

2009

## Enabling technology for non-rigid registration during image-guided neurosurgery

andriy Yuri Fedorov

*College of William & Mary - Arts & Sciences*

Follow this and additional works at: <https://scholarworks.wm.edu/etd>



Part of the [Computer Sciences Commons](#)

---

### Recommended Citation

Fedorov, andriy Yuri, "Enabling technology for non-rigid registration during image-guided neurosurgery" (2009). *Dissertations, Theses, and Masters Projects*. Paper 1539623540.

<https://dx.doi.org/doi:10.21220/s2-ggjp-hn81>

This Dissertation is brought to you for free and open access by the Theses, Dissertations, & Master Projects at W&M ScholarWorks. It has been accepted for inclusion in Dissertations, Theses, and Masters Projects by an authorized administrator of W&M ScholarWorks. For more information, please contact [scholarworks@wm.edu](mailto:scholarworks@wm.edu).

ENABLING TECHNOLOGY FOR NON-RIGID REGISTRATION DURING  
IMAGE-GUIDED NEUROSURGERY

Andriy Yuri Fedorov

Berdyansk, Ukraine

Master of Science, College of William and Mary, 2003

Bachelor of Science, Ternopil Academy of National Economy, Ukraine, 2001

A Dissertation presented to the Graduate Faculty  
of the College of William and Mary in Candidacy for the Degree of  
Doctor of Philosophy

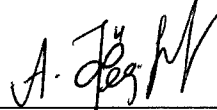
Department of Computer Science

The College of William and Mary  
May 2009

## APPROVAL PAGE

This Dissertation is submitted in partial fulfillment of  
the requirements for the degree of


Doctor of Philosophy



---

Andriy Yuri Fedorov

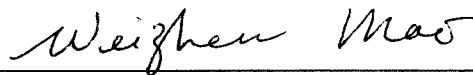
Approved by the Committee, April 2009



---

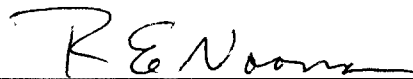
Committee Chair

Professor Nikos Chrisochoides, Computer Science  
College of William and Mary



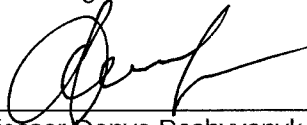
---

Associate Professor Weizhen Mao, Computer Science  
College of William and Mary



---

Professor Robert Noonan, Computer Science  
College of William and Mary



---

Assistant Professor Denys Poshyvanyk, Computer Science  
College of William and Mary



---

Professor Ron Kikinis, Radiology  
Harvard Medical School

## ABSTRACT PAGE

In the context of image processing, non-rigid registration is an operation that attempts to align two or more images using spatially varying transformations. Non-rigid registration finds application in medical image processing to account for the deformations in the soft tissues of the imaged organs. During image-guided neurosurgery, non-rigid registration has the potential to assist in locating critical brain structures and improve identification of the tumor boundary. Robust non-rigid registration methods combine estimation of tissue displacement based on image intensities with the spatial regularization using biomechanical models of brain deformation. In practice, the use of such registration methods during neurosurgery is complicated by a number of issues: construction of the biomechanical model used in the registration from the image data, high computational demands of the application, and difficulties in assessing the registration results. In this dissertation we develop methods and tools that address some of these challenges, and provide components essential for the intra-operative application of a previously validated physics-based non-rigid registration method.

First, we study the problem of image-to-mesh conversion, which is required for constructing biomechanical model of the brain used during registration. We develop and analyze a number of methods suitable for solving this problem, and evaluate them using application-specific quantitative metrics. Second, we develop a high-performance implementation of the non-rigid registration algorithm and study the use of geographically distributed Grid resources for speculative registration computations. Using the high-performance implementation running on the remote computing resources we are able to deliver the results of registration within the time constraints of the neurosurgery. Finally, we present a method that estimates local alignment error between the two images of the same subject. We assess the utility of this method using multiple sources of ground truth to evaluate its potential to support speculative computations of non-rigid registration.

# Table of Contents

Acknowledgments	vi
List of Tables	vii
List of Figures	viii
<b>1 Introduction</b>	<b>2</b>
1.1 Contributions of this Work . . . . .	5
1.2 Thesis Structure . . . . .	7
<b>2 Image Registration</b>	<b>8</b>
2.1 Background . . . . .	8
2.2 Robust Estimation of Volumetric Deformation . . . . .	13
2.2.1 Pre-operative Processing . . . . .	16
2.2.2 Intra-operative Processing . . . . .	18
2.3 Requirements . . . . .	24
<b>3 Mesh Generation</b>	<b>26</b>
3.1 Image-to-Mesh Conversion . . . . .	30
3.2 Application Requirements . . . . .	33
3.3 Related Work . . . . .	37
3.4 Open Source Mesh Generation Tools . . . . .	45
3.4.1 Piecewise-Linear Surface Recovery . . . . .	45
3.4.2 Delaunay Meshing . . . . .	49
3.4.3 Advancing Front Meshing . . . . .	52
3.4.4 Lattice-Based Meshing . . . . .	54
3.5 Construction of Sizing Function . . . . .	63
3.6 Evaluation Framework . . . . .	66
3.7 Results . . . . .	68
3.7.1 Evaluation of Lattice-Based Meshing . . . . .	68
3.7.2 Application-Specific Evaluation . . . . .	71
3.8 Summary . . . . .	81
3.9 Contributions . . . . .	83

<b>4</b>	<b>High-Performance Image Registration</b>	<b>85</b>
4.1	Cluster-based Implementation . . . . .	88
4.1.1	Related Work . . . . .	88
4.1.2	Motivation . . . . .	91
4.1.3	Implementation . . . . .	97
4.1.4	Evaluation . . . . .	102
4.1.5	Discussion . . . . .	107
4.2	Grid-Enabled Infrastructure . . . . .	108
4.2.1	Motivation . . . . .	108
4.2.2	Grid Computing for Medical Applications . . . . .	115
4.2.3	Implementation . . . . .	118
4.2.4	Evaluation . . . . .	124
4.2.5	Discussion . . . . .	130
4.3	Summary . . . . .	132
4.4	Contributions . . . . .	134
<b>5</b>	<b>Registration Accuracy Assessment</b>	<b>136</b>
5.1	Related Work . . . . .	139
5.2	Robust Error Estimation Methodology . . . . .	144
5.2.1	Image Pre-Processing . . . . .	144
5.2.2	Edge Detection . . . . .	146
5.2.3	Development of the Robust Local Hausdorff Distance Metric . . . . .	148
5.3	Evaluation . . . . .	152
5.3.1	Synthetic Ground Truth . . . . .	153
5.3.2	Physically-Realistic Simulated Ground Truth . . . . .	157
5.3.3	Observer-Derived Ground Truth . . . . .	159
5.4	Summary . . . . .	161
5.5	Contributions . . . . .	163
<b>6</b>	<b>Discussion and Future Work</b>	<b>164</b>
<b>A</b>	<b>Software Tools for Image-to-Mesh Conversion</b>	<b>168</b>
A.1	Triangulated Surface Recovery . . . . .	169
A.2	Delaunay Mesh Generation and Refinement from Binary Image . . . . .	170
A.3	Lattice-Based Image Meshing . . . . .	171
A.4	Background Mesh Construction . . . . .	172
A.5	Utilities . . . . .	172
<b>B</b>	<b>Execution of Non-Rigid Registration on TeraGrid</b>	<b>174</b>
<b>C</b>	<b>Software Tools for Local Accuracy Alignment Estimation</b>	<b>180</b>
	<b>Bibliography</b>	<b>184</b>
	<b>Vita</b>	<b>200</b>

## ACKNOWLEDGMENTS

Graduate school has been a life-changing experience for me. Thinking back, I would perish on this “journey to Ithaca” without my family, friends, colleagues and just good people I met during these years. First of all, I thank Nikos Chrisochoides for being my guide and mentor. It is his support, advice and wisdom that often helped me to stay afloat. I thank Allen and Meg Tucker, who gave me this most unique opportunity to study in the USA – I hope I made some progress! I thank all the professors whose classes I took for educating me, and for setting high standard of excellence in teaching. Steve Pieper, Simon K. Warfield and Ron Kikinis introduced me to the world of medical image computing and have been very patient answering all my questions.

This research would not be possible without the collaboration with my co-authors, and especially without the efforts of Andriy Kot and Eric Billet. I am thankful to Olivier Clatz who provided me with the motivating application, and helped a lot with understanding his implementation of the registration algorithm. Andrey Chernikov and Barbara Monteith reviewed this manuscript and provided helpful comments. I thank all of my dissertation committee for their insightful suggestions and help in improving this work.

Through the years of graduate school, I was supported in part by grants from NSF (ACI-0085969, ANI-0203974, CNS-0312980, CNS-0521381, EIA-9982853, CSI-0719929 and CCF-0833081), John Simon Guggenheim Memorial Foundation, Howard Hughes Medical Institute and Brain Science Foundation. This research was supported by an allocation through the TeraGrid Advanced Support Program. This work was performed [in part] using computational facilities at the College of William and Mary which were enabled by grants from Sun Microsystems, the National Science Foundation, and Virginia’s Commonwealth Technology Research Fund. These facilities would be useless to me without help from Tom Crockett, the true Wizard of SciClone.

I acknowledge outstanding staff and personnel of the university, in particular Vanessa Godwin, Jacquelyn Johnson and Stephen Sechrist, who understand the challenges of being an international student, and who are always ready to do more to help. I thank Professor Noonan and his family for their friendly support during my first weeks on this continent. Special thanks go to my good friends, officemates and roommates: Ken Blakely, Alexey Matveev, Andrey Chernikov, Andriy Kot, Elena Mingazova, Denis Kenzior, Natasha Sokolova, Chaman Singh Verma, Irena Zaburanna, George Zagaris, Patrick Vora and his parents, Melissa McDowell, Ray Plante, Besam Khidhir, Darrell Craig, Pam Sardeson and many many others. I will not forget the good times we had together, and all the things you did to help me in so many ways. I will miss you. I thank Sensei Steve Pohle and Sensei Ali Mohajerani for guiding me in my first step down yet another way. I am honored and lucky to be your humble student.

There are not enough words to describe my love and gratitude to my parents, my grandmother and all of my family for bringing me up, and for believing in me. I wish we could be closer together all these long years of my graduate school – I missed you immensely. Tania – your love and your support made this possible.

# List of Tables

2.1	Parameter space for the non-rigid registration method of Clatz et al. [53].	18
3.1	Surface approximation quality.	70
3.2	Element shape quality evaluation.	71
3.3	Element shape quality evaluation following deformation.	72
3.4	Registration points, true outliers, and their coverage by the mesh surfaces.	77
3.5	Application-specific metrics for the evaluated meshes.	78
4.1	Execution times of block matching with different parameters.	94
4.2	Execution time (seconds) of the intra-surgery part of the restructured implementation at various stages of development.	104
4.3	Execution time (seconds) of block matching with static work distribution on SciClone cluster.	106
4.4	Execution time (seconds) of block matching with dynamic work distribution on SciClone cluster.	106
4.5	Landmark error before registration (mm)	111
4.6	Absolute improvement in registration accuracy as a function of changing parameters.	111
4.7	Configurations of the utilized TeraGrid resources and corresponding processing times for the time-critical NRR workflow components.	126
5.1	Outliers percentage for synthetic tumor growth data.	158
5.2	Accuracy of error assessment for the tumor resection data.	161
A.1	Software dependencies of I2MTK tools.	168
A.2	Command line flags for triangulated surface recovery from binary image input.	169
A.3	Command line flags for adaptive Delaunay meshing from binary image input.	170
A.4	Command line flags for adaptive lattice-based meshing from binary image input.	171
A.5	Command line flags for adaptive lattice-based meshing from binary image input.	173
C.1	Parameters for the configuration file used by the accuracy assessment tool.	182
C.2	Parameters for the configuration file used by the accuracy assessment tool (continued).	183



# List of Figures

1.1	Intra-operative brain shift and recovered deformation field. . . . .	3
2.1	Processing steps during non-rigid image registration for IGNS. . . . .	14
3.1	Mesh vertex cell complex and registration points. . . . .	34
3.2	A hierarchy of the studied approaches to mesh generation. . . . .	37
3.3	Surface mesh recovery with Marching Cubes. . . . .	39
3.4	Geometric closeness and surface normal approximation. . . . .	46
3.5	Classification of good- and ill-shaped tetrahedral elements. . . . .	51
3.6	Subdivision templates used in <i>RGM</i> meshing approach. . . . .	56
3.7	Tetrahedral mesh at different stages of RGM meshing. . . . .	59
3.8	Signed distance transform example. . . . .	60
3.9	Synthetic deformation field construction and example. . . . .	67
3.10	Cross-cuts of the compared meshes adapted to the registration point distribution. . . . .	76
3.11	Histograms of the number of registration points assigned to mesh vertices for the analyzed meshes. . . . .	78
3.12	Boxplots comparing registration point distributions for the compared meshes for three registration cases. . . . .	79
3.13	Visualization of the mesh quality metrics related to element skewness. . .	79
4.1	Execution time breakdown for the original implementation of NRR on a high-end 4-way workstation. . . . .	94
4.2	Timeline of the processing steps with the restructured NRR implementation.	98
4.3	Scalability of the time-critical part of the registration code. . . . .	103
4.4	Impact of the parameter changes on the accuracy of registration at selected landmarks. . . . .	112
4.5	Landmark error as a function of the block size and outlier rejection rate. .	112
4.6	NRR workflow diagram for single registration execution. . . . .	122
4.7	NRR workflow diagram for speculative execution. . . . .	124
4.8	Execution timeline for single instance of NRR workflow on TeraGrid resources. . . . .	127
4.9	Execution timeline of the speculative NRR workflow on TeraGrid resources.	127
4.10	Estimation of the queue wait times for the used TeraGrid resources with the QBETS service. . . . .	130
5.1	Left: Binary image. Right: Corresponding greyscale image. . . . .	150

5.2	Visualization of a slice showing alignment error analysis results. . . . .	156
5.3	Comparison of the error estimation accuracy for the analyzed robust Hausdorff Distance based metrics. . . . .	157
5.4	Synthetic tumor deformation and recovered alignment error estimates. . .	158
5.5	Expert-estimated locations of the used landmarks for a pair of IGNS images.	160
5.6	Estimation of the alignment error for the pair of synthetic images. . . .	161

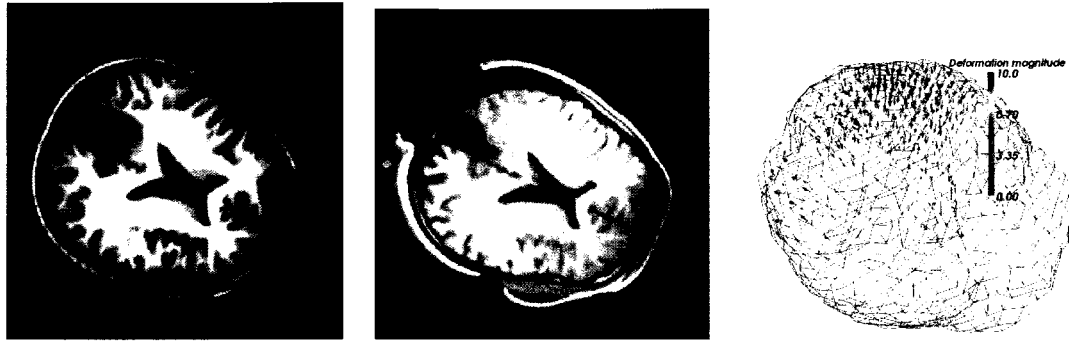
ENABLING TECHNOLOGY FOR NON-RIGID IMAGE  
REGISTRATION DURING IMAGE-GUIDED NEUROSURGERY

# Chapter 1

## Introduction

Cancer is one of the leading causes of death in the United States and around the world. According to the American Brain Tumor Association, the expected incidence of brain tumors in the US population was estimated at 52,236 new cases in 2008 [16, 43]. Neurosurgical resection is a primary treatment of brain tumors [17]. The goal of the resection is maximum removal of the tumor tissue with the minimum damage to the surrounding healthy tissue.

Image-guided neurosurgery (IGNS) utilizes pre-operatively acquired medical images to improve tumor localization and the precision of tumor resection, and to minimize negative consequences for the patient. However, intra-operative shifts of the brain structure are common during open skull brain surgery. This happens due to the leakage of cerebrospinal fluid (CSF), the injection of medications, brain tissue swelling, etc. There is evidence in the literature that such deformation can be in the order of up to 25 mm [64]. An illustration of the brain shift phenomenon is given in Figure 1.1. As a result of the intra-operative shifts, preoperative imaging data may become invalid in relation to the



**Figure 1.1:** Left: axial slice of a pre-operative MRI showing tumor. Center: intra-operative MRI showing brain shift and tumor resection. Right: tetrahedral mesh with the recovered deformation. MRI images courtesy of Surgical Planning Laboratory, Brigham and Women’s Hospital.

intra-operative brain configuration.

Recent technological advances attempt to remedy the problem of brain shift by providing means to track brain deformation. One such technology is the intra-operative MRI (iMRI) [116]. Using iMRI during a tumor resection procedure, surgeons can obtain MR imaging data that reveal brain deformations. This can be done with the use of open MR scanners, where the surgery is taking place with the patient located inside the scanner. Alternatively, the patient can be moved during surgery for iMRI acquisition. The primary goal of iMRI is to provide intra-operative imaging to the surgeons, showing tumor location and residual tumor volume to the surgeons. This is essential because the task of maximum safe tumor resection is complicated by the visual similarities between healthy and diseased tumor tissue, as well as the proximity of life-critical brain structures.

It was shown that the use of iMRI significantly improves the resection margins [54]. In turn, the improved resection margins result in a better prognosis for the patient [105, 161]. However, one of the remaining problems is the inability of iMRI scanners to fully

substitute the pre-operative imaging due to their limited capabilities as compared to closed-bore scanners, and due to the time constraints imposed on the acquisition process. Therefore, efforts are underway to develop methods and tools that use iMRI to deform the pre-operative MRI in order to account for intra-operative deformation [74, 166, 53]. The process of aligning the corresponding features of two images is known as *image registration*. Non-rigid registration uses spatially varying, non-global transforms, which account for local deformations of the imaged object.

A number of approaches to non-rigid registration have been proposed to date. In this dissertation we focus on the specific method developed by Clatz et al. [53], who developed a physics-based technique for registering pre-operative MRI data to the images acquired in the open magnet scanner during image-guided neurosurgery. This method combines biomechanical modeling of brain deformation with the estimates of deformation displacements from the intra-operative images. The authors evaluated the robustness of this method retrospectively using clinical data. The research presented in this dissertation enabled prospective application of this method by Archip and collaborators intra-operatively [8, 7]. The feasibility and accuracy of registration was confirmed during 11 neurosurgery procedures at Brigham and Women’s Hospital in Boston, MA.

This dissertation presents the development of *enabling technology* to register images during image-guided neurosurgery, and includes tools and methods essential for completing non-rigid registration within the constraints imposed by the neurosurgical procedure. Specifically, we developed technology for the registration method presented by Clatz et al., a method with the established accuracy and applicability. The objective of this work is not the development of an all-inclusive set of technologies required to

accomplish this task, but rather the development of some of the essential components of these technologies.

## 1.1 Contributions of this Work

This work describes a set of both practical and novel methods, and their implementations, that are important for the task of non-rigid registration in an intra-operative setting. We focus on a registration technique with established accuracy, which was evaluated off-line on the retrospective data. Prior to the completion of the work presented in this dissertation, the application of this registration technique was not feasible during the course of neurosurgery.

The first component developed in this dissertation is the construction of patient-specific biomechanical models of the brain from pre-processed MR imaging data. This model is required by the formulation of the registration technique. The studied problem, which we define as image-to-mesh conversion, presents new challenges to mesh generation approaches previously developed for Computer Aided Design (CAD) applications. We present and evaluate a number of approaches suitable for this task using both conventional quantitative metrics and the application specific quality measures. The methods we develop are based on open source software that facilitates their use in similar applications that require construction of tetrahedral meshes from images. Image-to-mesh conversion tools are essential to enable non-rigid registration computation.

Ensuring that the time required for registration computation is compatible with neurosurgery time constraints is the second topic studied in this dissertation. We develop a distributed implementation of the non-rigid registration technique, and use distributed

remote cluster resources connected with the hospital by a high-speed network as the platform for the execution of the restructured implementation. As a result, we were able not only to reduce the computation time to under 5 minutes, but also demonstrated that it is feasible to deliver remotely computed registration results intra-operatively.

A well-known difficulty in using complex image processing operations like non-rigid registration is in the selection of parameters that result in the best accuracy of the method. The optimal parameters may depend on the properties of the input data. Moreover, our preliminary studies of the registration parameter space show that different parameter combinations may be optimal for minimizing registration error at different image locations. When the parameters need to be selected within strict time constraints, optimal parameter selection becomes particularly challenging. In this dissertation we explore how the power of computational grids can be used to assist in improving registration accuracy. We present a feasibility study of *speculative execution* of non-rigid registration on a distributed cyberinfrastructure that connects thousands of computing nodes, freely available for research studies. During speculative execution, non-rigid registration is computed simultaneously with different parameter combinations. While non-rigid registration is a computationally challenging task itself, speculative execution of registration requires special infrastructure to distribute, monitor and analyze the results of execution of multiple registration instances. Such infrastructure is needed to augment the high-performance implementation of a single registration instance. In this dissertation we develop a framework for conducting speculative computations on TeraGrid, the research cyber-infrastructure of the United States [188].

For the application of speculative execution, it is essential to develop measure that



can determine the success or failure of non-rigid registration. In general, there are no automatic techniques to accomplish such evaluation for clinical data, because ground truth (the true deformation value at an image location) is rarely available intra-operatively. Moreover, even retrospective evaluation of non-rigid registration, when there are no constraints on the analysis time, presents significant challenges. Recognizing the difficulty of collecting ground truth data, and the spatial variability of the registration accuracy, we propose a new technique for calculating the alignment error at a given image location based on the definition of Hausdorff distance.

## 1.2 Thesis Structure

The dissertation is organized as follows:

- Chapter 2 summarizes background information related to the application of non-rigid registration, and describes the registration method.
- Chapter 3 describes the problem of image-to-mesh conversion in general, as well as the application-specific requirements imposed by the non-rigid registration method.
- Chapter 4 presents the high performance implementation of non-rigid registration, and the framework for its speculative execution.
- Chapter 5 describes the design, methodology and evaluation of this novel approach for automated assessment of image alignment.
- Chapter 6 summarizes the present research and suggests future work related to this dissertation.

## Chapter 2

# Image Registration

The research presented in this dissertation is motivated by the specific application of registering medical image data for the purposes of image-guided neurosurgery. In this Chapter we present the necessary background required to understand the basics of image registration and its context within the clinical application. We also describe the specific registration method targeted in this work, with the emphasis on its aspects and requirements addressed in the subsequent chapters.

### 2.1 Background

The use of Magnetic Resonance Imaging (MRI) for studying brain tissue has gained wide acceptance due to its superior capabilities in imaging soft tissue, absence of ionizing radiation and good contrast resolution for soft tissues [134]. MRI uses powerful magnetic field to systematically disrupt the spin of protons in the body, causing them to produce rotating magnetic fields, which are detected by the MR scanner coils. The strength of the signal depends on the physical properties of the tissue, and determines the values

of the image intensities derived during image reconstruction. MR image acquisition is parametrized by different *pulse sequences*, that are optimized for improved contrast while imaging different kinds of tissues. Typical MRI scans are obtained using so-called T1- or T2-weighted sequences. With T1 sequence, tissues that contain more non-bound water, like cerebral spinal fluid (CSF) appear dark, and those tissues that contain bound water molecules, like fat and brain tissues, are brighter. On the opposite, bright regions in the T2 scan correspond to the tissues rich in non-bound water.

Additional information about brain can be obtained using specialized sequences that are gaining more acceptance with the improvements in image acquisition and processing technology. Diffusion Tensor MRI (DT-MRI) measures the diffusion of water, which can be used to infer the orientation of white matter fibers. Functional MRI (fMRI) can be used to identify the cortex regions responsible for the specific kind of activity by spatially measuring blood oxygenation at high temporal resolution while the patient is asked to perform certain activities. Preliminary studies [149, 126] showed the value of fMRI and DT-MRI for the neurosurgery.

The validity of the preoperatively acquired image data is often compromised during the course of open skull neurosurgery due to significant deformation and shift of the brain tissue. This deformation is caused in particular by leakage of cerebrospinal fluid, injection of medications and swelling. Because of this deformation, the spatial alignment of the preoperative data with the *in situ* configuration of the brain structures is no longer accurate, and additional processing is required to recover this correspondence.

Image registration is the procedure of aligning the corresponding features of two or more images. In the context of IGNS, the objective of registration is to align the features

of the pre-operatively acquired data with the intra-operative image. The pre-operative image is therefore called *floating*, or *moving* image, since it will be transformed into the coordinate space of the intra-operative image, which is called *target*, *fixed*, or *reference* image.

Medical image is essentially a collection of samples of some physical value taken at regularly spaced locations within the imaged volume. We will use the notation suggested by Hill et al. [106] to introduce the most important related definitions. Image domain  $\Omega$  is defined as:

$$\Omega = \tilde{\Omega} \cap \Gamma_{\varsigma},$$

where  $\tilde{\Omega}$  is a bounded continuous set, also known as image *field of view*, and  $\Gamma$  is a sampling grid characterized by the anisotropic *spacing*  $\varsigma = \{\varsigma^x, \varsigma^y, \varsigma^z\}$ . In the context of registration, image spacing is usually different for floating and target images. In the case of IGNS, pre-operative image usually has better spatial resolution, while intra-operative image is optimized for reduced acquisition time. Value of the sample at the sampling grid is a mapping from some physical quantity into a scalar value. Commonly used imaging sequences, e.g., T1 weighted imaging, define a mapping  $A$  from a point in physical space  $\Omega$  into  $\mathbb{R}$ :

$$A : x \in \Omega \mapsto A(x) \in \mathbb{R}.$$

A volumetric sample value in a three-dimensional image is represented by image *voxel*, which is defined as a cuboid-shaped Voronoi region of size  $\varsigma$  centered at the sample grid point.

The result of image registration is a mapping  $\mathbf{T}$  that defines a transformation between the coordinate space of the moving image and the coordinate space of the reference

image. Given  $\mathbf{T}$ , the registered image is constructed by *resampling* the floating image. During resampling, the intensity value of each voxel in the coordinate space of fixed image is identified by first locating the corresponding point in the floating image space, and then interpolating the value of intensity at that point from the values of the neighboring voxels in the floating image, e.g., using the nearest-neighbor or linear interpolation functions.

Depending on the type of the transformation recovered, modality of the images, imaged subject and other criteria, the registration methods can be classified into different categories [36, 106]. With respect to the modality of the images that are registered, registration can be intra-modal or inter-modal. Intra-modal registration is applied to the images that were acquired using the same imaging technique, and usually the same pulse sequence, in the case of MRI. Inter-modal registration is usually a more difficult problem, because different image modalities may not necessarily be capable of showing the same anatomical features. For example, CT is ideal for imaging bone tissue, while in MRI bones have poor contrast. Image registration finds its applications both in aligning imagery of the same patient (intra-subject registration) and different patients (inter-subject registration).

Depending on the type of transformation, registration can be *rigid*, when the object is changing its position under rotation and/or translation, or *non-rigid*, when the transformation varies with the spatial location within the image. Affine transformation is similar to rigid, since it defines a single, global transformation, but in addition to translation and rotation it also includes scaling and shear. Affine transformations are typically applicable in inter-subject registration.

Finally, depending on the approach used to derive the transformation, the registration methods are classified into feature-based and intensity-based techniques. The methods in the first group rely on the identification of the same locations in images being registered, which can include surfaces or points. The methods from the second group are usually more general and robust, because the preliminary step of feature identification is not necessary. Intensity-based registration methods derive the registration transformation iteratively by optimizing certain voxel *similarity measure* between the images. The choice of the optimum similarity measure depends on the expected relationship between the intensities of the corresponding regions in the registered images. For example, in case of intra-modality registration when the images differ only by noise the sum of squared differences (SSD) measure can be sufficient [106, 207].

Image registration does not admit general solutions. Typically, registration methods differ depending on the application, which is defined by the kind of images, and the anatomical structures being registered. However, any registration method generally includes the same building blocks, which are the transform (global or non-rigid), interpolation function that allows to evaluate image values at non-voxel locations, similarity measure (or goodness-of-fit function) to assess the quality of alignment, and some optimization technique to guide the derivation of the transformation based on the goodness-of-fit value [109, 120].

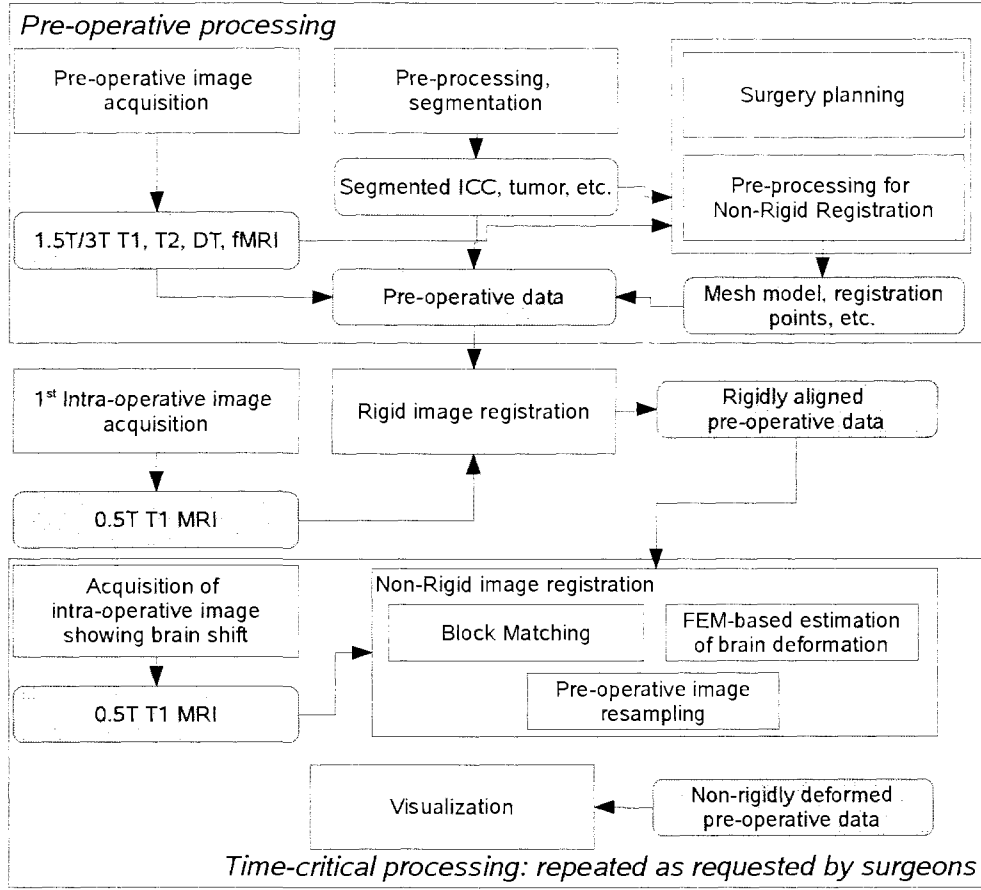
Registration of the brain MRI data for IGNS is typically a multi-step process, which includes both rigid and non-rigid components. The rigid transformation captures the differences in the position of the patient during pre-operative acquisition compared to the position during surgery. Local deformations that happen due to intra-operative

brain shift are recovered by non-rigid registration.

Registration, and especially non-rigid registration, is an inherently ill-posed problem. The objective of non-rigid registration is to produce the deformation vector at each voxel location of the image. At the same time, this vector has to be derived based only on the values of intensity in the reference and floating images [76]. Therefore, the non-rigid registration constructs some mapping from lower- to higher-dimensional space, which cannot be uniquely defined. Moreover, assuming that registration should guarantee alignment of certain homologous points, the alignment can be achieved with different transformations or deformation fields. The important question is therefore how to make the result of registration realistic. This is usually achieved by utilizing some form of smoothing, or regularization of the solution. A number of the registration methods that have been developed for brain shift recovery incorporate physical model of brain deformation to ensure realistic registration results. We next proceed to the description of one such method, that motivates the work in the subsequent chapters.

## **2.2 Robust Estimation of Volumetric Deformation**

The research described in this dissertation is targeting technology development for the specific non-rigid registration method developed by Clatz et al. in [53]. This method was designed for intra-modality same-subject registration of the pre-operative high-resolution brain MRI data with the intra-operative scans acquired in the open magnet MRI. In this Section we summarize the highlights of the registration method and introduce related definitions. The high-level diagram of the image processing steps during registration is shown in Figure 2.1. Registration-related processing includes pre- and intra-operative



**Figure 2.1:** Processing steps during non-rigid image registration for IGNS.

components, with only some steps being time-critical.

The pre-operative step includes acquisition and pre-processing of the patient data. As part of the standard clinical protocols multi-modal MRI scans of the patient are obtained prior to the surgery for the evaluation and surgery planning purposes. The pre-operatively acquired data is analyzed so that the tumor is localized with respect to the critical brain structures. The same data can be used for the pre-processing step of the registration, which does not require intra-operative image data. The registration method relies on the patient-specific biomechanical model of the patient brain, constructed from the segmented brain volume, and on the sparse estimation of displacements at selected



registration points. The locations of those points correspond to the regions of the image which shows significant structure, which can be identified in the intra-operative scans.

During the day of the surgery in the open MRI facility, the patient is placed on the operating table, and their head is rigidly fixated in a position that is most convenient for the operation. This position will generally be different from the position of the patient's head during pre-operative image acquisition. Rigid registration is used to account for this difference, and is applied to the pre-operative data to bring it into correspondence with the actual patient's position. This processing step is not time-critical, since there is usually a delay of more than an hour between the fixation of the patient's head and beginning of the tumor resection.

The time-critical component related to non-rigid registration is initiated by the acquisition of an MR scan showing brain deformation. Once this scan is available, it is used together with the results of the pre-operative processing for registration. First, the displacement vectors are estimated at the locations of the registration points by means of block matching. Next, the biomechanical model of brain deformation is used to iteratively discard outlier measurements from the results of block matching and derive the dense deformation field. Finally, the dense deformation field is applied to the pre-operative data for subsequent visualization.

Next we discuss the processing steps in more detail, following the original presentation of Clatz et al. [53].

### 2.2.1 Pre-operative Processing

The first component of the pre-processing stage is the construction of the tetrahedral discretization of the brain. The tetrahedral (finite element) discretization is subsequently used for numerical solution of the differential equation describing the mechanical behavior of the brain tissue. This task will be studied in more detail in Chapter 3. Here we give a brief overview of the approach to tetrahedral model construction described in the original paper.

The voxels that correspond to the intra-cranial cavity (ICC) in the pre-operative image are identified by means of semi-automatic segmentation. Image segmentation is a fundamental research problem in medical image analysis, and in general no automatic methods exist for segmenting an arbitrary structure from the image. Segmentation produces a binary image  $M$ , where each voxel is marked as inside or outside the ICC:  $M(x) : A(x) \mapsto \{0, 1\}$ . This binary image implicitly describes the ICC. The tetrahedral model is then constructed by first recovering the explicit triangulated surface of the ICC using the Marching Cubes (MC) algorithm [129], which is next *decimated* to reduce the number of surface triangles. The decimated triangulation of the ICC surface is finally used as the input for the Delaunay-based mesh generator *GHS3D* [182], which produces the tetrahedral mesh  $\mathcal{M} = (\mathcal{V}, \mathcal{T})$ .  $\mathcal{V}$  is the set of vertices, and  $\mathcal{T}$  is the set of mesh tetrahedra.

The second component of the initialization prepares the data for sparse estimation of the initial deformation. Given the pre-operative image  $A$ , this step produces a set of points in voxel coordinates  $\mathbf{f} \in \mathcal{F}$ ,  $\mathbf{f} = (x, y, z)$ . We will refer to these points as *registration points*. Deformation vector will be estimated in the intra-operative part of

the algorithm at each of these registration points. The selection of registration points is an iterative process itself. Initially, all of the points that correspond to the centers of the voxels inside the overlap of the segmented ICC mask  $M$  and tetrahedral mesh  $\mathcal{M}$  are included in the list of registration points  $\mathcal{F}_{init}$ . Note, that due to surface decimation, not all of the mask will be inside the mesh.

For each registration point  $\mathbf{f}$ , the mean and variance of the intensity in the voxels within the cuboid region of size  $B = (B_x, B_y, B_z)$  centered at the registration point are calculated. The points in the  $\mathcal{F}_{init}$  list are ordered by their corresponding variance. The list is traversed starting from the highest variance value. A registration point is selected in the list  $\mathcal{F}$  if the following two constraints are satisfied: (1) the voxel corresponding to the registration point is not connected with the voxel corresponding to the already selected point, given the specific connectivity definition, and (2) the total number of the selected points is below  $r_1|\mathcal{F}_{init}|$ . Connectivity defines whether the voxels corresponding to the selected registration points must not share faces (6-connectivity), faces and edges (18-connectivity) or faces, edges and corners (26-connectivity). The parameters used during registration points selection and their default values, as suggested by Clatz and collaborators, are summarized in Table 2.1.

Based on the finalized list of the registration points, normalized structure tensor [14]  $\mathbf{T}(\mathbf{f})$  is calculated using the following definition:

$$\mathbf{T}(\mathbf{f}) = \frac{\sum_{i=\mathbf{f}^x-B_x+1}^{\mathbf{f}^x+B_x-1} \sum_{j=\mathbf{f}^y-B_y+1}^{\mathbf{f}^y+B_y-1} \sum_{k=\mathbf{f}^z-B_z+1}^{\mathbf{f}^z+B_z-1} [\nabla F(i, j, k) \odot \nabla F(i, j, k)^T]}{tr(\sum_{i=\mathbf{f}^x-B_x+1}^{\mathbf{f}^x+B_x-1} \sum_{j=\mathbf{f}^y-B_y+1}^{\mathbf{f}^y+B_y-1} \sum_{k=\mathbf{f}^z-B_z+1}^{\mathbf{f}^z+B_z-1} [\nabla F(i, j, k) \otimes \nabla F(i, j, k)^T])},$$

where  $\nabla F(i, j, k)$  is the value of gradient at a given registration point:

$$\nabla F(x, y, z) = \left[ \frac{F(x-1, y, z) - F(x+1, y, z)}{s_A^x} \quad \frac{F(x, y-1, z) - F(x, y+1, z)}{s_A^y} \quad \frac{F(x, y, z-1) - F(x, y, z+1)}{s_A^z} \right]^T.$$

**Table 2.1:** Parameter space for the non-rigid registration method of Clatz et al. [53].

parameter	default setting
Young modulus ( $E$ )	694 $Pa$
Poisson's ratio ( $\nu$ )	0.45
selected fraction of registration points ( $r_1$ )	10%
block connectivity ( $bConn$ )	26
search block dimensions ( $B$ )	$7 \times 7 \times 7$
search window dimensions ( $W$ )	$11 \times 11 \times 25$
search step ( $sStep$ )	$1 \times 1 \times 1$
number of rejection steps ( $n_{iter}$ )	10
rejected blocks fraction ( $r_2$ )	25%
energy trade-off ( $\alpha$ )	$\frac{trace(K)}{n}$
error model breakup point ( $\lambda$ )	0.5
CG precision ( $r_{CG}$ )	0.001

$\mathbf{T}(\mathbf{f})$  is a second order positive-definite tensor. Eigen-decomposition applied to the structure tensor gives the direction of gradient together with the certainty in the gradient localization, as defined by the eigenvalues. Structure tensor is used in the subsequent formulation to increase the confidence in the deformation estimation.

Pre-operative image data is acquired in the coordinate space which is generally not aligned with the position of the patient during the surgery. As part of the standard procedure for image acquisition during IGNS within iMRI, the initial MRI scan is acquired once the patient head is fixed for the operation. Rigid registration is applied to the pre-operative data to align the pre-operative data with the patient position in the iMRI scanner.

### 2.2.2 Intra-operative Processing

The time-critical component of the registration is initiated with the acquisition of an intra-operative MRI showing brain shift.

The sparse displacement field between the floating and reference images is estimated

with the aid of volumetric block matching [53]. For each registration point  $\mathbf{f} \in \mathcal{F}$ , block matching iteratively searches for such a location of the region  $B$  centered at this point in the floating image (defined as search block) that maximizes the similarity metric with respect to the overlapping portion of the search region  $W$  (search window) within the reference image. The search for optimum matching block is parametrized by the sizes of the block and window regions, and by the voxel similarity metric. Clatz et al. suggest the use of Normalized Cross Correlation (NCC) [53] as a measure of image block similarity. Given the region  $B_F$  of the floating image centered at a certain registration point and the overlapping portion of the search window in the reference image  $B_R$ , NCC is defined as:

$$NCC = \frac{\sum_{i \in B} (B_T(i) - \bar{B}_T)(B_F(i) - \bar{B}_F)}{\sqrt{(\sum_{i \in B} (B_T(i) - \bar{B}_T)^2)(\sum_{i \in B} (B_F(i) - \bar{B}_F)^2)}}.$$

$\bar{B}_T$  and  $\bar{B}_F$  correspond to the average intensity values within the block in the reference and floating image respectively.

NCC measure is computed for every possible overlap of the block region with the window region, and the location which provides the maximum value of NCC is selected as the estimated new position of the registration point in the target image. The numeric value,  $c(\mathbf{f})$ , of NCC for the registration point  $\mathbf{f}$  is between 0 and 1, and serves as the measure of confidence in the estimated displacement. We note the high computational complexity of the block matching procedure. Considering the sizes of three-dimensional block and window are defined in pixels as  $B = \{B_x, B_y, B_z\}$  and  $W = \{W_x, W_y, W_z\}$ , the bound on the number of operations is

$$O(B_x B_y B_z \times W_x W_y W_z \times |\mathcal{F}|).$$

The block matching result contains outlier displacements: for some registration points the maximum value of NCC will not correspond to the location of the image block in the target image. This can happen due to a number of reasons. For example, registration point may correspond to the region of the image that does not show sufficient anatomical structure; the anatomical region that appears in the search block may not be present in the target image due to resection, and the resulting match is arbitrary; target image may be acquired after injection of contrast agent, which will result in non-linear changes in intensity values therefore violating the assumptions required for validity of the results obtained with NCC.

