Abstract

Deformable Models in Image-Guided Neurosurgery

Oskar Škrinjar

2002

The core of the thesis is the idea of deformable-model-based information recovery from medical images, with the aim to reduce the impact of soft tissue deformation, noise, and image artifacts. Soft tissue deformation is the common denominator of many medical imaging problems. For this reason the main part of the thesis addresses the problem of soft tissue deformation recovery. Two volumetric deformable models based on soft tissue biomechanics are presented and used for deformation compensation. Experiments reported by other researchers as well as ones done by our group suggest that the complexity of soft tissue deformation renders deformable-model-based recovery very difficult. These findings lead to the concept of deformable model guidance. Rather than letting the model predict the soft tissue deformation based only on pre-deformation data, the approach we take is to guide the model by information available during the deformation. The models are guided by limited surface information with the goal to recover the deformation in the full volume. Another deformable-model-based information recovery is presented for the case of extraction of 2D structures embedded in 3D medical image volumes. The deformable model is based on the physical properties of the 2D structures, which significantly reduces the search space and enhances the quality of recovered information.

These methods are applied to image guided neurosurgery, where the top priority is the accuracy of surgical navigation systems. In particular,

we describe intraoperative brain deformation compensation, with a stereo system used for model guidance. In addition, we show how deformable-model-based information recovery can be used to help localize implanted electrodes from postoperative 3D image volumes. Both applications are a part of a larger project aimed at unifying anatomical, functional, and electro-physiological data into one coordinate system.

Deformable Models in Image-Guided Neurosurgery

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
in Candidacy for the Degree of
Doctor of Philosophy

by

Oskar Škrinjar

Dissertation Director: James Scott Duncan

May 2002

Acknowledgments

A number of people helped the work presented in this thesis. Here I would like to note with appreciation the thoughtful assistance and contributions of these people and unnamed others.

First, I would like to acknowledge the support, guidance, and help from my advisor James Duncan, who was a great research advisor, committee member, course instructor and director of the Image Processing and Analysis Group at Yale University. I would also like to thank Roman Kuc for serving on my committee, broadening my views, and for being an excellent course instructor; moreover, I am thankful for a great time I had while working as a teaching assistant for his courses. I am grateful to Hemant Tagare for numerous discussions and advice, as well as for excellent teaching of a differential geometry course and seminars on topology. Thanks also go to David Kriegman, whose courses I truly enjoyed, for serving on my committee and for useful discussions during my first two years at Yale. Peter Belhumeur was also a great help, not only as a committee member, but as a very good research adviser and course instructor, too. I am thankful to Larry Staib for serving on committee, as well as for useful discussions and help. I would like to thank Keith Paulsen from Dartmouth College for serving as my external reader. My research would be much more difficult, if not impossible, without the support and help from Dr. Dennis Spencer and Kevin McCarthy from Department of Neurosurgery, Yale University, who provided our lab with medical images and medical expertise through numerous discussions. I am also thankful to Anand Rangarajan for very useful discussions and advice. I would like to acknowledge the help from

Turan Onat, whose teaching and research discussions I enjoyed a great deal, and Gary Porvik, for useful research advice, both from Department of Mechanical Engineering, Yale University. A very useful collaborator was Todd Constable, who provided both images and image acquisition expertise. I am thankful to Xenios Papademetris not only for providing computer support, but also, as a senior graduate student and later a post-doc, for sharing with me his knowledge of programming and computer graphics. I would like to thank Ravi Bansal for numerous discussions and for contributing to the social life of our group. I am also thankful to James Rambo for discussions on computer graphics. Colin Studholme was an excellent collaborator, and I am thankful to him for all the help and for letting me use his code. I am also thankful to Steven Haker, Athos Georghiades, Ohad Ben-Shahar, and Jonas August for useful discussions. Many thanks to Arya Nabavi and Ron Kikinis from Surgical Planning Lab, Harvard Medical School, for collaboration and for providing our lab with intraoperative MR images. I would also like to thank all the current members and the alumni of the Image Processing and Analysis Group, for making this a pleasant place to work, James Beaty, Sudha Chelikani, Haili Chui, Jan Dik, Melissa Koudelka, Ning Lin, Gang Liu, Francois Meyer, Reshma Munbodh, Pengcheng Shi, Rik Stokking, Shawn Walker, Yongmei Wang, Lawrence Win, Jing Yang, Xiaolan Zeng, and George Zubal. Thanks also go to Carolyn Meloling for all her help. Finally, I would like to thank my wife Marija for being supportive, understanding, and loving from the very first day.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Structure of the Thesis

The goal of this thesis is the development of methods for deformable-model-based information recovery from medical images, with the aim to reduce the impact of soft tissue deformation, noise, and image artifacts. These methods are applied to the problem of intraoperative brain deformation recovery and to the problem of localization of implanted subdural electrodes from postoperative magnetic resonance (MR) images.

The thesis is divided into two parts. Part I (Chapters 2, 3, 4) presents the theoretical background for the developed methods, while Part II (Chapters 5, 6) discusses image guided neurosurgery applications.

The work related to this thesis is discussed in Section 1.3. In addition, relevant literature is referenced whenever a method or application is introduced.

Chapter 2 introduces two biomechanical models for soft tissue deformation analysis: a damped spring-mass model and a model based on contin-

uum mechanics. The two models are compared and their advantages and disadvantages discussed. The concept of deformable model guidance is presented in Chapter 3, which is motivated by the complexity of soft tissue deformation. While the goal of the work presented in Chapters 2 and 3 is soft tissue deformation recovery from medical images using biomechanics as prior knowledge, Chapter 4 deals with the problem of 2D structure shape recovery from medical images in presence of artifacts and noise. The core of the shape recovery method is a deformable model which is based on physical properties of the 2D structures.

The methods developed in Chapters 2 and 3 are applied to the problem of intraoperative brain deformation in Chapter 5. The shape recovery method of Chapter 4 is used in Chapter 6 for the problem of localization of implanted subdural electrodes from postoperative MR images.

Appendix A gives an overview of mathematical concepts used throughout the thesis, while Appendix B provides remarks on cubic spline interpolation.

1.2 Introduction to the Problem

Soft tissue deformation, image noise and artifacts are frequently encountered problems in medical image analysis. In order to reduce their impact, we propose to use deformable models as a means of incorporating prior knowledge in image analysis. In the case of soft tissue deformation, the use of biomechanics-based deformable models constrains possible displacement fields enhancing the quality of the recovered deformation. Similarly, shape recovery of structures embedded in medical images can benefit from the use of deformable models based on the physical properties of the structures. Such a strategy reduces the search space and provides a physically correct

solution.

In this work we develop deformable-model-based information recovery methods for two image guided surgery applications, with an aim to make the methods general and applicable to related problems. The first application is the intraoperative brain deformation compensation, which is a complex soft tissue deformation recovery problem. The second application is the localization of implanted subdural electrodes from postoperative MR images, a shape recovery problem affected by image artifacts and noise.

Surgical navigation systems provide the surgeon with a display of preoperative and intraoperative data in the same coordinate system. However the systems currently in use in neurosurgery are subject to inaccuracy caused by intraoperative brain deformation (brain shift), since they typically assume that the intracranial structures are rigid. Experiments show brain shift of up to one centimeter, making it the dominant error in the system. A consequence is that the displayed preoperatively acquired brain images differ from the intraoperative brain, causing surgical navigation to be less reliable. In order to reduce this error and make surgical navigation systems more reliable we propose a biomechanical-model-based approach for brain shift compensation. The model is guided by limited intraoperative exposed brain surface data, with the aim to recover the deformation in the full volume. In order to validate the method we have done experiments in the operating room (OR) at Yale New Haven Hospital and at our Image Processing and Analysis Group, as well as performed numerous simulation studies. In addition, we have used intraoperative MRI data provided by our collaborators from Harvard Medical School.

Subdural electrodes are often used in epilepsy surgery in order to map brain function and locate seizures. The patient carries implanted electrodes

for several days, and over this time the electrodes are monitored for seizures and stimulated to determine brain function and the results are recorded. To effectively use these results, one needs to relate electrode locations to brain structures of interest. Postoperative MR images are often used to find the locations of electrode grids, but this task is affected by image artifacts and noise. We have developed a deformable-model-based method for recovery of the location and shape of the implanted electrodes from postoperative MR images. The model is based on physical properties of the implanted electrode grids. Electrode grids, which are 2D structures, are never subject to forces strong enough to stretch or compress them. For this reason, the model is based on the idea of preserving intrinsic surface distances. This approach helps find the location and shape of the electrode grids by reducing the effect of image artifacts and noise. Final 3D displays of extracted electrode grids embedded in medical images allow neurosurgeons and neurologists to easily visualize electrodes and relate their locations to brain structures of interest.

1.3 Related Work

Problems in image guided surgery have been addressed by many authors. Researchers at MIT and Harvard Medical School worked on automatic registration strategies in image guided surgery ([28], [29]), and their colleagues from Neurological Institute at McGill University addressed the use of multimodality imaging as an aid to the planning and guidance of neurosurgical procedures ([57]). Problems in surgical navigation were discussed in works by Chabrerie ([11]) and Dorward ([17]). Gering ([24]) and Stokking ([62]) worked on the integrated visualization in image guided surgery, while Studholme ([66], [67], [65]) addressed the problem of image fusion using

an information theoretic framework. Hata ([33]), Hill ([34]), Kansy ([38]), Maurer ([48]), and Nabavi ([53]) are among the researchers who used an intraoperative magnetic resonance imaging (MRI) scanner to enhance the performance of existing surgical navigation systems.

A survey of constitutive relations for human brain tissue was presented by Pamidi ([56]), who discussed, among other things, Kelvin solid model, which is a basis of our damped spring mass brain model. Spring mass models have often been used due to their simplicity and speed. E.g. they were used by Lee ([42]) for modeling facial deformations for animation purposes. Another work on constitutive modeling of brain tissue was done by Miller ([52]).

Soft tissue deformation is a complex phenomenon and in most case it is very difficult, if not impossible, to predict the deformation without guiding the model by information available during the deformation. This was observed by Hill ([34]) in his study of intraoperative brain deformation images obtained using interventional MR imaging. This assumption is the basis of our approach of modeling intraoperative brain deformation.

While some researcher, e.g. Gering ([24]), Gobbi ([26]), and Kansy ([38]), used acquired intraoperative data directly for surgical navigation, a few groups tried to utilize intraoperative information to correspondingly update typically richer preoperative data. Audette ([5]) used a range system to reconstruct the exposed brain surface, with an aim to compute intrasurgical brain deformation. In our work, we use a similar approach (instead of a range system we use a stereo camera system) but enhance the method by using a biomechanical model. In order to reconstruct and track the deforming brain surface, we use a method that builds up on the method suggested by Akgul ([2]). While they suggested the use of a deformable dual mesh approach,

we put this method in touch with a biomechanical model, in order to solve both surface reconstruction and tracking and in-volume tissue deformation. Edwards ([18]) also used the idea of guiding the model. He suggested a three component model in order to model rigid, fluid and deformable solid parts of the head. The work presents a 2D model, and while it is extendable to 3D, the tissue deformation modeling is not based on physics. A group of researcher used continuum mechanics in order to model tissue deformation. In a very nice series of papers, Miga (e.g. [49], [50], [51]) used continuum mechanics and consolidation physics to represent deformation characteristics of the brain. He did in vivo experiments using porcine data to validate the model generated deformation prediction. In addition, he analyzed the impact of anatomical constraints on brain deformation and incorporated them in the model, which is an approach we have taken, too. We also used continuum mechanics to model brain tissue deformation, but we advanced the approach by introducing model guidance, which helps recover complicated deformations that cannot be predicted by non-guided biomechanical models.

Several groups measured and reported the magnitude of the brain deformation. Hill ([35]) estimated the median brain surface shift after the dura had been opened to range from .3 mm to 7.4 mm. Bucholz ([10]) reported the average brain shift for cases in which hematoma or tumors were removed to be 9.5 mm and 7.9 mm, respectively. Similar values for the brain shift were reported by Maurer ([47]), Reinges ([59]), and Roberts ([61]).

We also note related work on: modeling of brain deformation due to tumor growth by Kyriacou ([41]), biomechanical model based non-rigid registration of brain images suggested by Ferrant ([21], [22]) and Hagemann ([31]), finite element modeling of the head under impact conditions by

Claessens ([13]), and optical flow for measurement of brain deformation by Hata ([33]).

In the development of our stereo surface reconstruction algorithm we have used a few image similarity measures. A comparison of several image similarity measures was done by Holden ([36]), while Studholme ([67]) presented normalized mutual information as an overlap invariant entropy image similarity measure.

For the problem of localization of implanted subdural electrodes, Chabrerie ([11]) suggested a manual approach, while we introduced an automated method. There is much work on interactive surface manipulation, e.g. by Markosian ([45]) and Zorin ([84]), but we haven't encountered work on manipulation of surfaces subject to local isometry. This is another contribution we made to facilitate localization of implanted subdural electrodes.

1.4 Contributions of this Work

There are three major contributions of this work:

- The development of a biomechanics-based deformable model suitable for intraoperative brain deformation compensation. The assumptions the model is derived from are based on our experience from experiments done in the OR at Yale New Haven Hospital as well as on results reported in the literature.

- The development of a stereo-camera-based deformable model guidance strategy. The complexity of soft tissue deformation renders deformation recovery very difficult. For this reason we introduce the concept of model guidance to help the model recover the deformation. The

stereo-camera-based model guidance strategy provides an automated and fast surface data acquisition and model updating. Such an approach is very well suited for surgical applications since it is not invasive and is relatively inexpensive.

- The development of an intrinsic-surface-distance-preserving deformable model. The model is applied to the problem of recovery of location and shape of implanted subdural electrode grids from postoperative MR images, but can be used in other problems where 2D structures are subject to local isometry.

Part I

Theoretical Foundations

Chapter 2

Biomechanical Deformable Models

2.1 Introduction

Soft tissue deformation is the common denominator of many medical imaging problems. While there are several different scenarios where soft tissue deformation plays the central role, here we look at objects that non-rigidly deform over time (e.g. the brain deforms during the surgery). The goal is to recover the deformation, i.e. the displacement vector field as a function of both space and time, $\boldsymbol{u}(\boldsymbol{r},t) = [u_x(\boldsymbol{r},t)\ u_y(\boldsymbol{r},t)\ u_z(\boldsymbol{r},t)]^T$.

In many cases the tissue biomechanics as well as factors affecting the deformation are only partly known and usually very complex. For this reason, particularly in the applications where precision is the first priority, typically it is not possible to develop a model that would predict the deformation with acceptable accuracy. Rather, the model has to be guided by information available during the deformation (e.g. during brain surgeries one

can use intraoperative information to guide biomechanical brain models) in order for the method to be reliable.

In this chapter, we present two biomechanical models for soft tissue deformation recovery, while in Chapter 3 we explore ways to guide the models with limited data available during the deformation. In our initial efforts to recover soft tissue deformation we used a damped spring mass model (see Section 2.2 for details) for its simplicity, speed, and ability to model slow and small soft tissue deformation. We apply this approach to the problem of intraoperative brain deformation recovery (see Chapter 5). As we further explored the problem of deformation compensation, we moved to a continuum mechanics model (presented in Section 2.4) which is also able to recover small soft tissue deformation, and although computationally more expensive, it overcomes drawbacks associated with the former model.

2.2 A Damped Spring-Mass Model

The goal of this work is to develop a biomechanical model that models soft tissue deformation, incorporates effects of gravity, and can be guided by limited data available during the deformation. Since one of the aims is to perform deformation recovery in real-time, i.e. faster or equal to the real deformation, we decided to use a damped spring-mass model because of its simplicity, speed, and ability to model small soft tissue deformation. We apply this model to the problem of brain shift, which is a small deformation (usually less than 5%) relative to the brain size.

2.2.1 Soft Tissue Modeling

Here we concentrate on relatively small and slow soft tissue deformation. According to our findings and findings of other groups ([10], [35]) brain shift is a relatively small deformation and a slow process (it takes between 30 and 60 minutes for the brain to achieve a steady state). This fact facilitates our approach to soft tissue modeling. We employ a linear stress-strain relation, which is a good approximation for a small tissue deformation. The model consists of a set of discrete interconnected nodes each representing a small part of the tissue. Nodes have masses depending on the size of the volume they represent and on the local tissue density. Each connection is modeled as a parallel connection of a linear spring and dashpot, known as the Kelvin solid model ([56]). As for the nodes, the connection parameters can depend on their position in the tissue. The Kelvin solid model is a model for a viscoelastic material subject to slow and small deformations. It is also a rather simple approach, which is a desirable property since the model deformation should be computed in real time, i.e. faster or at least at the speed of the tissue deformation. The constitutive relation for the Kelvin solid model is

$$\sigma = q_0\epsilon + q_1\dot{\epsilon}, \tag{2.1}$$

where σ is stress and ϵ strain, while q_0 and q_1 are local parameters. The dotted variables represent the time derivatives, e.g. $\dot{\epsilon} = \frac{d}{dt}\epsilon$.

Equation (2.1) can be rewritten in the following way. If two nodes are at positions $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$, have velocities $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$, and are connected in the above fashion, then the force acting on the first node is

$$\boldsymbol{f_{inner}}(\boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{v_1}, \boldsymbol{v_2}) = [k_s(\|\boldsymbol{r_2} - \boldsymbol{r_1}\| - r_{12}) - k_d(\boldsymbol{v_2} - \boldsymbol{v_1}) \cdot \boldsymbol{n_{21}}]\,\boldsymbol{n_{21}}, \tag{2.2}$$

where k_s is the stiffness coefficient, k_d is the damping coefficient and r_{12} is the rest length of the spring connecting the two nodes. In a general case they can vary from connection to connection depending on the local material properties. Vector $\boldsymbol{n_{21}}$ is the unit vector from $\boldsymbol{r_1}$ to $\boldsymbol{r_2}$. Note that the same force acts on the other node but in the opposite direction.

2.2.2 Model Equations

Newton's Second Law for each node j in the model gives

$$m^j \boldsymbol{a}^j = m^j \boldsymbol{g} + \sum_{i=1}^{n^j} \boldsymbol{f_{inner}}^j_{s_i^j}, \tag{2.3}$$

where m^j is the node's mass, $\boldsymbol{a}^j$ is its acceleration, $\boldsymbol{f_{inner}}^j_{s_i^j}$ is the interaction between nodes j and s_i^j defined by (2.2), and $\boldsymbol{g}$ is the gravity acceleration, while $\{s_1^j, s_2^j, \ldots, s_{n^j}^j\}$ is the set of n^j neighboring nodes of the node j. Equation (2.3) represents a system of second order nonlinear ordinary differential equations.

One can define state variables to be $\boldsymbol{x}_{2j-1} = \boldsymbol{r}^j$ and $\boldsymbol{x}_{2j} = \boldsymbol{v}^j$ for $j = 1, \ldots, N$, where N is the number of the brain model nodes, $\boldsymbol{r}^j$ is the position vector of the j-th node, and $\boldsymbol{v}^j$ is its velocity. Obviously, $\dot{\boldsymbol{x}}_{2j-1} = \boldsymbol{x}_{2j}$. The expression for $\dot{\boldsymbol{x}}_{2j}$ can be obtained directly from (2.3), since $\dot{\boldsymbol{x}}_{2j} = \frac{d}{dt}\boldsymbol{x}_{2j} = \boldsymbol{a}^j$. The expression depends only on state variables but not on their time derivatives. It follows that (2.3) can be rewritten in a compact state-space form, $\dot{\boldsymbol{\mathcal{X}}} = \boldsymbol{f}(\boldsymbol{\mathcal{X}})$, where $\boldsymbol{\mathcal{X}}$ is the vector of the state variables and $\dot{\boldsymbol{\mathcal{X}}} = \frac{d}{dt}\boldsymbol{\mathcal{X}}$. It is assumed that the model starts deforming from a rest position, i.e. $\boldsymbol{v}^j(t=0) = \boldsymbol{0}$ for all j. The initial node positions, $\boldsymbol{r}^j(t=0)$, are set by the mesh generator (Section 5.2.2).

The system in the state-space form is suitable for a numerical integration

([58]). In this case, the fourth order Runge-Kutta method with adaptive step size was employed.

2.3 Comparison of Spring-Mass and Continuum Mechanics Models

The main advantages of spring-mass models are their simplicity and computational speed. E.g. they are used in [42] for modeling facial deformations for animation purposes, since they are much faster to compute then continuum mechanics models.

However, one of the drawbacks of spring-mass models is that their parameters (e.g. spring constant, i.e. stiffness coefficient) depend on the model mesh. This means that if one wants to change the mesh density, e.g. to use a finer (denser) mesh, he would need to change model parameters to achieve the same model behavior. The problem is that it is not clear how to change the model parameters (except for the case of 1D models, when it is straightforward). A consequence is that typically one can not find spring-mass model parameters in the literature, and even if they are reported, they can be used only with model meshes which have the same (or very similar) density as the mesh density of the model that was used with the reported parameters. For the same reason one cannot use non-uniform model (multiresolution) meshes with spring-mass models, since it is not clear how to set model parameters for regions with different mesh densities. On the other hand, parameters used with the continuum mechanics models (e.g. Young's modulus and Poisson's ratio) are independent of the model mesh, and therefore their values can be found in the literature and one can change the model mesh density or use non-uniform meshes.

In Chapter 3 we will discuss strategies for model guidance with limited data available during the deformation. It turns out that model guidance by surface data can be done in a physically and mathematically correct way in the case of continuum mechanics models (through the displacement boundary conditions of the model partial differential equations), while one needs to use ad-hoc strategies for model guidance in the case of spring-mass models.

In addition, one often wants to control the incompressibility of the deformable model. This is of a particular importance in soft tissue deformation, since soft tissues are mainly water, making them almost incompressible. Model incompressibility can directly be controlled by Poisson's ratio in the case of continuum mechanics models. In the case of spring-mass models it is not clear how to do it (e.g. how to achieve incompressibility that corresponds to a specific value of Poisson's ratio).

Finally, while continuum mechanics model are physically correct, this is not the case with spring-mass models, at least in the case of linear elasticity. In the limiting case, when the spring length is let to approach zero, it turns out that the obtained differential equation is of the first order, while the equation obtained starting from the continuum mechanics (Navier equation, see Section 2.4) is a second order differential equation.

Even though continuum mechanics models have several advantages over spring-mass models, the latter are often used due to their simplicity and particularly due to their computational speed.

2.4 A Continuum Mechanics Model

In spite of the simplicity and speed of spring-mass models, we have decided to move to computationally more expensive continuum mechanics models for the reasons explained in Section 2.3.

2.4.1 Soft Tissue Modeling

Our experience with the damped spring-mass model applied to slow deformation cases suggests that one can neglect dynamic components (components involving velocity and acceleration) in the model (brain shift is a relatively slow process). This simplifies the model, and eliminates the need for dynamic model parameters (e.g. damping coefficient). Whenever new data for guidance become available one can solve the model equations constraining (guiding) the equations by the data.

We base our approach on the following three assumptions:

- **Relatively simple model.** Due to the complexity of the soft tissue deformation, not only it is difficult to model some of the deformation causing factors, but also it is not clear how to set model parameters (any increase in the model complexity inevitably involves more parameters). Therefore we base our approach on a simple model that incorporates the main tissue characteristics (elasticity and almost incompressibility). The complexity of the deformation is made up by model guidance.

- **Static model.** Since our goal is to model relatively slow soft tissue deformation with negligible dynamic components, we use a static model.

- **Model guidance.** The model has to be guided by data available during the deformation.

Model guidance will be discussed in Chapter 3, while here we present the very model.

2.4.2 Model Equations

For small deformations (brain shift is a small deformation relative to the brain size) it is a good approximation to use a linear stress strain relation and infinitesimal strain. Although soft tissues typically are not isotropic, since the directions of anisotropy are usually not available[1], we assume isotropic materials. If parts of the deformable objects are known to be fixed[2], one can fix the corresponding parts of the model. Furthermore, since we consider relatively slow deformation cases with negligible dynamic components, we use a static model.

The linear stress - strain relation for isotropic materials is given by

$$\boldsymbol{\sigma} = \boldsymbol{C}\boldsymbol{\epsilon}, \tag{2.4}$$

where $\boldsymbol{\sigma} = [\sigma_x\ \sigma_y\ \sigma_z\ \tau_{xy}\ \tau_{yz}\ \tau_{zx}]^T$ is the stress vector, $\boldsymbol{\epsilon} = [\epsilon_x\ \epsilon_y\ \epsilon_z\ \gamma_{xy}\ \gamma_{yz}\ \gamma_{zx}]^T$ is the strain vector, and $\boldsymbol{C} = \frac{E}{(1+\nu)(1-2\nu)}\boldsymbol{G}$ is the material stiffness matrix,

[1] Diffusion tensor imaging might become a standard way to obtain directional information about soft tissues.

[2] E.g. due to the toughness of falx and tentorium, the movement of the two brain structures is negligible in most cases.

with

$$\boldsymbol{G} = \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}.$$

The material stiffness matrix depends on two parameters, Young's modulus (E) and Poisson's ratio (ν) ([72]). The displacement vector $\boldsymbol{u} = (u_x\ u_y\ u_z)$ is related to the strain vector through the following equation

$$\boldsymbol{\epsilon} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \boldsymbol{u}. \tag{2.5}$$

Since a static model is assumed, the equations relating stress components and body force are equilibrium equations,

$$\begin{aligned} &\tfrac{\partial \sigma_x}{\partial x} + \tfrac{\partial \tau_{yx}}{\partial y} + \tfrac{\partial \tau_{zx}}{\partial z} + F_x = 0, \quad \tau_{yx} = \tau_{xy}, \\ &\tfrac{\partial \tau_{xy}}{\partial x} + \tfrac{\partial \sigma_y}{\partial y} + \tfrac{\partial \tau_{zy}}{\partial z} + F_y = 0, \quad \tau_{zx} = \tau_{xz}, \\ &\tfrac{\partial \tau_{zx}}{\partial x} + \tfrac{\partial \tau_{yz}}{\partial y} + \tfrac{\partial \sigma_z}{\partial z} + F_z = 0, \quad \tau_{zy} = \tau_{yz}, \end{aligned} \tag{2.6}$$

where $\boldsymbol{F} = (F_x, F_y, F_z)$ is a body force (gravity in this case).

We are interested in obtaining the displacement field throughout the volume of the soft tissue, and therefore the goal is to obtain equations in displacements only. By using the systems of equations (2.4), (2.5), and (2.6),

and by eliminating stress and strain components, one can obtain

$$\begin{aligned}
\nabla^2 u_x + \tfrac{1}{1-2\nu}\tfrac{\partial}{\partial x}(\tfrac{\partial u_x}{\partial x} + \tfrac{\partial u_y}{\partial y} + \tfrac{\partial u_z}{\partial z}) + \tfrac{F_x}{\mu} = 0, \\
\nabla^2 u_y + \tfrac{1}{1-2\nu}\tfrac{\partial}{\partial y}(\tfrac{\partial u_x}{\partial x} + \tfrac{\partial u_y}{\partial y} + \tfrac{\partial u_z}{\partial z}) + \tfrac{F_y}{\mu} = 0, \\
\nabla^2 u_z + \tfrac{1}{1-2\nu}\tfrac{\partial}{\partial z}(\tfrac{\partial u_x}{\partial x} + \tfrac{\partial u_y}{\partial y} + \tfrac{\partial u_z}{\partial z}) + \tfrac{F_z}{\mu} = 0,
\end{aligned} \tag{2.7}$$

where $\mu = \frac{E}{2(1+\nu)}$. These three equations are elliptic PDEs in displacements only and are known as Navier equations ([72]).

We need to solve Eq. 2.7 with given displacement boundary conditions. Since they are linear PDEs, and since differentiation is a linear operator, one can separately find the solution $\boldsymbol{u'} = (u'_x, u'_y, u'_z)$ for the equations with zero boundary conditions, and the solution $\boldsymbol{u''} = (u''_x, u''_y, u''_z)$ for the equations with zero body force, and the total solution will be $\boldsymbol{u} = \boldsymbol{u'} + \boldsymbol{u''}$. However, gravity acts all the time, both before and during the deformation, and therefore $\boldsymbol{u'}$ will be the same in both cases. Since we are interested in the displacement field between the deformed and undeformed state, we do not need to compute $\boldsymbol{u'}$. Thus, we need to solve only for $\boldsymbol{u''}$, i.e. solve Eq. 2.7 with the given boundary conditions and zero body force. One should notice that gravity affects $\boldsymbol{u''}$ through boundary conditions (since the soft tissue deforms partly because of gravity, and a part of the soft tissue surface will be used as boundary conditions). Another interesting observation is that Young's modulus does not affect the displacement field ($\boldsymbol{u''}$), since the body force is zero in this case, and therefore the last terms in Eq. 2.7 containing E (hidden in μ) disappear. Thus, the only model parameter to be set is Poisson's ratio.

Chapter 3

Deformable Model Guidance

3.1 Introduction

Deformable models are used in many applications, e.g. in image processing, computer vision, computer graphics, biomedical engineering, mechanics of materials, to mention a few. The common characteristic of all the deformable models is that they have a finite set of parameters that describe their structure and behavior. Very often some of the model parameters are not known, or not exactly known. For example, mechanical properties of a non-homogeneous object vary in space, sometimes they vary in time, and can also vary across a class of the objects. For some types of applications the initial geometry of the model is not known (e.g. snakes). In the case of soft tissue deformation, usually the tissue biomechanics as well as factors affecting the deformation are only partly known and relatively complex.

For these reasons, particularly in applications where precision is the first priority, typically it is not possible to develop a forward model that is able to recover deformation (i.e. compute the displacement field) with acceptable

accuracy. Rather, the model has to be guided by data available during the deformation in order to increase precision and reliability.

There are different types of data available for model guidance: points, e.g. recorded by a point localizer ([76]), surface data: obtained by a range system ([5]) or by a stereo camera system ([82]), and volumetric data obtained by volumetric image acquisition systems (MRI, CT, and ultrasound). One can also take only single 2D images using volumetric image scanners, obtaining 2D (planar) data ([38]). The main disadvantage of using a point localizer is that the acquisition is time consuming (and disturbing in the case of surgery), while volumetric data acquisition, which main advantage is that it shows in-volume deformation, is often costly (e.g., intraoperative MR scanners can be afforded by few hospitals). For surface acquisition, range systems are usually more expensive than stereo systems, but they do not suffer from the correspondence problem, that is one of the main problems in stereo surface reconstruction. The only volumetric data acquisition system available to us was a 3D ultrasound probe. We have tried to use it a few times, but the images (of brain tissues) were not of sufficient quality to allow deformation analysis. For this reason we have initially used a point localizer, and have later moved to a stereo system due to faster and more automated acquisition.

Although any type of data can be used for guidance of any deformable model, as we developed and improved biomechanical models we were also able to advance model guidance strategies. This will be shown in the following sections of the chapter, where we present a few strategies for biomechanical model guidance and discuss their advantages and disadvantages.

