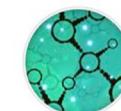


# BioImage-IT

Integration of data-management with analysis

Sylvain Prigent, Ludovic Leconte,  
César Augusto Valades Cruz, Léo Maury, Charles Kervrann, Jean Salamero  
STED/Serpico team (Curie/Inria)



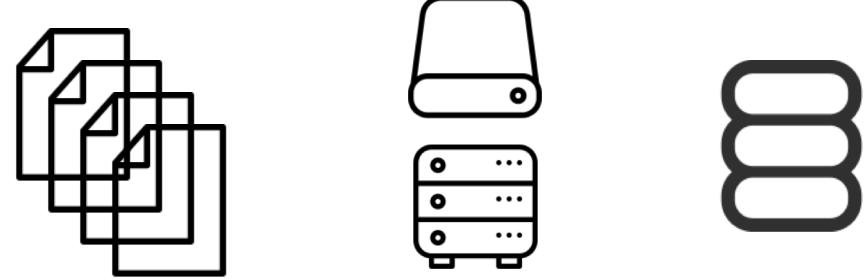
FRANCE-BIOIMAGING

Why BioImage-IT ?

# Data management and analysis is tedious...

## Problem 1: Data management

- Hard drives: storage without metadata
- Servers: storage without metadata
- data traceability ?



## Problem 2: Processing & visualization tools

- Many languages
- For different systems (Linux, Apple, Microsoft)
- Documentations & test dataset ?

