A Method for Identifying Geometrically Simple Surfaces from Three Dimensional Images

J. David MacDonald

School of Computer Science,
McGill University, Montreal

November, 1997

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements of the degree of Doctor of Philosophy.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank the numerous people who contributed to this thesis in their various ways. Firstly, I would like to thank my two supervisors. Dr. Alan Evans deserves much credit for providing a stimulating research environment, and for supporting me in many ways throughout the course of this dissertation. The guidance and enthusiasm of Dr. David Avis has been gratefully appreciated, as well as his patience and understanding. Both supervisors deserve thanks for their perseverance in this undertaking. Their encouragement has been a significant factor in the fulfillment of this research.

I would like to thank all of the members of the McConnell Brain Imaging Centre. They are a great bunch to work with, and their assistance, in the form of encouragement, software, and scientific advice, as well as friendship, is irreplaceable. In particular, I would like to thank Dr. Louis Collins, Rick Hoge, Vasco Kollokian, Peter Neelin, John Sled, Greg Ward, Mark Wolforth, and Dr. Alex Zijdenbos for providing a large base of software, as well as data and help with many aspects of this research. Dr. Colin Holmes and Dr. Noor Kabani deserve thanks for helping with the neuroanatomical side of things, providing much needed data and expertise. I would like to thank Michel Audette for his thorough proofreading of this text and his many valuable suggestions. There are many others who have helped me in too many ways to mention. I can only say: thank you all.

I would like to thank those who served as members of my Ph.D. proposal committee and Ph.D. defense committee. Their time and energy is gratefully acknowledged.

The assistance of the faculty and staff of the School of Computer Science throughout the years deserves much appreciation.

Of course, none of this work could have been done without funding. I would like to express appreciation for funding of various aspects of this work from the Natural Sciences and Engineering Research Council of Canada, the McDonnell-Pew Program in Cognitive Neurosciences, the Medical Research Council of Canada, the Human Brain Map Project, NIMH and NIDA.

Finally, I would like to thank my wife, Joanne, who has done everything imaginable to help me, except write it herself.

Abstract

Computational neuroanatomy is an exciting new area where digital tools are used with great advantage in the analysis of structure and function of the brain. One major area of research in this field involves automatically creating computerized representations of the brain which are in a form suitable for neuroanatomic analysis. The principle drawback of contemporary methods of generating these digital models is the incompleteness and ambiguity of the input data, which is typically under-sampled relative to the features of interest. A method is presented here for creating surface-based models of neuroanatomy that address the data incompleteness issue with an integrated combination of two model-based approaches. The first approach involves applying proximity and self-intersection restrictions on surfaces in order to create a plausible neuroanatomical model in the face of data with topologies inconsistent with medical knowledge. The second approach involves identifying multiple surfaces simultaneously, with inter-surface constraints, in order to use general neuroanatomical information to correct areas where the data is incomplete or ambiguous. The overall method is one of deforming a set of polyhedral meshes, with the above constraints and others incorporated into an objective function, which is minimized to find the best fit of a model to the data. Validation of this method on simple phantoms as well as in the task of segmentation of human cortical surfaces is demonstrated. Discussion of limitations and future work is presented.

Resumé

Le neuroanatomie informatique est une nouvelle sphère d'activité où des outils informatiques sont employés avantageusement dans l'analyse de la structure et du fonctionnement du cerveau. Un domaine de recherche important de cette discipline se consacre à la création automatique de représentations numeriques du cerveau sous une forme qui couvient à l'analyse neuroanatomique. La lacune majeure des méthodes contemporaines pour génerer de tels modèles numeriques est la nature incomplète et ambiguë des données, qui sont typiquement sous-échantillonnées par rapport aux caractéristiques pertinentes. Une méthode neuroanatomique, servant à créer des modèles basés sur des surfaces, et s'addressant à ce probleme d'inachèvement des données en intégrant deux techniques basés sur des modèles, est présentée ici. La première technique implique l'application de contraintes de proximité et d'auto-intersection sur les surfaces en vue de créer un modèle neuroanatomique plausible lorsque confronté à des données dont la topologie est incompatible avec le savoir médical. La deuxième technique implique la création simultanée des surfaces multiples, régies par contraintes inter-surfaces, dans le but d'exploiter de l'information neuroanatomique génerale pour corriger les régions où les données sont incomplètes ou ambiguës. Globalement la méthode consiste à déformer un ensemble de maillages polyhédriques, où les contraintes énumérées ci-haut et autres sont intégrées dans une fonction objective, qui est minimisée en vue d'obtenir la meilleure approximation des données par les modèles. La validation de cette méthode sur de simples mannequins ("phantoms"), ainsi que la segmentation de surfaces corticales humaines sont démonstrées. Une

discussion des limites de la méthode et du travail à venir est également présentée.

Original Contributions

The work described in this dissertation is primarily that of the author, in consultation with his two co-supervisors. The author formulated a unique surface deformation method first described in [MAE93], and later refined in [MAE94]. A major contribution of this work has been the ability to identify anatomical features that are not explicitly visible in an image, using multiple image boundaries to estimate the position of anatomical boundaries. A second major contribution is the use of self-intersection constraints to enforce topological constraints on identified objects. The combination of these two features into a surface deformation framework is demonstrated to improve the identification of cerebral cortical surface in MR images.

The surface deformation software was designed and implemented by the author. In addition, there were many software tools involved in the fulfillment of this dissertation. The tools *MNI-Display*, *MNI-register*, and *MNI-ray-trace* were created by the author for display and manipulation of volumes and surfaces. In addition, dozens of utility programs were written by the author for performing various operations on volumes, surfaces, and curves. The image registration software, tissue classification software, RF non-uniformity correction software, MRI simulator software, and MR volume libraries were provided by other members of the McConnell Brain Imaging Centre.

Chapter 1

Introduction

Since the advent of digital processing methods, there has been an increasing number of techniques for digital acquisition of measurements of physical objects, based on such modalities as visible and invisible light, heat, magnetic fields, radio waves, and many other methods covering almost the entire energy spectrum. The vast quantity of information generated often precludes the use of all but automated methods for processing this data. In general, the raw data from the various sensors must be transformed into more convenient representations and structured in ways that facilitate advanced digital processing and analysis. This general problem of information transformation and representation manifests itself in many different applications, each with its own specific variations. One particular case is a large set of applications which requires the transformation of three dimensional density information into structural representations of three dimensional objects.

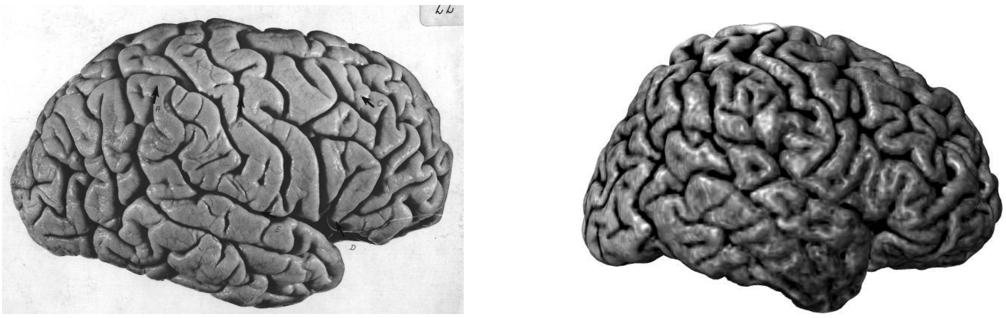
Within the domain of medical imaging, **magnetic resonance imaging**, known as **MRI** or **MR imaging**, is a prominent method of acquiring structural information about organisms in a non-destructive way. However, the raw data acquired constitutes a very low level representation of the information, with various sources of errors in the signal. In order to perform advanced processing, the MR image must be transformed into some digital representation that can be related to the wealth of anatomical information that is available from other sources, notably textbooks and experienced anatomists.

This document constitutes a doctoral dissertation in the field of computer science, with specific application to the domain of computational neuroscience. The problem being addressed is that of automatically generating digital models of neuroanatomical structures from three dimensional images such as MRI. One of the outstanding problems in this large research area is the question of how to respond to the non-optimality of the input data, which is noisy, under-sampled, and incomplete. Some contemporary methods of creating digital models are predominantly data driven, which is insufficient for automatic use due to the inaccuracy of the input data. Other more successful methods attempt to use model-based constraints to fill in the information missing in the data, but the facilities provided for incorporating model information are rather limited. A new method is presented here which is shown to provide improved neuroanatomical modeling of medical images, principally by integration of two model-based approaches.

The first approach addresses topological errors due to noise, under-sampling, and other imaging artifacts. Due to these factors, the image data from magnetic resonance imaging typically does not have a topology consistent with the knowledge gleaned from studies of actual human brains. As a result, purely data driven methods, as well as many model-based approaches, can be confounded by the incorrect topology in the data. Furthermore, most contemporary model-based methods are susceptible to producing models which are not correct relative to medical knowledge, in particular, objects which intersect themselves or other objects. In response to these limitations, the first approach presented here is that of intersection avoidance in the process of creating digital models. The resulting models are guaranteed not to intersect themselves or each other, making them more consistent with the real world objects which they are meant to represent. Although intersection avoidance is explored here within the context of neuroanatomical modeling, the idea can be generally applied to a broad array of image recognition tasks. It is shown here that the addition of such constraints to the construction of digital models can improve the correctness of the resulting solution, without adding a prohibitive computational cost.

The second approach addresses an aspect of imaging errors in the data which is particularly problematic in the domain of neuroanatomy. The human brain contains large surfaces which are very tightly folded, resulting in highly convoluted areas where under-sampling and noise degrade or remove the appearance of boundaries between two touching surfaces. The approach chosen to reduce the effects of such edge degradation uses anatomical knowledge about the relative position of different types of tissues in the human brain. The identification of one surface boundary is improved by constraining it with the position of another boundary and *a priori* knowledge of the relationship between the two. A general method of creating multiple component models with inter-component constraints is shown to improve the identification of cortical surfaces in the face of sub-optimal data. Again, this idea can be carried over into other domains, where sets of inter-connected objects provide a better model of the data than single objects.

The following chapter details the problem being addressed and provides a computational neuroanatomical context for dealing with the problem and its particular challenges. Relevant techniques from the literature are presented with a discussion of the advantages and disadvantages of each as it relates to the problem domain. A separate short chapter is devoted to a survey of general edge detection methods, concluding with a description of established feature-based tissue classification algorithms which can be successfully used to provide edge detection in medical images. After laying this groundwork, the proposed solution is presented. Description of the object representation, the objective function with its various components, as well as the method of minimizing to find a solution is presented in detail. A separate chapter investigates the effect of each of the objective function constraints on very simple phantom data. The following chapter discusses relevant implementation details of the surface deformation method. The discussion of the validation of this method is divided into two chapters. The first of these chapters presents results of tests constructed on simulated data and small phantoms, so that the correct answer is in some way known or assumed, with the result that quantifiable error estimates can be found. The second validation chapter demonstrates results on real data where the true answer is not readily available, but validation consists of showing that the method produces results qualitatively and quantitatively consistent with other neuroanatomical analysis. Figure 1.1 shows a photograph of a human brain and a computer generated image of a brain surface created by the algorithm described in this dissertation. The final chapter summarizes the results of this work, the weaknesses of the method, and presents some ideas for further work.



a) b)

Figure 1.1: a) Photo of human brain. b) Computer generated model.

Chapter 2

Problem Definition

2.1 Motivation

Science can be defined as the process of measuring the world in which we live, followed by attempts at discerning higher level meaning from this data. Methods of measurement have progressed far beyond the simple but useful yardstick to provide much more detailed and accurate data. Lately, digital information processing technology has enabled the sampling of large amounts of real world data, through such methods as photo-digitization, sonar, radar, and a host of other sensing modalities. The growing number of these types of datasets, combined with the increasing amount of information present in a given instance, points to the necessity for automated techniques for processing this data.

The type of processing which is required is quite naturally linked to the particular application domain. However, in general, the processing step is essentially a task of interpretation. Mimicking the techniques of centuries of scientific investigations, the computer must combine and correlate the data measurements to produce a model of the underlying process. This model not only represents a high-level form of the measured data, but also constitutes a more powerful mechanism for dealing with the associated real world phenomenon.

One example of where this computerized interpretation can be advantageously applied is in the area of neuroscience, the study of the anatomy and physiology of the brain. By far, the brain, and the human brain in particular, remains as the most complicated and intriguing organ. A multitude of researchers are perplexed by the mysteries of how it works, why it is shaped as it is, how it evolved, how it is affected by injury and disease, and how to diagnose and treat disorders of the brain.

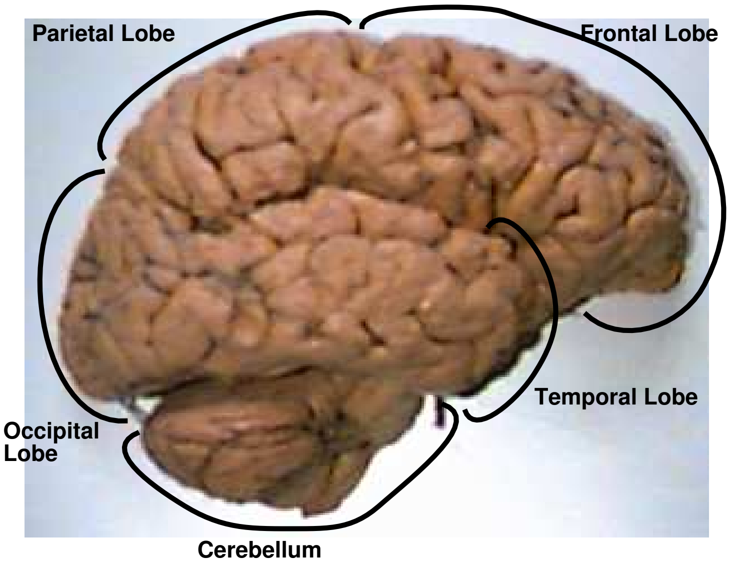
Neuroanatomy, the study of the functional and physical nature of the brain, forms an integral part of the foundation of neuroscience research. It is readily evident that in order to attack the mysteries of the brain, one of the first steps is simply to understand the various parts that comprise the organ. Over the years, this has resulted in considerable effort applied to dissecting animals of all types, in attempts to discern both the physical and functional workings of the brain. Often this work is distilled into a neuroanatomical **atlas**, such as the characterizations of cerebral sulci described by Ono et al [OKA90], or the atlas of the human brain defined in a standardized coordinate system by Schaltenbrand et al [SB59] The labeled drawings and descriptions in such atlases can be thought of as a higher level physical model of the underlying anatomy, which is used both as a detailed reference by students of neuroanatomy and by researchers interested in relating data from their investigations to established neuroanatomical knowledge.

A quick survey of typical neuroanatomical research illustrates the need for methods of creating highly structured and detailed models as tools for understanding the physical nature of the brain. Neuroanatomists are constantly looking for ways to go beyond qualitative assessments of brain structure and function, and provide quantitative methods to analyze the brain. Steinmetz et al [SRJ+90] manually identify regions on ten cadaver brains to determine if the surface area differs significantly between the left and right areas of the brain. Similarly, James [Jam92] investigates the unfolded shape of the human cerebral cortical surface and attempts to measure its surface area. Such use of real brain specimens is invaluable in the analysis of brain shape, but the evaluation of digitally acquired information is rapidly achieving importance in this pursuit. An area of considerable ongoing neuroscience research is **multi-modality registration**, which involves combining and reconciling information from two or more different types of imaging study. Evans et al [EMN+92] provide methods of mapping functional images to anatomical images of the brain, in attempts to determine which parts of the brain respond to certain stimuli. Similarly, Steinmetz and Seitz [SS91] investigate the functional anatomy of language processing. One of the main hurdles encountered is the inherent anatomical variability across the many subjects involved and the difficulty of identifying and relating the corresponding neuroanatomical structures across individuals. These and many other examples of neuroscience research [GKVPF89, DF92, OSTG89, SRH+89, SFF89] provide a strong motivation for developing computational tools for modeling the anatomical structure of the brain.

The problem addressed by this dissertation is one of creating digital representations of the human brain from one particular type of medical imaging data. Before defining the problem completely, it will be helpful to provide a brief overview of the structure of the human brain and related terminology, and a description of MR imaging, which provides the input for the problem at hand.

2.2 Neuroanatomical Domain

Structurally, the human brain is a very complex organ consisting of many components with widely varying shapes and sizes. Brain tissue is often classified into two types, **gray matter** and **white matter**. The gray matter consists mostly of neuron cell bodies, whereas the white matter is predominantly the axons or other processes of the neuron. The brain is divided into two roughly symmetric hemispheres. The **cerebral cortex**, depicted in Fig. 2.1, is the external layer of gray matter. The surface of the cerebral cortex (**cortical surface**) is highly convoluted, consisting of deep crevices which are termed **fissures** or **sulci**, the singular of the latter being **sulcus**. The



a) Right view

b) Top view

Figure 2.1: Photographs of a human brain specimen

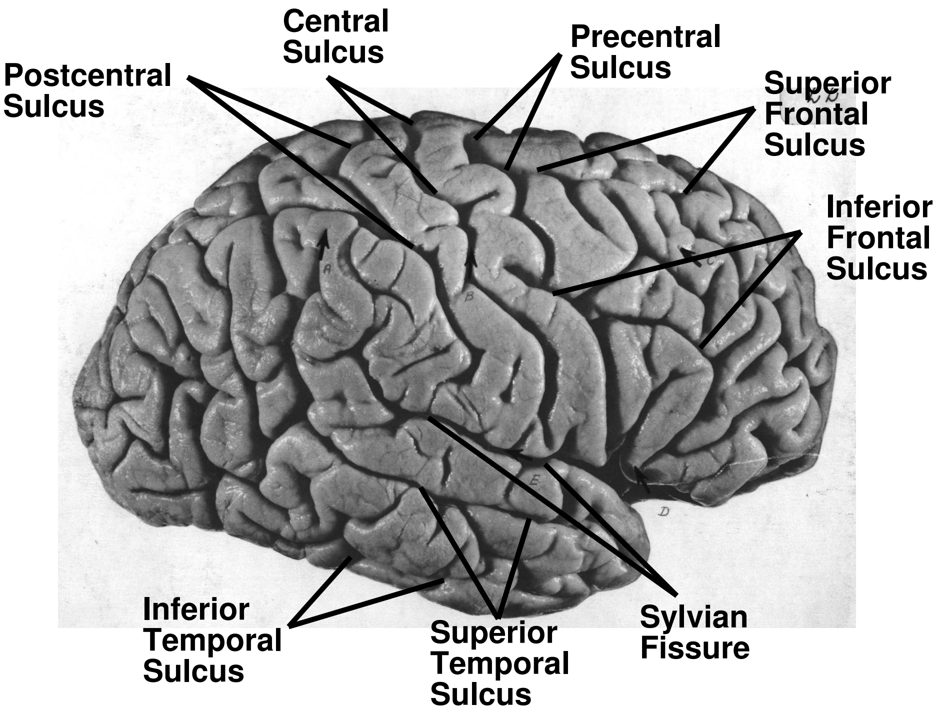


Figure 2.2: Normal sulcal topology.

folds themselves are termed **gyri**, the singular being **gyrus**. Several major sulci are presented in Fig. 2.2. Each hemisphere is divided into four main regions: the **frontal lobe**, the **parietal lobe**, the **occipital lobe**, and the **temporal lobe**. The boundaries of these lobes are defined in part by three major sulci: the **central sulcus**, the **parieto-occipital sulcus**, and the **sylvian fissure**. The hemispheres themselves are separated by the **inter-hemispheric fissure**. Although the primary emphasis here is on the outer surface of the cerebral cortex, it is important to note that the cortical gray matter is actually made up of several layered surfaces, totaling about 5 millimetres in thickness. The brain is bathed in a liquid called **cerebral-spinal fluid** or **CSF**. The **ventricles** are a set of CSF-filled cavities within each hemisphere that are quite prominent and distinct in shape and appearance. Other relevant structures in the brain are the **cerebellum**, a highly convoluted mass of gray matter, white matter, and deep nuclei at the rear of the brain, and the **brain stem**, a set of several small structures at the base of the brain which provides the link to the spinal cord. Neuroanatomists often view planar cross sections of the brain, usually classified into one of three orientations, each loosely defined by anatomical features. A **sagittal** section is approximately perpendicular to the line segment joining the ears. A **coronal** section is roughly perpendicular to the line segment from the nose to the back of the head. A **transverse** section is approximately perpendicular to the line defined by the neck. Photographic images of these three orthogonal cross sections through a human brain specimen are presented in Fig. 2.3.

2.3 Input Data – Magnetic Resonance Images

During the last two decades, neuroanatomists have begun investigating the structure of the human brain using a method called **Magnetic Resonance Imaging**, commonly referred to as **MRI** or **MR imaging**. MR imaging is a non-invasive technique for measuring the response of certain types of objects, usually organic tissue, to magnetic stimuli. An MR image of an object provides a three dimensional view of an object, where areas of similar composition appear as regions of common image value, and borders between different structures often appear as gradients in the image. A brief description of the MR imaging process is presented here, followed by a discussion of the nature of the signal being measured and sources of noise and other errors.

Within an MR scanner, there is a static magnetic field maintained by a superconducting magnet. When a target object is placed in this field, its molecules undergo what is termed **bulk magnetization**, where nuclear spins are oriented along the direction of the magnetic field. Then a series of radio frequency pulses are applied to induce local perturbations of the magnetic field within the target object, which causes the object to emit radio energy. The energy emission is measured by a **Radio Frequency coil** (or **RF coil**) placed around the object. The energy emitted depends on the strength, frequency, and timing of the pulses, as well as the intrinsic magnetic characteristics of the tissue in the object, and is often characterized by two time

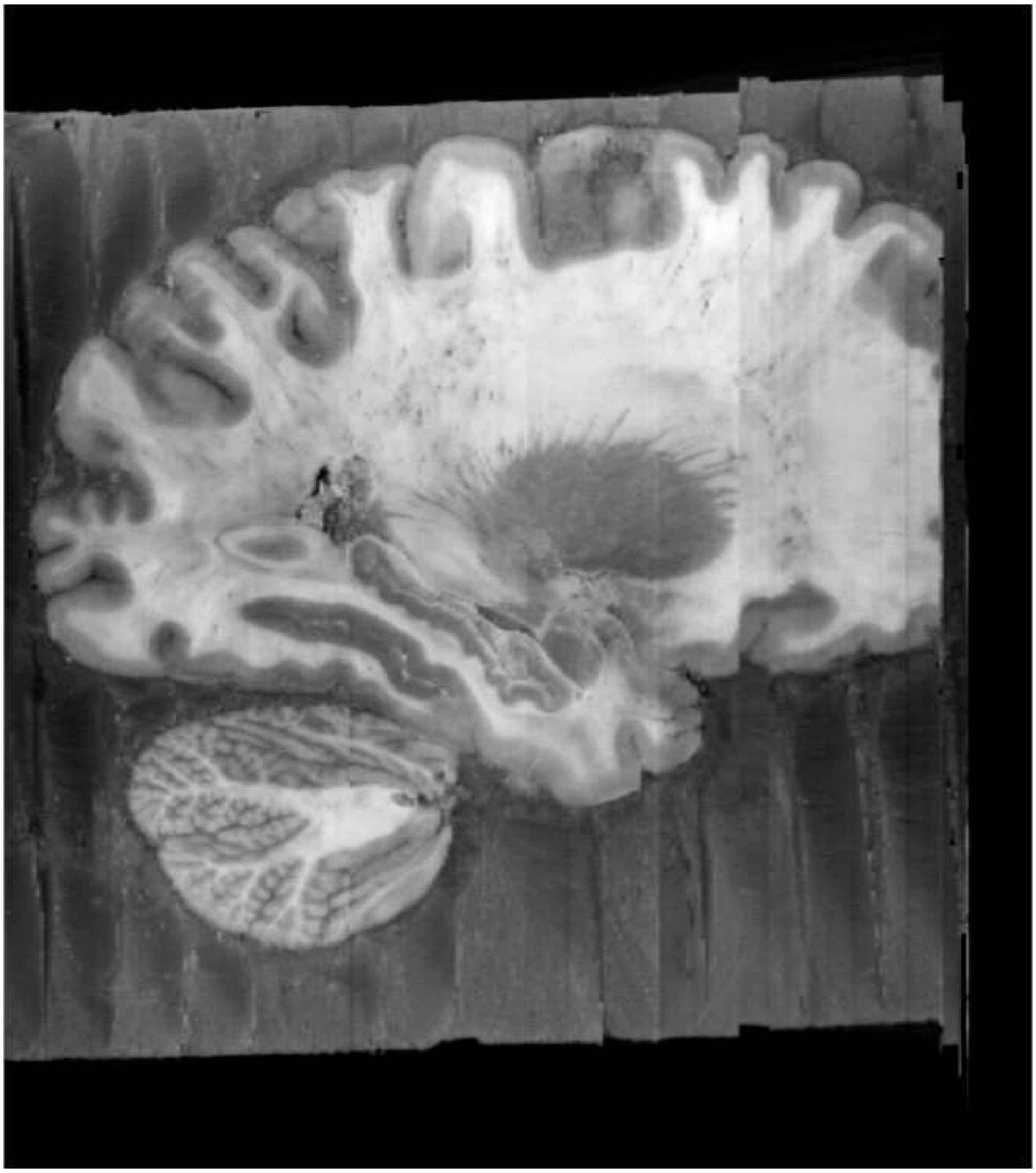


Figure 2.3a: Sagittal section through human brain specimen.

constants, **T1** and **T2**. Many pulse sequences have been devised which are optimized to favour either the T1 or T2 time constant, and the resulting images are termed **T1-weighted** and **T2-weighted** images, respectively. In addition, pulse sequences can generate values related to the number of protons in local regions of the target and hence produce **proton-density-weighted** or **PD-weighted** images. Each of these types of MR image provides different contrast characteristics between tissue

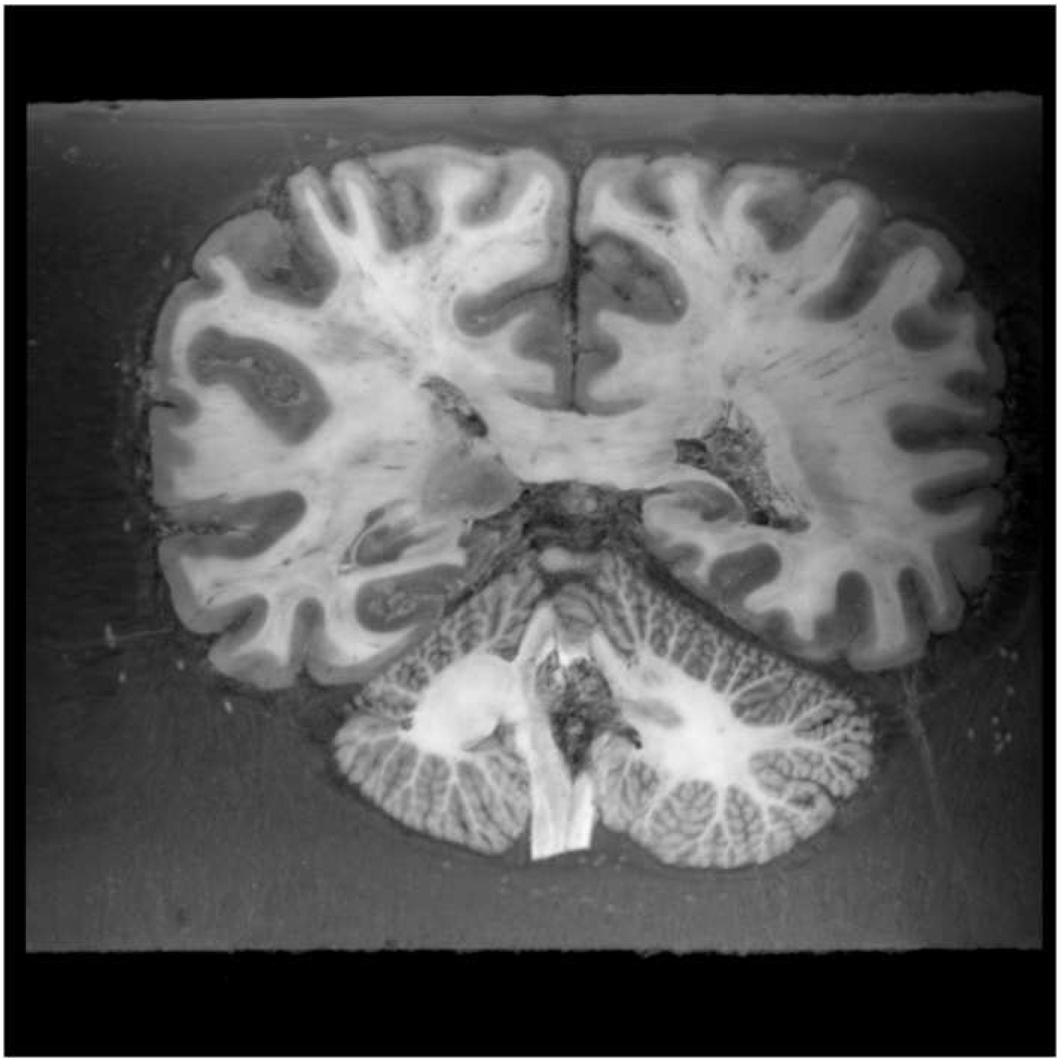


Figure 2.3b: Coronal section through human brain specimen.

types, and each is therefore more suited than the others at imaging particular tissue combinations.

An MR system uses Fourier analysis of the radio frequency data to produce a two or three dimensional regular grid sampling of the target object. The separations between samples are usually on the order of 1 millimetre and the number of samples is on the order of 256 in each of the three coordinate directions. A three-dimensional image is often referred to as a **volume** or **image volume**, in order to emphasize its three dimensional nature. Consequently, the three dimensional box centred at a

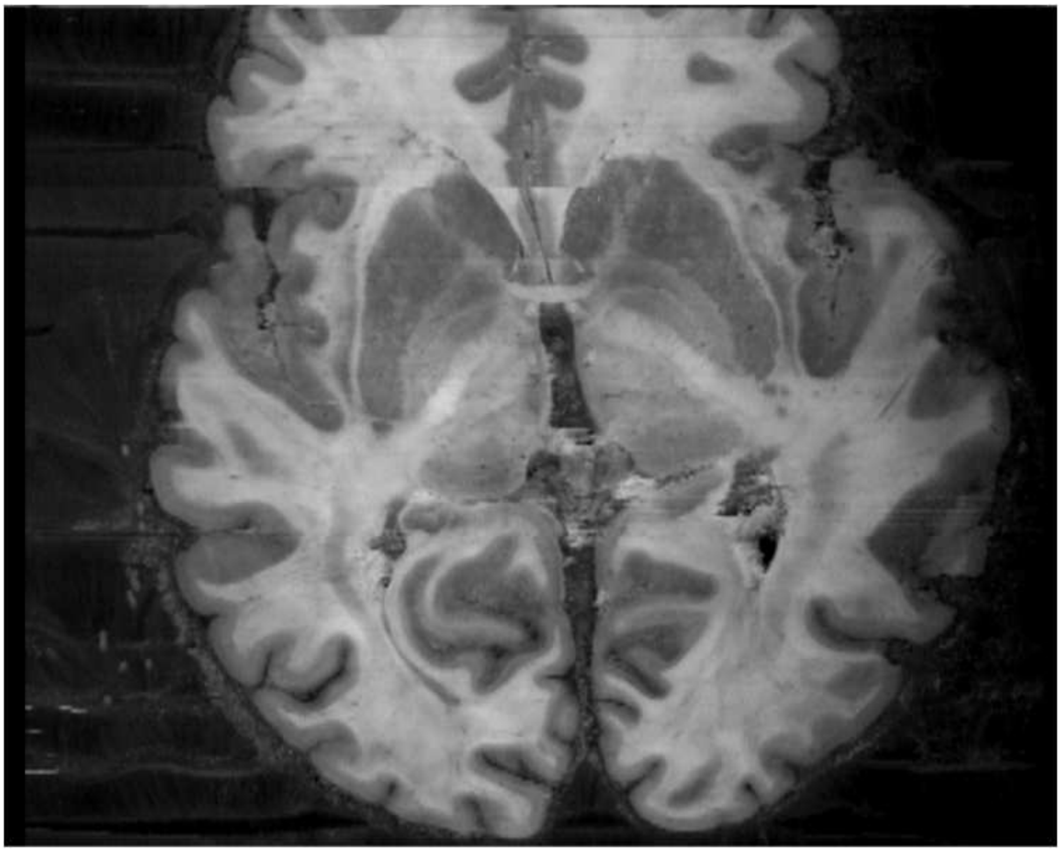


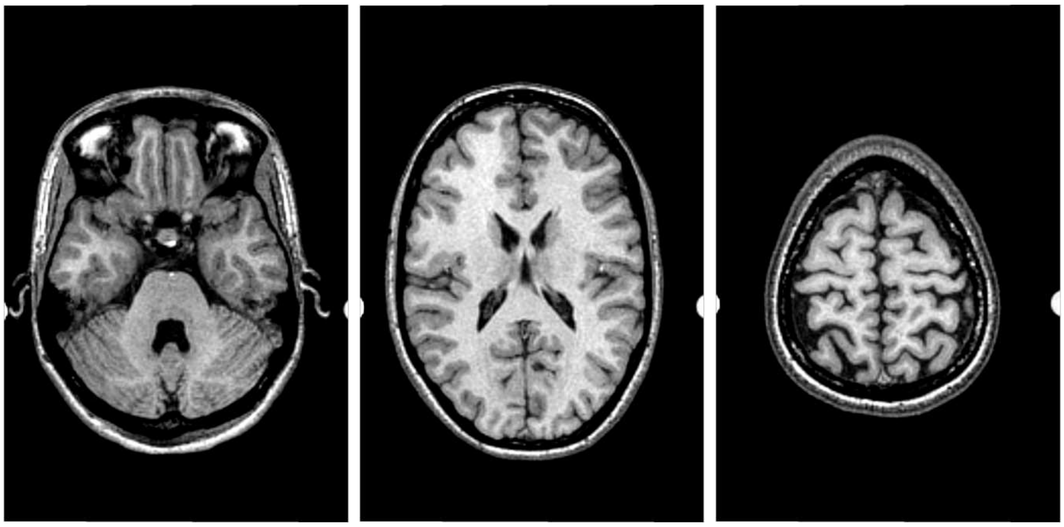
Figure 2.3c: Transverse section through human brain specimen.

Figure 2.3:

sample point with axial sizes equal to the sampling interval is termed a **voxel**, which is a contraction of the term **volume element** based on the two dimensional analog, the **picture element** or **pixel**. For convenience, the samples are often treated as having been generated with a perfect box filter over each voxel, or as a point sample at the centre of each voxel. Figure 2.4 depicts a set of parallel slices through T1-weighted, T2-weighted, and PD-weighted MR volumes of the same normal human subject. In this example, each volume is sampled on a grid of 172 by 256 by 256 with an isotropic sampling interval of one millimetre, resulting in 11 million sample points.

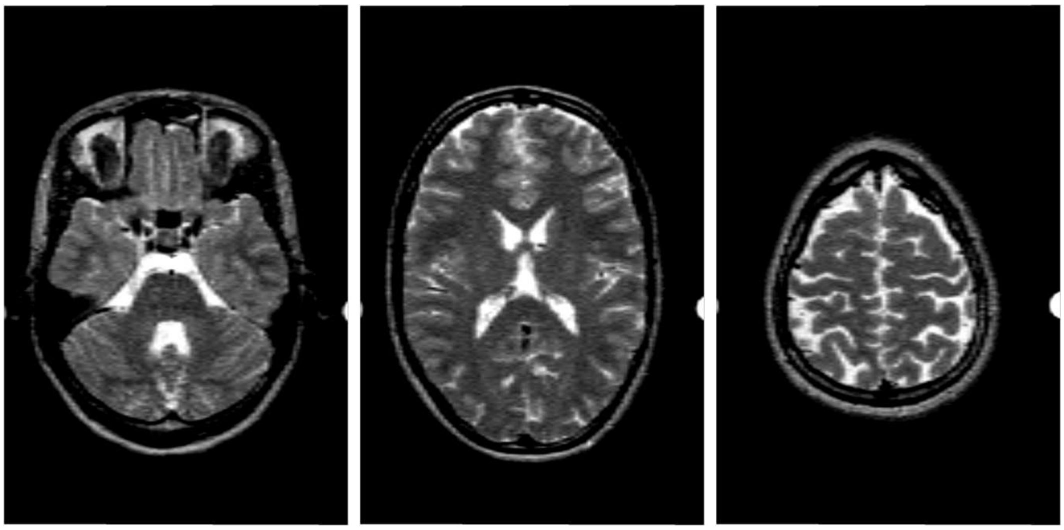
Mathematically, we can treat the data as an array of n_i by n_j by n_k values:

$$\{I_{i,j,k} \ : \ I_{i,j,k} \in \Re, \ \ 1 \leq i \leq n_i, \ \ 1 \leq j \leq n_j, \ \ 1 \leq k \leq n_k\}\,.$$



z=-20 mm   z=20 mm   z=60 mm

2.4a) T1-Weighted Images

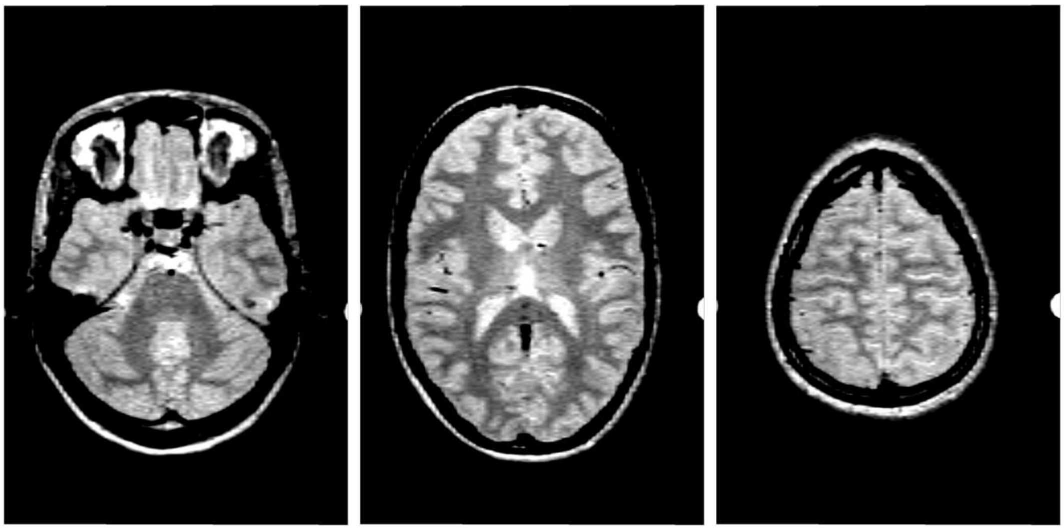


z=-20 mm   z=20 mm   z=60 mm

2.4b) T2-Weighted Images

Any of the many methods of interpolation and approximation [BBB87] such as B-splines, Hermite splines, or simple trilinear interpolation may also be used to create a continuous differentiable function which can be evaluated anywhere in the domain of the image:

$$F(x, y, z), \quad \nabla F(x, y, z), \quad \nabla^2 F(x, y, z), \ ...$$



z=-20 mm　　　　z=20 mm　　　　z=60 mm

2.4c) Proton-Density-Weighted Images

2.4d) Position of MR slices in a), b), and c).