3.2 Spring-Mass Model Guidance

In Section 5.3.1 we will show how to estimate spring-mass model parameters using limited data. Here we will assume that model parameters are known, and we will describe a way to readjust (guide) the model whenever a set of surface points on the deforming object (that we try to model) become available. The model tries to predict the object deformation at the moment of new data, new data are then used to readjust the model, and so on. The denser the data are both in space and time, the smaller the error between the model and the object.

In order to guide the model when (at time t) a new set of surface points $(x_i(t),\ i = 1, \ldots, N)$ is recorded, we do the following. For each point $x_i(t)$, we compute the displacement vector from the closest point p_i on the model surface to the point $x_i(t)$, and then artificially apply the displacement vector to the surface node closest to the point p_i. This is done for all the surface points at once. The imposed displacement constraints will propagate to other nodes through spring connections as the numerical integration proceeds. By doing this, one brings the model surface closer to the surface points. An example of a guided spring-mass model is given in Section 5.3.1.

The problem with this model guidance strategy is that it is completely ad-hoc, and it is not clear what the best way is to guide spring-mass models by surface point data. This is the second major reason (the first one was the fact that model parameters are mesh dependent) for moving to continuum mechanics-based models. In this case, the surface measurements can be used as displacement boundary conditions for the model partial differential equations. This is both mathematically and physically a correct way of guiding the model by surface data. Another reason for abandoning spring-

mass models is that they are not physically correct. They are just uniaxial models organized in a 3D (or 2D) matrix structure ("wire model"), with an empty space in between. For this reason spring-mass models do not directly model shear and incompressibility properties of tissues. These problems are avoided by using continuum mechanics.

3.3 Continuum Mechanics Model Guidance

As mentioned above, the idea behind the guidance of continuum mechanics models by surface data is to use the surface data as displacement boundary conditions for the model partial differential equations. A validation of such an approach for the case of a brain model using intraoperative MR imaging is presented in Section 5.3.2. Here we discuss surface data acquisition and integration with the deformable model in an automated fashion. For this purpose, we employ a stereo camera system to acquire surface data and suggest a way to use the data for model guidance. This work was reported in [79].

3.3.1 FEM Analysis

In order to solve system (2.7) we use a finite element method (FEM) based on the principle of virtual work. It has been shown ([83]) that the principle of virtual work is equivalent to solving system (2.6), which system (2.7) is derived from. According to the principle, in equilibrium, for arbitrary displacement $\boldsymbol{u}^*$, and the corresponding strain $\boldsymbol{\epsilon}^*$, the following equation holds,

$$\int_R \boldsymbol{\epsilon}^{*T} \boldsymbol{\sigma} dV = \int_R \boldsymbol{u}^{*T} \boldsymbol{F} dV, \tag{3.1}$$

where R is the region over which the displacement field should be determined. In a finite element framework, the region is partitioned into a set of finite elements. Within each element the displacement field is approximated using the following form,

$$\boldsymbol{u}(\boldsymbol{r}) = \sum_{1}^{N_e} n_i(\boldsymbol{r})\boldsymbol{u}_i, \tag{3.2}$$

where N_e is the number of nodes for the finite element, $\boldsymbol{u}_i$ is the displacement of node i, and $n_i(\boldsymbol{r})$ is the corresponding shape (interpolation) function. In order to guarantee that $\boldsymbol{u}(\boldsymbol{r_j}) = \boldsymbol{u}_j$ for all j and arbitrary nodal displacements, the shape functions have to satisfy $n_i(\boldsymbol{r}_j) = \delta_{ij}$, for all combinations of i and j, where δ_{ij} is Kronecker's delta. The goal of the FEM analysis is, assuming shape functions, to determine the nodal displacements $\boldsymbol{u}_i$, which in turn, according to (3.2), completely define the displacement field.

Assuming that $\boldsymbol{u}^*$ has the same form as $\boldsymbol{u}$ (see [44] for details), using (3.1) and taking into account that $\boldsymbol{u}_i{}^*$ are arbitrary, one can obtain a system of linear equations in nodal displacements $\boldsymbol{u}_i$,

$$\boldsymbol{A}\boldsymbol{w} = \boldsymbol{b}, \tag{3.3}$$

where $\boldsymbol{A}$ and $\boldsymbol{b}$ are a matrix and vector, respectively, whose elements are known (they are computed in the derivation of system (3.3)), and the elements of vector $\boldsymbol{w}$ are the x, y, and z components of the nodal displacements, i.e. $\boldsymbol{w} = [u_{1x}\ u_{1y}\ u_{1z}\ \cdots\ u_{Nx}\ u_{Ny}\ u_{Nz}]^T$, and $\boldsymbol{u}_i = [u_{ix}\ u_{iy}\ u_{iz}]^T$. The total number of nodes is N. Typically, $\boldsymbol{A}$ is a large sparse matrix. For given boundary conditions, i.e. the values of some of the nodal displacements, one can solve (3.3) for the rest of the nodal displacements, and then using (3.2) determine the displacement field.

The method is independent of the (bio)mechanical model used, as long as it reduces to the system of linear equations of the form (3.3). In other words, instead of starting from (2.4), (2.5), and (2.6), one can use some other model, convert the problem to a system of linear equations through an FEM analysis, and then proceed to the next step of the method.

3.3.2 Stereo Guidance

We have decided to use a pair of stereo cameras as a means to guide the deformable model. This strategy is a step forward compared to the surface point based model guidance, since the data acquisition is automated and it allows one to obtain more surface data in less time.

The basic idea is to reconstruct the object's exposed surface (in the brain surgery case, a part of the brain surface is exposed through the craniotomy) using stereo, and then use the reconstructed surface as a boundary condition for the model PDEs. While the stereo reconstruction and model deformation can be done completely independently of each other, we take advantage of treating them jointly in order to overcome or reduce the problems of stereo reconstruction: correspondence, surface specularities (during the surgery the brain surface is wet, which causes specularities), and camera differences. Since we use a static biomechanical model, each time stereo camera images are taken, the model is updated, i.e. the displacement field is computed.

The following definition introduces the notion of the set of admissible displacement fields.

Definition 3.1 *The set of admissible displacement fields $\mathcal{U}$ is the set of displacement fields defined by equation (3.2), with $\boldsymbol{w}$ satisfying equation (3.3).*

The set $\mathcal{U}$ is the set of all FEM (approximate) solutions to (2.7) for any boundary conditions. One should note that if there are fixed parts of the object (in the case of brain surgery, the falx and tentorium are fixed) then the corresponding nodes have zero displacements, and the equations in system (3.3) corresponding to those nodes should be omitted (see [44] for details). The system reduces to a system of linear equations of the same form as (3.3). If vector $\boldsymbol{b}$ in (3.3) is a zero vector[1], then $\mathcal{U}$ is a finite dimensional linear vector space[2] defined by the null-space of $\boldsymbol{A}$. When vector $\boldsymbol{b}$ is not a zero vector, then $\mathcal{U}$ is not a linear vector space, but the method can still be used.

The approach we have employed is, rather then starting from a pair of stereo images and solving for corresponding points, to search $\mathcal{U}$ for the displacement field that is the most compliant with the stereo images. The advantage is that the stereo correspondence problem is avoided (by testing different $\boldsymbol{u} \in \mathcal{U}$, one assumes correspondence), and the result is a displacement field that satisfies the biomechanical model equations. In addition, when searching for the "best" displacement field, if one starts from the displacement field from the previous frame and penalizes non-smooth exposed surfaces, then the generated displacement field is less affected by surface specularities, than it would be if the deformation and surface reconstruction were treated separately (since specularities make the stereo correspondence problem very difficult). However, the problem is that the search space is high dimensional. Its dimension is $3N$ (where N is the number of non-fixed nodes in the model), since each of N nodal displacements has three

[1] As explained in Section 2.4.2, we need to solve (2.7) with a zero body force, which causes $\boldsymbol{b}$ to be zero.

[2] Its vectors are displacement fields.

components. There are typically thousands of nodes in the model.

One can reduce the search space by expressing the displacements of nodes not on the exposed surface in terms of the displacements of the nodes on the exposed surface. This can be achieved by writing (3.3) as

$$\begin{bmatrix} \boldsymbol{A^R} & \boldsymbol{A^{ES}} \\ \boldsymbol{A^1} & \boldsymbol{A^2} \end{bmatrix} \begin{bmatrix} \boldsymbol{w^R} \\ \boldsymbol{w^{ES}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b^R} \\ \boldsymbol{b^{ES}} \end{bmatrix}, \tag{3.4}$$

where $\boldsymbol{w^R}$ contains the displacement components (x, y, and z) of the rest of the nodes (nodes not on the exposed surface), $\boldsymbol{w^{ES}}$ contains the displacement component of the nodes on the exposed surface, $\boldsymbol{A^R}$, $\boldsymbol{A^{ES}}$, $\boldsymbol{A^1}$, and $\boldsymbol{A^2}$ are the corresponding blocks of matrix $\boldsymbol{A}$, and $\boldsymbol{b^R}$ and $\boldsymbol{b^{ES}}$ are the corresponding parts of vector $\boldsymbol{b}$. It is assumed that the nodes are ordered such that $\boldsymbol{w} = [\boldsymbol{w^R}^T \ \boldsymbol{w^{ES}}^T]^T$. From (3.4) one can obtain that

$$\boldsymbol{w^R} = \boldsymbol{A^R}^{-1} \left(\boldsymbol{b^R} - \boldsymbol{A^{ES}} \boldsymbol{w^{ES}} \right). \tag{3.5}$$

The other equations in system (3.4) should be disregarded, since they correspond to the exposed surface nodes, which displacements will eventually be specified, and therefore their corresponding virtual displacements are zero (which implies that these equations should be disregarded, see [44]). Matrix $\boldsymbol{A^R}$ will be regular ($\boldsymbol{A^R}$ has to be regular in order for (3.5) to make sense) if there are enough nodes with specified boundary conditions (fixed nodes and exposed surface nodes). One should take advantage of the fact that $\boldsymbol{A^R}$ is usually a large sparse matrix, since this can significantly reduce the computational time.

System (3.5) allows one to compute the displacements of the rest of the nodes given the displacements of the exposed surface nodes. This leads to the notion of the reduced set of admissible displacement fields.

Definition 3.2 *The reduced set of admissible displacement fields $\mathcal{U}^r$ is the subset of $\mathcal{U}$ with arbitrary $\boldsymbol{w}^{ES}$, and $\boldsymbol{w}^{R}$ given by (3.5).*

Instead of searching $\mathcal{U}$ for the displacement field that is the most compliant with the stereo images, we search $\mathcal{U}^r$, which is a much smaller search space. If $\boldsymbol{b}$ is a zero vector, then $\mathcal{U}^r$ is a subspace of $\mathcal{U}$.

We use a perspective camera model, and a calibration object for camera calibration (see [71]). The function that projects a point $\boldsymbol{r}$ in the space to the image plane of camera i ($i = 1, 2$) is denoted as $\boldsymbol{P_i}(\boldsymbol{r})$, i.e. $\boldsymbol{P_i} : \boldsymbol{R}^3 \mapsto \boldsymbol{R}^2$, where $\boldsymbol{R}$ is the set of real numbers. Once the camera calibration is done, functions $\boldsymbol{P_i}$ are known. The image of camera i is a function I_i that maps points from the camera image plane to gray levels, i.e. $I_i : \boldsymbol{R}^2 \mapsto \boldsymbol{R}$. Finally, the function from points in the space to image gray levels of camera i, is the composition of functions $\boldsymbol{P_i}$ and I_i, i.e. the image gray level of camera i corresponding to a point $\boldsymbol{r}$ is $I_i(\boldsymbol{P_i}(\boldsymbol{r}))$.

Let $\boldsymbol{S^0}$ denote the exposed surface before deformation[3].

Definition 3.3 *The exposed surface corresponding to $\boldsymbol{u} \in \mathcal{U}^r$ is $\boldsymbol{S^u} = \left\{\boldsymbol{r} \mid \boldsymbol{r} = \boldsymbol{p} + \boldsymbol{u}(\boldsymbol{p}), \boldsymbol{p} \in \boldsymbol{S^0}\right\}$.*

Definition 3.4 *The backprojection of the image of camera i to surface $\boldsymbol{S}$ is function $B_i^{\boldsymbol{S}} : \boldsymbol{S} \mapsto \boldsymbol{R}$, defined by $B_i^{\boldsymbol{S}}(\boldsymbol{r}) = I_i(\boldsymbol{P_i}(\boldsymbol{r})), \forall \boldsymbol{r} \in \boldsymbol{S}$.*

The goal is to find $\boldsymbol{u} \in \mathcal{U}^r$ for which $B_1^{\boldsymbol{S^u}}$ and $B_2^{\boldsymbol{S^u}}$ correspond "the best". In order to compare (backprojected) images and define "the best" image

[3]In the case of brain surgery, immediately after the dura is opened, i.e. just before the brain starts deforming, the surgeon can outline the exposed brain surface using a localizer (which is a part of standard surgical navigation systems). Since the brain surface (before deformation) can be obtained by segmenting a preoperative MR scan, it follows that the exposed surface will be known before the brain starts deforming.

correspondence, we have tested four image similarity measures: mean square difference (MSD), mutual information (MI), normalized mutual information (NMI), and normalized cross correlation (NCC). For their definitions and more details see [36] and [67]. The problem with MSD is that it assumes an identity transformation between gray levels of the two images. This assumption is not valid in the case of non-Lambertian surfaces and in the presence of camera differences (e.g. different gains). NCC assumes an affine transformation between grey levels, while NMI and MI assume a general (algebraic) transformation. The main drawback of this approach is that the full backprojection images from the two cameras are compared. This leads to very strong local minima where the surface is typically positioned close to the true surface with smaller misplaced surface parts. In order to avoid this problem, we compare the backprojections from the two cameras locally, at neighborhoods[4] of corresponding points, rather than globally, as a whole. It cannot be done by using NMI (or MI) since it requires entire images, or their relatively large parts, to be compared. This is necessary for reliable estimation of probability density functions used in NMI (and MI). For this reason we use NCC as a local image similarity measure. Since it is used locally, we assume a local affine transformation between the gray levels of the two backprojections, which is an approximation of a general algebraic transformation at the global level.

Let $\partial \boldsymbol{S}(\boldsymbol{r})$ be a neighborhood on surface $\boldsymbol{S}$ of point $\boldsymbol{r} \in \boldsymbol{S}$. The normal-

[4]The neighborhood size is one of the parameters of the method. Since we discretize the model surface (with line segments in 2D and triangles in 3D), we use the two line segments sharing the node as its neighborhood in the case of 2D models, and the triangles sharing the node as its neighborhood in the case of 3D models.

ized cross correlation between $B_1^{\boldsymbol{S}}$ and $B_2^{\boldsymbol{S}}$ over $\partial \boldsymbol{S}(\boldsymbol{r})$ is

$$E^{\boldsymbol{S}}(\boldsymbol{r}) = \frac{\int \left(B_1^{\boldsymbol{S}} - \overline{B_1}\right)\left(B_2^{\boldsymbol{S}} - \overline{B_2}\right) dS}{\sqrt{\int \left(B_1^{\boldsymbol{S}} - \overline{B_1}\right)^2 dS}\sqrt{\int \left(B_2^{\boldsymbol{S}} - \overline{B_2}\right)^2 dS}},$$

where $\overline{B_i}$ $(i = 1, 2)$ is the mean backprojection intensity, i.e. $\overline{B_i} = \frac{\int B_i^{\boldsymbol{S}} dS}{\int dS}$. All the integrals are over $\partial \boldsymbol{S}(\boldsymbol{r})$. It can be shown that $\left|E^{\boldsymbol{S}}(\boldsymbol{r})\right| \leq 1$, and that $\left|E^{\boldsymbol{S}}(\boldsymbol{r})\right| = 1$ iff $B_1^{\boldsymbol{S}} = kB_2^{\boldsymbol{S}} + n$ over $\partial \boldsymbol{S}(\boldsymbol{r})$.

Since for the proper deformation recovery it is important to have point to point correspondence on the exposed surface over time, we define $E_i^{\boldsymbol{S}}(\boldsymbol{r})$ $(i = 1, 2)$ to be the normalized cross correlation between $B_i^{\boldsymbol{S}}$ over $\partial \boldsymbol{S}(\boldsymbol{r})$ and $B_i^{\boldsymbol{S^0}}$ over $\partial \boldsymbol{S^0}(\boldsymbol{p})$, where $\boldsymbol{r}$ and $\boldsymbol{p}$ are corresponding points. The objective function to be maximized is $E^{\boldsymbol{S}} = \int_{\boldsymbol{S}} E_{total}^{\boldsymbol{S}}(\boldsymbol{r}) dS$, where $E_{total}^{\boldsymbol{S}}(\boldsymbol{r}) = E^{\boldsymbol{S}}(\boldsymbol{r}) + E_1^{\boldsymbol{S}}(\boldsymbol{r}) + E_2^{\boldsymbol{S}}(\boldsymbol{r})$ is the "local energy". Term $E^{\boldsymbol{S}}(\boldsymbol{r})$ tries to find the best match between $B_1^{\boldsymbol{S}}$ and $B_2^{\boldsymbol{S}}$, while $E_1^{\boldsymbol{S}}(\boldsymbol{r})$ and $E_2^{\boldsymbol{S}}(\boldsymbol{r})$ try to enforce tracking, i.e. the best match between $B_i^{\boldsymbol{S}}$ and $B_i^{\boldsymbol{S^0}}$, or in other words, they try to find the best match between the backprojections in the current frame and their corresponding backprojections in the first (pre-deformation) frame.

One can improve results obtained by maximizing $E^{\boldsymbol{S}}$ by enforcing surface smoothness (this particularly makes sense in the case of brain surface, since it is very smooth). A good smoothness measure is strain energy ($\int_R \boldsymbol{\epsilon}^T \boldsymbol{\sigma} dV$), since a more curved surface implies higher strain energy and vice versa. The problem is that it is computationally too expensive. For each optimization iteration one would need to solve the whole FEM system in order to evaluate strain energy, which would render the approach too slow. Another approach, which is widely used, is to minimize an objective function of the form $\lambda E_s^{\boldsymbol{S}} - E^{\boldsymbol{S}}$, where $E_s^{\boldsymbol{S}}$ typically contains second or-

der derivatives of the surface. The problem is that there is no reliable way of setting λ, the parameter that controls the surface smoothness (higher λ means smoother surface), other than tuning. To avoid this problem we pose the optimization problem in the following way,

$$\hat{\boldsymbol{u}} = \arg \max_{\boldsymbol{u} \in \mathcal{U}^r} E^{\boldsymbol{S}^{\boldsymbol{u}}}, \max_{\boldsymbol{r} \in \boldsymbol{S}^{\boldsymbol{u}}} k(\boldsymbol{r}) < k_{max}, \tag{3.6}$$

where $\hat{\boldsymbol{u}}$ is the optimal displacement field, i.e. the solution,

$$k(\boldsymbol{r}) = \max\left(|k_1(\boldsymbol{r})|, |k_2(\boldsymbol{r})|\right),$$

$k_1(\boldsymbol{r})$ and $k_2(\boldsymbol{r})$ are the principal curvatures (see [16]) of $\boldsymbol{S}^{\boldsymbol{u}}$ at $\boldsymbol{r}$, and k_{max} is the maximal allowed curvature. In other words, we try to find a displacement field that maximizes $E^{\boldsymbol{S}^{\boldsymbol{u}}}$ subject to the constraint that the corresponding exposed surface does not have curvature greater than k_{max}. Parameter k_{max} controls the surface smoothness (smaller k_{max} implies a smoother surface) and can be estimated[5] (e.g. by measurements) since it has a physical meaning, as opposed to parameter λ.

This is a nonlinear optimization problem and we use a dual surface iterative scheme to solve it.

3.3.3 Optimization

Whenever a new frame (a new pair of stereo images) is acquired, we solve (3.6) for new displacement field. The initial idea was to start from the exposed surface from the previous frame, try to perturb it and find the one that will optimize (3.6). Experiments show that this approach in some cases suffers from the problem of local minima. In order to avoid this problem, we

[5]For brain surface we use $k_{max} = \frac{1}{r_{min}}$, $r_{min} = 20\ mm$.

have developed a dual surface optimization scheme, similar to the scheme presented in [2].

The idea is based on the fact that from a frame to the next frame the exposed surface will not move more than certain amount. If the maximal frame-to-frame displacement, d_{max}, is known, then one can define $\boldsymbol{S}_1^t = \{\boldsymbol{r} \mid \boldsymbol{r} = \boldsymbol{p} + d_{max}\boldsymbol{n}(\boldsymbol{p}), \boldsymbol{p} \in \boldsymbol{S}^p\}$, and $\boldsymbol{S}_1^b = \{\boldsymbol{r} \mid \boldsymbol{r} = \boldsymbol{p} - d_{max}\boldsymbol{n}(\boldsymbol{p}), \boldsymbol{p} \in \boldsymbol{S}^p\}$, where $\boldsymbol{S}_1^t$ is the "top" exposed surface, $\boldsymbol{S}_1^b$ is the "bottom" exposed surface, $\boldsymbol{S}^p$ is the exposed surface from the previous frame, and $\boldsymbol{n}(\boldsymbol{p})$ is the normal of $\boldsymbol{S}^p$ at point $\boldsymbol{p}$. The exposed surface for the new frame will be between $\boldsymbol{S}_1^t$ and $\boldsymbol{S}_1^b$. The scheme deforms the "top" and "bottom" exposed surface in a search for the optimum of (3.6), while pushing the two surfaces toward each other. Starting from $i = 1$, the scheme steps are:

1. $\boldsymbol{S}^g = \left\{\boldsymbol{r} \mid \boldsymbol{r} = \boldsymbol{p} + \delta\nabla E^{\boldsymbol{S}_i^g}(\boldsymbol{p}), \boldsymbol{p} \in \boldsymbol{S}_i^g\right\}, g = t, b$. This is gradient ascent with step size δ.

2. $\boldsymbol{S}^g \leftarrow \mathcal{K}(\boldsymbol{S}^g), g = t, b$. Operator $\mathcal{K}$ enforces the maximal curvature of $\boldsymbol{S}^g$ to be less than k_{max}. Since surfaces are triangulated, if the discrete approximation of the maximal curvature at a node is greater than k_{max}, that node is moved in the direction of the surface normal to the closest point where the maximal curvature is equal to k_{max}.

3. Move corresponding points of $\boldsymbol{S}^t$ and $\boldsymbol{S}^b$ toward each other (the one with smaller "local energy" will move more). This step guarantees that the scheme will converge (for details see "Convergence Enforcement" at the end of this section). $\boldsymbol{S}_{i+1}^g \leftarrow \boldsymbol{S}^g, g = t, b$.

4. if $d(\boldsymbol{S}_{i+1}^t, \boldsymbol{S}_{i+1}^b) > \epsilon$ then: $i \leftarrow i + 1$, go to step 1. $d(\boldsymbol{S}_{i+1}^t, \boldsymbol{S}_{i+1}^b)$ is the maximal distance between corresponding points on the two surfaces

(we use $\epsilon = .1\ mm$).

5. Use the average (over corresponding points) of $\boldsymbol{S}_{i+1}^t$ and $\boldsymbol{S}_{i+1}^b$ to determine $\boldsymbol{w}^{\boldsymbol{ES}}$. Use (3.5) to compute $\boldsymbol{w}^{\boldsymbol{R}}$, and (3.2) to determine the displacement field.

This dual surface scheme does not guarantee that all local minima will be avoided, but experiments show that it always outperforms the single surface scheme.

Convergence Enforcement

Let d_i denote the maximal distance between corresponding nodes of the "top" and "bottom" surface meshes for the i-th iteration, i.e. $d_i = d(\boldsymbol{S}_i^t, \boldsymbol{S}_i^b)$. The optimization scheme will converge if d_i will approach zero, i.e. if $\lim_{i\to\infty} d_i = 0$. Note that it is not enough that $\forall i\ d_{i+1} < d_i$, since, although it will guarantee convergence, the scheme might not converge to zero. It is also important that convergence is achieved with a reasonable speed. In practice, we assume that the scheme converges after i iterations if $d_i < \epsilon$, for a pre-specified value of ϵ.

In order to achieve convergence in a controlled way, we do the following. In step 3. of the optimization scheme, we move corresponding nodes from the two meshes toward each other such that $d_{i+1} = kd_i$, where k is a dimensionless parameter such that $0 < k < 1$. It is straightforward to show that $d_i = k^i d_0$, where d_0 is the maximal distance between corresponding nodes of the two meshes before the optimization. Since $0 < k < 1$, then $\lim_{i\to\infty} k^i = 0$, and consequently $\lim_{i\to\infty} d_i = 0$, i.e. the scheme converges. Moreover, it is an exponential convergence. If one wants to achieve a convergence in n iterations, that it is easy to show that one should take a value

for k that satisfies $k < \left(\frac{\epsilon}{d_0}\right)^{1/n}$.

The convergence enforcement does not guarantee that the curvature criterion will be preserved. However, the curvature criterion is enforced in each iteration of the optimization scheme (step 2) for both the top and bottom surface, and since the two surfaces converge toward each other, then the final surface will satisfy the curvature criterion.

3.3.4 Simulation Studies

In Chapter 5 we apply this approach to the problem of brain shift using real data, while here we test it by simulation studies.

For simulation studies, we artificially deformed a virtual exposed brain surface (this is the "true surface"), texture-mapped an image to the surface, positioned a pair of virtual cameras (camera calibration assumed), and projected the exposed surface to the cameras over time. Only the camera images were used as the input for the deformation recovery algorithm, while the "true surface" was used for validation. Model meshes were generated using an in-house meshing algorithm.

The purpose of 2D simulations is to visually demonstrate the method. Figures 3.1 and 3.2 are typical examples.

For 3D simulations we used realistic parameter values: exposed brain surface diameter = 7 cm (slightly smaller than craniotomy sizes typical for epilepsy surgery), cameras 1 m away from the brain (so as to not disturb the surgery), angle between cameras = 40°, camera resolution = 256×256 pixels, camera sensor size = 6 mm. Fig. 3.3 illustrates a deformation simulation and recovery of an initially bulging and then sinking brain. The mean error for surface reconstruction for this case over all surface nodes over

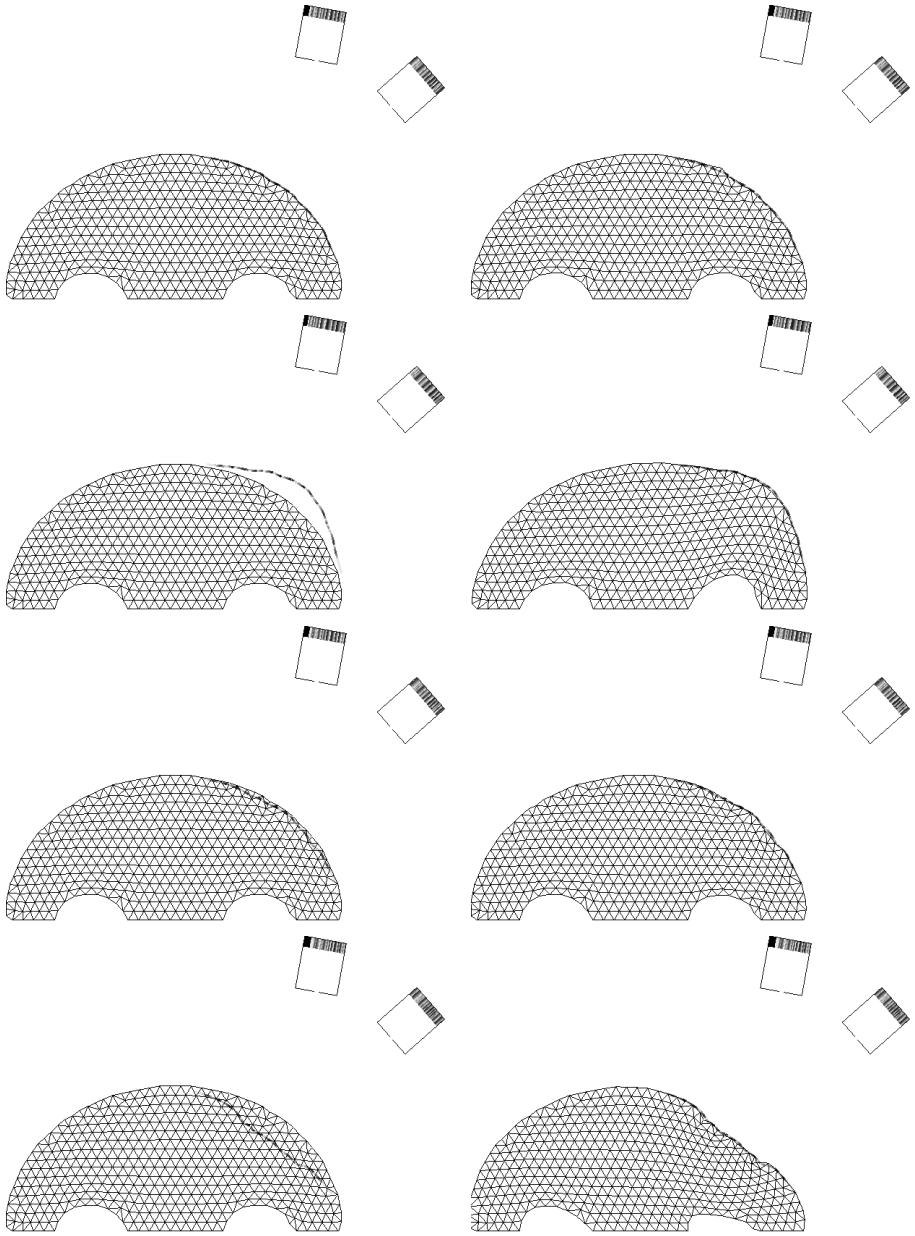


Figure 3.1: **An Example of 2D Deformation Recovery.** The left column is a time sequence of the true surface displayed with the undeformed model mesh, while the right column shows the true surface and the updated model mesh using the computed displacement field. A pair of virtual cameras is used for model guidance. The bottom row of model nodes was fixed ("falx").

