                        |        |                           |  |  |  |  |  |
| • Online: <a href="https://clij.github.io/clij2-docs">https://clij.github.io/clij2-docs</a> |                        |        |                           |  |  |  |  |  |
| CLIJ documentation can be found   |                        |        |                           |  |  |  |  |  |
| • In CLIJ dialogs under the menu Plugins > ImageJ on GPU (CLIJ2)                            |                        |        |                           |  |  |  |  |  |
| • Embedded in Fiji script editor - just start typing  |                        |        |                           |  |  |  |  |  |
| • Online: <a href="https://clij.github.io/clij2-docs">https://clij.github.io/clij2-docs</a> |                        |        |                           |  |  |  |  |  |

# Online support

Wei-Chen CHU weichen Mar 25

Hi, Robert  
I have a similar related question:  
In case of large images that exceeds the capacity of GPU memory.  
Are there any functions in CLIJ can allow lazy- loading/processing?  
(Something like "Using virtual stack" in the Bioformat importer)

1 ❤️ 🔍 ... ↗ Reply



<https://image.sc>

Robert Haase haesleinhuepf clij & clesperanto maintainer Mar 26

Hi @weichen ,  
we had a discussion earlier about potential strategies in these threads:

Hey @rfrs , what kind of denoising / analysis where you thinking of? I'm also working with light sheet data a lot and we use `downsampleSliceBySliceHalfMedian` in routine to reduce our amount of data while imaging. To fit your data into GPU memory of your computer, you have several strategies: Process it slice by slice using `pushCurrentZSlice`. There is also an [example macro](#) for this. Process tiles using the methods `pushTile` and `pullTile` as also discussed in [this thread](#). Please note that `pushTi...`

Segmentation of islets in fluorescence image ■ Image Analysis

Awesome! Let me know how it goes! For working with huge images, there are two new commands in CLIJx: `pushTile` and `pullTile` which allow you to process a huge image piece wise. Here is some example code: <https://github.com/clij/clij2-docs/blob/master/src/main/macro/processTiles3D.ijm> for 

```
(x = 0; x < numTilesX; x++) { for (y = 0; y < numTilesY; y++) { for (z = 0; z < numTilesZ; z++) { Ext.CLIJx_pushTile(original, x, y, z, tileSizeWidth, tileSizeHeight, tileSizeDepth, margin, margin, margin); Ex...
```

If you work with Python, you could also use `dask` for processing the data tile-by-tile: [https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/32\\_tiled\\_image\\_processing/tiled\\_nuclei\\_counting\\_quick.html](https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/32_tiled_image_processing/tiled_nuclei_counting_quick.html)

Furthermore, cropping and downsampling are often alternatives for answering a scientific question without processing every single pixel.

Let us know if this helps!

Best,  
Robert

✓ Solution 2 ❤️ 🔍 📌 ↗ Reply

<https://forum.image.sc/t/clij-plugin-gpu-error/79034/5>



ICOB Imaging Core

104

# Citation of CLIJ

- If you work with CLIJ and friends, please cite the paper(s).

The screenshot shows three side-by-side web pages related to the CLIJ paper:

- Nature Methods Article Page:** Shows the full article details including authors (Robert Haase, Loic A. Royer, Peter Steinbach, Dibrov, Uwe Schmidt, Martin Weigert, Nicola Maghelli, Eugene W. Myers), publication date (18 November 2019), and links to the journal issue (Nature Methods 17, 5–6(2020)) and the SpringerLink chapter.
- SpringerLink Chapter Page:** Shows the chapter details from the book "Bioimage Data Analysis Workflows – Advanced Components".
- bioRxiv Preprint Page:** Shows the preprint details including the title ("Interactive design of GPU-accelerated Image Data Flow Graphs and cross-platform deployment using multi-lingual code generation"), authors (Robert Haase, Akanksha Jain, Stéphane Rigaud, Daniela Vorkel, Pradeep Rajasekhar, Theresa Suckert, Talley J. Lambert, Juan Nunez-Iglesias, Daniel P. Poole, Pavel Tomancak, Eugene W. Myers), DOI (<https://doi.org/10.1101/2020.11.19.386565>), and a note about peer review.

<https://www.nature.com/articles/s41592-019-0650-1>

[https://link.springer.com/chapter/10.1007/978-3-030-76394-7\\_5](https://link.springer.com/chapter/10.1007/978-3-030-76394-7_5)

<https://www.biorxiv.org/content/10.1101/2020.11.19.386565v1>

Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

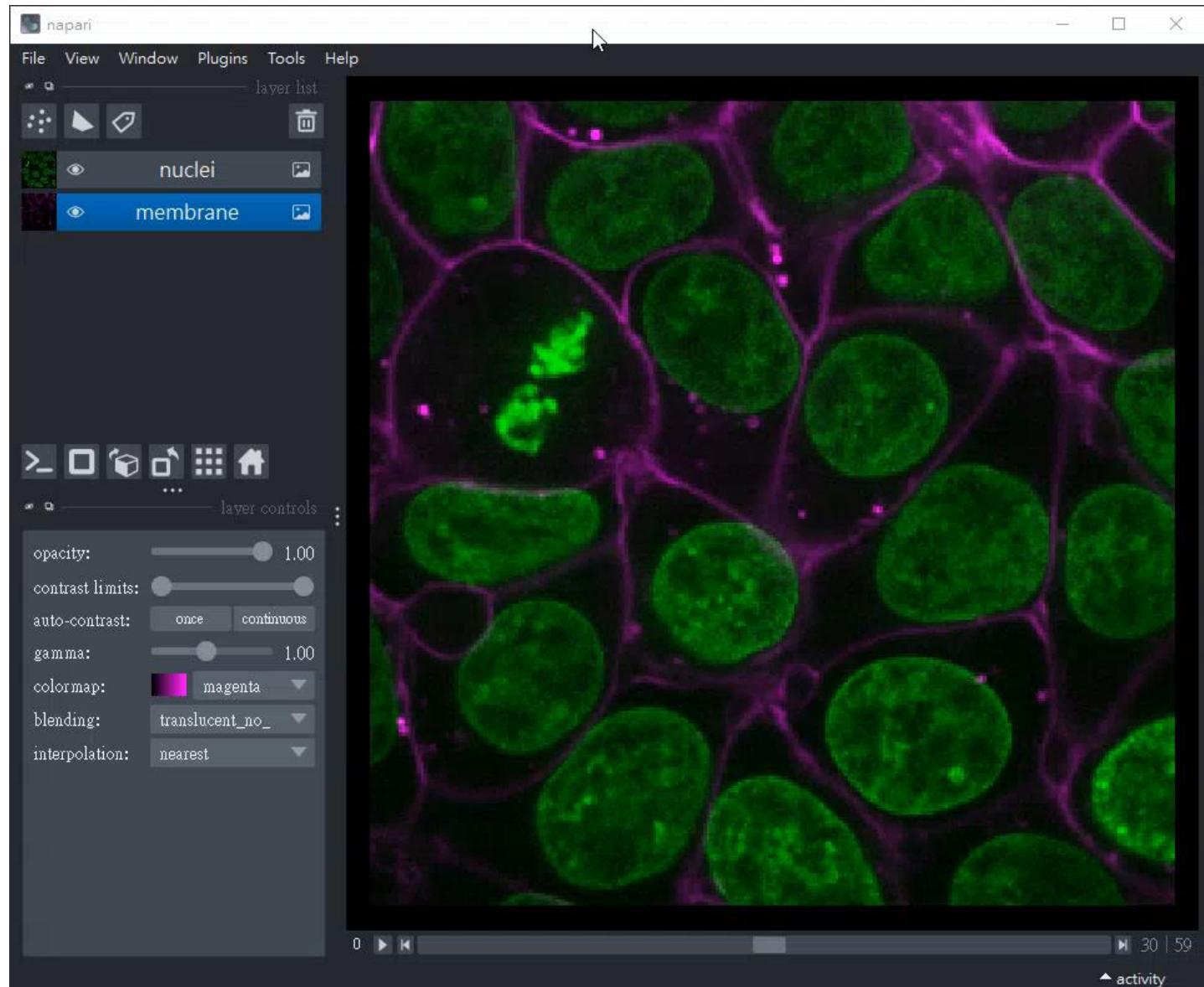
105

# From FIJI to Napari



# napari

- Multi-dimensional image viewer in python
- 3D Visualization!



