MNE-Python: Intracranial Electrode Localization Toolbox

About me

1. Alex Rockhill (GitHub: @alexrockhill)
2. University of Oregon / PhD student / Department of Human Physiology / 2nd year / graduation expected in 2024
3. Time zone: Pacific Standard Time (PST) Los Angeles (Eugene, OR)

Code contribution

- https://github.com/mne-tools/mne-python/pull/6910
- https://github.com/mne-tools/mne-python/pull/9087

Project information

1. Sub-org name

MNE-Python

2. Project Abstract

Intracranial electrophysiology is commonly recorded during and after invasive brain surgeries in epilepsy patients to determine where seizure activity starts. These recordings are also used to learn about normal brain function under various experimental conditions. The first step in analyzing intracranial electrophysiology recordings is determining the location of the electrode contacts in the coordinate frame of a subject's T1-weighted magnetic resonance image (MRI). The subject must also have a computed tomography (CT) image taken during or after the implantation surgery. The pre-surgical MRI has better resolution of the brain anatomy than the CT, whereas the CT shows the positions of the electrode contacts. These two 3D images allow the location of the electrode contacts to be associated with anatomical brain structures. MNE-Python has existing tools for analyzing time series data recorded by intracranial electrode contacts but there is a gap in the functionality in being able to label the location of intracranial electrode contacts. The proposed project is to add a graphical user interface (GUI) that allows users to localize electrode contacts as an enhancement to MNE-Python.

3. Detailed description

Introduction

There is a need for an open-source software tool to localize electrode contacts in a CT image and label the anatomical location of these contacts based on MRI data that has a high development standard. In a typical intracranial electrophysiology workflow, the raw data collected by the experimenter consists of a preoperative MRI, a postoperative CT and the time series data from the electrophysiology recording. Currently, MNE-Python contains examples of how to analyze and visualize intracranial electrophysiology recordings (e.g. https://mne.tools/dev/auto_tutorials/misc/plot_ecog and https://mne.tools/dev/auto_tutorials/misc/plot_seeg) but these examples assume that the electrode contact locations have already been determined. Electrode contact localization software exists (e.g. Brainstorm https://neuroimage.usc.edu/brainstorm/) but these software tools are either commercial products or rely on commercial software such as MATLAB and can be difficult to integrate into a typical MNE-Python workflow. These constrain the ability of users to modify and contribute to the source code and access to the software in order to reproduce findings, just to mention two of many examples of the benefits of open-source software. In addition, most of these contact localization workflows are designed for electrocorticography (ECoG), while stereo-electroencephalography (SEEG) is an increasingly common implantation being used by epilepsy teams around the world (Yan et al., 2019).

A working electrode contact localization GUI and overall pipeline was published in 2017 https://github.com/ChangLabUcsf/img_pipe. This project was improved by myself for code efficiency and readability (https://github.com/alexrockhill/img_pipe). New functions were added for user-friendliness and to deal with SEEG . Yet, the software package is not yet up to standards of usability and documentation for continued open source development and is not integrated into the MNE-Python environment yet. This project proposal is to integrate the existing code into MNE, refining the codebase in the process to reach the high software engineering standards of MNE-Python. Permission for adaptation from the authors of the original code has been obtained.

Workflow

The electrode contact localization GUI is the centerpiece of the proposed module to be added but there are several other steps to the pipeline. First, a freesurfer reconstruction using a subject's T1-weighted image must be computed to label brain areas as well as combined volumetric and surface coregistration to morph those brain areas to a common atlas. Next, the CT must be aligned to the MRI. Then, using electrode contact names from the electrophysiology recording, a GUI allows the user to go through and select the location of each electrode contact, associating it with the channel name from the recording. The user ends with labelled electrodes both in the patient's anatomical coordinates and in coordinates of a common atlas.

CLI/API

A possible command line call might look something like this:

```
$ mne_ieeg_localize --t1 ./anat/T1.mgz --ct ./ct/ct.mgz --raw
./ieeg/sub-1_ses-1_task-mytask-raw.fif  --fs_subj_dir ./derivatives/sub-1
```

Here, the output structure would default to BIDS format including the CT, (after being aligned to the MRI) the MRI and the ieeg data file and the channel location data.