In addition to the presence of outliers in the block matching results, there will be regions of the image without any estimations of the deformation because the distribution of the registration points can be highly non-uniform. Hence, the challenge of registration is to minimize the influence of the outliers, and estimate dense brain deformation from a sparse and irregular set of displacements. One solution is to formulate the problem in terms of energy minimization. Consider the brain in its rest state, without any deformation and any forces applied. Displacements at the selected registration points provide estimations of brain deformation. The total energy of the system  $\mathbf{W}$  can be defined as the sum of the mechanical and matching energy components. Let  $m = |\mathcal{F}|$  (the number of registration points), and  $n = |\mathcal{V}|$  (the number of mesh vertices). The  $\mathbf{U}$  is the vector of size  $3n$  (three degrees of freedom for each vertex in the mesh) that describes displacement of each mesh vertex.  $\mathbf{D}$  is the vector of size  $3m$ , that contains the displacements recovered by block matching.

A tetrahedron  $\mathbf{t}_i \in \mathcal{T}$  is defined by the four vertices  $\mathbf{p}_j$ ,  $j = 0, \dots, 3$ . Mesh de-

formation is described by the displacement vector at each mesh vertex  $\mathbf{u}_i = [\mathbf{U}]_i^{3 \times 1} = (u_i^x, u_i^y, u_i^z)$ . Then the displacement vector at any point  $\mathbf{X} = (x, y, z) \in \mathbf{t}$  can be found as:

$$\mathbf{u}(\mathbf{X}) = \sum_{j=0}^3 h_j(\mathbf{X}) \mathbf{u}_j,$$

where  $h_j(\mathbf{X})$ ,  $j = 0, \dots, 3$  are the linear shape functions [63], defined as:

$$\begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_0^x & \mathbf{p}_1^x & \mathbf{p}_2^x & \mathbf{p}_3^x \\ \mathbf{p}_0^y & \mathbf{p}_1^y & \mathbf{p}_2^y & \mathbf{p}_3^y \\ \mathbf{p}_0^z & \mathbf{p}_1^z & \mathbf{p}_2^z & \mathbf{p}_3^z \\ 1 & 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}.$$

$\mathbf{H}$  is defined as a  $3m \times 3n$  matrix that interpolates displacements at the registration points from the displacements at the mesh vertices. For each combination of mesh vertex  $\mathbf{v}_i \in \mathcal{V}$  and registration point  $\mathbf{f}_j \in \mathcal{F}$ , there is a  $3 \times 3$  submatrix  $[\mathbf{H}]_{j,i}^{3 \times 3}$ . Let  $\mathcal{S}(\mathbf{v}_i)$  be the set of all tetrahedra in  $\mathcal{T}$  that are incident on vertex  $\mathbf{v}_i \in \mathcal{V}$  (i.e., share this vertex). The set of tetrahedra in  $\mathcal{S}$  is known as the *vertex cell complex*. The submatrix  $[\mathbf{H}]_{j,i}^{3 \times 3}$  is non-zero and is defined as:

$$[\mathbf{H}]_{j,i}^{3 \times 3} = h_i^{\mathbf{t}} \cdot I$$

if and only if  $\exists \mathbf{t} \in \mathcal{T} : \mathbf{f}_j \in \mathbf{t}, \mathbf{t} \in \mathcal{S}(\mathbf{v}_i)$  (registration point  $\mathbf{f}_j$  is located inside the tetrahedron  $\mathbf{t}$ ).

$\mathbf{S}$  is a  $3m \times 3m$  block-diagonal matrix that reflects the confidence in the deformation estimate recovered by block matching at a given registration point. A  $3 \times 3$  submatrix  $[\mathbf{S}]_{j,j}^{3 \times 3}$  which corresponds to the  $j$ th registration point is defined as the structure tensor  $\mathbf{T}$  evaluated at that point scaled by the normalized value of cross-correlation  $c$  evaluated

during block matching:

$$[\mathbf{S}]_{j,j}^{3 \times 3} = \frac{|\mathcal{V}|}{|\mathcal{F}|} c(\mathbf{f}_j) \mathbf{T}(\mathbf{f}_j).$$

Given the definitions of the displacement vectors  $\mathbf{U}$  and  $\mathbf{D}$ , interpolation matrix  $\mathbf{H}$  and the weighting matrix  $\mathbf{S}$ , the matching energy is defined as:

$$\mathbf{W}_{matching} = (\mathbf{H}\mathbf{U} - \mathbf{D})^T \mathbf{S} (\mathbf{H}\mathbf{U} - \mathbf{D}).$$

Minimization of the matching energy with respect to  $\mathbf{U}$  results in such a configuration of the mesh that minimizes the error term ( $\|\mathbf{H}\mathbf{U} - \mathbf{D}\|$ ), which is the difference between the displacements  $\mathbf{D}$  recovered by block matching and those interpolated from the displacements at the mesh vertices. The values of the mesh vertex displacements that minimize the error term can be found with the Least Squares method [91]. Such formulation is termed as *interpolation formulation* by some sources [53], since with sufficiently dense mesh (e.g., when there is a mesh vertex corresponding to each of the registration points) the error term can be reduced to zero.

Although the interpolation formulation minimizes the registration error point-wise, the vertices of the mesh that do not contain any registration points do not move, and the resulting deformation may not have any physical meaning.

Physical constraints that approximate brain deformation under prescribed load conditions are introduced into the formulation to make the result more realistic. The mechanical energy is the energy that is exerted as a result of body deformation:

$$\mathbf{W}_{mechanical} = \mathbf{U}^T \mathbf{K} \mathbf{U},$$

where  $\mathbf{K}$  is the stiffness matrix that captures the physical properties of the modeled



object [63]. Mechanical energy is proportional to the amount of deformation, and the total energy of the system is:

$$\mathbf{W} = \mathbf{W}_{mechanical} + \mathbf{W}_{matching} = \mathbf{U}^T \mathbf{K} \mathbf{U} + (\mathbf{H} \mathbf{U} - \mathbf{D})^T \mathbf{S} (\mathbf{H} \mathbf{U} - \mathbf{D}).$$

Therefore, with this updated formulation, the final configuration of the system is the state of equilibrium between the internal mechanical and matching forces. The mesh vertex displacements at the equilibrium state are found by differentiating with respect to  $\mathbf{U}$  and solving  $\frac{\partial \mathbf{W}}{\partial \mathbf{U}} = 0$ :

$$\frac{\partial \mathbf{W}}{\partial \mathbf{U}} = [\mathbf{K} + \mathbf{H}^T \mathbf{S} \mathbf{H}] \mathbf{U} - \mathbf{H}^T \mathbf{S} \mathbf{D} = 0.$$

The internal forces of the body resist deformation and the external forces are driving the model deformation based on the sparse displacements from block matching. As a result, even with the arbitrarily dense mesh discretization, the error term ( $\|\mathbf{H} \mathbf{U} - \mathbf{D}\|$ ) cannot become zero. Therefore, this formulation is referred to as *approximation formulation*.

The robust estimation method proposed by Clatz et al. is based on the idea of iterative convergence from the approximation to interpolation formulation. This is achieved by introducing a force term  $\mathbf{F}$  that is updated at each iteration increasing from zero to the value which balances the mechanical forces of resistance:

$$[\mathbf{K} + \mathbf{H}^T \mathbf{S} \mathbf{H}] \mathbf{U} = \mathbf{H}^T \mathbf{S} \mathbf{D} + \mathbf{F}.$$

The iteration scheme is defined as follows:

$$\mathbf{F}_i \Leftarrow \mathbf{K} \mathbf{U}_i$$

---

**Algorithm 1** Iterative estimation of brain deformation with outlier rejection.

---

**Input:**  $\mathcal{F}$ ,  $\mathbf{K}$ ,  $\mathbf{H}$ ,  $\mathbf{S}$ ,  $\mathbf{D}$ ,  $r_2$ ,  $n_{iter}$  (see Table 2.1)

**Output:** Mesh deformation defined by  $\mathbf{U}$

```

1:  $F_i \Leftarrow 0$ 
2: for  $i = 0$  to  $n_{iter}$  do
3:    $\mathbf{F}_i \Leftarrow \mathbf{K}\mathbf{U}_i$ 
4:    $\mathbf{U}_{i+1} \Leftarrow [\mathbf{K} + \mathbf{H}^T\mathbf{S}\mathbf{H}]^{-1}[\mathbf{H}^T\mathbf{S}\mathbf{D} + \mathbf{F}_i]$ 
5:   for all  $\mathbf{f}_m \in \mathcal{F}$  do
6:     compute error function  $\xi(\mathbf{f}_m)$ 
7:   end for
8:   reject  $\frac{r_2|\mathcal{F}|}{n_{oi}}$  points with highest  $\xi(\mathbf{f}_m)$ 
9:   for all  $\mathbf{f}_m \in \mathcal{O}(\mathcal{F}, i)$  do
10:    remove the contribution of  $\mathbf{f}_m$  from  $\mathbf{H}^T\mathbf{S}\mathbf{H}$  and  $\mathbf{D}$ 
11:   end for
12:    $\mathbf{U}_{i+1} \Leftarrow [\mathbf{K} + \mathbf{H}^T\mathbf{S}\mathbf{H}]^{-1}[\mathbf{H}^T\mathbf{S}\mathbf{D} + \mathbf{F}_i]$ 
13: end for
14: repeat
15:    $\mathbf{F}_i \Leftarrow \mathbf{K}\mathbf{U}_i$ 
16:    $\mathbf{U}_{i+1} \Leftarrow [\mathbf{K} + \mathbf{H}^T\mathbf{S}\mathbf{H}]^{-1}[\mathbf{H}^T\mathbf{S}\mathbf{D} + \mathbf{F}_i]$ 
17: until convergence

```

---

$$\mathbf{U}_{i+1} \Leftarrow [\mathbf{K} + \mathbf{H}^T\mathbf{S}\mathbf{H}]^{-1}[\mathbf{H}^T\mathbf{S}\mathbf{D} + \mathbf{F}_i].$$

The attempt to remove outliers from the block matching data is made by discarding those points where the *error function* exceeds a predefined threshold. The error function  $\xi(\mathbf{f}_j)$  for registration point  $\mathbf{f}_j$  is defined as follows:

$$\xi(\mathbf{f}_j) = \|[\mathbf{S}]_{j,j}^{3 \times 3}[(\mathbf{H}\mathbf{U})_j - \mathbf{D}_j]\|.$$

The complete algorithm for iterative deformation estimation is given in Algorithm 1.

## 2.3 Requirements

Non-rigid registration during IGNS is a computation, which can impact life and health of the patient. In order to show that a particular algorithm and its implementation

can become part of an established clinical workflow, it undergoes thorough evaluation. The objective of the neurosurgery is maximum removal of tumor tissue with minimum damage to the healthy structures. The primary goal of non-rigid registration is to assist in this task, therefore, the first requirement is to improve identification of the tumor margin by providing accurate estimate of the deformation field. *Accuracy* is the main requirement to NRR.

The outcome of registration must be correct under varying conditions of image acquisition, noise, amount of resection and brain shift. The ability of the method to function correctly under different external conditions is known as *robustness*, and is another essential requirement. Robustness of the algorithm is related to its *sensitivity*, which can be defined as the variation in the output of registration as a result of changes to its inputs. In addition to the differences introduced by the case-specific details of anatomy, tumor size/location, and image acquisition, the result of the registration depends on the selection of the registration parameters summarized in Table 2.1.

Finally, the results of registration in the context of IGNS can only be relevant and useful if delivered as requested by the neurosurgeon. Acquisition of the intra-operative MRI introduces a significant disruption into the resection procedure. The non-rigid registration procedure should add minimum extra wait time to the resection procedure, and should be performed within minutes of iMRI acquisition.

Improving the accuracy, robustness and speed of the described non-rigid registration algorithm are the focus of this dissertation. In the following chapters we develop methods and tools that allow to deliver an implementation of the non-rigid registration method to satisfy these critical requirements.

## Chapter 3

# Mesh Generation

Study of the dynamic natural phenomena often requires solution of partial differential equations. Analytical solutions of differential equations describing the behavior of complex systems is usually not possible due to complicated geometry and boundary conditions. Instead, an approximate solution is sought. Finite Element Method (FEM) [211] is a numerical technique that allows to find such approximate solutions to partial differential equations. In particular, FEM is commonly used for solving continuum mechanics equations in biomechanical modeling [63]. A prerequisite step to application of the FEM is the approximation of the continuous geometric domain using discrete elements of simple shapes. The resulting discretization is often called *finite element mesh*, and the process of constructing such mesh is known as *mesh generation*.

Most of the research in mesh generation has been stimulated by the applications in the domain of Computer Aided Design (CAD). In the traditional formulation of the problem that has been adopted by the CAD community, the three-dimensional geometric domain  $\Omega \subset \mathbb{R}^3$  is defined by its boundary  $\partial\Omega$ , which consists of surfaces and curves

in their parametric representation [211]. Sections of the domain surface are defined by the so called *patches*, or *faces*, which are sections of the parametric surfaces in the form of B-splines or NURBS (Non-Uniform B-Splines) [160]. Edges of these patches are portions of spline curves. Construction of such parametric representation takes place prior to volumetric mesh generation, and is usually done manually using CAD software tools. The resulting surface  $\partial\Omega$  usually precisely corresponds to the true boundary of the modeled domain.

Relatively recently, representations of the object boundary with piecewise-linear elements, or facets, started gaining wider acceptance [153]. Such representations can be extracted from the previously meshed volumes, when the CAD surface is not available any more, or when it is inherently not known, as is the case for most medical data. Most of the mesh generation tools currently available accept the triangular faceted surfaces as input description of the domain for volumetric mesh generation.

Most computational schemes which use FEM require *conformal* meshes. A mesh of a closed bounded domain in three dimensions is conformal when all of the following requirements hold: (1) the union of all mesh elements is equal to the input domain, (2) each of the elements has non-empty interior, (3) the intersection of interiors of any two elements is empty, and (4) the intersection of any two elements is the empty set, a vertex, an edge, or a face [87]. Mesh generation methods can be classified according to the type of the simple shape used in discretization, with the tetrahedral and hexahedral discretizations being the most popular ones. Tetrahedral meshes are often preferred in the modeling applications used in medical research, because they are easier to construct, better approximate complicated shapes, and are more appropriate for remeshing [63].

The use of FEM has gained significant popularity in medical applications in the recent years. In particular, FEM was used for electro-magnetical source localization in the brain [205], modeling irradiation in tumor therapy [155], head collision impact simulation [124], tumor growth modeling [136], prediction of human facial shape after craniofacial surgery [122], and finally non-rigid FEM image registration [75, 53]. Consequently, there has been increased interest in adapting the traditional approaches and developing custom mesh generation methods suitable for these new applications of FEM.

In general, there are no patient-specific CAD models of internal organs suitable for FEM analysis in medical applications. The primary source of information about the shape and configuration of the given organ is medical imaging, such as MRI. However, delineation of the organ within the image is a complex task, and reconstruction of the CAD model from such delineation is often not feasible or not practical. The lack of precise geometric models and the need to construct volume tessellations from the image data is an important limitation, which complicates the use of existing off-the-shelf meshing tools. Application-specific requirements to the volume mesh make the problem even more challenging. In practice, there is no single widely accepted method to address the mesh generation needs of all applications.

The work presented in this Chapter is primarily motivated by the lack of end-to-end solutions for constructing tetrahedral meshes from the medical image data. We are not the first to address this problem. However, to the best of our knowledge, for the first time we propose an end-to-end approach to mesh generation from medical image data that has been implemented and distributed as open source software. We present this approach together with the analysis of its limitations, and compare it with two

existing techniques that are based on traditional meshing tools. We note, that all of the approaches we consider are available for wider use as open source software, and have been evaluated on the common sets of data in the context of physics-based non-rigid registration.

In this Chapter we discuss the specifics of mesh generation for the non-rigid registration method of Clatz et al. [53]. We begin by formulating the problem of constructing meshes from the image data. We discuss the generic requirements typical for the applications of FEM to the problems, where input domain is represented as an image. Next we formulate the unique application requirements motivated by the understanding of the algorithm discussed in the previous Chapter. We develop a hierarchy of the relevant mesh generation methods from the literature, with the focus on their suitability for NRR.

Based on this discussion, we present three specific approaches to tetrahedral mesh generation from medical image data. One of these approaches have been developed as part of the research presented in this dissertation. The other two techniques use well-established off-the-shelf components. The latter two techniques require special pre-processing of the image data, since they have not been designed for the image-to-mesh conversion task. Each of the evaluated methods is a representative of a separate group of traditional approaches to mesh generation. We develop an evaluation framework and discuss specific quantitative metrics to compare the three mesh generation approaches. We conclude this Chapter with the qualitative and quantitative comparison of the considered tetrahedral mesh generation methods in the context of the non-rigid registration application.

### 3.1 Image-to-Mesh Conversion

As discussed in Section 2.1, image  $A$  is a mapping of points from the image domain  $\mathbb{R}^3$  to  $\mathbb{R}$ ,  $A : \mathbf{x} \in \Omega \mapsto A(\mathbf{x})$ . Within the image, there is a tissue, or organ of interest, that corresponds to the modeled geometric domain. The objective of mesh generation is to construct a discretization of that organ, so that its surface is accurately represented by the surface of the discretized model. Obviously, the mapping of sample points provided by the image  $A$  alone is not sufficient to achieve this task. The minimum extra information that is required to construct a finite element discretization is a rule that defines which of the image samples are located inside the organ.

In this research we are mostly concerned with the medical applications that provide a *segmentation* of the object of interest. The output of segmentation is a binary image, i.e.,  $A(\mathbf{x}) \in \{0,1\}$ , with the subset  $\{\mathbf{x} \in \Omega | A(\mathbf{x}) = 1\}$  corresponding to the voxels located inside the object. Let  $\Sigma$  be the surface that separates zero and non-zero voxels of this binary image. The surface  $\Sigma$  is defined implicitly, as we only know whether a given voxel is inside or outside the object. Based on the inside/outside assignment of the image voxels in relation to the object of interest, we can construct an approximation of the implicit function  $\phi(x)$ , such that the object surface is defined by the zero level set of  $\phi(x)$ ,  $\{x : \phi(x) = 0\}$  [138]. Such approximation can be evaluated by computing the *signed distance transform* on the segmented image. Signed distance transform produces distance map image, where the absolute value at each voxel defines the distance to the closest boundary pixel of the input binary image. The sign of the voxel value is negative for the locations corresponding to the non-zero voxels in the input image, and positive otherwise. A number of algorithms for fast computation of Euclidean distance transform



have been proposed [61, 133].

The objective of mesh generation for FEM computations from the binary image data is to construct a conforming tetrahedral mesh  $\mathcal{M} = (\mathcal{V}, \mathcal{T})$ , which satisfies the following *generic requirements*:

**R1** The mesh boundary (surface triangulation) should be close to  $\Sigma$ .

**R2** Mesh size should be minimized.

**R3** Mesh elements should not have small angles.

We consider these requirements generic, since they are important for any application that uses FEM computations, where the input domain is described as an image. The implicit surface  $\Sigma$  is always an approximation of the true object boundary due to the intrinsic error with respect to the true surface. The exact continuous boundary of the organ can rarely be obtained because of the limitations imposed by the resolution of the image acquisition device, and the difficulties of precise location of the true boundary in the image. Instead, the piecewise-continuous boundary is usually estimated from the sparse sample of boundary points. The problem of recovering the parametric surface from the organized set of samples has been addressed by the computational geometry community, and a number of methods have been developed, as we discuss in the next Section. In general, imprecisely recovered surface can result in the unrealistic outcome of the modeling [129]. Also, in case of the specific registration method we consider, surface approximation error is particularly important. High surface approximation error will inevitably lead to the exclusion of the displacement estimations for the registration points located outside the brain surface.

The size of the mesh determines the size of the system of equations that need to be solved in the FEM formulation, and therefore the time required for that solution. Locally, the size of the mesh elements determines the bounds on the solution accuracy [211]. Both the computation time and accuracy of the solution have important implications for medical imaging. Therefore, flexibility of a mesh generation method in terms of the size of the mesh it produces (number of mesh elements) and the ability to adjust the size of the mesh elements locally is important. Local size of the mesh is commonly defined by means of a *sizing function*. Given the specific sizing function and domain description, an important property of a mesh generation method is its ability to construct the mesh with as few elements as possible while maintaining other requirements. Such property is known as mesh *size optimality* [176].

Finally, generic requirement R3 is important because the small angles in the mesh lead to instability of the solution and poor conditioning of the stiffness matrix constructed by the FEM. Ill-conditioned systems of equations result in slower convergence for iterative methods, and may require additional preconditioning [177], which in turn increases solution time.

Development of the mesh generation methods that attempt to meet the three generic requirements has been the focus of the mesh generation community for decades. On the other hand, the application-specific requirements are usually addressed by constructing a customized mesh sizing function, or developing customized mesh generation methods. In this paper we explore the first approach. Therefore, the ability to accept a user-defined sizing function is an essential feature for a mesh generation method to be considered for our application.

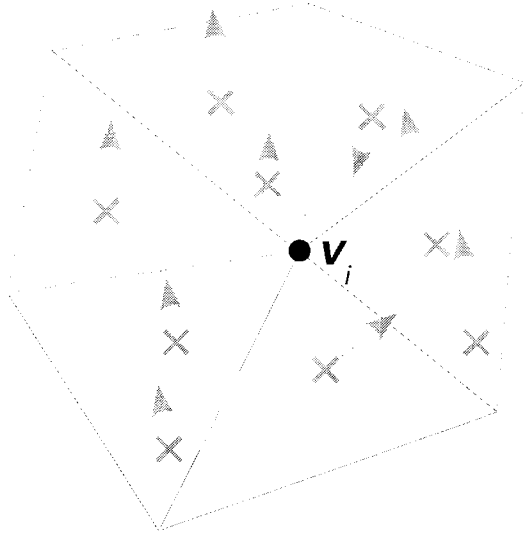
## 3.2 Application Requirements

The difficulty of defining the ideal tetrahedral mesh for the targeted non-rigid registration method is due to the fact that the mesh has a dual role in the formulation.

First, it is used to model the mechanical component of the energy. The mesh is used to construct the stiffness matrix that captures physical properties of the brain. The desirable characteristics for this part of the formulation are well-studied in the FEM and mesh generation communities [177, 211].

However, the tetrahedral mesh is also used to regularize, or smooth, the displacements recovered by block matching locally within the mesh vertex neighborhood. This neighborhood, or mesh vertex cell complex, shown in Figure 3.1 for a two-dimensional mesh, includes all the mesh elements adjacent to the specific vertex. The displacement recovered at each of the registration points within the mesh vertex cell complex affects the displacement at the corresponding mesh vertex. Therefore, it is important to maintain the empirically obtained ratio between the number of mesh vertices and the number of registration points under 0.1 (at least 10 registration points per mesh vertex) [53]. Ideally, this ratio should be maintained for every mesh vertex. The matrix  $\mathbf{H}^T \mathbf{S} \mathbf{H}$  has a non-zero  $3 \times 3$  entry for each mesh vertex and edge with the cell complexes containing registration points. The corresponding sub-matrices can be expressed as the sum over the registration points in a cell. The diagonal  $3 \times 3$  sub-matrix that corresponds to the mesh vertex  $\mathbf{v}_i$  can be calculated as the following summation over the registration points in the cell complex  $\mathcal{S}$  of  $\mathbf{v}_i$ :

$$[\mathbf{H}^T \mathbf{S} \mathbf{H}]_i = \sum_{\forall T \in \mathcal{S}(\mathbf{v}_i)} \sum_{\forall k \in T} h_{\mathbf{v}_i}^T(k)^2 \frac{\text{tr}(\mathbf{K})}{np} c(k) \mathbf{T}(k).$$



**Figure 3.1:** Mesh vertex cell complex in two dimensions; displacement at vertex  $v_i$  approximates displacements at the registration points, marked with ‘x’.

Here,  $h_{\mathbf{v}_i}^T(k)$  is the barycentric coordinate of the  $k$ th registration point with respect to  $\mathbf{v}_i$  in the containing tetrahedron  $\mathcal{T}$ ,  $n$  is the number of mesh vertices,  $p$  is the number of the registration points,  $c(k)$  is the correlation coefficient from block matching, and  $\mathbf{T}(k)$  is the image structure tensor at  $k$ th registration point.

The result of non-rigid registration is a deformation field, which provides mapping between the voxels in floating and fixed images. The value of the deformation at a given voxel is computed by interpolating the displacements from the mesh vertices after the last iteration of the outlier rejection [63]:

$$\forall \mathbf{x} \in \mathcal{T} : \mathbf{u}(\mathbf{x}) = \sum_{j=0}^3 h_{\mathbf{v}_j}^T(\mathbf{x}) \mathbf{u}(\mathbf{v}_j).$$

Note that the iterative procedure of outlier rejection described in Chapter 2 requires solving the linear system of equations at each iteration within the time-critical part of the computation. The size of this system depends on the number of degrees of freedom

in the mesh model.

Based on the registration formulation, we derive the following application-specific requirements to mesh generation:

**R4** Equi-distribution of the registration points with respect to mesh vertex cells.

**R5** Minimization of the approximation error with respect to displacement recovered by block matching at registration points.

**R6** Prevention of tetrahedron inversion during mesh deformation.

The requirement R4 is critical due to the presence of outliers in the output of block matching. If the vertex cell complex that corresponds to a mesh vertex contains small but non-zero number of registration points, that vertex becomes sensitive to the outlier registration points [52]. Empty vertex cells do not pose such a problem, since their deformation will be determined by the neighboring mesh elements. The empirically derived ratio of the registration points to the number of vertices used by Clatz et al. is about 15 matches per vertex [53]. During the process of outlier rejection, non-rigid registration discards registration points. Therefore, their distribution will change, and the optimal configuration of the mesh may need to be adjusted. However, in this research we consider only construction of the initial mesh, and not its refinement.

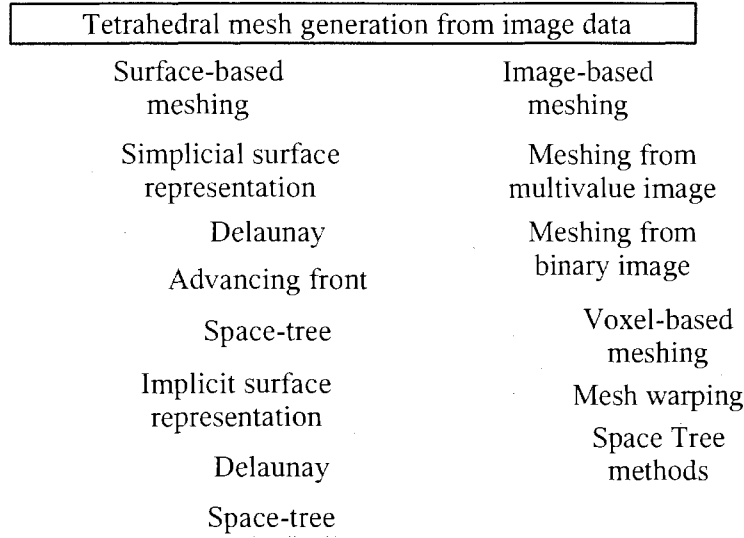
We introduce the requirement R5 since the same distribution of registration points can be achieved with the elements of different size. As an example, consider the mesh and distribution of registration points shown in Figure 3.1. Assuming that there are no other registration points in the cells of the vertices located on the cell boundary, their locations can change without affecting the assignment of registration points at vertex

$\mathbf{v}_i$ . However, larger elements are known to increase errors due to insufficient level of discretization [211]. Therefore, the meaning of the requirement R5 is to use the smallest possible elements, while maintaining the desired distribution of the registration points.

Finally, the last requirement R6 constrains the minimum element size in the mesh. Small elements undergoing large deformations are more likely to collapse (element volume is zero) or invert (element volume is negative). As we discussed in Chapter 2, the deformation vectors are updated during each iteration of the outlier rejection, and the final displacements at the mesh vertices are used to interpolate the dense deformation field within each tetrahedron. While the interpolation error does not depend on the tetrahedron shape, inversion or collapse of a tetrahedron will result in an unrealistic deformation field, e.g., points inside the different tetrahedra can map to the same image location.

The problem of moving meshes has been previously addressed in the literature [20, 178]. The proposed solutions suggest dynamic remeshing of the domain to prevent inversion of the elements. However, this is not an option in case of registration. Remeshing would require modifications to the original registration algorithm, because of the stiffness matrices  $\mathbf{K}$ ,  $\mathbf{S}$  and  $\mathbf{H}$  will need to be regenerated. In this work we consider the construction of the mesh suitable for registration computations without changing the registration algorithm.

In the next Section we overview the existing approaches to mesh generation, and consider their applicability for the non-rigid registration method.



**Figure 3.2:** A hierarchy of the studied approaches to mesh generation.

### 3.3 Related Work

An intrinsic difficulty of generating meshes from the binary image data is the processing and recovery of the object geometry. General-purpose mesh generators used in CAD applications expect that the object boundary is parametrized, i.e., it is defined by means of constructive solid geometry primitives, or explicitly (e.g., through the boundary discretization, as a collection of patches). Therefore, in order to convert the binary image into a tetrahedral mesh, one can either (1) recover the surface of the object in such a form that is acceptable by the traditional methods for volumetric meshing, followed by a conventional mesh generation process, or (2) use a mesh generation method, which operates directly on the binary image. We present the taxonomy of the available mesh generation categories in Figure 3.2.

The traditional approaches to tetrahedral mesh generation can be classified into three groups: (1) constrained Delaunay, (2) Advancing Front, and (3) Adaptive Space-Tree

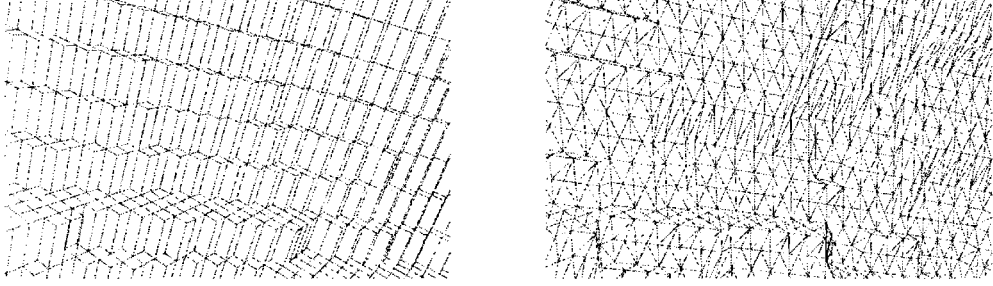
meshing. We define the term *space-tree* as a generalized regular subdivision of space, which includes octree and lattice subdivisions, adopting the terminology used previously by Bader et al. [18]. We discuss these methods in more detail in the subsequent sections, and refer the reader to the survey by Owen [152] for a detailed description of the traditional approaches to mesh generation.

Most of the practical mesh generation methods used in CAD are capable to construct volumetric meshes from faceted representation of the surface. Specifically, watertight surface triangulation is a commonly acceptable input for volume mesh generator. Thus, methods for constructing triangular approximations of the implicitly defined surfaces  $\Sigma$  are of particular interest in the context of meshing image data. Some of the requirements for the recovered triangular surfaces approximating  $\Sigma$  are the following: (1) same topology as  $\Sigma$ , (2) small (bounded) approximation error with respect to  $\Sigma$ , (3) good shape of the triangles in the surface discretization, as defined by the triangle aspect ratio, and (4) minimum number of triangles. The first two requirements are related to the accurate representation of the object, while the other two are essential for the quality of elements during volume mesh generation.

A straightforward approach to recover an iso-surface is by means of the Marching Cubes (MC) algorithm [129]. However, the original version of this algorithm may produce a triangulation with topological problems [144]. Another practical concern is the inability of the MC algorithm to generate adaptive surface triangulations, i.e., the size of the surface triangles is about the same independent of the local surface feature size. The surface produced by MC has “jagged” artifacts because of voxel sampling, which may create subsequent problems with the simulation [62, 32]. Surface mesh simplifica-



tion, or remeshing, is often a necessary post-processing step when MC is used [67, 39]. Topological problems of the surface recovered with Matching Cubes have been addressed in the recently proposed modifications of the original MC method [144]. Although surface triangulations constructed with the MC method have a number of problems, wide availability of its implementations as well as its speed made MC the method of choice for surface reconstruction in a number of studies, e.g., see [53, 38].



**Figure 3.3:** Left: Voxel surface extracted from a binary image. Right: Triangulated surface approximating the isosurface of the mask as recovered by Marching Cubes algorithm.

Triangular surface representation can also be recovered by more advanced techniques, which provide theoretical guarantees about the recovered surface. Since the seminal work of Amenta and Bern [6], a number of algorithms have been proposed for solving the problem of robust surface reconstruction from sparse point sample [123, 66, 31]. In order to provide theoretical guarantees, such methods expect that the sample is sufficiently dense. The definition of sufficient density is related to the surface feature size. Most of the available methods rely on the refinement of restricted Delaunay triangulation that approximates the surface represented by the finite set of sample points. A number of methods have been presented that ensure topological and geometric closeness of the recovered surface to the surface represented by the sample. The methods that have been developed to date differ in their performance, robustness to noise and theoretical

guarantees. Some of the methods have been implemented in publicly available packages. In particular, `Dellso` package by the group of Tamal Dey implements the isosurface meshing algorithm described in [66], and `CGAL` contains the implementation of the method by Boissonnat and Oudot [31].

Recently, a number of methods have been proposed that build volumetric mesh directly from the implicit representation of the surface [35, 125, 150]. These methods deserve a separate category in the classification, because they do not require simplicial surface representation, but at the same time their use is not restricted to the case of meshing image data. An approximation of the implicit function describing the object is easy to obtain from a binary image. Therefore, methods based on meshing implicit surfaces are directly applicable to meshing image data. Molino et al. [35] suggested the use of body centric cubic lattice with subsequent adaptive refinement and adjustment of the vertices to fit the zero level set of the implicit function. Oudot et al. [150] proposed a Delaunay meshing approach that combines surface recovery with volume tessellation and refinement. Recently, Labelle and Shewchuk [125] proposed a lattice-based method that operates on implicitly-defined surfaces and generates meshes with the theoretical guarantees on the minimum dihedral angle. None of these methods has been implemented and made available for evaluation. Meshing volumes bounded by implicit surfaces as the research area is still in its early stages with the few practical results available. A notable exception is the method developed by Persson and Strang [158] for meshing implicit surfaces, which is accompanied by a MATLAB implementation. However, this technique relies on the Delaunay refinement, which in three dimensions is known to be susceptible to generate sliver elements (nearly flat tetrahedra with the

volume close to zero). In order to be practical, methods that use Delaunay refinement methods must be accompanied by some sliver elimination procedures.

Overall, most of the methods used to construct finite element meshes rely on recovery of the triangular surface followed by volumetric meshing. A representative processing pipeline was described by Cebal and Lohner [41, 40]. The authors apply different levels of smoothing and improvement of surface mesh prior to the application of the Advancing Front method.

While surface recovery followed by conventional mesh generation is an effective way of constructing meshes from medical images, a number of direct approaches were proposed. The methods which operate on images rather than on the parametrized boundaries can be further separated into the following two categories, as shown in Figure 3.2.

The methods from the first category do not require segmentation and create meshes from the multi-value image data [88, 208]. The assumption made in such methods is that the pixels that correspond to the same tissue have similar intensity, and the object boundary can be defined with sufficient accuracy by some isosurface value. Given this assumption, these methods attempt to minimize the error of approximating the isosurface, while maintaining good quality of the mesh tetrahedra. Such approach to mesh generation is practical for volume rendering and certain FEM applications. It can also be acceptable when the organ is easily detectable by intensity thresholding, e.g., this is the case for bone segmentation from CT image data. However, brain segmentation from MRI is a challenging problem in image segmentation and can not be solved by thresholding only [55, 207]. Tissues that do not belong to the brain may have similar intensity patterns to the intensity of the brain tissue. Direct isosurface-based meshing of

the brain volume from the multivalued image may lead to large errors in the recovered surface of the object. Due to this reason mesh generation methods that operate on isosurface extracted from a grayscale image are not suitable for meshing brain volume from unsegmented MRI.

The second category of the image-based mesh generation methods operate on binary images, produced by specialized image segmentation algorithms. Among the methods in this category we separate three groups.

The methods in the first group construct the volumetric mesh with the surface that approximates the boundary of the segmented object at the level of voxel resolution [41, 102, 193]. Therefore, we define those techniques as *voxel-based meshing*. More than a decade ago, Frey et al. [88] introduced one of the first techniques for voxel-based meshing. Frey et al. proposed to construct tetrahedral tessellation of the voxelized volume by subdividing each individual voxel in a consistent manner to avoid hanging mesh nodes. Consequently, the resulting tetrahedral meshes has the number of tetrahedra in the same order as the number of voxels in the input data without the flexibility to control the size of the mesh. Hartmann and Kruggel [122] propose an approach that is based on hierarchical subdivision of the labeled voxel volume, with larger tetrahedra corresponding to the areas distant from the surface. Archip et al. [10] describe a technique that is based on Delaunay triangulation of the points corresponding to the centers of surface voxels from the segmented images. The resulting mesh is a convex hull of the input domain, which is subsequently adapted to achieve surface conforming tessellation by culling the outside mesh elements.

The main concern in using voxel-based meshes is the inability to control the overall

mesh size as well as local size of the mesh elements. The sizes of surface triangles in the meshes constructed using this approach are comparable with the size of voxels, i.e., the surface triangulation is not adaptive. The use of such techniques is problematic for time-critical applications, because the control over the element size is limited due to the fixed high resolution of the surface discretization.

A large number of methods that are based on space-tree decompositions have been proposed recently for meshing binary images. The advantage of these techniques over voxel-based meshing methods is in the ability to control the number of the mesh elements. Conceptually, the approaches based on adaptive space-trees have a long history in traditional mesh generation. Yerry and Shephard [206] were some of the first to present an octree-based approach to 3-d mesh generation. Mitchell and Vavasis [135] describe a quadtree-based algorithm with theoretical bounds on the mesh size. These ideas have been adopted to construct tetrahedral meshes from binary images [74, 137, 72, 192]. Such methods recover the surface by finding the points of intersection of the adaptive space-tree with the surface of the object defined as a binary image. Mesh quality near the surface can be compromised, as the newly inserted mesh nodes can be arbitrarily close to the existing nodes. Mesh optimization [85] is commonly used as a post-processing step. In practice, the methods from this group are well-suited for meshing binary images, and were shown to be quite effective for a number of medical applications. Some of these methods were designed and evaluated on the segmented brain MRI data [74, 137, 72]. Unfortunately, few of the published methods for image-to-mesh conversion are accompanied by their implementations. In addition, such methods are usually not evaluated in the context of their ability to build the mesh following the predefined sizing function.

The methods based on surface matching use a template volume mesh, which is warped to match the surface of the modeled object [38, 19]. While the advantages of this approach are good surface fidelity, control over the mesh size, and high speed, the quality of the elements undergoing deformation during warping can be compromised. Mesh optimization is a commonly used post-processing step for the methods in this group. This concept is most suitable for meshing objects that have similar geometries. The geometry of the intra-cranial cavity (ICC) is quite similar between different subjects. However, the mesh element sizing depends on criteria R4-6, which are patient-specific. Therefore, it is not feasible to construct a single template mesh for our application to satisfy patient-dependent point distributions.

Overall, we observe that a great variety of methods for tetrahedral meshing of binary images have been developed. Most of these methods were proposed and evaluated in the context of their fitness to a specific application. Little or no attention is usually paid to the comparison of the newly proposed techniques with the existing methods, and few implementations are available to conduct such an evaluation for a new application. Also, most of the effort is usually directed to developing a method that delivers good practical results, and not on establishing theoretical guarantees about the produced meshes. The lack of such guarantees makes it even more difficult to select the most appropriate algorithm from the range of seemingly similar methods.

Given the considered NRR algorithm, we derived the requirements to mesh generation, see Section 3.1. These requirements can now be used to customize and evaluate readily available, established methods to address the problem of mesh generation for the NRR application. The results of such evaluation can be used next to identify prob-

lems within the existing approaches and justify the development of new mesh generation methods for this application. However, before such an investment is justified, the possibility of using existing off-the-shelf tools must be carefully examined.

## 3.4 Open Source Mesh Generation Tools

### 3.4.1 Piecewise-Linear Surface Recovery

Traditionally used methods and tools for tetrahedral mesh generation operate on parametric or faceted representations of the domain surface. In order to leverage these classical approaches to mesh generation for image-to-mesh conversion, the faceted surface representation needs to be recovered as a first step of processing. Due to reasons discussed earlier, simple approaches, like Marching Cubes, are not suitable for solving this problem.

In general, the piecewise-linear triangulation of a surface should have the following desirable characteristics [66]:

1. Topology: the recovered surface should be *homeomorphic* to the original surface.

In simple terms, if two shapes are homeomorphic, they can be transformed one into another by means of continuous bending and stretching [203] (e.g., the shape of a donut is homeomorphic to the shape of a one-handled cup, but not to the shape of a pretzel or a ball).

2. Accuracy: the approximated surface should be sufficiently close to the original surface. Most common measures of the surface approximation accuracy are based on *Hausdorff distance* and angle between normals of the surface and its approxi-



**Figure 3.4:** Approximation (solid line) of the true surface (dotted line) with high (left) and low (right) error of normal approximation [147].

mation. Let  $X$  be the set including all points on the original surface,  $Y$  – the set of points on the approximated surface, and  $d(x, y)$  – the distance between points  $x$  and  $y$ . Then

$$d_H(X, Y) = \max\left\{\max_{x \in X} \min_{y \in Y} d(x, y), \max_{y \in Y} \min_{x \in X} d(x, y)\right\}$$

is the Hausdorff distance between  $X$  and  $Y$ . As we show in Figure 3.4, good approximation error defined with the Hausdorff distance may not be sufficient, depending on the specific application. As an example, smooth surfaces that closely approximate true surface normals are important in surface rendering applications.