<https://napari.org/>

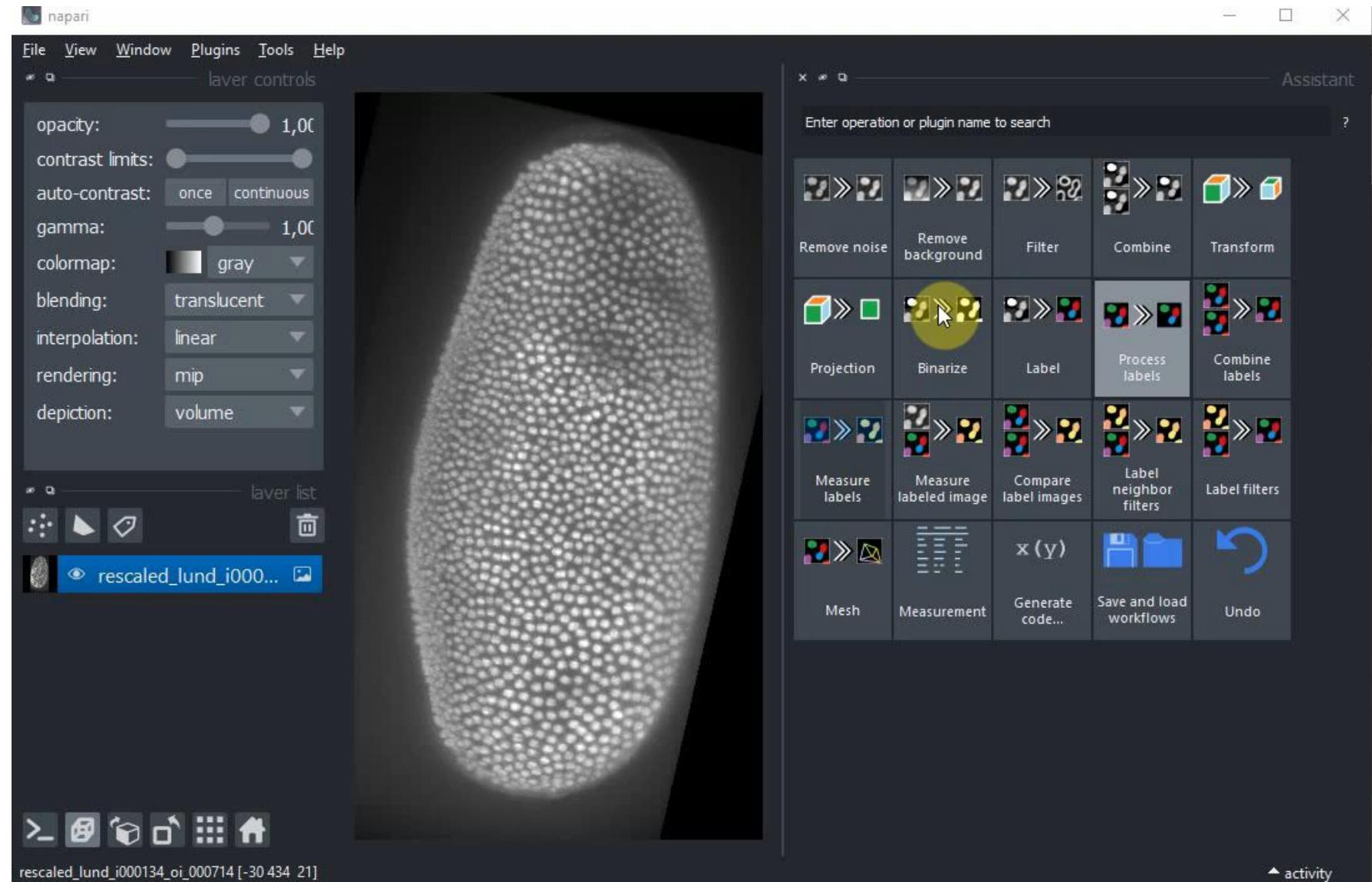


ICOB Imaging Core

108

# The Napari Assistant

- 3D visualization
- Undo [redo]



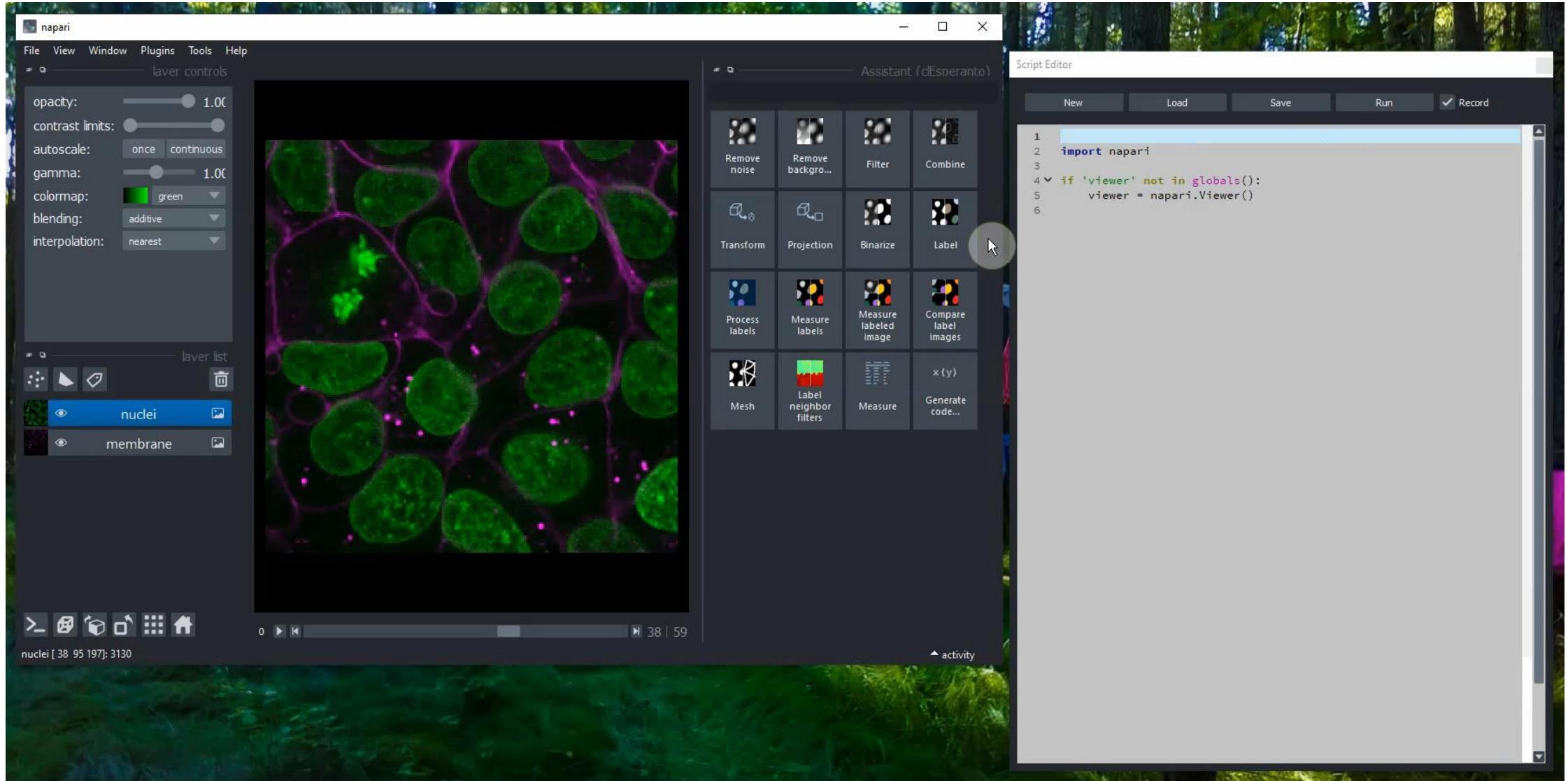
<https://www.napari-hub.org/plugins/napari-assistant>

Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

# Record code instead of writing it!



Adapted from the learning material of Dr. Robert Haase



ICOB Imaging Core

110

# clEsperanto

cle.

