An automated workflow for parallel processing of large multiview SPIM recordings

Christopher Schmied1, Peter Steinbach1, Tobias Pietzsch1, Stephan Preibisch1,2,3, Pavel Tomancak1

1 Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany

2 HHMI Janelia Research Campus, Ashburn, Virginia, USA

3 Max Delbrück Center for Molecular Medicine, Berlin Institute for Medical Systems Biology, Berlin, Germany.

**Supplementary Note**

The pipeline presented here is comparable in functionality and performance particularly to the pipeline developed by Fernando Amat at the Janelia Research Campus (Amat et al. 2015). While the Amat pipeline is faster owing to the full reliance on GPU processing, it is limited to processing of datasets generated using the SimView microscope (or similar) where precisely aligned orthogonal views are acquired. The Amat pipeline relies on Matlab R2013b and the following toolboxes: Curve Fitting, Image Processing, Statistics, Optimization, Signal Processing, Parallel Computing. This limits its deployment on a large computer cluster, which is nevertheless possible and has been implemented. The Amat pipeline replaces a previous version used in Tomer et al. 2012.

Additionally, several groups implemented dedicated fusion algorithms. Rubio Guivernau et al. 2012 introduces wavelet fusion running in MATLAB. It is not integrated into any SPIM processing pipeline and the source code cannot be found. Schmid and Huisken 2015 implemented a fast 2D approximation of 3D multi-view deconvolution in Fiji. In the near future it will be integrated into Multiview Reconstruction Fiji plugins and thus will be fully available to our cluster-processing pipeline.

Finally, Zeiss Microscopy GmbH implemented Preibisch et al. 2010 bead-based registration in their proprietary software along with custom fusion and multi-view deconvolution algorithms. Since the details of these algorithms are not publicly accessibly, we do not compare our approach to them in the context of an academic paper. They are however fully functional for processing of Lightsheet Z.1 data.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **paper** | **algorithm** | **software** | **HPC** | **license** | **datasets** | **processing times** | **advantages** | **disadvantages** |
| Preibisch et al. 2010, 2014,  Schmied et al. 2014  and  this manuscript | Registration  Fusion  Deconv.  Dataviz | Fiji plugins | Yes | GPL  ver. 2 | Drosophila Zebrafish Platynereis  Parhyale | See manuscript and Suppl. Tables 1 and 2 | Versatile, integrated, GUI, visualization via BigDataViewer (Pietzsch et al. 2015), automated parallel processing, deals with commercial data formats, GPU acceleration | Relies on beads embedded in the rigid mounting medium or the ability to segment blobs in the sample (via the Difference of Gaussian detector). |
| Amat et al. 2015 | Registration  Fusion  Deconv.  Dataviz | Matlab scripts | Yes | open source,  no license,  MATLAB  + toolboxes  licenses | Drosophila  Zebrafish  Parhyale  Platynereis | processes image data at a rate faster than the data acquisition rate of the light-sheet microscope, i.e. in the order of seconds. | Independent of embedding and beads, provides compression, uses web based CATMAID for visualization, fast GPU accelerated processing | Works specifically with orthogonal views of the SimView microscope, cannot register arbitratry views, i.e. Lightsheet Z.1 data and similar |
| Rubio Guivernau et al. 2012 | Fusion | Matlab scripts | No | no source available,  MATLAB license | Sea urchin  Zebrafish | Registration: 24 minutes Fusion : 64 minutes\* | Unique fusion method not available elsewhere | Lack of integration with other aspects of SPIM data processing.  No source code. |
| Schmid and Huisken 2015 | Deconv. | Fiji Plugin | No\*\* | GPL  ver. 3 | Zebrafish | 25 to 75 times faster compared to Preibisch et al. 2014 | Very fast plane-wise deconvolution, with GPU acceleration allows real time processing | Based on 2D approximation of 3D PSF, works only on systems where the rotation axis lies within the image plane |
| ZEN Black | Registration  Fusion  Deconv.  Dataviz | Commercial | No | hardware  or  software license key | Drosophila, Zebrafish, Parhyale  Platyneries | Commercial implementation of registration algorithm in Preibisch et al. 2010, custom multiview deconvolution | Applicable to Lightsheet Z.1 data, fully integrated | Closed. |

\* The algorithm was benchmarked against algorithms in Preibisch et al. 2010 which performed registration in 1.8 minutes and fusion in 1.5 minutes on the same data.