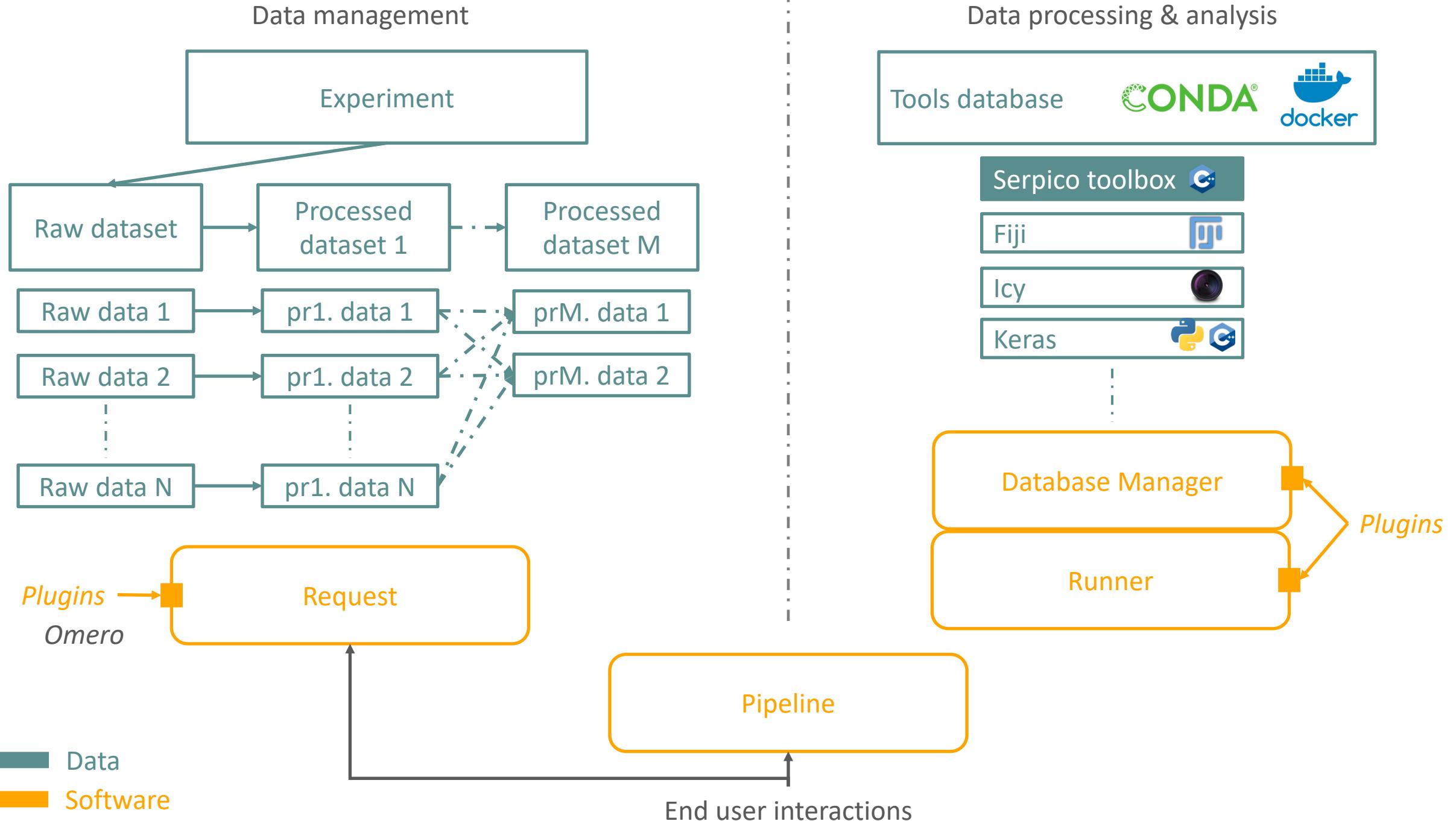


## Problem 3: link data to processing & visualization

- Manual scripting
- Systems configurations (ex: send data to cluster)



# What is Biolimage-IT ?



# User interface

## Home

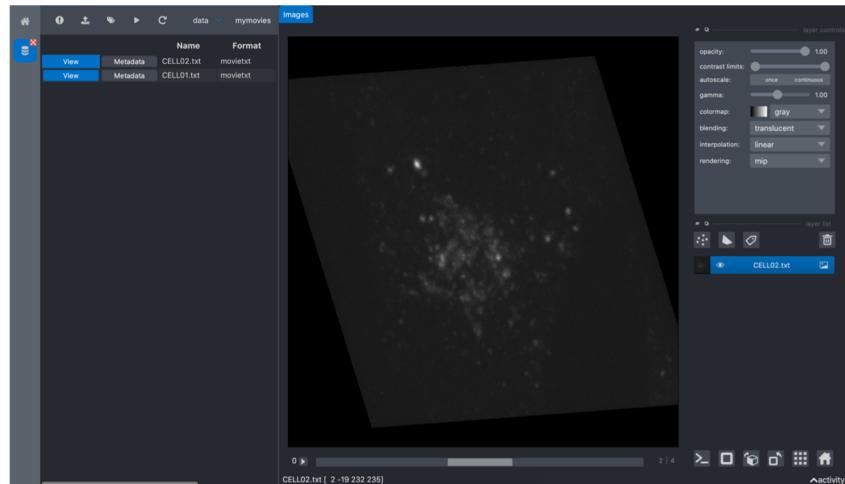
The screenshot shows the home interface of a software application. At the top, there is a navigation bar with a house icon, three dark grey buttons labeled "New experiment", "Toolboxes", and "Browse", and a "Home" button. Below the navigation bar is a section titled "EXPERIMENTS" containing a table with four rows:

	Name	
1	Tutorial experiment	2020-11-21
2	createtest	2021-11-01
3	myexperiment	2021-10-3

On the right side of the experiments table, there is a "Author" column with the value "sprigent". A modal dialog box titled "CREATE EXPERIMENT" is open in the center of the screen. It contains fields for "Destination" (set to "imageIT/workspace"), "Experiment name" (empty), and "Author" (set to "sprigent"). A "Create" button is at the bottom of the dialog.

