O8T Medical Imaging Engineer Coding Challenge

Revision	4
Last Updated	25-Feb-2025

Introduction

The medical imaging pipeline in the Quicktome product employs state-of-the-art techniques to process neurological MRI scans and provide insights for neurosurgeons for surgery planning. It is part of a large, complex set of software components which run in the cloud. Hospitals send brain scans from their PACS to our cloud platform, the data is processed, and the results visualised by clinicians in a web-based GUI which displays 3D volumes, tractography, parcellations and more.

You are tasked with developing a simple pipeline for preprocessing brain imaging data. This challenge will test your ability to handle image data, implement efficient processing algorithms, select suitable libraries, functions and parameters, and create maintainable, well-documented code.

Task Description

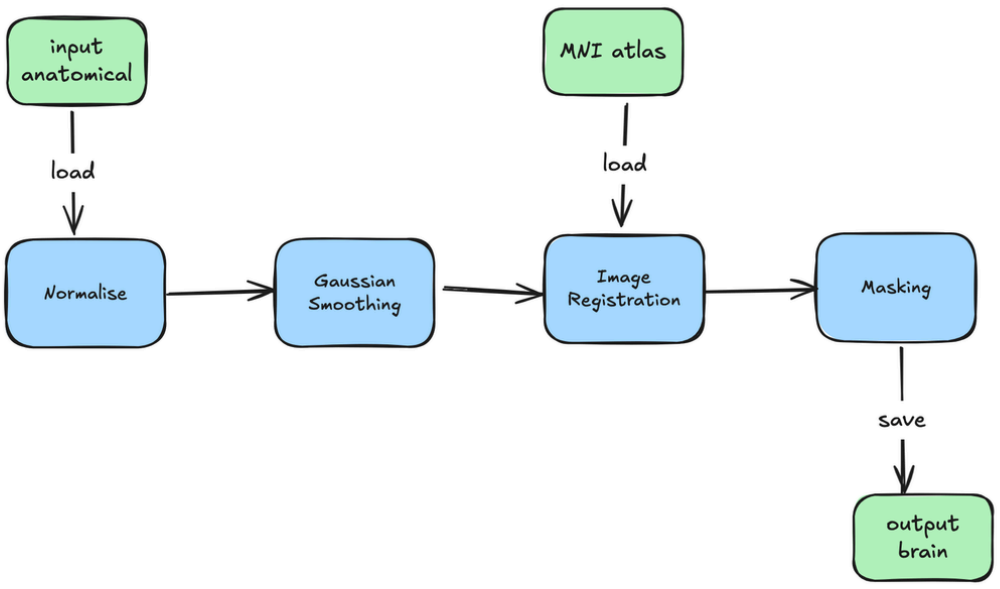
Create a Python-based processing pipeline that performs *atlas-based skull stripping* using the following steps on a 3D brain scan (T1 in DICOM or NIFTI format):

1. Load and validate the input volume
2. Implement basic preprocessing:
   - Normalise image intensities (Z-score or min-max)
   - Apply Gaussian smoothing
3. Perform atlas-based skull stripping:
   - Load the MNI brain atlas template and mask
   - Register the input image to the atlas template
   - Apply the transformed brain mask to extract the brain region
4. Save the segmented brain to an output file
5. Implement proper error handling and logging throughout the pipeline

The solution should be a command-line tool which takes suitable arguments for input and output filename, filter parameters, etc and have sensible defaults. For example:

```
python process_mri.py                   \
    --input subject2_anat.nii.gz        \
    --output processed_output.nii.gz    \
    --sigma 1.5
```

The resulting data flow might look like:



Extensions (Optional)

For candidates who complete the main task quickly, the following extensions can be attempted:

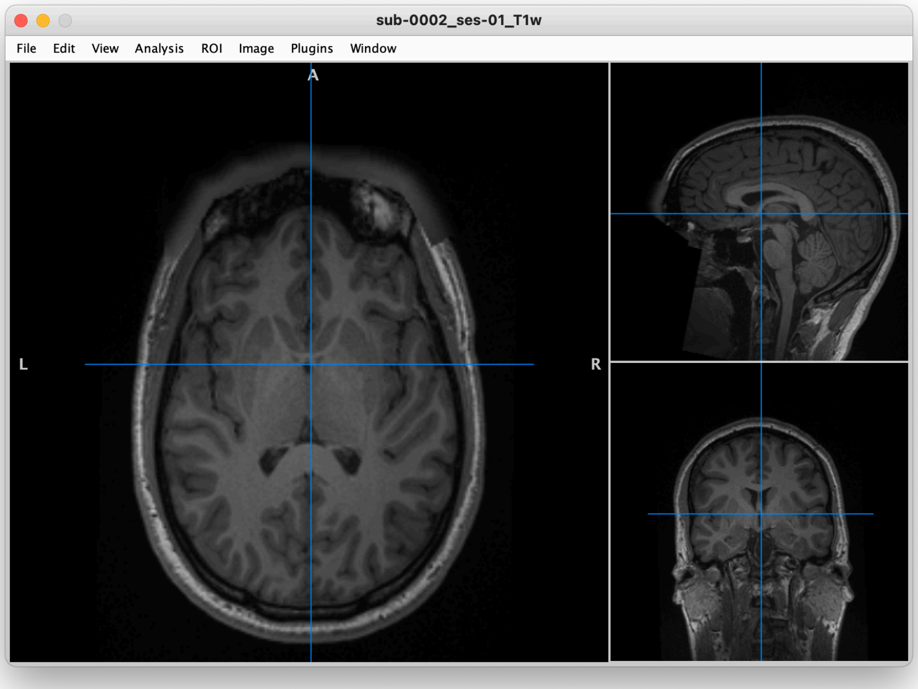
1. Add quality assessment metrics to evaluate the skull stripping results
2. Implement a simple reporting mechanism to summarize processing results
3. Add visualization of results (before/after preprocessing), eg checkerboard comparison

Source Data

Choose an MRI scan of a human brain as your input data. There are numerous free sources for neurological MRI datasets, including:

- ADNI
- OpenNeuro
- Human Connectome Project
- UK BioBank

For example, you could choose the anatomical scan from Subject 2 in the MEG-BIDS dataset.



Sample T1 Anatomical

MNI152 Atlas

The atlas registration should use the following unbiased average of the MNI152 database:

- ICBM 2009a Nonlinear Symmetric Atlas (NIFTI) (56MB)

This contains multiple modalities; ensure you select the one matching the input data.

Software

For this task, please write your solution in modern, idiomatic Python.

You are free to choose any libraries you wish to solve the problem. Common libraries include:

- numpy
- nibabel
- SimpleITK
- ...

The Mango viewer can be used to load and view data in many common medical imaging formats.

Technical Requirements

- Use Python 3.11+ and a modern, idiomatic style
- Implement as a command-line tool
- Include unit tests for key components
- Provide clear documentation and inline comments
- Include logging for tracking pipeline progress and errors
- Handle edge cases and invalid inputs gracefully

Evaluation

Criteria for evaluating the submission include:

- clarity of code
- efficiency
- error handling and robustness
- idiomatic solution
- separation of concerns (eg processing/file IO)
- maintainability
- PEP-8 conformance
- Software Engineering best practices
- unit testing & coverage
- comments & documentation

During the interview, you will be asked to present your work, including how the problem was solved, why certain decisions were made (eg library selection), what challenges were faced, how they were overcome, etc.

Note on AI Tools

If you use an AI assistant (eg ChatGPT or Claude) during this challenge, please document where and how they were utilized. Be prepared to discuss your experience including:

- How the AI helped shape your approach or implementation
- Limitations you encountered and how you addressed them
- What aspects required your expertise beyond what the AI provided
- How you validated and refined the AI-generated code
- How you might approach similar problems in the future with or without AI assistance

This information helps us understand your problem-solving process and technical judgment, not to penalize AI usage.

Submission

The submission can be made as either:

- a link to a Github (or equivalent) repository, or
- a self-contained ZIP file with the solution sent via email

The deliverables should include:

- Source code implementing the pipeline
- `README.md` with:
  - Installation/setup instructions
  - Usage examples
  - Description of the approach
  - Assumptions made
  - Potential improvements
- Unit tests
- Screenshot of sample output demonstrating the pipeline's functionality
- Brief documentation of the processing steps and implementation decisions

Do *not* include binaries, dependencies or large files (use `.gitignore`) but rather include links to input data in the `README` used to test the tool.

A solution should be possible in a few hours; don’t spend more than ~4 hours on the task.

Thanks and good luck!

🧠