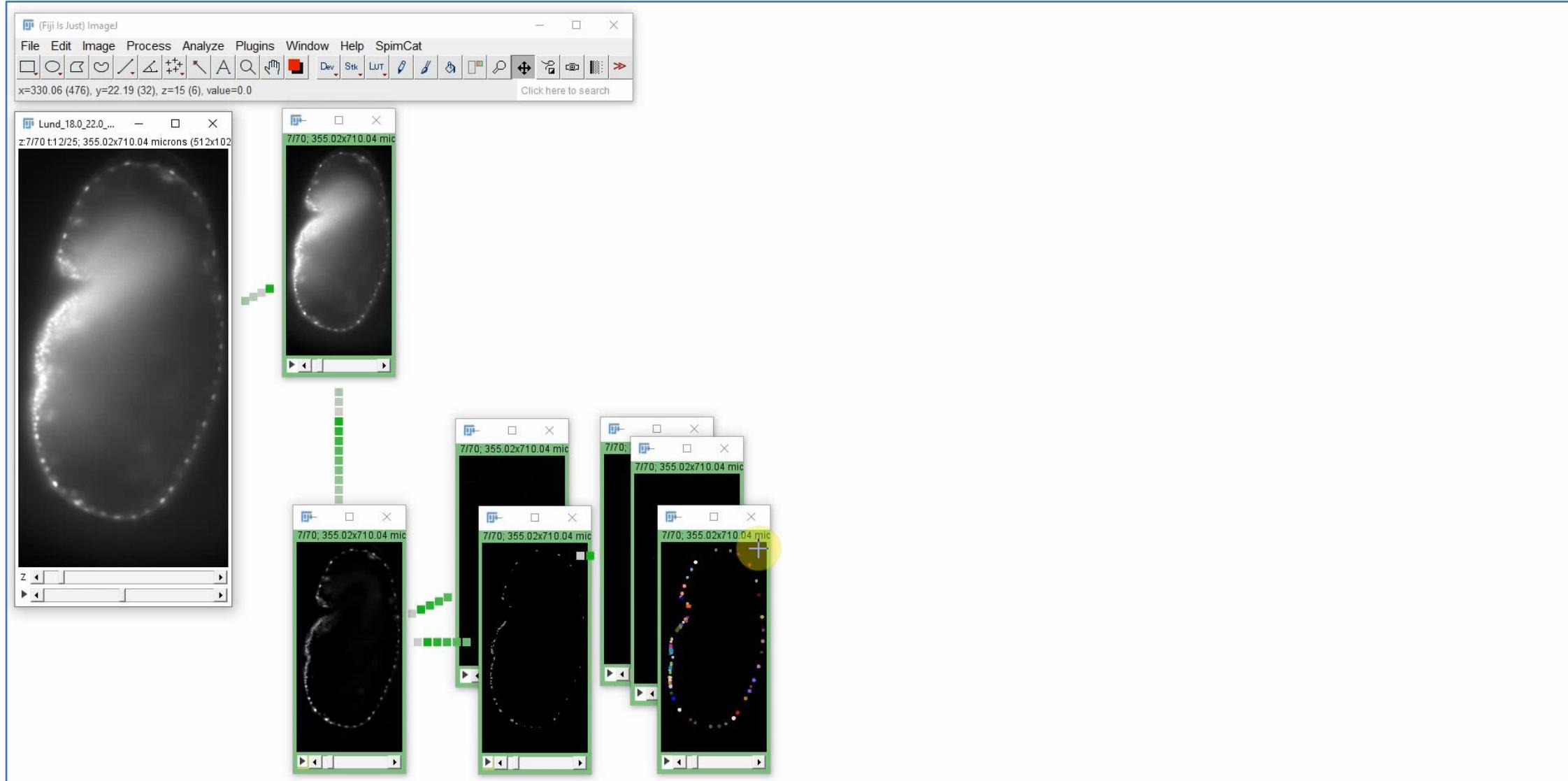
- Use the same command for Fiji, Python, Matlab, and Icy for image analysis!
- GPU- acceleration!

The screenshot shows the Fiji software interface with the title bar '(Fiji Is Just) Image'. The menu bar includes File, Edit, Image, Process, Analyze, Plugins, Window, Help, and SpimCat. Below the menu is a toolbar with various icons. A search bar says 'Click here to search'. The main window has a 'Developer Menu' button and a 'segmentation.py (Running)' tab. The code editor displays the 'segmentation.py' script:

```
24 # push image to GPU
25 input = clijx.push(input);
26
27 # reserve memory for output, same size and type as input
28 blurred = clijx.create(input);
29 thresholded = clijx.create(input);
30 labelled = clijx.create(input);
31
32 # blur, threshold and label the image
33 clijx.blur(input, blurred, 5, 5, 0);
34 clijx.automaticThreshold(blurred, thresholded, "Otsu");
35 I
36
37 # show result
38 clijx.show(labelled, "labelled");
39 IJ.run("glasbey");
40
41 # clean up
42 input.close();
43 blurred.close();
44 thresholded.close();
45 labelled.close();
46
47
```

The code editor has tabs for Run, Batch, Kill, and persistent. Buttons for Show Errors, Clear, and a right-pointing arrow are at the bottom. To the left is a file browser showing a directory structure with folders like python, macro, resources, main, beanshell, groovy, java, javascript, and python sub-folders containing various scripts. A yellow circle highlights the variable 'I' in the code.

# From Fiji to napari

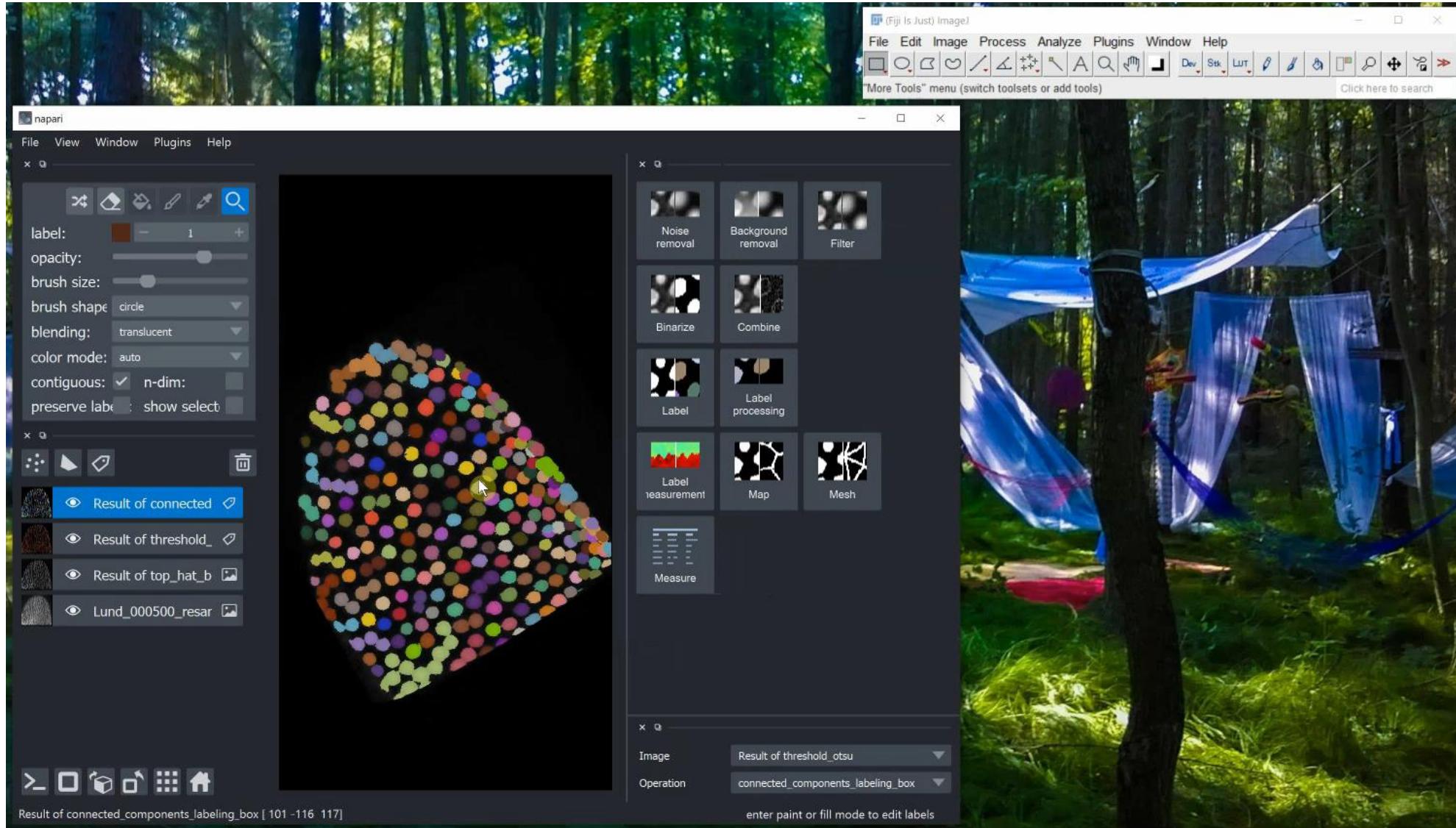


Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

# From Napari to Fiji



Adapted from the learning material of Dr. Robert Haase



**ICOB Imaging Core**

# Checklist



Continue using Fiji if, ...