Figure 2.4: T1-, T2-, and PD-weighted MR images.

$$x_{min} \leq x \leq x_{max},$$

$$y_{min} \leq y \leq y_{max},$$

$$z_{min} \leq z \leq z_{max}.$$

2.3.1 Partial Volume Effects

An MR image has certain characteristics deriving from the nature of the imaging system involved. Each sample value is actually a convolution of a **point spread function** with the target object over a finite subspace. The point spread function is simply the weighting function that defines how the tissue at a particular geometric location contributes to samples in the surrounding region. The point spread function in MR systems is characterized as a sinc function ($\frac{\sin ax}{ax}$), where the distance between the two central zero crossings is twice the width of a voxel, as depicted in Fig. 2.5. Because of the finite width nature of the point spread function, as well as the finite number of samples, MR images are subject to what is called the **partial volume effect**. The partial volume effect occurs in regions which are not of homogeneous composition, where the value of a particular sample includes weighted contributions from varying tissue types, thus causing information loss. In addition, the finite spacing between samples introduces the standard limitation from fundamental sampling theory, the **Nyquist limit**. The Nyquist equation dictates that details higher in frequency than the Nyquist limit, which is half the sampling rate, will not, in general, be captured in the final image. This, combined with the blurring nature of the finite width point spread function, results in the partial volume effect, which is most evident near the boundaries between different structures. Edges are either less apparent in the image or not visible at all. The tightly folded configuration of the cortical surface makes the partial volume effect particularly problematic. There are many places in the brain where the cortex on one side of a sulcus is very close to or actually touches the cortex on the opposite side, as depicted in Fig. 2.6a. Figure 2.6b illustrates how the MR image would incorrectly portray the touching parts of the cortex as connected.

Examining a typical T1-weighted MR image illustrates some of the consequences of partial volume effects. Figure 2.7 shows a magnified view of the temporal lobe region. Near the centre of the figure, the concave indentation in the white matter

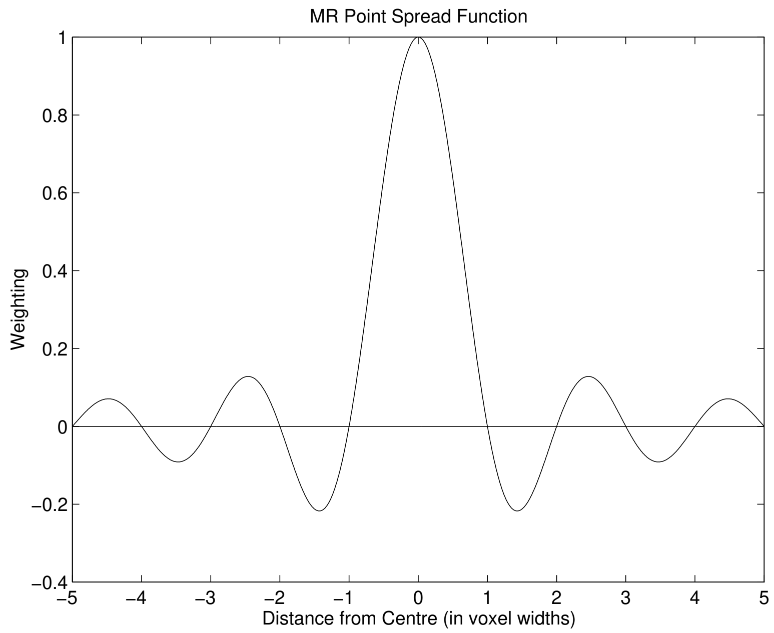
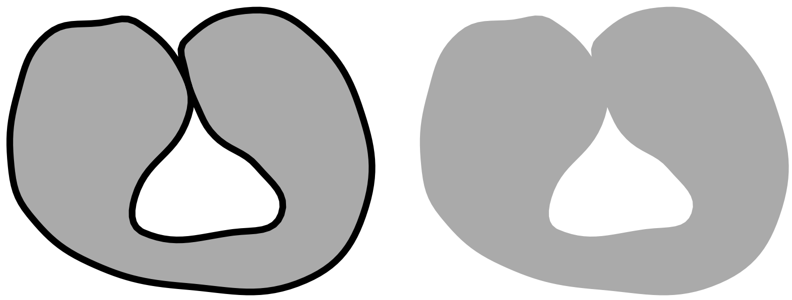


Figure 2.5: The typical point spread function of an MR imaging system.



a) b)

Figure 2.6: a) True data with touching gyri. b) resulting image data, where gyri are no longer distinct.

clearly indicates sulcal folding based on the anatomical assumption that the gray matter follows the shape of the white matter. However, the image in this area does not show any clear boundaries corresponding to the sulcus. No conventional image processing algorithms have been found to satisfactorily detect edges in such a situation. Due to the prevalence of these types of close proximity configurations in the typical

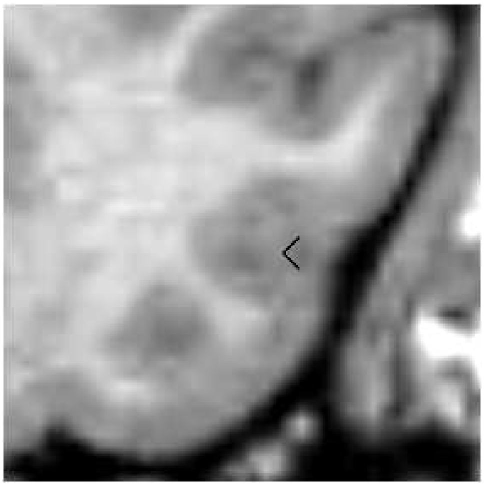
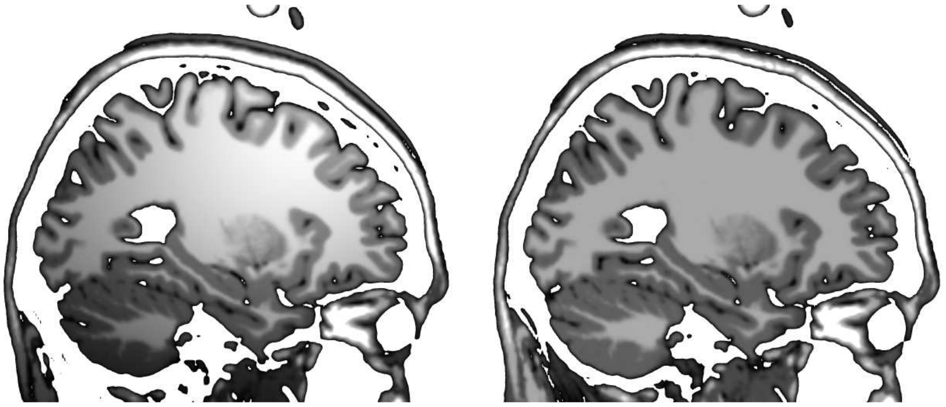


Figure 2.7: Partial volume effect of blurring a sulcus, pointed to by the $<$ symbol, in a T1-weighted MR image.

human brain, even an order of magnitude improvement in resolution of MR imaging systems would probably still not correct these problems, and it is not evident that partial volume errors will be reduced in MR images for quite some time. Therefore, incorporating *a priori* knowledge from the neuroanatomical domain into the image recognition task represents a promising alternative to purely data-driven methods.

2.3.2 RF inhomogeneity

An imaging artifact peculiar to MR imaging systems is the problem of **RF inhomogeneity** or **RF non-uniformity**. Like any antenna, the RF coil system has a non-uniform spatial sensitivity. As a result, the measured value of a given tissue type will vary depending on the position of the tissue relative to the RF coil. This typically results in an image whose mean intensity increases slowly along some direction in three dimensions. Figure 2.8a shows a simulated MR slice with an exaggerated RF inhomogeneity (about three times normal), where the image grows brighter diagonally from bottom left to top right. Figure 2.8b shows the same simulated image without RF inhomogeneity. The maximum difference in scaling across a single MR



a) b)

Figure 2.8: a) RF image inhomogeneity. b) no RF image inhomogeneity.

image due to RF inhomogeneity is typically on the order of 20 % of the image value.

2.3.3 Image Noise

As in most sensor equipment, noise due to many sources may influence the resulting image. The use of gradient magnetic fields makes the system susceptible to errors due to eddy currents around metallic objects in the vicinity of the scanner, such as its own components or articles in the subject's clothing or body. The image noise inherent in an MR image is often modeled as a Gaussian noise distribution in the complex number system, where the mean of the real and imaginary parts are independent. The resulting noise distribution is termed **Rician** [Nis95], which is similar to a Gaussian distribution in areas of high signal. However, in areas of low signal value, the distribution is more complex because the final signal is computed from the magnitude of its real and imaginary components, which effectively maps negative values to positive ones. Noise processes that decrease a signal value below zero therefore can result in an increase in the final signal value. The intensity of the noise is typically on the order of 3 % of the signal intensity.

2.3.4 Movement Artifacts

The length of time required to perform an MR scan introduces the possibility of errors in the image due to dynamic changes in the subject. For living specimens, it is often difficult to remain absolutely still for the duration of the scan, which is typically on the order of half an hour. Even a subtle movement such as a swallowing motion can have a pronounced effect on the accuracy of an MR image with one millimeter sampling. In general, living organisms being scanned are subject to dynamic changes such as movement of voluntary and involuntary muscles, blood flow, and other changes in organ shape and size due to normal bodily processes. Large movement artifacts often show up as sharp discontinuities across a plane of the image volume, and are difficult to correct. In general, the movement artifacts in cooperative subjects are more subtle and no correction is attempted. A good introduction to the use of magnetic resonance in medical imaging may be found in [CDM84]. A more detailed description of MR characteristics such as noise, RF inhomogeneity, and the point spread function may be found in [Nis95].

2.4 Problem Definition

Having examined the nature of MR imaging data, the problem addressed by this dissertation is proposed:

Problem : **Given three dimensional magnetic resonance images, create digital representations of the entire cerebral cortical surface that are geometrically simple and corrected for partial volume effects.**

The basic requirements of a solution are:

Robust segmentation	The cerebral cortical surface of normal human individuals must be reliably segmented, as measured against one or more experienced neuroanatomists,
Geometrically simple	the models created must be geometrically **simple**, that is, they must not self-intersect,
Partial-volume corrected	the method must correct for the ambiguity of partial volume effects in areas where neighbouring gyri are in close proximity or touching, by correctly locating the entire sulcus between the gyri,
Many datasets	the method must be applicable to a large number of datasets,
Arbitrary resolution	the solution must be extensible to an arbitrarily high level of detail,
Fully Automatic	no user intervention must be required, and
General	the method must be applicable to images produced by a variety of MR imaging protocols.

2.4.1 Segmentation

The essential task is that of object recognition or **segmentation**. In a typical MR volume of a human head, there are hundreds of neuroanatomical components visible. The cerebral cortical surface is a relatively large and highly convoluted neuroanatomical structure that is very difficult to segment by hand. Using interactive tools, an experienced observer may take several days to label the cortex in a high resolution MR volume. This dissertation will be concerned primarily with automatically segmenting the outer layer of the cortical surface, although application to other components will not be excluded. Due to the typically high cost of manual intervention in neuroanatomical segmentation, the method must be fully automatic, with no user input required. The technique should be applicable to T1-weighted, T2-weighted, and PD-weighted MR images, and be insensitive to the various imaging parameters of these protocols.

In order to validate an automatic method of segmentation, there are two general methods that may be used. The first is to create data for which the correct answer is somehow known or assumed, and to test the results of a solution against this "gold standard". The second method is to use real data, for which the answer cannot be reasonably known, and create reasonable qualitative and quantitative estimates of error. In the case of segmentation of the cortical surface of the human brain, a manual segmentation of a single normal MR volume has been performed by a neuroanatomist, and this can be used as a gold standard to test segmentation methods. It has been established that intra-observer errors in manual labeling can be significant (as high as 12 % of structure volume in the case of small structures such as the caudate [CEHP95, CHPE96]), which represents a flaw in the purity of the gold standard. However, due to the large amount of effort involved, it is not feasible to achieve a better gold standard, such as, for instance, a composite of labelings by several experts. The form of this data is a labeling of voxels into distinct anatomical components. Assuming this dataset as the correct answer, the number of voxels mislabeled by an automatic segmentation procedure can be determined. In addition, validation must be performed on a number of MR volumes to determine if the results are qualitatively reasonable and consistent with a neuroanatomical understanding of the cortical surface.

2.4.2 Geometrically Simple

The term **simple** is generally used to describe a set of objects (line segments or triangles, for example) where no pair of objects share points, except on the boundaries between objects. Less formally, the term **non-self-intersecting** is also used. In the case of a **polygon**, which is defined as the set of line segments connecting successive points, as well as the first and last points, of an ordered list of three of more points in the plane, the term simple means that adjacent edges share only one point, and no other pair of edges share a point. A simple **polyhedron** is a connected set of polygons in three space where the intersection between any two polygons is either the empty set or an edge of both polygons, and each edge is a member of exactly two polygons. Contrary to conventional research, the constraint that the models created be geometrically simple is one of the principal requirements of a solution, rather than merely a desirable feature. This constraint is important in order to increase the confidence that the modeled surface is a faithful reproduction of the real cortical surface, each layer of which is known not to intersect itself. Assuming that noise and partial volume effects will continue to plague MR images, it is therefore desirable to make algorithms as insensitive to these effects as possible. One of the central postulations of this dissertation is that restricting the output of the segmentation process to simple surfaces avoids many incorrect solutions.

2.4.3 Partial Volume Correction

In addition to constraining the modeled objects to be simple, an additional criterion is that the method should be more insensitive to partial volume effects than conventional methods, which typically either ignore partial volume effects, or use simple image processing operators for an intuitive, but limited, solution. In particular, areas of neighbouring gyri that appear in the image as connected tissue should be correctly identified with a boundary between them. It is very important that the method locate the complete depth of each sulcus, in order to most accurately represent the cortical surface. One of the difficulties with adding this criterion to the problem definition is that it is very hard to determine if a particular solution is correct. However, simple test cases can be devised where the answer is known, and application to real MR data can be qualitatively evaluated by experienced neuroanatomists. In effect, one hopes to duplicate the high level information used by an experienced neuroanatomist who factors out partial volume effects using a comprehensive understanding of brain anatomy.

2.4.4 Automatic Operation

The segmentation method must be applicable to a large number of datasets, as even a single neuroanatomical study may consist of several hundreds of individual scans. Therefore the operation of the segmentation process must involve no user intervention on individual datasets. In addition, as the required resolution of the digital models increases over time due to higher resolution data and more ambitious studies, the segmentation process must be extensible to an arbitrarily high resolution. This criterion also precludes the practical use of human intervention in the process, reinforcing the need for fully automatic methods.

2.4.5 Representation

The problem statement stipulates that the digital models representing the solution are surface representations of anatomical structures. The choice of surface representation, as opposed to a volume representation, for instance, arises from the types of post-processing typically applied to digital brain models. In order to assist analysis of the anatomical structures by experienced neuroanatomists, it is advantageous to be able to provide various types of visual depictions of a segmented structure, in particular, as a three dimensional object which looks similar to a photograph of the actual anatomical structure. However, it is more important to provide a representation that facilitates quantitative analysis. The segmented object representation must allow such rudimentary operations as measurement of distances, surface areas, and volumes of anatomical structures and their subcomponents. In addition, more sophisticated quantitative characterizations of geometric shape are currently being investigated by neuroanatomists. A surface representation can generally satisfy these requirements, and therefore has been chosen as the form of the output for the problem. Which type of surface representation to choose remains to be discussed in a subsequent chapter.

Chapter 3

Previous Work

There continues to be considerable research relating to processing of MR images. Consequently, there are many techniques that address some or all of the various aspects of the segmentation problem described in chapter 1. The following is a discussion of various techniques which are relevant to the problem at hand. Rather than being competing methods, many of these methods complement each other and typically, several techniques are used in combination to provide an overall solution. The methods discussed are divided into two classes. The following section describes those methods that can be classified as **volumetric pre-processing**, and the subsequent sections deal with techniques for image segmentation. The applicability of each method as well as its relative advantages and disadvantages are explored.

3.1 Volumetric Pre-Processing

Many general image processing techniques are relevant to the three dimensional images produced by MR scanners. Three methods of pre-processing the MR images for subsequent image analysis are described here: image registration, tissue classification, and RF inhomogeneity correction.

3.1.1 MR Image Registration

There are a multitude of methods proposed to handle the task of **image registration** or **spatial normalization**, which is the process of spatially transforming an image to be aligned within some target coordinate system. The goal of MR registration is to provide a standardized frame of reference for subsequent analysis. For processing of human brain MR images, the initial step is typically to apply a simple rigid geometric transformation consisting of three dimensional scaling, rotation, and translation, to align the dataset into a standard coordinate system, often termed a **stereotaxic** coordinate system. This can aid analysis by introducing a certain amount of predictability into the positions of various anatomical structures. This image registration task is addressed by both manual and automatic methods. Manual methods [LHH+91, EMN+92] involve locating corresponding coordinates (either based on anatomy or on artificially introduced objects (**fiducials**)) in both the given image and a standard target image to which all images are mapped. The set of pairs of three-dimensional coordinates is used to define a transformation between positions in the two images. The given image is transformed by this mapping to produce an image that is in registration with the target.

The reliance on human observers for the definition of corresponding points motivates a quest for more automatic methods of image registration. As a result, there are several automated methods which attempt to match image features, such as intensity values or gradients, across a pair of images [Bcc89, CNPE94, KGC+89, RTL+93, TJP+93]. An initial guess for the transformation is given, either by manual intervention, or by an automated procedure such as **principal axes analysis** [RTL+93]. The similarity between the image to be transformed and the target image is measured and the parameters of the transformation are adjusted to attempt to maximize this measure. The measure used is typically a function such as the cross-correlation of intensity values in the original image with the values in the transformed position in the target image. These methods are being used successfully to factor out the gross position, orientation, and size differences of various individual brains, and can be thought of as a very coarse means of segmentation.

The work of Talairach and Tournoux [TT88] in manually defining a standardized coordinate system based on identifiable anatomical landmarks is adapted into a digital framework by various methods. The method of Collins et al [CNPE94] involves the definition of a stereotaxic coordinate system, based on the atlas of Talairach and Tournoux. Using manual methods of landmark identification, the reference model image is placed into stereotaxic space. An automatic three dimensional image registration algorithm can then be applied to register any target image with the model image. Figure 3.1 depicts the registration of an image into stereotaxic space from **native** space, the intrinsic coordinate system of the MR scanner. Subsequent neuroanatomical analysis of the target image is greatly facilitated by its standardized orientation and positioning within the frame of reference defined by the stereotaxic coordinate system.

3.1.2 Tissue Classification

As described in chapter 1, the human brain has several different tissue compositions, the principal ones being gray matter, white matter, and cerebral spinal fluid (CSF). Depending on the imaging parameters of the MR acquisition, these different tissue types give varying signal responses, which are used to discern tissue type distribution in the image. By using several MR images of a single brain, including the T1-, T2-, and PD-weighted images, a three dimensional feature vector is derived for each voxel, consisting of the intensity values for the voxel in the three images. Pattern classifiers such as neural networks, nearest neighbour methods, and Gaussian-modeled linear discriminants [1] are invoked on training data where the tissue types are known, in attempts to create an accurate classifier for arbitrary datasets which are acquired with similar imaging parameters to the training data [ZDM93, CCR+93, KCS+92,

[1] [Nil90] provides a good introduction to these and other pattern classifiers

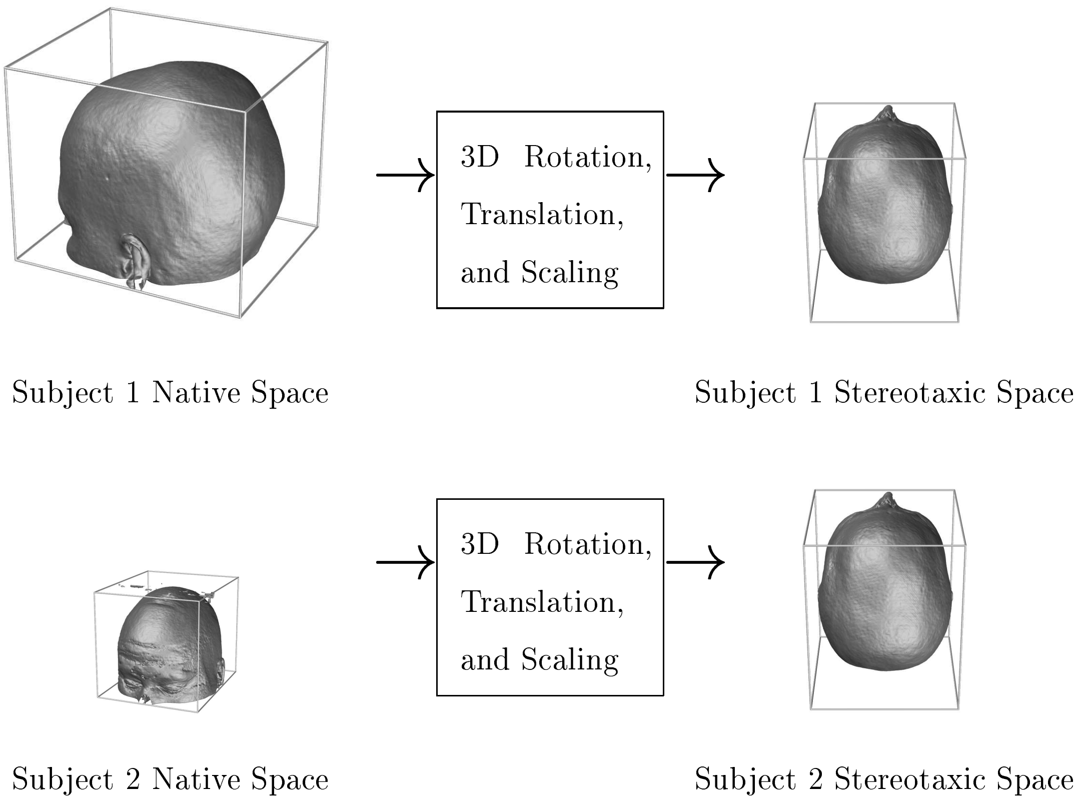


Figure 3.1: Transformation from native space to stereotaxic space.

LHH$^+$91, RLW93]. This constitutes a low-level segmentation of the brain, which does not necessarily correspond to an anatomical segmentation of the brain, since different anatomical structures may have the same tissue type.

Figure 3.2 illustrates the results of a typical tissue classification algorithm generating an image with four classes: gray matter, white matter, CSF, and background. Although tissue classification methods by themselves are not sufficient for segmentation of anatomical objects, it may be possible to use the tissue class information to assist other methods of segmentation. In particular, tissue classification will be revisited as an edge detection step for MR images in the next chapter.

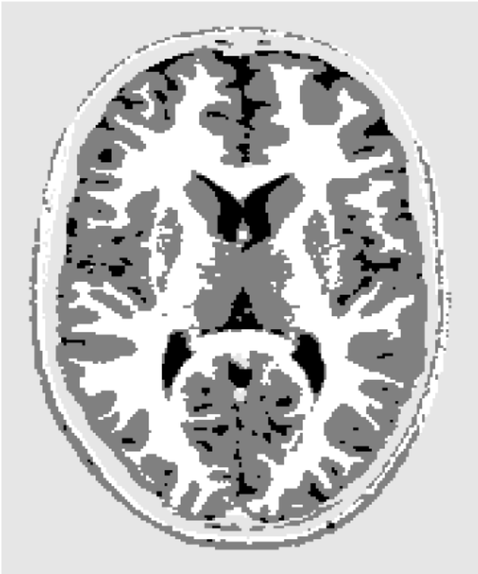


Figure 3.2: One slice through a 4-class volume (white matter is white, gray matter is gray, CSF is black, background is off-white).

3.1.3 RF Correction

The unique nature of the RF image inhomogeneity distortion that plagues MR imaging invites specialized methods to reduce or remove this artifact from images (Fig. 3.3). These methods can be loosely classified into two categories, where the principal difference is whether or not the RF correction is performed in conjunction with tissue classification. When used with tissue classification, the typical method attempts to estimate two unknowns: the tissue classes of the voxels and the spatially varying RF field over the volume. This is accomplished by fixing the estimate of one of the unknowns in order to compute an estimate for the second unknown, which is then used to estimate the first unknown. These two steps are repeated until the sequence converges. A popular method which uses this methodology is called the **Expectation-Maximization** algorithm [WIGKJ94].

The other general class of RF correction methods does not explicitly involve tissue classification, but instead uses the intensity histogram to achieve a similar result. By comparing local histograms of image intensity in different spatial locations, the shift

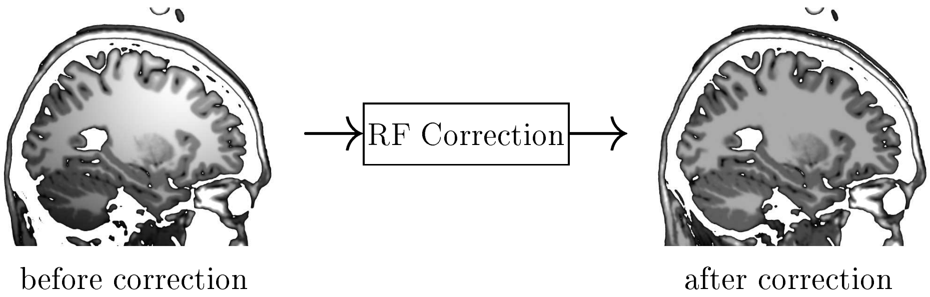


Figure 3.3: RF image inhomogeneity correction reduces the slow-varying intensity gradient across the image.

in the histogram is identified, and thus an estimate of the RF inhomogeneity field can be derived. One example of this method [SZE97] will be revisited in the next chapter.

Having examined various methods of pre-processing images, a survey of image segmentation algorithms is presented. Any method which divides images into distinct regions or extracts models of specific regions of the image may be considered an image segmentation algorithm. Some important differences among the various methods described in the following sections are the representation of the segmented objects, the level of user intervention required, and the level of model information used to constrain the process.

3.2 Contours

Many methods of image segmentation are based upon choosing a threshold which corresponds to the image intensity at the boundaries between objects in the image, or using gradient-based edge detectors to label particular regions of the image as boundary. Boundary points detected on two-dimensional slices of the image are then connected together into a curve or contour. This partitions the image into a set of connected components (Fig. 3.4), thereby achieving a segmentation of the image. The advantage of contours is that the problem of three-dimensional segmentation is initially converted to a more tractable two-dimensional problem. Often two objects which are connected in the three-dimensional space are not connected on many of the slices through the volume. Even if they are connected, manual segmentation techniques and morphological operators are simpler and more successful when applied independently to each slice, due to the ability of conventional two-dimensional technology to more easily display image slices than entire volumes. Two dimensional image contouring methods applied to three dimensional images are prevalent [CL88, EPO91, FKU77, GD82, MSS92] and generally attempt to construct a three dimensional surface from stacks of contours extracted from parallel slices. The essence of the task is find correspondences between positions on contours of neighbouring slices. The difficulty is that when these methods are applied to complex shaped objects, it may be very difficult or impossible to decide how to connect points on contours to those on neighbouring slices. Because this often occurs when a contour splits into two on the next slice, this is usually called the **branching problem**, and is a significant factor in limiting the use of contour methods in higher dimensional applications. Methods that attempt to solve these problems can involve very complex algorithms [EPO91]. In actuality, the initial advantage of reducing the problem from three to two dimensions is offset by the fact that the complexity of the higher dimensionality problem has not been eliminated, but rather delayed until the final connection phase. However, because of the wealth of published experience in manual and automatic methods of contouring datasets, contouring is a segmentation tool that should not be ignored, and can prove useful for various aspects of the segmentation problem, including preprocessing data and building models for other segmentation methods.

3.3 Isosurfaces (3D Contours)

The inefficiency of contours in dealing with three dimensional data has led many researchers to explicitly extend the idea of contouring to three dimensions. The ob-

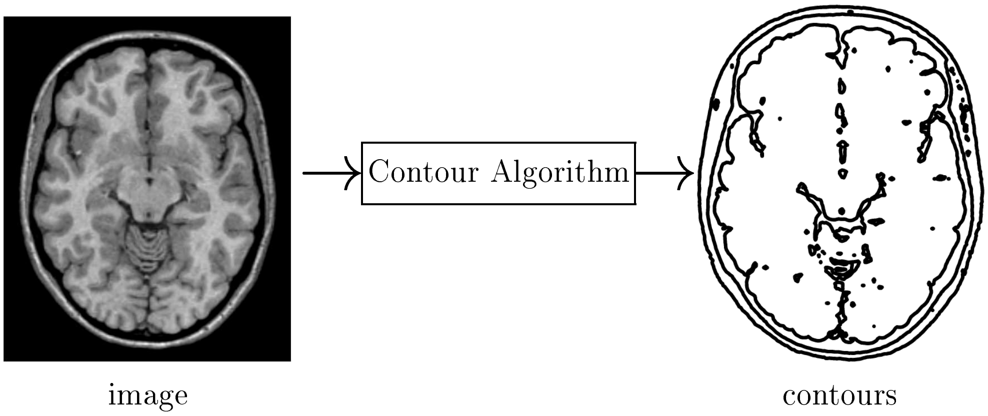


Figure 3.4: 2D image contouring.

vious approach is to construct surfaces of the boundaries of the objects, again using an image threshold. Since the surfaces correspond to sets of points which are of equal values, they are called **isosurfaces**. Isosurface algorithms [Blo88, BW90, CLL$^+$88, HW90, KCHN91, LC87, WMW86] are the three dimensional analog of two dimensional contouring and most isosurface methods are implemented in a very similar fashion. Typically, the volume is tessellated into small, simple geometric objects, and the thresholded surface is approximated in each, building up a set of connected polygons. The most cited example is the method of Lorensen and Cline [LC87] which uses rectilinear boxes as the tessellating object, leading them to term the method **marching cubes**, in reference to the algorithm's processing of the cubes one by one. By examining the sign of the difference between the desired threshold and each of the eight corner nodes of a single box, a set of one to six triangles is created to approximate the surface through the box. Figure 3.5 illustrates one situation where four nodes are below the threshold and four are above. In this case, the approximation to the surface is a set of four triangles, whose vertices are found by linear interpolation along the relevant box edges. The algorithm simply examines the eight vertices of each box in the dataset, constructing a small number of triangles for each one, and results in a connected surface for each connected object in the volume, a simple example of which appears in Fig. 3.6. Interestingly, although the marching cubes method is by far the most commonly referenced method, it has the serious problem of generating surfaces that may have holes, where the triangle edges do not join well across adjacent voxels in some cases. A similar method published one year earlier by Wyvill et al [WMW86] avoids this problem and produces a correct surface. More recent methods usually do not suffer from this problem either, such as the method of using tetrahedral tessellations by Hall and Warren [HW90] or a similar one by Payne and Toga [PT90]. However, despite the limitations of the method of Lorensen and Cline, the term *marching cubes* has become synonymous with isosurface construction algorithms.

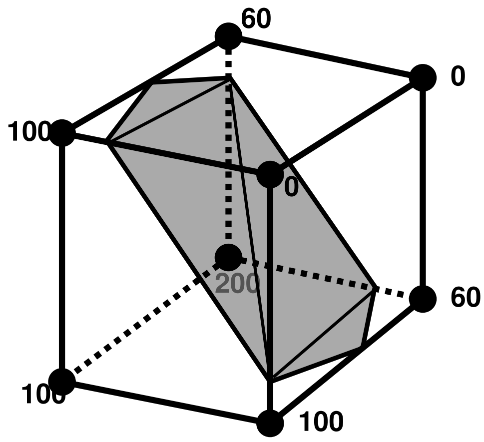


Figure 3.5: Example of marching cubes algorithm approximating the isosurface of value 80.

Although the simplicity and speed of contouring and isosurface algorithms is quite attractive, their use for surface segmentation is limited by the implicit assumption that objects that are distinct in reality appear as disconnected regions within the image. However, this assumption is often invalid, such as when objects are close together, similar in intensity, or affected by noise in the image. In these cases, an isosurface algorithm may create a single geometric object which corresponds to several distinct objects, or several distinct geometric objects which correspond to a single object, both of which may be considered incorrect segmentations of the image data. Thus, isosurface algorithms are generally used in conjunction with other manual and

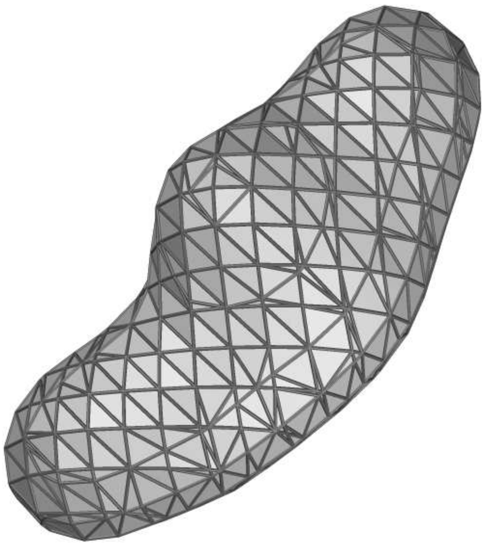


Figure 3.6: Triangulated surface generated by the marching cubes algorithm.

automatic segmentation algorithms to produce a surface representation of a previously segmented image. In addition, the lack of topological constraints in these algorithms means that holes in the data due to partial volume and other effects result in surfaces with similar potentially incorrect topological characteristics.

3.4 Morphological Operators

One method to partition the volume into separate objects is to use **morphological operators**. Morphological operators consist of simple transformations on the volume which affect the local connectivity of voxels. A good introduction is presented in [Ser82]. An image is first thresholded or classified in some way to create a binary image, where each voxel indicates whether or not an object may exist at that location. A **fill** operation labels each voxel within a homologous region in the image. An **erode** operation on a binary image changes all object voxels which have a non-object voxel in its neighbourhood (within a certain small number of voxels, defined by a specific kernel) to non-object status. This results in shrinking object regions in the image. A **dilate** operation performs the opposite operation and results in expanding object regions. A **close** operation consists of a dilate operation, followed by an erode operation, and effectively smoothes the boundaries of objects, filling in small holes. An **open** operation consists of an erode operation, followed by a dilate operation, and results in disconnecting regions which started out as very tenuously connected. Results of these operations on two dimensional images are presented in Fig. 3.7. Although the most obvious effects occur in loosely connected regions, it is important to note that all boundaries in the image may potentially be perturbed. Application of these types of operations on a thresholded volume can be used effectively in segmentation to break up an image into meaningful subregions. However, the choice of the sequence of operations and the relevant neighbourhood size will vary depending on the application and the particular characteristics of the dataset and the object being segmented. Morphological operators are thus best applied when the input data is sufficiently close to being disconnected into the desired segmentation. Unfortunately, this is not usually the case in the realm of MRI, and morphological operators have had only limited success for automated neuroanatomical segmentation, often because the sequence of morphological operations has to be empirically determined for each particular dataset. The limitations of morphological operators for automatic MRI segmentation are expressed by Hohne and Hanson: "Application of morphology operators depends strongly on intuition (at least in the case of complicated objects), which suggests that they would be used most effectively in an interactive mode" [HH92].

Morphological operators can be used to fill in holes due to partial volume effects and noise. The method of Dale and Sereno [DS93] uses a tissue classifier to define the volume of white matter voxels in an MRI volume. Then a sequence of three dimensional fill operations is performed to remove holes from the white matter. This guarantees that the surface of the resulting white matter volume is isomorphic to a sphere. However, the addition of non-white matter voxels to the list of white matter voxels calls into question the accuracy of the resulting surface.

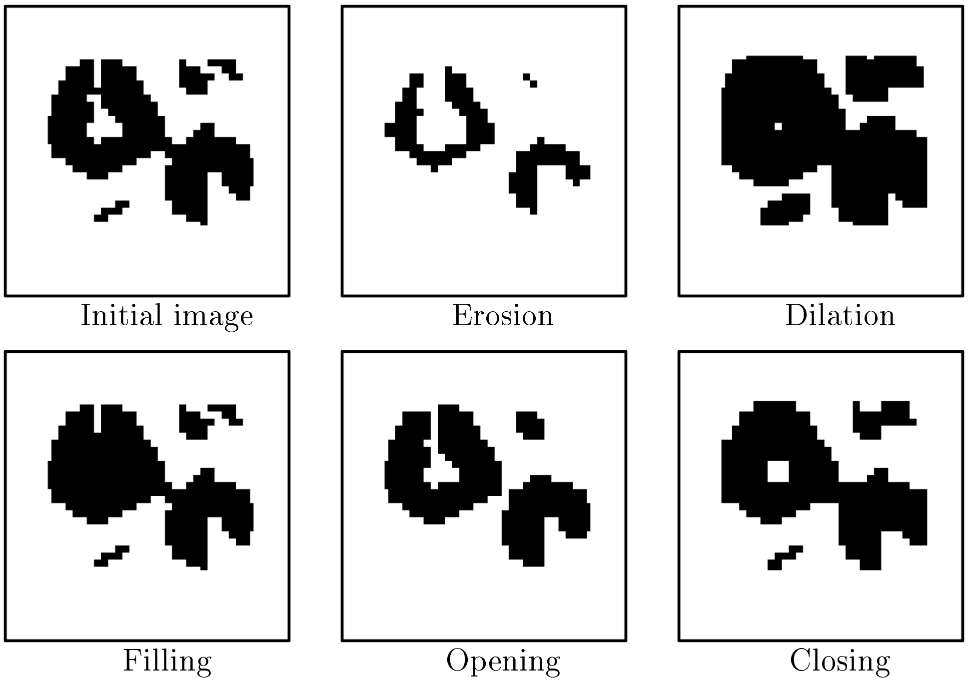


Figure 3.7: Results of five different morphological operators on an image.

3.5 Data-Driven Methods Versus Model-Driven Methods

While the methods described have been used successfully in a semi-automatic context, none of them are powerful enough to provide automatic segmentation of the cortical surface from MR images. The essential weakness of most of these algorithms is that they are almost purely **data-driven**; the only information available to the process is the data itself. Higher levels of information are available only indirectly, usually as input from an experienced human observer. As a result, researchers are investigating methods of incorporating *a priori* knowledge (application-specific information), into the process of image segmentation, in hopes to develop general-purpose fully automatic algorithms. Many of these latter techniques share the concept of a model which is matched to the image, providing a constraint on the segmentation process in a **model-driven** fashion.

3.5.1 Three Dimensional Image Registration as Model-Based Segmentation

Image registration methods that involve local transformations, in addition to global affine ones, provide more of a one-to-one correspondence between positions in images being registered, which makes them applicable to the domain of model-based segmentation. Using higher order transformations, image registration methods can achieve image segmentation and matching [CPDE92, Bcc89]. The essential idea is to register an arbitrary MR image to a standard MR image, the model, which has been labeled in some fashion. The registration provides a mapping from each position and associated classification label in the model to one in a target image, thereby attaching a label to each position in the target image, and effectively segmenting it. This provides a very flexible method of segmentation, in that any individual image may be used as the model image. Furthermore, for each model, several sets of segmented labels may be used, so one registration of an image to the model may provide a suite of several complementary segmentations of the image. Critical to the effectiveness of these methods is the accuracy of the matching between similar structures. In general, these methods work well for large objects such as the four major lobes of the cerebral cortex and some of the regularly shaped anatomical structures such as the thalamus and putamen, but fail to robustly match many of the other structures of the brain, particularly cortical features, due to their inherent variability in size, shape, and topology across subjects. In addition, these methods generally restrict the allowable transformations to continuous deformations of three dimensional space, which limits the ability to represent discontinuities or topological changes between two anatomical images.

