AUTOMATIC SEGMENTATION OF BRAIN STRUCTURES FOR RADIOTHERAPY PLANNING

By

Pallavi V. Joshi

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements for

the degree of

MASTER OF SCIENCE

in

Electrical Engineering

May, 2005

Nashville, Tennessee

Approved:

Dr. Benoit M. Dawant

Dr. Mike Fitzpatrick

Dedicated to my Parents

ACKNOWLEDGMENTS

I would like to thank my academic advisor, Dr. Benoit Dawant, for his faith in me and for constantly inspiring me during my research work. His constant encouragement, timely critical evaluation and enthusiasm for my work resulted in the successful completion of my thesis. I am also thankful to Dr. Mike Fitzpatrick for his guidance during my graduate studies at Vanderbilt. I would also like to thank him for serving on my supervisory committee.

I wish to thank Dr. Ken Niermann for being so helpful and showing tremendous interest in my research work. His support assured the timely completion of my thesis. I also thank Dr. Dennis Duggan for providing access to the patient images and to the Radiation Oncology planning software.

I want to thank my parents for their faith in me and for all they did to support me for my higher studies. Their wisdom and love has always been the driving force behind my achievements. I also want to thank my fiancé, Sachin Deshmukh, for his motivation and for always being there for me. I also thank my friend Jayeeta and all my other friends for making me feel that I can achieve anything in this world. Special thanks to Pierre-Francois D'Haese and all my friends at the MIPLAB for their help and suggestions throughout my research work.

Finally, I thank God for being there with me, come what may.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Overview

In the past few decades unprecedented advances have been made in the field of medical imaging. The development and commercialization of new imaging technologies such as Computed Tomography (CT), Magnetic Resonance Imaging (MRI), digital subtraction angiography, Doppler ultrasound imaging, and various imaging techniques based on nuclear emission (PET, SPECT, etc.) have enabled robust and reliable image processing techniques for detection and diagnosis of diseases [1]. Physicians can very effectively use these technologies for both diagnosing and tracking the progress of illness and injuries. The various imaging technologies have emerged to assist the visualization of internal structures in the body so as to assist the physicians in surgical planning, simulation, diagnosis and various other applications. Image segmentation, an important component of Medical Image Processing, can be effectively used for the study and interpretation of medical images. It deals with partitioning the image into several regions useful for a particular task. For example, image segmentation can be used in the detection of organs such as the heart, liver, lungs or the different structures in the brain [1]. In this thesis, automatic image segmentation has been used for labeling of brain structures such as the brainstem, cerebellum, pituitary, eyes, optic nerves and the chiasm to assist in Radiotherapy planning. The next section describes what is Radiotherapy planning and why is it being used.

Radiation Therapy Planning

Radiation Therapy is an effective way to treat cancer in almost any part of the body. It involves careful and accurate use of high intensity radiation beams to destroy the cancerous cells. Radiotherapy can treat cancer because these high intensity radiations destroy the cancer cells' ability to reproduce and thus the patient can survive cancer. Modern technology uses highly effective three dimensional imaging technologies such as MRI and CT, computerized treatment planning and modern sophisticated machines to generate the high energy radiations. A radiation oncologist may use radiations produced by a machine outside the patients' body; this is called "External Beam Radiation Therapy" [4]. Radiations may also be given by putting radioactive sources inside the patients' body, which is called "Brachytherapy". External beam therapy is a method of delivering a beam of high energy X-rays to the location of the patient's tumor. This high energy beam is generated by linear accelerators or cobalt machines. The figure shown below shows a patient positioned on the treatment couch of a linear accelerator [4].

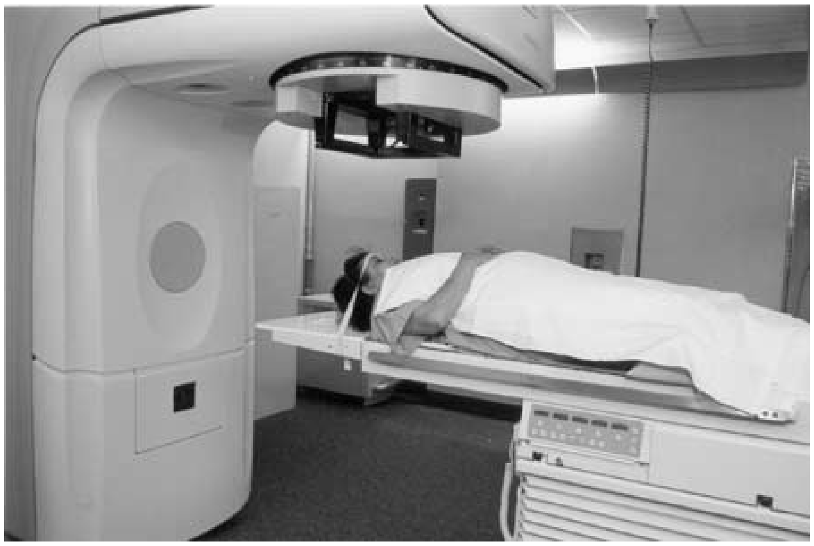


Figure 1.1: A patient positioned for radiation therapy planning [4].

The process of "External Beam Therapy" can be divided into three parts [4],

- Simulation: The radiation oncologist locates the tumor volume and the region to be treated on the CT images of the patient. The dosimetrist and the radiation oncologist then determine the optimum arrangement of the radiation beams needed to treat the patient.

- Treatment Planning: In this, the dosimetrist and the radiation oncologist calculate the radiation dose that will be delivered to the cancer cells. They also calculate the amount of radiation dose that will get delivered to the surrounding healthy tissues. This process of treatment planning is carried out on a special computer which will have a treatment planning software like the “Eclipse” [9] installed on it. This process basically makes use of "3-D conformal Radiotherapy". It includes careful development of complex plans which deliver highly "conformed" (focused) radiation to the cancer cells while sparing normal surrounding tissues. Intensity Modulated Radiation Therapy (IMRT) is a recently developed and highly effective method for destroying cancerous cells with minimal effect on the other body structures of the patient. The idea behind IMRT is to deliver a lethal radiation dose to the tumor using multiple beams coming from different angles and orientations. The beam shapes and orientation are chosen so as to maximize the irradiation of tumor while minimizing the irradiation of surrounding healthy tissues. Figure 1.2 depicts an IMRT plan carried out on a patient at the Radiation Oncology Department.

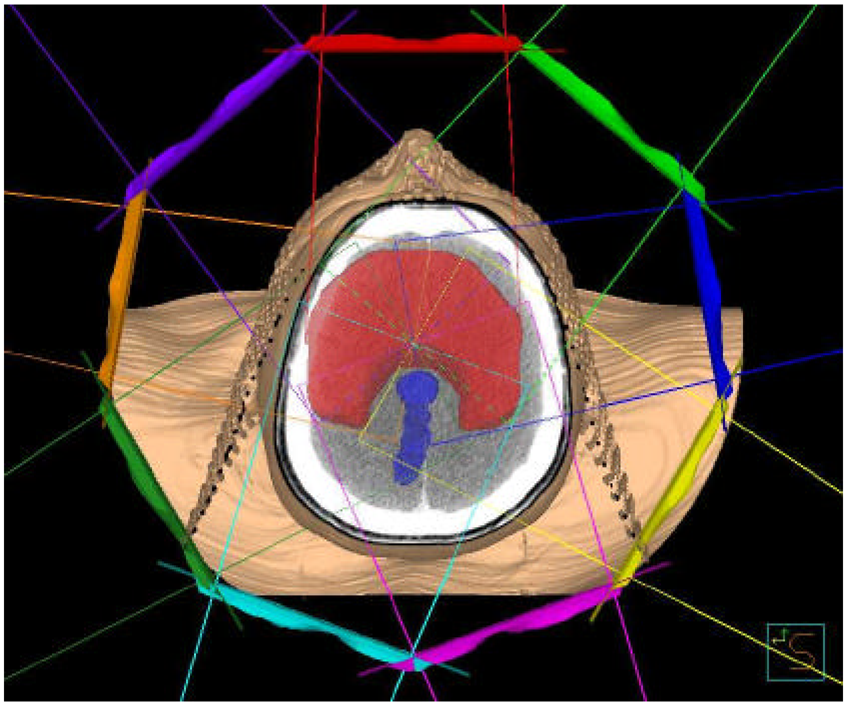


Figure 1.2: IMRT planning done on a patient.

[www.medical.philips.com/.../ ros/products/p3imrt/]

- Treatment Delivery: Once the simulation and planning is completed, the actual treatment begins.

The next section elaborates on the necessity of automatic delineation in IMRT.

Need of automatic segmentation in IMRT

IMRT relies on the careful and accurate delineation of structures to be irradiated (tumor) as well as of the structures (such as brainstem, cerebellum, eyes, etc.) to be spared. This delineation is currently done manually by segmenting the structures on a slice-by-slice basis. Manual delineation is extremely time consuming and is not reliable due to considerable inter and intra rater variability. It can also be challenging because radiation oncologists have historically not been well trained in cross sectional imaging. It is also difficult to accurately delineate complex 3D structures manually. Because of all these reasons and as the number of patients to be treated increases, these structures cannot always be segmented accurately, which may lead to suboptimal plans [15]. As a result of these human limitations, manual analysis of these images becomes impractical and computer-aided segmentation techniques are desired. Accurate auto-segmentation techniques will decrease the time taken by the physicians to contour the structures and thus make IMRT more useful. The next section briefly describes the various automatic segmentation techniques used in Medical Imaging.

Automatic Image Segmentation Techniques

Image segmentation plays an important role in medical image analysis. The objective of segmentation is to partition an image into regions. It essentially distinguishes the object of interest in an image from the rest of the image i.e. the image background. Image segmentation approaches can be classified depending on the technique used [2]. Segmentation techniques can be broadly classified as:

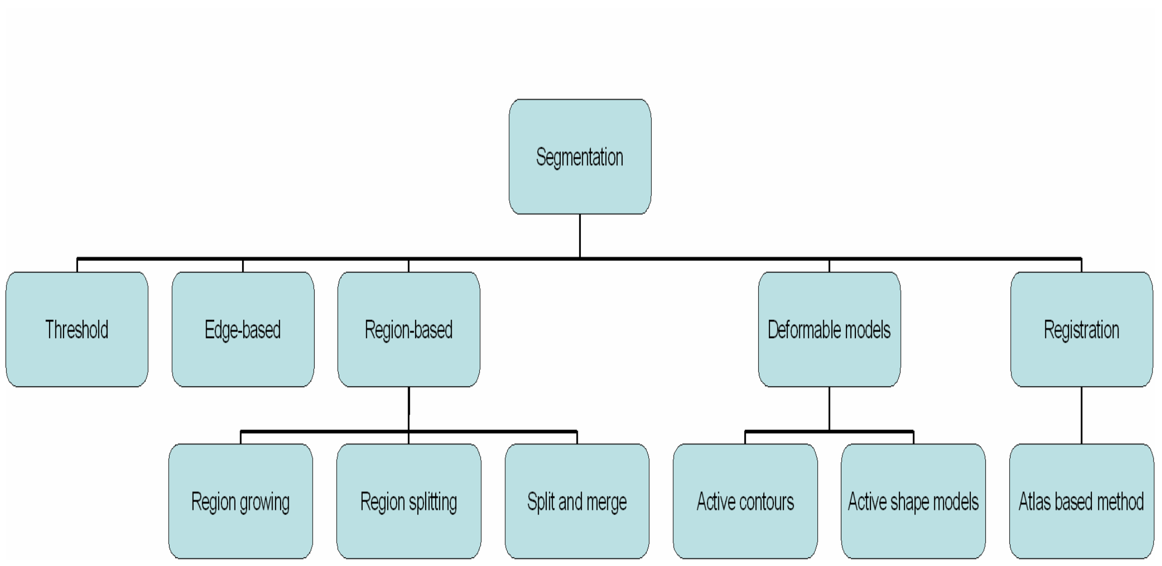


Figure 1.3: Various methods used in Medical Image Segmentation

The next few subsections will describe briefly each of the segmentation techniques mentioned in figure 1.3.

Thresholding

Gray-level thresholding is the simplest segmentation method. In this method, the object of interest is represented by an intensity level greater than a certain value and the rest of the image has different intensity levels. We then select such a gray level value and assign all the image voxels greater than or equal to this value as belonging to a particular object of interest. All the intensity levels less than the threshold are referred to as object background. If there is only one object of interest in the image, we use a single threshold. In the case of multiple objects, multiple thresholds are calculated and the image is thus divided into several useful regions of interest. If I(x,y) is the original image to be thresholded and T(x,y) is the thresholded image, then mathematically, thresholding can be represented by the following equation,

$$T(x,y) = 1, \text{ if } I(x,y) >= T$$

$$T(x,y) = 0, \text{ if } I(x,y) < T$$

There are various ways of choosing this threshold for segmentation. The two methods commonly used are, shape-based and optimal thresholding techniques [2].

Shape-based techniques make use of the shape of the gray-level histogram of the image. In the case of a bi-modal histogram, the valley point is chosen as the threshold value. These methods also sharpen the histogram peaks and the valleys between them so as to facilitate the thresholding operation. Optimal thresholding relies on the optimization of a merit function [2]. These methods are classified as non-parametric and parametric methods [2]. Non-parametric methods try to optimize some statistical measure such as the within-class variance, the between-class variance or the entropy. Otsu [34] proposed a technique in which the optimum threshold is selected as the one that maximizes s_B^2 / s_T^2, with s_T^2 the total variance and s_B^2 the between class variance. In parametric thresholding, the shape of the histogram of the image is assumed to be a mixture of Gaussian distributions. The parameters of this distribution are then estimated using maximum likelihood techniques [2], [17].

<u>Edge-based techniques</u>

In edge-based segmentation techniques, the processed image is described in terms of the edges between different regions. Edges can be detected using various edge detection operators. The most commonly used ones are the Laplacian operator, the Sobel operator, the Prewitt operator and the Canny edge detector. The Sobel operator is used to find the gradient magnitude at each point in an input grayscale image. The operator consists of a pair of 3x3 convolution masks given by,

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Here G_x and G_y are designed to respond maximally to the edges in the horizontal and vertical directions, respectively [5]. The absolute value of the gradient at any pixel in the image is then given by,

$$|G| = \sqrt{G_x^2 + G_y^2} \tag{1.1}$$

The direction of the gradient is given by,

$$q = \arctan(G_y / G_x) \tag{1.2}$$

The Prewitt operator can be specified by the following two masks,

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \text{and } G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

The Canny edge detector was designed to be an optimal edge detector. It works in a multi-stage process. First the image is smoothed with a Gaussian convolution to filter out any noise. The next step is to find the edge strength by taking the gradient of the image. For this purpose, any of the above operators can be used. Usually, the Sobel operator is preferred. Once the gradient is obtained, its direction can also be calculated as mentioned in equation (1.2). The next step is to relate the edge direction to a direction that can be tracked in the image. Then, non-maximal suppression is used to trace along the edge in the edge direction and set to 0 any pixel value that is not considered to be an edge. Finally, hysteresis thresholding is used to eliminate breaking up of edge contours. It uses two thresholds (high and low) T1 and T2. Any pixel having value greater than T1 is considered to be an edge pixel. All the pixels connected to this edge pixel and having a value greater than T2 are also considered as edge pixels [6], [7], [8].

The Laplacian operator is mathematically defined as,

$$L(x,y) = \frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2} \tag{1.3}$$

A discrete approximation to the Laplacian operator can be given as,

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

A Laplacian operator can detect edges because the position of an edge can be estimated by finding the zero crossings of the second derivative of the image [5]. Mathematical morphology can also be used for edge detection.

Region-based techniques

Region-based techniques segment the image I into N regions R_i based on some similarity criterion [2]. This process can be described as follows [2],

$$I = \bigcup_{i=1}^{N} R_i \tag{1.4}$$

$$R_i \bigcap R_j = 0, i \neq j \tag{1.5}$$

$$P(R_i) = TRUE \text{ for } i = 1,\ldots,N \tag{1.6}$$

$$P(R_i \bigcup R_j) = FALSE \text{ for } i \neq j \tag{1.7}$$

Here, $P(.)$ is a logical predicate. It specifies the properties that must be satisfied by all the pixels in a region. Region-based segmentation algorithms can be classified into the following categories:

- Region Growing: This method groups pixels together based on a similarity criterion. The algorithm starts with a few seed points. The pixels which satisfy the chosen similarity criterion are then added to these seed points. All those pixels satisfying the same similarity measure will form a single region of interest. The similarity criteria used can be the gray level, texture, moments or color information from the image. Connectivity information is also used by the algorithm to get meaningful results [5].

- Region Splitting: This method divides a region into several regions such that they satisfy the conditions stated in equations (1.4)-(1.7). The algorithm starts by considering the entire image as a single region R and selecting a predicate P. If $P(R) = FALSE$, the region is subdivided into quadrants. For each region R_i, if P is FALSE for any of the regions, that region is further subdivided into quadrants. This process is continued until there are no more regions to be formed [5].

- Split-and-merge: If only splitting was used, we may get adjacent regions with the same properties. To avoid this, a splitting algorithm is followed by the merging one. In this, the adjacent regions are merged whose combined pixels satisfy the predicate P. Such an algorithm is called the split and merge algorithm [5].

Deformable models

Classical methods of image segmentation such as edge detection and thresholding may not be easily applicable to noisy medical images. Since in most cases medical images are noisy, the method comprising of deformable models is often used for segmentation of medical images [3]. In this method, initial curves or surfaces are defined in a region and this curve or surface will evolve under the influence of internal curve or surface forces and external image forces. Two main implementations of deformable models can be found in the following papers: 'Snakes: Active Contours' by Kass et al. [10] and 'Active Shape Models' by Cootes et al [11]. The following sections describe them in brief.

Snake contours represent parametric curves or surfaces that evolve to fit the given object of interest by minimizing an energy functional. These contours are driven by the internal curve or surface forces and the external image forces. The internal forces keep the curve smooth and continuous while the external forces draw the contour towards the object edges. Mathematically, the deformable contour $\mathbf{X}(s) = (X(s), Y(s))$, $s \in [0,1]$, minimizes the following energy functional [3], [10],

$$E(\mathbf{X}) = S(\mathbf{X}) + P(\mathbf{X}) \tag{1.8}$$

$$S(\mathbf{X}) = \frac{1}{2}\int_0^1 a(s)\left|\frac{\partial \mathbf{X}}{\partial s}\right|^2 + b(s)\left|\frac{\partial^2 \mathbf{X}}{\partial s^2}\right|^2 ds \tag{1.9}$$

$$P(\mathbf{X}) = \int_0^1 P(\mathbf{X}(s))ds \tag{1.10}$$

The first term in equation (1.8) is the internal energy functional with parameters a and b used to control the stretching and bending of the curve. The second term in equation (1.8) is the potential energy function. The curve that minimizes the energy function must satisfy the following Euler-LaGrange equation:

$$\frac{\partial}{\partial s}\left(a\frac{\partial \mathbf{X}}{\partial s}\right) - \frac{\partial^2}{\partial s^2}\left(b\frac{\partial^2 \mathbf{X}}{\partial s^2}\right) - \nabla P(\mathbf{X}) = 0 \tag{1.11}$$

Various external forces can be used to drive these active contours such as: Gaussian Potential force, Pressure force, Distance Potential force, Gradient Vector flow, etc.

Active Shape Models (ASMs) make use of prior shape information from a training set to segment an object of interest from a new image. It involves the selection of landmark points for all the objects in the training set. These sets of landmarks are then aligned with each other by using rigid transformations (translations, rotations, scaling). A Point Distribution Model (PDM) is then constructed from these aligned landmark points. This involves finding the mean shape and the covariance matrix. The eigenvectors of this covariance matrix that correspond to the largest eigenvalues represent the direction of maximum shape variation for that object. Once the mean shape $\overline{M}$ is obtained it can be used to obtain any new shape by using the PDM. The equation can be stated as,

$$M = \overline{M} + Pb \tag{1.12}$$

where P is the matrix of eigenvectors and b is the vector of shape parameters that varies within specific limits set by the eigenvalues. This model is then fitted to the object of interest in the testing images by varying the pose parameters (rotation, translation and scaling) as well as the shape parameters [3], [11].

Registration

Registration is the process of aligning two images namely the source image and the target image of the same or different objects. It determines a spatial transformation which maps every point in the target image domain to its corresponding point in the source image domain. Registration is often a precursory step in image processing applications. This is because information obtained from the different imaging modalities gives different kind of information. An integration step is often required to combine the useful data from the different images. This calls for registration which will bring both the images in the same coordinate frame. We can therefore easily fuse the images obtained by various imaging modalities. There are two main types of registration methods, rigid registration and non-rigid registration. Rigid registration is typically used for accurate intra-patient registration problems. The distances and the straightness of all lines are preserved in this method [1]. It is composed of 3 translations and 3 rotations and thus has 6 degrees of freedom. Non-rigid registration is typically used for inter-patient registration problems and has numerous degrees of freedom. Atlas-based segmentation methods are segmentation techniques that make use of rigid and non-rigid registrations. This is our method of choice for segmenting brain structures to assist in radiotherapy planning. The method has been explained and discussed extensively in the next chapter.

Summary and Organization

We have used the 'Atlas-based method' for segmenting brain structures to assist in radiotherapy planning. This thesis tests the hypothesis that a fully automatic, atlas-based segmentation method can be used to segment most brain structures needed for radiotherapy planning [15].

This work is organized as follows. Chapter II explains the original atlas-based segmentation method in detail. Chapter III describes the various methods which have been studied and implemented in order to improve the segmentation results obtained by the original atlas-based segmentation technique. The entire process of segmentation is divided into several parts which have to be combined into a single program for easy and convenient execution of a pipeline. Chapter IV explains this integration process in detail. Finally we present the results obtained by the various methods used for segmenting the brain structures. The results obtained have been validated using several quantitative measures of validation. Chapter V explains the various validation methods used as well as the results obtained with them. Finally we conclude with 'Conclusion and Future work' in Chapter VI. This presents the various conclusions derived from the numerous experiments we conducted during this study. Chapter VI also includes recommendations for future investigation which may build upon this work.

CHAPTER II

ATLAS-BASED SEGMENTATION

This chapter explains the atlas-based segmentation technique used in this study to segment brain structures for radiotherapy planning. Atlas-based segmentation makes use of registration to achieve image segmentation. This method was originally used in [15] to achieve the segmentation of brain structures.

Atlas-based Segmentation Method Overview

Atlas-based segmentation essentially consists of the following three steps:

1. Rigid Registration
2. Non-rigid Registration
3. Model Projection

In this method, segmentation is achieved by registering the atlas or the reference image volume to the patient or the target image volume. Registration of the reference to the target image is achieved by applying a combination of rigid and non-rigid registration algorithms. The structures (in this case brain structures) to be segmented are first defined on the atlas volume. These atlas structures are then projected onto the patient volume by applying the rigid and non rigid transformations obtained above.

