DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

- @Pending Reviewers

Submitted: N/A
Published: N/A

Derotation: a Python package for correcting rotation-induced distortions in line-scanning microscopy

Laura Porta[1], Adam L. Tyson[1, 2], and Troy W. Margrie[1]

1 Sainsbury Wellcome Centre, University College London, UK **2** Gatsby Computational Neuroscience Unit, University College London, London, United Kingdom

Summary

Line-scanning microscopy, including multiphoton calcium imaging, is a powerful technique for observing dynamic processes at cellular resolution. However, when the imaged sample rotates during acquisition, the sequential line-by-line scanning process introduces geometric distortions. These artifacts, which manifest as shearing or curving of features, can severely compromise downstream analyses such as motion registration, cell detection, and signal extraction. While several studies have developed custom solutions for this issue (Vélez-Fort et al., 2018) (Hennestad et al., 2021) (Sit & Goard, 2023) (Voigts & Harnett, 2020), a general-purpose, accessible software package has been lacking.

`derotation` is an open-source Python package that algorithmically reconstructs movies from data acquired during sample rotation. By leveraging recorded rotation angles and the microscope's line acquisition clock, the software applies a precise, line-by-line inverse transformation to restore the original geometry of the imaged plane. This correction enables reliable cell segmentation during rapid rotational movements, making it possible to study yaw motion without sacrificing image quality (Figure 1).

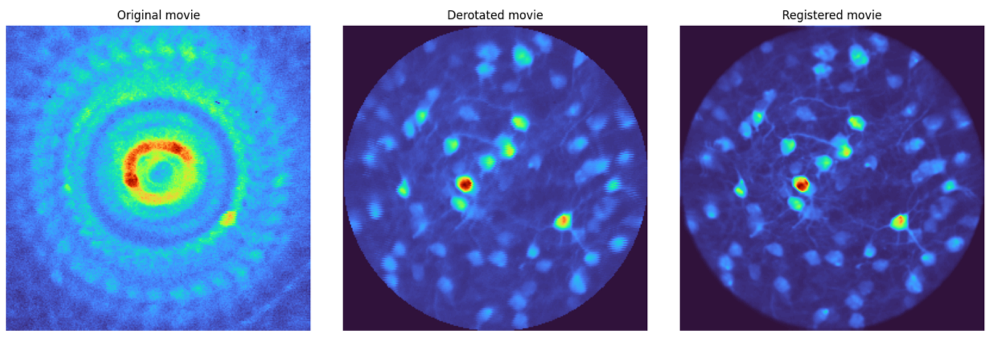

Figure 1: Example of `derotation` correction. On the left, the mean image of a 3-photon movie in which the sample was rotating. In the center, the mean image after derotation, and on the left the mean image of the derotated movie after suite2p registration (Pachitariu et al., 2016). As you can see, already after derotation the cells are visible and have well defined shapes.

Statement of Need

Any imaging modality that acquires data sequentially, such as line-scanning microscopy, is susceptible to motion artifacts if the sample moves during the acquisition of a single frame. When this motion is rotational, it produces characteristic "fan-like" distortions that corrupt the morphological features of the imaged structures (Figure 2). This significantly complicates, or even prevents, critical downstream processing steps like cell segmentation and automated region-of-interest tracking.

This problem is particularly acute in systems neuroscience, where researchers increasingly combine two-photon or three-photon calcium imaging with behavioral paradigms involving head rotation (Vélez-Fort et al., 2018) (Hennestad et al., 2021) (Sit & Goard, 2023) (Voigts & Harnett, 2020). In such experiments, where head-fixed animals may be passively or actively rotated, high-speed angular motion can render imaging data unusable without correction. The issue is even more acute in imaging modalities with lower frame rates, such as three-photon calcium imaging. While individual labs have implemented custom scripts to address this, there has been no validated, open-source, and easy-to-use Python tool available to the broader community.

