



ZELDA: a 3D Image Segmentation and Parent-Child relation plugin for Microscopy Image Analysis in napari

Rocco D'Antuono

Crick Advanced Light Microscopy STP

The Francis Crick Institute (London, UK)

Image data science for with
Python and Napari

11 August 2023

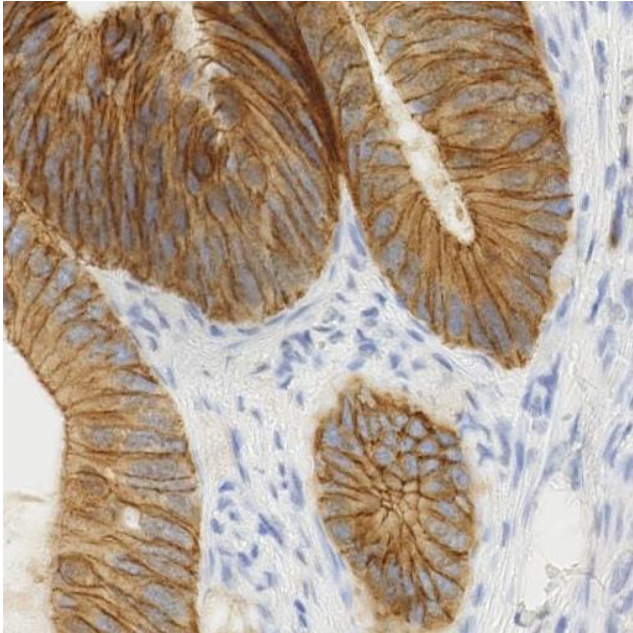


<https://doi.org/10.3389/fcomp.2021.796117>

<https://github.com/RoccoDAnt/napari-zelda>

Example of scripting: IPython console

Obtain segmented region outlines



```
1 from skimage import data
2 from skimage.color import rgb2gray
3 from skimage import filters
4
5 viewer.add_image(data.immunohistochemistry(), rgb=True, blending='additive')
6 binary=rgb2gray(viewer.layers[0].data)>0.6
7 edge_sobel = filters.sobel(binary)
8 viewer.add_image(edge_sobel, blending='additive', colormap='yellow')
```

A screenshot of the Napari software interface. The top menu bar includes 'File', 'View', 'Window', 'Plugins', and 'Help'. Below the menu bar, there are two panels: 'layer controls' and 'layer list'. The 'layer controls' panel has sliders for 'opacity' (set to 1.00), 'contrast limits', and 'gamma' (set to 1.00). It also has dropdown menus for 'autoscale' (set to 'once'), 'colormap' (set to 'yellow'), 'blending' (set to 'additive'), and 'interpolation' (set to 'nearest'). The 'layer list' panel shows two layers: 'Edges' (selected) and 'Immunohistochemi...'. The main view area shows the microscopy image with yellow outlines overlaid. At the bottom, there is a 'Jupyter QtConsole 5.1.1' window showing the execution of the Python script from the previous block. The console output shows the script being executed successfully. A blue arrow points from the 'Edges' layer in the layer list to the 'layer controls' panel. A red arrow points from the Python code block to the IPython console.

Fig. 2, <https://analyticalscience.wiley.com/doi/10.1002/was.0004000232>

What biologists want?



“open-source point and click software”

*“better software
for “3D/Volume” and “Tissue/Histology” analysis”*

COBA survey

Jamali et al. 2022. *Biological Imaging*, 1, E4.
doi:10.1017/S2633903X21000039

The lexicon of BIA

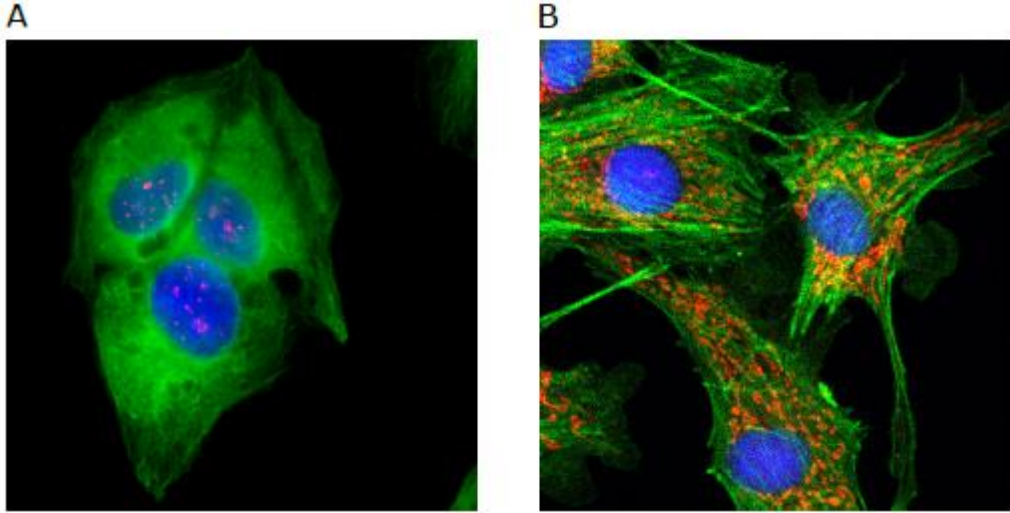


*“information about tools is non-uniform
and often focuses on technicalities”*

“aim to help bioimage analysts to identify and edit workflows”