3. Quality: triangles of the approximated surface should have good quality, as measured by minimum angle. In general, skinny triangles with angles approaching  $0^\circ$  are not desirable.
4. Optimality: the number of triangles in the approximated surface should be minimized for the given approximation accuracy.

In this work we used the implicit surface meshing approach proposed by Oudot et al. [31] to approximate the surface of the segmented binary mask. This method is available in the public domain as implemented in CGAL, Computational Geometry Algorithms Library [42]. Next we introduce some basic definitions from the field of



computational geometry, and give a brief description of the surface meshing algorithm we used.

The idea of the method takes its origin from the notion of  $r$ -sample, which was introduced by Amenta and Bern [6]. Following the notation by Oudot et al. [31],  $r$ -sample  $E$  of a surface  $S$  is a set of points that is sufficiently dense with respect to the distance to the *medial axis* of  $S$ . The property of  $r$ -sample is that no point  $p$  on  $S$  is farther away from  $E$  than the value of local feature size at that point  $r \cdot LFS(p)$ . In three dimensions, medial axis is defined as a locus of the centers of spheres that are tangent to the surface  $S$  in two or more points, e.g., this is a set of point that are equidistant from at least two points on the surface. Given the definition of the medial axis, the Local Feature Size at point  $p$ ,  $LFS(p)$ , is defined as the distance to  $p$  from the nearest point of the medial axis [6].

Delaunay triangulation (or tetrahedralization, in three dimensions) is such a triangulation of points that no point is contained in the circumsphere of any of its tetrahedra. The key result achieved by Amenta and Bern [6] was that with the sufficiently small values of  $r$ , the Delaunay triangulation of  $E$  contains a piecewise-linear surface  $T$ , which has a number of desirable properties relating it to  $S$  [31]:

- $T$  is homeomorphic to  $S$ .
- the error between the normal of each facet in the piecewise-linear surface  $T$  and the normals at the facet points, which belong to  $S$ , is bounded by  $O(r)$ .
- the Hausdorff distance between  $T$  and  $S$  is  $O(r^2)$ .

In practice, the use of the  $r$ -sample notion is not trivial, since it is difficult to con-

struct, and for a given sample of surface points, it is hard to check whether the sample satisfies the requirements of an  $r$ -sample. Oudot et al. [31] introduced the definition of *loose  $\epsilon$ -sample*. Given loose  $\epsilon$ -sample  $E$ , the authors show that restricted Delaunay triangulation of  $E$  is a good approximation of  $S$  in topological and geometric sense. Moreover, they present an algorithm, that allows to construct such a sample of points and the corresponding triangulation for a surface defined by an oracle. Given any line segment, this oracle should return the points of its intersection with the original surface  $S$ . The algorithm is guaranteed to terminate, and allows to specify the bound on the triangle angle in the output triangulation to avoid skinny facets.

The definition of the original surface by means of oracle is quite convenient while meshing iso-surfaces in images. Given a distance map constructed from the binary image, for any line segment we can approximate the intersection point of this segment with the zero level set by iteratively bisecting the segment and evaluating the distance value at the bisection point until some tolerance is reached.

The piecewise-linear surface that can be constructed by using the technique by Oudot et al. can be used to define the input domain for the established mesh generation tools we present next.

The advantages of using the discussed method for implicit surface triangulation are:

- Strong theoretical guarantees provided by the algorithm.
- Robust open source implementation, which is easy to adjust to work directly with the binary image data.
- Well-documented and maintained code base, with the improvements introduced

regularly.

- High performance of the code, with the surface meshes of typical brain ICC generated within seconds.

Still, there are problems one has to face while using this approach. First, surface mesh generation is separated from volume mesh construction. Second, the implementation is not flexible to provide control over surface mesh sizing, which may create difficulties in adjusting to user-defined element size distribution during volume meshing.

### 3.4.2 Delaunay Meshing

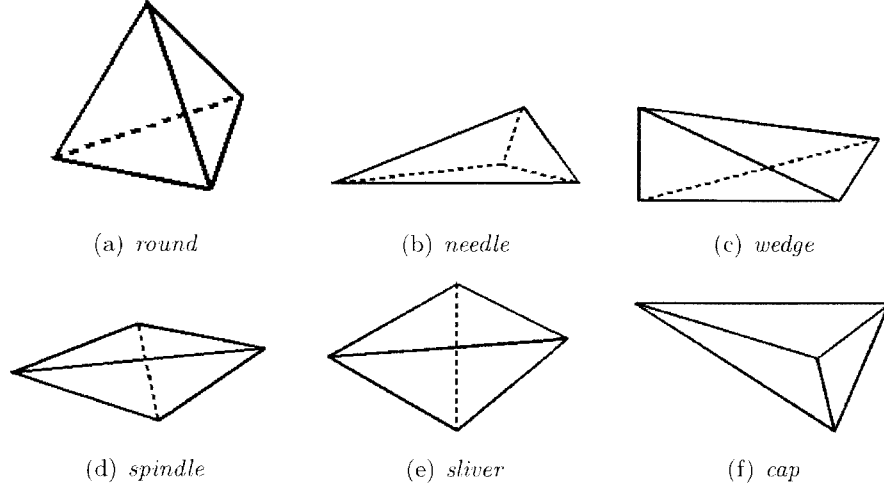
Triangulation of a point set in three dimensions defines such a subdivision of the convex hull enclosing these points into tetrahedra that the intersection of any two tetrahedra is either empty or contains a common face, and the set of the vertices of all the tetrahedra coincide with the input point set. Delaunay triangulation is a specific type of triangulation, which satisfies the Delaunay criterion: the circumsphere of each edge, face and tetrahedron in three dimensions is empty. Delaunay refinement is a procedure of inserting new points into an existing triangulation in such a way that the Delaunay criterion is maintained [93]. It was shown, that insertion of the new points, which are called *Steiner* points, during Delaunay refinement at certain well-defined locations results in introducing mesh elements of improved aspect ratio. The traditional Delaunay refinement algorithms insert Steiner points at the center of the circumsphere of the ill-shaped element [176]. Some of the recently introduced modifications to the insertion procedure use alternative strategies to provide more flexibility in the mesh refinement process [45, 194].

Mesh refinement is a procedure of mesh adaptation that is commonly required in numerical solution schemes to adjust the mesh according to the *a posteriori* error estimation. The reason why Delaunay refinement has been attracting so much attention in the mesh generation community is that the mesh produced by Delaunay refinement enjoys a number of useful theoretical properties. Specifically, the circumradius to shortest edge ratio can be bounded. In two dimensions this corresponds to the bounds on the minimum angle, preventing the numerical problems that can be caused by nearly flat elements [177]. Delaunay meshes are also known to be size-optimal, and the process of refinement is guaranteed to terminate [176]. Yet another desirable property of the Delaunay refinement is in its amenability to execution in parallel [47], which is essential for memory-limited and time critical mesh refinement applications.

In three dimensions, Delaunay refinement does not guarantee bounds on the minimum dihedral angle. Most of the ill-shaped elements shown in Figure 3.5 can be eliminated by Delaunay refinement, but not slivers. Extreme values of dihedral angles in tetrahedral elements lead to high gradient interpolation errors and poor conditioning of the stiffness matrix, which can affect the solution accuracy and convergence of iterative solvers [177].

The specific type of Delaunay triangulation that is of practical interest for FEM applications is Constrained Delaunay Triangulation (CTD). Unlike pure Delaunay triangulation, CDT preserves object boundaries, maintaining the surface approximation accuracy. Existing CDT algorithms have limitations on minimum dihedral angle in the object boundary and require its piecewise-linear or parametric definition [46, 175, 44].

We use a specific algorithm and implementation of constrained Delaunay triangu-



**Figure 3.5:** Good- (a) and ill-shaped (b-f) tetrahedra [25].

lation and refinement in three dimensions proposed by Si [179]. Compared to other existing Delaunay refinement methods, this algorithm presents a number of practical innovations. First, it does not impose restrictions on the minimum input angle at the expense of the guarantees provided. Second, the method has been designed specifically to allow the use of background mesh, which governs the insertion of new mesh points. Finally, it includes post-processing step that attempts to eliminate slivers that could be introduced in the mesh during refinement.

The algorithm has been implemented by Hang Si in the *TetGen* software [180], which we evaluate in the subsequent sections. The implementation of Delaunay refinement in *Tetgen* provides significant flexibility in controlling mesh size, which is important for realistic modeling applications. Mesh size can be adjusted spatially by constructing application-specific sizing function. The total number of mesh elements can be controlled in a number of ways. First, the method allows to specify  $\alpha$  parameters that essentially impose limits on the maximum density of the mesh vertices locally. Second,

the algorithm allows to bound the maximum volume of the tetrahedron. Therefore, it is possible to control tetrahedron volume even without specifying the background mesh.

### 3.4.3 Advancing Front Meshing

Advancing Front Technique (AFT) is a mesh generation heuristic which builds mesh iteratively starting from the triangulated boundary of the domain [128, 131, 164]. The placement of newly inserted points is optimized with respect to the existing mesh elements to improve the quality of the new elements. The task of optimum point placement can become particularly difficult when the meshing fronts advancing from the opposite boundaries are close to each other. In such cases, AFT may revert to discarding parts of the previously generated mesh layers to resolve the conflict, resulting in possible difficulties reaching termination of the meshing process.

AFT cannot directly construct volumetric discretization of a domain defined by parametrized surface patches. In the cases when only the parametric surface definition is available, surface triangulation must be constructed prior to volume meshing. Quality of meshes produced by AFT depends on the surface mesh quality. While AFT does not give any guarantees about the final mesh, in practice this method has been shown effective for a number of engineering applications. This is explained in part by the lack of bounds provided by the Delaunay methods on the minimum dihedral angle in three dimensions. In practice, AFT methods may deliver meshes with better angle distribution and fewer elements compared to Delaunay refinement methods. The reason is that while the Delaunay refinement produces meshes of optimal size, the constant involved in the theoretical bound may be too large for practical applications. The AFT

approach does not introduce any new points on the surface which makes it perfect for parallelization with splitting the original object domain in multiple subdomains. However, the same property makes the method inherently inflexible in constructing meshes with user-defined sizing. Surface triangulation must be constructed with this sizing distribution taken into account.

The *NETGEN* approach proposed by Shöberl in [173] is a representative algorithm from the broader class of the advancing front methods. Unlike the Delaunay-based *Tetgen* implementation we considered in the previous Section, this mesh generation tool is flexible in the definition of the domain boundary. Both the triangulated surface mesh and parametric surface representation are acceptable inputs. The difference in processing is that the surface defined parametrically will be triangulated by *NETGEN* as part of pre-processing. The major distinction of *NETGEN* from the similar AFT techniques is in the attempt to generalize the rules used to construct new elements, and define them in such a way that the same rules can be applied for two-dimensional meshing, three-dimensional surface and volume meshing.

The algorithm implemented in *NETGEN* organizes the construction of the mesh by maintaining the vector of the current boundary elements ordered by their quality and distance to the original domain boundary [173]. Point insertion is governed by the abstract rules that define the location of the new point depending on the configuration of the boundary element and its so called *environment*, which encloses the neighboring elements. Rules in two dimensions generate one point at a time, or connect existing points in the mesh. These rules are directly applicable both for constructing two-dimensional planar or surface triangulations. Similarly to *Tetgen*, *NETGEN* provides control over

the size of the elements in the mesh. This can be done by specifying mesh sizing at the defined points and controlling the maximum edge bound. The source code of *NETGEN* tool is publicly available [143].

### 3.4.4 Lattice-Based Meshing

As an alternative to using the traditional methods that require faceted surface approximation, we developed a novel tool that constructs tetrahedral mesh directly from the binary image describing the domain. The mesh generator is based on the approach presented by Molino et al. [34, 138]. This meshing method was originally developed for simulating large scale deformations. We adopt this approach for a different application of meshing binary images, and develop its customized implementation.

The approach we develop can be classified as space-tree based mesh generation technique. The advantage of this method over other space-tree approaches is in the choice of the lattice: instead of using adaptive octree, as in [74, 136, 26], body-centric cubic lattice (BCC) is used for the construction of the initial domain discretization. This lattice results in significantly better quality of the initial tetrahedral tessellation, has Delaunay properties and consists of tetrahedra which differ from the equilateral tetrahedron as little as possible with regular space tiling [90].

Another useful feature of the considered technique is that it does not require triangular mesh of the domain boundary. Instead, the object is represented by a function, absolute value of which gives the distance to the object surface for any point in space. The distance value is negative, if the point is located inside the object volume, and positive otherwise.



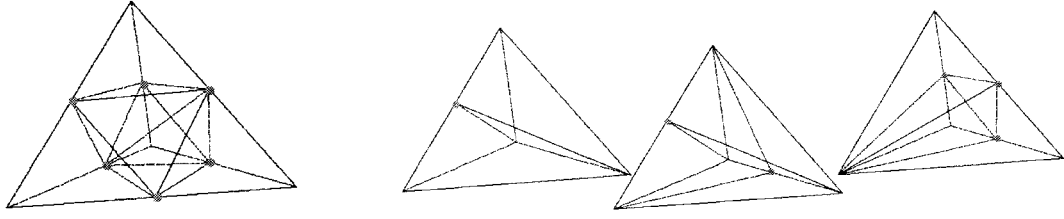
Next we summarize the processing steps of the developed mesh generation method.

**Step 1: BCC generation** The coarse BCC lattice is constructed to completely cover the binary mask. The BCC initialization is done by overlapping two cubic lattices, with the vertices of the second lattice being the centers of the cubes in the first lattice. The BCC is constructed by introducing a new set of diagonal edges that connect each vertex in the first lattice with the vertices of the containing cube in the second lattice. Each tetrahedron contains two vertices from each of the lattices, where the same-lattice vertices are connected with the lattice-orthogonal edges, and vertices from the different lattices are connected with the diagonal edges.

The size of the lattice is controlled by the spacing of the initial Cartesian grid used in cubic lattice construction. The lattice tetrahedra are all identical up to translation and rotation, and satisfy the Delaunay property. A tetrahedron is discarded if all four of its vertices are located outside the meshed mask.

**Step 2: BCC *red* refinement** The lattice is refined following the application-defined criteria. The elements which are selected for subdivision (e.g., those which cross the object surface) are subdivided following the 1:8 rule (see Figure 3.6). Each of the edges of the tetrahedron are split by the mid-point, and eight new *children* tetrahedra are formed from the initial tetrahedron (*parent*) following the template. We use the shortest inner diagonal during the subdivision, therefore the subdivision pattern is stable [138], i.e., subsequent *red* subdivisions of the *children* tetrahedra will not lead to diminishing quality.

Conformity of the mesh surface to the surface of the segmented mask is enforced by subdividing the tetrahedra that have vertices located both inside and outside the binary



**Figure 3.6:** Subdivision templates used in *RGM* mesh construction: *red* (leftmost) and three *green* templates.

mask. The tetrahedra obtained by *red* subdivision that have all four vertices outside the binary mask are discarded.

**Step 3: BCC *green* refinement** Following the *red* refinement, the tetrahedra which share the *red*-split edges, but were not subdivided, will contain *hanging nodes*. Hanging nodes are the vertices introduced at the midpoint of an edge in one element, but are not valid tetrahedron vertices at those elements [211]. The mesh which contains such hanging nodes is not conforming, and introduces complications during FEM solution. Generally, non-conforming meshes are not desirable, therefore we perform additional refinement in order to eliminate hanging nodes from the mesh using the *green* subdivision templates. The set of templates is shown in Figure 3.6. If the split edge configuration does not correspond to either one of the existing templates, all of the edges are split, and the *red* subdivision template is applied.

Application of the *red* refinement template during the *green* refinement stage results in the new hanging nodes introduced into the mesh. Therefore, multiple iterations of *red* and *green* refinement may be required.

Steps 2 and 3 (*red-green* refinement) are repeated until the application-defined mesh resolution is achieved. It is important to keep the mesh quality bounded. This is why during the iterative refinement only the elements produced as a result of *red* subdivision

can be subdivided. In case an edge has to be split for a *green* element, the whole group of *green* siblings is substituted with their parent, which is then *red*-refined.

**Step 4: Candidate mesh selection** This step of the algorithm prepares the mesh for the final processing stage. Once again, the elements which are located completely outside the object surface set are discarded. Next, a set of tetrahedra is selected, such that for each edge of a tetrahedron in the set at least 50% of the edge is located within the object volume. The vertices of those tetrahedra form the set of *enveloped* vertices. The mesh elements which do not include at least one of the enveloped vertices are then discarded. This heuristic proved to work well in practice to improve the final mesh quality. At this point, the mesh topology is finalized. The mesh elements are identical up to scaling, translation and rotation to one of the element configurations formed by *red-green* subdivision, i.e., the candidate mesh has guaranteed quality.

**Step 5: Surface boundary compression** The candidate mesh roughly approximates the object surface, but the surface vertices are not aligned with the object boundary. The goal of mesh boundary alignment, or compression, is to improve the boundary match<sup>1</sup>. A straightforward approach would be to project the candidate mesh surface vertices to the surface of segmentation mask. It is possible that some tetrahedra may collapse as a result of such operation, and create an invalid mesh. The quality of the mesh following boundary compression may also be compromised with arbitrarily small dihedral angles. Therefore, more elaborate method should be applied.

Instead of snapping the boundary vertices to the surface, the nodes are iteratively moved towards the segmentation surface guided by the distance map. In order to avoid

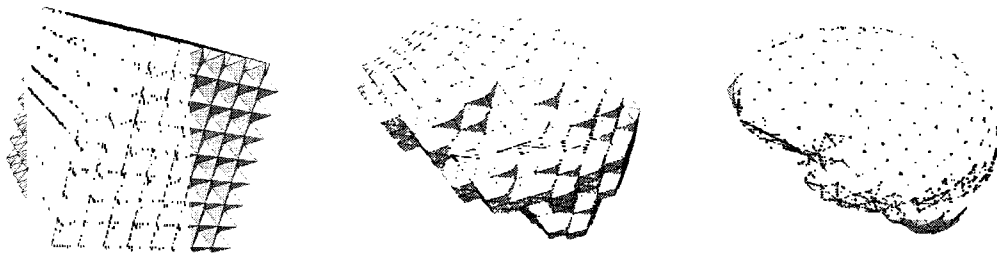
---

<sup>1</sup>Here, *boundary compression* should not be confused with compression for reduction in size, as in data compression. We use the terminology following the original description of the technique in [138]

drastic element flipping and quick deterioration of element quality, the following heuristic is used. A displacement vector is defined for each surface vertex at each iteration of boundary compression. The surface node displacement vector direction is defined by the *vertex normal*, i.e., the averaged normals of the faces incident on a particular mesh surface node. The displacement vector is scaled by the distance to the zero level set of the distance map. During the first 10 boundary compression iterations the vertices are moved by 20% of the distance to the surface, and during the subsequent iterations the displacement vector is scaled by the full distance value.

Surface compression inevitably leads to deformation of the mesh elements near the surface. Two methods have been implemented to improve the volume mesh quality. The first technique is physics-based. It models deformation of the candidate mesh using the linear elastic material model [211]. The reader is referred to [74, 166] for the formulation details. In summary, we use candidate mesh discretization as the model, and prescribe the boundary forces acting on the surface vertices based on the distance to the segmentation surface. The deformation of the linear elastic body calculates the positions of the model vertices given the boundary forces. These positions are calculated as the solution for linear system of equations  $F = Kx$ , where  $F$  is the vector of forces,  $x$  is the displacement vector, and  $K$  is the stiffness matrix calculated based on the mesh and the prescribed material properties [211].

An alternative approach which we implemented applies local mesh smoothing and optimization to the mesh after each surface compression iteration. The implementation has been integrated with both GRUMMP [148] and Mesquite [33] mesh optimization packages. After each boundary compression iteration, the locations of the mesh vertices



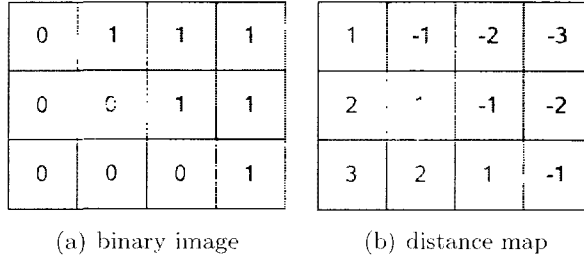
**Figure 3.7:** Processing steps of the RGM mesh generator. Left: initial BCC lattice. Center: tetrahedral mesh surface after discarding outside elements. Right: surface of the mesh with the adjusted vertex locations.

are updated to improve the shape of the mesh elements (in case of GRUMMP, the element connectivity can also be automatically modified if this results in mesh quality improvement). We present the mesh at different stages of processing in Figure 3.7.

The rest of the document uses the following notation and abbreviations: Red-Green Mesher (*RGM*) for the implemented mesh generation technique, *RGMp* for *RGM* with physics-based compression, and *RGMo* for optimization-based candidate mesh boundary compression.

We have implemented the presented method as two templated classes which use the Insight Toolkit (ITK) [120, 109] design patterns. These classes are now part of NAMIC SandBox [141].

The implementation expects the segmented binary voxel mask as input. The first of the classes, `BinaryMaskTo3DAdaptiveMeshFilter`, implements Steps 1 through 4 described above. Mesh generation and refinement is guided by the signed discrete distance transform of the input binary mask. The signed distance transform assigns integer valued Euclidean distance to the closest boundary voxel for each input voxel. An example of distance transform calculation for a small image in two dimensions is shown in Figure 3.8. The distance value is negative for the voxels located inside the object volume.



**Figure 3.8:** Signed distance transform example. Grayed squares correspond to the pixels inside the object, and the distance map contains the distance to the binary object boundary.

and positive otherwise. The distance value at a specific point in space is calculated by linear interpolation from the neighboring voxels.

The core of the first of the implemented classes is the data structure that maintains mesh connectivity. The **Vertex** field contains vertex coordinates. Vertices are shared and referenced by pointers in the referencing edge and tetrahedron data structures. The **Edge** keeps pointers to the two endpoints, and the pointer to the midpoint, which is initialized when the edge is subdivided. Each edge also keeps a list of pointers to all adjacent tetrahedra. The structured nature of the refined lattice limits the maximum number of tetrahedra which can share a particular edge, thus the time required to traverse all tetrahedra neighbors is bounded by a constant. The **Tetra** (tetrahedron) data structure keeps pointers to the six edges, the type of subdivision (e.g., *red* or *green*, which edges are subdivided), the pointer to the list of children tetrahedra (if a tetrahedron was *green* subdivided), and a pointer to the parent tetrahedron (if exists). The data structures were designed to facilitate refinement and minimize memory requirements. During the *red* stage of iterative refinement, a list of non-conforming tetrahedra is maintained and updated during the *green* stage, until a conforming mesh of the desired resolution is constructed.

The second filter, `VolumeBoundaryCompressionMeshFilter` takes as an input non-uniform mesh that does not conform to the domain, and applies physics- or optimization-based surface compression procedure to compress the mesh surface to the binary mask boundary. The physics-based approach to boundary compression requires solution of a system of linear equations, which is done with *PETSc* [12].

The following advantages make the RGM method suitable for image-to-mesh conversion in the context of the NRR application:

- The BCC lattice tetrahedra are as close as possible to equilateral tetrahedron, providing significant improvement in element shape as compared to the elements generated by tessellating an adaptive octree, thus we start from a better initial discretization as compared to octree-based methods previously suggested [74, 192, 26, 137].
- The *red-green* refinement procedure creates limited number of element configurations and does not allow significant propagation of the refinement.
- The mesh boundary compression stage is independent of the lattice refinement procedure and can be easily substituted with a different implementation. Any of the methods suggested for surface recovery in octree-based techniques are potentially applicable.
- The method was designed specifically for modeling large deformations, which is important for registering intra-operative images during IGNS, where significant deformations can occur.

- Lattice-based topology and structure of the mesh gives bounds on mesh connectivity and can be advantageous to FEM applications (note: this is applicable only for *RGMp*, because *RGMo* is likely to change the mesh topology);
- *red green* refinement can possibly be used to re-mesh local regions of the mesh without remeshing the whole object (*RGMp* only).
- Lower rate of quality degradation for physics-based vs. optimization-based boundary compression has been reported [138] (the experimental study we have performed did not confirm this observation for the evaluated data).
- The method does not require explicit object boundary representation, thus there is no need to develop a separate technique for triangular surface recovery and/or remeshing, which is necessary for CDT and AFT methods.
- There is trade-off between the volume mesh quality and the accuracy of boundary approximation. The volume mesh before the boundary compression has perfect quality, while the boundary compression procedure is iterative and can be stopped if the quality of the volume starts to deteriorate.

Nevertheless, there are certain disadvantages of the implemented approach:

- The choice of the displacement step during boundary compression does not guarantee the surface deformation will result in topologically correct mesh (i.e., depending on the element size, there is a possibility of having an element collapse during one iteration step).



- The user must choose the initial lattice spacing and the number of mesh refinement levels.
- There are no guarantees about the aspect ratio or minimum dihedral angles in the mesh after the surface compression.
- The implementation does not allow to mesh multiple segmented tissues.
- *RGMp* does not update the mesh connectivity, and thus is more susceptible to the degraded volume mesh quality during boundary compression. Neither of the boundary compression approaches updates surface connectivity.
- *RGMp* involves the solution of the problem equal or at least comparable in complexity to the numerical computations performed by the solver component during non-rigid registration.

Overall, in the absence of any implementations for automatic image-to-mesh conversion tools in the public domain, the advantages provided by the presented implementation make it a practical choice for meshing binary images.

### 3.5 Construction of Sizing Function

In general, applications that use tetrahedral meshes require different sizes of elements depending on their spatial location within the domain. The time required for NRR is directly proportional to the number of points in the mesh, while the solution error is reduced with higher discretization of the domain. Therefore, a balance between the computation time and solution accuracy is usually required by refining the domain with

higher resolution in the areas of interest, or in the areas where the *a posteriori* error exceeds some threshold. Next we describe a way to adjust the size of tetrahedra in order to meet the application-specific requirements of the NRR method. Specifically, we describe how to adapt the element size to the distribution of the registration points.

The conventional approach to definition of mesh density is through the use of *mesh sizing function*. Given a set of points  $\mathcal{V}$ , mesh sizing function  $H$  defines the desired length of the edge at a point  $H : p \in \mathcal{V} \mapsto \mathbf{R}^+$  [179]. It can be defined analytically, but more often its values are prescribed at the vertices of the background mesh. When the background mesh is used, the value of the sizing function at the non-vertex locations can be derived by interpolation. Both *Tetgen* and *NETGEN* accept a background mesh to control local mesh size.

We use the sizing function defined at the vertices of the background mesh to adjust the distribution of mesh element sizes to the distribution of the registration points for application-specific mesh generation. Note, that the background mesh does not need to be conforming to the object boundary. The only requirement is that the meshed domain should be completely enclosed within the background mesh to allow interpolation of the sizing function at any point within that domain [211].

We use the same CGAL-recovered surface mesh for the construction of both background mesh and the mesh used for NRR. The background mesh is constructed using *Tetgen* by setting the bound on the maximum tetrahedron volume. The bound is selected in such a way that the maximum tetrahedron volume corresponds to an equilateral tetrahedron with the edge equal to the average edge length in the CGAL-reconstructed surface mesh. The sizing value at each mesh vertex is initialized with the distance to the

$k$ th closest registration point to reflect the density of the registration point distribution.

The motivation behind the initialization of the background mesh is that the shape of a perfect mesh vertex cell complex is close to a ball with the radius prescribed by the background mesh. We use CGAL [42]  $k$ -neighbor search to find the  $k$  closest registration points and the distance to the furthest point for each background mesh vertex. Our goal is to have around 30 registration points in the cell complex of each vertex. However, based on the experimental results, the mean value in distribution of the registration points both for *Tetgen* and *NETGEN* was not approaching the desired bound when we set  $k = 30$ . Experimentally, we arrived at a result that the best distribution is obtained by using larger values for  $k$  (we used  $k = 100$ ), and adjusting the *Tetgen* mesh by reducing the *alpha* parameters of the implementation, see [179]. The *NETGEN* mesh was constructed using the same background mesh, but the sizing values were scaled down by constant to have similar number of nodes compared to the adaptive *Tetgen* mesh.

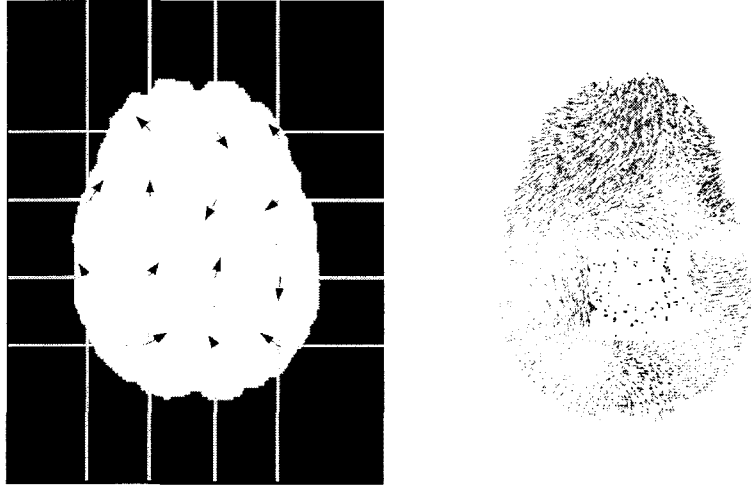
The sizing of the adaptive mesh constructed with *RGM* was controlled by a custom subdivision function. This function is called for each tetrahedron during the refinement at each mesh resolution, and returns *true* if the tetrahedron requires subdivision. We calculate the number of registration points inside the cells of the four tetrahedron vertices. The tetrahedron is subdivided if the number of the registration points in each vertex cell exceeds point density threshold  $p_t$ . We used two subdivision resolutions in all cases. The process of adaptive mesh construction with *RGM* is parametrized by the value point density threshold and the spacing of the initial lattice, see [72]. These parameters were experimentally chosen so that the size of the adaptive mesh (the number

of mesh vertices) is approximately the same as the size of the adaptive *Tetgen* mesh.

### 3.6 Evaluation Framework

The evaluation of the mesh generation techniques consists of the two components. First, since we introduce a new mesh generation method, (*RGM*), we need to analyze its performance in the context of the existing meshing tools. We approach this by comparing a number of quality metrics for the meshes constructed from the same binary image data using *RGM* and the conventional mesh generation tools that operate on triangulated surfaces. The commonly accepted metrics we use allow to compare shape of the mesh tetrahedra and assess the accuracy of surface approximation.

The second part of the evaluation is focused on the application-specific requirements. The specific metrics we use in the evaluation require estimation of the registration error at the registration points, which is based on the knowledge of the ground truth deformation at each of those points. The issue of validating the accuracy, i.e., establishing ground truth deformation in non-rigid registration of medical images is a difficult problem. We further discuss the issue of validation in Chapter 5. Since we cannot derive ground truth deformation from the clinical data, in the evaluation of mesh generation methods we use synthetic ground truth. We construct a synthetic deformation field, apply it to real MR images of brain, and use non-rigid registration to recover the synthetic deformation. Such a validation approach that is based on synthetic ground truth data is quite common in evaluating non-rigid registration methods [103]. The availability of the true deformation field allows us to assess the registration error and other related metrics at arbitrary image locations.



**Figure 3.9:** Left: Random deformation vectors are generated at the knots (red circles) of the deformation field grid that are located within the user-defined mask. Right: Example deformation field.

We use the method described by Rogelj et al. [167] to construct a synthetic deformation field. We first construct a sparse point sample at the knots of an isotropic sampling grid overlayed with the image. This procedure is illustrated in Figure 3.9, where sample points are shown with red circles. Deformation vector  $d_i = (d_i^x, d_i^y, d_i^z)$  is constructed at each of the grid points. Each of the vector components are drawn from a sum of Gaussian distributions, with separately initialized distributions for each coordinate component. The distributions are parametrized by the mean  $\mu = 0$  and variance  $\sigma$ . We set the distribution variance to be within the 5% of the brain dimension, and include additional checks to ensure that the resulting deformation does not violate the assumptions used in linear elastic models [63]. The dense deformation field is constructed by using Thin Plate Splines interpolation at non-knot image points within the binary mask. An example of the deformation field generated by this approach is given in Figure 3.9. The implementation of the framework for constructing the synthetic deformation field has been presented in [27].

## 3.7 Results

### 3.7.1 Evaluation of Lattice-Based Meshing

The objective of this evaluation component is to establish that the new mesh generation method we introduce is comparable with the conventionally used methods with respect to common mesh quality metrics.

Three ICC segmentations from the real patient data (from [53]) were used to construct tetrahedral meshes with the *RGM* method. Two sets of meshes were generated. In one case we perform mesh boundary compression using the physics-based method (*RGM-p*). The other set of meshes was constructed by using optimization-based boundary adaptation (*RGM-o*).

Clatz et al. [53] used the following processing steps for constructing tetrahedral meshes. First, the dense triangular surface mesh was extracted from the binary mask of the patient ICC segmentation. This surface was next decimated in order to reduce the size with the surface decimation package *Yams* [86]. The decimated mesh was then used as the input for volumetric meshing with the commercial Delaunay mesh generation package *GHS3D* [182]. We were able to obtain the original meshes used by Clatz et al. [53] for the purposes of this study, but we did not have access to the commercial tools *Yams* and *GHS3D*. The meshes used in the original paper by Clatz et al. comprised the first set of meshes used in the comparison, marked further as *GHS3D*.

The second set of meshes was constructed with another commercial mesh generation tool, *SolidMesh* [131]. This tool uses AFT combined with the Delaunay criteria for optimizing point insertion during mesh front propagation. The third set of meshes was

constructed with *Tetgen* [180], a tool we discussed earlier. In both cases we attempted to adjust the meshing parameters to obtain the number of elements as close as possible to 10K constraint specified by Clatz et al. [53].

Both *SolidMesh* and *Tetgen* tools require a piecewise-linear surface boundary as input. For these two methods we use the surface of the *RGM-o* mesh to describe the input volume domain.

First, we evaluate the quality of surface approximation for each of the methods. We compute the Hausdorff distance from the triangular surfaces reconstructed from segmented images using Marching Cubes algorithm to the surface extracted from volume meshes. We present the results of that evaluation in Table 3.1. Hausdorff distance was computed using the M.E.S.H. software [15]. Note, that we do not evaluate surface approximation for *Tetgen* and *SolidMesh* meshes, as their surfaces are identical to those of the *RGM-o* meshes.

Second, we evaluate the initial quality of element shape using aspect ratio and minimum dihedral angle. We define aspect ratio as  $\frac{|K|_\infty}{2\sqrt{6}r}$ , where  $|K|_\infty$  is the length of the longest edge, and  $r$  is the radius of the tetrahedron inscribed sphere. These quality metrics were calculated using VTK 4.4 [121] and are shown in Table 3.2. Perfect aspect ratio value is 1, and dihedral angles close to  $0^\circ$  or  $180^\circ$  and larger are considered problematic.

Finally, we evaluate the change in quality of the meshes caused by deformation. The deformed meshes were obtained as a result of running non-rigid registration on the image data that corresponds to each case. Aspect ratio and minimum dihedral angles in each of the five meshes after the deformation are shown in Table 3.3.

**Table 3.1:** Surface approximation quality.

case ID	method	number of surface triangles	Hausdorff distance		
			max	mean	RMS
1	RGM-o	1828	9.16	0.47	0.66
	RGM-p	1868	8.73	1.54	1.88
	GHS3D	1284	11.06	0.98	1.25
2	RGM-o	2000	6.48	0.55	0.75
	RGM-p	2022	7.97	1.40	1.69
	GHS3D	1284	10.37	1.00	1.34
3	RGM-o	1858	7.25	0.47	0.63
	RGM-p	1896	7.07	1.48	1.81
	GHS3D	1354	8.96	0.87	1.14

The evaluation shows that for all of the cases surface approximation for the meshes constructed by *RGM* is at least as good as for the meshes constructed by *GHS3D*. Because of the adaptive mesh structure, *RGM* meshes have larger number of surface elements with about the same number of tetrahedra, see Table 3.2. This also contributes to a better surface approximation. The element shape quality of the meshes generated with *RGM* is also comparable and in some cases significantly better than *GHS3D*, as can be seen from Table 3.2. Delaunay meshes generated with *Tetgen* have quality comparable with the *GHS3D* meshes, but they have more elements, with small volume elements near the mesh surface.

The changes of the mesh quality after deformation is most drastic for Delaunay meshes (e.g., see case 3 in Table 3.3). This is explained by the presence of small tetrahedra near the mesh surface where elements experience largest strains. On average, quality of the elements for all of the methods decreases as the result of deformation. However, for all of the methods (except *Tetgen*, where we observed large aspect ratios for cases 1 and 3) the deformed meshes preserve the quality and would probably not require remeshing if they were deformed further. We have not observed significant difference in



**Table 3.2:** Element shape quality evaluation.

<i>case</i> <i>ID</i>	<i>method</i>	<i>tets</i>	<i>aspect ratio</i> <i>min/average/max</i>	<i>min dih. angle</i> <i>min/average/max</i>
1	RGM-o	7334	1.03/1.44/2.83	33.3/52.5/77.2
	RGM-p	7565	1.02/1.36/2.56	25.8/55.6/75.6
	GHS3D	7886	1.05/1.61/11.64	6.8/47.8/81.5
	Tetgen	21514	1.04/1.97/7.34	11.4/41.7/80.5
	SolidMesh	8942	1.02/1.37/3.40	17.1/54.0/79.9
2	RGM-o	7473	1.02/1.48/4.09	30.7/52.0/79.3
	RGM-p	7556	1.02/1.40/2.75	23.6/54.8/76.7
	GHS3D	8202	1.05/1.62/6.68	11.1/47.7/83.3
	Tetgen	23907	1.04/1.97/6.24	13.8/41.8/81.8
	SolidMesh	10266	1.03/1.36/3.45	17.6/54.1/80.6
3	RGM-o	7497	1.01/1.46/3.23	32.5/52.1/79.7
	RGM-p	7743	1.01/1.37/4.27	14.3/55.4/77.0
	GHS3D	8235	1.02/1.60/19.1	3.07/47.7/83.2
	Tetgen	23173	1.04/1.98/6.37	9.0/41.6/82.2
	SolidMesh	9255	1.03/1.37/3.25	20.5/54.0/79.9

quality change between physics- and optimization-compressed *RGM* meshes.

We emphasize, that both *GHS3D* and *SolidMesh* methods are in the commercial domain, and that both *SolidMesh* and *Tetgen* require (high quality) surface mesh as input, while *RGM* is open source and operates directly on binary images. We observe that optimization-based surface compression procedure results in better quality meshes. Therefore, this is the method we will use in the subsequent experiments.

### 3.7.2 Application-Specific Evaluation

Previously, we introduced a number of application-specific requirements that we believe are important in the context of non-rigid registration. In this Section we first evaluate mesh generation techniques in the context of these metrics. As part of this evaluation, we explore the question whether the use of sizing function results in the meshes that better meet the application-specific requirements. In order to conduct the evaluation, we use

**Table 3.3:** Element shape quality evaluation following deformation.

<i>case</i>	<i>method</i>	<i>aspect ratio</i>	<i>min dihedral angle</i>
<i>ID</i>		<i>min/ave/max</i>	<i>min/ave/max</i>
1	RGM-o	1.02/1.46/2.80	25.85/52.00/78.79
	RGM-p	1.03/1.39/3.36	20.68/54.73/77.03
	GHS3D	1.04/1.65/12.97	5.39/47.33/82.66
	Tetgen	1.04/2.06/104.52	0.41/40.92/82.29
	SolidMesh	1.02/1.40/3.34	17.53/53.44/79.54
2	RGM-o	1.02/1.49/4.42	28.28/51.78/77.46
	RGM-p	1.02/1.41/2.90	23.40/54.43/78.00
	GHS3D	1.04/1.64/6.34	11.06/47.37/81.41
	Tetgen	1.04/2.07/15.60	4.71/40.97/84.61
	SolidMesh	1.01/1.38/3.54	17.45/53.68/80.33
3	RGM-o	1.02/1.49/3.83	17.51/51.46/79.24
	RGM-p	1.02/1.41/5.16	13.51/54.26/76.59
	GHS3D	1.05/1.64/22.38	2.43/47.164/83.84
	Tetgen	1.05/9.91/1717.15	0.0004/40.02/83.84
	SolidMesh	1.02/1.41/3.56	17.93/53.26/80.37

three mesh generation tools to construct two sets of meshes. The first set includes the meshes which have uniform distribution of element sizes, and the meshes in the second set are constructed using the sizing function defined in Section 3.5. In order to have a fair comparison, we attempt to construct meshes in such a way that they have similar number of vertices within each group (uniform and adaptive), and compare them using the same set of quantitative metrics.

We use two groups of metrics. The first group includes those mesh properties, which can be directly optimized during the process of mesh construction. These include element shape and surface approximation accuracy. We assess the element shape by the minimum dihedral angle for each tetrahedron of the mesh. Surface approximation accuracy is evaluated as the percentage of the registration points covered by the mesh, which is a practical measure for the NRR application.

