PyView: A general purpose tool for analyzing calcium imaging data

Ajayrama Kumaraswamy[1], **Georg Raiser**[2], **and C Giovanni Galizia**[1¶]

1 University of Konstanz, Department of Biology, Universitatsstr. 10, D-78457 Konstanz **2** Champalimaud Centre for the Unknown, Avenida de Brasília, 1400-038 Lisboa, Portugal ¶ Corresponding author

DOI: 10.21105/joss.04936

Software

- Review
- Repository
- Archive

Editor: Charlotte Soneson

Reviewers:

- @adamltyson
- @Yannan-Chen

Submitted: 14 October 2022
Published: 08 February 2023

Summary

Optical imaging allows to record network activity in many neurons simultaneously, yielding a direct access to network activity in the brain (Göbel & Helmchen, 2007). The development of transgenic animals that express genetically encoded calcium reporters, such as GCamP, has further increased the possibilities to study neural networks, and other reporters suitable for measuring membrane potential or specific second messengers are being developed in several labs (Rad et al., 2017). All of these techniques use changes in fluorescent light to quantitatively capture changes in neuron physiology, and all necessitate high-level image analysis tools in order to analyze the physiological data in a quantitative and consistent way (Galizia & Vetter, 2004). Here, we propose PyView (https://github.com/galizia-lab/pyview), a software for optical imaging analysis. PyView splits data treatment into two steps: a powerful GUI for interactive data analysis, with flexible selection of evaluation parameters, and a second step for batch processing, yielding time traces, 2D images or 3D movies across all experimental measurements with identical parameter settings. The program is modular, and easy to expand with tools for dedicated analyses and/or experimental questions. The program is open source, written in Python, and can be expanded to act as a wrapper for other programs that address single steps in a data analysis pipeline. Information about example workflows, galleries of examples outputs, and guides for installing, using and developing PyView are organized in a wiki (https://github.com/galizia-lab/PyView/wiki).

Statement of need

Optical imaging experiments, including calcium imaging experiments, can have long durations, in particular if performed in vivo. Typically, they consist of many measurements with different experimental settings (stimulus type and intensity, pharmacological treatments, animal age or genotype, to name but a few). The resulting data needs pre-processing of various kinds, depending on the system that is being investigated (e.g. olfactory or visual system, or animal species), the activity reporter used (e.g. GCamP, FURA), and the optical properties of the setup (e.g. 2-photon imaging, widefield imaging; single wavelength or ratiometric imaging). Bleaching and baseline correction, movement correction, and scattered light correction are necessary, and selection of appropriate regions of interest and exclusion of areas that are not neurons are among the most important steps here. Adjusting parameters for data treatment can be tedious, and necessitates interactive visualization and iterative experimentation by the researcher.

Conversely, statistical analysis across experimental animals, and across experimental parameters (e.g. different stimuli for sensory systems, or different animal groups), necessitate that all datasets are treated in an identical way. Therefore, data generation for statistical analysis should be done not using an interactive mode, but rather using batch processing, with identical parameter settings across all datasets and nonetheless using the same code as during GUI experimentation.

Scope and Alternatives

Several packages for analysis of calcium imaging data have been developed in recent years (Pnevmatikakis, 2019). These include CaImAn (Giovannucci et al., 2019), Suite2p (https://www.suite2p.org/), CALIMA (Radstake et al., 2019), SIMA (https://github.com/losonczylab/sima), MOCO (Dubbs et al., 2016), Toolbox-Romano (Romano et al., 2017), OpenFluo (Dupuy et al., 2009), ILTIS (https://github.com/grg2rsr/ILTIS), ImageBee (Strauch et al., 2013) and plug-ins to Fiji (Schindelin et al., 2012). Each is specialized in its own way: movement correction, selection of regions of interests or interactive visualization. However, most labs still program their batch-processing with flexibly selected parameters ad-hoc for each experiment, necessitating high-level programming expertise in a biological lab. Here we propose a modular approach, that is also able to include existing packages into the pipeline (e.g. ILTIS, which is already included), and available to use also for labs without expertise in computer programming.