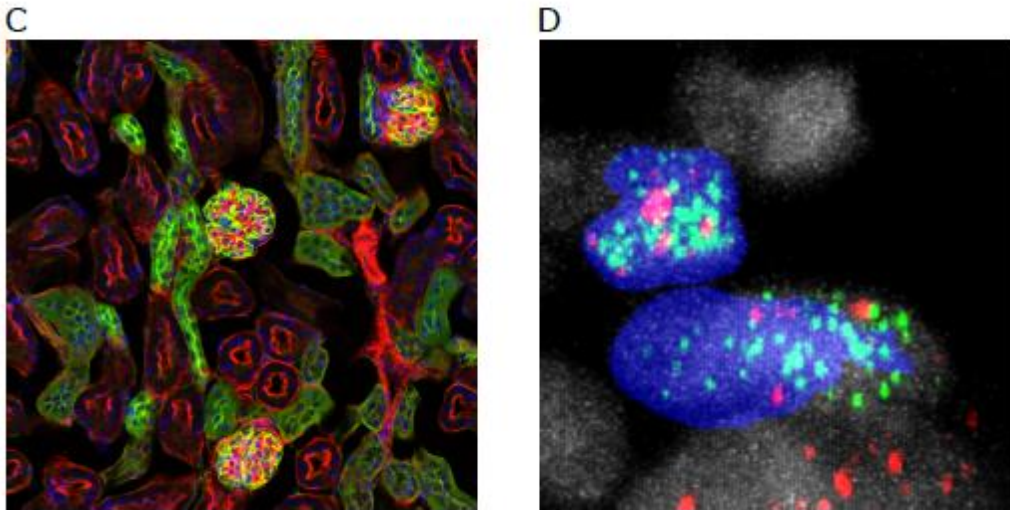
biii.eu

Some common bioimage analysis problem



(A) “2D counting”

(B) “2D segmentation” and “parent-child relation”.



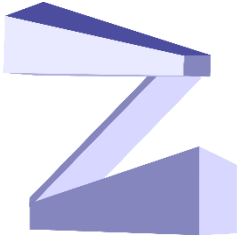
(C) “3D cell counting” or “3D object segmentation”

(D) “3D object segmentation” and “parent-child relation”

D'Antuono, Rocco, and Giuseppina Pisignano. 2022.

“ZELDA: A 3D Image Segmentation and Parent-Child Relation Plugin for Microscopy Image Analysis in Napari.”

Frontiers in Computer Science 3: 115. <https://doi.org/10.3389/fcomp.2021.796117>



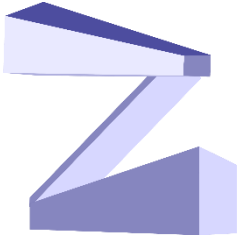
<https://github.com/RoccoDAnt/napari-zelda>

Software features:

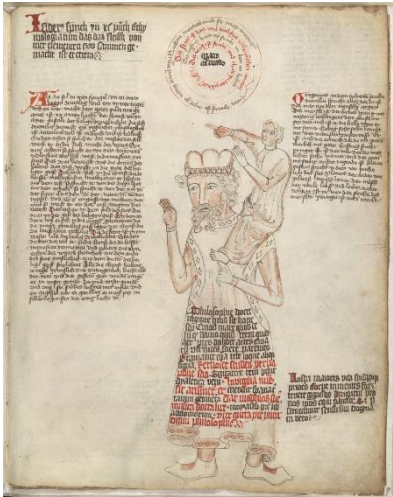
- Guided segmentation
- Graphical workflow
- Versatile
(3D,
Parent-Child,
Workflow composition,
Automatic data plotter)
- Reasonably Limited!
- Benchmarked (2D/3D)

The screenshot displays the napari interface with the ZELDA plugin. The main window shows a 3D microscopy image with segmented objects. The left sidebar shows layer controls and a layer list. The bottom panel shows the ZELDA workflow steps: Image calibration, Gaussian Blur, Threshold, Distance Map, Show seeds, Segment, Measure segmented objects, and Plot results and save graphs. The right sidebar shows the ZELDA launch dialog with protocols and a 'Run' button.