\*\*Parallelization can in principle be implemented.**Supplementary References**

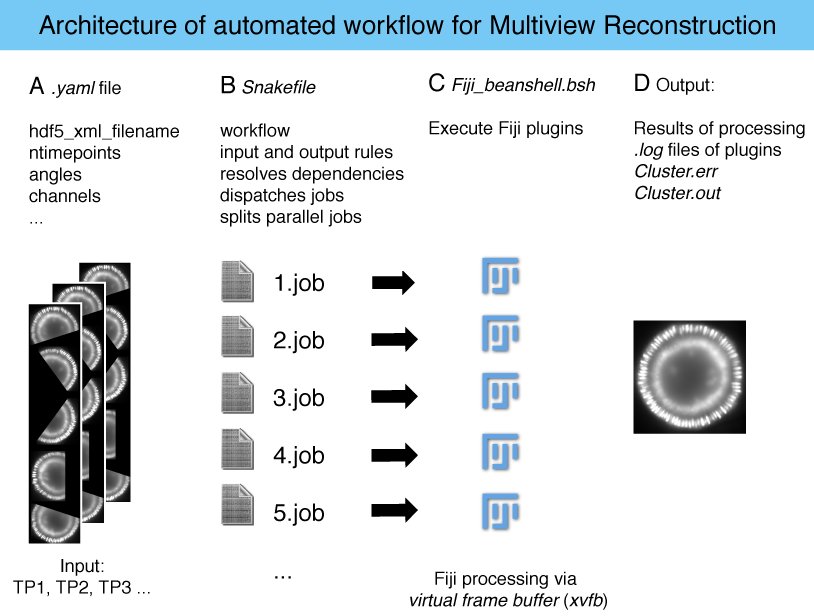
Amat F., Höckendorf B., Wan Y., Lemon W.C., McDole K., Keller P.J. (2015) **Efficient processing and analysis of large-scale light-sheet microscopy data** *Nat Protoc.*, 10(11):1679-96.

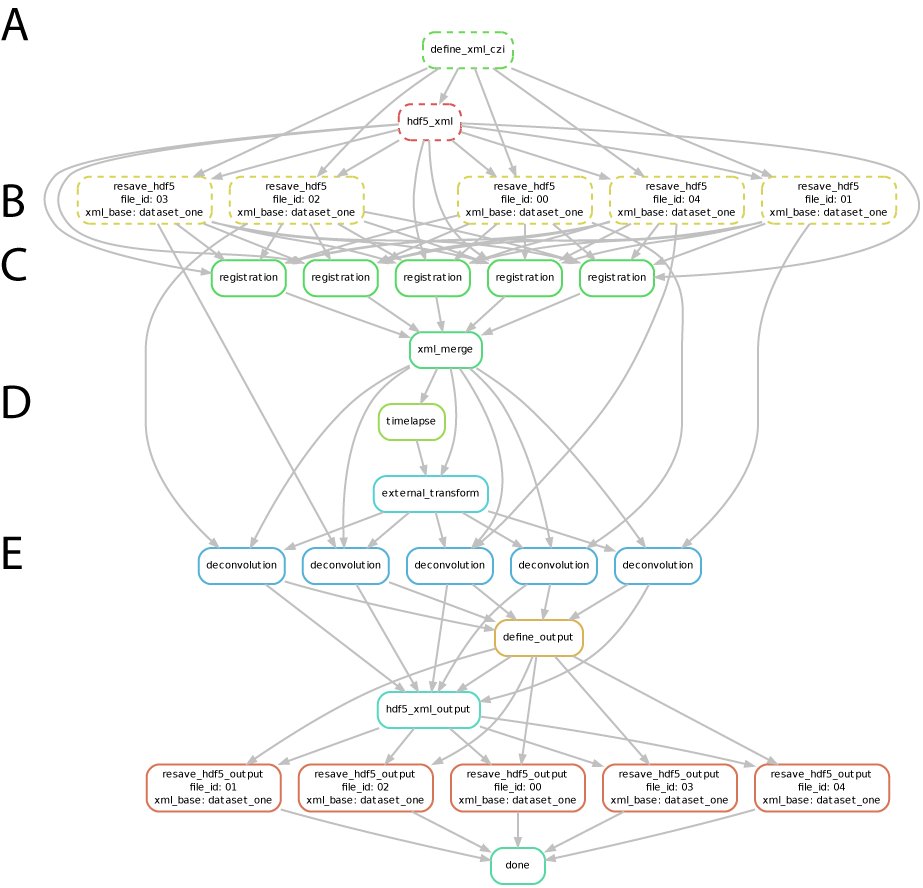
Tomer R., Khairy K., Amat F., Keller P.J. (2012) **Quantitative high-speed imaging of entire developing embryos with simultaneous multiview light-sheet microscopy** *Nat Methods*, 9(7):755-63