3.5.2 Deformable Models

A very promising method of model-based image analysis involves the use of deformable models. A curve, surface, or volume that approximates the target object is deformed to fit an image volume. One of the earliest and most referenced deformable model techniques applied to image segmentation is the **Snakes** method of Kass et al [KWT88]. A snake is a spline curve which changes its configuration to minimize its energy, composed of two opposing terms: an attraction to image features such as edges, and a set of internal forces which constrain shape and position. An initial configuration is subjected to the forces and deformed until it reaches equilibrium where its energy is minimized. A simple formulation of the energy equation to be minimized is

$$E_{snake} = \int_0^1 \alpha(s)|v_s(s)|^2 \ + \ \beta(s)|v_{ss}(s)|^2 \ + \ \gamma(s)I(v(s));$$

where the spline curve is defined by

$$v(s) = (x(s), y(s)), \ 0 \leq x \leq 1,$$

with first and second derivatives, $v_s(s)$ and $v_{ss}(s)$, respectively, and the image data is represented by the function $I(x, y)$. The functions, $\alpha(s)$, $\beta(s)$, and $\gamma(s)$, represent weights that control the relative effects of each of the three terms. A snake subjected to this equation attempts to move itself to areas of minimum or maximum image intensity, depending on the sign of $\gamma(s)$, subject to the constraints imposed by the first two terms. The first term, involving the gradient of the spline curve, $v_s(s)$, makes the snake act like an infinitely thin membrane preferring not to stretch or compress in length. The second term, involving the second derivative, v_{ss}, makes it act like a thin plate, preferring not to bend from the model configuration. The weighting functions, $\alpha(s)$ and $\beta(s)$, control the relative strengths of these two constraints. The ultimate effect of these two terms is to control the amount of stretching/compression and bending of the curve away from a model curve shape. The essential idea is that in areas where the data is ambiguous or ill-defined, the model shape will be imposed upon the snake. In other areas where the image edges are less ambiguous, the snake will be driven by the data. The choice of the weighting functions must be chosen interactively or empirically by the user, a drawback shared with most of the deformable methods. Figure 3.8 portrays a simple image and resulting deformation of a snake. The stretching and bending energy cost of the snake has prevented it from attaching to the small noise-like dot in the upper right of the image. The sensitivity of the snake to noise and small features is dependent on the choice of weights for the regularization terms of stretching and bending.

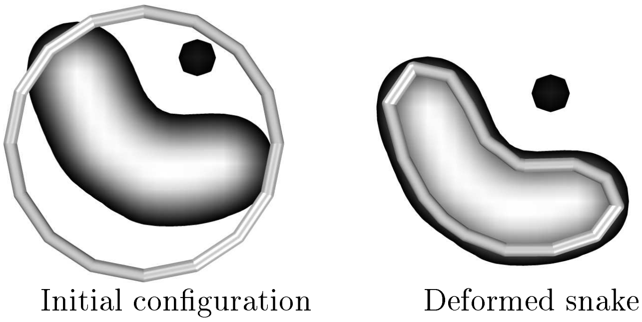


Figure 3.8: Active contour algorithm with snake initialized to be a circle (grey).

Kass et al describe several methods of representing the energy term, including alternate formulations of the image term, $I(v(s))$, involving first and second order edge detection terms, as well as additional constraints such as geometrical proximity forces. A multi-scale approach is suggested, to lessen the sensitivity of the minimization process to local minima, starting with a blurred image and slowly decreasing the blur while the energy of the snake is being minimized. The use of a blurred image is important to increase the range of attraction of the spline to image edges, by increasing the effective width of edges. The actual implementation involves using finite differences to approximate derivatives on a piecewise spline, and the minimization is achieved using implicit and explicit Euler steps in an iterative technique, with order $O(n)$ computations per step, where n is the number of parameters in the spline. Later work by the authors and others have extended these concepts to three dimensions by deforming surfaces to fit two and three dimensional images. The computational complexity of the original Snakes method can be considerable in three dimensions at high resolutions, as a large matrix inverse must be computed at each step. This can be quite expensive, especially when the image forces are quite strong, in which case Kass et al indicate that “the explicit Euler steps of the external forces will require much smaller step sizes" [KWT88]. The many related methods inspired by the ground-breaking Snakes paper use a wide variety of cost functions and minimization techniques, in attempts to optimize the use of active contour models to particular tasks [TWK88, TWK87, CC93, SMG^{+}93b, CHTH93, GA93, HEM92].

The success of the Snakes method as a general purpose model-based image segmentation tool has resulted in its application to a wide variety of domains. The complexity of segmenting neuroanatomical structures from high resolution medical images has motivated creation of many algorithms which can be conceived of as variations on the Snakes method. A survey of some of these most relevant to medical image analysis are presented here, with strengths and weaknesses outlined. A more comprehensive survey may be found in [MT96].

3.5.3 Cohen, Cohen, and Ayache

The authors Cohen, Cohen, and Ayache have several papers describing variations of the Snakes method for segmentation of medical images [CAC91, CCA91, CCA92, CC90, CC93]. The early work of Cohen and Cohen explores the use of a finite element implementation of the Snakes algorithm for curves on slices of MR. Later work adapts the Snakes techniques to three dimensional image segmentation [CCA92]. Again a finite element approach is substituted for the finite difference approach of the original Snakes method, resulting in an improved coverage of the domain of the surface without increasing the number of node points. Their method involves user input of a fairly low resolution initial guess, and produces low resolution surface representations. Use of an automatically generated initial guess would presumably work equally well, and eliminate the need for user intervention. However, its use in an automatic context and applicability to recognizing the highly convoluted and detailed surfaces of the human cortex has not been explored.

3.5.4 Dale and Sereno

Dale and Sereno [DS93] use a simpler, geometric approach to segmenting the human cerebral cortex. MRI data is initially classified into gray and white matter, using a tissue classification algorithm. Using some simple morphological operations, a white matter volume is created which has no topological holes, and whose boundary consists of a two dimensional manifold (isomorphic to a sphere). The boundary of this volume represents the interface between the gray and white matter, and is therefore a good initial guess for locating the gray-CSF boundary, which is the objective of the method. The polyhedral surface of the white matter is then expanded by moving vertices towards the gray-CSF boundary, while also constraining each vertex to be attracted to the centroid of its neighbours. The result is a polyhedral mesh approximating the cerebral cortical surface. While this provides a very high resolution description of the cortical surface with efficient use of computer resources, it has the problem of potentially creating non-simple (self-intersecting surfaces), and does not address the partial volume problem. However, the idea of using the gray-white boundary to guide the search for the cortical surface boundary is an important one that will be revisited later in this dissertation.

3.5.5 Davatzikos and Bryan

Davatzikos and Bryan [DB95] present an active contour method which models the cortical gray matter as a finite thickness sheet. The sheet has a two dimensional parameterization, making it suitable for subsequent morphometric analysis. Rather than parameterizing the complete folded cortical surface, the method parameterizes the outer boundary of the cortex, only entering the upper portion of each sulcus. Location of the depths of the sulcus is performed as a second step, where a curve is initialized at the top of each sulcus and pushed down into the depths by a method similar to the Snakes algorithm. While this method is novel in its attempts to locate the depths of the sulci, the total surface of the cortex is not contained in a single model. Coordinating the two models used, that is, the outer surface model and the set of deep curves, into a single model which faithfully represents the true folded cortex is a nontrivial task.

3.5.6 Sandor and Leahy

Sandor and Leahy [SL97] propose a method to automatically locate the cerebral cortical surface from MR images, and to impose a labeling on the resulting sulci. The authors rely on an edge detection method to define boundaries, followed by a series of morphological operations to create both a smooth voxel representation of the outer limits of the cortex, and a set of voxels corresponding to holes in the volume inside this hull. The assumption is that the holes represent either sulci, noise, or interior brain structures. By choosing the holes that are connected to the outside of the brain, the set of sulci are distinguished from the other two types of holes. A smooth atlas of the cerebral cortex, labeled with points on the extremities of sulci, is warped by a three dimensional Snake method to fit the smooth brain. The labels are then transferred to the smooth brain and to the set of voxels previously labeled as sulci. This method has been demonstrated to label the principal sulci almost completely automatically, but has a few disadvantages in the context of the problem addressed in this thesis. The surface representation does not go very deep into the sulcus, because the authors have chosen to model only the exterior portion of the cortex, stating that "without user interaction deformable models can not be guaranteed to converge to complex and convoluted image features". There are several morphological operators involved in smoothing the volume, which raises the question of how much error is introduced into local boundary positions. Finally, only the parts of sulci that are connected to the exterior are actually labeled as sulci, thus relying on the success of the morphological operator in opening up the entire depths of each sulcus, without modifying the data so much in other areas as to make the results incorrect.

3.5.7 Staib

The work of Staib et al [SD92b, SL, SD92a] uses deformable Fourier surface models composed of sinusoidal basis functions. Different topologies of surface can be modeled with a variable number of parameters, depending upon the resolution desired. Similar to the previous methods, a cost function is devised that is integrated over the two dimensional parameter space of the surface. The data is preprocessed with Gaussian smoothing to reduce the effects of noise, and filtered with a 3 x 3 x 3 Zucker-Hummel operator to create a smooth boundary. Gradient ascent is used to optimize the solution, starting with a rough initial guess. While this method produces models which facilitate shape analysis by encapsulating gross shape into a few parameters, application to the very complex problem of capturing the total cortical surface and deep sulci has not been demonstrated. Correcting deep sulci for partial volume effects and preventing non-simple surfaces is not addressed.

3.5.8 Level Sets

Mallaidi, Sethian, and Vemuri [MSV95] present a novel version of the active contours deformable method. The principal contribution is the re-parameterization of the deforming curve or surface as a level set of a higher dimensionality function. Whereas the original snakes formulation consisted of the points on a parametrically defined curve, $(x(u), y(u))$, the level set method represents the curve by the set of zero points of a two dimensional function, $F(x, y) = 0$. An initial function F is defined, and evolved over time by solving a partial differential equation. The advantage of this formulation is its ability to handle unknown image topologies. The evolving surface automatically splits into multiple components or combines several components into one, depending on the topology of the objects in the image. This provides an efficient solution to the problem of self-intersection avoidance, since level set curves cannot cross, although they can touch. This ability of the algorithm to change the models topology depending on the image data can be an attractive feature in some image segmentation tasks. However, in cases where a specific topology of the object is desired, such as with the cortical surface, this method suffers from the lack of a mechanism to force the algorithm to maintain a particular topology.

3.5.9 Modal Analysis Methods

A considerable amount of work by Pentland has broad application to many vision problems [PHS93, PH91, PS91, PW89, Pen90, Pen89, Pen88, WP92]. These methods generally use a simple object such as a **superquadric** surface to provide concise three dimensional representations derived from typically two dimensional image components. Superquadrics are essentially ellipsoids with a few extra parameters to provide pinched and tapered shapes. Local shape differences are provided by the augmenting of the superquadric models with local offset deformations. Although generally applicable to a wide range of vision problems, the method is suited best for identifying smooth part models from two dimensional images or three dimensional range data, and the segmentation of complex shapes in three dimensional medical images has not been demonstrated. However, related work by Pentland and colleagues [PW89, SP93] in parameterizing shape and measuring shape differences using **modal analysis** has been applied in a medical imaging context. The work of Nastar and Ayache [NA93a, NA93b], uses modal analysis to represent and track two and three dimensional objects that deform over time. Modal analysis uses traditional mechanical engineering techniques to break down shapes into sequences of successively finer modes of vibration. The non-rigid lower order modes can be used to perform matching of similar structures with a very succinct parameter space. Modal analysis can be a powerful tool for subsequent analysis of surfaces segmented from MR images, as well as a technique for matching. However, its applicability to segmenting highly convoluted cortical structures requires more investigation.

3.6 Why Conventional Deformable Surfaces are Insufficient

The current methods of deforming surfaces to fit three dimensional images just described fail to adequately address the problem of creating topologically consistent, simple cortical surfaces with partial-volume corrected deep sulci in three respects,

- possibility of self intersection,
- sensitivity to partial volume effects, and
- inability to accurately represent the total cortical surface.

Firstly, these methods rely on the model shape constraints (stretching and bending) for a "regularization" effect. That is, by keeping the surface more resistant to bending and stretching, it is less likely to wrap over on itself and become self-intersecting. There are two problems with this, one being the fact that this is only an encouragement not to self-intersect, not a guarantee. The second problem is that, as more regularization is imposed, the surface is less likely to deform to fit the particular dataset. A second problem with the existing active methods is that the partial volume effect is not fully addressed. A single surface is used to try to connect together detected edge features essentially using a simple connectivity constraint, based on a continuous surface whose stretching and bending is constrained. However, in order to account for places in the image where gyral boundaries are blurred by undersampling, it would seem that more sophisticated models of the possible configurations of human cerebral cortex are required, in order to infer folded gyral configurations in these areas. A lesser, but not insignificant, problem is the computational tractability of applying existing methods to a surface as complicated as the human cortical surface. Of the methods surveyed, only that of Dale and Sereno makes an attempt to find the entire cortical surface in a high-resolution representation. However, this method provides no enforcement of self-intersection avoidance or explicit strategy to circumvent partial volume effects.

3.7 Numerical Minimization Methods

Many of the algorithms covered so far rely in one form or another on minimization of a multi-dimensional function. Minimization is simply the process of finding a point in the parameter space for which the function value is less than or equal to the function value for all other points in the parameter space. The ability to efficiently find a minimum is critical to many of the methods, and therefore, a survey of some numerical techniques for minimization of functions is presented. A reasonable starting point for practical implementation of standard numerical algorithms is the book, *Numerical Recipes* [PFTV88]. The practical differences between the various algorithms involve whether or not derivatives of the function must be calculated, the amount of storage involved, and the sensitivity of the algorithm to local minima.

3.7.1 Gradient Descent

The most basic method of function minimization consists of simply walking downhill from the current position. This is usually termed **gradient descent** because it involves evaluating the first derivative of the objective function, and stepping in the negative direction until the directional minimum (minimum along a line) is found. Performing this in an iterative fashion will eventually terminate at a point in the domain where the gradient is zero. However, this point may not necessarily be the global minimum, but might be a local minimum, or some other feature, such as a saddle point. It is difficult for any algorithm to avoid local minima, but the feature of being trapped in saddle points and other singularities is a significant disadvantage of the gradient descent method. In addition, depending on the curvature of the function being minimized, the gradient descent method can have a slower convergence rate than other methods.

3.7.2 Conjugate Gradients

The method of conjugate gradients addresses the main problem of more naive approaches that simply follow gradients, that of slow convergence and getting trapped in singularities such as saddle points. Conjugate gradient methods rely on a simple formula to compute search directions based on linear combinations of gradients, but in such a way that the successive search directions are roughly mutually orthogonal. This means that minimizing in a particular conjugate direction does not "undo" the minimization achieved in the previously searched directions. In the case of a quadratic function, this method is guaranteed to find the single global minimum in a number of iterations equal to the size of the parameter space. For more complex functions, it has been found to behave very well, although it can still be trapped in local minima. One particularly attractive feature of many conjugate gradient methods is that the storage involved is linear in the dimensionality of the function. A practical conjugate gradient algorithm may be found in [PFTV88].

3.7.3 Golden Section Search

The Golden section algorithm [PFTV88] is a straightforward method of finding a minimum of a one-dimensional function, and therefore, a method of finding the minimum of a multi-dimensional function along a line. The Golden section search is an iterative algorithm involving shrinking the interval of search until a minimum is found to within some specified tolerance. It is essentially the optimization analog of the binary search method of root finding. The Golden section search is initialized with three points along the line, where the the function value at the inner point is less than that of the outer two. The outer two points define the interval of search for the minimum. An iteration consists of computing the function value at a fourth point within the interval, and replacing one of the two outer points with the fourth point, thus producing a new set of three points with a smaller interval. A method of computing the fourth point based on the outer two and the Golden ratio (the value 0.618034), results in a method where the size of each successive interval is the Golden ratio times the previous. The advantages of this method are that is does not require computation of function derivatives and has a predictable convergence rate, exponentially decreasing the size of the interval. The golden section search is often used as the line minimization component of gradient descent and conjugate gradient methods.

3.7.4 Simplex

The **simplex** [PFTV88] method of function minimization (not to be confused with the simplex method of linear programming) is a geometric approach to finding the minimum of a function. The most attractive feature of the algorithm is that no function derivatives are required to be computed. A simplex is the convex hull of $d+1$ points in d dimensions, where d is the number of dimensions of the domain of the function to be minimized. An iteration of the simplex involves trying one of several different geometric contortions on the simplex to reduce the cost of its maximal vertex. These contortions involve contracting and expanding edges, and the resulting behaviour of the simplex has inspired the algorithm to be referred to as **amoeba**. However, the requirement of quadratic storage space ($d+1$ points of d values each) is prohibitive when dealing with functions where d is very large.

3.7.5 Simulated Annealing

All the numerical minimization techniques described thus far have the trait of searching locally from an initial position, and typically find a local minimum which is very dependent on the initial position. The method of **simulated annealing** introduces randomness into the search, in an attempt to step over local minima and increase the chance of finding the global minimum. Kirkpatrick et al [KGV83] present simulated annealing as an approximation to the process of cooling a physical material from a liquid to a solid. They apply the method to several optimization tasks, including the

Traveling Salesman problem. An initial parameter configuration is randomly modified in an iterative fashion. At first the perturbations of the parameters are quite random, but as the iterations progress, the likelihood of a particular change in configuration is increasingly forced to be proportional to the decrease in function value attributable to the change. The function being minimized is analogous to the energy state of a physical substance, and the increasing preference of the parameters to move towards lower function values is analogous to the increase in probability of the substance to move to a lower energy state as the temperature is lowered. This effectively results in a random search at first, smoothly changing into an approximate local gradient-following search near the end. Although the method of simulated annealing has been used successfully on a number of problems, it has the drawback that the temperature reduction "schedule" must be empirically determined for each problem, so that the temperature is reduced slowly enough to ensure a good chance of finding the global minimum, but fast enough to provide a tractable solution. As the authors say, the choice of this schedule requires "insight into the problem being solved and may not be obvious".

3.7.6 Genetic Algorithms

Another more recently proposed class of optimization methods are termed **genetic algorithms**, whose stochastic nature is closely related to simulated annealing, and likewise has an analogy to a real world process. The principal distinction from simulated annealing methods is the method of generating randomized positions in parameter space from previous ones. Genetic algorithms emulate biological evolution, which attempts to optimize the performance of a species using genetic combinations of existing members to create new ones. A "population" of configurations of the parameters is created as a set of initial guesses to the optimum value of the function. The parameters of the objective function are treated as a sequence of genes, and pairs of members of a population breed by combining their genes into a new sequence using simple rules of gene splicing. Borrowing from nature's rule of survival of the fittest, members of the population which have a lower function value are more likely to thrive and enter into reproductive liaisons with other members. The result is that the population evolves into a set of members which optimize the objective function, and careful selection of the reproductive process increases the likelihood of finding the global minimum. Similar to simulated annealing, this method has the drawback of requiring a reproduction strategy for combining estimates into new estimates, the formulation of which may not be obvious.

3.8 Self-intersection and Proximity

One of the main limitations of contemporary surface segmentation research is that creation of non-simple configurations is not precluded: there is no guarantee that the resulting object does not intersect itself. Detection and prevention of non-simple surfaces is directly related to intersection and proximity testing, an area of extensive interest in computational geometry. Applications include collision detection and path planning in robot movement, VLSI circuit design, and animation. An introduction to many of the data structures and overall strategies for solving these types of problems can be found in [PS85].

One ubiquitous class of algorithms for solving geometric queries is termed **plane-sweep**. Originally formulated for problems in the plane, it involves a vertical line that sweeps across the plane from left to right, encountering various "event" points, which cause a sweep line status to be updated. Hinrichs et al [HNS88] use this method to provide a simple and optimal worst case ($\Theta(n \log n)$) algorithm to find the closest pair in a set of n points in the plane. As the vertical line sweeps across the plane, an ordered list of a subset of the points that are to the left of the plane is maintained. When the vertical line encounters a new point in the set, this point is tested with a subset of the points in the ordered list, to see if a new closest pair has been found. The plane-sweep technique can also be generalized to higher dimensions, and has been applied to problems such as detecting intersection of line segments or rectangles.

The method of **bucketing** or **fixed grid** techniques generally have a poor worst case time complexity, but perform very well in terms of expected time, under non-pathological conditions. Bucketing methods impose a uniformly spaced grid over the space of the problem, and partition geometric data into distinct buckets. Solving problems such as the nearest neighbour to a point in space is a simple matter of finding the relevant bucket and testing against the small number of data points in the bucket [Knu73]. Bucketing methods are simple to implement and generalize easily to higher dimensions.

In addition to these methods, there is a plethora of hierarchical data structures applicable to geometric proximity queries. Some of the more common ones include the multidimensional binary tree (**kd-tree**) [PS85], the **quadree**, and variants thereof [SW, Sam], as well as **binary splitting plane** (**BSP**) trees [FKN80], the **segment** tree, and the **range** tree [Ben79].

A brief survey of published methods of solving proximity problems demonstrates the wealth of methods available for assisting in solving the problem of creating geometrically simple surfaces in the context of segmentation. Dickerson et al [DE96] use Delaunay triangulations to solve various inter-point distances in two or more dimensions. Gupta et al [GJS96] use plane sweep techniques to detect collisions and determine minimum inter-object separation within sets of points, line segments, and axes-parallel hyper-rectangles which are moving along linear trajectories. Canny [Can86a] presents algebraic formulae for detecting collision between moving polyhedra, whereas efficient algorithms for determining minimum distance between circles and line segments in 3-space are presented by Neff [Nef90] and Lumelsky [Lum85], respectively. Gilbert et al [GJK88, GF90] provide a mathematical programming method to find the distance between arbitrary convex sets in any dimensional space. These methods and others may be advantageously applied to the task of surface segmentation of geometrically simple surfaces.

Chapter 4

Edge Detection Algorithms

The formulation of a high level geometric representation of an object in an image relies on the identification of low level salient features. These features may be contours, image gradients, or the result of some image processing algorithm which defines edges or boundaries of objects in an image. The issue of locating boundaries in images, or **edge detection** is dealt with in this chapter. Edge detection is the process of locating positions in an image which are likely to be boundaries between different objects in the scene being imaged. This is an important step in that it generally represents a reduction of the data from a large number of samples to a much smaller number of candidate edges. This process is generally hampered by noise, which may create false edges or change the apparent spatial position of an edge. In the realm of neuroanatomical segmentation, edge detection is often used for the data-driven portion of the segmentation process, in order to provide the higher level model-based method with usable estimates of localized image edges. There is a wealth of published research presenting methods to locate edges in multidimensional images of all types. Here an overview of the methods applicable to three dimensional MR images is presented, followed by a detailed description of one particular method of tissue classification which is chosen for the preprocessing edge detection step in the solution to the MR segmentation task.

4.1 Marr-Hildreth Edge Detection

The work of Marr and Hildreth [MH80] is an often-referenced method of edge detection that is generally applicable to many image processing situations. The basis of their method is that an image should be subjected to a smoothing filter to reduce the effects of noise on the edge detection process. Since edges are effectively areas of change in the image intensity, noise detection algorithms often involve computing image derivatives, which are sensitive to intensity noise. The optimal filter is defined by two conflicting constraints, the requirement of smoothing the image to reduce noise and the desire to minimize the resulting error in the spatial position of reported edges. Marr et al observe that under certain reasonable assumptions about the noise in the image, the use of a Gaussian blurring function,

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma}\, e^{-\frac{x^2}{2\sigma^2}},$$

provides an optimal tradeoff between these two factors. An edge occurs where there is a peak in the first directional derivative of the smoothed image. This corresponds to positions where the second directional derivative changes from positive to negative, which is a subset of what are termed **zero-crossings**, where the second directional derivative changes sign. The second directional derivative of a Gaussian function is convolved with the image, and zero values in the resulting image are used to define edges. This procedure is applied to an image several times, using different filter widths, to capture sets of edges that occur at various spatial scales. The results are combined by choosing edges that occur at two or more neighbouring scales. This method, commonly referred to as **M-H edge detection**, has become a widely referenced technique in medical imaging, and in image processing literature in general.

4.2 Canny Edge Detection

The method of edge detection by Canny [Can86b] is similar to that of Marr-Hildreth and has also been applied to a variety of image processing tasks. The two M-H criteria for an optimal filter are augmented with a third constraint. Similar to the M-H criteria, there must be a low probability of false positives and false negatives with respect to the reporting of edges, and the positions of edges reported by the algorithm must be as close as possible to the true edge. As a third constraint, there must be only one response from each edge. The mathematical expression of these three criteria results in a numerical optimization problem that attempts to maximum the signal-to-noise ratio, while minimizing the estimated error between reported and actual edge positions, and minimizing the likelihood of multiple responses from single edges. For a given expected type of edge, a filter can be found which optimizes the three criteria. An image is convolved with this filter, which effectively produces a smoothed gradient image, where local maxima are assumed to be edges. For detection of sharp edges, or **step** edges, Canny found that the optimal filter is very similar to the first derivative of a Gaussian. Therefore, for practical edge detection in images of two or more dimensions, Canny, like Marr and Hildreth, also convolves the image with a Gaussian function, then computes directional second derivative zeros in order to mark the edges. The magnitude of the gradient is used to estimate the edge strength. This method is therefore quite similar to that of Marr and Hildreth, but Canny has extended his method to handle variable signal-to-noise ratios throughout the image, by locally adapting the filter. The sensitivity and generality of the method has made Canny edge detection a popular choice for multidimensional image processing.

4.3 Monga, Deriche, and Rocchisani

The Canny edge detection method was originally applied mostly to two dimensional images, and the complexity of the algorithm limited its applicability to high resolution three dimensional images. Monga, Deriche, and Rocchisani [MDR91] present a variation of Canny edge detection optimized for the case of three dimensional images, in an attempt to produce a more computationally tractable method for processing of medical images. Their method changes the boundary conditions of Canny's filters and results in a different filter type. Application in three dimensions starts by creating the gradient image with three passes of one dimensional filters. Local gradient extrema are identified by approximating gradients at a voxel using the neighbouring voxel values in a finite difference fashion. Finally, a method of choosing the subset of the extrema which correspond to edges is applied, based on locating areas of high gradient magnitude, subject to connectivity constraints with other areas of high gradient magnitude. The authors have demonstrated implementation of their algorithm on three dimensional MR images of the human heart, which suggests the amenability of this method to edge detection of MR images of the human brain.

4.4 Anisotropic Diffusion Filters

The method of **anisotropic diffusion filtering** [PM90, GKKJ92] differs from other edge detection methods in that it varies the smoothing of the image depending on proximity to edges. The essential idea is to smooth within homogeneous regions, rather than across them. This is achieved by diffusion algorithms which blur local regions by a variable amount, related to the magnitude of the gradient. Areas of high gradient magnitude are assumed to be near edges, and the amount of smoothing performed is minimal, whereas near weaker edges or interior to objects, higher amounts of smoothing are performed. Application of anisotropic diffusion filters involves iteratively diffusing the entire image, until convergence is achieved, where the amount of change between iterations is below some tolerance. The effect of the adaptive smoothing is similar to that of the previously described methods, in that it attempts to preserve the spatial location of edges, while applying some amount of smoothing to them.

4.5 Tissue Classification as Edge Detection

The methods of edge detection just discussed have wide applicability and have been used successfully in medical imaging contexts. However, magnetic resonance imaging has the relatively unique characteristic of being able to generate multiple images of a single object with different tissue contrasts. In chapter 3, a brief overview of methods which take advantage of this feature was presented. Tissue classification algorithms use T1-weighted, T2-weighted, and PD-weighted MR images to classify voxels into a small set of neuroanatomically based types, including gray matter, white matter, and CSF. In effect, this is a type of implicit edge detection that is optimally tuned for magnetic resonance imaging. Edge features can be considered to exist on the boundaries between any two adjacent voxels which have differing tissue classes. An advantage of this over more general edge detection methods is that edges are not only identified, but also classified into different types, such as gray-CSF and gray-white edges. Based on the success of existing MR tissue classification algorithms, the remainder of this dissertation will assume that the input images have been preprocessed with an established tissue classification algorithm. This convenient assumption is actually not a very limiting restriction, as it can be shown that any of the general methods of edge detection may be quite easily incorporated into the overall method of MR image segmentation presented here.

The actual method of tissue classification is an implementation of the work of Zijdenbos et al [ZDM93, ZER$^+$96]. This method approaches tissue classification as a feature matching process using a training set, with no user intervention required. Before classification, MR volumes are registered into a standard stereotaxic coordinate system using an affine image registration algorithm described in chapter 3 [CNPE94]. A volume containing the probability of each voxel being each of the various tissue classes has been generated from a sample of several hundred MR volumes which have been classified by a semi-automatic method. The training set is defined as a set of voxel positions in this composite image (**probabilistic volume**) with high probabilities of particular tissue types. These voxel positions are then used to derive a sample of image values from the T1-, T2-, and PD-weighted MR images of a particular target individual. The image values from the three target volumes and the corresponding tissue classes from the probabilistic volume are used to initialize an artificial neural network. Each voxel in the target image is then independently classified by the neural network into one of the defined tissue types, resulting in a discrete three dimensional volume, where each voxel has one of a small number of integer labels. Figure 4.1 depicts the results of this tissue classification algorithm on a typical MR volume set.

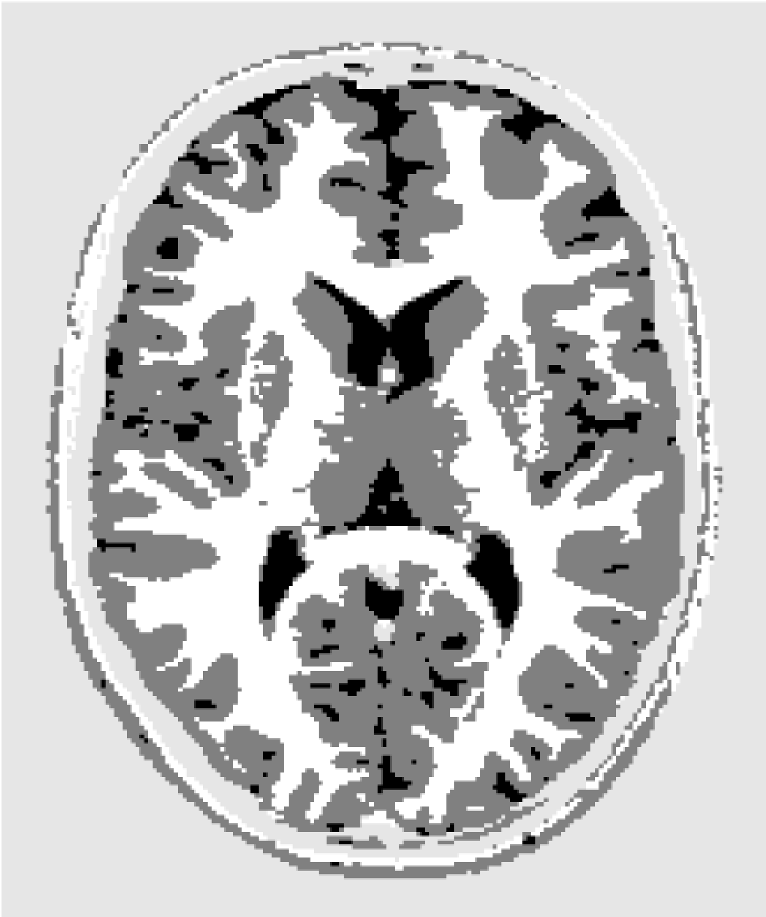


Figure 4.1: One slice through a tissue-classified volume. Areas of changes in class type are considered to be edges (white matter is white, gray matter is gray, CSF is black, background is off-white).

This tissue classification method relies on a preprocessing step that reduces the effects of RF inhomogeneities in the three dimensional volume. The method of Sled et al [SZE97] assumes that the effect of RF is an intensity scaling that is spatially slowly varying. Consequently, the RF can be estimated from localized histograms and modeled as a smooth spline function. The resulting estimate is subtracted from the MR image to produce an image with markedly reduced inhomogeneity. Sled et al perform experiments to test the effect of RF correction on tissue classification, image registration, and surface extraction. In all three cases, the sensitivity of the algorithms to RF inhomogeneity is greatly reduced when the data is first preprocessed with the RF inhomogeneity reduction algorithm.

Chapter 5

Solution: Non-intersecting Object Deformation Environment (NODE)

5.1 Overview

A novel method for creating geometrically simple representations of cortical surfaces from MR images is presented here. The foundation of the technique is a surface deformation approach similar to the Snakes algorithm of Kass et al. The method proposed is novel in that it allows simultaneous identification of multiple inter-related surfaces combined with constraints which guarantee geometrically simple surfaces. Homology between model points and deformed points can be constrained through the use of curvature-based shape matching. The use of multiple surfaces will be shown to provide some correction for partial volume effects by incorporating more neuroanatomical *a priori* information into the model constraints. Specifically, inter-surface proximity constraints are used to specify relationships among multiple surfaces of the segmentation process and to prevent two surfaces from intersecting each other. Intra-surface proximity constraints prevent individual surfaces from becoming non-simple. An objective function with intuitive user-chosen weighting parameters is devised to measure how closely a particular surface or set of surfaces simultaneously approximates the image data and adheres to a set of model constraints. The parameter space imposed by the surfaces involved is searched to find a minimum of the objective function, and the corresponding surface is output. Because this technique provides a framework for deformation of objects with intersection constraints, it is referred to as the **Non-intersecting Object Deformation Environment** (**NODE**) algorithm.

5.2 Representation

The choice of surface representation for the solution presented in this dissertation is a very straightforward decision. All surfaces are represented here as arbitrary polyhedra, for many reasons. Most computationally tractable surface proximity algorithms are polyhedron-based. Using more complex representations, even such prevalent ones as piecewise spline patches, would greatly increase the complexity of computing proximities between parts of surfaces. Similarly there is a wealth of algorithms for dealing with polyhedra, although this is of lesser importance, as many other surface representations also have an extensive history of practical use. More software supports polygon and polyhedron formats than any other type, so this choice of format is practical from the point of view of data interchange and flexibility of analysis. In addition, the choice of polyhedral surface representation is not very limiting because algorithms exist to convert to many other representations, including piecewise spline patches, superquadrics, and spherical harmonics. Finally, most implementations of deformable non-polyhedral surface representations approximate the integral of the image component of the cost function as a set of discrete points, because closed form solutions of surface integrals of functions of MR images are generally not feasible. Evaluating at discrete points on the surface is essentially the same as using a polyhedral representation and evaluating at vertices, so in this respect, nothing is gained in using a more sophisticated representation.

5.3 Objective Function

The objective function is simply a scalar goodness-of-fit measure for a given configuration of surfaces. Its domain is therefore the set of parameters of the surfaces involved, and its range is the set of real numbers. The objective function is a weighted sum of various model and data components, each of which is described in a subsequent section. The objective function, $O(S)$, may be defined generally as a weighted sum of N_t terms, each of which may be thought of as a data or model term:

$$O(S) = \sum_{k=1}^{N_t} w_k T_k,$$

where S is a set of N_s polyhedral surfaces:

$$S = \{S_i \;:\; S_i \text{ is a polyhedral surface, } 1 \leq i \leq N_s\},$$

w_k is a weighting factor, and T_k represents one of the terms defined in the following sections. Before describing each of the possible objective terms, some definitions are presented:

$\bar{x}_v = (x_v, y_v, z_v)$,	the 3D position of vertex v in a deforming polyhedral mesh,
$\bar{\hat{x}}_v = (\hat{x}_v, \hat{y}_v, \hat{z}_v)$,	the 3D position of vertex v in a static model polyhedral mesh,
n_v ,	the number of vertices in a polyhedral mesh.
n_p ,	the number of polygons in a polyhedral mesh. and
m_v ,	the number of neighbours of vertex v,
$n_{v,j}$,	the j'th neighbour of vertex v, and
$\bar{N}_v$,	the surface normal at vertex v, defined as the unit normal to the polygon consisting of the counterclockwise ordered neighbours of the vertex.

5.3.1 Image Value

The image value term represents the proximity of the surface to edges in a particular image dataset. This term decreases as the surface approaches the edges in the image, which are defined as the contours of a user-defined threshold:

$$T_{image} = \sum_{v=1}^{n_v} (V(\bar{x}_v) - t)^2,$$

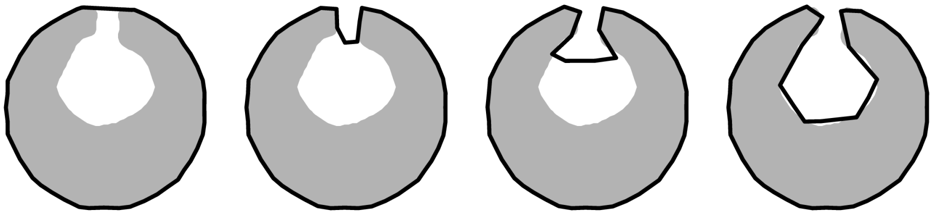
where t is the threshold denoting the image value that best defines the image edges, and $V(x, y, z)$ is the continuous function representing the three dimensional image volume. As suggested in the original Snakes paper, and subsequent research [Bcc89, VRL93], image blurring may be performed in order to reduce the effects of local minima by smoothing the cost function. Rather than fitting the surface to a particular contour, it is also possible to have the surface attracted to the lowest values in its vicinity, by setting the threshold, t, to a value less than or equal to the lowest value in the volume. Similarly, one could set the value of t to a very large value and use a gradient magnitude volume to define V, in order to fit the surface to the edge defined as the maximum local gradient magnitude.

5.3.2 Image Boundary Distance

In order to increase the power of locating image boundaries that are relatively far from the current surface position, an alternate image term is introduced. This term is based on the distance of a vertex from the nearest image boundary in the direction of the surface normal, and is defined as

$$T_{boundary\_dist} = \sum_{v=1}^{n_v} d_B(\bar{x}_v, \bar{N}_v, t)^2$$

where $d_B(\bar{x}_v, \bar{N}_v, t)$ is the distance to the nearest image contour of the threshold, t, from the vertex, v, along the line defined by the surface normal, $\bar{N}_v$. The search is performed along the surface normal in both directions from each vertex. In practice, this term also requires a value for the maximum search distance, after which unsuccessful searches are truncated and the value of d_B is set to the maximum distance. The choice of the maximum search distance controls the range of attraction that image edges will have on surface vertices. The intention of this term is to reduce



a) b) c) d)

Figure 5.1: a) Result using image value term. b) Result using simple boundary distance. c) Intermediate step where objective function value has increased. d) Result using modified boundary distance term.

the frequency of local minima in the cost function, by increasing the distance across which a surface may be attracted to an edge.