# User interface

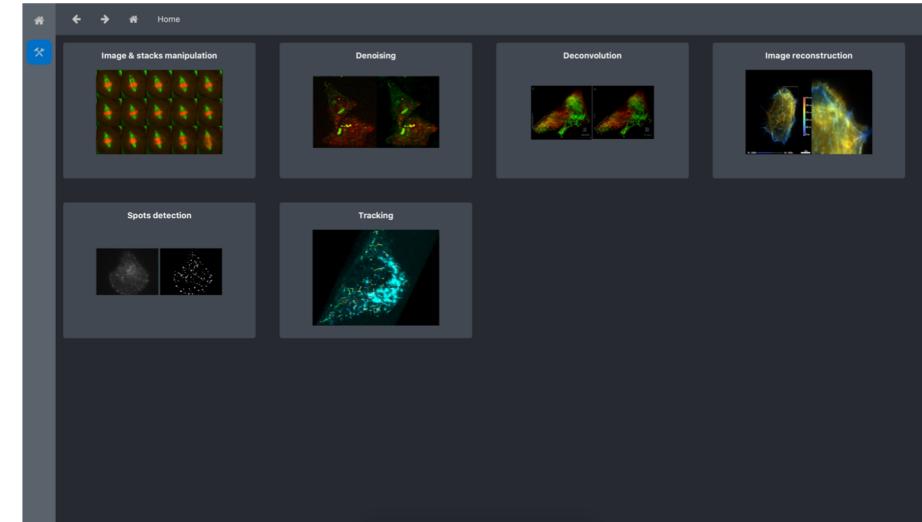
## Experiment editor



## Tool documentation

A screenshot of the ND-SAFIR Denoising tool documentation page. The left sidebar lists several tutorials with their names and versions, such as 'noise2void' (0.2.1), 'Median 4D' (0.1.3), 'Ndsafir 4D' (3.0.0), 'Median 2D' (0.1.3), 'Ndsafir 3D' (3.0.0), 'Ndsafir 2D' (3.0.0), 'Median 3D' (0.1.3), 'SPITFIR(e) 3D' (0.1.3), 'SPITFIR(e) 2D' (0.1.3), and 'SPITFIR(e) 4D' (0.1.3). The main content area features a section titled 'Overview' with two microscopy images showing green and red fluorescence. Below this, a detailed description of ND-SAFIR is provided: "ND-SAFIR is a software for denoising n-dimentionnal images especially dedicated to microscopy image sequence analysis. It is able to deal with 2D, 3D, 2D+time, 3D+time images have one or more color channel. It is adapted to Gaussian and Poisson-Gaussian noise which are usually encountered in photonic imaging. Several papers describe the detail of the method used in ndsafr to recover noise free images (see references)." A 'Tutorial' button is also visible in the sidebar.

## Toolboxes finder



## Tool runner

A screenshot of the Tool runner interface. On the left, there's a 'File' menu, a 'Folder' button, and an 'Experiment' button. Below these are sections for 'Inputs' (with an 'Input Image' field set to '14um\_120.tif'), 'Parameters' (with fields for Noise, Patch, Noise factor, Time series, and Iterations), and 'Exec' (with a 'Run' button). A 'Progress' bar at the bottom indicates the process is at 100% completion, having started at 0% and processed 1/1 data. The main area shows a microscopy image of cells with green and red fluorescence, and on the right, there are 'layer controls' for opacity, contrast limits, and gamma, along with colormap, blending, interpolation, and rendering options. At the bottom, there's a progress bar labeled 'o.CELL01\_HELA\_MPR42\_GFP\_20ms\_0s\_22 steps\_0.4um\_120.tif [-86 216 69]' and a toolbar with various icons.

# Developers tools

Interact with the python API

```
1 from bioimagepy.experiment import Experiment
2
3 ## create an experiment
4 my_experiment = Experiment()
5 my_experiment.create(name="myexperiment",
6                      author="Sylvain Prigent",
7                      uri="./")
8
9 print("experiment created")
```

```
1 # import the data
2 my_experiment.import_dir(dir_uri='./synthetic_data/data/',
3                           filter=".tif$",
4                           author='Sylvain Prigent',
5                           format='tif',
6                           date='2019-03-17',
7                           copy_data=True)
8
9 # show the experiment directory
10 list_files('./myexperiment')
```

## Pipeline

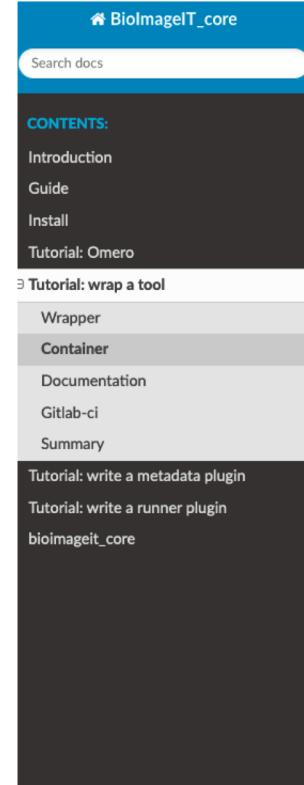
```
1 from bioimagepy.experiment import Experiment
2 from bioimagepy.pipeline import Pipeline
3
4 ConfigAccess('../bioimagepy/config_sample.json')
5 my_experiment = Experiment('../userdata/myexperiment')
6 pipeline = Pipeline(my_experiment)
```

```
1 process1 = pipeline.add_process(ProcessAccess().get('svdeconv2d_v0.1.0'))
2 process1.set_parameters('sigma', 2, 'regularization', 2, 'weighting', 0.1, 'method', 'SV')
3 process1.set_dataset_name('deconv')
4 process1.add_input('i', 'data')
5 process1.run()
```

```
1 process2 = pipeline.add_process(threshold_particles)
2 process2.set_parameters('threshold', 'Default dark')
3 process2.set_dataset_name('particles')
4 process2.add_input('input', 'deconv')
5 process2.run()
```

```
1 process3 = pipeline.add_process(ProcessAccess().get('Wilcoxon_v1.0.0'))
2 process3.set_dataset_name('wilcoxon')
3 process3.add_input('x', 'particles', 'Population=population1', 'count')
4 process3.add_input('y', 'particles', 'Population=population2', 'count')
5 process3.run()
```

Integrate your own tools



The tool wrapper is in the file called `sampletool.xml`. Its content is shown below. It is a classical Galaxy Project wrapper. The command is `meanfilter` and take three arguments, the input image `-i`, the output image `-o` and one parameter `-r` the radius of the filter. The two important information in this wrapper are the `<requirements>` section that contains the address of the Docker image container that we will be generated with the `gitlab-ci`, and the address of the documentation in `<help>` that will be generated by the `gitlab-ci` pages.