The second group includes quantitative metrics, which cannot be directly optimized

by the existing mesh generation methods. Let  $\bar{\mathbf{D}}_i$  be the ground truth displacement at the registration point  $i$ , which aligns given point in the floating image with the corresponding point in the target image. This value is known from the synthetic ground truth deformation field. Following the notation introduced in Section 2,  $\mathbf{D}_i$  is the displacement recovered at the  $i$ th registration point by block matching,  $\mathbf{U}$  is the set of displacements at mesh vertices, and  $\mathbf{H}$  is the interpolation matrix from the values of displacements at the mesh vertices  $\mathbf{U}$  to the displacements at the registration points  $\mathbf{H}\mathbf{U}$ . The interpolated displacement at the  $i$ th registration point is defined as  $[\mathbf{H}\mathbf{U}]_i$ . Also,  $\mathbf{T}_i$  is the structure tensor evaluated at the  $i$ th registration point. Using this notation, we define the set of application-specific quantitative evaluation metrics:

1. *Approximation error* at a registration point is defined as  $\|\mathbf{S}_i\mathbf{D}_i - [\mathbf{H}\mathbf{U}]_i\|$ . We assess the accuracy of approximation by the percentage of the registration points, where the magnitude of this error exceeds 1.0. We call those registration points “error points”, while reporting results (errors below this threshold are in the sub-voxel range). We use the proportion of the registration points  $P_e$  where the error exceeds 1.0 mm as a measure of the approximation error:

$$P_e = \frac{|\{\mathbf{f} \in \mathcal{F} : \|\mathbf{S}_i\mathbf{D}_i - [\mathbf{H}\mathbf{U}]_i\| > 1.0\}|}{|\mathcal{F}|} \cdot 100\%.$$

2. *Outlier detection sensitivity*, defined as the ratio of the true outliers within the discarded registration points to the total number of the discarded registration points. The set of true outliers,  $\mathcal{F}_o$ , in the block matching result is defined as a

registration point, where  $\|\mathbf{T}_i \mathbf{D}_i - \bar{\mathbf{D}}_i\| > 1.0$ :

$$\mathcal{F}_o = \{\mathbf{f} \in \mathcal{F} : \|\mathbf{T}_i \mathbf{D}_i - \bar{\mathbf{D}}_i\| > 1.0\}.$$

The registration algorithm defines the  $i$ th registration point as an outlier based on the absolute value of the difference between the estimated deformation  $\mathbf{H}\mathbf{U}_i$  and the result of block matching at that point  $\mathbf{S}_i \mathbf{D}_i$ . The registration points are ranked by this value. The top percentage is defined by  $K = \frac{r_2}{n_{iter}}$  is rejected. Here  $r_2$  and  $n_{iter}$  are the registration parameters listed in Table 2.1. Let  $\mathcal{F}^k$  be the set of registration points at the beginning of the  $k$ th outlier rejection iteration, and  $\mathcal{F}_r^k$  – the set of registration points discarded by the registration algorithm at the  $k$ th iteration:

$$\mathcal{F}_r^k = \{\mathbf{f} \in \mathcal{F}^k : \|\mathbf{S}_i \mathbf{D}_i - \mathbf{H}\mathbf{U}_i\| > K_{\mathbf{f} \in \mathcal{F}^k}^{th}(\|\mathbf{S}_i \mathbf{D}_i - \mathbf{H}\mathbf{U}_i\|)\}.$$

We define  $P_o$  as the proportion of the true outliers that were correctly identified by the registration algorithm:

$$P_o = \frac{\sum_k^{n_{iter}} |\mathcal{F}_r^k \cap \mathcal{F}_o|}{|\mathcal{F}_o|} \cdot 100\%.$$

We use  $P_o$  as a measure of sensitivity in outlier detection.

3. *RMS of the absolute error* at the registration points, absolute error being defined as  $\|[\mathbf{H}\mathbf{U}]_i - \bar{\mathbf{D}}_i\|$ .
4. *Distribution of the registration points* with respect to mesh vertices.

Note that the goal of the study is not to tune the NRR parameters for optimal registration results. We attempt to perform a controlled analysis of the impact of mesh on the NRR performance, while keeping fixed the other registration algorithm parameters that can influence registration accuracy.

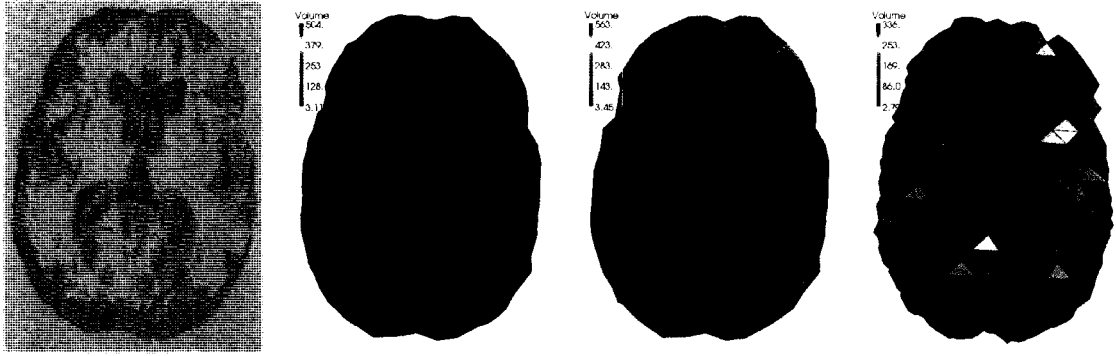
In the previous Section we compared the meshes constructed with *GHS3D* and *SolidMesh* tools, both of which are commercial products. In the application-specific evaluation we cannot use *GHS3D*, because we do not have access to the tool itself. In addition, *SolidMesh* cannot be evaluated, because we have the freely distributed version of this tool, which is not capable to construct adaptive meshes based on sizing function defined at background mesh vertices. Therefore, in the current study we use *NETGEN* as a representative method among AFT, in addition to the previously evaluated *RGM* and *Tetgen*.

We constructed synthetic deformation fields and performed NRR on the MRI scans 1 through 3 in the set of 18 images available from the Internet Brain Segmentation Repository (IBSR)<sup>2</sup>. The synthetic deformations were generated with 20 “knots” in the deformation grid on average [27]. The deformation magnitude at each “knot” was under 5% of the brain size to maintain the validity of the linear elastic physical model [63, 53].

The parameters used for CGAL surface mesh generation [42] were: angular bound 30°, surface radius and distance bounds 10.0, surface precision bound 0.001. Each method was used to construct two meshes for each registration case. The prescribed element size was uniform throughout the volume of the first mesh. The second mesh was constructed to adapt the element size according to the sizing function designed in

---

<sup>2</sup>The MR brain data sets were provided by the Center for Morphometric Analysis at Massachusetts General Hospital and are available at <http://www.cma.mgh.harvard.edu/ibsr/>.



**Figure 3.10:** Left to right: selected registration points, and adaptive tetrahedral mesh cuts (same slice) generated with *Tetgen*, *NETGEN*, and *RGM*. Tetrahedra are colored according to their volume, from blue (smallest) to red (largest).

Section 3.5. We adjusted the implementation-specific parameters to have the uniform and adaptive meshes with approximately 1.5K and 6K vertices, respectively.

We present cross-cuts of the adaptive meshes in Figure 3.10. The adaptive meshes generated with *NETGEN* have a layer of relatively large elements near the surface of the mesh. This is explained by the nature of AFT, which does not insert new points on the triangulated surface. We use the CGAL triangulation as the support surface, and let *NETGEN* to construct a new triangulation to respect the prescribed element sizing. However, the re-triangulated surfaces contained small triangles, which do not obey the prescribed edge sizing.

The synthetic registration cases differ in the number of registration points, and in the number of true outliers. Moreover, because of the differences in mesh surfaces recovered by CGAL and *RGM* respectively, different percentage of those points are located inside the mesh domain, as summarized in Table 3.4. This is an important observation, because the percentage of outliers impacts the registration error, and the reduction of the registration error is our ultimate objective. Also, due to the lower precision of surface approximation by *RGM*, about 4% fewer registration points and 1%

**Table 3.4:** Registration points, true outliers, and their coverage by the mesh surfaces.

case id	reg. points	outliers,%	reg. points inside,%		outliers inside,%	
			CGAL	RGM	CGAL	RGM
IBSR01	56447	7.8%	95.6%	91.1%	6.9%	5.5%
IBSR02	57526	16.2%	94.9%	90.8%	14.3%	12.4%
IBSR03	46525	18.8%	95.3%	90.5%	16.0%	13.1%

to 3% fewer outliers are located within the *RGM*-generated meshes. The distribution of outliers is non-uniform, and many of them are located close to the surface.

None of the meshes contains sliver elements. We observe that the minimum dihedral angle was the largest,  $14^\circ$ , in the *NETGEN*-generated meshes. The values of this metric for *Tetgen* and *RGM* are  $8^\circ$  and  $5^\circ$  respectively.

Based on the results presented in Figures 3.11 and 3.12, the use of custom mesh sizing can significantly improve the distribution of registration points compared to uniform-sized meshes. In conjunction with the sizing function, *Tetgen* achieves the best distribution results overall. Although the average values for the distributions are similar for all meshes, *Tetgen* meshes have lower maximum values and better distribution: the distribution curve approaches normal distribution with the mean close to the desired number ( $k = 30$ ). The advantage of *Tetgen* over *NETGEN* is that mesh points can be inserted at arbitrary locations on the surface during refinement. *RGM* is limited even more than *NETGEN*, as new points can be inserted only at the periodic predefined locations, based on the initial lattice structure. Nevertheless, the distributions in *RGM* meshes are consistently better compared with *NETGEN*. This might be caused by large elements near the surface of the *NETGEN* meshes.

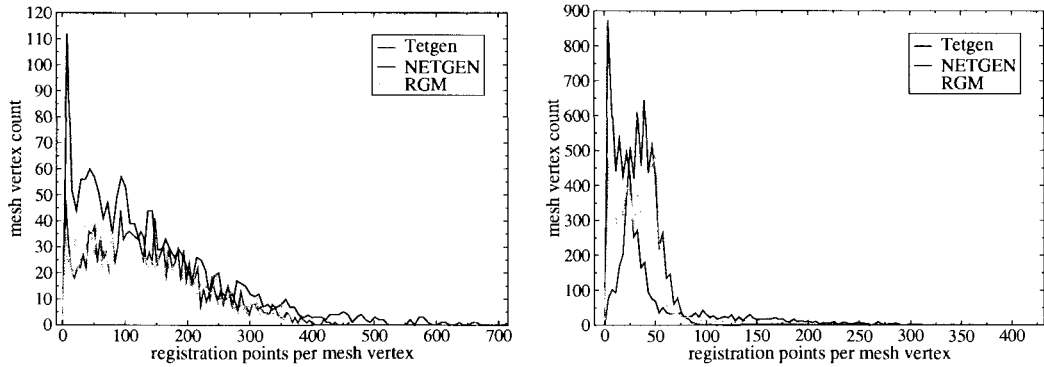
Note that empty vertex cells do not pose a problem. The corresponding mesh nodes will move following the neighboring vertices during registration. Problems can be caused

**Table 3.5:** Application-specific metrics for the evaluated meshes.

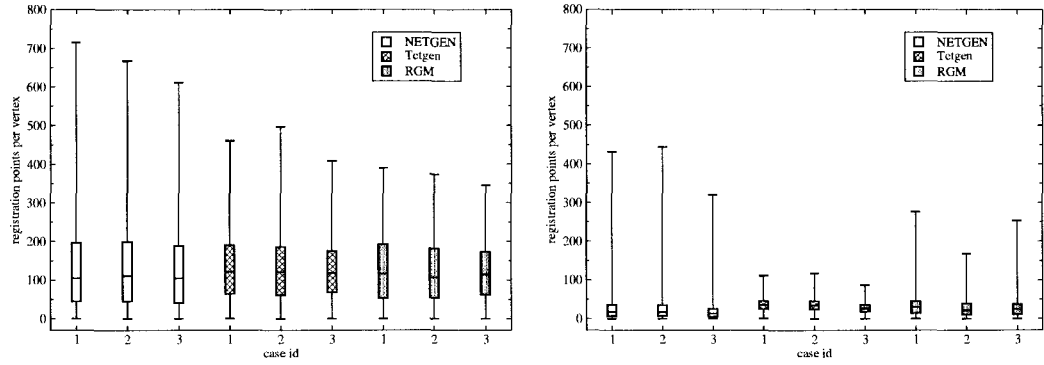
case id	metric	uniform size meshes			adaptive meshes		
		Tetgen	NETGEN	RGM	Tetgen	NETGEN	RGM
IBSR01	mesh points	1617	1596	1607	6044	6020	6209
	$P_e$	2.4%	2.6%	1.9%	1.6%	1.8%	1.1%
	$P_o$	53.5%	52.6%	43.8%	52.3%	52.5%	42.8%
	RMS error	1.60	1.62	1.61	1.69	1.65	1.60
IBSR02	mesh points	1682	1617	1696	6166	6255	6993
	$P_e$	6.7%	7.1%	5.9%	4.9%	5.8%	4.3%
	$P_o$	64.7%	64.4%	59.4%	63%	64.7%	60.5%
	RMS error	1.98	1.92	1.82	2.55	2.27	2.23
IBSR03	mesh points	1410	1413	1404	6631	6503	6033
	$P_e$	7.7%	8.1%	6.3%	4.8%	6.4%	4.5%
	$P_o$	72.9%	71.2%	65.7%	70.7%	74.6%	69.3%
	RMS error	2.61	2.53	2.25	3.52	3.08	3.05

by few registration points (the contribution of outliers is not smoothed by the correctly recovered displacements), or by too many registration points in the cell (increased approximation error).

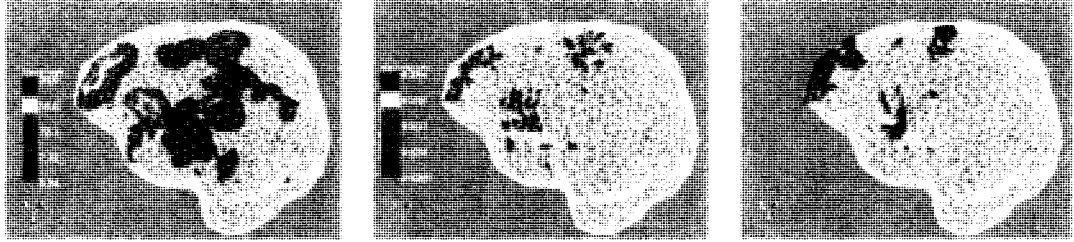
The non-rigid registration was performed with the default parameters suggested by Clatz et al. [53]. The fraction of selected registration points  $r_1$  was set to 5%. The quantitative metrics that are not directly optimized by mesh generation are summarized in Table 3.5. The approximation accuracy is consistently improved for all mesh

**Figure 3.11:** Distribution of the number of registration points per mesh vertex: uniform-graded meshes (left) and adaptive refined meshes (right); IBSR01.





**Figure 3.12:** Boxplot for the distribution of the number of registration points per mesh vertex cell complex with the unrefined meshes (left) and meshes constructed using the sizing function adapted to point distribution (right).



**Figure 3.13:** Left: Image voxels with the largest values of the error with respect to ground truth. Center: mesh elements with the minimum eigenvalue of the dilation matrix below 0.1. Right: inverted mesh elements after NRR. IBSR02, *Tetgen* adaptive mesh, wireframe shows the undeformed mesh.

generation methods when the refined meshes are used. However, this is the only metric that is clearly connected with the size of the elements of the mesh. We do not observe improvement in either outlier detection sensitivity or RMS of the registration error. On the contrary, RMS error is increasing in the refined meshes. We suggest that there are two major reasons why this may be the case.

First, the adaptively refined meshes are more susceptible to the element inversion during NRR. The areas of the high error in the deformation field recovered by NRR are spatially co-located with the areas of the mesh, where tetrahedra invert or become highly skewed. We use the tetrahedron measure proposed by Baker [20] and identify highly skewed mesh elements by the minimum eigenvalue of the element dilation ma-

trix. Figure 3.13 shows spatial correspondence of the locations of skewed and inverted elements with the areas with the largest registration error. The sizing function that we used does not take into consideration the expected magnitude of deformation.

Second, refined meshes decrease the error of approximating displacements recovered by block matching both at the outlier and non-outlier registration points. Ideally, the mesh should be constructed in such a way that in each mesh vertex cell the number of outliers is less than the number of correct displacements. In general outliers are distributed non-uniformly. As we decrease the size of the mesh elements, it becomes more likely that the registration points inside some cells will be dominated by the outliers, causing higher error with respect to the true deformation.

We tried to resolve the first problem by modifying the sizing function to reflect the deformation magnitude averaged over the  $k$  closest registration points. The parameters for CGAL surface mesh recovery were chosen according to the maximum of the averaged deformation magnitude near the object surface. None of the evaluated meshers was able to follow the prescribed sizing distribution closely. Both *Tetgen* and *NETGEN* created large tetrahedra near the mesh surface. We cannot attempt to improve the fitness of the mesh to the sizing function by reducing the default values of the *alpha* parameters, as we have done for the meshes evaluated previously. The *alpha* parameters control the bound on the shortest edge length at a mesh point, see Lemma 1 by Si [179]. Their reduction introduces small volume elements. The construction of meshes that adapt to the degree of deformation requires further study.

### 3.8 Summary

In this Chapter we approached the problem of constructing finite element meshes in the context of a specific non-rigid registration method. The task of constructing such meshes has been traditionally complicated by the fact that the domain of interest is described as a set of image voxels. However, in addition to the intrinsic difficulty of constructing meshes from the image data, the adaptation of the mesh to meet the requirements of a specific image processing application is particularly challenging.

We presented a list of specific requirements imposed by the non-rigid registration application on tetrahedral mesh, and completed a quantitative evaluation of the existing approaches. Based on the results of the literature review, we compiled a taxonomy of the available meshing methods commonly used in medical applications, and summarized their differences and limitations in the context of the considered application. We observe that there are no tools for direct conversion of the image data to tetrahedral volume mesh. Existing methods commonly employ multiple processing steps (surface recovery, decimation, volume mesh generation) to construct such meshes, but the details and deficiencies of such multi-step approaches are often left outside the research discussion.

We attempted to fill the gaps in the existing body of literature and tools by presenting an end-to-end approach for direct conversion of binary images into tetrahedral meshes. We also describe a detailed procedures for using conventional meshing tools that work with surface triangulations to mesh binary image data, and present a quantitative comparison of the meshes constructed with different methods for discretization of ICC segmentations derived from the real patient data. *RGM*, the mesh generation tool we developed, constructs meshes of quality comparable with the state of the art

commercial packages for the geometries used in NRR. This tool has been made available as open source project, and was used for constructing patient-specific models for NRR during IGNS procedures at Brigham and Women’s Hospital in Boston, MA in 2006.

There are no general solutions in mesh generation – application requirements vary, and mesh generation tools have to be adjusted to meet those requirements. Based on the specific set of requirements we formulated for the NRR application, we developed a set of quantitative metrics that allow to assess the goodness of a particular mesh for registration purposes. Specifically, we studied the issue of adapting the distribution of mesh element sizes to the distribution of the registration points. The quantitative comparison of the considered mesh generation methods allowed us to conclude that Delaunay-based mesh generation in conjunction with the robust tools for implicit surface triangulation achieve the best quality mesh in terms of distribution of the registration points. However, the remaining issue is the one of constructing such a mesh that optimally conforms to the two conflicting requirements: distribution of the registration points and the magnitude of deformation.

Possibly the most important question that has not been studied is the impact of the mesh on the registration accuracy with the clinical data. We stress that the question of the meaningful impact of the mesh generation method on the registration accuracy cannot currently be evaluated due to the lack of ground truth data for clinical registration cases. The difficulties of assessing the accuracy of non-rigid registration are further discussed in Chapter 5. In this Chapter we described the list of guidelines and a number of solutions for mesh generation, however further evaluation is necessary, contingent on the availability of clinical ground truth data.

### 3.9 Contributions

The main contributions of the work presented in this Chapter are the following:

- a novel tool for direct conversion of binary images into tetrahedral meshes has been presented and evaluated in the context of NRR applications;
- application-specific requirements to mesh generation for NRR application have been summarized;
- an evaluation framework for comparing suitability of tetrahedral meshes in the context of NRR has been proposed and used for evaluation of the considered mesh generation tools;
- a detailed description of the end-to-end procedure for constructing tetrahedral meshes using off-the-shelf software has been presented and analyzed.

The results presented in this Chapter have been published in the following papers:

1. A.Fedorov, N.Chrisochoides, R.Kikinis, S.K.Warfield. Tetrahedral mesh generation for medical imaging. MICCAI'05 Open source workshop, 2005 (appears in Insight Journal, <http://hdl.handle.net/1926/35>)
2. N.Archip, A.Fedorov, B.Lloyd, N.Chrisochoides, A.Golby, P.Black, S.K.Warfield. Integration of patient specific modeling and advanced image processing techniques for image guided neurosurgery. Proceedings of SPIE Medical Imaging, 2006, pp.422-429
3. A.Fedorov, N.Chrisochoides, R.Kikinis, S.K.Warfield. An evaluation of three approaches to tetrahedral mesh generation for deformable registration of MR images.

Proceedings of IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2006, pp.658-661

4. B.Joshi, A.Fedorov, N.Chrisochoides, S.K. Warfield, S. Ourselin. A Quantitative Assessment of Approaches to Mesh Generation for Surgical Simulation. *Engineering with Computers*, 24(4):417-430, 2008
5. A.Fedorov, N.Chrisochoides. Tetrahedral Mesh Generation for Non-rigid Registration of Brain MRI: Analysis of the Requirements and Evaluation of Solutions. *Proceedings of 17th International Meshing Roundtable*, 12-15 October 2008, pp.55-

72

## Chapter 4

# High-Performance Image Registration

Accuracy is the most important requirement for any non-rigid registration method. However, once the accuracy of the method has been established, it can only be useful during IGNS if the computation is feasible to complete within the time allocated for image processing during the procedure. Non-rigid registration methods for recovering brain shift have been studied for more than a decade, but many of them were found to be impractical due to the high computation time they require.

The NRR method by Clatz et al. has been designed from the very beginning with the goal of supporting IGNS image processing. The authors performed a retrospective study of the implementation using the image data from past neurosurgeries, and estimated the computation time required for intra-operative component of the registration to be under one minute [53]. However, the practical constraints of the implementation and high variability of the execution time depending on the input data did not allow to achieve

such result during the initial attempts to apply this method intra-operatively.

The research results presented in this Chapter were originally motivated by a practical need. During the years of 2006 and 2007, image-guided craniotomies of brain tumors were being performed on average every month in the open magnet facility at Brigham and Women’s Hospital in Boston, MA. State of the art tools available at that time did not provide any capabilities to account for intra-operative brain deformation, which motivated the feasibility study of using the method designed by Clatz et al. in the clinical setting.

An attempt to use the implementation for intra-operative prospective evaluation of the method using a 4-processor high-end workstation resulted in the conclusion that the execution time of NRR on a single high-end workstation (next to the operating room) exceeds by far the time a surgeon can wait to proceed with the tumor resection after each intra-operative scan. Ideally, the surgeon would like to have registration results immediately, but certainly the time to process the images should not exceed the time required to obtain a full brain iMRI, which takes about 4-5 minutes. Better performance of the registration process could conceptually be achieved using one of two approaches. We can either apply less computationally demanding models which provide the same accuracy of the solution, or we can attempt to improve the scalability and efficiency of the existing implementations. In this Chapter we explore the second approach to make the computation of the NRR method discussed earlier feasible and practical for IGNS.

As we show in this Chapter, high performance computing (HPC) resources prove to be essential to enable in-time completion of NRR. However, in addition to making NRR feasible, HPC resources hold the potential to improve the *accuracy* of registration.



In practice, the non-zero registration error is inevitable, independently of the level of complexity in the registration algorithm. This is due to the fact that existing models of brain deformation are heuristics and approximations. There are technological limitations (e.g., insufficient contrast in the intra-operative iMRI to detect resection boundaries, low image resolution), lack of knowledge about the physical tissue properties, absence of precise patient-specific biomechanical models of the brain, etc. The heuristics used in the registration methods are characterized by certain parameters that allow to tune their behavior. The selection of the optimal values for such parameters is usually not straightforward, because they depend on the images that are registered. The parameter settings can be optimized retrospectively, but such settings may not be optimal for the prospective application of the method.

The problem of optimum parameter selection has been previously recognized in the image registration community, and a number of approaches have been proposed to address this issue. Skrinjar et al. [183] develop a framework for selecting optimum stiffness, which parametrizes the biomechanical model of brain deformation. The proposed solution is to initialize stiffness to a small non-zero value, and iteratively increase it until the distance between the modeled locations of the surface points to the true surface location is close to zero. Ino et al. [110] introduce the concept of *speculative parallelism* in the context of registration, where the same registration problem is approached with the different sets of parameters at the same time. In any scenario of optimum parameter estimation, multiple instances of registration must be evaluated. Given the non-trivial amount of computation needed for a single instance of NRR, the task of parameter search may become infeasible.

In this Chapter we focus on the use of HPC resources for the NRR method of Clatz et al. Our first objective is to develop a high-performance implementation that meets the IGNS time constraints and can be used for prospective studies in clinical research. The second goal is to evaluate the possibilities of enabling parametric search for NRR. We consider both the ubiquitous cluster of workstations and the widely distributed National cyber-infrastructure (TeraGrid) as platforms to facilitate these highly demanding computations.

## **4.1 Cluster-based Implementation**

### **4.1.1 Related Work**

Image processing computations that operate on three-dimensional medical data often require non-trivial time when performed sequentially. Such tasks as multi-channel pattern classifications used in medical image segmentation, complex biomechanical simulations or optimization in multidimensional parameter space often used in medical image registration are inherently complex. Some of these computations hold the potential to be useful in decision making during clinical procedures, which impose tight time constraints on the delivery of the computation results.

In this Section we focus on the computations related to image processing for the purposes of image-guided neurosurgery. A number of approaches and frameworks have been proposed to leverage the high performance power of local cluster computing resources for this clinical application, as we discuss next. The major results in developing HPC systems for such problems are coming from relatively few groups, which have corre-

sponding needs. Until recently, the use of high quality intraoperative imaging modalities was limited and available at few clinical centers due to high cost of the needed imaging systems.

Christensen and collaborators were some of the first to discuss the use of parallel computing resources for solving time-consuming problems related to brain MRI processing [51]. They describe a method for constructing anatomical brain atlases customized to patient image data. The authors present a system implemented on a massively parallel SIMD architecture, with the parallelized PDE solver used in deformation modeling and parallel evaluation of the three-dimensional transformations. The authors state that parallelization allowed to reduce processing time from weeks to hours for typical datasets.

Warfield et al. [198] presented a parallel implementation of the brain tissue classification and rigid registration based on the segmented label overlap similarity metric. The implementation includes load balancing component based on *workpiles*, which essentially correspond to the master-worker distribution scheme we adopted in our distributed implementation of NRR. The implementation developed in [198] was used to process a large number of brain scans MRI scans with the reported processing time within 10 minutes.

In another development , Warfield et al. [199] presented some of the first results in intra-operative processing of iMRI. They consider the problem of intra-operative segmentation of brain iMRI. The segmentation method is based on voxel-wise classification of tissue, which can be completed independently for each voxel. The authors demonstrate linear speedup of segmentation on a 20-processor HPC workstation, which

allows to segment 60 slices of data in about 20 seconds. Remarkably, the developed method was subsequently applied and evaluated during real cases of neurosurgery and liver cryo-ablation [200]. The same group later developed a high-performance method for intra-operative non-rigid registration, which uses linear biomechanical model [197]. The authors used PETSc [159] library for parallel assembly and solution of the linear system. Although the authors report clinically acceptable timing results delivered by their implementation, the evaluation was restricted to off-line experimental studies.

The approaches to parallelizing registration computation are highly dependent on the specific registration method. For example, the methods considered in [199, 198] can be parallelized without any need for intra-processor communication using the master-worker approach. On the other hand, the parallel linear solver used by the framework described in [197] requires MPI-enabled communication. Rohlfing et al. [168] develop an implementation of free-form non-rigid registration method based on B-spline interpolation on a shared memory supercomputer. Global image similarity and gradient computations contribute most of the execution time, as they are repeatedly evaluated during the optimization process. The authors present parallel implementation of those steps, and report speedups close to linear in the application of this framework for off-line processing. Another distributed implementation of a registration method based on free-form deformation was developed by Ino et al. [111]. Stefanescu et al. [186, 187] developed a parallel implementation of the demons-based non-rigid registration method. Demons-based registration [191] presents the advantage of being able to recover small-scale deformations, as compared to registration based on B-splines. The authors of [186] report speedup of 11 on a cluster of 15 workstations. However, neither of these methods

incorporates biomechanical models of the tissue in the registration process.

An approach closely related to the registration method we target in this work was studied by Ourselin et al. in [151]. Although the authors considered the problem of rigid image registration, the registration method is based on deriving the global transform with the aid of block matching, which was parallelized using equi-distribution of work among the processing nodes of a cluster. The implementation was used for retrospective analysis of image data.

Overall, we observe that a large number of registration methods have been implemented on computing cluster resources. Such systems are motivated by excessive computational time required for processing three-dimensional image data. The parallelization approach is always customized to the specific registration method, and the common denominator in such implementations are the communication libraries used, e.g., MPI [140], and load balancing schemes based on the master-worker model. Some of the reported results are evidently suitable for intra-operative application of the methods. Nevertheless, very few methods (namely, only the framework developed in [199]) were actually applied and evaluated during the neurosurgery. We explain this by the difficulty of coordinating computation in the real time-critical setting, possible dependency of the performance on the input data and parameters, and insufficient robustness and accuracy of the registration approach.

#### **4.1.2 Motivation**

The method proposed by Clatz et al. has been designed specifically to be applicable during IGNS. Therefore, a number of design decisions have been made to make the

computation feasible within minutes:

1. Iterative rejection of the outlier registration points results in the changes in the interpolation and stiffness matrices  $\mathbf{B}$ ,  $\mathbf{H}$  and  $\mathbf{S}$ . The matrices are updated to account for these changes by locally modifying the submatrices corresponding to the discarded registration points.
2. The tetrahedral mesh used in the registration does not change between the iterations of the algorithm. The estimated mesh deformation is reflected in the deformation vector  $\mathbf{U}$ , but the mechanical stiffness matrix  $\mathbf{K}$  remains constant.
3. The authors recognize that the block matching component of registration is the most time-consuming processing step, and develop its parallel implementation using PVM [23].

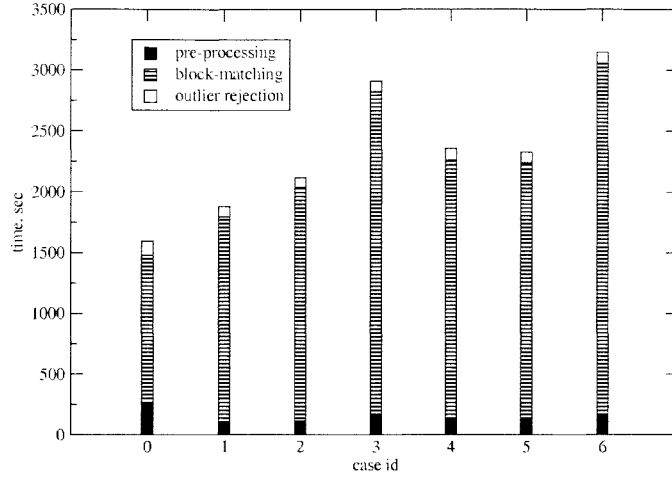
However, the method has been tested and evaluated retrospectively, in the environment of a research lab. The intricacies of running the code during neurosurgery and providing the result to the operating team often cannot be anticipated in advance, as cannot be anticipated all the difficulties related to using a heuristic method on an untested data.

During the initial attempts to run the original implementation during the course of neurosurgery, the following observations have been made:

- The only computational resource readily available for non-rigid registration next to the operating room is a high-end workstation. However, execution time of the prototype implementation by Clatz et al. on a high-end multi-processor worksta-

tion varied between 40 minutes and 1 hour, depending on the input data, as shown in Figure 4.1.

- In the absence of local computing facilities, the external HPC resources were required. The characteristics of the external execution environment are not known in advance. It is desirable for the parallel implementation to be robust and efficient on both time-shared and dedicated resources, with possible heterogeneity of the computing nodes.
- The functional complexity of the implementation (initialization, parameter checking, data transfer) did not allow to easily separate the individual components of the computation, and added significantly to the total time required to execute the code with the clinical data.
- The components that contribute most to the total execution time of the implementation are data pre-processing and block matching, with the block matching component highly amenable to parallelization. The parallel implementation of block matching by Clatz et al. cannot be ported on commodity clusters, since it is based on PVM, which is not a commonly supported parallel programming environment (specifically, it is not supported on either SciClone [56] or TeraGrid [188], the computational resources used in this study).
- The implementation is highly vulnerable to failures: a hardware fault during intraoperative part of the registration requires restart of the computation. Basic fault-tolerance functionality should be part of the implementation.



**Figure 4.1:** Execution time breakdown for the original implementation of NRR on a high-end 4-way workstation.

**Table 4.1:** Time (minutes) required for sequential block matching with 100K registration points using Normalized Cross Correlation similarity metric (Intel Xeon 3.7GHz).

block dimension	search window dimension							
	7	9	11	13	15	17	19	21
5	0.4	0.8	1.5	2.7	3.6	5.2	7.3	9.6
7	0.9	2.0	3.6	5.9	8.9	13.1	17.9	23.9
9	2.6	5.6	9.9	16.1	24.5	35.3	48.9	65.2
11	4.5	9.6	17.2	28.2	42.9	62.0	85.9	113.5

Based on the initial evaluation of the prototype implementation, block matching is the most time-consuming step, which requires parallelization. Table 4.1 shows the timing results for sequential execution of block matching with 100K registration points on a high-end workstation. The search window size is a parameter chosen to reflect the maximum expected deformation magnitude. The typical resolution of MR image is  $\approx 1$  mm, and the reported values of the brain shift are in the order of 10-20 mm [64]. Therefore, the size of the block matching search window must be twice the expected deformation, making the search window dimension of 21 pixel typical. Based on the results in Table 4.1, a block matching search parametrized by such values cannot be completed sequentially within the neurosurgery time constraints.



Given the initial observations, we identified the need to provide a computational infrastructure, capable of delivering the NRR results within the clinical time constraints. The components of such infrastructure must include high-performance computing resources, low-latency network connection between the remote site and the hospital network, and the software implementation, that can leverage these resources.

Nowadays, clusters of workstations (CoWs) are readily available at most research institutions and universities. However, the use of such facilities is normally optimized for off-line computations, i.e., when the results of the computation do not need to be delivered within strict time constraints. Most clusters are accessed in the *space-shared* mode. A user wishing to use the cluster for computation submits so called *job request*, which describes the code to be executed, inputs and outputs, number of cluster nodes, and the expected time needed for the computation to complete. The cluster *batch scheduler* manages the queue of job requests and assignment of the cluster resources. In the space-shared scheduling mode, once the cluster nodes are assigned to the task, the task has exclusive access to this subset of cluster resources.

In the case of NRR, the computation is triggered by the availability of the intra-operative MRI scan. Once this scan is available, the computation must proceed as quickly as possible. Typically, scheduling a job is associated with queuing delays, which depend on the current availability of the resources and job load of the cluster. Certain batch scheduling systems allow preemption of the currently scheduled jobs in favor of high-priority requests. However, job preemption requires the ability to save the current state of the jobs being preempted, so that they can be safely restarted. In practice, job preemption mode is rarely supported. An alternative to job preemption is advance

reservation of the cluster nodes. The problem related to this is the sub-optimal utilization of the resources, which is why advance reservations are rarely available on the production systems [146], and can be accomplished by special arrangement only.

In order to enable the timely computation of NRR, considering the lack of parallel computing resources at Brigham and Women’s Hospital at the time this study was performed, we decided to use resources of the SciClone Cluster [56] at the College of William and Mary. SciClone is a heterogeneous cluster that consists of roughly 200 nodes, which, as we show further, is sufficient for timely computation of the block matching registration step. Moreover, SciClone site is connected with the national LambdaRail [142] backbone network through the 10 Gbps link, providing low latency high bandwidth means for transferring intra-operative data and registration results. Parallel computations are facilitated by the numerous implementations of the Message Passing Interface (MPI) [140] available on SciClone. The access to the SciClone resources is managed by the PBS scheduler. However, for the purposes of this study, we were able to eliminate the PBS scheduling delay by special arrangement to have exclusive access to the SciClone resources at the time of the neurosurgery. In order to provide a backup computing facility for the fault-tolerance considerations, we also use the student computer lab at the Computer Science department at the College of William and Mary.

The described environment provides ample computational capabilities accessible via a high-end network connection. However, in order to take advantage of this computational infrastructure, significant restructuring of the NRR code was necessary. In the next Section we focus on the architecture of the parallel distributed NRR implementation.

### 4.1.3 Implementation

Based on the initial evaluation of the NRR performance, the most straightforward, but very important modification necessary was the separation of the monolithic computation into separate processing stages. The processing modules are pre-processing (selection of the registration points), block matching between the pre-operative and iMRI showing brain shift, iterative outlier rejection (solver) and resampling of the pre-operative image using the displacements recovered at mesh vertices. This separation is motivated not only by performance, but also by the fault-tolerance considerations.

The complete processing timeline is shown in Figure 4.2. The processing involves acquisition of data in the hospital located in Boston, transfer of the data and its pre-processing at the College of William and Mary, and transfer of the computation results back to the hospital for the subsequent post-processing and visualization. The technical aspects that needed to be considered in the high-performance implementation are load balancing and fault tolerance.

**Load balancing** Within the time-critical computational components, the block matching is the step that requires performance improvements. The processing required by the block matching is by definition embarrassingly parallel. In order to find a match for a given block, we need the block center coordinates, and the regions of the fixed and floating images bounded by the block matching window [53]. No communication between processors is necessary. A straightforward work assignment is static equi-distribution of the registration points to the computing nodes. Static assignment eliminates the need for communication during block matching computation. However, this approach is not practical for a number of reasons.



First, the computing environment is highly heterogeneous, and processing time required for a single registration point varies depending on the processor speed. Moreover, the computing resources at the computer lab are available only in the time-shared mode, i.e., multiple users may utilize a given node at the same time. Therefore, the CPU load is highly variable and cannot be predicted. This creates motivation for dynamic redistribution of work, or load balancing. Second, even in the presence of perfect static assignment of work, lack of communication between the computing node will make failure detection impossible, leading to the partial loss of the block matching results. Based on these two considerations, we implemented parallel block matching using hierarchical master-worker load balancing scheme.

Multi-level dynamic load balancing method which we adopted for parallel block matching is different from the static multi-level graph partitioning methods presented in [1], and is closer to the hierarchical partitioning approaches described in [189, 70]. We use initial estimation of the combined computational power of each cluster involved in the computation (based on CPU clock speed) for the weighted partitioning of the work-pool and initial assignment of work. This initial estimation is adjusted at runtime using a combination of master-worker and work-stealing [30, 204] methods. At the level of individual cluster, we implement conventional master-worker model. The input images are loaded in memory at all the computing nodes. The master node (selected as the rank 0 node assigned by MPI) is responsible for maintaining the list of registration points, and the results of block matching for each of these points. These data are maintained in the work-pool data structure. Each item of the work-pool contains the three coordinates of the block center, the displacement vector estimated by block matching, and the value of

correlation for the optimal location. The registration points are assigned to the worker nodes in small batches. Upon completion of the registration, the worker reports the result back to master and requests more work, if available.

We maintain the hierarchical structure of load balancing by designating a separate master node at each of the administrative domains involved in the computation. Each master node maintains a copy of the global work-pool, which are identical in the beginning of the computation. The portion of the work-pool assigned to the specific cluster is partitioned in meta-blocks (a sequence of blocks), which are passed between the cluster nodes using the master-worker model. As soon as all the matches for a meta-block are computed, they are communicated back to the master, and a new meta-block is requested. In case the portion of the work-pool assigned to a master is processed, the master continues with the “remote” portions of work (i.e., those, initially assigned to other clusters). As soon as the processing of a “remote” meta-block is complete, it is communicated to all the other master nodes to prevent duplicated computation.

**Fault tolerance** We perform data pre-processing sequentially on the fastest machine at the computer lab. Parallel block matching uses combined computational facilities of the computer lab and SciClone, and the FEM solver is executed sequentially on the fastest machine at the computer lab.

The implementation is loosely coupled, with the results of separate processing steps communicated over the file system. This introduces the first level of fault tolerance, i.e., if the failure takes place at any of the stages, we can restart just the failed phase of the algorithm and recover the computation.

The communication between different clusters is handled via TCP sockets. In case of

a hardware problem on one of the clusters, the rest of the unaffected sites can continue with the execution. A monolithic code which uses a conventional MPI implementation would crash in the case of a single node failure, because conventional MPI implementations do not include provisions for automatic checkpoint and restart [140].

The redundant workpool data structure maintained at each of the master nodes assists not only in load balancing, but also in providing the second level of defense against hardware failures. Based on the statistical data collected over the years, hardware failures are usually caused by the disruptions of power, or faults in air conditioning at SciClone. Such problems cannot be anticipated, and the recovery of the cluster following the failure takes hours. In case there is a failure that makes one of the clusters inaccessible, the block matching computation can continue, and the results already calculated will be secured at the replica of the work-pool at the other master node.

Both the frequency of work-pool updates within each cluster and the intra-cluster synchronization is an adjustable trade-off between the communication time and a possibility of duplicate block processing. This situation is possible when a block has been processed on one site, but the work-pool update has not yet been propagated. However, note that frequent synchronization of work-pools can be done by a separate background thread not involved in the computation, and the communication latencies associated with intra-cluster updates can be generally hidden by employing non-blocking communication feature of existing communication libraries. Overall, these side effects are negligible compared to the benefits we gain by our ability to save the computation state.

The distributed implementation was exposed to the hospital users as a web service. In order to use the service, the client-side script was developed to implement initial data

correctness checks, contact the server and transfer the computation results back to the hospital, as shown in Figure 4.2.

#### 4.1.4 Evaluation

The performance evaluation is based on non-rigid registration of seven image datasets acquired at BWH. Two of these seven registration computations were accomplished during the course of surgery, while the rest of the computations were done retrospectively. The surgery for case 6 was postponed in the last moment due to patient's health complications, and the analysis was performed retrospectively. All of the intra-operative computations utilized SciClone, which was reserved in advance for the neurosurgery use.

All the experiments for the implementation we present here were performed in the combined computing resources of two clusters: (1) a cluster of 30 Dell Precision 360n single-core CPU at 3 to 3.6GHz and 1GB SDRAM 333MHz which is used by computer science students for class assignments and projects (CSLab) and (2) SciClone cluster [56]. These clusters belong to different administrative domains (users need separate accounts to access these resources). SciClone is a heterogeneous cluster maintained at College of William and Mary. It is arranged as seven sub-clusters which can be used individually or together. The extensive details of hardware configuration and structure of SciClone are available elsewhere [56]. The configuration details of the SciClone sub-clusters follow, since they have changed since the time experimental evaluation was performed.