Rubio-Guivernau J.L., Gurchenkov V., Luengo-Oroz M.A., Duloquin L., Bourgine P., Santos A., Peyrieras N., Ledesma-Carbayo M.J. (2012) **Wavelet-based image fusion in multi-view three-dimensional microscopy** *Bioinformatics*, 28(2):238-45.

Schmid B., Huisken J. (2015) **Real-time multi-view deconvolution** *Bioinformatics*, 31(20):3398-400.

**Supplementary Data**

**Supp. Fig.1. Workflow architecture,** for processing a multiview dataset, here shown a time-lapse recording of the expression of Histone-YFP during *Drosophila melanogaster* embryogenesis with 5 views, the parameters are determined prior the automated processing and stored in a *.yaml* configuration file (A). These parameters are passed to a *Snakefile,* which contains the logic of the workflow (B). Upon execution of *snakemake* and presence of the input files (e.g. images) *snakemake* dispatches the jobs which call Fiji *beanshell scripts* to carry out the processing using Fiji via a *virtual frame buffer* (C). The output generated by the workflow triggers the next batch of jobs once the input rules of the next step are fulfilled. The processing additionally writes log files and the cluster error and output files (D).

****

**Supp. Fig.2. Dependency graph of the *snakemake* workflow,** *snakemake* draws a directed acyclic graph (DAG) by resolving the file dependencies between the different processing steps. Here we show an example of a DAG for processing a dataset with 5 time points. Jobs are dispatched when the input rule of the first processing step is fulfilled (A). The next batch is sent when all outputs of the processing step are created and the input rule of the next step is fulfilled (B-E). Independent tasks in the same processing step are dispatched in parallel, i.e. parallel processing of time points (B, E).

1. Processing switches

2. Define dataset

2.1. General Settings

hdf5\_xml\_filename: '"dataset"',

ntimepoints: 90,

angles: "0,72,144,216,288",

channels: "green",

illumination: "0",

2.2. Settings for .czi files

first\_czi: "name.czi",

2.3. Settings for .tif datasets

image\_file\_pattern: 'img\_TL{{t}}\_Angle{{a}}.tif',

multiple\_channels: '"NO (one channel)"',

3. Detection and registration

3.1 Channel settings

reg\_process\_channel: '"All channels"',

source\_channel: "red",

target\_channel: "green",

reg\_interest\_points\_channel: '"beads"',

type\_of\_detection: '"Difference-of-Gaussian"',

3.2 Settings for Difference-of-Mean

3.3 Settings for Difference-of-Gaussian

sigma: '1.3',

threshold\_gaussian: '0.025',

4. Time-lapse registration

reference\_timepoint: '45',

5. Weighted-average fusion

downsample: '1',

minimal\_x: '274',

minimal\_y: '17',

minimal\_z: '-423',

maximal\_x: '1055',

maximal\_y: '1928',

maximal\_z: '480',

6. Multiview deconvolution

6.1. External transformation

external\_trafo\_switch: "\_transform",

matrix\_transform: '"0.5, 0.0, 0.0, 0.0, 0.0, 0.5, 0.0, 0.0, 0.0, 0.0, 0.5, 0.0"',

6.2. Deconvolution settings

iterations: '15',

minimal\_x\_deco: '137',

minimal\_y\_deco: '-8',

minimal\_z\_deco: '-211',

maximal\_x\_deco: '527',

maximal\_y\_deco: '964',

maximal\_z\_deco: '240',

detections\_to\_extract\_psf\_for\_channel: '"beads"',

7. Resave output

8. Software directories

9. Fiji resource settings

10. Advanced settings

**Suppl. List 1. Parameter List for processing,** list of the key input parameters in the *.yaml* file for multiview, multi-channel multi-illumination side processing of time-lapse SPIM recordings. The parameters are sorted to match the logic of the processing workflow: define and resave to hdf5 (2) > detect and register (3) > time lapse registration (4) > fuse (5 + 6) and resave output (7). The key parameters (highlighted in red) are recorded during manual processing of the reference time point in the GUI.

