ECE 4250 Final Report: Registration Based Segmentation of MRI Scans

Hamilton Lee (Cornell ECE ‘22)

Introduction

The goal of this project is to perform segmentation on MRI brain scans. The segmentation will be performed on 2D mid coronal slices of a MRI volume. The regions of interest are the Cerebral Cortex and Cerebral White Matter of both the left and right hemispheres of the brain. The training data provided are six MRI scans and their respective segmentations. Two validation data are provided along with their segmentations. The 9 other scans are the testing data which do not include their respective segmentations. The primary method of segmentation used is registration based segmentation. This involves finding an optimal geometric transformation (ϕ_i) that takes all the training images into the image we want to perform segmentation on with respect to some loss metric. Each optimal transform for each training image is performed onto the training segmentations to transfer the labels over to a candidate segmentation. Each training data will give 1 candidate segmentation. A function will decide pixel values for the final segmentation based on each candidate segmentation.

The metric used to determine whether a segmentation is successful will be the Jaccard overlap defined as:

$$\frac{|A \cap B|}{|A \cup B|}$$

Another metric one may be interested in is the Dice coefficient defined as:

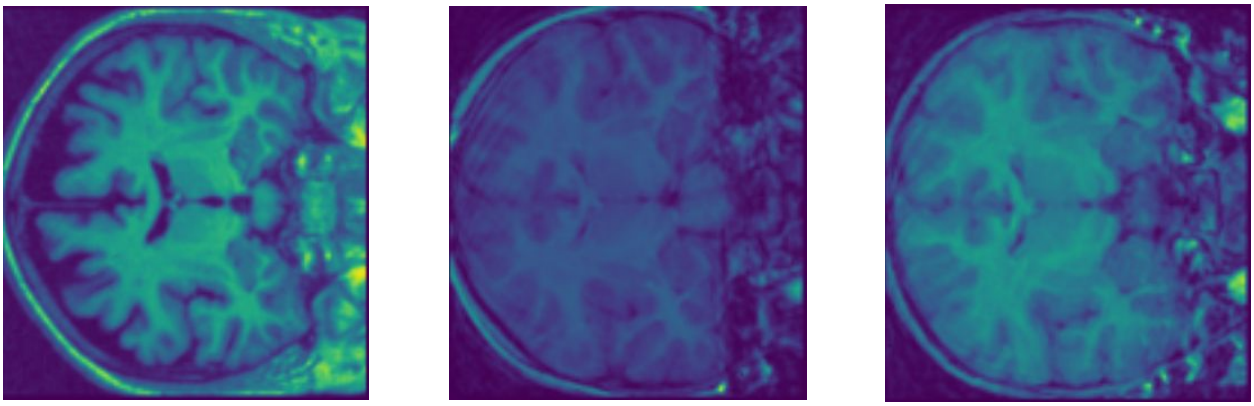
$$\frac{2 \cdot |A \cap B|}{|A| + |B|}$$

In both of these, A is the “ground truth” segmentation while B is segmentation generated by the segmentation algorithm. Both of these metrics quantify how similar the generated segmentation is to the ground truth where a value of 0 is not similar at all and a value close to 1 means a very good segmentation.

There are 4 main areas from which to improve the segmentation method from the process presented in Milestone 2 of the final project. These areas are discussed in individual sections below.

Pre-processing

Pre-processing is the way in which the MRI input images are processed before being put into the algorithm to find a resultant segmentation. This is perhaps more of an issue with multi-modal scans, but it is still important in this application. This is because, despite all input images being MRI scans (same mode), the distribution of pixel intensities among scans varies widely. For some optimization modules and loss metrics, this could lead to instability because some pixels may incur a large penalty when optimizing. Beyond this, MRI inhomogeneity presents an issue. MRI inhomogeneity is introduced into an MRI scan when magnetic fields are introduced in an inhomogeneous way. This can lead to spatial distortion, blurriness, intensity loss, and so on [1]. An example is presented in Fig. 2.



(a) (b) (c)

Figure 1.

In Figure 1, despite all images being of the same modality (MRI), intensity values vary widely among all 3 images. Also, notice in image (c), there are visible wave patterns over coronal sections. These correspond to MRI inhomogeneity.

