# Efficient and Reliable Methods for Direct Parameterized Image Registration

*Rupert Brooks*

Department of Electrical & Computer Engineering
McGill University
Montréal, Canada

September 2008

---

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

THIS PAGE INTENTIONALLY BLANK

## Acknowledgments

I would like to first thank my supervisor, Tal Arbel, who first encouraged me to explore the field of image registration. She has suffered through multiple readings of all the material in this thesis, including my frequently very unclear initial presentations of ideas, and has always found a way to improve the work. Many thanks also go to my advisory committee: D. Louis Collins, Doina Precup and James Clark, who have followed the highs and lows of this research and provided sage advice along the way. D. Louis Collins also generously included me in his IGNS research group meetings and discussions, something I have appreciated greatly. Bruce Pike helped prepare me for, and helped administer my comprehensive exam.

Isabelle, je ne peux pas exprimer toute mon appréciation dans ces quelques phrases. Je te remercie pour notre vie ensemble, pour être toi, pour ta patience et pour ton aide en lisant plusiers chapitres de cette thèse. Thanks also to my mother, Nancy, and siblings Rose, Tina and Kris for their patience and understanding, and for not asking *too* frequently when I would finally finish.

Tal Arbel, Frank Ferrie, David Lowther, Andraws Swidan and Martin Levine gave me the privilege of acting as teaching assistant during their courses. I would like to particularly thank Frank Ferrie and David Lowther for giving me the job of preparing the lab exercises for ECSE211. This experience was very rewarding, and not *only* because it involved playing with Lego.

To my friends and colleagues at CIM and the MNI, especially Cathy, Matt, Rola, Frank R., Prasun, Prakash, Shufei, Scott, Peter, Carlos, Sandra, Laurence, Simon and Jerome thanks for everything. You are good friends, wise colleagues, and many of you have even helped me move. Extra thanks to Cathy for her continual friendship and collaboration, and also for carefully checking the math in Chapter 7 and Appendix F. Your thesis is next.

Grad students have to eat, even if only ramen and sardines, and so we rely on funding. Thank you Tal, for generously funding a long Ph.D., and never flinching at conferences in exotic locations. NSERC provided for 2 years of funding, and Doina Precup provided funds to attend IJCAI'07. Finally, note that most of the computational experiments in this thesis used the CLUMEQ and RQCHP clusters, and would have been impossible otherwise.

## Abstract

This thesis examines methods for efficient and reliable image registration in the context of computer vision and medical imaging. Direct, parameterized image registration approaches work by minimizing a difference measure between a fixed reference image, and the image warped to match it. The calculation of this difference measure is the most computationally intensive part of the process and for faster registration it either has to be calculated faster, or calculated fewer times. Both possibilities are addressed in detail.

Efficiency and reliability are addressed in four ways (1) Methods are presented for generalizing the Gauss-Newton Hessian approximation to the non-least squares case, and for the optimal selection of scaling factors for the transformation parameters. Both of these enhance performance by enabling optimization algorithms to perform fewer evaluations of the difference measure. The performance of a wide range of optimization algorithms is analyzed both theoretically and experimentally, and guidelines are presented for optimizer selection based on the characteristics of the registration problem. (2) Using only a portion of the available pixels results in faster calculation but suffers from a potential loss of accuracy. An algorithm is presented which applies formal deliberation control methods to managing this tradeoff. By managing the amount of image data used at every evaluation of the cost function, the algorithm adapts to the nature of the images and the stage of the optimization. This adaptive approach allows greater efficiency without sacrificing reliability. (3) It is shown that the scale used to compute the derivative is a critical factor to consider when selecting subsets of pixels for registration, that has largely been ignored in previous work. Finally, (4) two existing efficient registration approaches, the inverse compositional, and efficient second order algorithms, rely on specialized optimizer update steps and specialized parameterizations. A generalization of these methods is presented that both identifies the connections between them, and eliminates the need for these specialized components.

Throughout the thesis, application specific approaches have been avoided. Both 2D and 3D images from both computer vision and medical imaging applications have been used throughout. Consequently each of the efficient registration methods can be applied, alone or in combination, to a very wide range of problems.

## Résumé

Cette thèse examine différentes méthodes efficaces et fiables de recalage dans les contextes de la vision numérique et de l'imagerie médicale. Les approches directes minimisent une mesure de différence entre une image de référence (fixe) et sa version déformée. Le calcul de cette mesure constitue la partie la plus intensive du processus. Pour un recalage rapide, la mesure doit être calculée plus rapidement, ou moins souvent. Les deux possibilités sont discutées en détail dans la thèse.

Le rendement et la fiabilité sont abordés de quatre manières différentes. (1) Le nombre de calculs peut être diminué en généralisant l'approximation de Gauss-Newton de la matrice hessienne pour une fonction de coût autre que de type moindres carrés, ou par la sélection optimale d'un facteur d'échelle pour les paramètres de la transformation. Les performances d'une variété d'algorithmes d'optimisation sont analysées, et des lignes directrices sont proposées pour la sélection d'un optimiseur basée sur les caractéristiques du problème de recalage. (2) L'utilisation d'une portion des pixels résultent en un calcul plus rapide de la mesure de différence, mais peut engendrer une perte d'exactitude. Un algorithme basé sur le principe du contrôle de délibération est proposé afin de gérer ce compromis. La gestion de la quantité d'information utilisée à chaque évaluation de la fonction de coût permet une adaptation à la nature des images ainsi qu'au stade de l'optimisation. Cette approche permet un haut rendement sans sacrifier la fiabilité. (3) Le facteur d'échelle utilisé pour le calcul des dérivées a rarement été abordé. Il est démontré que ce facteur est critique pour la sélection de sous-ensembles de pixels pour le recalage. Finalement, (4) La méthode compositionnelle inverse et la méthode à haut rendement du second ordre sont basées sur des itérations de l'optimiseur ainsi que sur des paramétrisations spécialisées. Une généralisation de ces deux méthodes est présentée en relevant leurs liens, et en éliminant le recours aux éléments spécialisés.

La thèse aborde le recalage d'une manière générale, et les approches développées pour des applications spécifiques ont été évitées. Des données bi et tridimensionnelles provenant des contextes de la vision numérique et de l'imagerie médicale ont été utilisées pour tester les différents aspects discutés. Par conséquent, les méthodes proposées de recalage efficace peuvent être appliquées, seules ou de façon combinée, à une grande variété de problèmes.

# Contents

# List of Tables

# List of Figures

# List of Algorithms

# Chapter 1

# Introduction

Image registration is the process of finding the spatial relationship between two or more images. This is a very fundamental idea, and image registration forms a key component of a wide range of applications, including intraoperative image guidance (*e.g.,* [5, 54, 160]), template based tracking (*e.g.,* [66, 71]), the construction of anatomic atlases from sets of patient data (*e.g.,* [76, 112]), the georeferencing of remotely sensed images (*e.g.,* [55, 204]), vision guided control of robotic manipulators (visual servoing, *e.g.,* [60, 139]), superresolution (*e.g.,* [98, 111]) and video coding (*e.g.,* [72]), to name just a few. It is not surprising then, that registration is one of the oldest and most frequently recurring problems in computer vision and medical imaging.

All these applications benefit from fast and reliable methods, but certain applications stand out as being particularly time-sensitive. Tracking, visual servoing and video coding, for example, must often be done at or near video frame rates. However, I was particularly motivated to conduct this research by image registration problems arising in image guided neurosurgery (*e.g.,* [5, 54, 160]). Surgery on the brain is a carefully planned and precise process and the surgical planning is usually done on images taken preoperatively. However, during the procedure, the patient's brain changes shape, resulting in a brainshift of up to 5 cm [129], which reduces the effectiveness of the preoperative images for guidance. The shift must be determined by relating the coordinate systems of the preoperative, and the intraoperative images, in other words, by solving an image registration problem.

This motivating problem is typically characterized by having large 3D volumetric images of different modalities, using curved – but not arbitrarily complex – warps, and having a fairly good starting estimate of the final transformation. Direct, parameterized, image registration algorithms are particularly appropriate for this context [91, 145], but they have a tendency to be slow. Speed is important in this context, and it goes without saying that reliability is also critical.

However, while this was my original motivation, this is not a thesis about image guided surgery problems, but a thesis about direct image registration. Image registration has an enormous range of applications, and has been addressed by many authors, in many contexts. In this work I have approached the direct image registration problem itself and abstracted and incorporated results from many fields. I believe that engineering should proceed from general principles to specific solutions, and in each component of this thesis I have tried to first abstract a general theory, and then rigorously test it by experimenting on a range of image types. Best results will always be obtained by tuning an approach to the problem at hand. It is not my intention to avoid that tuning, but to guide it in a principled way. Throughout this work it has been my goal to discover the abstract principles that should be used to guide specific implementations for direct image registration problems.

In Chapter 4, I address the issue of optimizer selection in direct registration problems. Of necessity, this work has a large experimental component, but before performing the experimental analysis, I present a theoretical analysis of why the algorithms perform as they do. It is my hope that this will enable future engineers to make good judgments about optimization choices for their application, without having to run a battery of empirical tests themselves. In Chapter 5, I address the issue of how many pixels to use. Rather than giving a particular number as an answer, I have instead provided a system to automatically determine how many to use. I also ask the question: Why should we choose only a single amount? The system I present adapts the amount at every step. Chapter 6 questions the commonly used heuristic of selecting pixels of high derivative. I show that the scale used to compute the derivative upon which the selection is based is a critical factor in determining the reliability of such an approach. This understanding enables prediction of when that heuristic will be effective, and when it will not. Finally, in Chapter 7 I discuss the

inverse compositional and the efficient second order methods. These are examples of methods that have become popular in some fields, *i.e.,* computer vision and robotics, but have remained nearly unknown in others, *i.e.,* medical imaging. It is my view that the fundamental contributions of both of these methods are clever, effective ways of computing the derivatives of the cost function. Viewed in this manner, I show how these approaches can be generalized to remove the remaining complexities of their current implementations. It is my hope that this will make their adoption in a wide range of problems easy and practical.

This is neither a wholly theoretical, nor a wholly experimental thesis. I have approached each component with the goal of abstracting general themes, but in each case I have thoroughly tested the application of these ideas. In many cases, the experimental results give further interesting insights into the general principles at work. The experimental cases may seem slightly abstracted, one step removed from real clinical problems. This is intentional – a major focus was to keep the experiments objective, controlled, and avoid dependence on special characteristics of the data. Real clinical or other problems come with baggage that may impede clear and objective comparisons of algorithmic approaches.

In the following section, I will first define the problem of image registration and describe the categories of ways in which it can be made more computationally efficient. Section 1.2 describes the specific contributions of this thesis. Section 1.3 then lists the parts of this thesis that have been published, that are being reviewed, and work that is closely related, but not specifically covered in this thesis. Finally, Section 1.4 describes the overall structure of the remainder of this document.

## 1.1 Image Registration

There is a general consensus [41, 62, 138, 181, 208] that registration approaches can be classified into two main groups: feature based methods, and direct methods. Feature based registration methods derive a set of features from the images in question, usually establish correspondence between those feature sets, and derive a registration from these sets of corresponding features.

Direct methods, on the other hand, operate by optimizing a measure of image

difference, $D$, defined directly on the intensities of the images to be registered. In the standard direct registration approach, a pair of images are registered. One of the images, referred to as the fixed, or reference image, $I_f(\boldsymbol{X})$, is held fixed while the other, referred to as the moving or template image, $I_m(\boldsymbol{X})$, is deformed by some transformation, $\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi})$, [145]. Here $\boldsymbol{X}$ is the coordinates of the lattice of image pixels, and $\boldsymbol{\phi}$ is the parameters of the transformation. The set of transformation parameters which minimizes the difference between the two images is considered to be the correct solution. This can be expressed as an optimization problem:

$$\boldsymbol{\phi}_{opt} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}} \left( D(I_f(\boldsymbol{X}), I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))) \right) \tag{1.1}$$

By far the most computationally intensive part of a direct image alignment process is the evaluation of the image difference measure, $D$. There are of course methods to speed up the process without fundamentally changing the calculation, perhaps through better programming (*e.g.,* [178]), or through parallelization (*e.g.,* [7, 121, 162]). But to increase efficiency through algorithmic changes can either be addressed by

1. reducing the number of evaluations of $D$ that are required, or,
2. speeding up the calculation of $D$ itself

Reducing the number of evaluations of $D$ must be achieved by improving the efficiency of the optimization process. As for speeding up the calculation of $D$ itself, this can be achieved in two ways. Either

1. by reducing the number of pixels processed, or,
2. by precalculating and caching parts of the difference measure.

This classification of the different approaches is shown conceptually in Figure 1.1. In this thesis I have made novel contributions to each of these possibilities.

## 1.2 Contributions

The optimization algorithm is a key component of direct image registration, but its effect on the image registration process in isolation has not been studied frequently.

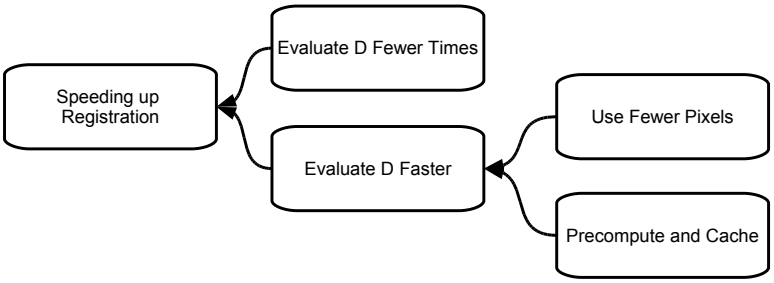



**Figure 1.1** A conceptual view of speeding up direct registration

Previous studies [26, 120, 137] have also focused on very specific registration problems and been very empirical in nature. As a result, it is difficult to generalize their results to different image registration problems. I have investigated the choice of optimizer from a principled perspective, so that my conclusions provide guidelines for future designers of registration algorithms to use in choosing their optimization algorithm. These conclusions are tested rigorously on both 2D and 3D images, for five optimization algorithms representative of those widely used in current registration techniques, over transformations ranging in complexity from 3 to 1029 parameters and for the three most widely used image difference measures. However, before different optimization algorithms could be compared in an unbiased way, two issues which had not been sufficiently addressed by previous work had to be resolved.

First, for the use of Newton-Raphson type optimization methods a Hessian, or approximate Hessian for the cost function must be calculated. For least squares problems, an excellent approximation known as the Gauss-Newton approximation exists [82, 158]. I have shown how this can be generalized to other cost functions, which is important because, for certain problem configurations, the Newton-Raphson optimizer is the method of choice. As a product of that generalization, I have also identified flaws in the existing implementations of Hessians [70, 185] for mutual information. The flawed Hessian approximation seriously degrades the performance of Newton-Raphson optimizers.

Next, it is generally known that scaling of the transformation parameters is important to achieve good results in optimization [59, 80, 85, 151]. However, for image registration problems this issue has either been ignored, or treated in an ad-hoc way.

I have identified the role of these scaling parameters, and presented an optimal means of calculating them. My experimental results show that image registration performance on matrix based transforms is greatly enhanced by using the proper parameter scaling factors.

One of the main ways of speeding up the evaluation of $D$ is to use only some of the pixels to compute it. However, previous work had not addressed the issue of exactly how many pixels are actually required. I was the first to adapt a framework developed in Real-Time Artificial Intelligence, the anytime algorithms [65, 106], to determining how many pixels are required at each stage of the image registration process. A key element of my approach is that the amount of data used is not selected once and fixed, but is adapted at each particular evaluation of the image difference measure. The experiments show clearly that using an adaptive approach achieves speed gains while maintaining the reliability and accuracy of the registration. In contrast, selecting an arbitrary, fixed amount of pixels to use can increase processing speed, but only at the expense of compromising reliability and accuracy.

If only some pixels will be used it is natural to ask if some are better than others. There is kind of a conventional wisdom in the field that selecting certain pixels based on their gradient leads to faster direct image registration. A framework for selecting pixels which addressed this more formally was proposed in [66]. However, the existing method suffers from a sometimes severe decrease in reliability. I have identified the cause of this problem as being due to ignoring the role of the scale of the derivative. The experiments show that the shape of the cost function is changed significantly by this pixel selection process, and its capture radius is proportional to the scale of the derivative used in pixel selection. Thus, the problem can be mitigated by determining the appropriate scale from the uncertainty of the transformation parameters.

The inverse compositional (IC) alignment algorithm [12] is a well-known efficient image registration approach. It achieves faster evaluation of $D$ in the second of the potential ways listed above – by precalculating and caching part of the calculation. However, its implementation is complicated because it requires an optimization algorithm that uses compositional update steps while most standard optimization algorithms use additive steps. I have shown how it is related to another method developed in the field of robotics, the efficient second order method (ESM) [139, 140], which ap-

proaches the problem of registration efficiency by improving the optimization method. The ESM method was only applicable when the transform was a Lie group parameterized along its exponential map. I have shown that both these methods can be viewed as improved means of computing the derivatives of $D$ with respect to different sets transformation parameters. The derivatives can be converted to the desired parameter space by a simple application of the chain rule. This allows the advantages of these methods to be kept, while removing the restrictions on optimization steps, and parameterizations. Experiments were performed on both 2D and 3D datasets using three different optimization algorithms and three different cost functions. These experiments conclusively show that my generalization of the IC algorithm is as fast as the original IC algorithm, and in certain cases it proves much more reliable. The generalization of the ESM algorithm shows a statistically significant improvement in reliability over the classical image registration implementation.

### 1.2.1 Summary of Contributions

This research has made the following original contributions:

1. **A comparison of optimization algorithms in a manner that allows selection of an optimizer based on problem characteristics**
2. **A generalization of the Gauss-Newton Hessian approximation to non-least squares cost functions**
3. **A method for automatically determining an optimal linear parameterization for image registration problems.**
4. **The first algorithm to use anytime algorithms for deliberation control in image registration.**
5. **An explanation of why pixel selection may reduce reliability and ways to mitigate this.**
6. **A generalization of both the inverse compositional alignment and efficient second order methods which removes restrictions on their applicability.**

## 1.3 Published Work

I will now outline all the publications related to the work in this thesis:

### Publications Submitted for Review

[34] Rupert Brooks and Tal Arbel. Generalizing inverse compositional and ESM image alignment. *International Journal of Computer Vision*, 2008. SUBMITTED FOR PUBLICATION.

### Peer-Reviewed Journal Publications

[39] Rupert Brooks, Tal Arbel, and Doina Precup. Anytime similarity measures for faster alignment. *Computer Vision and Image Understanding*, 110(3):378–89, 2008.

### Peer-Reviewed Conference Publications

[38] Rupert Brooks, Tal Arbel, and Doina Precup. Fast image alignment using anytime algorithms. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI2007)*, pages 2078–2083, Hyderabad, India, January 2007.

[33] Rupert Brooks and Tal Arbel. The importance of scale when selecting pixels for image registration. In *Proceedings of the 4th Canadian Conference on Computer and Robot Vision (CRV2007)*, pages 235–242, Montreal, Canada, May 2007.

[32] Rupert Brooks and Tal Arbel. Generalizing inverse compositional image alignment. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR2006)*, volume 2, pages 1200–1203, Hong Kong, August 2006.

### Other Publications

[36] Rupert Brooks and Tal Arbel. Improvements to the itk::KernelTransform and subclasses. *Insight Journal*, March 2007. DSpace handle `http://hdl.handle.net/1926/494`.

[35] Rupert Brooks and Tal Arbel. A homogeneous transform class for the ITK. *Insight Journal*, March 2007. DSpace handle `http://hdl.handle.net/1926/493`.

[37] Rupert Brooks, D. Louis Collins, and Tal Arbel. Scaling angles and distances to maximize efficiency of image registration. Short paper presented at the 8th International Conference on Medical Image Computing and Computer Aided Intervention (MICCAI 2005), October 2005. available online `http://www.ia.unc.edu/MICCAI2005/ShortPapers/`.

### Related Work

The work in the following papers does not directly appear in this thesis, but is closely related to this research.

[40] Rupert Brooks, D. Louis Collins, Xavier Morandi, and Tal Arbel. Deformable ultrasound registration without reconstruction. In *Proceedings of the 11th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI'08)*, pages 1023–1031, New York, USA. 2008.

[6] Michel Audette, Rupert Brooks, Robert Funnell, Gero Strauss, and Tal Arbel. Piecewise affine initialized spline-based patient-specific registration of a high-resolution ear model for surgical guidance. In *MICCAI Workshop on Image Guidance and Computer Assistance for Soft-Tissue Interventions*, New York, USA. 2008.

## 1.4 Structure of the Thesis

Peer-review is essential to the scientific method, and publication in peer reviewed forums was an important focus throughout this research. As a result, this document is structured in a very modular fashion. Each chapter and even certain sections of chapters are relatively independent. In what is perhaps an uncommon structure for a thesis, I have located the experimental results in the text immediately following the theory they are designed to test. It is hoped that the reader will find this convenient, as it avoids cross-referencing between widely separated areas of the text.

