Publication and Citation of Scientific Software with Persistent Identifiers

Martin Fenner, for Martin Hammitzsch and the SciForge project
Technical Lead Article-Level Metrics
Public Library of Science

sciforge

PLOS

Scientific software has become an essential component of the research process.

but

Software development in general is not perceived as a scientific achievement.

sciforge

A project funded by the German Research Foundation (DFG) at GFZ Potsdam, coordinator Martin Hammitzsch

Establish the missing link between papers and data publications.

Make software recognized as scientific achievement.

Leverage open access and open science.

Establish standard software engineering rules, best practices and processes in science.

Make software recognized as scientific achievement.

Software Journals and Articles

OPEN ACCESS Freely available online PLoS one

TrakEM2 Software for Neural Circuit Reconstruction

Albert Cardona[1]*, Stephan Saalfeld[2], Johannes Schindelin[2], Ignacio Arganda-Carreras[3], Stephan Preibisch[2], Mark Longair[1], Pavel Tomancak[2], Volker Hartenstein[4], Rodney J. Douglas[1]

1 Institute of Neuroinformatics, University of Zurich and ETH Zurich, Zurich, Switzerland, **2** Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany, **3** Massachusetts Institute of Technology, Boston, Massachusetts, United States of America, **4** Molecular Cell and Developmental Biology Department, University of California Los Angeles, Los Angeles, California, United States of America

Abstract

A key challenge in neuroscience is the expeditious reconstruction of neuronal circuits. For model systems such as *Drosophila* and *C. elegans*, the limiting step is no longer the acquisition of imagery but the extraction of the circuit from images. For this purpose, we designed a software application, TrakEM2, that addresses the systematic reconstruction of neuronal circuits from large electron microscopical and optical image volumes. We address the challenges of image volume composition from individual, deformed images; of the reconstruction of neuronal arbors and annotation of synapses with fast manual and semi-automatic methods; and the management of large collections of both images and annotations. The output is a neural circuit of 3d arbors and synapses, encoded in NeuroML and other formats, ready for analysis.

Citation: Cardona A, Saalfeld S, Schindelin J, Arganda-Carreras I, Preibisch S, et al. (2012) TrakEM2 Software for Neural Circuit Reconstruction. PLoS ONE 7(6): e38011. doi:10.1371/journal.pone.0038011

Editor: Aravinthan Samuel, Harvard University, United States of America

Received March 22, 2012; **Accepted** April 28, 2012; **Published** June 19, 2012

Funding: This work was funded primarily by Kevan A. Martin and the Institute of Neuroinformatics, University of Zurich and ETH Zurich; and also by grant NIH 1-R01 NS054814-05 to VH and grant SNSF 31003A_132969 to AC. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: sapristi@gmail.com

Introduction

There is a growing consensus that detailed volumetric reconstructions of thousands of neurons in millimeter-scale blocks of tissue are necessary for understanding neuronal circuits [1,2]. Modern electron microscopes (EM) with automatic image acquisition are able to deliver very large collections of image tiles [3–8]. Unfortunately, the problems of acquiring the data have so far been easier to solve than that of interpreting it [9,10]. Increasingly, neuroscience laboratories require automated tools for managing these vast EM data sets using affordable consumer desktop computers.

Here, we present such a tool. It is an open source software package, named TrakEM2, that is optimised for neural circuit reconstruction from tera-scale serial section EM image data sets. The software handles all the required steps: rapid entry, organization, and navigation through tera-scale EM image collections. Semi- and automatic image registration is easily performed within and across sections. Efficient tools enable manipulating, visualizing, reconstructing, annotating, and measuring neuronal components embedded in the data. An ontology-controlled tree structure is used to assemble hierarchical groupings of reconstructed components in terms of biologically meaningful entities such as neurons, synapses, tracts and tissues. TrakEM2 allows millions of reconstructed entities to be manipulated in nested groups that encapsulate the desired abstract level of analysis, such as "neuron", "compartment" or "neuronal lineage". The end products are 3D morphological reconstructions, measurements, and neural circuits specified in *NeuroML* [11] and other formats for functional analysis elsewhere.