To correct for differing intensity values across MRI scans, the histograms of the MRI training data can be altered. An exact histogram specification can be used to take the intensities of a scan into the desired range [2]. There are multiple approaches for doing so. One method is to perform histogram equalization on every MRI scan. This, however, assumes that the intensity distribution is equally distributed along all values. This is clearly not the case. Another alternative would be to perform histogram matching on each training data to the input image. This is a better alternative because it preserves, in some respects, the pixel distribution of the mid coronal slice. The implicit assumption is that the pixel distributions for all mid-coronal slices are relatively similar. A better alternative might be to perform histogram matching not to the input image, but one of the training images that has a desirable histogram to perform the rest of the segmentations with. For example, in Figure 1 (b), the intensity values are relatively low contrast and largely similar compared to (a) and (c). This may lead to some issues when performing calculation of the loss corresponding to a transformation when optimizing. Instead of depending on the quality of the input image to get a good target for histogram matching, setting one image a priori as the target for histogram matching will guarantee less variability due to histogram matching.

	Left Hemisphere	Right Hemisphere
Subject 7 Cerebral White Matter (No Matching)	0.619521	0.620530
Subject 7 Cerebral Cortex (No Matching)	0.668229	0.652807
Subject 7 Cerebral White Matter (With Matching)	0.622001	0.622889
Subject 7 Cerebral Cortex (With Matching)	0.670113	0.649591

Table 1.

The values in Table 1. illustrate the values of the jaccard overlap of Subject 7's segmentation with and without histogram matching with all other parameters held constant. The transformation used in this example is an affine transformation with a weighted majority segmentation for segmentation. The values for histogram matching are slightly improved meaning that histogram equalization may lead to better alignment when optimizing.

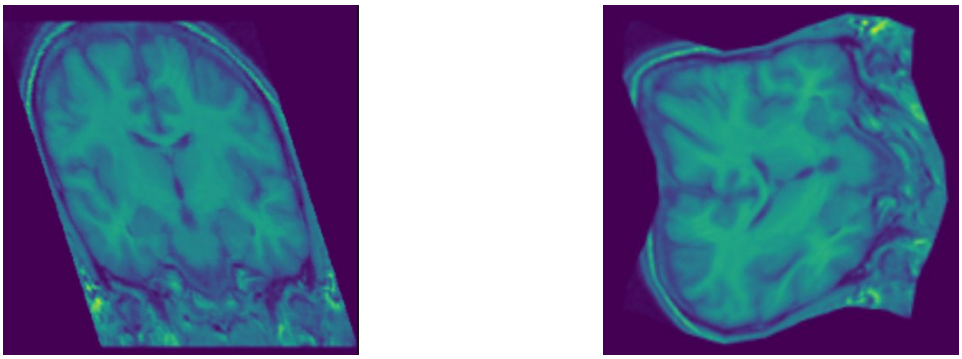
There are many advanced methods to correct for histogram inhomogeneity, but in this case, a simple bilateral mean filter will get rid of most of the noise. It will not get rid of larger structures like the wave patterns observed in Figure 1 (c). More advanced methods must be applied to correct for those artifacts.

Geometric Transformations

Choosing an optimal transformation is one of the most important steps in improving results for registration based segmentation. The standard transformation used in Milestone 2 implements a restricted form of the general affine transformation that doesn't allow for shear transformations. The most natural method to test would be to check whether or not generating a general affine transformation will improve results. An affine transformation has the form:

$$\begin{bmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Parameters a, b, c, and d determine the rotation, shearing, and scaling parameters for the transformation whereas t_x and t_y determine shift parameters. This matrix is applied to each of the coordinates of the pixel grid. Affine transformations are still limited in the sense that they map parallel lines to parallel lines globally and thus cannot account for local variations in the image. For a more local approach, a piecewise affine transformation can be used. For the piecewise affine transformation implemented in the project, the user sets input control points and output control points on a pixel coordinate grid. The computer will then generate a mesh of triangular regions with fixed vertices on the set of control points via Delaunay triangulation. This algorithm guarantees that no fixed control points are in the circumcenters of any drawn triangles [3]. Affine transformations are then found on each of the local triangular regions. The difference between the two is illustrated below:



(a) (b)

Figure 2.

Figure 2 (a) is the affine transform while Figure 2 (b). The piecewise affine is able to capture more local changes.

The main advantage to this transformation is that it is able to approximate local changes linearly. The main problem with this method is that it is computationally expensive and takes a long time to run. In the example below, only 16 control points were optimized. This poses an issue because in some cases, with the limited number of control points, the optimizer chose to distort the image in such a way that it moves the boundaries of the regions of interest causing problems with the jaccard overlap. If the number of control points are able to be increased, the transformation should prove more accurate. The Jaccard overlaps comparing affine transforms and piecewise affine transforms are illustrated below.

