

NiTransforms: A Python tool to read, represent, manipulate, and apply n -dimensional spatial transforms

Mathias Goncalves¹, Christopher J. Markiewicz^{1, 2}, Stefano Moia⁴, Satrajit S. Ghosh^{2, 3}, Russell A. Poldrack¹, and Oscar Esteban¹

¹ Department of Psychology, Stanford University, Stanford, CA, USA ² McGovern Institute for Brain Research, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA ³ Department of Otolaryngology, Harvard Medical School, Boston, MA, USA ⁴ Basque Center on Cognition Brain and Language, San Sebastian, Spain

DOI: [10.21105/joss.03459](https://doi.org/10.21105/joss.03459)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Editor: [Øystein Sørensen](#) ↗

Reviewers:

- [@robbisg](#)
- [@PeerHerholz](#)

Submitted: 29 June 2021

Published: 10 September 2021

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Introduction

Spatial transforms formalize mappings between coordinates of objects in biomedical images. Transforms typically are the outcome of image registration methodologies, which estimate the alignment between two images. Image registration is a prominent task present in almost any image processing workflow.

Statement of need. In neuroimaging, the proliferation of image registration software implementations has resulted in a disparate collection of structures and file formats used to preserve and communicate the transformation. This assortment of formats presents the challenge of compatibility between tools and endangers the reproducibility of results. Some tools are available that permit some conversions between formats, either within neuroimaging packages or standalone such as Convert3D ([Yushkevich, n.d.](#)). However, they are typically limited either in compatible packages and/or application coverage (e.g., only linear transforms).

Summary. *NiTransforms* is a Python tool capable of reading and writing transforms produced by the most popular neuroimaging software (AFNI ([Cox & Hyde, 1997](#)), FSL ([Jenkinson et al., 2012](#)), FreeSurfer ([Fischl, 2012](#)), ITK via ANTs ([Avants et al., 2008](#)), and SPM ([Friston et al., 2006](#))). Additionally, the tool provides seamless conversion between these formats, as well as the ability of applying the transforms to other images. The tool has already been integrated into *fMRIPrep* ([Esteban et al., 2019](#)), a popular neuroimaging preprocessing pipeline that leverages many of the neuroimaging software already mentioned. *NiTransforms* is inspired by *NiBabel* ([Brett et al., 2006](#)), a Python package with a collection of tools to read, write and handle neuroimaging data, and will be included as a new module.

Audience. Computer vision researchers and experts using Python, developers of neuroimaging workflows built on AFNI, FSL, FreeSurfer, ITK/ANTs, or SPM, developers of neuroimaging visualization tools.

Implementation

We first mathematically formulate the problem of spatial alignment of images and highlight common pitfalls. We then justify the architectural design of *NiTransforms* and describe the major elements of the implementation.

Methods

Let \vec{x} represent the coordinates of a point in the reference coordinate system R , and \vec{x}' its projection on to another coordinate system M :

$$T: R \subset \mathbb{R}^n \rightarrow M \subset \mathbb{R}^n$$

$$\vec{x} \mapsto \vec{x}' = f(\vec{x}).$$

In an image registration problem, M is a moving image from which we want to sample data in order to bring the image into spatial alignment with the reference image R . Hence, f here is the spatial transformation function that maps from coordinates in R to coordinates in M . There are a multiplicity of image registration algorithms and corresponding image transformation models to estimate linear and nonlinear transforms.

The problem has been traditionally confused by the need of *transforming* or mapping one image (generally referred to as *moving*) into another that serves as reference, with the goal of *fusing* the information from both. An example of image fusion application would be the alignment of functional data from one individual's brain to the same individual's corresponding anatomical MRI scan for visualization. Therefore, "applying a transform" entails two operations (Figure 1): first, transforming the coordinates of the samples in the reference image R to find their mapping \vec{x}' on M via $T\{\cdot\}$, and second an interpolation step, as \vec{x}' will likely fall off-the-grid of the moving image M . These two operations are confusing because, while the spatial transformation projects from R to M , the data flows in reversed way after the interpolation of the values of M at the mapped coordinates \vec{x}' .