Segmentation algorithm for a single population of "objects"



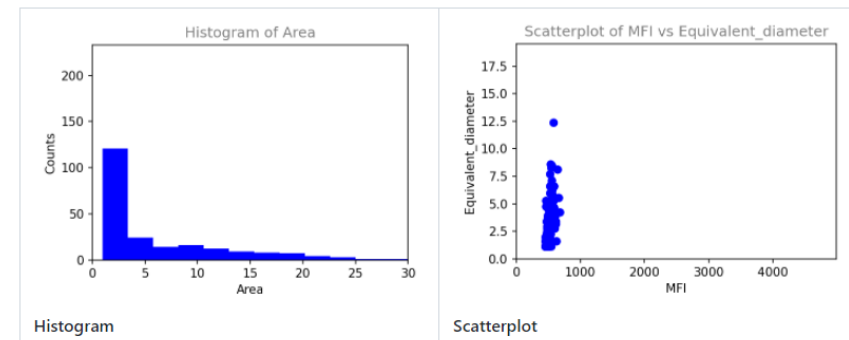
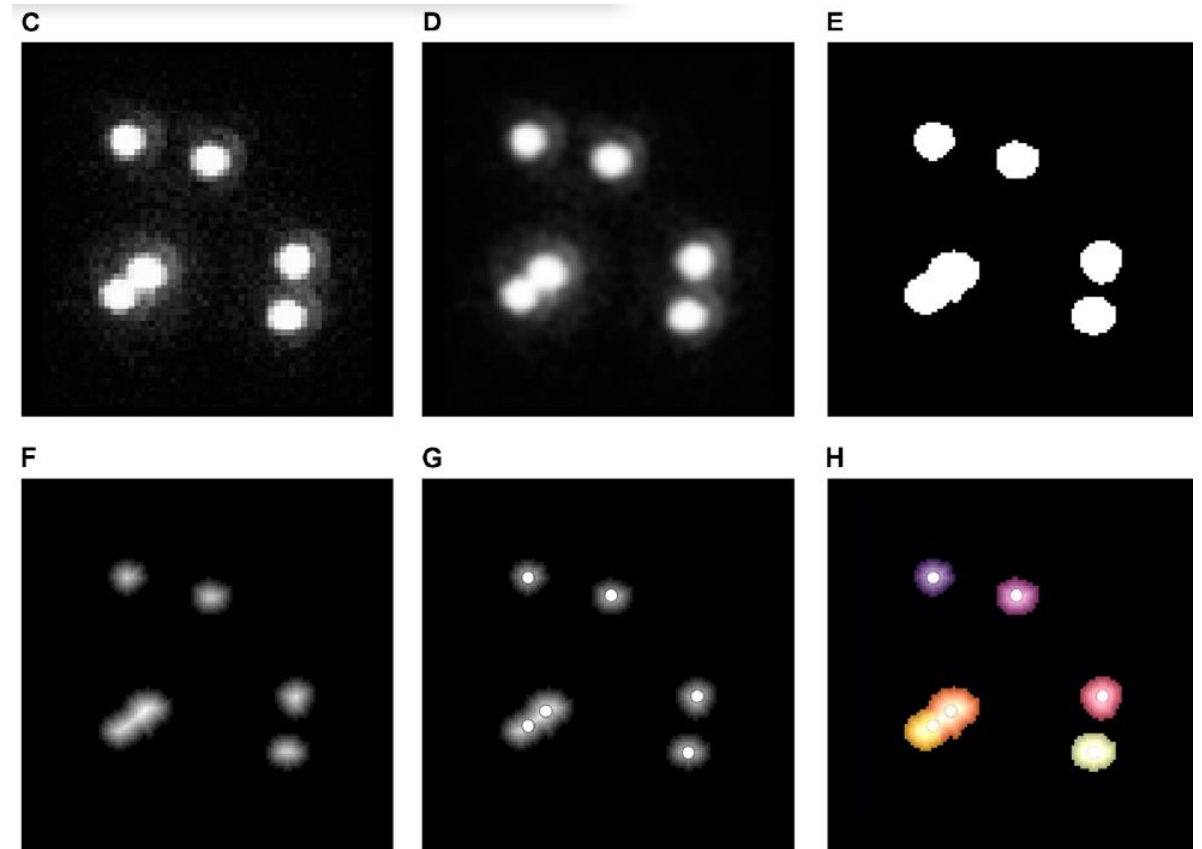
<https://github.com/RoccoDAnt/napari-zelda>

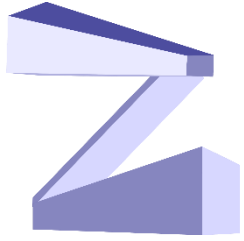


From wikimedia.org

"standing on the sholders [sic] of Giants:":