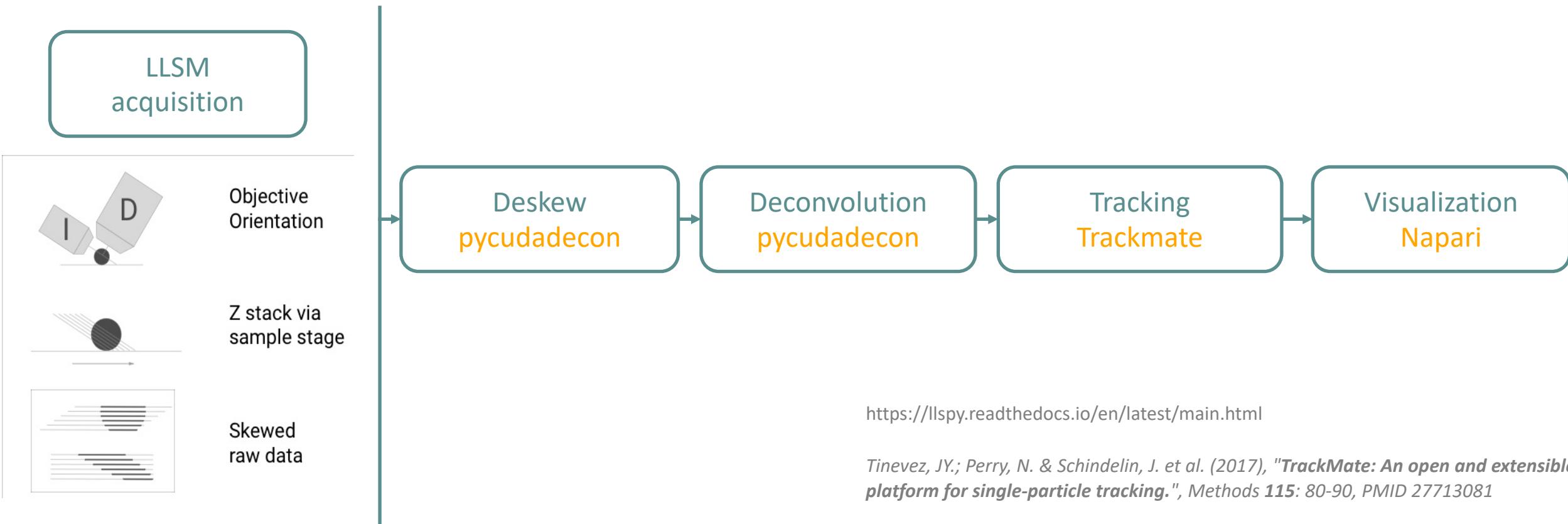
```
<tool id="sampletool" name="SampleTool" version="0.1.0" python_template_version="3.5">
  <requirements>
    <container type="docker">registry.gitlab.inria.fr/
      bioimage-it/sampletool:766efe8b8397fef0115e20f...
    </container>
  </requirements>
  <command detect_errors="exit_code"><![CDATA[
    meanfilter -i ${i} -o ${o} -r ${r}
  ]]></command>
  <inputs>
    <param type="data" name="i" format="tiff" label="Input Image" help="2D image" />
    <param argument="-r" type="integer" value="3" label="Radius" help="Filter radius (in pixels)" />
  </inputs>
  <outputs>
    <data name="o" format="tiff" label="Output image" />
  </outputs>
  <tests>
    <test>
      <param name="i" value="celegans.tif" />
      <param name="r" value="3" />
      <output name="o" file="celegans_result.tif" compare="sim_size" />
    </test>
  </tests>
  <help><![CDATA[
    https://bioimage-it.gitlabpages.inria.fr/sampletool/
  ]]></help>
  <citations>
  </citations>
</tool>
```