The patient dataset used in this study consists of a total of 20 patients. Out of these 20 patient volumes, an atlas volume was visually selected as being representative of the population, i.e., having an average brain size and shape, average ventricular size, and with only a very small tumor [15]. The atlas structures are drawn manually on every slice by a radiation oncologist. The structures used in this study are the brainstem, cerebellum, pituitary, eyes, lenses, optic nerves and the optic chiasm. Delineation was performed either on the sagittal, coronal, or the axial view depending on the structure, using a segmentation tool developed in-house. As it is difficult for a human operator to draw contours on consecutive slices that lead to smooth 3D shapes, they were post processed to obtain smooth 3D surface models on the atlas [15].

The atlas-to-subject registration was performed in two steps. The first step was the rigid registration of the atlas volume to the target patient volume. This helps in the initial alignment of the source and target volumes. The next step is the non-rigid registration between the rigidly registered atlas and the patient volume. Figure 2.1 gives a pictorial representation of the atlas-based registration method.

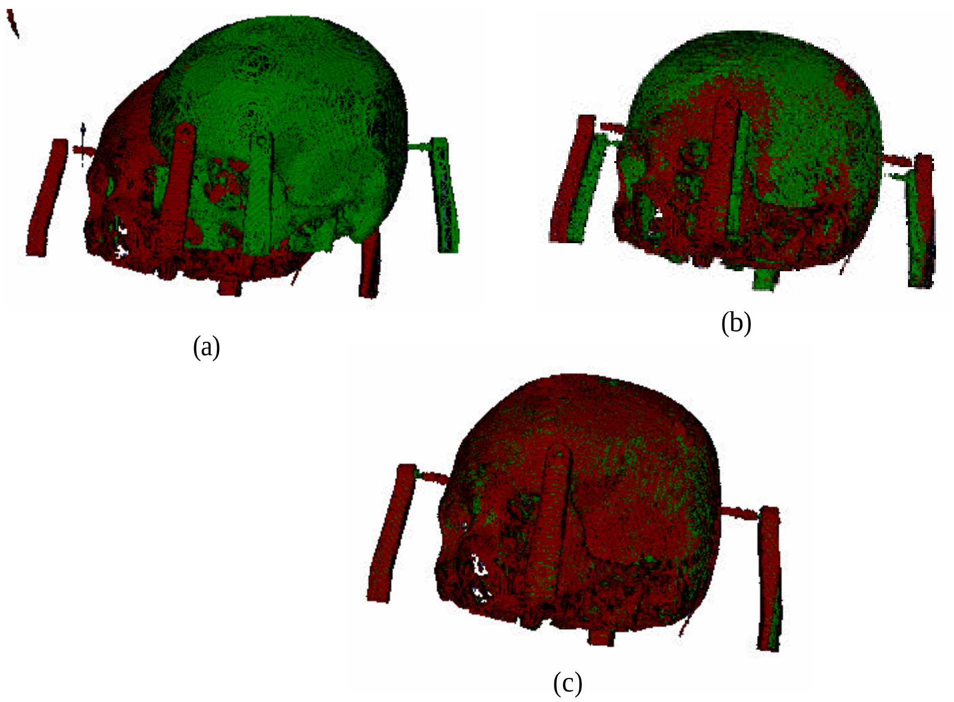


Figure 2.1: Atlas-based segmentation. The red skull is the target image and the green skull is the source or atlas image. (a) Source and target before registration. (b) Deformed source and target after rigid registration. (c) Deformed source and target after non-rigid registration.

The next section describes what registration is and how the rigid and non-rigid registration techniques have been implemented in this experiment.

What is Registration?

"Registration is the determination of a geometrical transformation that aligns points in one view of an object with the corresponding points in another view of that object or another object."[1]

The two images to be registered are called the source and the target images. The registration algorithm takes the source image and aligns it with the target image. It thus gives a mathematical mapping from the source image domain to the target image domain. In medicine, image registration is often an essential step in automated medical image analysis [1]. Registration has numerous applications in the fields of image guided surgery, distortion correction in fMRI images, atlas-based segmentation, etc. Image registration methods can be broadly classified into the following categories: point-based methods, surface-based methods, and voxel intensity based methods.

Voxel intensity based methods have become the method of choice in most of the clinical applications. This is because they use only the gray level information from the images and thus allow easy and robust implementations [13]. All the intensity-based registration methods are based on the optimization of some similarity measure. The problem of registration thus consists of transforming the source image $B(\mathbf{x})$ to "best" match the target image $A(\mathbf{x})$ under a chosen similarity criterion [12], [13]. In our experiment, the atlas volume is the input or the source image to be registered to the target or the patient volume so that they get aligned with each other. This alignment is a two step process: rigid and non-rigid registration.

Rigid or Affine Registration

"Rigid transformations are defined as geometrical transformations that preserve all distances, straightness of lines and the non-zero angles between straight lines." [1]. Rigid transformations consist of three rotations (q_x, q_y, q_z) and three translations (t_x, t_y, t_z) and thus have 6 degrees of freedom. We also consider anisotropic scaling in the three directions (sx, sy, sz) and 3D skewing (Gx, Gy, Gz) and hence get a total of 12 degrees of freedom. These transformations with 12 degrees of freedom are called affine transformations. Thus if T is an affine transformation, then,

$$\mathbf{x}' = T(\mathbf{x}) \tag{2.1}$$

$$\text{where, } T = \begin{bmatrix} S.G.R & t \\ 0 & 1 \end{bmatrix}$$

$$R = Rx.Ry.Rz \tag{2.2}$$

Here R is a 3x3 Rotation matrix, G is the 3x3 skewing matrix, S is the 3x3 scaling matrix and t is a 3x1 translation vector [1]. In the above equations, $\mathbf{x}'$ is the transformed point and $\mathbf{x}$ is the input point. If $\mathbf{y}$ is the corresponding point in the target image then the registration is good if $\mathbf{x}' \sim \mathbf{y}$; $(\mathbf{x}' - \mathbf{y})$ is called as the Target Registration Error (TRE) [1].

There are numerous rigid registration algorithms in medical imaging and they can be classified as being either frame-based, point landmark based, surface-based or voxel-based. We have used voxel-based method in this study. The rigid transformation is iteratively calculated by optimizing a similarity measure. The similarity measure we use is the Normalized Mutual Information (N.M.I) between corresponding voxel pairs from the overlapping region of the target image $A(\mathbf{x})$ and the transformed source image $TB(\mathbf{x})$ [13].

The N.M.I is assumed to be maximal when the images are geometrically aligned. Thus we have to iteratively find the transformation T which maximizes the N.M.I given by:

$$NI(A,TB) = \frac{H(A) + H(TB)}{H(A,TB)} \tag{2.3}$$

Here $H(A)$ and $H(TB)$ are the marginal entropies for the target and the transformed source images respectively, and $H(A,TB)$ is their joint entropy. These entropies can be evaluated directly from the discrete joint probability distribution of intensity values $p_{A,TB}(a,b)$:

$$H(A,TB) = -\sum_{a,b} p_{A,TB}(a,b)\log(p_{A,TB}(a,b)) \tag{2.4}$$

$$H(A) = -\sum_{a} p_{A}(a)\log(p_{A}(a)) \tag{2.5}$$

$$H(TB) = -\sum_{b} p_{TB}(b)\log(p_{TB}(b)) \text{, where} \tag{2.6}$$

$$p_{A}(a) = \sum_{b} p_{A,TB}(a,b) \text{, and} \tag{2.7}$$

$$p_{TB}(b) = \sum_{a} p_{A,TB}(a,b) \tag{2.8}$$

In our implementation, the images are initially positioned such that their centers coincide and the corresponding axes of both images are aligned and have the same direction. Powell's multidimensional direction set method is then used to maximize N.M.I, using Brent's one dimensional optimization algorithm for line minimizations.

The rigid registration algorithm explained above has been proposed by Maes F. [14]. All the equations above have been adopted from [1], [13]. Table 2.1 summarizes the rigid registration algorithm.

Table 2.1: Rigid registration algorithm

```
Load the images $A(\mathbf{x}), B(\mathbf{x}')$.
    • Create the images at the lowest user specified resolution
    • Set the optimization order, step size and initial values of rotations, translations,
      scaling and skewing parameters
      For I = 1 to number of resolutions
          • Set the Powell parameters for optimization
          • Begin Powell's optimization
          • Update the values of rotations, translations, scaling, skewing
          • Upsample $A(\mathbf{x}), B(\mathbf{x}')$
      End For
Output $B(\mathbf{x}')$ and $\mathbf{x}'$.
```

<u>Non-rigid Registration</u>

Non-rigid transformations have much more degrees of freedom and are used for inter-patient registration. Non-rigid registration consists of finding displacement vectors for every voxel in the image that can correctly align objects in one image with the corresponding objects in another image. In this case too, we use voxel intensity based registration method. The algorithm described here was originally developed by Rohde [12] and [13]. It is called the “Adaptive Bases Algorithm”. All the equations described below have been adopted from [12] and [13].

The registration task consists of iteratively deforming a source image $B(\mathbf{x})$ to "best" match it with a target image $A(\mathbf{x})$, optimizing the chosen cost function. The cost function used here is the Normalized Mutual Information (N.M.I) between the two images and registration is achieved by its maximization. Mathematically, it can be represented as,

$$T_{optimal} = {}_{\operatorname{argmax}(T)} F(TB(\mathbf{x}), A(\mathbf{x}), T)\,, \tag{2.9}$$

where F is the chosen similarity measure, T is the deformation field, $\mathbf{x}$ is the coordinate vector in three dimensional image space.

The task of deforming an image can be thought of as a re-arrangement of its sampling coordinates [13]. Mathematically,

$$TB(\mathbf{x}) = B(\mathbf{x} + \mathbf{v}(\mathbf{x})), \text{where} \tag{2.10}$$

$$\mathbf{x} = (x, y, z), \text{and} \tag{2.11}$$

$$\mathbf{v}(\mathbf{x}) = (v_x(\mathbf{x}), v_y(\mathbf{x}), v_z(\mathbf{x})) \tag{2.12}$$

where $\mathbf{v}$ is the deformation field in the three dimensions. It is constructed through a linear combination of compactly supported radial basis functions $\Phi(r)$:

$$\mathbf{v}(\mathbf{x}) = \sum_{i=1}^{N} \mathbf{c}_i \Phi(\|\mathbf{x} - \mathbf{x}_i\|) \tag{2.13}$$

$$\mathbf{c}_i = (c_i^x, c_i^y, c_i^z) \tag{2.14}$$

$$\Phi(\mathbf{x}) = R(\frac{\|\mathbf{x} - \mathbf{x}_i\|}{a_i}) \tag{2.15}$$

$$\Phi(r) = (1-r)_+^4 (3r^3 + 12r^2 + 16r + 4) \text{ for } r >= 0 \tag{2.16}$$

where $(1-r)_+^4 = \max(1-r, 0)$, N is the total number of basis functions composing the deformation field $\mathbf{v}$, $\mathbf{c}_i$ are the coefficients of the basis functions, $\mathbf{x}_i$ is the origin of the i^{th} basis function and a_i is the radius of the i^{th} basis function. Using the above formula (2.13) we can find the value of the function at any coordinate $\mathbf{x}$.

After modeling the deformation field by the linear combination of radial basis functions, we have to optimize their coefficients under the chosen similarity measure (N.M.I). We choose the Steepest Descent combined with a line search as the optimization procedure. We use a multiscale approach coupled with a multiresolution strategy to register the images. By resolution we mean the spatial resolution of the image while scale is related to the region of support and the number of basis functions. We find transformations that try to match large and coarse objects before proceeding to the more local detailed ones. Following this approach the final deformation field is computed as:

$$\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3 + \ldots + \mathbf{v}_M \tag{2.17}$$

where M is the number of levels chosen (a level refers to a particular combination of scale and resolution). This multiscale and multiresolution approach is usually successful at avoiding local minima during optimization.

Once the parameters are set through a parameter file, we begin the registration by first creating the source and target images at the user specified resolution at the current level. We then place basis functions (at the user specified scale) on a regular grid and we model the deformation field as:

$$d(\mathbf{x}) = \mathbf{x} + \sum_{k=1}^{m-1} \mathbf{v}_k(\mathbf{x}) + \sum_{i=1}^{N} \mathbf{c}_i \Phi(\frac{\mathbf{x}}{s} - \mathbf{k}_i)\text{, with } \mathbf{k}_i = \frac{\mathbf{x}_i}{s} \tag{2.18}$$

Here, $\mathbf{x}_i$ is the position of the basis function, s is their scale, and $\sum_{k=1}^{m-1} \mathbf{v}_k(\mathbf{x})$ is the sum of the deformation fields obtained up to level m-1.Then the gradient $\mathbf{G}$ of the cost function with respect to the coefficients of the basis functions is evaluated. The value of $\mathbf{G}$ is used to determine which regions in the images $A(\mathbf{x})$ and $B(\mathbf{x} + \sum_{k=1}^{m-1} \mathbf{v}_k(\mathbf{x}))$ are most likely to be misregistered at the current level. The idea behind this is: if the magnitude of the cost function with respect to the coefficient $\mathbf{c}_i$ is large then the cost function is not at a minimum with respect to $\mathbf{c}_i$. This means that the region where the corresponding basis function is located is misregistered. Once the regions of misregistration are identified, each disjoint region is optimized independently using a steepest descent algorithm combined with a line search.

This algorithm is called as the Adaptive Bases Registration Algorithm [12], [13] and is summarized in table 2.2.

Table 2.2: Non-rigid registration algorithm as proposed by Rohde [13]

```
Initialize the images $A(\mathbf{x}), B(\mathbf{x}')$ at the lowest user specified resolution and scale. Set $\mathbf{v}$ to zero.
        For I = 1 to number of resolutions
        For J = 1 to number of scales at the current resolution
                • Create a regular grid $\Theta$ at the current resolution and scale and compute regions of support for the basis functions
                • Identify regions of misregistration
                • Optimize each region independently from one another
        End For
        Upsample $A(\mathbf{x}), B(\mathbf{x}')$
End For
Output $B(\mathbf{x}')$ and $\mathbf{x}'$.
```

Model Projection

The final part in atlas-based segmentation is the projection of atlas structures on the patient volume. This is done by using the rigid and non-rigid deformation fields obtained from the above algorithms. For every structure defined on the atlas, the deformation field will give its equivalent position in the target or patient volume coordinate space. Thus, segmentation of those structures will be achieved on the patient volume. The target-defined masks are then converted to meshes for contour extraction. Figure 2.2 shows the atlas structures and the projected patient structures.

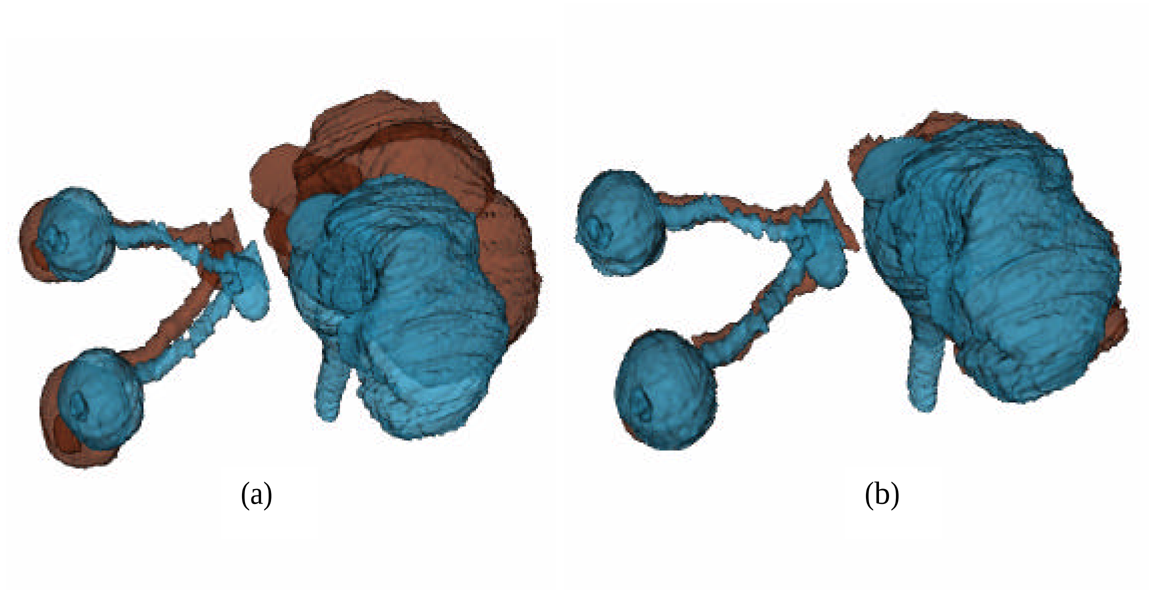


Figure 2.2: (a) Structures before registration and (b) Structures after rigid and non rigid registration. The blue structures are target defined structures and the red structures are atlas defined structures.

The next section describes the whole project pipeline.

Project Pipeline

The entire project consists of several programs which forms the project pipeline. This is shown in the following figure:

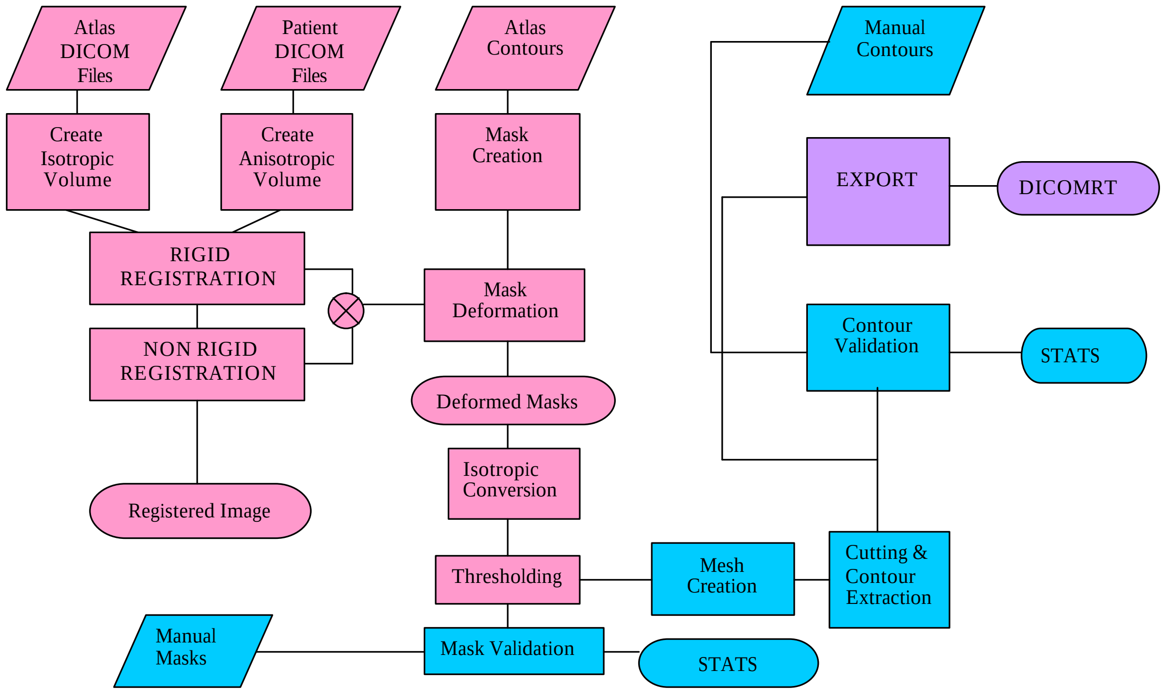


Figure 2.3: Project pipeline.

The whole pipeline is divided into three main parts: atlas-based segmentation, validation, and creating and exporting DICOM-RT files. In the figure above, they have been highlighted with three different colors: pink, blue and lavender in that order. A brief description of the different programs and their usage in the pipeline is given next.

Part I: Atlas-based Segmentation

3D Volume Creation: The DICOM files for the patients as well as the atlas are provided by the Radiation Oncology Department at the Vanderbilt Clinic. These DICOM files are then converted to raw image files using an in-house program written in IDL. Usually the images have the following dimensions: 256x256x124 in the X, Y and Z directions respectively with voxel dimensions of 0.9375x0.9375x1.3 mm in the three directions. Once we get the raw image file for the atlas, it can be resampled to obtain the isotropic volume: 256x256x256 with voxel dimensions of 1x1x1 mm^3. The atlas is resampled into an isotropic volume because the contours drawn by the physician on the atlas are on an isotropic volume. The patient volume is kept anisotropic since this gives a computational time saving of nearly 45 minutes.

Rigid Registration: The algorithm [14] described previously is used for rigid registration of the atlas to the patient volume. This provides initial alignment for the two volumes. The rigid registration program is executed by running a .exe file along with a parameter (script) file. This parameter file provides the paths for the source and target images and also all the registration parameters. The program has been written in the C language [33]. The atlas registered to the target is saved along with the rigid deformation field.

Non-Rigid Registration: The non-rigid algorithm described in the previous section has been implemented in-house using the C++ language. Here also there is an .exe file which uses a parameter file with all the non rigid registration parameters as well as the source and target image paths. The output of the rigid registration is the input to the non rigid program. The final deformed atlas image is saved along with the non rigid deformation field.

Atlas Mask Creation: A physician has drawn contours on the atlas volume to delineate 10 structures on the atlas. The structures delineated are: brainstem, cerebellum, chiasm, and pituitary, left eye, right eye, left lens, right lens, left optic nerve and right optic nerve. These contours are then converted to binary masks for each structure using a program written in-house in IDL.

Mask Deformation: The atlas masks are then deformed and projected on the patient CT volume. This is done by applying the rigid and non-rigid deformation fields obtained from the above two steps. To project the masks on the CT image volume, MR to CT rigid registration has to be performed for the target image. The program for mask deformation was also written in-house (using C language) and can be executed by running a .exe file which uses a parameter file. This parameter file specifies the location and names of the atlas structures and also of the deformation fields saved by the rigid and non-rigid programs.

Isotropic Conversion: The deformed masks obtained on the target MR volume are then resampled to get the isotropic volumes. This conversion is necessary because the atlas contours were initially drawn on the isotropic volume. Hence the patient masks also should be on isotropic volumes to facilitate their comparison. The code for this was written in-house using IDL.

Mask Thresholding: The deformed masks will not be binary like the atlas masks but will be gray level volumes. This is because when we deform the masks by applying the deformation fields we have to interpolate to get the intensity values at the transformed coordinates. Due to interpolation effects the volume is no longer a binary volume but has a range of gray values. These gray level volumes are then thresholded at half the intensity value of the masks (in our case it is 128). The program for this was written in C++.

Part II: Validation

Two methods have been used to compare manual and automatic contours. The first one involves comparing shape and size of the segmented masks, the second compares contours on a point by point basis. The manual contours were drawn by three physicians

(Dr. Anthony Cmelak, Dr. Edwin Donnelly and Dr. Ken Niermann) on randomly selected slices on each volume included in the study.

Mask Validation: To quantitatively compare masks obtained by the automatic segmentation and the masks obtained manually we use the following similarity measure:

$$S = \frac{2N(C_1 \cap C_2)}{N(C_1) + N(C_2)} \tag{2.19}$$

where $N(C_1)$ and $N(C_2)$ are the number of pixels included in the automatic and manual structure respectively. The code for mask comparison is written in C++ and the statistics obtained are tabulated in an Excel spreadsheet.