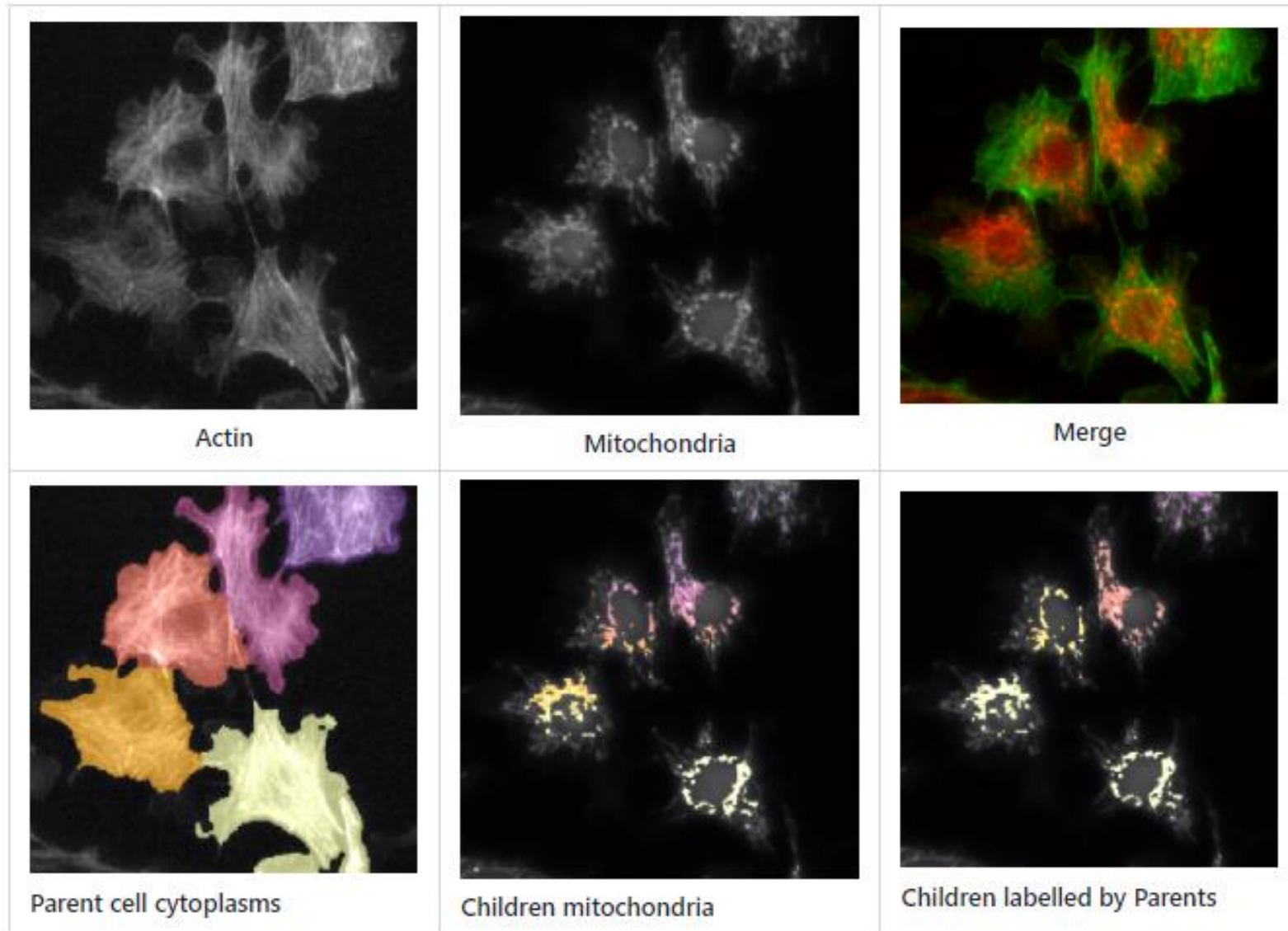
- MorphoLibJ (Legland et al., 2016)
- CellProfiler (McQuin et al., 2018)
- Scikit-Image (Walt et al. 2014)
- ...



