1. Convert MRI scans from DICOM to NIfTI format

Task:

You will be given a set of MRI scans in DICOM format. Your task is to develop a Python script that efficiently converts these scans into the more research-friendly NIfTI format. This is a crucial step in facilitating further analysis and processing of the images.

Solution:

The solution is implemented in the convertor.py script. It utilises the dicom2nifti library. After running the script, a new .nii.gz file will be generated in the patient's case folder (PatientID_Modality).

2. Plot MRI scans and 3. Resampling and resizing

Task:

Once you have converted the MRI scans into the NIfTI format, you will need to create a Python script that can display the images in an intuitive and user-friendly manner. This will require you to not only read and display the images but also ensure that they are visualised in a way that is easy to understand them.

In medical imaging, it is common to encounter MRI scans with varying voxel sizes and resolutions, which can be attributed to differences in acquisition protocols, scanner hardware, or patient-specific factors. To ensure consistency across datasets and improve the computational efficiency of subsequent processing steps, it is essential to resample and resize these images to a common resolution and voxel size.

Solution:

The entire solution is presented in the resize_and_plot.py script. The script first opens the image using the Nibabel library and then generates basic figures for both technical and non-technical users. If the user requires a more detailed analysis of the image, I recommend using existing tools available online or for download.

Resampling is achieved using the resize method, allowing image transformation based on a target shape and scaling factor.

4. N4 correction of MRI scans

Task:

MRI scans can often suffer from intensity non-uniformities due to various factors, such as magnetic field inhomogeneities. To overcome this issue, you will need to implement an N4 bias field correction algorithm using Python. This will help normalize the intensities across the images, making it easier to compare and analyze the scans.

Solution:

The N4 bias field correction is implemented in the n4_bias_correction.py script. The main method is perform_correction, which first creates a mask using the Otsu thresholding technique and then applies it to the N4 bias field correction. To verify that the correction was successful, a sample is created and saved in the output/n4_bias_example directory.

5. Brain Extraction

Task:

Finally, you will develop a Python script to perform brain extraction on the MRI scans. This process involves isolating the brain tissue from the surrounding non-brain tissues, such as skull and scalp. Accurate brain extraction is essential for many downstream analysis tasks, such as segmentation and registration.

Solution:

The brain extraction is performed using the brain_extractor.py script. The extraction process involves converting the image into a binary mask using Otsu's thresholding, performing multiple erosions on the entire scan, selecting the second largest object in the binary image, and then performing a dilation. This resulting mask is then applied to the entire image. An example of the extraction is depicted in the output/extract_example directory.

Recommendations, Advantages, and Disadvantages of the Solutions:

It is recommended to have the code reviewed by another person to ensure its correctness and efficiency.

For the N4 bias field correction method, it is advisable to consult someone experienced with the algorithm to compare the results and ensure its effectiveness.

The brain extraction method is relatively simple and may not work well in some instances. It assumes a sufficient separation of the brain from the skull and assumes that the brain is the second largest object in the binary mask. For more accurate brain extraction, considering using tools like FSL or employing a neural network approach could be beneficial. The advantage of this solution is its simplicity and adjustable parameters.