The API will be documented with Sphinx in the style of MNE and syntax will look something like this:

First, we need the image data from the subject. The T1-weighted MRI from which the anatomy is determined and the CT from which the electrode contact locations are determined.

```
>>> from  mne_ieeg_localize import align_ct_to_mri
>>> ct_fname = './ct/ct.mgz'
>>> mri_fname = './anat/T1.mgz'
```

Then, we align the CT to the MRI so that the electrode locations are in terms of the subjects anatomy via the freesurfer reconstruction of the T1-weighted image.

```
>>> ct_aligned = align_ct_to_mri(ct_fname, mri_fname)
```

Next, we prepare to launch the electrode contact picking GUI. The freesurfer reconstruction directory is an added input so that we can visualize the pial surface (this is important for ECoG to check that the electrodes contacts are properly localized to the pial surface).

```
>>> fs_subj_dir = './derivatives/sub-1'
>>> ct_aligned_fname =  './ct/ct_aligned.mgz'
>>> raw_fname = './ieeg/sub-1_ses-1_task-mytask-raw.fif'
>>> raw = mne_localize.localize_electrodes(raw_fname, ct_fname, mri_fname, fs_subj_dir)
```

At the end, the raw object contains the locations of each of the electrode contacts in the T1-weighted image coordinate frame.

Implementation Details

Some notes on potential implementation roadblocks and challenges along with potential solutions are outlined below:

- Guiding users through the freesurfer pipeline: Currently there are wrappers for freesurfer functions, command line examples in the documentation would be better as these are freesurfer functions and don't interface with the core functionality besides as inputs.
- CT-MRI coregistration: This requires nipy dependency but the nipy function call

(https://nipy.org/nipy/api/generated/nipy.algorithms.registration.histogram_registration) is near depreciation so this could be rewritten and the dependency could be removed. Additionally, the coregistration could be improved by having the user select fiducials so that the initial alignment is close. Currently the initial alignment is often very far away because of a translation which requires a very cumbersome manual intervention (e.g. in freeview).

- Making the core GUI functionality compatible with typical laptop processor speeds: img_pipe currently functions on a laptop with a second or more of lag between user interactions and minimal lag on a typical workstation. By using a visualization toolkit (VTK) plot embedded in the PyQt5/6 GUI, the processing should be sped up to be as responsive as comparable 3D brain imaging-manipulation GUIs (e.g. freesurfer) instead of relying on matplotlib to handle dynamic plotting. The GUI is shown in Figure 1.
- Adding automated contact selection: Using a marching cubes algorithm, a reasonable segmentation of electrode contacts based on user-provided (and adjustable) constraints on the volume of the contacts has already been achieved but needs to be improved and integrated into the GUI. This would allow the user not to have to click around in 3D to find the center of the contact but, instead, to be led directly to each contact (the user still needs to determine which channel each contact corresponds to and skip any false positives).
- Adding volumetric models for more exact localizations: With electrode contact segmentation based on the anatomical image, it will be possible to use the volumetric model of the 3D contact to determine a more accurate estimate of the position of the contact than a user-selected voxel or even using the brightest voxel in the vicinity.
- Writing tests and documentation: There are a few tests written but the majority of tests and documentation need to be written. There will be tests for all the core functions and several tests using fake clicks on a non-interactive backend. Documentation will use Sphinx and the typical MNE-Python aesthetic.