*Whirlwind* : 64 single-CPU Sun Fire V120 servers, 650MHz, 1GB RAM, 36.4GB HDD;

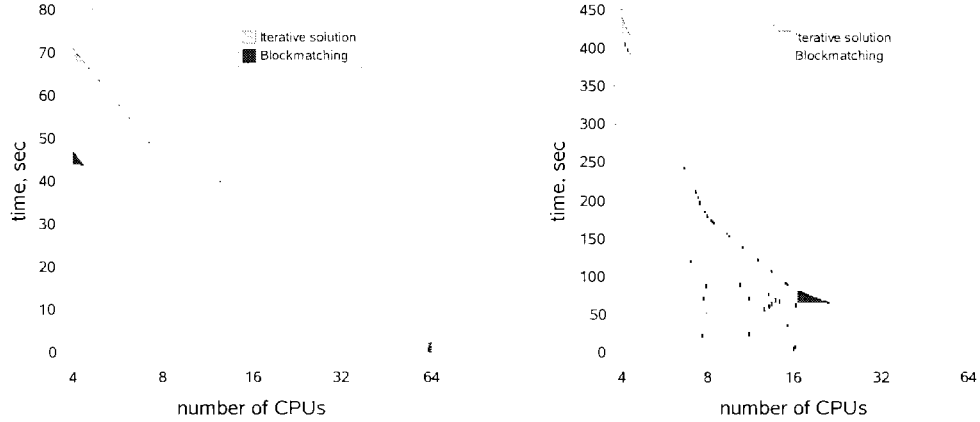
2) *Typhoon*: 64 single-CPU Sun Ultra 5 workstations, 333MHz, 256MB RAM, 9.1GB HDD;

3) *Twister*: 32 dual-CPU Sun Fire 280R servers, 900MHz, 2GB RAM, 72.8GB



HDD; 4) *Tornado*: 32 dual-CPU Sun Ultra 60 workstations, 360MHz, 512MB RAM, 18.2GB HDD; 5) *Gulfstream*: 6 dual-CPU Sun Ultra 60 workstations, 360MHz, 512MB RAM, 45.5-63.7 GB HDD; 6) *Hurricane*: 4 quad-CPU Sun Enterprise 420R servers, 450MHz, 4GB RAM, 18.2GB HDD; 7) *Vortex* : 4 quad-CPU Sun Fire V440 servers, 1.28GHz, 8-16GB RAM, 292GB HDD. Three additional heavily configured server nodes act as front-ends and file servers for the entire system, and two network server nodes provide application gateways between SciClone’s various internal and external networks. In aggregate, at the time of the experimental evaluation SciClone provided 212 nodes with a total of 311 CPUs, 236 GB of RAM, 15.1 TB of disk capacity, and 362 GFLOPS peak floating point performance. We used all sub-clusters of SciClone except *Typhoon*, which at the time was the least powerful resource.

The quantitative results of the evaluation are summarized in Tables 4.2, 4.4, and 4.2 and in Figure 4.3.



**Figure 4.3:** Scalability of the time-critical part of registration implementation with block sizes  $3 \times 3 \times 3$  (left) and  $11 \times 11 \times 11$  (right).

*Portability* of the code has been improved. Original implementation used PVM [23] that is not widely supported nowadays. The new implementation is based on MPI [184].

**Table 4.2:** Execution time (seconds) of the intra-surgery part of the restructured implementation at various stages of development.

Setup	Case ID						
	1	2	3	4	5	6	7
using original PVM implementation	1558	1850	2090	2882	2317	2302	3130
SciClone (240 procs), no load-balancing	745	639	595	617	570	550.4	1153
Pre- and post-processing on single CS node, BM on SciClone (240 procs), no load-balancing	226	123	182	189	217	174	189
Pre- and post-processing on single CS node, pre-processing in advance, BM on SciClone (240 procs), dynamic load-balancing	35	53	56	47	42	42	48
Pre- and post-processing on single CS node, pre-processing in advance, BM on SciClone (240 procs) and CSLab(29 procs), dynamic 2-level load-balancing	<b>30</b>	<b>40</b>	<b>42</b>	<b>37</b>	<b>34</b>	<b>33</b>	<b>35</b>

From porting the code from BWH-HPC cluster at BWH to SciClone in dedicated mode at W&M we improved the performance of the application by 3.5 times (see first and second rows of the Table 4.2). One would expect a linear improvement (i.e., 60 times) due to the scalable nature of the algorithm. However, this was not feasible due to the following three reasons: (1) monolithic design of the original implementation, i.e., it was designed to run on the same static configuration from the beginning to the end; (2) the fastest processors of *SciClone* were much slower (in terms of CPU speed) than the nodes available at the hospital and *CSLab*, and (3) presence of work load imbalances.

*Decoupling* of the pre-processing phase from intra-operative image registration. This allowed us to: (1) execute the sequential pre-processing phase on the fastest workstation available and (2) *mask latencies* caused by MPI initialization of the parallel component of the implementation. Initialization of MPI, depending on the implementation, may

include startup of the daemon processes at each of the nodes, which adds up to a non-negligible delay when using hundreds of nodes. As a result we were able to reduce the response time of the non-rigid registration on average by more than 12 minutes (see second and third rows of Table 4.2 and Figure 4.2). The maximum improvement we observed so far was more than 16 minutes (case 1).

*Multi-level dynamic load balancing* over multiple clusters reduced the execution time by more than 14 minutes when we used 240 processors and the decoupled implementation of the code (compare the second row with the fourth and fifth rows of Table 4.2). The flexibility we achieved from the portability of the code and these two optimizations (decoupling and dynamic load balancing) together reduced the absolute response time of the parallel block matching code alone by two orders of magnitude. We can complete this step in 1225 seconds (case 1) and 2890 seconds (case 7) using the local resources available at the hospital, while with all the computational resources of multiple clusters we could possibly utilize, we now complete it in less than 30 seconds, see Table 4.4. These experimental data indicate, that we gain about 50% improvement in the performance of the application due to dynamic load balancing, i.e., we can achieve the same performance with half of the resources. This is a particularly important result for hospitals, where space is at premium.

*Web-services* became an option after the decoupling and implementing the functionality to handle platform-dependent aspects of heterogeneous and time-shared clusters. The use of web-service and hence the elimination of the human factor from the loop reduced the response time by roughly 20 minutes. Originally, the initialization involved two teams, one at BWH and another at W&M, and four different administrative do-

**Table 4.3:** Execution time (seconds) of block matching with static work distribution on SciClone cluster.

ID	Number of processors					
	40	80	120	160	198	240
1	377.82	199.96	142.14	110.53	91.33	60.84
2	173.41	101.49	158.08	89.72	52.75	29.70
3	316.20	159.34	105.50	81.40	65.28	53.54
4	407.82	218.86	150.56	113.17	93.67	71.17
5	353.96	183.20	127.19	105.06	83.43	57.09
6	298.97	151.16	104.72	81.30	64.13	49.53
7	300.99	151.48	105.10	80.11	64.37	49.49

**Table 4.4:** Execution time (seconds) of block matching with dynamic work distribution on SciClone cluster.

ID	Number of processors					
	40	80	120	160	198	240
1	200.84	107.77	77.67	63.41	54.65	33.27
2	103.65	77.62	125.95	111.62	36.89	17.06
3	217.70	109.74	75.45	56.62	47.33	38.80
4	242.35	125.64	91.23	70.19	57.80	43.98
5	189.17	100.95	73.00	57.73	49.75	32.47
6	152.13	81.20	57.97	45.65	37.49	27.51
7	150.81	78.98	57.35	45.91	37.21	27.91

mains, taking up to 30 minutes of time. With the use of web-services, the time to start the application at BWH and initiate its execution at W&M was reduced to 75 seconds (57s to send the images from BWH to W&M and 18s to retrieve the output at BWH from W&M). These data vary due to network traffic, but consistently we observe them to be below 2 minutes total, which is the bare-minimum cost for data transfer and I/O for the application. Computations for Case 7 (run retrospectively) took less than 7 minutes (as measured between the time the data was submitted for computation from BWH until the result was received back from W&M). *This is an improvement in the response time by 13 to 15 times.*

*Fault-tolerance* became very important attribute of the code, since by increasing the

hardware components and the complexity of the distributed computing environment we increase the chances for failures. Our implementation can safely run the registration procedure on several clusters in different locations at a modest overhead of 6% over the non-fault-tolerant version, and deliver results of the computation even in case of the failures of all but one cluster involved.

#### **4.1.5 Discussion**

We developed a parallel and distributed web-service based implementation of the NRR code that is capable of delivering registration results within 5 minutes. Moreover, we showed that this result can be achieved in the absence of the high-performance infrastructure within the hospital, as the existing communication networks can provide seamless access to the remote computing sites. The result we demonstrated confirmed, for the first time, the feasibility of near real-time image fusion for brain MRI using landmark tracking across the entire image volume. This became possible because we were able to achieve: (1) the reduction of the execution time of the parallel block matching from 2890 seconds at BWH to less than 30 seconds at W&M (for the last intra-operative non-rigid registration case); (2) effective use of a large number of processors, with dynamic load balancing, we improved the performance of parallel block matching by 50% compared with the static work distribution; (3) the reduction of the overheads associated with manual initialization and transfer of data from BWH from 20-30 minutes to about 60 seconds; (4) ease-of-use; and (5) the first fault-tolerant and web-service based non-rigid registration code using landmark tracking across the entire volume.

The computations we perform are supporting clinical research activities underway at

BWH. It has yet to be shown whether the software infrastructure we developed can be used for *decision-making* during the course of a neurosurgery. It is clear, however, that there is great potential in using ubiquitous CoWs for performing intensive time-critical computations for medical applications within the clinical time constraints.

## 4.2 Grid-Enabled Infrastructure

### 4.2.1 Motivation

Registration accuracy, robustness, and performance can all be affected by the values of NRR parameters. Sensitivity of the method is defined as the impact of the small changes in the algorithm inputs and parameters on the result of computation. Sensitivity of the specific NRR method we study has previously not been evaluated. In this Section we present an initial evaluation of the NRR sensitivity, which motivates our studies in developing a Grid-enabled infrastructure for parametric studies and speculative execution of NRR.

The default parameter values for the NRR computation were suggested in [53]. Previously, results of registration were reported with the block dimensions fixed to  $7 \times 7 \times 7$ , and cumulative rejection rate of 25%. These settings were shown to give significant improvement in registration accuracy as compared to rigid registration only. This result was verified in multiple studies using clinical data both retrospectively and prospectively [53, 8, 7].

High dimensionality of the parameter space precludes its exhaustive evaluation. At the same time, some of the parameters (e.g., block size used in block matching, or

percentage of the rejected blocks) belong to well-defined ranges. Block size cannot exceed window size, and has dimensions expressed in multiples of the corresponding voxel dimensions. The outlier ratio cannot be higher than 50% of the measured displacements [169]. The optimum selection of those parameter values is non-trivial, as we do not know in advance what will be the mixture of outliers in the data, and what will be the magnitude of deformation. We select the parameter subspace of the registration algorithm by varying outlier rejection fraction  $r_2$  and block size  $B$  as follows:  $B \in \{5, 7, 9, 11\}$ ,  $r_2 \in \{5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$ . The rest of the parameters were fixed to their default values, which we discussed in Chapter 2.

We evaluated the sensitivity of the algorithm to changes in the selected parameters on clinical data using expert-placed corresponding brain landmarks in the floating and target images from [53]. The initial alignment accuracy is evaluated by measuring the Euclidian distance between the corresponding landmark locations in the floating and reference data. Deformation field derived during non-rigid registration is then used to warp the floating image to the reference image. The post-registration accuracy is measured by computing Euclidian distances between the corresponding landmarks in the warped (registered) and reference images. Preference was given to this method over the automated assessment (e.g., the method described in [8]), because it allows for the collection of localized accuracy measurements throughout the image volume.

The impact of the selected parameters on registration accuracy was evaluated on 6 MRI datasets previously used in [53]. Ideally, warped landmarks of the floating image should coincide with the ones located on the reference image. The accuracy of registration at the landmark locations can be assessed by computing distances between the

corresponding landmarks in floating and reference images (before registration), and between floating and registered images (after registration). The registration accuracy is considered good, if the registration error at a given landmark location does not exceed the smallest voxel dimension. Note, that identification of anatomical landmarks is challenging for clinical data and depends on the specifics of a particular dataset. This made identification of the same number of landmarks in each of the images not possible.

First we consider the impact of varying the block size on the initial sparse displacement field recovered by block matching. The landmark-measured error before non-rigid registration for each case is shown in Table 4.5. Table 4.6 summarizes the improvement in accuracy for each of the landmarks with the optimum combination of  $B$  and  $r_2$ , as compared to the default parameter settings. Based on the experimental data, in most cases, good registration accuracy is achieved using the default parameters as suggested by Clatz et al. [53]. For example, for Case 2 (see Figure 4.4), we did not achieve any noticeable improvement in accuracy by varying the studied parameters. This was not true for Case 3, where such improvement was as high as 2.9 mm. In both cases, however, there were landmark points, where registration error exceeded voxel dimensions.

The experimental results suggest, that the optimum value of outlier rejection is varied in different locations of the image. For example, if we consider landmarks 5 and 12 in Case 1, the optimal combination of the studied parameters is different in each case, see Figure 4.5. When the rate of outlier rejection at a certain location is insufficient, registration error will increase due to the influence of incorrect matches. Otherwise, when the rejection rate is too conservative, correct matches may be discarded, which will lead to higher approximation error.



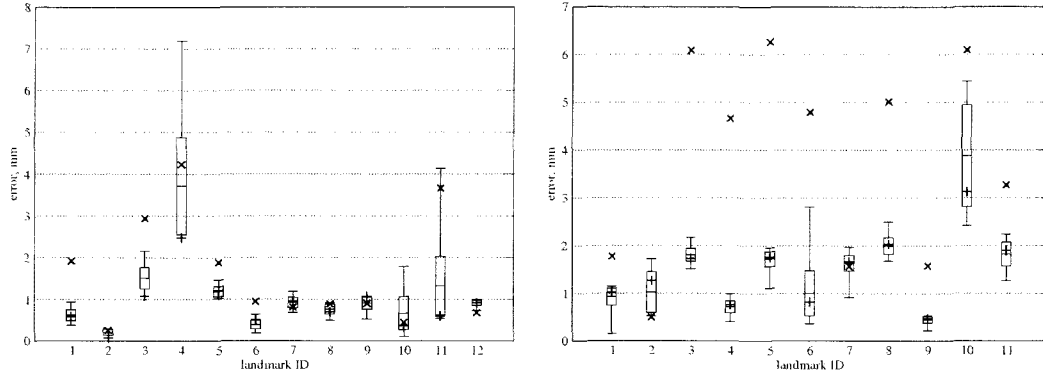
**Table 4.5:** Landmark error before registration (mm)

Case	1	2	3	4	5	6	7	8	9	10	11	12
1	0.9	2.4	1.7	1.8	5.0	3.3	1.7	2.1	2.6	2.0	2.1	1.8
2	1.9	0.8	3.6	1.7	1.7	0.8	0.8	1.2	1.2	0.8	3.6	0.8
3	6.7	3.7	0.9	9.7	3.7	3.0	2.0	–	–	–	–	–
4	1.8	0.9	6.8	4.7	5.7	4.5	1.3	4.6	1.3	6.4	3.7	–
5	0.0	0.8	1.2	0.8	4.2	6.1	9.0	6.5	6.5	9.0	–	–
6	0.8	1.9	3.5	0.8	2.7	0.8	1.7	0.0	–	–	–	–

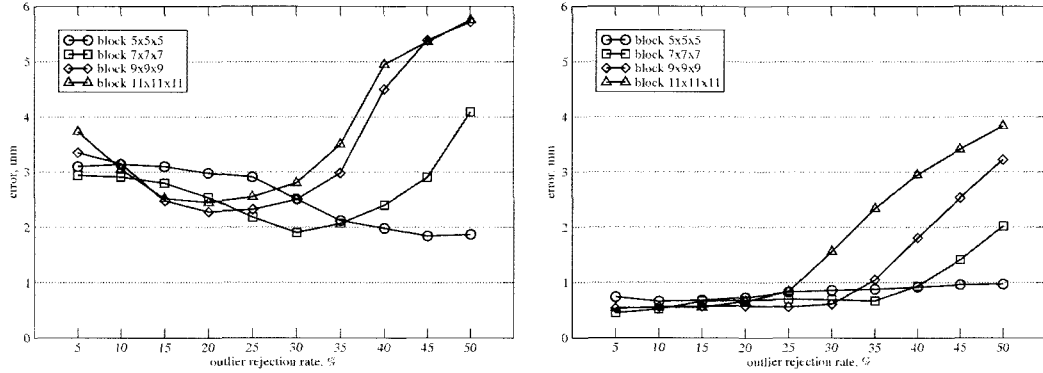
**Table 4.6:** Absolute improvement in accuracy using optimal  $B$  and  $r_2$  compared to  $B = 7$ ,  $r_2 = 25\%$ (mm)

Case	1	2	3	4	5	6	7	8	9	10	11	12
1	0.2	0.1	0.2	0.2	0.3	0.1	0.3	0.4	0.0	0.2	0.2	0.3
2	0.2	0.0	0.0	0.0	0.2	0.3	0.3	0.2	0.5	0.3	0.1	0.2
3	0.6	1.0	0.2	2.9	0.0	0.1	0.3	n/a	n/a	n/a	n/a	n/a
4	0.9	0.8	0.2	0.4	0.7	0.5	0.7	0.4	0.3	0.7	0.7	n/a
5	0.0	0.3	0.0	0.0	0.8	0.5	0.4	0.0	0.3	0.4	n/a	n/a
6	0.2	0.1	0.2	0.1	2.0	0.1	0.0	0.1	n/a	n/a	n/a	n/a

One approach to address the problem of sub-optimal parameter selection is to use *speculative execution* of registration, as suggested by Ino et al. [110]. The idea of speculative execution is to compute multiple registrations on the same input data using different parameter settings. Variability between the results can then be used to estimate sensitivity of the method, e.g., by automatically calculating certain metrics to assure registration accuracy. A number of such metrics for NRR have been proposed by Christensen et al. [49]. Archip et al. [8] used Hausdorff distance between edges identified automatically to prospectively evaluate the accuracy of the registration achieved by the algorithm of Clatz et al. [53]. Recently, we proposed a robust modification of the HD metric, which allows local evaluation of the registration accuracy in a region of the image [71]. Automated approaches for NRR assessment are particularly useful in the context of the speculative execution.



**Figure 4.4:** Distribution of landmark errors before registration ("x"), and after registration with the default parameters ("+"): case 2 (left), case 4 (right).



**Figure 4.5:** Influence of the block size and rejection rate on landmark error: case 1, landmark 5 (left), and case 1, landmark 12 (right).

One of the practical problems in performing speculative registration is the enormous amount of computation required. For example, based on the evaluation results presented earlier in Table 4.1, we need more than 11 hours of computation on a high-end workstation (the timings were collected for the highly optimized code on Intel Xeon 3.7GHz), to sequentially complete the parameter study for block matching only. Considering that there may be 3-4 different similarity metrics that must be evaluated, the range of valid values for the outlier rejection in the solver, and the need to assess the accuracy of each NRR result, the total time required to perform exhaustive evaluation on a *single dataset* is in the order of days of sequential processing.

While the implementation presented in the previous Section proved to be practical and useful during neurosurgery research studies, it has a number of deficiencies. These deficiencies are particularly important in the context of using the framework for speculative NRR computations. Although the idea we described to manage workload distribution is not constrained to be used with two clusters, in practice it is not straightforward to extend this implementation to larger scale.

As discussed earlier, we were able to disable the batch scheduling system to ensure that the NRR computation gets immediate access to the SciClone resources. Direct access to computational resources at any other non-affiliated computational cluster cannot be arranged, and automatic handling of the interaction with the batch scheduling systems must be part of the framework. Security is a critical concern for NRR, since the patient data is passed over the unsecured network. In our case, the communication was done via authenticated secure shell sessions (SSH). With the growing number of computing sites located in different administrative domains, manual user authentication at each of the sites is not an option. Speculative execution and parametric studies of NRR motivate the use of highly distributed infrastructure with the established mechanisms for security and authentication, data transfer and resource management. Implementation of speculative registration should be portable and flexible to enable easy inclusion of new computational resources.

The idea of harnessing the power of the large number of computational resources to solve a single problem has earlier motivated the concept and development of *Grid computing*. Following the definition, computational grid is a “...hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to

high-end computational capabilities” [83]. The three defining features that distinguish Grid from cluster computing have been outlined by Foster in [81]. First, the resources involved in the computation must not be subject to centralized control. This facilitates expansion and openness of the Grid. Second, Grid infrastructure must be built based on “...standard, open, general-purpose protocols and interfaces.” Third, Grid must be capable to deliver nontrivial qualities of service.

In the last decade significant effort has been focused on development of the supporting standards and software [24], deploying production grid systems worldwide and porting applications on those systems [154]. One such production system under continuous improvement and development is USA-based TeraGrid [188]. As of May 2007, TeraGrid was connecting 11 high-end computational sites within the USA, providing “...more than 250 TFLOPS of computing capability and more than 30 petabytes of storage” and therefore making TeraGrid “...the world’s largest, most comprehensive distributed cyberinfrastructure for open scientific research” [188]. Currently, TeraGrid connects 11 computational centers providing cumulative peak performance of 1124 teraflops. The capabilities of TeraGrid are continuously growing, providing computational and storage resources otherwise unavailable to any single research institution worldwide.

The computing resources of TeraGrid cyberinfrastructure satisfy the definition of a Grid system. The resources of TeraGrid are not subject to centralized control: each of the 11 computational sites represents a separate administrative domains. The TeraGrid infrastructure is built based on open, community-defined standards. The core document that defines architecture of the Grid, as it is implemented in TeraGrid, is Open Grid Services Architecture (OGSA) [84]. OGSA provides a high-level definition of for

most of the services that are utilized on a Grid: security and authentication, resource management and data transfer. Open Grid Services Infrastructure (OGSI) is another community standard, that specifies technical details of how Grid services must be implemented [80]. In the latest developments, OGSI has been superseded by the extensions to Web Service Resource Framework (WSRF), which defines procedures for accessing and providing open services [4]. Following OGSA and WSRF one can develop an open Grid infrastructure, and access the resources of such a Grid. Globus Toolkit [3] is an open-source software that implements the standards defined by OGSI. Globus Toolkit is the core middleware that enables TeraGrid implementation.

The features of Grid in general and its specific implementation in TeraGrid national cyberinfrastructure make it the platform of choice for the highly demanding multi-parameter computations of NRR. The capabilities of the Grid have been previously recognized and used for a number of medical applications. In the next Section we discuss the applications of Grid computing for clinical problems, and in non-rigid registration in particular.

#### **4.2.2 Grid Computing for Medical Applications**

Concurrently with the development of the grid technology, we witness significant advances in medical image acquisition, analysis and visualization. Medical images of high resolution and of different modalities are essential in prevention, diagnosis and treatment of many diseases. As the related technology is improving and clinical studies complete, the demand for analysis of the medical data will be growing. Distributed computational and storage resources, like the TeraGrid, provide a promising platform

to meet such demands. Medical applications that require processing and archival of large amounts of data contributed by geographically distributed medical centers were naturally some of the first to embrace grid computing. Representative projects include MammoGrid (maintenance and analysis of mammography data; Europe) [79] and Biomedical Informatics Research Network (BIRN) (analysis of functional MRI; USA) [28]. The infrastructures developed by these projects (by design) are highly specialized for the application domain.

In the context of our project, we are interested in using *general purpose* research grid environment for the NRR computations. This idea has been evaluated in the past for different medical applications. Dong et al. [69] present results of modeling whole human arterial tree on TeraGrid. Grid Enabled Neurosurgical Imaging Using Simulation (GENIUS) project [130] targets modeling and visualization of cerebral blood flow within 15-30 minutes to assist in surgical planning. The common feature of these two applications is that they show linear scalability up to thousands of cores. Therefore, they can leverage virtually any large number of nodes available to the application.

The lack of global TeraGrid resource management system makes it difficult to use large number of nodes at different sites for cross-site runs without prior arrangements. The aforementioned projects circumvented this problem by using the advance reservation capabilities available at some TeraGrid sites. However, advance reservations are not generally favored by the site administrators, as they tend to reduce the resource utilization [146]. This is possibly the primary reason why advance reservations are not available at most of the TeraGrid sites. Moreover, in the case of intra-operative processing for IGNS, the actual time of the surgery is not known precisely well in advance.

The surgery may also be canceled or postponed due to the last minute considerations, e.g., changes in the patient's condition. The GENIUS project [130] considered the use of SPRUCE [22], urgent computing system under development on TeraGrid. This idea is more practical, as it allows to dynamically affect the scheduling decisions to favor selected jobs. In practice, to the best of our knowledge, only one site of the TeraGrid (UC/ANL) provides SPRUCE-directed job preemption, and at some other limited number of sites SPRUCE can be used to schedule the SPRUCE jobs to run first, once the resources become available.

Non-rigid image registration is usually amenable to parallelization. However, its scalability is highly dependent on the specific algorithm used, and the NRR methods known to us [75, 48, 112, 198] do not require more than hundreds of nodes. Still, such computational resources are not available locally at most hospitals, and a number of groups studied how the grid resources specifically can be used to increase the performance of medical image processing computations. Lippman and Kruggel [127] present a framework for intra-operative image registration on the grid. However, they use customized grid environment (GEMSS [92]), and do not evaluate the procedure on a general research grid, like the TeraGrid. Stefanescu et al. [187] describe a parallel implementation of non-rigid registration deployed on a cluster of workstations as a web service (we evaluated this approach in [48]). Gropp et al. [100] study rigid registration, which is a computationally simpler problem, and discuss its different aspects in the context of grid computing. Glatard et al. [94] present a grid-enabled service-based infrastructure for comparing the performance of rigid registration implementations. Their work does explore the use of general-purpose grid resources, and uses workflow model for describing

and scheduling the execution. However, their evaluation is done using the European grid infrastructure. The approaches to grid application development and workflow definition are different from those that can be applied on TeraGrid, and cannot be directly used in our application. In general, we observe that the studies of the NRR methods that are used in practice do not leverage general purpose grid systems.

### 4.2.3 Implementation

We leverage the cluster-based implementation presented earlier to develop the Grid-based NRR infrastructure. The Grid-based implementation follows the workflow design. In a workflow, the computation is coordinated among the participating resources by processing tasks according to a defined set of rules to achieve an overall goal [107]. Workflow can be implemented by manually organizing the interaction of the components, but in practice computational workflows are automated. Workflow management systems are used to define and execute workflows, and organize the results produced during their execution.

Essentially, our cluster-based implementation of NRR represents a workflow, since the processing is not done in a monolithic component. The processing steps are loosely coupled, with the results of pre-processing passed to intra-operative computation as files. However, the coordination of the workflow is not done by a generic workflow management system. This significantly complicates the task of migrating, porting and extending the currently defined workflow.

Without going into the details of OGSA or Globus Toolkit architecture, consider the high-level procedure of job submission to a Grid system. Before access to the Grid



resources can be granted, the user and the resource must be mutually authenticated. Globus Toolkit implements authentication procedure using public key cryptography facilitated by X.509 certificates [202]. Every user or resource participating in the Grid activities must possess a valid certificate issued by a recognized Certificate Authority (CA). Note, that since the centralized control is not present in a Grid system, each of the participating administrative domains requires separate mutual authentication. In presence of multiple resource providers, the task of authentication may become unmanageable. This problem is resolved in Globus Toolkit by *proxy certificates*, which are issued on behalf of the user and delegate restricted credentials within the Public Key Infrastructure [202]. Proxy certificates allow to implement the so called *single sign-on* feature, when the user needs to be authenticated only once to use all of the Grid resources, given all of those resources recognize a common Certificate Authority.

After authentication the user is granted access to the Grid resources. OGSA defines that the Grid resources are available via Grid services, which are extensions of Web services. Again, due to the lack of centralized control, the management of the resources at each specific Grid site is done by a *Local Resource Manager*, which is usually a batch scheduler like PBS or LSF. OGSA defines a unified way to discover those resources, submit new job requests, monitor the job status, and communicate the results of the computation back to the client. Within the Globus Toolkit, resources at a particular site are managed by a Grid Resource Acquisition and Management system (GRAM) [73]. GRAM is a service that can be accessed via the site-specific public contact address, and allows to abstract specifics of local resource management with a well-defined API.

Finally, the last component which is critical for executing jobs on a Grid resource

is the mechanism to provide data transfer functionality. The conventional means of transferring the data cannot be used due to their incompatibility with the standards defining Grid Security Infrastructure, insufficient reliability and performance. The data transfer protocol that is used in Globus-supported grids is GridFTP [2]. From the user perspective, GridFTP is not significantly different from regular FTP, but it provides robust, secure and efficient file transfer, particularly optimized for transferring bulk data.

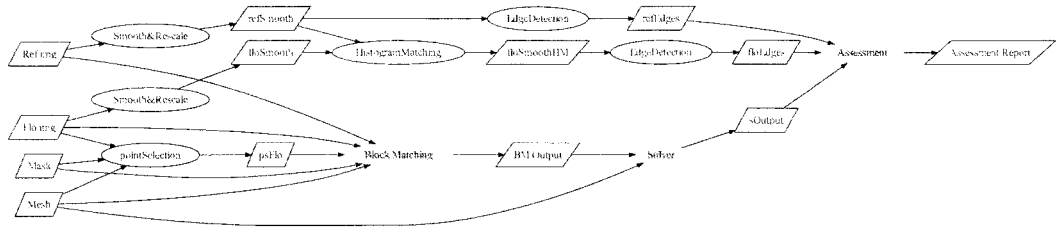
Each TeraGrid site runs its own GRAM and GridFTP servers, as part of server installation of Globus Toolkit. These services can be accessed via the publicly available portable Globus client implementations.

While TeraGrid resources can be accessed directly for individual job submission and data transfer, doing this on the large scale or as part of workflow execution is not practical. We adopted Swift workflow scripting and management system [210] to implement and deploy NRR workflow. This choice was motivated by the following observations.

1. Swift has been developed and evaluated to support grid implementations that are based on Globus Toolkit, which allows to use this system without any modifications to schedule workflows on TeraGrid.
2. Swift uses a scripting language, SwiftScript, to define the workflow. SwiftScript is a powerful way of abstracting interaction of the processing tasks, which allows to define composite data inputs, dependencies between the processing tasks, and provides familiar control structures like loops and conditional structures, which allow flexible control over workflow definition and execution.

3. Swift execution management system is highly flexible in its capabilities of interacting with the various resource providers. It uses programmatic access to Globus-managed resources by means of CoG Karajan [195]. Swift provides the flexibility of using the same unmodified workflow description script to interact with conventional resources, like those managed by local batch scheduler, or even accessed in time-shared mode via direct login.
4. Swift implements basic fault-tolerance of workflow execution at the individual task level, which is critical for NRR computations. In case a particular task fails to deliver the expected output data, Swift will re-schedule its execution, possibly on a different site.
5. Task-level load-balancing is one of the functionalities provided by the Swift workflow execution infrastructure. The execution traces for the same computational task are continuously collected. These traces are used to dynamically select the best performing site when the task is scheduled next.
6. Data provenance mechanisms in Swift allow to abstract the site-specific file management issues. All the inputs and outputs of the individual tasks are transparently communicated, and can be referenced within the SwiftScript through simple data pointers.

Swift provides the means to define and execute the workflow, which consists of individual processing tasks. Each of the processing tasks must be available as an executable at each of the sites, that will be involved in the workflow computation. The details of running a specific task are provided to Swift in the so called *translation catalog* available



**Figure 4.6:** NRR workflow diagram for single registration execution (shaded are the time-critical components of the workflow).

at the client (submission) site. The translation catalog contains the identifier of the remote site where the executable is installed, together with the optional information on its invocation. For example, in order to execute an MPI-parallelized task, the number of nodes must be explicitly specified. In order to guarantee that the task execution will be able to complete on the systems managed by a site-specific Local Resource Manager (LRM, e.g., PBS or LSF), estimated wall clock time must also be included in the task description.

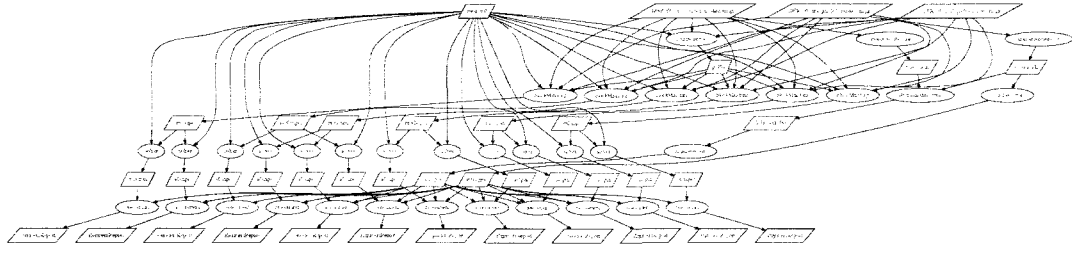
The execution sites are described in the Swift *site catalog*, located at the workflow submission site. The site catalog contains a *pool element* for each execution site. The pool element specifies the details about file transfer and task invocation services at the given site. For the local resources, the file transfer method can be a local copy operation supported by the filesystem, and task invocation is local execution. For the pool elements corresponding to the TeraGrid sites, file transfer is done through GridFTP (each TeraGrid site provides GridFTP service URL for file operations), and the remote resources are accessed via the Globus GRAM4 [82] gatekeeper endpoint. Task submission for the remote sites is a complex operation that involves mutual authentication, secure transfer of the job descriptor and monitoring of the job status. However, all these

complex details are handled transparently by the Swift execution engine.

In order to implement the Grid-based workflow using the Swift system, we restructured the cluster-based implementation to have loosely-coupled computation of NRR with the separate tasks for each processing step. As part of the preparation for Swift workflow execution, each of the executables must be compiled at a remote site, and basic profiling of the code is required to obtain the approximate execution time on typical input data. This is needed to estimate the expected walltime for each task.

**Single NRR workflow** The NRR workflow diagram of a single NRR procedure together with the accuracy assessment module is shown in Figure 4.6 (the development of the accuracy assessment module is discussed in the next Chapter). The block matching task is parallelized using MPI, and has been deployed on the TeraGrid sites for remote parallel execution. The other components of the workflow are executed on the local resources (single node of the W&M SciClone cluster). This NRR workflow corresponds to the base case for computation supported by our cluster-based implementation we discussed earlier, augmented with the accuracy assessment module.

**Speculative NRR workflow** Speculative execution of NRR literally requires just a few additional lines of SwiftScript to iterate through the user-defined parameter space. Using the Swift filesystem mappers, each intermediate and the end result of the speculative computation is given a unique self-descriptive name, and stored in a user-defined location on the local filesystem. For the speculative NRR workflow, block matching, solver, and NRR assessment tasks are executed on the remote TeraGrid resources. The rest of the tasks are computed locally. The complete diagram of the speculative execution of the NRR workflow for the parametric study of a parameter subset of block



**Figure 4.7:** NRR workflow diagram for speculative execution of the registration with  $B \in \{7, 9\}$ ,  $W \in \{15, 17, 19\}$ , and  $r_2 \in \{0.2, 0.3\}$  (this Figure is only for the purposes of illustrating the connections between the workflow components).

matching block size ( $B$ ), search window size ( $W$ ) and the value of fraction of the rejected registration points ( $r_2$ ) is shown in Figure 4.7.

We believe the Swift-based workflow description of NRR as a significant improvement over the custom implementation in [48]. In this new implementation we are able to easily include additional computing resources for NRR execution. Modification of the procedure to include new intermediate steps, or even major changes in the workflow (like speculative execution) minimum updates to the existing workflow description.

#### 4.2.4 Evaluation

We used three sites of TeraGrid (TACC Lonestar, NCSA Abe, NCSA Mercury) to install components of the workflow that we selected for remote execution. These sites were chosen among the currently available 11 TeraGrid sites because they provide the most convenient development environment (multi-core Intel nodes). Our implementation of the NRR has been originally developed and tuned in similar environments. We expected that porting of the codes would require less effort for the selected sites as compared to, for example, BlueGene/L or Cray XT3 platforms, also available within TeraGrid.

Porting of the NRR code included setup of the supporting open source libraries,

which were not available at the sites we used. As part of the setup, we had to update the local firewall configuration on SciClone to allow incoming connections from the TeraGrid sites. Open firewall ports is a necessary condition for using remote Globus GRAM service. Access to the TeraGrid resources is simplified with single sign-on procedure and Globus Security Infrastructure [82]. The user password is entered once at the local site (SciClone), and after that all job submissions and data transfers to and from the TeraGrid sites are transparent from the SciClone user perspective.

The hardware and software configuration for the selected sites together with the benchmarked execution times for the workflow components are summarized in Table 4.7. There are two observations to be made. First, hardware configurations were introduced at different times on different TeraGrid sites. Therefore, we observe significant heterogeneity and disparity in processing power for the available nodes. Second, it is not possible to use the same compiler and MPI implementation on all TeraGrid sites. This creates significant difficulties for porting legacy codes, since different compilers may require different configurations and, sometimes, may simply fail to compile the code. As the results of the benchmark runs for the installed components show (see Table 4.7), execution times vary significantly for the same level optimization with different compilers, exaggerated even more by the non-uniformity of the clock speeds.

**Single NRR instance execution** We performed experimental evaluation of the single instance of NRR workflow (see Figure 4.6) at each of the three sites of the TeraGrid in Table 4.7. In each case, block matching was executed on 10 nodes at a remote site, while the rest of the workflow components were run on the single node of the SciClone cluster. The execution timeline for NCSA Abe cluster is summarized in Fig-

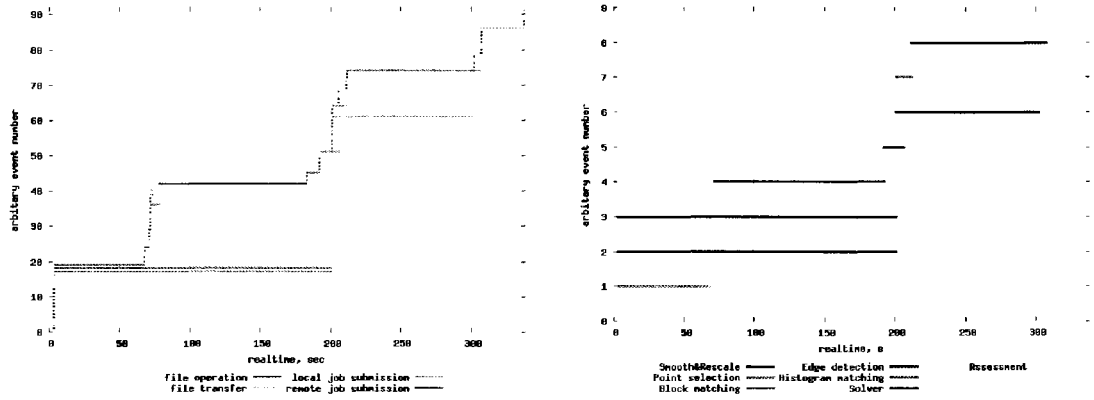
**Table 4.7:** Summary of the hardware and software environments, and processing times on a single node for the time critical components of the NRR workflow: sequential block matching (C1), solver (C2), registration assessment (C3). Benchmark setup: 55K registration points,  $B = 7$ ,  $W = 17$ ,  $\approx 4.5K$  mesh vertices.

	W&M SciClone	TACC Lonestar	NCSA Abe	NCSA Mercury
	Opteron 2218	Intel Xeon	Intel 64 Clowertown	Intel Itanium 2
	2.6GHz, 8GB	2.6GHz, 8GB	2.33GHz, 8GB	1.5GHz, 4GB
	gcc 4.1.2, -O3	icc 10.1, -O3	gcc 3.4.6, -O3	gcc 3.2.2, -O3
	MVAPICH2-1.0.1	MVAPICH-1.0.1	MPICH-VMI 2.2.0-3	MPICH-GM 1.2.5
C1	275 sec	294 sec	418 sec	608 sec
C2	6 sec	27 sec	12 sec	21 sec
C3	24 sec	22 sec	182 sec	495 sec

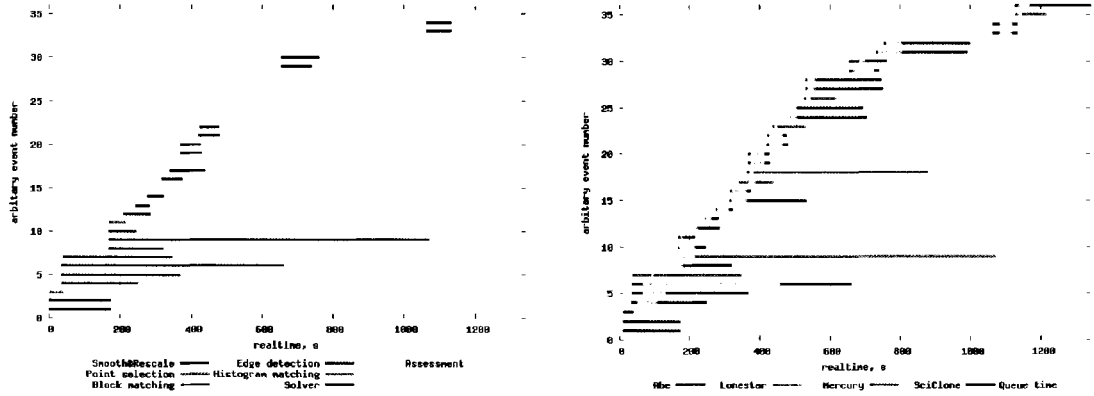
ure 4.8. Although we did not use any advance resource reservation capabilities, which are available at some sites of TeraGrid, we were able to complete execution and collect traces without major queuing delays at each of the tested sites. We observed that the file transfer delays (maximum below 30 sec) were negligible in comparison to the speed improvement in block matching computation due to parallelization in each case. This is in part attributed to high-performance GridFTP protocol, which in our experience provides faster transfer of image data as compared to SFTP that we used in [48]. The NRR workflow was completed in less than 6 min. This includes point selection component, which can be evaluated before the time-critical part of the computation.