Contour Comparison: The masks are first converted into meshes using a program written in C++ (VTK). From these meshes the contour points are extracted. These contour points are then compared to the manual contour points. For this point by point comparison, we define an envelop as a band around the structure that encompasses all three manual contours plus one pixel on the inside and one pixel on the outside [15]. The number of automatic contour points that fall inside this envelop are then computed. The statistics are tabulated in an Excel spreadsheet. The code for contour comparison has been written in Matlab.

Part III: DICOM-RT file export

Creating and Exporting DICOM-RT files: For exporting the contours as DICOM-RT files, the contours have to be defined on the CT volume. This is because the radiation oncologists perform the radiotherapy planning on a CT volume. Thus the contour points are extracted from the CT meshes and these are used to create the DICOM-RT structure files. The code for this is written in C++ using VTK [29] and DCMTK [31]. The code was originally written by Pierre-Francois D'Haese [15]. These RT structure files are then exported to the Radiation Oncology Department at Vanderbilt University for treatment planning.

Summary

This chapter described the atlas-based segmentation technique used in this thesis for automatic segmentation of brain structures [15]. This method has been used as the basis for further work in this thesis. The segmentation results obtained from this method have been studied and we have tried to improve the results obtained by implementing three different techniques. These techniques have been explained comprehensively in the next chapter.

CHAPTER III

METHODS OF IMPROVING SEGMENTATION

Overview

This chapter gives a detailed explanation of three different methods that have been implemented to improve the accuracy of segmentations obtained by atlas-based method. The methods implemented are:

1. Fusion of CT and MRI volumes.
2. Mesh deformation.
3. Classifier combination using an Expectation Maximization (EM) algorithm.

Each of the following sections elaborates on one method that has been implemented in this study. In all cases, the atlas volume is isotropic (256x256x256 with a voxel spacing of 1x1x1 mm) and the target volume is anisotropic (256x256x124 with a voxel spacing of 0.9375x0.9375x1.3 mm). The target is kept anisotropic as it permits a saving of 30-40 minutes in the registration process.

Fusion of CT and MRI volumes

This section describes the CT-MR fusion method for improving segmentation results. The method is expected to give better results for some structures like the cerebellum and pituitary due to additional bone information available from the CT volume. The results have been validated in chapter V. The basic idea behind CT-MR fusion is to give edge definition to the structures in MRI by using high intensity information from CT, in an attempt to improve results for structures that have poor boundary definitions in MR imaging.

CT -MR Fusion Process

MRI imaging provides remarkably clear and detailed picture of the internal organs and tissues of the brain. It however does not show bones. The CT imaging technique displays the bones in white color (high intensity >= 1300) as compared to the rest of the structures in the brain (intensity ~ 1000). Thus a possible solution can be to create a volume which has internal organs and tissues from the MRI image and the bones from the CT image. This generates an image which has sharp intensity difference at the brain and skull boundary. The following figure shows a CT-MR fused image:

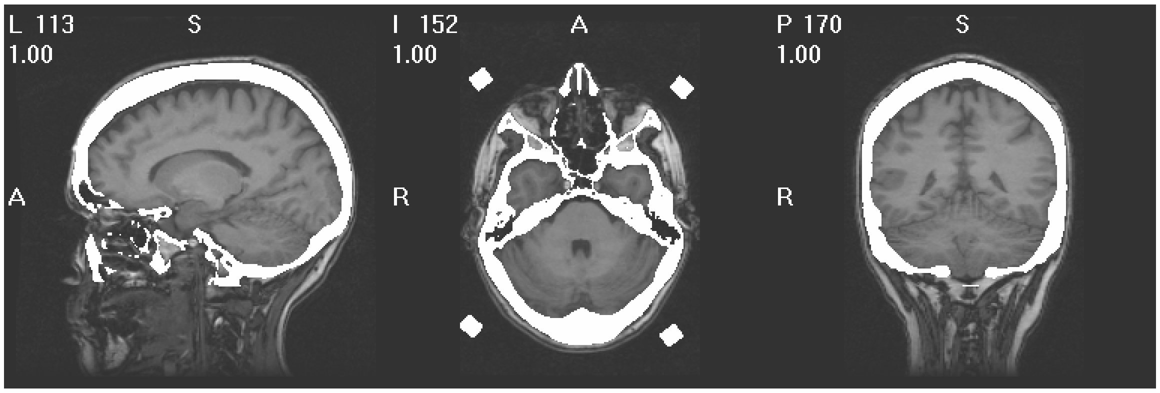


Figure 3.1: CT-MR fused image

To achieve this fusion, first the CT volume has been registered rigidly to the MRI volume of the same patient. This rigid registration aligns the skull bones in both the imaging modalities. Once both the volumes have the same coordinate system, the coordinates of CT bone voxels are determined by applying a threshold of >1300 to the CT-MR registered volume. These coordinates are then replaced with an intensity of 300 in the original MR volume. Thus an image with tissues from MRI and bones from CT is obtained. This fused image is used in the registration process. The same procedure is used to create a CT-MR fused atlas or reference image volume. This fused atlas is then registered rigidly and non-rigidly to the patient fused volume. The results obtained are then validated and compared with the original method.

Implementation and Algorithm

This method has been implemented as a part of the whole pipeline. The atlas-based segmentation process can be considered as one pipeline. The different programs implemented are part of this pipeline. The algorithm used can be summarized in table 3.1:

Table 3.1: CT-MR fusion algorithm

Give patient MRI and patient CT volumes as input to the program

- Register patient CT to MR using the rigid registration algorithm.
- Threshold the CT to MR registered volume at a value of 1300 to get the coordinates of bones in the image.
- Replace these coordinates with a value of 300 in the original MRI volume.
- Rigidly register the fused atlas to the fused patient image.
- Non-rigidly register the rigidly deformed fused atlas to the fused target image.

Mesh Deformation

This section describes the mesh deformation technique in atlas-based segmentation. We have tried to deform the meshes instead of the masks to investigate if this would bring an improvement in the accuracy of the atlas-based segmentation results. The results have been validated in chapter V.

Concepts

In the original study, segmentation was achieved by deforming the binary masks defined on the atlas image [15]. Essentially, the masks were deformed using the inverse transformation from the target to the atlas image. The forward transformation (from atlas to target) is not applied because it can cause incorrect results. If the forward transformation is applied to the binary masks, the corresponding transformed points in the target image domain will not necessarily fall on the target image grid. They will therefore be assigned to the nearest voxel on the grid and this voxel will be assigned the intensity of the corresponding voxel from the source image. Thus, two different voxels in the atlas image may get assigned to the same voxel in the target image because of rounding that takes place in the target image space. This will result in an ambiguous situation where the same voxel may get assigned to two different intensities from the atlas image and thus increasing the possibility of a faulty segmentation.

To eliminate this ambiguity and to increase the reliability of segmentation, the reverse transformation (from target to atlas) is used for deforming the binary masks on the atlas. In this situation, the inverse transform (from target to atlas) is applied to every voxel (or pixel in 2D) on the target image grid to get the corresponding transformed points in the source image domain. The intensities of these transformed points are then assigned to the corresponding voxels in the target image domain. But the transformed points will hardly ever fall on the source image grid. This situation calls for interpolation to obtain the intensity at a particular point in the source image domain. This interpolated intensity is then assigned to the corresponding voxel in the target image. Thus the intensity at every voxel in the target image is determined by "pulling" intensities from the source image. However, due to interpolation, the intensities assigned to the target image voxels will not be binary values but will be grayscale values. This is especially true for the voxels at the boundary of the structure. To minimize interpolation effects, the deformed gray-level masks are then thresholded at half the intensity value of the masks (128 in this case). But the thresholding of deformed masks cannot always be a reliable method of segmentation.

In the case of mesh deformations, we deform the meshes directly by applying the forward transformation. Meshes are essentially a set of points in 3D space, defining the topology and geometry of the binary image masks. This method deals only with a set of points in the image domain and not the entire image grid. The deformed mesh can be directly obtained by applying the forward transformation to these set of points in the source image domain. Since the mesh points do not fall on a regular grid, we need to interpolate the deformation field in order to find the displacement field at that point. In this study, we have used cubic interpolation for the deformation field. Once the corresponding transformed points are obtained in the target image domain, masks can be created. The detailed process of mesh creation and deformation is discussed in the following sections. The contours and masks obtained from this method are validated in the chapter V.

Mesh Creation

Meshes are a set of points generated from the image data using a marching cubes algorithm. The binary masks of the brain structures are given as input to the marching cubes algorithm. The threshold to be used for iso-surface generation must also be specified. The meshes are created using Visualization Tool Kit (VTK) [29]. VTK has many filters for iso-surface generation from the image data. The filter used in this experiment is the "vtkMarchingCubes" filter. The meshes created from this filter have their coordinates in the world coordinate system (i.e. in millimeters) with respect to the origin of the image data. The wireframe representation of the meshes generated is shown in the figure 3.3 below:

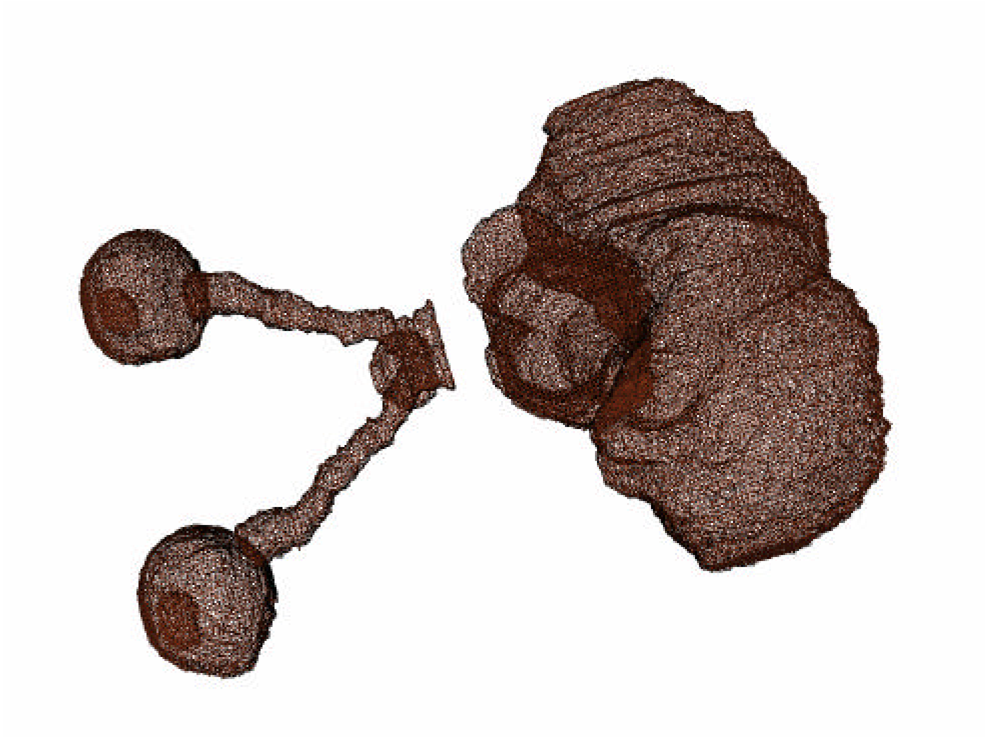


Figure 3.2: Wireframe representation of the meshes generated by the 'Marching Cubes' algorithm.

<u>Mesh Deformation</u>

Once the meshes have been obtained the forward transformation (atlas to target) is applied to the mesh points. First a rigid transformation (rotations, translations and scaling) is applied to the mesh. The rigid parameters are applied using VTK filters for transformation. In VTK, the class "vtkTransform" can be used to describe affine transformations in three dimensions, which are internally represented by a 4x4 homogeneous transformation matrix [29]. This transformation can be manipulated using the Translate(), Rotate() and Scaling() methods defined for the class. The rigid registration parameters are set using these methods and a transformation matrix in the world coordinate system is created. Once the transform is created, it has to be applied to the mesh points.

Before applying the rigid transformation, the orientation of the mesh needs to be changed. This is done because; prior to applying the rigid transformation, both the source and the target images have to be oriented in the similar fashion. It should however be noted that the orientation need not be changed if the transformation matrix is created in the image coordinate system. In both the cases, one should make sure that the VTK coordinate system is similar to the one used by the registration algorithm. The only difference between these two coordinate systems is that VTK considers the lower left hand corner of the image as the (0,0) voxel, whereas our registration algorithm considers the upper left hand corner as the (0,0) voxel. In short, the Y axis is flipped in VTK. To ensure that both the systems are equivalent, we read the images in VTK such that its Y axis gets flipped. The orientation used is depicted in the following figure:

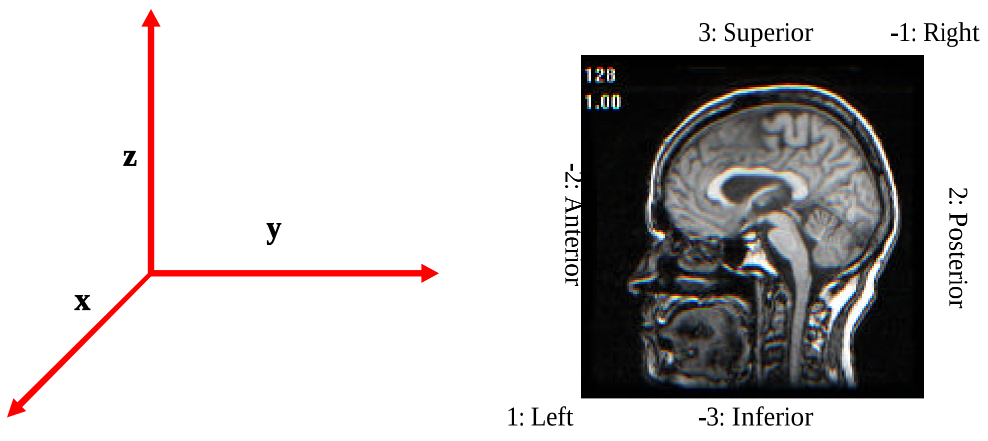


Figure 3.3: Orientation used in rigid registration algorithm.

As can be seen from the figure 3.3, the orientation is such that the positive X axis goes from right to left of the patient, the positive Y axis goes from anterior to posterior of the patient and the positive Z axis goes from inferior to superior of the patient. To transform the orientation of the sagittal MR image into the above orientation, the following transformation matrix is applied to the mesh,

$$\begin{bmatrix} 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This transformation matrix can be specified using the above figure as (2,-3,-1). It is achieved using another VTK filter called "vtkTransformFilter". In VTK terminology, the meshes are referred to as polygonal data. The vtkTransformFilter operates on this polygonal data and transforms its point coordinates in accordance to the vtkTransform matrix provided to it. An important point to be noted here is that all the rotations are around the center of the image and hence the origin of the image has to be changed to its center. The orientation of the mesh is later changed back to its original orientation by applying the inverse transform. This is necessary because the non-rigid registration program processes the images in their original orientation.

The next step is the application of the non-rigid deformation to the rigidly transformed mesh points. In the original method, the non-rigid deformation field gives the displacements in X, Y and Z directions for every voxel in the source image. This information is saved in the form of $(\mathbf{x}+\mathbf{dx}),(\mathbf{y}+\mathbf{dy})$ and $(\mathbf{z}+\mathbf{dz})$ for every voxel. To extract only the displacements, the voxel coordinates are subtracted from these numbers to get only $\mathbf{dx}, \mathbf{dy}$ and $\mathbf{dz}$. These displacements are in the units of voxels. Since the coordinates of the mesh points are in the units of millimeters (mm.), the displacements have to be first converted to units of mm. This is achieved by multiplying each displacement vector by its voxel spacing. The 3D deformation field is then imported as a vtkImageData. Once this is done, a non-rigid transformation is created using the "vtkGridTransform" class. It has a method "SetDisplacementGrid" that accepts a vector field. The vector field is a vtkImageData created above having three components (the x, y and the z displacement vectors) per voxel. This non-rigid deformation grid is then applied to the mesh using another VTK filter "vtkTransformPolyDataFilter". The output of this filter is the final deformed mesh in the target image domain.

Mesh Smoothing

Mesh smoothing has been applied to get better contours. The motive is to relax a mesh by making its points more evenly distributed [29]. To achieve this objective, a VTK filter "vtkWindowedSincPolyDataFilter" has been used. This filter adjusts point coordinates by using a windowed sinc function interpolation kernel. This function uses standard signal processing low pass filters (windowed sinc functions) to smooth the polygonal data. The transfer functions of these low pass filters are approximated by Chebyshev polynomials [29]. Smoothing is accomplished by applying the filter in an iterative process. This smoothing operation reduces high frequency information from the geometry of the mesh. After the meshes have been deformed they are smoothed by applying the above filter.

Contour Extraction

Contour points can be extracted from the meshes. To achieve this, the mesh polygonal data is cut using a "vtkPlane". To create a plane, the VTK class "vtkPlane" has been used and for the cutting operation a "vtkCutter" filter has been utilized. The vtkPlane is specified by a point and a normal. To cut a mesh with a plane, the plane has to pass through the cutting point of the mesh and should be perpendicular to the direction along which the data is being cut. The manual contours for each different structure have been drawn on a different orientation depending on the visibility of that structure. For example, the brainstem can be more easily contoured on the axial view, the cerebellum on the sagittal view and so on. Depending on the direction (sagittal, coronal or axial) in which the contours were originally drawn by the physicians, the plane is accordingly oriented perpendicular to that direction. The region where the cutter intersects the mesh will represent a set of points on a single slice. These points are extracted and imported as contour points for a structure. The contours obtained by mesh deformation are then validated using the techniques discussed in chapter V. The contour points are such that they represent the structure on an isotropic volume (256x256x256 with 1mmx1mmx1mm spacing). This can be easily accomplished by setting the origin of the plane as the center of an isotropic volume and then cutting the mesh at a distance of 1 mm. from the origin. This has been done because the manual contours were originally drawn on isotropic image volumes and is hence essential while validating the results. In the original implementation, this was achieved by reslicing the anisotropic volume using interpolation.

Mask Creation

The last step is the creation of masks from the meshes. This has been accomplished by using two VTK filters "vtkPolyDataToImageStencil" and "vtkImageStencil". These filters convert the polygonal mesh into an isotropic image data using a cookie-cutter operation. The images obtained are nothing but the binary masks of brain structures on the target image. All the results are validated in chapter V.

Implementation and Algorithm

The program for mesh deformation has been written using the Visualization ToolKit (VTK) in the C++ language [29]. The program takes the patient ID and the atlas ID as the inputs and finds the path for the rigid and non- rigid deformation fields for that patient. The path of the atlas mask is already hard coded inside the program. For every structure, a mesh is created and deformed using the above described method and the output binary mask is saved at the specified path. The contour points generated on every slice for every structure are also saved into a ".pts" file. The algorithm is summarized in table 3.2:

Table 3.2: Algorithm for mesh deformation

```
Give the patient ID and the atlas ID as input to the program
    For every structure of the atlas
        • Read the binary mask and create a mesh using vtkMarchingCubes
        • Change the origin to its center
        • Change the orientation of the mesh to (2,-3,-1)
        • Apply Rigid transformation using vtkTransformFilter
        • Change the orientation back to original for non-rigid deformation
        • Apply the non-rigid deformation field using vtkGridTransform
        • Smooth the mesh using vtkWindowedSincPolyDataFilter
        • Cut the mesh so as to get contour points on an isotropic volume
        • Create a binary mask from the mesh using vtkImageStencil
        • Save the mask and the contour points.
    End For loop
```

Classifier combination using Expectation Maximization (EM) algorithm

This section describes the EM algorithm for combining outputs from multiple classifiers. The algorithm used is called the 'Simultaneous Truth and Performance Level Estimation' (STAPLE) algorithm [18], [19]. The segmentations obtained are expected to be better than the single classifier segmentation. This hypothesis has been proved and results are validated in chapter V. This concept was originally used in [20] for combining classifiers in atlas-based segmentations.

Background

It has been shown in the pattern recognition community that the decisions made by a combination of classifier systems is more accurate than that made by an individual classifier [25]. The basic notion is that a group of classifiers tends to predict better than a single one. The goal of any pattern classification system is to achieve the best possible classification [26]. This objective led to the development of different classification methods for a single pattern recognition problem. It has been observed that different classifier systems produce different outputs as the sets of patterns classified/misclassified by different classifiers would not be exactly similar. This observation has led to the notion of "classifier combination". Different classifiers can be developed by having different representations for the input to the classifier (different features sets) and/or having different classification principles for each individual classifier [24], [26]. The outputs of these different classifiers are then combined by using different combination strategies. The various classifier combination methods described in this section have been adapted from a paper by J. Kittler [26]. The notations used are also similar to those used in [26].

Suppose that there are R different pattern classification systems. In our case, the patterns have been represented by the same feature vector but the classification schemes are different for each classifier. Let $\mathbf{x}_1,.....,\mathbf{x}_R$ represent the inputs of different classifiers. Here $\mathbf{x}$ denotes the feature vector which is same for all the classifiers and the subscript 1… R denotes the classifier number. Thus, $P(w_1 \,|\, \mathbf{x}_j)$ represents the posteriori probability of classifier j for class w_1. The various combination strategies have been summarized below.

Suppose that the voxel Z has to be assigned to any one of the m different classes.

- Product rule: It can be stated as:

$$assign\ \ Z \to w_j\ \ if$$

$$P^{-(R-1)}(w_j)\prod_{i=1}^{R} P(w_j \,|\, \mathbf{x}_i) = \max_{k=1}^{m} P^{-(R-1)}(w_k)\prod_{i=1}^{R} P(w_k \,|\, \mathbf{x}_i) \qquad (3.1)$$

- Sum Rule: The sum rule for fusion can be stated as:

$$assign\ \ Z \to w_j\ \ if$$

$$(1-R)P(w_j) + \sum_{i=1}^{R} P(w_j \,|\, \mathbf{x}_i) = \max_{k=1}^{m}\left[(1-R)P(w_k) + \sum_{i=1}^{R} P(w_k \,|\, \mathbf{x}_i)\right] \qquad (3.2)$$

- Max Rule: This rule is derived from the sum rule and can be stated as:

$$assign\ \ Z \to w_j\ \ if$$

$$(1-R)P(w_j) + R\max_{i=1}^{R} P(w_j \,|\, \mathbf{x}_i) = \max_{k=1}^{m}\left[(1-R)P(w_k) + R\max_{i=1}^{R} P(w_k \,|\, \mathbf{x}_i)\right] \qquad (3.3)$$

- Min Rule: This rule is derived from the product rule and can be stated as:

$$assign\ \ Z \to w_j\ \ if$$

$$P^{-(R-1)}(w_j)\min_{i=1}^{R} P(w_j \,|\, \mathbf{x}_i) = \max_{k=1}^{m} P^{-(R-1)}(w_k)\min_{i=1}^{R} P(w_k \,|\, \mathbf{x}_i) \qquad (3.4)$$

- Median Rule: This rule can be stated as:

$$assign \;\; Z \to w_j \;\; if$$

$$\operatorname*{med}_{i=1}^{R} P(w_j \mid \mathbf{x}_i) = \max_{k=1}^{m} \operatorname*{med}_{i=1}^{R} P(w_k \mid \mathbf{x}_i) \tag{3.5}$$

- Majority Vote Rule: This rule can be derived from the sum rule and under the assumption of equal priors. The hardening of the a posteriori probabilities $P(w_k \mid \mathbf{x}_i)$ produces binary valued functions Δ_{ki} as:

$$\Delta_{ki} = 1 \text{ if } P(w_k \mid \mathbf{x}_i) = \max_{j=1}^{m} P(w_j \mid \mathbf{x}_i)$$

$$\Delta_{ki} = 0 \text{ otherwise}$$

The majority vote rule can then be stated as:

$$assign \;\; Z \to w_j \;\; if$$

$$\sum_{i=1}^{R} \Delta_{ji} = \max_{k=1}^{m} \sum_{i=1}^{R} \Delta_{ki} \tag{3.6}$$

These conventional methods for combining individual classifiers weigh each classifier equally [25]. In this study, we combine the individual classifiers by first estimating their performance parameters and then combining these individual classifiers by weighing them according to their estimated performances. The method implemented is based on Warfield's algorithm [18], [19] of estimating the ground truth and performance parameters using the EM algorithm.