	Left Hemisphere	**Right Hemisphere**
Subject 15 Cerebral White Matter (Affine)	0.532491	0.609189
Subject 15 Cerebral Cortex (Affine)	0.502757	0.582573
Subject 15 Cerebral White Matter (Piecewise Affine)	0.556836	0.594281
Subject 15 Cerebral Cortex (Piecewise Affine)	0.521384	0.564982

Table 2.

With all other parameters kept constant, the changes between the two methods for 16 control points are not significant. This is because, as mentioned above, boundaries of the ROIs are over and under-warped because the grid is not dense enough to provide a good transformation.

It is often hard to approximate local changes in an image with linear transformations. Nonlinear-methods are better suited to capturing local changes. One of the simplest non-linear transformations that can be implemented is the polynomial transformation. Intensities are mapped based on the coordinate transformation:

$$x = \sum_{i=0}^{N} \sum_{j=0}^{N} (a_{ji} \cdot u^{(j-i)} \cdot v^{i})$$
$$y = \sum_{i=0}^{N} \sum_{j=0}^{N} (b_{ji} \cdot u^{(j-i)} \cdot v^{i})$$

This is for output coordinates (x, y) and input coordinates (u, v). N is the order of the polynomial of interest. a_{ij} and b_{ij} are the coefficients for these polynomials. The results of optimizing this transformation are displayed below for subject 15. The other parameters are kept the same as affine and piecewise affine transforms. An order 3 polynomial is used.

	Left Hemisphere	**Right Hemisphere**
Subject 15 Cerebral White Matter (Polynomial)	0.539699	0.602663
Subject 15 Cerebral Cortex (Polynomial)	0.504474	0.574913

Table 3.

The results for a polynomial transform are comparable to the affine and piecewise affine because an order 3 polynomial would not be able to capture significant local variations in the image. High order polynomials are necessary which requires large computational ability.

Freeform Deformations

Freeform deformations are the most accurate transformations that are tested. This is because freeform deformations most accurately capture local transformations around certain neighborhoods of pixels. Unfortunately, implementations of freeform deformations are quite memory and computationally expensive as many of them require monte carlo methods for estimating deformation grids. So they are not suitable for direct implementation in python. Luckily, Simple ITK has a C backed python library which makes performing these transformations possible in jupyter [5].

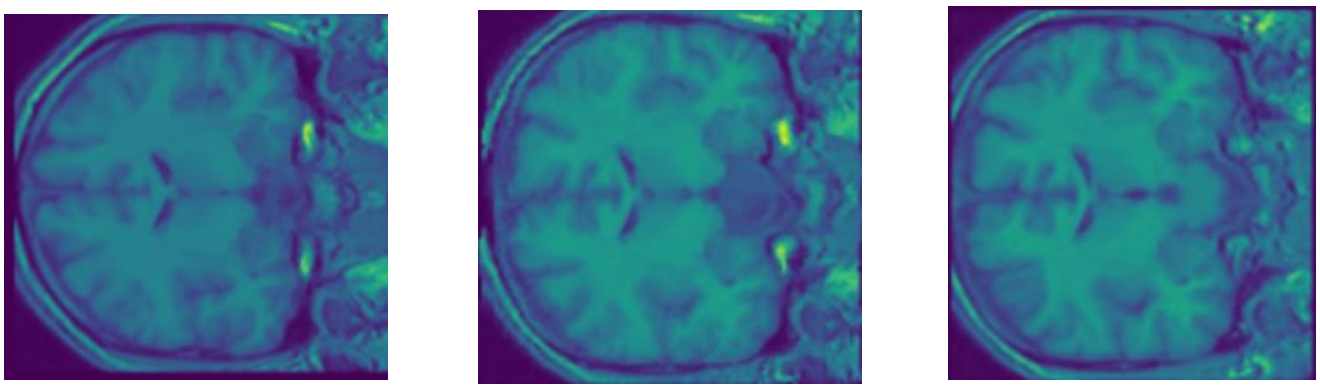
The first freeform deformation that is tested is a B-Spline based deformation. A spline is a parametric curve defined by user set control points. A B-Spline is a basis for all such curves of a given degree. A local spline deformation on a deformation field of control points can be expressed as a tensor product of the given form [6]:

$$T(\Phi) = \overline{x} + \mathbf{B} \otimes \Phi = \overline{x} + \sum_{i=1}^{3} \sum_{j=1}^{3} \sum_{k=1}^{3} B_i B_j B_k \Phi$$

Where $\bar{x}$ is the coordinate vector. The coordinate deformation grid Φ and $\bar{x}$ are indexed over r, s, t. $\mathbf{B}$ is the space spanned by the B-splines indexed over i, j, k. A deformation field is optimized to take the moving image to the fixed image using gradient descent. Unfortunately, using the Simple ITK library, the spline transformations tended to overfit the moving image to the fixed image leading to strange deformations that did not transfer well into the segmentations.