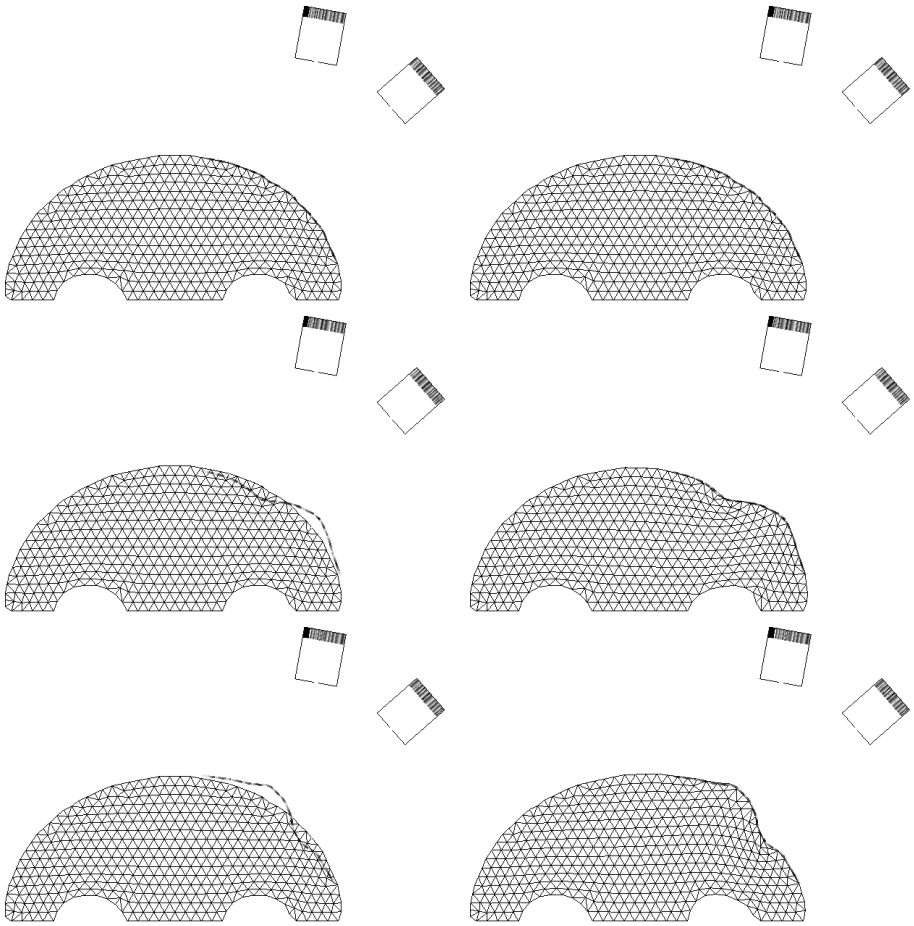


Figure 3.2: **Another Example of 2D Deformation Recovery.**

all frames was .17 mm (std = .09 mm, max = .88 mm), while the maximal displacement was 28 mm. We did over 50 deformation simulation studies artificially applying various deformation patterns to the exposed surface with magnitudes of up to 30 mm, which is more than the magnitude of actual brain deformations. The mean reconstruction error was always under .2 mm and max error under 1 mm. It takes about 2 min to compute one frame on a 933 MHz Pentium III (surface has about 300 nodes).

In order to make simulations more realistic, we added simulated specularities to the exposed brain surface images. Specularities might move over time, as it happens in brain surgery due to people walking around in the OR and lights being turned on and off or moved. We varied the position, shape, and size of specularities over frames and from study to study, but kept them similar in size and shape to actual specularities that appear on exposed brain surfaces. Some of the frames from one of the simulations are shown in Fig. 3.4. For this case, the mean reconstruction error was .20 mm (std = .15 mm, max = 1.33 mm). We did 10 more deformation simulations with added specularities similar to those shown in Fig. 3.4. The mean reconstruction error for all the cases was .2 mm or better, while the maximal error for the worst case was 1.62 mm.

We have added noise to simulated images to examine the dependence of surface reconstruction error on image noise standard deviation. The added noise was zero mean Gaussian noise with standard variation varied from zero to a value when the images cannot be visually distinguished from the noise. Fig. 3.5 shows how the image noise affects the surface reconstruction error in a typical case. Repeated simulations indicate that the method is robust to relatively small amounts of noise.

In addition, we have varied the angle between the two virtual cameras

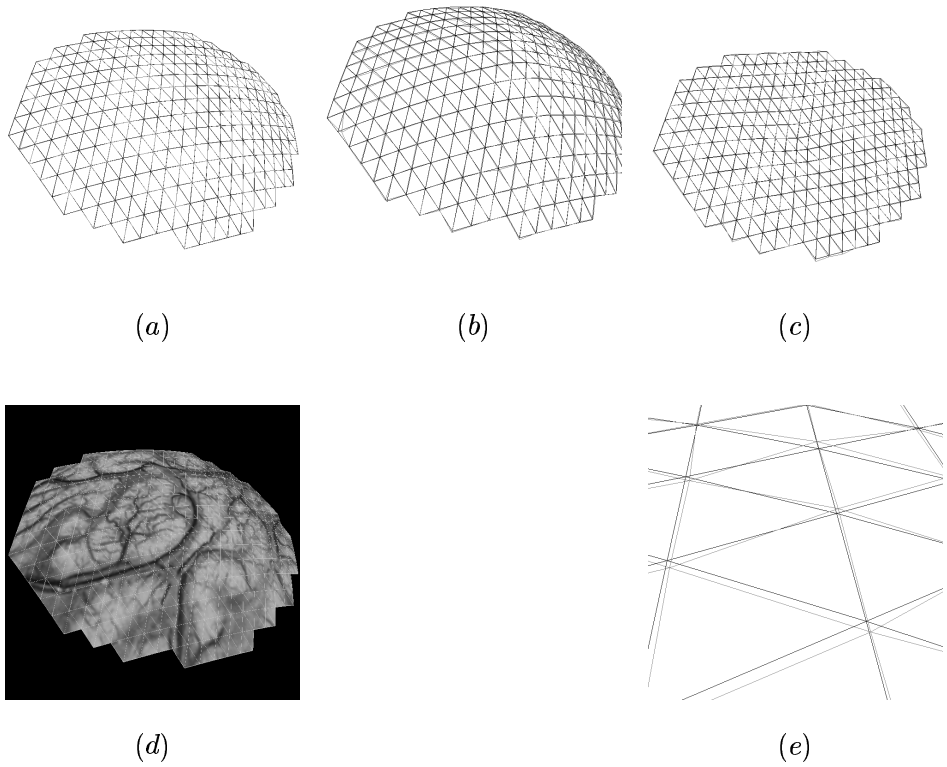
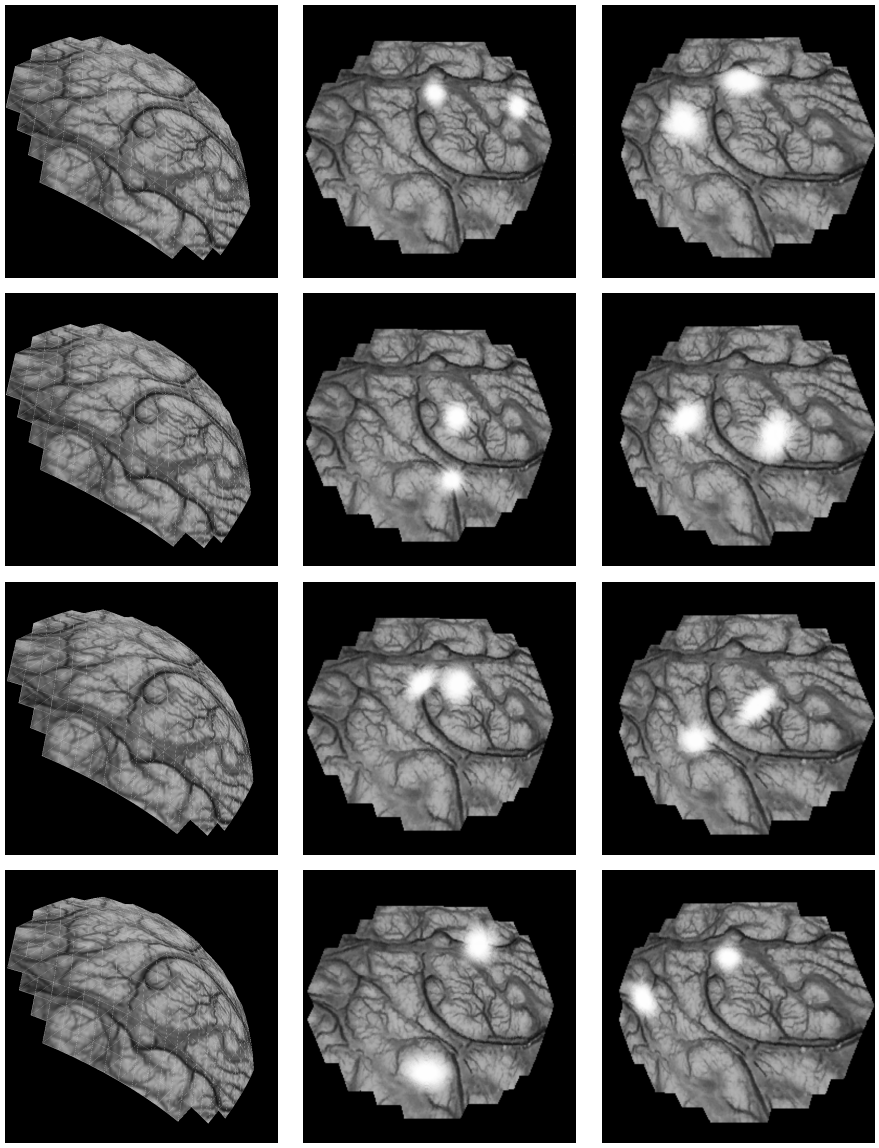


Figure 3.3: **Simulation of an Initially Bulging and then Sinking Brain.** (d) an exposed brain surface with a texturemap, (a) pre-deformation exposed brain surface mesh, (b) the exposed surface mesh at the peak of bulging (12 mm max displacement relative to the initial surface), (c) the exposed surface mesh at the peak of sinking (28 mm max displacement relative to the peak of bulging), and (e) a part of the zoomed-in true (solid) and recovered (dashed) exposed surface mesh at the peak of sinking (triangle side lengths ≈ 4 mm).



(a) (b) (c)

Figure 3.4: **Simulation of Specular Reflections on the Brain Surface.** (a) frames of a deforming virtual brain surface, (b) and (c) are the corresponding frames from the two cameras with randomly added specularities that move over time.

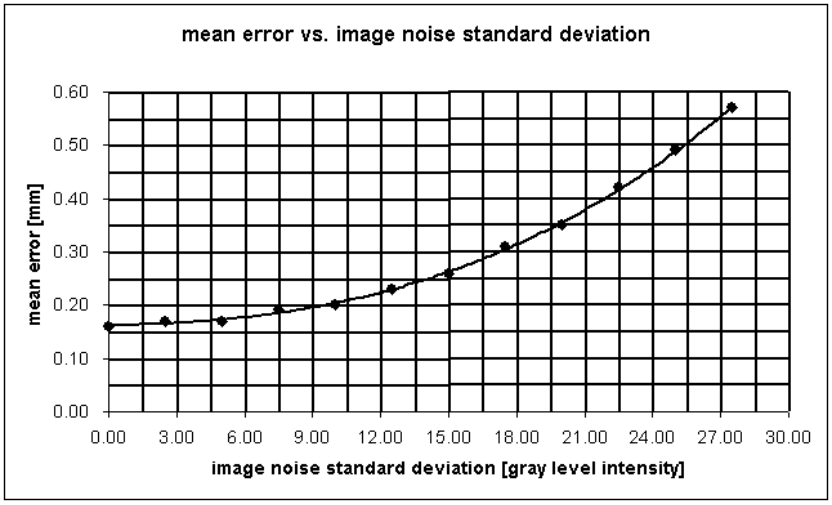


Figure 3.5: **Surface Reconstruction Error vs. Image Noise Standard Deviation.** The graph shows that the mean surface reconstruction error increases with the image noise standard deviation. The noise is additive zero mean Gaussian noise and image intensities range from 0 to 255. For relatively small values of image noise standard deviation (less than 10) the surface reconstruction error is almost constant, which indicates that a small amount of noise does not significantly affect the performance of the method.

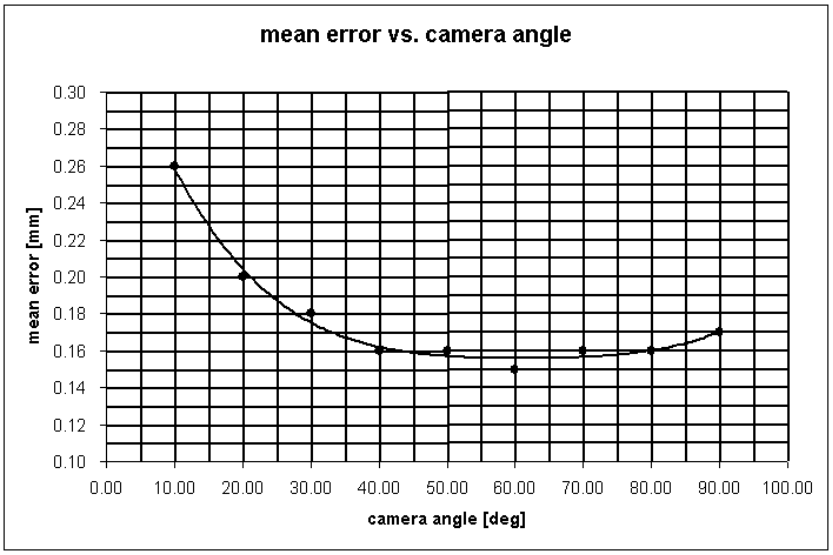


Figure 3.6: **Surface Reconstruction Error vs. Angle Between the Cameras.** The graph shows how the mean surface reconstruction error depends on the angle between the virtual cameras. The minimum error corresponds to an angle of 60 degrees. For higher angles the error increases because the images from the two cameras differ too much, while for smaller angles the error increases because the depth reconstruction error increases.

and examined how it affects the surface reconstruction error. A typical case is shown in Fig. 3.6. It indicates that there is an optimal angle between the cameras that minimizes the surface reconstruction error.

3.3.5 Surface Tracking Studies

In this section we present the result of applying the described algorithm to track deforming surfaces. In such cases (when there is no associated mechanical model) the same algorithm can be used, except that the step 5

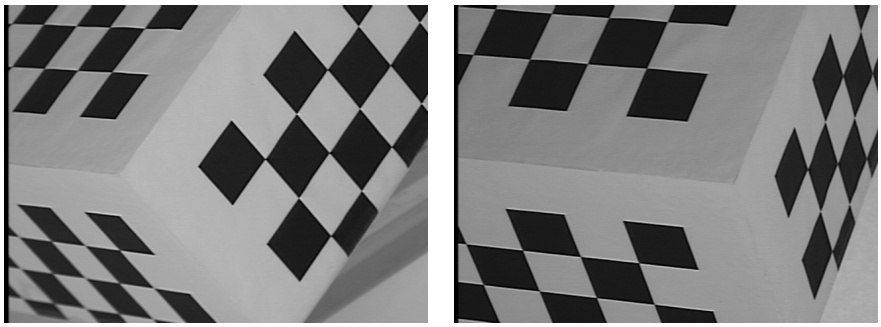


Figure 3.7: **Stereo System Calibration Object.** These two images are the views from the two cameras of the stereo system calibration object. The known geometry of the object is used to calibrate the system, i.e. to determine the camera model parameters.

of the optimization scheme (presented in Section 3.3.3) is omitted. For these experiments we used a pair of NTSC cameras, with a 640 by 480 resolution and a 48 dB signal to noise ratio (SNR). The baseline of the stereo system was 60 cm, while the distance from the scene (a deforming surface) to the cameras was about 110 cm.

We used the perspective camera model and the camera calibration procedure explained in [71]. Views from the two cameras of the object used for calibration are shown in Fig. 3.7.

After the calibration was done, we estimated the accuracy of the system. In order to draw any conclusions from the surface tracking algorithm results, one needs to know the accuracy of the stereo system (e.g., if the error of tracking a deforming surface is 2 mm, this does not mean anything if the accuracy of the stereo system is not known). For this task, we imaged a ruler from the two cameras (Fig. 3.8), and manually found corresponding points on the ruler marks in the two images. Then, we computed the as-

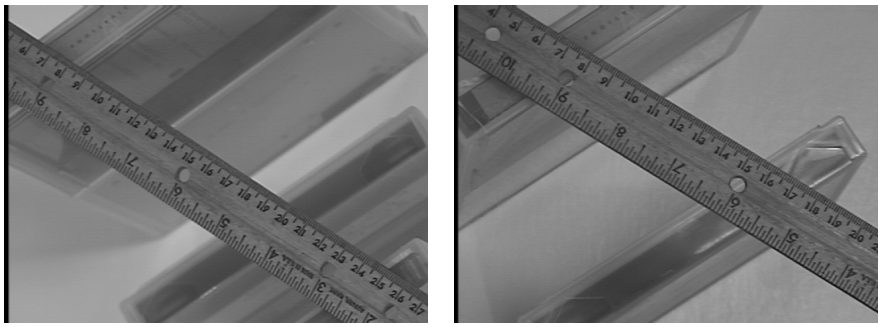


Figure 3.8: **Stereo System Accuracy Estimation.** A ruler was images from the two cameras. By manually associating corresponding points of ruler marks in the two camera images, we were able to estimate the accuracy of the stereo system.

sociated 3D point (using the camera model, which parameters were know after the calibration had been done) for each pair of corresponding points on the ruler marks in the two images. Finally, we computed the distance between the 3D point corresponding to the mark of say 11 cm, and the 3D point corresponding to the mark of say 17 cm. Since we know that the distance should be 60 mm (17 cm - 11 cm) we were able to compute the error between the stereo system generated distance between the two points and the true distance. We repeated the procedure for all pairs of marks and computed the mean and maximal error as well as the standard deviation. The same measurements were done for five different positions of the ruler. The results, summarized in Table 3.1, indicate that the stereo system accuracy in reconstructing distance between points in the space is well within a millimeter.

Once the calibration was done and the stereo system accuracy was estimated, we imaged a few deforming surfaces and tracked them using the

ruler position	mean [mm]	std [mm]	max [mm]	sample size
1	0.22	0.15	.62	78
2	0.20	0.16	.54	91
3	0.23	0.16	.66	91
4	0.19	0.15	.61	153
5	0.23	0.16	.69	105

Table 3.1: **Stereo System Accuracy.** The table contains the errors (mean, std, max) of stereo system reconstructed distances between pairs of ruler marks measured against the corresponding true distances. The sample size (the number of pairs of ruler marks) for the five cases is not the same because depending on the ruler position different number of ruler marks were visible in both cameras images (which is necessary to determine the distance between them).

proposed algorithm. In order to obtain the "true" surface deformation, we selected a set of points in one of the camera images in the first frame. The points were used as nodes of the surface triangulation. Then we manually set the corresponding points in the image of the other camera and the corresponding points in both camera images in all other frames. These point pairs were used to reconstruct the point locations in the space using the camera model, i.e. to obtain the "true" surface over time. Three frames of a deforming surface together with manually set corresponding points are shown in Fig. 3.9.

We applied the algorithm to three deforming surfaces and compared the algorithm computed surfaces to the "true" surfaces at manually set nodes. Then we computed the mean, standard deviation, and maximal error over all nodes over all frames for the three cases. The results are presented in Table 3.2. The mean error was about 1 mm while the maximal error was under 2 mm. Since the stereo system accuracy is a few times better than the accuracy of surface tracking, one can consider the figures in Table 3.2 to be reliable. One way to reduce the surface tracking errors is to use cameras with higher resolution and SNR.

These experiments, although done with surfaces that do not exhibit specular behavior, are an encouraging step toward building a stereo-camera-guided system for brain deformation compensation. Since the brain typically deforms for several millimeters, the stereo surface tracking should have an accuracy of at least a millimeter for the system to be useful.

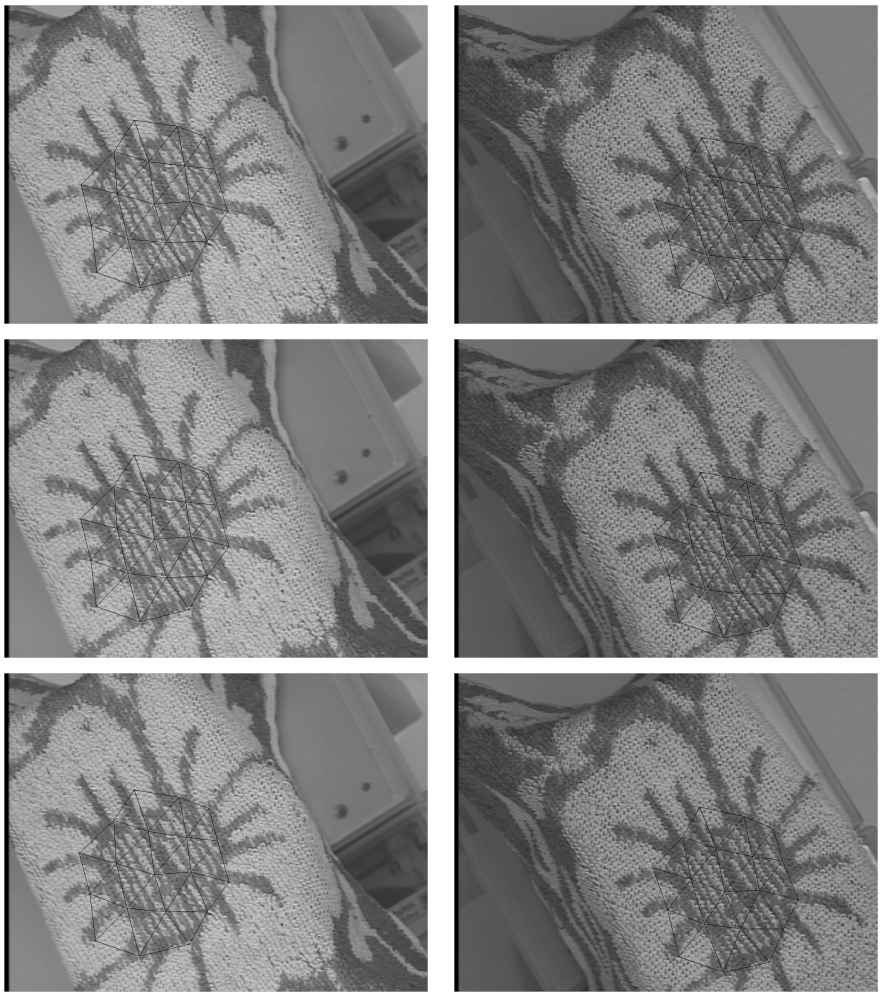


Figure 3.9: **Stereo Frames of a Deforming Surface.** Three stereo frames (each frame has two camera images) are shown together with manually set corresponding points organized in a triangulated structure. The change from a frame to a frame is barely visible since the surface was more than a meter away from the cameras and it deformed for only several millimeters.

surface	mean [mm]	std [mm]	max [mm]	displacement [mm]
1	1.11	0.46	1.87	4.6
2	1.04	0.41	1.77	3.6
3	1.02	0.49	1.71	6

Table 3.2: **Surface Tracking Errors.** The table contains the errors on node positions (mean, std, max) of three tracked surfaces measured against the "true" nodes which space positions were determined by manually selecting corresponding points in camera images and then using the camera model. The last column shows the maximal displacement of surface nodes.

Chapter 4

Isometrically Deforming Surface Model

4.1 Introduction

The goal of this section is to design a surface model that can be utilized in a recovery of 2D structures embedded in 3D images. In particular, we are interested in modeling deformable surfaces with a specific property - under a deformation the intrinsic surface distance between any two surface points does not change, i.e. the surface allows only locally isometric deformation (Definition A.50).

We start by discussing continuous solutions to the problem and then we present a discrete solution based on a damped spring surface model (net). The model is damped in order to prevent oscillations and it is iteratively solved until it reaches a steady state, i.e. until all the springs reach their rest lengths. By doing this one preserves distances along the surface (intrinsic surface distances). Nonlinear springs are added to approximately enforce

C_1 continuity of the surface. The method can be extended to surface deformations under which intrinsic distances change.

We apply the method to interactive manipulation of subdural electrode grids in post operative MRI datasets used in neurosurgery, since electrode grids are not in-plane extended or compressed while manipulated during the implantation (Section 6.3). The model is general and can be used in other applications as well, i.e. whenever there is a need to deform a surface in a locally isometric way.

First, we analyze properties of continuous solutions to the problem, give some theoretical results, then present a motivation for using a discrete solution based on a damped spring surface model, and finally explain the method and show results. This work was reported in [78].

There is much work on interactive surface manipulation (e.g. see [45] and [84]), but we haven't encountered work on manipulation of surfaces subject to local isometry.

4.2 Continuous Solution

Our initial idea was to find a surface parameterization that would have certain number of adjustable parameters, and for any choice of the parameters, it would preserve intrinsic surface distances. Thus, by changing the parameters, the surface would move and deform in a proper way (with preserved intrinsic distances at every point all the time). E.g. if the user selects a point on the surface and interactively moves it, the rest of the surface would move accordingly. The underlying model would assure that the intrinsic distances are preserved. Here we present two theorems on which we base our approach.

Theorem 4.1 [1] *A flat surface is isometricly deformed if and only if it can be parameterized with patches* $\mathbf{r}(u,v)$ *that satisfy*

$$\left\| \frac{\partial \mathbf{r}}{\partial u} \right\| = 1, \tag{4.1}$$

$$\left\| \frac{\partial \mathbf{r}}{\partial v} \right\| = 1, \tag{4.2}$$

and

$$\frac{\partial \mathbf{r}}{\partial u} \cdot \frac{\partial \mathbf{r}}{\partial v} = 0. \tag{4.3}$$

Proof. The distance along a curve on the surface is $\int_{\mathcal{C}} \left\| \frac{d\mathbf{r}}{dt} \right\| dt$, where $\mathcal{C}$ is the projection of the curve to the parametric space and t is the parameter of $\mathcal{C}$. Since $\frac{d\mathbf{r}}{dt} = \frac{\partial \mathbf{r}}{\partial u}\frac{du}{dt} + \frac{\partial \mathbf{r}}{\partial v}\frac{dv}{dt}$, it follows that the distance is $\int_{\mathcal{C}} \sqrt{\left\| \frac{\partial \mathbf{r}}{\partial u} \right\|^2 \left(\frac{du}{dt}\right)^2 + 2\left(\frac{\partial \mathbf{r}}{\partial u} \cdot \frac{\partial \mathbf{r}}{\partial v}\right)\left(\frac{du}{dt}\frac{dv}{dt}\right) + \left\| \frac{\partial \mathbf{r}}{\partial v} \right\|^2 \left(\frac{dv}{dt}\right)^2} dt$.

$\Longrightarrow$ If the equations (4.1), (4.2) and (4.3) are satisfied then the distance becomes $\int_{\mathcal{C}} \sqrt{\left(\frac{du}{dt}\right)^2 + \left(\frac{dv}{dt}\right)^2} dt$. Since it does not depend on $\mathbf{r}$, i.e. no matter now the surface is deformed, the distance over the surface is preserved for any curve $\mathcal{C}$, it follows that the surface is isometricly deformed.

$\Longleftarrow$ Let the surface be isometricly deformed, i.e. the distance over the deformed surface is the same as the corresponding one over the flat surface for any curve $\mathcal{C}$. Since this must hold for any curve, then for $u(t) = a + t$, $v(t) = b$ and $t_1 < t < t_2$, with a and b being constants, the distance becomes $\int_{t_1}^{t_2} \left\| \frac{\partial \mathbf{r}}{\partial u} \right\| dt$. But for the flat surface the distance is $t_2 - t_1$ (or it is

[1] This theorem, although independently derived, is not a new theoretical contribution, but rather a version of Lemma 4.5 (Section 6.4 from [55]) applied to flat surfaces.

proportional to $t_2 - t_1$, and the curve can always be reparametrized so that the distance is $t_2 - t_1$), and since the distance hasn't changed, it follows that $\int_{t_1}^{t_2} \left\| \frac{\partial \mathbf{r}}{\partial u} \right\| dt = t_2 - t_1$. By taking the derivative with respect to t_2 of the left and right side of the last equation one obtains (4.1). Similarly, by using $u(t) = a$, $v(t) = b + t$ and $t_1 < t < t_2$ one can obtain (4.2), and by using $u(t) = a + t$, $v(t) = b + t$ and $t_1 < t < t_2$ one can obtain (4.3). The equations (4.1), (4.2) and (4.3) have to be satisfied for all values of u and v since the previous analysis holds for all a, b, t_1 and t_2. $\diamond$

The equations in Theorem 4.1 are first order, nonlinear partial differential equations. They can be stated as

$$\left\| \frac{\partial \mathbf{r}}{\partial p} \right\| = A, \quad \left\| \frac{\partial \mathbf{r}}{\partial q} \right\| = B, \quad and \quad \frac{\partial \mathbf{r}}{\partial p} \cdot \frac{\partial \mathbf{r}}{\partial q} = C, \tag{4.4}$$

where $A > 0$, $B > 0$, and $\|C\| < AB$. With the following change of parameters

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} A & B\cos\phi \\ 0 & B\sin\phi \end{bmatrix} \cdot \begin{bmatrix} p \\ q \end{bmatrix},$$

where $\cos\phi = \frac{C}{AB}$, (4.4) can be simplified to the equations in Theorem 4.1, i.e. the equations (4.4) are not more general than the equations in Theorem 4.1.

The problem is to find solutions $\mathbf{r}(u, v)$ to the equations of Theorem 4.1 that would have certain number of parameters. By changing the parameters one would move and deform the surface, but the equations would be satisfied for any values of the parameters. One would like to have enough independent parameters, i.e. degrees of freedom, or in other words, enough flexibility to deform the surface. A few questions arise: Does a solution exist at all

?, Is there a general solution ?, Can one find a closed form solution, in a convenient (maybe polynomial) form ?

A solution does exist for the equations and it is a plane, i.e.

$$\mathbf{r}(u,v) = \mathbf{r_0} + u \cdot \mathbf{A} + v \cdot \mathbf{B},$$

where $\mathbf{r_0}$ is a constant vector, $\mathbf{A}$ and $\mathbf{B}$ are linearly independent vectors with $\|\mathbf{A}\| = A$, $\|\mathbf{B}\| = B$ and $\mathbf{A} \cdot \mathbf{B} = C$. This is a solution of Equations 4.4, and it can be reparametrized to satisfy equations of Theorem 4.1. Another known solution is a cylinder. A solution that is a plane can only be deformed by a rigid body transformation, while a cylinder, in addition to rigid body transformation, can have its radius changed. However, planes and cylinders are not useful solutions for this application since they cannot represent a general locally isometric deformation of the plane.

Theorem 4.2 *A rigid body transformation of a solution to the equations of Theorem 4.1 is again their solution.*

This theorem is an obvious result, but we give a proof here for completeness.