Architecture: Software

PyView relies heavily on several existing tools and packages in Python, incorporating their strengths into a general and extendable framework. All of them have been listed in the file "setup.py" at the root of the PyView GitHub repository (https://github.com/galizia-lab/pyview)

PyView has been structured for good readability, modularity, testability and extensibility. It consists primarily of three parts:

PyView-Core

This contains all the classes, functions and definitions that form the core of PyView.

PyView-GUI

This is a GUI written in PyQt5 with which users can use the functionalities implemented in PyView-core: configure parameters, save configuration files, load data, create representative images and movies, and export processed data into CSV files for further analysis.

ILTIS

ILTIS (https://github.com/grg2rsr/ILTIS) is a stand-alone package used for interactive calcium imaging data visualization, a fork (https://github.com/galizia-lab/ILTIS) of which has been integrated into PyView-GUI. This allows interactive visualization of imaging data and inspecting individual image frames or pixelwise time traces. ILTIS can also be used to manually segment images and save the results into files.

Architecture: Data processing

The basic idea is to analyze data in two steps: 1. load and interactively play with the data in PyVIEW-GUI to find analysis parameters suitable for a set of experimental conditions 2. non-interactive batch processing using scripts based on PyVIEW-Core to generate images, movies, spreadsheets of time traces, etc.

Details on the architecture are explained in the wiki as a tutorial: https://github.com/galizia-lab/pyview/wiki/Tutorial

Future development

The package will be further developed on a continuous basis. Input from the community is welcome, in particular to implement reading new data formats, and to export data to other packages.

The next version of PyView will leverage CalmAn (Giovannucci et al., 2019) for correcting movement artifacts, for automatically estimating spatial footprints and temporal activity of neural sources. PyView will also incorporate rNMF (Soelter et al., 2014; Strauch et al., 2013) for source segmentation and SimpleElastix (Marstal et al., 2016) for correcting anisotropic movement artifacts. Since PyView can integrate multiple pipelines into itself, it will also be useful to compare their outcomes. A similar approach has been taken in electrophysiology by SpikeInterface (https://open-ephys.org/spikeinterface), which allows different spike sorting algorithms to be compared on a single dataset (Buccino et al., 2020). Guides for running unit tests on existing modules as well as contributing new modules are provided in the section "Developer Guide" of the PyView wiki (https://github.com/galizia-lab/PyView/wiki/Developer-Guide).

Strengths

Among the strengths of this package, we note the following: PyView is free and completely open source, and built with Python and various Python libraries that are equally free and available. The package is modular and easily extendable – our hope is that it will grow thanks to the input of many labs across the world. It is built to incorporate existing and future tools.

Furthermore, PyView can be used with only little computer knowledge, and programming knowledge is not necessary. Thus, it is suitable for experimenters with all levels of programming experience, and can be used by biology or medical science undergraduates. Similarly, it can be used as a data analysis tool in teaching classes that focus on the biological result, and be fruitfully used to teach the difference between "few animals interactive data analysis" and "many animals statistical analysis with standardized settings".

Thanks to the necessity to structure the data in appropriate folders, and to list all choices in dedicated parameter files (.yml files), data analysis choices remain transparent and explicit to the user.

Limitations and caveats

In the current version, all data that is analyzed in PyView GUI is loaded into memory, creating problems for very large data sets. In most cases, this does not create problems in batch analysis mode, where single measurements are analyzed sequentially.

The power of the program lies in its flexibility, and that the user has extensive choices to control how data is treated, which output is generated, and how it is formatted – this comes at a price: there are many parameters to be set. Even though we tried to name these parameters in an intuitive way, we note that beginners are often overwhelmed.