Image registration is a large domain, and a review only focused narrowly on the work relevant to each contribution would not present a broad perspective to the reader. The following chapter reviews the domain of image registration from a high level, historical perspective. This big picture is then supplemented in each chapter with a focused review of the literature relevant to the particular contributions being discussed there. Despite the modular structure, nearly all the work presented was performed using a common set of algorithms, a common implementation and a common statistical testing framework. Chapter 3 describes these components that are common to all the subsequent chapters. An effort has also been made to use consistent mathematical notation throughout, and the reader may find the mathematical glossary in Appendix A helpful.

Recall that image registration can be made more efficient by either better optimization algorithms, which evaluate $D$ fewer times, by computing $D$ faster by using only some of the data, or by computing $D$ faster through precalculation of certain components. Chapter 4 is focused on the first option and discusses the choice of optimization algorithms for image registration problems. In order to make this choice, it is necessary to properly scale the transformation parameters, and, for Newton-Raphson optimizers, to properly compute approximate Hessian matrices. Chapter 4 also describes a generalization of the Gauss-Newton Hessian approximation and a method for determining optimal scaling parameters that are used in the remainder of the thesis. Chapter 5 addresses the second option by investigating the issue of how many pixels should be used and presents a framework for managing the trade-off between computation time and accuracy. Chapter 6 considers the popular idea of choosing pixels of high derivative, and shows that previous approaches have ne-

glected to consider the scale over which the derivative is valid. Chapter 7 is mainly focused on the precalculation and caching approach. It presents a generalization of the inverse compositional and ESM alignment algorithms which allows them to be used without requiring compositional update steps, or special parameterizations of the transformation. Finally, the overall conclusions of the thesis and suggestions for future investigation are described in Chapter 8.

# Chapter 2

# Overview of the Image Registration Field

Image registration is required for a very wide range of applications, in fields including, to name a few, photogrammetry, machine vision, and medical imaging. Contributions to image registration have flowed in from many different sources, and occasionally been forgotten and reinvented. This chapter is intended to define the direct parameterized registration problem and to give a high-level overview of the image registration field. It describes the context of development, and the connections between the various algorithms which this thesis builds upon. This background applies to all the subsequent chapters. In keeping with the modular nature of this thesis, each chapter then deals with a focused area of image registration, and contains a focused literature review relevant to that specific topic. This chapter stays rather conceptual while Chapter 3 deals with the technical background required.

In the following section, a formal definition for the image registration problem is given, together with a discussion of its ill-posed nature and methods of regularization. Section 2.2 provides a very high-level and historical overview of the field, including a classification of several different types of registration. This thesis is focused on efficient direct parameterized image registration. An overview of parametric direct methods is given in Section 2.3 and Section 2.4 discusses previous work that improves the efficiency of these methods. Section 2.5 provides a review and sets the stage for the chapters to come.

## 2.1 Definitions

Image alignment or registration is the problem of finding a mapping between the coordinate systems of two (or more) images. To discuss registration, we must inevitably discuss correspondence as a good registration is usually described as one that identifies meaningful correspondences between features in each image. In this thesis, registration will be defined as the problem that, given two images defined over *spaces* $\mathcal{U}, \mathcal{V}$ parameterized by some coordinate systems $\boldsymbol{u}, \boldsymbol{v}$, find the mapping, $\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{v})$. Correspondence, on the other hand, is defined as the problem that, given two *sets* of features or elements in each image, $\mathcal{P}, \mathcal{Q}$ find a bipartite graph $\mathcal{G} = \{\mathcal{P} \cup \mathcal{Q}, \mathcal{E}\}$ where $\mathcal{E}$ is a set of weighted edges between nodes in $\mathcal{P}, \mathcal{Q}$ representing correspondences. If the sets $\mathcal{P}, \mathcal{Q}$ are assigned coordinates in some spaces $\mathcal{U}, \mathcal{V}$, then, clearly, a registration of $\mathcal{U}$ and $\mathcal{V}$ will imply some correspondence, and a correspondence between $\mathcal{P}$ and $\mathcal{Q}$ implies some registration.

As mentioned in Chapter 1, image registration approaches are generally classified as either feature-based, or direct. Broadly speaking, the feature-based approaches first search the space of correspondences and then derive the mapping, while direct approaches to image alignment search the space of possible mappings itself.

In the standard direct registration approach, a pair of images are registered. One of the images, referred to as the fixed or reference image, $I_f$, is held fixed while the other, referred to as the moving or template image, $I_m$, is deformed by some transformation, $\mathbf{W}(\mathbf{x}, \boldsymbol{\phi})$ [145]. Here, $\boldsymbol{W}$ is a class of transformation functions (or warps) parameterized by a vector, $\boldsymbol{\phi}$. That transformation which minimizes the difference between the two images is considered to be the correct solution. This can be expressed as an optimization problem:

$$\boldsymbol{\phi}_{opt} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}} \left( D(I_f(\boldsymbol{X}), I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))) \right) \tag{2.1}$$

This formulation of the problem provides a general framework, illustrated in Figure 2.1, for addressing image registration problems. To implement a registration algorithm, one chooses an image difference measure, transformation, interpolator and an optimization algorithm, connects them together and runs the optimization [91, 109, 145].

Although it may seem trivial, it is worth taking a moment to clarify the definition of an image. An image, $I$, has a kind of dual nature in the literature. It is frequently considered to be a function of space, $I(\mathbf{x})$, where $\mathbf{x}$ is a coordinate in the space where the image is defined [145]. This definition allows us to consider the derivative of the image, and the value of the image function in between pixel locations (which in practice is found through interpolation). However, an image is also considered as a set of pixel values. If we take the derivative of a function of an image, such as the difference measure $D$, we consider this derivative to be with respect to each pixel in the image. The image at a set of pixel positions will be written as $\boldsymbol{I} = I(\mathbf{X})$, where $\mathbf{X}$ is the set of coordinates of each pixel in the image, and $\boldsymbol{I}$ is thus a vector with length equal to the number of pixels in the image. Note that throughout the thesis the word *pixel* will be used to denote elements of both two and three dimensional images.

It is also worth noting that the image difference measure is also frequently called a similarity measure. Optimization algorithms are conventionally discussed in terms of minimization, and to remain consistent throughout the thesis, the terms difference measure or cost function will be used, and smaller values will indicate better matching. However, the term "image distance" which is sometimes used, will be avoided. The difference measure does not have to be a distance in the technical sense. For example it may be negative or disobey the triangle inequality.

This pairwise expression of the problem (Equation 2.1) can be extended to deal with multiple images, in what is called groupwise registration. This can be done by extending the cost function to operate over multiple images (*e.g.,* [14, 27]), by combining multiple pairwise cost functions (*e.g.,* [172]), or conceivably both.

### 2.1.1 Ill-posedness and Regularization

In its most general form, that is when any transformation at all is allowed, the image registration problem is highly ill-posed due to non-uniqueness of the solution. If all possible transformations are allowed, then for any difference measure it is possible to create many perfect matches. Regularization is therefore necessary. Most commonly, the problem is implicitly regularized by restricting the type of transformation used, which restricts the space of allowable vector fields [49]. Frequently, this can be justified based on the physical reality of the problem. For example, when registering two

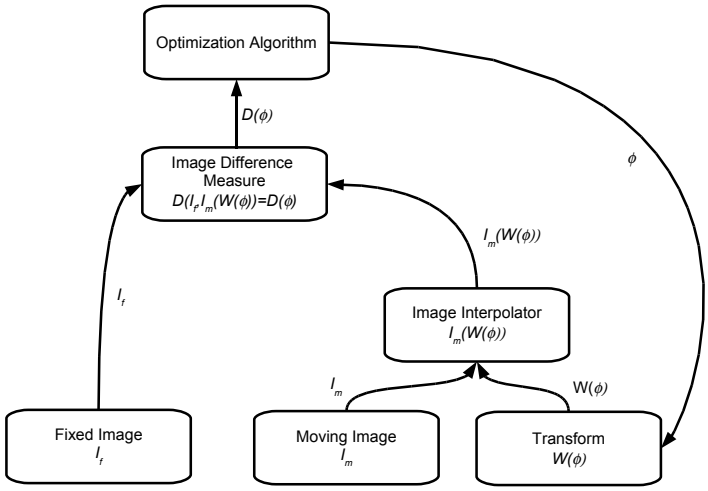



**Figure 2.1** A conceptual view of the standard, pair-wise registration approach. Registration consists of optimizing an image difference measure between a fixed image, and a warped (interpolated) moving image. The details of optimization algorithms, image difference measures and interpolators are reasonably independent and may be studied separately. Figure based on [109].

scanned volumes of a patient's head, it is reasonable to assume they are related by a rigid transform. In this thesis, only implicit regularization has been used.

When implicit regularization is not enough, there are three types of explicit regularization in common use [49]. Consistent image registration [52] formulates the problem bidirectionally. This technique penalizes deviations from perfect symmetry when switching the roles of the fixed and moving images and is particularly useful when dealing with warping functions that do not have a closed-form inverse. The optimization problem may be also be regularized by adding a competitive regularizer term to Equation 2.1. Common examples of this include the use of the elastic strain energy [8], or constraints on changes in volume [163]. Finally, the problem may be regularized incrementally, by penalizing optimization steps that do not conform to some criteria. The viscous fluid registration model [53] fits into this category. All of these

regularizations can be included to create an extended version of Equation 2.1 [49]:

$$\begin{aligned}\boldsymbol{\phi}_{opt} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}} \Big( & \lambda_{D_{i_f}} D_1(\boldsymbol{I}_f, \boldsymbol{I}_m, \boldsymbol{\phi}) + \lambda_{D_{i_r}} D_1(\boldsymbol{I}_m, \boldsymbol{I}_f, \boldsymbol{\phi}^{-1}) + ... \\ & + \lambda_{D_j} D_j(\boldsymbol{I}_1, \boldsymbol{I}_2, ..., \boldsymbol{I}_n, \boldsymbol{\phi}) + ... \\ & + \lambda_{R_{comp}} R_{comp}(\boldsymbol{\phi}) + ... \\ & + \lambda_{R_{incr}} R_{incr}(\boldsymbol{\phi}, t)\Big)\end{aligned} \tag{2.2}$$

where the $\lambda_{...}$ are weights, and each pairwise image difference measures, $D_i$, is computed both forwards, $D_{i_f}$ and backwards (*i.e.,* reversing the role of fixed and moving images), $D_{i_r}$. The $D_j$ represent difference measures defined over multiple images, and both competitive, $R_{comp}$, and incremental regularizers, $R_{incr}$, are included. The incremental regularizers are functions of the time step, $t$, as well as the images and transform.

While this thesis is focused on the pairwise image registration problem, using implicit regularization only, the applicability of this research is general. Observe that the general expression (Equation 2.2) still consists mainly of the optimization of image difference measures, the $D_i$. Thus speeding up image registration still requires either fewer or faster evaluations of these measures.

## 2.2 Previous Work: Overview

Numerous surveys of image registration exist. Some are broad in scope, and some are specific to a particular application. The books by Hajnal *et al.* [91], Modersitzki [145] and Goshtasby [89] give a good broad coverage of the subject, while the collection [168] is focused on non-parametric registration. The survey by Brown [41] gives a good overview of early work (although missing the work of Anuta [3, 4] and Hobrough [102]). The surveys [100, 138, 155] are focused on medical imaging applications, and the book by Toga [187] gives a very in depth discussion of registration exclusively on brain images. In contrast, the overview of direct methods by Irani and Anandan [110] is very focused on computer vision applications, as is the survey by Zitova and Flusser [208] which emphasizes feature based methods. Toga and Thompson [188] provide a discussion of registration in the specific context of creat-

ing neuroanatomical atlases. Szeliski's survey [181] is focused on the registration of images for stitching purposes – mainly panorama creation – but is nevertheless quite comprehensive.

It would would be very difficult, and probably of very limited use, to attempt to itemize every instance of image registration in the literature. Instead, this section will describe the field at a very broad level. It begins with a historical overview of the developments in the field, to explain how the framework of Figure 2.1 came to be in current use today. This is followed by a more focused discussion of work specifically related to improving the efficiency of this image registration framework. Of course, each of the contributions in the thesis derives from a specific chain of related previous work. Technical reviews of related work will be deferred until the chapters discussing each contribution.

### 2.2.1 Historical Development

#### Early Work

In 1957, Gilbert Hobrough demonstrated a remarkable machine to his colleagues which was capable of performing automatic photogrammetric stereoplotting [47]. While previous devices could register photographs along one axis or another, this was the first device capable of automatically registering images in two dimensions. This was achieved using entirely analog electronics through a clever use of a random scanning pattern [101, 102]. The achievement was of great practical significance because at that time nearly all geographic mapping was created by a human operator manually drawing the terrain contours while stereo viewing the aerial photos. By 1965 [103], a machine had been developed capable of refining a ten parameter distortion model using only analog means.

The idea reappeared independently, this time using digital images on a digital computer, in Anuta's 1969 paper [3]. This describes direct image alignment by searching for a maximum of correlation in the space of possible translations. The procedure used an exhaustive search of the transformation space, which makes it reminiscent of template matching approaches. This was followed by a flurry of rather remarkable papers which set forth many of the broad divisions in the field that continue to exist today.

The differentiation between direct and feature based approaches to registration appeared around this time and development of both proceeded side by side. For example, some of the earliest reported work in recovering vector fields of deformation from image sequences was done to recover cloud motion for meteorological analysis. In 1971, Leese *et al.* [128] proposed a block matching, cross-correlation approach while Endlich *et al.* [73] proposed a feature based technique side by side in the very same journal issue. Direct methods are most appropriate for multimodal image registration problems involving curved, but diffeomorphic, transformations. They are thus a much closer fit to the medical imaging problems that originally motivated this research. This thesis, and the rest of this discussion, is therefore focused exclusively on direct image registration methods.

Besides the main distinction in the field of registration between direct and feature based approaches, direct registration itself can be subdivided into four different types of approaches: iconic methods, Fourier based methods, non-parametric or optical flow methods, and – the subject of this thesis – parametric registration methods.

### Types of Direct Image Registration

Iconic methods lie in some sense in-between the feature based techniques and the direct techniques. The term *iconic*, coined in [49], includes a number of block or patched based registration methods. These model complex transformations by solving for a collection of local, simple transformations which are then smoothed together somehow, often with other constraints being included. These works tend to take the form of end to end algorithms, intended to function as black-boxes for image registration. Collins *et al.* [58] developed the ANIMAL algorithm which models complex deformations as locally simple transformations of blocks. Shen and Davatzikos developed the HAMMER algorithm [170]. This method is difficult to classify as it has many of the characteristics of feature based methods, such as searching for item to item correspondence, but also does some pixel based computations. The PASCHA algorithm proposed by Cachier *et al.* [49] attempts to include elements of all the different image registration methods discussed (feature-based, consistent, competitive and incremental regularizers). All three of these examples have been developed in the context of medical imaging. These methods can be related back to the central theme

of this thesis by noting that they can be viewed as a collection of small local direct parametric registrations. The work presented here could be applied to these methods on a component by component basis.

Fourier based methods were developed as a faster way to perform image registration by performing the search in the Fourier space [4]. A more sophisticated method using phase correlation was proposed by Kuglin and Hines somewhat later [122]. When applicable, Fourier based methods can lead to rapid, one-step solutions of the registration problem. However, they require that a simple mathematical relationship exist between the Fourier transform of the current and transformed image. This is not generally true for the transforms of interest in this thesis, and so their development is not pursued further here.

Optical flow is the 2D vector field created by the projection of 3D motion through a camera, or other imaging device. The concept has its roots in the psychology of motion perception discussed by Gibson [84]. However, the optical flow vector field can also be viewed as a transformation that registers two images, and optical flow techniques have played an important part in the development of image registration. In 1981, Horn and Schunk [105] presented the first method for determining optical flow in a computer vision context. Numerous methods followed, which have been compared in the extensive review [17]. The majority of the optical flow methods directly generated vector flow fields regularized with some sort of smoothness constraint. The key point here is that the transforms are not parameterized. Currently, the terms *nonparametric image registration techniques* and *optical flow techniques* are frequently used interchangeably (*e.g.,* [97, 199]), although there is no optical flow, as such, when performing intersubject MRI registration.

Parametric methods are distinguished by using a parameterized functional mapping to express the relationship between the coordinate systems of the two images being registered. This idea was originally proposed by Barrow *et al.* [18], but their work is somewhat neglected, and the origin of parametric methods is usually ascribed to Lucas and Kanade's 1981 registration paper [136]. As parametric methods are the main focus of this thesis, they will be reviewed in more detail in the following section.

## 2.3 Parametric Image Registration Methods

There were two key elements in Lucas and Kanade's [136] paper. One was the idea of using arbitrary parameterized functional transformations, such as rotations, shears, or homographies, between the spaces of the images. Lucas and Kanade [136] also used a least squares cost function, and optimized it using an iterative Gauss-Newton optimization method. This approach was to become one of the canonical approaches to image registration in the computer vision field, being repeatedly used by many other authors (*e.g.,* [12, 24, 182]). All of the work presented in this thesis is influenced by the Lucas-Kanade method. In fact, the efficient inverse compositional method [11, 12], discussed at length in Chapter 7, was published 20 years after [136] and is entitled "Lucas-Kanade 20 Years On" to celebrate the great influence this approach has had.

The Lucas-Kanade method [136] can be described quite well in terms of the framework shown in Figure 2.1. The image difference measure used was mean squared difference, the optimizer was of a Gauss-Newton type, and the transformations used were matrix transforms, such as rigid and affine transforms. Key elements that remained to be developed were an image difference measure that was robust to significant variations in the character of the images, and the use of more sophisticated transformations, that could model image changes more complex than those caused by, for example, looking at a planar object from a different camera angle.

### 2.3.1 Development of the Mutual Information Measure

Early image registration work primarily used correlation [3, 102], mean absolute difference [15], and mean squared difference [136] measures to quantify the goodness of a match between images. The observation that correlation based measures could be improved by normalization [157] lead to the normalized correlation measure that is still widely used. However, particularly in medical applications, the images to be registered have such different natures that correlation based measures cannot effectively measure the quality of the registration. For example, the similarities between PET (Positron Emission Tomography) and MR (Magnetic Resonance) images cannot be captured by correlation alone. Various information theoretic criteria were proposed as cost functions for registration by several authors [57, 99, 180, 205] in the early

1990s. In 1995, mutual information (MI) was proposed independently by [56, 194]. MI captures in a very general way the idea that there is some relationship between the structures in each image, and it proved very effective for registering images of varying characters. Applications of direct image registration using MI quickly became widespread. By 2003 a survey by Pluim *et al.* [155] listed over 90 reported studies of mutual information registration of medical images.

### 2.3.2 Development of Parameterized Deformable Transformations

Certain kinds of image registration problems, such as the change in appearance of a planar object when viewed from different angles could be addressed with fairly simple matrix based or low-order polynomial transformations. However, the complex transformations of highly deformable objects such as the brain motivated many different techniques to model them. It was recognized early on that the deformable image registration problem was highly ill-posed, and various deformation models were proposed to regularize it. Widrow [203] proposed to use a "rubber-sheet" transformation, a clear precursor of elastically regularized transforms, for the registration of chromosome images, although he did not propose a truly automatic method to achieve the registration. Fischler and Elschlager [78] used a model consisting of points connected by virtual springs to match a canonical face to images of faces and models of terrain to images of terrain. They proposed a matching algorithm, but the images had to be broken down descriptively by hand first. It was not until the paper by Burr [45, 46] that an end-to-end system for deformably registering images was developed, using an iterative solution method. Bajcsy and Broit [8] were the first to explicitly use elasticity theory to define a regularizer; they also applied the matching process to 3D volumes, a difficult task given the limited computer power of the time.

Spline based transforms were also introduced to model deformations of an image that could not be represented by one of the matrix group transforms. Goshtasby proposed piecewise linear [87] and cubic [88] mappings. Evans *et al.* [75] generalized the thin plate spline [31, 94] originally used for modeling aircraft wings to model deformations in three dimensions. Szeliski and Couglan [183] developed a series of multiresolution splines that were also applied to medical image registration [127]. In 1999, Rueckert *et al.* [164] proposed a multi-resolution deformation model based on

the tensor product of B-splines defined over a control point grid which has become very popular in medical imaging (*e.g.,* [124, 133, 142, 163]). Both the thin plate spline, and the B-spline deformation models are used extensively in Chapter 4.

The emphasis on deformable transformations was greatest in the medical imaging field. This was because in medical imaging, the mapping between images can usually be treated as diffeomorphic[1]. In the computer vision field, complex transformations were usually caused by shifts in camera position that involved occlusions and and other non-closed form transformations. This, coupled with the development of very reliable feature based approaches, *e.g.,* the scale invariant feature transform (SIFT) [134, 135], lead the emphasis in computer vision toward feature based methods. However, there has recently been a resurgence of interest in parametric image registration techniques that can address the special characteristics of images taken by a camera, from two different viewpoints [21, 113]. Techniques have also been proposed which can process self-occlusions which may occur when imaging a deformable object with a camera [83]. These advances point to a promising future for direct image registration approaches in computer vision.

