Bioimage informatics

An automated workflow for parallel processing of large multiview SPIM recordings

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

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

*To whom correspondence should be addressed.
Associate Editor: Robert Murphy

Received on 30 July 2015; revised on 13 November 2015; accepted on 25 November 2015

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, multichannel, multiillumination 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 and implementation: 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-workflows. Documentation can be found here: http://fiji.sc/Automated_workflow_for_parallel_Multiview_Reconstruction.

Contact: schmied@mpi-cbg.de

Supplementary information: Supplementary data are available at Bioinformatics online.

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. Selective Plane Illumination Microscopy (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; Schmied et al., 2014; Preibisch, unpublished (https://github.com/fiji/SPIM_Registration)) 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
An automated workflow for parallel SPIM processing

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 and 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 and 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; Pietzsch et al., 2015; Preibisch, unpublished [https://github.com/fiji/SPIM_Registration]), 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 (Supplementary Fig. 1). The biologist’s goal is to analyze, for instance, cellular behavior within a HPC cluster environment (Supplementary Fig. 1).

To map and dispatch the workflow logic to a single workstation or on a HPC cluster, we use the automated workflow engine snakemake (Koster 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 (Supplementary 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 (Supplementary Fig. 1). 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 (Supplementary List 1). The workflow is executed by passing the .yaml file to snakemake on the command line (Supplementary 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 (Supplementary List 2). If the parameters are chosen correctly and the local or HPC resources are sufficient (Supplementary 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) (Koster 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 (Supplementary 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 h of Drosophila embryonic development the processing becomes real time with respect to the acquisition. Using deconvolution on a cluster with only 4 GPUs (Supplementary Table 1) still brings a more than 3-fold speed up (Supplementary 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 13 h with typical cluster workload from other users.

4 Conclusion and outlook

The biologist’s goal is to analyze, for instance, cellular behavior using time-lapse SPIM recordings. The steps between data acquisition and analysis are of rather technical interest. Our pipeline

Fig. 1. Automated workflow for multiview processing. Workflow for SPIM image processing (A-E) using parallelization (B, C and E). Shown on the right are slices in the BigDataViewer of a Drosophila embryo expressing histone H2Aα-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.

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/32/7/1112/1744153 by guest on 02 August 2018
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).

Acknowledgements

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.

Conflict of interest: none declared.

References