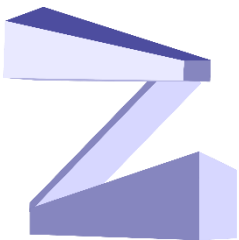


<https://github.com/RoccoDAnt/napari-zelda>

Parent-Child relation



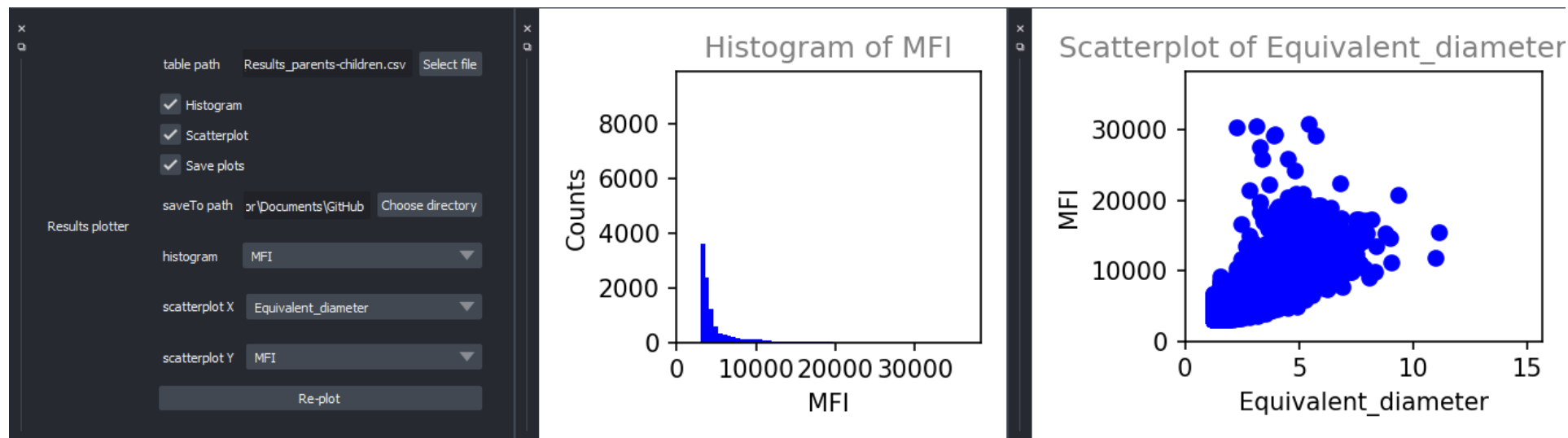
Parent objects	GB: sigma=2.0-> Th_parents=60.0-> DistMap-> Maxima: min_dist=10
Children objects	GB: sigma=0.3-> Th_children=450.0 -> DistMap-> Maxima: min_dist=2



<https://github.com/RoccoDAnt/napari-zelda>

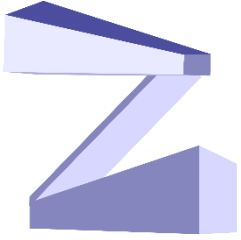
Parent_label	Area	Equivalent_diameter	MFI
4	61524	44.50092806	21400.02779
6	51327	41.89248994	22328.22453
0	99	5.214739423	19818.31818
5	7704	22.263476	20383.78855
5	7771.5	22.32830885	20670.41228
4	74502	47.43259198	24658.9672
5	99049.5	52.15608401	24016.61083
5	56317.5	43.20844824	22260.36548
1	73584	47.23696771	24755.67967
6	77463	48.05283162	25165.21064
5	13725	26.98930273	22365.67377

Data Plotter



matplotlib.backends.backend_qt5agg

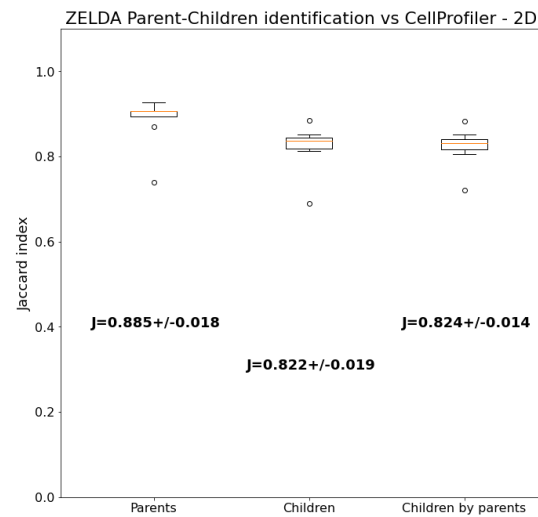
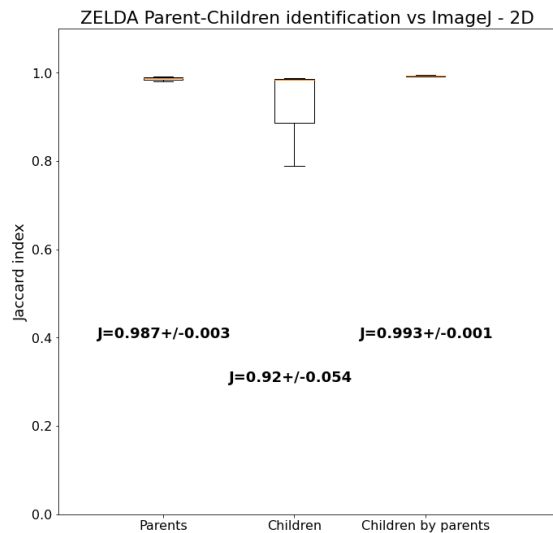
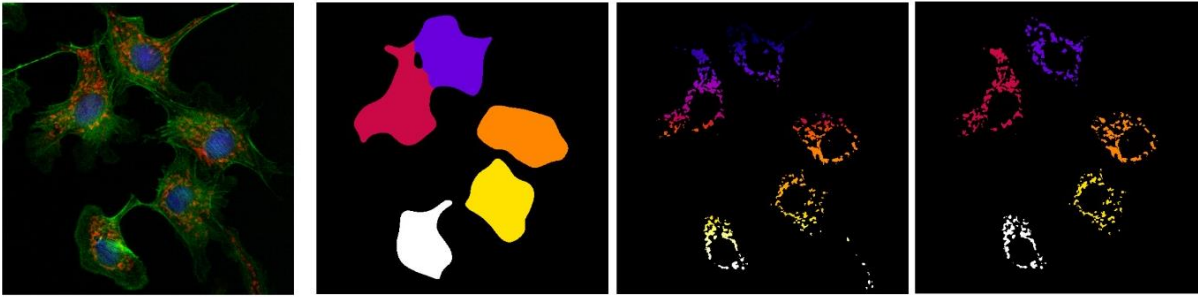
(https://matplotlib.org/2.2.2/_modules/matplotlib/backends/backend_qt5agg.html)