## 2.4 Efficient Parametric Registration Techniques

At the current time, the direct, parameterized image registration approaches have evolved into a widely used framework (shown in Figure 2.1) for image registration. Recall from Chapter 1 that the primary computational cost of this framework is the evaluation of the image difference measure, $D$. The optimization component of Figure 2.1 controls the number of times $D$ will be evaluated, while the remainder of the framework mainly impact the speed with which $D$ can be evaluated. The issue of efficiency within this framework has been addressed through specific implementations of efficient components, through parallel processing, through multiresolution approaches, through the selection and use of subsets of pixels, and through the inverse compositional and related methods.

[1] A mapping is diffeomorphic if it is differentiable and has a well defined inverse [201].

### 2.4.1 Efficient Components

The modularity of the direct parameterized registration framework has encouraged the development of computationally efficient approaches for each of its components. Several efficient components have been based on the B-spline framework for signal processing proposed by Unser *et al.* [189]. In [189] they proposed to use B-spline basis functions for nearly all levels of efficient signal processing systems. This proposal was in fact successfully achieved. The B-spline freeform deformations proposed by Rueckert *et al.* [164] are explicitly designed for efficiency, presumably to address some of the significant computational inefficiencies in landmark based spline models. In this work, the deformation is modeled using B-spline kernels defined on a grid of control points. Because the kernels are compactly supported and separable, their implementation is computationally efficient. Thévenaz and Unser [185] proposed a efficient means of calculating mutual information using a B-spline Parzen windowed joint histogram. This method is combined with the B-spline freeform deformations of Rueckert *et al.* [164] and B-spline interpolation [189] in the multimodal deformable registration by Mattes *et al.* [142]. In this thesis, the B-spline deformation model [164] and Thévenaz and Unser's mutual information [185] are among the techniques used.

Optimizers for image registration can be approached using either a local, or global type optimizer. Local optimizers are more commonly used [91, 181, 208], but can suffer from being trapped in local minima if they are not initialized close enough to the final solution. Global optimizers will always converge to the global minimum of the image difference measure, but they can be orders of magnitude slower than a local approach. The efficiency of global optimization methods has been addressed by Chen *et al.* [51] who proposed a tunneling search method for efficient global optimization, and Wachowiak and Peters [197], who used parallelization of direct search approaches to make global optimization more efficient. However, in this thesis, local optimization approaches have been used exclusively, which corresponds to the assumption that a plausible starting estimate is available. The efficiency of local optimizers in a registration context has mainly been addressed through empirical studies such as those by Bernon *et al.* [25, 26], Maes *et al.* [137] and Klein *et al.* [120]. These studies have focused on particular registration problems and it is not always clear how to generalize their results. Chapter 4 of this thesis addresses this issue by developing

principles for optimizer selection which are then verified through experiment. The Efficient Second order Method (ESM) [22, 139, 140], which is discussed at greater length in Chapter 7, is an example of a specific local method that exploits special structure within the image registration problem to create a better approximation to the Hessian matrix of second derivatives at each step. This better approximation allows more efficient and reliable optimization.

The efficiency of the interpolation method has also been addressed. As mentioned, Unser *et al.* [189] have proposed B-spline based methods for accurate and efficient image interpolation, including the representation of an image entirely as a B-spline scale space. Čapek and Poušek [191] discuss using integer rather than floating point operations for faster volume resampling. Finally, although the work of Salvado and Wilson [167] is focused on removing interpolation artifacts rather than speed, they also do address the efficiency of interpolation methods.

### 2.4.2 Parallel Implementations

Several authors have addressed the efficiency of image registration by proposing parallel implementations. Wachowiak and Peters [196, 197] used parallelized direct search methods for rigid 3D volume registration. Rohlfing and Maurer [162] developed a shared memory parallel implementation of a B-spline deformable registration framework. Through the use of a large number of processors (64 to 128) they were able to achieve significant speedups. Several authors have also implemented both parametric and nonparametric image registration algorithms on graphics processing units (GPU)s (*e.g.,* [121, 166]).

These methods achieve efficiency by spreading the calculation across multiple processors. It could in some way be considered that rather than being more efficient, these methods are bringing more resources to bear on the same problem. In this thesis, parallelization has not been considered, but the approaches discussed here could easily be generalized to a parallel environment.

### 2.4.3 Multiresolution Approaches

Coarse to fine, or multiresolution, approaches were being applied to deal with the problems of speed and local minima in the search as early as 1973 [147]. During the

1980s, numerous authors [9, 67, 74] were applying multiresolution approaches to the registration problem in more or less ad-hoc ways for both computational efficiency and to avoid local minima (see also early work in stereo [147]). Bergen *et al.* [24] formalized this approach and applied it to parametric registration to produce a framework for multiresolution image registration that is now considered a standard method. In this approach, the images to be registered are reduced in size in a (usually Gaussian) scale space pyramid. Optimization is started on the top level of the pyramid. As each optimization stage completes, the results from that stage are used to initialize the next, higher resolution stage. The approach used in this thesis (see Section 3.1.12) is closely related to that discussed in [24].

### 2.4.4 Pixel Selection

As the method of [3] was relatively slow, several methods for speeding up direct image registration and/or template matching were soon proposed. Barnea and Silverman proposed the Sequential Similarity Detection Algorithms (SSDA) in [15] and Nagel and Rosenfeld [148] proposed a method for selecting optimal pixels in template matching applications. These methods are the very early precursors of the work presented in Chapters 5 and 6. The idea of using only some of the pixels in the image became fairly widespread, but formal analysis of exactly how many or which ones was not immediately addressed.

In the field of computer vision, *tracking* is the process of following an object through a time series of images. *Template based tracking* [29, 90] addresses this problem by starting with a template image of the object to be followed, and then registering this template to subsequent frames in the image series. As tracking an object is obviously a time sensitive matter, template based tracking became a driver for a considerable amount of work in efficient image registration. Dellaert and Collins [66] proposed to speed up registration for the tracking problem by using only a small set of pixels that best reduce the uncertainty in the resolved transformation parameters. A recent, alternative approach was proposed by Benhimane [23] who determine which pixels best fit a low order Taylor series approximation to the image. It is implied that these pixels will then be optimal for reducing the uncertainty in the transformation parameters, as they best agree with the optimization model used. In Chapter 6 the

framework of Dellaert and Collins [66] is examined, and it is shown that the issue of scale has been ignored in the role of the derivative used for pixel selection. Unless taken into account, this omission can lead to performance degradation.

Tracking was also one of the main motivations for the development of the last category of efficient approaches, the inverse compositional algorithm and related methods.

### 2.4.5 Inverse Compositional Methods

The image registration problem has two special characteristics which can be exploited to gain efficiencies. There is a symmetry between the roles of the fixed and the moving images, and the transformations parameters could be optimized compositionally instead of additively. Hager and Belhumeur [90] first exploited the symmetry between the fixed and moving images to develop a fast template based tracking approach. Shum and Szeliski [172] exploited the compositional aspect to increase efficiency in a panoramic mosaicking application. Baker and Matthews [11, 12] identified and classified all the ways that these special characteristics could be used and proposed the inverse compositional registration algorithm. This algorithm achieves efficiency by estimating an update step for the warp of the fixed image, and composing the inverse of that step with the current moving image warp.

The previously mentioned ESM algorithm [22, 139, 140] was developed independently for tracking and visual servoing. It is shown in this thesis that it is closely related to the inverse compositional algorithm. Where a classical approach computes an update step for the moving image warp parameters, and the inverse compositional approach computes one for the fixed image warp parameters, the ESM approach computes *both* and combines them. In this way it is closely related to the work of Keller and Averbuch [114], although the ESM method addresses certain mathematical issues that Keller and Averbuch ignored.

Current implementations of both the IC and ESM algorithms suffer from certain restrictions. For one, the optimizer step must be made compositionally in the transformation space. As most common optimization algorithms are defined with additive update steps, this complicates the implementation of these methods. Furthermore, the ESM method is also restricted to operate on transformations parameterized using the exponential map of their Lie group, which may not always be practical or avail-

able. Chapter 7 shows how the ESM and IC algorithms are related, and how they may be generalized to remove these restrictions while maintaining their advantages.

## 2.5 Summary

Registration is a problem with a wide variety of applications and the development of current techniques spans several decades. Current approaches to image registration can first be classified into direct and feature based methods, and the direct methods can be further subdivided into Fourier-based, non-parametric and parametric approaches. It is the direct, parametric image registration approaches that are of interest in this thesis. These approaches can be viewed as being made from four interconnected components (Figure 2.1): An optimization algorithm, an image difference measure, a transformation and an interpolation technique. The basic form of this model took shape in the papers of Barrow *et al.* [18], and especially Lucas and Kanade [136].

Since that time each of the components has increased in sophistication. Particularly important developments related to the implementation framework described in this thesis are the development of multiresolution approaches [24], the use of mutual information as a cost function [56, 194], the use of the thin plate spline [31, 75, 94] and the B-spline parametric deformation models [142, 164].

This research is focused on improving the efficiency of image registration. Of particular relevance to the work presented here are the studies of optimization efficiency [25, 26, 120, 137], methods that propose to use only some of the pixels [15, 23, 66, 148], and the inverse compositional methods [11, 12, 22, 90, 139, 140, 172]. Each of these will be discussed in greater depth and related to the current work in the corresponding chapters. In each case general, deeper principles unifying the methods in question is sought, and its effectiveness is then verified by experiment. Therefore, before the original contributions of this thesis can be described, it is necessary to discuss the experimental framework in which the research was conducted. The following chapter provides technical discussion of the image registration approach which is common to all the subsequent chapters.

# Chapter 3

# Experimental Framework

This thesis proposes a number of specific methods for improving the performance of image registration algorithms. To test these propositions, experiments were performed which consist of running sets of image registrations using the old and the proposed algorithms. As shown in Figure 2.1 the direct, parametric image registration process can be viewed as having four components: the image difference measure, the transformation, the interpolation method, and the optimization algorithm. The work in this thesis has focused on aspects of the difference measure and the optimization, while using standard approaches for the other aspects. For the most part, the experiments in Chapters 4–7 were conducted using a common implementation, evaluation criteria, and statistical testing framework. To avoid repetition, these common elements will be described together in this chapter.

In the first section, the technical aspects of each of the components used in the experiments is described. Once the experiments were performed it was necessary to evaluate them. In each case, the registration results are compared in terms of speed, accuracy and reliability. Section 3.2.1 describes how the output of the different registration runs was compared and the statistical tests that were used to verify whether the observed results could legitimately be used to draw conclusions.

## 3.1 Image Registration Framework

All the work presented in this thesis has been implemented by using and extending the Insight Toolkit [109]. This toolkit was developed to support medical imaging research and supplies well-tested implementations of registration algorithms. In research of this sort, there is always the risk of researcher bias, in that more time is usually spent crafting the newer algorithm being proposed than is spent on the old baseline algorithm being compared to. Inadvertent bugs in the baseline algorithm can lead to artificially good results. Using this toolkit has the advantage that the baseline implementation is well-tested, and implemented independently from this research, which should minimize this problem. Most importantly, it also enhances reproducibility of the results, as the same framework is available to other researchers. The following sections describe each of the components of the registration system in turn: the image difference measures, the interpolation method, the transformations, and the optimization algorithms.

### 3.1.1 Image Difference Measures

Three of the most widely used image difference measures were used for the experiments in this thesis. The normalized correlation (NCC) measure [3, 102] was the first measure used for image registration. It is invariant to linear changes in intensity between the two images under test. The mean squared difference (MSD) measure [136] expresses the image registration problem as a least squares problem. Using this measure implies that the two image intensities are expected to be identical when they are correctly registered. It is particularly appropriate when the images come from the same instrument, under similar conditions. The mutual information (MI) measure [56, 194] is the most recent and general of the three. It measures the existence of some relationship, however complex, between the intensities of each image.

Roche *et al.* [161] have presented an elegant interpretation of all three of these measures in terms of the joint histogram of the intensities of the two images. Mean squared difference corresponds to the expectation that the joint histogram should contain a line of slope exactly one at correct registration. Using the normalized correlation difference measure corresponds to the expectation that the joint histogram

should contain a line, but the slope and intercept may be arbitrary. Finally, mutual information merely expects that the entropy of the joint histogram should be lowest at correct registration. That is, that as much as possible of the variation in the intensities in one image can be predicted by the variations in the other at corresponding points.

When images are in correct alignment, it is expected that the squared difference should be low, and the correlation and mutual information should be high. For consistency, all measures will be defined so that low numbers indicate a good match, and therefore all optimization problems will be minimization problems. Thus the negative normalized correlation, and the negative mutual information are the measures that are really used. For brevity, however, the word negative may be omitted in much of the discussion. All the equations and definitions, will be constructed so that lower numbers indicate better registration.

In order to perform optimization, we must be able to compute each measure, its derivative, and occasionally its Hessian. In this section we discuss the computation of each measure and its derivative, as these are common to all experiments in this thesis. The calculation of approximate Hessians was specifically examined as part of this thesis, so that discussion will be deferred to Chapter 4.

In general, the image difference measure may be viewed as a function of the warped moving image,

$$D(\boldsymbol{\phi}) = D(I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi})))$$

Its derivative with respect to the transformation parameters then takes the form

$$\frac{\partial D}{\partial \boldsymbol{\phi}} = \sum_i \frac{\partial D}{\partial \boldsymbol{I}_{m_i}} \frac{\partial \boldsymbol{I}_{m_i}}{\partial \boldsymbol{W}} \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{\phi}}$$

where the summation over $i$ runs over all the pixel positions of the fixed image. Here $\frac{\partial D}{\partial \boldsymbol{I}_{m_i}}$ is the derivative of the image measure with respect to each pixel, $i$, which is specific to each measure. The $\frac{\partial \boldsymbol{I}_{m_i}}{\partial \boldsymbol{W}}$ is the derivative of the moving image at the warped position of the point, $\boldsymbol{X}_i$, and $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{\phi}}$ is the Jacobian matrix of the warp at $\boldsymbol{X}_i$.

### 3.1.2 Mean Squared Difference

The mean squared difference measure, $D_{MSD}$, is simply an average of the squared differences in intensity between corresponding pixels.

$$D_{MSD}(\boldsymbol{\phi}) = \frac{1}{N}\sum_{i=1}^{N}(I_f(\boldsymbol{X}_i) - I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}))^2 \tag{3.1}$$

where $N$ is the number of pixels in the image. The gradient of this measure is easy to compute:

$$\begin{aligned}\nabla_{\boldsymbol{\phi}} D_{MSD}(\boldsymbol{\phi}) = \frac{2}{N}\sum_{i=1}^{N}&\left[(I_f(\boldsymbol{X}_i) - I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}))\right)\\ &\cdot\nabla_{\boldsymbol{W}} I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}))\nabla_{\boldsymbol{\phi}}\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})]\end{aligned} \tag{3.2}$$

### 3.1.3 Normalized Correlation

Normalized correlation is appropriate for comparing images where there is a linear relationship between the intensities in the fixed and moving images, such as photographs under different lighting conditions or satellite images in different spectral bands. Note that to remain consistent with our discussion of minimization, we define our cost function, $D_{NCC}$, as the negative of normalized correlation. The $D_{NCC}$ measure is a ratio of two terms

$$D_{NCC} = \frac{-u}{v} \tag{3.3}$$

where

$$u = \sum_i \left[(\boldsymbol{I}_{f_i} - \bar{I}_f)(\boldsymbol{I}_{m_i} - \bar{I}_m)\right],$$

and

$$v = \left[\sum_i \left[(\boldsymbol{I}_{f_i} - \bar{I}_f)^2\right]\sum_j \left[(\boldsymbol{I}_{m_j} - \bar{I}_m)^2\right]\right]^{\frac{1}{2}}.$$

where $\bar{I}_f$ and $\bar{I}_m$ are the mean intensities for the fixed and moving images, respectively. The gradient is then

$$\frac{\partial D_{NCC}}{\partial \boldsymbol{\phi}} = \frac{1}{v^2}\left(u\frac{\partial v}{\partial \boldsymbol{I}_m} - v\frac{\partial u}{\partial \boldsymbol{I}_m}\right)\frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}$$

Assuming that the mean values of the image, $\bar{I}_m$, do not appreciably change as the parameters change, the derivatives of $u$ and $v$ with respect to a particular pixel in the moving image, $I_{m_i}$, are:

$$\frac{\partial u}{\partial \boldsymbol{I}_{m_i}} = (\boldsymbol{I}_{f_i} - \bar{I}_f),$$

and,

$$\frac{\partial v}{\partial \boldsymbol{I}_{m_i}} = \frac{1}{v}\sum_k \left[(\boldsymbol{I}_{f_k} - \bar{I}_f)^2\right](\boldsymbol{I}_{m_i} - \bar{I}_m),$$

so the complete gradient of the $D_{NCC}$ measure is:

$$\begin{aligned}\nabla_{\boldsymbol{\phi}_m} D_{NCC} = \frac{-1}{v}\cdot\sum_i \left[(\boldsymbol{I}_{f_i} - \bar{I}_f)\cdot\nabla_{\boldsymbol{\phi}_m}\boldsymbol{I}_{m_i}\right] \\ +\frac{u}{v^3}\cdot\sum_k \left[(\boldsymbol{I}_{f_k} - \bar{I}_f)^2\right]\cdot\sum_j \left[(\boldsymbol{I}_{m_j} - \bar{I}_m)\cdot\nabla_{\boldsymbol{\phi}_m}\boldsymbol{I}_{m_j}\right]\end{aligned} \tag{3.4}$$

### 3.1.4 Mutual Information

The mutual information (MI) image similarity measure [56, 194] is useful for images of different modalities. The implementation used in this thesis is based on the efficient mutual information implementation proposed by Thévenaz and Unser [185], which relies on a B-spline Parzen windowed representation of the joint probability distribution of the intensity levels in the two images. To understand the implementation, it is helpful to first consider how the joint distribution could be directly computed, without Parzen windowing. Specifically, let $b_{f_k}$, where $k = 1 \ldots K$, be a set of $K$ bins of width $d_f$ for the intensity values in the fixed image starting at $b_{f_0} = \min_{\mathbf{x}} I_f(\mathbf{x})$. Similarly, let $b_{m_l}$ be the bins for the intensity values in the moving image, where $l = 1 \ldots L$, the bins begin at $b_{m_0} = \min_{\mathbf{x}} I_m(\mathbf{x})$ and have width $d_m$. Then the joint distribution, $\boldsymbol{P}$, is an array of size $K \times L$. The entry $\boldsymbol{P}_{kl}$ is equal to the number of pixels in $\boldsymbol{I}_f$ for which the intensity falls in bin $k$ and the intensity of the corresponding pixels in the

transformed image falls in bin $l$, normalized by the total number of pixels, $N$:

$$\boldsymbol{P}_{kl}(\boldsymbol{\phi}) = \sum_{i=1}^{N} \delta\left(k, \left\lceil \frac{I_f(\boldsymbol{X}_i) - b_{f_0}}{d_f} \right\rceil\right) \delta\left(l, \left\lceil \frac{I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})) - b_{m_0}}{d_m} \right\rceil\right).$$

Here $\delta$ is an indicator function, equal to 1 if its two arguments are equal, and zero otherwise. Thévenaz and Unser [185] use instead a soft version to compute the entries in the table, based on B-spline Parzen windows, so that the joint distribution becomes:

$$\boldsymbol{P}_{kl}(\boldsymbol{\phi}) = \sum_{i=1}^{N} \frac{1}{N} \beta_0\left(k - \frac{I_f(\boldsymbol{X}_i) - b_{f_0}}{d_f}\right) \beta_3\left(l - \frac{I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})) - b_{m_0}}{d_m}\right) \tag{3.5}$$

where $\beta_0$ and $\beta_3$ are 0th and 3rd order B-spline Parzen windows respectively. The joint distribution can simply be normalized by dividing by the number of pixels because the B-spline Parzen windows satisfy the *partition of unity* constraint. That is, their contributions to the table for each pixel will always sum to one, regardless of the pixel value (see [185, p. 2085] for details). The mutual information can then be computed using the usual formula [61]:

$$D_{MI}(\boldsymbol{\phi}) = \sum_{k=1}^{K} \sum_{l=1}^{L} \boldsymbol{P}_{kl}(\boldsymbol{\phi}) \log \frac{\boldsymbol{P}_{kl}(\boldsymbol{\phi})}{\left(\sum_{k'} \boldsymbol{P}_{k'l}(\boldsymbol{\phi})\right)\left(\sum_{l'} \boldsymbol{P}_{kl'}(\boldsymbol{\phi})\right)} \tag{3.6}$$

Note that the second factor in the denominator above is just the intensity histogram of the fixed image, which is computed only once, before the optimization process.