Proof. For this proof it is assumed that all the vectors are column vectors. Let $\mathbf{r}$ be a solution to the equations of Theorem 4.1 and let $\overline{\mathbf{r}}$ be its rigid body transformation, i.e. $\overline{\mathbf{r}} = \mathbf{R} \cdot \mathbf{r} + \mathbf{t}$, where $\mathbf{R}$ is a rotation matrix and $\mathbf{t}$ is a translation vector. Since $\frac{\partial \overline{\mathbf{r}}}{\partial u} = \mathbf{R} \cdot \frac{\partial \mathbf{r}}{\partial u}$ and $\frac{\partial \overline{\mathbf{r}}}{\partial v} = \mathbf{R} \cdot \frac{\partial \mathbf{r}}{\partial v}$ it follows that $\left\| \frac{\partial \overline{\mathbf{r}}}{\partial u} \right\| = \sqrt{\frac{\partial \overline{\mathbf{r}}}{\partial u}^T \cdot \frac{\partial \overline{\mathbf{r}}}{\partial u}} = \sqrt{\frac{\partial \mathbf{r}}{\partial u}^T \cdot \mathbf{R}^T \cdot \mathbf{R} \cdot \frac{\partial \mathbf{r}}{\partial u}} = \sqrt{\frac{\partial \mathbf{r}}{\partial u}^T \cdot \frac{\partial \mathbf{r}}{\partial u}} = \left\| \frac{\partial \mathbf{r}}{\partial u} \right\| = 1$, because $\mathbf{R}^T \cdot \mathbf{R} = \mathbf{I}$ (^{T} stands for transposition). Similarly, one can obtain that $\left\| \frac{\partial \overline{\mathbf{r}}}{\partial v} \right\| = \left\| \frac{\partial \mathbf{r}}{\partial v} \right\| = 1$. Finally, $\frac{\partial \overline{\mathbf{r}}}{\partial u}^T \cdot \frac{\partial \overline{\mathbf{r}}}{\partial v} = \frac{\partial \mathbf{r}}{\partial u}^T \cdot \mathbf{R}^T \cdot \mathbf{R} \cdot \frac{\partial \mathbf{r}}{\partial v} = \frac{\partial \mathbf{r}}{\partial u}^T \cdot \frac{\partial \mathbf{r}}{\partial u} = 0$. Thus, $\overline{\mathbf{r}}$ is a solution of the equations of Theorem 4.1. $\diamond$

Theorem 4.1 provides a framework for designing a surface model (either a continuous or a discrete one), while Theorem 4.2 allows one to rigidly move the whole model assuring that it is still a valid solution, i.e. that it satisfies equations of Theorem 4.1.

We haven't found any solution to the equations of Theorem 4.1 other than planes and cylinders, and we do not know if there is a general solution. For this reason we have employed an approximate, discrete solution to the problem using a damped spring model (net).

4.3 Discrete Solution

The motivation for using a spring net is the idea to relax the net so that the springs reach their rest lengths. If the spring net densely covers the surface, and the spring rest lengths are set so that initially all the springs are at rest, then the net will, whenever fully relaxed, have proper distances, i.e. distances along the springs will be preserved. If the net is dense, the distances over the whole surface will be preserved with a small deviance. In order for this approach to be valid, the connectivity of the net has to such that the distances are preserved in all directions (this is approximately achieved by introducing diagonal springs, see Fig. 4.1). One needs to introduce damping to the spring model, since otherwise the net would oscillate forever under any change. This approach will not provide a continuous solution, but rather a discrete one that can be interpolated.

Thus, we model the surface with a set of nodes interconnected by springs. For each node on the surface one can write the Second Newton's Law,

$$m\frac{d^2\mathbf{r}}{dt^2} + d \cdot \frac{d\mathbf{r}}{dt} + k \cdot \sum_i (\|\mathbf{r} - \mathbf{r_i}\| - l_i) \cdot \frac{\mathbf{r} - \mathbf{r_i}}{\|\mathbf{r} - \mathbf{r_i}\|} = 0,$$

where m is the node's mass, $\mathbf{r}$ position, d is the damping coefficient, k is the spring constant (stiffness coefficient), the sum is the sum over all neighboring nodes of the node, $\mathbf{r_i}$ is the position of the i-th neighbor, and l_i is the rest length of the spring between the node and its i-th neighbor. The first term is the inertial force term, the second one is the damping force, and the third is the sum of the spring forces acting on the node.

One can, assuming that node masses are very small, drop the inertial force term, and use quasi-static analysis. The governing equation then reduces to

$$\frac{d\mathbf{r}}{dt} + K \cdot \sum_i (\|\mathbf{r} - \mathbf{r_i}\| - l_i) \cdot \frac{\mathbf{r} - \mathbf{r_i}}{\|\mathbf{r} - \mathbf{r_i}\|} = 0,$$

where $K = \frac{k}{d}$. In order to be numerically solved, this equation has to be discretized, where time becomes the iteration index. The equation can be written as

$$\Delta\mathbf{r} = C \cdot \sum_i (\|\mathbf{r} - \mathbf{r_i}\| - l_i) \cdot \frac{\mathbf{r_i} - \mathbf{r}}{\|\mathbf{r} - \mathbf{r_i}\|}, \tag{4.5}$$

where $C = K \cdot \Delta t$. This reduction from a second order to a first order equation speeds up the computation while still achieving the goal. One can write equation (4.5) for each node in the net, obtaining a system of equations.

The method works as follows. When a node is moved (interactively by the user), compute $\Delta\mathbf{r}$ for each node, update their positions ($\mathbf{r} \mapsto \mathbf{r} + \Delta\mathbf{r}$), recompute new $\Delta\mathbf{r}$ for each node, update all positions, an so on, until the net is relaxed, i.e. until the system converges. Note that this is Euler's numerical integration scheme ([58]). The criterion we use to stop the iterative process is

$$\max_i \frac{|\|\mathbf{l_i}\| - l_i|}{l_i} < \epsilon,$$

where $\|\mathbf{l_i}\|$ is the (current) length of the i-th spring, and l_i is its rest length. We use $\epsilon = .01$, which means that no spring is extended or compressed more than 1 % of its rest length, or in the other words, the distances along the springs are preserved with maximal error of 1 %. We use this model for subdural electrode grid model manipulation (Chapter 6). The largest grid has ten electrodes in one direction, and since the inter-electrode distance is 1 cm, the total grid length is 9 cm. This means that the length of the grid model will not differ from 9 cm by more than .9 mm (1 % of 9 cm), i.e. the worst case error will be less than the resolution (1 mm) of the MR images that are used.

The next step is to set a value for the constant C in the equation (4.5). One can exactly analyze a single damped spring. In the quasi-static approach, the governing differential equation is

$$\frac{dx}{dt} + C \cdot (x - x_0) = 0,$$

where x_0 is the rest spring length. The general solution is

$$x(t) = Ae^{-Ct} + x_0.$$

The constant A depends on the initial condition, i.e. on the initial position of the spring. Now it is easy to see that the greater C, the sooner the spring will reach its rest length. It also follows from (4.5) that for a small C, the convergence is slow. However, a too large C makes the system unstable, i.e. the iterative process does not converge. A nice property is that the convergence does not depend on the physical size of the surface,

i.e. if the process converges for a value of C, then the rescaled surface (each node position $\mathbf{r_i}$ is rescaled to $\alpha \cdot \mathbf{r_i}$) will also converge for the same value of C. This can be easily seen from (4.5) since the scale α cancels. Experiments with different values of C and different number of nodes in the net show that in all cases the system becomes unstable for values of C greater than .3. Since the above analysis suggests that C should be as great as possible, we use $C = .25$.

We use a net of nodes organized in a rectangular matrix, inter-connected with springs. A little thought reveals that it is necessary to have diagonal springs in addition to side ones. This is demonstrated in Fig. 4.1. Now, under any deformation, when the net is relaxed, the squares will very closely remain squares, i.e. the intrinsic distances will be preserved very closely. This organization of the springs is motivated by Theorem 4.1. The springs along the sides of the squares tend to enforce (4.1) and (4.2), while the diagonal springs are introduced to preserve angles, i.e. to enforce (4.3). However, it is clear that both side and diagonal springs at the same time tend to preserve distances and angles.

Another problem is that even with diagonal springs, when the net is relaxed, sharp corners might occur. This is illustrated in Fig. 4.2. In order to overcome this problem we introduce a nonlinear spring. It is a regular spring that can be turned on and off. Normally it is not active, and it is activated if the angle between two neighboring links becomes too small. In this case it pushes back the nodes keeping the angle large enough. The geometry of the problem is explained in Fig. 4.3. Using the law of cosines one obtains

$$d = a\sqrt{2(1 - \cos t)}.$$

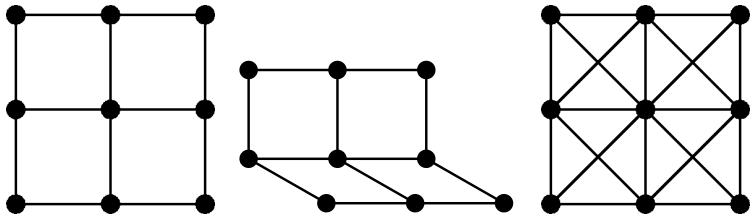


Figure 4.1: **Surface Model Spring Net Structure.** The black dots are nodes while the lines between the nodes are springs. If the net is organized as in (a), then it could reach (b) and be fully relaxed, i.e. all the springs have their rest lengths (the net (b) is still in one plane). This is clearly not allowed, since the diagonal distances between the nodes are not preserved. To prevent this from happening, we add diagonal springs, as in (c).

The curvature is the change of the angle of the normal per unit distance, i.e. in this case the curvature is $k = \frac{t}{a}$. Since $\cos(s) = -\cos(t)$ it follows that

$$d = 2a \cos \frac{ka}{2}. \tag{4.6}$$

By specifying the maximal curvature, using (4.6) one can compute at what distance d the spring will turn on. The spring will prevent nodes from coming closer than what is allowed, approximately enforcing C_1 continuity. The nonlinear springs make the net smooth with the given maximal curvature.

Thus, the user can now manipulate the surface, and the underlying model will deform the surface properly, assuring that the intrinsic distances are preserved with the given error (1% in our case). The user can rigidly move the surface to a new position if needed, and Theorem 4.2 guarantees that the

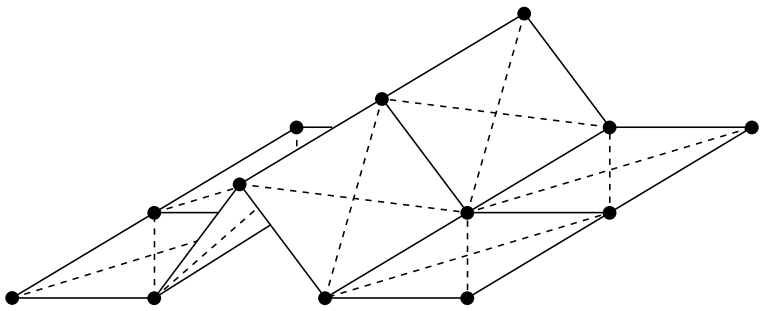


Figure 4.2: **An Example of a Spring Net Sharp Corner.** This figure shows a relaxed net (all springs, including the diagonal ones, shown as dashed lines, have reached their rest lengths) that has a sharp corner. The middle squares can be arbitrary close, making the corner arbitrary sharp. To prevent this we introduce nonlinear springs that approximately enforce C_1 continuity.

surface at the new position will also have the intrinsic distances preserved (which is intuitively clear).

4.4 Results

Figure 4.4 shows a few stages of the deformation of an electrode grid driven by the user interaction. The tool for interactive electrode grid manipulation (Section 6.3) we have developed based on the presented model allows the user to fix some of the electrodes, and when a non-fixed electrode is moved by the user only the free (non-fixed) electrodes will move. The user can move non-fixed electrodes without restrictions as long as the intrinsic surface distances are preserved. E.g., if a non-fixed electrode is neighboring to a fixed electrode, and if the user tries to move the non-fixed electrode too far

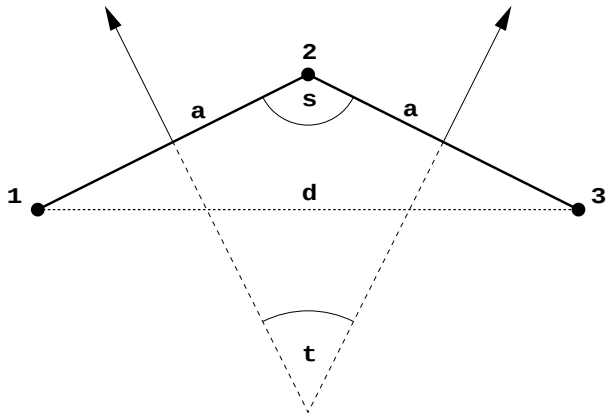


Figure 4.3: **A Nonlinear Spring in the Surface Model.** Three nodes (1, 2 and 3), that initially make a straight line, are connected with two springs, each of length a. If the angle s between the two springs becomes too small, a spring between nodes 1 and 3 of length d is activated and pushes the nodes back. The angle t is the angle change in the normal and can be related to the curvature.

surface\path	1	2	3	4	5	6	7
1	.035	.075	.007	.061	.509	.022	.051
2	.045	.057	.023	.016	.038	.054	.039
3	.110	.060	.075	.658	.049	.039	.086

Table 4.1: **Intrinsic Surface Distance Preservation Error.** Errors [percent] of seven computer generated random surface paths of random length for each of the three extensively deformed surfaces relative to the corresponding paths in the undeformed (flat) surfaces.

from the fixed electrode, the model will not allow that.

Table 4.1 shows the relative errors of various random paths over several extensively deformed surfaces (the errors are relative to the corresponding paths in the flat, undeformed surfaces). One can see that most of the errors are less than 0.1%, and this is because the iterative procedure is stopped when the most extended or compressed spring is less than 1% different (longer or shorter) from its rest length, but most of the springs are far less than 1% different from their rest lengths. These data indicate that the model indeed preserves the intrinsic surface distances under deformation with a specified relative error.

4.5 Discussion

This work shows that the presented deformable model in the form of a damped spring net can be used to model a surface with an interesting geometric property - preserved intrinsic surface distances. The method is an easy and intuitive way for the user to manipulate surfaces (e.g. electrode

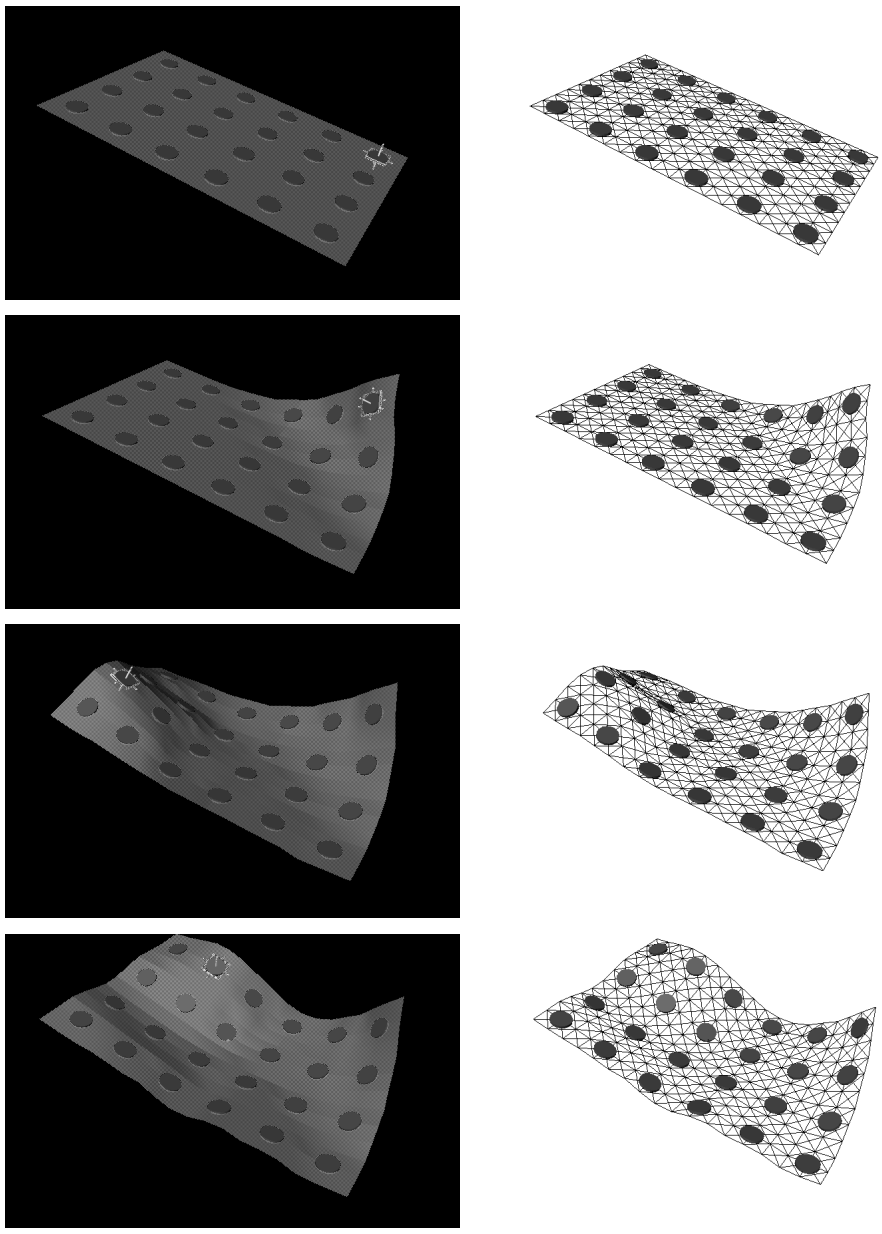


Figure 4.4: **A Sequence of States of a Deformed Surface Model.** The left and right figures show a sequence of an electrode grid deformed states and the corresponding model, respectively. The model is composed of nodes interconnected with springs (the lines in the right figures) and it guarantees that the intrinsic surface distances are preserved. In the left images one can see the selected electrode that is manipulated by the user.

grid surfaces). It is simple enough so that it runs with an interactive speed, while still achieving the specified goal.

A flat surface has a zero Gaussian curvature at all points. Since the Gaussian curvature is preserved under isometry (*Theorema Egregium* of Gauss, see [16] or [55]), it stays zero all the time at all surface points. One can use a damped spring net to manipulate a surface that initially does not have zero or constant Gaussian curvature. The net, no matter how manipulated, will preserve the initial Gaussian curvature at each point of the surface with desired accuracy. This is the case because the damped spring net allows for only locally isometric surface deformation. In addition, the damped spring net can be used for surfaces of shapes other than rectangular (e.g. sometimes the surgeon cuts out a part of the electrode grid, making it irregular, Section 6.3).

Part II

Application to Surgical Navigation Systems

Chapter 5

Intraoperative Brain Deformation Compensation

5.1 Introduction

Surgical navigation systems are systems used before the surgery for surgical planning and during the surgery for surgical navigation. Their goal is to help the surgeon prepare for and perform the surgery. These systems are capable of displaying various imaging modalities in one coordinate system, and merging pre- and intraoperative data. They use magnetic resonance images (MRI) for displaying anatomy, computerized tomography (CT) images for bony structures, magnetic resonance angiography (MRA) images for blood vessels, functional magnetic resonance images (fMRI) for function, and positron emission tomography (PET) and single-photon emission computerized tomography (SPECT) images for metabolic activity of the tissues that are imaged.

The use of surgical navigation systems has become a standard way to

assist the surgeon in navigating within the intraoperative environment, planning, and guiding the surgery. One of the most important features of these systems is the ability to relate the position of surgical instruments to features in preoperative images. Ideally, they should provide a 3D display of the anatomical structures of interest and include visualization of surgical instruments within the same coordinate system. In order to be reliably used, surgical navigation systems should be as accurate as possible, preferably to within the voxel size of the images used ([29]). Most of the current systems use preoperatively-acquired 3D data and register it to the patient coordinate system ([28], [29], [57]). However, they assume that the organs being operated on are rigid, and are consequently subject to error due to soft tissue deformation.

Here we concentrate on the problem of compensation for intraoperative brain deformation (commonly referred to as brain shift), although a similar approach can be applied to related problems involving soft tissue deformation. Preoperative data are registered to the patient coordinate system at the beginning of the surgery. While this can be done with a precision to within 1 mm at the beginning of the surgery ([29]), since the brain deforms, the accuracy of the system deteriorates as the surgery proceeds. The median brain shift after the dura had been opened of points on the brain surface was estimated to range from 0.3 mm to 7.4 mm ([35]). Since the deeper brain structures deform less than the outer ones, the largest error is at the cortical surface. It is clear that a surgical navigation system based on the rigid brain assumption cannot achieve a precision better than a few millimeters in the outer brain structures. The brain deforms even more after interventions (e.g. post-resection). Furthermore, the average brain shift for cases in which hematoma or tumors were removed was reported to be 9.5 mm and

7.9 *mm*, respectively ([10]). In such cases the error is even larger. In this work with do not address tissue resection and removal problems.

Brain shift contributes to the inaccuracy of surgical navigation systems more than any other source of error. Since the accuracy of surgical navigation systems is the top priority in making them useful, this problem has been addressed by several groups. The initial work was on estimating and reporting brain shift ([10], [17], [33], [35], [47], [59], [61]), while later efforts were aimed at compensating for the brain shift using a deformable model ([18], [49], [50], [51], [76], [80], [81]) and intraoperative brain imaging ([26], [33], [34], [48], [53]). We also note related work on: modeling of brain deformation due to tumor growth ([41]), biomechanical model based non-rigid registration of brain images ([21], [22], [31]), finite element modeling of the head under impact conditions ([13]), and brain tissue constitutive modeling ([52]). For a more detailed discussion on the work relevant to the brain shift problem see Section 1.3.

Brain shift is a complex phenomenon, and here we list factors that, not necessarily in the order of importance, affect brain deformation: gravity, mechanical tissue properties, administered drugs, loss of cerebro-spinal fluid (CSF), interaction of CSF and brain tissues, anatomical constraints, tissue resection and removal, intracranial pressure, geometrical complexity, and patient variability. The bulk of the brain deformation is typically in the gravity direction. Brain is not a homogeneous structure, i.e. different brain tissues have different mechanical properties, which affects the way the tissues deform. Some of the drugs administered during the surgery affect the brain volume and consequently influence the deformation. When the dura is cut, a part of the CSF leaks out causing ventricles to partly collapse, which is reflected in further brain tissue deformation. Certain parts of the brain

and surrounding structures are rigid or very stiff (skull, brain stem, tentorium, falx) imposing constraints on the soft tissue deformation. When the dura is cut, the intracranial pressure the brain was exposed to is relieved, which typically causes the brain to initially bulge out before sinking in the direction of gravity. Some of these factors are patient dependent, e.g. the way the patient responds to administered drugs, the amount of lost CSF or the mechanical properties of pathological tissues. Given the above list, it becomes obvious that it is very difficult to reliably model brain deformation without use of intraoperative information. This assumption is the basis of our approach. Similar observations were reported in [34]. The use of intraoperative information for model guidance was suggested by a few groups ([5], [18], [47], [76], [81]). Although there are surgical navigation systems that use intraoperative data, e.g. input from intraoperative scanners ([24], [38]), they do not update available preoperative images (high resolution preoperative MRI, CT, MRA, fMRI, PET, SPECT and others) when the brain deforms, and therefore the precision with which they display preoperative data is still limited by the error due to brain shift.

Here we present an approach for dealing with the problem of brain shift that relies on a combination of intraoperative input and a biomechanical deformable brain model. We start by giving an overview of the general steps needed for the approach. Then we introduce a damped spring-mass brain model guided by sparse points delineated on the exposed brain surface, and point out drawbacks of this method. In order to overcome the drawbacks, we move to a continuum mechanics based brain model guided by exposed brain surface data. Finally, we present a partial validation of the continuum mechanics based brain model using intraoperative MR image sequences.

5.2 System Overview

Our approach to brain shift compensation is to run an intraoperatively guided 3D brain model during the surgery and use the model output to display preoperative data (deformed according to the current model state). Before the surgery one can acquire anatomical (MRI, CT) and functional (functional MR, SPECT, PET, ...) images, segment them, generate surfaces of the segmented structures of interest, and then deform all of them intraoperatively based on the current model state. If the model deformation prediction is close to the actual brain deformation, then the displayed images and structures of interest (that are deformed according to the current model state) are closer to the current actual brain state than they would be if one didn't use the brain shift compensation, making the surgical navigation system more precise and reliable.

Therefore we propose a biomechanical-model-based brain shift compensation system composed of the following tasks: preoperative image acquisition, segmentation, mesh generation, registration of the model to the intraoperative environment, model setup and guidance, and visualization of model-updated preoperative data.

5.2.1 Segmentation, Visualization, and Registration

The first step after the preoperative image acquisition is the brain tissue segmentation, i.e. labeling of each voxel of the 3D image of the head as brain tissue or non-brain tissue. For this task, we have adopted the automatic brain segmentation algorithm described in [62].

For object surface rendering, we have used the algorithm presented in [25]. Some of the surfaces produced by this algorithm can be seen in Figs.

5.1, 5.2, and 5.3.

In order to display and use brain surface data for model guidance, a rigid body transformation between the patient and the preoperative image coordinate systems has to be established. For this purpose, we used a set of fiducial markers placed on the skin of the patient. In the operating room (OR), the marker coordinates were recorded using a mechanical localizer [54]. In addition, the markers were manually localized in the preoperative MRI dataset (markers have to be imageable in MR and/or CT scanners). Then, the optimal (in the least squares sense) rigid body transformation between the two sets of marker locations was computed using the method described in [4]. Once the rigid body transformation is determined, any point recorded by the localizer can be mapped to the preoperative image coordinate system.

5.2.2 Mesh Generation

The next step is to generate the model mesh from the segmented brain tissue. Here we use hexahedral ("brick") elements[1], having 8 nodes at the vertex positions. The segmented object (the brain tissue in this case) is the input to our mesh generator, which generates an unstructured mesh ([43]). The algorithm first generates a regular 3D matrix of bricks over the full 3D image. The brick side length is the only parameter of the algorithm. Each brick that has at least a half of its volume inside the segmented object is kept,

[1]The term "brick element" (or just "brick") is used in FEM analysis, and we use it for the continuum mechanics based brain model, since we solve it using a FEM. However, we use the same term ("brick element") for the spring-mass model, since the nodes are organized in "bricks", and the "brick" mesh structure is used for the trilinear interpolation. We hope that there will be no confusion because the two models are described separately.

and others are discarded. The kept bricks will compose the final mesh, while their nodes will be finely readjusted. The nodes are divided into two groups. Each node that has all of its neighboring nodes left is called an interior node, and all other nodes are called surface nodes. Each surface node is moved to the closest point on the surface of the segmented object. Note that surface nodes before moving were not far from the surface of the segmented object. Finally, the interior nodes are smoothed using a Laplacian-type smoother[2] ([43]), in order to enhance the regularity of the mesh, which is needed by FEM solvers. A typical output of the mesh generator is shown in Fig. 5.1. The meshes we use do not capture all of the fine details of the segmentation output, but they still achieve a reasonable performance in terms of accuracy and speed. We have determined the mesh size (brick side length) by testing various sizes and comparing the accuracy and speed of the method. A much finer mesh, that would capture all brain geometric details (e.g. sulcal structures), would have too many nodes and would slow down the computation, while not achieving a significant improvement in accuracy.

5.2.3 Intraoperatively-Guided Biomechanical Brain Model

In our initial efforts to recover intraoperative brain deformation, we used a damped spring mass model (Section 5.3.1) for its simplicity, speed, and ability to model slow and small soft tissue deformation. As we further explored the problem of brain shift compensation, we moved to a continuum

[2]Each interior node is moved to the mean position of its neighboring nodes, while surface nodes are kept fixed. This is iteratively repeated until interior nodes achieve a steady state, i.e. until the displacement of the interior node that moved the most in the current iteration is less than a given value.

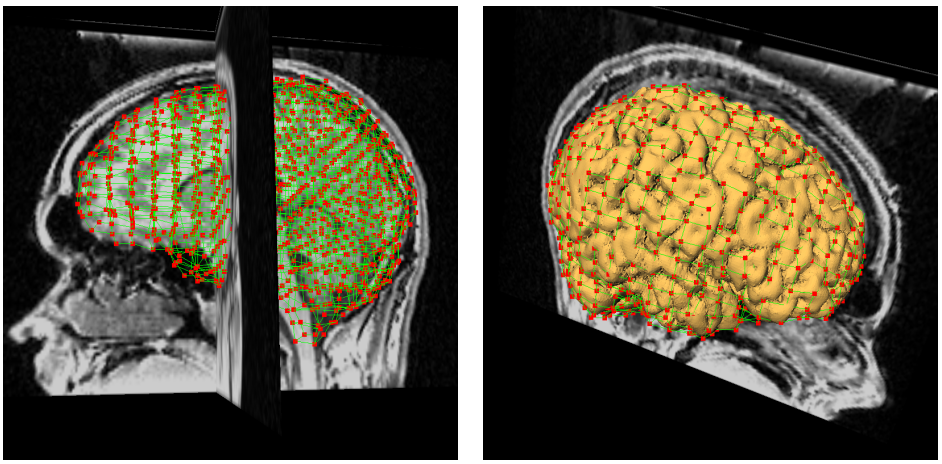


Figure 5.1: **A Typical Brain Model Mesh.** The left figure shows the mesh, while the right one shows the mesh and the outer brain surface. The mesh has over 2000 nodes and 1500 elements (bricks).

mechanics-based model (Section 5.3.2) which is also able to recover small soft tissue deformation, and although computationally more expensive, it overcomes drawbacks associated with the former model. Both models are guided by exposed brain surface data. While for the spring-mass model we used a few brain surface points to guide the model, as we moved to the continuum mechanics-based model we used a surface reconstruction of the exposed brain surface for model guidance.

5.2.4 Interpolation

The result of solving the model equations (Chapter 2) is a set of node displacements over time. One usually wants to display updated (deformed) preoperative images and surfaces of objects of interest. For this purpose, we employ trilinear interpolation to determine the displacement field in be-

tween model nodes. This interpolation scheme provides C_0 continuity of the displacement field.

It turns out that this task requires both the forward and inverse trilinear interpolation. The forward trilinear interpolation equations can be written in the following matrix form

$$[x\ y\ z]^T = A[1\ \alpha\ \beta\ \gamma\ \alpha\beta\ \alpha\gamma\ \beta\gamma\ \alpha\beta\gamma]^T, \tag{5.1}$$

where x, y and z are coordinates in the global coordinate system, while α, β and γ are the corresponding local coordinates in the coordinate system of the "brick" element. The 24 elements of the 3×8 matrix A depend directly on the (known) displacements at the eight nodes of the "brick" element[3], and can therefore be considered known. Their computation is straightforward (the local coordinates take values 0 or 1 at the eight "brick" nodes). While one just needs to evaluate (5.1) to do the forward trilinear interpolation, i.e. to obtain the global coordinates for given local coordinates, it is more difficult to do the inverse trilinear interpolation, i.e. to determine the local coordinates for given global coordinates. The inverse problem can be solved explicitly, but the solution expressions are cumbersome (there are three solutions since it is a cubic equation in local coordinates). After testing the explicit solution we have found it not practical, and have decided to numerically solve the inverse interpolation. We use an iterative bisection method for this purpose. We do the bisection search in the three local coordinate directions (see [58]). The method converges rapidly achieving the given precision (.1 mm) in several iterations.

An example of a deformed surface is given in Figure 5.3 (b,c), while

[3]The displacements at the nodes of the "brick" element are determined by solving the model equations.

model updated images are shown in Figure 5.4 (c, f).