Use cases and Input Data formats

Lüdke et al. (2018) recorded odor responses in Drosophila brains and used ILTIS for selecting ROIs, and a previous version of PyView (VIEW, written in IDL® language) for data analysis. Odor responses in eight different olfactory receptor populations where characterized for a large panel of odorants and mixtures in Drosophila using the calcium reporter GCamP, and VIEW was used to quantify the data (Münch & Galizia, 2017). In fact, VIEW had been developed over more than two decades (Galizia et al., 1999; Sachse et al., 1999), and put to fruit in analyzing olfactory coding in the insect brain (Galizia, 2014; Münch & Galizia, 2016; Paoli & Galizia, 2021). However, it was difficult to share among labs due to the proprietary nature of the language used (IDL®). These studies motivated us to create a new version of the analysis pipeline, with improved features, and written in a language that is freely available (Python): PyView as presented here.

Currently the following data formats can be loaded into PyView: Till Photonics files (TillVision), generic TIFF files/OME files (https://www.openmicroscopy.org/ome-files) with metadata (as created, for example, by Till Photonics LiveAcquisition software), Zeiss confocal/2-photon data, Leica confocal/2-photon data (.lif). Example datasets for all implemented formats have been published at https://doi.gin.g-node.org/10.12751/g-node.4c44i5. Readers for other formats are easy to implement.

Conclusions

PyView provides a flexible package for optical imaging data analysis, affording rigorous data treatment using identical settings across collections of many measurements.

Acknowledgments

Thanks for providing sample data sets, testing the software, and giving useful advice go to all members of the Galizia lab, in particular to Alja Lüdke, Hanna Schnell, Marco Stucchi, Marco Paoli, Inga Petelski, Sercan Sayin, Paul Szyszka. Datasets were also provided by Bente Berg and Xi Chu from NTNU Trondheim, and by Martin Giurfa, Marco Paoli, and Maria Gabriela de Brito Sanchez from the Centre de Recherches sur la Cognition Animale, Toulouse. Funded by the Federal Ministry of Education and Research (BMBF) and the Baden-Württemberg Ministry of Science as part of the Excellence Strategy of the German Federal and State Governments

References

Buccino, A. P., Hurwitz, C. L., Garcia, S., Magland, J., Siegle, J. H., Hurwitz, R., & Hennig, M. H. (2020). SpikeInterface, a unified framework for spike sorting. *eLife*, *9*, e61834. https://doi.org/10.7554/elife.61834

Dubbs, A., Guevara, J., & Yuste, R. (2016). moco: Fast motion correction for calcium imaging. *Frontiers in Neuroinformatics*, *10*, 6. https://doi.org/10.3389/fninf.2016.00006

Dupuy, F., Casas, J., Bagnères, A.-G., & Lazzari, C. R. (2009). OpenFluo: A free open-source software for optophysiological data analyses. *Journal of Neuroscience Methods*, *183*(2), 195–201. https://doi.org/10.1016/j.jneumeth.2009.06.031

Galizia, G. C. (2014). Olfactory coding in the insect brain: Data and conjectures. *European Journal of Neuroscience*, *39*(11), 1784–1795. https://doi.org/10.1111/ejn.12558

Galizia, G. C., Sachse, S., Rappert, A., & Menzel, R. (1999). The glomerular code for odor representation is species specific in the honeybee Apis mellifera. *Nature Neuroscience*, *2*(5), 473–478. https://doi.org/10.1038/8144

Galizia, G. C., & Vetter, R. (2004). Optical methods for analyzing odor-evoked activity in the insect brain. In *Methods in insect sensory neuroscience*. CRC Press. https://doi.org/10.1201/9781420039429

Giovannucci, A., Friedrich, J., Gunn, P., Kalfon, J., Brown, B. L., Koay, S. A., Taxidis, J., Najafi, F., Gauthier, J. L., Zhou, P., Khakh, B. S., Tank, D. W., Chklovskii, D. B., &

Pnevmatikakis, E. A. (2019). CaImAn an open source tool for scalable calcium imaging data analysis. *eLife*, *8*, e38173. https://doi.org/10.7554/elife.38173