Because a 3rd order B-spline is differentiable, the gradient of the joint histogram, $\nabla_{\boldsymbol{\phi}} \boldsymbol{P}_{kl}(\boldsymbol{\phi})$, can be computed and stored in $\sharp(\boldsymbol{\phi})$ tables, each of dimension $K \times L$. Specifically, the elements of these tables are given by

$$\nabla_{\boldsymbol{\phi}} \boldsymbol{P}_{kl}(\boldsymbol{\phi}) = \sum_{i=1}^{N} \frac{1}{d_m N} \frac{\partial \beta_3\left(l - \frac{I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})) - b_{m_0}}{d_m}\right)}{\partial \boldsymbol{I}_{m_i}} \cdot \frac{\partial \boldsymbol{I}_{m_i}}{\partial \boldsymbol{\phi}}. \tag{3.7}$$

(The specifics of B-splines, $\beta_0(x), \beta_1(x), ...$, and their derivatives $\frac{\partial \beta_0(x)}{\partial x}, \frac{\partial \beta_1(x)}{\partial x}, ...$ can be found in [189].) In [185] it is shown that when using the above formulation the

derivative of $D_{MI}$ is:

$$\nabla_{\boldsymbol{\phi}} D_{MI}(\boldsymbol{\phi}) = \sum_{k}^{K} \sum_{l}^{L} \nabla_{\boldsymbol{\phi}} \boldsymbol{P}_{kl}(\boldsymbol{\phi}) \log \frac{\boldsymbol{P}_{kl}(\boldsymbol{\phi})}{\sum_{k'}^{K} \boldsymbol{P}_{k'l}(\boldsymbol{\phi})} \tag{3.8}$$

In a practical implementation, calculation of the MI measure and its gradient proceeds by first accumulating the joint distribution, $\boldsymbol{P}$, and its derivatives, $\nabla_{\boldsymbol{\phi}} \boldsymbol{P}$, in a loop over all the pixels. Once accumulated, these can be normalized, and then converted into the actual MI and its gradient using Equations 3.6 and 3.8.

### 3.1.5 Interpolation

The interpolator is a critical part of the image registration process. Except in very unusual circumstances, the points in the fixed image are not mapped directly onto lattice points in the moving image by the transform. To come up with values for positions off the sampling lattice, interpolation must be used. All interpolation methods face a tradeoff between computation time, and accuracy. All experiments in this thesis have used linear interpolation exclusively in the registration process. This was chosen for performance reasons, as it is much faster to compute than the alternatives.

It has been pointed out by several authors [153, 167], however, that linear interpolation can lead to artifacts in the cost function that can interfere with the registration process. Upon investigation, it was found that these artifacts occur primarily when the transformation consists of a translational shift, which causes the sampling points to consistently go in and out of alignment with respect to one another. For transforms that are not exclusively translational, these artifacts are quite minor. Figure 3.1 shows an example. The mean squared difference cost function has been plotted between an image and a transformed version of itself. (The image used is shown in Figure 4.1.) Only in the case of a pure translational shift do the interpolation artifacts appear.

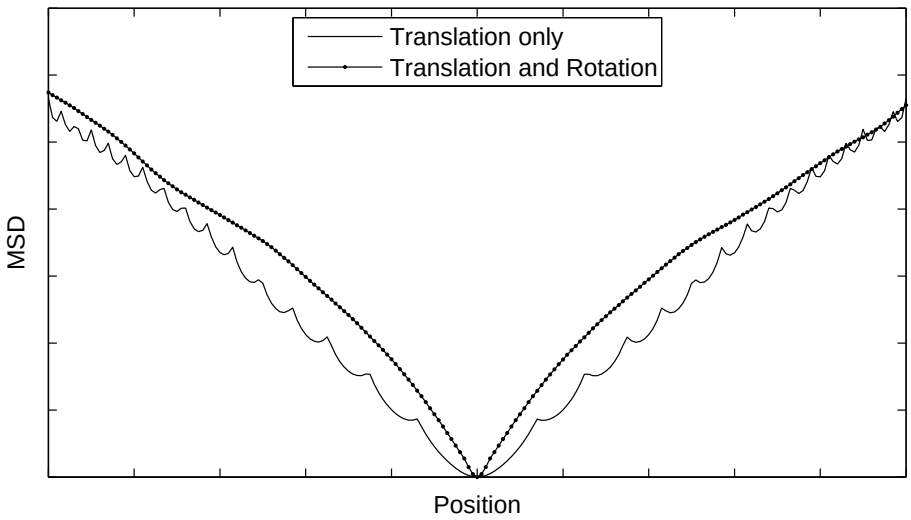



**Figure 3.1** Mean squared difference cost function of an image compared to a transformed version of itself using linear interpolation. The plain line shows the case where the moving image is transformed using translation only, while the dotted line shows the case of a transformation involving both angular and translational components. Interpolation artifacts are clearly visible in the translational case, but when there is some angular shift involved these disappear. The translation ranges from $\pm 16$ pixels in both x and y directions. For the dotted line, the angle also varies over $\pm 0.28$ radians.

Interpolation also affects the calculation of the image gradient. Computing the gradient of the cost function requires the image gradient to be computed at a point in the moving image which does not in general fall on the sampling lattice. There are two approaches. One is to compute the gradient from the image every time it is needed, the other is to compute the gradient of the moving image once, and interpolate it when it is needed. The latter approach has been chosen, using a nearest neighbor interpolation. This is the default method used in the ITK (Insight Tool Kit) library [109]. It is much more efficient than recomputing the gradient at every step, and our experience shows that it is equally effective.

### 3.1.6 Transformations

The transformation, or warp, defines the mapping from the coordinate system of the fixed image to the coordinate system of the moving image. The majority of the work

presented here (Chapters 5, 6, and 7) is implemented on matrix-based transformations. The work on optimizer selection and scaling (Chapter 4) is extended to both matrix based and spline based transformations. The matrix based transforms correspond roughly to what are sometimes called the linear transforms, while the spline based transforms may also be described as curved or deformable transforms.

Transformations are defined as families of functions mapping from the space of one image into the space of another, $W(x, \boldsymbol{\phi})$, where the family of functions is parameterized by $\boldsymbol{\phi}$. Being functions, they conceptually support the operations of composition, and inversion. The composition operation

$$\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi}_C) = \boldsymbol{W}(\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi}_A), \boldsymbol{\phi}_B)$$

will be represented with a $\circ$. For brevity, a slight abuse of notation will be used to write:

$$\boldsymbol{\phi}_C = \boldsymbol{\phi}_B \circ \boldsymbol{\phi}_A$$

to represent the composition of two transforms, *i.e.,*

$$\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi}_C) = \boldsymbol{W}(\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi}_A), \boldsymbol{\phi}_B)$$

Note that, in general, composition is not commutative, *i.e.,* $\boldsymbol{\phi}_B \circ \boldsymbol{\phi}_A \neq \boldsymbol{\phi}_A \circ \boldsymbol{\phi}_B$. In a similar manner to composition, $\boldsymbol{\phi}^{-1}$ will be used to refer to the inverse of a transform, *i.e.,*

$$\boldsymbol{x} = \boldsymbol{W}(\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi}_A), \boldsymbol{\phi}_A^{-1})$$

### 3.1.7 Matrix Transformations

There are a number of widely used transformations that can be represented as the action of a matrix group [10] on the space of homogeneous coordinates. That is

$$\begin{bmatrix} \hat{\boldsymbol{x}} \\ \lambda \end{bmatrix} = \boldsymbol{M} \cdot \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} \tag{3.9}$$

| Transformation | Description |
|---|---|
| Rotation | Rotates the coordinates. $\boldsymbol{A}$ is a rotation matrix, $\boldsymbol{T}, \boldsymbol{E} = \boldsymbol{0}$ |
| Translation | Applies a shift in each coordinate. $\boldsymbol{A} = \boldsymbol{I}$, $\boldsymbol{T}$ may be a non-zero vector, $\boldsymbol{E} = \boldsymbol{0}$ |
| **Rigid (Euclidean)** | Rotates and translates the coordinates. The set of rigid transformations includes all translations and rotations. |
| Uniform Scaling | Scales the coordinates. $\boldsymbol{A} = \gamma \boldsymbol{I}$ and $\boldsymbol{T}, \boldsymbol{E} = \boldsymbol{0}$ |
| Similarity (7-Parameter) | Rotates, translates and scales the coordinates uniformly. Preserves angles. In 3D, 7 parameters are required. The set of similarity transformations includes all the rigid transformations, and all the uniform scalings. |
| Scaling | Scales the coordinates differently along each axis. $\boldsymbol{A}$ is a diagonal matrix |
| Rigid+Scaling (9 parameter) | Rotates, translates and scales the coordinates with possibly a different scaling in each direction. The term "9 parameter" only makes sense in 3D, where 9 parameters are required. The set of rigid+scaling transformations includes all the similarity transformations, and all scalings. |
| **Affine** | Preserves parallel lines. $\boldsymbol{E}$ must be zero. The affine transformation includes all the rigid+scaling transformations. |
| **Projective or Homography** | Preserves collinearity, intersection and cross-ratio. The set of projective transformations includes all the affine transformations. |

**Table 3.1** The matrix based transforms form a hierarchy of transformation groups. Transformations shown in bold are used in the experiments in this thesis.

where $\boldsymbol{M}$ is a member of one of the matrix groups. $\boldsymbol{M}$ can be further broken down into

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{T} \\ \boldsymbol{E} & \gamma \end{bmatrix} \tag{3.10}$$

Here, if $dim$ is the dimension of the image coordinate system, then $\boldsymbol{A}$ is a $dim \times dim$ matrix which contains the affine components of the translation, $\boldsymbol{T}$ is a $dim \times 1$ vector containing the translational components, and $\boldsymbol{E}$ is a $1 \times dim$ vector holding the elation components. The $\gamma$ component is a scalar, which for affine and Euclidean transformations is equal to one [96]. The right hand side of 3.9 is the homogeneous coordinate of $\boldsymbol{x}$. Homogeneous coordinates [30, 96] are elements of the projective space $\mathbb{P}_{d+1}$, that is, they map to coordinates in the affine space $\mathbb{R}^d$ by dividing by the last element, $\lambda$, above.

The matrix transforms may be composed by matrix multiplication, and inverted by matrix inversion. These operations always give rise to another transformation of the same type, meaning that these types of transformations form Lie groups [10]. In fact, there is a hierarchy of matrix based transformations, each of which is a Lie group [10, 96]. These are summarized in Table 3.1. Note that all these transformations are linear in the homogeneous coordinates, but the homography is not linear in the coordinates themselves. The homography should not be viewed as a curved transformation, however, as it preserves straight lines.

The transforms may be parameterized in various ways. In this thesis, we use the Euler angle parameterization of rigid transforms. In the two dimensional case, this is simply parameterizing a rigid transformation by an angle and two translations. In the three dimensional case, this parameterizes the 3D rotation as successive rotations about the coordinate axes. Order matters, since matrix multiplication does not commute. The order used is to first rotate around the Y axis, then around the X axis and finally around the Z axis as that is the default within ITK.

Euler angles are not an ideal parameterization of 3D rotations. As rotations approach right angles, the mapping can become singular. However, Euler angles have the advantage of forming a vector space, meaning that they can be optimized using additive update steps. The most common alternative parameterization, using quaternions, requires specialized updating steps because quaternions cannot be simply

added [93]. The registrations which were performed as part of this research involved angular changes of less than 90 degrees, and Euler angles performed entirely satisfactorily. The remaining matrix based transforms are parameterized using their matrix elements directly. Details of the parameterizations can be found in Appendix F.

Matrix based transforms are frequently centered. That is, they are considered to act about a center of rotation. In practice, this is implemented by pre and post multiplying by a translation to the center,

$$\hat{\boldsymbol{x}} = \boldsymbol{T} \cdot \boldsymbol{M} \cdot \boldsymbol{T}^{-1}\boldsymbol{x}, \tag{3.11}$$

where $\boldsymbol{T}$ is a translation that moves the origin to the center point. Because the translation, $\boldsymbol{T}$, is also a matrix group transformation, this could be represented as a single matrix transformation (although with different parameters) centered on the origin. The advantage of centering the transformation is that it frequently makes the derivatives with respect to the parameters well behaved, leading to better optimization performance [109]. For the work presented in this thesis, unless otherwise stated, all transformations were centered on the center of the image.

### 3.1.8 Thin Plate Splines

A deformation field can also be represented by the change in position of a set of landmark points. The deformation on each point is then well defined, and the problem is to interpolate the deformation between the points. One widely used method is the thin plate spline (TPS). As the name suggests, it was originally developed for modeling the deformation of thin plates of metal [94]. It was proposed for the modeling of biological shape by Bookstein [31], and extended to 3D by Evans *et al.* [75]. The TPS is one of a more general class of transformations called *kernel splines* because they interpolate deformations based on a radial basis function kernel centered at each landmark [63, 177].

The kernel splines have the advantage of being easy to define, however they also have some disadvantages. The final positions of all points depend on the positions of all the landmarks, which means that in an image registration context all pixels affect and are affected by all parameters. As the number of landmark points and therefore

the number of parameters rises their computational cost increases significantly (see also Section 4.6.1). The kernel splines do not have closed form methods for composition or inversion. Indeed it is possible to generate non-diffeomorphic warps with them, that is, warps for which an inverse does not exist, or which are not invertible.

The implementation of thin plate splines in ITK had to be modified to be suitable for use in the registration framework. These modifications, and details of the formulation can be found in Appendix B.

### 3.1.9 B-spline Transformations

Rueckert [164] proposed to model deformations using a grid of control points over the image extent, with interpolation between the points using a tensor product of cubic B-splines. That is, the deformation along a particular coordinate dimension, $i$, at a point is given by [142, 164] (for the 3D case):

$$D_i\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \sum_j \beta_3\left(\frac{x - x_j}{\Delta \boldsymbol{p}_x}\right) \beta_3\left(\frac{y - y_j}{\Delta \boldsymbol{p}_y}\right) \beta_3\left(\frac{z - z_j}{\Delta \boldsymbol{p}_z}\right) \boldsymbol{\phi}_{j_i} \tag{3.12}$$

where the $j$ iterates over the neighborhood of the point, $\begin{bmatrix} x_j & y_j & z_j \end{bmatrix}$ are the coordinates of each control point, $\Delta \boldsymbol{p}$ is the control point spacing, and $\beta_3$ is a third order B-spline. The deformation is parameterized by a shift for each control point and each dimension, the $\boldsymbol{\phi}_{j_i}$ above. The definition of B-splines and their derivatives may be found in [189].

The cubic B-spline is only non-zero over neighbors two distant from the current point in each coordinate which means that the support is local – that is not all parameters affect all points which means they can be implemented efficiently. Also, the interpolation is truly separable, so the interpolation of the deformation in $x$ is not affected by the interpolation of the deformation in $y$, and so on. However, because the set of points used to model the deformation must be defined on a regularly spaced grid, in certain cases many more points are required to use a B-spline model than are necessary for a kernel spline model. This occurs for two reasons. If the deformation is concentrated in a small area, many of the control points are effectively wasted

(modeling nothing). In addition, since the deformation must be computed using neighboring control points on each side of the point of interest, it is necessary to pad the control point grid outside the area of interest. For example a "$5 \times 5$" B-spline, having 25 control points covering a 2D image, would also have to have a padding of 3 points on the edge (see Figure 3.2). These extra points add significantly to the number of parameters. This also means that outside the region of support, the transformation is undefined. This can easily lead to a "rip" – a non-differentiable portion of the warp – around the edges. Also, like the kernel splines, there are no closed form exact inverse or composition methods, and it is possible to define uninvertible warps.

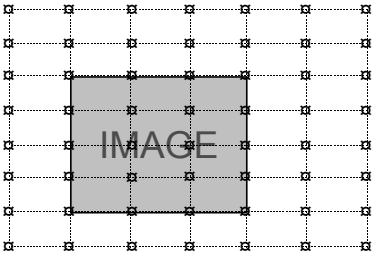



**Figure 3.2** Example of a B-spline grid. This example has $4 \times 5$ control points over the region of the image, but these must be padded with 3 extra rows and columns outside the region of interest. The transformation shown here is only well defined up to the edge of the image.

### 3.1.10 Optimization

The optimization algorithm forms a key component of the image registration process. Since the work of Lucas and Kanade [136], the majority of direct image registration algorithms have used algorithms suitable for unconstrained minimization of convex cost functions [41, 181, 208]. Of course, the necessary assumptions to use these algorithms generally do not hold perfectly. Firstly, the problem is rarely entirely unconstrained. However, in practice the constraints are usually so broad that they can be ignored. A more serious issue is that the image difference measure is rarely convex. A convex function, $f(\boldsymbol{x})$, defined over a convex set $\boldsymbol{X}$ must satisfy [104],

$$f(a\boldsymbol{x} + (1-a)\boldsymbol{y}) \leq af(\boldsymbol{x}) + (1-a)f(\boldsymbol{y}); \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{X}; \quad 0 \leq a \leq 1$$

which means that a straight line drawn between any two points on the function will be greater than or equal to the function. A somewhat weaker constraint is that the function may not have local minima. A function with a single global minimum is quasiconvex. More formally a function, $f(x)$, is quasiconvex on a convex domain $X$ if [104]

$$f(a\boldsymbol{x} + (1-a)\boldsymbol{y}) \leq \max(f(\boldsymbol{x}), f(\boldsymbol{y})); \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{X}; \quad 0 \leq a \leq 1$$

which implies that its sublevel sets are convex.

For most image registration problems, the function can be assumed to be quasiconvex provided the optimization is started near enough to the true minimum. For the practical problems motivating this thesis, this is feasible. When a patient is scanned using an MRI or CT scanner they are oriented in a roughly consistent manner in the scanning device. Image registration is needed for a refinement of an already plausible transformation, rather than for starting the process from scratch. However, if the optimization process is started outside the range over which the function can be considered to be quasiconvex, it may converge to an erroneous local minimum. The distance from the true minimum that the optimization algorithm can be started and still be expected to converge is referred to as the *capture radius*. The capture radius tends to be more a property of the cost function, than the optimization algorithm.

### 3.1.11 Unconstrained Convex Optimization Methods

Algorithms for unconstrained optimization of quasiconvex functions can be divided into two classes [80, 85, 131]. The so-called direct search methods are distinguished by neither explicitly nor implicitly using a model of the function being optimized and work by performing only greater than/less than comparison of function values [131]. One direct search algorithm, the Nelder-Mead *downhill simplex* [149], will be used in this thesis. (This algorithm is also called the Amoeba algorithm.) This is the direct search method most commonly used in medical image registration (see, *e.g.,* [138, 155]). The downhill simplex algorithm searches an $N$-dimensional space by maintaining a cluster of $N+1$ points (a simplex) in that space. At each iteration, the worst point (in terms of the objective function value) is replaced by trying one of

several alterations of the simplex. If none of the alterations are effective, the simplex is shrunk by moving all the points toward the current best point. The algorithm stops when the simplex reaches a minimum size. A detailed description of the algorithm can be found in [144], and the implementation used in this work is available in open source form as part of the ITK toolkit [118].

The remaining optimization approaches could be called *model-based algorithms* as they all form some sort of implicit or explicit model of the function at each iteration. Based on that model, a new step in the parameter space is computed. There are two ways of computing this step in common use. Line search approaches first choose a search direction, and then search for a minimum along that direction [192]. Trust-region approaches define a radius, $R$, around the current iterate where the model of the function is believed to be accurate [59]. They then solve the following constrained minimization problem:

$$\Delta\boldsymbol{\phi} = \underset{\Delta\boldsymbol{\phi}}{\operatorname{argmin}} f(\boldsymbol{\phi} + \Delta\boldsymbol{\phi}) \quad | \quad ||\Delta\boldsymbol{\phi}|| \leq R$$

where $f(\boldsymbol{\phi} + \Delta\boldsymbol{\phi})$ is the current model of the function. This problem is referred to as the *trust-region subproblem*. If the new function value meets some sufficient decrease conditions, the step is accepted, otherwise the trust-region boundary is shrunk.

In this thesis, five model based optimization algorithms are used, which will now be described in detail.

*Powell's method* [156] requires only function values in order to create its model. It performs a line search along each parameter direction in turn. This is followed by a line search along the direction from the starting point to the best point found so far. It can be shown that this constructs an implicit quadratic model of the function [131]. A detailed description of the algorithm can be found in [192], and the implementation used in this work is available in open source form as part of the ITK toolkit [116].

*Gradient descent* is used to refer to a wide range of methods which take steps in the direction of the gradient. They differ mainly in how the step size is chosen. Two gradient descent methods are used in this thesis. The first is the regular step gradient descent optimizer provided in the ITK library. This optimizer, and modifications of it are used in Chapters 5 and 6. The implementation of this algorithm is freely available

[117], but a written description is not. Therefore a written description is provided here, as Algorithm 1.

**Algorithm 1** Regular Step Gradient Descent Optimizer [109]

**Set** start position $\boldsymbol{\phi}_0$; iteration counter $n = 0$; scaling matrix $\boldsymbol{S}$; step size $s$
**Repeat**
  **Compute** the gradient at the current position $\nabla D(\boldsymbol{\phi}_{(n)})$
  **Compute** the scaled gradient, $\nabla D^* = \boldsymbol{S} \times \nabla D(\boldsymbol{\phi}_{(n)})$
  **If** the scaled gradient has changed direction by more than 90° **then**
    **Set** $s = \frac{s}{2}$  // Reduce the step size
  **End if**
  **Compute** the update to the parameters, $\Delta\boldsymbol{\phi}_{(n)} = \frac{\nabla D^*}{||\nabla D^*||} \cdot s$
  **Set** $\boldsymbol{\phi}_{(n+1)} = \boldsymbol{\phi}_{(n)} + \Delta\boldsymbol{\phi}_{(n)}$ and $n = n + 1$
**Until** the convergence criteria are reached

The second gradient descent method is a trust-region gradient descent algorithm implemented as part of this thesis. This method is based on the theory in [59], and is a reasonably straightforward application of trust-region methods to gradient descent. Details are provided in Algorithm 2.