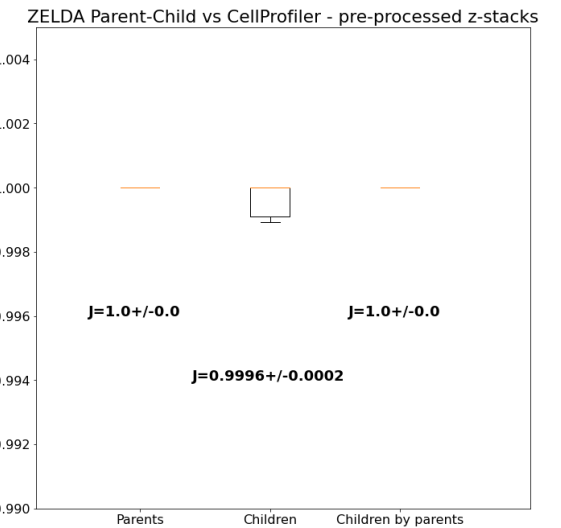
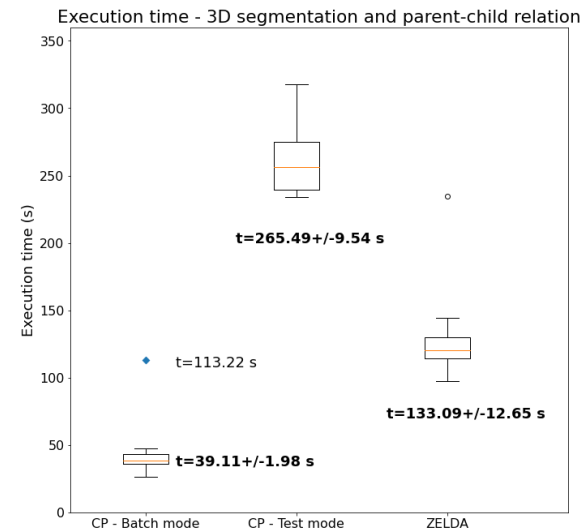
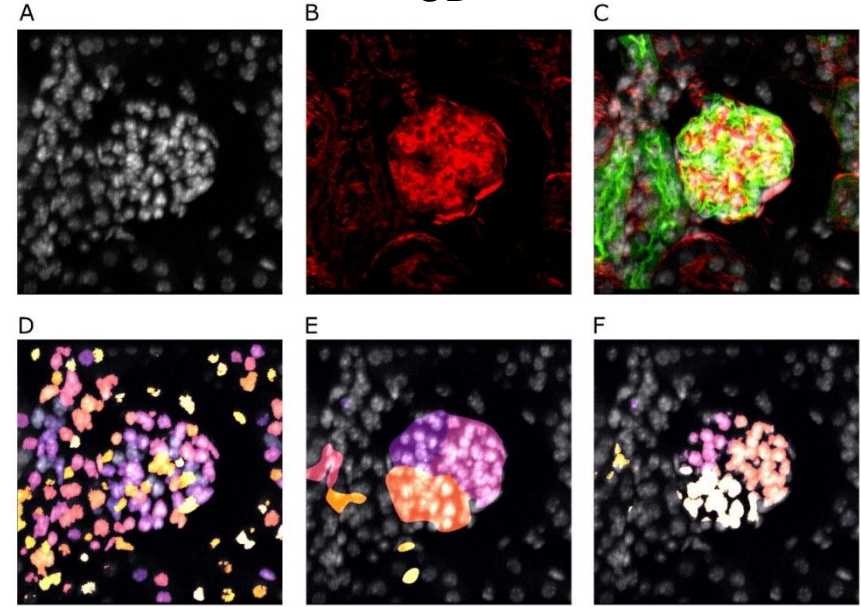
Benchmarked against ImageJ and CellProfiler

