MISA++: A standardized interface for automated bioimage analysis

Ruman Gerst[1,2], Anna Medyukhina[1], and Marc Thilo Figge[1,3]

[1] Applied Systems Biology, Leibniz Institute for Natural Product Research and Infection Biology – Hans-Knöll-Institute, Jena, Germany
[2] Faculty of Biological Sciences, Friedrich-Schiller-University Jena, Germany
[3] Institute of Microbiology, Faculty of Biological Sciences, Friedrich-Schiller-University Jena, Germany

1. Analysis of big volume image data

Modern imaging techniques, such as lightsheet fluorescence microscopy (LSFM), allow the capture of whole organs in three spatial dimensions. This includes the evaluation of structural changes such as a reduced number of glomeruli in kidneys [2] or the formation of bronchus-associated lymphoid tissue (BALT) caused by lung inflammation [3]. The analysis of these big volume image data requires a combination of user-friendly and highly efficient tools.

High-performance tools
e.g. C++ / OpenCV

- ⊕ Fast runtime
- ⊕ Full control over hardware
- ⊖ Complex code syntax
- ⊖ Hard to debug
- ⊖ Need compilation
- ⊖ Hard to integrate

User-friendly tools
e.g. ImageJ

- ⊕ Easy to program and debug
- ⊕ Graphical user interfaces
- ⊕ Easy to integrate
- ⊖ Slow runtime
- ⊖ Limited optimization
- ⊖ Limited server support

Idea: Make high-performance tools easy to integrate into user-friendly tools via our framework [1].

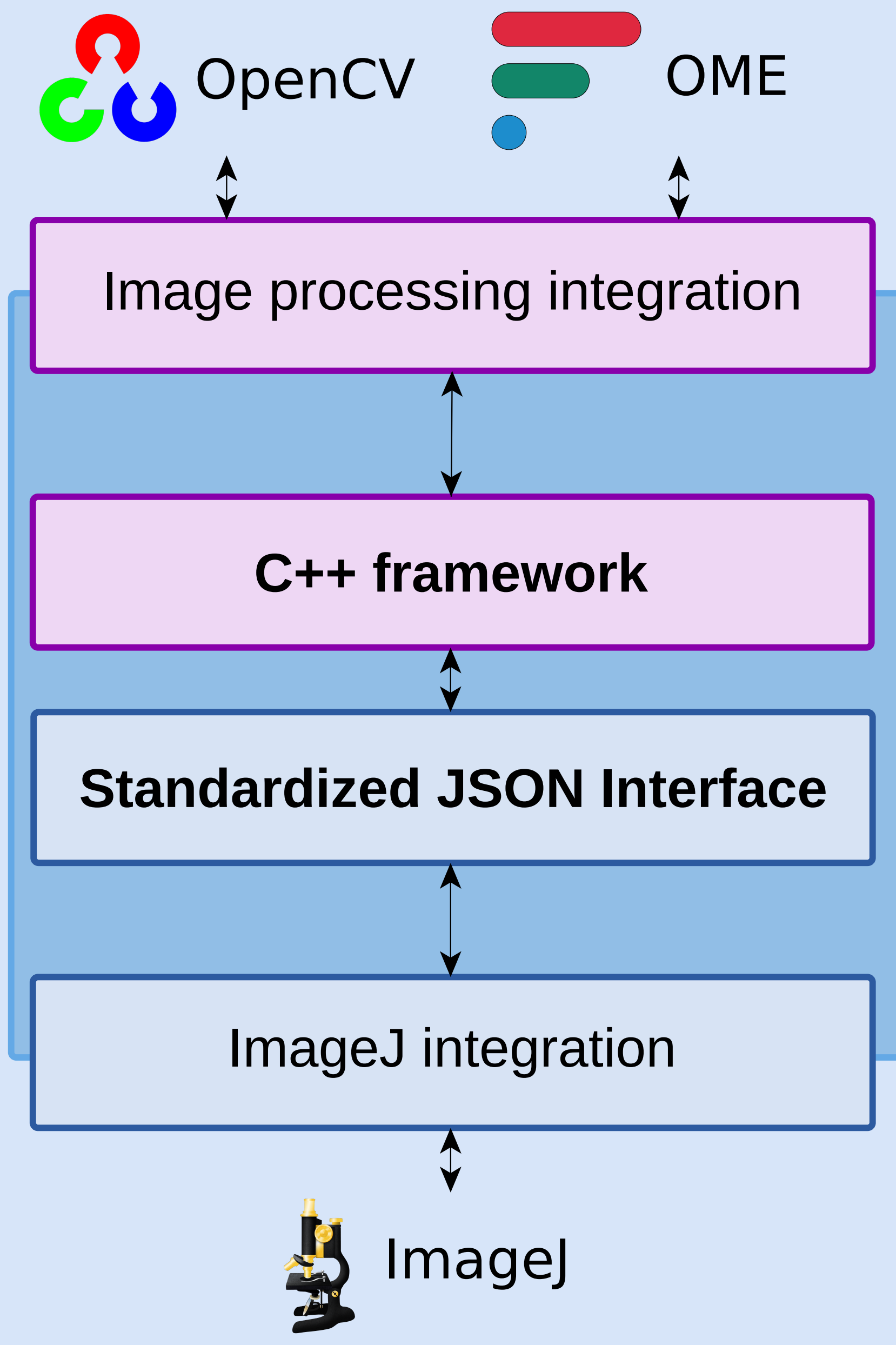
- ☑ High performance
- ☑ Easy to use for non-developers