The fundamental difference between the two implementations is that by using a formal trust-region framework, the proofs of convergence for trust-region algorithms can be expected to hold. The practical effect is that this algorithm is more robust and reliable than Algorithm 1. For low dimensional parameter spaces, the effect is minor, but for the higher dimensional spaces dealt with in Chapter 4 the effect could be significant. Algorithm 2 was developed mainly to carry out the work in Chapter 4, and has also been used in Chapter 7.

The reason both gradient descent methods are used in the thesis is chronological rather than technical. The second method had not been implemented when the work in Chapters 5 and 6 was completed. It was not considered worthwhile to reimplement that work on the newer algorithm. The experiments in those chapters are not particularly sensitive to the differences between the algorithms, and it was anticipated there would be no change in the conclusions.

Second order methods for optimization are those that use second derivatives of the cost function as part of their internal model. These methods apply Newton's method for finding zeros of a function [150] to finding zeros of the gradient. For

**Algorithm 2** Trust-Region Gradient Descent optimization.

**Set** start position $\boldsymbol{\phi}_0$; iteration counter $n = 0$; trust-region radius $R$; scaling matrix $\boldsymbol{M}$
**Compute** $\nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(0)})$ and $D(\boldsymbol{\phi}_{(0)})$
**Repeat**
  **Compute** $\boldsymbol{\phi}_{(n+1)}$ at intersection of negative scaled gradient and trust-region boundary.
  **Compute** Expected improvement $E_{(i)} = (\boldsymbol{\phi}_{(n+1)} - \boldsymbol{\phi}_{(n)}) \cdot \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(n)}) - D(\boldsymbol{\phi}_{(n)})$
  **Compute** $\nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(n+1)})$ and $D(\boldsymbol{\phi}_{(n+1)})$
  **Compute** $\rho$, the ratio of real to expected improvement. $\rho = \frac{(D(\boldsymbol{\phi}_{(n+1)}) - D(\boldsymbol{\phi}_{(n)}))}{E_{(n)}}$
  **If** $\rho < c_0$ **then**
    // This is an unexpectedly poor result, so the model is wrong. Reject the step and shrink the trust region.
    $R = \gamma_0 R$
  **Else if** $\rho < c_1$ **then**
    // The improvement is acceptable relative to the model, so accept the step
    $n = n + 1$
  **Else**
    // This improvement is good or excellent relative to the model. Accept the step, and expand the trust region
    $n = n + 1; R = \gamma_1 R$
  **End if**
**Until** convergence criteria reached

The constants $c_0$, $c_1$, $\gamma_0$ and $\gamma_1$ are 0.001, 0.1, 0.1 and 2 respectively. These values were chosen based on the recommendations in [59].

strictly quasiconvex functions, a zero of the gradient will occur at the global minimum. Frequently called Newton methods, these methods are more properly called Newton-Raphson methods as the basic update step, still used today, was first expressed by Raphson [28, 159, 206]:

$$\Delta\boldsymbol{\phi} = -\left[\mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi})\right]^{-1} \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) \tag{3.13}$$

where $\mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi})$ is the Hessian matrix of second derivatives of the cost function, $D(\boldsymbol{\phi})$. The Newton-Raphson iteration will find the solution of a quadratic minimization problem in one step, and if started near enough to the minimum these methods are known to converge to a solution extremely rapidly [59, 151]. There are two specific second order methods that are relevant to the discussion in this thesis, the Levenberg-Marquardt method, and the Gauss-Newton method.

The update step is very effective when the function is convex, as the Hessian matrix will be positive definite. However, difficulties arise when the Hessian matrix, $\mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi})$, is indefinite, which may easily arise on quasiconvex functions. Levenberg [130] and Marquardt [141] proposed instead to solve

$$\Delta\boldsymbol{\phi} = -\left[\mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) + \lambda \boldsymbol{I}\right]^{-1} \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) \tag{3.14}$$

where $\lambda$ is chosen to make the Hessian well behaved. Levenberg and Marquardt developed their work for the least-squares case, and certain authors [158, 186] have believed that their work only applies in that case. However, Equation 3.14 is general and can be applied to any cost function. The Levenberg-Marquardt approach is a particular heuristic solution to a problem that trust-region Newton-Raphson methods solve in a variety of ways [59]. In this thesis, a trust-region Newton-Raphson optimization algorithm implemented as part of this thesis has been used as a second order method. The Steihaug-Toint method of solving the trust-region subproblem (Equation 3.13) is applied. In brief, this method applies a conjugate gradient method to solving Equation 3.13 under the trust-region constraint. The reader is referred to [59, p. 205] for details of the Steihaug-Toint method. Details of the algorithm, other than the solution of the trust-region subproblem are given in Algorithm 3. This method may be viewed as comparable to the more widely known Levenberg-Marquardt methods.

**Algorithm 3** Trust-Region Newton-Raphson optimization.

**Set** start position $\boldsymbol{\phi}_0$; iteration counter $n = 0$; trust-region radius $R$; scaling matrix $\boldsymbol{M}$
**Compute** $\mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(0)})$, $\nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(0)})$ and $D(\boldsymbol{\phi}_{(0)})$
**Repeat**
  **Compute** $\boldsymbol{\phi}_{(n+1)}$ using Steihaug-Toint method [59, p. 205].
  **Compute** Expected improvement $E_{(n)} = (\boldsymbol{\phi}_{(n+1)} - \boldsymbol{\phi}_{(n)}) \cdot \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(n)}) - D(\boldsymbol{\phi}_{(n)})$
  **Compute** $\mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(0)})$,$\nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(n+1)})$ and $D(\boldsymbol{\phi}_{(n+1)})$
  **Compute** $\rho$, the ratio of real to expected improvement. $\rho = \frac{(D(\boldsymbol{\phi}_{(n+1)}) - D(\boldsymbol{\phi}_{(n)}))}{E_{(n)}}$
  **If** $\rho < c_0$ **then**
    // This is an unexpectedly poor result, so the model is wrong. Reject the step and shrink the trust region.
    $R = \gamma_0 R$
  **Else if** $\rho < c_1$ or step is within the trust region **then**
    // The improvement is acceptable relative to the model, so accept the step
    $n = n + 1$
  **Else**
    // This improvement is good or excellent relative to the model and the step is at the edge of the trust region. Accept the step, and expand the trust region
    $n = n + 1; R = \gamma_1 R$
  **End if**
**Until** convergence criteria reached

The constants $c_0$, $c_1$, $\gamma_0$ and $\gamma_1$ are 0.001, 0.1, 0.1 and 2 respectively. These values were chosen based on the recommendations in [59]. The fact that the expected improvement is not calculated with a quadratic model is intentional, as empirical tests showed that this seems to improve the efficiency significantly.

The particular case of minimizing a sum of squared errors arises very frequently. Gauss [82] was the first to use the least squared error criterion and developed an approximation to the Hessian matrix for use in Equation 3.13. Optimization using this Hessian approximation is frequently called Gauss-Newton optimization. This can be confusing however, because any second order technique can be applied using the Gauss-Newton Hessian approximation. In this thesis the convention will be adopted that all methods using a variation of Equations 3.13 or 3.14 will be called Newton-Raphson methods. Gauss's technique will be referred to as the Gauss-Newton approximation of the Hessian. This method is discussed in detail in 4.3, where it is generalized to non-least squares cost functions.

Quasi-Newton methods are a class of methods intended to gain the rapid convergence of the Newton-Raphson methods without incurring the computational difficulty of computing the Hessian matrix. They require only gradient information, and build up an estimate of the Hessian matrix, or its inverse, from the changes observed between successive gradients. There are various ways of building up this estimate, but the method that is currently considered superior [151, 192] is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [43, 79, 86, 169] method. In this thesis a limited memory version of the BFGS method is used (referred to as LBFGS). The implementation is described in detail in [48, 207], and the source code is freely available as part of the ITK library [115].

### 3.1.12 Multiresolution Approach

For most practical direct image registration problems, multiresolution approaches must be applied in order to avoid becoming trapped in local minima. The framework proposed by Bergen *et al.* [24] is still what is widely used today. The images to be registered are reduced in size using a process of Gaussian blurring and subsampling. The optimization is begun on the lowest resolution of the pyramid. The optimal parameters found at each level are then used as a starting point for the next level of optimization.

In this work we used a Gaussian scale space pyramid, with each level being reduced by a factor of two from the previous level. As a heuristic guide, the number of multiresolution levels was chosen so that the top level would have at least as many

pixels as a $64 \times 64$ image in two dimensions, and a $32 \times 32 \times 32$ image in three dimensions.

Certain authors [154, 185] have questioned the use of a Gaussian scale space for registration with mutual information. However, the approach is widely used for registration with MI. A simple investigation shows that the Gaussian scale space achieves the desired effect on the cost function for MI. Figure 3.3 shows the mutual information of an image (actually the image in Figure 4.1) with a transformed version of itself at a set of points ranging from $\begin{bmatrix}-0.28\text{radians} & -16\text{pixels} & -16\text{pixels}\end{bmatrix}$ to $\begin{bmatrix}0.28\text{radians} & 16\text{pixels} & 16\text{pixels}\end{bmatrix}$ in the rigid transformation space. Note how as the resolution is reduced, the function progressively flattens out and the region where it slopes nicely down toward the minimum gets wider. This multiresolution approach is effectively widening the capture radius of the MI function.

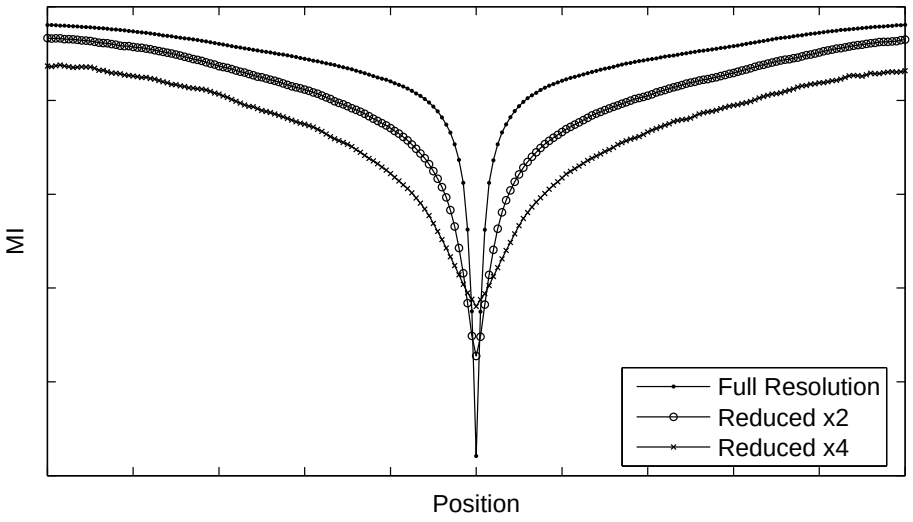



**Figure 3.3** Mutual information cost function at three levels of a Gaussian scale space pyramid. The line with small dots was produced using the full resolution images, the line with circles by reduction by a factor of 2, and the line with crosses by reduction by a factor of 4. Note how the derivatives of the function become less extreme, and how the point of inflection moves further from the optimum at lower resolutions.

This concludes the description of the image registration framework used in this research. Most of the experiments take the form of performing many similar registrations with the component of interest being varied. The following section discusses

how the results of those registration runs were interpreted.

## 3.2 Evaluation of Results

The primary means of empirically testing the claims in this thesis is by running many image registrations with each algorithm, and comparing the results. These experiments are performed mainly on image pairs between which there is a known transformation. For each such pair, a set of starting transformations is chosen. Each starting transformation is composed with the true transformation, and this is provided as a starting point for the optimization process. Each algorithm is provided with exactly the same inputs, and therefore corresponding runs are directly comparable.

In general, each experiment involves collection of data on the accuracy, efficiency and reliability of the registration process. The efficiency is measured both by measuring the time required to perform the calculation, and keeping track of the number of function evaluations, and optimizer iterations. Times were collected as elapsed clock time required for the actual performance of the algorithm. Running time is notoriously variable on multitasking computer systems, but several factors combine to make these reported times as reliable as possible. Firstly, reported times are for the algorithm itself, and I/O times have been ignored. Secondly, the vast majority of the reported experiments were performed on cluster machines, where the machine is dedicated to the image registration task during the programs run. Thus the variability in runtime caused by competition for resources between processes should be minimized.

The accuracy is measured by computing the mean target registration error (mTRE) between the final transform and the true transform. In order to evaluate the results of a registration, it is necessary to quantify how "close" a particular registration is to the right answer. There are a number of possible measures of distance between transformations. Van de Kraats *et al.* [190] have proposed that registration comparisons should use the mTRE over the region of interest. In [190] this is computed using a grid of points over the image extent. The same method is used in this thesis. For transformations based on the matrix groups, we use a $10 \times 10$ grid (in 2D, a $10 \times 10 \times 10$ grid in 3D) of points evenly spaced in the image extent. For transformations commonly

considered deformable (*i.e.,* spline based transforms) a $50 \times 50 \times 50$ grid of points is used.

Finally the reliability is considered in terms of how often the algorithm failed. The algorithm could fail in one of two ways. The more obvious way is that the optimization fails to converge, and an error is reported. A rule was also imposed that if the mTRE was greater than 5 pixels, the run would be considered to have failed. This approach was taken because a single run that converges incorrectly could artificially distort the mTRE. Fortunately, this criterion was rarely ambiguous. Failed runs were not considered in the comparisons of runtime and accuracy.

### 3.2.1 Statistical Testing

Since each algorithm being tested was run on the same set of test inputs, paired comparisons can be used to test for significance. That is, algorithms can be compared on a run by run basis with identical inputs. This provides a much finer ability to detect differences between the algorithms. In all cases, the tests were performed at the 95% confidence level, after applying any corrections.

Numerical quantities such as running time and mTRE were tested for statistical significance using the paired t-test [171, pp. 433–61]. The paired t-test tests the hypothesis that the differences between corresponding samples have a mean of zero. A rejection of this hypothesis justifies a conclusion that there is a significant difference in the results of two algorithms. The paired t-test is appropriate when using interval data – that is that the magnitude of the numbers has real meaning, and the differences can be considered to have a normal distribution.

Letting the mean of the $N$ differences between the matched pairs be $\bar{X}$, and the standard deviation of these differences be $\sigma$, the $t$-statistic is

$$t = \frac{\sqrt{N}\bar{X}}{\sigma}$$

If the mean of the differences is zero, this statistic will be drawn from a $t$-distribution with $N - 1$ degrees of freedom [171, pp. 433–61].

The failure of an algorithm is a binary event, it either failed or it succeeded. Thus a different significance test is needed in this case, and the McNemar test [171, pp. 491–

508] was used. The McNemar test sets up a contingency table listing all the possible outcomes. Specifically,

| | Algorithm 1 passed | Algorithm 1 failed |
|---|---|---|
| Algorithm 2 passed | $a$ | $b$ |
| Algorithm 2 failed | $c$ | $d$ |

The McNemar test examines whether there is a difference between $b$ and $c$ using a sign test. If there is no difference between the algorithms (the null hypothesis), $b$ will be drawn from a binomial distribution $B(b + c, 0.5)$. A rejection of this hypothesis justifies a conclusion that there is a significant difference in failure rate between the algorithms.

Statistical hypothesis testing suffers from the problem that the tests can only show statistical differences, not statistical "sameness". For example, the tests cannot detect whether both algorithms have the same failure rate. Instead, what can be stated is whether or not a difference in their failure rates can be detected. When multiple comparisons are performed as in these experiments, pure chance dictates that some Type I errors, or false rejections of the null hypothesis, $H_0$, will occur.

To account for this problem, a correction must be applied to the test statistic which increases the difference required to declare a particular result significant. For the mTRE and timing data the Tukey correction was applied. This replaces the test statistic from the student t distribution with one from the studentized range distribution [171, pp. 534–5], which accounts for the extra comparisons. The failure rate tests were adjusted using the Bonferroni-Dunn correction. The Bonferroni-Dunn correction simply recomputes the test statistic at a significance level divided by the number of comparisons being made, which reduces the possibility of Type I error in accordance with the Bonferroni inequality [171, pp. 531–4].

These corrections do however reduce the power of the tests and tend to increase the likelihood of a Type II error, or false acceptance of $H_0$. Depending on the hypothesis under test, either of these types of errors might artificially bolster the argument. For instance, the correction for multiple comparisons could easily hide a statistical difference in algorithm failure rates. Conversely, not correcting for multiple comparisons could artificially indicate a difference in running time where none exists. Therefore, all three possible cases are reported. Where the corrected tests rejected $H_0$, it can be

concluded that the performance of the two algorithms differ. When $H_0$ was rejected using pairwise comparisons, but accepted when adjusted for multiple comparisons the result is considered ambiguous, and finally when $H_0$ was accepted by all tests, the data do not indicate a performance difference between these algorithms.

### 3.2.2 Comparing Deformable Transforms

Understanding the accuracy of a deformable registration is not as simple as the case of the matrix based transformation. In the case of matrix based transforms, errors in the recovered transformation affect the entire image in a more or less even way. Residual error is lowest at some point, and then increases radially away from that point. However, in the case of a deformable transformation different regions may be registered to greatly differing levels of accuracy. A very accurate part of the transformation can tend to average away a very inaccurate part, and vice-versa. Therefore, rather than reporting mTRE, histograms of the resulting errors are examined. These show better the change in the overall error produced by the registration. To generate these histograms the residual errors on the same $50^d$ grid of points used for the mTRE calculation are kept, and their distribution plotted.

Nevertheless, the ultimate goal of the experiments is to be able to make a statement such as "method A outperforms method B". For the case of deformable transformations, just like the matrix based transforms, paired runs can be compared. For each such pair, one run (A) will be considered better than another (B) if the following criteria are met:

1. The runs have a statistically significantly different median error when all the data is considered, and,

2. the median error of A is lower than the median error of B.

A statistically significant difference in median is detected with the rank-sum, or Mann-Whitney "U", test [171, pp. 181-94] on all the errors from both cases. This test orders the set of errors, and determines if either case falls above or below the other in rank enough times to reject the null hypothesis. This test was chosen as it is suitable for non-normal data, and the distributions of errors very clearly does not follow a normal distribution.

## 3.3 Summary

In order to test each of the improvements to image registration algorithms proposed in this thesis, software has been implemented which can perform registration on two and three dimensional images. The software was implemented using the ITK library, which provides well tested implementations of standard registration algorithms. Using a well-tested standard implementation helps to reduce the chance of researcher bias.

For all experiments, registration was performed using linear interpolation and a multiresolution technique based on Gaussian pyramids created for each image. Depending on the nature of the components being tested, various transforms are used, including two and three dimensional rigid and affine transforms, two dimensional homographies and the deformable thin plate spline and B-spline models. A range of optimizers are also used, including the downhill simplex algorithm, Powell's method, two variations on gradient descent, a limited-memory BFGS algorithm and a trust-region Newton-Raphson technique.

The majority of the experiments to be shown in Chapters 4, 5, 6, and 7 take the form of comparisons of two algorithms. These comparisons are performed by running each algorithm on identical input and observing the time required, accuracy and success rate. Statistical testing techniques are used to ensure that the observed results are in fact significant. The following chapter begins the theoretical and experimental contributions of this thesis with an analysis of the optimization component of image registration.

# Chapter 4

# Optimizer Selection and Scaling in Image Registration

The previous chapters have shown how the direct parameterized registration problem is expressed as an optimization problem over the parameters of the transformation. It is obvious that the selection and configuration of the optimization algorithm used has a huge influence on the success of any approach. However, most reported studies of optimization algorithms in this context have compared optimizers for specific registration contexts. If a particular registration problem is not the same as those studied, it is not clear how to generalize the previous work. The primary goal of this chapter is to determine how the specific registration problem characteristics affect the performance of optimization algorithms in order to develop a principled way of selecting and configuring an appropriate algorithm for any given context. However, comparing the performance of optimization algorithms is only meaningful if they are correctly configured and supplied with valid input. Two significant factors affecting performance have not been well analyzed in previous work. These are (1) the calculation of approximate Hessians when Newton-Raphson optimizers are used, and (2) the scaling of the transformation parameters.

To use Newton-Raphson optimization algorithms for image registration, it is necessary to have a valid approximate Hessian calculation. For least-squares problems the Gauss-Newton approximation to the Hessian has proved very effective. In this thesis, the Gauss-Newton approximation is generalized to non-least squares cost functions

for the first time. It is shown that the resulting approximate Hessians are effective for both normalized correlation and mutual information. It is also shown that previous approximations to the Hessian of mutual information are ineffective.

It is known that the scaling of the parameter space has a significant effect on the performance of optimization algorithms. Image registration work in the literature often reports using scaling factors (*e.g.,* [26, 109, 172, 194]), but these are determined in an ad-hoc way. In this chapter, the issue of scaling is analyzed and a method for automatically determining an optimal set of scaling factors for any given optimization problem is developed. It is shown that the scaling of the transformation parameters plays a critical role in optimizer performance on matrix based transforms.