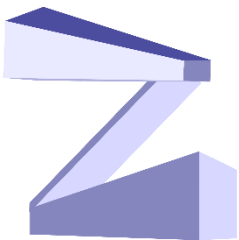
<https://github.com/RoccoDAnt/napari-zelda>

2D



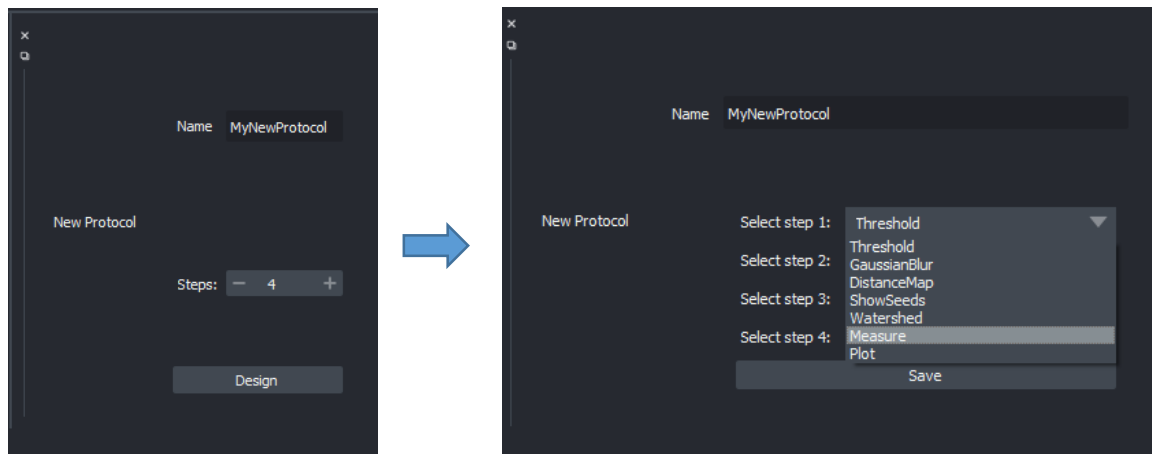
3D





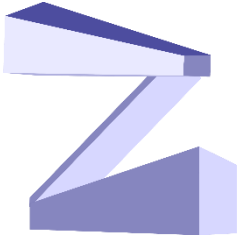
<https://github.com/RoccoDAnt/napari-zelda>

Workflow design and sharing



```
{
  "name": "MyNewProtocol",
  "widget": "MyNewProtocol_protocol_widget",
  "steps": [
    {
      "step_number": 1,
      "step_name": "Threshold"
    },
    {
      "step_number": 2,
      "step_name": "ShowSeeds"
    },
    {
      "step_number": 3,
      "step_name": "Watershed"
    },
    {
      "step_number": 4,
      "step_name": "Measure"
    }
  ]
}
```

Newly designed protocol (from .json file)



TRY IT OUT!

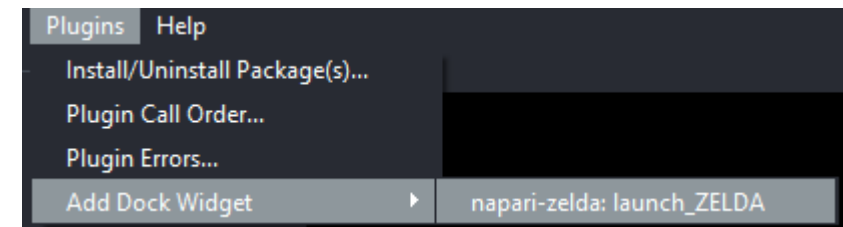
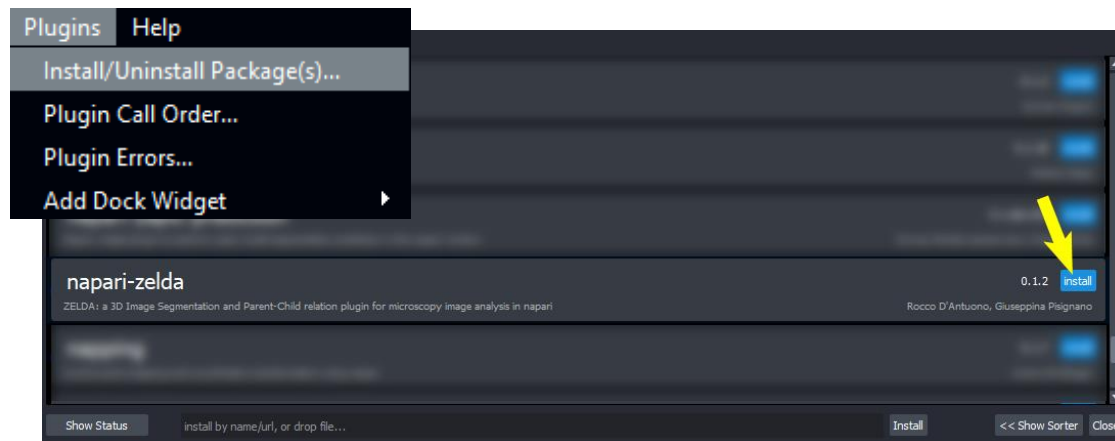
<https://github.com/RoccoDAnt/napari-zelda>