2. Framework

MISA++
Modular Image Stack Analysis for C++

Available on GitHub https://github.com/applied-systems-biology/misa-framework

Modular **I**mage **S**tack **A**nalysis for C**++** (MISA++) is a platform-independent open source framework that simplifies developing efficient image analysis tools. It provides standardized data and parameter handling, parallelization, documentation, and integration into user-friendly software via an ImageJ plugin.



High-performance tools

- ☑ CPU and memory-efficient
- ☑ Runs on servers

Our framework

- ☑ Automated parallelization
- ☑ Memory management
- ☑ Standardized parameters
- ☑ Standardized input/output
- ☑ Standardized documentation

User friendly tools

- ☑ Easy data analysis
- ☑ Graphical interfaces

3. Glomeruli segmentation

Glomeruli are functional structures within the renal cortex that are damaged by diseases and toxins [1]. Our algorithm consists of five steps that first segments the tissue and then extracts the glomeruli.

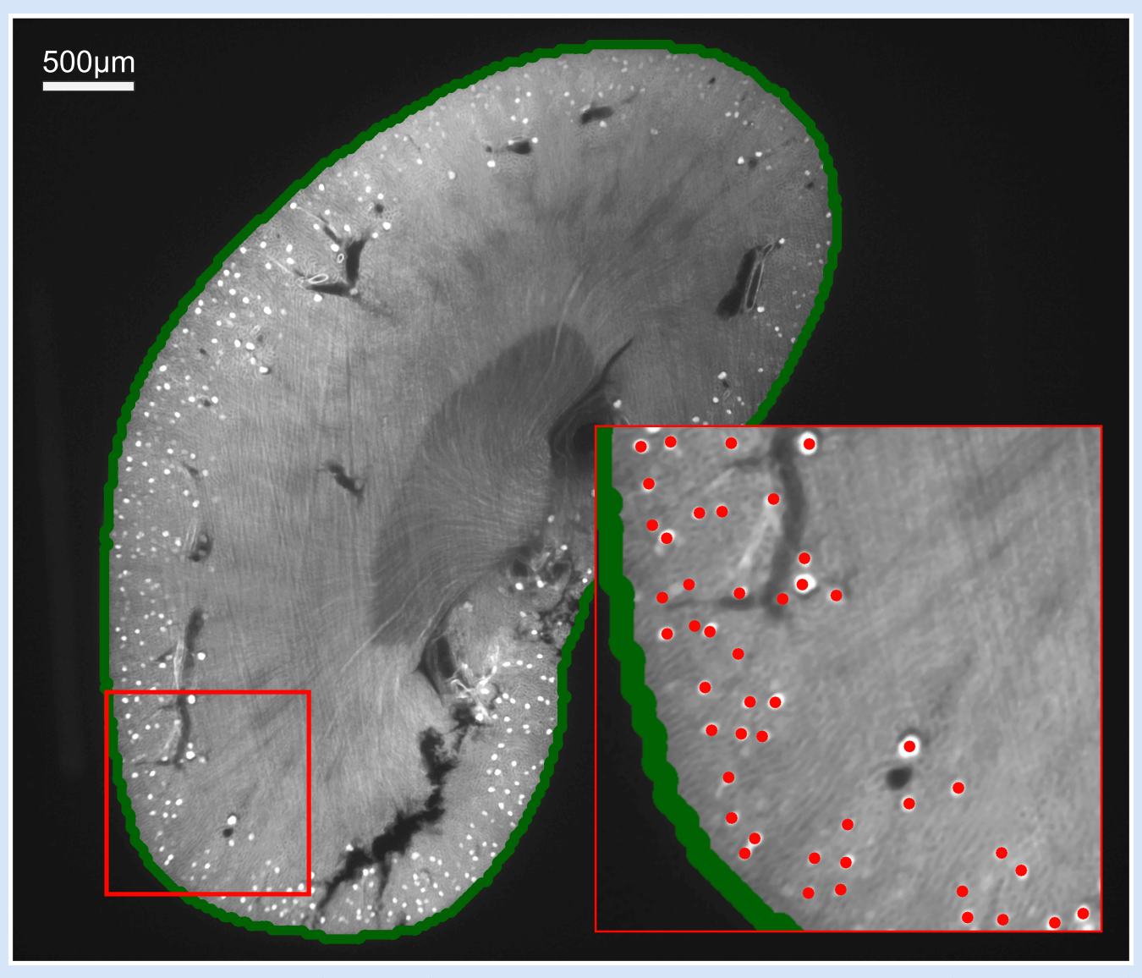
a. 2D tissue segmentation
Percentile thresholding