**Speculative NRR execution** The timeline of speculative NRR execution for a relatively small parameter search space is shown in Figure 4.9 (this corresponds to the workflow diagram in Figure 4.7). The total time required to complete such speculative execution was in this particular case about 20 min. Observe, that the execution of the single block matching component on Lonestar site reportedly took more than 800 sec in the speculative run. We attribute this to the intermittent overload of the remote GRAM service, which might have prevented the timely propagation of the job update status to





**Figure 4.8:** Execution timeline for single instance of NRR on NCSA Abe site. Left: file operations and task execution summary. Right: task executions colored by task type (file operation times are included).



**Figure 4.9:** Speculative NRR execution timeline for the parameter space in Figure 4.7. Left: task executions colored by the execution site. Right: tasks executions colored by the task type (file operation times and queued times are included). *This figure is best viewed in color*

the submitting client.

The resources available within the TeraGrid are sufficient for concurrent execution of all independent branches of the workflow shown in Figure 4.7. Therefore, in the ideal case, ignoring all overheads, one could expect that the total time required for speculative execution would be equal to the longest execution of single registration. We measured this time by executing a single instance of NRR on each of the sites, and the maximum execution time we observed was under 10 min (without accounting for the queue delays).

In practice, a number of factors are contributing to the increased processing time, as compared to the ideal case:

(1) As we show in Table 4.7, time required to complete the same task varies depending on the selected TeraGrid site. The current procedure of the workflow scheduling by Swift execution engine does not take into account execution time during the process of resource selection (the same task is available for execution at all three sites).

(2) Due to the fact that the resources at each site are available through the remote LRM, some of the jobs will be delayed in the queue. In our case, as shown in Figure 4.9, cumulative queuing delays for all jobs were about 20 minutes (!), with the maximum individual queued time of 385 sec.

(3) There are certain constraints in the Swift scheduling procedure, that limit the maximum number of jobs that are allowed to run at a given site at the same time (the so-called *job throttle* limit). Because of this, there only 4, not 6, concurrent executions of block matching initially, see Figure 4.9. The job throttle limits are in place to prevent overloading the GRAM service with the large number of concurrent job submissions.

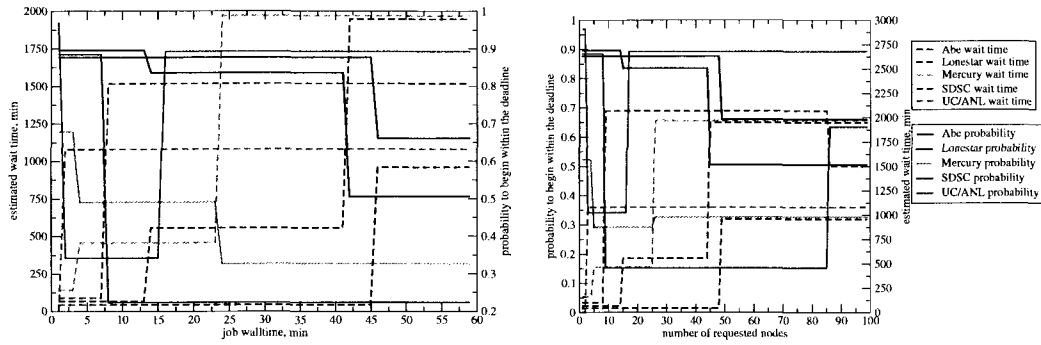
(4) There are multiple minor overheads due to file staging (transfers of input and output files), task scheduling, Swift book-keeping, etc. Current scheduling policy used in Swift does not account for minimization of the file transfers (e.g., in Figure 4.9, observe that assessment task is performed on sites different from where solver was run to produce registration results). These are, however, minor issues compared to the first three items.

There is on-going work to address the limitations that negatively affect the performance of throughput-oriented applications, like speculative NRR, on TeraGrid. Although in our evaluation we did not experience queue delays, they are to be expected in

the general case. We expect one of the problems that will need to be addressed for the intra-operative application of speculative execution will be how to reduce the impact of the queue delays at the remote site. One approach, which has been considered by the community, is to estimate the remote LRM queue wait time, and use to adjust the task scheduling (queue prediction services are provided by all the TeraGrid sites). As we show in Figure 4.10, wait time is dependent on the job execution time and the number of processors requested. Moreover, scheduling policies are not uniform across the TeraGrid sites (e.g., some sites favor long jobs, and some sites give preference to short-running tasks), and are changing with the time. The dynamic selection of the job configuration can potentially improve job throughput.

Conceptually different approach is currently under development within the broader grid workflow community. The idea is to use multi-level scheduling. Using this approach, a group of nodes within a cluster is allocated, and then used to schedule multiple smaller tasks of the workflow. Falkon [165] implements this concept, and has been shown to achieve very high throughput for the applications they evaluated. The use of multi-level scheduling also helps to alleviate the restriction on the maximum task concurrency. The limits currently in place are the safeguards to prevent overloading of the remote GRAM service by the large number of jobs. Scheduling of smaller jobs within the larger allocation would not require interaction with GRAM. We expect to see significant benefits in using Falkon and other similar techniques under development (e.g., *Coasters* resource provider developed within Swift) for executing speculative NRR workflows.

File transfer costs were insignificant for the reduced parametric studies we performed. However, for the full-scale parametric studies workflow scheduling will need to take into



**Figure 4.10:** Estimation of queue wait times and probability that the estimate is correct by QBETS service [145] for some of the TeraGrid sites (job start within 30 min with requested 95% confidence in the result). Left: estimation as a function of job walltime, 16 nodes requested. Right: estimation as a function of the number of nodes, 30 min job wall time.

account data affinity to minimize communication. Also, since the execution time of the solver module is comparable with the time required to transfer its input, it is very inefficient to schedule solver as a separate task. Multi-level scheduling and job clustering may improve execution locality and minimize file transfers.

#### 4.2.5 Discussion

As a result of this work, we developed a workflow-based implementation of the single and speculative workflows for the NRR method, and completed their preliminary evaluation at three TeraGrid sites.

Speculative execution with different combinations of parameters could be a necessary precaution in the applications, where accuracy is important, but the rules for selecting optimum parameter combinations are not clear. In the context of neurosurgery, speculative execution can be applied in two scenarios. First, immediately following the acquisition of the first intra-operative scan (but before the skull opening), speculative computation can be initiated to explore those parameters of the algorithm that depend on the properties of the intra-operative image. As an example, this can involve the study

of the best similarity metric to be used in block matching. It was suggested elsewhere that the selection of the best similarity metric depends on the properties of the image, other than its modality [196]. The second application of speculative execution is during the time-critical intra-operative registration.

The implementation we developed is not tied to a specific collection of execution sites, as the framework presented in [48], but is generic enough due to its use of Swift, and the broad range of resource providers supported by Swift (e.g., Globus GRAM2 and GRAM4, PBS and LSF batch schedulers, etc.). Therefore, the implementation can be deployed with minimum effort on any conventional cluster within a hospital, or can be executed on the TeraGrid resources, to enable large-scale population and parameter studies of the NRR method.

The important result of our evaluation is that it is feasible to use TeraGrid resources for intra-operative NRR. Although we were able to complete test runs without significant queue delays for the single NRR instance, advance reservations should be used, whenever possible, to guarantee resource availability during the surgery. Our preliminary results of evaluating speculative NRR workflow executions indicate, that with the current technology we cannot achieve the required task throughput to do speculative execution within the time constraints of IGNS. However, the work under development in the relevant projects, e.g., Falkon [165], is very promising. We expect to achieve significant performance improvements by using multi-level scheduling techniques, as compared to the conventional direct interaction with the remote batch scheduler via Globus GRAM4 for each workflow task. Again, due to our use of Swift for workflow implementation, we can leverage these new execution models without any modifications to the NRR code or

workflow script.

The open question that cannot be answered by this study (yet) is the following. Given the availability of the local mid-size computing cluster, is the effort to migrate an application like NRR on the TeraGrid justified? As we discovered during the course of this work, significant time investment may be required to port a code on the TeraGrid and learn the site-specific usage procedures. Efficient use of the TeraGrid resources is complicated by the queue delays, scheduled and emergency shutdowns of the TeraGrid sites and Globus services we use, and by the technical difficulties we described in the previous Section. On the other hand, although mid-size clusters (100-200 nodes) become more and more ubiquitous, we believe they are not yet affordable by an average hospital or research group. We also believe that local clusters (which by the way are usually also managed by a batch scheduler) are insufficient for large-scale computations like NRR speculative execution, or large-scale population studies. It remains to be seen how practical the full-scale speculative studies can be on the TeraGrid. However, the preliminary results we present are sufficient to affirm the advantages of the Swift-based workflow NRR implementation, which can be used immediately both on TeraGrid and conventional clusters for NRR evaluation. At the same time, to achieve good performance of the speculative NRR execution we need to evaluate non-conventional, throughput-oriented workflow scheduling techniques.

### 4.3 Summary

The ability to deliver the results of NRR within the clinical time constraints is essential. In this Chapter we presented a distributed implementation of the NRR workflow that

makes near real-time execution possible with the use of distributed cluster resources.

The first major difference of the new implementation compared to the original version developed by Clatz et al. [53] is in the restructuring of the registration process to use multiple processing steps within a workflow. Such restructuring allows for the first level of fault tolerance during the execution, and facilitates independent execution of the pre-processing steps before the time-critical computation. The second major improvement we implemented is the high-performance parallel MPI version of the block matching. We used a master-worker load distribution scheme to dynamically adjust to the heterogeneity of the computing resources. In addition to this, master-worker load distribution applied within the multi-cluster environment allows to provide an additional layer of fault-tolerance. The CoW-based implementation presented in this Chapter has been applied to perform non-rigid registration of the pre-operative MRI to the iMRI acquired during the neurosurgery.

The NRR method we study is characterized by the large parameter space. We perform an exploration of a relatively small parameter subspace and present the initial results on sensitivity of the method to the parameter selection. Based on the expert-identified brain landmark accuracy assessment, the registration error is non-uniform over the registered volume. We report the experimental results that show the difficulties in selecting the optimum parameter combination for the given dataset. This difficulty motivates the development of the grid-enabled workflow implementation of the registration, which utilizes Swift workflow management system [210] and the resources of TeraGrid [188] to enable speculative execution of the registration. We present initial results of running parametric search on multiple geographically distributed computational

sites from different administrative domains using the workflow-based implementation.

Compared to the existing distributed systems designed for NRR, the cluster-based distributed registration version discussed in this Chapter was applied and evaluated prospectively during neurosurgeries. The preliminary results of evaluating our distributed workflow implementation on the TeraGrid shows the feasibility of using Grid resources for the needs of intra-operative registration.

## 4.4 Contributions

The contributions of this part of research are the following:

- for the first time ever, results of volume matching registration delivered intra-operatively within 5 minutes;
- presented NRR infrastructure routinely used for image registration during IGNS research studies (>11 MRT surgery cases in 2006-2007);
- feasibility study of using TeraGrid for neurosurgery registration research;
- practical NRR workflow-based Grid-enabled NRR implementation deployed on TeraGrid.

The results presented in this Chapter have been previously published in the following papers:

1. N.Archip, O.Clatz, S.Whalen, D.Kacher, A.Fedorov, A.Kot, N.Chrisochoides, F.Jolesz, A.Golby, P.Black, S.K.Warfield. Non-rigid alignment of pre-operative MRI, fMRI,



and DT-MRI with intra-operative MRI for enhanced visualization and navigation in image-guided neurosurgery. *Neuroimage*, 35(2):609-624, 2007

2. N.Chrisochoides, A.Fedorov, A.Kot, N.Archip, P.Black, O.Clatz, A.Golby, R.Kikinis, S.K.Warfield. Toward Real-Time, Image Guided Neurosurgery Using Distributed and Grid Computing. *IEEE/ACM Supercomputing* 2006
3. N.Archip, A.Fedorov, B.Lloyd, N.Chrisochoides, A.Golby, P.Black, S.K.Warfield. Integration of patient specific modeling and advanced image processing techniques for image guided neurosurgery. *SPIE Medical Imaging 2006: Visualization, Image-Guided Procedures, and Display*, 2006
4. A.Fedorov, N.Chrisochoides. Toward Improved Tumor Targeting for Image Guided Neurosurgery with Intra-operative Parametric Search using Distributed and Grid Computing. *IEEE International Parallel and Distributed Processing Symposium, NSFNGS Workshop*, pp.1-5, 2008

## Chapter 5

# Registration Accuracy

## Assessment

The goal of non-rigid registration for image-guided neurosurgery is to update the pre-operative MRI to accurately represent the deformed brain anatomy. Therefore, critical decisions made by the surgeon may be based on the registered images. Lack of accuracy in registration result may lead to imprecise location of a life-critical brain structure and irreparable damage to the healthy brain and to the patient. At the same time, accuracy of a given image processing method is only one of the components that has to be considered while comparing it with other methods developed for the same task [115]. Such complete evaluation is known as *validation*. Validation is a mandatory step that must be completed prior to the application of any method in the clinical context. Although there are no standards for performing validation, some of the criteria that are commonly used in addition to accuracy are reproducibility, robustness and consistency of the method [97, 114].

In general, complete validation is a difficult task, because the true result expected from the image processing method is rarely known. In part this is explained by the incomplete information about the object available from the medical image. This incomplete data is corrupted by multiple sources of error and uncertainty, which may be related, for example, to biological variability in the imaged tissue or to the specifics of the image acquisition process. These problems are further exacerbated by the variability of interpretation of the same image by different experts (*observers*, using the validation terminology [115]). Another difficulty in assessing and comparing accuracy of different methods is the lack of common datasets and the lack of common metrics, which also depend on the application and specific processing method.

For the practical purposes, *accuracy* of the image processing method is a required prerequisite, and one of the key components of the complete validation of the technology [89]. In the context of image registration, accuracy can be defined as determination of alignment between the points in one image with the corresponding (homologous) points in the other image [101]. In order to evaluate the accuracy of registration, we need so called *gold standard*, or *ground truth*, – the true transformation that recovers the alignment of all the homologous points in two images. If such gold standard is known, we can use it to compute a “figure of merit” to quantify the distance to ground truth. Such an approach allows to formalize accuracy validation as hypothesis testing, with the known techniques and tools for its statistical analysis [114]. However, the ground truth is rarely available for the general clinical case, which makes calculation of the figure of merit problematic, and requires different means of validation.

Jannin et al. [114] define four types of datasets commonly used in validation, de-

pending on the availability of ground truth in that data: (1) numerical simulations, e.g., [167]; (2) realistic simulations from clinical data sets [172]; (3) physical phantoms; or (4) clinical datasets. For the first two types of validation data, the ground truth is known. In the case of phantom studies, the ground truth can be measured or recovered. Some examples of phantom studies for brain registration were employed in accuracy assessment studies previously [163, 65, 5, 117]. However, the established accuracy on synthetic data of phantom data does not eliminate the need of evaluating the method on real data. In the cases when clinical data is used for validation, ground truth cannot be recovered, and a *substitute* of ground truth is usually given by experts in the imaging for the specific application [114]. For example, in the case of non-rigid registration, homologous points may be manually identified in the images before and after registration. The problem associated with this approach of estimating the ground truth is the consistency of locating the homologous points. Generally, ground truth obtained with the aid of human observers is called a *bronze standard*, or *fuzzy gold standard* [114].

In the previous Chapter we discussed the distributed framework, one of the potential uses of which is to facilitate speculative execution of non-rigid registration. One of the approaches to perform such execution assumes the capability of evaluating and comparing the accuracy of the solution obtained with the different registration parameters during registration of clinical data. This requires some kind of ground truth, or its substitute, to be available. Obviously, derivation of such ground truth cannot possibly involve human observers, because of the limited time and prohibitively large amount of data.

In this Chapter we develop and evaluate a novel metric that can be used to assess

the accuracy of non-rigid registration automatically. This metric is based on the Hausdorff distance (HD) between the *edges* in the image pairs before and after registration. This measure can be evaluated automatically, since it does not require the known correspondence between the image points, and does not require human interaction due to automatic edge identification. The use of such a metric can not substitute accuracy validation. However, in the conditions when the time for validation is limited, automatic measure may be the only option.

We evaluate the recent advances in improving robustness of HD, and apply the novel accuracy assessment measures we develop to evaluate the accuracy of pairwise alignment for brain MRI under non-rigid deformation. Based on the results of our evaluation, the presented approach significantly improves the accuracy of the previously used alignment metrics that are based on the conventional HD. The implementation of the presented approach is available as open source software, accompanied by the detailed description of the parameters we used to obtain the reported results on publicly available BrainWeb data [27]<sup>1</sup>.

## 5.1 Related Work

In this Section we present related work in the development of the quantitative measures that allow to assess the accuracy of registration. Rigid registration, where alignment between images is established with the aid of global translation and rotation has been the area where such metrics were studied and developed originally. In a number of medical applications, particularly in neurosurgery, the problem of registering the image with the

---

<sup>1</sup>BrainWeb simulation interface can be accessed at <http://www.bic.mni.mcgill.ca/brainweb/>.

position of the patient during the procedure can often be formulated as the alignment of two sets of points. Such points, which can be reliably identified, are called *fiducial* points. Examples of such fiducial points can be anatomical landmarks, or artificial landmarks implanted for the purposes of registration. The sets of landmarks identified in both images can be used for point-based rigid registration, which seeks least-squares fit of the corresponding fiducial points. Such an alignment problem can be reduced to the Orthogonal Procrustes problem, that has known solutions [174].

One measure of error for point-based registration that can be easily evaluated is fiducial registration error (FRE). FRE is usually reported as the root mean square (RMS) distance between the corresponding fiducial points after registration defined by some rigid transform  $\mathbf{T}$  [101]:

$$FRE^2 = \sum_i^N |\mathbf{T}(\mathbf{p}_i) - \mathbf{q}_i|^2.$$

The FRE measure is directly related to the reliability of locating fiducial points in the images. Such reliability is defined as the fiducial localization error (FLE), that captures the difference between the assumed position of the fiducial and its ground truth location. The relationship between FRE and FLE has been established by Sibson et al. [181]:

$$RMS(FRE) = \sqrt{(1 - 2/N)}RMS(FLE).$$

The fundamental problem in using FRE as the measure of registration accuracy is that small FRE is not a sufficient condition for small error at any point in the image. Target registration error (TRE) [181] measures the error in the clinically important region, e.g., at the boundary of the tumor, which is of practical importance. There

are theoretical results that relate TRE to FRE in the case of rigid registration [78]. The FRE, FLE and TRE measures were introduced by Maurer et al. [132] and proved effective for evaluating point-based registration methods. However, the problem is much more complex for non-rigid registration. Due to the fact that potentially arbitrary local deformations can be introduced by registration, any set of points can be brought into alignment, even if those points are totally irrelevant. The connection between TRE and FRE cannot be established for non-rigid registration, and TRE can rarely be evaluated.

Few studies exist, where TRE was evaluated for non-rigid registration based on the simulated ground truth, e.g., see Schnabel et al. [172]. In general, however, TRE is difficult to estimate, because the task of finding features or landmarks in the clinically relevant region may be very difficult. An easier task is often to outline the corresponding regions in the compared images, and evaluate the so called *regional correspondence* [60]. The two most common quantitative measures for estimating region overlap are the Dice similarity coefficient (DCE) [68] and the Tanimoto coefficient (TC), also known as the Jaccard coefficient [113]. For two overlapping regions  $A$  and  $B$  DSC and TC are defined as follows [59]:

$$TC = \frac{N(A \cap B)}{N(A \cup B)}, \quad DSC = \frac{2N(A \cap B)}{N(A) + N(B)},$$

where  $N()$  corresponds to the number of voxels that correspond to the segmented region. DCE and TC evaluate the overlap of two labels, but cannot be applied to evaluate more than two volumes, and do not allow to compare the degree of mis-alignment. In particular, the same overlap coefficient can correspond to two different cases that differ if we consider point-wise registration error. Some of these deficiencies have been addressed by the generalized overlap measures developed by Crum et al. [59].

In general, very few comprehensive validation studies of non-rigid registration methods exist. We refer the reader to the reports by Grachev et al. [98] that relies on anatomical landmarks, and by Hellier et al. [104] that uses regional overlap measures.

The major difficulty in applying overlap measures and TRE in assessing non-rigid registration accuracy remains the definition of the relevant regions or points, which in general requires involvement of human observer. Complete validation studies that rely on such data necessarily include analysis of the variability of rater results, and possibly estimation of ground truth from the inputs of multiple experts, as proposed by Warfield et al. [201]. This problem has been recognized by the community, and a number of alternative assessment measures were developed that do not require human observer.

Consistency of registration can be evaluated based on analysis of the transformation obtained with the registration tool. Fitzpatrick et al. [101] describe the idea of registration circuit that is constructed by registering image  $A$  to  $B$ ,  $B$  to  $C$  and  $C$  to  $A$ . In the ideal case, each point of image  $A$  can be tracked back to  $A$  through the registration transforms. Christensen et al. [50] suggest the use of inverse consistency metric, which is based on the assumption that the composition of the forward and backward transformations with the same input images should result in the identity transform. Hellier et al. [104] use the Jacobian transform, which is the first partial derivative of the transformation, applied to the deformation field to find the points of singularity. Metrics that are based on analyzing transformations may allow to detect the failure of registration. However, they are not sufficient to conclude success.

Recently, Archip et al. [11] proposed to use the value of mutual information similarity measure between the images to assess the accuracy of registration. This approach has a



number of problems, outlined by Crum et al. in relation to the use of image similarity measures for registration in general [60]. In particular, good image similarity value does not necessarily imply correctness of registration, it does not allow to assess the magnitude of the error, and it cannot be used to analyze spatial error distribution.

The Hausdorff distance is a very common measure in pattern recognition and computer vision to evaluate mismatch between the two sets of points. A number of methods have been proposed to identify features (points, edges, lines) in medical images [37, 29]. The resulting feature images can be used as input for the HD measure. The HD is not based on point correspondence, which makes it somewhat tolerant to the differences in the two sets of features compared. However, it is highly sensitive to noise. A number of robust modifications to the HD have been proposed to suppress the impact of noise and improve robustness of the HD, since one of the first papers in this direction by Huttenlocher et al. [108]. The most recent surveys of the modifications to the HD are available in [209, 21]. Until recently, the HD has found limited use as a measure for the evaluation of image alignment. Peng et al. [156] used a robust version of the HD to register outlines of brain in two dimensions. Morain-Nicolier et al. [139] applied the HD to quantify brain tumor evolution. Finally, Archip et al. [9, 8, 11] assess the performance of non-rigid image registration of brain MRI with the 95% partial HD, but do not discuss the reliability of this approach.

To the best of our knowledge, the HD-based approach to image alignment assessment has not been comprehensively evaluated before. A number of the robust versions of the HD exist, but they have not been evaluated for three-dimensional images and in the context of NRR for medical imaging.

## 5.2 Robust Error Estimation Methodology

The methods we are developing focus primarily on registration assessment for image guided neurosurgery. The objective of the NRR is to align the pre-operative image with the intra-operative data. Consequently, the objective of the evaluation procedure is to confirm that alignment indeed improved following NRR, and quantify the level of mis-alignment before and after registration. We approach this by considering two pairs of images. The first image is always the fixed (target) image used in registration. For the purposes of assessment, the second image can either be the floating image or the registered image. By evaluating the alignment of fixed *vs* floating and fixed *vs* registered images we attempt to assess the error of alignment before and after registration, respectively. However, the formulation of the problem remains the same. Next we describe in detail the processing steps required for the derivation of the alignment metric.

### 5.2.1 Image Pre-Processing

A common procedure that precedes analysis of the images, in particular for medical imaging applications, is pre-processing. The specifics of pre-processing are highly dependent on the application, and may target improvements of the visual appearance of the image for better interpretation by the human observer (e.g., contrast or sharpness improvement, enhancement of certain features), or as a prerequisite in order to improve the quality of the subsequent processing operations. The latter is the reason that necessitates pre-processing for the purposes of registration accuracy analysis.

The objective of the accuracy assessment study is essentially to identify and quantify the differences between the images due to non-rigid deformation of the image features.

Therefore, it is very important that all other sources of difference between the images are suppressed, and the corresponding features are identified in both images.

The first step of pre-processing is to remove small spurious noise features from the analyzed images. A straightforward approach to noise removal in spatial domain is based on linear filtering [96], which is usually implemented as a convolution operation. Convolution is defined by the mask of constant size  $m \times n$  with the coefficients  $w_k$  (in two dimensions), which is also known as a convolution kernel. Convolution of an image with the kernel means generation of a new image, where intensity  $R$  at a pixel location is calculated as follows:

$$R = \sum_{k=1}^{mn} w_k z_k.$$

Here,  $z_k$  are the image intensity values that overlap with the convolution kernel. Convolution based linear filtering allows to easily implement averaging and weighted averaging operations, that suppress the high frequency image features. Gaussian function used to initialize the convolution kernel provides flexibility to precisely control the range of high frequencies in the resulting image by varying the size of the convolution kernel [77]. The convolved image can also be viewed as a result of solving the differential equation for heat propagation with the initial conditions defined to be the intensities of the input image [157].

Application of a linear convolution operation results in an image that appears blurred, with high frequency noise features suppressed. The negative result of such blurring is that the meaningful image features, like the boundaries of distinctive image regions, are also diffused. Perona and Malik studied this problem, and proposed an alternative approach using anisotropic diffusion [157] that limits smoothing at the region boundaries.

We apply anisotropic diffusion filtering to the input images as the first pre-processing step to preserve the region boundaries that define image features, and eliminate image noise.

The basic assumptions made in our accuracy assessment scheme is that the images correspond to the same brain, and they are acquired under the same image acquisition protocol. The second pre-processing operation addresses the fact that the intensities of the image in the corresponding brain regions may be different in each of the images. We improve the contrast of the input images by applying adaptive contrast enhancement algorithm presented by Stark in [185]. Adaptive contrast enhancement is a time consuming procedure, and in the time-critical applications we instead use conventional histogram equalization [96]. Finally, we rescale the intensity ranges of both images to be within the same range.

### **5.2.2 Edge Detection**

The first pre-processing step is needed to prepare the input data for selection of the features, that are used as inputs for the actual accuracy assessment algorithm. Image features are defined by abrupt changes in the image intensity. In the medical MRI, image intensity is related to the physical properties of the imaged tissue. Therefore, changes in intensity correspond to the boundaries between the tissues that have different properties. Edge detection operation attempts to identify the set of connected pixels that define the region boundaries, extracting structural information from multi-valued images.

Robust edge detection is not trivial for realistic images. Sharp change in image intensity is a relative definition, that requires certain threshold to define how sharp that

change should be. The definition of the threshold is related to the normal variation of the image intensities within the homogeneous region and the level of edge detail required.

We use the edge detection filter proposed by Canny [37] to identify the features in the input data. The filter consists of the four main steps [96]. First, the input image is smoothed with the Gaussian filter, that we discussed in the previous Section. Second, the magnitude and angle of the gradient are computed on the smoothed image. High values of the gradient magnitude define the locations in the image that are likely to correspond to the edge features. However, there will generally be multiple non-zero gradient magnitude pixels that correspond to the same edge. The next processing step is non-maxima suppression operation, which is thinning the initial approximation of the edges. This is accomplished by locally estimating the edge direction that is closest to the gradient angle at a point, and discarding the edge points that have their gradient magnitude values less than those of its neighbors. The goal of this step is to produce the edges that are exactly one pixel wide.

The last step of Canny edge detection is concerned with filtering the remaining non-zero gradient magnitude pixels to eliminate the “false” edges. This step is accomplished with two thresholds: a low threshold,  $T_l$ , and high threshold,  $T_h$ . The suggested ratio of these thresholds is two or three to one [37]. The final edge image is constructed by combining the images obtained by thresholding the output of the previous step with the high and low thresholds. The result of thresholding with  $T_h$  produces edges that correspond to stronger gradients in the initial image. The weaker edges can be identified by removing the strong edges from the set of edges obtained with  $T_l$ . Those weaker edges are used to “repair” disconnected strong edges by analyzing neighborhood connectivity

as described in [96].

Both the pre-processing and edge-detection steps are facilitated by the implementations of the image processing algorithms in the Insight Toolkit [109, 120].

### 5.2.3 Development of the Robust Local Hausdorff Distance Metric

Given two three-dimensional images,  $\mathcal{I}$  and  $\mathcal{J}$ , we attempt to estimate the point-wise alignment error between the homologous points of these images. Let  $\mathcal{A}$  and  $\mathcal{B}$  be the binary images, as defined earlier in Chapter 2, with the non-zero voxels corresponding to the feature (edge) points extracted from  $\mathcal{I}$  and  $\mathcal{J}$  respectively. Binary images  $\mathcal{A}$  and  $\mathcal{B}$  are the result of pre-processing and edge detection operations described in the previous Sections. Let  $A = \{a_1, \dots, a_n\}$  and  $B = \{b_1, \dots, b_m\}$  of  $\mathbb{R}^3$  be the sets of points that correspond to the non-zero voxels in  $\mathcal{A}$  and  $\mathcal{B}$ , respectively.

The directed Hausdorff Distance (HD) between the two sets of points  $h(A, B)$  is defined as the maximum distance from any of the points in the first set to the second set. The symmetric HD between the two sets, denoted  $H(A, B)$ , is the maximum of the directed HD values for the two sets [108]:

$$h(A, B) = \max_{a \in A} (d(a, B)), \text{ where } d(a, B) = \min_{b \in B} \|a - b\|,$$

$$H(A, B) = \max(h(A, B), h(B, A)).$$

As suggested in [21], we use the same notation for images, considering  $A$  and  $B$  the sets of non-zero pixels of  $\mathcal{A}$  and  $\mathcal{B}$ , respectively:  $h(\mathcal{A}, \mathcal{B}) = h(A, B)$ . In the rest of this section we refer to symmetric Hausdorff Distance whenever we use HD.

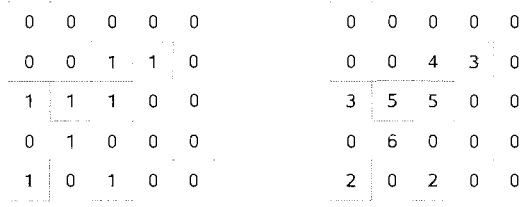
In the case when the points used for calculation of  $d(a, B)$  and  $d(b, A)$  happen to be the homologous points (i.e., when they correspond to the same image feature in both images),  $H(A, B)$  would be the maximum (global), alignment error. Even in this perfect correspondence case, the first problem of using HD for alignment assessment becomes apparent: HD can only estimate the maximum error. The second problem comes from the sensitivity of the metric to noise and lack of point correspondence: the estimated value of the error will not be the same as the maximum error, in the general case. Simple versions of the robust HD measure were proposed to alleviate this problem. The partial Hausdorff distance is defined as a quantile of the ranked distances between the two point sets, originally proposed by Huttenlocher et al. [108]. Archip et al. [8, 11] use 95%-HD, which is defined as the 0.95-quantile partial distance between the two sets. However, 95%-HD is a global measure, and does not allow to assess the error locally without modifications to the calculation procedure.

The local-distance map (LDMap) proposed by Baudrier et al. [21] for the comparison of two-dimensional images extends the definition of the HD. LDMap allows to derive the *local* measure of dissimilarity  $H_{loc}(A, B, x)$  between the two binary images  $A$  and  $B$  at the point  $x$  as follows:

$$\forall x \in \mathbf{R}^3 : H_{loc}(A, B, x) = |\mathcal{A}(x) - \mathcal{B}(x)| \times \max(d(x, A), d(x, B)). \quad (5.1)$$

$H_{loc}(A, B, x)$  is a symmetric measure, which is connected to the conventional HD definition by the relation  $H(A, B) = \max_x (H_{loc}(A, B, x))$  [21].

The advantage of the LDMap measure  $H_{loc}(A, B, x)$  is that it can be used for localized estimation of the alignment error. Ideally, the value of  $H_{loc}$  should be the same as



**Figure 5.1:** Left: Binary image. Right: Corresponding greyscale image.

the distance between the corresponding points in the images. However, because there is no point correspondence in the HD definition, the values of  $H_{loc}$  would generally deviate from the values of the alignment error.

We attempt to add the notion of point correspondence to the definition of the LDMap by using greyscale modification of the HD originally proposed by Zhao et al. [209] for matching two-dimensional images corrupted by noise. In order to achieve this, we first transform the input binary images, produced by the feature detection procedure, into non-binary (so called, greyscale) images  $\tilde{\mathcal{A}}$  and  $\tilde{\mathcal{B}}$ . These greyscale images have the same size as the initial binary images, with each voxel corresponding to the non-zero voxel in the input binary image initialized to the total number of non-zero voxels in its neighborhood. A two-dimensional example of greyscale image construction is shown in Figure 5.1. By construction,  $\tilde{\mathcal{A}} : x \in \Omega \mapsto \{0, 1, \dots, 27\}$ . Pixel value is zero, if the corresponding pixel in the binary image is zero, and the maximum number of non-zero neighbors is 26 in a three-dimensional image neighborhood.

The definition of the distance between a point a point set should be updated, to take into account the intensity value in the greyscale image. Let  $a$  be a greyscale image pixel with non-zero value, and  $\tilde{\mathcal{A}}(a)$  be that greyscale image value. The definition of distance from  $a$  to greyscale image  $\tilde{\mathcal{B}}$  is based on the use of tolerance parameter  $g$  that specifies



the maximum allowed difference in the intensity between the pixel values in  $\tilde{A}$  and  $\tilde{B}$ . It is defined as the minimum distance a pixel in  $\tilde{B}$  with such an intensity value that differs from  $\tilde{A}(a)$  by no more than  $g$ :

$$d(a, \tilde{B}) = \min_{b \in \tilde{B}} \|a - b\|, \quad \tilde{A}(a) - g \leq \tilde{B}(b) \leq \tilde{A}(a) + g.$$

Zhao et al. suggested selecting  $g = 1$  for two-dimensional images [209]. We set  $g = 2$  to account for the increase in image dimensionality. The symmetric Hausdorff Distance can now be redefined to account for greyscale image values using the modified formula for  $d(a, \tilde{B})$ :

$$\begin{aligned} h(\tilde{A}, \tilde{B}) &= \max_{a \in \tilde{A}} (d(a, \tilde{B})), \\ H(\tilde{A}, \tilde{B}) &= \max(h(\tilde{A}, \tilde{B}), h(\tilde{B}, \tilde{A})). \end{aligned}$$

The use of greyscale definition of the Hausdorff distance is based on the assumption that non-rigid deformation of an image should not change the local topological properties of the edge features. We use the number of non-zero pixels in the fixed neighborhood as the measure of topological connectivity. This is a measure that can be easily evaluated, and can be used to discard those points that are located at a close distance, but do not have similar connectivity.

Accordingly, the greyscale local HD can now be evaluated based on this updated point distance definition:

$$\forall x \in \mathbf{R}^3 : GH_{loc}(\tilde{A}, \tilde{B}, x) = |\mathcal{A}(x) - \mathcal{B}(x)| \times \max(d(x, \tilde{A}), d(x, \tilde{B})). \quad (5.2)$$

Based on the  $GH_{loc}$ , we define the corresponding global measure, greyscale Hausdorff distance (GHD)  $GH(A, B)$  between the two binary images, as  $GH(A, B) = \max_x (GH_{loc}(\tilde{A}, \tilde{B}, x))$ .

The calculation of  $GH_{loc}$  results in a greyscale image, where every non-zero pixel corresponds to the value of local greyscale Hausdorff distance. As the final step, we apply local smoothing operation, to reduce the impact of noise on the error estimation. We define robust greyscale HD locally based on the least trimmed squares robust statistics [169] on the values of  $GH_{loc}$  in the region around each feature point. The robust greyscale HD  $RGH_{loc}(\tilde{A}, \tilde{B}, x)$  is calculated as the average of the ordered values of  $GH_{loc}(\tilde{A}, \tilde{B}, x)$  in the fixed size window centered at  $x$ , after discarding 20% percent of the top distance values within this window (trimmed mean value). Similarly to the HD and GHD, we define the robust greyscale Hausdorff distance (RGHD)  $RGH(A, B) = \max_x (RGH_{loc}(\tilde{A}, \tilde{B}, x))$ .

### 5.3 Evaluation

In this Section we perform an evaluation of the presented error assessment method. The objective of this evaluation is to establish accuracy of the estimated error measures. We approach this evaluation by utilizing different sources of ground truth, as we discussed previously. As the first source of ground truth we consider known synthetic deformation field applied to a real image. In the second experiment, the ground truth is represented by the deformation field obtained by realistic simulation. Finally, we use the observer-recovered ground truth from the real images obtained during neurosurgery.

In all three evaluation approaches, the performance of an evaluation metric is measured as its ability to recover the deformation magnitude (thus, misalignment error value) at the locations of the detected edge features in the input images. The recovered error estimate can then be compared to the ground truth deformation at this point. In the first and second evaluation scenario, the ground truth deformation is available at

every point of the image. In the third evaluation study that is based on observer ground truth, we know the estimate of the deformation only at few points identified by the observer. Therefore, in the cases when these point do not coincide with the edge points recovered by the edge detector, ground truth comparison cannot be accomplished.

The questions we attempt to answer in this evaluation are (1) how the values of  $H_{loc}$ ,  $GH_{loc}$  and  $RGH_{loc}$  locally compare to the ground truth alignment error, and (2) how the robust versions of the HD (GHD and RGHD) compare with the conventional HD.

### 5.3.1 Synthetic Ground Truth

We first evaluate the effectiveness of the proposed accuracy assessment methodology for non-rigid registration of brain MRI using the synthetic ground truth data. We use the method described by Rogelj et al. [167] to construct a synthetic deformation field. The synthetic deformation field is applied to the original grayscale image, followed by feature detection step performed on both the original and deformed images. The proposed error recovery methodology is then employed to estimate the misalignment between the original and the deformed images at the selected feature points. The true misalignment value, which is the magnitude of the synthetic deformation at a point, ideally should be equal to the error value recovered by the either of the proposed assessment measures.

The procedure used for generating the synthetic deformation field is the same as used for evaluating mesh generation methods in Chapter 3 and in [167, 27]. We assign a random deformation vector at each of the grid control points, with the components of the vector drawn from a Gaussian distribution parametrized by mean  $\mu = 0$  and variance  $\sigma$ . The variance parameter can be used to control the deformation field, with larger local

deformation values obtained with the larger values of variance. Deformation vectors at the control points are initialized in such a way that the magnitude of the deformation does not exceed 10% of the brain dimensions, and there are no discontinuities in the deformation field (no intersections of the deformation vectors at the control points). The dense deformation field is constructed by using Thin Plate Splines interpolation from the deformation vectors at the control points that fall within the binary mask.

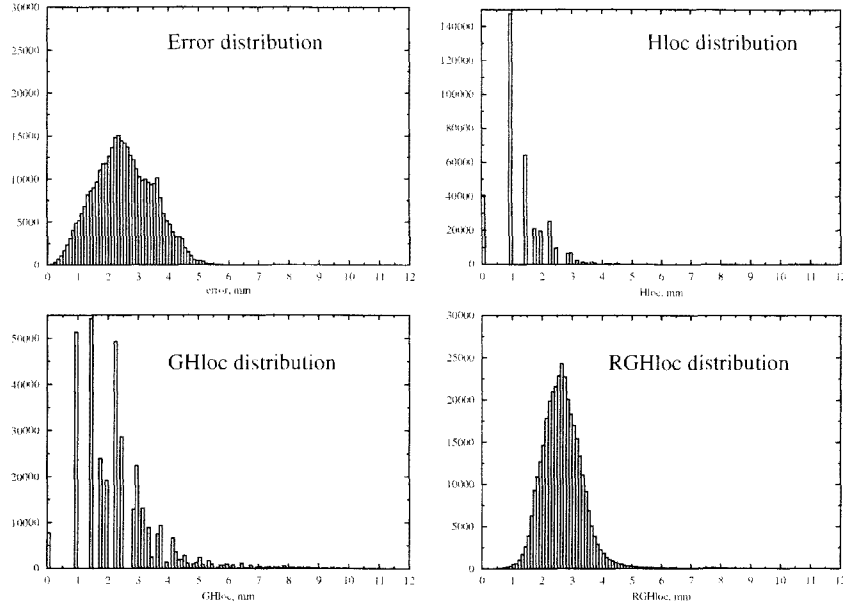
The synthetic deformation is next applied to the MR images of the brain. In this study we use brain MRI available from the BrainWeb MRI simulator [57]. BrainWeb simulator was used to create two normal subject T1 images with the following parameters of the simulator: 1 mm slice thickness, 0% intensity non-uniformity. In order to evaluate the impact of noise in the input data, we perform evaluation on two image sets with different noise levels. BrainWeb simulator was used to generate images with no noise, and images corrupted with the 9% Gaussian noise.

We compare the local estimation methods with ground truth error using two measures: distribution of error and percentage of outliers. Ideally, the distribution of local error estimates ( $H_{loc}$ ,  $GH_{loc}$  and  $RGH_{loc}$ ) will closely mimic the true error distribution. Also, a good estimate of error should have minimum number of outliers. Let  $d_i$  be the distance at voxel location  $i$ , as measured by a local estimation method. Let  $e_i$  be the true error at the same voxel, defined as the magnitude of the ground truth deformation vector defined at this voxel. We further refer to the value  $e_i$  as the alignment error. Any voxel  $i$  where  $|d_i - e_i| > 2mm$  is defined as an outlier. The meaning of outlier measurement is that the absolute difference between the distance from the feature located at this particular voxel exceeds the true distance between the corresponding points by

more than 2 mm. We choose 2 mm because the deformation field is in physical space and the HD distance implementation is limited to 1 mm spacing of the input images we use. Therefore, errors as large as  $\sqrt{3}$  cannot be prevented.

The distributions of the alignment error values, and  $H_{loc}$ ,  $GH_{loc}$ , and  $RGH_{loc}$  are shown in Figure 5.2. We observe that the true distribution of error is approximated closer by the robust HD measures. With the simple definition of  $HD_{loc}$ , topological properties of the edge features are not considered, which causes underestimation of the error value. As can be seen from the figure, the problem of underestimation is still relevant for the RGHD measure, since the distribution shows lower variance. This is expected, because RGHD results in local smoothing of the distance estimates. Although the RGHD provides closer approximation of the original distribution, it comes at a cost that maximum errors will be discarded because of the use of least trimmed squares robust statistics [169].