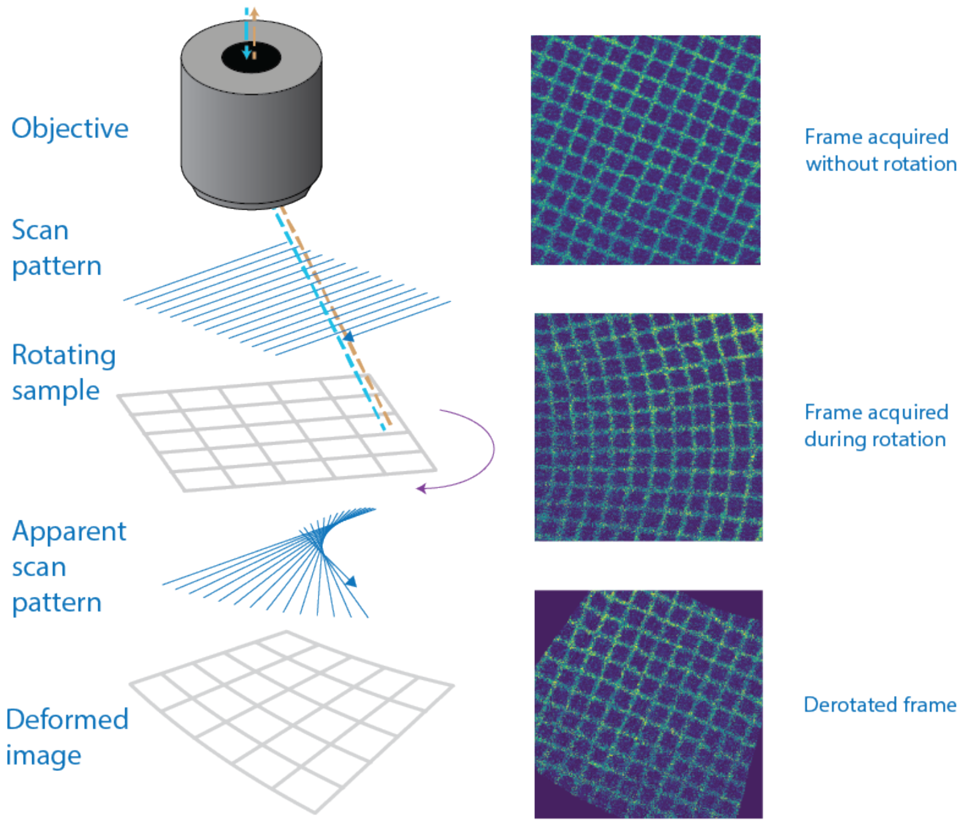

Figure 2: Schematic of line-scanning microscope distortion. Left: scanning pattern plus sample rotation lead to fan-like artifacts. Right: grid imaged while still (top), while rotating (middle), and after `derotation` (bottom), showing alignment restoration.

`derotation` directly fills this gap by providing a documented, tested, and modular solution for post hoc correction of imaging data acquired during rotation. It enables researchers to perform quantitative imaging during high-speed rotational movements. By providing a robust and accessible tool, `derotation` lowers the barrier for entry into complex behavioral experiments and improves the reproducibility of a key analysis step in a growing field of research.

Functionality

The core of the `derotation` package is a line-by-line affine transformation. It operates by first establishing a precise mapping between each scanned line in the movie and the rotation angle of the sample at that exact moment in time. It then applies an inverse rotation transform to each line around a specified or estimated center of rotation. Finally, the corrected lines are reassembled into frames, producing a movie that appears as if the sample had remained stationary.

Data Ingestion and Synchronization

The package accepts two types of input formats depending on the processing approach:

For pipeline workflows (`FullPipeline` and `IncrementalPipeline`):
These pipelines are designed for experimental setups with synchronized rotation and imaging data. The required inputs are:

- An array of analog signals containing timing and rotation information, typically including the start of a new line and frame, when the rotation system is active, and the rotation position feedback;
- A **CSV file** describing speeds and directions.

For low-level core function:
Advanced users can bypass the pipeline workflows and use the core transformation function directly by providing the original movie and an array of rotation angles for each line.

This modular design allows users with custom experimental setups to integrate the `derotation` algorithm into their own analysis scripts while still benefiting from the core transformation logic.