Multiple classifiers in Atlas-based segmentation

Different classifiers can be generated by using a different classification principle for each of the individual classifiers. In atlas-based segmentation, multiple classifiers can be obtained by applying different registration methods to the same atlas or the same registration method to different atlases [21], [23], [24], [25]. In this study, the same registration method has been applied to different atlases. This has been done because different segmentations will result from different atlases depending on the quality of registration [25]. The feature set representing input to the classifiers is kept the same. In this case, the input to the classifier is an image which is represented by its voxel intensities. Four different atlases have been used. Each different atlas along with the registration method will thus represent a unique classifier for the voxels of the input image [25].

Classifier Combination Method

The method used for combining the individual classifier outputs is as proposed by Warfield et al. [18], [19]. This method uses an EM algorithm for 'Simultaneous Truth and Performance Level Estimation' (STAPLE). A detailed study of the different combination methods has been conducted by Torsten Rohlfing [22], [25]. According to this study, the STAPLE algorithm gives better results than any of the other combination methods discussed above [19], [21], [25]. The STAPLE algorithm takes a group of segmentations as input and computes a probabilistic estimate of the true segmentation and of the performance parameters for each classifier. The source of each segmentation is an automated image segmentation obtained from one of the four atlases. This algorithm has been developed as an instance of the EM algorithm which will be discussed in the next section [18].

Expectation Maximization (EM) Algorithm

This section gives a brief overview of Maximum Likelihood (ML) estimation and of the EM algorithm. The notations and the equations used below have been adapted from [17] and [32]. ML estimation is used to estimate the parameters of a distribution based upon the observed data which is drawn from that distribution. Suppose that there are n samples drawn independently from the probability distribution $p(\mathbf{x} \mid q)$. Let q denote a set of parameters characterizing the distribution $p(\mathbf{x} \mid q)$. The task is to estimate q from the observed samples. The joint probability of all the samples is,

$$p(\mathbf{x} \mid q) = \prod_{k=1}^{n} p(\mathbf{x}_k \mid q) \tag{3.7}$$

The equation (3.7) is called as the likelihood function denoted by $l(q)$. Thus the maximum likelihood estimate of q is the value $\hat{q}$ that maximizes the log of likelihood $p(\mathbf{x} \mid q)$. This is achieved by taking the first derivative of the log likelihood function with respect to q and equating it to zero. Mathematically this can be written as:

$$l(q) = \ln p(\mathbf{x} \mid q) \tag{3.8}$$

$$\hat{q} = \underset{q}{\operatorname{argmax}}\, l(q) \tag{3.9}$$

$$\nabla_q l = 0 \tag{3.10}$$

Maximum likelihood techniques can be used when the data is uncorrupted, i.e. when the complete data is available. But to achieve the learning of parameters when the entire data is not available, Expectation-Maximization (EM) algorithm has to be used. The notations used below have been adapted from [32]. Suppose that $\mathbf{x}$ represents the complete data and $\mathbf{y}$ represent the observed data (with missing features) [32]. The complete data $\mathbf{x}$ is not directly observable and thus the likelihood function $f(\mathbf{x} \mid q)$ cannot be directly calculated and thus q cannot be estimated using the ML algorithm. In such situations, the parameter q can be estimated iteratively by maximizing the expectation of log $f(\mathbf{x} \mid q)$ given the data

$\mathbf{y}$ and the current estimate of q. The Expectation (E)-step of the EM algorithm can be stated as,

$$Q(q \mid q^k) = E[\log f(\mathbf{x} \mid q) \mid \mathbf{y}, q^k] \tag{3.11}$$

Here, q^k is the estimate of q at iteration number k. Thus, this equation gives the conditional expectation of the likelihood of the complete data given the observed data and the previous estimate q^k. Once the likelihood of the complete data is estimated the ML estimate of q can be obtained by maximizing Q.

$$q^{k+1} = \underset{q}{\operatorname{argmax}}\, Q(q \mid q^k) \tag{3.12}$$

Equation (3.12) is called the Maximization (M)-step of the EM algorithm. The E-step and the M-step are iterated until convergence is reached.

Application of EM Algorithm

The EM algorithm described above has been used for combining multiple classifier outputs in this study. This method was originally proposed in [18], [19], [23]. The notations used below have been adapted from [18]. This section describes the usage of the EM algorithm in the case of atlas-based segmentations. In our case, there are a total of four expert segmentations $j = 1....4$. Each segmentation has its own dataset D_j comprising of image voxels. Each segmentation is characterized by its sensitivity and the specificity denoted by p_j, q_j. The sensitivity and specificity parameters are defined as the "true positive fraction" and the "true negative fraction" respectively [19]. Each sample or voxel i in D_{ij} has two features: one is its own intensity value (in our case it is 0 or 1 as we deal with binary images) and the other is the binary ground truth value for that voxel. In this case therefore the binary ground truth value for each voxel is unknown and is considered as the missing feature. The complete data therefore consists of the segmentation decisions at each voxel D_{ij}, which are known, and the true segmentation T_i which is not known [19]. Since the complete data is not available, the complete data log likelihood cannot be determined but has to be estimated [19]. Thus, we can say that our data in not complete and it consists of some missing features. As discussed earlier, if the complete data is not available, EM algorithm can be used to determine the unknown data (ground truth in this case) and also for the estimation of the performance parameters. The EM algorithm first estimates the conditional expectation of the complete data log likelihood function, given the known data and an initial guess of the performance parameters. This can be achieved by using equation (3.11). Once the complete data log likelihood is estimated, the performance parameters can be estimated by applying equation (3.12). This sequence of likelihood function and parameter estimation is repeated until convergence is reached [19]. The details of the algorithm and notations used are described in the next section.

Concepts and Notation

The notations and equations given in this section are similar to those used by Warfield in his papers [18], [19]. We have also closely followed his derivations throughout this section. Let D represent the segmentation decisions made by all the classifiers. This is the observed data. Let T represent the unknown binary true segmentation. Then the complete data will be (D,T). Let p and q represent the sensitivity and specificity of the automatic segmentations. These are the parameters to be estimated for each classifier. The probability density function of the complete data will then be as given in the following equation,

$$f(D,T \mid p,q) \tag{3.13}$$

The performance level parameters to be estimated should be such that they maximize the complete data log likelihood function given by:

$$(\hat{p},\hat{q}) = \arg\max_{p,q} \ln f(D,T \mid p,q) \tag{3.14}$$

The sensitivity and specificity parameters are given by:

$$p_j = P(D_{ij} = 1 \mid T_i = 1) \tag{3.15}$$

$$q_j = P(D_{ij} = 0 \mid T_i = 0) \tag{3.16}$$

Let $q_j = (p_j, q_j)^T, \forall j \in 1...R$ be the unknown parameters describing the performance of segmentation j. In this case, j goes from 1 to 4. Since the complete data log likelihood function is not available, the conditional expectation of the complete data log likelihood function is estimated. This is referred to as the E-step of the EM algorithm. The E-step requires the calculation of the following function:

$$Q(q \mid q^{k-1}) = E\left[\ln f(D,T \mid q) \mid D, q^{k-1}\right] \tag{3.17}$$

Here $k-1$ refers to the value of the previous estimate of the parameters and k is the current iteration number. This function essentially gives the conditional expectation of the complete data log likelihood function. Since T is unknown it is considered here as a binary random variable. This random variable is binary because it can have two values either zero, representing the presence or one, representing the absence of the structure at the concerned voxel. The Expectation E of a random variable $\mathbf{x}$ can be defined as,

$$E(\mathbf{x}) = \int \mathbf{x} f(\mathbf{x}) d\mathbf{x} \tag{3.18}$$

To find E of any function of this random variable $g(\mathbf{x})$ we use,

$$E\{g(\mathbf{x})\} = \int g(\mathbf{x}) f(\mathbf{x}) d\mathbf{x} \tag{3.19}$$

In the case of discrete random variables, the integration gets replaced by summation. Thus, from equation (3.19), equation (3.17) can be written as,

$$Q(q \mid q^{k-1}) = \sum_T \ln f(D,T \mid q) f(T \mid D, q^{k-1}) \tag{3.20}$$

We have from the definition of joint probability,

$$f(D,T \mid q) = \frac{f(D,T,q)}{f(q)} \tag{3.21}$$

Substituting equation (3.17) by the joint probability equation in (3.21) we get,

$$Q(q \mid q^{k-1}) = E\left[\ln \frac{f(D,T,q)}{f(q)} \mid D, q^{k-1}\right] \tag{3.22}$$

On expressing q as (p,q) and multiply dividing by $f(T, p, q)$,

$$Q(q \mid q^{k-1}) = E\left[\ln \frac{f(D,T,p,q)\, f(T,p,q)}{f(T,p,q)\, f(p,q)} \mid D, p^{k-1}, q^{k-1}\right] \tag{3.23}$$

By the definition of joint probability we have the following equation,

$$f(D \mid T, p, q) = \frac{f(D,T,p,q)}{f(T,p,q)} \tag{3.24}$$

Assuming T to be independent of the performance level parameters we get,

$$f(T,p,q) = f(T)\, f(p,q) \tag{3.25}$$

Thus the final equation can be written as,

$$Q((p,q) \mid (p^{k-1}, q^{k-1})) = E\left[\ln(f(D \mid T,p,q)\, f(T)) \mid D, p^{k-1}, q^{k-1}\right] \tag{3.26}$$

The next step is determining the parameters that maximize this function which is referred to as the M-step. Mathematically this can be represented as:

$$(p^k, q^k) = \underset{p,q}{\operatorname{argmax}} E\left[\ln(f(D\,|\,T,p,q)\,f(T))\,|\,D, p^{k-1}, q^{k-1}\right] \tag{3.27}$$

Here (p^k, q^k) is the estimate of the segmentation performance parameters at iteration k. The E-step and M-step estimation sequence is repeated until convergence is reached. The following sections explain in detail the E-step and the M-step used for this algorithm [19].

E-step

In this section, the equation for estimating the unknown ground truth is derived.

By Bayes formula we can write,

$$f(T\,|\,D, q^{k-1}) = \frac{f(D\,|\,T, q^{k-1})\,f(T)}{\sum_T f(D\,|\,T, q^{k-1})\,f(T)} \tag{3.28}$$

$$f(T\,|\,D, q^{k-1}) = \frac{\prod_i \left[\prod_j f(D_{ij}\,|\,T_i, q_j^{k-1})\right] f(T_i)}{\sum_{T_1} \cdots \sum_{T_N} \prod_i \left[\prod_j f(D_{ij}\,|\,T_i, q_j^{k-1})\right] f(T_i)} \tag{3.29}$$

Here, i is the voxel index and j is the segmentation index. The above equation gives the conditional probability density of the true segmentation given the observed segmentations and the previous estimate of the performance parameters. It can be written for each voxel i as,

$$f(T_i\,|\,D_i, p^{k-1}, q^{k-1}) = \frac{\prod_j f(D_{ij}\,|\,T_i, p_j^{k-1}, q_j^{k-1})\,f(T_i)}{\sum_{T_i} \prod_j f(D_{ij}\,|\,T_i, p_j^{k-1}, q_j^{k-1})\,f(T_i)} \tag{3.30}$$

Here, $\sum_{T_i}$ indicates the summation over all the possible values of T_i for every voxel i. Since T is a binary random variable it can take only two possible values zero or one. The above equation can therefore be written as,

$$f(T_i \mid D_i, p^{k-1}, q^{k-1}) = \frac{\prod_j f(D_{ij} \mid T_i, p_j^{k-1}, q_j^{k-1}) f(T_i)}{\prod_j f(D_{ij} \mid T_i = 1, p_j^{k-1}, q_j^{k-1}) f(T_i = 1) + \prod_j f(D_{ij} \mid T_i = 0, p_j^{k-1}, q_j^{k-1}) f(T_i = 0)} \tag{3.31}$$

In these equations, $f(T_i)$ represents the a priori probability of T_i. A voxel-wise independence assumption has been made here. Also, the individual segmentations are independent of each other. This allows us to write the joint probability equal to the product of the individual probabilities [19]. Since the true segmentation is treated as a binary random variable, the conditional probability can be written as,

$$f(T_i = 0 \mid D_i, p^{k-1}, q^{k-1}) = 1 - f(T_i = 1 \mid D_i, p^{k-1}, q^{k-1}) \tag{3.32}$$

Consider the following expressions,

$$a_i^{k-1} = f(T_i = 1) \prod_j f(D_{ij} \mid T_i = 1, p_j^{k-1}, q_j^{k-1}) \tag{3.33}$$

$$a_i^{k-1} = f(T_i = 1) \prod_{j:D_{ij}=1} f(D_{ij} \mid T_i = 1, p_j^{k-1}, q_j^{k-1}) \prod_{j:D_{ij}=0} f(D_{ij} \mid T_i = 1, p_j^{k-1}, q_j^{k-1}) \tag{3.34}$$

According to the definitions of sensitivity and specificity the above expression can be written as,

$$\prod_{j:D_{ij}=1} f(D_{ij} \mid T_i = 1, p_j^{k-1}, q_j^{k-1}) = \prod_{j:D_{ij}=1} p_j^{k-1} \tag{3.35}$$

$$\prod_{j:D_{ij}=0} f(D_{ij} \mid T_i = 1, p_j^{k-1}, q_j^{k-1}) = \prod_{j:D_{ij}=0} (1 - p_j^{k-1}) \tag{3.36}$$

Similarly,

$$\prod_{j:D_{ij}=0} f(D_{ij} \mid T_i = 0, p_j^{k-1}, q_j^{k-1}) = \prod_{j:D_{ij}=0} q_j^{k-1} \tag{3.37}$$

$$\prod_{j:D_{ij}=1} f(D_{ij} \mid T_i = 0, p_j^{k-1}, q_j^{k-1}) = \prod_{j:D_{ij}=1} (1 - q_j^{k-1}) \tag{3.38}$$

Let,

$$a_i^{k-1} = f(T_i = 1) \prod_{j:D_{ij}=1} p_j^{k-1} \prod_{j:D_{ij}=0} (1 - p_j^{k-1}) \tag{3.39}$$

$$b_i^{k-1} = f(T_i = 0)\prod_{j:D_{ij}=0} q_j^{k-1} \prod_{j:D_{ij}=1} (1 - q_j^{k-1}) \quad (3.40)$$

Equation (3.31) can then be written as,

$$W_i^{k-1} = f(T_i = 1 \mid D_i, p^{k-1}, q^{k-1}) = \frac{a_i^{k-1}}{a_i^{k-1} + b_i^{k-1}} \quad (3.41)$$

The above equation gives the probability of the true segmentation at voxel i being equal to one. The E-step requires the calculation of the log likelihood function of the entire data. The derivation for this is given in the next section.

M-step

Given the conditional probability of true segmentation W_i^{k-1}, the values of the performance level parameters can be calculated.

Consider equation (3.27),

$$(p^k, q^k) = \underset{p,q}{\operatorname{argmax}}\, E\left[\ln(f(D \mid T, p, q) f(T)) \mid D, p^{k-1}, q^{k-1}\right] \quad (3.42)$$

This can be written as follows considering voxel-wise independence,

$$(p^k, q^k) = \underset{p,q}{\operatorname{argmax}} \sum_j \sum_i E\left[\ln f(D_{ij} \mid T_i, p_j^{k-1}, q_j^{k-1}\right] \quad (3.43)$$

Here $f(T)$ has not been considered since it is independent of (p, q).

Thus for each automatic segmentation j ,

$$
\begin{aligned}
(p_j^k, q_j^k) &= \arg\max_{p_j, q_j} \sum_i E\left[\ln f(D_{ij} \mid T_i, p_j^{k-1}, q_j^{k-1}) \mid D, p^{k-1}, q^{k-1}\right] \\
&= \sum_{T_i} \sum_i \ln f(D_{ij} \mid T_i, p_j^{k-1}, q_j^{k-1}) f(T_i \mid D, p^{k-1}, q^{k-1}) \\
&= \sum_{i:D_{ij}=1} \ln f(D_{ij} \mid T_i = 1, p_j^{k-1}, q_j^{k-1}) f(T_i = 1 \mid D, p^{k-1}, q^{k-1}) \\
&+ \sum_{i:D_{ij}=0} \ln f(D_{ij} \mid T_i = 1, p_j^{k-1}, q_j^{k-1}) f(T_i = 1 \mid D, p^{k-1}, q^{k-1}) \\
&+ \sum_{i:D_{ij}=0} \ln f(D_{ij} \mid T_i = 0, p_j^{k-1}, q_j^{k-1}) f(T_i = 0 \mid D, p^{k-1}, q^{k-1}) \\
&+ \sum_{i:D_{ij}=1} \ln f(D_{ij} \mid T_i = 0, p_j^{k-1}, q_j^{k-1}) f(T_i = 0 \mid D, p^{k-1}, q^{k-1})
\end{aligned}
$$

$$
(p_j^k, q_j^k) = \sum_{i:D_{ij}=1} W_i^{k-1} \ln p_j + \sum_{i:D_{ij}=0} W_i^{k-1} \ln(1 - p_j) + \sum_{i:D_{ij}=0} (1 - W_i^{k-1}) \ln q_j + \sum_{i:D_{ij}=1} (1 - W_i^{k-1}) \ln(1 - q_j)
$$

Differentiating the above equation with respect to p_j and setting this equal to zero gives the maximum value of the sensitivity. The final expression obtained is as follows,

$$
\hat{p}_j^{\,k} = \frac{\sum_{i:D_{ij}=1} W_i^{k-1}}{\sum_{i:D_{ij}=1} W_i^{k-1} + \sum_{i:D_{ij}=0} W_i^{k-1}} \tag{3.44}
$$

Similarly expression for q_j can be obtained as,

$$
\hat{q}_j^{\,k} = \frac{\sum_{i:D_{ij}=0} (1 - W_i^{k-1})}{\sum_{i:D_{ij}=1} (1 - W_i^{k-1}) + \sum_{i:D_{ij}=0} (1 - W_i^{k-1})} \tag{3.45}
$$

This sequence is iterated until convergence is reached.

Implementation and Algorithm

The above algorithm has been implemented using C++ programming language. The program takes four automatic segmentations for each structure as input and calculates the ground truth and the performance parameters for each automatic segmentation. The ground truth segmentation is saved as an image and the parameters are exported to an Excel file for validation purpose. The entire algorithm is summarized in table 3.3 below.

Table 3.3: Algorithm for the classifier combination method.

```
Set the path for the four automatic segmentations as input to the STAPLE program.
        For every structure
            • Set the sensitivity and specificity parameters to 0.9
            While convergence is not reached
                • Find the ground truth segmentation from the current estimate
                • Find the new estimate of sensitivity and specificity from the
                  ground truth obtained above.
            End While loop
            • Save the ground truth as a binary raw image.
            • Save the final performance parameters into a text file.
        End For loop
```

The above algorithm has been performed independently for every class (structure). It is therefore called as the binary STAPLE algorithm [25]. Instead of performing the STAPLE algorithm for each class, all the classes can be treated simultaneously [25] and [19]. However, we have not used this multi-class algorithm because it has been proved by T. Rohlfing in his paper [25] that the repeated application of the binary STAPLE algorithm outperforms the multi-class method.

Summary

This chapter explained all the three algorithms used to improve the accuracy of the original atlas-based segmentation method. The results obtained using these algorithms have been validated in chapter V. All the different programs constituting the atlas-based segmentation pipeline have been combined together into a single program. This pipeline architecture is explained in the next chapter.

CHAPTER IV

PROJECT PIPELINE

This chapter explains the various programs forming the pipeline of atlas-based segmentation method. It also describes the method of integrating all the programs into a single program. This helps in automatic execution of the pipeline without any user intervention. The pipeline has been created using the CT-MR fusion method. It essentially has been created for the physicians in the Radiation Oncology Department, Vanderbilt University.

Concept of a Pipeline

As has been described in the previous chapters, atlas-based segmentation consists of several different parts. Over the years, these parts have been implemented using different programming languages. The essential components of atlas-based segmentation can be summarized as,

- DICOM to raw image conversion (IDL)
- CT to MR rigid registration (C language)
- CT-MR fusion (MATLAB)
- Fused Atlas to fused patient rigid registration (C language)
- Atlas to patient non-rigid registration (C++ language)
- Model Projection on CT volume(C language)
- Mask Threshold (C language)
- Mesh Creation (VTK with C++ language)

- Contour extraction and DICOM-RT file creation (VTK with C++ language)

All these different programs are executed one after the other to achieve segmentation of the brain structures. The serial execution of these programs is termed as the "Project Pipeline". The rigid and non-rigid registration algorithms can be executed using a script file. This file contains the information about the source and target images and their dimensions, voxel spacing and orientations. The registration parameters are also set through these script files. To begin the registration process, the executables of the rigid and non-rigid programs along with the respective script files have to be executed from the DOS prompt. To run these programs serially, a batch file has to be created in DOS. The CT-MR fusion is achieved by using a code written in MATLAB. For projecting the atlas binary masks onto the target image, a mask deformation program has been written in the C language. This program is also executed using a script file. This file contains the paths and names of all the binary masks defined on the atlas. The paths of the rigid and non-rigid deformation fields are also provided to this program via the script file. The atlas masks are then deformed using the inverse transformation from the target to the atlas. The obtained masks are thresholded at a value of 128 using another C program. The thresholded masks are then given to the mesh creation program written in C++ using VTK. These meshes are then given as input to the DICOM-RT program, which extracts the contour points and creates the DICOM-RT structure files. Thus the various programs are executed individually and this pipeline requires user intervention at nearly every stage.

Need for Automatic Execution of Pipeline

As is discussed in the previous section, the execution of the pipeline requires manual intervention at every stage. The entire segmentation process takes about 3.5 hours for completion. The rigid and non-rigid registrations itself take nearly 3 hours. The user cannot wait for so long to finish one step of the pipeline. He therefore has to come back and run the rest of the programs to complete the segmentation process. This is very inconvenient, especially for the physicians who will be using the programs for radiotherapy planning. Thus, though the segmentation method is automatic, its execution is not automated. This creates a necessity of automatic execution of the pipeline in order to realize the project practically. The following sections describe the method used to integrate the pipeline into a single program.