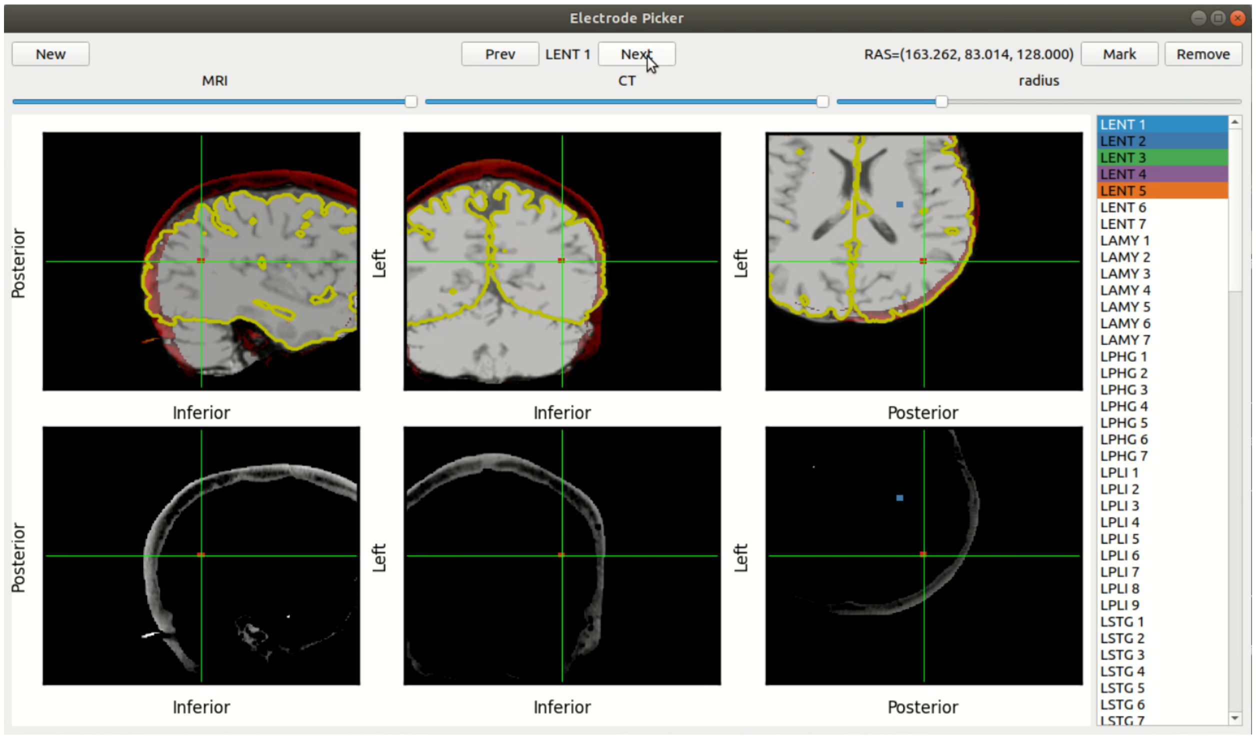

Figure 1. The GUI for the electrode picker. The top three subplots show the MRI with a mostly transparent CT overlaid (the electrodes and skill are not transparent). The bottom three subplots show just the CT. By navigating around the brain, the user is able to find bright voxels in 3D where the electrodes are and mark them, associating them with one of the channel names from the recording on the right.

4. Weekly timeline

There are several enhancements but the scope of the project is such that by working full-time for two-and-a-half months, a complete and functional beta version of the package should be finished. Writing the tests and documentation should take approximately 4-5 weeks. The core functionality is already written but should be able to be significantly improved in 2-3 weeks. If the core functionality enhancements are not able to be finished in that time frame, they can be added in follow-up pull requests, while still delivering a viable product.

June 7, 2021 - June 11, 2021

- Use mne project template (https://github.com/mne-tools/mne-project-template) to create a new project called mne-ieeg-localize, do preliminary set up
  - API goal: remove mne_localize.recon_all, mne_localize.warp_to_template and put in documentation

June 14, 2021 - June 18, 2021

- Integrate existing code from (https://github.com/alexrockhill/img_pipe)
  - API goal: mne_ieeg_localize.coreg_ct_mr

June 21, 2021 - June 25, 2021

- Improve computation so that code can execute on typical laptop processor speed using VTK backend
    - API goal: mne_ieeg_localize.coreg

June 28, 2021 - July 9, 2021 (two weeks)

- Improve algorithm for automatic detection and electrode labelling
    - API goal: within GUI button/interaction to automatically segment marching cube CT volumes into devices for faster identification

July 12, 2021 - July 16, 2021

- Write tests to core functionality
    - API goal: mne_ieeg_localize.export/export button to export locations of the electrodes and their labels in subject scanner coordinates as well as atlas coordinates

July 19, 2021 - July 23, 2021

- Write tests for graphical user interface using fake clicks to non-interactive backend

July 26, 2021 - July 30, 2021

- Finish tests

August 2, 2021 - August 6, 2021

- Write examples and documentation

August 9, 2021 - August 16, 2021

- Write examples and documentation

Conclusion

Overall, this project is well-poised to be finished in the allotted time span and having a complete processing pipeline for intracranial electrophysiologists contained within MNE-Python would increase the user base of mne and further scientific discovery powered by open source software.

References

Hamilton LS, Chang DL, Lee MB and Chang EF (2017) Semi-automated Anatomical Labeling and Inter-subject Warping of High-Density Intracranial Recording Electrodes in Electrocorticography. Front. Neuroinform. 11:62. doi: 10.3389/fninf.2017.00062

Yan, H., et al. (2019). Method of invasive monitoring in epilepsy surgery and seizure freedom and morbidity: A systematic review. Epilepsia.