Comparison of Figs. 5.1a with 5.1b illustrates how the $T_{boundary\_dist}$ term represents an improvement over the T_{image} term. However, the method as described is still insufficient for accurately locating the depths of sulci. The reason is that Fig. 5.1b corresponds to a local minimum in the objective term, $T_{boundary\_dist}$. For the surface to pass from this configuration to the final configuration in Fig. 5.1d, it would have to pass through a configuration similar to Fig. 5.1c, which has a higher objective function cost because more vertices are far away from the image boundaries. The correct result presented in Fig. 5.1d is achieved through modifying the boundary search mechanism.

The modification of the search mechanism involves performing the search for image boundaries at fixed intervals during the deformation, rather than every time the objective function is evaluated during minimization. Each iteration consists of computing the nearest image boundary point for each vertex, then taking a few minimization steps of the resulting objective function while holding the image boundary points constant. While this will cause the surface to follow convoluted sulci all the way into the depths, it introduces a new problem. The successive states of the objective function are no longer monotonically decreasing, as it is possible for the function to temporarily increase, before decreasing again. This means that termination criteria for the minimization can not depend solely on changes in function value, but must also include some other criterion, such as distance moved per iteration, or maximum distance of a vertex from the boundary. The disadvantage of an objective function that does not monotonically decrease is far outweighed by the advantage of this algorithm to accurately locate deep and convoluted sulci.

5.3.3 Stretch

One of the terms that imposes model constraints is the stretch term, which increases as lengths between vertices are stretched or compressed relative to a user-defined model surface representing the ideal lengths. The term represents a normalized deviation from this ideal:

$$T_{stretch} = \sum_{v=1}^{n_v} \sum_{j=1}^{m_v} \left(\frac{\sqrt{(x_v - x_{n_{v,j}})^2 + (y_v - y_{n_{v,j}})^2 + (z_v - z_{n_{v,j}})^2} - L_{v,j}}{L_{v,j}} \right)^2,$$

where $L_{v,j}$, the ideal length of an edge, is defined as the corresponding length in the model polyhedron:

$$L_{v,j} = \sqrt{(\hat{x}_v - \hat{x}_{n_{v,j}})^2 + (\hat{y}_v - \hat{y}_{n_{v,j}})^2 + (\hat{z}_v - \hat{z}_{n_{v,j}})^2}.$$

The intention of this term is to make distances between corresponding pairs of vertices on the model and deformed surface roughly equivalent. This term is analogous to the term involving the magnitude of the first derivative of the spline in the original Snakes formulation, which makes the snake act like a membrane which constrains stretching and compression.

5.3.4 Curvature

The other term that provides a model-based shape constraint is the curvature term, which controls the amount of bending of the surface away from the model shape. On curves, curvature is defined as the reciprocal of the radius of an inscribed circle at a point on the curve. The curvature at a point on a surface is usually represented by two numbers, the mean curvature and the Gaussian curvature, which are based on the minimum and maximum curvatures defined by cross sections on osculating planes through the surface at that point. For the purposes of controlling surface shape, a more easily computed scalar measure of deviation from flatness is sufficient for representing the curvature at a vertex of a polyhedral surface. The measure of curvature is based on the ratio of height to base length of the pyramid-type structure defined by a vertex and its neighbours. The signed perpendicular height of the vertex above the plane of best fit through its neighbours is divided by the average distance between the neighbours and their centroid to produce a curvature estimate. Vertices in flat areas have a value near zero, convex vertices have positive values, and concave vertices have negative values. Similar to the stretch term, the curvature term measures deviation of surface vertices from the curvature, $\hat{C}_v$, at the corresponding vertices of a model surface:

$$T_{curvature} = \sum_{v=1}^{n_v} \left(\frac{(\bar{x}_v - \tilde{x}_v) \bullet \bar{N}_v}{B_v} - \hat{C}_v \right)^2$$

where $\tilde{x}_v$ is the centroid of the neighbours of vertex v, B_v is the average distance from the centroid to each of the neighbours, $\bullet$ is the vector dot product operator, and $\hat{C}_v$ is the curvature computed similarly from the model surface. This term is analogous to the term involving the second derivative of the spline in the original Snakes formulation which causes the spline to function like a thin plate, constraining the bending of the spline.

The curvature term is relatively easy to compute, but assumes that the surface consists of a mesh isomorphic to a sphere. In order to handle other cases, such as deforming sheets or more complicated topologies, one could devise other measures of curvature. One simple possibility is to measure the angle between every pair of triangles that share an edge. Similar to the curvature term, the bending term would assign a higher penalty to angles which deviate further from the model configuration. For the application to cortical surface segmentation, closed surface models are used, and alternate forms of the bending term are not explored further.

5.3.5 Vertex-to-Point Constraints

Another method of guiding the deformation is to constrain a particular vertex to remain close to a specified distance from a fixed point:

$$T_{anchor} = (\sqrt{(\bar{x}_v - x_A) \bullet (\bar{x}_v - x_A)} - d_A)^2$$

where x_A is a user specified anchor point, and d_A is the preferred distance that vertex v should be from the anchor point. A value of zero for d_A results in a vertex being attracted to the point. In practice, this term may be used to guide a specified portion of a deforming surface towards a particular neuroanatomical landmark of interest, or to allow a user to interact with a surface.

5.3.6 Vertex-to-Vertex Constraints

A similar constraint to anchoring a vertex to a fixed point is to constrain two vertices to be a preferred distance apart:

$$T_{vertex-vertex} = (\sqrt{(\bar{x}_v - \bar{x}_w) \bullet (\bar{x}_v - \bar{x}_w)} - d_B)^2$$

where d_B is the preferred distance between vertex v and vertex w. Again, this can be used either to keep two vertices close together or far apart. In the case where vertex v and w correspond to neighbouring vertices in a polyhedron, this term becomes equivalent to the stretching term, $T_{stretch}$. A more interesting situation occurs where the pair of vertices belong to two distinct surfaces, in which case the $T_{vertex-vertex}$ term may be used to maintain a specified distance between two surfaces.

5.3.7 Self-Intersection Constraints

In order to maintain a well-behaved surface during the deformation process, it is critical to define self-intersection constraints on each surface. Prevention of self-intersection is achieved by associating a high cost with non-simple topologies and configurations where non-adjacent polygons are in close proximity. This maintains a smooth objective function that increases in cost as parts of the surface move closer to intersection. The formulation of this term therefore measures the distance between every pair of polygons in a polyhedral mesh,

$$T_{self-intersect} = \sum_{i=1}^{n_{p-1}} \sum_{j=i+1}^{n_p} \begin{cases} (d_p(P_i, P_j) - d_{i,j})^2, & \text{if } d_p(P_i, P_j) < d_{i,j} \\ 0, & \text{otherwise,} \end{cases}$$

where $d_p(T_i, T_j)$ is the smallest Euclidean distance between the i'th polygon, P_i, and the j'th polygon, P_j, and $d_{i,j}$ is a distance threshold. In practice, pairs of adjacent polygons are not included in the above equation, as their $d_p(P_i, P_j)$ and $d_{i,j}$ must both be zero. Figure 5.2 illustrates the value of the self-intersection term as a function of the distance between two polygons. The function is zero whenever the two polygons are greater than $d_{i,j} = 5$ millimetres apart, but increases to a prohibitive cost as the distance closes to zero. If this term has a sufficiently high weight associated with it and all values of $d_{i,j}$ for non-adjacent polygons are positive, then geometric simplicity of the surface is maintained throughout the deformation. This term could also have been based on the more easily computed distance between vertices or between a vertex and a polygon, but the distance between polygons was chosen because the former two distance constraints cannot, in general, prevent self-intersecting topologies.

5.3.8 Surface-to-Surface Intersection Constraints

Similarly to the self-intersection constraints, the distance between any pair of surfaces may be constrained:

$$T_{self-intersect} = \sum_{i=1}^{n_p} \sum_{j=1}^{n_q} \begin{cases} \left(d_p(P_i, Q_j) - \hat{d}_{i,j}\right)^2, & \text{if } d_p(P_i, Q_j) < \hat{d}_{i,j} \\ 0, & \text{otherwise,} \end{cases}$$

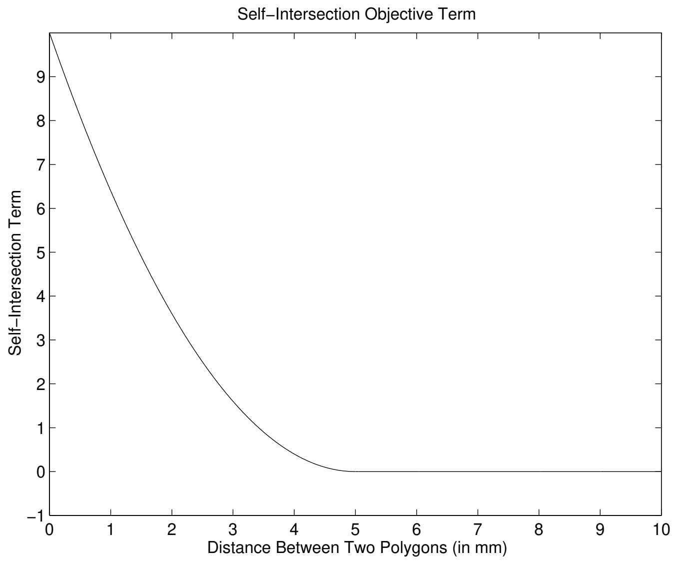


Figure 5.2: The contribution to the self-intersection objective term as a function of distance between two polygons in a polyhedron.

where n_p and n_q are the number of polygons in the two surfaces, $d_p(P_i, Q_j)$ is the distance between a polygon on one surface and a polygon on the second surface, and $d_{i,j}$ is the threshold distance below which two polygons contribute positive values to the objective function being minimized. With an appropriate weight and values of $\hat{d}_{i,j}$, this term may be used to prevent inter-surface intersection or to maintain a specified minimum inter-surface distance.

5.4 Elaboration on Objective Terms

For reasons of clarity, the definitions of objective terms have been presented in a somewhat simplified form. However, in practice, it is possible to involve more than one data volume in the objective function, as well as multiple distinct surfaces, and even many stretch and curvature models for each surface, each with its own set of weights and other parameters. In addition, the formulation of some of these objective terms is developed further in the following sections.

5.4.1 Directional Edges

Examination of human neuroanatomy and three dimension MR images reveals that there are often several candidate boundaries near any given point in the image. In order to increase the likelihood of the minimization procedure choosing the correct edge, the boundary search term is refined to include a directional constraint based on the orientation of local image gradients. In addition to choosing a threshold for each desired edge, a direction parameter, g, either positive or negative, can also be specified to constrain the search to areas where the dot product of the image gradient and the outward surface normal is less than or greater than 90 degrees, respectively. This requires a modification of the distance calculation in the boundary distance term,

$$T_{boundary\_dist} = \sum_{v=1}^{n_v} d_B(\bar{x}_v, \bar{N}_v, g, t)^2,$$

where d_B now finds the distance to the nearest image contour along the search direction, N_v, ignoring those points where the gradient is facing the wrong way with respect to N_v, as defined by the sign of g. This refined boundary search constraint is useful in differentiating cerebral cortex edges in T1-weighted images, which have inward facing image gradients, from the inside skin edges, which are relatively close to the cortex in places, but have outward facing image gradients.

5.4.2 Differential Weights

Until now, the weights have been specified on a per-term basis, but it is also possible to vary the weight of a term depending on various criteria. For instance, the image boundary distance term may have a different weight depending on whether the boundary was found inside or outside the surface, to allow a preference for the surface to be just outside the image edges. In the case of cortical surface extraction, it may also be appropriate to allow a surface to bend in the concave direction more easily than in the convex direction, in order to better interpolate deep sulci. One other way to apply differential weights is to effectively impose a maximum deviation from ideal, by attaching a higher weight once the deviation increases past some point, similar to the formulation of the self-intersection term described previously. This allows imposition of such constraints as maximum stretch or curvature deviation from a model surface. The general form of such a term is:

$$T_{max} = \begin{cases} (m_{actual} - m_{ideal} - max\_diff)^2, & \text{if } m_{actual} - m_{ideal} > max\_diff \\ (m_{actual} - m_{ideal} - min\_diff)^2, & \text{if } m_{actual} - m_{ideal} < min\_diff \\ 0, & \text{otherwise.} \end{cases}$$

where m_{actual} and m_{ideal} are the appropriate actual and ideal measures from one of the previously described objective terms. Terms of this type contribute nothing to the objective function unless the difference between the actual and ideal is outside the range $[min\_diff, max\_diff]$.

5.4.3 Choice of Weights

A typical objective function may consist of several different weights and parameters, each of which must be chosen in some way. Due to the generality of the objective function terms and the dependence on the specific segmentation task and data, there is no theoretically grounded method to choose good weights. However, the weighting parameters provide direct control on the tradeoffs between the various objective terms and therefore represent an intuitive mechanism for constructing surface deformation tasks. Experimentation with the various parameters is performed in a subsequent chapter. In order to make the weights independent of surface sampling and volume data scaling, each weight may be normalized by the number of vertices or polygons, or volume data range, as appropriate.

5.4.4 Continuity of Objective Function

The objective terms presented have generally taken the form of a squared residual, not only for reasons of computational efficiency, but also to provide a smooth objective function. The large number of parameters involved in each surface, combined with the complexity of the image data, introduce the possibility of many local minima in the objective function. Using squared residuals as the basic components of the objective function helps keep the objective function C^1 continuous, that is, both function value and gradient are continuous. However, it must be noted that the boundary search term is potentially discontinuous, because an infinitesimal change in the surface normal may cause the distance to a boundary to have a discontinuous change in value.

5.4.5 Oversampling

The objective terms described thus far have typically been evaluated on a per-vertex basis, which effectively samples the objective function at the resolution of the surfaces involved. However, most of the terms, particularly the image terms, admit the possibility of being sampled other than at the vertices, in general, on an arbitrary sampling over the surface. This can prove useful in preventing the deforming surface from missing data small relative to the spacing of vertices, while not increasing the number of surface parameters over which minimization is performed. In particular, oversampling the image value term and the boundary search term over all the triangles in the surface can be an advantage in fitting a surface to an image.

5.5 Minimization

A typical MR volume consists of over 10 million voxels, and the surface required to represent an anatomical object spanning most of the volume requires a large number of parameters. For the cortical surfaces represented by polyhedral surfaces, the number of vertices is on the order of 100 000, resulting in 300 000 or so parameters over which minimization occurs. It is computationally intractable to perform an exhaustive search of this parameter space, so minimization is achieved by starting with a good initial guess, performing a multi-scale search, and using a minimization method that searches locally from the current position in parameter space. The initial guess is a surface that is expected to be close to the correct answer, and the method of choosing the initial guess depends highly on the application context. The multi-scale concept is widely prevalent in the literature of objective function minimization and in this case, involves starting with a coarse under-sampled set of surfaces, and iteratively minimizing the objective function, while increasing the sampling of the surfaces.

5.5.1 Conjugate Gradient Minimization

The conjugate gradient method of function minimization is a robust and efficient algorithm based on using the first derivative of the objective function to direct the search. It is chosen in this case because it is usually not compromised by singularities such as saddle points. In addition, its storage requirement is linear in the number of dimensions of the objective function, which is important when dealing with hundreds of thousands of dimensions. There are multiple variations of the conjugate gradient method, but the one chosen here is the straightforward algorithm described in [PFTV88].

5.5.2 Termination Criteria

As mentioned in section 5.3.2, the termination criterion can not be simply to stop when the objective function stops decreasing by a threshold amount. Since the objective function can increase and decrease somewhat while the surface is far from the image boundaries, it is necessary to devise other termination criteria. One obvious method that is often used in dynamic systems where no explicit objective function exists is to stop when the movement of the parameters decreases below some level. In this case, one could use the average or maximum distance moved by the vertices to decide when to stop the minimization. Another termination criterion is to stop when all vertices (or oversampled points) on the deforming surface are within a specified distance from an image boundary. This criterion is best applied when there is little chance of holes in the image where it is impossible for a single closed surface to conform exactly to the image. For instance, deforming a sphere to fit an image of a torus could never bring all points on the surface close to the image boundaries, due to a mismatch of topologies between the data and the model. Therefore, it is better to use a combination of two termination criteria where the algorithm stops when the motion of the surface vertices decreases below some level, or all points are within some specified distance of image boundaries.

5.5.3 Multi-scale Approach

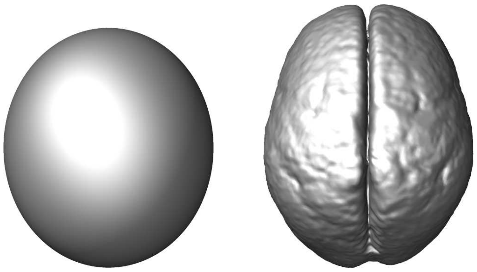
In order to speed up convergence of deformation problems, a multi-scale approach is often employed [MH80, Can86b, CEHP95, Bcc89, VRL93]. The intended effect is that the initial stages generate a rough overall fit to highly blurred data for relatively little computational effort, with the effects of noise and small image features being ignored by the low resolution surfaces. As the iterative decrease in scale progresses, the surfaces interpolate smaller features in the data, until the desired amount of detail is achieved, modulated by the amount of computation time allowed. In implementing the multi-scale approach for the surface deformation described herein, it was found more expedient not to blur the target volume. If blurring is performed, narrow sulci may only show up at relatively high resolutions, at which point the computational cost of deforming the surface into a deep sulcus is much higher than if the sulcus can be located when the surface has fewer parameters. The use of oversampling of the boundary function on the deforming surface provides a similar averaging advantage to that arising from using blurred volumes, which effectively compensates for the lack of blurring.

5.6 Creation of the Cortical Surface Model

Critical to the success of a deformable surface is the model used to constrain the segmentation process. Here, the model may be used as both the initial guess and as the shape model during the deformation. The model should capture the shape and position of the typical object being segmented. In the case of the human cerebral cortical surface, a model was created as follows. A set of 53 normal individual MR volumes were registered into a stereotaxic coordinate system, by an automatically computed, linear transform. This factors out rotations and translations, as well as scaling in each of the three coordinate dimensions. A voxel-by-voxel average was performed in Talairach space to create an average MR volume, depicted in Fig. 5.3a. This volume was then thresholded using an empirically chosen threshold to create a binary volume, shown in Fig. 5.3b. The binary volume was then segmented by hand on a voxel-by-voxel basis, to remove the cerebellum, brain stem, skin, and other non-cortical features, with the result depicted in Fig. 5.3c. The surface of this segmented, binary volume was then assumed to be representative of the size and shape of the typical cortical surface in Talairach space. The final step was to create an explicit surface model from the segmented volume. This was accomplished by creating an ellipsoid that had the rough size and shape of the cortex (Fig. 5.4a), then applying the surface fitting method described previously to deform the ellipsoid to fit the boundaries in the segmented volume. The resulting model is shown in Fig. 5.4b, and is used as the initial model for much of the validation discussed in subsequent chapters. This model encapsulates the gross shape, orientation, and position of all normal brains mapped into stereotaxic space. Accordingly, it represents a good starting model for deforming to capture any individual brain, as well as a frame of reference for investigating local anatomical variations.

a) b) c)

Figure 5.3: a) Cross section of average of 53 normal individual MR volumes. b) Cross section of thresholded average. c) Cross section of segmented thresholded average.



a) b)

Figure 5.4: a) Ellipsoid before deformation to the segmented average MR volume. b) Ellipsoid after deformation.

Chapter 6

Basic Tests of Deformation

As a prelude to exploring the application of the deformation method to realistic data, a series of short examples are used to test and demonstrate some of the features of the NODE algorithm. The datasets used are fabricated to be small and simple, in order to focus on one aspect of the deformation process. The effect of each of the components of the cost function is dealt with separately, followed by demonstrations of how several components are used together to achieve specific results. It is important to note that all tests are performed with closed polyhedral surfaces and three dimensional volumes, but that the results are often presented visually in the form of two dimensional cross-sections in order to minimize confusion caused by three dimensional image complexity.

6.1 Boundary Interpolation using Image Value

The essential task is to fit a deformable surface to the boundary of an object apparent in an image. One of the cost function components that can be used to locate boundaries is the image value term, T_{image}, where the difference between a constant and the image value at each vertex is to be minimized. As with other deformable methods, the use of this term in practice requires that the image be smoothed somehow. Discontinuities and sharp edges in the image decrease the likelihood that a vertex will be able to follow the gradient to reach the desired threshold. Figure 6.1 illustrates a cross section of the initial configuration of a surface and volume, and the final position after deformation using the image value component. The final position has not changed the surface at all, because the vertices are positioned so far from the boundary that they are on an image value plateau, and local gradient information is insufficient to indicate the direction to the boundary. Most methods of deformable surface involve blurring the image to avoid such image plateaus. Figure 6.2 depicts the same situation as the previous figure, except that the image has been blurred with a three dimensional box filter, that is, each filtered voxel value is the average of the original values within a rectilinear box centred at the voxel. The result on the surface deformation is that all vertices are successfully moved to the desired boundary position. In effect, the blurring of the data increases the distance of attraction of vertices to edges. This need for increasing the region of attraction around edges by blurring or other types of transformations is observed in much of the deformable model research, notably the seminal Snakes paper [KWT88].

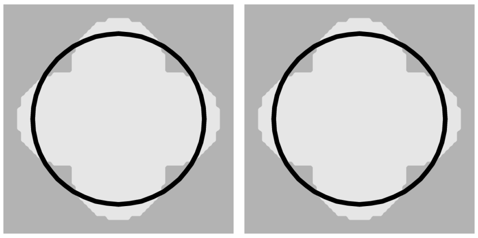


Figure 6.1: Cross sections of initial and final surfaces using image value term on non-blurred data.

6.2 Boundary Interpolation using One Dimensional Search

The failure of the image value term to locate edges relatively far from the surface vertices without blurring is addressed by the other cost function term that deals with

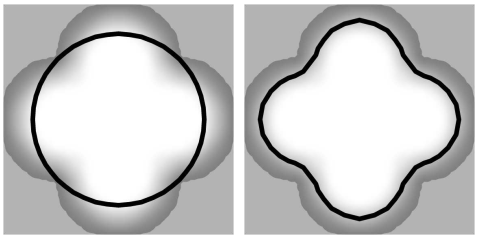


Figure 6.2: Cross sections of initial and final surfaces using image value term on blurred data.

boundary location, $T_{boundary\_dist}$. The boundary search term involves searching inward and outward along the surface normal to locate an image value equal to the desired threshold. In order to compare its behaviour to that of the image value term, it has been applied to the same two datasets as in the previous section. Figure 6.3 illustrates that the boundary distance term correctly locates boundaries that are a significant distance from the initial position of the vertices, where the T_{image} term failed to do so. Figure 6.4 shows that in the case of blurred data, the result is equivalent to the image value method, as would be expected. This indicates that the use of the $T_{boundary\_dist}$ term obviates the need for image blurring to increase the distance of attraction of edges.

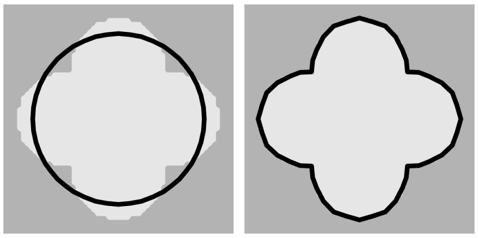


Figure 6.3: Cross sections of initial and final surfaces using boundary search term on non-blurred data.

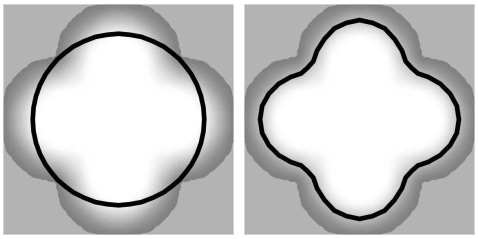
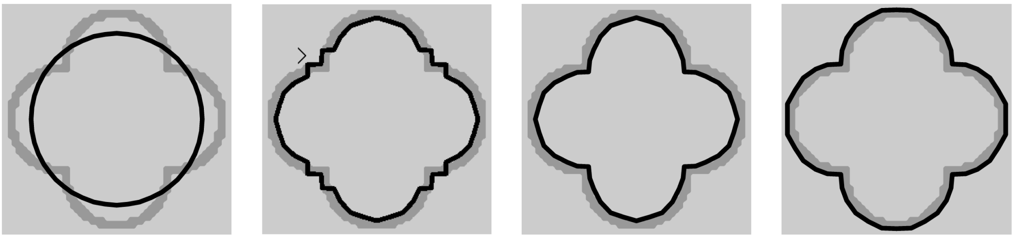


Figure 6.4: Cross sections of initial and final surfaces using boundary search term on blurred data.

6.3 Boundary Direction Constraints

The refinement of the boundary search term, $T_{boundary\_dist}$ to include gradient direction information is useful to distinguish between neighbouring edges. Figure 6.5a illustrates the cross section of an initial model and an image volume, which has two concentric sets of edges. Figure 6.5b shows that if no directional constraint on the boundary search is used, the model is incorrectly deformed to fit parts of both concentric boundaries. Figure 6.5c shows how including an inward-facing gradient directional constraint allows the desired object boundary to be identified. Using an outward-facing gradient constraint results in location of the other boundary, as shown in Fig. 6.5d.

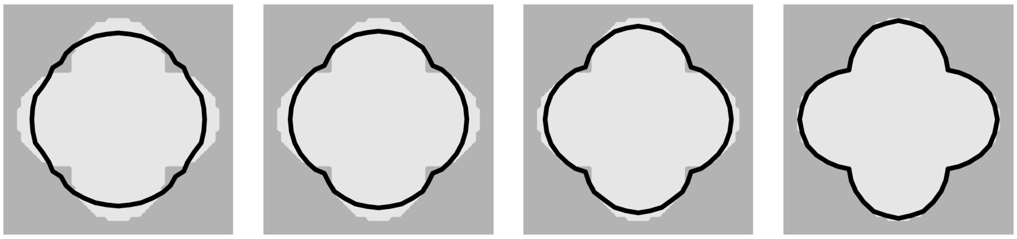


a) b) c) d)

Figure 6.5: Cross sections of boundary search: a) initial configuration. b) final configuration without directional boundaries (symbol $>$ shows area of confusion). c) final configuration searching for inward-facing gradient. d) final configuration searching for outward-facing gradient.

6.4 Stretching Constraints

One method of imposing regularization on the deforming object is to control the amount of stretching and compression of the surface with the term, $T_{stretch}$. The same initial configuration as Fig. 6.1 is used. The surface was deformed to fit the image with a cost function including both a boundary search term and a stretch term. Figure 6.6 presents the results of several such fits, each with a different weight assigned to the stretch term. This demonstrates that the amount of regularization imposed by the stretch term can be selected from a smooth continuum to achieve the desired effect. How to choose appropriate values will be dealt with in a subsequent chapter.

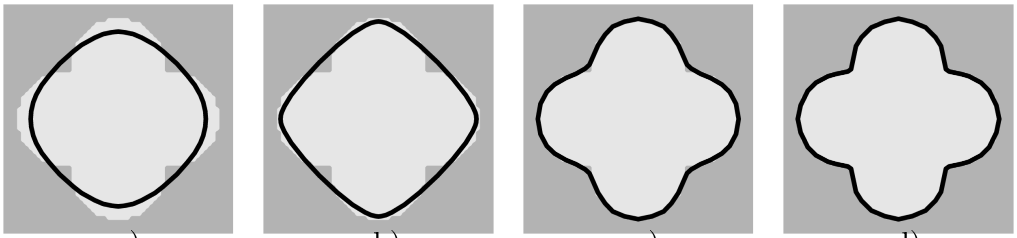


a) b) c) d)

Figure 6.6: Cross sections of deformed surfaces with stretch weights decreasing from a) through d).

6.5 Curvature Constraints

The second method of imposing regularization on the deforming object is to control the amount of bending of the surface, through the use of the curvature term. The usual initial configuration is used (Fig. 6.1), and the deformation involves minimizing the boundary search term and curvature term. The results of various values for the curvature weights are shown in Fig. 6.7. The results are very similar to that of the stretch term, again demonstrating a smooth continuum of tradeoff between the amount of surface deformation and boundary interpolation. Although the stretch and curvature term provide similar effects, there is a distinction and one may be more appropriate than another in a given context, depending on whether an application is more concerned with distances on a deforming surface, or the shape of the surface using curvature measurements.

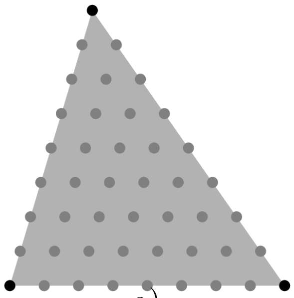


a) b) c) d)

Figure 6.7: Cross sections of deformed surfaces with curvature weights decreasing from a) through d).

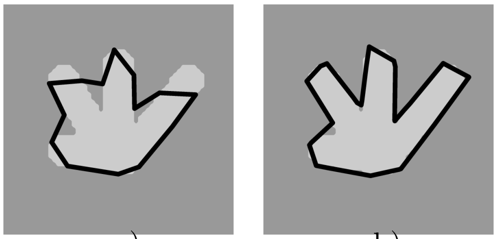
6.6 Oversampling

The effect of oversampling the boundary search term is to make the surface more likely to detect features small relative to the size of the polygons comprising the deforming model. Figure 6.8 illustrates a uniform grid superimposed upon a single triangle where the boundary search term is evaluated at each grid point in addition to the three vertices. Figures 6.9a and 6.9b show the results of two deformations of a low resolution surface where the objective function is sampled only at the vertices in the first case, and oversampled at 39 extra positions per triangle in the second case. As would be expected, the oversampled surface more closely interpolates the image data, whereas without oversampling, only the vertices of the deforming object are deformed to be near the boundaries.



a)

Figure 6.8: Grid depicting oversampling. Vertices are black, oversample points are gray.



a) b)

Figure 6.9: a) boundary sampling only at vertices. b) oversampling between vertices results in a better fit.

6.7 Vertex-to-Point Constraints

Any vertex in the deforming surface may be assigned an attractive or repulsive force with respect to a given three dimensional position, using the T_{anchor} term. Fig. 6.10 illustrates the results of several surface deformations, where the same vertex is attracted to a given point (illustrated by a small sphere) with different weights in each case. Again, a smooth continuum of tradeoff between interpolation of the image and proximity of the vertex to the fixed point is possible. Fig. 6.11 illustrates the results of similar deformations, when the vertex-to-point term is used to repel the vertex from the given point.

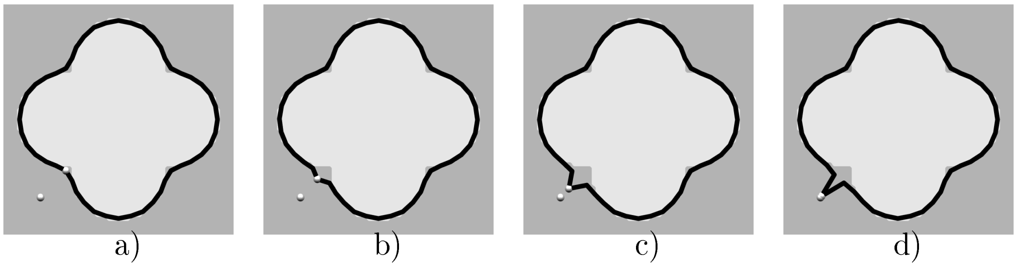


Figure 6.10: Cross sections showing vertex-to-point attraction with increasing weight from a) to d).

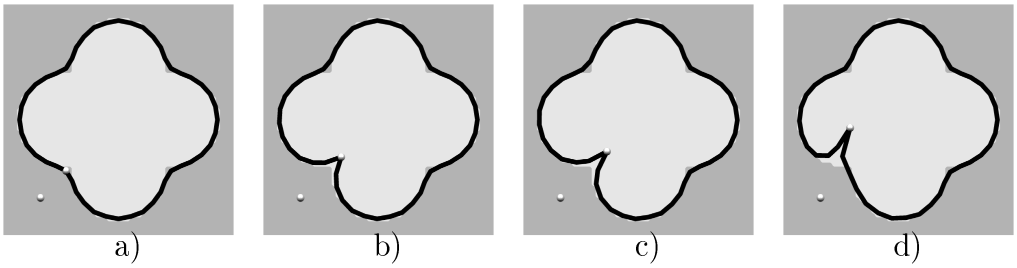
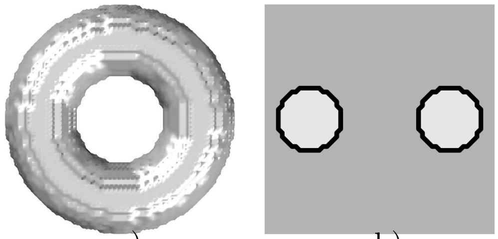


Figure 6.11: Cross sections showing vertex-to-point repulsion with increasing weight from a) to d).

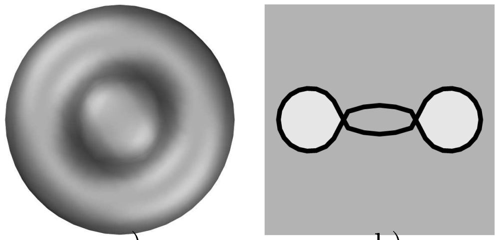
6.8 Self-Intersection Constraints

One example of where self-intersection constraints impose a pronounced effect on the outcome of surface deformation is in cases where the image data has a topological hole. Fig. 6.12 shows a three dimensional view and a cross-section view of an image volume containing a torus. Fig. 6.13 shows a cross section part way through the deformation process as an ellipsoid is being deformed to fit the image volume with no self-intersection constraints. The deformation process causes the surface to intersect itself as it attempts to wrap the surface around the hole. Imposing self-intersection constraints causes the deformation to stop when the two parts of the surface on either side of the hole meet each other, as depicted in Fig. 6.14, where non-adjacent points on the surface are prevented from coming within one millimetre of each other. While it may be argued that neither solution is actually correct due to a mismatch of topology between the model and the data, this provides a good test of using self-intersection constraints to force the model topology onto the data.



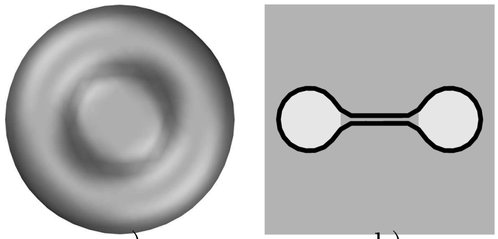
a) b)

Figure 6.12: a) Three dimensional view of torus surface. b) Cross section of torus image and surface.



a) b)

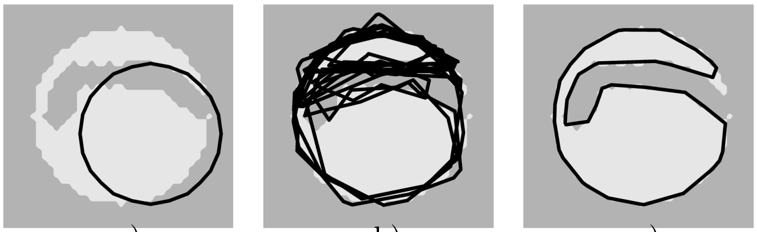
Figure 6.13: a) Three dimensional view of surface deformed with no self-intersection constraints. b) Cross section of image and surface from a).



a) b)

Figure 6.14: a) Three dimensional view of surface deformed with self-intersection constraints. b) Cross section of surface and volume.

Self-intersection constraints are also very important in data which has convoluted geometry, in particular, medical images of the human brain. Figure 6.15a illustrates an initial configuration of surface and image volume which can lead to a problem of self-intersection. Figure 6.15b depicts the cross section of the surface part way through the deformation of the surface without self-intersection constraints. The surface doubles over on itself, and eventually tries to wrap around the image object several times, never actually achieving it, which is clearly an incorrect solution. Incorporating self-intersection constraints into the process prevents this behaviour and allows the deformation process to correctly locate the image object, (Fig. 6.15c). It is interesting to note that, even in this case where the topology of the model matches that of the image, deformation to image boundaries can result in a non-simple surface due to the position of the model relative to the image volume, unless self-intersection constraints are applied. With highly convoluted geometry, such as in MR brain volumes, it appears very important to impose self-intersection avoidance on the process.



a) b) c)

Figure 6.15: a) Cross section of initial configuration that can lead to self-intersection problems. b) Cross section of surface deformed without self-intersection constraints. c) Cross section of surface deformed with self-intersection constraints.

6.9 Circumventing Partial Volume Effects

The primary objective of assembling the various objective terms presented in chapter 5 into a single surface deformation technique is to locate the total cortical surface from MR images, which are confounded by partial volume effects. A test is performed on a very simple constructed dataset. Fig. 6.16a shows a cross section of an image of a single sulcus, where CSF is coloured light gray, gray matter is dark gray, and white matter is white. A worst-case scenario partial volume effect is demonstrated by having no CSF within the sulcus, and thus no apparent sulcus, from the point of view of the boundary between the gray and CSF regions. Figures 6.16b and 6.16c show the deformation of a sphere to fit the gray-CSF and gray-white boundaries, respectively. The observation that the gray-white surface contains a well formed sulcus, even though the gray-CSF does not, motivates methods of improving the gray-CSF surface using the extra information provided by the gray-white interface. Two such techniques are presented here.

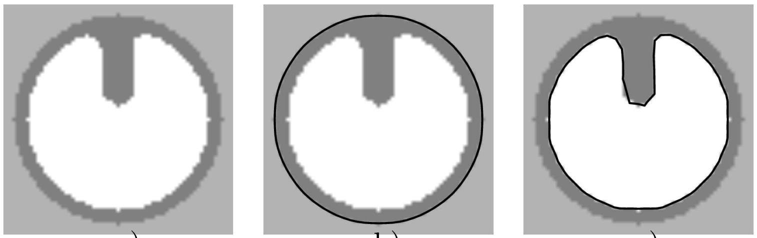
6.9.1 One Surface – Two Boundaries

To force the gray-CSF boundary deeper into the sulcus, an assumption is introduced that the cortical surface boundary should be a certain distance from the gray-white boundary, based on neuroanatomical knowledge. The corresponding cost function includes both a boundary search term for the gray-CSF boundary and a second boundary search term for the gray-white boundary, where the latter term, incorporating an offset distance of five millimetres, is weighted higher than the first. In terms of the objective function components described in a chapter 5, the overall function being minimized is of the form:

$$
\begin{aligned}
& w_1 \; T_{stretch} \; + \\
& w_2 \; T_{curvature} \; + \\
& w_3 \; T_{boundary\_dist}(1.5, 0) \; + \; \cdot \\
& w_4 \; T_{boundary\_dist}(2.5, 5) \; + \\
& w_5 \; T_{self-intersect}(0.25)
\end{aligned}
$$

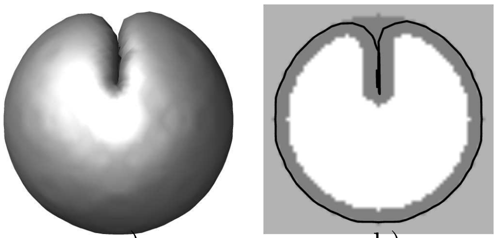
The $T_{boundary\_dist}(1.5, 0)$ term measures the distance along the surface normal to an image contour of 1.5, which corresponds to the gray-CSF boundary in a classified image. Similarly, the $T_{boundary\_dist}(2.5, 5)$ term measures distance to a point 5 millimetres away from the gray-white image boundary, defined by the value 2.5 in a classified image. The self-intersect term is non-zero whenever two non-adjacent triangles of the deforming surface come within 0.25 millimetres of each other. The stretch and curvature terms are used to keep the polyhedron relatively smooth.