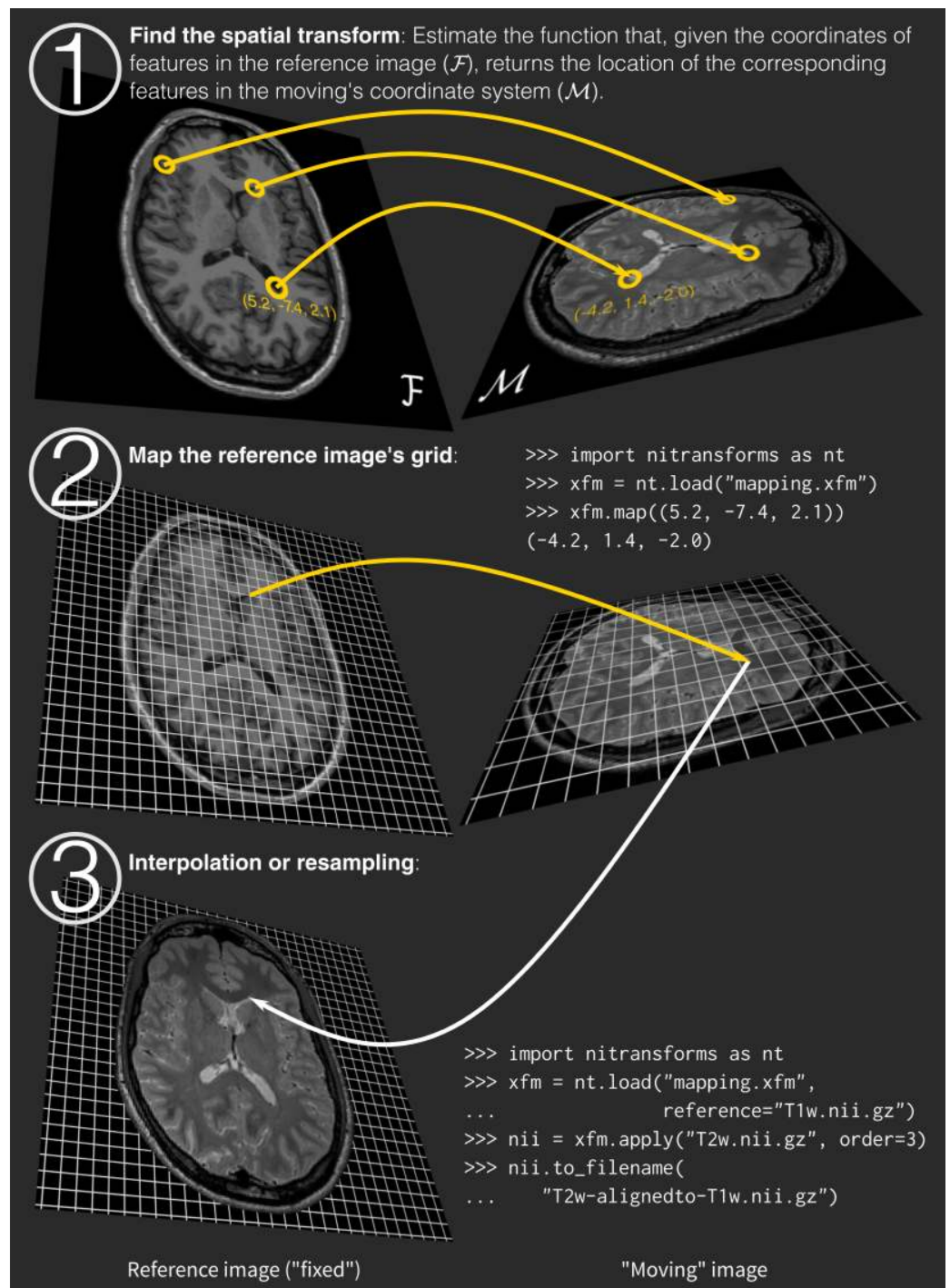


Figure 1: Resampling a 3D image via a spatial transform to fuse the information of one into another image.