Modifications in the original pipeline

In order to integrate all the programs into a single program, most of the original programs have been modified to suit this requirement. Since rigid and non-rigid registration programs have to be executed using a script file on DOS prompt, they cannot be directly used in any other C program. To facilitate its usage in every application, Dynamic Link Libraries (DLLs) have been created for the rigid and non-rigid registration algorithms. These libraries essentially consist of “rigid/non-rigid classes” (C++ class notion), which have different methods to set the registration parameters. Thus there is no need of a separate script file to set the registration parameters. The registration will begin by invoking the ‘Execute()’ method in both the libraries. The registered images and the deformation fields can be saved as well as can be obtained directly by calling the respective methods. For CT-MR fusion as well, the code has been written in C++ and is easily incorporated into the pipeline.

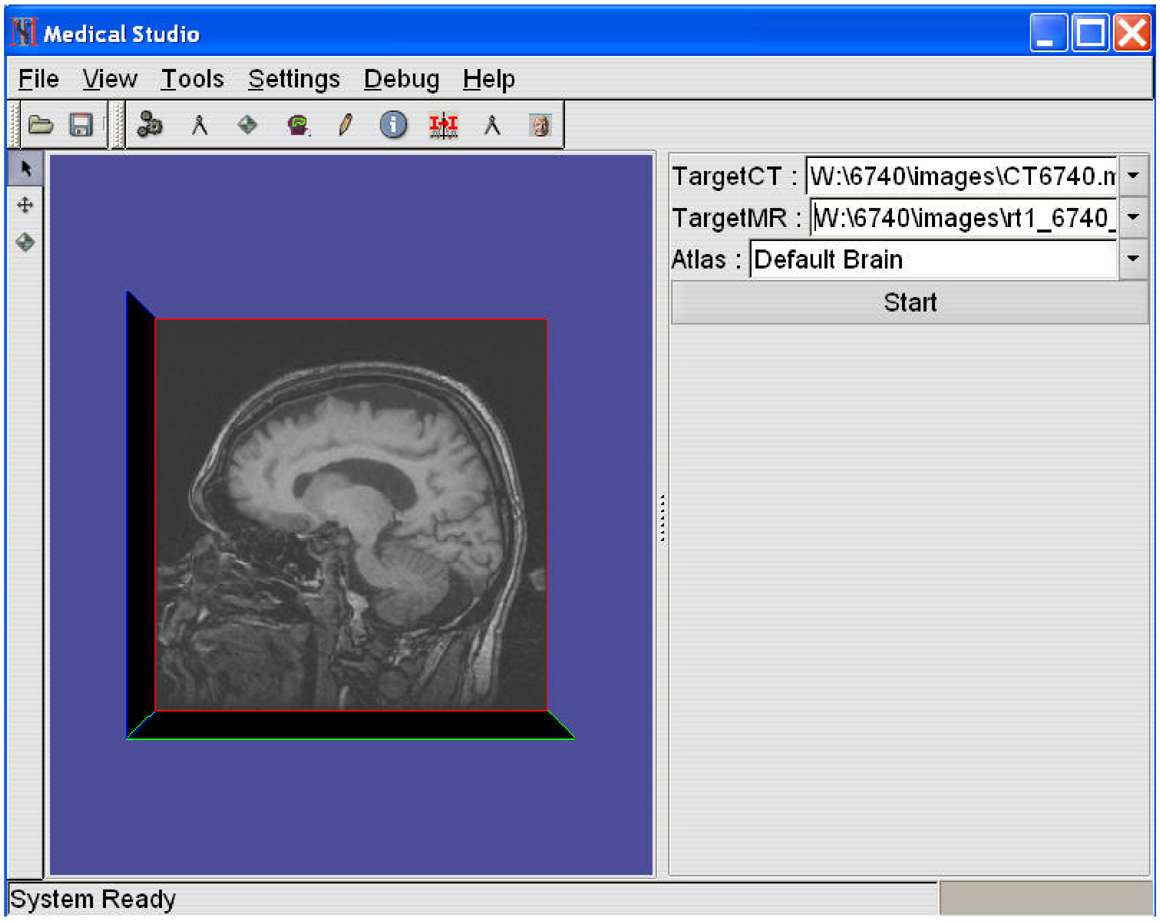
Once the images have been registered, the atlas masks have to be deformed using another program similar to the registration program. The mask deformation program has also been written using the C language. Since this also has to be executed using a script file, we create a DLL for the deformation program. This DLL consists of a “deform” class and has various methods to set the paths of the deformation fields. To begin mask deformation, a “Deform()” method has to be invoked. Three deformation fields have to be applied to the atlas masks so as to project them on the target CT volume. This is essential because the Radiation Oncology Department uses CT images to carry out the radiation therapy planning. Once the masks have been deformed, these are given as input to the mesh creation program written in C++ using VTK. These meshes are then given as input to the DICOM-RT program. This code extracts contour points from the meshes and creates the DICOM-RT structure files. These contours are then loaded in the Radiation Oncology Department. The DICOM-RT code can be directly inserted into the pipeline as it is entirely written in C++. Thus the new components of the pipeline can be summarized as,

- CT to MR rigid registration (DLL in C++)
- CT-MR fusion (C++ language)
- Atlas to Patient rigid registration (DLL)
- Atlas to Patient non-rigid registration (DLL)
- Mask deformation (DLL)
- Mesh creation and contour extraction (C++ language using VTK)
- DICOM RT file creation (C++ language using VTK)

Thus, all the components of the pipeline can be executed serially using a single C++ program. An end user segmentation module has been created for easy interface. Its architecture has been discussed in the next section.

Segmentation Module Architecture

An end user segmentation tool can be created from the integrated pipeline. This can be achieved using the free software called "Medical Studio" [30]. It is based on popular open source libraries like VTK for visualization [29], ITK for image processing, GTK for graphical user interface and DCMTK [31] for DICOM compatibility. Medical Studio is built on a plug-in architecture and thus allows easy integration of new functionalities. The entire C++ code which was written previously to run the pipeline automatically has been imported into Medical Studio. Then a DLL is created called "libAutoSegment.dll". This DLL acts as the segmentation module plug-in into the Medical Studio environment. Since Medical Studio already has DICOM compatibility, DICOM files can be directly loaded into the program using the Medical Studio GUI (Graphical User Interface). Thus the conversion of DICOM files to raw image files need not be done manually. The atlas masks should be stored in the same directory where the Medical Studio executable is located. The segmentation module also has its own GUI which will be displayed once the plug-in gets loaded into the environment. This GUI has three combo boxes, one for the target CT volume, one for the target MR volume and one for the Atlas image volume. The "START" button on the GUI has to be pressed to begin program execution. This module can be easily loaded on any machine in the Radiation Oncology Department. With its simple and user friendly interface, any physician can generate automatic contours and import them in their planning software. The figure 4.1 shows a screen shot of Medical Studio with the segmentation module GUI.



TargetCT : W:\6740\images\CT6740.m

TargetMR : W:\6740\images\rt1_6740_

Atlas : Default Brain

System Ready

Figure 4.1: GUI of the segmentation module developed using Medical Studio

CHAPTER V

RESULTS AND VALIDATION

This chapter summarizes the results generated by all the methods described in Chapter III. For each patient, there are three manual segmentations drawn by three physicians from the Radiation Oncology Department at Vanderbilt University. These manual contours have been drawn on randomly selected slices for every structure. This is done because it is extremely time consuming to draw the contours slice by slice for the whole structure. The manual contours are then compared to the automatically generated contours for validation. In this thesis, three validation techniques have been used to determine quantitatively how "well" the automatic segmentation is as compared to the manual ones. This chapter explains the two validation techniques used and compares the automatic segmentation with the manual segmentation. Finally, the segmentation results generated by all the methods are compared to determine which one gives the best results.

Validation Techniques

Verification of the performance of automatic segmentation algorithms is important because they may often lack accuracy and precision. A common means to verify the accuracy of automatic image segmentation is by behavioral comparison. The result of the automated algorithm is compared to the interactive segmentation generated by experts. If the two results are close to an acceptable tolerance the automated algorithm is considered acceptable for the purpose. Manual segmentations generated by different experts suffer from inter-rater and intra-rater variability. Thus it is necessary to derive “ground truth” segmentation from the group of expert segmentations. There are various ways to combine the expert segmentations. The method that we use here is the "STAPLE" algorithm proposed by Warfield et al. [18], [19]. This algorithm is the same as the one used to find the ground truth segmentation in the multi-classifier combination method explained in Chapter III. The only difference being that instead of combining automatic segmentations we combine the three manual segmentations drawn for every structure.

There are various metrics proposed for the comparison of automatic segmentations with the segmentations produced by a group of experts. Following are the methods used in this study to validate the atlas-based segmentations:

<u>Dice Similarity Coefficient or S-index:</u>

This method was first developed by L. R. Dice [28]. It compares the shape and size of the manual and auto-segmented masks. It calculates an index of similarity between each measurement pair (binary masks). The similarity index between two measurements $S(S \in [0,1])$ is defined as the ratio of twice the common area to the sum of the individual areas [8], [15]. For a structure s, the similarity index is defined as:

$$S(s) = \frac{2\left|V_{manual}^{(s)} \cap V_{atlas}^{(s)}\right|}{\left|V_{manual}^{(s)}\right| + \left|V_{atlas}^{(s)}\right|} \tag{4.1}$$

where, $V_{manual}^{(s)}$ denotes the number of voxels labeled as belonging to structure s by manual segmentation and $V_{atlas}^{(s)}$ denotes the number of voxels labeled as belonging to structure s by atlas-based segmentation. For a perfect mutual overlap of manual and atlas-based segmentation, S has a value of 1. It will have smaller values in case of imperfect overlaps.

The manual segmentation used here is the ground truth obtained by combining several manual segmentations with the STAPLE algorithm [18].

Distance Measure:

Mask-based comparison is a good similarity measure for volumetric measurements but it does not provide precise information on the contours themselves [15]. This method directly compares the manual and automatic contours. The ground truth contour is calculated from the group of manual segmentations by using the STAPLE algorithm [18]. The automatic contour is then compared to this ground truth contour on a point-by-point basis. For every point on the automatic contour, its minimum Euclidean distance is calculated from the ground truth contour. We then determine the percentage of automatic contour points that fall within 1 pixel (1 mm.) distance from the ground truth contour. The higher the percentage, the better is the automatic segmentation. We also calculate the mean distance error and the maximum distance error of the automatic contour from the manual ground truth contour.

The segmentation decisions made by the different atlas-based methods are discussed in the next section. For every method, the results have been validated by the two techniques explained above.

Results

This section presents the results obtained by the three methods implemented in chapter III. We also provide the results obtained by the original atlas-based method discussed in chapter II for comparison. For every method the S-indices and the distance measures have been computed. We have performed a total of five experiments to determine which method gives the best segmentation result for the atlas-based method.

Experiment 1

This experiment was performed to study the effect of fusing CT and MR image volumes on the segmentation results. This section gives the S values and distance measures for the original atlas-based segmentation method compared with the CT-MR fusion method.

Mask-based comparison: Here we present the results obtained from the 10 patients in the dataset. The actual dataset consists of 20 patients but we have CT images for only 10 patients. The table shows average S-indices for 10 structures obtained from all these 10 patients. The masks obtained by the CT-MR fusion method have been compared with the ground truth mask obtained from the manual segmentations drawn by three experts. Table 5.1 gives the S-indices for both the original atlas-based method and the CT-MR fusion technique.

Table 5.1: Mask-based comparison (S-indices) between automatic and manual masks for the CT-MR fusion method and the original atlas-based method.

	MR only	CT-MR
Brainstem	0.90	0.90
Cerebellum	0.88	0.89
Chiasm	0.62	0.64
Pituitary	0.45	0.55
Left eye	0.88	0.86
Right eye	0.86	0.86
Left lens	0.65	0.68
Right lens	0.74	0.71
Left optic nerve	0.49	0.50
Right optic nerve	0.62	0.62

A comparison between the two methods can be illustrated with the help of the following bar graph.

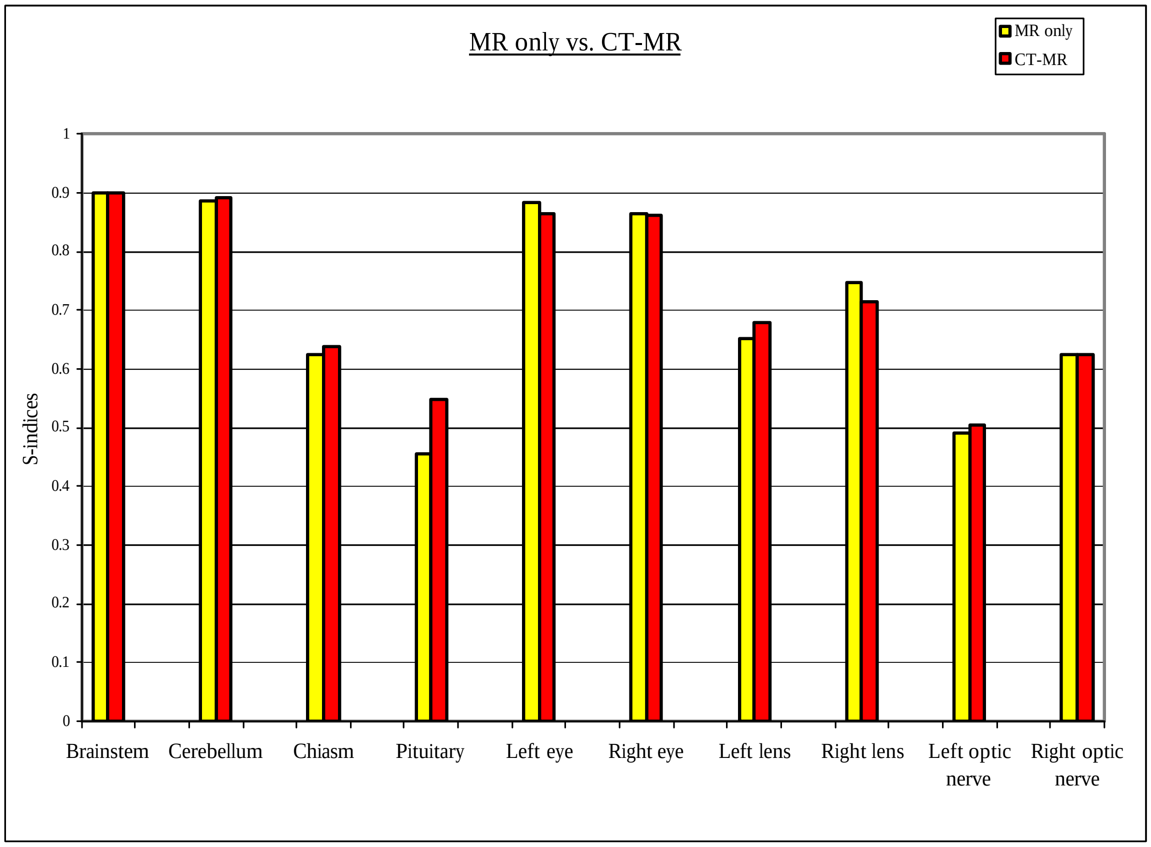


Figure 5.1: Mask-based comparison between the original method and the CT-MR fusion method.

Contour-based comparison: This section gives the contour-based comparison between automatic and manual contours (ground truth). The percentage of automatic contour points that fall within 1 mm. and 2 mm. distance of the ground truth contour has been calculated. We also calculate the mean and the maximum distances of the contour points from the ground truth contour. Again, the ground truth contour has been calculated by using the STAPLE algorithm. Table 5.2 gives these distance measures for the original atlas-based method as well as for the CT-MR fusion method.

Table 5.2: Contour-based comparison between the manual ground truth and the automatic segmentation for original atlas-based method and CT-MR fusion method.

	MR only % In=1mm.	CT-MR % In=1mm.	MR only % In=2mm.	CT-MR % In=2mm.
Brainstem	79.60	78.21	93.09	92.67
Cerebellum	45.22	48.83	69.04	70.70
Chiasm	50.36	51.90	73.31	74.78
Pituitary	33.83	35.17	54.90	58.70
Left eye	65.25	60.76	83.23	79.96
Right eye	63.54	62.40	83.34	81.94
Left lens	74.00	79.52	95.80	95.87
Right lens	77.68	73.91	94.58	95.16
Left optic nerve	45.57	42.08	69.29	69.03
Right optic nerve	60.80	61.57	84.80	86.36

A comparison between the two methods can be demonstrated with the help of the bar graph shown in figure 5.2.

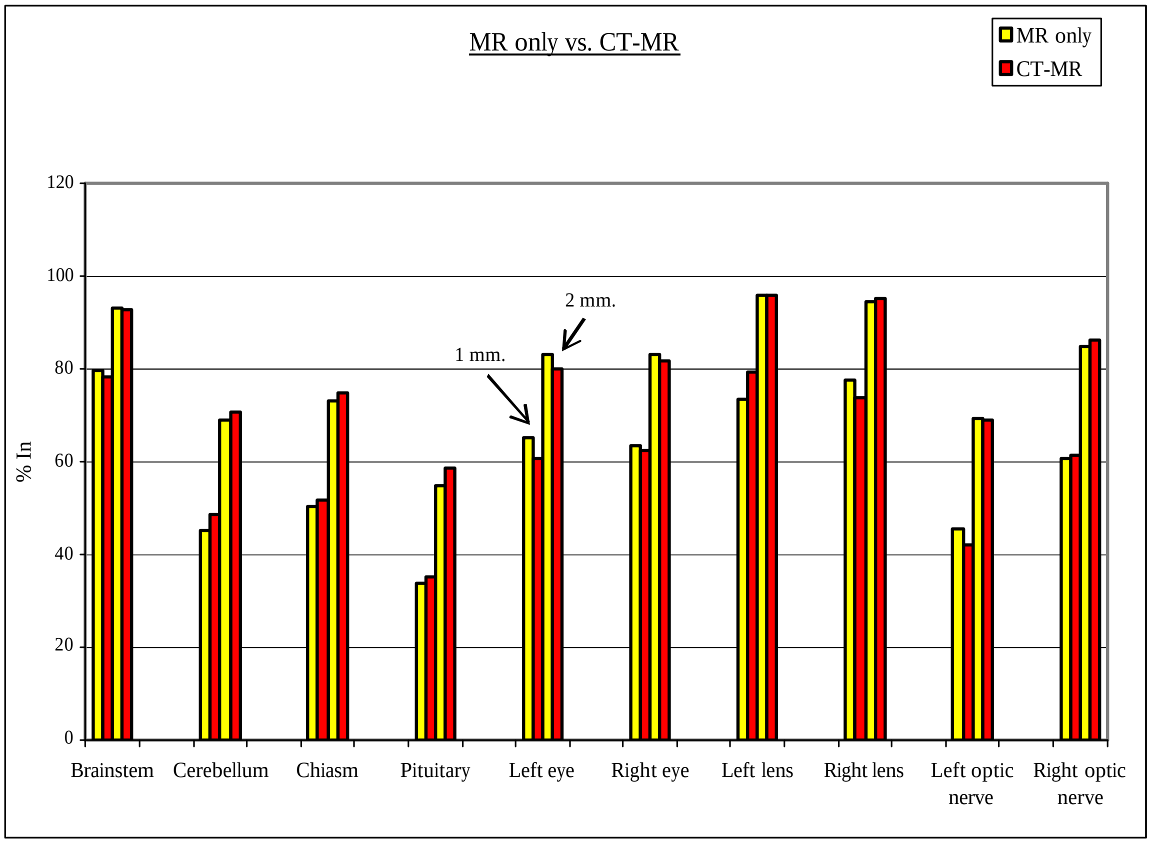


Figure 5.2: Comparison between the original method and the CT-MR fused method.

Discussion

In this section we discuss the problems encountered in the original method and explain the results obtained by the CT-MR fusion method. As can be seen from tables 5.1 and 5.2, the brainstem, eyes, lenses, chiasm and the optic nerves do not show any improvement in their segmentation. However, the pituitary gland and the cerebellum show some improvement and the possible reasons for this improvement are discussed below.

Improvement in the cerebellum contour: Atlas-based segmentation method sometimes leads to poor results in the region around the cerebellum. The main problem is usually observed in the lower part of the cerebellum. This is due to poor edge definition as well as intensity inhomogeneities that affect MR image volumes [15]. In the original study, this issue was addressed by post processing the obtained contours using a geometric deformable model algorithm [15]. In this thesis, we try to minimize this problem by combining CT and MR volumes of the patient. The figure 5.3 shows the problem with a cerebellum contour:

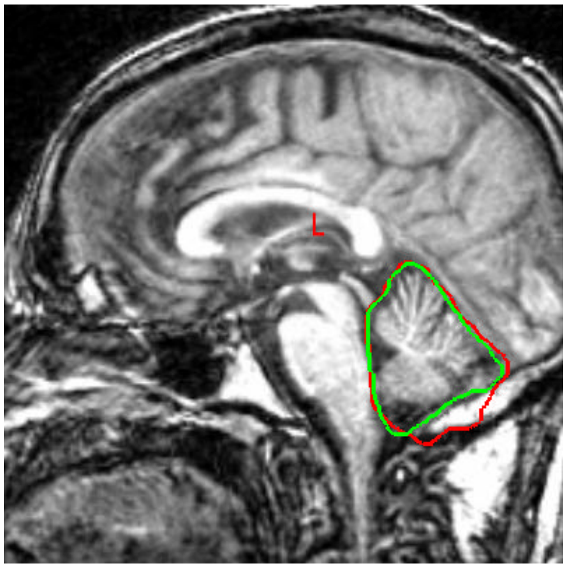


Figure 5.3: Cerebellum leakage in the skull. The red contour is the original contour and the green contour is the CT-MR fused contour.

As can be seen, the red contour (original method) leaks into the skull which results in poor segmentation of the cerebellum. This occurs due to poor edge definition in the lower part of the cerebellum. In the CT-MR fusion method, the results in this part improve due to a sharp intensity difference at the border of the cerebellum and the skull bone. The resulting contour is shown in green color in figure 5.3 above. This improvement can also be seen with the help of the comparative bar graphs in figures 5.1 and 5.2. It can be seen that there is only a slight improvement (1% for masks and 4% for contours) in the cerebellum segmentation. A possible reason why a significant improvement in the cerebellum segmentation cannot be observed is that there are not many patients in our dataset which have leaking in the skull bone. The slices which have such leaking for the cerebellum should also get selected by the experts for drawing manual segmentations. In addition to the leakage issue, the cerebellum also suffers from some other problems which will be discussed later.