5.3 Biomechanical Brain Model

5.3.1 A Damped Spring-Mass Brain Model

In our research we are mainly concerned with (but not limited to) issues surrounding epilepsy surgery. To quantitatively investigate such a case, we have recorded a set of six points (anatomical landmarks) on the exposed brain surface approximately every eight minutes during the surgery starting when the dura was opened. The mean shift in the direction perpendicular to the brain surface was about 3 mm. The initial and final set of points both displayed over the same pre-deformation brain surface generated from a preoperative MR scan are shown in Fig. 5.2. By initial moment, we mean the moment when the dura was cut, and the final moment is when the brain settled down and achieved a steady state. This result clearly shows the need for a high quality intraoperative 3D acquisition system and/or a method for brain shift compensation. Tradeoffs among different approaches to these problems are discussed later in the chapter.

Here we use the spring-mass model presented in Section 2.2.

Contact Algorithm for Brain-Skull Interaction

The brain-skull interaction as modeled in our initial efforts in [80], is a highly nonlinear function, and significantly slows down the adaptive step-size numerical integration. The consequence was that the steady-state for this previous 3D model (with about 1000 nodes and 5000 connections) was reached in approximately four hours, which is much slower than the real

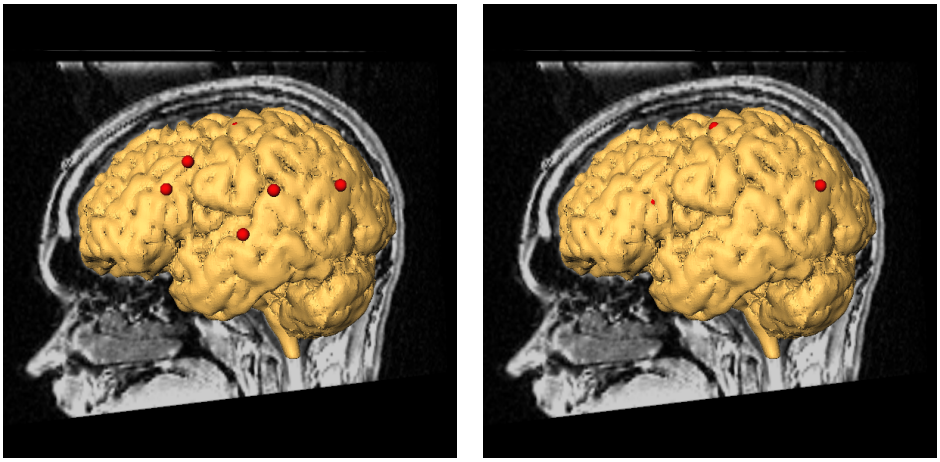


Figure 5.2: **Intraoperatively Recorded Points on the Exposed Brain Surface** Intraoperatively recorded points on the exposed brain surface at the beginning of the surgery are shown at left, while their positions about 45 minutes later relative to the same pre-deformation brain surface are shown at right. The points moved in the direction of gravity (which is perpendicular to the sagittal plane) for a few millimeters and they are hidden under the pre-deformation brain surface (only one of the points is still visible in the figure at right). Since the brain deformed (in the direction of the gravity vector), the surface points moved relative to the pre-deformation brain surface.

brain deformation, and therefore the model cannot be used for display updating during the surgery.

To increase the computational speed, we have moved to a contact algorithm based approach. Prior to the simulation, the skull and brain tissue have to be segmented. Ideally, a preoperative MRI scan would be used for brain tissue segmentation and a preoperative CT scan for skull segmentation (the MR and CT scans would be registered, e.g. by the algorithm suggested

in [66]). However, since we didn't have preoperative CT scans available, we did an approximate skull segmentation from preoperative MR images, using a combination of automated and manual processing steps. The brain-skull interaction is not directly a part of the model equations, but rather it is incorporated via numerical integration, through a contact algorithm. As the model evolves over time, when a node enters the skull area, it is returned to its previous position (to its position from the previous step in the numerical integration). This prevents nodes from entering the skull, but permits them to come arbitrarily close to it (more precisely, close up to the precision set in the numerical integration) and move along the skull surface if pulled by forces that are not perpendicular to the skull surface. Effectively, nodes can move freely unless they reach the skull, in which case they can move only in the direction tangential to the skull surface. This behavior is identical to the one achieved by the brain-skull interaction suggested in [80], but it is much faster to simulate. As a result, the 3D model needs about 10 minutes to reach a steady state, which is faster than the actual brain deformation (which is, according to our surgical colleagues and our own measurements, approximately 45 minutes).

Parameter estimation

One of the problems in soft tissue deformation recovery is a reliable model parameter estimation. The approach we have employed here is to use intraoperative measurements to estimate model parameters. Although our model allows for local parameter control, we still assume a homogeneous model for two reasons. First, it is very difficult to estimate the brain tissue parameters locally and second, there are contradictory reports in the literature regard-

ing white and gray stiffness properties. For our approach, even in the case of a homogeneous model, there are two parameters to be estimated: the stiffness coefficient k_s and the damping coefficient k_d in (2.2).

Let $S(t)$ be the brain surface generated by the model at time t, and let $x_i(t),\ i = 1, \ldots, N$ be the recorded brain surface points (in our case $N = 6$) at time t. Further, let $d(x, S)$ denote the signed distance between the point x and the closed surface S, where $|d(x, S)|$ is the distance between x and S, and $d(x, S)$ is negative if the point is inside the surface, and positive if it is outside. We treat the brain surface as a closed surface. The average signed distance of the recorded points to the brain surface at time t is

$$d(t) = \frac{1}{N} \sum_{1}^{N} d(x_i(t), S(t)), \tag{5.2}$$

and the total average signed distance (over time) is

$$d = \frac{1}{M} \sum_{1}^{M} d(t_i), \tag{5.3}$$

where M is the number of times brain surface points were recorded, and $t_i,\ i = 1, \ldots, M$, are the corresponding times.

We used an off-line parameter estimation, where the whole sequence (over time) of the recorded brain surface points was utilized. Model simulations indicated that the steady state did not depend on the damping coefficient, but only on the stiffness coefficient. This conclusion coincides with the exact mathematical analysis of simple one-object systems (e.g. a mass connected to a spring that is fixed at the other end, with a friction between the mass and the support). The damping coefficient determines how fast the steady state will be reached, while the stiffness coefficient determines the final shape of the brain.

For this reason, we use the steady state to estimate the stiffness coefficient. The time of the steady state is t_M, and it denotes the time when the last set of points was recorded (after the brain settled down, i.e. achieved a steady state). Since reports by other researchers as well as experiments done by our group suggest that the brain shifts mainly in the direction of gravity, we use the following approach to estimate the model parameters. We pick a relatively small value k_1 for the stiffness coefficient, such that $d(t_M)$ (defined by Eq. 5.2) is positive[4]. Similarly, we pick a relatively large value k_2 for the stiffness coefficient, such that the corresponding $d(t_M)$ is negative. The next step is to use $k_{new} = (k_1 + k_2)/2$ as a stiffness coefficient, run the model again, and get a new value for $d(t_M)$. If $d(t_M) > 0$, we set $k_1 = k_{new}$. Otherwise, we set $k_2 = k_{new}$. Then we repeat the steps, i.e. continue with the bisection search, until $d(t_M)$ is close enough to zero. The last k_{new} is used as the "optimal" stiffness coefficient. For all the steps, we use an arbitrary value for the damping coefficient since it does not affect the steady state.

Once the stiffness coefficient is estimated, the damping coefficient is determined in a similar fashion (using the bisection search), but this time by using d (defined by Eq. 5.3) rather than $d(t_M)$, i.e. by reducing the average signed distance over time.

The idea was to estimate the model parameters on a number of patients and then use their average values for future patients for brain deformation compensation. We intraoperatively recorded exposed brain surface points

[4]For a small stiffness coefficient the model is "soft", and it will settle down significantly, i.e. more than the actual brain, causing all the recorded points at time t_M to be outside the brain model surface. For this reason each point will have a positive signed distance to the brain model surface.

for four patients, but due to technical difficulties we were able to use the data only of two of them. From the two patients we determined the average model parameter values. These parameter values were used for all other experiments with the damped spring mass model discussed in this chapter.

While one could record brain surface points for more patients and get a possibly better estimate for the model parameters, there are a few problems related to this approach. The main problem is that the model parameters are mesh dependent, i.e. if a denser (or sparser) mesh is required, one would need to readjust the model parameters to achieve the same model behavior. However, it is not clear how to mathematically readjust the model parameters for a denser mesh, except for a 1D model. For the same reason one cannot use a nonuniform mesh (mesh with different node densities in different regions). Since the model parameters are mesh dependent they cannot be found in the literature. Furthermore, if we want to use the average values for the model parameters estimated from a number of patients, we would need to use model meshes of the same density for all the patients. Mesh dependence of the model parameters is one of the reasons why we have decided to move to continuum mechanics based modeling, rather than further investigating damped spring-mass models. Model parameters in continuum mechanics based models are mesh independent, which avoids all the aforementioned problems.

Table 5.1 shows the average distance between the rigid (initial) gray/CSF brain surface and recorded brain surface points over time (i.e. during the operation) in the row "surface movement". In addition, the row "model error" contains the average error between the model predicted position of the gray/CSF brain surface and the positions of the recorded brain surface points over time. This table contains data for a single patient undergoing an

time[min:sec]	0:00	7:40	14:40	19:40	24:40	34:52	49:00	max
(a)	0.34	1.38	2.21	2.30	2.74	3.24	3.29	3.29
(b)	0.34	0.45	0.30	0.13	0.20	0.32	0.04	0.45

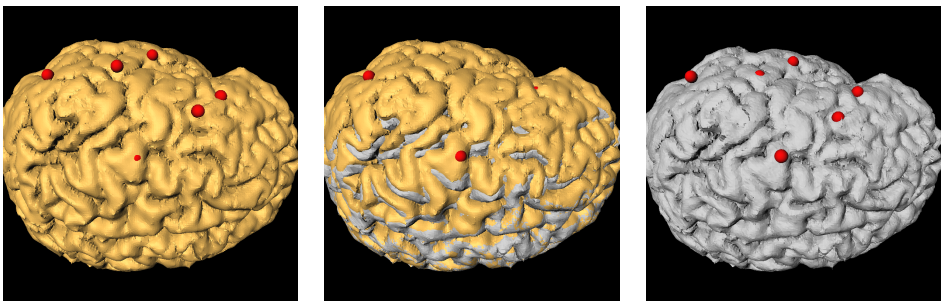
Table 5.1: **Average Brain Surface Movement and Model Error.** Row (a) contains true brain surface movement [mm], while row (b) contains the model error [mm].

epilepsy surgery. The surgeon touched six points (recorded their positions with the mechanical localizer) every eight minutes (on average). One can see that the distance between the initial gray/CSF brain surface and the recorded brain surface points increases over time and ultimately reaches 3.29 mm. The model with optimal parameter settings (determined off-line from two patients) has maximal error of 0.45 mm over time. While the error between the model prediction and the actual surface points is relatively small, one should keep in mind that the model parameters were estimated on just two patients, and that the results shown in Table 5.1 present almost the best fit of the model to the measured points on the brain surface of one of the two patients.

Intraoperative Model Guidance

In Section 3.2 we have presented a way to guide the spring-mass model. An example of a guided brain deformation model output is shown in Fig. 5.3.

To validate the approach one would need to obtain for multiple subjects dense time sequences of 3D brain images using intraoperative sensing, and then compare the model predictions to the actual deformations in the full brain volume. This can be done by using intraoperative MRI data, or maybe



(a) **(b)** **(c)**

Figure 5.3: **An Example of a Guided Brain Model Output. (a)** shows the recorded points at the beginning of the surgery with the initial (pre-deformation) brain surface. Note that the points are on the brain surface. **(b)** represents the final (steady state) brain surface points with initial brain surface (yellow surface) and final brain surface (gray surface). One can see that the brain surface points moved inside the initial brain surface. This is due to the effect of gravity that pulled the brain downwards. **(c)** represents the final brain surface points and final brain surface after model-guided updating. The points are again on the brain surface. The final brain model surface was computed using the final model state, while the final points are the measurements on the brain surface when the brain settled down.

data from an intraoperative CT or ultrasound scanner.

However, since the only available intraoperative data for these experiments are exposed brain surface points recorded over time, we randomly selected two of the points to guide the model and compared the model predictions against other four points. The results are given in Table 5.2.

Although the error for the guided model is reduced with respect to the case with no brain compensation, this test was done only on one patient, and

time[min:sec]	0:00	7:40	14:40	19:40	24:40	34:52	49:00	max
(a)	0.34	1.38	2.21	2.30	2.74	3.24	3.29	3.29
(b)	0.34	0.12	0.38	0.44	0.35	0.49	0.14	0.49

Table 5.2: **Average Brain Surface Movement and Guided Model Error.** Row (a) contains true brain surface movement [mm], while row (b) contains the guided model error [mm].

with very sparse (both in time and space) intraoperative data. An extensive validation using intraoperative volumetric imaging is necessary to determine the reliability of this approach.

The model presented in this section is able to run in real time. By real time we mean the following. The brain deforms with certain speed (it takes about 45 minutes to assume a steady state). On the other hand it takes a certain time to simulate the brain deformation on a computer, i.e. to deform the model. However, at say 5 minutes after opening the dura (5 minutes of the actual, surgical time) the corresponding model state (the state that corresponds to the 5th minute of the actual time) has already been computed and stored in the memory, and can be used for displaying (deformed) images and surfaces. Thus, the simulation of the brain deformation is computed faster than the actual brain deformation, i.e. in real time. If this is not the case, i.e, if the simulation of brain takes more time than the actual brain deformation, then the model could not be used during the surgery (in real time) for displaying deformed images and surfaces. The model meshes used here had over 2000 nodes, 11500 connections, and 1500 elements (bricks), and it took less than 10 minutes on an Octane SGI workstation (R10000 250 MHz processor) to reach a steady state. This is a significant improvement

over our previous model ([80]).

5.3.2 A Continuum Mechanics Based Brain Model

In spite of the advantages of spring-mass models, we have decided to move to continuum mechanics based models for reasons explained in Section 2.3. The model mathematics is presented in Section 2.4.

For this model, we use a pair of stereo cameras overlooking the exposed brain surface to acquire intraoperative information about the deforming brain. The idea is to reconstruct and track the exposed brain surface as it deforms during the surgery. If this can be done reliably, one can use the reconstructed brain surface as displacement boundary conditions for the model PDEs. Each time the surgeon moves her or his hands and surgical tools out of the way of the cameras, snapshots from the two cameras are taken, exposed brain surface is reconstructed, the surface is used to guide the model, and once the model is deformed, it can be used to update (properly warp) all preoperative images available. The advantage of this intraoperative data acquisition over the manual point delineation (Section 5.3.1) is that it is automated, less disturbing for the surgeon, and it provides more data.

As pointed out in Section 2.4.2, the only model parameter to be set is Poisson's ratio. We have tested a range of values for ν, and the one that yielded the smallest error (a partial validation is presented in Section 5.3.2) was $\nu = .4$, which is a value used by other groups as well ([21]). We assume that the model is homogeneous since there is no reliable way known to us for setting the model parameter for different brain structures.

Partial Validation

In this section we will investigate how well a continuum mechanics-based brain model can predict in-volume deformation using only partial (exposed brain) surface data for model guidance.

In order to test the method, we used intraoperative MR image sequences. We manually segmented the pre-deformation brain regions of interest (cerebral hemisphere at the side of the craniotomy, falx, and tentorium). Then we rigidly registered the preoperative (undeformed) and intraoperative (deformed) brain images using a normalized mutual information based registration algorithm ([65]). The purpose of this step was to align the undeformed and deformed brain images, i.e. to remove the rotation and translation from the transformation between the two images. We employed a finite element method to determine the deformation governed by Eq. 2.7. A mesh composed of hexahedral ("brick") elements (with 5 mm approximate side lengths) was generated using the segmented data and the in-house mesh generator explained in Section 5.2.2. The generated meshes (of the cerebral hemisphere at the side of the craniotomy excluding ventricles) had about 6,500 nodes and about 5,000 "brick" elements. Here we used the anatomical constraints that the falx and tentorium are practically fixed, and we fixed the corresponding model nodes. For this reason it is enough to consider only the half of the brain at the side of the craniotomy, since the other part does not deform. We are aware that, although this assumption holds in most cases, there are exceptions where the falx moved during the surgery. In order to simulate exposed brain surface tracking[5] we manually segmented the de-

[5]In a complete system brain surface tracking would be done by using a pair of stereo cameras, as explained in Section 3.3.2.

formed brain from the intraoperative scan and generated its surface. Then we computed the displacement at each point $\boldsymbol{r_1}$ of the undeformed brain surface S_1 (only at the part of the brain surface that was visible through the craniotomy, i.e. at the exposed brain surface), as $\boldsymbol{\Delta r} = \boldsymbol{r_2} - \boldsymbol{r_1}$, where $\boldsymbol{r_2}$ is the point on the deformed brain surface S_2 closest to the point $\boldsymbol{r_1}$, i.e. obtained as $\arg_{\mathbf{r_2} \in S_2} min||\mathbf{r_2} - \mathbf{r_1}||$. Finally, the computed displacements at the exposed brain surface were used as boundary conditions for the model PDEs.

In this section we present a partial validation of the method using intraoperative MRI for two cases: a sinking brain and a bulging brain. In both cases, we used intraoperative MR images after the dura was opened and brain deformed, but before any major resection occurred. The intraoperative MRI datasets had 60 slices of a 256 by 256 size, with .9375 mm in-plane resolution and 2.5 mm slice thickness. Their quality, i.e. the signal to noise ratio (SNR), was lower than the quality (SNR) of typical preoperative MR images. For both cases we generated the model and displacement boundary conditions as explained above. We used ABAQUS to compute the model deformation. For a model of about 6,500 nodes and about 5,000 "brick" elements, it took about 80 seconds to solve the equations on an SGI Octane R12K machine. This time is almost practically applicable, since it would mean that after a couple of minutes after obtaining exposed brain surface data, one would get updated MR images and other preoperative data. In order to validate the computed deformation, we manually selected a set of anatomical landmarks in the preoperative scan of the (undeformed) brain at various locations throughout the volume of the cerebral hemisphere at the side of the craniotomy. For landmarks, we used points at anatomical structures that can relatively easily be identified in both preoperative and

intraoperative images. Then we manually found the corresponding landmarks in the intraoperative scan of the (deformed) brain. Finally, using the displacement field computed by the model, we determined the positions of the "model predicted landmarks" in the deformed brain corresponding to the landmarks in the undeformed brain, and compared them to the corresponding manually set landmarks in the deformed brain.

Table 5.3 shows the true displacements of 14 landmarks, the computed corresponding displacements, and the errors between the computed and true landmarks in the deformed brain for the sinking brain and for the bulging brain, respectively. One can see that the maximal true landmark displacement was 3.8 mm (3.6 mm) while the maximal error was 1.4 mm (1.3 mm) for the case of the sinking (bulging) brain. Fig. 5.4 shows an MR image slice of a preoperative brain, the corresponding intraoperative image slice of the deformed brain, and the corresponding model-updated image slice of the deformed brain.

5.4 Discussion

We presented a brain shift compensation method based on a biomechanical model guided by limited intraoperative data. We started by a simple and relatively fast damped spring-mass brain model. The problems associated with the model parameters and guidance are overcome by an approach that relies on a continuum mechanics-based brain model.

Clearly, this system uses only intraoperative surface information and it cannot perform well after tissue resections. In addition to using this system before resections, one can use it in the case of subdural electrode implantation (often performed as a first stage of epilepsy surgery) where no

tissue is removed, but the brain still deforms due to gravity, loss of CSF and other mentioned factors.

The approach using continuum mechanics-based brain shift compensation indicates that exposed brain surface information might be enough to recover pre-resection brain deformation with an error comparable to the scan resolution. The used intraoperative MR scans had 2.5 mm slice thickness, with in-plane .9375 mm by .9375 mm pixels, while the maximal error of the predicted brain deformation in the presented cases was 1.4 mm.

In addition, this work compares the spring mass and the continuum mechanics-based models for brain shift compensation. The main advantages of the spring mass model are its simplicity and computational efficiency. However, it suffers from mesh dependence on model parameters and from the lack of a good model guidance strategy. Both of the problems are overcome by the continuum mechanics-based model, which disadvantage is that it is computationally more expensive. Furthermore, the continuum mechanics-based model allows for incompressibility control through Poisson's ratio[6], while it is not clear how to achieve it in the case of the spring-mass model.

As noted above, an alternative to using biomechanical model-based brain shift compensation is to use an intraoperative MRI and/or CT system. The primary advantage of these approaches is that they provide the actual state of the brain at the time of imaging, i.e. during the surgery, while a disadvantage is that they are very expensive, and therefore not affordable to all hospitals. They also restrict surgical access to the patient, prevent standard metal surgical tools from being used, their spatial resolution is typically not as high as that of preoperative MRI, and one must interrupt the surgery

[6]Poisson's ratio controls model incompressibility. This parameter is dimensionless, it can relatively reliably be estimated, and its values are available in the literature.

for a few minutes for each image acquisition. Even with the advantages of intraoperative MR scanners, one can see only the current (intraoperative) anatomical state of the brain, but the functional and segmented data is still available only in the preoperative state. Therefore the use of a deformable model would be helpful even if an intraoperative scanner is available since the model can be guided by the rich intraoperative data from the scanner, and it can correspondingly deform any preoperative data. The main advantages of the use of a deformable model over the use of intraoperative scanners is its much lower cost and its ability to deform any data type, while the main disadvantage is its lower precision and reliability since it is practically impossible to accurately model all the complex phenomena that influence the brain deformation during the surgery.

Landmark	Case I			Case II		
	$\|\boldsymbol{t}\|$	$\|\boldsymbol{c}\|$	$\|\boldsymbol{e}\|$	$\|\boldsymbol{t}\|$	$\|\boldsymbol{c}\|$	$\|\boldsymbol{e}\|$
1	**.7**	**.3**	**.8**	**2.7**	**2.0**	**.8**
2	**.9**	**.5**	**1.4**	**1.8**	**1.6**	**1.0**
3	**.6**	**.7**	**.4**	**.6**	**1.1**	**.6**
4	**.1**	**.2**	**.2**	**3.6**	**2.4**	**1.3**
5	**2.3**	**1.7**	**.7**	**2.6**	**2.6**	**.8**
6	**2.9**	**2.4**	**1.3**	**.8**	**.5**	**.4**
7	**2.1**	**1.4**	**1.4**	**1.3**	**.8**	**.9**
8	**1.0**	**.7**	**.4**	**1.1**	**1.2**	**.8**
9	**1.9**	**1.3**	**1.2**	**1.4**	**1.5**	**.9**
10	**2.7**	**1.8**	**1.3**	**.7**	**.8**	**.5**
11	**.8**	**.4**	**.4**	**.7**	**.5**	**.7**
12	**.8**	**.5**	**.8**	**.4**	**.2**	**.5**
13	**2.1**	**1.9**	**1.0**	**2.4**	**2.0**	**1.2**
14	**3.8**	**3.0**	**1.2**	**.5**	**.3**	**.7**

Table 5.3: **Model-Predicted Landmark Position Error.** Case I (sinking brain) and Case II (bulging brain): true landmark displacements ($\boldsymbol{t}$), computed landmark displacements ($\boldsymbol{c}$), and error between true and computed landmark locations ($\boldsymbol{e} = \boldsymbol{c} - \boldsymbol{t}$), for 14 landmarks. All values are in millimeters.

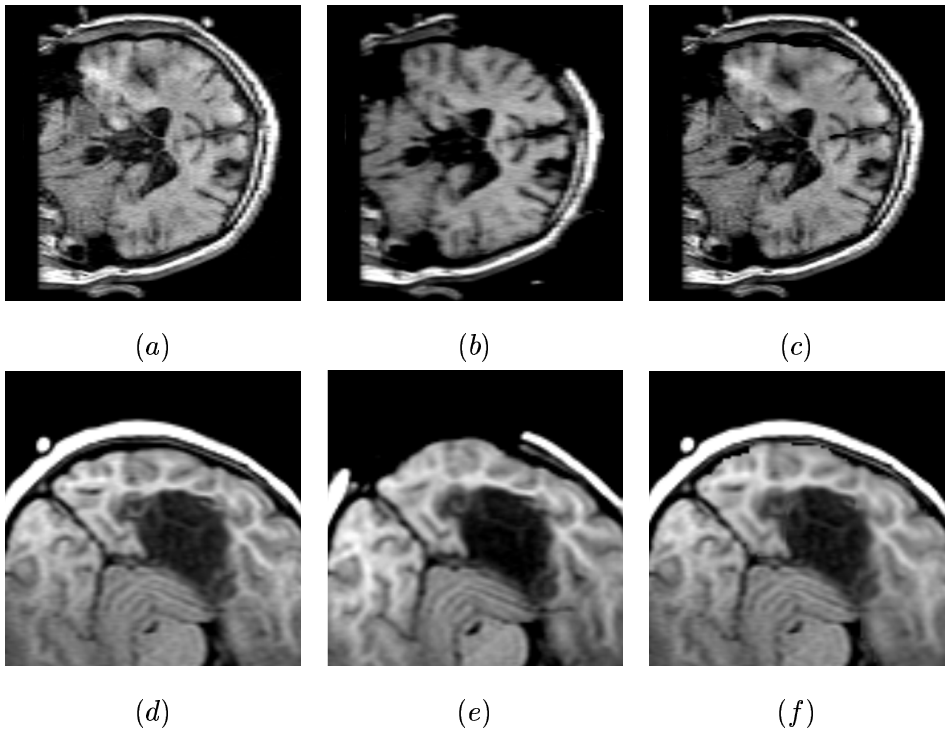


Figure 5.4: **Model-Updated Preoperative MR Brain Images.** (a) A preoperative coronal slice of a sinking brain, (b) the corresponding intraoperative slice of the deformed brain, (c) the corresponding model-computed slice of the deformed brain. Axial slices (d), (e), and (f) correspond to the bulging brain case (undeformed, deformed, and model-computed, respectively). Note that in both cases the exposed brain surfaced in the computed slice moved similarly as the corresponding surface in the intraoperative slice.

Chapter 6

Localization of Implanted Subdural Electrode Grids, Strips and Depth Electrodes

6.1 Introduction

Subdural electrodes are often used in epilepsy surgery in order to map brain function and locate seizures. The patient carries implanted electrodes for several days, and over this time the electrodes are monitored for seizures and stimulated to determine brain function and the results are recorded. To effectively use these results, one needs to relate electrode locations to brain structures of interest. This is conventionally done by using postoperative MR or CT scans (scans taken after electrodes have been implanted), and looking for electrodes in the scans. Since it takes a significant effort to picture in mind the spatial relations between electrodes and structures of interest directly from postoperative images, we have developed a combination

of automatic and manual tools for localization and visualization of implanted subdural electrode grids, strips and depth electrodes. The tool automatically locates electrode positions, and allows the user to interactively correct, if needed, the result of the automatic localization. Both the automatic and interactive tools are based on the idea of preserving intrinsic surface distances, used as a geometric constraint in the algorithms. In addition, the user can color electrodes based on the function they correspond to. The final 3D display allows neurosurgeons and neurologists to easily visualize electrodes and consequently concentrate solely on medical analysis.

Electrodes are typically not implanted as single, separate electrodes, but rather as electrode groups kept together by a supporting rubber material. There are three types of electrode groups: electrode grids, electrode strips, and depth electrodes. An electrode grid is at least a 2 by 2 matrix of electrodes, i.e. there are at least two rows and at least two columns. Electrode grids are sometimes cut by the surgeon if they cannot fit in the place otherwise. The surgeon can make a few holes in the grid as well as remove some of the border electrodes. An electrode strip is a 1 by N array of electrodes, i.e. it is a single row, and it must have at least two electrodes. A depth electrode is also a single row of at least two electrodes.

Figure 6.1 shows an electrode grid in the process of implantation. Once electrodes are implanted, the skull is closed with wire contacts to all the electrodes available outside the skull, and the patients carries the implanted electrodes for several days. Electrodes are used to map brain function by stimulations and determine seizure locations by monitoring electrode activations. In order to effectively use stimulation and activation results, one needs to relate electrode locations to some of the head and brain structures. This is usually done by taking a scan (MR or CT) of the patient after the

electrodes have been implanted (so called postoperative scan). Figure 6.2 shows typical postoperative MR images of a patient with implanted subdural electrodes. However, clinicians find this approach limiting for the following reason. If one looks at individual 2D image slices, since usually there are just a few electrodes in a slice, it is not clear where they are in the grid (e.g. a grid can have 64 electrodes, arranged in an 8 by 8 matrix), and it is difficult to relate them to brain structures. Similar problem is encountered if postoperative images are volume rendered. For this reason we developed a tool for extraction and visualization of electrode grids (and electrode strips and depth electrodes). The tool extracts the grid as a whole, rather than as a set of non-related electrodes, by fitting a smooth, curved surface through the estimated centers of the electrodes. A by-product is the ability to reliably estimate the orientation of the electrodes[1]. Knowing the centers and orientations of the electrodes, and the surface of the grid supporting material, the whole grid can be realistically displayed in 3D, together with brain structures of interest. The grid supporting material is usually a transparent rubber. Although the supporting material is rubber, grids are never subject to forces strong enough to stretch them. In addition, the electrodes in the grid corresponding to particular functional areas can be colored, further helping visualize the correspondence between functional areas and brain structures. By using our tool, it is now much easier to reliably locate functional areas with respect to other structures, and plan and guide the surgery.

To the best of our knowledge there are no such methods available in the literature. While our method is automatic, there are two other approaches

[1]Electrodes used at Yale New Haven Hospital are disk-shaped and are 3.8 mm in diameter and about .5 mm in height.

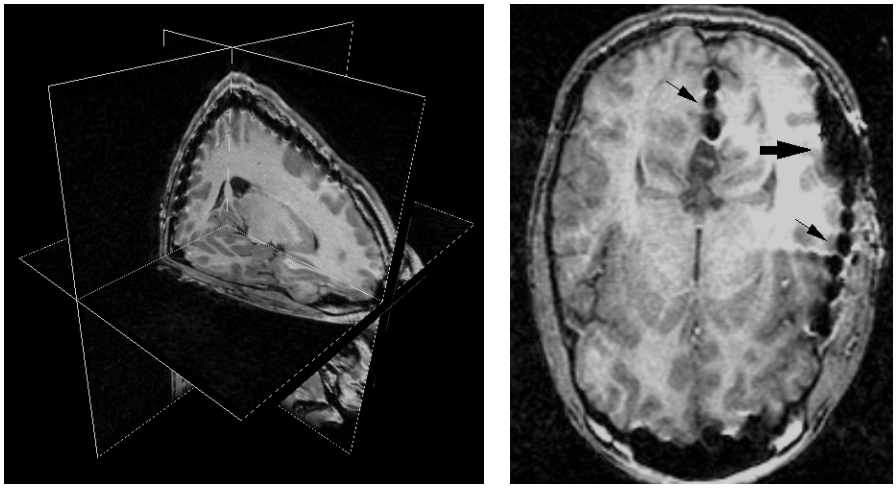
commonly employed: manual localization of electrodes ([11]), which is time consuming, or use of a postoperative CT scan for electrode localization. The advantage of using postoperative MR images is that anatomical structures are better visible compared to CT images, which is very important for making further conclusions. The problem is that metal electrodes corrupt the magnetic field in MR scanners causing image artifacts (voids), which makes electrode localization from MR images more difficult. Although there are no such artifacts due to electrodes in postoperative CT images, and therefore their localization is easier, postoperative MR images are more commonly used due to better imaging of anatomy. Our method was reported in [74] and [75].