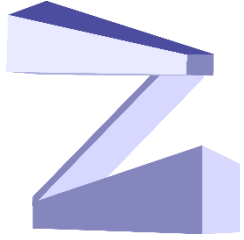
1. Install napari



```
conda create -y -n napari-env python=3.8
conda activate napari-env
pip install "napari[all]"
```

2. Install ZELDA plugin





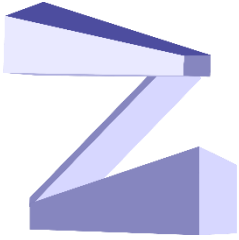
<https://github.com/RoccoDAnt/napari-zelda>

Acknowledgements:

- **Giuseppina Pisignano** (University of Bath)
- **Reviewers and Editors** for having asked many useful improvements
- First users: **Ana Stojiljković** (University of Bern), **Ewelina Bartoszek** (University of Basel), **Peter Carlton** (Kyoto University)

Read more:

D'Antuono R and Pisignano G (2022)
ZELDA: A 3D Image Segmentation and Parent-Child Relation
Plugin for Microscopy Image Analysis in *napari*.
Front. Comput. Sci. 3:796117. doi: 10.3389/fcomp.2021.796117



<https://github.com/RoccoDAnt/napari-zelda>

magicgui



TextEdit:	text edit value...		
FileEdit:	/home	Select file	
RangeEdit:	start 0	stop 10	step 2
SliceEdit:	start 0	stop 10	step 2
DateTimeEdit:	31 Dec 1999 11:30:00		
DateEdit:	14 Sep 1752		
TimeEdit:	12:20:00		

<https://napari.org/magicgui/>

References

```

full_name [Napari Developer]: Ramon y Cajal
email [yourname@example.com]: ramon@cajal.es
github_username_or_organization [githubuser]: neuronz52
# NOTE: for packages whose primary purpose is to be a napari
plugin, we
# recommend using the 'napari-' prefix in the package name.
# If your package provides functionality outside of napari, you
may
# choose to leave napari out of the name.
plugin_name [napari-foobar]: napari-growth-cone-finder
module_name [growth_cone_finder]:
napari_growth_cone_finder
short_description [A simple plugin to use with napari]:
# you can select from various plugin template examples
include_reader_plugin [y]:
include_writer_plugin [y]:
include_dock_widget_plugin [y]:
include_function_plugin [y]:
    
```

```

Select docs_tool:
1 - mkdocs
2 - sphinx
3 - none
Choose from 1, 2, 3 [1]: 3
Select license:
1 - BSD-3
2 - MIT
3 - Mozilla Public License 2.0
4 - Apache Software License 2.0
5 - GNU LGPL v3.0
6 - GNU GPL v3.0
Choose from 1, 2, 3, 4, 5, 6 (1, 2, 3, 4, 5, 6) [1]:
INFO:post_gen_project:Moving files for mkdocs.
    
```

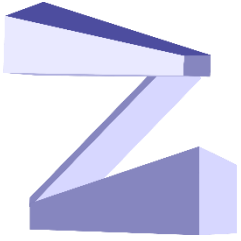
<https://github.com/napari/cookiecutter-napari-plugin>



The cornucopia of open-source software



<https://pypi.org/>



Additional slide on magicgui

<https://github.com/RoccoDAnt/napari-zelda>

Graphical User Interface:

- magicgui

- cookie-cutter plugin

- pypi.org

```
63 @magicgui(labels=False,  
64     label={'widget_type': 'Label', 'value': "Threshold"},  
65     Otsu={'widget_type': 'CheckBox', 'name': 'Otsu_threshold'},  
66     threshold={'widget_type': 'FloatSlider', "max": 65535.0, 'min': 0.0},  
67     call_button="Apply",  
68     persist=True  
69 )  
70 def threshold_one_pop(viewer: 'napari.Viewer', label, Otsu, layer: Image, threshold: int = 1)-> napari.types.ImageData:  
71     if layer:  
72         if Otsu==True:  
73             threshold = skimage.filters.threshold_otsu(np.array(layer.data))  
74             print(threshold)  
75             th=layer.data>threshold  
76             viewer.add_image(th, scale=layer.scale, name='Threshold th='+str(threshold)+' of '+str(layer.name))  
77
```