Improvement in the pituitary gland segmentation: Along with the improvement in cerebellum segmentation, some improvement has also been observed in the segmentation of the pituitary gland. This is shown in figure 5.4 below:

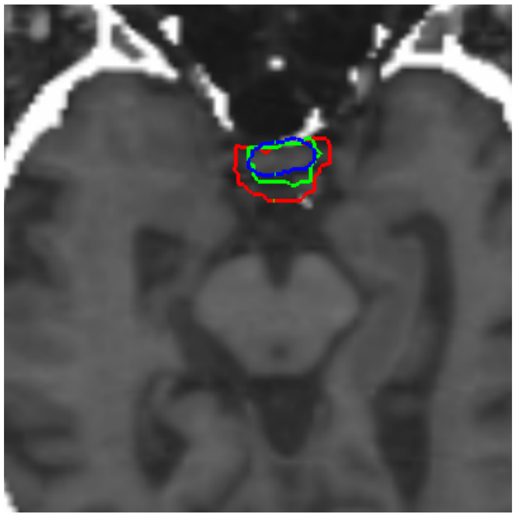
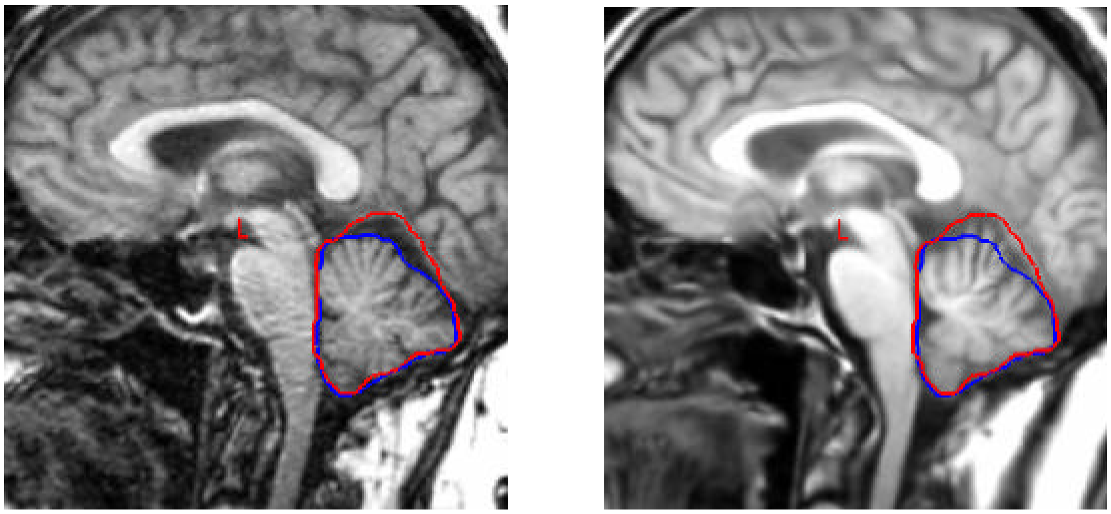


Figure 5.4: Improvement in pituitary gland segmentation.

In figure 5.4, the red contour is the original contour, the green contour is the CT-MR fused contour and the blue one is the manual ground truth contour. As can be seen, the pituitary gland is surrounded by bones when the CT and MR images are fused. In the original MR volume, the pituitary had very poor edge definition which causes leaking and incorrect segmentation. But once the CT and MR volumes are fused the pituitary gets defined better, which improves its segmentation. The pituitary gland segmentation improves by 10% for the masks and 2% for the contours. This is also evident from the comparative graphs shown in figures 5.1 and 5.3 above.

Cerebellum Problem: The problem of cerebellum leakage has been minimized by using CT-MR fusion technique. But this method only prevents the leakage in the skull and not in the brain cortex. The situation is illustrated in figure 5.5:



(a) (b)

Figure 5.5: Cerebellum leakage in the brain cortex. (a) Target image and (b) source deformed to the target.

The above figure shows a target image on the left side and the atlas deformed to the target on the right side. The blue contour is the manually drawn contour and the red contour is the automatically generated contour. As can be seen, the cerebellum contour leaks in the brain cortex. This happens because of the gap that is present in the upper part of the cerebellum and the cortex. This gap is not recognized by the registration because a similar gap is not present in the atlas image. The quality of the segmentation of the cerebellum could be improved if this gap gets recognized by the registration algorithm. To achieve this objective, the multi-classifier method incorporates two out of four atlases which have a similar gap between the cerebellum and the cortex. The results of multi-classifier method will be discussed later.

Experiment 2

This experiment was performed to demonstrate the difference between the mask-deformed and mesh-deformed segmentations. In the previous experiment, it was shown that the CT-MR fusion method was slightly better than the original MR only method. We however had only 10 patients with CT volumes. Hence in all the further experiments we have used MR volumes alone, as it gave us a dataset of 20 patients to work with. It should however be noted that the results obtained in the following experiments would likely improve a little if the CT-MR fusion method was also used.

Mask-based comparison: Here the mask-deformed masks and the mesh-deformed masks are compared to the ground truth mask. The results have been obtained from 20 patients and the table 5.3 below presents average S-indices for all the ten structures.

Table 5.3: S-indices for mask and mesh deformed segmentations.

	Mask	Mesh
Brainstem	0.92	0.92
Cerebellum	0.89	0.89
Chiasm	0.54	0.53
Pituitary	0.49	0.48
Left eye	0.87	0.85
Right eye	0.87	0.86
Left lens	0.70	0.69
Right lens	0.77	0.74
Left optic nerve	0.51	0.48
Right optic nerve	0.61	0.60

A graph is shown in figure 5.6 which compares the two methods.

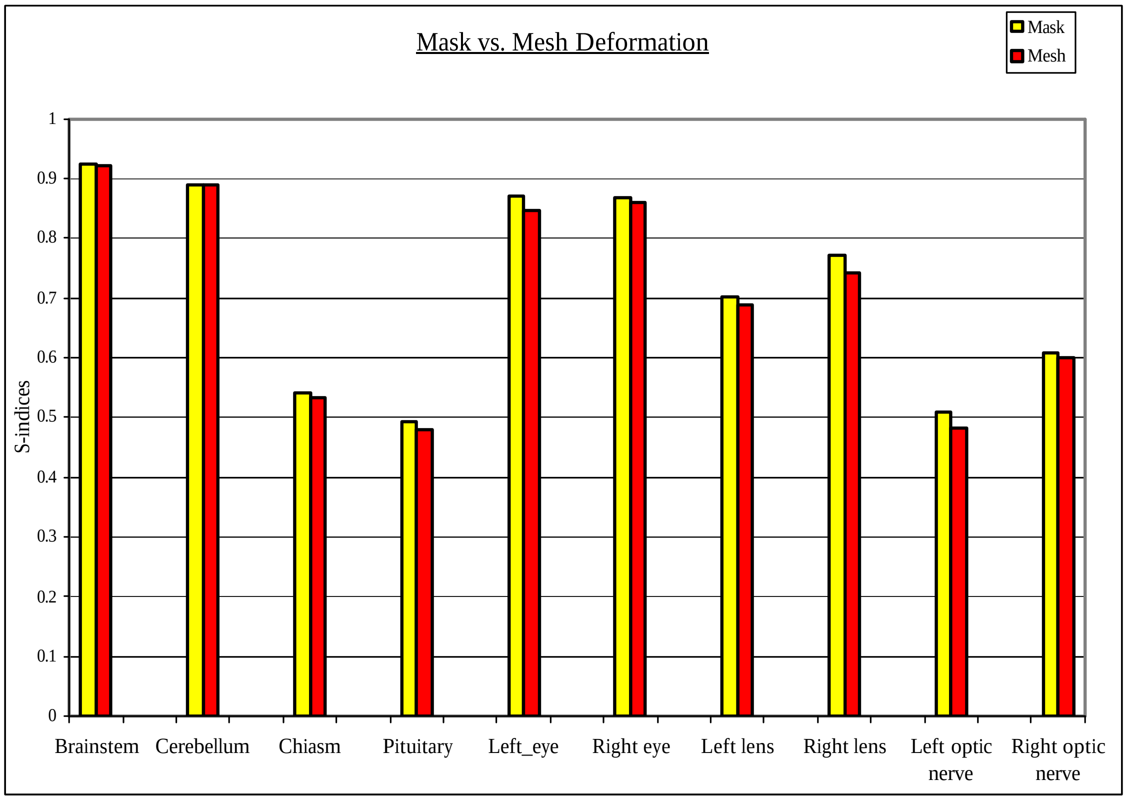


Figure 5.6: Comparison between mask and mesh deformation

Contour-based comparison: The mask-deformed and the mesh-deformed contours are compared with the ground truth contour. The average values of the distances are tabulated in table 5.4.

Table 5.4: Contour-based comparison for mask-deformed and mesh-deformed contours.

	Mask % In=1mm.	Mesh % In=1mm.	Mask % In=2mm.	Mesh % In=2mm.
Brainstem	81.58	74.80	94.08	93.73
Cerebellum	47.26	39.836	70.98	70.03
Chiasm	45.84	37.36	68.91	63.91
Pituitary	35.32	27.45	52.77	48.88
Left eye	67.30	57.90	83.98	82.86
Right eye	67.55	56.52	85.59	84.13
Left lens	71.99	69.48	96.67	97.11
Right lens	78.73	65.25	95.53	92.46
Left optic nerve	47.69	35.18	73.01	68.72
Right optic nerve	58.92	47.05	83.87	77.47

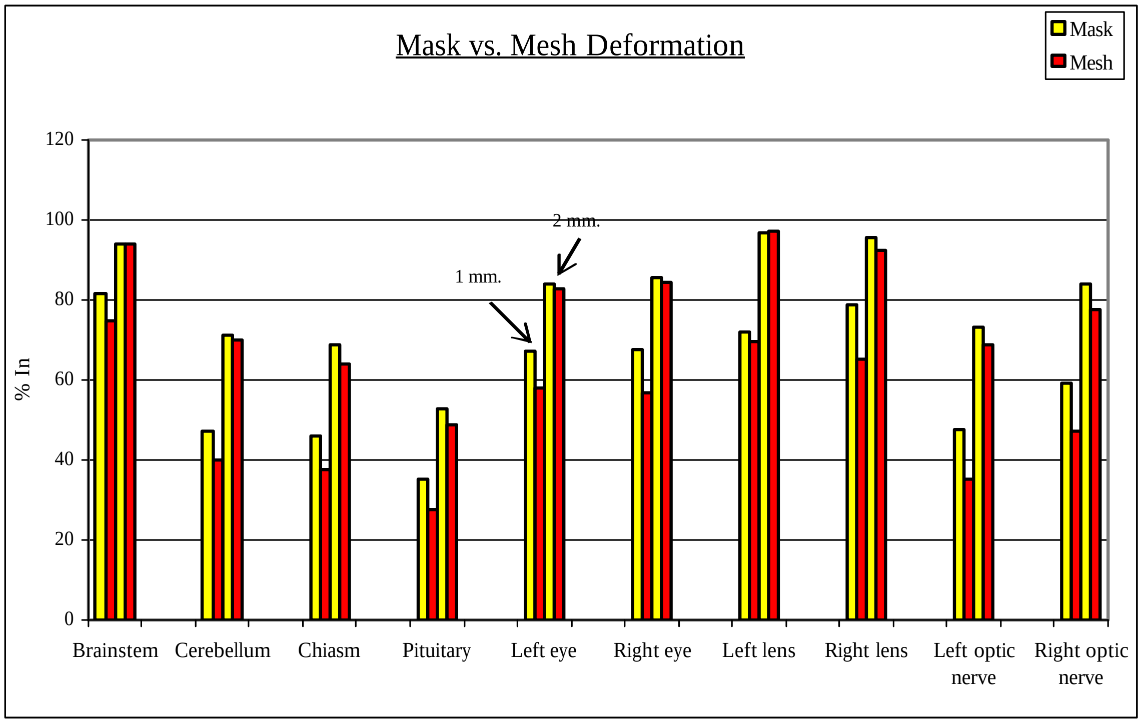


Figure 5.7: Comparison between mask and mesh deformation

Discussion

As can be seen from figures 5.6 and 5.7, the results obtained from mesh deformation are not better than those obtained by deforming the binary masks. In fact for most of the structures the mesh deformed contours are worse than the original atlas-based contours. To explain why the results are not better for mesh deformations we present a few segmentation results (contours) obtained by both the methods in figure 5.8 below.

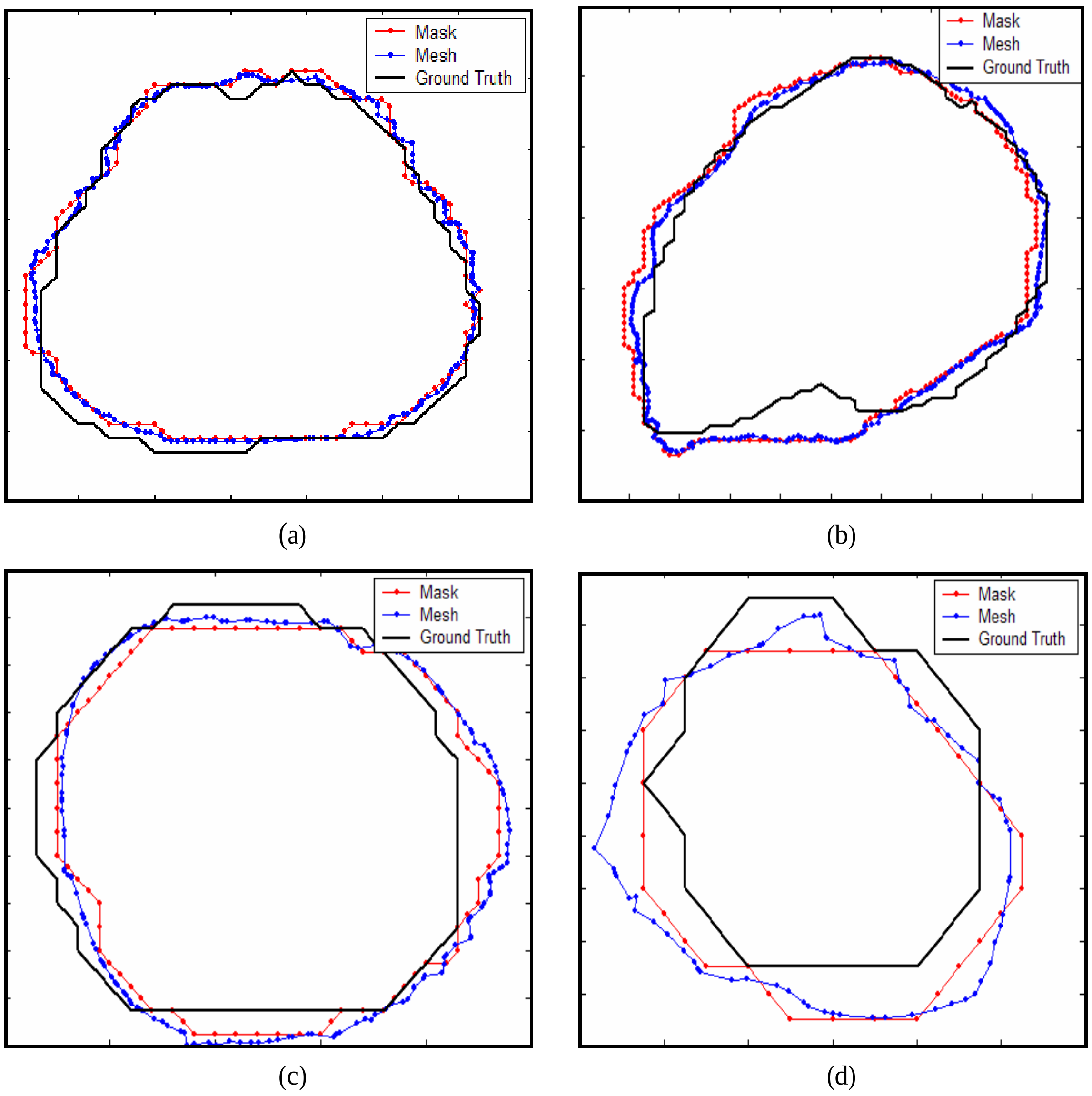
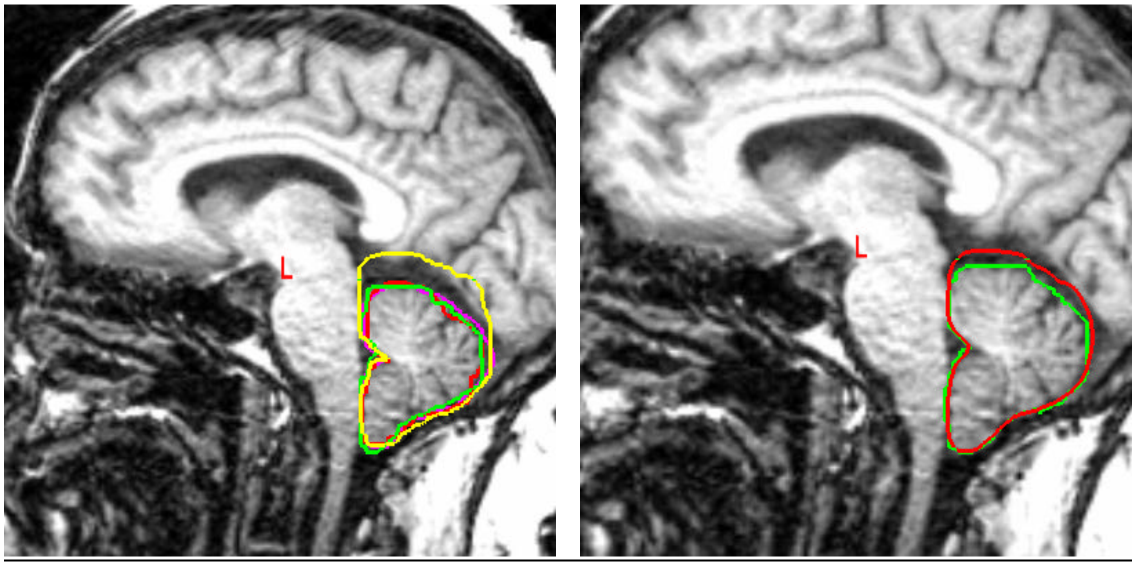


Figure 5.8: Contours generated by mask and mesh deformation. (a) Contours for brainstem (b) contours for cerebellum (c) contours for left eye and (d) contours for right optic nerve.

In figure 5.8, the red contour is the original mask deformed contour, the blue contour is the mesh deformed contour and the black contour is the manual ground truth contour. As can be seen the two contours are not very different from each other. We have tried a few variations in the mesh deformed contours. The mesh deformed contours were smoothed using a low pass filter so that the high frequency content could be minimized as much as possible. It was observed that the smoothed contours gave better results than the non-smoothed ones. We had also tried decimation of the meshes before extracting the contours. From all these small experiments, it was concluded that the contours extracted from smooth meshes without decimation gave the best results. However, these contours are not better than the original mask deformed contours. The exact reason for this has not been elucidated. A possible cause is the interpolation of the deformation field that occurs because the contour points are not defined on a regular grid. Interpolation effects also occur with the mask deformation method but these are mitigated by the thresholding of the deformed mask. There is no such thresholding with mesh deformations. Since the meshes were not found to be better than the masks, in all the further experiments we have used mask deformations.

Experiment 3

This experiment was performed to highlight the differences between the average contour and the STAPLE ground truth in classifier combination. In this section we consider some synthetic contours and find their average contour as well as the STAPLE ground truth. The figure 5.9 below shows the four contours considered in this experiment.



(a) (b)

Figure 5.9: Difference between average contour and STAPLE ground truth contour (a) shows the four contours to be combined (b) shows the average contour (red) and STAPLE ground truth contour (green).

As shown in figure 5.9(a), we have four contours each shown with a different color. Three of these contours are similar and indicate correct segmentation of the cerebellum. But the fourth one (yellow) represents incorrect segmentation. These contours have been combined using two different classifier combination methods namely by taking the average contour and by finding the STAPLE automatic ground truth. Figure 5.9(b) shows the average contour in red color and the automatic ground truth contour in green. As can be seen, the automatic ground truth is closer to the manual ground truth while the average contour shows some leaking in the upper part. This is because the STAPLE algorithm works probabilistically and will therefore generate an output very similar to the three similar segmentations. It combines the segmentations by weighing them according to their performances. On the other hand, the average contour will get affected by the bad segmentation because it will treat all the contours equally. Thus, in such a situation the STAPLE automatic ground truth will generate a better result as compared to the average contour. If all the contours to be combined are similar, i.e. all the different classifiers are of the same quality, then the STAPLE and the average methods will generate more or less the same outputs. This is shown in figure 5.10 below.

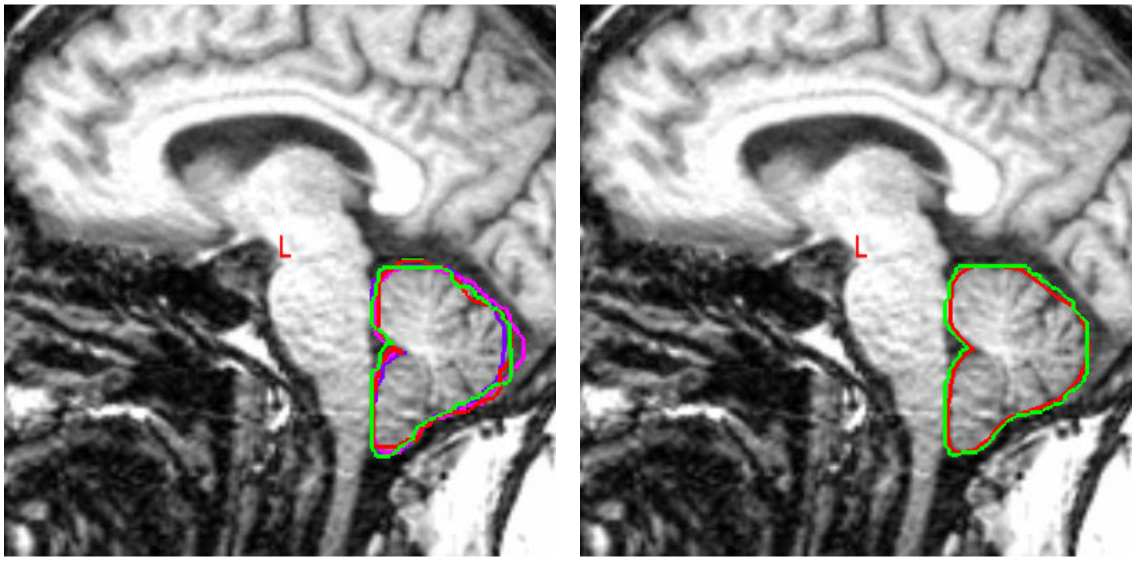


Figure 5.10: Difference between STAPLE and average contour when all the contours to be combined are similar. (a) Contours to be combined (b) average (red) and STAPLE (green) contour.