TrakEM2 has been used successfully for the reconstruction of targeted EM microvolumes of *Drosophila* larval central nervous system [7], for array tomography [12], for the reconstruction and automatic recognition of neural lineages in LSM stacks [13], for the reconstruction of thalamo-cortical connections in the cat visual cortex [14] and for the reconstruction of the inhibitory network relating selective-orientation interneurons in a 10 Terabyte EM image data set of the mouse visual cortex [8], amongst others.

Results

From Raw Collections of 2d Images to Browsable Recomposed Sample Volumes

An EM volume large enough to encapsulate significant fractions of neuronal tissue and with a resolution high enough to discern synapses presents numerous challenges for visualization, processing and annotation. The data generally consists of collections of 2d image tiles acquired from serial tissue sections (Figure 1; [7,8]) or from the trimmed block face (Block-face Serial EM or SBEM, [3,15]; focused ion beam scanning EM or FIBSEM, [6]) that are collectively far larger than Random Access Memory (RAM) of common lab computers and must be loaded and unloaded on demand from file storage systems. Additional experiments on the same data sample may have generated light-microscopical image volumes that must then be overlaid on the EM images, such as in array tomography [12,16] or correlative calcium imaging [8,15]. TrakEM2 makes browsing and annotating mixed, overlaid types of images (Figure S1) over terabyte-sized volumes fast (Text S1, section "Browsing large serial EM image sets") while enabling the independent manipulation of every single image both from a point-and-click graphical user interface (GUI; Figure 1e, S2, S3,

Describe software in the traditional journal article format, ideally with special considerations for software (e.g. software repositories, peer review)

Software journals are a new concept similar to data journals – only a few examples currently exist.

Some of the most highly cited papers in traditional journals are software (or data) papers, e.g.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., et al. (2000). The Protein Data Bank. Nucleic Acids Research, 28(1), 235–242. doi:10.1093/nar/28.1.235

Peer Review

Software Sustainability Institute

Journal of open research software

- *Is the software in a suitable repository?*
- *Does the software have a suitable open licence?*
- *If the Archive section is filled out, is the link in the form of a persistent identifier, e.g. a DOI? Can you download the software from this link?*
- *If the Code Repository section is filled out, does the identifier link to the appropriate place to download the source code? Can you download the source code from this link?*
- *Is the software license included in the software in the repository? Is it included in the source code?*
- *Is sample input and output data provided with the software?*
- *Is the code adequately documented? Can a reader understand how to build/deploy/install/run the software, and identify whether the software is operating as expected?*
- *Does the software run on the systems specified? (if you do not have access to a system with the prerequisite requirements, let us know).*
- *Is it obvious what the support mechanisms for the software are?*

Code Review

Pilot study with professional Mozilla developers doing code review on code snippets from already published PLOS Computational Biology papers. Focus on

- Version control and packaging
- Comments and documentation
- Tests
- Readability and code structure

Positive feedback from authors and reviewers, limitation was lack of context (domain expertise or direct contact)

Software Repositories

General or specific for language and/or scientific domain

Almost always open source software with source code

No concept of global persistent identifiers or long-term preservation

Preservation Repositories

Journal of Open Research Software distinguishes:

- A source code repository holds many versions of the software as it is being developed
- A preservation or institutional repository will preserve a set of files deposited for the long term

Both Figshare and Zenodo integrate with Github
Neither repository offers long-term storage of executable code (e.g. storing all software dependencies or virtual machines)

Persistent Identifiers

Persistent identifiers for software are not (yet) common practice.