We compare relative robustness of the evaluated accuracy measures, and their susceptibility to noise by evaluating the percentage of outlier estimations with the increasing variance parameter in the synthetic deformation field. The results are plotted in Figure 5.3, showing the comparison between the different global and local error estimation measures. We observe that conventional global HD metric overestimates the maximum error in most cases, and always overestimates the mean error. This metric is highly susceptible to image noise: note the difference between the error reported for 0% and 9% noise levels with the global HD measure. Comparing the 95% HD and the RGHD measure we developed, we observe that both of these metrics are consistently growing as the average/maximum errors increase. However, neither of these measures cannot be

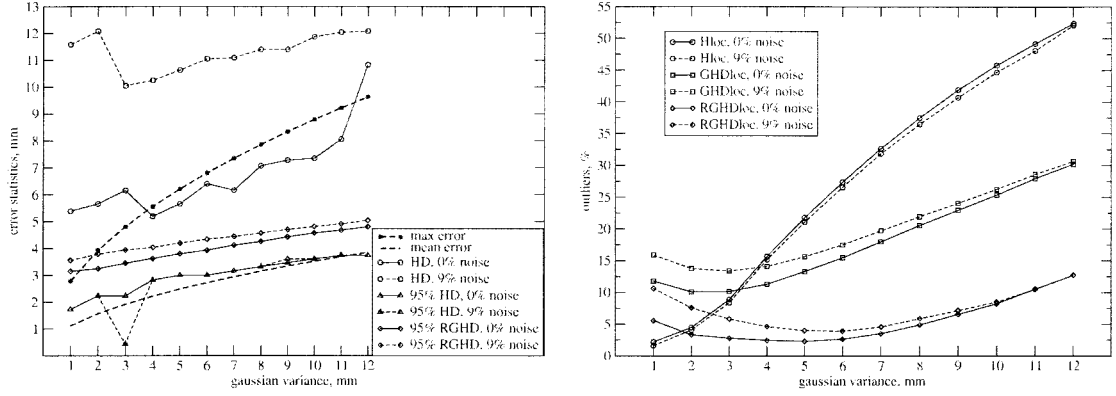


**Figure 5.2:** Distribution of the alignment error and the  $HD_{loc}$ ,  $GHD_{loc}$ , and  $RGHD_{loc}$  values for the same synthetic deformation case (three-dimensional BrainWeb image data, deformation as a sum of Gaussians,  $\mu = 0$ ,  $\sigma = 5$ ).

used as the estimate of the maximum registration error, since the reported metrics by definition exclude maximum local estimates.

Comparison of the percentage of outliers in local error measure estimations is shown in Figure 5.3. First, we observe that the number of outliers is increasing as we increase the variability and deformation magnitude in the synthetic deformation field by changing the Gaussian variance. Although the percentage of outliers is growing for both robust and non-robust metrics, it is consistently lower for the metrics we developed. At the same time we observe that the robust metrics are more sensitive to image noise, and contain more outliers for low values of the deformation.

Overall, the robust metrics contain less outliers, which is reflected in the more stable behavior of GHD and RGHD in comparison to the HD: RGHD is consistently increasing as the alignment error increases, and it is always above the mean error value (see



**Figure 5.3:** Comparison of the error estimation accuracy for the analyzed robust Hausdorff Distance based metrics. Left: Error statistics for synthetically deformed BrainWeb images with and without noise, and the derived values of the Hausdorff distance based estimations. Right: The change in the proportion of outliers as a function of Gaussian variance.

Figure 5.3, left). Thus, for large deformations (deformations as large as 10-15 mm have been reported during open skull craniotomy) RGHD is a more appropriate measure.

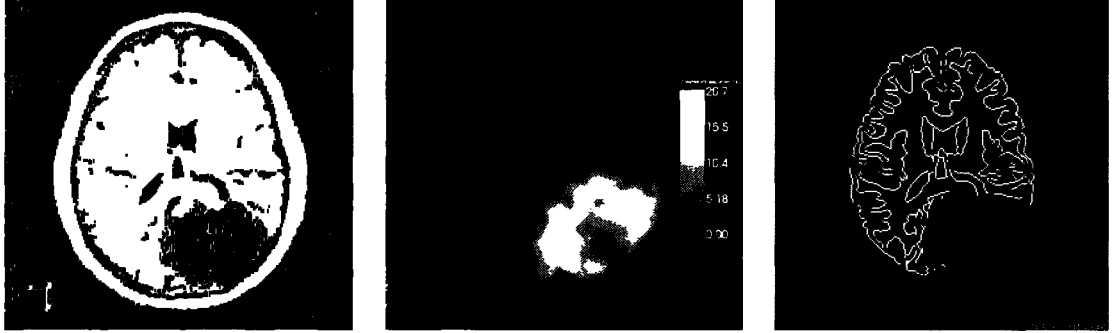
### 5.3.2 Physically-Realistic Simulated Ground Truth

We used simulated brain tumor growth images to assess error estimation performance for more realistic deformation modes, and for the images of different contrast. The simulated images were created from the BrainWeb anatomical data as described by Prastawa et al. [162]. We used two versions of the simulated data: (1) with the intensity distribution close to that of the healthy subject image, and (2) with the intensity distribution derived from the real tumor data. Edge detection was done on the images with the regions corresponding to the tumor excluded. The misalignment was estimated between the healthy subject data and the image with the simulated tumor for the same subject at each feature point of the edge images. The recovered distances were compared with the true deformation magnitude from the tumor growth simulation (deformation field being the sum of the tumor mass effect and infiltration-induced deformations).

**Table 5.1:** Outliers percentage for synthetic tumor growth data (only points corresponding to non-zero ground truth deformation are considered).

id	same contrast		diff. contrast		diff. contrast, enhanced	
	$H_{loc}$	$RGH_{loc}$	$H_{loc}$	$RGH_{loc}$	$H_{loc}$	$RGH_{loc}$
1	7.9%	4.6%	32.7%	42.9%	12.1%	9.5%
2	21.7%	16.4%	30.3%	34.1%	20.2%	15.4%
3	4.1%	3.6%	34.5%	44.7%	10.4%	7.1%

The outlier statistics is summarized in Table 5.1. Case 2 was the most complex, with the two infiltrating tumors of large volume located one next to another. Edge detection is particularly problematic in the edema region, which in this particular case extends over most of the deforming tissue region. This explains large number of outliers for set 2. Figure 5.4 helps to appreciate the complexity of error recovery for tumor set 2: there are very few edges detected in the area of the deformation, and the tumor area is almost indistinguishable from the large edema region. Nevertheless, robust HD estimation consistently has less outliers than the HD.



**Figure 5.4:** Synthetic tumor deformation and recovered alignment error estimates. Left: Synthetic tumor, case 2. Center: Deformation field produced by the tumor growth simulation (tumor mass effect and infiltration), colored by magnitude. Right: Edges recovered from the simulated tumor image. The same slice is shown in all images.



### 5.3.3 Observer-Derived Ground Truth

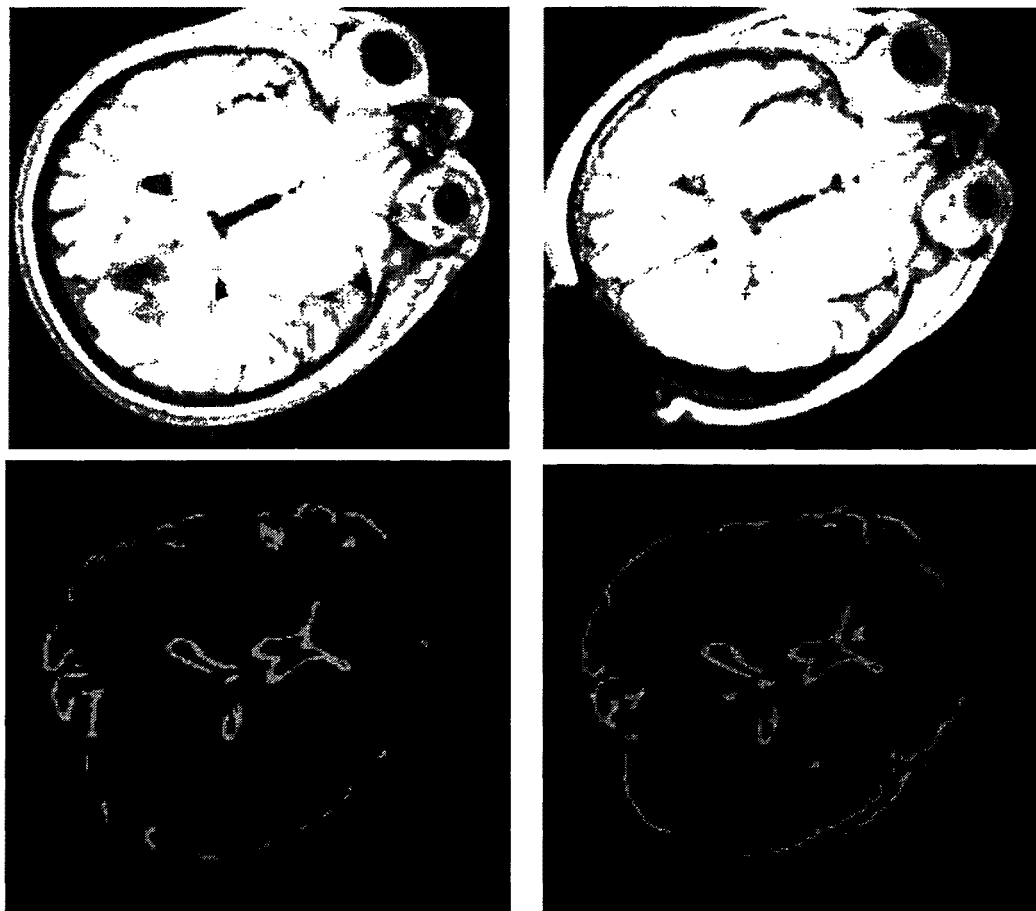
We used three data sets from the public SPL repository of the tumor resection cases<sup>2</sup>. A radiologist was asked to locate 10 corresponding anatomical landmarks in the pre- and intra-operative brain MRI T1 images. Some of the landmarks identified for the first imaging case together with the edge features used in the automated analysis are shown in Figure 5.5. We used the total of three sets of intra-operative data, and only one observer. The effort required for identification of such landmark points is significant, and we were not able to recruit more observers to cross-validate correctness of landmark locations.

The error recovered using the HD-based techniques was compared with the expert-estimated error. The results are summarized in Table 5.2. On average, the RGHD measure shows better accuracy compared to the HD. At the same time, these results identify one of the problems associated with the automated error estimation. The automatically estimated measures rely on the consistent identification of the anatomical features in the analyzed images. Therefore, the error cannot be estimated in the regions of the image where such features were not located. We observe that this was the case for a number of anatomical landmarks, when there were no edge points in the landmark vicinity.

We note that accuracy improvement of the robust error estimation measures in comparison with the conventional HD-based estimates is in the sub-voxel range. Also, both of the metrics disagree with the observer-estimated displacement magnitude in the order of 1-2 mm. In relation to these observations, one of the very important issues that has

---

<sup>2</sup>[http://www.spl.harvard.edu/pages/Special:PubDB\\_View?dspaceid=541](http://www.spl.harvard.edu/pages/Special:PubDB_View?dspaceid=541)



**Figure 5.5:** Expert-estimated locations of the used landmarks for a pair of IGNS images .Top: Pre- and intra-operative images with some of the anatomical landmarks selected. Bottom: Edge features extracted from the pre- and intra-operative images.

not been studied is the reliability of the observer-estimated ground truth data. As we discussed previously, the derivation of ground truth in validating non-rigid registration on clinical data is inherently unsolvable problem. We do not suggest that the developed techniques can serve as a substitute for validation of the method. Rather, we propose an alternative method to estimate the alignment error between the non-rigidly deformed images in the absence of ground truth.

**Table 5.2:** Accuracy of error assessment for the tumor resection data in mm; empty entries correspond to image locations without edge features to assess error.

land-mark	case 1			case 2			case 3		
	expert	$H_{loc}$	$RGH_{loc}$	expert	$H_{loc}$	$RGH_{loc}$	expert	$H_{loc}$	$RGH_{loc}$
1	2.59	—	—	2.51	—	—	4.43	1	3.07
2	0.48	—	—	1.52	—	—	4.53	1.41	2.25
3	0.48	1	0.92	2.99	1.41	1.53	3.96	—	—
4	0.48	1	1.2	1.36	1.73	1.16	2.15	—	—
5	2.59	1	0.82	0.98	—	—	2.88	2	2
6	1.07	—	—	2.4	—	—	3.66	—	—
7	2.45	—	—	2.04	—	—	3.49	2.24	2.22
8	1.44	1	0.63	1.92	1	1.36	4.43	1.41	2.94
9	3.36	2.24	3.45	3.04	—	—	3.96	—	—
10	1.44	1	1.11	1.36	1	1.43	1.98	1.41	1.05
avg difference w.r.t. expert		0.77	0.69		0.81	0.57		2.04	1.37



**Figure 5.6:** Local estimation of misalignment using RGHD, all are images show the same slice. Left: Undeformed image, BrainWeb. Center: Deformed image, Gaussian kernel variance 5 mm. Right: LDMap of the deformed and undeformed images, voxel values initialized to  $RGH_{loc}$ .

## 5.4 Summary

We have presented an HD-based approach to estimation of image alignment error. Based on the evaluation results, the RGHD measure we propose can be more robust compared to the conventional Hausdorff Distance measure in terms of outliers in local distance estimation, and thus can potentially improve the accuracy of the image alignment assessment. While our primary application is the assessment of the non-rigid registration

results, validation of the proposed method itself on real neurosurgery data is complicated by the absence of ground truth. However, based on the evaluation, it can be used to improve the confidence in registration results.

The synthetic tumor growth data used in our evaluation may be more challenging than estimation of the pre-, intra-operative and registered image alignment. In the latter case, the images have similar content: tumor and edema are present on both images, and the edges detected from those images are more similar. We show that RGHD improves error estimation accuracy locally for anatomical landmarks, thus we expect that globally RGHD is also more accurate on neurosurgery data than the HD measure.

The evaluated techniques, and specifically RGHD – the most robust of the evaluated methods – can serve multiple purposes in registration assessment. First, they can be used as a global similarity metric between the two images, as well as for local alignment assessment. This mode of operation is particularly useful for automatic assessment of the non-rigid registration results during large-scale unsupervised parametric studies. Second, localized assessment of registration error can also be applied in conjunction with the visual assessment to provide quantitative error measurements. An example is shown in Figure 5.6. We emphasize, that the proposed method cannot substitute validation studies. Instead, it can be used in conjunction with other accuracy assessment methods for the patient studies, where accuracy is critical, processing time is highly limited and there are no means to compare the registration result with the ground truth. A promising area of our future work is the evaluation of the proposed measures in conjunction with the consistency tests of the deformation fields obtained during the NRR, and sensitivity of the measures to parameter selection of a specific NRR method.

## 5.5 Contributions

Following are the contributions of the research presented in this Chapter:

- A novel robust method for alignment error estimation based on local Hausdorff distance definition was proposed;
- An open-source implementation of the method has been developed;
- The developed method has been evaluated using different sources of ground truth data, establishing the advantages in robustness of the proposed approach over the conventional Hausdorff distance estimations.

The results presented in this Chapter previously appeared in the following publications:

- A.Fedorov, E.Billet, M.Prastawa, A.Radmanesh, G.Gerig, R.Kikinis, S.K.Warfield, N.Chrisochoides. Evaluation of Brain MRI Alignment with the Robust Hausdorff Distance Measures. In Proc. of 4th International Symposium on Visual Computing (ISVC'08), 2008, pp.594-603
- E.Billet, A.Fedorov, N.Chrisochoides. The Use of Robust Local Hausdorff Distances in Accuracy Assessment for Image Alignment of Brain MRI. ISC Insight Journal, 2008

## Chapter 6

# Discussion and Future Work

In this dissertation we discussed three aspects related to non-rigid registration for image-guided neurosurgery. These aspects belong to quite different domains: high performance and distributed computing, tetrahedral mesh generation, and image analysis for automated accuracy assessment of non-rigid registration. As a result, we developed a number of methods and tools that attempt to make NRR computation feasible, more accurate and more reliable for the purposes of intra-operative image guidance.

Non-rigid registration is an inherently ill-posed problem in medical image analysis, that requires customized solutions depending on the clinical application, imaging modality and anatomical region. While developing robust non-rigid registration algorithms is a complex task, the complete solution of the problem includes deployment of the method in the clinical setting. As we showed in this dissertation, the various aspects related to such deployment of the method requires understanding of a number of related issues.

We believe that this work presents tangible results that will facilitate development and application of the broad range of image registration and processing algorithms in the

clinical setting. Our study of the mesh generation problem in the context of the NRR resulted in the development and analysis of the three reproducible approaches to this problem. Two of these approaches include off-the-shelf tools used in traditional mesh generation. In this work, we show how these tools can be combined to mesh binary images.

As we showed in this dissertation, construction of the mesh for the registration method we studied requires consideration of multiple generic and application-specific criteria. These criteria present conflicting requirements, and the question of combining these requirements to produce the final mesh requires further study. In particular, an interesting question could be what is the optimal distribution of the registration points that can be achieved, and how to improve the mesh sizing to consider the magnitude of mesh deformation.

The most important issue that has been addressed in this work to the very limited extent is the validation of the developed techniques for image meshing and accuracy assessment with the clinically relevant data. This problem is inherent to validation of the non-rigid registration methods. Validation of the presented results with the anatomical landmark data obtained using multiple expert observers is necessary to establish the clinically significant impact of the meshing methods on registration accuracy.

The presented methods of image-to-mesh conversion consider only the case of single-material, homogeneous meshes. Constructing multi-material tetrahedral meshes using the conventional mesh generation tools by surface meshing followed by volume meshing is more challenging. The surface recovery method we evaluated as implemented in CGAL is not capable of meshing multiple surfaces at the same time. However, this is

not a limitation inherent to the surface recovery algorithm, which is indeed capable of handling recovery of multiple surfaces [31].

We developed a distributed implementation of the clinically evaluated NRR method. Recognizing the difficulties of parameter selection for this method, we proposed to address this problem using speculative execution. Speculative computations require significant processing resources. We developed an implementation that is capable of leveraging the computing power of the national TeraGrid cyberinfrastructure to enable this computation.

The feasibility of using geographically distributed Grid resources for the purposes of non-rigid registration has been evaluated. There are some important practical issues, which still complicate use of the developed framework on the TeraGrid during neurosurgery. The first difficulty is due to the presence of local job batch scheduler does not provide guarantees on the start time of the job execution. We discussed a number of approaches that can help to alleviate this problem. First, there are wait time prediction services, like QBETS [145], available as services on TeraGrid. These services provide potential to improve scheduling decisions while choosing the execution site. Second, we used direct submission of jobs to the batch scheduler, which is not the most optimal for throughput-oriented execution. Recently introduced alternative scheduling mechanisms allow to use single job submission request to subsequently execute multiple application instances. This significantly reduces time due to minimized interaction with the scheduler and reduced queue times.

The second difficulty related to the use of TeraGrid is that the interaction with the remote GRAM using the standard Globus Toolkit installation requires open ports in



the client firewall configuration to receive updates about the job status. Based on our experience, configuration of the hospital intranet often includes multiple levels of network security, which may make firewall openings not possible. The issue of firewalls has been recognized by some groups as one of the most difficult practical issues in using grid resources. In order to address this problem, the client workstation may require special configuration, or may need to be located outside the firewall. Alternative solutions proposed in the literature describe customized implementation of communication with the Grid resources, that does not require firewall openings [99, 170]. Study of these solutions within the Globus Toolkit implementation presents a subject of interesting future work.

We contributed an automated approach for estimating alignment error between the MRI images of the same subject brain. The novelty of this method is in its ability to provide localized error estimates, and in reporting the error measure as the Euclidean distance at an image feature location. Such error reporting compares the method we developed favorably with the accuracy estimates based on similarity measure calculation, or based on global Hausdorff Distance. The clinical relevance of the proposed technique was evaluated on the set of three sets of MRI data collected during neurosurgery, with the anatomical landmarks identified by one observer. As it is the case for the mesh generation component of this dissertation, the study of this new accuracy assessment measure would greatly benefit from the evaluation on a larger set of clinical data.

## Appendix A

# Software Tools for Image-to-Mesh Conversion

In this Appendix we provide brief usage instructions for the image-to-mesh conversion tools developed in Chapter 3. All of these tools are written in C/C++. We refer to the software toolkit developed in this dissertation as ImageMeshingToolkit (I2MTK). We refer to the root directory of the source code distribution as `I2MTK`. The described tools are located in `I2MTK/Tools` directory.

**Table A.1:** Software dependencies of I2MTK tools.

name	purpose
CMake [118]	platform-independent configuration
ITK [120]	image processing and image IO
VTK [121]	mesh data structures for mesh IO support
Mesquite [171]	mesh optimization
CGAL [42]	implicit surface recovery and spatial search data structures support
Tetgen [180]	Delaunay mesh generation and refinement

The developed software depends on a number of tools and libraries, as summarized in Table A.1. Prior to compilation of the Mesquite library, the contents of `I2MTK/Mesquite`

must be copied to the root directory of the Mesquite source code. These files contain modifications required for programmatic access to the Mesquite data structures. Upon installation of the prerequisite libraries, your I2MTK distribution must be updated to include the correct path to your Mesquite distribution following the instructions in `I2MTK/CMakeLists.txt`, configured and compiled.

## A.1 Triangulated Surface Recovery

`I2MTK/Tools/surface-recovery.cxx`.

This tool wraps the surface recovery functionality provided by CGAL, implements direct surface recovery from binary image, and saves the surface mesh in VTK POLY-DATA format [119].

**Table A.2:** Command line flags for triangulated surface recovery from binary image input.

flag	explanation	suggested value
<code>-input <i>name</i></code>	name of the input binary image in any image format supported by ITK IO	n/a
<code>-output <i>name</i></code>	name of the output mesh saved in any format supported by VTK IO	n/a
<code>-surfaceRadiusBound <i>value</i></code>	surface recovery parameter that regulates the maximum radius of the circumsphere of a triangulated facet centered on the facet (the detailed explanation is available in CGAL documentation)	5..10
<code>-surfaceDistanceBound <i>value</i></code>	surface recovery parameter that specifies the bound on the distance between the facet and the center of its circumsphere)	5..10

The command line options are summarized in Table A.2.

## A.2 Delaunay Mesh Generation and Refinement from Binary Image

I2MTK/Tools/image-mesher-adaptive.cxx.

This tool combines the functionality of CGAL surface recovery with the Tetgen algorithms for Delaunay meshing and refinement.

**Table A.3:** Command line flags for adaptive Delaunay meshing from binary image input.

flag	explanation	suggested value
-input <i>name</i>	name of the input binary image in any image format supported by ITK IO	n/a
-inputPointList <i>name</i>	name of the input VTK unstructured grid file [119] with the list of registration points	n/a
-output <i>name</i>	name of the output mesh saved in any format supported by VTK IO	n/a
-surfaceRadiusBound <i>value</i>	surface recovery parameter that regulates the maximum radius of the circumsphere of a triangulated facet centered on the facet (the detailed explanation is available in CGAL documentation)	5..10
-surfaceDistanceBound <i>value</i>	surface recovery parameter that specifies the bound on the distance between the facet and the center of its circumsphere)	5..10
-alpha1 <i>value</i>	volume mesh generation parameter (the detailed explanation is available in Tetgen [180] documentation)	0.4..0.8
-alpha2 <i>value</i>	volume mesh generation parameter (the detailed explanation is available in Tetgen [180] documentation)	0.6..0.9
-nnN <i>value</i>	desired number of the registration points per mesh vertex cell complex	n/a

The command line options are summarized in Table A.3.

### A.3 Lattice-Based Image Meshing

I2MTK/Tools/biofemir-mesher.cxx

This tool implements the functionality of RGM mesher, which is based on BCC lattice refinement, followed by iterative surface compression to the binary label surface. This tool operates directly on the binary image data, as described in Chapter 3. Mesquite library [171] is used by this tool for mesh quality improvement following boundary compression.

**Table A.4:** Command line flags for adaptive lattice-based meshing from binary image input.

flag	explanation	suggested value
-input <i>name</i>	name of the input binary image in any image format supported by ITK IO	n/a
-inputPointImage <i>name</i>	name of the input image with the registration points in any format supported by ITK IO	n/a
-output <i>name</i>	name of the output mesh saved in any format supported by VTK IO	n/a
-nnN <i>value</i>	desired number of the registration points per mesh vertex cell complex	n/a
-resolutions <i>value</i>	number of red-green refinement iterations	2
-bccSpacing <i>value</i>	value of BCC spacing for the initial lattice as a relative to the corresponding image dimension, i.e., numeric value of the spacing is calculated as $imageDimension/bccSpacing$	n/a
-bccSpacingValue <i>value</i>	numeric value of BCC spacing	n/a
-maxIterations <i>value</i>	number of surface compression iterations	n/a
-minAngleAllowed <i>value</i>	minimum value of the dihedral angle allowed during surface compression; if the mesh contains angle smaller than this value, boundary compression procedure is terminated	n/a

The command line options are summarized in Table A.4. Most of these options

should be set depending on the desired properties of the mesh. Specifically, the values of BCC spacing allow to control the size of the mesh. Bounds on the number of surface compression iterations and minimum dihedral angle provide flexibility in terms of the trade-off between the surface approximation accuracy and quality of the mesh elements.

## A.4 Background Mesh Construction

`I2MTK/Tools/biofemir-bg-mesher-nn.cxx`

This tool constructs the background mesh based on refinement of BCC lattice that completely encloses the input binary image. The values of the sizing function are initialized as described in Section 3.5. The resulting tetrahedral background mesh can be saved in any file format supported by VTK IO, and also in the format required for background sizing field description by Tetgen and NETGEN.

The command line options are summarized in Table A.5.

## A.5 Utilities

`I2MTK/Tools/mesher-reader.cxx` performs conversion between the tetrahedral meshes saved in different formats. Supported formats include those available in VTK IO, and the formats produced and supported by Tetgen and NETGEN.

`I2MTK/Tools/surface-reader.cxx` provides conversion between the triangular surface meshes in different formats.

`I2MTK/Tools/meshqual-driver.cxx` reports statistics on mesh quality. The quality metrics for each tetrahedron can also be saved in a VTK file. This class wraps the quality

**Table A.5:** Command line flags for adaptive lattice-based meshing from binary image input.

flag	explanation	suggested value
-input <i>name</i>	name of the input binary image in any image format supported by ITK IO	n/a
-inputPointImage <i>name</i>	name of the input image with the registration points in any format supported by ITK IO	n/a
-output <i>name</i>	name of the output mesh saved in any format supported by VTK IO	n/a
-nnN <i>value</i>	desired number of the registration points per mesh vertex cell complex	n/a
-resolutions <i>value</i>	number of red-green refinement iterations	2
-bccSpacing <i>value</i>	value of BCC spacing for the initial lattice as a relative to the corresponding image dimension, i.e., numeric value of the spacing is calculated as $imageDimension/bccSpacing$	n/a
-bccSpacingValue <i>value</i>	numeric value of BCC spacing	n/a
-outputPrefix <i>name</i>	if specified, background mesh will be saved in the files named <i>name.msz</i> and <i>name.b.node</i> for NETGEN and Tetgen respectively.	n/a

assessment functionality implemented in *vtkMeshQuality* class provided by VTK [121].

`I2MTK/Tools/meshAddPtCnt.cxx` takes the tetrahedral mesh and the list of points, and produces a mesh with initialized value of point count in the mesh vertex cell complex for each mesh vertex.

## Appendix B

# Execution of Non-Rigid Registration on TeraGrid

In this Appendix we describe the setup and infrastructure for executing the non-rigid registration workflow over the TeraGrid [188] resources. Execution of the registration workflow assumes that the registration process is run as a sequence of processing steps available separate modules, which correspond to the decoupled implementation of NRR described in Chapter 4. The modules used in the workflow are the following:

- *smooth* wraps the anisotropic diffusion filtering;
- *detectEdges* wraps the Canny edge detection filter, and produces a binary output with the edges extracted from greyscale image;
- *pointSelection* prepares the list of registration points identified in the floating (pre-operative) image;
- *blockMatchingMPI* is a parallel implementation of block matching based on MPI;



- *solver* is the implementation of the iterative outlier rejection procedure;
- *ResampleRunAssessment* applies the procedure of error estimation on the result of non-rigid registration.

The *pointSelection*, *blockMatchingMPI* and *solver* are the components of the restructured implementation of the original NRR code used in [53].

The setup of the Grid-based NRR infrastructure consists of the preparations that must be done on the submitting site (client side) and on the site where the computation will be taking place (server side).

The processing modules depend on the ITK and VTK libraries, which are the same as mentioned in Appendix A. These libraries must be available on the server side for workflow component execution. Additionally, the block matching module depends on the availability of an MPI implementation on the server side. No restriction are in place on the specific implementation of MPI that should be used, which reinforces portability of the implementation. We mention OpenMPI [190] and MPICH [13] as the two popular MPI libraries. All the modules that will be executed remotely must be compiled and available as executables on each of the server sites.

On the client side, Globus Toolkit (GT) compatible with the version that supports TeraGrid must be installed (we used GT version 4.0.7). Swift package [58] must be available on the client side for workflow description, execution and management. Swift location is referred to by the environment variable `SWIFT_HOME`. As part of the client side setup, a range of TCP ports open for incoming connections must be available, following the GT setup guidelines [95].

Execution of jobs on TeraGrid requires that the submitting user has active accounts and resource allocations on the TeraGrid sites that will be used for job submissions.

Once the workflow modules are installed on the server sites, the client side must be configured to describe the computational resources that will be used for workflow execution, and the details on the module location and execution by means of Site catalog (grid resource description) and Translation catalog (workflow component description). As an example, the following XML listing describes the configuration for a single remote resource, as provided in the site catalog in

`SWIFT_HOME/etc/sites.xml:`

```
<config>
<pool handle="SDSC-GT4">
  <gridftp url="gsiftp://tg-gridftp.sdsc.teragrid.org:2811/" />
  <execution provider="gt4" jobmanager="PBS"
url="https://tg-login1.sdsc.teragrid.org:8443/wsrf"/>
  <workdirectory>/users/fedorov/scratch</workdirectory>
</pool>
</config>
```

The description of each resource (the “pool” element) must include the contact string for GridFTP service, the execution provider, which is a GRAM service for TeraGrid sites, and the location of scratch directory on the remote resource. This directory will be used to store intermediate data relevant for the job execution.

The details on the availability of the workflow components are provided in the translation catalog stored in `SWIFT_HOME/etc/tc.data`. The translation catalog provides all the details about the workflow component that are necessary for its remote execution: the full path to the executable, the walltime needed for its completion, and the number of nodes, if the executable is an MPI application.

We refer the reader to the supporting documentation of Swift [58] and TeraGrid [188] for further information on configuration of the translation and site catalogs, as they should be tailored to the resources and configurations available in each particular case.

Using the Swift script, the parametrized execution of the NRR workflow can be described as follows:

```
### Description of the types used in the script
type vtkfile{} // .vtk file: meshes and registration points
type imagefile{} // various image files supported by ITK readers
type volfile{} // .vol file: mesh format expected by NRR
type messagefile{} // message files: stderr and stdout

### Invocations of the workflow components
# Detect registration points
(vtkfile output) selectPoints(imagefile image, imagefile mask, volfile mesh,
    string connexity, string fracVar, string blockSize){
    app {
        pointSelection "--mesh" @mesh "--floatingImage" @image "--fracVarRejected"
            @fracVar "--blockConnexityType" @connexity "--mask" @mask
            "--blockHalfSizeX" @blockSize "--blockHalfSizeY" @blockSize
            "--blockHalfSizeZ" @blockSize "--output" @output;
    }
}

# Smooth and rescale input images for assessment processing
(imagefile out) smoothImage(imagefile input){
    app {
        smoothAndRescale @input @out;
    }
}

# Detect edges in such a way that the total number of edge
# points is about 10% of the image mask points
(imagefile output) detectEdges(imagefile input){
    app {
        EdgeDetection @input "10" @output;
    }
}

# Take the input edges and the mesh containing deformation vectors at the
# vertices, and compute the alignment score
(messagefile out) runAssessment(imagefile flo_edges, imagefile ref_edges, vtkfile mesh){
    app {
        ResampleRunAssessmentRegion "--floating-edges" @flo_edges
            "--reference-edges" @ref_edges "--metric" "rghd"
            stdout=@filename(out) "--transform-mesh" @mesh;
    }
}
```

```

# Estimate the mesh deformation based on the registration points displacements
(vtkfile meshOut) solver(volfile mesh,vtkfile bmResult, string spacing[]){
  app {
    solver "--mesh" @mesh "--pointFileName" @bmResult "--voxelSizeX"
    spacing[0] "--voxelSizeY" spacing[1] "--voxelSizeZ" spacing[2]
    "--output" @meshOut;
  }
}

# Compute block matching displacements at the registration points
(vtkfile output) blockMatching(imagefile fImage, imagefile rImage,
  imagefile mask, volfile mesh, string blockSize, string windowSize, vtkfile points){
  app {
    blockMatchingMPI "--mesh" @mesh "--floatingImage" @fImage "--output" @output
    "--mask" @mask "--blockHalfSizeX" @blockSize "--blockHalfSizeY"
    @blockSize "--blockHalfSizeZ" @blockSize "--pointFileName" @points
    "--referenceImage" @rImage "--blockHalfStepX" "1" "--blockHalfStepY" "1"
    "--blockHalfStepZ" "1" "--blockHalfWindowX" @windowSize
    "--blockHalfWindowY" @windowSize "--blockHalfWindowZ" @windowSize;
  }
}

#####

### Parameters
# - point selection
string psConnexity = "26";
string psFracVar = ".95";
string psBlockSize = "3";
# - block matching: include speculative execution for block and window sizes
string bmBlockSize[] = ["3","4"];
string bmWindowSize[] = ["7","8","9"];
# - solver: include speculative execution for the outlier percentage
string sFracRejected[] = ["0.2","0.3"]; # estimated percentage of outliers
string inputImageSpacing[] = ["0.9375","0.9375","1.5"];

# Inputs
imagefile floImage<"floating.nii.gz">; # floating image
imagefile refImage<"reference.nii.gz">; # reference image
imagefile maskImage<"mask.nii.gz">; # binary mask image
volfile mesh<"mesh.vol">; # tetrahedral mesh

### Intermediate files
imagefile floSmooth<"floating_smooth.nii.gz">; # result of smoothing
imagefile refSmooth<"reference_smooth.nii.gz">;
imagefile floEdges<"floating_edges.nii.gz">; # result of edge detection
imagefile refEdges<"reference_edges.nii.gz">;
vtkfile psFlo<"psFlo.vtk">; # list of registration points

### Outputs
vtkfile deformationMesh<"mesh_def.vtk">; # mesh with deformation vectors

### Beginning of workflow description

# filter input image

```

```

floSmooth = smoothImage(floImage);
refSmooth = smoothImage(refImage);
# detect edges
floEdges = detectEdges(floSmoothHM);
refEdges = detectEdges(refSmooth);
# identify registration points
psFlo = selectPoints(floImage, maskImage, mesh, psConnexity, psFracVar, psBlockSize);

# iterate through all possible combinations of block/window sizes and
# percentage of outliers
foreach blockSize in bmBlockSize {
  foreach windowSize in bmWindowSize {
    vtkfile bmOutput<single_file_mapper;
      file=@strcat("Results/bmOutput-",blockSize,"-",windowSize,".vtk")>;
    # perform block matching
    (bmOutput) = blockMatching(floImage, refImage, maskImage, mesh,
      blockSize, windowSize, psFlo);

    foreach fRejected in sFracRejected {
      vtkfile sOutput<single_file_mapper;file=@strcat("Results/sOutput-",
        blockSize,"-",windowSize,"-",fRejected,".vtk")>;
      # perform iterative outlier rejection
      (sOutput) = solver(mesh,bmOutput,inputImageSpacing);

      messagefile alignmentReport<single_file_mapper;file=@strcat("Results/score-",
        blockSize,"-",windowSize,"-",fRejected,".dat")>;
      # estimate registration error
      alignmentReport = runAssessment(floEdges,refEdges,sOutput);
    }
  }
}

```

## Appendix C

# Software Tools for Local Accuracy Alignment Estimation

This Appendix describes the developed tool for estimating the alignment error between the two images. The accuracy assessment tool was developed jointly with Eric Billet, and this short guide is largely based on the detailed paper describing the details of implementation and parameter settings by Billet et al. [27].

The accuracy assessment tools are written in C++, and are based on the functionality provided by the individual processing filters of Insight Toolkit [120]. The analysis functionality is implemented in the *RunAssessment* tool, which accepts on input the configuration file that specifies inputs and processing parameters. The parameters listed in the configuration file are explained in Tables C.1 and C.2. Note, that the parameters corresponding to each row in the tables should be available in the same line of the configuration file.

In the calculations of GHD and RGHD, the local values of the distance are initialized

to -100 for the edge points where the value cannot be determined for the selected analysis parameters.

In case the ground truth deformation field is available, as it is the case for when synthetic ground truth is used, the results of the accuracy assessment can be compared with the *evaluation* tool we supply. This tool accepts as input local distance map (generated by the *RunAssessment* tool using the HD, GHD or RGHD approach), the ground truth deformation field, and the edges extracted from the analyzed images. All of these are saved by the *RunAssessment* tool. Give this data, the *evaluation* tool reports statistics on the accuracy of error estimation stored in the local distance map.

**Table C.1:** Parameters for the configuration file used by the accuracy assessment tool.

parameter	explanation	suggested value
input images	the names of the files with the two images to be compared	n/a
input image masks	the names of the files with the two images corresponding to the masks for regions of interest in the analyzed images	n/a
percent of the edge voxels	an integer specifying the percentage of voxels of the mask image, which defines the desired number of edge voxels to be detected in the image	10
output edge images	the names of the two files where the edges detected from the input images should be saved	n/a
local distance map	the name of the file where the local distance map image will be stored	n/a
contrast enhancement	a flag that defines the image enhancement procedure to be used prior to edge detection. The allowed options are: <ul style="list-style-type: none"> <li>• 0: no contrast enhancement</li> <li>• 1: adaptive contrast enhancement [185]. Note, that this operation is very time-consuming and for typical brain dataset takes about an hour</li> <li>• 2: histogram matching [96]. This operation is very fast, but is not as robust as the adaptive procedure</li> </ul>	n/a
number of threads	should be adjusted to the system configuration for the efficient execution of the parallelized processing components	n/a
type of metric	should have one of the following values, that correspond to the developed metrics discussed in Chapter 5: <ul style="list-style-type: none"> <li>• HD: conventional local distance map (LDMap)</li> <li>• GHD: greyscale local distance map that includes topological edge structure</li> <li>• RGHD: greyscale local distance map with robust smoothing</li> </ul>	n/a



**Table C.2:** Parameters for the configuration file used by the accuracy assessment tool (continued).

parameter	explanation	suggested value
neighborhood radius	the radius of the image neighborhood to be used for construction of the greyscale local distance image. Note, that in three dimensions radius of 1 corresponds to the neighborhood dimensions 3x3x3	1
tolerance	this parameter corresponds to the value $t$ used in Chapter 5 for finding the corresponding point in greyscale distance image calculations	2
maximum deformation	this parameter limits the search neighborhood during the search of the corresponding point. This is an application-dependent parameter that should be set based on the expected maximum deformation.	n/a
robust statistics window	this parameter is a non-zero integer number, which should be set only if RGHD metric is used. This number defines the radius of the neighborhood over which the robust statistics metrics will be calculated.	5
robust statistics percentage	a number between 0 and 1 that is used in robust statistics calculation. The robust statistics window is centered around each of the edge voxels, and initializes the local RGHD value at that voxel based on the robust statistics calculated over the values of previously estimated GHD at the voxels inside the window.	0.8
minimum number of edge points	non-zero integer number that defines the minimum number of non-zero edge voxels within the smoothing neighborhood to proceed with the RGHD calculations. In the case when the number of edge points within the smoothing window is below the specified number.	

# Bibliography

- [1] A. ABOU-RJEILI AND G. KARYPIS. Multilevel algorithms for partitioning power-law graphs. In *IEEE International Parallel and Distributed Processing Symposium*, page 10, 2006.
- [2] W. ALLCOCK, J. BRESNAHAN, R. KETTIMUTHU, AND M. LINK. The Globus striped GridFTP framework and server. In *Proc. of Supercomputing'05*, page 54, 2005.
- [3] GLOBUS ALLIANCE. The Globus Toolkit, 2009. <http://www.globus.org/toolkit/>.
- [4] GLOBUS ALLIANCE. The WS-Resource Framework, 2009. <http://www.globus.org/wsrf/>.
- [5] R. ALTEROVITZ AND K. GOLDBERG. Comparing algorithms for soft tissue deformation: accuracy metrics and benchmarks. Technical report, Berkeley Automation Laboratory, UC Berkeley, 2003.
- [6] N. AMENTA AND M. BERN. Surface reconstruction by Voronoi filtering. In *Proc. of 14th SCG*, pages 39–48. ACM Press, 1998.
- [7] N. ARCHIP, O. CLATZ, S. WHALEN, S. P. DIMAIO, P. M. BLACK, F. A. JOLESZ, A. GOLBY, AND S. K. WARFIELD. Compensation of geometric distortion effects on intraoperative magnetic resonance imaging for enhanced visualization in image-guided neurosurgery. *Neurosurgery*, 62(3 Suppl 1), March 2008.
- [8] N. ARCHIP, O. CLATZ, S. WHALEN, D. KACHER, A. FEDOROV, A. KOT, N. CHRISOCHOIDES, F. JOLESZ, A. GOLBY, P.M. BLACK, AND S.K. WARFIELD. Non-rigid alignment of pre-operative MRI, fMRI, and DT-MRI with intraoperative MRI for enhanced visualization and navigation in image-guided neurosurgery. *NeuroImage*, 35(2):609–624, 4/1 2007.
- [9] N. ARCHIP, A. FEDOROV, B. LLOYD, N. CHRISOCHOIDES, A. GOLBY, P. M. BLACK, AND S. K. WARFIELD. Integration of patient specific modeling and advanced image processing techniques for image guided neurosurgery. In *Proceedings of SPIE 6141 Medical Imaging 2006: Visualization, Image-Guided Procedures, and Display*, K.R. Cleary and R.L. Galloway Jr., editors, volume 61411E, 2006.