As can be seen the average contour and the STAPLE automatic ground truth are more or less similar as is evident from figure 5.10(b). Thus, it can be concluded that the STAPLE algorithm will work better if some classifiers are good and some are bad. Since some of our classifiers can be good and some can be bad we have used STAPLE algorithm to find the combined output. In addition, for STAPLE to produce good results, the classifier errors need to be as independent as possible. If there are two contours representing correct segmentation and two representing bad segmentation, the STAPLE will generate better results than the average contour if the bad segmentations do not make the same mistakes. If there are multiple classifiers misclassifying the same set of voxels, STAPLE will get biased and will misinterpret their agreement as an evidence of their correctness. In such a situation, the average contour may give better results. Thus for STAPLE to produce good results, the individual classifiers have to be independent and different from each other. Each of our classifiers can be considered different because each has a different atlas and different atlases will produce different outputs depending on the quality of registration. In the next experiment, we have validated the results obtained by the STAPLE algorithm. In the following experiments, we have only used STAPLE to combine the segmentations as we did not have an accurate and an easy way of finding the average contour.

Experiment 4

This experiment was conducted for the multi-classifier method implemented in chapter III. In this, we have compared the STAPLE output with the original atlas-based output. All the validation is with respect to the ground truth manual segmentation.

Mask-based comparison: Here we have presented the S-indices obtained for all the four atlases used in classifier combination. We also present the S-indices for the combined output using the STAPLE algorithm. Table 5.5 shows the average S-indices for 20 patients.

Table 5.5: S-indices for the individual atlases and the STAPLE output.

	Atlas #1	Atlas #2 (original)	Atlas #3	Atlas #4	STAPLE
Brainstem	0.93	0.92	0.92	0.84	0.94
Cerebellum	0.90	0.89	0.86	0.86	0.92
Chiasm	0.77	0.54	0.70	0.61	0.79
Pituitary	0.48	0.36	0.41	0.51	0.53
Left eye	0.91	0.88	0.88	0.84	0.93
Right eye	0.89	0.86	0.80	0.81	0.91
Left lens	0.49	0.61	0.49	0.579	0.679
Right lens	0.47	0.71	0.54	0.57	0.73
Left optic nerve	0.66	0.47	0.55	0.60	0.70
Right optic nerve	0.64	0.59	0.51	0.62	0.70

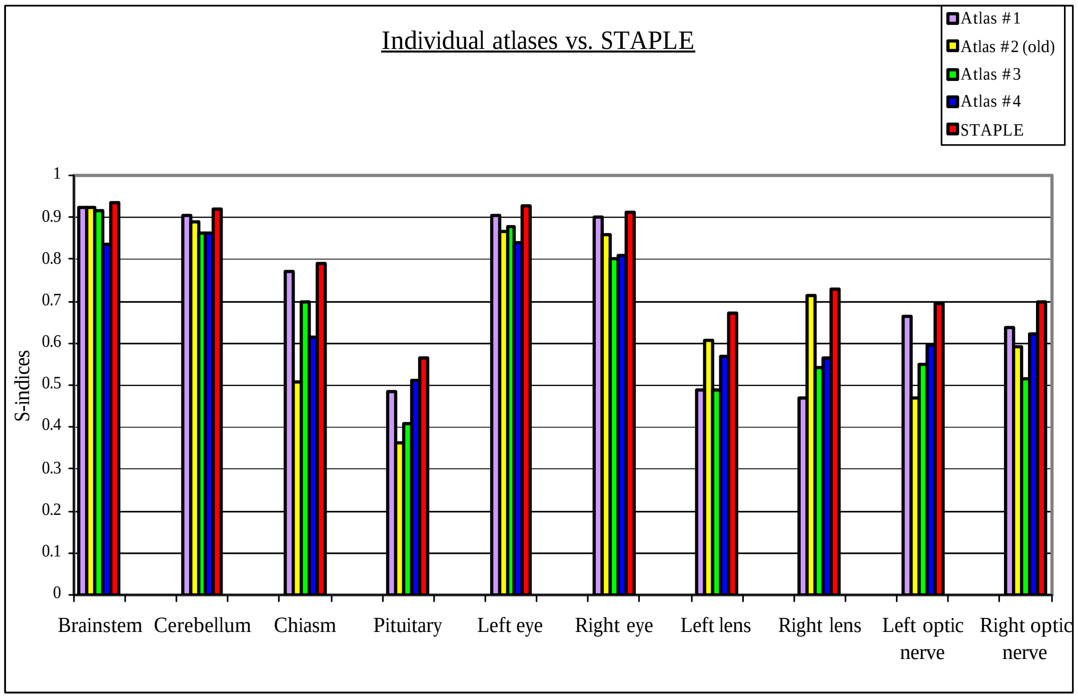


Figure 5.11: Mask-based comparison of original atlas-based segmentation and the STAPLE output.

Contour-based comparison: Here we present the distance measures for the individual atlases as well as the STAPLE output. Table 5.6 gives the average distance measures for all the 20 patients.

Table 5.6: Percent of points within 1 mm. for the atlases, STAPLE and average contours.

	Atlas #1	Atlas #2 (original)	Atlas #3	Atlas #4	STAPLE
Brainstem	78.85	81.58	76.1	68.5	80.7
Cerebellum	47.2	47.3	33.78	50.05	53.5
Chiasm	79.8	45.84	74.1	88.03	76.54
Pituitary	36	34.5	34.66	42.26	39.36
Left eye	59.34	67.29	45	61.78	69.28
Right eye	59.52	67.55	43.56	58.14	67.15
Left lens	47.7	72	60.48	78.71	76.67
Right lens	50.12	78.73	62.77	68.99	71.31
Left optic nerve	66.71	47.69	48.71	71.05	62.4
Right optic nerve	67.2	58.92	53.08	67.15	61.67

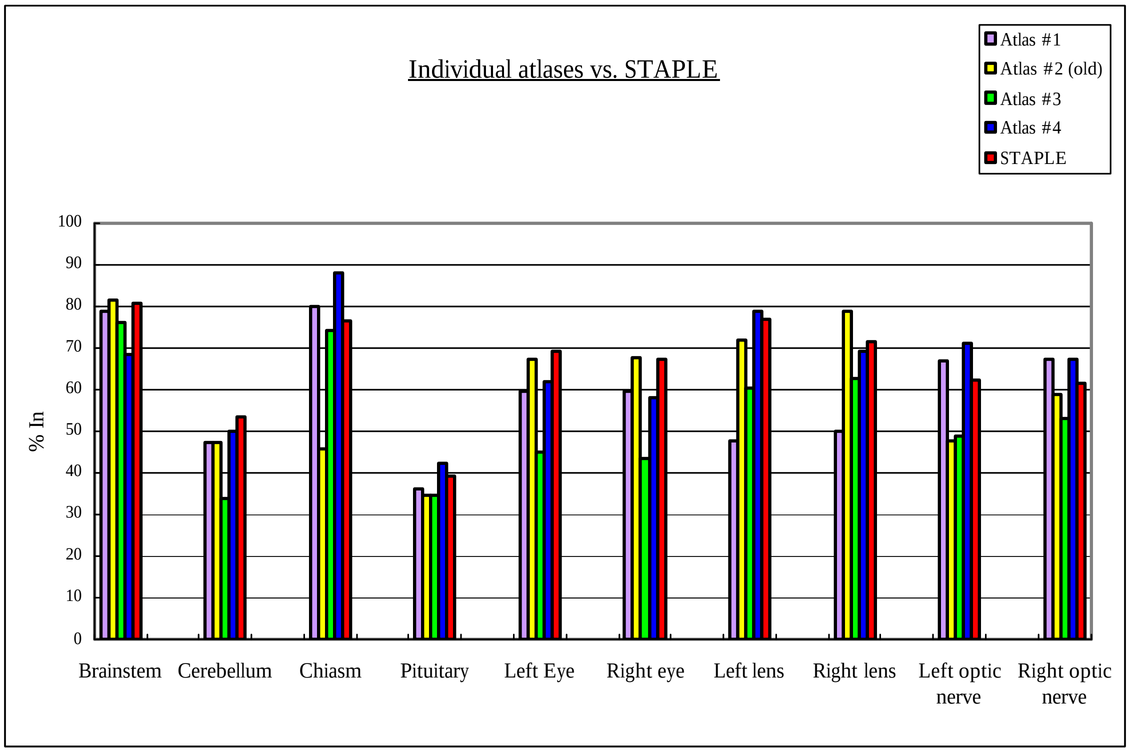
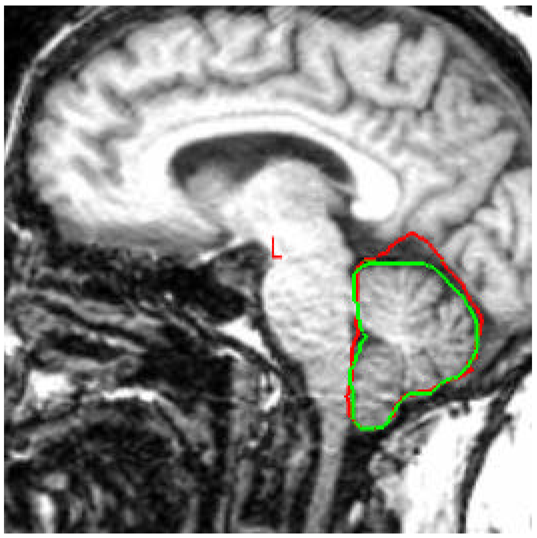


Figure 5.12: Contour-based comparison of original atlas-based results and STAPLE results.

Discussion

It is evident from figures 5.11 and 5.12 that the STAPLE output gives better results than the original atlas-based segmentation (atlas #2). To support our validation results, we also present the actual contours for some structures in this section. The output is not a lot better for the brainstem since it is one of the easiest structures to segment. Thus all the classifiers will generate similar outputs and the combined output will be similar to the original one. However in our case, the fourth atlas generates slightly bad results for the brainstem. But since the STAPLE works probabilistically, it produces an output similar to the three correct and similar segmentations.

It can be seen that the cerebellum shows some improvement. As was discussed previously, leakage in the cortex is a major problem with the cerebellum. To minimize this leakage, we have introduced two atlases which have a similar gap between the cerebellum and the brain cortex. The figure 5.13 shows one such patient where the cerebellum contour improves due to less leakage in the cortex, when using several atlases combined with the STAPLE algorithm.



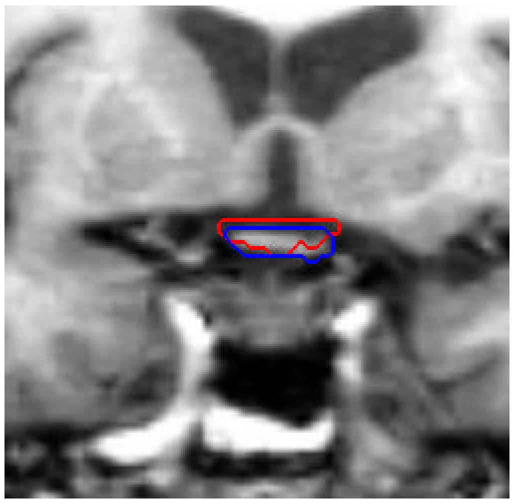
Red = old contour

Green = STAPLE

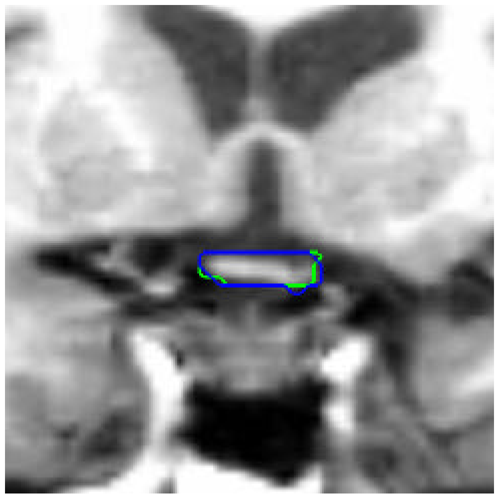
Figure 5.13: Improvement in the cerebellum contour using STAPLE algorithm.

The red contour is the original atlas-based contour and the green one is the STAPLE output. Here one can see that the leakage in the cerebellum is minimized to a great extent. If, however, the original atlas had a gap then the contour would not have leaked in the cortex. To summarize, the combined classifier output will be better even if there are some outlier cases in the individual atlases.

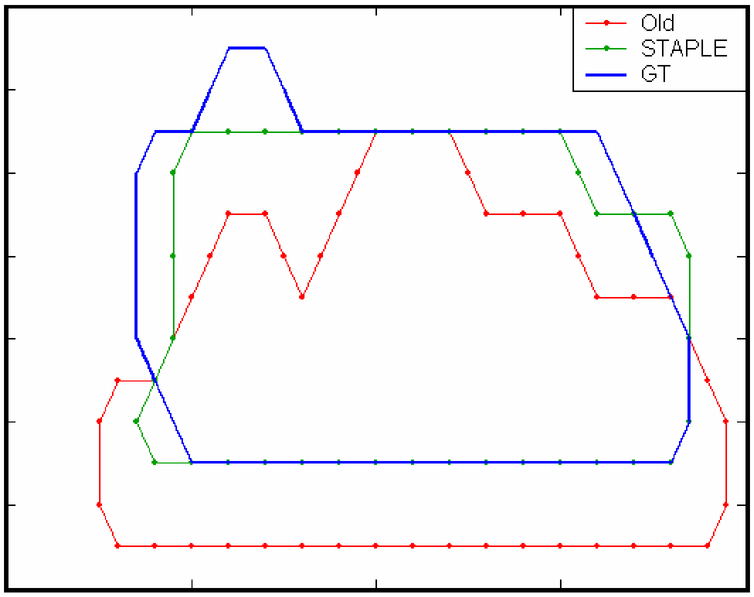
There is a considerable improvement in the segmentation of the optic chiasm and the pituitary gland. This is shown in figure 5.14 below for the optic chiasm.



(a)



(b)



(c)

Figure 5.14: Improvement in the segmentation of the optic chiasm. (a) Original contour (red) (b) STAPLE contour (green) (c) zoomed in view of all the contours. Blue is the manual GT contour.

This is because the chiasm gets segmented accurately by three out of the four atlases and thus the STAPLE algorithm produces an output very similar to these three correct segmentations. It should also be noted that the chiasm had been drawn accurately on all the four atlases by the physicians. However, the accuracy of segmentation largely depends on the quality of registration for a particular atlas. Some get registered very well while some do not. We therefore get different results with different atlases.

For some patients, in the case of the pituitary gland, two out of four atlases produce good segmentations and still the combined segmentation is better. This is because the two segmentations which are bad do not make similar mistakes. In other words, they make independent errors and thus the STAPLE algorithm will detect them as erroneous and generate an output similar to the two correct segmentations. Thus, the overall segmentation is better than the original segmentation. It should however be noted that the pituitary gland is not easy to segment manually and therefore we cannot guarantee that its manual segmentation on the atlas will always be accurate. Figure 5.15 below shows the improvement in the pituitary gland.

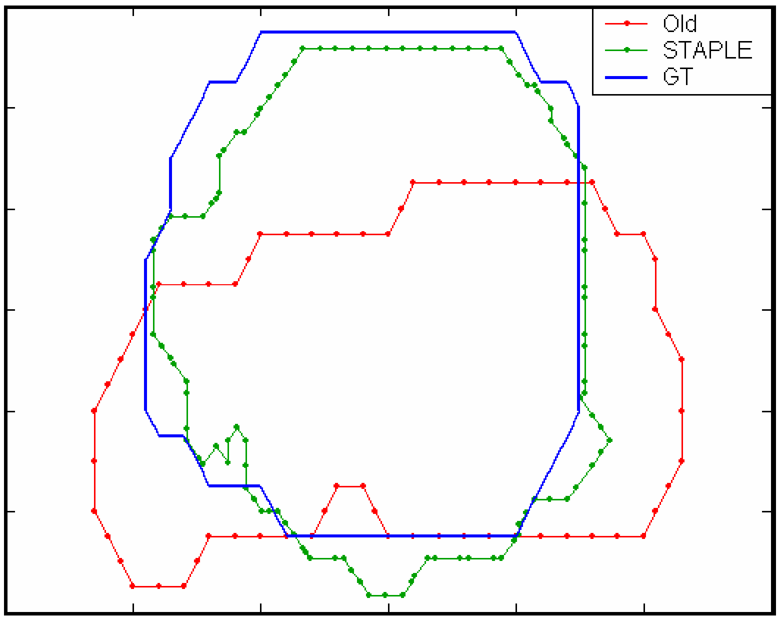


Figure 5.15: Improvement in the pituitary gland segmentation using STAPLE.

The segmentation of both the eyes is improved a little after combining the classifiers. Again all the atlases segment the eyes with more or less the same accuracy. Thus the final output does not show considerable amount of improvement. In the case of the left lens, three atlases produce good segmentations while the fourth one produces bad results. This is because lenses are absent on one of the atlases. The combined output of all these atlases will therefore show improvement.

We have observed contradictory results in the case of the right lens. This is because, for some patients, three out of four atlases have produced bad results for the right lens. For such patients, the only atlas to generate better results was the original atlas. The following figure supports this statement.

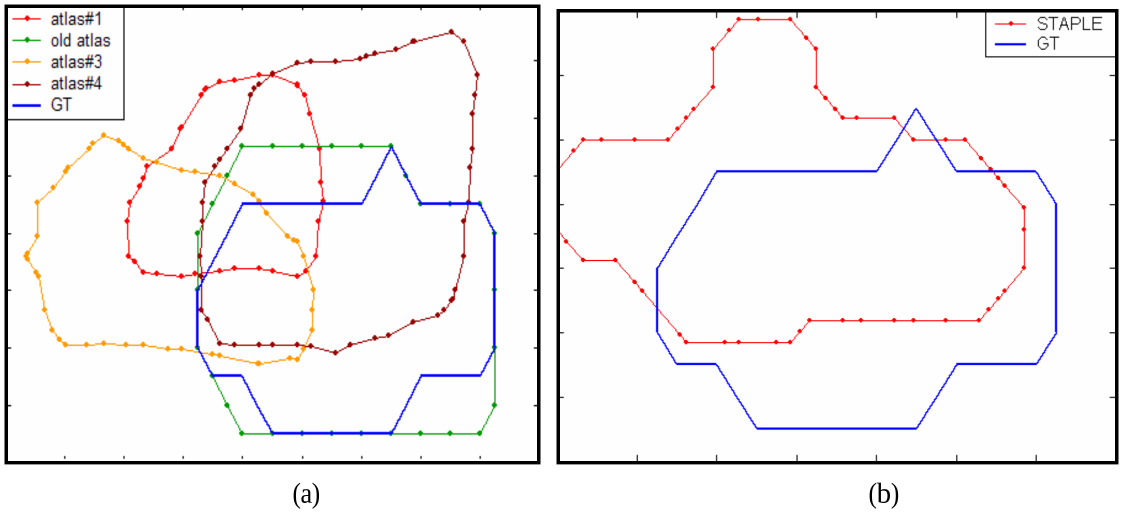


Figure 5.16: Degradation in the segmentation of the right lens. (a) Contours obtained from four atlases. (b) STAPLE contour.

In figure 5.16(a), the green contour is the original contour and in figure 5.16(b) the red contour is the STAPLE output. It is worse than the original contour due to bad segmentations by the other three atlases. This supports our statement that for STAPLE to generate good results the majority of the classifiers should generate good segmentations.

Considerable improvement is also achieved in the case of both the optic nerves. In the case of the optic nerves, the first and the fourth atlases generated very good results for most of the patients. But, the second atlas (original atlas) and the third atlas produced bad results. Hence the final STAPLE output is better than the two bad ones but worse than the two good ones. STAPLE results deteriorate in such a case if multiple atlases commit the same mistakes. However, even if two of the four atlases segment the structure correctly and the other two make independent and different errors, STAPLE can produce a better output. This observation also supports our statements made earlier that STAPLE will fail if the errors made by multiple classifiers are not independent. Figure 5.17 shows the results for one such optic nerve.

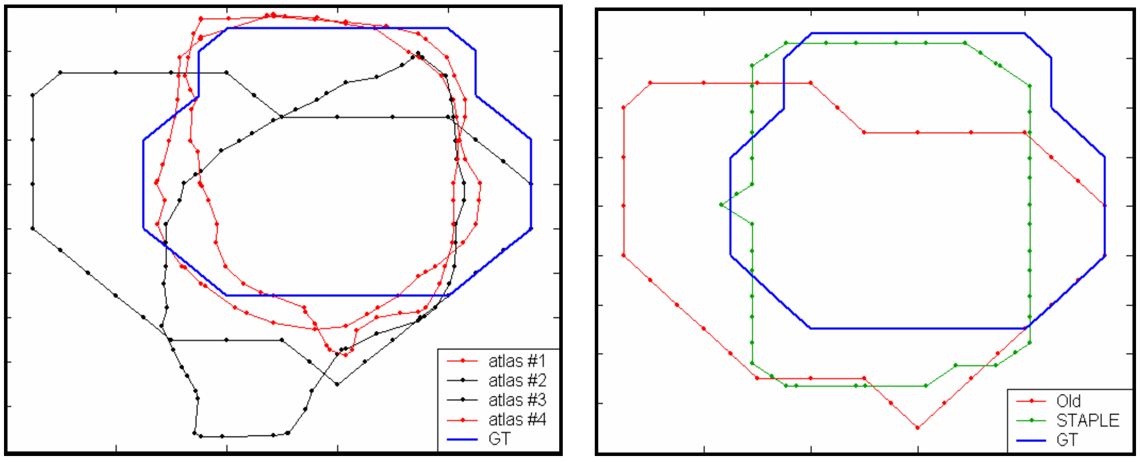


Figure 517: Improvement in the segmentation of the optic nerves. (a) Segmentations to be combined. (b) STAPLE contour (green) ground truth contour (blue).

It should also be noted that the results for smaller structures like the chiasm, pituitary, and the optic nerves are not as good as those for bigger structures like the brainstem, cerebellum and the eyes. This is because the smaller structures are harder to segment manually than the bigger ones, which also affect the segmentation results. Also, the S-indices for smaller structures are small. This is because the S-indices strongly depend on the object size [22]. If 5 voxels have been misclassified for a 50 voxel large structure, the S-index will be 0.95, whereas if the object size is 10, the S-index will be 0.8. Thus for small structures, the contour-based validation method may be a better approach.

It can be concluded from all the above experiments that the classifier combination method gives the best results for atlas-based segmentation.

Experiment 5

This experiment was performed to compare the individual manual segmentations, drawn by the three raters, with the manual ground truth segmentation. The statistics are then compared with those obtained with the classifier combination method. This has been done for all the ten structures and for all the 20 patients. The following sections give mask-based and contour-based comparisons of the manual and automatic segmentations with the manual ground truth.

Mask-based comparison: Here we present the average S-indices obtained by comparing the individual manual segmentations with the manual ground truth segmentation. These results are then compared with the S-indices obtained by comparing the STAPLE automatic segmentation with the manual ground truth segmentation. Figure 5.18 shows a comparison with the help of a bar graph.