6.2 Automatic Electrode Localization

Electrodes used at Yale New Haven Hospital produce sphere-shaped artifacts of about 10 mm in diameter in MR scans (there is a sphere artifact at the position of each electrode). Artifacts appear to be dark (void), hide nearby tissue, and in addition, they are often mixed with other head structures that are normally dark (e.g. bone, cerebro-spinal fluid - CSF). Moreover, the wires connected to electrodes produce artifacts, especially where they are joined together (at the place they leave the skull). Fig. 6.2 shows typical artifacts in postoperative MR scans.

Because of artifacts and image noise it is very difficult to reliably estimate electrode positions if electrodes are treated independently of each other. It is even more complicated to estimate electrode orientations (the orientation of electrode disks) since sphere-shaped artifacts carry almost no orientation information, making the orientation estimation very sensitive to

Figure 6.1: **Electrode Grid Implantation.** An 8 by 8 subdural grid of electrodes is placed on the brain surface at the craniotomy. The white wires coming out the gap between the brain and the skull belong to electrode strips that have already been implanted.



(*a*) (*b*)

Figure 6.2: **Postoperative MR Images of a Patient with Implanted Subdural Electrodes.** Figure (a) shows three orthogonal sections through a postoperative MR scan. One can see artifacts caused by electrodes of an 8 by 8 grid between the top of the brain and the skull. It is not easy to determine where those electrodes are in the grid and how they are related to brain structures of interest. Figure (b) shows an axial slice of another patient. Artifacts are usually sphere-shaped (small arrows), but sometimes (big arrow) are so dominant that it is very difficult even for the human eye to locate electrodes.

noise. The idea is to treat an electrode grid (e.g. an 8 by 8 grid) as a whole. There are two main reasons for doing this. First, this includes prior knowledge (the geometry of the grid is known prior to implantation) and second, errors in estimating electrode positions tend to cancel out. Our method is based on a combination of nonlinear and predictive filtering subject to differential geometry constraints. The algorithm fits a smooth, curved surface through the estimated electrode positions, and even if an estimated electrode position is significantly off, geometric constraints push it very close to its true position. As a by-product, once the surface is fitted, one can determine the surface normals at electrode positions and use them to properly orient electrode disks. Electrode orientations are not as important as electrode positions (the aim is to locate functional areas), but they help visualize electrode grids more realistically.

The constraint that electrode grids are deformable but not stretchable is used. Electrode grids when implanted are deformed to fit into a place, but are not subjected to force strong enough to stretch or compress them. If $\boldsymbol{x}$ is a parametrization of an electrode grid surface (Definitions A.46 and A.47), then this constraint reduces to the fact that the integral

$$\int_C \left\| \frac{d\boldsymbol{x}}{dt} \right\| dt \tag{6.1}$$

is invariant under grid deformations for a fixed curve C, where t is a parameter along the curve. This has to hold for any curve on the grid surface.

Due to artifacts, noise, and many degrees of freedom of an electrode grid, it is very difficult in one step to reliably fit a smooth grid model of a known geometry that satisfies the previous constraint. Rather, the geometry and smoothness constraints are enforced through a few steps, each step introducing some portion of the constraints. The main steps of the method are:

nonlinear filtering, predictive filtering, surface regularization, and surface interpolation.

6.2.1 Nonlinear Filtering

The purpose of this step is to find a set of possible electrode centers. Typically, patients are implanted a few grids and strips of electrodes often totaling to more than a hundred electrodes. Since each electrode causes a dark artifact (void), the first idea was to cross correlate a 3D kernel similar to the sphere-shaped artifact over the postoperative MR 3D image[2]. Maxima of the normalized cross-correlation would be possible electrode centers. The problem with this is how to set the kernel. While the center of the kernel has to be a dark sphere, the rest of it has to be "white", or some kind of transition to "white". This pattern depends on the structures surrounding particular electrode, and it is not the same for all electrodes. For this reason, we have moved from linear filtering to nonlinear one, i.e. we do not perform ordinary cross-correlation, but maintain the same basic strategy to look for a sphere-shaped dark region surrounded by lighter tissue. Prior to nonlinear filtering the dataset is smoothed by a 3 by 3 by 3 kernel to reduce the effect of noise. The form of the 3D kernel we use for the nonlinear filtering is shown in Fig. 6.3(b). Each voxel in the dataset is checked by doing the following steps:

- The kernel is centered at the current voxel.
- If any voxel in the KERNEL CENTER exceeds the THRESHOLD value, the current voxel is discarded and the next one is checked.

[2]A 3D image is a set of 2D (MR) image slices stacked together.

Type	Parameter	Value	Description
Grid Related	DISK DIAMETER	3.8 mm	Electrode diameter
	DISK HEIGHT	0.5 mm	Electrode height
	OUTER MARGIN	5.0 mm	Supporting material margin
	DISTANCE	10.0 mm	Inter-electrode distance
Artifact Related	KERNEL CENTER	7.0 mm	Inner "dark" area
	KERNEL SPHERE	10.5 mm	The whole "artifact"
	MIN DISTANCE	7.0 mm	Minimal electrode distance
	MAX DISTANCE	14.0 mm	Maximal electrode distance
	MIN DIAGONAL	10.0 mm	Minimal square diagonal
	MAX DIAGONAL	17.0 mm	Maximal square diagonal
	THRESHOLD	40	"Dark" is under threshold

Table 6.1: **Automatic Electrode Localization Algorithm Parameters.**

- The number of voxel values under the THRESHOLD in the KERNEL CENTER and over the THRESHOLD out of the KERNEL CENTER but still in the KERNEL SPHERE is counted. The count is called SUM.

The parameters used in the algorithm are summarized and briefly explained in Table 6.1. All parameters are capitalized. There are two groups of parameters. Grid-related parameters depend solely on the geometry of the used electrode grids, and are further described in Fig. 6.3(a). Artifact-related parameters depend on the effect metal electrodes cause in the imaging system (MR scanner).

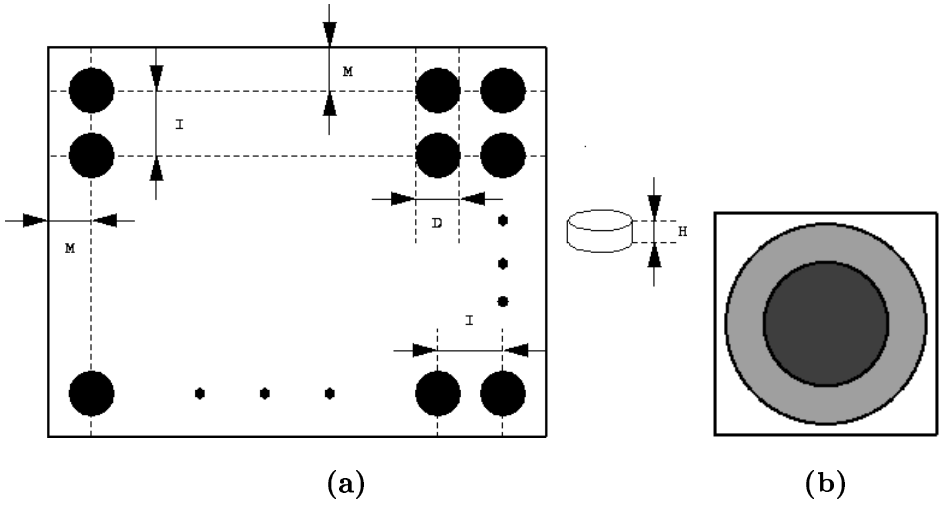


Figure 6.3: **Electrode Grid Parameters.** Figure (a) shows the grid related parameters, that define its geometry. The black circles represent the electrodes arranged in a grid, while a single electrode is show at the right. M is OUTER MARGIN, D is DISK DIAMETER, H is DISK HEIGHT and I is DISTANCE. Figure (b) shows a section through the 3D kernel. The innermost (the darkest) region is kernel center (its diameter is KERNEL CENTER) and the gray ring together with the kernel center is kernel sphere (its diameter is KERNEL SPHERE).

Thus, each voxel is either disregarded or its position and SUM are recorded. The bigger the sum, the more likely the voxel is an electrode center. However, there are typically many (thousands) of voxels that were recorded during the nonlinear filtering. At this point the first geometric constraint is introduced. Since electrodes on the grid we use are 10 mm apart (DISTANCE parameter), then any two estimated electrode centers should not be closer than a certain distance (MIN DISTANCE parameter). By imposing this constraint many of the estimated centers are rejected (the estimated centers that are kept are those with higher SUM).

The output of this step is a set of positions that are likely to be close to positions of possible electrode centers. The estimated positions are guaranteed to be at least MIN DISTANCE apart. Typically, a majority of the electrode centers are found, some are missed, and there are some other structures that are wrongly chosen as electrode centers. An example of estimated electrode centers is shown in Fig. 6.4(a).

6.2.2 Predictive Filtering

The purpose of this step is, using the estimated electrode centers, to label some of them, discard the rest and predict the positions of missing ones. The grid patterns are searched from the largest (typically 8 by 8) to the smallest ones. Each time a pattern is found, the electrodes belonging to that grid are removed from the list of estimated electrode positions, and then next largest grid pattern is looked for. However, since there are typically a few hundred estimated electrode centers, it would be computationally too expensive to check say an 8 by 8 grid pattern against all possible combinations of estimated electrode centers, including cases with missing electrodes. For this

reason we first determine what we call "low level topology". We look for all pairs of estimated centers that might be neighboring electrodes on a grid - these pairs are referred to as "links". Any two estimated centers are at least MIN DISTANCE apart (that was enforced in previous step), and two estimated centers make a "link" if they are at most MAX DISTANCE apart. Next, we look for all sets of four estimated centers that might compose a "square" on the grid, i.e. that make a 2 by 2 "subgrid". A "square" must have sides composed of already determined "links", and its diagonals must be at least MIN DIAGONAL and at most MAX DIAGONAL. An example of estimated centers and "low level topology" is shown in Fig. 6.4(b).

Now, having the "low level topology" available, it is much easier to search for grid patterns. For each "square" the pattern is searched for by sliding it over that "square". In other words, a pattern is positioned over the current "square", such that the "square" is in position (1,1) in the pattern, this case is checked, then the pattern is moved such that the "square" is in position (1,2) in the pattern, this case is checked, and so on until the "square" visited all positions in the pattern, both for "vertical" and "horizontal" orientation of the pattern (if the pattern has the same number of rows and columns, then "vertical" and "horizontal" orientations are the same, i.e. only one orientation is checked). For a particular case, say when the current "square" is in position (2,4) in the grid pattern, the nodes in the grid pattern neighboring to the square are predicted using the estimated positions of the four vertices of the "square". Predicted node positions that are close to some estimated electrode centers are replaced by the estimated ones. Further, the new neighboring nodes in the grid pattern are predicted using the the positions of the "square" vertices and already predicted node positions, and so on until the whole pattern is filled. By doing this, the pattern tries to

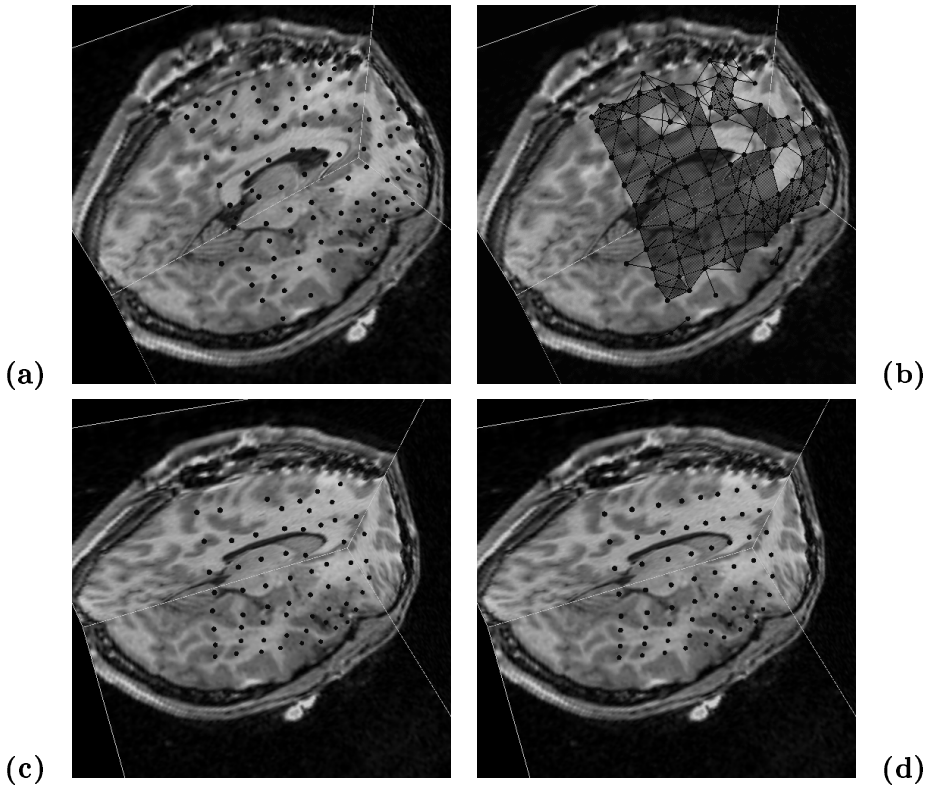


Figure 6.4: **Electrode Grid Extraction Steps.** Figure (a) shows a part of the estimated electrode centers in a postoperative MR scan (some of the estimated centers are occluded by the three orthogonal slices through the dataset). One can see a pattern of an 8 by 8 grid, but still the estimated centers are not regularly arranged, some are missing and there are false ones. Figure (b) represents the "low level topology" ("links" and "squares") as defined in the paper. Figure (c) shows the best match for an 8 by 8 grid pattern using the estimated electrode centers and "low level topology", while predicting the missing electrodes. The non-regularity of electrode positions is due to errors in estimated electrode center locations. It is significant in the top row of the grid, since the wires are coming out of the grid at that side causing increased artifacts. Figure (d) shows the electrode positions of the best match after regularization. One can see that the electrode positions form a more regular grid compared to the one in Figure (c).

follow the estimated centers, but if some are missing their positions are predicted. Once this process is finished, it is counted how many node positions, links and squares in the pattern match the estimated electrode positions, "links" and "squares" from the "low level topology". This sum is used as a measure to find the best pattern match. Thus, for each "square" the pattern is moved over all positions (and for both orientation), and the best match is determined. An example of the best pattern match is shown in Fig. 6.4(c).

6.2.3 Regularization and Surface Interpolation

Estimated electrode positions and predicted positions of missing electrodes have errors. However, these errors tend to cancel when a smooth surface of a known geometry (representing the grid) is fitted through the best pattern match. The reason why the errors tend to cancel is that there is no preferred direction for errors in estimated and predicted electrode centers. We suggest the following way to regularize the grid. If $\boldsymbol{r_1}$, $\boldsymbol{r_2}$, $\boldsymbol{r_3}$ and $\boldsymbol{r_4}$ are positions of four electrodes making a square in the best pattern match (the vertices of the square are connected as follows: 1-2-3-4-1), and

$$\begin{aligned}
\boldsymbol{x_t} &= \frac{\boldsymbol{r_4}-\boldsymbol{r_1}+\boldsymbol{r_3}-\boldsymbol{r_2}}{|\boldsymbol{r_4}-\boldsymbol{r_1}+\boldsymbol{r_3}-\boldsymbol{r_2}|}, & \boldsymbol{y_t} &= \frac{\boldsymbol{r_2}-\boldsymbol{r_1}+\boldsymbol{r_3}-\boldsymbol{r_4}}{|\boldsymbol{r_2}-\boldsymbol{r_1}+\boldsymbol{r_3}-\boldsymbol{r_4}|}, \\
\boldsymbol{n} &= \boldsymbol{x_t} \times \boldsymbol{y_t}, & \boldsymbol{c} &= \frac{\boldsymbol{r_1}+\boldsymbol{r_2}+\boldsymbol{r_3}+\boldsymbol{r_4}}{4}, \\
\boldsymbol{a} &= \frac{\boldsymbol{x_t}+\boldsymbol{y_t}}{|\boldsymbol{x_t}+\boldsymbol{y_t}|}, & \boldsymbol{d} &= \frac{\boldsymbol{a}\times\boldsymbol{n}}{|\boldsymbol{a}\times\boldsymbol{n}|}, \\
\hat{\boldsymbol{x}} &= \frac{\boldsymbol{a}+\boldsymbol{d}}{|\boldsymbol{a}+\boldsymbol{d}|}, & \hat{\boldsymbol{y}} &= \frac{\boldsymbol{a}-\boldsymbol{d}}{|\boldsymbol{a}-\boldsymbol{d}|},
\end{aligned}$$

then it is not difficult to see that $\boldsymbol{c}$ is the origin, and $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{y}}$ are orthogonal unit vectors of a planar coordinate system fitted through the four vertices and centered at the square center. Furthermore, $\hat{\boldsymbol{x}}$ is approximately in

direction 2-3 (and 1-4), while $\hat{\boldsymbol{y}}$ is approximately in the direction 1-2 (and 4-3). This is used to improve the vertex positions, knowing the inter-electrode distance (DISTANCE), as follows:

$$\begin{bmatrix} \boldsymbol{r_1} \\ \boldsymbol{r_2} \\ \boldsymbol{r_3} \\ \boldsymbol{r_4} \end{bmatrix}_{new} = \boldsymbol{c} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \frac{DISTANCE}{2} \begin{bmatrix} -1 & -1 \\ -1 & 1 \\ 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}} \\ \hat{\boldsymbol{y}} \end{bmatrix}.$$

This computation is done for each square in the grid. Since a vertex may be shared by two, three, or four squares, its new position is computed as the average of its new positions in each of the squares it belongs to. Let $\boldsymbol{\mathcal{R}}$ be a vector of all the electrode positions in the grid pattern. Combining previous equations for all the "squares" in the grid pattern, one can relate the electrode positions for two subsequent iterations, as $\boldsymbol{\mathcal{R}}_{k+1} = f(\boldsymbol{\mathcal{R}}_k)$, where k is the iteration index. The entire process is iteratively repeated until the electrodes assume steady state positions, i.e. until $\|\boldsymbol{\mathcal{R}}_{k+1} - \boldsymbol{\mathcal{R}}_k\|_\infty < \epsilon$ [3]. The procedure converges in practice. By doing this, the grid keeps its shape, while distances between electrodes are forced to come closer and closer to the ideal inter-electrode distance. An example of grid regularization is shown in Fig. 6.4(d).

Regular grids can now easily be interpolated by spline or some other smooth surfaces. However, since electrode grids are just slightly curved, each square is approximately flat, and practically there is no need to do better than a linear piece-wise interpolation. Due to the aforementioned reasons, this interpolation provides almost a smooth surface and it very

[3] Infinity norm of a vector is its maximum element (Definitions A.24 and A.26).

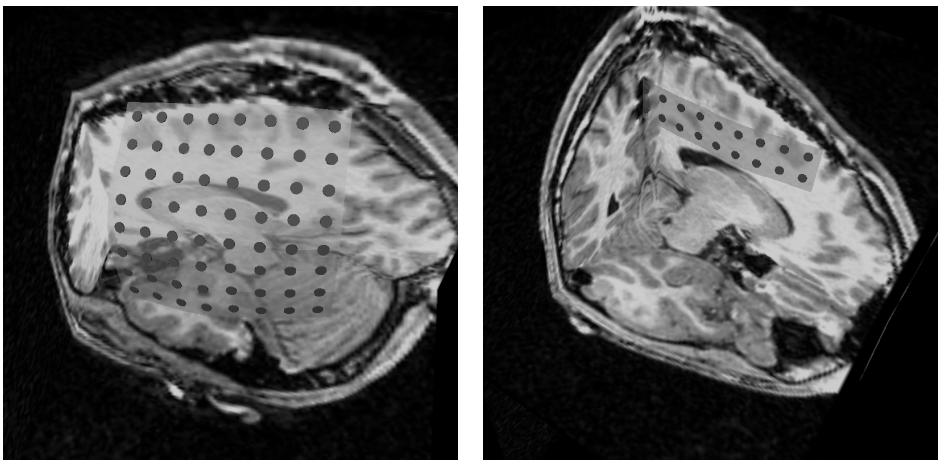


Figure 6.5: **Examples of Electrode Grid Extraction.** These two figures show examples of the final electrode grid, represented as a smooth surface with disk-shaped electrodes properly oriented. The electrodes can be colored to denote certain functional areas.

closely satisfies the invariance of (6.1). In addition, the surface can be extrapolated to model the grid-supporting material margins, and surface normals can be computed to properly orient disk-shaped electrodes. Two examples are shown in Fig. 6.5. A linear surface interpolation was used for these examples, while a spline one is described in Section 6.4.

6.2.4 Summary

We have compared the algorithm output to manually localized electrodes, concluding that the maximal error was within 1.5 mm, while the mean error was within 1 mm, which is sufficient for clinical applications. Table 6.2 presents the comparison results for 5 patients each having at least 64 electrodes (an 8 by 8 electrode grid), and some having 84 electrodes (an 8

subject	number of electrodes	mean [mm]	max [mm]
1	64	0.8	1.3
2	84	1.0	1.5
3	64	1.0	1.4
4	64	0.9	1.3
5	84	0.8	1.4

Table 6.2: **Automatically Determined Electrode Location Errors.** The table presents the errors of automatically determined electrode locations for 5 subjects. Each subject had at least 64 electrodes implanted. The automatically determined electrode locations were compared to the "true" electrode locations, which were determined manually.

by 8 and a 2 by 10 electrode grids) implanted.

The whole procedure takes a few minutes on an SGI Octane machine. The method, as currently implemented, works for rectangular grids of electrodes, but can easily be adjusted to work for non-rectangular grids and strips of electrodes as well.

This method is a step forward in electrode localization and visualization, especially when it is compared to what was used before: either manual electrode localization, or just mere looking at postoperative MR images.

6.3 Interactive Electrode Manipulation

In Section 6.2 we presented an automatic algorithm for extraction of implanted electrode grids form postoperative MR images. Although the algorithm has an acceptable accuracy (typical error of electrode location is

1 - 1.5 mm), due to big artifacts sometimes there is a need for interactive (manual) post-processing, particularly because this is a medical application, and the results should be reliable. Since electrode grids are not stretched or compressed during the implantation this is a property that one wants to preserve during the manipulation of the grid surface. E.g. the user would interactively move one of the electrodes on the grid and the whole grid would deform correspondingly so that the distances along the grid surface (also known as intrinsic surface distances) are preserved. In other words, by moving one electrode the user would indirectly move others (mainly from the electrode's neighborhood) in a physically correct way. Such surface deformation is known as locally isometric transformation, or local isometry. In Chapter 4 we developed a deformable surface model that allows only for locally isometric surface deformation, i.e. it preserves intrinsic surface distances. We use this model to allow the user to interactively manipulate electrode grid models that were generated by the automatic electrode localization algorithm. An example of interactive electrode strip manipulation is shown in Figure 6.6.

This tool allows clinicians to correct, if needed, the output of the automatic electrode grid localization algorithm, or to directly use it for interactive electrode grid positioning, without even using the automatic algorithm. While an interactive electrode grid manipulation tool is necessary as a backup, the model-based approach treats electrode grids as structures taking into account their physical properties. This renders electrode grid manipulation easier than the case when each electrode is manipulated independently.

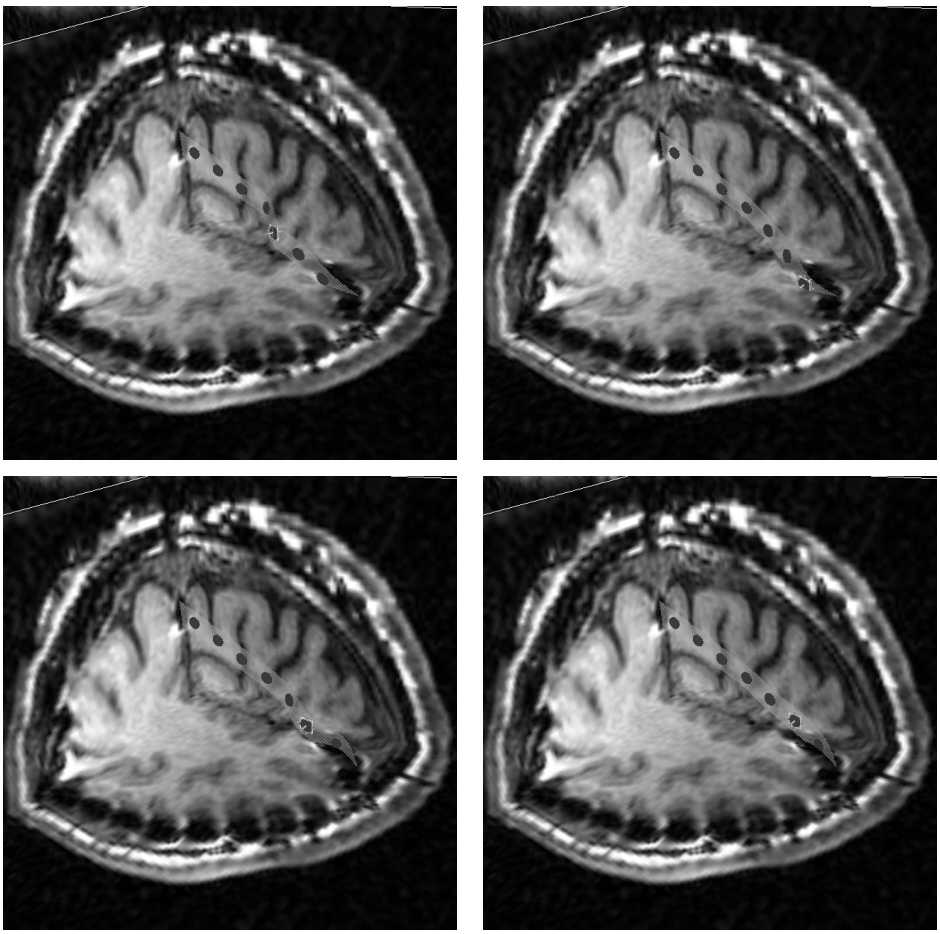


Figure 6.6: **Interactive Electrode Strip Manipulation.** These four figures demonstrate an electrode strip manipulation. The user selected an electrode from the strip and interactively moved it in the direction perpendicular to the electrode disk until the strip assumed an improved shape. Electrodes can be interactively moved in tangent directions, too. As the user moved the electrode, the rest of the strip moved in a physically correct way. Then the user selected another electrode and repeated the process until the electrode strip was properly positioned. Note that the strip was not stretched or compressed during the manipulation, i.e. that the distances along the strip were preserved.

6.4 Electrode Visualization

In order to properly visualize electrodes, one needs to draw the very disk shaped electrodes, as well as the supporting rubber material. Once this is done, it is easy to set the color of each electrode depending on the function it corresponds to. To display electrode disks, for each disk one needs to know its center, the direction of the disk axis, the disk radius and the disk height. The disk radius and height are known since the electrode specifications can be obtained from the electrode manufacturer; e.g. electrodes used at Yale New Haven Hospital (manufactured by [1]) have the following specifications: for electrode grids and strips, the electrode disks are 2 mm in radius, .5 mm in height, with 10 mm center to center inter-electrode distance in both directions, and with 4 mm supporting rubber material margins from the border electrode centers, while the depth electrodes are .5 mm in radius, 2.5 mm in height, with 6.5 mm center to center inter-electrode distance and 2 mm supporting rubber material margins.

Since electrode localization methods, both automatic and manual, output electrode centers, the only unknowns are the electrode disk orientations, i.e. the directions of the disk axes and the supporting rubber material surface points.

The way to determine the orientation of electrode disks and to generate the supporting rubber material surface differs for electrode grids, electrode strips, and depth electrodes, and the following three sections describe each case separately.

6.4.1 Visualization of Electrode Grids

The idea is to interpolate a surface through the electrode centers and then having the surface, to generate the supporting rubber material points (as points on the interpolated surface) and also compute the surface normals at each electrode position, obtaining the orientation of the electrode disks. The problem is that the surgeon sometimes cuts out one or more parts of the electrode grid before it is implanted, making holes in the grid surface or removing some of the border electrodes. This makes the surface interpolation problem more difficult, since a preferably C_1 surface should be fit through a non-rectangular set of points. Our approach is to first interpolate smooth curves along the rows and along the columns of the electrode grid, where it is possible (where the electrodes are removed, the interpolation curves are interrupted), and then generate surface from the interpolated curves. There are a few reasons for using this approach. First, it is computationally less complex compared to the interpolation of a polynomial surface through the electrode centers. We interpolate cubic splines along the rows and columns of the electrode grid. Let's say that the electrode grid has N by N electrodes. For each row or column the cubic spline has to be interpolated through N points, and this task is of complexity $O(N)$ (Appendix B). Since there are $2N$ lines (N rows and N columns), the complexity of interpolating the rows and columns is $O(N^2)$. We generate the interpolated surface from the row and column spline curves using Coons patches blended by cubic Hermit polynomials which provide a C_1 surface ([7], [20]). Having the curves available, there is no computation involved with Coons patches other than a direct evaluation. At the other hand, the interpolation of a polynomial surface through N by N electrodes involves inverting matrices of size N^2,

which is a task of $O(N^6)$ complexity, since the complexity of matrix inversion is $O(M^3)$, where M is the matrix size ([58]). When the surface is not cut, i.e. when the electrode centers make a rectangular set of points, they can be interpolated using tensor product interpolation, which has $O(N^4)$ complexity (section 15.12 of [20]). Even in this case, it is slower than our approach ($O(N^2)$ complexity). The speed is important when the electrode grids are interactively deformed by the user, and the surface interpolation has constantly to be recomputed. Another advantage of using spline curve interpolation through the rows and columns of the electrode grid, is that it can be directly applied to the case of electrode strips and depth electrodes, which have only one row, and naturally need a curve interpolation. Finally, since electrode grids are never very curved surfaces, even a simple linear interpolation achieves acceptable results as far as visualization of supporting material surface is concerned. We have decided to generate a C_1 surface to avoid "broken surface" artifacts (due to linearly interpolated C_0 surface) for cases when a grid of electrodes is significantly zoomed by the user. Figure 6.7 shows the steps in the visualization of a regular (not cut) electrode grid, while Figure 6.8 presents the case of a non-regular (cut) electrode grid of the same size. Once the surface is interpolated, we extrapolate it at borders, to generate the margins of the supporting rubber material. For the cut parts, when one or two quarters of a "square"[4] are missing (i.e. when one or two electrodes of the "square" are removed by the surgeon), we still use Coons Patches to generate (interpolate) the non-missing part

[4]A "square" is defined by the four neighboring electrodes forming a square. When one, two or three of the four electrodes are removed by the surgeon, one needs to generate (interpolate) only part of the "square" surface. When all four electrodes are removed, no "square" surface is displayed.

of the "square" surface, while when three quarters are missing, we use a transitional surface (section 21.3 of [20]) to generate (interpolate) the left quarter of the "square".