This chapter then deals with three intertwining issues – (1) approximate Hessian calculation, (2) parameter scaling, and (3) optimizer selection. It is therefore structured in three main investigative components. Each component is presented with its own set of experiments so that the topic is complete, and the result of the analysis can be applied in the subsequent investigations. The following section discusses previous work. To avoid repetition, Section 4.2 then presents the experimental data that is used in all three investigations. Section 4.3 develops of a generalization of the Gauss-Newton Hessian approximation, with simulations on a simple problem to show the effectiveness of the approximation, and simulated image registration experiments to verify the effectiveness of the approach. Section 4.5 discusses the importance of parameter scaling, and derives a method for automatic selection of optimal scaling of the transformation parameters. The scaling is tested on a wide range of simulated and real image registration problems. The experiments show conclusively that using these optimal scaling factors improves performance on matrix based transforms. Once the issues of Hessian calculation and scaling have been addressed, it is finally possible to return to the motivating question of this chapter: Given a registration problem, which optimizer should be used? Section 4.6 first analyzes the expected behavior of the optimization algorithms. Simple toy problem experiments are presented to illustrate the results of this analysis, followed by extensive experiments on simulated and realistic image registration problems. Based on the results of the analysis and the experiments, the key characteristics of registration problems that affect optimizer performance have been identified, and general guidelines for optimizer selection have

been developed. Finally Section 4.7 reviews the combination of results found, and summarizes the results in the form of guidelines.

### Publications

A very preliminary version of the work on parameter scaling in Section 4.5 was described in:

[37] Rupert Brooks, D. Louis Collins, and Tal Arbel. Scaling angles and distances to maximize efficiency of image registration. Short paper presented at the 8th International Conference on Medical Image Computing and Computer Aided Intervention (MICCAI 2005), October 2005. available online `http://www.ia.unc.edu/MICCAI2005/ShortPapers/`.

## 4.1 Previous Work

Recall that the direct, parameterized image registration problem can be expressed as the following optimization problem,

$$\boldsymbol{\phi}_{opt} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}} \left( D(I_f(\boldsymbol{X}), I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))) \right), \tag{4.1}$$

which considers the optimal transformation parameters, $\boldsymbol{\phi}_{opt}$, to be those that minimize a difference measure between a fixed image, $I_f$, and a transformed moving image, $I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))$ [91, 145].

Many different algorithms for unconstrained optimization have been applied to this problem. Table 4.1 summarizes the use of different algorithms listed in two surveys of medical image registration [138, 155]. From these surveys, it appears that the downhill simplex, Powell's method and gradient descent are very popular approaches. The Levenberg-Marquardt technique is popular in [138], but much less popular in [155]. This can perhaps be explained by the fact that [155] surveys mutual information registration only, and calculation of the MI Hessian is difficult. An examination of the more recent literature suggests there is a trend toward the use of the LBFGS quasi-Newton optimizer for B-spline transforms (*e. g.* [124, 142, 165, 198]), quite possibly due to its easy availability in the ITK [109] library. In the computer vision

field, a survey itemizing a count of different approaches could not be found, but two recent surveys [181, 208] support the idea that most approaches use a Newton-Raphson optimization of a squared difference measure with the Hessian approximated using the Gauss-Newton method. These approaches are based on the Lucas-Kanade algorithm [136]. For computer vision problems where this approach is not viable, no clear preference is apparent.

| Optimization Approach | Count in [138] | Count in [155] |
|---|---|---|
| Downhill Simplex | 12 | 16 |
| Powell's Method | 17 | 16 |
| Misc. Direction Set | 8 | - |
| Misc. Gradient Descent | 11 | 16 |
| Quasi-Newton | - | 2 |
| Levenberg-Marquardt | 12 | 2 |
| Newton-Raphson | 3 | - |
| Other | 24 | 7 |

**Table 4.1** Number of image registration papers using various optimization algorithms reported in survey papers. The category of other contains methods to which the discussion in this paper does not apply, such as simulated annealing, genetic algorithms, etc.

Several authors have previously examined the question of which optimizers are best for a particular registration application. The work of Floca and Dickhaus [81] takes perhaps the ultimate empirical approach to this question. They propose a system to exhaustively explore the space of possible optimizers and parameter settings for registration problems. In effect, they optimize over optimizers. While interesting, this approach may not scale well to anything other than very small, focused, problems.

Bernon *et al.* [25, 26] compared the downhill simplex and Powell optimizers in the context of registering segmented MR and SPECT data using a mutual information (MI) cost function. Overall, [25, 26] concluded that downhill simplex was better than Powell's, but this result was only reached after much adjustment of the original problem configuration. Their initial results showed that Powell's algorithm was much more reliable than the downhill simplex, but much slower. However, they ultimately changed their problem construction to a sort of multiresolution approach which improved the performance of the downhill simplex algorithm. In [26] they

briefly addressed the issue of how performance varied with the number of parameters by comparing the performance of their algorithms on a 6-parameter rigid and a 9-parameter similarity transform. They reported that the performance of both algorithms was much worse on the 9 parameter one.

Maes *et al.* [137] compared Downhill Simplex, Powell, gradient-descent, Polak-Ribiere conjugate gradient, LBFGS and Levenberg-Marquardt algorithms in the context of multiresolution affine registration with MI. All algorithms were adjusted to converge to the same level of accuracy, and there were no failures. Thus the primary measurement of algorithm quality was timing, and they found that the downhill simplex, conjugate gradient and Levenberg-Marquardt were faster than the others. Their calculation of the Hessian for the Levenberg-Marquardt approach is unusual, an issue discussed further in Section 4.3. Regrettably, however, there was no discussion of scaling or parameterization in [137], which this chapter will show is a critical factor in determining performance.

Klein *et al.* [120] compared a number of optimization algorithms for the problem of deformable registration with B-splines. They compared the algorithms in terms of speed and accuracy. Accuracy was measured by the average displacement error after recovery of a simulated deformation field, and the overlap of segmented structures in real data. This work included both deterministic approaches – where they compared two types of gradient descent, Nonlinear Conjugate Gradient (NCG), and LBFGS – and stochastic optimization approaches, including Keifer-Wolfowicz, Robbins-Monro and an evolutionary strategy. Overall, their conclusions were that the LBFGS, and NCG performance were roughly similar, with NCG being somewhat less computationally efficient. Gradient descent was found to perform somewhat poorer on both criteria than either the LBFGS or NCG methods. Of the stochastic approaches, the Robbins-Monro method proved superior, and was the method that they concluded was ideal.

Trust-region optimization has not been widely used in image registration. However, Liu and Chen [132] have made a strong argument that trust-region optimization approaches were faster than line-search based approaches for a template based tracking problem. They were performing Newton-Raphson optimization using a histogram-based measure between a template, and its location in a new frame. Because the

tracking context is time sensitive, they conclude with a clear preference for the faster algorithm, and did not investigate precision or reliability.

Thévenaz and Unser [186] proposed a modified Levenberg-Marquardt algorithm for optimization of MI. Their modification is based around an approximation to the MI Hessian that they develop by truncating the Taylor series expression. They present good results in terms of accuracy with their algorithm but the issues of speed or number of iterations are not discussed. However the other algorithms against which they compare differ in many aspects, in particular in how the MI is computed. Dowson and Bowden [70] perform the same truncation of the Taylor series. Optimization methods were not the focus of their work, and they use only Levenberg-Marquardt optimization. In Section 4.3 this Hessian approximation will be discussed and it will be shown that it does not describe the true curvature of the MI function well. The findings in this thesis are that Newton-Raphson type optimization schemes with this approximation are no more effective than gradient descent. Fortunately, a better approximation is possible.

Each of the optimization comparison studies in the literature has concentrated on a specific problem domain, and the question of how to generalize the results has not been addressed. Previous work on Newton-Raphson optimization of MI, it will be shown, has not properly approximated the Hessian, and in [70, 185, 186] has not addressed the question of whether using the Hessian actually helps optimization performance. Finally, the issue of scaling the transformation parameters has not been formally addressed. This chapter will address all these issues using a common experimental framework and data. To avoid repetition, these common elements are discussed in the following section.

## 4.2 Materials and Methods

To address the issue of optimizer choice in a generalizable way, this chapter investigates the performance of five optimization algorithms: (1) the downhill simplex algorithm, (2) Powell's algorithm, (3) a Trust-Region Gradient Descent algorithm, (4) the LBFGS quasi-Newton algorithm, and (5) a Trust-Region Newton-Raphson algorithm. Details on all these approaches have been given in Section 3.1.10. These

optimizations have been applied to registrations using mean squared difference (MSD, Section 3.1.2), normalized correlation (NCC, Section 3.1.3) and mutual information (MI, Section 3.1.4). These optimization algorithms and image difference measures are applied to eight different transformations and six different datasets which are detailed below.

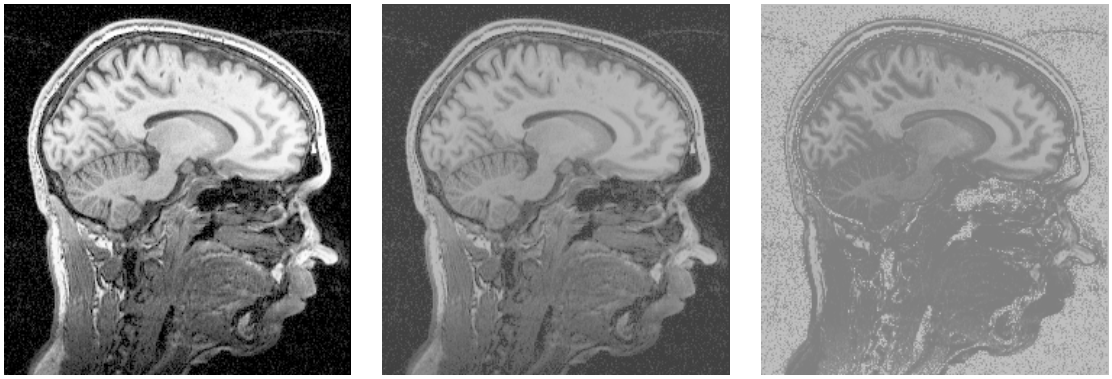
### 4.2.1 Transformations Used

The image registration experiments have been selected to use a range of transformations of different types, with different numbers of parameters. Specifically the experiments use four matrix transforms (see Section 3.1.6): (1) a 2D rigid transformation, (2) a 3D rigid transformation, (3) a 2D homography transformation and (4) a 3D affine transformation; and four deformable transforms: (5) a 2D thin plate spline (TPS) transformation defined by 20 landmarks, (6) a 2D B-spline transformation defined on a $4 \times 4$ grid, (7) a 3D TPS transformation defined on 48 landmarks and (8) a 3D B-spline transformation on a $4 \times 4 \times 4$ grid. Transforms 1, 2, and 4 are most widely used in medical imaging contexts, whereas transform 3 is of particular importance in computer vision, since it models the relationship between images taken from a rotating camera. The four deformable transforms are also most widely used in medical imaging, although variations on the TPS transform have interesting computer vision applications as well [20, 21].

The TPS transform is defined by a set of corresponding landmark points in both images. Rather than manually selecting landmark points, a method was developed to create evenly distributed meshes of points suitable for use as landmarks in a thin plate spline. This method was used to create two point distributions to define transforms (5) and (7). The procedure for generating these point distributions is further described in Appendix C. As discussed in Section 3.1.9, the B-spline transformation is completely defined by specifying the size of the grid used to span the image. Note that the B-spline grid is padded, so a $4 \times 4$ transform also has 33 additional control points outside the image.

### 4.2.2 Intensity Distortion

Each of the image difference measures being used is appropriate for different situations. MSD is suitable when the image intensities match exactly, but NCC and MI are intended for cases with linear, and non-linear intensity distortion respectively. To simulate this the intensities of the images are distorted for the cases where MI and NCC are used. Figure 4.1 shows one of the images used for several experiments under each type of intensity distortion.



(a) Original image (b) linear distortion (c) nonlinear distortion

**Figure 4.1** (a) Sagittal T1-weighted MRI slice of the brain used as a base for 2D rigid and deformable synthetic registration problems. This figure also shows examples of the intensity distortions used in all NCC and MI registration cases. For the NCC measure, the intensities (which started in the range 0-255) were rescaled to be between 64 and 192 (b). For the MI measure, the following nonlinear distortion was applied. $I_{out} = 256 * (I_{in}/256 - 0.5)^2 + 127$ (c). This image was collected at the Montreal Neurological Institute.

### 4.2.3 Simulated Registration Problems

A simulated registration problem has been defined for each of the transformations of interest. For each transformation a typical image where this type of transformation would be relevant was selected and used as a base. For each case, five random transformations were created and the selected base images were warped by these transformations. For the matrix based transforms, the images were warped using the same type of transformation as was being recovered. Thus it was possible, at least in

principle, to recover the transformation exactly. For the deformable transformations, however, the images were warped by a random deformation of a different type than was recovered, specifically a $5 \times 5$ B-spline in 2D and a $5 \times 5 \times 5$ B-spline in 3D. Therefore the deformation cannot be recovered perfectly, which more closely approximates real conditions. In each case, the registration process to recover the synthetic warp was started from a set of different randomly generated starting positions, leading to a large enough number of runs for statistical testing to be performed. The number of starting positions and their ranges (in mTRE) from the true starting position are listed in Table 4.2 for all registration problems.

| Registration Problem | # Starts | # Cases | # Runs | mTRE Range |
|---|---|---|---|---|
| Synthetic Problems | | | | |
| 2D Rigid | 15 | 5 | 75 | 3–36mm |
| 2D Homography | 60 | 5 | 300 | 2.6–62pixels |
| 3D Rigid / Affine | 45 | 5 | 225 | 2–42mm |
| 2D Deformable | 5 | 5 | 25 | 2.8mm* |
| 3D Deformable | 5 | 5 | 25 | 3.1mm* |
| Realistic Problems | | | | |
| Axial Brain Slices | 60 | 8 | 480 | 8–60mm |
| Vertebra Volumes | 40 | 6 | 240 | 4–26mm |
| Deformable Phantom | 1 | 4 | 4 | 2.2–5.2mm* |

**Table 4.2** For each registration test problem a set of random start positions were determined (# Starts), and there were a number of image pairs to register (# Cases). Thus the total shown in # Runs was performed for each optimizer, scale factor, and image difference tested. The mTRE range column shows the range of mTRE for the starting positions used.
* Deformable mTRE's reported as *median* target registration error.

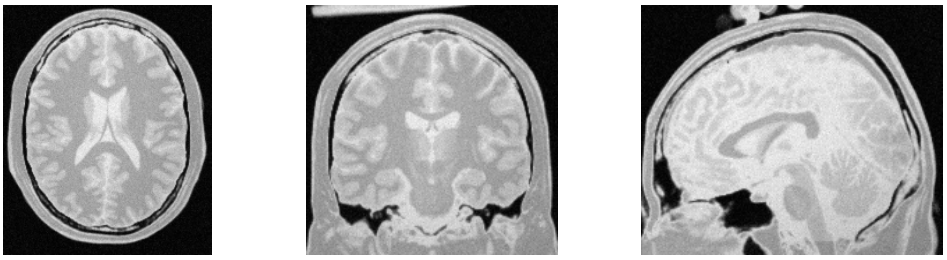


**Figure 4.2** Brainweb volume used for simulated registration experiments

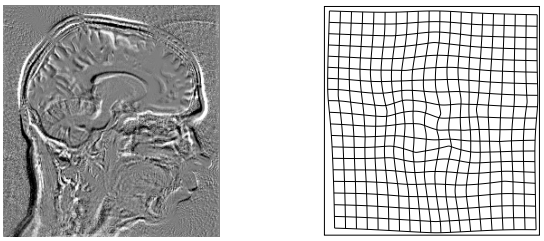
Copies of the base image were also given the intensity distortions shown in Figure 4.1 for use with the NCC and MI measures. The registration experiments were conducted using the warped image, which does not have intensity distortion, as a fixed image, and the (possibly intensity distorted) base images as the moving image. The image shown in Figure 4.1 was used for the 2D rigid, 2D TPS and 2D B-spline tests. Examples of the warped images used are shown in Figures 4.4a and 4.4b. For the 3D rigid and affine tests, a simulated T1 volume from the Brainweb [123] software was used (see Figure 4.2). For the 3D TPS and B-spline tests, the partially inflated case of the deformable phantom volumes used in the realistic experiments (described in the following section, see Figure 4.7b) was chosen as a base image. Examples warped by one of the random warps used are shown in Figure 4.4e.

(a) Original (b) Original cropped to ROI (c) Example of warped image

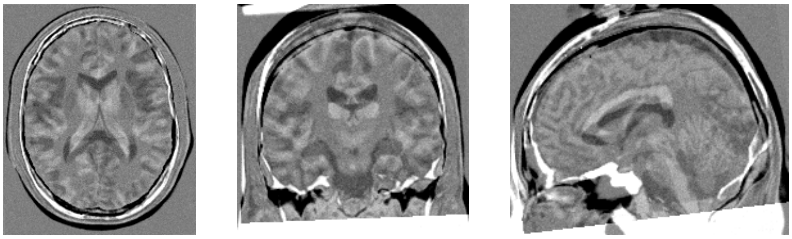
**Figure 4.3** Agra fort image used as base for homography tests, with example random warp.

The homography transformation is more relevant to computer vision than it is to medical imaging. Therefore a digital photo was used as a base image for these synthetic registration experiments. The image was warped by the randomly generated transformation, and the central part of the image was then cropped out. This cropping process avoided having the edge of the image in the region to be registered, which could have skewed the results. The original image, and an example of the warped input data are shown in Figure 4.3. All work was done with greyscale versions of the images.

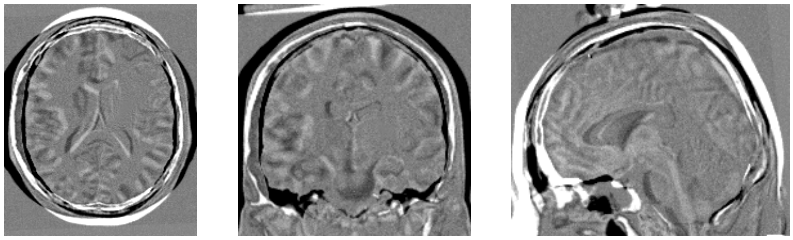
(a) 2D sagittal brain MR slice warped by rigid transformation



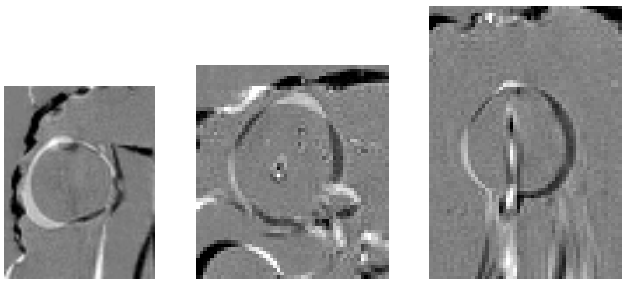
(b) 2D sagittal brain MR slice warped by deformable transform also showing deformation field



(c) Example cross sections through 3D simulated brain MR volume warped by rigid transformation



(d) Example cross sections through 3D simulated brain MR volume warped by affine transformation



(e) Example cross sections through MR image of deformable phantom warped by 3D deformable transformation

**Figure 4.4** Difference images showing typical random warps that were applied to each base image to generate the simulated registration problems.

### 4.2.4 Realistic Registration Problems

Simulated registration problems provide a precise gold standard for accuracy since the deformation is known exactly. However, real registration conditions are more difficult. Each image of a given image pair to be registered may have different patterns of noise, and different real distortions in the object being imaged. For multimodal datasets, the intensity relationship is generally more complex than the intensity distortion applied in the simulated case. For these reasons, it is desirable to do additional experiments on realistic image registration cases to confirm results found in the simulated cases.

Three realistic datasets were used:

1. **Axial Brain Slices:** This is a set of matched 2D slices taken from previously independently registered brain volumes in four different modalities: Proton Density (PD), T1-weighted, and T2-weighted magnetic resonance images (MRI) and a computed tomography (CT) image. These are shown in Figure 4.5. This data was provided by the Montreal Neurological Institute.

2. **Vertebral Volumes:** These are two sets of three previously registered volume images of cadaver vertebrae discussed in [190] and provided by the authors of that paper. Three modalities, computed tomography (CT), 3D X-ray (3DRX) and Magnetic Resonance Imaging (MRI) are provided. The CT volume was resampled to a 0.75mm isotropic pixel size to be roughly compatible in size with the other datasets. Example slices are shown in Figure 4.6.

3. **Deformable Phantom:** These are magnetic resonance (MR) images of a deformable phantom object created by Dr. Xavier Morandi. The phantom contains a balloon which can be inflated to cause a deformation. A volume of interest containing the balloon was selected and cropped out. This volume was selected as it undergoes significant deformation, also, by working on a cropped dataset, edge effects could be avoided when a synthetic deformation was applied. The volume was imaged in three states of inflation of the balloon, shown in Figure 4.7.