Another freeform deformation approach that is attempted is a diffeomorphic transformation using the demons algorithm. A diffeomorphism is an isomorphism between smooth differentiable manifolds. In other words, if there is a manifold, M_a which has an atlas U_a, then a diffeomorphism is a transformation ϕ that continuously maps each chart φ in U_a onto another smooth differentiable manifold M_b with atlas U_b. Because the differentiability is preserved, under the transformation, each chart φ, is still homeomorphic onto $\mathbb{R}^n$, so we can view the image coordinates in $\mathbb{R}^3$ as a manifold and perform these diffeomorphic warps on the manifold to deform the images [7]. It is useful to note that these transformations are, by definition, invertible. A non-diffeomorphic transformation is still useful in performing registration, but a smooth transformation might not be guaranteed because the transformation might be singular.

The optimal diffeomorphism is found through the demons algorithm. This algorithm treats the fixed image as a permeable grid. Force vectors drive the moving image into alignment with the fixed image. The deformation force fields are smoothed with gaussians to preserve differentiability [8]. An example is shown below:



(a) (b) (c)

Figure 3.

The image (a) is the moving image. Image (c) is the fixed image and image (b) is the warped version of image (a) into (c). It is evident this transformation does take into account the local differences between images as we can see the cerebral cortex in (b) far more closely follows (c)'s shape than (a). This would not be possible with the affine transform. With all other parameters kept constant (same as the piecewise affine transform), much better Jaccard overlaps are obtained for subject 15:

	Left Hemisphere	**Right Hemisphere**
Subject 15 Cerebral White Matter (Polynomial)	0.819123	0.843579
Subject 15 Cerebral Cortex (Polynomial)	0.799796	0.846921

Table 4.

This method can be improved through the use of image pyramids and multiscale image processing. The idea of multiscale image processing is that information about a specific image appears at different scales

and sizes [9]. The idea of an image pyramid is to decompose an image into multiple scales and then extract data at each given scale. The reason this is desirable is because if there is noise in the image, the multiple scale extraction will help reduce the effects of noise on the image. First registration is performed at the smallest scale. The resultant displacement field is scaled larger and becomes the initial displacement field for the next scale. This process recurs until the final full resolution image is reached. To clarify, at each smaller scale, the image is shrunk by some shrink factor and transformed using a filter. Classically, a gaussian filter is used to smooth the image at each level. This works to attenuate noise and to smoothly transition the displacement fields to convergence as new information of the brain topology slowly appears. An enhancement filter method is attempted where at each level the image is enhanced by different amounts. The idea of this method is that at different enhancement levels and scales, new information of the topology of the brain structure appears. Jaccard overlaps for each method are shown for subject 15. Evidently, the gaussian pyramid is more effective.

	Left Hemisphere	**Right Hemisphere**
Subject 15 Cerebral White Matter (Gaussian Pyramid)	0.854571	0.857051
Subject 15 Cerebral Cortex (Gaussian Pyramid)	0.820210	0.870606
Subject 15 Cerebral White Matter (Contrast Pyramid)	0.833887	0.849968
Subject 15 Cerebral Cortex (Contrast Pyramid)	0.807889	0.852332

Table 5.

Other Considerations and Full Pipelines

There are other considerations when considering generation of a segmentation pipeline. A major component in generating a good segmentation is how the final pixels are decided when presented with a set of candidate segmentations. The naive algorithm simply takes the majority voted intensity for each pixel in the output grid. This does not take into account the quality of the segmentations themselves. It does not make sense for a badly generated segmentation to be given the same weight of a segmentation that is extremely close to the ground truth image. There are several ways to improve this. First method that was tried was a naive weighted majority voting. The weights for each segmentation are found by taking the loss function values for that segmentation and fitting it to some curve, for example a sigmoid. This didn't work too well since if each pixel is to be viewed as a random variable with each candidate pixel as a measurement of that variable, there is no reason that a prior distribution would be known. Another method that gives better results is to not take majority votes just across one pixel, but use a 3x3 neighborhood instead. The idea is that ROIs are generally continuous therefore most pixels shouldn't be the boundary pixels especially for regions like the cerebral cortex and white matter. Using a 3x3 neighborhood will decrease the likelihood the pixel value chosen is an outlier. The most successful method tested is a correlational approach where a vote for each pixel across all candidate segmentations is weighted based on a correlation between the transformed training image that generated that segmentation and the fixed image. The correlation is taken around an 11x11 neighborhood of the pixel in question. The