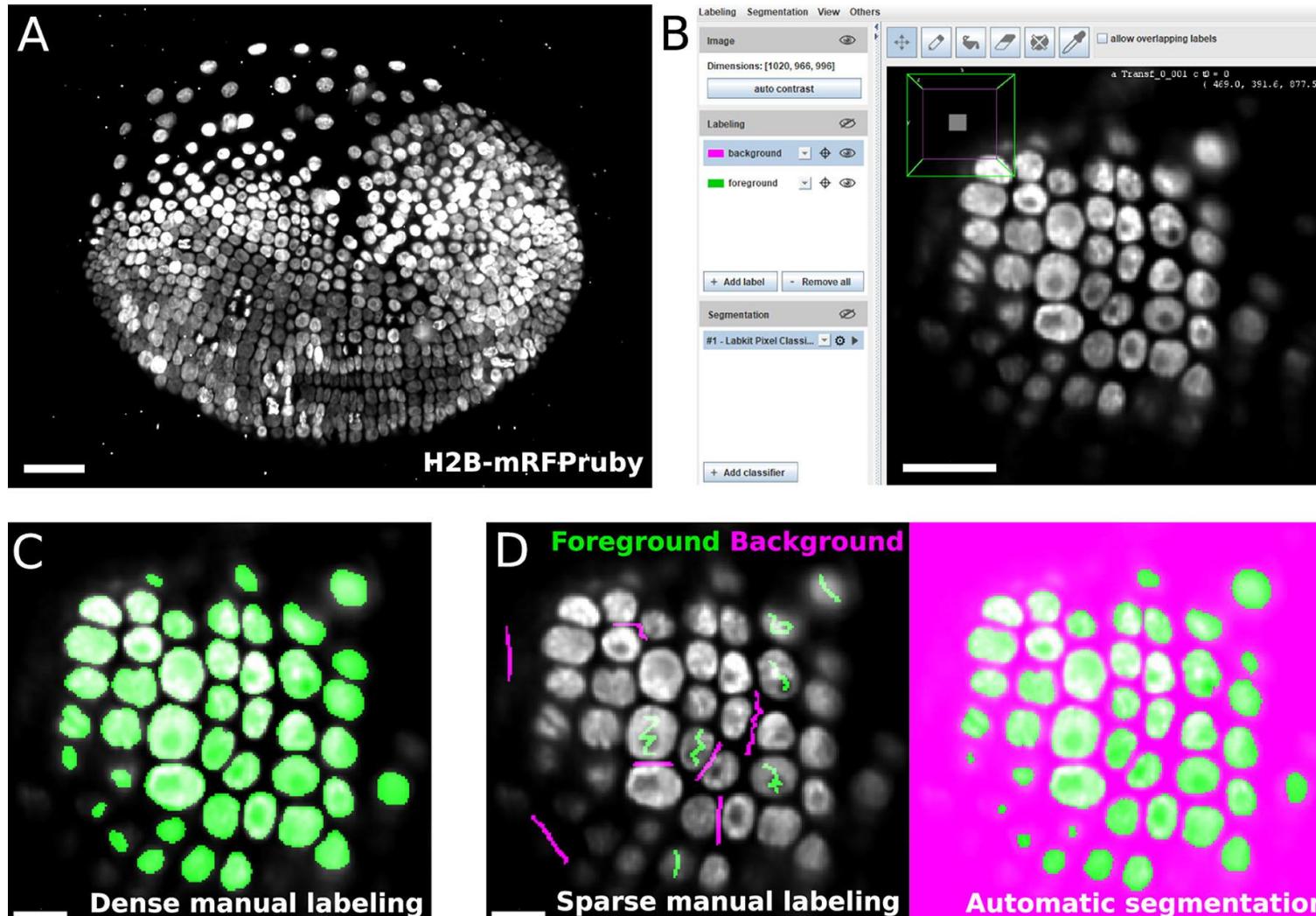
- **it does the job you need.**
- you don't speak python (yet).
- you want to do [cell-]tracking or **image stitching**.
- work with **big data**
- you're ok with just applying advanced deep learning models
- want to keep things **simple**. (especially installation)



Try out Napari if

- you work with **3D** (+time, channel) data
- your data fits in memory
- you want to use **machine learning** for analyzing image/object data (also: ilastik)
- tools you need are listed on the **napari hub**
- you speak python, or
- you want to **learn python!**

# Pixel classification by LABKIT



- GPU-acceleration by CLIJ (OpenCL)
- Integrate with Imaris (Since Ver 9.9)

[https://imaris.oxinst.com/learning/vie\\_w/article/how-to-use-labkit-with-imaris](https://imaris.oxinst.com/learning/vie_w/article/how-to-use-labkit-with-imaris)

- Can handle with big data!

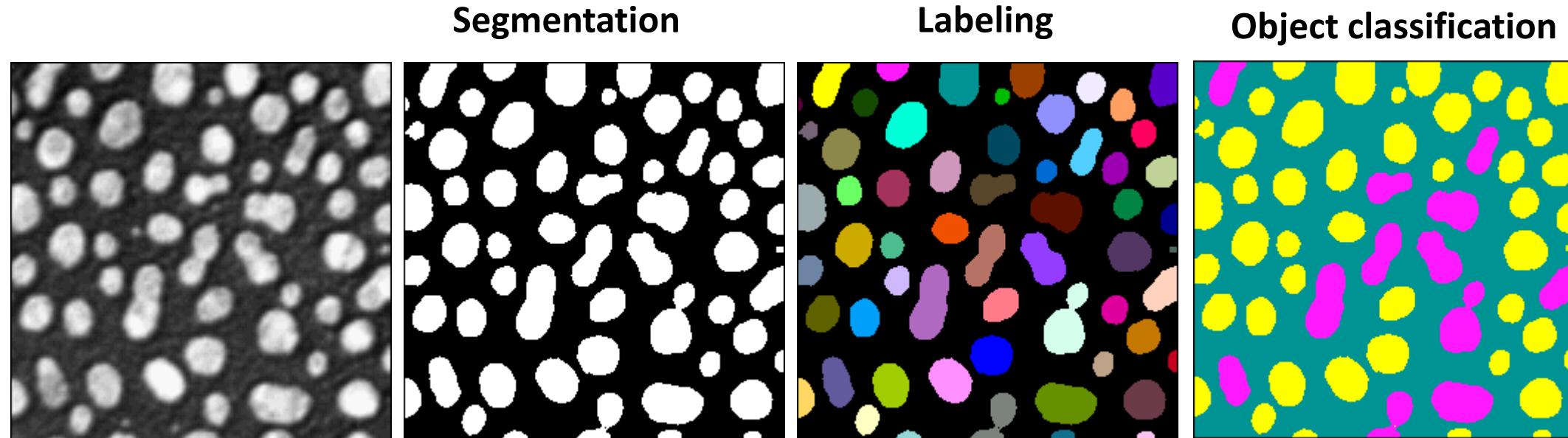
<https://www.youtube.com/watch?v=AdredeoUkrUU&t=19s>

# Pixel and Object classification by iLastik

- GPU-acceleration (Nvidia CUDA)
- Can handle with big data!