b. Tissue quantification
Number of pixels, volume

c. 2D glomeruli segmentation
TopHat background removal, Otsu thresholding

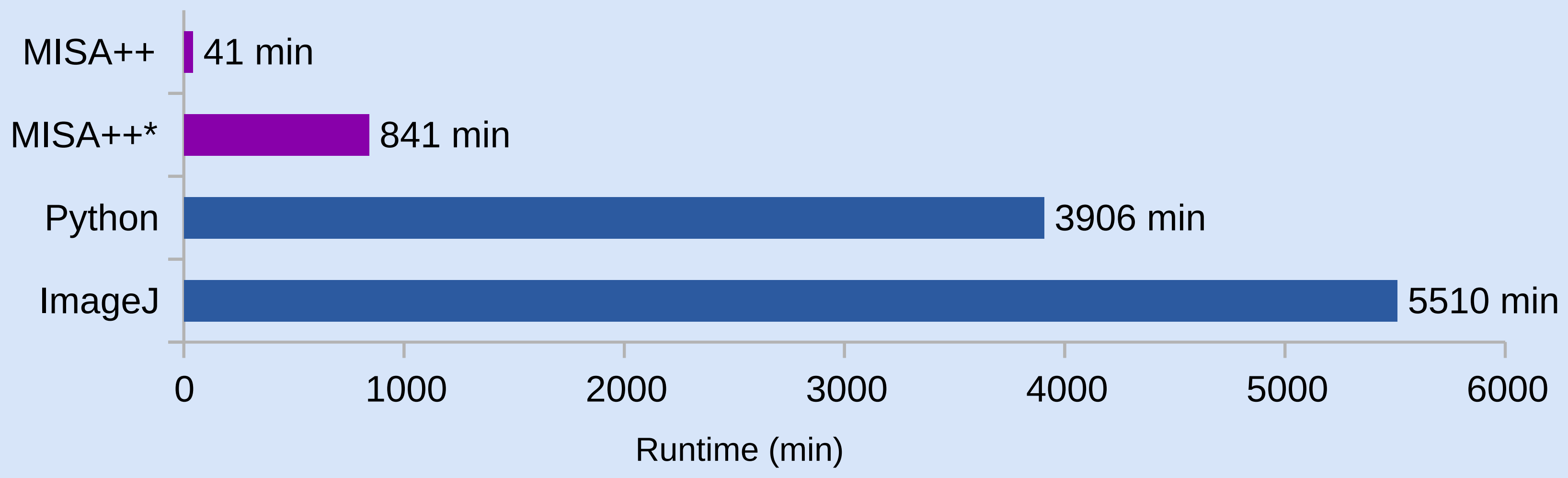
d. 3D glomeruli reconstruction
3D constrained connected components

e. Glomeruli quantification
Number of glomeruli, volume, diameter



Glomeruli (red) inside the tissue (green)