idea is that the quality of each candidate pixel should be proportional to how similar the image that generated that pixel is to the input image. The 11x11 neighborhood is determined by experimentation. Performing correlation on the whole image isn't as good because it doesn't take into account local similarity. This is a major area for improvement as this process is quite slow. It could probably be sped up by using some sort of convolutional filter. The results of this method are shown below for a standard affine transformation. All other parameters are kept constant except for the voting scheme:

	Left Hemisphere	**Right Hemisphere**
Subject 15 Cerebral White Matter (3x3 Majority)	0.513041	0.591162
Subject 15 Cerebral Cortex (3x3 Majority)	0.477410	0.555026
Subject 15 Cerebral White Matter (Correlation)	0.532491	0.609189
Subject 15 Cerebral Cortex (Contrast Pyramid)	0.502757	0.582573

Table 6.

The correlational approach, despite being computationally expensive, is better across all transformations that were tested.

A minor component to this project would be choosing which loss function to optimize loss with respect to. Some methods like diffeomorphic demons registration in SimpleITK have built-in loss functions, but for the simpler geometric transformations like affine and piecewise affine, the choice of loss function can make a difference in the quality of the segmentation. Three options are considered all with very similar effects: sum of squared differences, sum of absolute differences, and correlation. All three yielded similar results, so a sum of absolute differences is used for all operations that needed a loss function. Another possible metric would be mutual information. This is a metric that determines how well the signal intensities can be predicted for one signal given another. Mutual information is more useful for multi-modal processing when intensities can vary greatly across images, but because the images are MRI's variation in intensities can be fixed by histogram matching, it would not give much improvement over the other mentioned metrics.

The full pipelines for segmentation algorithms that were tested are a combination of many of the methods mentioned above. All the most successful pipelines generally follow a set sequence. First, preprocessing is applied. This includes performing elementary denoising via bilateral mean filters. Histogram matching to training image 4 is performed. It has been determined that subject 4's histogram contains an intensity distribution that works well with the transformation functions tested. If histogram matching is performed to the input image instead, a risk is run that the input scan contains a bad pixel intensity distribution. Next, a global transformation is performed via an affine transformation. This sets the initial transformation for more local transformations. Next a nonlinear deformation transformation is performed. Multiscale registration often yields better results. The composition of the affine transform and nonlinear deformation is transferred onto the segmentations of the training data. A correlational weighted majority algorithm is used to determine the pixel intensities of the final image.

I. Works Cited

[1] Jackson EF. MR Acceptance Testing and Quality Control (pdf). American Association of Physicists in Medicine Annual Meeting, 2009.

[2] D. Coltuc, P. Bolon and J. -. Chassery, "Exact histogram specification," in IEEE Transactions on Image Processing, vol. 15, no. 5, pp. 1143-1152, May 2006, doi: 10.1109/TIP.2005.864170.

[3] “Delaunay Triangulation.” Wikipedia, Wikimedia Foundation, 17 Apr. 2020, en.wikipedia.org/wiki/Delaunay_triangulation.

[4] Adamchik, Victor & Jeffrey, David. (2003). Polynomial transformations of Tschirnhaus, Bring and Jarrard. SIGSAM Bulletin. 37. 10.1145/1093528.1093530.

[5] https://simpleitk.readthedocs.io/en/master/

[6] Balci SK, Golland P, Shenton M, Wells WM. 2007. Free-Form B-spline Deformation Model for Groupwise Registration. Med Image Comput Comput Assist Interv 10:23-30. doi: 10.1901/jaba.2007.10-23

[7] “Local Diffeomorphism.” Wikipedia, Wikimedia Foundation, 17 Apr. 2020, en.wikipedia.org/wiki/Local_diffeomorphism

[8] BMC Med Imaging. 2020 Apr 23;20(1):41. doi: 10.1186/s12880-020-00439-6.

[9] http://www.cs.toronto.edu/~jepson/csc320/notes/pyramids.pdf