DataCite DOIs should be the preferred persistent identifier:

- do not invent yet another identifier
- DataCite metadata describe software well
- software and data often used together

Challenge are source code repositories without long-term preservation

Versioning

- Semantic versioning (MAJOR.MINOR.PATCH, e.g. 2.3.2) of software is evolving standard
- Resolving dependencies is a major challenge
- DataCite suggests to register new DOIs for major and minor versions
- DataCite metadata can describe relationship: isNewVersionOf, isPreviousVersionOf

Research Infrastructure

Support for scientific software with persistent identifiers needed in

- Institutional Repositories
- Research Information Systems (CRIS)
- Journal submission systems
- Reference Managers
- Kerndatensatz Forschung

Metrics

Impactstory

Dataset

Article-Level Metrics Hannover Medical School
(2013) *figshare.*
highly viewed +1 | highly discussed | saved | discussed

CrowdoMeter Tweet Classifications
(2012) *figshare.*
highly discussed | highly viewed | saved

CrowdoMeter Tweets
(2012) *figshare.*
highly viewed +2 | saved | discussed

Article-Level Metrics f
(2013) *figshare.*
viewed

Software

jekyll-travis
(2014) Integrate Jekyll with Github Pages and Travis CI to automatically build Jekyll site
highly recommended +1 | highly cited

orcid-feed
(2013) RSS feeds for ORCID profiles
highly recommended | cited

mfenner.github.io
(2013) My personal blog
highly recommended | cited

jekyll-orcid
(2013) Jekyll plugin that integrates with the ORCID service
highly recommended | cited

jekyll-scholmd
(2013) Auto-linking of scholarly identifiers in markdown files
highly recommended

Metrics

ScienceToolbox Login Sign up

nipy/nipype

python | Computer Science Applications | Biomedical Engineering | Neuroscience

README ✓ LICENSE ✓ Tests ✗ Virtualization ✗ Continuous integration ✓ ✓✓✓○○

96 stars and 105 forks, with 1 citations

Citations

Nipype: A Flexible, Lightweight and Extensible Neuroimaging Data Processing Framework in Python by Gorgolewski K, Burns CD, Madison C, Clark D, Halchenko YO, Waskom ML, Ghosh SS — Frontiers in Neuroinformatics (Jan 2011)

Follow @ScienceToolbox Fork

Dear recruiters:

While you read this, make sure that you remember that GitHub is not your C.V. and that these stats only provide a *biased and one-sided view*. This is just a toy. Don't take it too seriously!

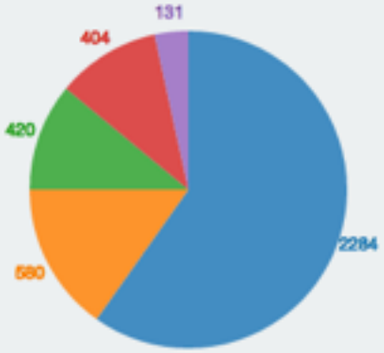
OK. I promise!

THE
OPEN SOURCE
REPORT CARD

Carl Boettiger

Carl Boettiger is a top notch useR (one of the 2% most active R users) who loves pushing code. Carl is a nine-to-fiver who works best in the afternoon (around 1 pm).

Carl has contributed to repositories in 20 languages. In particular, Carl seems to be a pretty serious **R** expert. The following chart shows the number of contributions Carl made to repositories mainly written in **R**, **JavaScript**, **CSS**, **XSLT**, and **Ruby**.



Open Licenses

The Open Source Institute (OSI) has reviewed approved licenses that comply with their Open Source definition. Popular licenses include

- Apache License 2.0
- MIT license
- BSD license
- GNU General Public License

Two topics of discussion are

- copyleft vs. permissive licenses (the former require the same license for derivative works)
- software in source code repositories without a license

Further Reading

Wilson, G., Aruliah, D. A., Brown, C. T., Hong, N. P. C., Davis, M., Guy, R. T., et al. (2012, October 1). Best Practices for Scientific Computing. arXiv.org.

Stodden, V., & Miguez, S. (2014). Best Practices for Computational Science: Software Infrastructure and Environments for Reproducible and Extensible Research. Journal of Open Research Software, 2(1), e21. doi: 10.5334/jors.ay

Osborne, J. M., Bernabeu, M. O., Bruna, M., Calderhead, B., Cooper, J., Dalchau, N., et al. (2014). Ten simple rules for effective computational research. *PLoS Comput Biol*, *10*(3), e1003506. doi:10.1371/journal.pcbi.1003506