Bioinformatics Advance Access published December 1, 2015

An automated workflow for parallel processing of large multiview SPIM recordings

Christopher Schmied [1], Peter Steinbach [1], Tobias Pietzsch [1], Stephan Preibisch [1,2,3] and Pavel Tomancak [1*]

[1]Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany, [2]HHMI Janelia Research Campus, Ashburn, Virginia, USA and [3]Max Delbrück Center for Molecular Medicine, Berlin Institute for Medical Systems Biology, Berlin, Germany.

Associate Editor: Prof. Robert Murphy

ABSTRACT

Summary: Selective Plane Illumination Microscopy (SPIM) allows to image developing organisms in 3D at unprecedented temporal resolution over long periods of time. The resulting massive amounts of raw image data requires extensive processing interactively via dedicated graphical user interface (GUI) applications. The consecutive processing steps can be easily automated and the individual time points can be processed independently, which lends itself to trivial parallelization on a high performance computing (HPC) cluster. Here we introduce an automated workflow for processing large multiview, multi-channel, multi-illumination time-lapse SPIM data on a single workstation or in parallel on a HPC cluster. The pipeline relies on *snakemake* to resolve dependencies among consecutive processing steps and can be easily adapted to any cluster environment for processing SPIM data in a fraction of the time required to collect it.

Availability: The code is distributed free and open source under the MIT license http://opensource.org/licenses/MIT. The source code can be downloaded from github: https://github.com/mpicbg-scicomp/snakemake-work-flows. Documentation can be found here: http://fiji.sc/Automated_workflow_for_parallel_Multi-view_Reconstruction.

Contact: schmied@mpi-cbg.de

1 INTRODUCTION

The duration and temporal resolution of 3D fluorescent imaging of living biological specimen is limited by the amount of laser light exposure the sample can survive. SPIM alleviates this by illuminating only the imaged plane thus reducing photo damage dramatically. Additionally, SPIM achieves fast acquisition rates due to sensitive wide-field detectors and sample rotation enables complete coverage of large, non-transparent specimen. Taken together, SPIM allows imaging of developing organisms *in toto* at single cell resolution with unprecedented temporal resolution over long periods of time (Huisken *et al*., 2004; Keller *et al*., 2008).

This powerful technology produces massive, terabyte size datasets that need computationally expensive and time-consuming processing before analysis. Existing software solutions implemented in Fiji (Preibisch *et al*., 2010, 2014; Preibisch, unpublished; Schmied *et al*., 2014) or in ZEISS ZEN black are performing chained processing steps on a single computer and require user inputs via a GUI. As the spatial and temporal resolution of the light sheet data increase, such approaches become inconvenient since processing can take days.

In controlled experiments, SPIM image processing is robust enough to be automated and key steps are independent from time point to time point. HPC is inherently designed for such time consuming and embarrassingly parallel tasks that require no user interaction. Therefore, we developed an automated workflow with minimum user interaction that is easily scalable to multiple datasets or time points on a cluster. In combination with the appropriate computing resources it enables for the first time processing of SPIM data that is faster than the total acquisition time required for collecting the raw images.

2 PROCESSING WORKFLOW

The Fiji SPIM processing pipeline uses Hierarchical Data Format (HDF5) as data container for the originally generated TIFF or CZI files by custom made (Pitrone *et al*., 2013) or commercial SPIM microscopes (Fig. 1A,B). Following format conversion, multiview registration aligns the different acquisition angles (views) within each time point (Fig. 1C), and subsequent time-lapse registration stabilizes the recording over time (Preibisch *et al*., 2010) (Fig. 1D). Fusion combines the registered views of one time point into a single volume by averaging or multiview deconvolution (Preibisch *et al*., 2010, 2014) (Fig. 1E,F). The result is a set of HDF5 files containing registered and fused multiview SPIM data that can be examined locally or remotely using the BigDataViewer (Pietzsch *et al*., 2015).