6.4.2 Visualization of Electrode Strips

In this case, since electrode strips have only one row of electrodes, we interpolate a cubic spline through the (row of) electrodes. In order to determine the orientation of the electrode disks, we use the estimated electrode orientations from the automatic localization of the electrodes or the exact electrode orientations from the manual manipulation of the electrode strip, whichever is the way the electrode strip was positioned. This is necessary, since in the case of electrode strips, one cannot fit a surface through one row of electrodes, and thus cannot compute the normal to the surface, and use it as the orientation of the electrode disks. Knowing the interpolated curve through the row of electrodes, and electrode orientations, at each electrode one can compute the vector orthogonal to the interpolated cubic spline and the electrode disk normal by a cross (also known as "vector") product. This vector and the tangent to the cubic spline are used as a basis of the tangent plane of the supporting rubber material surface at each electrode, and in between electrodes the basis is interpolated. Having the basis of the tangent plane of the material, one can generate its surface. Figure 6.9 shows the steps in the visualization of an electrode strip.

6.4.3 Visualization of Depth Electrodes

Depth electrodes, similarly as electrode strips, have one row of electrodes, and we interpolate a cubic spline through them. Having the interpolated

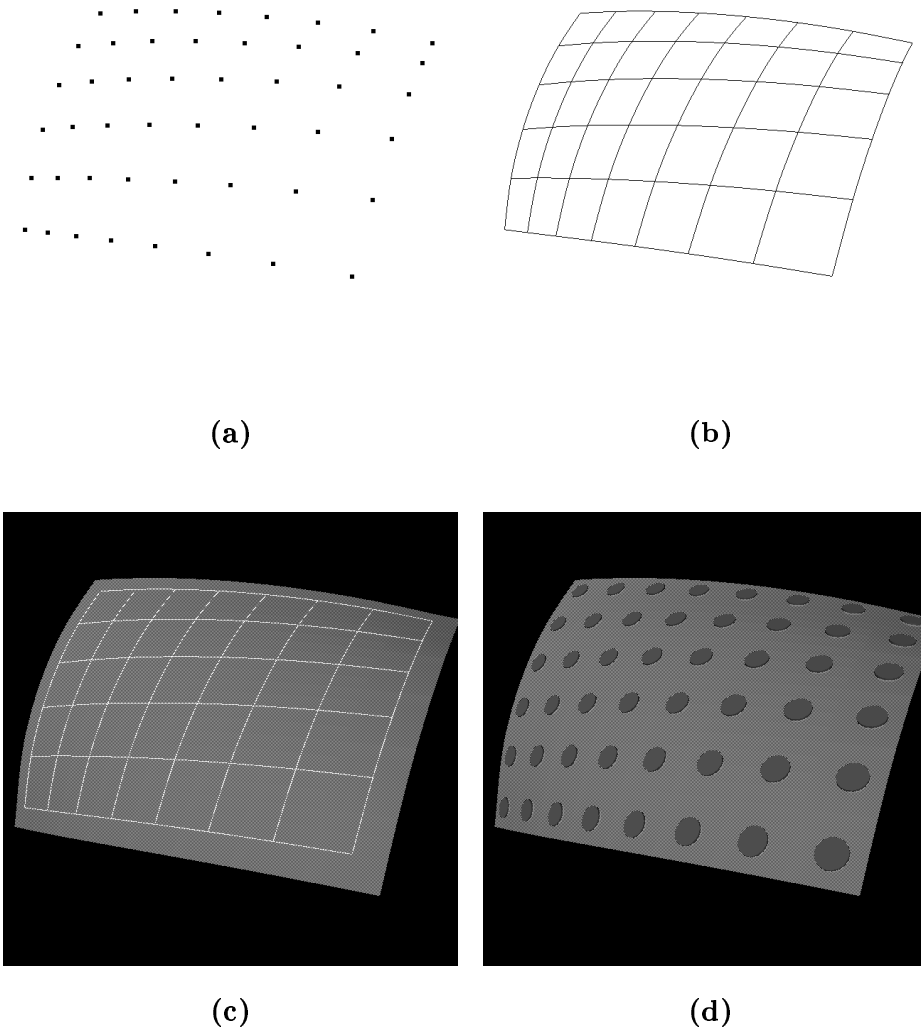


Figure 6.7: **Regular Electrode Grid Visualization.** **(a)** electrode centers output from the automatic or manual electrode localization, **(b)** the cubic spline curves interpolated through the rows and columns of the electrode grid, **(c)** the C_1 surface interpolated through the row and column curves using Coons patches blended by cubic Hermit polynomials, and **(d)** final display of the rubber supporting material surface and electrodes, which orientation is determined by the normal to the surface at their locations.

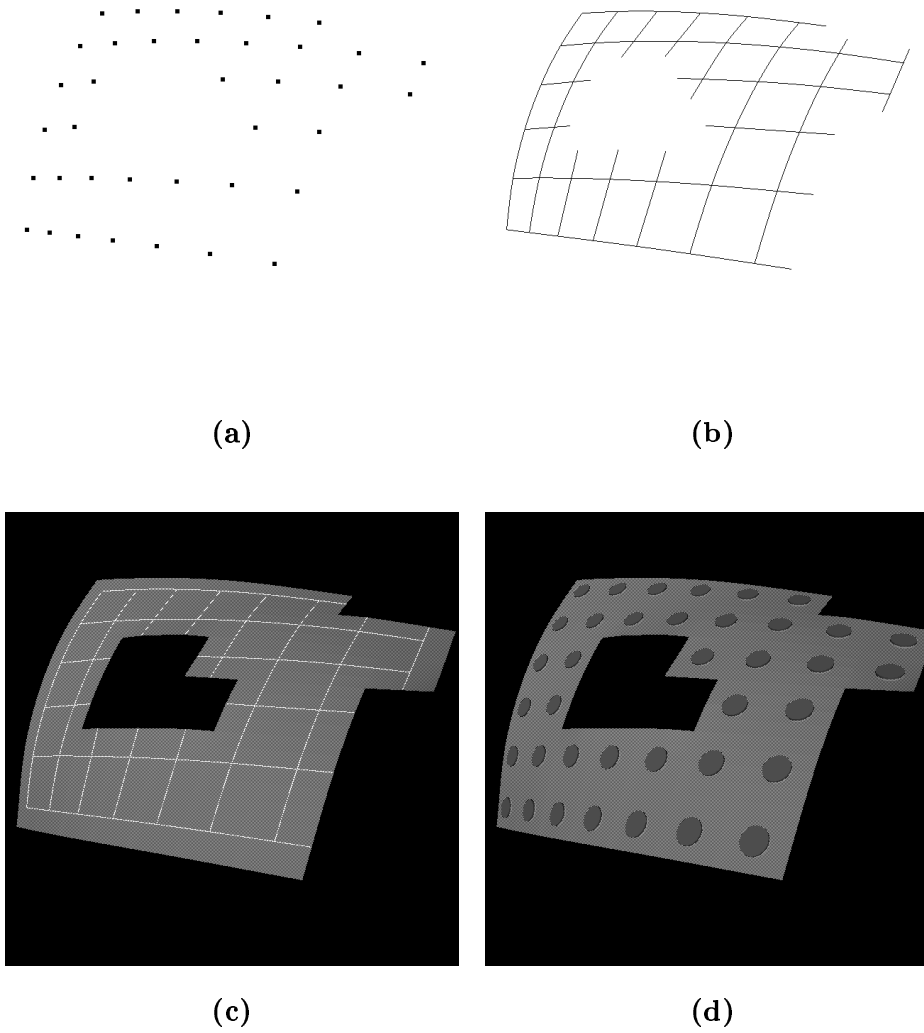


Figure 6.8: **Non-Regular Electrode Grid Visualization.** **(a)** electrode centers output from the automatic or manual electrode localization, **(b)** the cubic spline curves interpolated through the rows and columns of the electrode grid (the curves are interrupted where the electrodes are missing, **(c)** the C_1 surface interpolated through the row and column curves using Coons patches blended by cubic Hermit polynomials, and **(d)** final display of the rubber supporting material surface and electrodes, which orientation is determined by the normal to the surface at their locations.

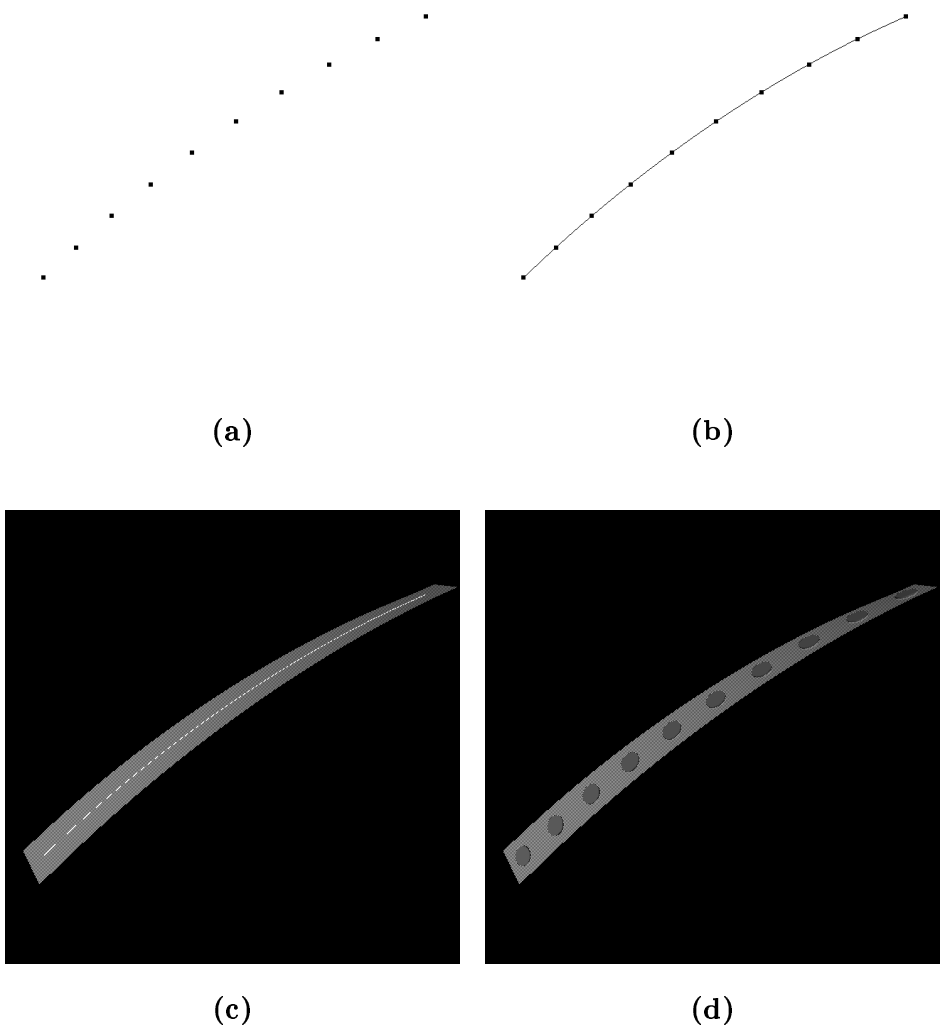


Figure 6.9: **Electrode Strip Visualization.** **(a)** electrode centers output from the automatic or manual electrode localization, **(b)** the cubic spline curve interpolated through the row of electrodes, **(c)** the interpolated C_1 surface using the interpolated cubic spline and provided electrode disk orientations, and **(d)** final display of the rubber supporting material surface and electrodes.

spline, it is straightforward to determine the orientation of electrode disks as the tangents of the cubic spline. The material surface is generated as a cylinder around the cubic spline. Figure 6.10 shows the steps in the visualization of a depth electrode.

There is a problem in visualization of depth electrodes due to the same radius of the very metal electrode disks and the cylindrical supporting rubber material. When the depth electrode is curved (and it is curved in practically all cases), the displayed material "cuts" the electrode disks hiding parts of them. This is illustrated in Figure 6.11 (a). To prevent this artifact from happening, we reduce the supporting rubber material cylinder radius a little bit, so that it does not "cut" the electrode disks, even when the depth electrode is maximally curved.

Figure 6.11 (b) shows the case of maximally curved depth electrode. Since depth electrodes, when implanted, are not very curved, we can set the minimal possible radius (r_{min}), corresponding to the maximal curvature[5]. Given the electrode disk radius ($r_e = d_e/2$) and the electrode disk height (h_e), it is not difficult to compute the maximal possible radius ($r_{max} = d_{max}/2$) of the supporting rubber material cylinder, so that it does not cut electrodes, even when the depth electrode is maximally curved. The expression is

$$r_{max} = r_{min} - \sqrt{(r_{min} - r_e)^2 + \frac{1}{4}h_e^2}.$$

Note in Figure 6.10 (d) that the cylindrical supporting material is of slightly smaller radius that the very electrodes (in reality the electrodes and the cylindrical material have the same radius), and that there are no

[5]For depth electrodes used at Yale New Haven Hospital, we set $r_{min} = 8.3\ mm$, which is a conservative figure, since depth electrodes are never that curved.

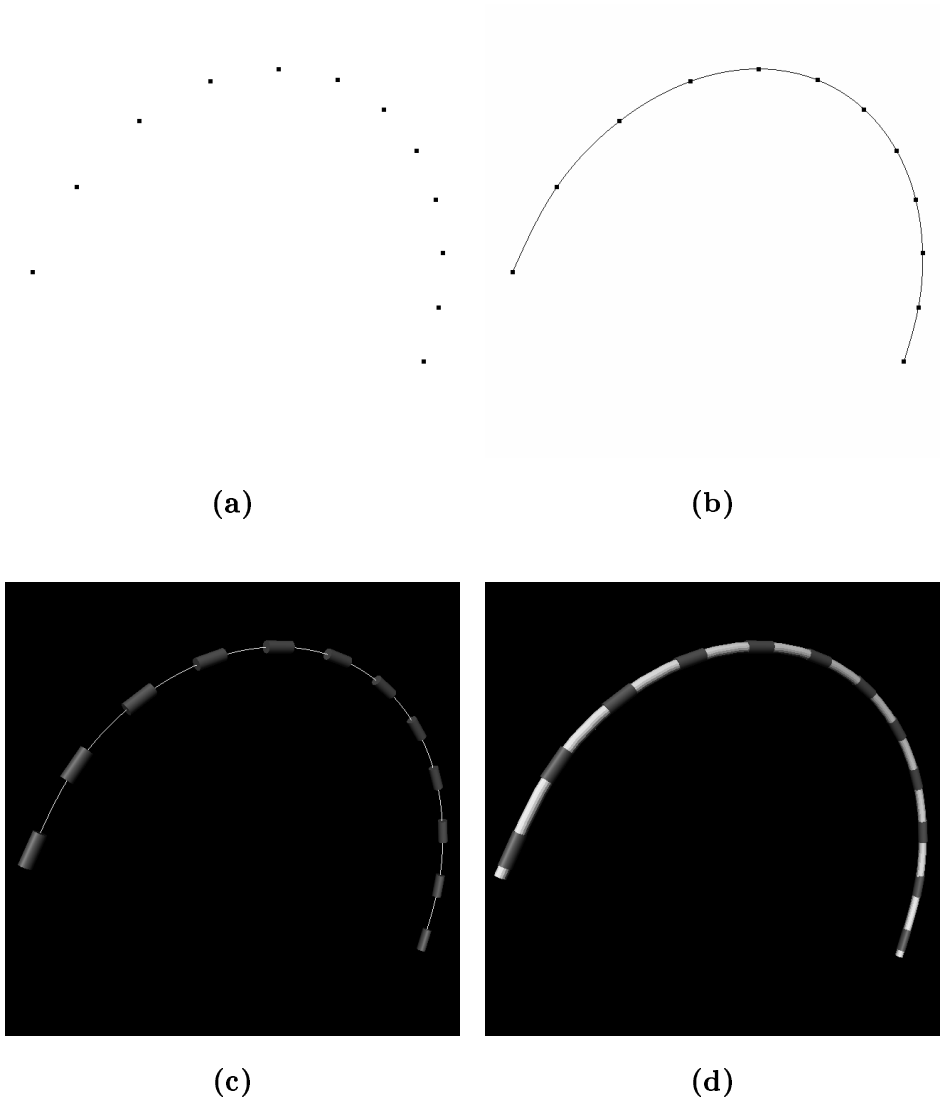


Figure 6.10: **Depth Electrode Visualization. (a)** electrode centers output from the automatic or manual electrode localization, **(b)** the cubic spline curve interpolated through the row of electrodes, **(c)** the electrode disks, which orientations are defined by the tangents of the cubic spline at the electrode centers, and **(d)** final display of the electrodes and the rubber supporting material surface generated as a cylinder around the cubic spline.

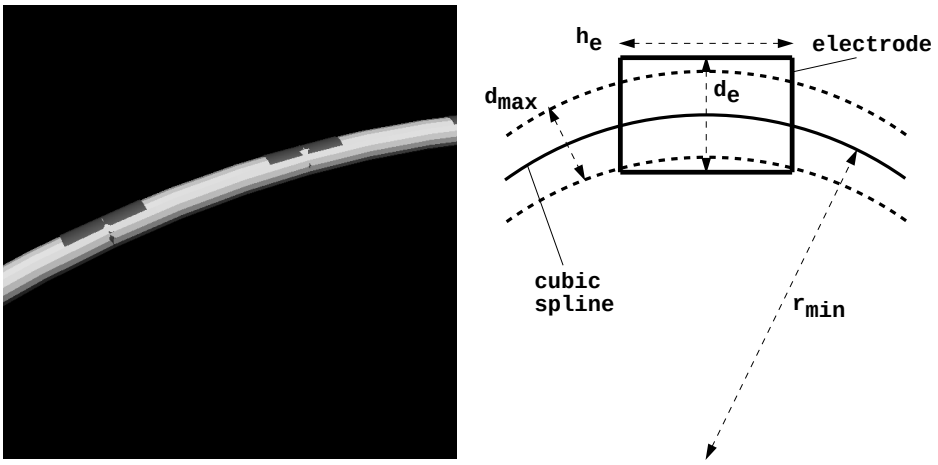


Figure 6.11: **Depth Electrode Visualization Artifacts. (a)** the cylindrical supporting rubber material cuts the electrodes, partially hiding them, **(b)** geometry of the maximally curved depth electrode, such that the cylindrical material still does not cut electrodes.

artifacts as ones in Figure 6.11 (a).

6.4.4 Examples of Localized Electrodes

To illustrate the usefulness of the presented tools, in Figure 6.12 we show a few examples of localized electrode grids from postoperative MR images. Such views allow for much easier analysis of spatial relations between functional locations and anatomical structures, compared to the use of non-processed postoperative MR images (e.g. see Figure 6.2).

6.5 Discussion

We addressed the problem of localization and visualization of subdural electrode grids, strips, and depth electrodes. While conventionally each electrode is independently manually localized in postoperative MR images, we suggested a combination of automatic and interactive steps for reliable electrode grid localization. The approach takes advantage of the fact that electrodes are grouped into grids (and strips and depth electrodes) connected by a supporting material. Electrode grids are never stretched or compressed when implanted, and this is built into the approach, which significantly helps electrode localization. Generated electrode grid models, shown together with anatomical and functional images, greatly enrich 3D displays, helping clinicians to better understand spatial relations between function and anatomy.

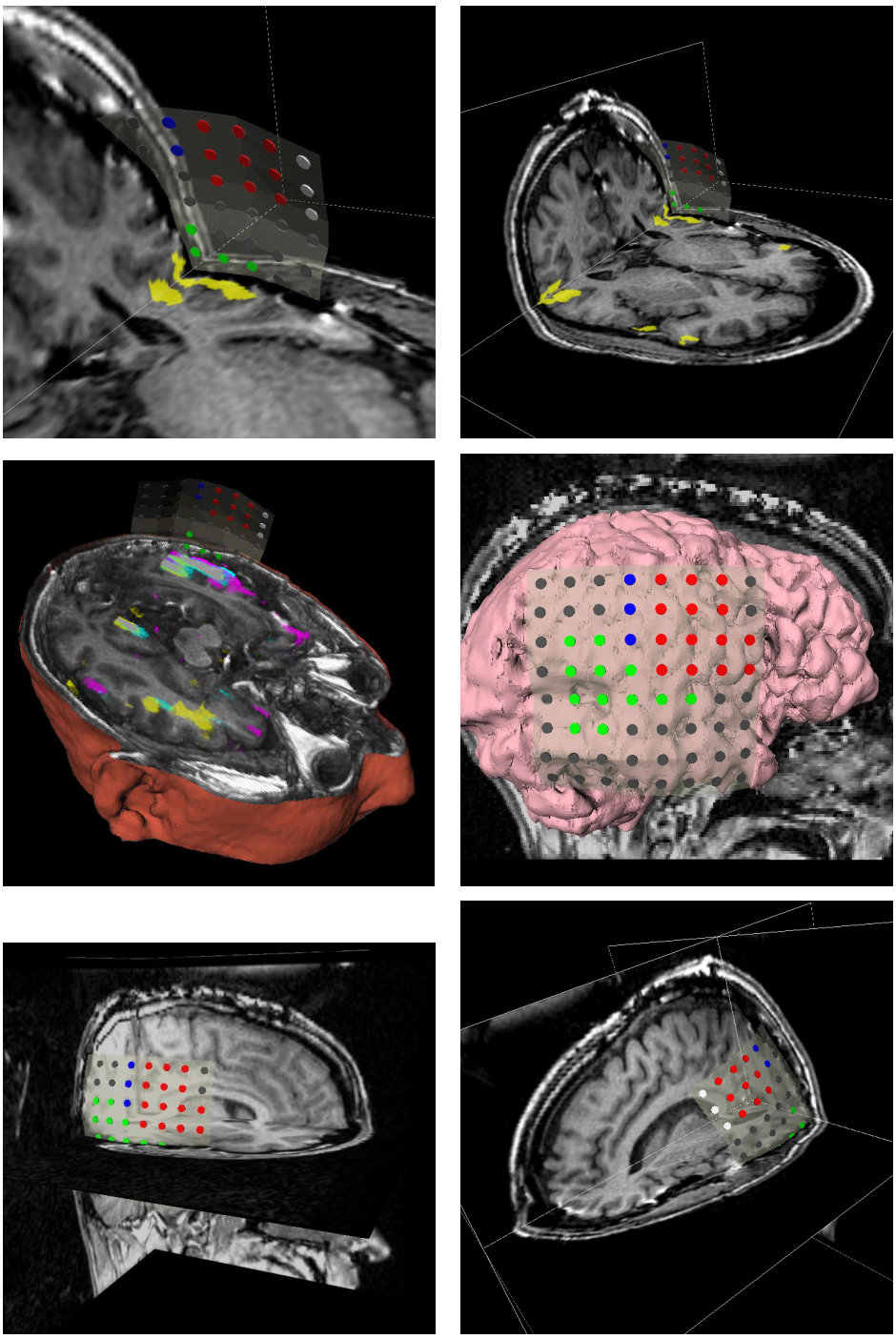


Figure 6.12: **Examples of Localized Electrode Grids.** Some of the electrodes are colored based on the brain function they correspond to. In the first three figures electrodes are displayed together with fMRI data.

Appendix A

An Overview of Mathematical Concepts

This appendix is an overview of mathematical concepts used throughout the thesis, providing their definitions as well as useful inter-relations. Although relevant literature is referenced in all the chapters, this overview makes the thesis self-contained. In addition, it introduces the terminology and notation used in the thesis. The material in the appendix is based on [3], [8], [9], [16], [27], [32], [39], [40], [63], and [64].

A.1 Algebra

A **set** is a collection of definite, well-defined objects (**set elements**) to form a whole. Sets are denoted by capital letters $(X, Y, ...)$, elements of sets are denoted by lowercase letters $(p, x, ...)$. If p is an element of X, it is denoted as $p \in X$, and $p \notin X$, if p is not an element of X.

Definition A.1 *A set Y is a* **subset** *of another set X (in symbols: $Y \subset X$)*

if every element $y \in Y$ is also an element of X.

Definition A.2 Difference *between sets X and Y (read X without Y) is*

$$X \backslash Y = \{x \mid x \in X \text{ and } x \notin Y\}.$$

Definition A.3 *The (***Cartesian***)* **product** *of sets X and Y is $X \times Y = \{(x, y) \mid x \in X, y \in Y\}$.*

The Cartesian product of set X with itself is denoted as $X^2 = X \times X$. Similarly, the definition can be generalized to the case of X^n, where n is a positive integer.

Definition A.4 *A subset f of the Cartesian product $X \times Y$ is called a* **function** *from X to Y (in symbols: $f : X \mapsto Y$) if from $(x, y_1), (x, y_2) \in f$ it follows that $y_1 = y_2$. The function is said to be single-valued.*

Definition A.5 *A function f is* **injective** *(also called* **one-to-one***) if $f(x_1) = f(x_2)$ always implies $x_1 = x_2$.*

Definition A.6 *A function $f : A \mapsto B$ is* **surjective** *(also called* **onto***) if for every $y \in B$ there is $x \in A$ such that $y = f(x)$.*

Definition A.7 *A function is* **bijective** *if it is injective and surjective.*

Definition A.8 *A (***binary***)* **operation** $*$ *on a set A is a function $* : A \times A \mapsto A$ (in symbols, the binary operation $*$ associates to any elements $x, y \in A$ an element $x * y \in A$).*

Definition A.9 *A pair $(A, *)$ of a set A and a binary operation $*$ on A is a* **group** *if the following four properties apply:*

1. *Closure:*

$$\forall x, y \in A,\ x * y \in A.$$

2. *Associative law:*

$$\forall x, y, z \in A,\ x * (y * z) = (x * y) * z.$$

3. *Existence of a neutral (identity) element* $e \in A$*:*

$$\forall x \in A,\ x * e = e * x = x.$$

4. *Existence of an inverse element* $x^{-1} \in A$ *for each* $x \in A$*:*

$$x * x^{-1} = x^{-1} * x = e.$$

Definition A.10 *A group* $(A, *)$ *is called* **Abelian** *(or* **commutative***) if the commutative law holds, i.e.:*

$$\forall x, y \in A,\ x * y = y * x.$$

Definition A.11 *A triple* $(A, *, \circ)$ *is a* **field** *if the following applies:*

1. $(A, *)$ *is an Abelian group with the neutral element zero (*0*; zero element).*

2. $(A \backslash \{0\}, \circ)$ *is an Abelian group. Its neutral element is often referred to as one (*1*, unit element).*

3. *Distributive law:*

$$\forall x, y, z \in A,\ x \circ (y * z) = (x \circ y) * (x \circ z)\,.$$

A.2 Vector Analysis

Definition A.12 *A (***linear***)* **vector space** *is a structure* $(M, (K, +, \cdot), \oplus, \odot)$, *where* M *is a set of so-called vectors,* $(K, +, \cdot)$ *is a field (often called scalar field),* $\oplus$ *is a binary operation on* M*, and* $\odot$ *is an external operation (often called scalar multiplication),* $\odot : K \times M \mapsto M$*, if the following applies for all* $a, b \in K$ *and* $\boldsymbol{x}, \boldsymbol{y}$[1] $\in M$*:*

1. $(M, \oplus)$ *is an Abelian group.*

2. *Associative law:*

$$a \odot (b \odot \boldsymbol{x}) = (a \cdot b) \odot \boldsymbol{x}.$$

3. *(First) distributive law:*

$$(a + b) \odot \boldsymbol{x} = (a \odot \boldsymbol{x}) \oplus (b \odot \boldsymbol{x}).$$

4. *(Second) distributive law:*

$$a \odot (\boldsymbol{x} \oplus \boldsymbol{y}) = (a \odot \boldsymbol{x}) \oplus (a \odot \boldsymbol{y}).$$

5.

$$\forall \boldsymbol{x} \in M,\ 1 \odot \boldsymbol{x} = \boldsymbol{x}.$$

Although a vector space is the whole structure $(M, (K, +, \cdot), \oplus, \odot)$, often just set M is referred to as a vector space. Using Definition A.12 one can show many useful results, e.g. $0 \odot \boldsymbol{x} = \mathbf{0}$ or $-1 \odot \boldsymbol{x} = -\boldsymbol{x}$, where 0 is the neutral element of $(K, +)$ group, 1 is the neutral (unit) element of $(K \backslash \{0\}, \cdot)$ group, -1 is the inverse element of 1 in $(K, +)$ group, $\mathbf{0}$ is the neutral element of $(M, \oplus)$ group, and $-\boldsymbol{x}$ is the inverse element of $\boldsymbol{x}$ in $(M, \oplus)$ group.

[1] Vectors are denoted by lowercase bold letters ($\boldsymbol{x}$, $\boldsymbol{y}$, ...)

It will be assumed that $\odot$ has precedence over $\oplus$. In order to simplify notation, $\odot$ will be omitted, and $\oplus$ will be replaced by $+$.

Definition A.13 *A nonempty subset W of a vector space V is called a* **subspace** *of V if W is itself a vector space with respect to the operations defined in V.*

Theorem A.1 *A nonempty subset W of a vector space V is a subspace if and only if $\mathbf{0} \in W$ and*

$$\forall \boldsymbol{x}, \boldsymbol{y} \in W \text{ and } \forall \alpha, \beta \in K,\ \alpha\boldsymbol{x} + \beta\boldsymbol{y} \in W.$$

Definition A.14 *Let V be a vector space and K associated (scalar) field. For an integer $n \geq 2$, $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n} \in V$, and $\alpha_1, \alpha_2, \ldots, \alpha_n \in K$, the expression*

$$\alpha_1\boldsymbol{x_1} + \alpha_2\boldsymbol{x_2} + \ldots + \alpha_n\boldsymbol{x_n}$$

is called a **linear combination** *of vectors $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}$.*

Definition A.15 *The* **span** *of vectors $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}$ is set*

$$S = \{\alpha_1\boldsymbol{x_1} + \alpha_2\boldsymbol{x_2} + \ldots + \alpha_n\boldsymbol{x_n} \mid \alpha_1, \alpha_2, \ldots, \alpha_n \in K\}.$$

It is said that S is **spanned** *or generated by those n vectors.*

Theorem A.2 *The span of vectors $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}$ is a vector space.*

Definition A.16 *A set of vectors $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}$ is* **linearly independent** *if $\alpha_1\boldsymbol{x_1} + \alpha_2\boldsymbol{x_2} + \ldots + \alpha_n\boldsymbol{x_n} = \mathbf{0}$ only if $\alpha_1 = \alpha_2 = \ldots = \alpha_n = 0$. Otherwise, the vectors are* **linearly dependent**.

Definition A.17 *A set of linearly independent vectors of a vector space that span the vector space is a* **basis** *of the vector space. The elements of a basis are called* **basis vectors**.

Theorem A.3 *All bases of a vector space have the same number of elements.*

Theorem A.3 leads to the definition of the dimension of a vector space.

Definition A.18 *The* **dimension** *of a vector space is the number of elements of a basis of the vector space.*

Let R denote the set of real numbers, and R^n the set of all n-tuples of real numbers.