Use case:  
Lattice light sheet image analysis pipeline

# LLSM original pipeline

Application: 3D+t tracking of endosomes using Lattice Lightsheet microscopy

Pipeline: needs at least 3 different software



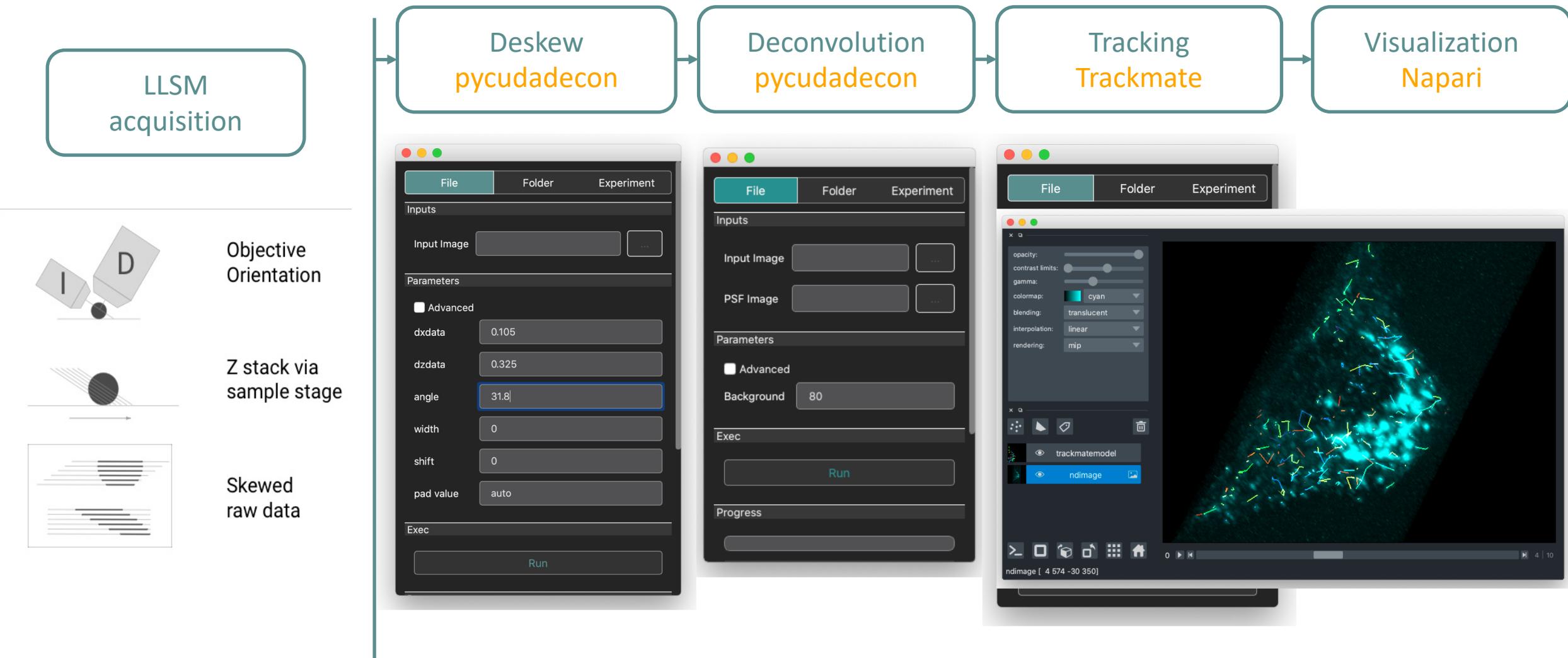
<https://llspy.readthedocs.io/en/latest/main.html>

Tinevez, JY.; Perry, N. & Schindelin, J. et al. (2017), "TrackMate: An open and extensible platform for single-particle tracking.", Methods 115: 80-90, PMID 27713081

napari contributors (2019). napari: a multi-dimensional image viewer for python.  
doi:10.5281/zenodo.3555620

# LLSM BiolImage-IT pipeline

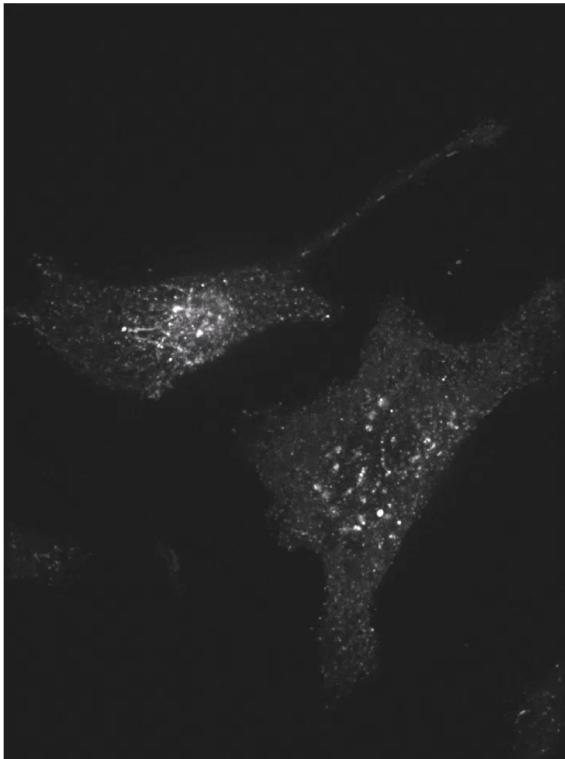
Pipeline: with BiolImage-IT -> interact with a single interface