Processing Pipelines

For ease of use, `derotation` provides two high-level processing workflows tailored to common experimental paradigms. These pipelines handle data loading, parameter validation, processing, and saving outputs, while also generating logs and debugging plots to ensure quality control.

- `FullPipeline` is engineered for experimental paradigms involving randomized, clockwise or counter-clockwise rotations. It assumes that there will be complete 360° rotations of the sample. As part of its workflow, it can optionally estimate the center of rotation automatically using Bayesian optimization, which minimizes residual motion in the corrected movie.
- `IncrementalPipeline` is optimized for stepwise, single-direction rotations. This rotation paradigm is useful for calibration of the luminance across rotation angles. It can also provide an alternative estimate of the center of rotation, fitting the trajectory of a cell across rotation angles.

Both pipelines are configurable via YAML files or Python dictionaries, promoting reproducible analysis by making it straightforward to document and re-apply the same parameters across multiple datasets.

Upon completion, a pipeline run generates a comprehensive set of outputs: the corrected movie, a CSV file with rotation angles and metadata for each frame, debugging plots, a text file containing the estimated optimal center of rotation, and log files with detailed processing information.

Synthetic Data Generation

For further development and testing, `derotation` includes a synthetic data generator that can create challenging synthetic datasets with misaligned centers of rotation and out-of-plane rotations. This feature is particularly useful for validating the robustness of the `derotation` algorithm and for developing new features.

The synthetic data can be generated using the following classes:

- `Rotator` class: Core class that applies line-by-line rotation to an image stack, simulating a rotating microscope.

- `SyntheticData` class: Creates fake cell images, assigns rotation angles, and generates synthetic stacks leveraging the `Rotator` class. It is a complete synthetic dataset generator.

Documentation and Installation

`derotation` is available on PyPI and can be installed with `pip install derotation`. It is distributed under a BSD-3-Clause license. Comprehensive documentation, tutorials, and example datasets are available at https://derotation.neuroinformatics.dev. Using Binder, users can run the software in a cloud-based environment with sample data without requiring any local installation.

Future Directions

Derotation is currently used to process 3-photon movies acquired during head rotation. Future directions can include further automated pipelines for specific motorised stages and experimental paradigms.

Acknowledgements

We thank Simon Weiler, Mateo Vélez-Fort, and Chryssanthi Tsitoura for providing three-photon imaging datasets used during development and testing, and for feedback on the package. We also thank Igor Tatarnikov for contributing to the development of the package and the whole Neuroinformatics Unit. This package was inspired by previous work on `derotation` as described in (Voigts & Harnett, 2020). (TODO: Add grant information.)

References

Hennestad, E., Witoelar, A., Chambers, A. R., & Vervaeke, K. (2021). Mapping vestibular and visual contributions to angular head velocity tuning in the cortex. *Cell Reports*, *37*(12), 110134. https://doi.org/10.1016/j.celrep.2021.110134

Pachitariu, M., Stringer, C., Dipoppa, M., Schröder, S., Rossi, L. F., Dalgleish, H., Carandini, M., & Harris, K. D. (2016). *Suite2p: Beyond 10,000 neurons with standard two-photon microscopy*. Neuroscience. https://doi.org/10.1101/061507

Sit, K. K., & Goard, M. J. (2023). Coregistration of heading to visual cues in retrosplenial cortex. *Nature Communications*, *14*(1), 1992. https://doi.org/10.1038/s41467-023-37704-5

Vélez-Fort, M., Bracey, E. F., Keshavarzi, S., Rousseau, C. V., Cossell, L., Lenzi, S. C., Strom, M., & Margrie, T. W. (2018). A Circuit for Integration of Head- and Visual-Motion Signals in Layer 6 of Mouse Primary Visual Cortex. *Neuron*, *98*(1), 179–191.e6. https://doi.org/10.1016/j.neuron.2018.02.023

Voigts, J., & Harnett, M. T. (2020). Somatic and Dendritic Encoding of Spatial Variables in Retrosplenial Cortex Differs during 2D Navigation. *Neuron*, *105*(2), 237–245.e4. https://doi.org/10.1016/j.neuron.2019.10.016