Theorem A.4 *Set R^n is a vector space of dimension n ($n \geq 1$).*

Theorem A.5 *All real $m \times n$ matrices (with m and n fixed) form a vector space of dimension mn.*

Definition A.19 *Let V and W be vector spaces with the same scalar field K. A function $T : V \mapsto W$ is a* **linear transformation** *if*

$$\forall \boldsymbol{x_1}, \boldsymbol{x_2} \in V, \text{ and } \forall \alpha_1, \alpha_2 \in K, \; T(\alpha_1 \boldsymbol{x_1} + \alpha_2 \boldsymbol{x_2}) = \alpha_1 T(\boldsymbol{x_1}) + \alpha_2 T(\boldsymbol{x_2}).$$

Definition A.20 *An* **isomorphism** *is a bijective linear transformation of vector spaces.*

Theorem A.6 *All vector spaces of dimension n are isomorphic to R^n.*

Definition A.21 *An isomorphism from a vector space V to R^n is called a* **co-ordinate system** *for V.*

Definition A.22 *Let $D \subset R^n$. A* **scalar field** *on D is a function $f : D \mapsto R$.*

Definition A.23 *Let $D \subset R^n$. A* **vector field** *on D is a function $\boldsymbol{f} : D \mapsto R^n$.*

Definition A.24 *Let V be a vector space. A* **vector norm** *is a function $f : V \mapsto R$ (standard notation: $f(\boldsymbol{x}) = \|\boldsymbol{x}\|$) that satisfies the following properties:*

1. $\|\boldsymbol{x}\| \geq 0, \ \forall \boldsymbol{x} \in V$.
2. $\|\boldsymbol{x}\| = 0 \ \Leftrightarrow \ \boldsymbol{x} = \boldsymbol{0}$.
3. $\|\lambda \boldsymbol{x}\| = |\lambda| \|\boldsymbol{x}\|, \ \forall \boldsymbol{x} \in V, \ \forall \lambda \in R$.
4. $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|, \ \forall \boldsymbol{x}, \boldsymbol{y} \in V$ *(triangle inequality).*

Definition A.25 *A* **unit vector** *with respect to the norm $\|\cdot\|$ is a vector $\boldsymbol{x}$ that satisfies $\|\boldsymbol{x}\| = 1$.*

Definition A.26 *A* **p-norm** *(or L_p norm) for space R^n is defined by*

$$\|\boldsymbol{x}\|_p = \left(|x_1|^p + |x_2|^p + \cdots + |x_n|^p\right)^{1/p},$$

where $p \geq 1$. In practice, most commonly used cases are $p = 1$ or 2, and a third norm, $\|\boldsymbol{x}\|_\infty$ (the latter as defined below), that is,

1. $\|\boldsymbol{x}\|_1 = |x_1| + |x_2| + \cdots + |x_n|$ *("L_1 norm").*

2. $\|\boldsymbol{x}\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$ *("Euclidean norm" or "L_2 norm").*

3. $\|\boldsymbol{x}\|_\infty = \lim_{p\to\infty} \|\boldsymbol{x}\|_p = \max_k |x_k|$. *("$L_\infty$ norm").*

Definition A.27 *Let $U : R^n \mapsto R$ be a scalar field. The vector field*

$$\boldsymbol{\nabla} U = \left(\frac{\partial U}{\partial x_1}, \frac{\partial U}{\partial x_2}, \cdots, \frac{\partial U}{\partial x_n}\right)$$

is the **gradient** *of U.*

Definition A.28 *Let $\boldsymbol{F} : R^n \mapsto R^n$ be a vector field. The scalar field*

$$div\boldsymbol{F} = \frac{\partial F_1}{\partial x_1} + \frac{\partial F_2}{\partial x_2} + \cdots + \frac{\partial F_n}{\partial x_n}$$

is the **divergence** *of $\boldsymbol{F}$, where $\boldsymbol{F} = (F_1, F_2, \ldots, F_n)$.*

Definition A.29 *Let $U : R^n \mapsto R$ be a scalar field. The scalar field*

$$\Delta U = \frac{\partial^2 U}{\partial x_1^2} + \frac{\partial^2 U}{\partial x_2^2} + \cdots + \frac{\partial^2 U}{\partial x_n^2}$$

is the **Laplacian** *of U.*

The formulas in Definitions A.27, A.28, and A.29 are in the Cartesian coordinates, and it is assumed that the partial derivatives exist.

A.3 Functional Analysis

Definition A.30 *A* **metric** *or* **distance** *on a non-empty set X is a function $d : X^2 \mapsto R$, for which the following applies for all $x, y, z \in X$,*

1. $d(x, y) \geq 0$,

2. $d(x, y) = 0 \;\Leftrightarrow\; x = y$,

3. $d(x, y) = d(y, x)$ *(symmetry),*

4. $d(x, y) \leq d(x, z) + d(z, y)$ *(triangle inequality).*

Definition A.31 *A* **metric space** *is a pair (X, d), where X is an arbitrary non-empty set (called the underlying set) and d is a metric on X.*

Definition A.32 *The* **Euclidean** *or L_2* **metric** *on R^n is defined by*

$$L_2(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}.$$

Definition A.33 *The* **Euclidean space** *is the metric space (R^n, L_2).*

Definition A.34 *The L_p metric ($p \geq 1$) on R^n is defined by*

$$L_p(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{k=1}^{n} |x_k - y_k|^p \right)^{1/p}.$$

In addition to L_2 metric, common are L_1 and L_∞ metrics (defined below).

Definition A.35 *The L_∞ metric on R^n is defined by*

$$L_\infty(\boldsymbol{x}, \boldsymbol{y}) = \lim_{p \to \infty} L_p(\boldsymbol{x}, \boldsymbol{y}) = \max_k |x_k - y_k|.$$

Definition A.36 *Let (X_1, d_1) and (X_2, d_2) be two metric spaces. A function $f : X_1 \mapsto X_2$* **preserves distances** *if*

$$d_2\left(f(x_1), f(x_2)\right) = d_1(x_1, x_2)$$

for all $x_1, x_2 \in X_1$. An **isometry** *is a bijective function that preserves distances. If there is an isometry $f : X_1 \mapsto X_2$ then the metric spaces (X_1, d_1) and (X_2, d_2) are* **isometric**.

Definition A.37 *The* **open ball** *with center c and radius $r > 0$ in a metric space (X, d) is defined as*

$$B_r(c) = \{x \in X \mid d(x, c) < r\}.$$

It is said that $B_r(c)$ is an open ball around c.

Definition A.38 *A non-empty set $U \subset X$ is an* **open set** *in a metric space (X, d) if for each $x \in U$ there is $r > 0$ such that $B_r(x) \subset U$.*

Definition A.39 *A set $V \subset X$ is a* **neighborhood** *of x in space X, if there is an open set U such that $x \in U \subset V$.*

Definition A.40 *Let X and Y be two metric spaces. A function $f : U \subset X \mapsto Y$ is* **continuous** *at $x \in U$ if given $\epsilon > 0$, there exists a $\delta > 0$ such that*

$$f(B_\delta(p)) \subset B_\epsilon(f(p)).$$

Function f is said to be continuous in U if f is continuous for all $p \in U$.

Definition A.41 *Let X and Y be two metric spaces. A function $f : X \mapsto Y$ is a* **homeomorphism** *if the following applies*

1. *f is bijective,*
2. *f is continuous,*
3. *f^{-1} is continuous.*

If there is a homeomorphism $f : X \mapsto Y$, two metric spaces X and Y are said to be **homeomorphic** *or* **topologically equivalent**.

A.4 Differential Geometry

Definition A.42 *A* **parametrized differentiable curve** *is a differentiable function $\alpha : I \mapsto R^3$ of an open interval $I = (a, b)$ of R into R^3. The set $\alpha(I) \subset R^3$ is called the trace of α.*

Definition A.43 *Let $\alpha : I \mapsto R^3$ be a parametrized differentiable curve. The vector $\alpha'(t)$ is called the* **tangent vector** *of the curve α at t.*

Definition A.44 *A parametrized differentiable curve $\alpha : I \mapsto R^3$ is said to be* **regular** *if $\alpha'(t) \neq 0$ for all $t \in I$. A point t where $\alpha'(t) = 0$ is called a* **singular point** *of the curve α.*

Definition A.45 *Let $F : U \subset R^n \mapsto R^m$ be a differentiable function. Each $p \in U$ is associated with a linear function $dF_p : R^n \mapsto R^m$ which is called the* **differential** *of F at p and is defined as follows. Let $w \in R^n$ and let $\alpha : (-\epsilon, \epsilon) \mapsto U$ be a differentiable curve such that $\alpha(0) = p$ and $\alpha'(0) = w$. By the chain rule, the curve $\beta = F \circ \alpha : (-\epsilon, \epsilon) \mapsto R^m$ is also differentiable. Then*

$$dF_p(w) = \beta'(0).$$

It can be shown that the above definition of $dF_p(w)$ does not depend on the choice of the curve which passes through p with tangent vector w, and that dF_p is, in fact, a linear function.

Definition A.46 *A subset $S \subset R^3$ is a* **regular surface** *if, for each $p \in S$, there exists a neighborhood V in R^3 and a function $\boldsymbol{x} : U \mapsto V \cap S$ of an open set $U \subset R^2$ onto $V \cap S \subset R^3$ such that*

1. $\boldsymbol{x}$ *is differentiable. This means that if the function is written as*

$$\boldsymbol{x}(u,v) = (x(u,v), y(u,v), z(u,v)) \,, \;\; (u,v) \in U$$

the functions $x(u,v)$, $y(u,v)$, *and* $z(u,v)$ *have continuous partial derivatives of all orders in* U.

2. $\boldsymbol{x}$ *is a homeomorphism. Since* $\boldsymbol{x}$ *is continuous by condition 1, this means that* $\boldsymbol{x}$ *has an inverse* $\boldsymbol{x}^{-1} : V \cap S \mapsto U$ *which is continuous; that is,* $\boldsymbol{x}^{-1}$ *is the restriction of a continuous function* $\boldsymbol{F} : W \subset R^3 \mapsto R^2$ *defined on an open set* W *containing* $V \cap S$.

3. *(The regularity condition.) For each* $q \in U$, *the differential* $d\boldsymbol{x}_q : R^2 \mapsto R^3$ *is one-to-one.*

The function $\boldsymbol{x}$ is called a **parametrization** or a system of (local) coordinates in (a neighborhood of) p. The neighborhood $V \cap S$ of p in S is called a **coordinate neighborhood**.

Definition A.47 *A* **parametrized surface** $\boldsymbol{x} : U \subset R^2 \mapsto R^3$ *is a differentiable function* $\boldsymbol{x}$ *from an open set* $U \subset R^2$ *into* R^3. *The set* $\boldsymbol{x}(U) \subset R^3$ *is called the* **trace** *of* $\boldsymbol{x}$. $\boldsymbol{x}$ *is regular if the differential* $d\boldsymbol{x}_q : R^2 \mapsto R^3$ *is one-to-one for all* $q \in U$ *(i.e. the vectors* $\partial\boldsymbol{x}/\partial u$, $\partial\boldsymbol{x}/\partial v$ *are linearly independent for all* $q \in U$*). A point* $p \in U$ *where* $d\boldsymbol{x}_q$ *is not one-to-one is called a* **singular point** *of* $\boldsymbol{x}$.

Note that a parametrized surface, even when regular, may have self-intersections in its trace.

Theorem A.7 *Let* $\boldsymbol{x} : U \subset R^2 \mapsto R^3$ *be a regular parametrized surface and let* $q \in U$. *Then there exists a neighborhood* V *of* q *in* R^2 *such that* $\boldsymbol{x}(V) \subset R^3$ *is a regular surface.*

Definition A.48 *A* **tangent vector** *to a regular surface S, at a point $p \in S$, is the tangent vector $\alpha'(0)$ of a differentiable parametrized curve $\alpha : (-\epsilon, \epsilon) \mapsto S$ with $\alpha(0) = p$.*

Theorem A.8 *Let $\boldsymbol{x} : U \mapsto S$ be a parametrization of a regular surface S and let $q \in U$. The vector subspace of dimension 2,*

$$d\boldsymbol{x}_q(R^2) \subset R^3,$$

coincides with the set of tangent vectors to S at $\boldsymbol{x}(q)$.

The plane $d\boldsymbol{x}_q(R^2)$, which passes through $\boldsymbol{x}(q) = p$, does not depend on the parametrization $\boldsymbol{x}$. This plane is called the **tangent plane** to S at p and will be denoted by $T_p(S)$. The choice of the parametrization $\boldsymbol{x}$ determines a basis $\{(\partial \boldsymbol{x}/\partial u)(q), (\partial \boldsymbol{x}/\partial v)(q)\}$ of $T_p(S)$, called the basis associated to $\boldsymbol{x}$. Sometimes it is convenient to write $\partial \boldsymbol{x}/\partial u = \boldsymbol{x}_u$ and $\partial \boldsymbol{x}/\partial v = \boldsymbol{x}_v$.

Definition A.49 *Given a point p on a regular surface S, there are two unit vectors of R^3 that are normal to the tangent plane $T_p(S)$; each of them is called a* **unit normal vector** *at p.*

Definition A.50 *Let S and $\bar{S}$ be two regular surfaces. A function $f : V \mapsto \bar{S}$ of a neighborhood V of $p \in S$ is a* **local isometry** *at p if there exists a neighborhood $\bar{V}$ of $f(p) \in \bar{S}$ such that $f : V \mapsto \bar{V}$ is an isometry. If there exists a local isometry into $\bar{S}$ at every $p \in S$, the surface S is said to be* **locally isometric** *to $\bar{S}$. S and $\bar{S}$ are locally isometric if S is locally isometric to $\bar{S}$ and $\bar{S}$ is locally isometric to S.*

Appendix B

Remarks on Cubic Spline Interpolation

Given a set of N function values (x_i, $i = 1, \ldots, N$), and their corresponding parameter values (t_i, $i = 1, \ldots, N$), one can compute a cubic polynomial

$$x(t) = at^3 + bt^2 + ct + d, \tag{B.1}$$

for each interval $[t_i, t_{i+1}]$, $i = 1, \ldots, N-1$, such that the piecewise cubic polynomial on the $[t_1, t_N]$ interval is continuous in second derivative at the interval boundaries; this is known as a cubic spline. One way of determining the coefficients of the cubic polynomial for each interval is to compute the values of the second derivative at the interval boundaries (Section 3.3 of [58]). This leads to a system of linear tridiagonal equations, which can be solved in $O(N)$ time ([58]). If the spline is uniform[1] (which is reasonable to use in the case of electrode center interpolation, since electrodes are equidistantly

[1] In uniform parametric interpolation the parameter is increased by one for each succeeding point in the sequence.

spaced in electrode grids, strips and depth electrodes), one can assume that for each interval the parameter t goes from 0 to 1, and in this case, the coefficients of the form B.1 can be computed as follows:

$$a = \frac{x_2'' - x_1''}{6},\ b = \frac{x_1''}{2},\ c = x_2 - x_1 - \frac{x_1''}{3} - \frac{x_2''}{6},\ d = x_1, \tag{B.2}$$

where x_1 and x_2 are the function values at the beginning ($t = 0$) and at the end ($t = 1$) of the interval, and x_1'' and x_2'' are the corresponding values of the second derivative. To summarize, knowing the function values and the values of the corresponding second derivatives (which can be computed as explained in Section 3.3 of [58]), one can compute the cubic polynomial coefficients using B.2 and then interpolate the function using B.1. The form B.1, which involves 5 multiplications and 3 additions, can be rearranged into the following form,

$$x(t) = ((at + b)t + c)t + d, \tag{B.3}$$

which has 3 multiplications and 3 additions, and is thus computationally more efficient. The form B.3, generalized to the case of N-th order polynomial, is known as *Horner's rule* (Section 3.7 of [7]).

Bibliography

[1] Ad-Tech Medical Instrument Corporation, Racine, Wisconsin. *Medical Supplies: http://www.adtechmedical.com/.*

[2] Y. S. Akgul and C. Kambhamettu. Recovery and tracking of continuous 3d surfaces from stereo data using a deformable dual-mesh. In *International Conference on Computer Vision*, Corfu, Greece, September 1999.

[3] M. A. Armstrong. *Basic Topology.* Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1990.

[4] K.S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, September 1987.

[5] Michel A. Audette, Kaleem Siddiqi, and Terry M. Peters. Level-set surface segmentation and fast cortical range image tracking for computing intrasurgical deformations. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 788–797, Cambridge, UK, September 1999.

[6] G. H. Barnett, D. W. Roberts, and R. J. Maciumas. *Image-Guided Neurosurgery: Clinical Applications of Surgical Navigation.* Quality Medical Publishing, first edition, 1998.

[7] Robert C. Beach. *Curves and Surfaces of Computer-Aided Design.* Van Nostrand Reinhold, New York, 1991.

[8] A. I. Borisenko and I. E. Tarapov. *Vector and Tensor Analysis with Applications.* Dover Books on Mathematics. Dover Publications, New York, 1979.

[9] I. N. Bronshtein and K. A. Semendyayev. *Handbook of Mathematics.* Springer-Verlag, Berlin, third edition, 1998.

[10] Richard D. Bucholz, David D. Yeh, Jason Trobaugh, et al. The correction of stereotactic inaccuracy caused by brain shift using an intraoperative ultrasound device. In *First Joint Conference, Computer Vision, Virtual Reality and Robotics in Medicine and Medical Robotics and Computer-Assisted Surgery (CVRMed-MRCAS'97)*, pages 459–466, Grenoble, France, March 1997.

[11] Alexandra Chabrerie, Fatma Ozlen, Shin Nakajima, et al. Three-dimensional reconstruction and surgical navigation in pediatric epilepsy surgery. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 74–83, Cambridge, MA, October 1998.

[12] H. Chui, J. Rambo, R. Schultz, et al. Registration of cortical anatomical structures via robust 3d point matching. In *Information Processing in Medical Imaging (IPMI)*, pages 168–181, Visegrad, Hungary, June/July 1999.

[13] M. H. A. Claessens. *Finite Element Modeling of the Human Head Under Impact Conditions.* PhD thesis, Eindhoven University of Technology, 1997.

[14] R. Courant and D. Hilbert. *Methods of Mathematical Physics.* John Wiley & Sons, 1994.

[15] P. Dierckx. *Curve and Surface Fitting with Splines.* Oxford University Press, 1995.

[16] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, Englewood Cliffs, New Jersey, 1976.

[17] N. L. Dorward, O. Alberti, B. Velani, et al. Early clinical experience with the easyguide neuronavigation system and measurement of intraoperative brain distortion. In *Minimally Invasive Techniques for Neurosurgery*, pages 193–196, 1997.

[18] P. J. Edwards, D. L. G. Hill, J. A. Little, et al. Deformation for image guided interventions using a three component tissue model. In *Information Processing in Medical Imaging (IPMI)*, pages 218–231, Poultney, Vermont, USA, June 1997.

[19] G. A. Evans, J. M. Blackledge, and P. D. Yardley. *Numerical Methods for Partial Differential Equations.* Springer-Verlag, New York, first edition, 2000.

[20] Gerald Farin. *Curves and Surfaces for Computer-Aided Geometric Design - A Practical Guide.* Computer Science and Scientific Computing. Academic Press, San Diego, fourth edition, 1997.

[21] M. Ferrant, S. K. Warfield, A. Nabavi, et al. Registration of 3d intra-operative mr images of the brain using a finite element biomechanical model. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 19–28, Pittsburgh, PA, USA, October 2000.

[22] Matthieu Ferrant. *Physics-based Deformable Modeling of Volumes and Surfaces for Medical Image Registration, Segmentation and Visualization.* PhD thesis, Université catholique de Louvain, April 2001.

[23] Y. C. Fung. *A First Course in Continuum Mechanics.* Prentice Hall, Englewood Cliffs, NJ, third edition, 1994.

[24] D. T. Gering, A. Nabavi, R. Kikinis, et al. An integrated visualization system for surgical planning and guidance using image fusion and interventional imaging. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 809–819, Cambridge, UK, September 1999.

[25] S. Gibson. Constrained elastic surface nets: Generating smooth surfaces from binary segmented data. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 888–898, Cambridge, MA, October 1998.

[26] D. G. Gobbi, R. M. Comeau, and T. M. Peters. Ultrasound probe tracking for real-time ultrasound/mri overlay and visualization of brain shift. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 920–927, Cambridge, UK, September 1999.

[27] Gene H. Golub and Charles F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, Maryland, second edition, 1989.

[28] W. E. L. Grimson, G. J. Ettinger, S. J. White, et al. Evaluating and validating an automated registration system for enhanced reality visualization in surgery. In *First International Conference on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed'95)*, pages 3–12, Nice, France, April 1995.

[29] W. E. L. Grimson, G. J. Ettinger, S. J. White, et al. An automatic registration method for frameless stereotaxy, image guided surgery, and enhanced reality visualization. *IEEE Transactions on Medical Imaging*, 15(2):129–140, April 1996.

[30] A. C. C. Guyton. *Basic Neuroscience: Anatomy and Physiology.* Harcourt, second edition, 1991.

[31] A. Hagemann. *A Biomechanical Model of the Human Head with Variable Material Properties for Intraoperative Image Correction.* PhD thesis, University of Hamburg, Logos Verlag Berlin, 2001.

[32] John W. Harris and Horst Stocker. *Handbook of Mathematics and Computational Science.* Springer - Verlag, New York, 1998.

[33] N. Hata, A. Nabavi, S. Warfield, et al. A volumetric optical flow method for measurement of brain deformation from intraoperative magnetic resonance images. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 928–935, Cambridge, UK, September 1999.

[34] D. L. G. Hill, C. R. Maurer, A. J. Martin, et al. Assessment of intraoperative brain deformation using interventional mr imaging. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 910–919, Cambridge, UK, September 1999.

[35] Derek L. G. Hill, Calvin R. Maurer Jr., Matthew Y. Wang, et al. Estimation of intraoperative brain surface movement. In *First Joint Conference, Computer Vision, Virtual Reality and Robotics in Medicine and Medical Robotics and Computer-Assisted Surgery (CVRMed-MRCAS'97)*, pages 449–458, Grenoble, France, March 1997.

[36] M. Holden, D. L. G. Hill, E. R. E. Denton, et al. Voxel similarity measures for 3-d serial mr brain image registration. *IEEE Transactions on Medical Imaging*, 19(2):94–102, February 2000.

[37] K. H. Huebner. *The Finite Element Method for Engineers.* John Wiley & Sons, third edition, 1994.

[38] K. Kansy, P. Wisskirchen, U. Behrens, et al. Localite - a frameless neuronavigation system for interventional magnetic resonance imaging systems. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 832–841, Cambridge, UK, September 1999.

[39] Ilija Kovačević and Nebojša Ralević. *Functional Analysis (in Serbo-Croatian).* Stylos, FTN, Novi Sad, Yugoslavia, 1997.

[40] Erwin Kreyszig. *Advanced Engineering Mathematics.* John Wiley & Sons, New York, sixth edition, 1988.

[41] S. K. Kyriacou and C. Davatzikos. A biomechanical model of soft tissue deformation, with application to non-rigid registration of brain images

with tumor pathology. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 531–538, Cambridge, MA, October 1998.

[42] Yuencheng Lee, Demetri Terzopoulos, and Keith Waters. Realistic modeling for facial animation. In *SIGGRAPH*, pages 55–62, Los Angeles, CA, August 1995.

[43] V. D. Liseikin. *Grid Generation Methods*. Springer-Verlag, Berlin, 1999.

[44] R. K. Livesley. *Finite elements: an introduction for engineers*. Cambridge University Press, Cambridge, UK, 1983.

[45] Lee Markosian, Jonathan M. Cohen, Thomas Crulli, and John Hughes. Skin: A constructive approach to modeling free-form shapes. In *SIGGRAPH*, pages 393–400, Los Angeles, CA, August 1999.

[46] W. Maurel, N. M. Thalmann, D. Thalmann, and Y. Wu. *Biomechanical Models for Soft Tissue Simulation*. Springer-Verlag, New York, first edition, 1998.

[47] C. R. Maurer, D. L. G. Hill, R. J. Maciunas, et al. Measurement of intraoperative brain surface deformation under a craniotomy. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 51–62, Cambridge, MA, October 1998.

[48] C. R. Maurer, D. L. G. Hill, A. J. Martin, et al. Measurement of intraoperative brain deformation using a 1.5 tesla interventional mr system: Preliminary results. *IEEE Transactions on Medical Imaging*, 17(5):817–825, October 1998.

[49] M. Miga, K. Paulsen, F. Kennedy, et al. Initial in-vivo analysis of 3d heterogeneous brain computations for model-updated image-guided neurosurgery. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 743–752, Cambridge, MA, October 1998.

[50] M. Miga, K. Paulsen, F. Kennedy, et al. Model-updated image-guided neurosurgery using the finite element method: Incorporation of the falx cerebri. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 900–909, Cambridge, UK, September 1999.

[51] M. Miga, A. Staubert, K. Paulsen, et al. Model-updated image guided neurosurgery: Preliminary analysis using intraoperative mr. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 115–124, Pittsburgh, PA, USA, October 2000.

[52] K. Miller and K. Chinzei. Constitutive modelling of brain tissue: Experiment and theory. *Journal of Biomechanics*, 30(11/12):1115–1121, November 1997.

[53] A. Nabavi, P. McL. Black, D. T. Gering, et al. Serial intraoperative mr imaging of brain shift. *Neurosurgery*, 48(4):787–798, April 2001.

[54] Ohio Medical Instrument Co., Inc., Cincinnati, Ohio. *Operation of the Mayfield^R Acciss^{TM} Stereotactic Workstation*, March 1997. OMI^R Surgical Products.

[55] Barrett O'Neill. *Elementary Differential Geometry.* Academic Press, 1997.

[56] M. R. Pamidi and S. H. Advani. Nonlinear constitutive relations for human brain tissue. *Journal of Biomechanical Engineering*, 100:44–48, February 1978.

[57] T. Peters, B. Davey, P. Munger, et al. Three-dimensional multimodal image-guidance for neurosurgery. *IEEE Transactions on Medical Imaging*, 15(2):121–128, April 1996.

[58] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C - The Art of Scientific Computing*. Cambridge University Press, Cambridge, UK, second edition, 1992.

[59] M. H. T. Reinges, G. Krombach, H. Nguyen, et al. Assessment of intraoperative brain tissue movements by frameless neuronavigation. In *Computer Aided Surgery*, page 2:218, 1997.

[60] Richard A. Robb. *Biomedical Imaging, Visualization, and Analysis*. John Wiley & Sons, first edition, 1999.

[61] D. W. Roberts, A. Hartov, F. E. Kennedy, et al. Intraoperative brain shift and deformation: A quantitative analysis of cortical displacement in 28 cases. *Neurosurgery*, 43(4):749–760, October 1998.

[62] Rik Stokking. *Integrated Visualization of Functional and Anatomical Brain Images*. PhD thesis, University Utrecht, 1998.

[63] Gilbert Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, Wellesley, Massachusetts, 1986.

[64] Gilbert Strang. *Linear Algebra and its Applications*. Harcourt Brace Jovanovich College Publishers, Orlando, third edition, 1988.

[65] C. Studholme, D. J. Hawkes, and D. L. G. Hill. A normalised entropy measure of 3d medical image alignment. In *SPIE Medical Imaging*, San Diego, CA, February 1998.

[66] C. Studholme, D. L. G. Hill, and D. J. Hawkes. Automated 3-d registration of mr and ct images of the head. *Medical Image Analysis*, 1(2):163–175, June 1996.

[67] C. Studholme, D. L. G. Hill, and D. J. Hawkes. An overlap invariant entropy measure of 3d medical image alignment. *Pattern Recognition*, 32(1):71–86, January 1999.

[68] M. E. Taylor. *Partial Differential Equations: Qualitative Studies of Linear PDE*. Springer-Verlag, New York, first edition, 1996.

[69] J. F. Thompson, N. P. Weatherrill, and B. Soni. *Handbook of Grid Generation*. CRC Press, first edition, 1998.

[70] S. P. Timoshenko and J. N. Goodier. *Theory of Elasticity*. McGraw-Hill, third edition, 1990.

[71] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, Upper Saddle River, NJ, 1998.

[72] Somasundaram Valliappan. *Continuum Mechanics Fundamentals*. A.A. Balkema, Rotterdam, 1981.

[73] Oskar Škrinjar, Alexandre Carpentier, Todd Constable, Colin Studholme, and James Duncan. A platform for visualization of anatomical, functional and sub-dural cortical stimulation data. In *Eighth Scientific Meeting and Exhibition*, Denver, Colorado, April 2000. International Society for Magnetic Resonance in Medicine (ISMRM).

[74] Oskar Škrinjar and James Duncan. Automatic electrode grid extraction from distorted post-op mr scans. In *Seventh Scientific Meeting and Exhibition*, page 2216, Philadelphia, PA, May 1999. International Society for Magnetic Resonance in Medicine (ISMRM).

[75] Oskar Škrinjar and James Duncan. Automatic extraction of implanted electrode grids. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 990–997, Cambridge, UK, September 1999.

[76] Oskar Škrinjar and James Duncan. Real time 3d brain shift compensation. In *Information Processing in Medical Imaging (IPMI)*, pages 42–55, Visegrad, Hungary, June/July 1999.

[77] Oskar Škrinjar and James Duncan. An automatic algorithm for skin surface extraction from mr scans. In *Eighth Scientific Meeting and Exhibition*, Denver, Colorado, April 2000. International Society for Magnetic Resonance in Medicine (ISMRM).

[78] Oskar Škrinjar and James Duncan. Preserving intrinsic surface distances - application to electrode grid manipulation. In *Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA)*, pages 54–60, Hilton Head Island, SC, June 2000. IEEE Computer Society.

[79] Oskar Škrinjar, Arya Nabavi, and James Duncan. A stereo-guided biomechanical model for volumetric deformation analysis. In *Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA)*, pages 95–102, Kauai, Hawaii, December 2001. IEEE Computer Society.

[80] Oskar Škrinjar, Dennis Spencer, and James Duncan. Brain shift modeling for use in neurosurgery. In *Medical Image Computing and Computer Aided Intervention (MICCAI)*, pages 641–649, Cambridge, MA, October 1998.

[81] Oskar Škrinjar, Colin Studholme, Arya Nabavi, et al. Steps toward a stereo-camera-guided biomechanical model for brain shift compensation. In *Information Processing in Medical Imaging (IPMI)*, pages 183–189, Davis, CA, June 2001.

[82] Oskar Škrinjar, Hemant Tagare, and James Duncan. Surface growing from stereo images. In *Computer Vision and Pattern Recognition (CVPR)*, pages pp. II: 571–576, Hilton Head Island, SC, June 2000. IEEE Computer Society.

[83] O. C. Zienkiewicz. *The Finite Element Method*. McGraw-Hill, London, third edition, 1977.

[84] D. Zorin, P. Schroder, and W. Sweldens. Interactive multiresolution mesh editing. In *SIGGRAPH*, pages 259–268, Los Angeles, CA, August 1997.

Index