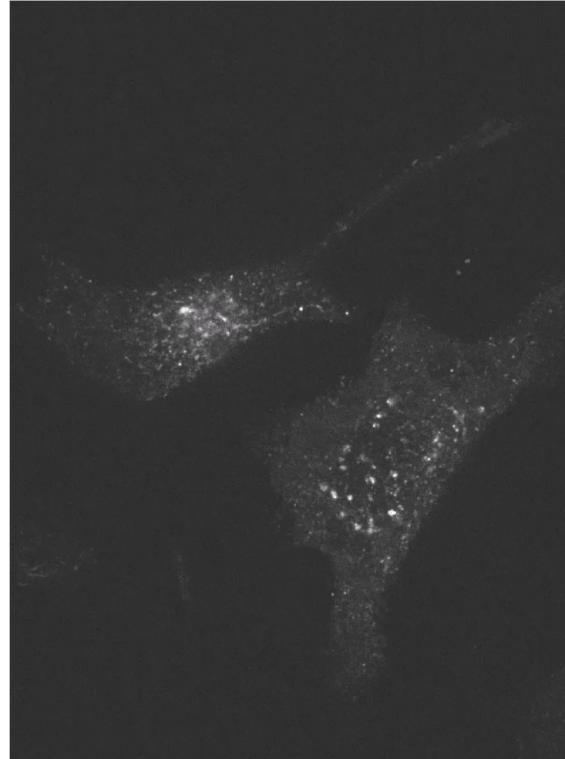
# Sted/Serpico toolbox

# Understood the complexity of the spots

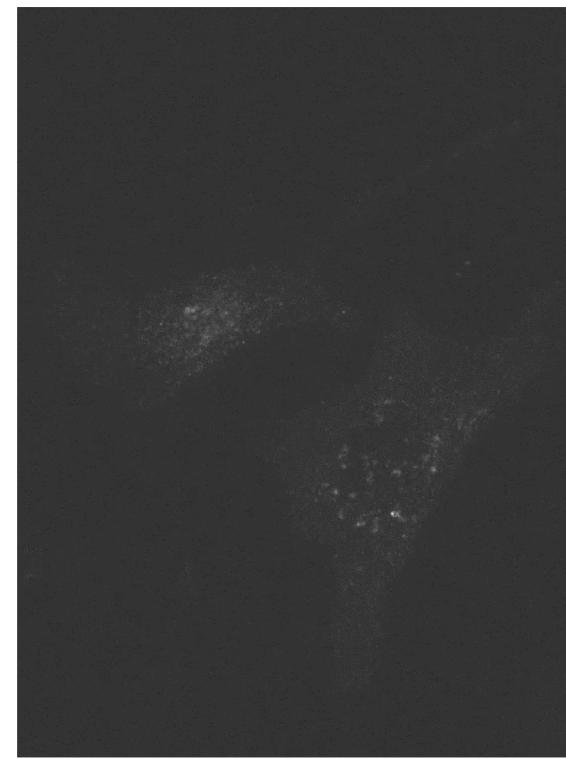
Normal power laser



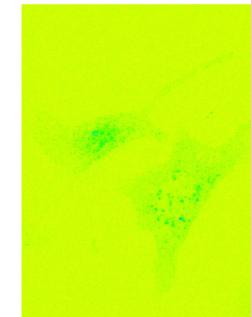
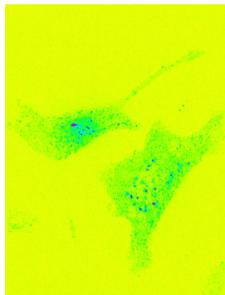
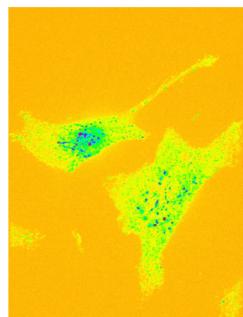
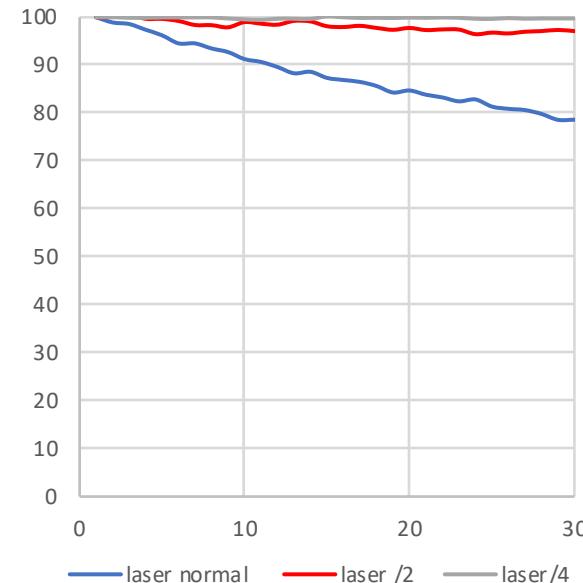
Laser divided by 2