All steps are implemented as plugins (Preibisch *et al*., 2010, 2014; Preibisch, unpublished; Pietzsch *et al*., 2015), in the open-source platform Fiji (Schindelin *et al*., 2012). We use these plugins by executing them from the command line as Fiji beanshell scripts (Suppl. Fig. 1). To overcome the legacy dependency of Fiji on the GUI we encapsulate it in a *virtual framebuffer* (*xvfb*) that simulates a monitor in the headless cluster environment (Suppl. Fig. 1).

*to whom correspondence should be addressed

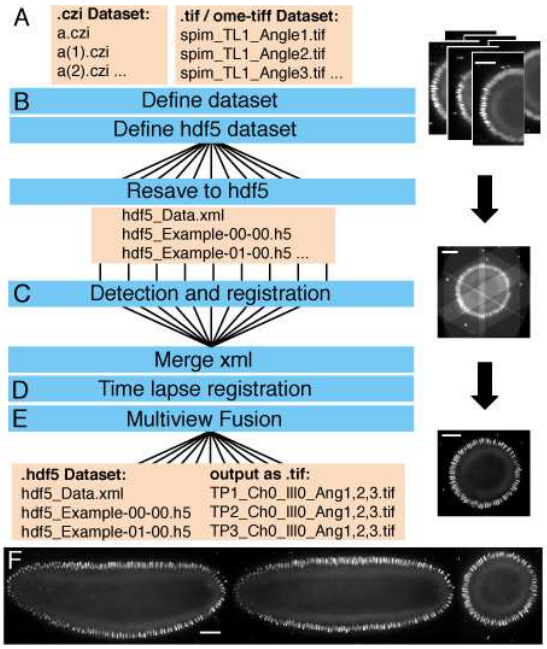

Fig. 1. Automated workflow for Multiview processing. Workflow for SPIM image processing (A-E) using parallelization (B, C and E). Shown on the right yz–slices in the BigDataViewer of a *Drosophila* embryo expressing histone H2Av-mRFPruby raw (A) registered (C) and deconvolved (E). Results of deconvolution with xy–, xz– and xz–slices through the fused volume of the same embryo (F). Scale bars represent 50 µm.

To map and dispatch the workflow logic to a single workstation or on a HPC cluster, we use the automated workflow engine *snakemake* (Köster and Rahmann, 2012). The workflow is defined using a *Snakefile* containing the name, input and output file names of each of the processing steps and python code calling the *beanshell scripts* (Suppl. Fig. 1). Upon invocation, the *snakemake* rule engine resolves the dependencies between individual processing steps based on the input files required and the output files produced during the workflow. It also creates the command that fits the input/output rule description and the template command as defined in the *Snakefile*. Most importantly, if single tasks on individual files are discovered to be independent, they are invoked in parallel (Suppl. Fig. 2). Each instance of *snakemake* for one dataset is independent and thus the workflow can be applied simultaneously to multiple dataset.

The required parameters for processing are collected by the user during GUI processing of an exemplary time point and entered into a *.yaml* configuration file (Suppl. List 1). The workflow is executed by passing the *.yaml* file to *snakemake* on the command line (Suppl. Fig. 1). Importantly, from the user perspective the launching of the pipeline on a HPC cluster and on a local workstation appears identical and require a single command (Suppl. List 2). If the parameters are chosen correctly and the local or HPC resources are sufficient (Suppl. Table 1 and 2) no further action from the user is necessary.

Snakemake supports multiple back ends to perform the command dispatch: local, cluster and *Distributed Resource Management Application API* (*DRMAA*)(Köster and Rahmann, 2012). The local back end creates a new sub shell and calls the command(s) required. The cluster back end is a general interface to HPC batch systems based on string substitution. *DRMAA* specifies a system library that interfaces all common batch systems based on a generalized task model, thus multiple batch systems are supported through one interface.