For datasets 1 and 2, ground truth transformations were provided with the data. For the deformable phantom object, no ground truth was easily available. To create

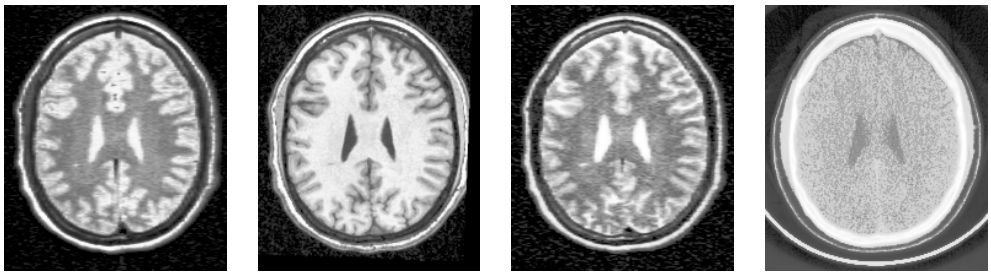
a measure of accuracy, the surface of the balloon object was segmented into a dense mesh using an automatic iso-surface finding algorithm. The resolution of this mesh was approximately 1mm. It is to be expected that a correct transformation should map points from the mesh of the balloon in the fixed image near to points in the mesh of the balloon in the moving image. The procedure for determining the accuracy of a registration of these volumes was as follows.

1. Warp the balloon mesh corresponding to the fixed image by the transformation found by the registration.

2. Match points in this warped mesh to the nearest point in the balloon mesh of the moving image.

3. Compute distances between these matched points. The distribution of these distances may be examined as discussed in Section 3.2.2.

Table 4.3 summarizes the properties of all the datasets, and the parameters of the registration experiments.

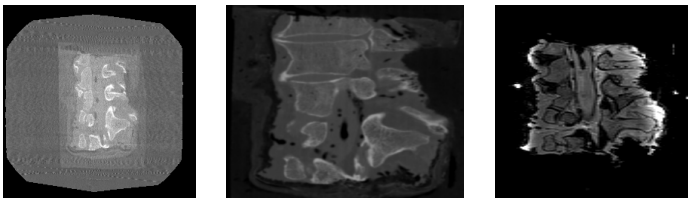


(a) PD (b) T1 (c) T2 (d) CT

**Figure 4.5** Multimodal axial slices.

### 4.2.5 Optimizer Settings

All of the optimization algorithms used require parameters – stopping criteria – for specifying their precision. Obviously if one algorithm was given much more refined stopping criteria than another, it would appear to be slower, but more precise. An



(a) 3DRX (b) CT (c) MR

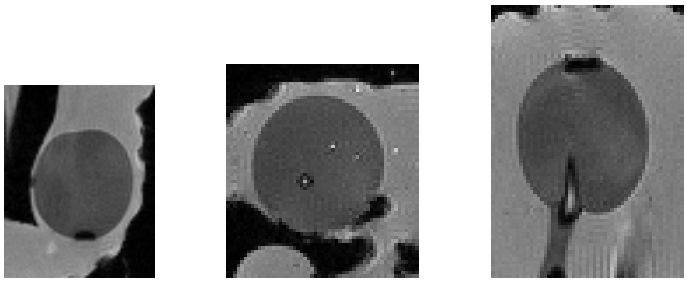
**Figure 4.6** Example cadaver vertebra images.

| Dataset | Size | Resolution | Levels | % Pxl | Experiments |
|---|---|---|---|---|---|
| Sagittal Brain Slice | $240 \times 256$ | 1mm | 2 | 100% | Simulated 2D Rigid, TPS, B-spline |
| Axial Brain Slices | $217 \times 181$ | 1mm | 2 | 100% | Realistic 2D Rigid |
| Agra Fort | $1000 \times 700$ | n/a | 4 | 30% | Simulated Homographies |
| Brainweb Simulated MR | $181 \times 217 \times 180$ | 1mm | 3 | 30% | Simulated 3D Rigid, Affine |
| Vertebrae | Varies | Varies | 3 | 100% | Realistic 3D Rigid |
| Deformable Phantom | $61 \times 78 \times 55$ | 1mm | 2 | 70% | Simulated 3D TPS, B-spline. Realistic 3D TPS, B-spline. |

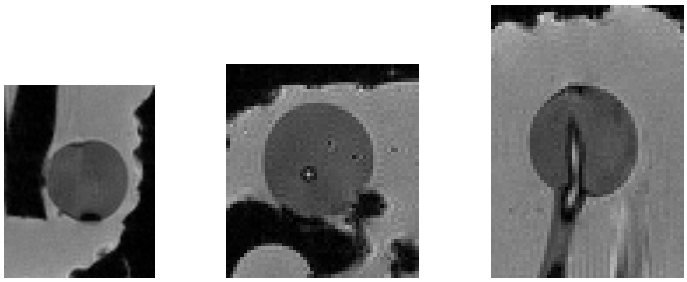
**Table 4.3** Summary of Registration and Dataset characteristics. The Levels column refers to the number of multiresolution levels which were used in the optimization. For speed of processing, certain datasets were registered using a random subset of the pixels in the image. The % pxl column indicates the size of that subset.

effort was made to choose the criteria in a straightforward manner and avoid "tweaking". Table 4.4 details the parameter settings for each algorithm. These were held fixed for all the registration experiments.

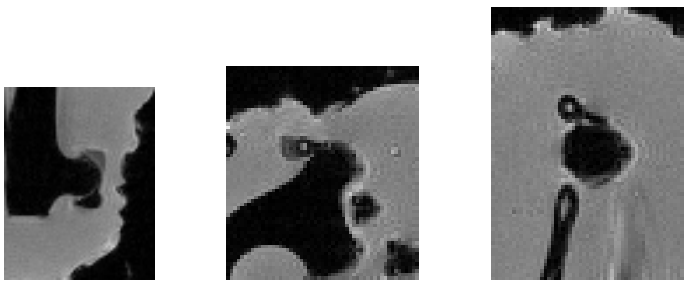
These transformations, datasets and optimizer settings define the experimental context used in the subsequent sections of this chapter. The following section begins the investigation into optimizer performance by analyzing methods of approximating the Hessian matrix in image registration problems.



(a) Cross sections through deformable phantom volume: Fully Inflated case



(b) Cross sections through deformable phantom volume: Partially inflated case



(c) Cross sections through deformable phantom volume: Deflated case

**Figure 4.7** Deformable phantom volumes.

| Algorithm | Criterion | Value | Reason |
|---|---|---|---|
| Downhill Simplex | $\phi$ tolerance | 0.01 | precision |
| | $D$ tolerance | 0.0001 | $p^2$ based on [85] |
| | Start simplex size | 1 | Similar to T-R |
| Powell | $\phi$ tolerance | 0.01 | precision |
| | $D$ tolerance | 0.0001 | $D_{tol} = p^2$ based on [85] |
| | Step size | 1 | Similar to T-R |
| BFGS | $D$ tolerance | 0.0001 | $D_{tol} = p^2$ based on [85] |
| | Minimum $\|\|\nabla D\|\|$ | 0.0001 | $\nabla D_{tol} = p^2$ based on [85] |
| | Minimum Step Size | 0.005 | $p/2$ |
| T-R Grad. Descent | Minimum $\|\|\nabla D\|\|$ | 0.0001 | $\nabla D_{tol} = p^2$ based on [85] |
| | Starting T-R | 1 | a safe step size |
| | Minimum Step Size | 0.005 | $p/2$ |
| Newton-Raphson | Minimum $\|\|\nabla D\|\|$ | 0.0001 | $\nabla D_{tol} = p^2$ based on [85] |
| | Starting T-R | 1 | a safe step size |

**Table 4.4** Settings and justifications for optimizer parameters. A nominal precision of $p = 0.01$ was used as a guideline for parameter setting.

## 4.3 Approximating the Hessian

The Hessian, or an approximation of it, is required for Newton-Raphson type optimizers. Most image registration applications in computer vision use a squared-difference cost function and form a Gauss-Newton approximation of its Hessian [136, 181, 208]. However, MSD is only applicable to a narrow range of image registration problems as it requires the assumption that intensities are identical in the images being matched. The NCC and MI image difference measures are more relevant when this assumption does not hold. Mutual information in particular is not a least-squares function, and calculating its Hessian is significantly more difficult than for MSD. This section examines the Gauss-Newton approximation for the squared difference case, and then shows how its fundamental insight can be extended to other cost functions.

### 4.3.1 The Gauss-Newton Approximation

The Gauss-Newton approximation of the Hessian is a method for applying the Newton-Raphson optimization to least-squares problems, *i.e.,* problems formulated as the

minimization of a sum of squared errors,

$$\boldsymbol{\phi}_{opt} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}} \left( D = [l - f(\boldsymbol{\phi})]^T \boldsymbol{Z} [l - f(\boldsymbol{\phi})]^T \right), \tag{4.2}$$

where $l$ is a vector of observations, and $f$ is the predictions of those observations from the model. The central term $\boldsymbol{Z}$ is the weight matrix, which is often simply the identity matrix, but may also be the inverse covariance matrix of the observations, if that is available. In an image registration context, the vector of observations corresponds to the pixels in the image, and the overall cost function is usually weighted by the number of pixels in the image to give the MSD image difference measure:

$$D_{MSD}(\boldsymbol{\phi}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \boldsymbol{Z}_{ij} (I_f(\boldsymbol{X}_i) - I_m(\boldsymbol{W}(\mathbf{x}_j, \boldsymbol{\phi}))^2. \tag{4.3}$$

This is equivalent to Equation 3.1 except for the addition of a weight matrix, $\boldsymbol{Z}$. This weight matrix, $\boldsymbol{Z}$, is nearly always the identity in an image registration context, but it is informative for the following discussion so it will be left in the derivation. The derivative of this cost function is given by

$$\nabla_{\boldsymbol{\phi}} D_{MSD}(\boldsymbol{\phi}) = \frac{-2}{N} \left[ (I_f(\boldsymbol{X}) - I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))) \right] \cdot \boldsymbol{Z} \cdot \frac{\partial I_m(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}} \tag{4.4}$$

and its Hessian:

$$\frac{\partial^2 D_{MSD}(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}^2} = \frac{2}{N} \frac{\partial I_m(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}}^T \cdot \boldsymbol{Z} \cdot \frac{\partial I_m(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}} - \frac{2}{N} \left[ I_f - I_m(\boldsymbol{\phi}) \right]^T \cdot \boldsymbol{Z} \cdot \frac{\partial^2 I_m(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}^2} \tag{4.5}$$

The Gauss-Newton approximation drops the second term, on the grounds (or perhaps in the hopes) that it is small. More will be said about the size of this term later.

The Gauss-Newton method has a nice visual interpretation [173]. Imagine each image as a point in the (high-dimensional) space of images. The point representing the moving image can be shifted in this space by changing the transformation parameters. The image registration problem is to move this point to minimize the distance between it and the point representing the fixed image. When the cost function is (weighted) mean squares, this distance corresponds to the usual (weighted) Euclidean distance,

as shown in Figure 4.8.

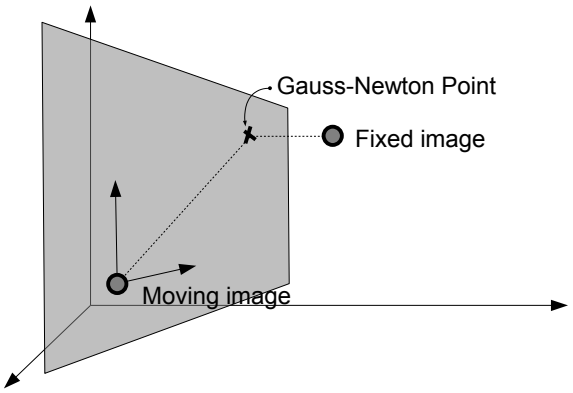



**Figure 4.8** The Gauss-Newton step. The derivative of the moving image with respect to the transformation parameters creates a hyperplane in the space of possible images. The Gauss-Newton step is the point on this hyperplane closest in a Euclidean sense to the fixed image.

The derivative of the moving image with respect to the $N$ transformation parameters creates a $N$-dimensional hyperplane in the image space. This hyperplane represents the expected appearance change given a small change in the parameters. In other words, it is the tangent to the appearance manifold defined by the transformation parameters. When the cost function is the mean squared difference between pixel values, the Gauss-Newton step is the point on this hyperplane that is closest to the fixed image. The key reason the Gauss-Newton method is effective in image registration is that the hyperplane induced by the derivative is a reasonable approximation of how the actual image warps due to the transformation.

### 4.3.2 How Good is the Gauss-Newton Approximation?

The validity of the Gauss-Newton approximation has not really been explored in the context of image registration, although its successful application (*e.g.,* [136, 172, 181]) suggests that it works well. Some estimate of the relative weights between the two terms in Equation 4.5 can be derived by considering that the first term is always positive semi-definite, while the second term may be positive definite, negative definite, or indefinite. Thus the first term will tend to grow proportional to the number of elements in $\boldsymbol{I}$. Assuming that the second term is not biased in some way, then it will

remain roughly the same size, or grow slowly as the size of $\boldsymbol{I}$ increases. Therefore, in general, the Gauss-Newton approximation is better the larger the number of elements in $\boldsymbol{I}$. It is important to note, however, that this argument depends on the assumption of no bias in the second term – if this is not true, the approximation may be poor regardless of the size of $\boldsymbol{I}$.

The best way to test the approximation may be by empirical means. This is fairly impractical in the true image registration case. In addition to time required for computation, the second term involves second derivatives of an interpolated image, thus image noise and interpolation artifacts would most likely make the results uninterpretable. However, it is feasible to examine the accuracy of the Gauss-Newton approximation using a proxy problem which can be calculated exactly.

Consider the 1D problem of aligning two signals, $f(x)$ and $g(W(x,p))$, where $W(x,p)$ is simply $x+p$. Thus $p$ is the single parameter of the very simple warp, $W$. Two one-dimensional "images" will be created by sampling at integral $x$ positions from $0-200$. Given these exact expressions for $g$, and $f$, $D_{MSD}$, its exact first and second derivatives, and the Gauss-Newton approximation to its second derivative can all be computed. Each of these can be used to create an approximate version of the function which can be compared, both visually and numerically, to the real cost function. Specifically, the image functions $f, g$ will be stretched sine waves:

$$f(x) = \sin\sqrt{x+200}; \qquad g(x,p) = \sin\sqrt{x+p}; \qquad x > 0 \tag{4.6}$$

These are shown for five different positions of the "warp" parameter, $p$, in Figure 4.9.

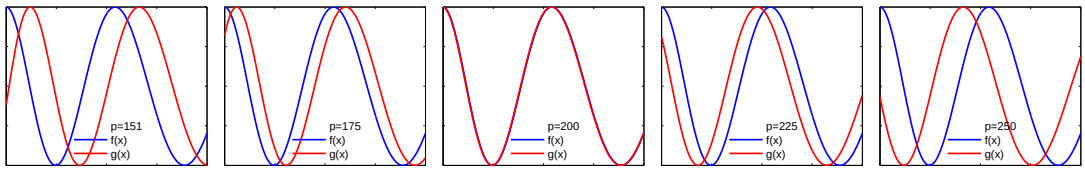
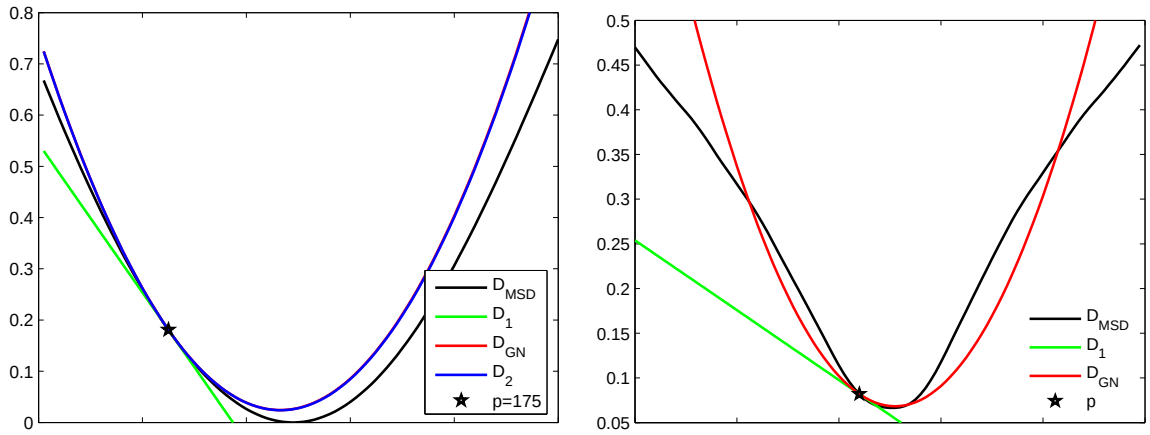



**Figure 4.9** The simple functions used for testing the MSD Gauss-Newton approximation. The fixed and moving one-dimensional images are shown for values of the parameter, $p$, ranging from 151 to 250. Correct registration of the functions occurs at $p = 200$.




(a) Calculated (b) Real

**Figure 4.10** The actual function, $D_{MSD}$, the full Taylor series approximation, $D_2$, the Gauss-Newton approximation, $D_{GN}$, and a linear approximation, $D_1$, for Mean Squared Difference. The calculated case (left) shows the comparison to a full, exactly computed Taylor series. The real case (right) shows the estimate of the cost function computed using the Gauss-Newton Hessian approximation for a real image.

The three approximations that will be compared are the linear approximation due to the gradient alone

$$D_{MSD_1}(p) = D(p_0) + (p - p_0)\frac{\partial D_{MSD}}{\partial g}\frac{\partial g}{\partial p}, \tag{4.7}$$

the Gauss-Newton approximation,

$$D_{MSD_{GN}}(p) = D_{MSD_1}(p) + 0.5(p - p_0)\frac{\partial g}{\partial p}^T\frac{\partial g}{\partial p}(p - p_0), \tag{4.8}$$

and the complete second order Taylor series approximation

$$D_{MSD_2}(p) = D_{MSD_{GN}}(p) + 0.5(p - p_0)\frac{\partial D_{MSD}}{\partial g}\frac{\partial^2 g}{\partial p^2}(p - p_0). \tag{4.9}$$

These are shown in Figure 4.10a. The Gauss-Newton approximation and the complete second order Taylor series approximation are visually nearly indistinguishable. As a numerical comparison, the average absolute percentage difference (over the entire

graphed range) between the estimates of the function three samples away using each method is used:

$$err = \frac{\sum_{i=3}^{N} \left[ |\hat{f}_{i-3}(p_i) - f(p_i)| + |\hat{f}_i(p_{i-3}) - f(p_{i-3})| \right]}{2 \sum_{i=3}^{N} |f(p_i) - f(p_{i-3})|} \tag{4.10}$$

where $\hat{f}_i$ is the approximation of the function centered on $p_i$. The Gauss-Newton approximation yields an error of 1.84%, while using the full Hessian yields an error of 1.82%. Both are clearly superior to the linear approximation, which yields an error of 5.43%. Clearly both estimate the function well, and the difference between them is minimal.

A visual example of the real image difference measure is shown in Figure 4.10b. This shows a plot of $D_{MSD}$ for the image in Figure 4.1 and a transformed version of itself. The function is computed along a line in the parameter space of 2D rigid transforms. At the indicated point in the curve the derivative and Gauss-Newton approximate Hessian have been collected and used to derive an approximation of the cost function. The approximation to the function based on the calculated Gauss-Newton Hessian is a good approximation of the real function.

## 4.4 Generalizing the Gauss-Newton Approximation

The Gauss-Newton approximation is very useful for least squares problems, but Hessian approximations are also necessary for the case of cost functions that are not least squares. Making an ad-hoc approximation without careful justification can lead to a poor Hessian approximation, which will tend to hinder, rather than help, the optimization performance.

Maes *et al.* [137] proposed very simple approach to generalizing this approximation, based on [158, Chap. 15]. They consider the cost function to be a single equation measuring the goodness of a model, and then apply least squares minimization to this model. That is, they minimize $D^2$, which has a derivative,

$$\frac{\partial D^2}{\partial \boldsymbol{\phi}} = 2 \cdot D \cdot \frac{\partial D}{\partial \boldsymbol{\phi}} \tag{4.11}$$

and a Hessian

$$\frac{\partial^2 D^2}{\partial \boldsymbol{\phi}^2} = 2 \cdot \frac{\partial D}{\partial \boldsymbol{\phi}}^T \frac{\partial D}{\partial \boldsymbol{\phi}} + 2 \cdot D \cdot \frac{\partial^2 D}{\partial \boldsymbol{\phi}^2} \tag{4.12}$$

The "Gauss-Newton" approximation, in this case, is to drop the second term in Equation 4.12.

Unfortunately, this approximation cannot be justified, as it is highly unlikely that the second term is small. Consider that the truncated term in the approximation above is $2D\frac{\partial^2 D}{\partial \boldsymbol{\phi}^2}$. This approximation has truncated *all* the second derivatives of the original cost function and thus is roughly equivalent to gradient descent on the original function.