Fig. 6.17 shows the result of deformation of this two-boundary model, where a reasonable approximation to the sulcus is attained. This ability to infer sulcal boundaries that are not evident in the data is of critical importance in the generation of models of the entire cortical surface. The use of self-intersection constraints ensures that while the two sides of the sulcus may be very close, they never touch and the resulting surface represents a realistic approximation to the real-world, tightly folded cortical surface. It would be difficult to achieve this complete cortical surface model in a general way with conventional methods of boundary detection, even using morphological operators.



a) b) c)

Figure 6.16: a) Cross section of image representing sulcus obscured by partial volume. b) Cross section of apparent gray-CSF boundary. c) Cross section of apparent gray-white boundary.



a) b)

Figure 6.17: a) Partial-volume corrected surface deformed to simultaneously fit two boundaries. b) Cross section of surface and volume.

6.9.2 Two Surfaces – Two Boundaries

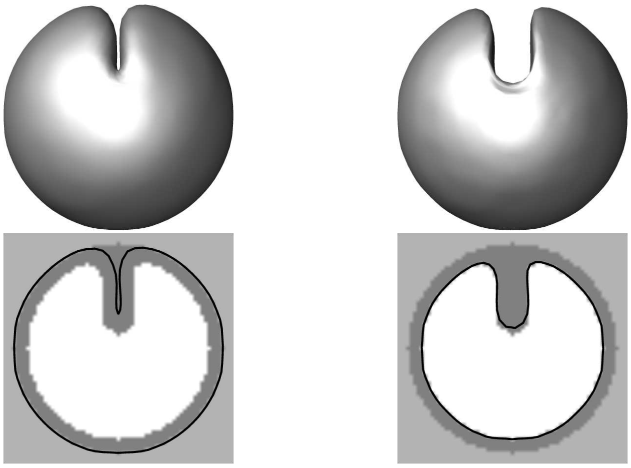
The previous section describes the use of a single surface simultaneously attracted to two concentric boundaries as a method to solve the partial volume problem with respect to deep sulci. A slight variation is presented here, which involves two surfaces attracted to the same two image boundaries defined in the previous solution. The outer surface is attracted to the gray-CSF boundary and the inner surface is attracted to the gray-white boundary. The two surfaces are constrained to be about five millimetres apart as they deform. Similar to the previous example, the inner surface has a higher weight attracting it to its image boundaries.

The function being minimized now consists of terms operating on two surfaces, S_1 and S_2:

$$\begin{aligned}
&w_1 T_{stretch}(S_1) \;+\; w_2 T_{curvature}(S_1) \;+\; w_3 T_{boundary_dist}(S_1, 1.5, 0)\\
&+w_5 T_{self-intersect}(S_1, 0.25) + w_6 T_{stretch}(S_2) \;+\; w_7 T_{curvature}(S_2)\\
&+w_8 T_{boundary_dist}(S_2, 2.5, 0) \;+\; w_9 T_{self-intersect}(S_2, 0.25)\\
&+w_{10} T_{vertex-vertex}(S_1, S_2, 3, 5, 7) \;+\; w_{11} T_{surface-surface}(S_1, S_2, 2).
\end{aligned}$$

There is a $T_{stretch}$, $T_{curvature}$, $T_{boundary\_dist}$, and $T_{stretch}$ for each of the two surfaces. In addition, the $T_{vertex-vertex}(S_1, S_2, 3, 5, 7)$ term constrains each pair of corresponding vertices in the two surfaces to prefer to be five millimetres from each other, with a prohibitively high cost preventing the distance from decreasing to less than three or greater than seven millimetres apart. Constraining the vertices of the two surfaces in this way does not, in general, guarantee that the two surfaces do not come into closer proximity. Therefore the final term, $T_{surface-surface}(S_1, S_2, 2)$, is used as a backup to prevent any triangle on one surface from coming within two millimetres of any triangle on the other surface.

The result of this deformation is presented in Fig. 6.18, where the inner surface represents the gray-white boundary and the outer surface represents the gray-CSF boundary and is similar to the result presented in the previous single surface example. The advantage of this method of partial volume correction over the one surface-two boundary technique is that there are two explicit surfaces which may be used for quantitative analysis, such as measuring the thickness of cortical gray matter.



a) Outer surface b) Inner surface

Figure 6.18: Two-surface solution to partial-volume problem.

The previous two examples illustrate how the deformation framework proposed can be used to solve a particular problem. Other solutions for this problem are also possible. One could first find the gray-white boundary, then let it expand outwards to the gray-CSF boundary, with constraints keeping it within a fixed distance of its original position. One could also design a one-surface two-boundary model where the surface stays within a couple of millimetres of the gray-CSF boundary, while being attracted towards the gray-white boundary. These suggestions and the two examples presented illustrate that the deformation framework proposed provides the flexibility to try different methods of achieving model-based segmentation.

Having examined the behaviour of the deformation process with respect to the various objective function terms, the next chapter investigates some of the implementation details, and the following two chapters deal with quantitatively and qualitatively evaluating the method on realistic data.

Chapter 7

Implementation

The implementation of NODE as described in chapters 5 and 6 is fairly straightforward. However, experience has shown that there are some aspects of the implementation that should be examined in more detail, in order to achieve the best results. A description of the computational methods of evaluating and optimizing the objective function is presented. Relevant polyhedral topologies are examined. The time complexity of the algorithm is analyzed empirically, and the areas of computation representing the largest bottlenecks are identified.

7.1 System Implementation

The deformation method is implemented in the C programming language and has been developed and tested on Silicon Graphics processors running Unix, versions IRIX 5.3 and IRIX 6.4. All timing statistics presented in this document have been generated on Silicon Graphics Origin 200 computers with R10000 processors running at a clock speed of 180 megahertz. Each computer has four processors sharing 128 megabytes of memory.

7.2 Deformation Algorithm

The essence of the deformation process is the minimization of an objective function. A conjugate gradient method has been found to provide efficient minimization of the function. The basic algorithm consists of repeating iterations of minimizations along lines in the parameter space of the surfaces until achieving the termination criteria as defined in section 5.5.2. The overall structure of the algorithm is

- Initialize positions of deforming surfaces.
- While termination criteria is not achieved.
    - Compute derivative of objective function.
    - Compute conjugate direction from derivative and previous conjugate direction.
    - Find minimum of objective function along conjugate direction.
    - Update vertex positions of deforming surfaces to this minimum.

A one dimensional Golden section search algorithm is performed to find the line minimum much like a binary search finds the root of a function. The primary efficiency arising from the conjugate gradient method is due to the restriction to one dimensional searches, rather than more fully searching around the current location in a very high dimensional space, which could require a large number of function evaluations. The Golden section search algorithm typically involves evaluating the objective function at about 20 to 60 points on a line in the function's parameter space. This repetition of function evaluations on a line provides at least two opportunities for making the computation more efficient, one related to the boundary search and the second related to self-intersection avoidance, both of which are now described.

In implementing the objective function as described, the most computationally expensive operation is the search for a boundary position along the surface normal. As mentioned in chapter 5, the boundary search term requires that the search for boundary points from a given configuration only be performed at the beginning of each set of line minimization iterations. This holding constant of the boundary points for the line minimization provides one of the opportunities for implementation efficiencies. In this case, the boundary search term of the objective function can now be reformulated as a trivially computed univariate quadratic function of distance along a line, rather than a much more complicated quadratic function of thousands of variables. Another significant increase in computational speed is gained by only performing the search for the boundary points in the image at the beginning of every n-th iteration, where n is a value of about five or more, thus reducing the computational cost of boundary searching to one fifth. The drawback is that by the end of the n line minimizations, if the vertices have moved considerably, the particular boundary points that the surface were pushed toward may no longer be close to the correct ones. However, in practice, this does not appear to be a problem, as one may modify the procedure to recompute the boundary points whenever the vertices have moved greater than a fixed distance, allowing a tradeoff between computation speed and accuracy of boundary points used in the objective function.

7.3 Intersection Avoidance

Another way the restriction to one dimensional searches allows efficient implementation is in the computation of polygon proximities. As will be seen in the time-complexity analysis, self-intersection and inter-surface intersection avoidance accounts for a significant portion of the computational cost. Since intra-surface proximity testing is a special case of inter-surface proximity testing, the discussion will be restricted to the intra-surface case for simplicity. However generalization to inter-surface proximity testing is straightforward. In order to evaluate the self-intersection objective terms at a given position in the parameter space, a list of all pairs of polygons that are within a certain distance d of each other must be generated. (Pairs that are adjacent in a polyhedron will be ignored by the objective function calculation). The current implementation has been restricted to triangles for simplicity, so the following discussions of the inter-surface proximity query will consider all polyhedra to be triangulations. Knowing that this query will be repeated at several dozen points on a line in parameter space provides the opportunity for more efficient algorithms for answering this set of queries.

Here the problem is posed more formally, as a new open problem in computational geometry. A set of n_v vertices moving at constant velocities is defined in 3-space by sextuples,

$$V = \{V_i = (x_i, y_i, z_i, dx_i, dy_i, dz_i), \quad 1 \le i \le n_v\}.$$

The position of vertex V_i is a function of a time parameter, t, such that

$$V_i(t) = (x_i + t\ dx_i, \quad y_i + t\ dy_i, \quad z_i + t\ dz_i).$$

A set of n_t triangles constituting a triangulation of these vertices is defined by triplets of integers,

$$T = \{T_j = (a_j, b_j, c_j) \quad , 1 \le a_j, b_j, c_j \le n_v, \quad 1 \le j \le n_t\},$$

The position of triangle T_j at a time, t, is thus defined as the positions of its three vertices,

$$T_j(t) = (V_{a_j}(t), \quad V_{b_j}(t), \quad V_{c_j}(t)).$$

Given this configuration of triangles and vertices, a distance, d, and a value of the time parameter, t, one can define a list of pairs of triangles that are within the distance d at the specified time-point:

$$L(V, T, d, t) = \{(m,\ n) : D(T_m(t), T_n(t)) \le d, \quad m \ne n, \quad 1 \le m, n \le n_t\},$$

where the function $D(T_m(t), T_n(t))$ is the minimum three dimensional Euclidean distance between any point of triangle $T_m(t)$ and any point of triangle $T_n(t)$.

At this point, the problem of computing $L(V, T, d, t)$ is quite similar to the types of problems encountered in motion planning, dynamic simulation, and computational geometry. However, there are several further refinements based on the use of this query in a surface deformation context which create a unique proximity query problem. Most importantly, due to the nature of the Golden Section search of parameter space along a line, the query is actually repeated at several values of t, and the problem is actually to generate a set of lists at n_l unordered time points. The open problem can now be defined as computing the following set of lists of triangle pairs,

$$\{L(V, T, d, t_k): \ 1 \leq k \leq n_l \ \ , \ t_{min} \leq t_k \leq t_{max}\} .$$

However, the value of each successive time point, t_k, is not known in advance, as it can only be computed after the lists from L_1 up to L_{k-1} have been generated. Other important features of this query within the context of surface deformation include the fact that the sets of triangles constitute non-convex polyhedra which are isomorphic to triangulations of a sphere, and the sizes of all the triangles are roughly the same. The value of d used is typically quite small relative to the domain of the triangles, usually under one millimetre, whereas the polyhedra being deformed are around 120 millimetres in diameter. In addition, the amount of movement of the vertices, as defined by the parameters, t_{min} and t_{max}, is also quite small, usually such that no vertex moves more than a couple of millimetres over this range.

Having defined this problem of generating a set of lists of triangles pairs, it should be noted that some of the computational geometry methods presented in chapter 3 could be adapted to solving this problem. A simple method would be to use an existing proximity testing algorithm to solve the query of each time point, t_k, independently. However, for ease of implementation, the problem is transformed from a query at a large number of time points to a smaller number of queries, by capitalizing on two observations. Firstly, it can be observed that the query is allowed to return a superset of the correct answer without adversely affecting the deformation algorithm. In addition, it is observed that the query is repeated many times on a fairly small interval of t, and the motion of the triangles is also small relative to the space in which they reside. The transformation to a smaller number of queries is performed by dividing the range, $[t_{min}, t_{max}]$ into uniform intervals of width, w. A single list of potential triangle pairs is created for each interval which contains one or more values of t, rather than for each value of t. The list computed for a given interval centred at t_c, $[t_c - \frac{w}{2}, t_c + \frac{w}{2}]$ must therefore be the union of all possible lists computed on this interval:

$$L_{\cup}(t_c - \frac{w}{2}, t_c + \frac{w}{2}, w) = \bigcup \left\{ L(V, T, d, t) : \; t_c - \frac{w}{2} \leq t \leq t_c + \frac{w}{2} \right\}.$$

The essential simplifying step can now be applied. The union of all possible lists over a continuum is itself a subset of a list computed at the centre of the interval but with an increased distance range:

$$L_{\cup}(t_c - \frac{w}{2}, t_c + \frac{w}{2}, w) \subseteq L(V, T, d + \sigma(w), t_c),$$

for an appropriate value of $\sigma(w)$. The value of $\sigma(w)$ may be computed as

$$\sigma(w) = s \; \frac{w}{2},$$

where s is the maximum distance two vertices can move relative to each other, per unit of t. The value of s is bounded by the **diameter** [1] of the set of three dimensional line directions of the vertices,

$$s = diameter(\{(dx_i, dy_i, dz_i) : \; 1 \leq i \leq n_v\}).$$

Thus a tradeoff has been performed, a smaller number of time points to query, in return for computing lists for larger values of d and returning supersets of the correct answer. The potential inefficiency of this method arises if the interval width, w, is too large, in which case a much larger number of triangles will be compared by the surface deformation objective function than if the list is computed for each evaluation point on the line. In practice, the intervals can be chosen small enough to avoid this problem, yet still large enough to be efficient, by satisfying several queries per interval.

Now a method to compute the list, $L(V, T, d, t_k)$, is presented. Several of the published techniques discussed in chapter 3 are applicable to the solution of this

[1] the largest magnitude of difference between vectors in the set

query, including plane sweep algorithms, bucketing methods, and hierarchical data structures such as bin-trees. The divide-and-conquer strategy is chosen here, due to its simplicity of implementation. The method is derived from the divide-and-conquer method of finding the closest pair of points in a set by Preparata and Shamos [PS85], combined with the box intersection testing of Gupta et al [GJS96]. The problem is first transformed into solving the query for three dimensional rectangular boxes, by fitting a bounding box around each triangle. The set of boxes is recursively split into small subsets, where proximity testing is only performed within subsets. The recursive partition of a set of boxes is performed on one of the three coordinate axes, (x, y, or z) at a time. Assuming that splitting is being performed along the x axis, the midpoint, m, of the x domain of the boxes is found, and two subsets are generated. The first subset is the list of all boxes in the set which intersect the half space defined by $x \leq m + \frac{d}{2}$. The second subset is the list of all boxes in the set which intersect the half space defined by $x \geq m - \frac{d}{2}$. The two half spaces are not disjoint and the resulting two subsets of boxes are also not necessarily disjoint. In order to avoid multiple testing of particular pairs, the boxes in the second subset that are also in the first subset are flagged. Each subset is recursively subdivided into smaller subsets until they contain a small number of boxes, or correspond to a small enough region of space (no smaller than twice the distance d). At this point, the distance between all pairs of boxes in the subset is tested, except those pairs corresponding to boxes that are flagged in such a way as to prevent duplicate tests of the same pairs. If a pair of boxes is found to be within the distance d, then the corresponding triangles are tested. In this way, the trivial computation of box proximities reduces the number of more costly triangle proximity computations. This algorithm is used for both inter-surface and intra-surface proximity testing. The only difference is that for intra-surface proximity testing, pairs of adjacent triangles are discarded, and whereas for the inter-surface case, two disjoint sets of triangles are compared and adjacency is not relevant.

Although this method has been found to be relatively efficient in practice, it has a worst case complexity of $O(n^2)$, where n is the number of triangles involved, and it is therefore an interesting problem to try to reduce the worst case complexity. However, it can be easily seen that this algorithm is already optimal, since a large enough value of d can be chosen to require returning all pairs of triangles in the set, which is of quadratic size. Therefore, it is more interesting to characterize the complexity as a function of the size of the output. Because the distance parameter, d, is usually a small fraction of the domain of the polyhedra being deformed, the size of the output, though quadratic in the number of triangles, is usually quite small. This motivates one to speculate on whether there is an algorithm which always solves the triangle pairs query in time linear in the output size.

7.4 Surface Tessellation

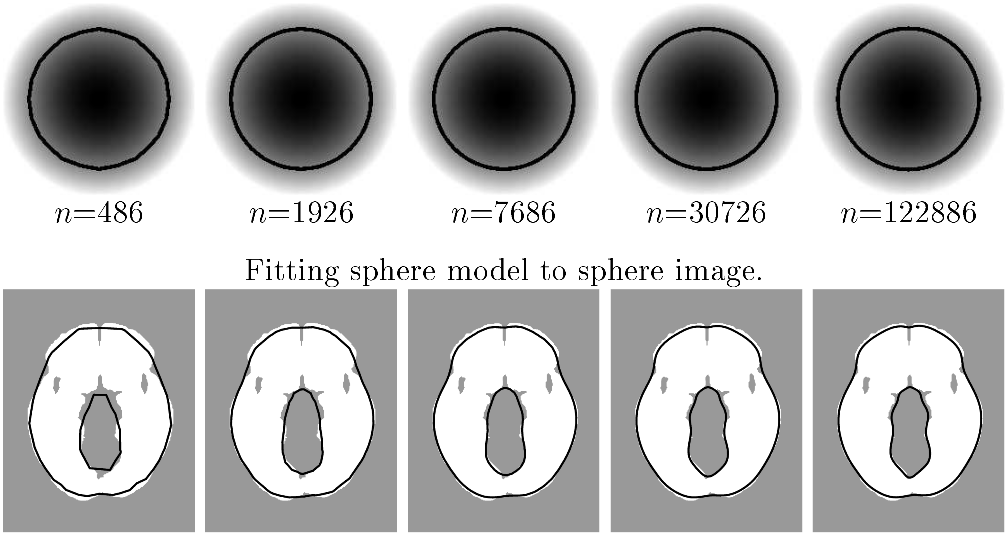
The surface deformation algorithm as currently implemented can be applied to any triangular mesh isomorphic to a sphere. Generally, the starting point for a deformation is a sphere or ellipsoid, or an object that is a deformation of one of these. The tessellation method used is to start with an icosahedron, the platonic solid of 20 triangular faces, and repetitively subdivide each triangle into four similar triangles, until the desired number of triangles are reached. The new vertices created by triangular subdivision are projected out to the sphere or ellipsoid. For many of the experiments where brain models are represented, there are five successive sizes used: 320, 1280, 5120, 20480, and 81920 triangles. The number of vertices in any triangulation of a sphere is $n_t/2 + 2$, where n_t is the number of triangles in the polyhedron. Therefore the numbers of parameters, three per vertex, defining these five surfaces are: 486, 1926, 7686, 30726, and 122886. These polyhedra were chosen over other tessellations because they are simple to compute, yet have a fairly uniform distribution of vertices over the surface. Many of the surfaces discussed here will have one of these five configurations.

It is also possible to use adaptively subdivided triangulations, in order to allow regions of the model to stretch as much as needed to match the image, but maintain a more uniform sampling interval by adding vertices where needed. The simple method used here is to subdivide all edges in a polyhedron which are greater than a certain length, and re-triangulate. By decreasing the length threshold and subdividing the surfaces during deformation, the polyhedral surface automatically adds vertices and increases the number of triangles as needed to match the image more closely. In a subsequent chapter, adaptive refinement of polyhedra is used for the segmentation of the total cortical surface using a two-surface model.

7.5 Time Complexity

Due to the complexity of the objective function and MR image data, it is difficult to derive a theoretical measure of the time complexity of the deformation algorithm. However, some theoretical observations are made, before resorting to empirical measurement of computational cost as a function of the number of parameters in the surfaces being deformed. The conjugate gradient method is guaranteed to find the minimum of a quadratic function in a number of steps no greater than the number of parameters. When applied to more complicated functions, it is difficult to predict the rate of convergence, but it seems reasonable to assume that it is at least of linear order in the worst case. Each iteration involves computing the objective function a few dozen times. All components of the objective function can be computed in linear time, except for one. The algorithm described in the previous section for computing triangle proximities is, in the worst case, of quadratic complexity. Multiplying the $O(n)$ number of steps by the $O(n^2)$ cost per step results in an expected worst-case time complexity of $O(n^3)$.

In addition to this brief theoretical analysis, some empirical analysis can provide more information on the running time of the surface deformation algorithm in practice. Empirical estimates of time complexity are generated by two different surface fitting tasks. The first task is the deformation of a spherical model to a differentlysized spherical image. The second task is to fit an ellipsoid to the average brain image described in section 5.6. The deformations were performed at five surface resolutions, 320, 1280, 5120, 20480, and 81920 triangles, using $T_{boundary\_dist}$, $T_{stretch}$, $T_{curvature}$, and $T_{self-intersect}$ terms. The termination criteria was that no vertex moves more than .01 millimetres over 10 consecutive iterations. Cross sections of the results of the two sets of deformations are presented in Fig. 7.1. The plot of computational time versus the number of parameters defining the surface is presented in Fig. 7.2, and in log-log form in Fig. 7.3. With only five sample points, it is difficult to characterize the nature of the time-complexity functions, but it is certainly a super-linear function and not inconsistent with an order $O(n^3)$ function.



n=486 n=1926 n=7686 n=30726 n=122886

Fitting sphere model to sphere image.

n=486 n=1926 n=7686 n=30726 n=122886

Fitting ellipsoid model to average cortex.

Figure 7.1: Cross sections of fitting to sphere and average image, with number of parameters, n.

7.5.1 Fitting Using a Multi-Scale Approach

From the previous set of experiments, it can be seen that the higher resolution surfaces require considerably more computational effort, and the computation time increases

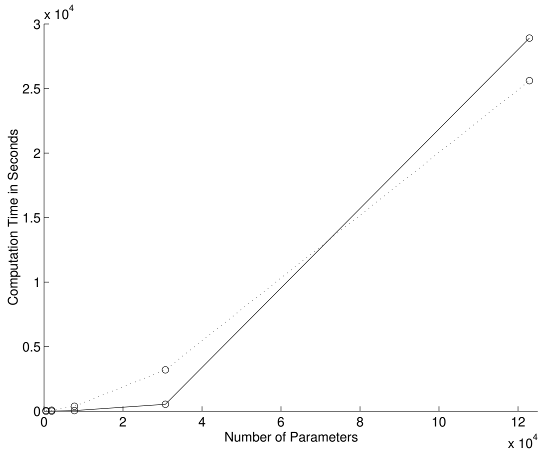


Figure 7.2: Plot of computation time versus number of parameters. Solid line = fitting to sphere, dotted line = fitting to average image.

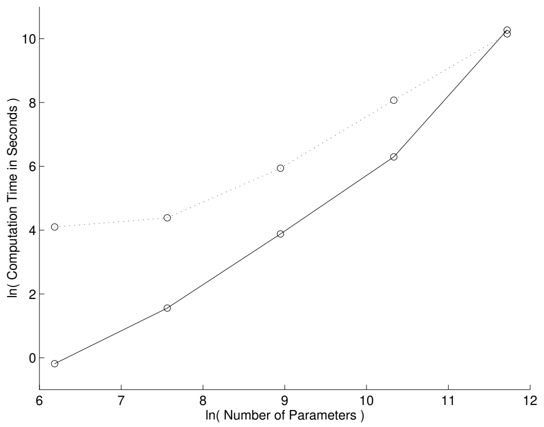


Figure 7.3: Log-log plot of computation time versus number of parameters. Solid line = fitting to sphere, dotted line = fitting to average image.

in a quadratic or cubic nature as the number of parameters increases. It is therefore desirable to perform the deformation in a multi-scale approach, so as to quickly achieve a coarse fit using a small number parameters, thus reducing the number of iterations required for the higher resolutions. The experiments of fitting a sphere surface to a spherical image and an ellipsoid to the average brain image are repeated, where the deformed polyhedron at each resolution of surface is subdivided into four times as many triangles, to be used as the initial configuration of the next finer resolution deformation. Accordingly, the time required for the smallest size, 320 triangles, is the same as in the previous experiments. However, the time required for the higher resolutions has been reduced by a factor of 70 for the highest resolution of the sphere-fitting experiment, and a factor of four in the case of fitting to the average cortex image. In addition the time complexity has become more linear in nature, as can be seen from the corresponding plot of time complexity, Fig. 7.4, and the log-log plot of Fig. 7.5. The segmentation results (Fig. 7.6) are as good or better as compared to the non-multi-scale experiments. This is a strong indication that the multi-scale approach represents a major gain in efficiency.

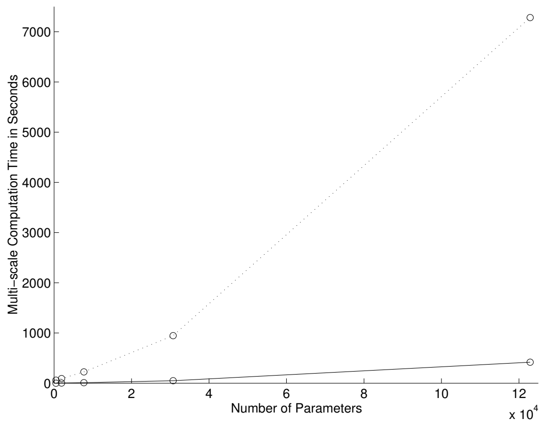


Figure 7.4: Plot of computation time versus number of parameters in a multi-scale deformation. Solid line = fitting to sphere, dotted line = fitting to average image.

7.6 Computational Bottlenecks

The computational cost of the current implementation of the surface deformation in typical applications with surfaces of 81 920 triangles can be broken down into six main components:

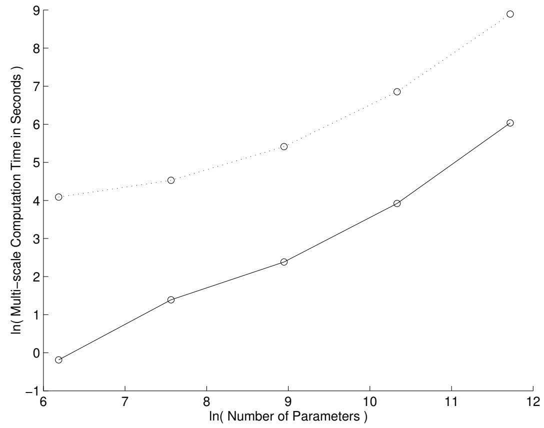


Figure 7.5: Log-log plot of computation time versus number of parameters in a multi-scale deformation. Solid line = fitting to sphere, dotted line = fitting to average image.

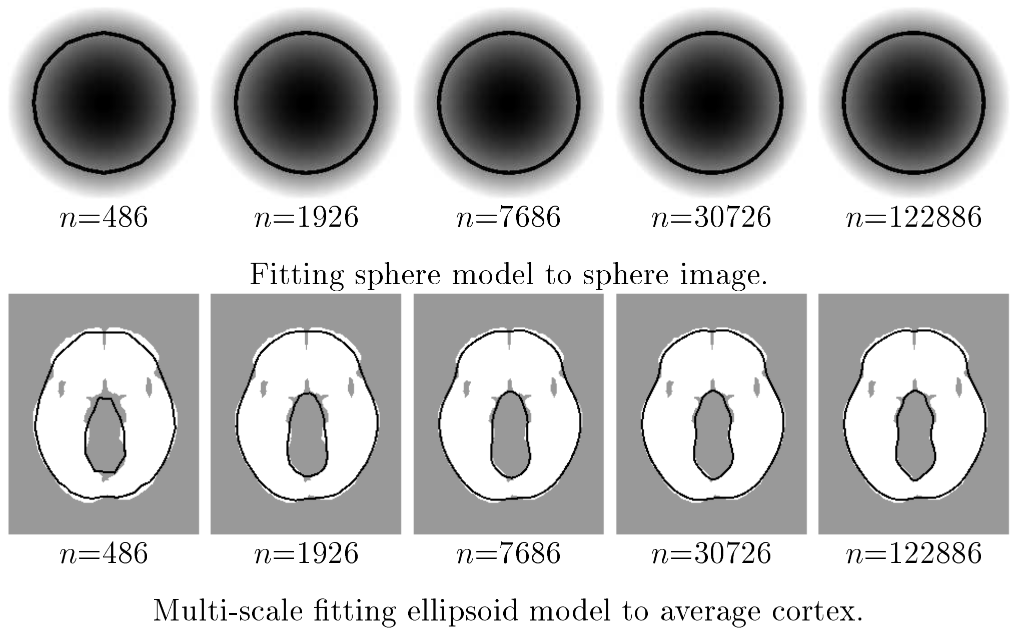


Figure 7.6: Cross sections of multi-scale fitting to sphere and average image, with number of parameters, n.

20 %	Generating pairs of triangles within distance d,
15 %	Measuring distance between triangle pairs in objective function calculation,
20 %	Searching image volume for boundaries along surface normals,
5 %	Evaluating stretch component.
20 %	Evaluating curvature component.
20 %	Remainder of program.

The first component above, the triangle set proximity query, takes one fifth of the time, and could conceivably be reduced through use of a more efficient algorithm. The second component represents the time spent by the objective function calculation to measure the distance between pairs of triangles. Those pairs for which the distance is less than some threshold, d, contribute a value to the function, based on the distance found. The number of calls to this inter-triangle distance function cannot be reduced significantly because measurements of the current implementation have shown that 94 percent of the triangle pair distance computations result in a value less than the distance d, and therefore must be computed in order to evaluate the self-intersection term. The final 20 percent of the time is not broken down further, as it is involves many different parts of the program. The present implementation is well optimized at present, but as the size of problems addressed by this algorithm increases, the relative computational times may change, and more efficient algorithms may provide significant improvements. In particular, the solution to the all-pairs triangle proximity problem may well become the most significant computational aspect due to its quadratic worst case complexity. Another possible improvement in computation time may result from removing the stretch or curvature term from the objective function. The usefulness of these two terms in the segmentation of total cortical surface is investigated in the next chapter.

Chapter 8

Validation on Simulated Data

The first step in validating the NODE deformation algorithm is to apply it in controlled situations, where the factors influencing the procedure are well understood. The method can be subjected to a variety of tests, where the correct answer is known or assumed, and measures of error devised. This chapter outlines a set of experiments where MR-derived phantom data were constructed in order to test specific features of the algorithm. Each section outlines what features of the algorithm are being tested, describes how the experiment was constructed, and presents numerical, graphical, and visual summaries of the results.

8.1 Rigid Surface-to-Image Matching Validation

A necessary feature of any deformation procedure is that it be robust in the face of intensity noise and geometric distortions. In particular, it is desirable that the final position of any deformation be relatively independent of the spatial position of the initial guess, as well as insensitive to noise and sampling artifacts which may distort the apparent boundaries. In order to test this, a simple task was formulated. A low resolution model of the cortex, consisting of 320 triangles, is fit to a target MR dataset to locate the cortical surface. By keeping the stretching and curvature weights relatively high, the cortex model is rigidly transformed to fit the global shape of an individual brain. In effect, this is performing a surface-to-image linear registration task. The MR dataset is first blurred slightly with a square filter of width three millimetres to smooth the objective function slightly. The consistency of this process is tested in the face of the following four confounds:

slice thickness,
image noise,
RF inhomogeneity, and
linear misregistration.

This experiment measures consistency, not accuracy, since there is no inherently correct surface. The correct answer is arbitrarily defined as the surface resulting from deformation to a particular gold standard image volume. In order to analyze the effects of each of the four confounds independently, the gold standard used in each case is the volume which has the lowest value of the confound parameter varied, or, in the case of varying the initial misregistration, an identity transform. The measurement of error in the final result is defined as the RMS error over the pairs of corresponding vertices in the deformed cortex model and the gold standard surface:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (z_i - \hat{z}_i)^2}{n}},$$

where (x_i, y_i, z_i) is the i-th vertex in the test surface and $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$ is its counterpart in the gold standard surface. This measure assumes that the two polyhedra are isomorphic to each other, which is the case in the context of the deformation-based segmentation discussed here.

A magnetic resonance simulator, called (**MRISIM**) [KEP96] is used to model the first three confounds. MRISIM is a program designed to model most of the characteristics of an MR imaging system, in order to create volumes that simulate MR images. Three sets of simulated images have been previously computed. The first set varies the noise level from 0 % to 10 % of the image intensity. The second set varies the slice thickness from one to 10 millimetres, and the third set varies the RF inhomogeneity from 0 % to 60 %. The fourth confound, linear misregistration, is simulated with random linear transforms, which have randomized three dimensional scalings, translations, and rotations. The degree of misregistration, σ_{in}, is defined as the RMS error over pairs of vertices in an initial model and the linearly transformed coordinates of the model. One hundred linear transforms have been created with a roughly uniform distribution of σ_{in} from zero to 100 millimetres.

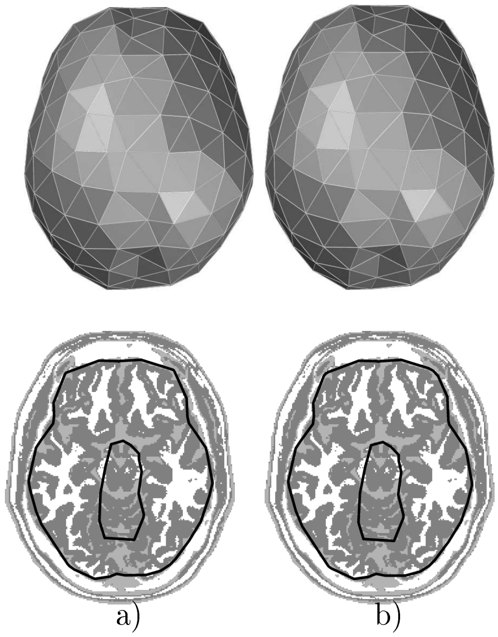


Figure 8.1: Deformation fitting cortex model to individual MR dataset. a) initial configuration b) final configuration (only small differences from initial).

Figure 8.1 shows the configuration of the surface before and after fitting the cortex model to a typical classified image volume. In the absence of a large initial misregistration error or other confounds, there is little change in the surface during this deformation, since the model derived from the average brain image already matches any individual brain in the stereotaxic space reasonably well at a resolution of 320 triangles. This procedure was tested against each of the four confounds, and the results presented in graphical form, Figs. 8.2, 8.3, 8.4, and 8.5.

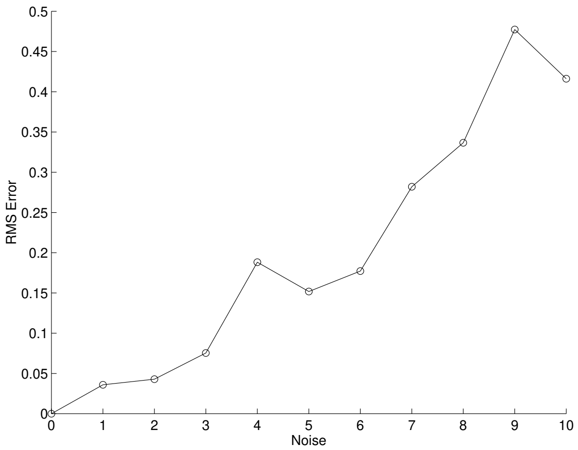


Figure 8.2: RMS error, σ, versus noise level.

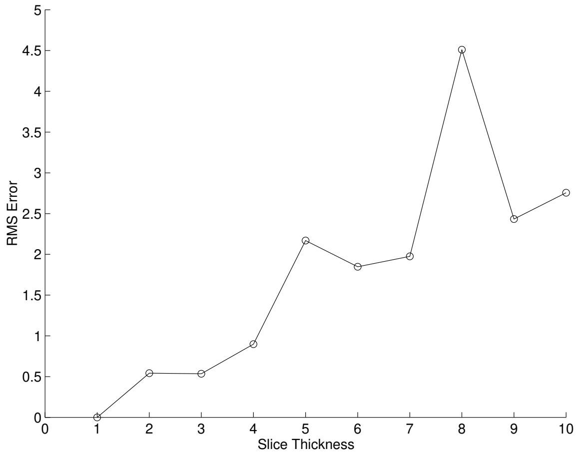


Figure 8.3: RMS error, σ, versus slice thickness.

The graph of σ versus percent noise shows a roughly linear increase to $\sigma = .47$mm at the highest noise level of 10 %. Given that the image data is sampled at one millimetre, this is a satisfactory value. The graph of the effects of RF inhomogeneity also show a linear trend to less than one millimetre at the highest RF level of 60 %. However, the graph of σ versus slice thickness shows a much steeper increase in σ, with a maximum value of almost five millimetres. This is due in part to the partial volume effect of large slice thicknesses causing the cortex and skin to be blurred together, which results in the cortex deformation incorrectly locking on to the outer

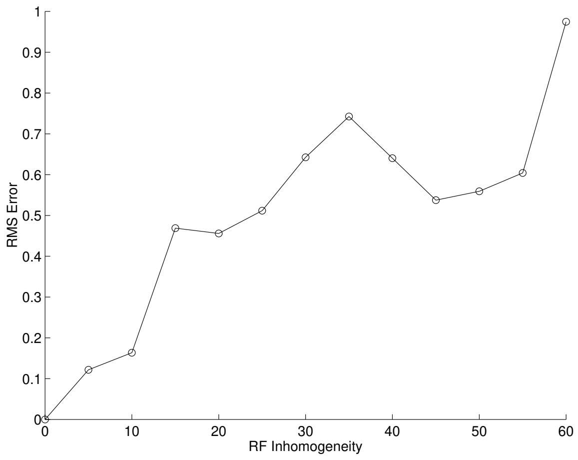


Figure 8.4: RMS error, σ, versus RF level.

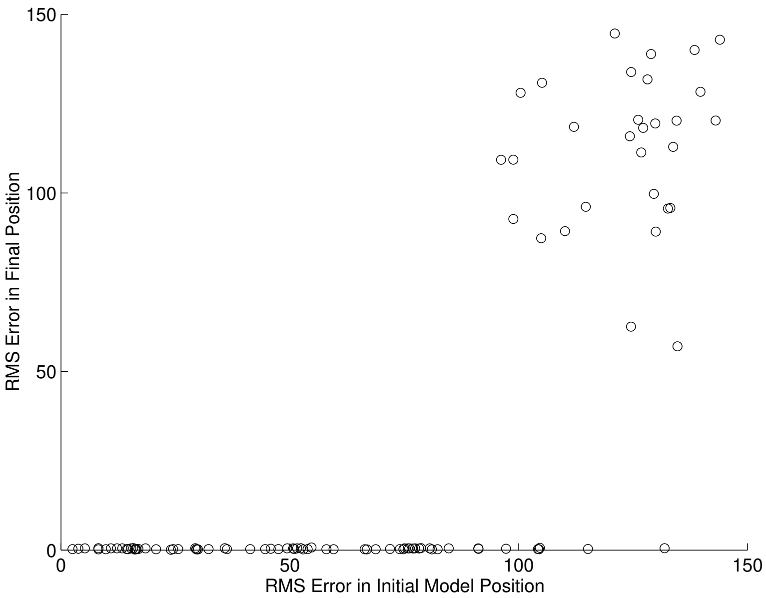


Figure 8.5: RMS error, σ, versus initial spatial error, σ_{in}.

skin surface. There are a few points in each graph where the value either drops or rises more sharply than would be expected, such as can be seen at the eight millimetre value on the slice thickness plot. Further investigation is required to determine if this is due to instability in the algorithm or can be explained by the particular interaction characteristics of the imaging confound and the deformation procedure.