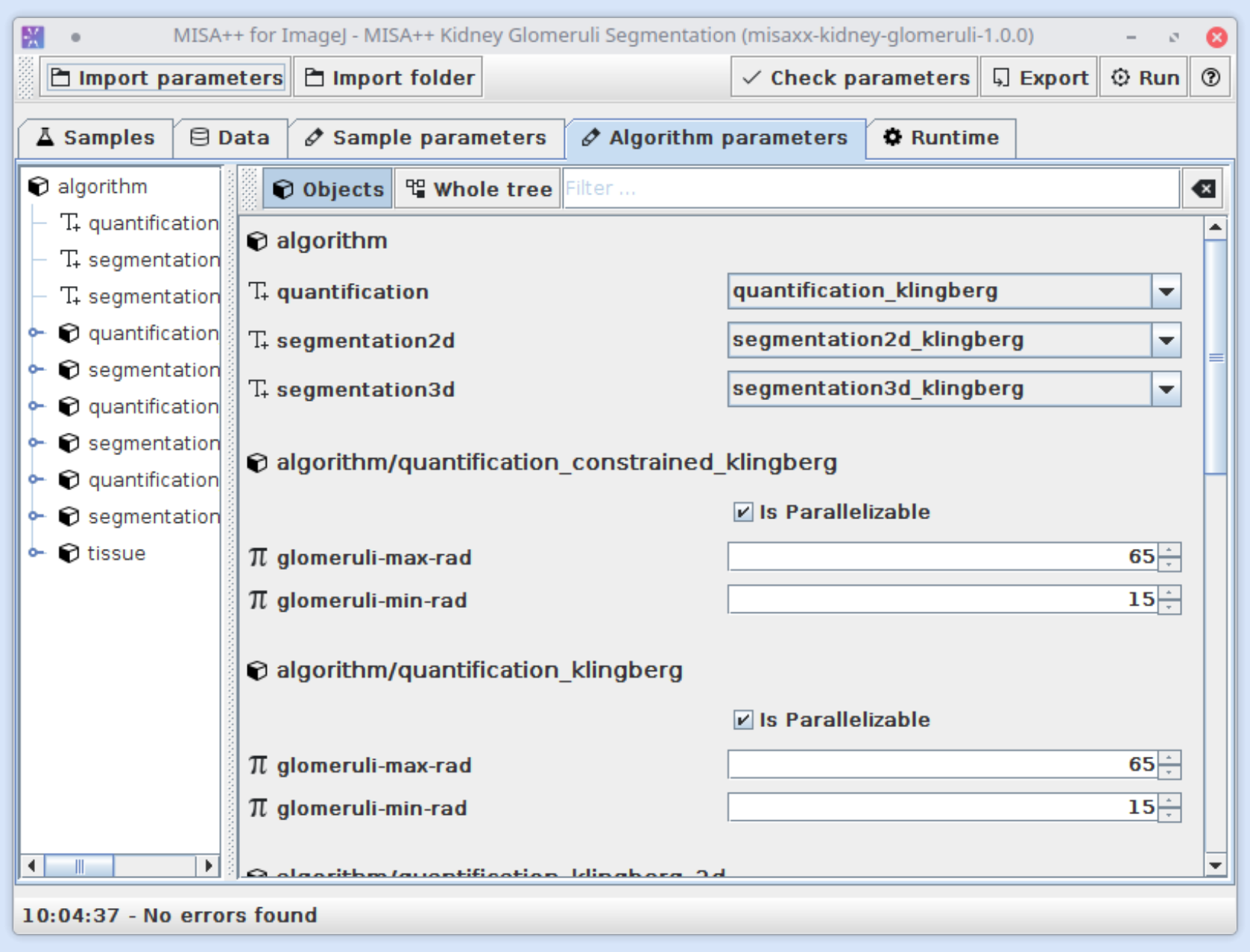
The original implementation is written in the Python language. By creating an advanced MISA++-based implementation, we could reduce the analysis runtime for the same data set on the same hardware by two orders of magnitude.



Compared to the implementation published by Klingberg et al. [1] (Python), MISA++ calculates the same work up to 95 times faster if all images are available at the same time. If the analysis is done one after another kidney (*), the calculation is still approximately 4 times faster. MISA++ was 134 times faster than a implementation in ImageJ. Parallelization using 30 threads.

4. ImageJ integration

We developed a plugin for ImageJ that integrates tools created with MISA++ into ImageJ. Users can setup and monitor analyses, and display results within a easy-to-use graphical user interface.



MISA++ for ImageJ - MISA++ Kidney Glomeruli Segmentation (misaxx-kidney-glomeruli-1.0.0)

algorithm
quantification: quantification_klingberg
segmentation2d: segmentation2d_klingberg
segmentation3d: segmentation3d_klingberg

algorithm/quantification_constrained_klingberg
☑ Is Parallelizable
glomeruli-max-rad: 65
glomeruli-min-rad: 15

algorithm/quantification_klingberg
☑ Is Parallelizable
glomeruli-max-rad: 65
glomeruli-min-rad: 15