Laser divided by 4



Photobleaching comparison



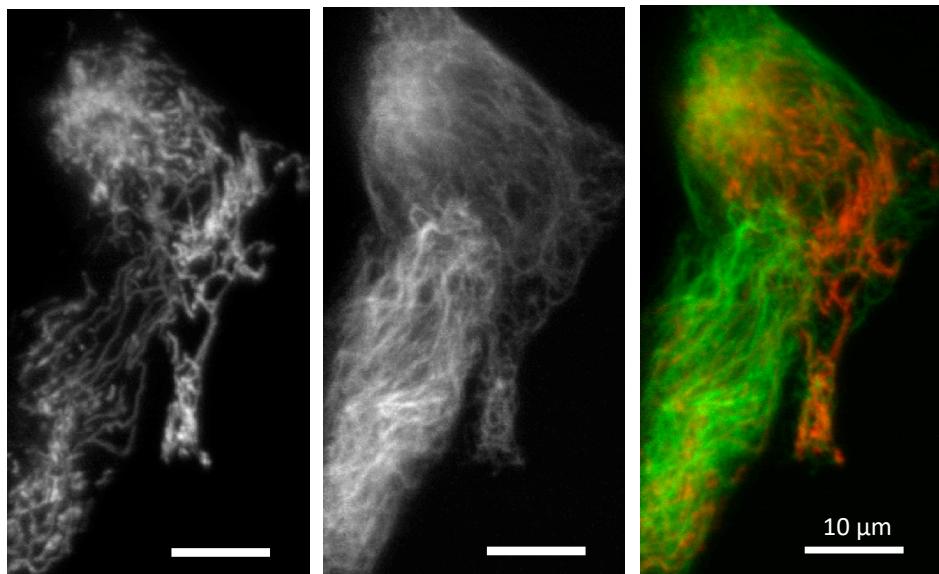
information:  
Spinning3 HEla M6PR eGFP  
Steps : 26 @0.3µm  
30 times points  
@10ms

# SPITFIR(e): deconvolution and denoising 2D, 3D, 3D+t

Smooth and sparse model dedicated to light microscopy:

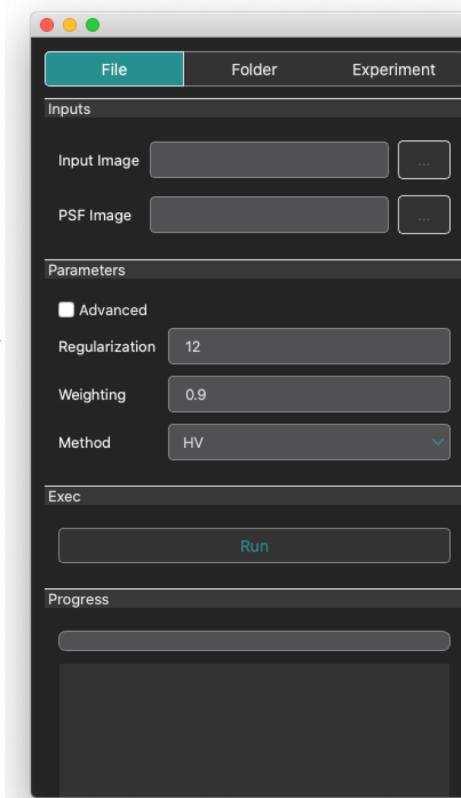
$$\text{HSV}_\rho(u) = \int_{\Omega} \underbrace{\sqrt{\rho^2 \|\mathcal{H}u(\mathbf{x})\|_F^2 + (1-\rho)^2 u(\mathbf{x})^2}}_{\|D_{2,\rho} u(\mathbf{x})\|_2} d\mathbf{x}$$