Confound	Range of Confound	Maximum RMS Error (σ)
noise	≤ 5 %	0.19 mm
RF	≤ 30 %	0.64 mm
slice thickness	≤ 3 mm	0.54 mm
misregistration	$\sigma_{in} \leq 20$ mm	0.60 mm

Table 8.1: Maximum RMS error, σ, in mm^2, as a function of imaging confounds.

The plot of RMS error, σ, versus initial misregistration, σ_{in} shows a very interesting result (Fig. 8.5). The registration process either produces an answer very close to the correct answer or very far away. Any surface with an initial RMS below 90 millimetres ($\sigma_{in} < 90$mm) is deformed to within 0.8 millimetres of the gold standard ($\sigma < 0.8$mm), two examples of which are depicted in Fig. 8.6. Any surface with a value of σ_{in} greater than 90 millimetres causes the deformation process to fail, with an error of $\sigma > 50$mm, two examples being shown in Fig. 8.7. This behaviour can be explained by using an analogy to fitting a two dimensional ellipsoid model to an image of an ellipsoid with only a single rotation parameter. If the initial rotation is greater than 90 degrees from the orientation of the image, it is easy to imagine that the best fit found is actually 180 degrees out of alignment. Fitting a three dimensional cortex to an image is more complex than this simple analogy, but examination of the cases where the procedure fails illustrates that a rotation of 90 degrees around an axis is usually involved.

Table 8.1 summarizes the results for the range of the confounds expected in typical MR images that have undergone an image registration into stereotaxic space. With respect to the task of segmenting the entire cortical surface, the effects of the four confounds mentioned can be minimized by using this rigid deformation procedure as a pre-processing step to define the position of a model to be deformed to fit the total cortical surface. In addition, the good results in the case of large initial misregistrations indicate that this method may be useful as a stand-alone image

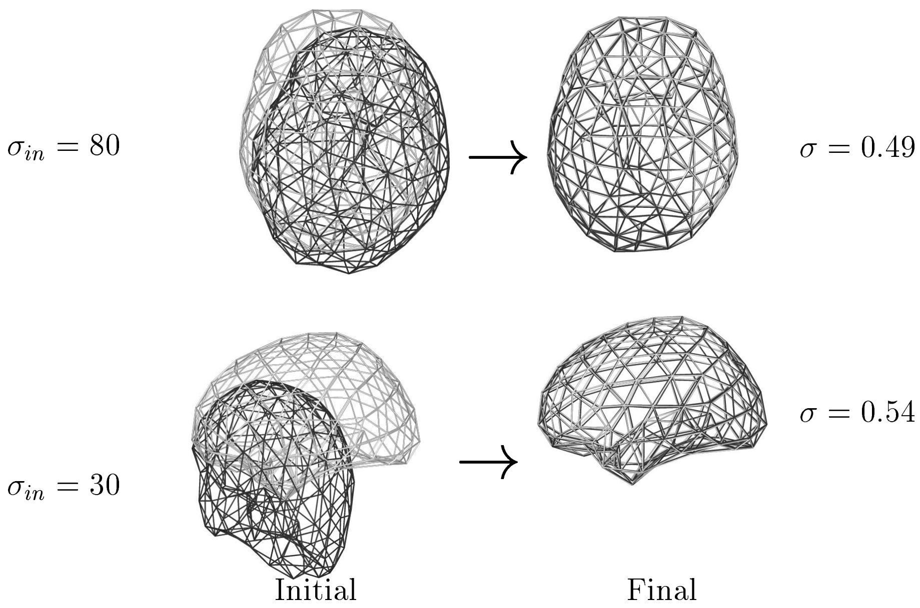


Figure 8.6: Initial and final configurations of two examples of successful image deformation (top view of example 1 and left view of example 2) (black = surface position, gray = gold standard).

registration method. It would be interesting to compare this method to established surface-based registration methods, such as that of Pelizzari et al [PCS+89], and to investigate whether the use of multiple models, e.g., cortex, cerebellum, and skin, can improve the image registration procedure.

8.2 Partial Volume Solution Using Two-Surface Model

Having examined the consistency of the deformation in the simple case of rigid transformation, the more interesting problem of locating the complete extent of sulci in complex MR images is now addressed. A small brain phantom which contains sulci characteristic of full brain MR images is created by scooping a small spherical piece of gray and white matter out of a classified image of a normal individual. Three cross

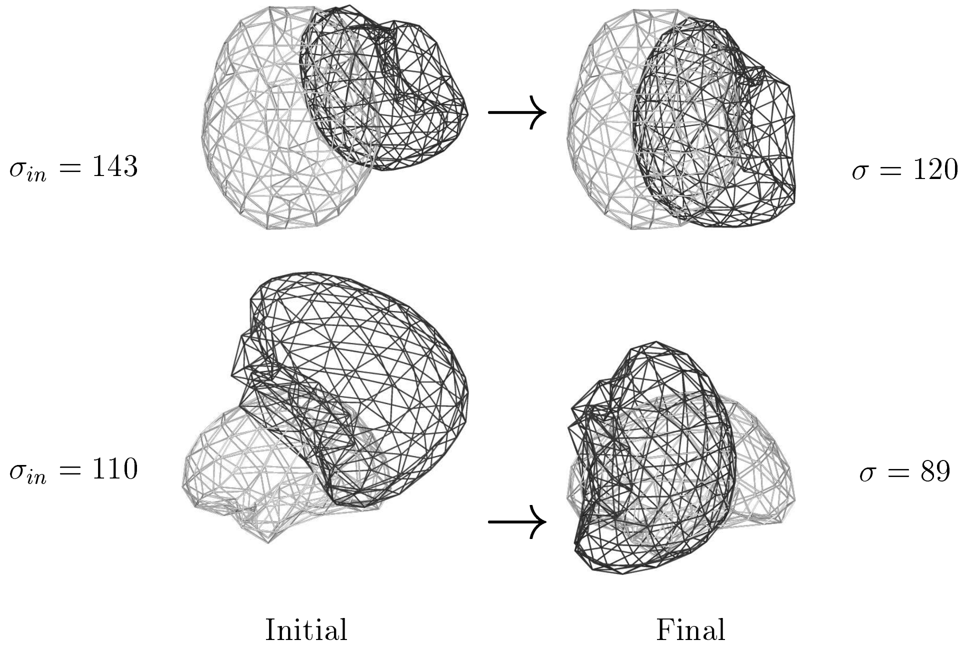


Figure 8.7: Initial and final configurations of two examples of failed image deformation (top view of example 1 and left view of example 2) (black = surface position, gray = gold standard).

sections of this phantom are presented in Fig. 8.8. The method used to locate the boundary is to start with two concentric spheres five millimetres apart. The inner surface is deformed to fit the gray-white boundary in the image while the outer surface is fitted to the gray-CSF boundary. Corresponding vertices of the two surfaces are constrained to be within three and seven millimetres apart, with the optimal value at five millimetres, using a $T_{vertex-vertex}$ objective term. A $T_{surface-surface}$ term is used to prevent any pair of points on the two surfaces from coming within one millimetre of each other. In order to allow the surface to stretch as much as required, adaptive subdivision of the polyhedral mesh is performed during the deformation, so as to increase the number of vertices in areas that are highly stretched, such as deep sulci. The initial model consists of 320 triangles, and the maximum allowable length is decreased from 20 millimetres to three millimetres during the deformation, causing the surface to be repetitively subdivided, with the final surface consisting of about 4500 triangles.

Figure 8.8: Parallel cross sections through small brain phantom.

The initial and final states of the deformation are presented in Fig. 8.9. For purposes of comparison, two other methods of locating the gray-CSF boundary are performed. One method simply repeats the deformation using a one-surface model, and the second uses the Marching Cubes algorithm to triangulate the gray-CSF boundary. Cross sections of all three methods are presented in sagittal orientation in Fig. 8.10, coronal orientation in Fig. 8.11, and transverse orientation in Fig. 8.12, for comparison purposes. There are some cross sections where small pockets of CSF on a slice have been identified by the Marching Cubes algorithm, but not by the two-surface model. However, these features, such as the ones visible on the X=-41 mm slice of the Marching Cubes algorithm as tiny circular regions, correspond to the deepest extremity of sulci, and three dimensional investigation in the vicinity (Fig. 8.13) reveals that the two-surface model comes within a millimetre or so of these features. Therefore, the differences between the two techniques is not really as large as would be inferred from looking at the single slice alone. Table 8.2 lists the number of triangles created by each of the three methods and the surface area of the gray-CSF surface found. The two-surface deformation produces a higher surface area than the one-surface method, indicating that it is achieving greater sulcal depth. However, even though the two-surface deformation achieves greater sulcal depth than the Marching Cubes method in places, the surface areas are about the same, and there are places where the Marching Cubes method provides a better approximation to the gray-CSF

Method	Number of Triangles	Surface Area, in mm^2
one-surface deformation	3310	10339
two-surface deformation	4514	12537
Marching Cubes	34472	12465

Table 8.2: Results of gray-CSF segmentation on small phantom.

boundary. This is most likely due to two factors. Firstly, the deformation method has smoothing constraints in the form of $T_{stretch}$ and $T_{curvature}$ terms, whereas the Marching Cubes method has no smoothing and responds to every edge feature, making a more crumpled surface and thus inflating the surface area. The effects of changing the stretch and curvature weights on the two-surface deformation are examined in the next section. A second factor is that the marching cubes surface contains more than seven times as many triangles as the two-surface deformation result, and therefore can more closely interpolate the image boundaries.

The importance of using a two-surface model is especially evident in areas where there are topological holes in the image data. For instance, the hole in the centre of the Y=-11 mm slices is not located by either the single-surface deformation or the Marching Cubes algorithm [1] because it is not connected to the rest of the gray-CSF boundary. However, the two-surface method produces a neuroanatomically reasonable surface that includes the hole, circumventing the partial volume effect. Given that the phantom is representative of typical sulcal and gyral topology in classifed MR images, the application of this method to the total cortical surface should be straightforward.

8.3 Choosing Weights

A significant problem with the NODE algorithm and with deformation algorithms in general is the need to provide weighting factors for the various components of the

[1] The version of Marching Cubes used does not exhaustively search all voxels, but instead finds a single connected component.

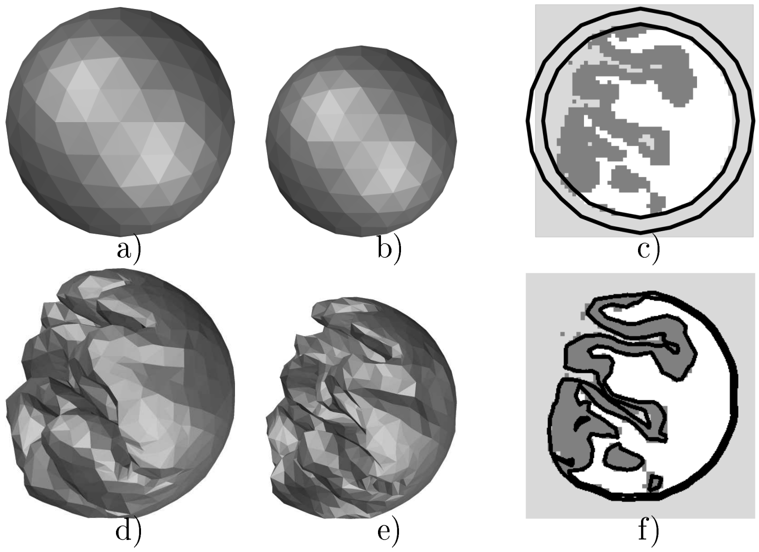


Figure 8.9: a) Initial gray-CSF surface. b) initial gray-white surface. c) cross section of volume and the two surfaces. d) Final gray-CSF surface. e) final gray-white surface. f) cross section of volume and the two surfaces.

objective function, for which there is no theoretical basis upon which to draw. The procedure used to choose weights for the NODE algorithm is to vary one weight at a time, usually starting with a very high value and reducing it until the desired effect has been achieved. The following terms are used in the formulation of the objective function for simultaneous deformation of two surfaces:

$$
\begin{aligned}
&T_{stretch},\\
&T_{curvature},\\
&T_{boundary\_dist},\\
&T_{self-intersect},\\
&T_{vertex-vertex},\\
&T_{surface-surface}.
\end{aligned}
$$

One of the critical terms is $T_{self-intersect}$, which is used to keep each surface from intersecting itself. The obvious method is to assign a weight to this term that is many orders of magnitude higher than the other weights, in order to make $T_{self-intersect}$

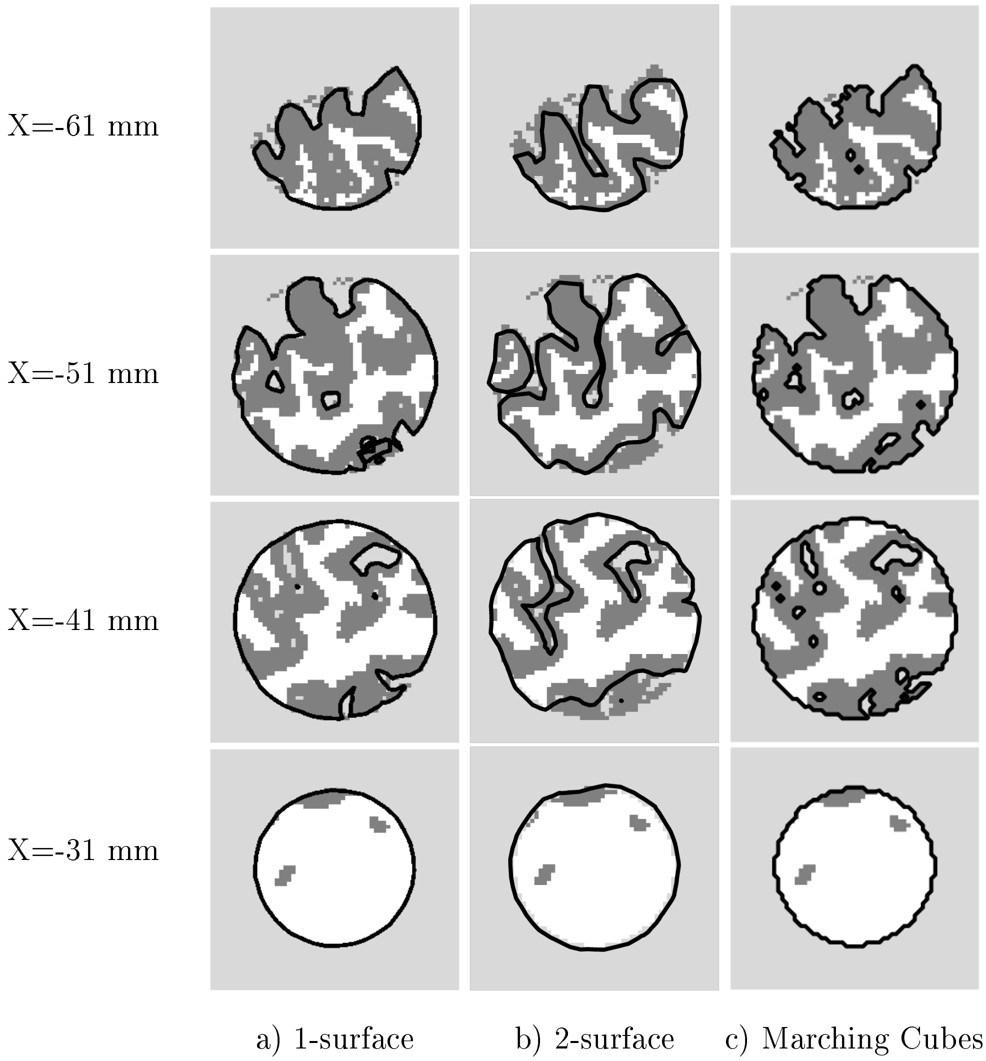


Figure 8.10: Sagittal slices through phantom volume and gray-CSF surface produced by a) one-surface deformation b) two-surface deformation c) Marching Cubes algorithm.

dominate when two portions of a surface come into close proximity. However, in practice, setting the weight too high causes the minimization procedure to take smaller steps, due to the high curvature in the objective function. This slowing down is analagous to what Kass et al observe about their Snakes algorithm, "(if) the external forces become large, however, the explicit Euler steps of the external forces will require much smaller step sizes" [KWT88]. Within the NODE framework, a solution to this

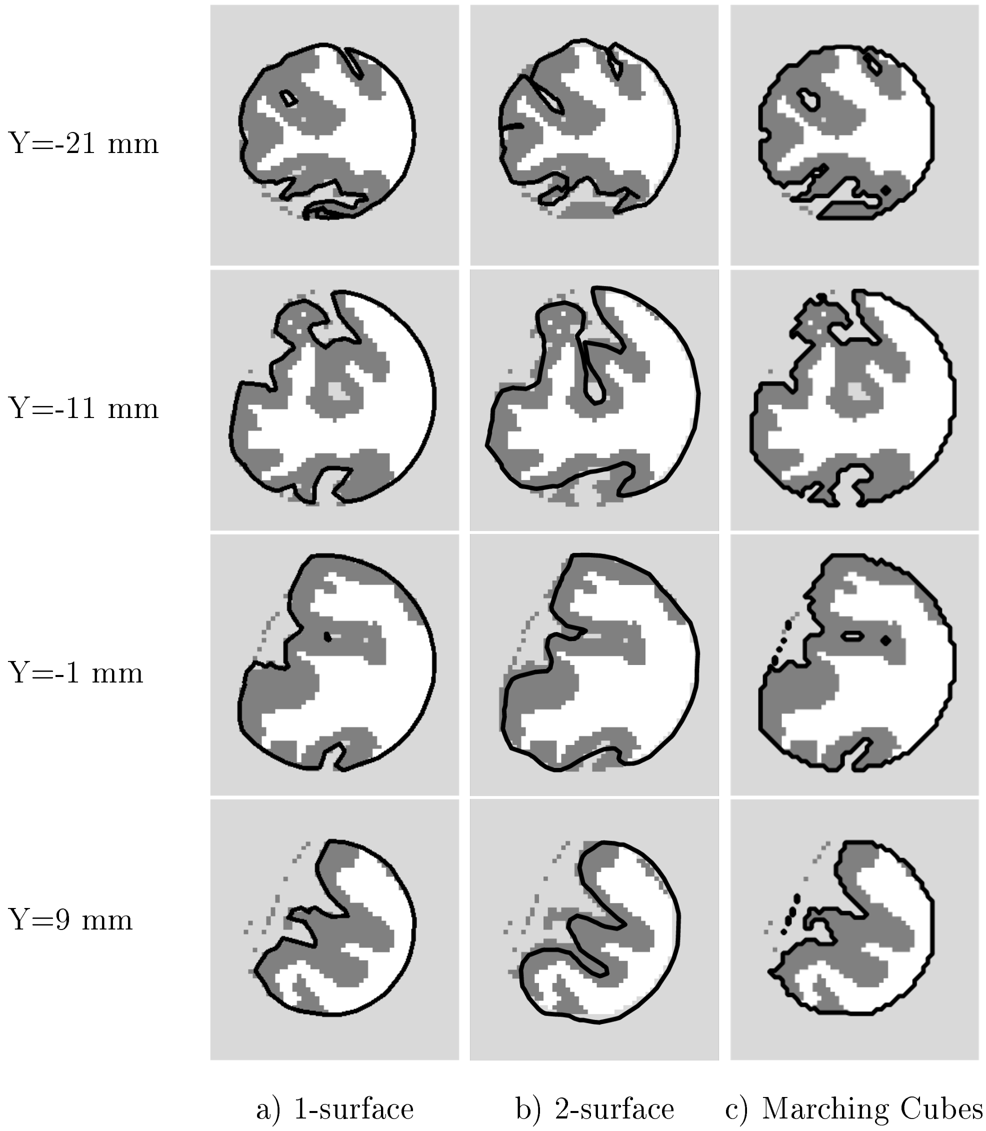
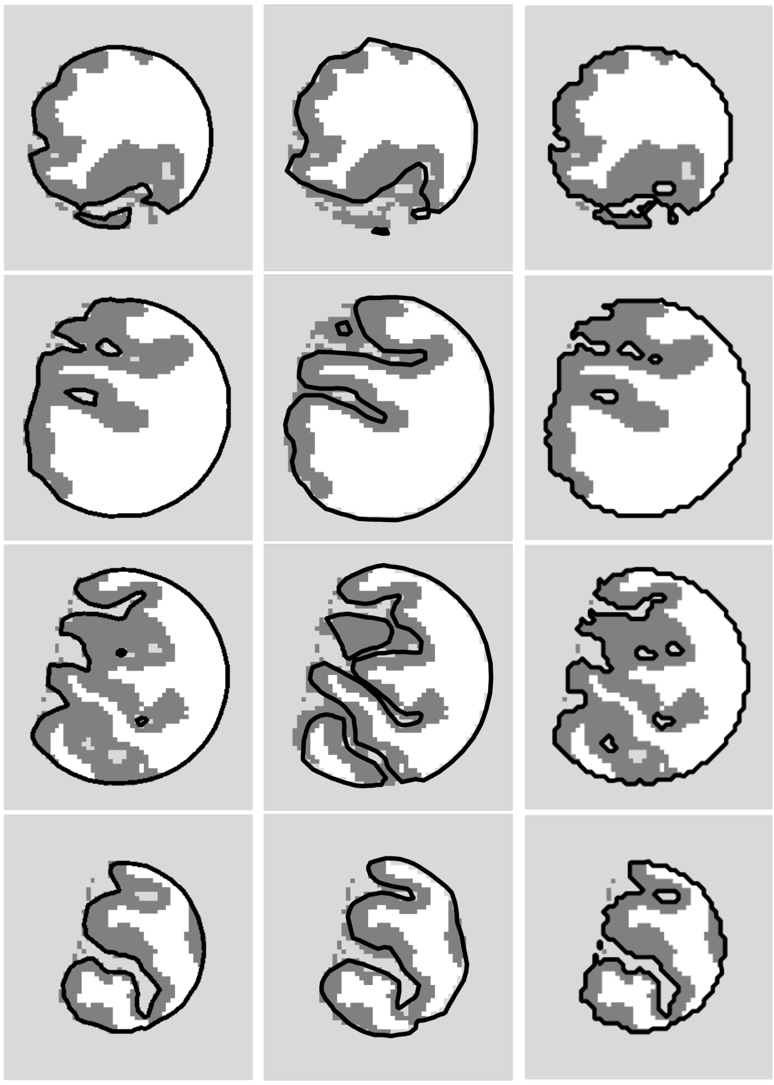


Figure 8.11: Coronal slices through phantom volume and gray-CSF surface produced by a) one-surface deformation b) two-surface deformation c) Marching Cubes algorithm.

problem is to use several $T_{self-intersect}$ terms, with increasing weights:



Z=19 mm

Z=29 mm

Z=39 mm

Z=49 mm

a) 1-surface b) 2-surface c) Marching Cubes

Figure 8.12: Transverse slices through phantom volume and gray-CSF surface produced by a) one-surface deformation b) two-surface deformation c) Marching Cubes algorithm.

Figure 8.13: Three orthogonal slices through sulcal extremity and gray-CSF surface produced by the two-surface deformation.

$$
\begin{aligned}
&0.1\ T_{self-intersect}(0.50) + \\
&1\ T_{self-intersect}(0.45) + \\
&10\ T_{self-intersect}(0.40) + \\
&100\ T_{self-intersect}(0.35) + \\
&1000\ T_{self-intersect}(0.30) + \\
&10000\ T_{self-intersect}(0.25) + \\
&100000\ T_{self-intersect}(0.20) + \\
&1000000\ T_{self-intersect}(0.15) + \\
&10000000\ T_{self-intersect}(0.10),
\end{aligned}
$$

where $T_{self-intersect}(d)$ refers to a self-intersection term that is non-zero whenever two triangles are less than d millimetres apart. The effect of this is to gradually increase the dominance of the self-intersection term as needed to prevent self-intersections. Another way to achieve the same effect is to replace the quadratic function inter-triangle distance in the $T_{self-intersect}$ term with higher order functions. However, the method of using multiple quadratic terms is already available in the current framework and still provides an objective term that is C^1 continuous. In addition, several self-intersection terms require only slightly more computation time than one term. The multiple term method is also used for the $T_{surface-surface}$ component. This method has been used with an identical set of terms to avoid intersections in almost all of the examples presented in this dissertation, ranging from objects of diameter 20 millimetres to around 200 millimetres. Thus, one choice of weights for the terms, $T_{self-intersect}$ and $T_{surface-surface}$, will generally suffice for a wide range of deformation tasks, minimizing the user specification required.

Having defined the intersection term weights, one can now devise weights for the remaining terms. The process starts with the one-surface case, using the set of $T_{self-intersect}$ terms defined above, the $T_{boundary}$ with a weight of one, and the $T_{stretch}$ term with an arbitrarily high weight. Deformations are repeated with decreasing values of the $T_{stretch}$ weight until the desired closeness to the boundary is achieved. Optionally, a weight for the curvature term may be found in a similar fashion, or depending on the segmentation task, one or both of the $T_{stretch}$ and $T_{curvature}$ terms may be removed from the objective function. Having defined weights for all the single surface terms, the weight for the $T_{vertex-vertex}$ term is computed similarly, starting with an arbitrarily high value.

The efficiency with which one chooses weights by this method depends on how precise a weight must be in order to achieve a desired deformation, and on whether the resulting surfaces respond smoothly to the choice of weights. The task of fitting a small phantom brain from section 8.2 is revisited here with a view towards understanding the effects of varying the objective term weights. The set of weights used in the deformation problem was selected by the trial-and-error method described. The weight values are now used as the starting point for another set of experiments. For each class of objective term, the deformation is repeated with the relevant weights scaled by a different factor each time, holding the weights of the other terms constant. The scale values used are

$$0,\ 0.01,\ 0.1,\ 1,\ 10,\ \text{and } 100,$$

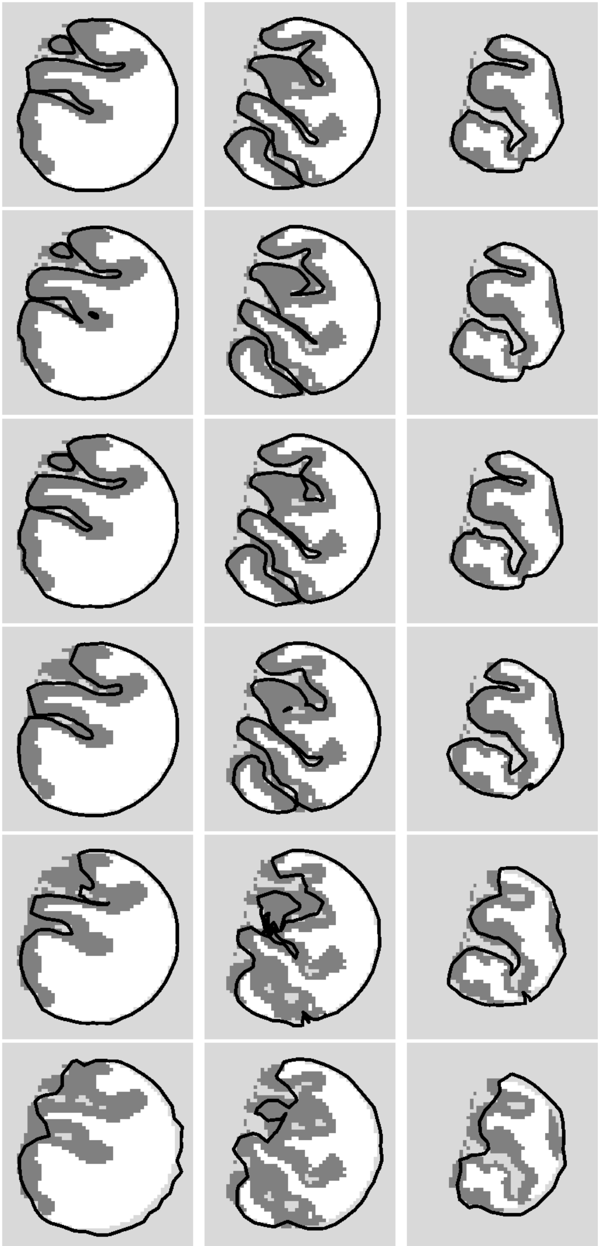
where a zero value indicates removal of the term from the objective function. Cross sections of the results for each of the six classes of objective term are presented in Figs. 8.14, 8.15, 8.16, 8.17, 8.18, and 8.19. The number of triangles and surface areas for each case are presented in Table 8.3.

The effect of changing the weights of the $T_{self-intersect}$ or $T_{surface-surface}$ terms is minimal, in terms of the number of triangles, surface area, and visual appearance of the surface. This indicates that the use of sets of increasing weights may alleviate the need to choose weights for inter-surface intersection and self-intersection constraints.

The behaviour of the weighting of the $T_{vertex-vertex}$ term is interesting, because the implementation actually consists of two separate terms. The first $T_{vertex-vertex}$ term encourages the two surfaces to be five millimetres apart. The second $T_{vertex-vertex}$ term constrains the inter-surface distance to stay within the range of three to seven millimetres by using a much higher weight. When the $T_{vertex-vertex}$ term is removed from the objective function, each surface is effectively deformed independently, and the resulting outer surface of the two-surface deformation is equivalent to that produced by the one-surface deformation. However, for the weight scales of 0.1 and 0.01, it appears that the first $T_{vertex-vertex}$ is not weighted high enough to make the surfaces stay near five millimetres apart, but the high value of the second $T_{vertex-vertex}$ term causes areas of high curvature in the objective function, preventing the minimization from proceeding deep into the sulci. Then, at the highest three values, the first $T_{vertex-vertex}$ is now strong enough to keep the two surface close to five millimetres apart, and the second $T_{vertex-vertex}$ term rarely comes into play. Further investigation is required to fully understand this behaviour, but it is possible that the use of sets of increasing weights such as those involved in the self-interection term may alleviate this problem. Notwithstanding this perplexing behaviour, it is encouraging to note that the results for the weight scales of 1, 10, and 100 are relatively stable.

The behaviour of the segmentation with respect to the $T_{boundary\_dist}$ term appears to have two distinct states. Below a scale value of 1, the term is too small to have any effect on the deformation, and the surface does not change from its initial state of a sphere. For weight scales of 1, 10, and 100, the results are relatively consistent.

The effects of the weights of the $T_{stretch}$ and $T_{curvature}$ terms are almost identical to each other. For the weight scales of 0, 0.01, 0.1, and 1, good segmentation results are achieved. Above the scale value of 1, a relatively poor interpolation of the boundary is achieved, due to the smoothing effects of the highly-weighted stretch or curvature term. The relatively good performance of the algorithm when either component is removed completely indicates one or both terms may be unnecessary in the deformation procedure. This observation resulted in a secondary experiment, to view the effects of leaving both the stretch and curvature terms out of the objective function. The resulting surface of 6682 triangles and a surface area of 13,983 square millimetres is presented in Fig. 8.20. Although this result appears to provide similar or better results, as compared to the previous deformations, there are regions where the surface contains several tight folds not corresponding to object boundaries. It appears that either the stretch or curvature term must be left in the objective function in order to prevent this type of irregularity.



scale = 0

scale = 0.01

scale = 0.1

scale = 1

scale = 10

scale = 100

Z = 29 mm　Z = 39 mm　Z = 49 mm

Figure 8.14: Effect of scaling $T_{stretch}$ weights.

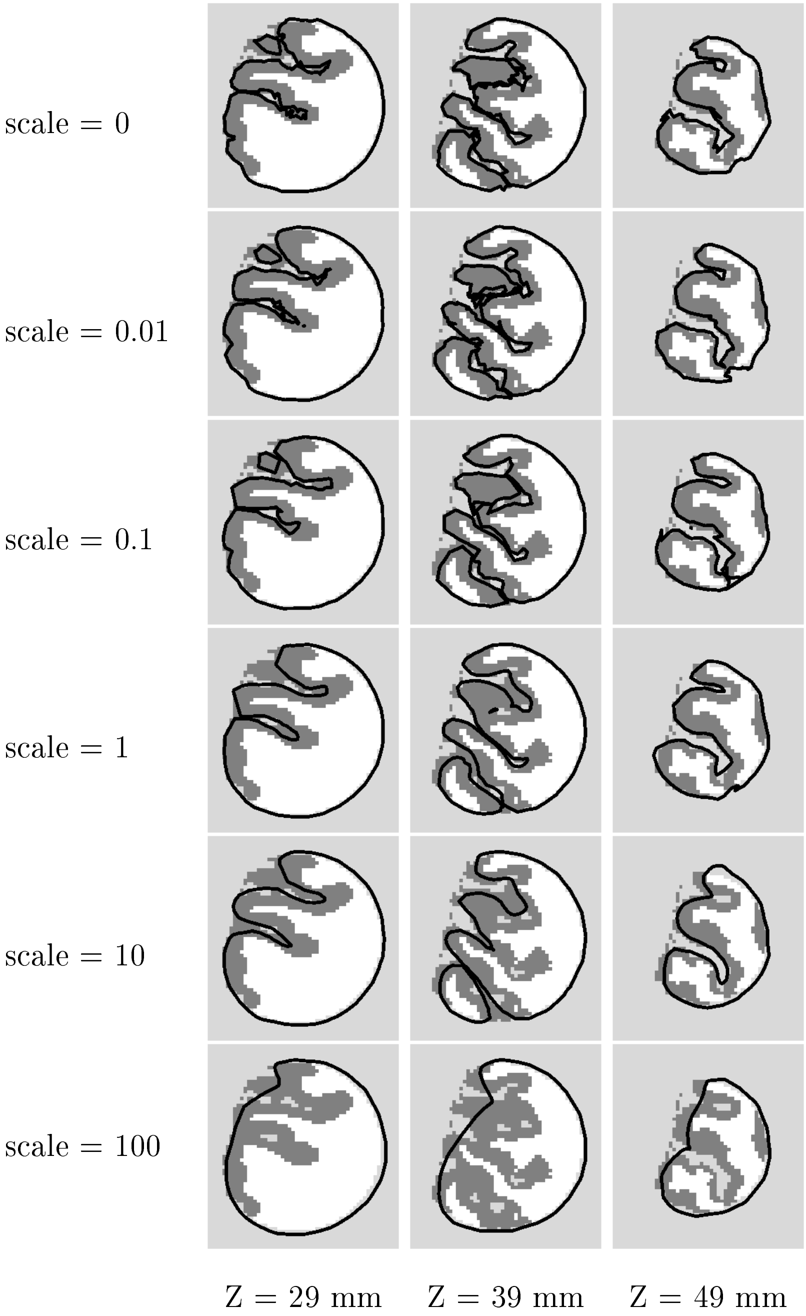
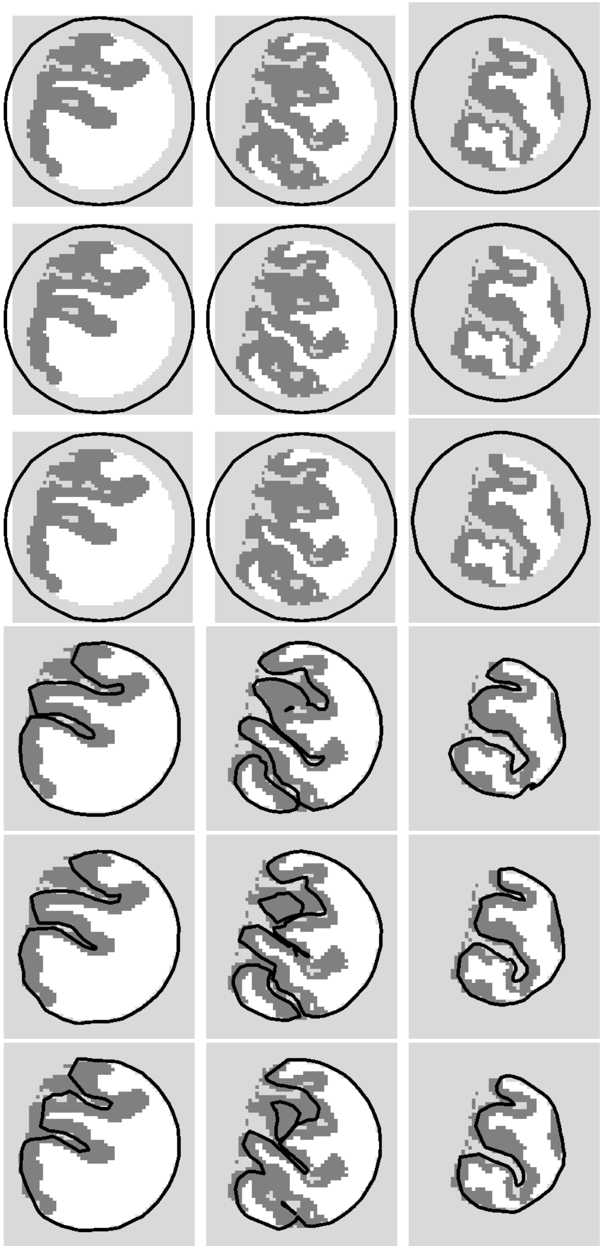


Figure 8.15: Effect of scaling $T_{curvature}$ weights.



scale = 0

scale = 0.01

scale = 0.1

scale = 1

scale = 10

scale = 100

Z = 29 mm   Z = 39 mm   Z = 49 mm

Figure 8.16: Effect of scaling $T_{boundary\_dist}$ weights.