- [10] N. ARCHIP, R. ROHLING, V. DESSENNE, P. J. ERARD, AND L. P. NOLTE. Anatomical structure modeling from medical images. *Comp. Meth. and Progr. in Biomed.*, 82(3):203–215, 2006.
- [11] N. ARCHIP, S. TATLI, P. MORRISON, F. JOLESZ, S. WARFIELD, AND S. SILVERMAN. Non-rigid registration of pre-procedural mr images with intra-procedural unenhanced ct images for improved targeting of tumors during liver radiofrequency ablations. In *Proc. of MICCAI'07*, pages 969–977, 2007.
- [12] ARGONNE NATIONAL LABORATORY. PETSc: Portable, extensible toolkit for scientific computation, 2005.
- [13] ARGONNE NATIONAL LABORATORY. Mpich, 2009. <http://www.mcs.anl.gov/research/projects/mpich2/>.
- [14] S. ARSENEAU. *Junction Analysis*. VDM Verlag Dr. Mueller e.K, April 2008.
- [15] N. ASPERT. M.E.S.H.: Measuring error between surfaces using the Hausdorff distance, v.1.13, 2004.
- [16] AMERICAN BRAIN TUMOR ASSOCIATION. Brain tumor facts, 2009. <http://www.abta.org/index.cfm?contentid=113>.
- [17] AMERICAN BRAIN TUMOR ASSOCIATION. Treating brain tumors: Surgery, 2009. <http://www.abta.org/siteFiles/SitePages/ABC3A809DEA9E6F5B1A2E4C935ED6188.pdf>.
- [18] M. BADER, H.-J. BUNGARTZ, A. FRANK, AND R. MUNDANI. Space tree structures for PDE software. In *Proc. of Computational Science – ICCS 2002*, pages 662–671, 2002.
- [19] L. BAGHDADI, D. STEINMAN, AND H. LADAK. Template-based finite-element mesh generation from medical images. *Computer Methods and Programs in Biomedicine*, 77(1):11–21, 2005.
- [20] T. J. BAKER. Mesh movement and metamorphosis. *Engng with Comp.*, 18(3):188–198, 10/20 2002.
- [21] E. BAUDRIER, F. NICOLIER, G. MILLON, AND S. RUAN. Binary-image comparison with local-dissimilarity classification. *Pattern Recognition*, 41(5):1461–1478, 2008.
- [22] P. BECKMAN, S. NADELLA, N. TREBON, AND I. BESCHASTNIKH. SPRUCE: A system for supporting urgent high-performance computing. In *Proc. of WoCo9: Grid-based Problem Solving Environments*, 2006.
- [23] A. BELGUELIN, J. DONGARRA, A. GEIST, R. MANCHEK, S. OTTO, AND J. WALPORE. Pvm: Experiences, current status, and future direction. In *Supercomputing '93 Proceedings*, pages 765–766, 1993.

- [24] F. BERMAN, G. FOX, AND A. J. G. HEY. *Grid Computing: Making The Global Infrastructure a Reality*. Wiley, 2003.
- [25] M. BERN, P. CHEW, D. EPPSTEIN, AND J. RUPPERT. Dihedral bounds for mesh generation in high dimensions. In *SODA '95: Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 189–196, 1995.
- [26] G. BERTI. Image-based unstructured 3d mesh generation for medical applications. In *Proc. of ECCOMAS*, 2004.
- [27] E. BILLET, A. FEDOROV, AND N. CHRISOCHOIDES. The use of robust local Hausdorff distances in accuracy assessment for image alignment of brain MRI. *ISC Insight Journal*, January-June, 2008. <http://hdl.handle.net/1926/1354>.
- [28] BIRN. Biomedical Informatics Research Network, 2008. <http://www.nbirn.net>.
- [29] B.LLOYD, G.SZEKELY, R.KIKINIS, AND S.K.WARFIELD. Comparison of salient point detection methods for 3d medical images. In *2005 MICCAI Open Source Workshop*, 2005.
- [30] R. BLUMOFE, C. JOERG, B. KUSZMAUL, C. LEISERSON, K. RANDALL, AND Y. ZHOU. Cilk: An efficient multithreaded runtime system. In *Proceedings of the 5th Symposium on Principles and Practice of Parallel Programming*, pages 55–69, 1995.
- [31] J.-D. BOISSONNAT AND S. OUDOT. Provably good sampling and meshing of surfaces. *Graphical Models*, 67(5):405–451, 2005.
- [32] S.K. BOYD AND R. MÜLLER. Smooth surface meshing for automated finite element model generation from 3d image data. *J. of Biomechanics*, 39(7):1287–1295, 2006.
- [33] M. BREWER, L. FREITAG-DIACHIN, P.M. KNUPP, T.LEURENT, AND D.J. MELANDER. The Mesquite mesh quality improvement toolkit. In *Proc. of 12th IMR*, pages 239–250, 2003.
- [34] R. BRIDSON. *Computational aspects of dynamic surfaces*. PhD thesis, Stanford University, 2003.
- [35] R. BRIDSON, J. TERAN, N. MOLINO, AND R. FEDKIW. Adaptive physics based tetrahedral mesh generation using level sets. *Engineering with Computers*, 21(1):2–18, 2005.
- [36] L.G. BROWN. A survey of image registration techniques. *ACM Comput. Surv.*, 24(4):325–376, 1992.
- [37] J. CANNY. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, 1986.

- [38] A. D. CASTELLANO-SMITH, T. HARTKENS, J. SCHNABEL, D. R. HOSE, H. LIU, W. A. HALL, C. L. TRUWIT, D. J. HAWKES, AND D. L. G. HILL. Constructing patient specific models for correcting intraoperative brain deformation. In *Proc. of MICCAI'01*, volume 2208, pages 1091–1098, 2001.
- [39] J. CEBRAL, MARCELO A. CASTRO, R. LÖHNER, JAMES E. BURGESS, RICHARD PERGOLIZZI, AND CRISTOPHER M. PUTMAN. Recent developments in patient-specific image-based modeling of hemodynamics. *Mecanica Computacional*, XXIII:1471–1482, 2004.
- [40] J. CEBRAL AND R. LÖHNER. From medical images to anatomically accurate finite element grids. *Intl J for Num Meth in Engng*, 51(8):985–1008, 2001.
- [41] J. R. CEBRAL AND R. LÖHNER. From medical images to CFD meshes. In *Proc. of 8th IMR*, pages 321–331, 1999.
- [42] CGAL. Computational Geometry Algorithms Library, 2009. Version 3.4, <http://www.cgal.org>.
- [43] S. R. CHANDANA, S. MOVVA, M. ARORA, AND T. SINGH. Primary brain tumors in adults. *American Family Physician*, 77(10):1423–1430, 2008.
- [44] S. W. CHENG, T. K. DEY, AND T. RAY. Weighted delaunay refinement for polyhedra with small angles. In *Proceedings of the 14th International Meshing Roundtable*, n/a, editor, pages 325–342, 2005.
- [45] A. CHERNIKOV AND N. CHRISOCHOIDES. Three-dimensional semi-generalized point placement method for delaunay mesh refinement. In *Proc. of 16th International Meshing Roundtable*, pages 25–44, 2007.
- [46] L. P. CHEW. Guaranteed-quality delaunay meshing in 3D. In *Proceedings of the 13th ACM Symposium on Computational Geometry*, pages 391–393, 1997.
- [47] N. CHRISOCHOIDES. Parallel mesh generation. *Numerical Solution of Partial Differential Equations on Parallel Computers*, 51:237–264, 2005.
- [48] N. CHRISOCHOIDES, A. FEDOROV, A. KOT, N. ARCHIP, O. CLATZ, R. KIKINIS, AND SIMON K. WARFIELD. Toward real-time image guided neurosurgery using distributed and grid computing. In *Proc. of the 2006 ACM/IEEE Conference on Supercomputing*, 2006.
- [49] G. E. CHRISTENSEN, X. GENG, J. G. KUHL, J. BRUSS, T. J. GRABOWSKI, I. A. PIRWANI, M. W. VANNIER, J. S. ALLEN, AND H. DAMASIO. Introduction to the non-rigid image registration evaluation project (NIREP). In *Proc. of WBIR 2006*, pages 128–135, 2006.
- [50] G. E. CHRISTENSEN AND H. J. JOHNSON. Consistent image registration. *Medical Imaging, IEEE Transactions on*, 20(7):568–582, 2001.

- [51] G.E. CHRISTENSEN, M.I. MILLER, M.W. VANNIER, AND U. GRENANDER. Individualizing neuroanatomical atlases using a massively parallel computer. *Computer*, 29(1):32–38, 1996.
- [52] O. CLATZ. About registration code, 2006. Personal communication, 26 Feb 2006.
- [53] O. CLATZ, H. DELINGETTE, I. F. TALOS, A. J. GOLBY, R. KIKINIS, F. A. JOLESZ, NICHOLAS AYACHE, AND S. K. WARFIELD. Robust non-rigid registration to capture brain shift from intra-operative MRI. *IEEE Trans. Med. Imag.*, 24(11):1417–1427, 2005.
- [54] E. B. CLAUS, A. HORLACHER, L. HSU, R. B. SCHWARTZ, D. DELLO-IACONO, F. TALOS, F. A. JOLESZ, AND P. M. BLACK. Survival rates in patients with low-grade glioma after intraoperative magnetic resonance image guidance. *Cancer*, 103(6):1227–1233, 2005.
- [55] H. E. CLINE, W. E. LORENSEN, R. KIKINIS, AND F. JOLESZ. Three-dimensional segmentation of MR images of the head using probability and connectivity. *J. of Comp. Assisted Tomography*, 14(6):1037–1045, 1990.
- [56] WILLIAM & MARY COMPUTATIONAL SCIENCE CLUSTER. SciClone Cluster Project, 2009. <http://compsci.wm.edu/SciClone>.
- [57] D. L. COLLINS, A. P. ZIJDENBOS, V. KOLLOKIAN, J. G. SLED, N. J. KABANI, C. J. HOLMES, AND A. C. EVANS. Design and construction of a realistic digital brain phantom. *Medical Imaging, IEEE Transactions on*, 17(3):463–468, 1998.
- [58] UNIVERSITY OF CHICAGO COMPUTATION INSTITUTE. Swift, 2009. <http://www.ci.uchicago.edu/swift/>.
- [59] W. R. CRUM, O. CAMARA, AND D. L. G. HILL. Generalized overlap measures for evaluation and validation in medical image analysis. *IEEE Transactions on Medical Imaging*, 25(11):1451–1461, 2006.
- [60] W. R. CRUM, L. D. GRIFFIN, D. L. G. HILL, AND D. J. HAWKES. Zen and the art of medical image registration: correspondence, homology, and quality. *NeuroImage*, 20(3):1425–1437, 2003.
- [61] O. CUISENAIRE AND B. MACQ. Fast and exact signed euclidean distance transformation with linear complexity. In *IEEE International Conference on Acoustics Speech Signal Processing*, pages 3293–3296, 1999.
- [62] P. W. DE BRUIN, F. M. VOS, F. H. POST, SARAH F. FRISKEN-GIBSON, AND A. M. VOSSEPOEL. Improving triangle mesh quality with SurfaceNets. In *Proc. of MICCAI 2000*, Scott L. Delp, Anthony M. DiGioia, and Branislav Jaramaz, editors, pages 804–813, 2000.
- [63] H. DELINGETTE AND N. AYACHE. *Soft tissue modeling for surgery simulation*, volume XII of *Handbook of Numerical Analysis: Special volume: Computational models for the human body*, pages 453–550. Elsevier, Netherlands, 1 edition, 2004.

- [64] C. DELORENZO. *Image-Guided Intraoperative Brain Deformation Recovery*. PhD thesis, Yale University, 2007.
- [65] C. DELORENZO, X. PAPADEMETRIS, K. P. VIVES, D. D. SPENCER, AND J. S. DUNCAN. A realistic brain phantom for 3d deformation recovery. In *Proc. of ISBI*, pages 9–12, 2007.
- [66] T. K. DEY AND J. A. LEVINE. Delaunay meshing of isosurfaces. In *Proc. of IEEE Int. Conf. on Shape Modeling and Applications (SMI '07)*, pages 241–250, 2007.
- [67] T. K. DEY, G. LI, AND T. RAY. Polygonal surface remeshing with Delaunay refinement. In *Proc. of 14th IMR*, pages 343–361, 2005.
- [68] L. R. DICE. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.
- [69] S. DONG, J. INSLEY, N. T. KARONIS, M. E. PAPKA, J. BINNS, AND G. KARNIADAKIS. Simulating and visualizing the human arterial system on the TeraGrid. *Future Generation Computer Systems*, 22(8):1011–1017, 10 2006.
- [70] J. FAIK. *A Model for Resource-Aware Load Balancing on Heterogeneous and Non-Dedicated Clusters*. PhD thesis, Rensselaer Polytechnic Institute, Troy, 2005.
- [71] A. FEDOROV, E. BILLET, M. PRASTAWA, A. RADMANESH, G. GERIG, R. KIKINIS, S. K. WARFIELD, AND N. CHRISOCHOIDES. Evaluation of brain MRI alignment with the robust Hausdorff distance measures. In *Proc. of ISVC 2008*, pages 594–603, 2008.
- [72] A. FEDOROV, N. CHRISOCHOIDES, R. KIKINIS, AND S. K. WARFIELD. Tetrahedral mesh generation for medical imaging. *ISC Insight Journal, 2005 MICCAI Open Source Workshop*, 2005. <http://hdl.handle.net/1926/35>.
- [73] M. FELLER, I. FOSTER, AND S. MARTIN. GT4 GRAM: A functionality and performance study. In *Proc. of Teragrid 2007 Conference*, pages 1–5, 2007.
- [74] M. FERRANT. *Physics-based Deformable Modeling of Volumes and Surfaces for Medical Image Registration, Segmentation and Visualization*. PhD thesis, Université Catholique de Louvain, 2001.
- [75] M. FERRANT, A. NABAVI, BENOIT M. MACQ, F. A. JOLESZ, R. KIKINIS, AND SIMON K. WARFIELD. Registration of 3d intraoperative MR images of the brain using a finite element biomechanical model. *IEEE Transactions on Medical Imaging*, 20(12):1384–1397, 2001.
- [76] B. FISCHER AND J. MODERSITZKI. Ill-posed medicine - an introduction to image registration. *Inverse Problems*, 24(3):034008 (16pp), 2008.
- [77] R. FISHER, S. PERKINS, A. WALKER, AND E. WOLFART. HIPR2 Image Processing Learning Resources, 2009. <http://homepages.inf.ed.ac.uk/rbf/HIPR2/>.

- [78] J. M. FITZPATRICK AND J. B. WEST. The distribution of target registration error in rigid-body point-based registration. *Medical Imaging, IEEE Transactions on*, 20(9):917–927, 2001.
- [79] THE CENTRE FOR COMPLEX COOPERATIVE SYSTEMS. MammoGrid, 2008. <http://www.cems.uwe.ac.uk/cccs/project.php?name=mammogrid>.
- [80] I. FOSTER. The anatomy of the Grid: Enabling scalable virtual organizations. In *Proc. of 7th International Euro-Par Conference Manchester on Parallel Processing*, pages 1–4, 2001.
- [81] I. FOSTER. What is the grid? - a three point checklist. *GRIDtoday*, 1(6), July 2002.
- [82] I. FOSTER. Globus Toolkit Version 4: Software for service-oriented systems. *Journal of Computer Science and Technology*, 21(4):513–520, 2006.
- [83] I. FOSTER AND C. KESSELMAN. *The Grid: A blueprint for a new computing infrastructure*. Morgan Kaufmann, San-Francisco, 1998.
- [84] I. FOSTER, C. KESSELMAN, J. M. NICK, AND S. TUECKE. *The Physiology of the Grid*, pages 171–198. Grid Computing: Making the Global Infrastructure a Reality. Wiley, 2003.
- [85] L. A. FREITAG, P. M. KNUPP, T. S. MUNSON, AND S. M. SHONTZ. A comparison of optimization software for mesh shape-quality improvement problems. In *Proc. of 11th IMR*, pages 29–40, 2002.
- [86] J. P. FREY. Yams: A fully automatic adaptive isotropic surface remeshing procedure. Technical Report RT-0252, INRIA, 2001.
- [87] J. P. FREY AND P.-L. GEORGE. *Mesh Generation*. HERMES Science Publishing, 2000.
- [88] P. FREY, B. SARTER, AND M. GAUTHERIE. Fully automatic mesh generation for 3-d domains based upon voxel sets. *Int. J. for Num. Meth. in Engng.*, 37(16):2735–2753, 1994.
- [89] D. G. FRYBACK AND J. R. THORNBURY. The efficacy of diagnostic imaging. *Medical Decision Making*, 11:88–94, 1991.
- [90] A. FUCHS. Almost regular triangulations of trimmend NURBS-solids. *Engineering with Computers*, 17(1):55–65, 2001.
- [91] W. GAUTSCHI. *Numerical analysis: An introduction*. Birkhauser, 1997.
- [92] GEMSS. Grid-Enabled Medical Simulation Services, 2008. <http://www.it.necelab.eu/gemss/>.
- [93] P.-L. GEORGE AND H. BOROUCHAKI. *Delaunay Triangulation and Meshing. Application to Finite Elements*. HERMES, / 1998.



- [94] T. GLATARD, X. PENNEC, AND J. MONTAGNAT. Performance evaluation of grid-enabled registration algorithms using bronze-standards. In *Proc. of MICCAI 2006*, pages 152–160, 2006.
- [95] GLOBUS ALLIANCE. Firewall howto, 2009. <http://dev.globus.org/wiki/FirewallHowTo>.
- [96] R.C. GONZALEZ AND R.E. WOODS. *Digital Image Processing*. Prentice Hall, August 2007.
- [97] C. S. GOODMAN. Hta 101: Introduction to health technology assessment, 2008.
- [98] I. GRACHEV, D. BERDICHEVSKY, S. L. RAUCH, S. HECKERS, D. N. KENNEDY, V. S. CAVINESS, AND ALPERT. A method for assessing the accuracy of inter-subject registration of the human brain using anatomic landmarks. *NeuroImage*, 9(2):250–268, February 1999.
- [99] S. GRAUPNER AND C. REIMANN. Globus grid and firewalls: Issues and solutions in a utility data center environment. Technical Report HPL-2002-278, Internet Systems and Storage Laboratory, HP Laboratories, 2002. <http://www.hpl.hp.com/techreports/2002/HPL-2002-278.pdf>.
- [100] W. GROPP, E. HABER, S. HELDMANN, D. KEYES, N. MILLER, J. SCHOPF, AND T. YANG. *Grid-based Image Registration*, volume 239 of *Grid-Based Problem Solving Environments*, pages 435–448. Springer, 2007.
- [101] J. V. HAJNAL, D. J. HAWKES, AND D. L. G. HILL. *Medical image registration*. CRC Press, 2001.
- [102] U. HARTMANN AND F. KRUGGEL. A fast algorithm for generating large tetrahedral 3d finite element meshes from magnetic resonance tomograms. In *IEEE Workshop on Biomedical Image Analysis*, page 184, 1998.
- [103] T. HEIMANN, I. OGUZ, I. WOLF, M. STYNER, AND H.-P. MEINZER. Implementing the automatic generation of 3d statistical shape models with ITK. *Insight Journal*, 2006 MICCAI Open Science Workshop, 2006.
- [104] P. HELLIER, C. BARILLOT, I. COROUGE, B. GIBAUD, G. LE GOUALHER, D. L. COLLINS, A. EVANS, G. MALANDAIN, N. AYACHE, G. E. CHRISTENSEN, AND H. J. JOHNSON. Retrospective evaluation of intersubject brain registration. *Medical Imaging, IEEE Transactions on*, 22(9):1120–1130, 2003.
- [105] S. J. HENTSCHEL AND R. SAWAYA. Optimizing outcomes with maximal surgical resection of malignant gliomas. *Cancer control : journal of the Moffitt Cancer Center*, 10(2):109–114, 2003.
- [106] D. L. G. HILL, P. G. BATCHELOR, M. HOLDEN, AND D. J. HAWKES. Medical image registration. *Physics in Medicine and Biology*, 46:R1–R45, 2001.
- [107] D. HOLLINSWORTH. The workflow reference model. Technical Report TC00-1003, Workflow Management Coalition, 1994.

- [108] D.P. HUTTENLOCHER, D. KLANDERMAN, AND W.J. RUCKLIDGE. Comparing images using the Hausdorff distance. *IEEE Trans. Pat. Anal. and Mach. Intel.*, 15(9):850–863, 1993.
- [109] L. IBANEZ AND W. J. SCHROEDER. *The ITK Software Guide*. Kitware Inc, 2003.
- [110] F. INO, Y. KAWASAKI, T. TASHIRO, Y. NAKAJIMA, Y. SATO, S. TAMURA, AND K. HAGIHARA. A parallel implementation of 2-d/3-d image registration for computer-assisted surgery. *Int J for Bioinformatics Research and Applications*, 2(4):341–357, 2006.
- [111] F. INO, K. OYAMA, AND K. HAGIHARA. A data distributed parallel algorithm for nonrigid image registration. *Parallel Computing*, 31(1):19–43, 1 2005.
- [112] F. INO, K. OYAMA, Y. KAWASAKI, A. TAKEUCHI, Y. MIZUTANI, J. MASUMOTO, Y. SATO, N. SUGANO, T. NISHII, H. MIKI, H. YOSHIKAWA, K. YONENOBU, S. TAMURA, T. OCHI, AND K. HAGIHARA. A high-performance computing service over the internet for nonrigid image registration. *International Congress Series*, 1256:193–199, 6 2003.
- [113] P. JACCARD. Measures of the amount of flora in the alpine zone. *New Phytologist*, 11:37–50, 1912.
- [114] P. JANNIN, J. MICHAEL FITZPATRICK, D. J. HAWKES, X. PENNEC, R. SHAHIDL, AND M. W. VANNIER. Validation of medical image processing in image-guided therapy. *IEEE Transactions on Medical Imaging*, 21(12):1445–1449, 2002.
- [115] P. JANNIN, E. KRUPINSKI, AND SIMON K. WARFIELD. Guest editorial validation in medical image processing. *IEEE Transactions on Medical Imaging*, 25(11):1405–1409, 2006.
- [116] F. A. JOLESZ. Future perspectives for intraoperative MRI. *Neurosurgery clinics of North America*, 16(1):201–213, 2005.
- [117] A. E. KERDOK, S. M. COTIN, M. P. OTTENSMEYER, A. M. GALEA, R. D. HOWE, AND S. L. DAWSON. Truth cube: establishing physical standards for soft tissue simulation. *Medical Image Analysis*, 7(3):283–291, 2003.
- [118] KITWARE INC. CMake: Cross Platform Make, 2009. Version 2.6, <http://cmake.org/>.
- [119] KITWARE INC. File formats for vtk version 4.2, 2009. <http://www.vtk.org/VTK/img/file-formats.pdf>.
- [120] KITWARE INC. ITK: The Insight toolkit, 2009. Version 3.11.0, <http://itk.org/>.
- [121] KITWARE INC. VTK: The Visualization toolkit, 2009. Version 5.2, <http://vtk.org/>.
- [122] R. M. KOCH, M. H. GROSS, F. R. CARLS, D. F. VON BUREN, G. FANKHAUSER, AND Y. I. H. PARISH. Simulating facial surgery using finite element models. In *Proceedings of SIGGRAPH'96*, pages 421–428, 1996.

- [123] R. KOLLURI, J. R. SHEWCHUK, AND J. F. O'BRIEN. Spectral surface reconstruction from noisy point clouds. In *Proc. of SGP'04*, pages 11–21, 2004.
- [124] A. H. KUIJPERS, M. H. CLAESSENS, AND A. A. SAUREN. The influence of different boundary conditions on the response of the head to impact: a two-dimensional finite element study. *Journal of Neurotrauma*, 12(4):715–724, 1995.
- [125] F. LABELLE AND J. R. SHEWCHUK. Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. In *Proc. of ACM SIGGRAPH*, page 57, 2007.
- [126] B. J. LAUNDRE, B. J. JELLISON, B. BADIE, A. L. ALEXANDER, AND A. S. FIELD. Diffusion tensor imaging of the corticospinal tract before and after mass resection as correlated with clinical motor findings: Preliminary data. *AJNR Am J Neuroradiol*, 26(4):791–796, April 2005.
- [127] H. LIPPMANN AND F. KRUGGEL. Quasi-real-time neurosurgery support by MRI processing via grid computing. *Neurosurgery Clinics of North America*, 16(1):65–75, 2005.
- [128] R. LOHNER AND J. CEBRAL. Parallel advancing front grid generation. In *Proceedings of the 8th International Meshing Roundtable*, pages 67–74, 1999.
- [129] W. E. LORENSEN AND H. E. CLINE. Marching cubes: A high resolution 3d surface construction algorithm. In *Proc. of ACM SIGGRAPH*, pages 163–169, 1987.
- [130] S. MANOS, S. ZASADA, M. D. MAZZEO, R. HAINES, G. DOCTORS, S. BREW, R. PINNING, J. BROOKE, AND P. V. COVENEY. Patient specific whole cerebral blood flow simulation: A future role in surgical treatment for neurovascular pathologies. In *Proc. of Teragrid'08*, 2008.
- [131] D. MARCUM. Efficient generation of high-quality unstructured surface and volume grids. *Engineering with Computers*, 17(3):211–233, 2001.
- [132] C. R. MAURER, J. J. MCCRORY, AND M. J. FITZPATRICK. Estimation of accuracy in localizing externally attached markers in multimodal volume head images. In *Proc. of SPIE Medical Imaging: Image Processing*, Murray H. Loew, editor, volume 1898, pages 43–54. SPIE, 1993.
- [133] C. R. MAURER, R. QI, AND V. RAGHAVAN. A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):265–270, 2003.
- [134] D. G. MITCHELL AND M. COHEN. *MRI Principles*. Saunders, December 2003. howpublished: Hardcover.
- [135] S.A. MITCHELL AND S.A. VAVASIS. Quality mesh generation in higher dimensions. *SIAM J. Comp*, 29(4):1334–1370, 2000.

- [136] A. MOHAMED. *Combining Statistical and Biomechanical Models for Estimation of Anatomical Deformations*. PhD thesis, Johns Hopkins University, 2005.
- [137] A. MOHAMED AND C. DAVATZIKOS. Finite element mesh generation and remeshing from segmented medical images. In *Proc. of IEEE ISBI*, pages 420–423, 2004.
- [138] N. P. MOLINO. *Mesh generation and fracture for deformable bodies*. PhD thesis, Stanford University, 2004.
- [139] F. MORAIN-NICOLIER, S. LEBONVALLET, E. BAUDRIER, AND S. RUAN. Hausdorff distance based 3d quantification of brain tumor evolution from MRI images. In *Proc. 29th Annual Intl Conf of the IEEE EMBS*, pages 5597–5600, 2007.
- [140] M.SNIR, S.W.OTTO, S.HAUSS-LEDERMAN, D.WALKER, AND J.DONGARRA. *MPI: The complete reference*. The MIT Press, 1998.
- [141] NAMIC. SandBox Repository, 2006. <http://svn.na-mic.org/NAMICSandBox/trunk>.
- [142] NATIONAL LAMBDARAIL. Participants, 2009. <http://www.nlr.net/>.
- [143] NETGEN. Automatic mesh generator, 2009. <http://www.hpfem.jku.at/netgen/>.
- [144] G.M. NIELSON. On Marching Cubes. *IEEE Trans. on Visualization and Computer Graphics*, 9(3):283–297, 2003.
- [145] D. NURMI, J. BREVIK, AND R. WOLSKI. QBETS: Queue bounds estimation from time series. In *Proc. of JSSPP’07*, pages 76–101, 2008.
- [146] D. NURMI, R. WOLSKI, AND J. BREVIK. VARQ: virtual advance reservations for queues. In *Proc. of HPDC’08*, pages 75–86, 2008.
- [147] Y. OHTAKE, A. BELYAEV, AND A. PASKO. Dynamic meshes for accurate polygonization of implicit surfaces with sharp features. In *SMI’01: Proceedings of the International Conference on Shape Modeling & Applications*, page 74, 2001.
- [148] C. OLLIVIER-GOOCH. GRUMMP: Generation and refinement of unstructured, mixed-element meshes in parallel, 2005. <http://tetra.mech.ubc.ca/GRUMMP/>, accessed 14 April 2006.
- [149] J. P. O’SHEA, S. WHALEN, D. M. BRANCO, N. M. PETROVICH, K. E. KNIERIM, AND A. J. GOLBY. Integrated image- and function-guided surgery in eloquent cortex: a technique report. *The international journal of medical robotics + computer assisted surgery : MRCAS*, 2(1):75–83, March 2006.
- [150] S. OUDOT, L. RINEAU, AND M. YVINEC. Meshing volumes bounded by smooth surfaces. In *Proc. of 14th IMR*, pages 203–219, 2005.

- [151] S. OURSELIN, R. STEFANESCU, AND X. PENNEC. Robust registration of multimodal images: towards real-time clinical applications. In *Proc. of Medical Image Computing and Computer-Assisted Intervention (MICCAI'02)*, pages 140–147. Springer, 2002.
- [152] S. J. OWEN. A survey of unstructured mesh generation technology. In *Proc. of 7th IMR*, pages 239–267, 1998.
- [153] S. J. OWEN, DAVID R. WHITE, AND TIMOTHY J. TAUTGES. Facet-based surfaces for 3d mesh generation. In *Proceedings of the 11th International Meshing Roundtable*, pages 297–312, 2002.
- [154] M. PARASHAR AND CRAIG A. LEE. Special issue on grid computing. *Proceedings of IEEE*, 93(3):479–483, 2005.
- [155] K. D. PAULSEN, X. JIA, AND J. M. JR SULLIVAN. Finite element computations of specific absorption rates in anatomically conforming full-body models for hyperthermia treatment analysis. *IEEE Transactions on Biomedical Engineering*, 40(9):933–945, 1993.
- [156] X. PENG, W. CHEN, AND Q. MA. Feature-based nonrigid image registration using a Hausdorff distance matching measure. *Optical Engineering*, 46(5):057201, 2007.
- [157] P. PERONA AND J. MALIK. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(7):629–639, July 1990.
- [158] P. O PERSSON AND GILBERT STRANG. A simple mesh generator in MATLAB. *SIAM Review*, 46(2):329–345, 2004.
- [159] PETSC. Portable, extensible toolkit for scientific computation. <<http://www-unix.mcs.anl.gov/petsc/petsc-2/>>, 2005.
- [160] L. A. PIEGL AND W. TILLER. *The NURBS Book (Monographs in Visual Communication)*. Springer, November 1996.
- [161] J. PIEPMEIER, S. CHRISTOPHER, D. SPENCER, T. BYRNE, J. KIM, J. P. KNISEL, J. LACY, L. TSUKERMAN, AND R. MAKUCH. Variations in the natural history and survival of patients with supratentorial low-grade astrocytomas. *Neurosurgery*, 38(5), May 1996.
- [162] M. PRASTAWA, E. BULLITT, AND G. GERIG. Synthetic ground truth for validation of brain tumor MRI segmentation. In *Proc. MICCAI 2006, LNCS 3749*, pages 26–33, 2006.
- [163] A. PUZRIN, O. SKRINJAR, C. OZAN, S. KIM, AND S. MUKUNDAN. Image guided constitutive modeling of the silicone brain phantom. In *SPIE Medical Imaging: Image Processing*, Robert L. Galloway, Jr, and Kevin R. Cleary, editors, volume 5744, pages 157–164. SPIE, 2005.

- [164] R. RADOVITZKY AND M. ORTIZ. Tetrahedral mesh generation based on node insertion in crystal lattice arrangements and advancing-front-delaunay triangulation. *Computer Methods in Applied Mechanics and Engineering*, 187(3-4):543–569, 2000.
- [165] I. RAICU, Z. ZHANG, M. WILDE, I. FOSTER, P. BECKMAN, K. ISKRA, AND B. CLIFFORD. Toward loosely coupled programming on petascale systems. In *Proc. of Supercomputing 2008*, 2008.
- [166] J. REXILIUS. *Physics-Based Nonrigid Registration for Medical Image Analysis*. MS thesis, Medical University of Luebeck, 2001.
- [167] P. ROGELJ, S. KOVAČIČ, AND J. C. GEE. Point similarity measures for non-rigid registration of multi-modal data. *Comp. Vis. and Image Underst.*, 92(1):112–140, 2003.
- [168] T. ROHLFING AND C. MAURER. Non-rigid image registration in shared-memory multiprocessor environments with application to brains, breasts and bees. *IEEE Transactions on Information Technology in Biomedicine*, 7(1):16–25, 2003.
- [169] P.J. ROUSSEEUW AND A.M. LEROY. *Robust Regression and Outlier Detection*. John Wiley & Sons, 1987.
- [170] A.L. ROWLAND, M. BURNS, J.V. HAJNAL, D. RUECKERT, AND D.L.G. HILL. Using Grid services from behind a firewall. In *Proc. of UK e-Science All Hands Meeting*, 2005. <http://www.allhands.org.uk/2005/proceedings/papers/402.pdf>.
- [171] SANDIA NATIONAL LABORATORIES. Mesquite: Mesh Quality Improvement Toolkit, 2009. Version 1.1.7, <http://www.cs.sandia.gov/optimization/knupp/Mesquite.html>.
- [172] J. A. SCHNABEL, C. TANNER, A. D. CASTELLANO-SMITH, A. DEGENHARD, M. O. LEACH, D. R. HOSE, D. L. G. HILL, AND D. J. HAWKES. Validation of nonrigid image registration using finite-element methods: application to breast mr images. *Medical Imaging, IEEE Transactions on*, 22(2):238–247, 2003.
- [173] J. SCHÖBERL. NETGEN: An advancing front 2d/3d-mesh generator based on abstract rules. *Computing and Visualization in Science*, 1(1):41–52, 07/20 1997.
- [174] P. SCHÖNEMANN. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, March 1966.
- [175] J. R. SHEWCHUK. Constrained Delaunay tetrahedralizations and provably good boundary recovery. In *Proceedings of the 11th International Meshing Roundtable*, pages 193–204, 2002.
- [176] J. R. SHEWCHUK. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry: Theory and Applications*, 22(1–3):21–74, may/ 2002.

- [177] J. R. SHEWCHUK. What is a good linear element? Interpolation, conditioning, and quality measures. In *Proc. of 11th IMR*, pages 115–126, 2002.
- [178] S. M. SHONTZ AND S. A. VAVASIS. A mesh warping algorithm based on weighted laplacian smoothing. In *Proceedings of International Meshing Roundtable 2003*, pages 147–158, 2003.
- [179] H. SI. On refinement of constrained Delaunay tetrahedralizations. In *Proc. of 15th IMR*, pages 509–528, 2006.
- [180] H. SI. Tetgen: A quality tetrahedral mesh generator and three-dimensional delaunay triangulator, 2009. Version 1.4.2, <http://tetgen.berlios.de/>.
- [181] R. SIBSON, A. BOWYER, AND C. OSMOND. Studies in the robustness of multidimensional scaling: euclidean models and simulation studies. *Journal of Statistical Computation and Simulation*, 13(3):273–296, 1981.
- [182] SIMULOG. GHS3D. <<http://www.simulog.fr/mesh/gener2.htm>>, 2005.
- [183] O. SKRINJAR, A. NABAVI, AND J. DUNCAN. Model-driven brain shift compensation. *Medical Image Analysis*, 6(4):361–373, 12 2002.
- [184] M. SNIR AND W. GROPP. *MPI: The Complete Reference*. The MIT Press, 1998.
- [185] J. A. STARK. Adaptive image contrast enhancement using generalizations of histogram equalization. *IEEE Transactions on Image Processing*, 9(5):889–896, 2000.
- [186] R. STEFANESCU, X. PENNEC, AND N. AYACHE. Parallel non-rigid registration on a cluster of workstations. In *Proc. of HealthGrid’03*, 2003.
- [187] R. STEFANESCU, X. PENNEC, AND N. AYACHE. Grid powered nonlinear image registration with locally adaptive regularization. *Medical Image Analysis*, 8(3):325–342, 9 2004.
- [188] TERA GRID. User Support & Documentation, 2008. <http://teragrid.org/userinfo/>.
- [189] J. D. TERESCO, J. FAIK, AND J. E. FLAHERTY. Resource-aware scientific computation on a heterogeneous cluster. *Computing in Science & Engineering*, 7(2):40–50, 2005.
- [190] THE OPEN MPI GROUP. Openmpi, 2009. <http://www.open-mpi.org/>.
- [191] J.-P THIRION. Image matching as a diffusion process: an analogy with maxwell’s demons. *Medical image analysis*, 2(3):243–260, /9 1998.
- [192] S. TIMONER. *Compact Representations for Fast Nonrigid Registration of Medical Images*. PhD thesis, MIT, 2003.
- [193] T. UDESHI. Tetrahedral mesh generation from segmented voxel data. In *Proc. of 12th IMR*, pages 425–436, 2003.

- [194] A. ÜNGÖR. Off-centers: A new type of steiner points for computing size-optimal quality-guaranteed delaunay triangulations. *Computational Geometry*, 42(2):109–118, 2009.
- [195] G. VON LASZEWSKI, M. HATEGAN, AND D. KODEBOYINA. *Java CoG kit workflows*, pages 340–356. Workflows for eScience. Springer, 2007.
- [196] D. ŠKERL, B. LIKAR, AND F. PERNUŠ. A protocol for evaluation of similarity measures for non-rigid registration. *Medical Image Analysis*, 12(1):42–54, 2008.
- [197] S. K. WARFIELD, M. FERRANT, X. GALLEZ, A. NABAVI, AND F. A. JOLESZ. Real-time biomechanical simulation of volumetric brain deformation for image guided neurosurgery. In *Supercomputing '00: Proceedings of the 2000 ACM/IEEE conference on Supercomputing (CDROM)*, page 23, Washington, DC, USA, 2000. IEEE Computer Society.
- [198] S. K. WARFIELD, F. A. JOLESZ, AND R. KIKINIS. A high performance computing approach to the registration of medical imaging data. *Parallel Comput.*, 24(9-10):1345–1368, 1998.
- [199] S. K. WARFIELD, F. A. JOLESZ, AND R. KIKINIS. Real-time image segmentation for image-guided surgery. In *Supercomputing '98: Proceedings of the 1998 ACM/IEEE conference on Supercomputing (CDROM)*, pages 1–7, Washington, DC, USA, 1998. IEEE Computer Society.
- [200] S. K. WARFIELD, A. NABAVI, T. BUTZ, K. TUNCALI, S. G. SILVERMAN, P. BLACK, F. A. JOLESZ, AND R. KIKINIS. Intraoperative segmentation and nonrigid registration for image guided therapy. In *MICCAI '00: Proceedings of the Third International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 176–185, London, UK, 2000. Springer-Verlag.
- [201] S. K. WARFIELD, K. H. ZOU, AND W. M. WELLS. Simultaneous truth and performance level estimation (STAPLE): An algorithm for the validation of image segmentation. *IEEE Transactions on Medical Imaging*, 23(7):903–921, 2004.
- [202] V. WELCH, I. FOSTER, C. KESSELMAN, O. MULMO, L. PEARLMAN, J. GAWOR, S. MEDER, AND F. SIEBENLIST. X.509 proxy certificates for dynamic delegation. In *In Proceedings of the 3rd Annual PKI R&D Workshop*, volume 60, 2004.
- [203] WIKIPEDIA. Homeomorphism, 2009. <http://en.wikipedia.org/wiki/Homeomorphism>.
- [204] I. WU. *Multilist Scheduling: A New Parallel Programming Model*. PhD thesis, School of Comp. Sci., Carnegie Mellon University, Pittsburg, PA 15213, July 1993.
- [205] Y. YAN, P. L. NUNEZ, AND R. T. HART. Finite-element model of the human head: scalp potentials due to dipole sources. *Medical and Biological Engineering and Computing*, 29(5):475–481, 1991.



- [206] M. A. YERRY AND M. S. SHEPHARD. Automatic three-dimensional mesh generation by the modified-octree technique. *Int. J. Num. Meth. Engng*, 20(11):1965–1990, 1984.
- [207] T. S. YOO. *Insight into images*. A K Peters, 2004.
- [208] Y. ZHANG, C. BAJAJ, AND B.-S. SOHN. 3d finite element meshing from imaging data. *Comp. Meth. in App. Mech. and Engng.*, 194(48-49):5083–5106, 2005.
- [209] C. ZHAO, W. SHI, AND Y. DENG. A new Hausdorff distance for image matching. *Pattern Recognition Letters*, 26:581–586, 2005.
- [210] Y. ZHAO, M. HATEGAN, B. CLIFFORD, I. FOSTER, G. VON LASZEWSKI, V. NEFEDOVA, I. RAICU, T. STEF-PRAUN, AND M. WILDE. Swift: fast, reliable, loosely coupled parallel computation. In *Proc. of 2007 IEEE Congress on Services*, pages 199–206, 2007.
- [211] O. C. ZIENKIEWICZ, R. L. TAYLOR, AND J. Z. ZHU. *The Finite Element Method: Its Basis and Fundamentals*. Butterworth-Heinemann, 2005.

## VITA

### Andriy Yuri Fedorov

Andriy Yuri Fedorov joined the graduate program of the College of William and Mary in 2001 after finishing four years of study at the Institute for Computer Information Technologies (IKIT) in Ternopil, Ukraine. Andriy Fedorov studied the communication aspects related to load balancing in asynchronous parallel applications, earning his M.S. degree in Computer Science from the College of William and Mary in December 2003. Most of 2005 Andriy Fedorov spent as a Visiting Research Fellow at Computational Radiology Laboratory within Surgical Planning Laboratory, Brigham and Women's Hospital, Boston. During that time Andriy Fedorov was first introduced to the field of medical imaging that motivated his research in the following years. Upon graduation from the PhD program, Andriy Fedorov is joining Surgical Planning Laboratory at Brigham and Women's Hospital and Harvard Medical School as a post-doctoral Research Fellow. His current research project is the development of ChangeTracker – a clinical research tool for volumetric measurement of meningioma brain tumors from MR images.