<https://www.youtube.com/watch?v=AedreoUkrUU&t=19s>

iLastik

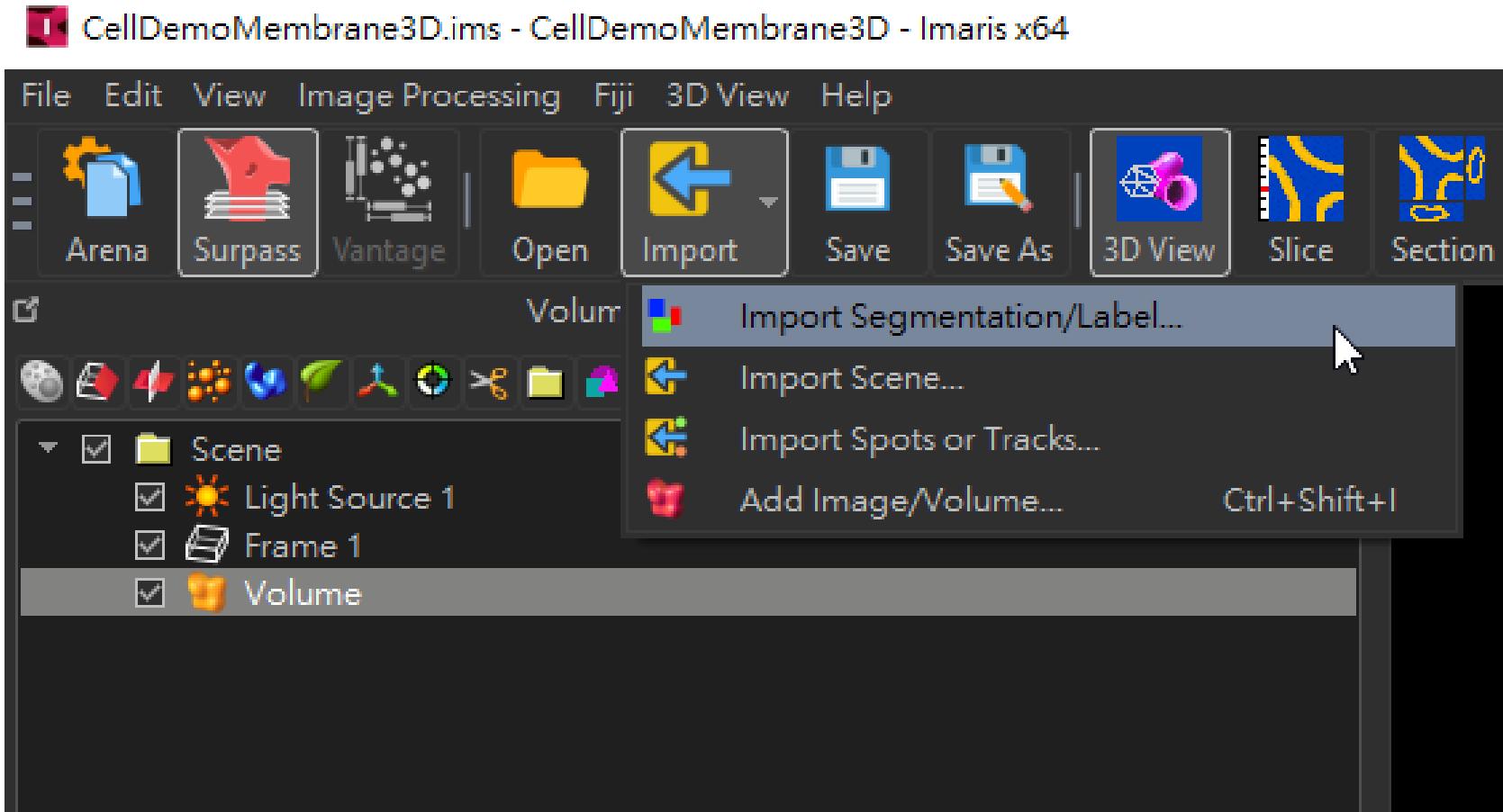


<https://www.ilastik.org/>



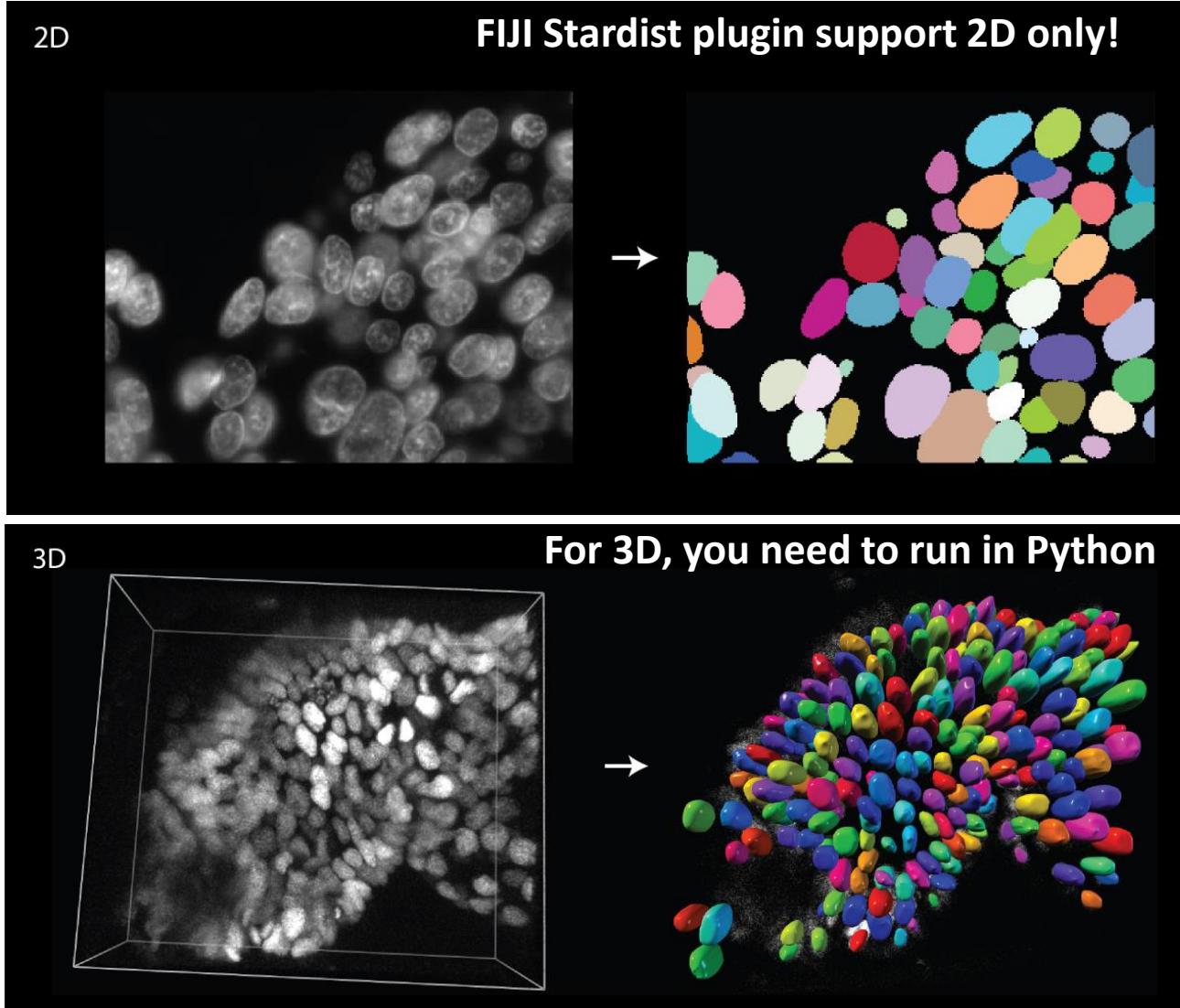
ICOB Imaging Core

# Mask /label can import to Imaris as a “Surface” (since Ver 9.9)



<https://imaris.oxinst.com/learning/view/article/how-to-import-label-images-from-ilastik>