Göbel, W., & Helmchen, F. (2007). In vivo calcium imaging of neural network function. *Physiology*, *22*(6), 358–365. https://doi.org/10.1152/physiol.00032.2007

Lüdke, A., Raiser, G., Nehrkorn, J., Herz, A. V. M., Galizia, G. C., & Szyszka, P. (2018). Calcium in Kenyon cell somata as a substrate for an olfactory sensory memory in Drosophila. *Frontiers in Cellular Neuroscience*, *12*, 128. https://doi.org/10.3389/fncel.2018.00128

Marstal, K., Berendsen, F., Staring, M., & Klein, S. (2016, June). SimpleElastix: A user-friendly, multi-lingual library for medical image registration. *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. https://doi.org/10.1109/cvprw.2016.78

Münch, D., & Galizia, G. C. (2016). DoOR 2.0 - comprehensive mapping of Drosophila melanogaster odorant responses. *Scientific Reports*, *6*(1). https://doi.org/10.1038/srep21841

Münch, D., & Galizia, G. C. (2017). Take time: Odor coding capacity across sensory neurons increases over time in Drosophila. *Journal of Comparative Physiology A*, *203*(12), 959–972. https://doi.org/10.1007/s00359-017-1209-1

Paoli, M., & Galizia, G. C. (2021). Olfactory coding in honeybees. *Cell and Tissue Research*, *383*(1), 35–58. https://doi.org/10.1007/s00441-020-03385-5

Pnevmatikakis, E. A. (2019). Analysis pipelines for calcium imaging data. *Current Opinion in Neurobiology*, *55*, 15–21. https://doi.org/10.1016/j.conb.2018.11.004

Rad, M. S., Choi, Y., Cohen, L. B., Baker, B. J., Zhong, S., Storace, D. A., & Braubach, O. R. (2017). Voltage and calcium imaging of brain activity. *Biophysical Journal*, *113*(10), 2160–2167. https://doi.org/10.1016/j.bpj.2017.09.040

Radstake, F. D. W., Raaijmakers, E. A. L., Luttge, R., Zinger, S., & Frimat, J. P. (2019). CALIMA: The semi-automated open-source calcium imaging analyzer. *Computer Methods and Programs in Biomedicine*, *179*, 104991. https://doi.org/10.1016/j.cmpb.2019.104991

Romano, S. A., Pérez-Schuster, V., Jouary, A., Boulanger-Weill, J., Candeo, A., Pietri, T., & Sumbre, G. (2017). An integrated calcium imaging processing toolbox for the analysis of neuronal population dynamics. *PLOS Computational Biology*, *13*(6), e1005526. https://doi.org/10.1371/journal.pcbi.1005526

Sachse, S., Rappert, A., & Galizia, G. C. (1999). The spatial representation of chemical structures in the antennal lobe of honeybees: Steps towards the olfactory code. *European Journal of Neuroscience*, *11*(11), 3970–3982. https://doi.org/10.1046/j.1460-9568.1999.00826.x

Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J.-Y., White, D. J., Hartenstein, V., Eliceiri, K., Tomancak, P., & Cardona, A. (2012). Fiji: An open-source platform for biological-image analysis. *Nature Methods*, *9*(7), 676–682. https://doi.org/10.1038/nmeth.2019

Soelter, J., Schumacher, J., Spors, H., & Schmuker, M. (2014). Automatic segmentation of odor maps in the mouse olfactory bulb using regularized non-negative matrix factorization. *NeuroImage*, *98*, 279–288. https://doi.org/10.1016/j.neuroimage.2014.04.041

Strauch, M., Rein, J., Lutz, C., & Galizia, G. C. (2013). Signal extraction from movies of honeybee brain activity: The ImageBee plugin for KNIME. *BMC Bioinformatics*, *14*(S18). https://doi.org/10.1186/1471-2105-14-s18-s4