10:04:37 - No errors found

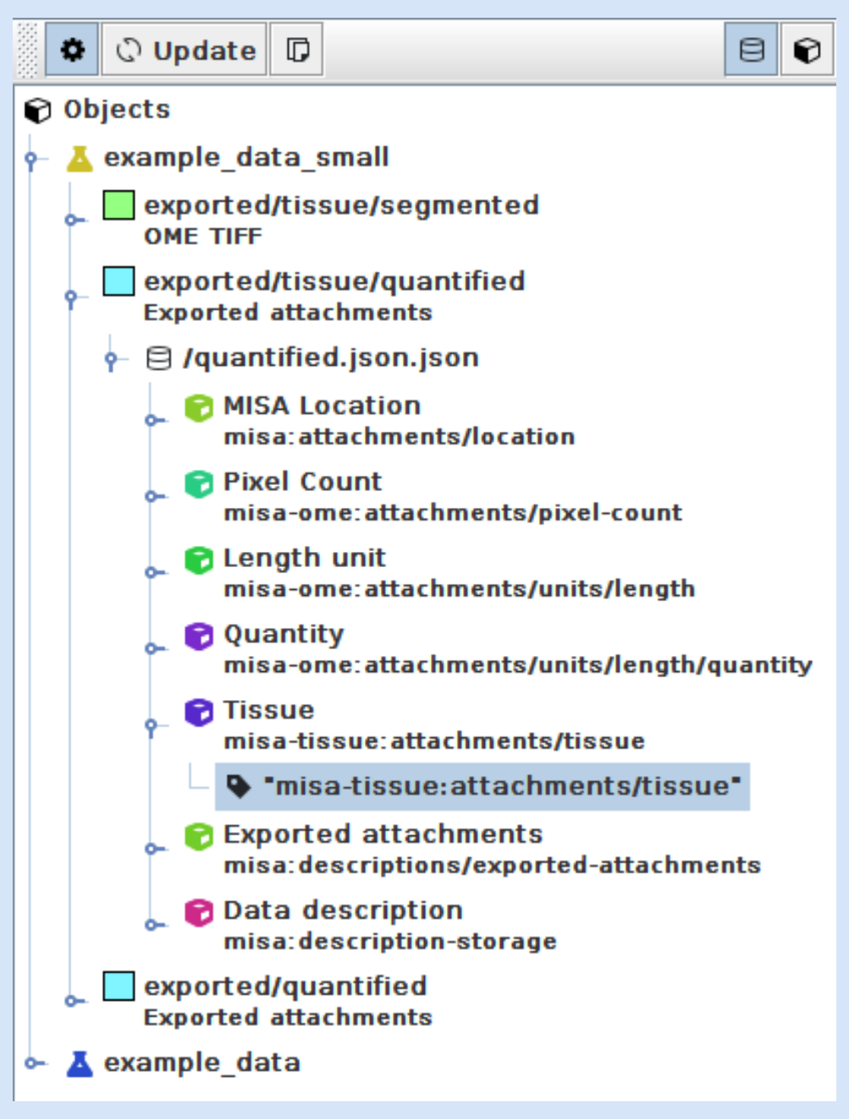
Features

- ☑ Manage MISA++ applications
- ☑ Import input data
- ☑ Change algorithm parameters
- ☑ Run analyses
- ☑ Display analysis results
- ☑ Create plots & tables
- ☑ Export analysis tasks
- ☑ Detailed runtime statistics
- ☑ Create pipelines

All features are available for any MISA++ application, made possible by the machine-readable documentation. All user interfaces are automatically generated.

In future, the ImageJ integration will be enhanced by upgrading it to JIPipe (**https://www.jipipe.org/**).

JIPipe



Objects
example_data_small
exported/tissue/segmented
OME TIFF
exported/tissue/quantified
Exported attachments
/quantified.json.json
MISA Location
misa:attachments/location
Pixel Count
misa-ome:attachments/pixel-count
Length unit
misa-ome:attachments/units/length
Quantity
misa-ome:attachments/units/length/quantity
Tissue
misa-tissue:attachments/tissue
"misa-tissue:attachments/tissue"
Exported attachments
misa:descriptions/exported-attachments
Data description
misa:description-storage
exported/quantified
Exported attachments
example_data

ruman.gerst@leibniz-hki.de

www.leibniz-hki.de

References

[1] Gerst *et al.* 2020. *SoftwareX*. 11: 100405.
[2] Klingberg *et al.* 2017. *JASN*. 28(2): 452–59.
[3] Mzinza *et al.* 2018. *Cell Mol Immunol*. 15: 875–887

ILRS JENA MIBINTACT
FRIEDRICH-SCHILLER-UNIVERSITÄT JENA