# Stardist



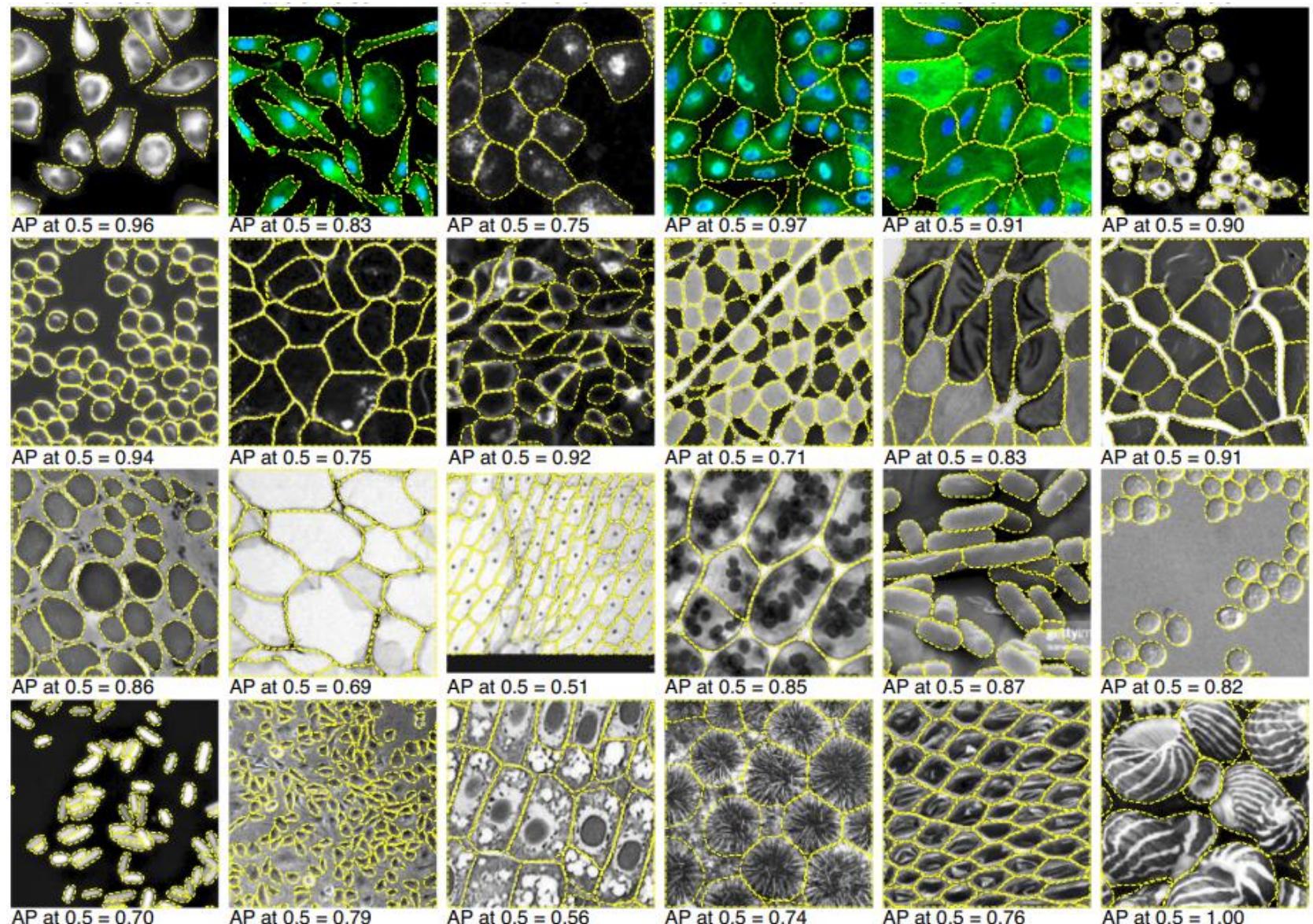
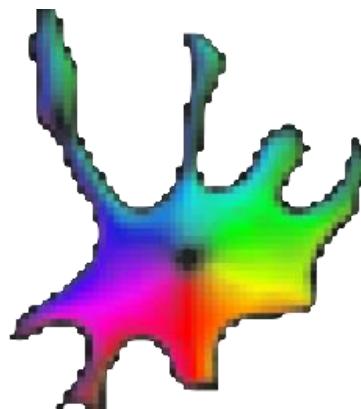
- Useful for nuclei segmentation (or something similar)
- GPU-acceleration (Nvidia CUDA)

Webinar:

[https://www.youtube.com/watch?v=Amn\\_eHRGX5M](https://www.youtube.com/watch?v=Amn_eHRGX5M)

# Cellpose/ Cellpose 2

- Deep Learning segmentation tools
- GPU-acceleration (Nvidia CUDA)



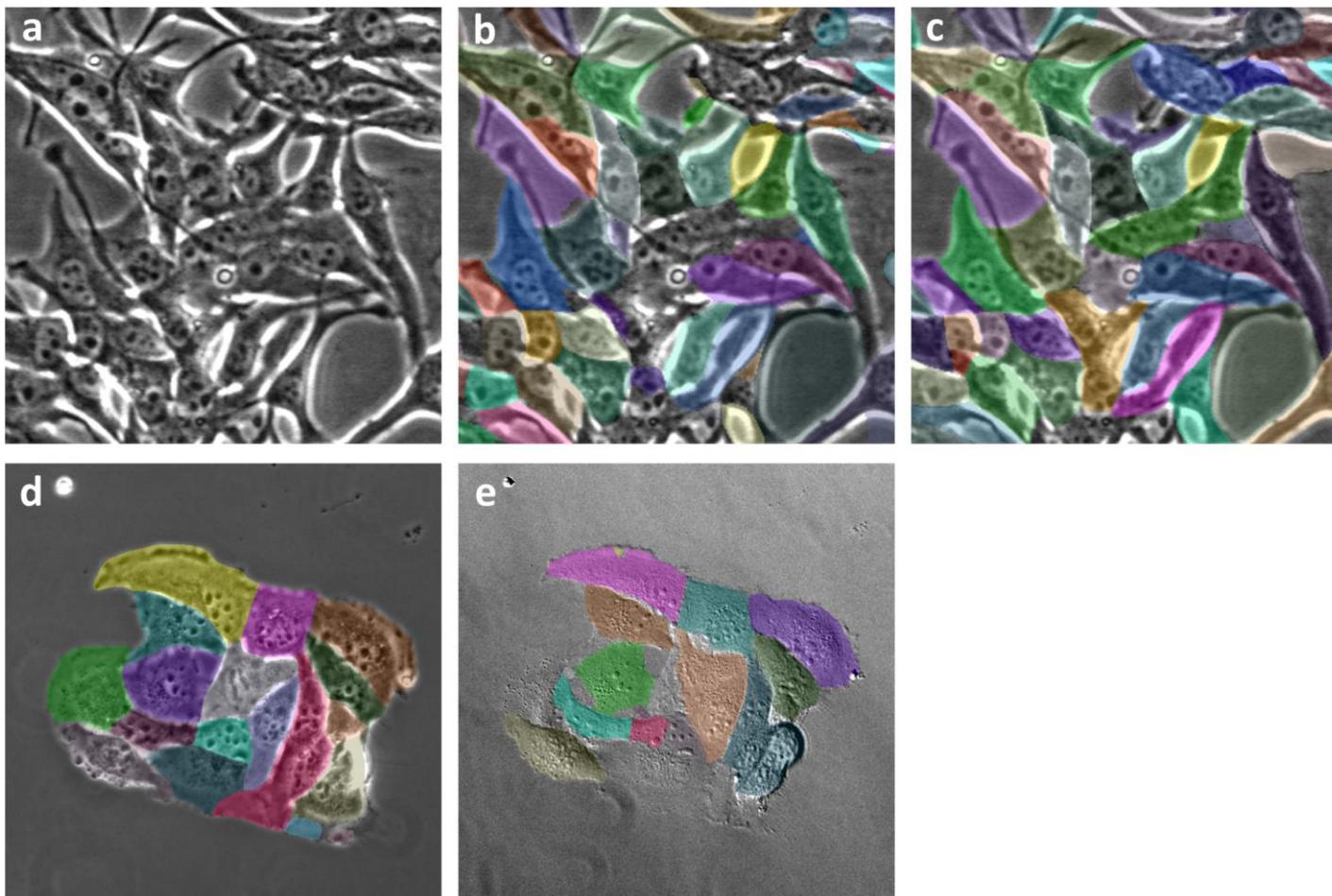
<https://github.com/MouseLand/cellpose>



ICOB Imaging Core

119

# Cellpose is a great tools for BF, Ph, DIC segmentation



My workflow:

Image of Ph/DIC channel

↓  
Cellpose  
(Train for your model is better)

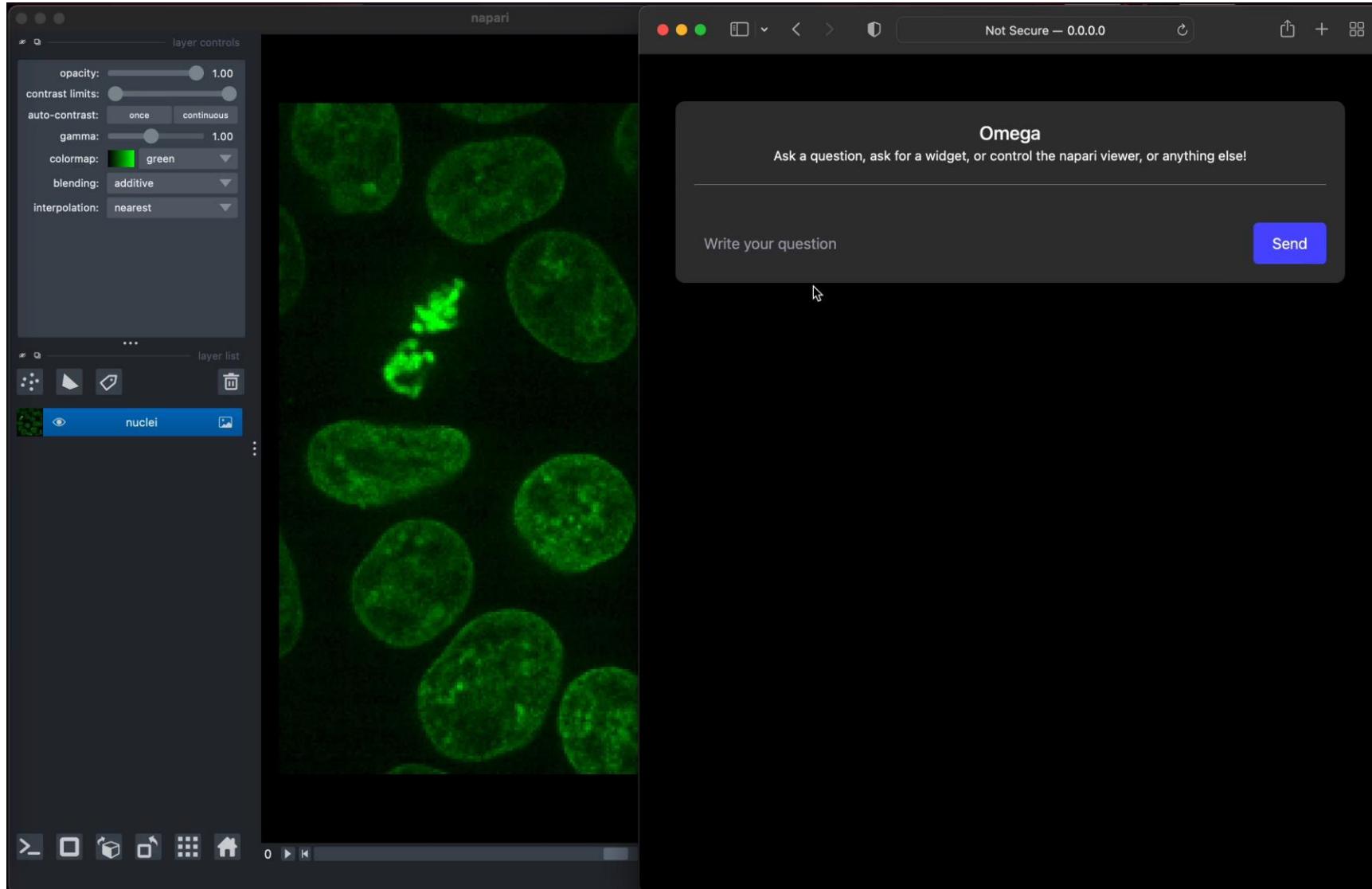
Segmentation Image file

↓  
pullLabelsToROIManager (CLIJ2)

ROI List

↓  
Measure on the  
fluorescence channel

# Napari-chatgpt (OMEGA)



<https://github.com/royerlab/napari-chatgpt>



**ICOB Imaging Core**

121

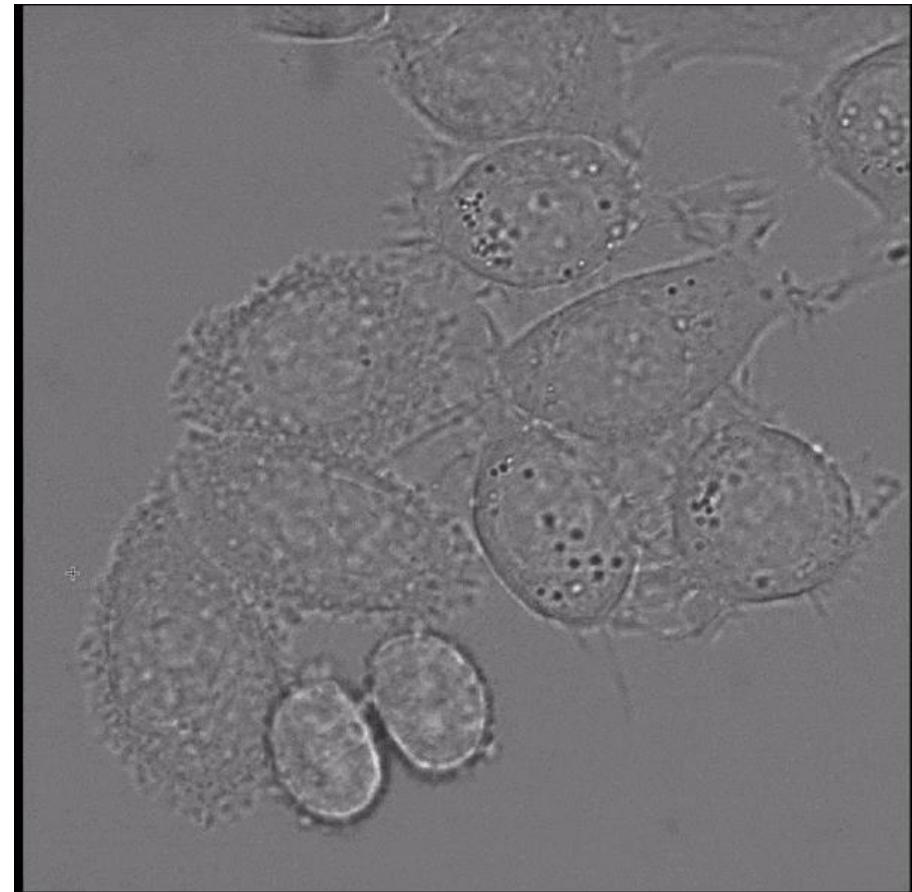
# Segment Anything Model (SAM) for Microscopy