Software Architecture

There are four main components within the tool: an `io` submodule to handle the structure of the various file formats, a `base` submodule where abstract classes are defined, a `linear` submodule implementing n -dimensional linear transforms, and a `nonlinear` submodule for both parametric and non-parametric nonlinear transforms. Furthermore, *NiTransforms* provides

a straightforward *Application Programming Interface* (API) that allows researchers to map point sets via transforms, as well as apply transforms (i.e., mapping the coordinates and interpolating the data) to data structures with ease.

To ensure the consistency and uniformity of internal operations, all transforms are defined using a left-handed coordinate system of physical coordinates. In words from the neuroimaging domain, the coordinate system of transforms is *RAS+* (or positive directions point to the Righthand for the first axis, Anterior for the second, and Superior for the third axis). The internal representation of transform coordinates is the most relevant design decision, and implies that a conversion of coordinate system is necessary to correctly interpret transforms generated by other software. When a transform that is defined in another coordinate system is loaded, it is automatically converted into *RAS+* space.

NiTransforms was developed using a test-driven development paradigm, with the battery of tests being written prior to the software implementations. Two categories of tests were used: unit tests and cross-tool comparison tests. Unit tests evaluate the formal correctness of the implementation, while cross-tool comparison tests assess the correct implementation of third-party software. The testing suite is incorporated into a continuous integration framework, which assesses the continuity of the implementation along the development life and ensures that code changes and additions do not break existing functionalities.

References

- Avants, B. B., Epstein, C. L., Grossman, M., & Gee, J. C. (2008). Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain. *Medical Image Analysis*, 12(1), 26–41. <https://doi.org/10.1016/j.media.2007.06.004>
- Brett, M., Markiewicz, C. J., Hanke, M., Cote, M.-A., Cipollini, B., McCarthy, P., Cheng, C., Halchenko, Y. O., Ghosh, S. S., Larson, E., Wassermann, D., & Gerhard, S. (2006). Open Source Software: NiBabel. *Zenodo*, 3458246. <https://doi.org/10.5281/zenodo.591597>
- Cox, R. W., & Hyde, J. S. (1997). Software tools for analysis and visualization of fMRI data. *NMR Biomed*, 10(4-5), 171–178. [https://doi.org/10.1002/\(SICI\)1099-1492\(199706/08\)10:4/5%3C171::AID-NBM453%3E3.0.CO;2-L](https://doi.org/10.1002/(SICI)1099-1492(199706/08)10:4/5%3C171::AID-NBM453%3E3.0.CO;2-L)
- Esteban, O., Markiewicz, C. J., Blair, R. W., Moodie, C. A., Isik, A. I., Erramuzpe, A., Kent, J. D., Goncalves, M., DuPre, E., Snyder, M., Oya, H., Ghosh, S. S., Wright, J., Durnez, J., Poldrack, R. A., & Gorgolewski, K. J. (2019). fMRIPrep: A robust preprocessing pipeline for functional MRI. *Nat Meth*, 16(1), 111–116. <https://doi.org/10.1038/s41592-018-0235-4>
- Fischl, B. (2012). FreeSurfer. *NeuroImage*, 62(2), 774–781. <https://doi.org/10.1016/j.neuroimage.2012.01.021>
- Friston, K. J., Ashburner, J., Kiebel, S. J., Nichols, T. E., & Penny, W. D. (2006). *Statistical parametric mapping: The analysis of functional brain images*. Academic Press. ISBN: 978-0-12-372560-8
- Jenkinson, M., Beckmann, C. F., Behrens, T. E. J., Woolrich, M. W., & Smith, S. M. (2012). FSL. *NeuroImage*, 62(2), 782–790. <https://doi.org/10.1016/j.neuroimage.2011.09.015>
- Yushkevich, P. A. (n.d.). Open Source Software: Convert3D Medical Imaging Processing tool. *Sourceforge*. <https://sourceforge.net/p/c3d>