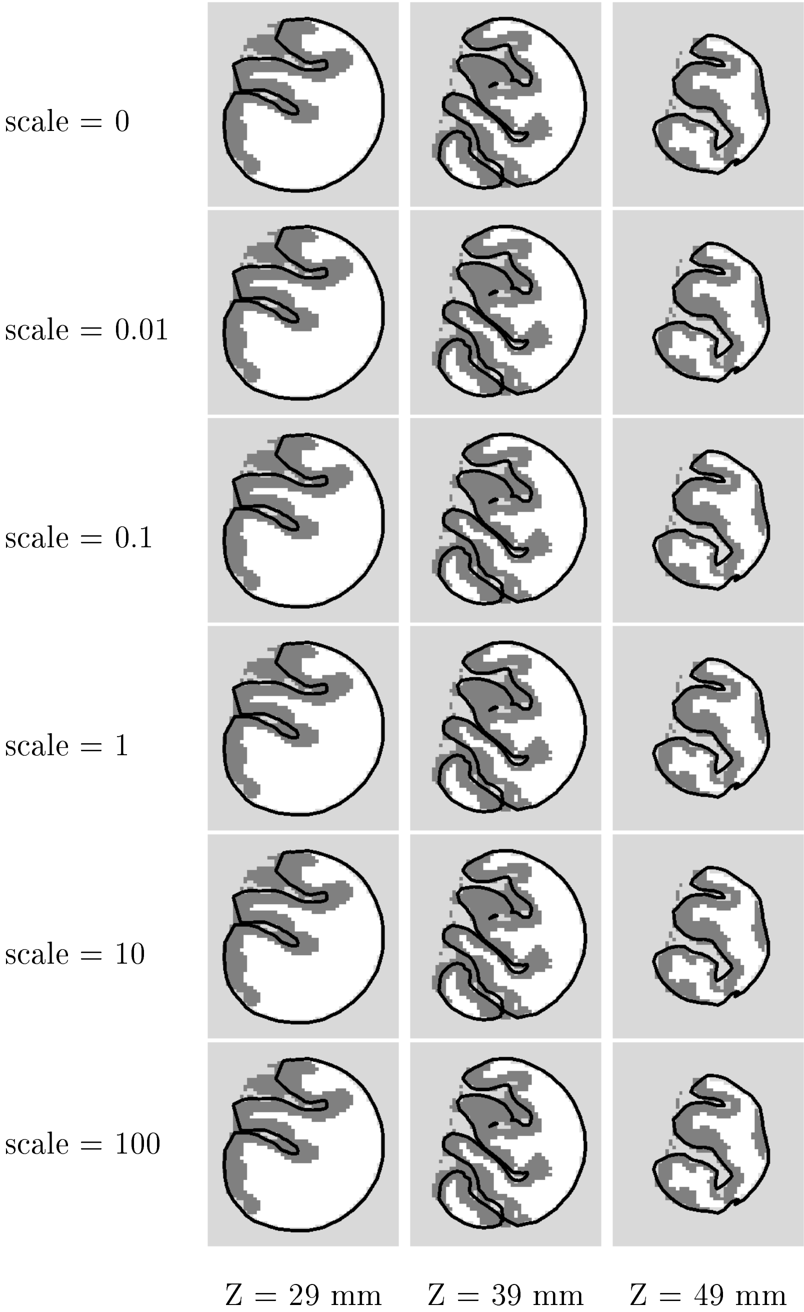


Figure 8.17: Effect of scaling $T_{self-intersect}$ weights.

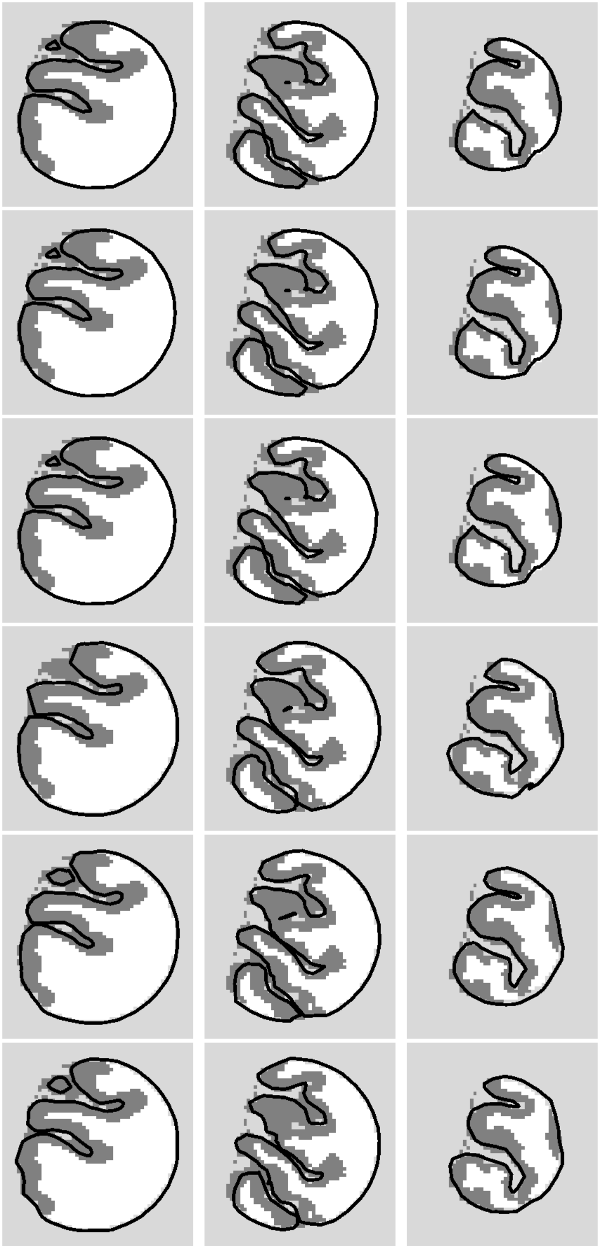


Figure 8.18: Effect of scaling $T_{surface-surface}$ weights.

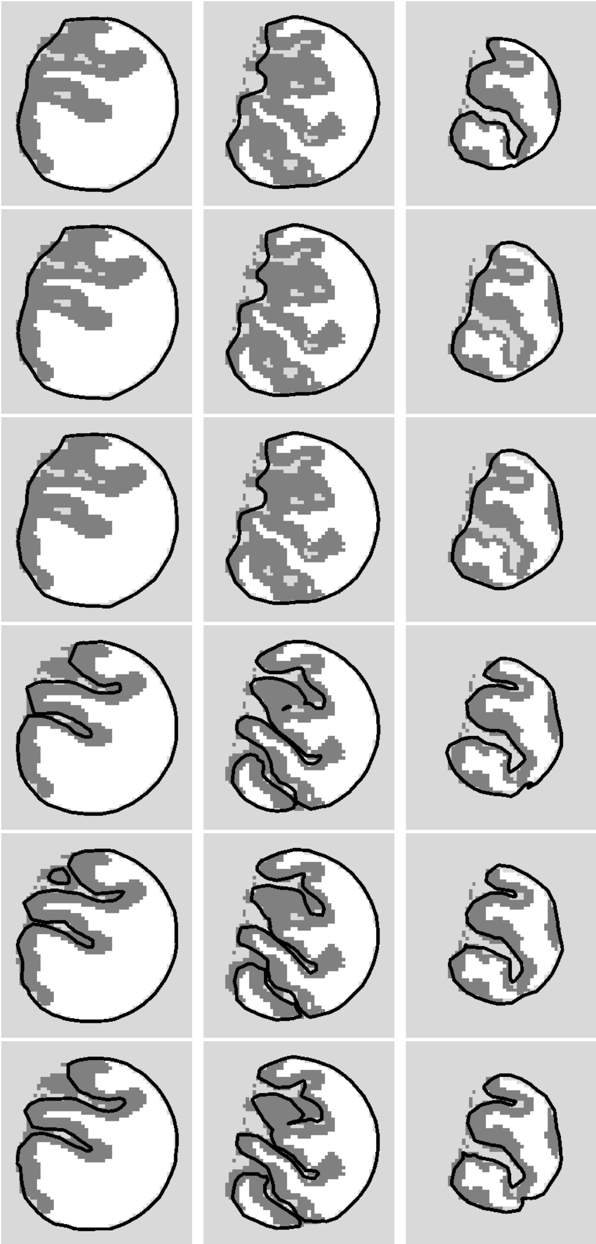


Figure 8.19: Effect of scaling $T_{vertex-vertex}$ weights.

Term	Weight Scale Factor					
	0	0.01	0.1	1	10	100
$T_{stretch}$	11900	11400	11800	11200	10200	8900
$T_{curvature}$	13700	13700	12500	11200	9800	7700
$T_{boundary\_dist}$	11100	11100	11100	11200	11300	11500
$T_{self-intersect}$	11200	11200	11200	11200	11200	11200
$T_{surface-surface}$	10800	10800	10800	11200	11200	11300
$T_{vertex-vertex}$	10300	7800	7800	11200	11200	11200

Surface area, in mm^2, as a function of scaling of objective term weights.

Term	Weight Scale Factor					
	0	0.01	0.1	1	10	100
$T_{stretch}$	6498	4944	5160	3422	2434	1282
$T_{curvature}$	3822	4058	3548	3422	3486	2360
$T_{boundary\_dist}$	1280	1280	1280	3422	3950	3894
$T_{self-intersect}$	3422	3422	3422	3422	3422	3422
$T_{surface-surface}$	3508	3508	3508	3422	3444	3570
$T_{vertex-vertex}$	3310	2376	2376	3422	3532	3546

Table 8.3: Number of triangles as a function of scaling of objective term weights.

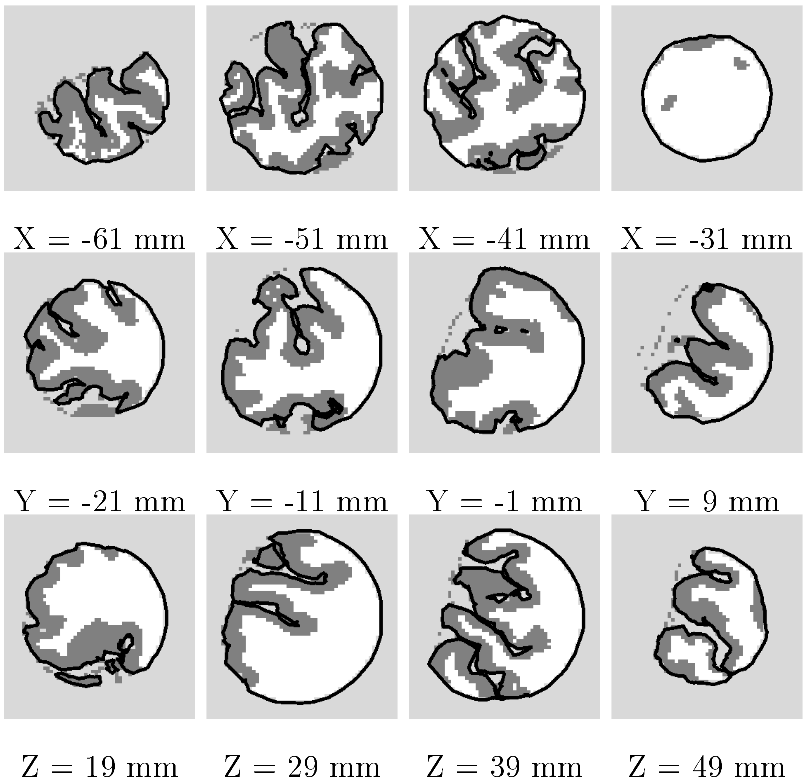


Figure 8.20: Cross sections of surface deformed without $T_{stretch}$ or $T_{curvature}$.

Chapter 9

Experiments on Real Neuroanatomical Data

Having performed validation of various aspects of the NODE deformation technique, the method is now applied to real neuroanatomical problems. The ultimate goal of this dissertation is to provide a method which can provide quantitative answers to basic neuroanatomical questions about the human cortical surface. The problem is to create a surface representation of the human cortical surface that localizes the depths of the cerebral sulci even in those places where partial volume effects confound the identification of gray matter boundaries. Although it is more difficult to accurately measure the performance of the algorithm in this context, it is possible to realize quantitative and qualitative assurances of the utility of the surface deformation. We begin by describing the data and method used.

9.1 Data

The data used for these experiments come from a pool of MR scans of 102 normal volunteers. T1-, T2-, and PD-weighted images at one millimetre isotropic resolution are acquired for each subject. The T1-weighted volumes are acquired using a sagittal volumetric 3D RF-spoiled gradient echo sequence with TR/TE=18ms/10ms, flip angle = 30 degrees, and 1 signal average. The PD- and T2-weighted data are acquired as two 2D multiple slice, dual-echo, fast spin-echo (FSE) datasets with TR/TE1/TE2=3300ms/35ms/120ms. The total scanning time is about 30 minutes. Post-processing on these datasets consists of RF inhomogeneity-correction, linear registration into stereotaxic space, and tissue classification.

9.2 Method

The method consists of the following two sequential steps. The first step involves locating the cerebral white matter voxels similar to the method of Dale and Sereno [DS93]. This is achieved by fitting a pair of low resolution (5120 triangles) average surfaces to the gray-CSF boundary and gray-white boundary using the two-surface method described in section 8.2, with the modification that larger weights are assigned to the terms, $T_{stretch}$ and $T_{curvature}$. These two terms provide a strong constraint on the shape of the deforming model, so that it maintains the global shape features of the average cortex, finding the best fit of this shape to the image volume. All white matter voxels outside the resulting gray-CSF surface are then labeled as gray matter as shown in Fig. 9.1.

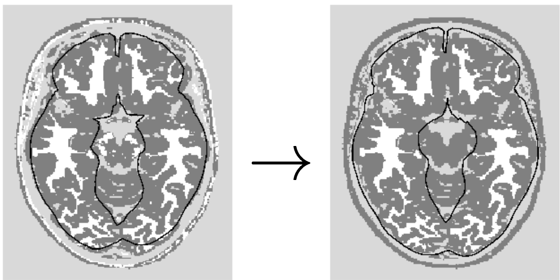


Figure 9.1: Step 1 in segmentation of gray-CSF boundary: masking out non-cerebral white matter voxels.

The second and final step fits a pair of average surface models to the masked volume produced by the first step. The absence of white matter voxels outside the cerebral cortex allows the procedure to be guided by the gray-white boundary, using the two-surface deformation parameters devised in section 8.2. A multi-scale approach is used, where initial surfaces of 320 triangles are adaptively subdivided as the deformation progresses. The initial maximum edge length is 40 millimetres, and the final maximum edge length is four millimetres, resulting in surfaces of almost 100 000 triangles. This process requires about 100 hours of computer time on a Silicon Graphics Origin 200 R10000 processor running at a clock-rate of 180 megahertz.

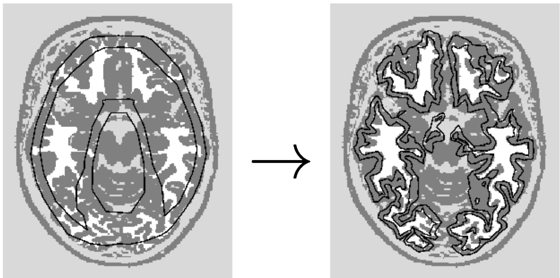


Figure 9.2: Step 2 in segmentation of gray-CSF boundary: simultaneous deformation of two surfaces to fit volume with non-cerebral white matter masked out.

9.3 Validation Against Manual Segmentation

In order to test the segmentation aspect of this technique, a realistic segmented dataset is required. Since it can take up to several days to segment the entire cerebral cortex manually at the voxel level, a large number of datasets is not available. However, one excellent dataset is available for use in validating the segmentation. An experienced neuroanatomist in the McConnell Brain Imaging Centre, Dr. Noor Kabani, has previously spent several months creating a detailed voxel-by-voxel labeling of a normal human MR volume into more than 70 distinct anatomical regions, constituting a detailed atlas of human neuroanatomy. There are several choices of methods for relating a cortical surface model to this atlas. One possibility is to compare points on the surface with voxels in the atlas which are on gray-CSF boundaries. A more straightforward way is to compare the set of voxels between the two deformed surfaces to the set of cortical gray voxels in the atlas. Ideally, both sets of voxels should correspond to the entire cortical gray matter.

Slices combining the cortical gray matter voxels labelings created by the neuroanatomist with the two-surface deformation method are presented in Fig. 9.3. A voxel can be classified into one of four classes, true-positives, true-negatives, false-positives, and false-negatives, depending on whether the voxel was labeled as cortical gray by each of the two methods. Table 9.1 shows the number of voxels corresponding to each of the four classes. It is evident that the set of voxels between the two deforming surfaces does not contain the full set of true cortical gray voxels; only 67 % [1] of the true gray voxels were correctly identified. The reason for this is that the choice of five millimetres for the distance between the gray-CSF and gray-white surfaces is not optimal. An indication of this arises from the observation that the average distance between the two deformed surfaces, 5.34 millimetres, is greater than the five millimetres constraint chosen. The data is trying to pull the two surfaces further apart, but the inter-surface $T_{vertex-vertex}$ term keeps this from happening. Based on the results of this preliminary experiment, a method of creating an improved inter-surface distance constraint is investigated in a subsequent section on cortical thickness maps.

It is also informative to look at the set of false-positive voxels, which are voxels that were incorrectly labeled as cortical gray by the two-surface deformation. The number of false-positive voxels is 127,004, which is 15 % [2] of the total number of cortical gray voxels in the manual segmentation. About two thirds of the false-positive voxels are in the problematic region of the inferior surface of the cortex. The manual labeling of the cortical gray voxels leaves a large hole in the inferior portion of the cortex, whereas the closed topology of the deforming surface crosses over this hole. The Y=-

[1] 557454 / (557454 + 270585) $\times$ 100 % = 67 %

[2] 127004 / (557454 + 270585) $\times$ 100 % = 15 %

30 mm slice of Fig. 9.3 shows this crossing over as the light gray false-negative region spanning the two hemispheres. Whether the cortical surface should be represented as with or without this hole is a subject of neuroanatomical debate, but investigation into choosing a deforming model with an appropriate neuroanatomical topology is required. If the false-positive voxels arising from the closing of the inferior hole in the cortex are ignored, the remaining set of false-positive voxels corresponds to about 5 percent of the total number of cortical gray voxels.

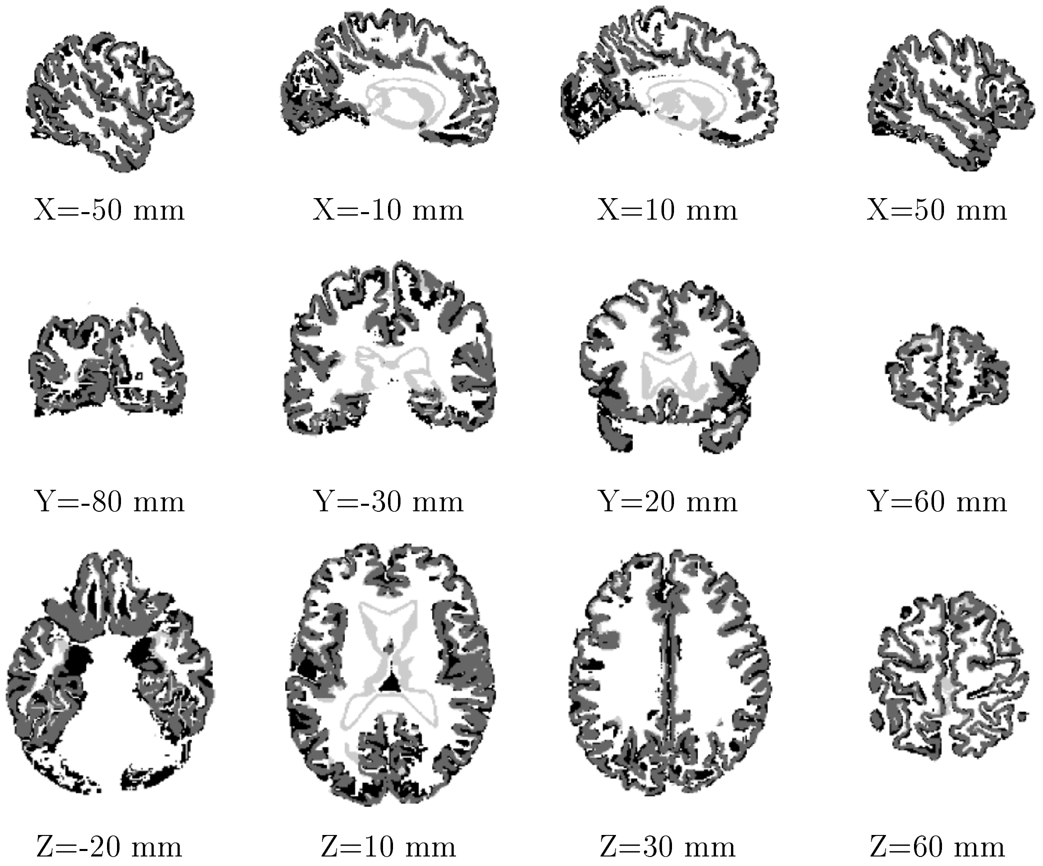


Figure 9.3: Comparison of manual labeling to automatic labeling. (dark gray = true-positives, light gray = false-positives, and black = false-negatives)

	True Gray	True Non-Gray
NODE Gray	557454	127004
NODE Non-Gray	270585	6154094

Table 9.1: Number of voxels labeled as cortical gray and non-cortical gray by NODE algorithm and manual labeling (assumed to be the "truth").

9.4 Comparison to Other Methods

The result of the two-surface deformation is compared with that of the one-surface deformation and the Marching Cubes algorithm. In the latter two cases, the surface mask computed as the first step of the procedure in section 9.2 is used to mask voxels exterior to the cortex, in order to create a volume that does not contain cerebellum or skin voxels which would confuse the algorithms. The surface models created by the three algorithms are shown in Figs. 9.4, 9.5, and 9.6. Slices through the volume and gray-CSF surface produced by each of the three algorithms are shown in sagittal orientation in Fig. 9.7, coronal orientation in Fig. 9.8, and transverse slices in Fig. 9.9. The number of triangles produced and the surface area of the gray-CSF surface for each algorithm is presented in Table 9.2. It is readily apparent that the single surface model fails to interpolate the depths of the sulci. The dual surface method produces a cerebral cortex model with about the same surface area as that of the Marching Cubes, but the single surface method produces a much lower surface area. There are many places where the Marching Cubes algorithm seems to produce better results It would be informative to compute the dual surface model at a resolution similar to that of the Marching Cubes, in order to better relate the two methods, but it is encouraging that the dual surface model provides a partial volume corrected surface with about the same surface area as that of the Marching Cubes algorithm, using only a tenth the number of triangles.

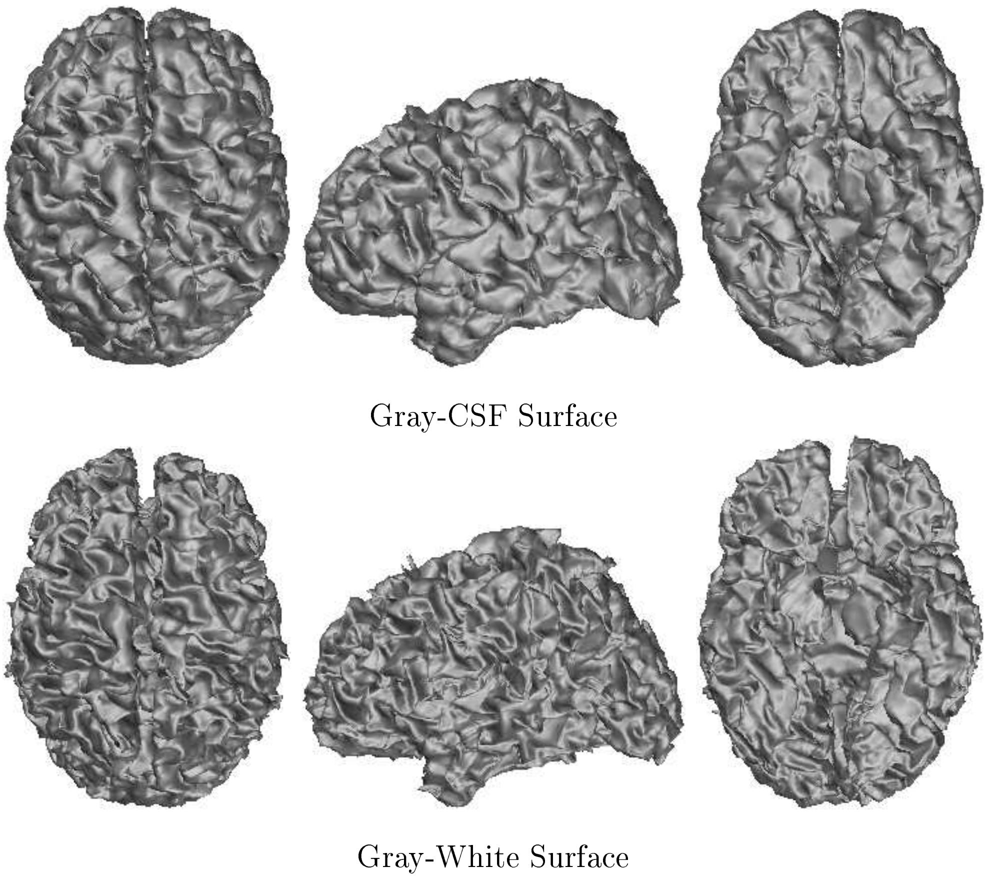


Figure 9.4: Top, left, and bottom views of gray-CSF and gray-white surfaces produced by the two-surface deformation model.

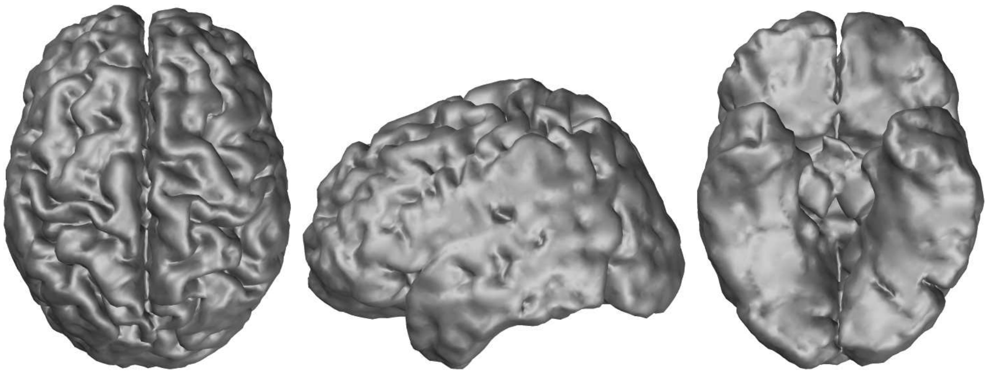


Figure 9.5: Top, left, and bottom views of gray-CSF surface produced by the one-surface deformation model.

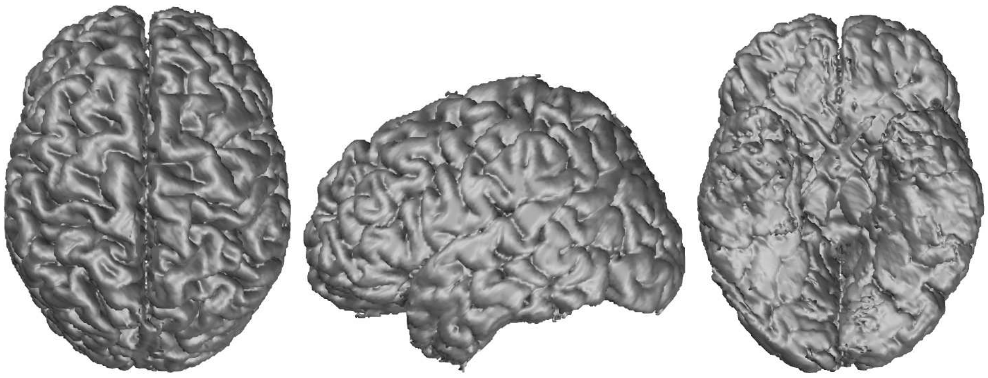


Figure 9.6: Top, left, and bottom views of gray-CSF surface produced by the conventional Marching Cubes algorithm.

Method	Number of Triangles	Surface Area, in mm^2
one-surface deformation	34352	121928
two-surface deformation	66176	226272
Marching Cubes	677880	240397

Table 9.2: Results of gray-CSF segmentation on individual MR image.

9.5 Surface Averaging and Flattening

An interesting characteristic of polyhedra that have been deformed by perturbing the vertices, such as in the NODE algorithm, is that there is a direct mapping between points on the initial polyhedron and the deformed polyhedra. The position of any vertex, $\bar{x}_v$ on one surface maps directly to the position of the corresponding vertex, $\bar{\hat{x}}_v$, on the other surface. Non-vertex points of a polyhedron are mapped by linear interpolation of the mappings of the three vertices defining the triangle in which the point resides.

The ability to map three dimensional positions between polyhedra has several practical uses. One application is the creation of an average surface from a set of surfaces. This is done by taking the centroid of each set of corresponding vertices across the surfaces. This implicitly assumes that the vertices of the surfaces are

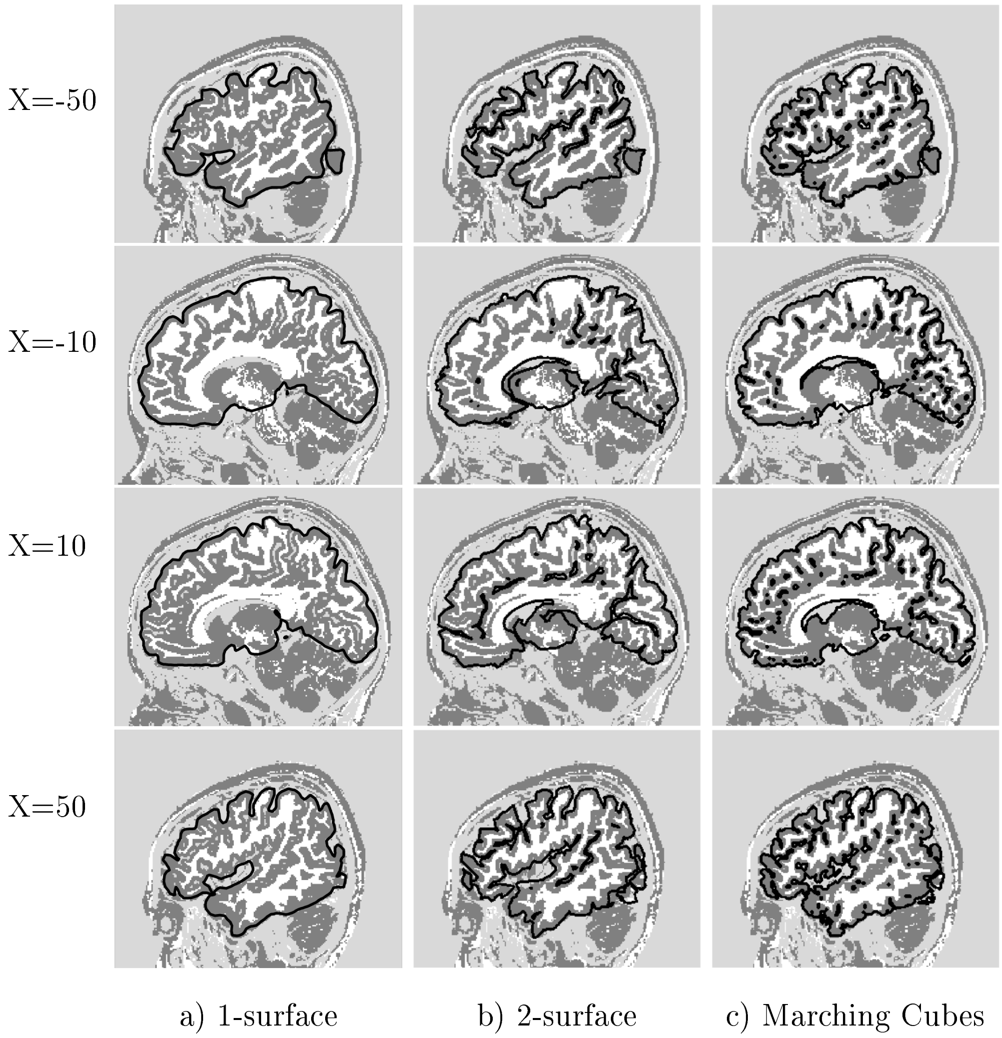


Figure 9.7: Sagittal slices through volume and gray-CSF surface produced by: a) one-surface deformation. b) two-surface deformation c) Marching Cubes algorithm.

homologous, that is, the i'th vertex of any cortical surface corresponds to a particular anatomical position. Although this is not actually true for the surfaces created by the NODE algorithm, in practice, there is some correlation of vertices, and the mean surface can have a practical use. Figure 9.10 depicts three views of the average of 102 surfaces created by the 1-surface NODE algorithm at a resolution of 81,920 triangles. This surface encapsulates the main features of any individual brain and thus it may be useful as a model for the cortical surface deformation.

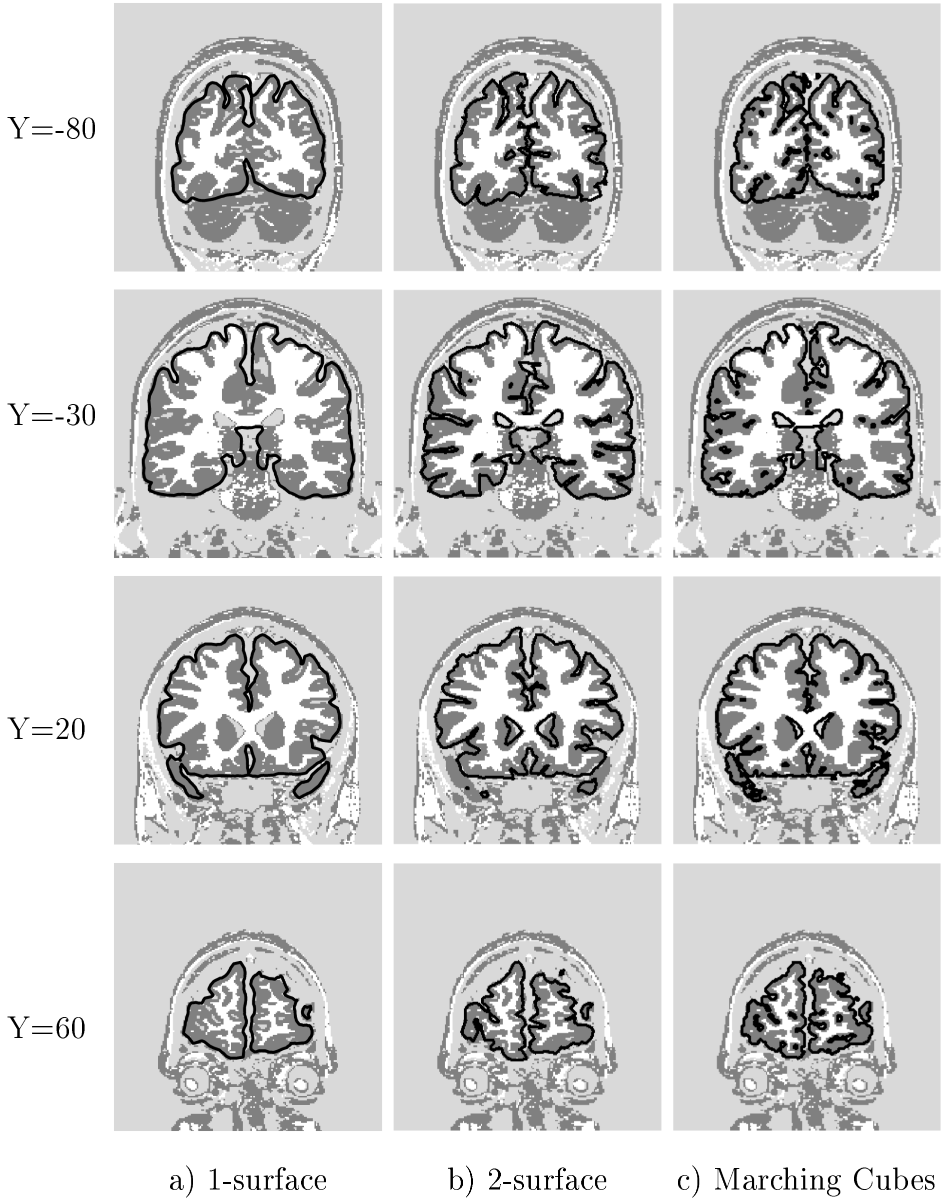


Figure 9.8: Coronal slices through volume and gray-CSF surface produced by: a) one-surface deformation. b) two-surface deformation c) Marching Cubes algorithm.

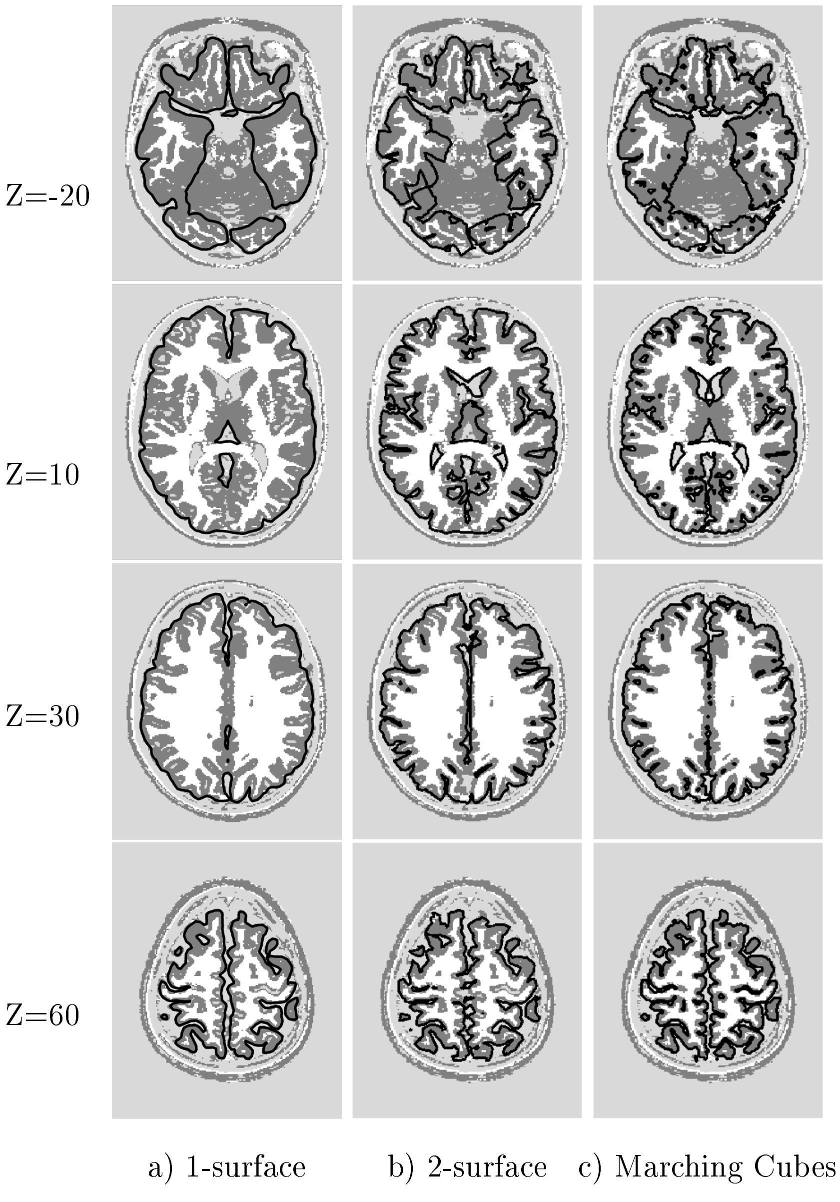


Figure 9.9: Transverse slices through volume and gray-CSF surface produced by: a) one-surface deformation. b) two-surface deformation c) Marching Cubes algorithm.