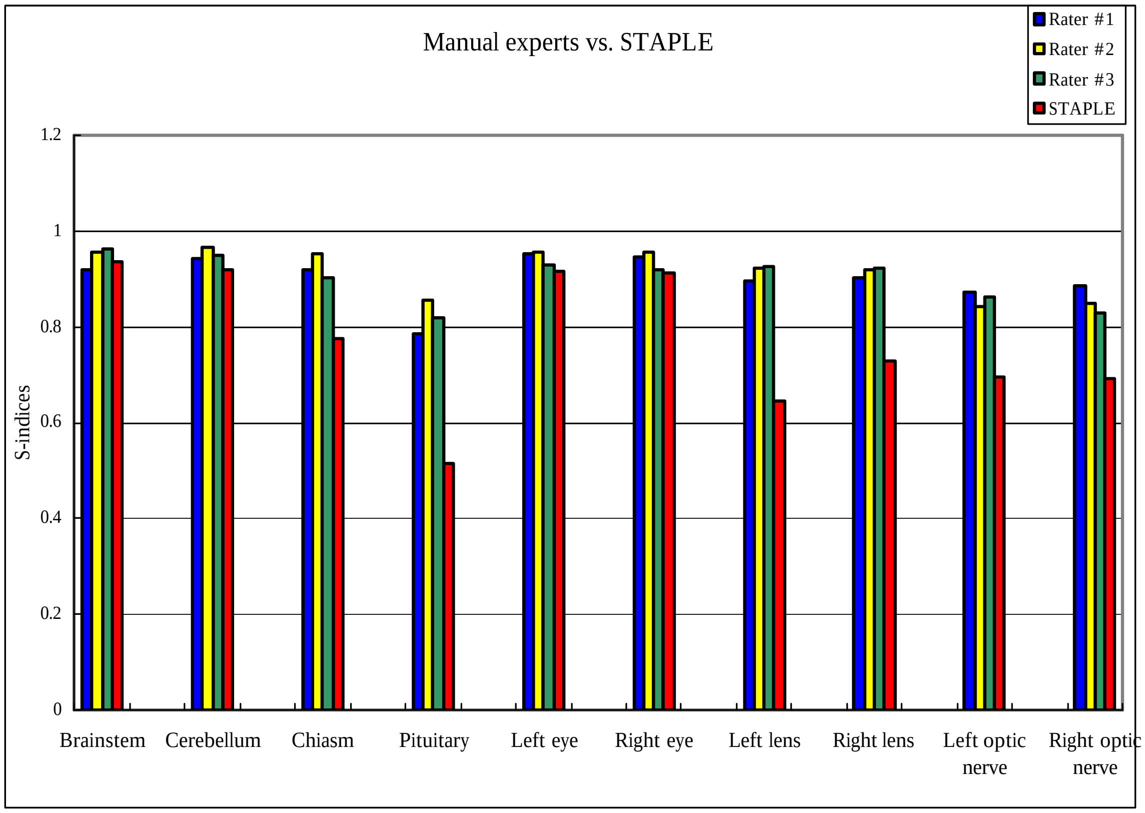


Figure 5.18: Mask-based comparison between the manual masks and the automatic masks obtained by the STAPLE algorithm.

<u>Contour-based comparison:</u> This section gives the average distance measures for the manual and automatic contours. Figure 5.19 shows the comparison with the help of a bar graph. The graph shows the percent of automatic and manual contour points that lie within a distance of 1 mm. from the ground truth contour.

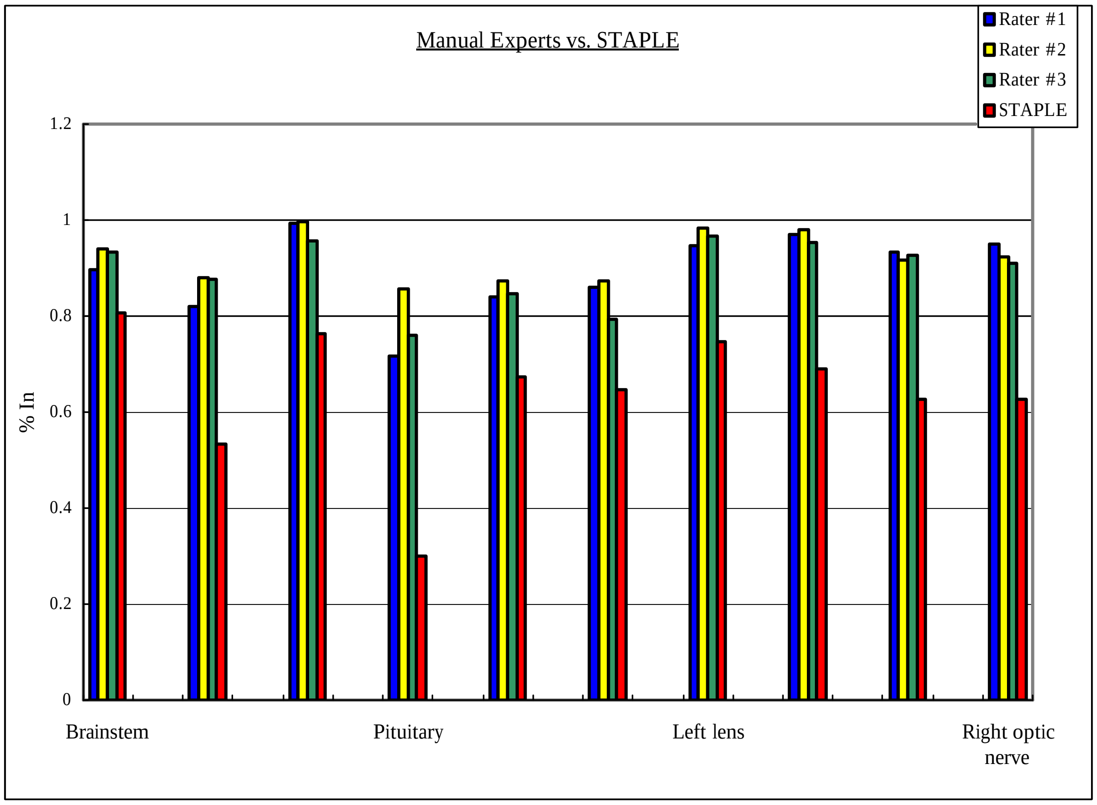


Figure 5.19: Contour-based comparison between manual and automatic contours.

Discussion

As can be seen from figures 5.18 and 5.19, the three individual manual segmentations are more or less similar to the ground truth for most of the structures. They are however different from each other in the case of the pituitary gland. This shows that the pituitary gland is difficult to segment manually. It also becomes difficult to segment the pituitary gland automatically because of its poor edge definition in MR images. The pituitary only gives an accuracy of 52% for the masks and 31% in the case of contours. This problem is alleviated when CT-MR fusion is used, in which case it improves the accuracy by 10%.

The optic nerve is another challenging structure for the intensity-based registration method that is used in this study. This is because, on a portion of its length, it appears as a dark structure surrounded by white pixels and on the remaining portions of its length, it is essentially invisible [15]. The good contrast on a fraction of its length ensures good registration whereas the other slices show poor registration of the optic nerve. This can be explained with the help of the following figure.

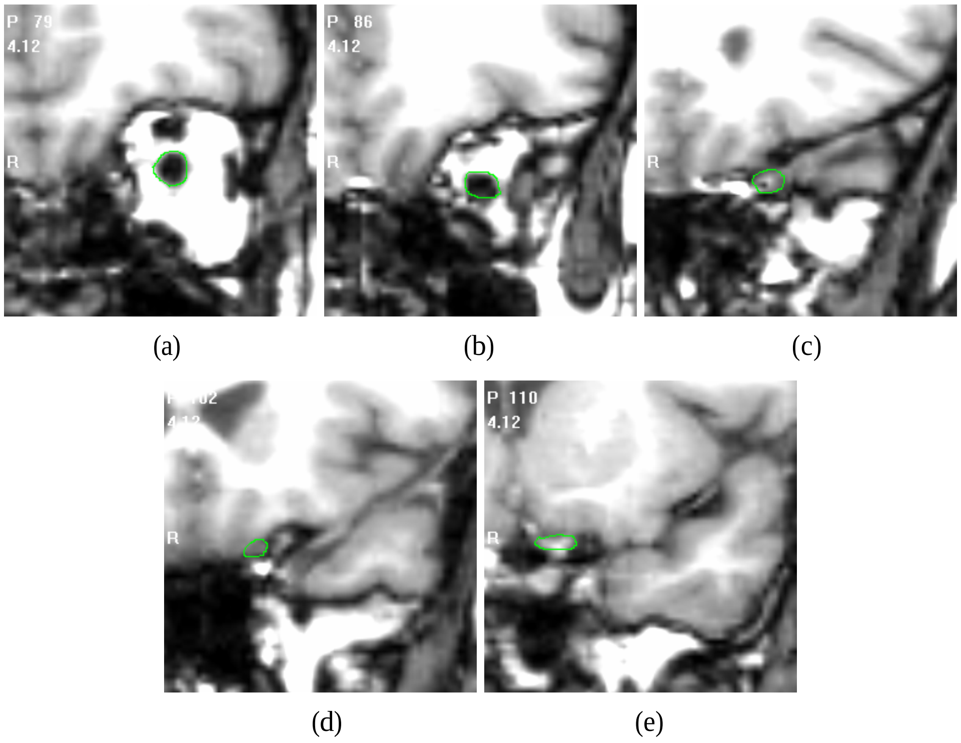


Figure 5.20: Left optic nerve on a coronal MR volume.

As can be seen, the optic nerve is surrounded by white tissues in figures 5.20 (a) and (b). In figures 5.20 (c), (d) and (e) the optic nerve is hardly visible. The expert will draw the optic nerve on such slices based on a priori anatomic information and by mentally interpolating the position of the nerve based on its position on the slices where it is visible [15]. This explains why we only get an accuracy of 70% for the masks and 62% for the contours.

Experiment 6

In this experiment, radiotherapy planning was carried out using the automatic contours. These contours were first loaded into the radiotherapy planning software, "Eclipse". Then a tumor volume was located and contoured on the patient image volume. This is called as the 'Gross Tumor Volume' (GTV). Then a margin of 0.5 cm was included in the GTV along with the areas of likely spread of the cancer cells. This is called as the 'Planning Tumor Volume' (PTV). Once this is done, the computer software determines the optimum arrangement of the radiation beams so that maximum radiations are received by the tumor cells and minimum by the surrounding healthy body tissues. To determine the validity of this automatic plan we have also manually contoured the surrounding brain structures. A radiation oncologist then calculates the amount of dose received by the manually delineated contours. These dose values (in cGy) are then compared with the actual constraints available for these structures. If the dose received by the structures is in tolerable limits, we conclude that the automatic plan can be an option for radiotherapy planning.

Table 5.7 shows the values of the doses received by the brain structures when the automatic contours were used for radiotherapy planning. The prescribed dose for the GTV is 5400 cGy. The table represents the maximum dose a % volume of the structure can receive. It also shows the actual amount of dose that the structure received.

Table 5.7: Doses received by manual contours using automatic plan

Structures	Maximum Dose (cGy)	% Volume	Actual Dose (cGy)	% Volume
GTV	5400	100	5400	100
PTV	5130 (95 %)	100	5130	99.68
Left eye	3000	0	2530	0
Right eye	3000	0	2320	0
Left lens	650	0	650	0
Right lens	650	0	640	0
Brainstem	4500	0	1070	0
Cerebellum	4500	0	550	0
Left optic nerve	4500	0	4000	0
Right optic nerve	4500	0	4490	0
Chiasm	4500	0	2500	0
Remaining brain	5000	0	1070	0
Remaining brain	3500	10	600	10
Remaining brain	1500	20	430	20

As can be seen, the amount of dose received by all the structures is within the maximum allowable limit. Thus, we can conclude that automatic planning can be an option especially when the tumor volume is not located close to the critical structures like the optic chiasm and the optic nerves. In the above case, the tumor was located between the upper parts of the eyes.

Summary

This chapter presented the results obtained by the three new algorithms implemented to improve the accuracy of atlas-based segmentations. The results obtained by the new methods were compared to the original method using tables and bar graphs. It was concluded that the multi-classifier approach using the STAPLE algorithm gave the best results compared to the other two algorithms. The conclusions and future work are discussed in detail in the next chapter.

CHAPTER VI

CONCLUSION AND FUTURE WORK

Radiation therapy planning requires accurate delineation of the brain structures to be irradiated as well those to be spared. Currently, this task is done manually which is extremely time consuming and is affected by inter-rater and intra-rater variability. In this thesis, we use an automatic method for segmentation of the brain structures to assist in radiation therapy planning. The atlas-based segmentation method was studied and various methods for improving the results of this technique were proposed. An end user segmentation tool was developed which could be easily loaded in the Radiation Oncology department. This tool executes the automatic segmentation method till the creation of DICOM-RT files.

Various methods were suggested and implemented for the improvement of the results generated by the original atlas-based method. The results obtained by the various methods were compared using two different validation techniques. In these validation methods, the automatic contour or mask was compared to the manual segmentation to determine how well the structure was segmented. The greater the value of these validation metrics, the better is the segmentation.

The first method investigated the advantages and disadvantages of fusing CT and MR image volumes. This method gave better results for the structures which get well defined on the fused image due to the additional bone information from the CT image. In this study, it was found that the cerebellum and the pituitary gland contours show slight improvement after fusion. We can go a step further and create a fused image which will have the internal brain structures from MRI and the part of the brain outside the skull taken from the CT. This may show improvement for the optic nerves and eyes which are clearly visible on the CT image.

The second method was implemented to study the effects of mesh deformation on the results of atlas-based segmentation. In the original method, we were deforming the binary masks defined on the atlas image by the application of the deformation field in the reverse direction (target-to-atlas). In this method, we deform the atlas-defined meshes by applying the deformation field in the forward (atlas-to-target) direction. It was concluded that the mesh deformations do not give any improvement but deteriorate the results slightly. A possible cause is the interpolation of the deformation field that takes place because the mesh points do not fall on a regular grid. These interpolation effects have been mitigated in the original method by applying an intensity threshold. Mesh deformations can be an interesting subject for future investigation. By considering the properties of the mesh, we can try to control the deformation of the mesh points. Also, smoother meshes could be obtained by getting Radial Basis Function (RBF) approximations to the mesh points. One could also try to get regular meshes by smoothing the deformation field itself.

In the third method, various atlases were used to represent different classifier systems for the image voxels. The outputs of these classifiers were combined to obtain better segmentations. This result is based on the popular theory in pattern classification that the combination of classifiers gives more accurate results than the individual classifiers. Each atlas with the registration method represents a unique classifier. We have combined four individual classifier systems using the STAPLE algorithm [18], [19]. It was concluded that for the majority of the structures, the STAPLE algorithm produces a better output than the original classifier. It was also observed that for STAPLE to produce good results, the individual classifiers have to be different and independent from each other. The more independent and the more different the classifiers are; the more likely the STAPLE will give better results than any other combination strategy. The following table shows the percent improvement achieved for each of the structures as compared to the original atlas-based segmentation method. It can be seen that there is no significant improvement for the brainstem and the eyes whereas for the cerebellum, chiasm, pituitary and the optic nerves significant improvement has been achieved by using the STAPLE algorithm.

Table 6.1: Percent improvement achieved by the STAPLE algorithm

Structures	% Improvement	
	Masks	Contours
Brainstem	2	0
Cerebellum	**3**	**5**
Chiasm	**28**	**30**
Pituitary	**17**	**6**
Left eye	5	-1
Right eye	5	-1
Left lens	**7**	**6**
Right lens	**1**	**-9**
Left optic nerve	**23**	**15**
Right optic nerve	**11**	**5**

Thus, it can be concluded that the results substantially improve if classifier combination is selected as the method of choice.

Classifier combination is however not always the best way to achieve segmentation. Classifier selection may actually be better in some cases. This is because; sometimes the combined output may not be better than the best individual classifier. It may therefore be better if we choose a classifier which gives the best result for that particular structure.

In this work, we were able to segment the brainstem, cerebellum, and eyes with an accuracy of 90% or more. The chiasm and the lenses, being smaller structures, gave an accuracy of around 75%. The segmentation of the optic nerves is however a difficult task and gave an accuracy of only 62%. In all the methods above, the optic nerves never gave excellent results. One of the possible future works can be to try and segment the optic nerves by using deformable models proposed in [10] and [11] because they make use of the priori shape information of the object to be segmented. This may produce better results for the optic nerves. The pituitary gland also never gave good results because of its poor edge definition in MR images. We could achieve an accuracy of only 40% for the pituitary gland. This problem was partially solved by using CT-MR fusion, in which case the pituitary gland showed an improvement of 10%.

In conclusion, even though the automatically generated contours do not produce accurate results on all the slices, it may produce acceptable contours on the majority of the slices. A trained physician can then edit the remaining inaccurate contours. This will likely reduce the interaction time required for delineation and thus help in IMRT planning. Also, automatic contours for 7 patients have been loaded in the Radiation Oncology department at Vanderbilt University and plans have been carried out on each one of them. Majority of these plans proved that automatic planning can be a feasible option if the tumor is not located very close to the critical structures.

REFERENCES

[1] J. M. Fitzpatrick, D. L. G. Hill, and C. R. Maurer, 'Image Registration', Volume II of the "Handbook of Medical Imaging", M. Sonka and J. M. Fitzpatrick, eds., SPIE Press, pp. 447-513, 2000.

[2] B. M. Dawant, A. P. Zijdenbos, 'Image Segmentation', Volume I of the "Handbook of Medical Imaging", M. Sonka and J. M. Fitzpatrick, eds., SPIE Press, pp. 71-120, 2000.

[3] C. Xu, D. L. Pham, J. L. Prince, 'Image Segmentation using deformable models', Volume I of the "Handbook of Medical Imaging", M. Sonka and J. M. Fitzpatrick, eds., SPIE Press, pp. 129-168, 2000.

[4] Radiation Therapy information from, www.radiologyinfo.org

[5] R. C. Gonzalez, R. E. Woods, "Digital Image Processing", 2nd edition, Prentice Hall, 2001.

[6] Canny edge detector from, http://ct.radiology.uiowa.edu/~jiangm/courses/dip/html/node93.html

[7] Canny edge detector from, http://64.233.179.104/search?q=cache:FE2hf6ST4N8J:www.s2.chalmers.se/undergraduate/courses/ess060/PDFdocuments/ForPrinter/Notes/CannyEdgeDetector.pdf+Canny+edge+detector&hl=en

[8] Canny edge detector from, http://www.pages.drexel.edu/~weg22/can_tut.html

[9] Radiation Oncology Department, The Vanderbilt Clinic.

[10] M. Kass, A. Witkin, D. Terzopoulos, "Snakes: Active contour models", International Journal of Computer Vision, vol. 1, no. 4, pp. 14-26, 1990.

[11] T. F. Cootes, C. J. Taylor, D. H. Cooper, J. Graham, "Active Shape Models-Their training and application", Computer Vision and Image Understanding, vol. 61, no. 1, pp. 38-59, 1995.

[12] G. K. Rohde, A. Aldroubi, B. M. Dawant, "The adaptive bases algorithm for non-rigid image registration.", IEEE Trans. Medical Imaging, vol. 22, no.11, pp. 1470-1479, 2003.

[13] G. K. Rohde, "The Adaptive Grid Registration Algorithm: A new spline modeling approach for automatic Intensity Based Nonrigid Registration.", Masters Thesis, Vanderbilt University, August 2001.

[14] Maes F., Collignon A., Vanderneulen D., Marchal G., Suetens P., "Multimodality image registration by maximization of mutual information.", IEEE Transactions on Medical Imaging, vol. 16, no. 2, pp. 187-198, 1997.

[15] PF D'Haese, V. Duay, Rui Li, A. du Bois d'Aische, A. Cmelak, E. Donnelly, K. Niermann, T. E. Merchant, B. Macq, B. M. Dawant, "Automatic segmentation of brain structures for radiation therapy planning", Medical Imaging Conference, SPIE 2003.

[16] PF D'Haese, V. Duay, T. E. Merchant, B. Macq, B. M. Dawant, "Atlas-based segmentation of the brain for 3-dimensional treatment planning in children with infratentorial ependymoma", MICCAI (2) 2003, pp. 627-634.

[17] R. O. Duda, P. E. Hart, D. G. Stork, "Pattern Classification", 2nd ed., pp. 84-126, 2001.

[18] S. K. Warfield, K. H. Zou, W. M. Wells, "Validation of image segmentation and expert quality with an Expectation-Maximization algorithm", MICCAI 2002, pp. 298-306, 2002.

[19] S. K. Warfield, K. H. Zou, W. M. Wells, "Simultaneous Truth and Performance Level Estimation (STAPLE): An algorithm for the validation of image segmentation", IEEE Transactions on Medical Imaging, vol. 23, no. 7, pp. 903-921, 2004.

[20] T.Rohlfing, D. B. Russakoff, C. Maurer, Jr., "An expectation maximization-like algorithm for multi-atlas multi-label segmentation", in Information Processing in Medical Imaging, Berlin Heidelberg, 2003, vol. 2732 of Lecture Notes in Computer Science, pp. 210-221, Springer-Verlag.

[21] T.Rohlfing, D. B. Russakoff, C. Maurer, Jr., "Performance-based multi-classifier decision fusion for atlas-based segmentation of biomedical images", in Proceedings of Second International Workshop on Biomedical Imaging, (Los Alamitos, CA), pp.255-260, IEEE press 2004.

[22] T. Rohlfing, R. Brandt, R. Menzel, C. R. Maurer, Jr., "Evaluation of atlas selection strategies for atlas-based image segmentation with application to confocal microscopy images of bee brains", NeuroImage, vol. 21, pp. 1428-1442, Apr. 2002.

[23] T.Rohlfing, D. B. Russakoff, C. Maurer, Jr., "Expectation Maximization strategies for multi-atlas multi-label segmentation", in Information Processing in Medical Imaging, vol. 2732 of Lecture notes in Computer Science, (Berlin Heidelberg), pp. 210-221, Springer-Verlag, July 2003.

[24] T.Rohlfing, C. Maurer, Jr., "Multi-Classifier framework for atlas-based image segmentation", in Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington D.C., USA, June 27-July 2, 2004, pp. 255-260, IEEE Press, 2004.

[25] T.Rohlfing, D. B. Russakoff, C. Maurer, Jr., "Performance-based classifier combination in atlas-based image segmentation using Expectation-Maximization parameter estimation", IEEE Transactions on Medical Imaging, vol. 23, pp. 983-994, 2004.

[26] J. Kittler, M. Hatef, R. Duin, J. Matas, "On Combining classifiers", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 3, pp. 226-239, Mar. 1998.

[27] L. Xu, A. Krzyzak, C. Y. Suen, “Methods of combining multiple classifiers and their applications to handwritten recognition”, IEEE Trans. Systems, Man, and Cybernetics, vol. 22, no. 3, pp. 418-435, 1992.

[28] Lee R. Dice, “Measures of the amount of ecologic association between species”, Ecology, vol. 26, no. 3, pp. 297-302, 1945.

[29] The Visualization ToolKit (VTK), www.vtk.org

[30] Medical Studio, www.medicalstudio.org

[31] The DICOM ToolKit (DCMTK) www.dcmtk.org

[32] T. K. Moon, “The Expectation-Maximization Algorithm”, IEEE Signal Processing Magazine, pp. 47-60, November 1996.

[33] R. Li, “Automatic placement of regions of interest in medical images using image registration”, Master’s thesis, Vanderbilt University, 2001.

[34] N. Otsu, "A threshold selection method from gray-level histograms", IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, no.1, pp. 62-66, 1979.