Local back end:

/path/to/snakemake/snakemake -j 1 -d /path/to/data/

To specify the configuration script for the queuing system:

--cluster-config ./cluster.json

For DRMAA back end add:

--drmaa " -q {cluster.lsf\_q} {cluster.lsf\_extra}"

For Lsf backend add:

--cluster "bsub -q {cluster.lsf\_q} {cluster.lsf\_extra}”

Flag for number of jobs run in parallel:

-j <number of jobs>

Flag for specifying data location:

-d /path/to/data/

Flag for dry run of snakemake:

-n

Force the execution of a rule:

-R <name of rule>

To save error and output files of cluster add:

--drmaa " -q {cluster.lsf\_q} {cluster.lsf\_extra} -o test.out -e test.err"

--cluster "bsub -q {cluster.lsf\_q} {cluster.lsf\_extra} -o test.out -e test.err“

**Suppl. List 2. List of commands for the *snakemake* workflow.**

|  |  |
| --- | --- |
| A Workstation | |
| Processor | 2x hexa-core Intel Xeon E5-2630 @ 2.30 GHz |
| Memory | 128 GB (16x8GB) @1600 MHz, DDR3 ECC RDIMM |
| GPU | 2x NVIDIA Quadro 4000, 2 GB of Memory, 256 CUDA cores |
| Hard Drive | 4x 4 TB SATA disks in RAID 5 configuration |
|  |  |
| B Cluster MPI-CBG "*Madmax*" | |
| Processor | 44x Nodes with 2x hexa-core Intel Xeon E5-2640 @ 2.50 GHz |
| Total No. of cores | 528 |
| Memory per node | 128 GB (16x8GB) @ 1333 MHz |
| GPU | 4x nodes with NVIDIA Tesla M2090 Fermi Generation @ 1.3GHz, 6GB GDDR5 Memory @ 1.85GHz, 512 CUDA cores |
|  |  |
| Lustre storage server | |
| Storage | Fully redundant Lustre volume with 200 TB of usable space |
| Storage architecture | 2 metadata servers in active/passive configuration with a shared disk enclosure and 4 object storage servers with 3 disk enclosures delivering 2.5 GB/s each and 10 GB/s aggregated |
| Interconnect | InfiniBand QDR 40Gbps fully non-blocking fat tree topology |

**Suppl. Table 1. Hardware used for processing,** parameters of the computer hardware for efficient SPIMage processing. Standalone desktop machine (A) and HPC cluster (B).

|  |  |  |  |
| --- | --- | --- | --- |
| Job name | Average memory (MB) | Average CPU time (sec) | #jobs |
| Define dataset | 2315 | 908 | 1 |
| Define hdf5 dataset | 2158 | 39 | 1 |
| Resave to hdf5 | 2827 | 530 | 90 |
| Detection and registration | 7189 | 1388 | 90 |
| Merge xml | 3 | 43 | 1 |
| Time lapse registration | 2534 | 953 | 1 |
| Average fusion | 7761 | 3806 | 90 |
| Deconvolution GPU | 27171 | 7485 | 90 |
| Define output | 3 | 23 | 1 |
| Define hdf5 output | 2 | 32 | 1 |
| Resave output to hdf5 | 4918 | 534 | 90 |

**Suppl. Table 2. Cluster processing resource summary.**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Local 1 TP | Local 90 TPs | Cluster 90 TPs |
| Resave to hdf5 | 3 | 262 | 15 |
| Detection and registration | 2 | 221 | 15 |
| Average fusion | 7 | 661 | 47 |
| Deconvolution (GPU) | 21 | 1874 | 740 |
| Resave output | 3 | 286 | 7 |
| Total with average fusion |  | 23 h 56 min | 1h 31min |
| Total with deconvolution (GPU) |  | 44 h 8 min | 13 h 9 min |

**Suppl. Table 3. Processing time comparison Local vs Cluster (min).**