<https://github.com/facebookresearch/segment-anything>



<https://arxiv.org/abs/2304.02643>

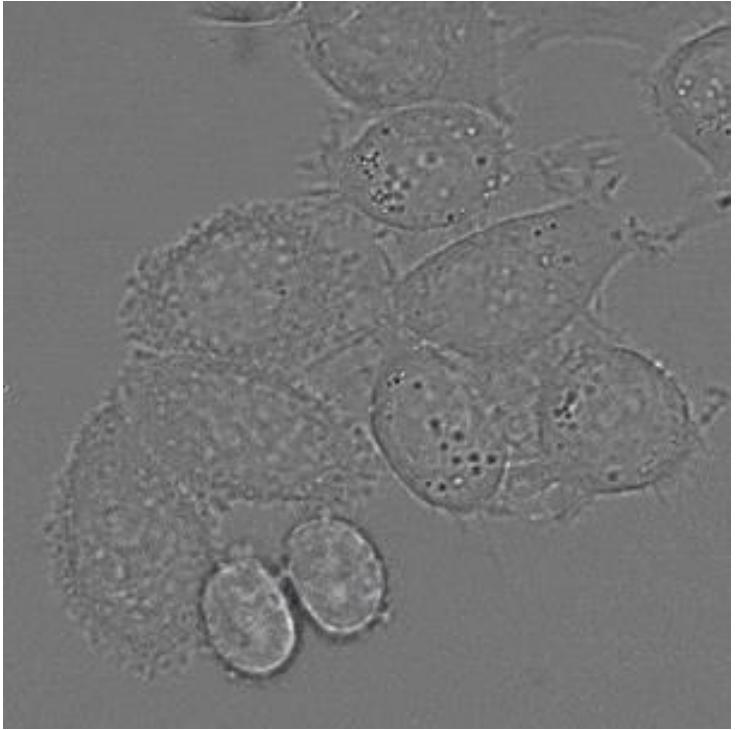


<https://github.com/computational-cell-analytics/micro-sam>

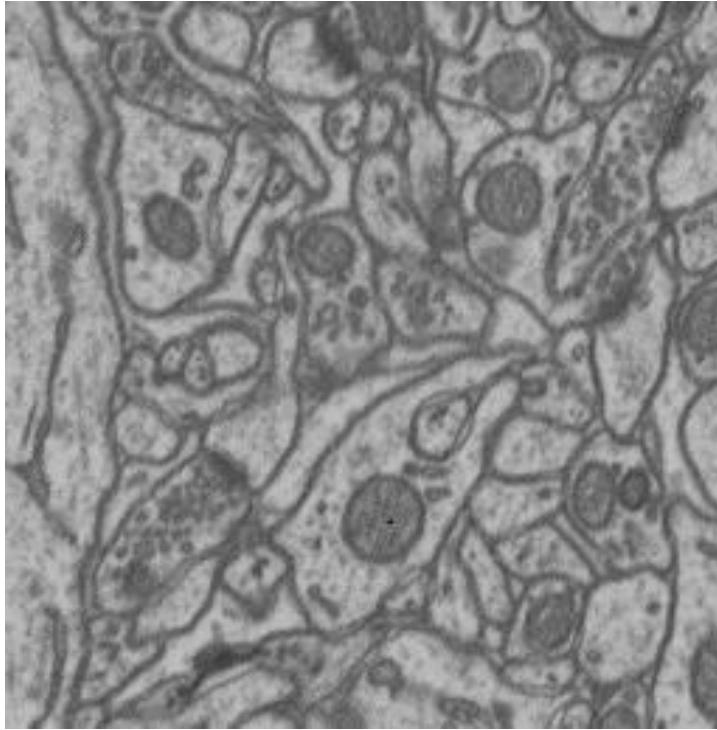


**ICOB Imaging Core**

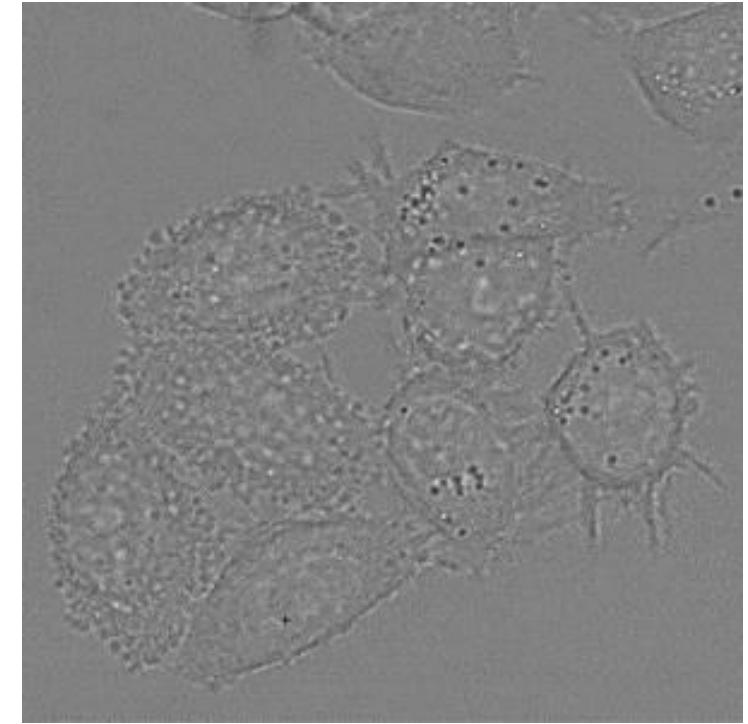
# Segment Anything Model (SAM) for Microscopy



2D

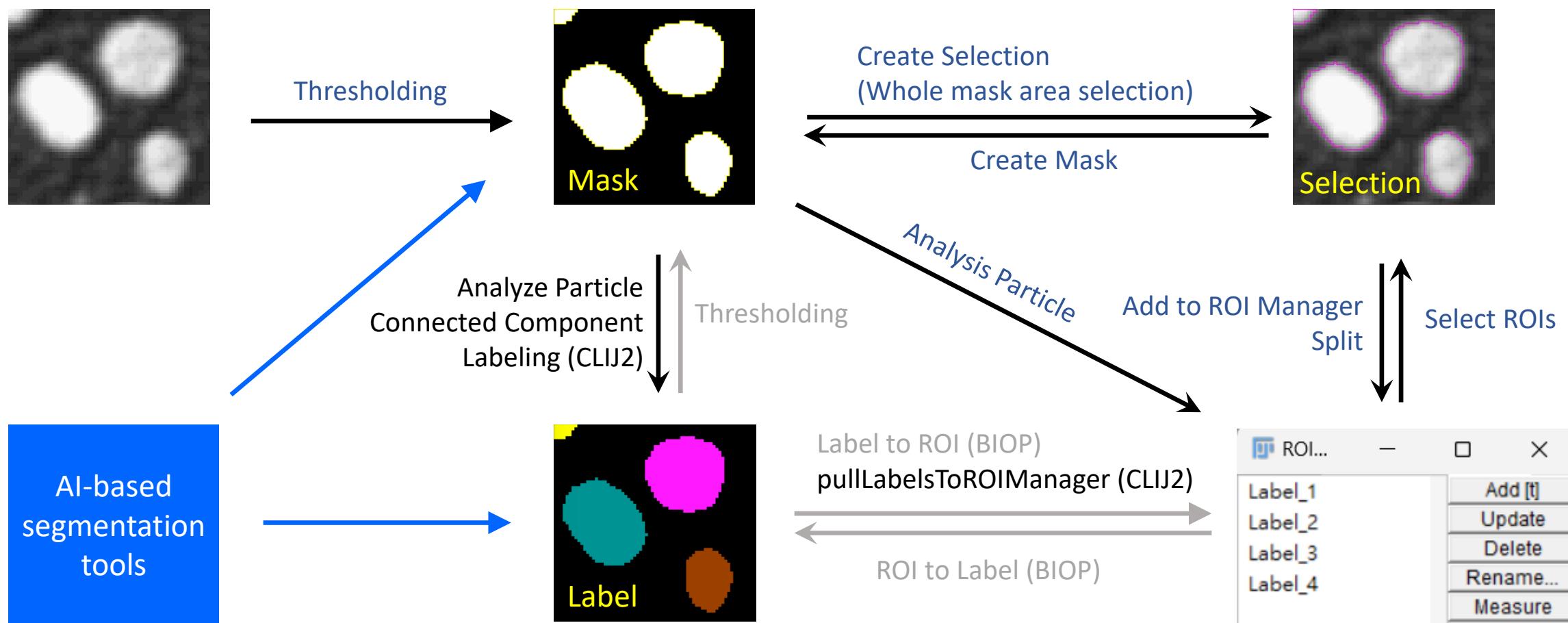


3D



2D + Tracking

# Segmentation with ROI list, Mask, and Label



This slide is important!



# Acknowledgements



**Dr. Robert Haase**  
“Physic of Life” (PoL),  
TU Dresden, Germany



**Dr. Peter Bankhead**  
University of Edinburgh, UK



<http://eubias.org/>



**ICOB Imaging Core**