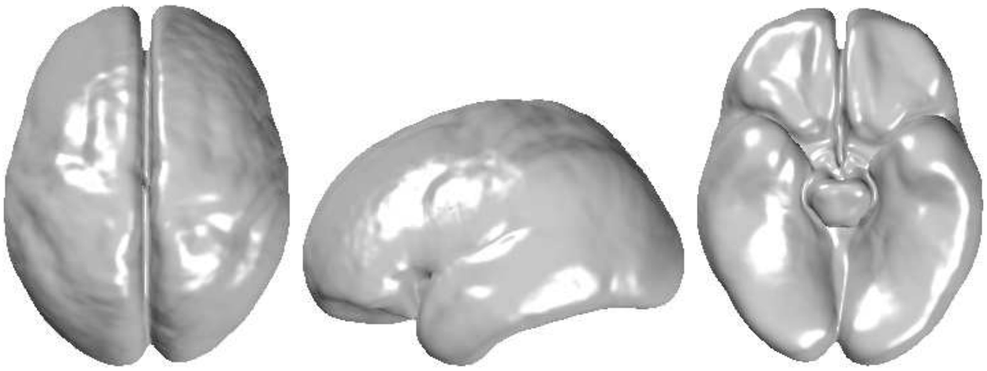
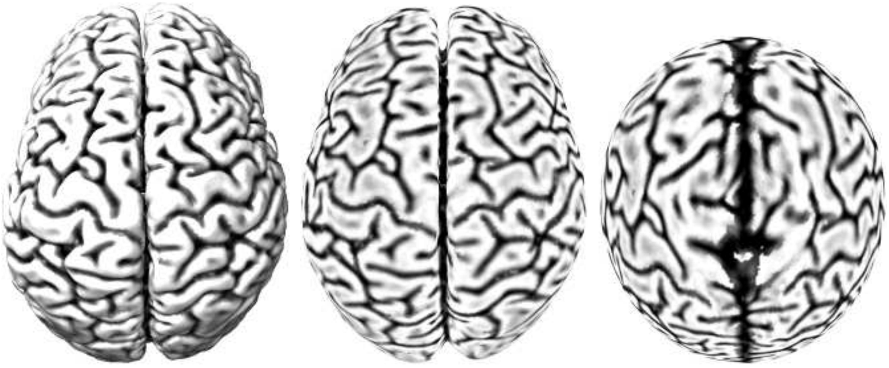


Figure 9.10: Top, left, and bottom views of average of 102 cortical surfaces produced by the one-surface NODE algorithm.

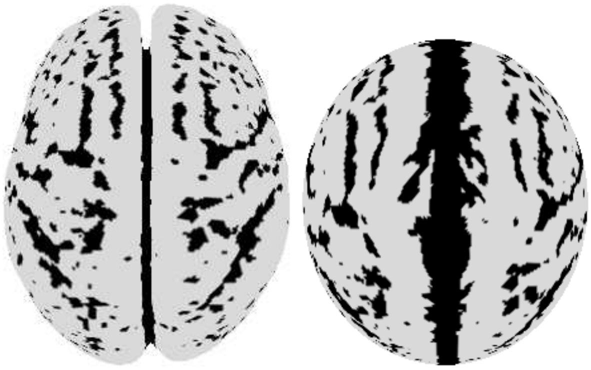
Another use of inter-polyhedral mapping is in the conversion of complex surfaces into smoother representations in order facilitate subsequent neuroanatomical analysis. Figure 9.11 shows the results of mapping an individual cortex to the average surface and an ellipsoid of the approximate shape of the brain. The surfaces are coloured with gray-scale values based on the local curvature, where dark areas correspond to the depths of sulci and lighter areas correspond to gyri. Figure 9.12 shows the average curvature of the 102 cortical surfaces mapped on to the average surface and on to an ellipsoid. It is interesting to note that even without any explicit homology constraints, the positions of several sulci are consistent enough to show up on the average of 102 surface curvatures. The flattened representations of individual or mean surfaces presented here provide an alternate coordinate system, which may prove useful for neuroanatomical analysis such as examining sulcal topology.

The averaging of vertex positions and curvature across surfaces relies on the topology of the triangulations being exactly the same. However, the gray-CSF surfaces created by the NODE algorithm have different triangulations, because the adaptive triangulation of the polyhedra results in different numbers of triangles for different sets of image data. However, the inter-polyhedral mapping can be used to convert each of the individual triangulations to a common triangulation topology. Whenever a surface is subdivided during the deformation, a copy of the initial model is sub-



a) b) c)

Figure 9.11: Curvature of individual cortex a) on the cortex, b) mapped to the average of 102 surfaces, and c) mapped to an ellipsoid.



a) b)

Figure 9.12: Average vertex curvature across 102 surfaces mapped to a) the average of 102 surfaces, and b) an ellipsoid.

divided in the same fashion. The resulting triangulation of the model matches the triangulation of the deformed surface, but is geometrically the same set of points as the initial model, and is referred to here as a re-parameterized model. The deformed surface can be re-triangulated to match the initial model by mapping vertices in two steps. The first step takes the coordinates of a vertex on the initial model and finds the triangle on the re-parameterized model which contains this point. Using the vertex indices of this triangle, the second step interpolates a geometric position within the corresponding triangle on the deformed surface. The resulting re-triangulated surface thus has the triangulation topology of the initial model, but the vertex positions define an object geometrically similar to the deformed surface. This method is used in the next section to create an average of several different sized surfaces.

9.6 Cortical Thickness Maps

The two-surface NODE algorithm has been used to automatically create 10 sets of gray-CSF and gray-white surface models from 10 normal human MR datasets. Each gray-CSF surface was re-triangulated to a resolution of 81,920 triangles, and the resulting average surface was computed (Fig. 9.13). A preliminary version of a cortical thickness map can then be produced by taking the mean distance of the surface at each vertex, shown superimposed on the average surface and on an ellipsoid in Fig. 9.14. The average cortical thickness of the 10 pairs of surfaces ranges from 5.78 to 6.53 millimetres. Although this data suffers from insufficient sample points ($N = 10$) and from the fact that there is little enforcement of homology across the surfaces, the resulting cortical thickness map demonstrates the applicability of the NODE algorithm to neuroanatomic analysis. In addition, the thickness map may also be used to define the constraining model for subsequent application of the two-surface deformation. This process of iteratively applying the model-based deformation, then using the output to improve the model, may result in increasingly sophisticated methods of cortical surface segmentation.

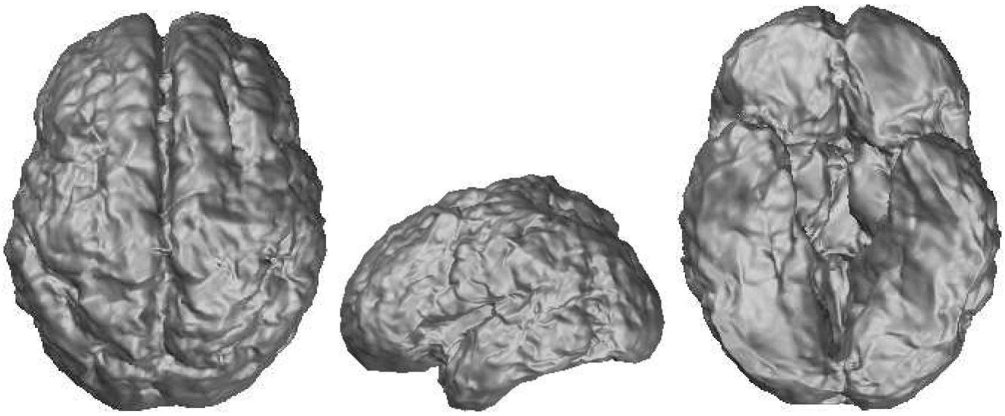


Figure 9.13: Top, left, and bottom views of average of 10 gray-CSF surfaces.

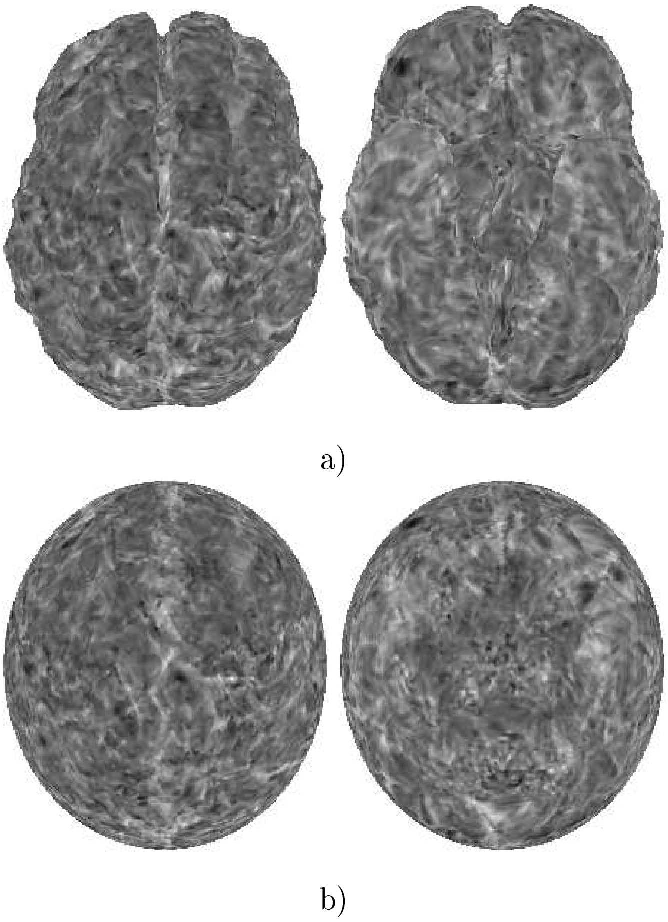


Figure 9.14: Top and bottom views of average cortical gray thickness ($N = 10$) mapped onto a) average surface, and b) an ellipsoid. The thickness ranges from 5.78 mm in the black regions to 6.53 mm in the white regions.

Chapter 10

Conclusion

10.1 Summary

The task of segmenting human cortical surfaces from three dimensional magnetic resonance images has been addressed. A method of deforming a polyhedral model by minimizing an objective function has been presented. The technique, referred to as the NODE algorithm, has been shown to provide a solution to locating the cortical surface in the face of partial volume artifacts. This has been achieved through the unique combination of self-intersection testing and inter-surface constraints on a multiple surface deformation model. The major contributions of this work to date include:

- a method of automatically locating the cortical surface from MR images has been presented and evaluated,
- inter-surface and intra-surface constraints have been shown to avoid non-simple topologies in deforming models,
- the use of multiple surface models has been shown to improve the localization of sulci occluded by partial volume effects, and
- a preliminary mean cortical thickness map has been produced, which can be refined and used to improve the segmentation process.

Although this solution represents a major step forward, it is by no means a final solution, and further work is required.

10.2 Further Work

The NODE method of two-surface deformation has been shown to achieve good results on a small brain phantom and on individual brain datasets. However, ways to improve the localization of the cerebral cortex should be investigated, as well as explorations of the deformation method at higher resolutions. The most obvious short-coming with the NODE technique is the necessity to choose objective terms and weights to achieve a particular image segmentation task. Although the objective function has been formulated to consist of components that have intuitive notions, such as stretching and bending, it is still necessary to investigate a range of weight combinations for each of the objective terms. For more complicated tasks involving several surface models, the number of weighting factors that must be chosen can be 10 or more, and the combinatorics of trial-and-error determination of optimum values requires considerable investigation. Some simple exploration of the sensitivity of results to the choice of weights has been presented in section 8.3, but further research into methods of automating the choice of weights is warranted.

10.2.1 Better Models of Gray and White Matter

The two-surface model with a preferred inter-surface distance has been shown to be sufficient for improving the segmentation of cortical surfaces. However, it is by no means the most sophisticated model one could devise. As mentioned in section 9.6, one could use the results of this model deformed to a large number of normal datasets to create a model of the gray matter thickness that varies across the domain of the cortex. In addition, adding models of the skin surface, brain stem, cerebellum, and dura to the multiple surface deformation process may also improve the localization of the cortical surface.

10.2.2 Triangle Proximity Query

One open problem that has arisen from this work is the surface proximity query required to avoid self-intersection and inter-surface intersection, described in section 7.3. Given a set of triangles, their three dimensional vertex coordinates and movement vectors, and a distance along the movement vectors, compute the list of triangles within a certain distance of each other. The fact that this query will be asked at various (initially unknown) positions along the movement vector makes this a unique problem that has not been fully addressed in the computational literature, but is potentially a critical problem for surface based deformation methods.

10.2.3 Homology and Shape Matching

One of the limitations of creating average surfaces by taking the mean of corresponding vertices is that the enforcement of homology of vertices is minimal at present. Exploration of ways to increase the homology of the deformed surfaces should be pursued. One possibility is to use curvature and stretch constraints to force the deforming model to remain similar to the shape of the average cortex model, which can result in matching regions of the model with similarly shaped regions of the image. Other possibilities include adding constraints based on features extracted from the image volumes, such as sulcal positions or anatomical landmarks. It would also be interesting to investigate whether multiple surface models and self-intersection avoidance can improve the results in this area.

10.2.4 Surface Flattening

Another area of considerable interest is that of mapping complex surfaces, such as convoluted cortical surfaces, into visually and mathematically simpler representations, such as flat sheets and spheres. Many such methods start with a complex three dimensional surface and attempt to deform it to become smoother, with constraints on distance and angular changes. It has been observed that these methods can result in flattened surfaces that have areas of overlap where the surface has folded over on itself [CDVE95], and it would therefore be interesting to use self-intersection avoidance to attempt to circumvent this behaviour. The surface deformation as currently described has all the components necessary to perform this type of surface flattening. Setting the desired curvature to zero would provide a flattening force for an arbitrary three dimensional surface. Stretch constraints can be used to help preserve distances between the three dimensional and flattened configurations of the surface. Self-intersection constraints can be used to prevent the problem of the surface folding over on itself during the flattening process. Comparison of this method to conventional techniques is therefore warranted.

As introduced in section 9.5, flattening a surface can also be achieved by inverting the process. The use of the NODE algorithm to deform ellipsoidal models to complex MR images results in a mapping of the resulting polyhedra to the original ellipsoidal models and averages of sets of surfaces. Investigations are warranted to determine how to encourage preservation of distances, angles, and areas in the mapping.

10.3 Conclusion

In conclusion, the goal of this dissertation to solve a particular image understanding problem has been achieved, through a novel combination of techniques. The method is sufficiently general to have a much wider base of application than the specific problem addressed, and investigation into other areas promises equally interesting results.

Bibliography

[AERT92] A. Aggarwal, H. Edelsbrunner, P. Raghavan, and P. Tiwari. Optimal time bounds for some proximity problems in the plane. *Inform. Process. Lett.*, 42(1):55–60, 1992.

[AGP91] Y. Amit, U. Grenander, and M. Piccioni. Structural image restoration through deformable templates. *Journal of the American Statistical Association*, 86(414):376–387, June 1991.

[AS95] David Adalsteinsson and James A. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118:269–277, 1995.

[BB88] Ronen Barzel and Alan H Barr. Modeling system based on dynamic constraints. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 22(4):179–188, Aug 1988.

[BBB87] R.H. Bartels, J.C. Beatty, and B.A. Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling.* Morgan Kaufmann, 1987.

[Bcc89] R. Bajcsy and Kovačič. Multiresolution elastic matching. *Computer Vision, Graphics, and Image Processing*, 46:1–21, 1989.

[Ben79] J. L. Bentley. Decomposable searching problems. *Information Procesing Letters*, 8(5):244–251, 1979.

[BGCK93] K. T. Bae, M. L. Giger, C. T. Chen, and C. E. Kahn Jr. Automatic segmentation of liver structure in ct images. *Journal of Medical Physics*, 20(1):71–8, Jan-Feb 1993.

[BGK92] C. Brechbühler, B. Gerig, and O. Kübler. Surface parameterization and shape description. In *Proceedings Visualization in Biomedical Computing 1992*, pages 80–144, 1992.

[Blo88] Jules Bloomenthal. Polygonization of implicit surfaces. *Computer Aided Geometric Design*, 5(4):341–355, Nov 1988.

[Boo89] F.L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585, June 1989.

[BW90] J. Bloomenthal and B. Wyvill. Interactive techniques for implicit modeling. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, pages 109–116, 1990.

[CAC91] Isaac Cohen, Nicholas Ayache, and Laurent Cohen. Segmenting, visualizing and characterizing 3d anatomical structures with deformable surfaces. *Proceedings of the Annual Conference on Engineering in Medicine and Biology*, 13:1183–1184, 1991.

[Can86a] John Canny. Collision detection for moving polyhedra. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(2):200–209, March 1986.

[Can86b] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, November 1986.

[Car76] M.P. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, New Jersey, 1976.

[CC90] Laurent D Cohen and Issac Cohen. A finite element method applied to new active contour models and 3d reconstruction from cross sections. *Proc 3 Int Conf Comput Vision*, pages 587–591, 1990.

[CC93] Laurent D Cohen and Isaac Cohen. Finite-element methods for active contour models and balloons for 2-d and 3-d images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1131–1147, Nov 1993.

[CCA91] Isaac Cohen, Laurent D Cohen, and Nicholas Ayache. Introducing new deformable surfaces to segment 3d images. In *Proceedings IEEE Comput Soc Conf Comput Vision Pattern Recognition*, pages 738–739, 1991.

[CCA92] I. Cohen, L.D. Cohen, and N. Ayache. Using deformable surfaces to segment 3-d images and infer differential structures. *Computer Vision, Graphics, and Image Processing*, 1992.

[CCR+93] S. Cagnoni, G. Coppini, M. Rucci, D. Caramella, and G. Valli. Neural network segmentation of magnetic resonance spin echo images of the brain. *Journal of Biomedical Engineering*, 15(5):355–62, Sep 1993.

[CDM84] T.S. Curry, J.E. Dowdey, and R.C. Murry. *Christensens's Introduction to the Physics of Diagnostic Radiology.* Lea and Febiger, Philadelphia, USA, 3rd edition, 1984.

[CDVE95] G. J. Carman, H. A. Drury, and D. C. Van Essen. Computational methods for reconstructing and unfolding the cerebral cortex. *Cerebral Cortex*, 5(6):506–17, Nov-Dec 1995.

[CEHP95] DL Collins, AC Evans, C Holmes, and TM Peters. Automatic 3D segmentation of neuro-anatomical structures f rom mri. In Y Bizais, C Barillot, and R DiPaola, editors, *Information Processing in Medical Imaging*, pages 139–152, Brest, France, Aug 1995. IPMI, Kluwer.

[CHPE96] DL Collins, CJ Holmes, TM Peters, and AC Evans. Automatic 3D model-based neuroanatomical segmentation. *Human Brain Mapping*, 3(3):190–208, 1996.

[CHTH93] T.F. Cootes, A. Hill, C.J. Taylor, and J. Haslam. The use of active shape models for locating structures in medical images. In *Proceedings of Information Processing in Medical Imaging*, pages 33–47, Flagstaff, Arizona, June 1993.

[CL88] James S Chen and Wei-Chung Lin. New surface interpolation technique for reconstructing 3-d objects from serial cross-sections. In *Proceedings - International Conference on Pattern Recognition 9th.*, pages 1100–1102, 1988.

[CLL+88] H. E. Cline, W. E. Lorensen, S. Ludke, C. R. Crawford, and B. C. Teeter. Two algorithms for the three-dimensional reconstruction of tomograms. *Journal of Medical Physics*, 15(3):320–7, May-Jun 1988.

[CNPE94] D.L. Collins, P. Neelin, T.M. Peters, and A.C. Evans. Automatic 3d intersubject registration of mr volumetric data in standardized talairach space. *Journal of Computer Assisted Tomography*, 18(2):192–205, March 1994.

[CPDE92] D.L. Collins, T.M. Peters, W. Dai, and A.C. Evans. Model-based segmentation of individual brain structures from mri data. In *Proceedings Visualization in Biomedical Computing 1992*, pages 10–23, 1992.

[CS89] Kenneth L. Clarkson and Peter W. Shor. Applications of random sampling in computational geometry, ii. *Discrete and Computational Geometry*, 4:387–421, 1989.

[CW83] F. Chin and C. A. Wang. Optimal algorithms for the intersection and the minimum distance problems between planar polygons. *IEEE Trans. Comput.*, C-32(12):1203–1207, 1983.

[D8̈8] M.J. Düurst. Additional reference to marching cubes (letter). *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 22:72–73, 1988.

[DB95] Chris Davatzikos and R Nick Bryan. Using a deformable surface model to obtain a shape representation of the cortex. *Proceedings of the IEEE International Conference on Computer Vision 1995*, pages 212–217, 1995.

[DE96] M. T. Dickerson and D. Eppstein. Algorithms for proximity problems in higher dimensions. *Comput. Geom. Theory Appl.*, 5:277–291, 1996.

[DF92] H. Damasio and R. Frank. Three-dimensional in vivo mapping of brain lesions in humans. *Archives of Neurology*, 49:137–143, February 1992.

[DS93] A. M. Dale and M. I. Sereno. Improved localization of cortical activity by combining eeg and meg with mri cortical surface reconstruction: a linear approach. *Journal of Cognitive Neuroscience*, 5:162–176, 1993.

[EMN+92] A.C. Evans, S. Marrett, P. Neelin, T. Gum, W. Dai, S. Milot, E. Meyer, and D. Bub. Anatomical mapping of functional activation in stereotactic coordinate space. *Neuroimage*, 1:43–53, 1992.

[EPO91] A B Ekoule, F C Peyrin, and C L Odet. Triangulation algorithm from arbitrary shaped multiple planar contours. *ACM Transactions on Graphics*, 10(2):182–199, Apr 1991.

[FB88] David R Forsey and Richard H Bartels. Hierarchical b-spline refinement. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 22(4):205–212, Aug 1988.

[FKN80] H. Fuchs, Z.M. Kedem, and B.F. Naylor. On vision surface generation by a proiri tree structures. *Computer Graphics*, 14(3):124–133, 1980.

[FKU77] H. Fuchs, Z.M. Kedem, and S.P. Uselton. Optimal surface reconstruction from planar contours. *Communications of the ACM*, 20(10):693–702, October 1977.

[FvDFH90] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes. *Computer Graphics: Principles and Practice.* Addison-Wesley, 2 edition, 1990.

[GA93] Andre Gueziec and Nicholas Ayache. Large deformable splines, crest lines, and matching. *Proceedings of SPIE - The International Society for Optical Engineering*, 2031:316–327, 1993.

[GD82] S. Ganapathy and T.G. Dennehy. A new general triangulation method for planar contours. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 16(3):69–75, 1982.

[GF90] Elmer G Gilbert and Chek-Peng Foo. Computing the distance between general convex objects in three-dimensional space. *IEEE Transactions on Robotics & Automation*, 6(1):53–61, Feb 1990.

[GJK88] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi. A fast procedure for computing the distance between complex objects. *??*, 4(2), 1988.

[GJS96] Prosenjit Gupta, Ravi Janardan, and Michiel Smid. Fast algorithms for collision and proximity problems involving moving geometric objects. *Computational Geometry Theory and Applications*, 6:371–391, 1996.

[GKKJ92] Guido Gerig, Olaf Kübler, Ron Kikinis, and Ferenc A. Jolesz. Nonlinear anisotropic filtering of mri data. *IEEE Transactions on Medical Imaging*, 11(2):221–231, jun 1992.

[GKVPF89] M. S. Gazzaniga, M. Kutas, C. Van Petten, and R. Fendrich. Human callosal function: Mri-verified neuropsychological functions. *Neurology*, 39(7):942–6, Jul 1989.

[GS93] R.J-M. Grognard and A.D. Seagar. Modeling surfaces using spherical harmonics. Technical report, CSIRO, Sydney, Australia, 1993.

[HEM92] M.E. Hyche, N.F. Ezquerra, and R. Mullick. Spatiotemporal detection of arterial structure using active contours. In *Proceedings Visualization in Biomedical Computing 1992*, pages 52–62, 1992.

[HH92] K. H. Hohne and W. A. Hanson. Interactive 3d segmentation of mri and ct volumes using morphological operations. *Journal of Computer Assisted Tomography*, 16(2):285–94, Mar-Apr 1992.

[HNS88] Klaus Hinrichs, Jurg Nievergelt, and Peter Schorn. Plane-sweep solves the closest pair problem elegantly. *Information Processing Letters*, 26(5):255–261, Jan 1988.

[HW90] Mark Hall and Joe Warren. Adaptive polygonalization of implicitly defined surfaces. *IEEE Computer Graphics and Applications*, 10(6):33–42, Nov 1990.

[Jam92] B. James. The unfolded shape of the brain. *Journal of the Royal Society of Medicine*, 85(9):551–2, Sep 1992. Review.

[KCHN91] A.D. Kalvin, C.B. Cutting, B. Haddad, and M.E. Noz. Constructing topologically connected surfaces for the comprehensive analysis of 3d medical structures. In *Proceedings of SPIE – Medical Imaging V: Image Processing*, pages 247–258, 1991.

[KCS+92] M. Kamber, D. L. Collins, R. Shinghal, G. S. Franci s, and A. C. Evans. Model-based 3D segmentation of multiple sclerosis lesions in dual-echo

MRI data. In *Visualization in Biomedical Computing*, volume 1808, pages 590–600, Chapel Hill, North Carolina, 13-16 October 1992. SPIE.

[KEP96] R. K.-S. Kwan, Alan C. Evans, and G. Bruce Pike. An extensible MRI simulator for post-processing evaluation. In *Proceedings of the International Conference on Visualization in Biomedical Computing*, 1996. In press.

[KGC+89] Stane Kovacic, Jim C Gee, Wallace S L Ching, Martin Reivich, and Ruzena Bajcsy. Three-dimensional registration of pet and ct images. *Proceedings of the Annual Conference on Engineering in Medicine and Biology*, 11(89):548–549, 1989.

[KGV83] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[Knu73] D.E. Knuth. *The Art of Computer Programming 3: Sorting and Searching*. Addison-Wesley Publishing Company, 1973.

[KWT88] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, pages 321–331, 1988.

[LC87] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 21(4):163–169, Jul 1987.

[LHH+91] E. D. Lehmann, D. J. Hawkes, D. L. Hill, C. F. Bird, G. P. Robinson, A. C. Colchester, and M. N. Maisey. Computer-aided interpretation of spect images of the brain using an mri-derived 3d neuro-anatomical atlas. *Medical Informatics*, 16(2):151–66, Apr-Jun 1991.

[Lum85] V. J. Lumelsky. On fast computation of distance between line segments. *Inform. Process. Lett.*, 21:55–61, 1985.

[MAE93] JD MacDonald, D Avis, and AC Evans. Automatic parameterization of human cortical surface from magnetic resonance images. In *Information Processing in Medical Imaging*, 1993.

[MAE94] JD MacDonald, D Avis, and AC Evans. Multiple surface identification and matching in magnetic resonance images. In *Proceedings Visualization in Biomedical Computing 1994*, pages 160–169, 1994.

[MBA93] Gregoire Malandain, Gilles Bertrand, and Nicholas Ayache. Topological segmentation of discrete surfaces. *International Journal of Computer Vision*, 10(2):183–197, Apr 1993.

[MBL+91] J.V. Miller, D.E. Breen, W.E. Lorensen, R.M. O'Bara, and M.J. Wozny. Geometrically deformed models: a method for extracting closed geometric models from volume data. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 25(4):217–226, July 1991.

[MCAG92] M.I. Miller, G.E. Christensen, Y. Amit, and U. Grenander. A mathematical textbook of deformable neuro-anatomies. Technical report, Washington University School of Medicine, St. Louis, MO, August 1992.

[MDR91] Olivier Monga, Rachid Deriche, and Jean-Marie Rocchisani. 3d edge detection using recursive filtering: application to scanner images. *CVGIP-Image Understanding*, 53(1):76–87, Jan 1991.

[MH80] D. Marr and E. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London*, 207:187–217, 1980.

[MS96] J. Matoušek and O. Schwarzkopf. A deterministic algorithm for the three-dimensional diameter problem. *Comput. Geom. Theory Appl.*, 6:253–262, 1996.

[MSS92] D. Meyers, Shelley Skinner, and Kenneth Sloan. Surfaces from contours. *ACM Transactions on Graphics*, 11(3):228–258, July 1992.

[MSV95] Ravikanth Malladi, James A Sethian, and Baba C Vemuri. Shape modeling with front propagation: a level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175, Feb 1995.

[MT85] M. McKenna and G. T. Toussaint. Finding the minimum vertex distance between two disjoint convex polygons in linear time. *Comput. Math. Appl.*, 11:1227–1242, 1985.

[MT96] Tim McInerney and Demetri Terzopoulos. Deformable models in medical image analysis: a survey. *Medical Image Analysis*, 1(2), 1996.

[NA93a] C. Nastar and N. Ayache. Non-rigid motion analysis in medical images: a physically based approach. In *Proceedings of Information Processing in Medical Imaging*, pages 17–32, Flagstaff, Arizona, June 1993.

[NA93b] Chahab Nastar and Nicholas Ayache. Physically based analysis of deformations in 3d images. *Proceedings of SPIE - The International Society for Optical Engineering*, 2031:182–192, 1993.

[NAF94] T. Cizadilo S. Arndt D.S. O'Leary V. Swayze N.C. Andreasen, G. Harris and M. Flaum. Techniques for measuring sulcal, gyral patterns in the brain as visualized through magnetic resonance scanning: Brainplot and brainmap. *Proceedings of the National Academy of Science, USA*, 90:93–97, January 1994.

[Nef90] C. A. Neff. Finding the distance between two circles in three-dimensional space. *IBM J. Res. Develop.*, 34:770–775, 1990.

[Nil90] Nils J. Nilsson. *The Mathematical Foundations of Learning Machines.* Morgan Kaufmann Publishers, San Mateo, California, 1990.

[Nis95] Dwight G. Nishimura. Principles of magnetic resonance imaging. Class notes, Stanford University, 1995.

[OKA90] M. Ono, S. Kubik, and C.D. Abernathey. *Atlas of Cerebral Sulci*. Georg Thieme Verlag, Stuttgart, 1990.

[OPY92] Mattias Ohlsson, Carsten Peterson, and Alan L Yuille. Track finding with deformable templates - the elastic arms approach. *Computer Physics Communications*, 71(1):77–98, Aug 1992.

[OSTG89] J. S. Oppenheim, J. E. Skerry, M. J. Tramo, and M. S. Gazzaniga. Magnetic resonance imaging morphology of the corpus callosum in monozygotic twins. *Annals of Neurology*, 26(1):100–4, Jul 1989.

[PA95] Janos Pach and Pankaj K. Agarwal. *Combinatorial Geometry*. John Wiley and Sons, Inc., New York, 1995.

[PCS+89] C. A. Pelizzari, G. T. Y. Chen, D. R. Spelbring, R. R. Weichselbaum, and C. T. Chen. Accurate three-dimensional registration of CT, PET and MRI images of the brain. *Journal of Computer Assisted Tomography*, 13(1):20–26, 1989.

[Pen88] Alex Pentland. Shape information from shading: A theory about human perception. *Second Int Conf on Comput Vision*, pages 404–413, 1988.

[Pen89] A. Pentland. Shape information from shading: a theory about human perception. *Spatial Vision*, 4(2-3):165–82, 1989.

[Pen90] A. Pentland. Automatic extraction of deformable part models. *International Journal of Computer Vision*, pages 107–126, 1990.

[PFTV88] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, England, 1988.

[PH91] Alex Pentland and Bradley Horowitz. Recovery of nonrigid motion and structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):730–742, Jul 1991.

[PHS93] Alex Pentland, Bradley Horowitz, and Stan Sclaroff. Non-rigid motion and structure from contour. *Proc IEEE Workshop Visual Motion*, pages 288–293, 1993.

[PM90] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, Jul 1990.

[PS85] F. Preparata and M. Shamos. *Computational Geometry – An Introduction*. Springer–Verlag, New York, 1985.

[PS91] Alex Pentland and Stan Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):715–729, Jul 1991.

[PT90] Bradley A Payne and Arthur W Toga. Surface mapping brain function on 3d models. *IEEE Computer Graphics and Applications*, 10(5):33–41, Sep 1990.

[PT92] Bradley A Payne and Arthur W Toga. Distance field manipulation of surface models. *IEEE Computer Graphics and Applications*, 12(1):65–71, Jan 1992.

[PW89] A. Pentland and J. Williams. Good vibrations: modal dynamics for graphics and animation. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 23(4):215–222, 1989.

[RLW93] E. Rinast, R. Linder, and H. D. Weiss. Neural network approach for computer-assisted interpretation of ultrasound images of the gallbladder. *European Journal of Radiology*, 17(3):175–8, Nov 1993.

[RTL+93] H. Rusinek, W.H. Tsui, A.V. Levy, M.E. Noz, and M.J. deLeon. Principal axes and surface fitting methods for three-dimensional image registration. *Journal of Nuclear Medicine*, 34(11), November 1993.

[Sam] Hanan Samet. The quadtree and related hierarchical data structures. *Computing Surveys v 16, June 1984, p 187-260.*

[SB59] G. Schalenbrand and P. Bailey. *Introduction to Stereotaxis With an Atlas of the Human Brain.* G. Thieme, Stuttgart, 1959.

[Sch81] J. T. Schwartz. Finding the minimum distance between two convex polygons. *Inform. Process. Lett.*, 13:168–170, 1981.

[SD92a] Lawrence H Staib and James S Duncan. Boundary finding with parametrically deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(11):1061–1075, Nov 1992.

[SD92b] L.H. Staib and J.S. Duncan. Deformable fourier models for surface finding in 3d images. In *Proceedings Visualization in Biomedical Computing 1992*, pages 90–104, 1992.

[Ser82] Jean Paul Serra. *Image analysis and mathematical morphology.* Academic Press, London, 1982.

[SFF89] H. Steinmetz, G. Fürst, and H-J. Freund. Cerebral cortical localization: application and validation of the proportional grid system in mr imaging. *Journal of Computer Assisted Tomography*, 13(1):10–19, 1989.

[SG93] A.D. Seagar and R.J.-M. Grognard. Fourier descriptors of curves and surfaces. *Computer Vision, Graphics, and Image Processing*, 1993.

[Sie91] Raimund Siedel. *New Trends in Discrete and Computational Geometry.* Springer-Verlag, Berlin, 1991.

[SL] Lawrence H Staib and Xianzhang Lei. Intermodality 3d medical image registration with global search. *Proc IEEE Workshop Biomed Image 1994*, pages 225–234.

[SL97] S. Sandor and R. Leahy. Surface-based labeling of cortical anatomy using a deformable atlas. *IEEE Transactions on Medical Imaging*, 16(1):41–54, Feb 1997.

[SMG+93a] J W Snell, M B Merickel, J C Goble, J B Brookeman, and N F Kassell. Model-based segmentation of the brain from 3-d mri using active surfaces. *Bioengineering, Proceedings of the Northeast Conference*, pages 164–165, 1993.

[SMG+93b] John W Snell, Michael B Merickel, John C Goble, James B Brookeman, and Neal F M D Kassell. Model-based segmentation of the brain from 3d mri using active surfaces. *Proceedings of SPIE - The International Society for Optical Engineering*, 1898:210–219, 1993.

[SP93] Stan Sclaroff and Alexander P Pentland. Modal models: energy-based implicit functions. *Proceedings of SPIE - The International Society for Optical Engineering*, 1828:14–23, 1993.

[SRH+89] H. Steinmetz, J. Rademacher, Y. Huang, H. Hefter, K. Zilles, A. Thron, and H-J. Freund. Cerebral aysmmetry: Mr planimetry of the human planum temporale. *Journal of Computer Assisted Tomography*, 13(6):996–1005, 1989.

[SRJ+90] H. Steinmetz, J. Rademacher, L. Jancke, Y. X. Huang, A. Thron, and K. Zilles. Total surface of temporoparietal intrasylvian cortex: diverging left-right asymmetries. *Brain & Language*, 39(3):357–72, Oct 1990.

[SS91] H. Steinmetz and R. J. Seitz. Functional anatomy of language processing: neuroimaging and the problem of individual variability. *Neuropsychologia*, 29(12):1149–61, 1991. Review.

[ST92] Richard Szeliski and David Tonnesen. Surface modeling with oriented particle systems. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 26(2):185–194, Jul 1992.

[SW] Hanan Samet and Robert E Webber. Hierarchical data structures and algorithms for computer graphics. *IEEE Computer Graphics & Applications v 8, May 1988, p 48-68– Related material:v8 p59-75 Jl '88.*

[SZE97] John G. Sled, Alex P. Zijdenbos, and Alan C. Evans. A comparison of retrospective intensity non-uniformity correction methods for MRI. In *Information Processing in Medical Imaging*, 1997. in press.

[SZL92] W.J. Schroeder, J.A. Zarge, and W.E. Lorensen. Decimation of triangle meshes. *Computer Graphics: A Quarterly Report of SIGGRAPH-ACM*, 26(2):65–70, July 1992.

[TJP+93] T.G. Turkington, R.J. Jaszczak, C.A. Pelizzari, C.C. Harris, J.R. MacFall, J.M. Hoffman, and R.E. Coleman. Accuracy of registration of pet, spect and mr images of a brain phantom. *Journal of Nuclear Medicine*, 34(9):1587–1594, September 1993.

[TM86] P.H. Todd and R.J.Y. McLeod. Numerical estimation of the curvature of surfaces. *Computer-Aided Design*, 18(1):33–37, January 1986.

[TT88] J. Talairach and P. Tournoux. *Co-Planar Stereotactic Atlas of the Human Brain: 3-Dimensional Proportional System: An Approach to Cerebral Imaging.* Verlag, Stuttgart, 1988.

[TWK87] D. Terzopoulos, A. Witkin, and M. Kass. Symmetry-seeking models and 3d object reconstruction. *International Journal of Computer Vision*, 1(3):211–221, October 1987.

[TWK88] Demetri Terzopoulos, Andrew Witkin, and Michael Kass. Constraints on deformable models: Recovering 3d shape and nonrigid motion. *Artificial Intelligence*, 36(1):91–123, Aug 1988.

[UH92] J.K. Udupa and G.T. Herman. Boundaries in multidimensional digital scenes: Theory and algorithms. Technical report, Department of Radiology, University of Pennsylvania, January 1992.

[VRL93] B.C. Vemuri, A. Radisavljevic, and C.M. Leonard. Multi-resolution stochastic 3d shape models for image segmentation. In *Proceedings of Information Processing in Medical Imaging*, pages 62–76, Flagstaff, Arizona, June 1993.

[WIGKJ94] W.M. Wells III, W.E.L. Grimson, R. Kikinis, and F.A. Jolesz. Statistical intensity correction and segmentation of mri data. In *Proceedings Visualization in Biomedical Computing 1994*, pages 13–24, 1994.

[Wil64] Douglass J. Wilde. *Optimum Seeking Methods*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1964.

[WMW86] G. Wyvill, C. McPheeters, and B. Wyvill. Data structure for soft objects. *The Visual Computer*, 2:227–234, 1986.

[WP92] John R Williams and Alex P Pentland. Superquadrics and modal dynamics for discrete elements in interactive design. *Engineering Computations (Swansea, Wales)*, 9(2):115–127, Apr 1992.

[WS89] Estarose Wolfson and Eric L Schwartz. Computing minimal distances on polyhedral surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(9):1001–1005, Sep 1989.

[WWW87] P Widmayer, Y. F Wu, and C. K Wong. On some distance problems in fixed orientations. *SIAM Journal on Computing v 16, Aug*, pages 728–46., 1987.

[ZDM93] Alex P Zijdenbos, Benoit M Dawant, and Richard A Margolin. Measurement reliability and reproducibility in manual and semi-automatic mri segmentation. *Proceedings of the Annual Conference on Engineering in Medicine and Biology*, 15:162–163, 1993.

[ZER+96] Alex P. Zijdenbos, Alan C. Evans, Farhad Riahi, John G. Sled, H.-C. Chui, and V. Kollokian. Automatic quantification of multiple sclerosis lesion volume using stereotaxic space. In *Proceedings of the International Conference on Visualization in Biomedical Computing*, 1996. In press.