3 RESULTS

We compared the performance of the pipeline on a 175 GB, single channel SPIM recording of a *Drosophila* embryo consisting of 90 time points and 5 views, processed either on a single computer or on a HPC cluster (Suppl. Table 1). The processing using average fusion takes almost precisely one day on a single powerful computer. In contrast, using the full cluster resource the dataset can be processed in 1 h 31 min, which represents a 16-fold speedup in processing. Since the time-lapse covers 23 hours of *Drosophila* embryonic development the processing becomes real time with respect to the acquisition. Using deconvolution on a cluster with only 4 GPUs (Suppl. Table 3) still brings a more than 3-fold speed up (Suppl. Table 3). A dataset of 2.2 TB in size with 715 time points (Schmied *et al*., 2014) would take an estimated week to process on a single computer. Using this method the processing is reduced to only 15 h with typical cluster workload from other users.

4 CONCLUSION AND OUTLOOK

The biologist‘s goal is to analyse, for instance, cellular behaviour using time-lapse SPIM recordings. The steps between data acquisition and analysis are of rather technical interest. Our pipeline leverages HPC to reduce the notoriously difficult and time-consuming SPIM data processing to a single autonomous command. Similar pipelines have been developed (Amat *et al*., 2015), however in our case the reliance on an open source platform (Fiji) allows us to execute the processing in parallel without any software associated costs. It is also possible to incorporate new algorithms from the Fiji ecosystem into the pipeline (Schmid and Huisken, 2015 and see Supplementary Note).

Future improvements of the workflow will provide greater accessibility to novice users by using the UNICORE GUI framework (Almond and Snelling, 1999). Ultimately, we aim for a completely unsupervised automated processing similar to grid computing practiced in fields facing similar big data challenges such as particle physics and molecular simulation (Bird, 2011; Gesing *et al*., 2012)

ACKNOWLEDGEMENT

We thank Stephan Janosch for valuable discussions and Akanksha Jain for testing the workflow. We thank the computer services of the MPI-CBG for their great general support and specifically Oscar Gonzalez, the members of the scientific computing facility and light microscopy facility.

FUNDING

P.T. and C.S. were supported by the HFSP Young Investigator grant RGY0093/2012. P.T. and T.P. were supported by the European Research Council Community‘s Seventh Framework Program (FP7/2007-2013) grant agreement 260746.

REFERENCES

Almond,J. and Snelling,D. (1999) UNICORE: Uniform Access to Supercomputing as an Element of Electronic Commerce. *Future Generation Computer Systems*, **613**, 1-10.

Amat,F. *et al*. (2015) Efficient processing and analysis of large-scale light-sheet microscopy data. *Nat Protoc.*, **10**, 1679-96.

Bird,I. (2011) Computing for the Large Hadron Collider. *Annual Review of Nuclear and Particle Science*, **61**, 99-118.

Gesing,S. *et al*. (2012) A Single Sign-On Infrastructure for Science Gateways on a Use Case for Structural Bioinformatics. *Journal of Grid Computing*, **10**, 769-790.

Huisken,J. *et al*. (2004) Optical Sectioning Deep Inside Live Embryos by Selective Plane Illumination Microscopy. *Science*, **305**, 1007-1009.

Keller,P. J. *et al*. (2008) Reconstruction of zebra- fish early embryonic development by scanned light sheet microscopy. *Science*, **322**, 1065-1069.

Köster,J. and Rahmann,S. (2012) Snakemake–a scalable bioinformatics workflow engine. *Bioinformatics*, **28**, 2520-2522.

Pietzsch,T. *et al*. (2015) BigDataViewer: visualization and processing for large image data sets. *Nature Methods*, **12**, 481-483.

Pitrone,P.G. *et al*. (2013) OpenSPIM: an open-access light-sheet microscopy platform. *Nature Methods* , **10**, 598-599

Preibisch,S., *et al*. (2010) Software for bead-based registration of selective plane illumination microscopy data. *Nature Methods*, **7**, 418-419.

Preibisch,S., *et al*. (2014) Efficient Bayesian-based multiview deconvolution. *Nature Methods*, **11**, 645-648.

Preibisch,S. unpublished: https://github.com/bigdataviewer/SPIM_Registration

Schindelin,J. *et al*. (2012) Fiji: an open-source platform for biological-image analysis. *Nature Methods*, **9**, 676-682.

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

Schmied,C. *et al*. (2014) Open-source solutions for SPIMage processing. *Quantitative Imaging in Cell Biology* , **123**, 505-529.