LLSM acquisition  
+ Deskew

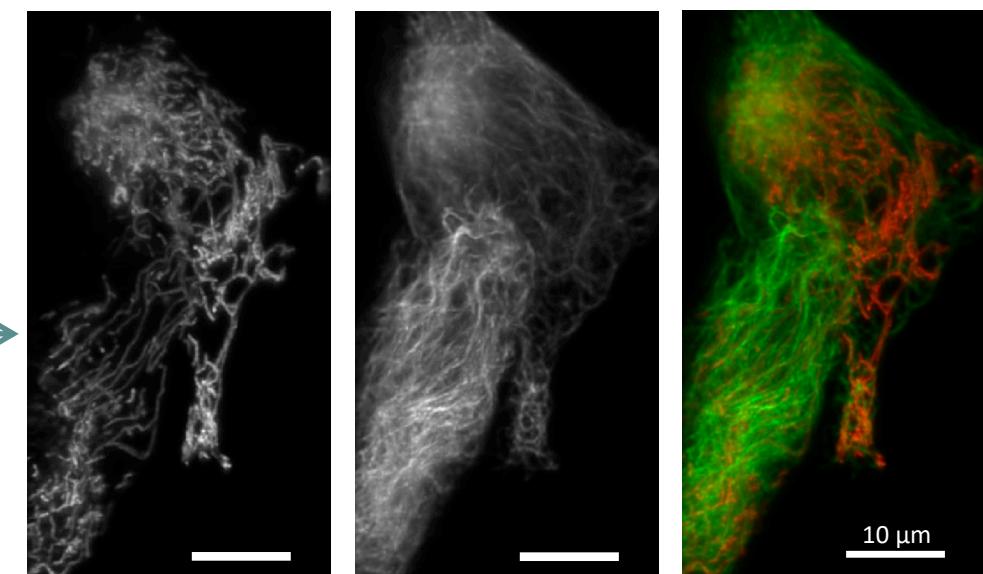


Microtubules (green) vs Mitochondria (red)  
2 s/volume  
RPE1 Cells

BioImage IT  
SPITFIR(e)



Result



Deskew+  
4D denoising+  
3D Deconvolution

# Resources

## Website

- [bioimageit.github.io](https://bioimageit.github.io)

The screenshot shows the BioImageIT website's tutorial section. It is divided into two main sections: "Using the graphical interface" and "Using the python library".

- Using the graphical interface:** Contains four cards:
  - Create an experiment
  - Browse the toolboxes
  - Run a data processing tool
  - Design a data processing pipeline
- Using the python library:** Contains three cards:
  - Create an experiment
  - Run a data processing tool
  - Design a pipeline

At the bottom, it says: "The python library tutorials are available as Jupyter notebooks [here](#)".

The screenshot shows a Jupyter notebook titled "tutorial1-experiment". The notebook has a header bar with "jupyter", "tutorial1-experiment", "Dernière Sauvegarde : 19/11/2020 (modifié)", "Se déconnecter", "Fichier", "Édition", "Affichage", "Insérer", "Cellule", "Noyau", "Widgets", and "Aide". Below the header is a toolbar with various icons.

**Create an experiment**

To create an *Experiment* we need first to import the `Experiment` class from `bioimageit.experiment`. Then we call the method `create` to create the *Experiment*. The three parameters are:

1. `name`: the name of the experiment
2. `author`: the person who create the experiment
3. `uri`: the directory where the experiment will be create

**Entrée [2]:**

```
from bioimageit_core.experiment import Experiment
## create an experiment
my_experiment = Experiment()
my_experiment.create(name="myexperiment",
                     author="Sylvain Prigent",
                     uri="./")
print("experiment created")
experiment created
```

The *Experiment* has been created in the same directory as this notebook in the folder `./myexperiment`. The function "list\_files" below shows the actual content of the experiment folder:

**Entrée [3]:**

```
import os
def list_files(startpath):
    for root, dirs, files in os.walk(startpath):
        level = root.replace(startpath, '').count(os.sep)
        indent = ' ' * 4 * (level)
        print('{0}/{1}'.format(indent, os.path.basename(root)))
        subindent = ' ' * 4 * (level + 1)
        for f in sorted(files):
            print('{0}{1}'.format(subindent, f))
```

## Codes repositories

- [github.com/bioimageit](https://github.com/bioimageit)
- <https://gitlab.inria.fr/serpico>

# Hands-on

2D, 3D, 3D+t Image denoising

# Acknowledgments

## Fundings

- France-BioImaging
- EIT-Digital
- Airbus - BPI

## Sted/Serpico team

- Jean Salamero
- Charles Kervrann
- Ludovic Leconte
- César Augusto Valades Cruz
- Léo Maury

## Collaborators

### Inria Mosaic team – ENS Lyon

- Guillaume Cerutti
- Jonathan Legrand

### Institut Curie - *eLabFTW*

- Nicolas Carpi

### Institut Curie – *DSI (ongoing)*

- Alexandre Fargeot