To apply the Gauss-Newton method to other cost functions, reconsider Equation 4.3. The general case is to consider an arbitrary cost measure, $D$, which is a function of a vector valued function, $\boldsymbol{I}$, which is, in turn, a function of a vector of model parameters, $\boldsymbol{\phi}$.

$$D = D(\boldsymbol{I}(\boldsymbol{\phi})) \tag{4.13}$$

This can be expanded to second order as a Taylor series as follows

$$\begin{aligned} D(\boldsymbol{I}(\boldsymbol{\phi})) \approx D(\boldsymbol{I}(\boldsymbol{\phi}_0)) + (\boldsymbol{\phi} - \boldsymbol{\phi}_0)\frac{\partial D}{\partial \boldsymbol{I}} \cdot \frac{\partial \boldsymbol{I}}{\partial \boldsymbol{\phi}} \\ +0.5(\boldsymbol{\phi} - \boldsymbol{\phi}_0)\frac{\partial \boldsymbol{I}}{\partial \boldsymbol{\phi}}\frac{\partial^2 D}{\partial \boldsymbol{I}^2} \cdot \frac{\partial \boldsymbol{I}}{\partial \boldsymbol{\phi}}(\boldsymbol{\phi} - \boldsymbol{\phi}_0) \\ +0.5(\boldsymbol{\phi} - \boldsymbol{\phi}_0)\frac{\partial D}{\partial \boldsymbol{I}} \cdot \frac{\partial^2 \boldsymbol{I}}{\partial \boldsymbol{\phi}^2}(\boldsymbol{\phi} - \boldsymbol{\phi}_0) + ... \end{aligned} \tag{4.14}$$

Observe that because the function being approximated is a composition of functions, there are two second order terms in its Taylor series. The Gauss-Newton approximation is to truncate the series one term earlier than a full second order expansion. This idea can be easily generalized to any functions of the form $D(\boldsymbol{I}(\boldsymbol{\phi}))$. Note however, that in the general case, the Gauss-Newton approximation to the Hessian is more complex. The weight matrix, $\boldsymbol{Z}$, in the least squares case corresponds to $\frac{\partial^2 D}{\partial \boldsymbol{I}^2}$ in Equation 4.14.

It is therefore necessary to look closely at the matrix $\frac{\partial^2 D}{\partial \boldsymbol{I}^2}$. For the image registration problem, taken literally, it would be a symmetric matrix the size of the number of pixels. Evaluating Equation 4.14 would require a summation of the outer products

between the derivatives of every pixel with every other pixel. This is clearly unrealistic to compute. However, for most image difference measures, an intuitive understanding suggests that pixels with the same intensity are equivalent, as far as the measure, $D$, is concerned. Therefore, this matrix should not need to be computed in its full generality for most measures. In the following it is shown that for two commonly used measures, NCC and MI, the calculation is indeed more tractable.

### 4.4.1 Normalized Correlation

Recall from Section 3.1.3 that normalized correlation NCC is appropriate for comparing images where there is a linear relationship between the intensities in the fixed and moving images. The $D_{NCC}$ measure is a ratio of two terms

$$D_{NCC} = \frac{-u}{v} \tag{4.15}$$

where

$$u = \sum_i \left[ (\boldsymbol{I}_{f_i} - \bar{I}_f)(\boldsymbol{I}_{m_i} - \bar{I}_m) \right],$$

and

$$v = \left[ \sum_i \left[ (\boldsymbol{I}_{f_i} - \bar{I}_f)^2 \right] \sum_j \left[ (\boldsymbol{I}_{m_j} - \bar{I}_m)^2 \right] \right]^{\frac{1}{2}}.$$

where $\bar{I}_f$ and $\bar{I}_m$ are the mean intensities for the fixed and moving images, respectively. The gradient is then

$$\frac{\partial D_{NCC}}{\partial \boldsymbol{\phi}} = \frac{1}{v^2} \left( u \frac{\partial v}{\partial \boldsymbol{I}_m} - v \frac{\partial u}{\partial \boldsymbol{I}_m} \right) \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}$$

Temporarily defining $\boldsymbol{A} = u \frac{\partial v}{\partial \boldsymbol{I}_m}$, $\boldsymbol{B} = v \frac{\partial u}{\partial \boldsymbol{I}_m}$, and $C = v^2$, the Hessian can be expressed as:

$$\begin{aligned} \frac{\partial^2 D_{NCC}}{\partial \boldsymbol{\phi}^2} = \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}^T \left( \frac{C \frac{\partial \boldsymbol{A}}{\partial \boldsymbol{I}_m} + \boldsymbol{B} \frac{\partial C}{\partial \boldsymbol{I}_m} - \boldsymbol{A} \frac{\partial C}{\partial \boldsymbol{I}_m} - C \frac{\partial \boldsymbol{B}}{\partial \boldsymbol{I}_m}}{C^2} \right) \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} + \\ \frac{1}{v^2} \left( u \frac{\partial v}{\partial \boldsymbol{I}_m} - v \frac{\partial u}{\partial \boldsymbol{I}_m} \right) \frac{\partial^2 \boldsymbol{I}_m}{\partial \boldsymbol{\phi}^2} \end{aligned} \tag{4.16}$$

The second term of Equation 4.16 is equivalent to what is dropped in the Gauss-Newton approximation for mean squared difference (Equation 4.5). For the same reasons as before, this term can be assumed small and dropped. To complete the details of the Hessian each of the parts of the first term must be considered.

Assuming that the mean values of the image, $\bar{I}_m$, do not appreciably change as the parameters change, the derivatives of $u$ and $v$ with respect to a particular pixel in the moving image, $\boldsymbol{I}_{m_i}$, are:

$$\frac{\partial u}{\partial \boldsymbol{I}_{m_i}} = (\boldsymbol{I}_{f_i} - \bar{I}_f),$$

and

$$\frac{\partial v}{\partial \boldsymbol{I}_{m_i}} = \frac{1}{v} \sum_k \left[ (\boldsymbol{I}_{f_k} - \bar{I}_f)^2 \right] (\boldsymbol{I}_{m_i} - \bar{I}_m)$$

Letting $F = \sum_k \left[ (\boldsymbol{I}_{f_k} - \bar{I}_f)^2 \right]$, and noting that it is a constant,

$$\boldsymbol{A} = \frac{u}{v} \cdot F \cdot (\boldsymbol{I}_{m_i} - \bar{I}_m),$$

and,

$$\frac{\partial \boldsymbol{A}}{\partial \boldsymbol{I}_{m_j}} = \frac{F}{v}(\boldsymbol{I}_{f_j} - \bar{I}_f)(\boldsymbol{I}_{m_i} - \bar{I}_m) - \frac{uF^2}{v^3}(\boldsymbol{I}_{m_j} - \bar{I}_m)(\boldsymbol{I}_{m_i} - \bar{I}_m) + \frac{uF}{v}\delta_{ij},$$

where $\delta_{ij}$ is an indicator function, equal to 1 if $i = j$ and 0 otherwise. Continuing,

$$\boldsymbol{B} = v \cdot (\boldsymbol{I}_{f_i} - \bar{I}_f),$$

$$\frac{\partial \boldsymbol{B}}{\partial \boldsymbol{I}_{m_j}} = \frac{F}{v} \cdot (\boldsymbol{I}_{f_i} - \bar{I}_f) \cdot (\boldsymbol{I}_{m_j} - \bar{I}_m),$$

and,

$$\frac{\partial C}{\partial \boldsymbol{I}_{m_j}} = 2F \cdot (\boldsymbol{I}_{m_j} - \bar{I}_m)$$

Combining the terms and simplifying, Equation 4.17 is obtained for the Gauss-Newton Hessian of $D_{NCC}$, which will be referred to as $\mathcal{H}_{GN-2}D_{NCC}$.

$$\mathcal{H}_{\boldsymbol{\phi}} D_{NCC} \approx \mathcal{H}_{GN-2} D_{NCC} =$$
$$\frac{2 \cdot \sum_i \left[(\boldsymbol{I}_{f_i} - \bar{I}_f)^2\right]}{v^3} \cdot \sum_i \left[(\boldsymbol{I}_{f_i} - \bar{I}_f) \cdot \nabla_{\boldsymbol{\phi}} \boldsymbol{I}_{m_i}\right] \cdot \sum_j \left[(\boldsymbol{I}_{m_j} - \bar{I}_m) \cdot \nabla_{\boldsymbol{\phi}} \boldsymbol{I}_{m_j}\right]^T$$
$$- \frac{3u \cdot \left[\sum_i \left[(\boldsymbol{I}_{f_i} - \bar{I}_f)^2\right]\right]^2}{v^5} \cdot \sum_i \left[(\boldsymbol{I}_{m_i} - \bar{I}_m) \cdot \nabla_{\boldsymbol{\phi}} \boldsymbol{I}_{m_i}\right] \cdot \sum_j \left[(\boldsymbol{I}_{m_j} - \bar{I}_m) \cdot \nabla_{\boldsymbol{\phi}} \boldsymbol{I}_{m_j}\right]^T$$
$$+ \frac{u \cdot \sum_i \left[(\boldsymbol{I}_{f_i} - \bar{I}_f)^2\right]}{v^3} \cdot \sum_i \left[\nabla_{\boldsymbol{\phi}} \boldsymbol{I}_{m_i} \cdot \nabla_{\boldsymbol{\phi}} \boldsymbol{I}_{m_j}^T\right] \tag{4.17}$$

This Hessian is lengthy to write, but tractable to compute. The first two terms (blue) of Equation 4.17 can be computed from summations over weighted gradients, and the third term (red) is the weighted summation of the outer product of $\frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}$ with itself. Each of the first two terms is arguably very small since they include summations of the form $\sum_i (\boldsymbol{I}_i - \bar{I})$ which should average to zero. This suggests that these terms can be ignored without effect, which yields another approximation that will be referred to as

$$\mathcal{H}_{\boldsymbol{\phi}} D_{NCC} \approx \mathcal{H}_{GN-1} D_{NCC} = \frac{u \cdot \sum_i \left[(\boldsymbol{I}_{f_i} - \bar{I}_f)^2\right]}{v^3} \cdot \sum_i \left[\nabla_{\boldsymbol{\phi}} \boldsymbol{I}_{m_i} \cdot \nabla_{\boldsymbol{\phi}} \boldsymbol{I}_{m_j}^T\right] \tag{4.18}$$

Both approximations will be examined in the following experiments.

As in the case of MSD, these Gauss-Newton approximations can be tested on a function that can be computed exactly. Normalized Correlation is intended to be used when there are linear intensity differences between the functions, so to simulate that, $f(x)$ is changed to:

$$f(x) = 0.7 \sin \sqrt{x + 200} + 1.15; \tag{4.19}$$

and proceed as before. This new function is shown for five different positions of the "warp" parameter, $p$, in Figure 4.11.

The results are shown visually in Figure 4.12a. A numerical examination over the entire graphed range, using Equation 4.10 gives the result that the approximate function based on the simple Gauss-Newton approximation to the Hessian ($\mathcal{H}_{GN-1} D_{NCC}(\boldsymbol{\phi})$, Equation 4.18) has an error of 0.51%, the approximate function

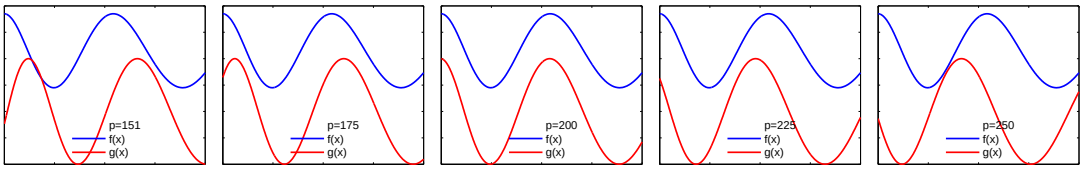
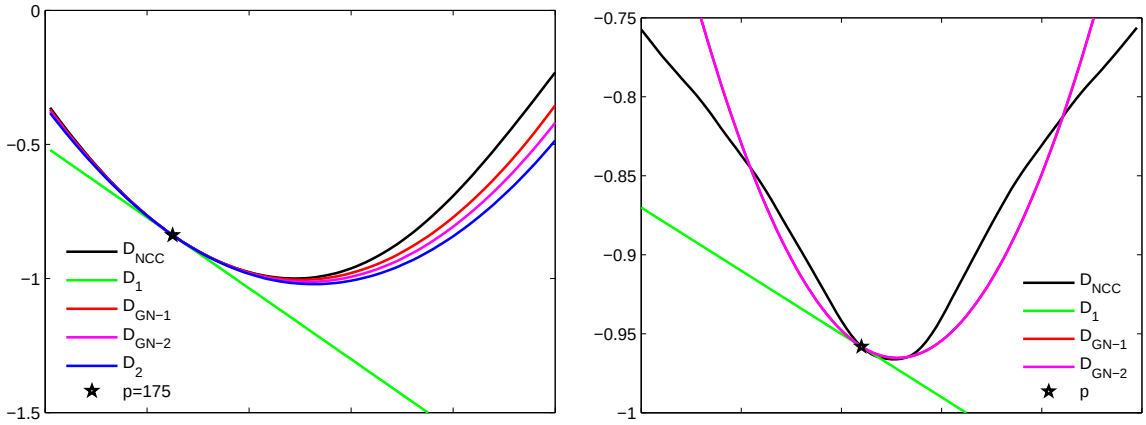



**Figure 4.11** The simple functions used for testing the NCC Gauss-Newton approximation. The fixed and moving one-dimensional images are shown for values of the parameter, $p$, ranging from 151 to 250. Correct registration of the functions occurs at $p = 200$.

based on the full Gauss-Newton approximation to the Hessian ($\mathcal{H}_{GN-2}D_{NCC}(\boldsymbol{\phi})$, Equation 4.17) has an error of 0.88% and the full second order Taylor series approximation has an error of 0.97%. Obviously all three are very close approximations to $D_{NCC}$. It may seem counterintuitive that a shorter truncation of the Taylor series gives more accurate results, but this simply means that there are third order terms which tend to cancel the truncated terms. Over the same distance, approximating the function using only the gradient yielded an error of 5.6% with Equation 4.10. From this it can be concluded that the Gauss-Newton Hessian approximation forms very nearly as good a local approximation to the function as the full Hessian and can be used as input to Newton-Raphson algorithms. An even simpler approximation (GN-1) made by dropping some small terms appears to be equally as good or better.

The values of $D_{NCC}$ were computed between the images shown in Figure 4.1a and 4.1b along a line through the 2D rigid parameter space. The gradient, and each of the two Gauss-Newton approximation were collected at the point shown and used to approximate the function. The image difference measure and these approximations are shown in Figure 4.12b. Note that the two approximations are nearly indistinguishable from each other, and both approximate the real $D_{NCC}$ well.

To test the effectiveness of these Hessian approximations two simulated registration experiments using the 2D TPS and 2D B-spline transforms were performed. The sagittal brain image warped by the synthetic 2D deformation was used (Figures 4.1 and 4.4b). No scaling factor was used. Figure 4.13 graphically reports the results of these registrations in terms of running time, mTRE, number of function evaluations and failure rate. These were tested for statistical significance using the framework




(a) Calculated (b) Real

**Figure 4.12** The actual function, $D_{NCC}$, the full second order Taylor series approximation, $D_2$, the approximation based on the complete Gauss-Newton Hessian approximation, $D_{GN-2}$, the approximation based on the simplified Gauss-Newton Hessian approximation, $D_{GN-1}$, and a linear approximation, $D_1$, for Normalized Correlation. The calculated case (left) shows the comparison to a full, exactly computed Taylor series. The real case (right) shows the Gauss-Newton Hessian approximations for a real image.

described in Section 3.2.1.

As the analysis suggested would be the case, no statistically significant difference between the two approaches to approximating the Hessian was found. A Newton-Raphson optimization using either of these Hessian approximations used significantly fewer iterations than a gradient descent optimization. As there is no detectable difference between the two approaches, it is concluded that the simpler of the two should be used. This approximation is used in all the following experiments that require a Hessian. Note that in the B-spline case, the gradient descent method was faster than the Newton method, despite requiring more iterations. This is due to the computational cost of computing the Hessian, an issue that will be discussed in more detail in Section 4.6.1.

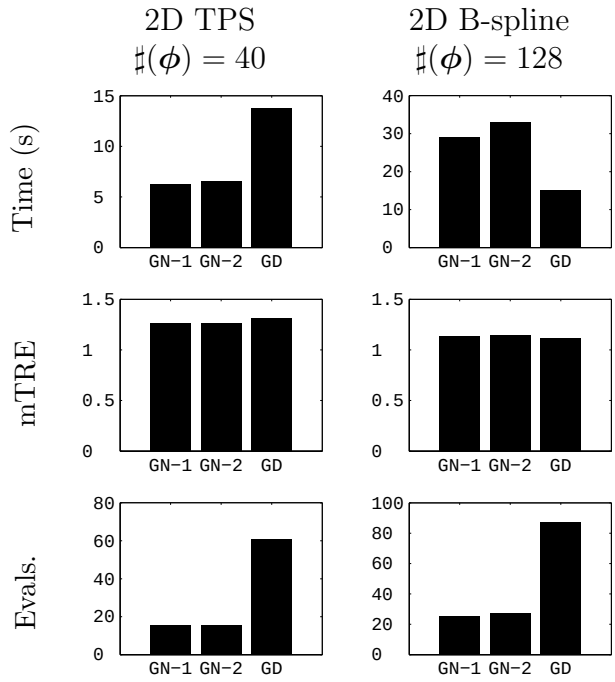



**Figure 4.13** Registration performance using a Newton-Raphson optimization technique with both approximate Hessian calculation methods for the NCC measure. Results are also compared against using gradient descent (GD). The two Hessian approximations, $\mathcal{H}_{GN-1}D_{NCC}$ and $\mathcal{H}_{GN-2}D_{NCC}$, give nearly identical results, so the simpler one $\mathcal{H}_{GN-1}D_{NCC}$ is preferred. (The mTRE is given in mm.)

### 4.4.2 Mutual Information

As previously discussed in Section 3.1.4 mutual information measures the existence of a statistical relationship between two signals, and is particularly useful for multi-modality image registration.[1] Recall that mutual information is a function of a joint probability distribution, $\boldsymbol{P}$, of the intensities in the images. In addition to the expression of Equation 3.6, (negative) mutual information can be expressed as a sum of entropies, $D_{MI} = H(\boldsymbol{P}_{f,m}) - H(\boldsymbol{P}_f) - H(\boldsymbol{P}_m)$, where $H(\boldsymbol{P}_i)$ is the Shannon entropy, $H(\boldsymbol{P}_i) = -\sum_i \boldsymbol{P}_i \log \boldsymbol{P}_i$ [61].

For brevity, the Gauss-Newton approximation for the joint entropy will be derived. The approximate Hessian of MI can then be evaluated by adding up the Hessians of its component entropies. The entropy of a probability distribution where each element, $\boldsymbol{P}_k$, is a function of the moving image, $\boldsymbol{I}_m$, which is in turn a function of some parameters, $\boldsymbol{\phi}$, is given by,

$$H(\boldsymbol{\phi}) = -\sum_k \boldsymbol{P}_k(\boldsymbol{I}_m(\boldsymbol{\phi})) \log(\boldsymbol{P}_k(\boldsymbol{I}_m(\boldsymbol{\phi}))), \tag{4.20}$$

and its derivative is then,

$$\frac{\partial H}{\partial \boldsymbol{\phi}} = -\sum_k \left[ \log(\boldsymbol{P}_k) \frac{\partial \boldsymbol{P}_k}{\partial \boldsymbol{I}_m} \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} + \boldsymbol{P}_k \frac{\partial \log(\boldsymbol{P}_k)}{\partial \boldsymbol{I}_m} \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \right] \tag{4.21}$$

$$\frac{\partial H}{\partial \boldsymbol{\phi}} = -\sum_k \left[ \log(\boldsymbol{P}_k) \frac{\partial \boldsymbol{P}_k}{\partial \boldsymbol{I}_m} \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} + \frac{\partial \boldsymbol{P}_k}{\partial \boldsymbol{I}_m} \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \right] \tag{4.22}$$

Since the sum of the probability distribution, $\boldsymbol{P}_k$, must be 1, the sum of the derivatives of $\boldsymbol{P}_k$ must be zero, and therefore the second term disappears. Taking the second derivative

$$\begin{aligned} \frac{\partial^2 H(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}^2} = -\sum_k \Bigg[ & \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \frac{\partial \boldsymbol{P}_k}{\partial \boldsymbol{I}_m} \frac{1}{\boldsymbol{P}_k} \frac{\partial \boldsymbol{P}_k}{\partial \boldsymbol{I}_m} \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} + \\ & \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \frac{\partial^2 \boldsymbol{P}_k}{\partial \boldsymbol{I}_m^2} \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \log(\boldsymbol{P}_k) + \log(\boldsymbol{P}_k) \frac{\partial \boldsymbol{P}_k}{\partial I_m} \frac{\partial^2 \boldsymbol{I}_m}{\partial \boldsymbol{\phi}^2} \Bigg] \end{aligned} \tag{4.23}$$

[1] To be consistent with the discussion of minimization, the negative mutual information is described here.

Both [185] and [70] approximate the Hessian by dropping the second (blue) and third (red) terms of Equation 4.23. Adding the entropies for the joint and moving image probability distributions, their approximation to the Hessian of MI is:

$$\begin{aligned}\frac{\partial^2 D_{MI}(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}^2} \approx \mathcal{H}_{TU} D_{MI} = &- \sum_{f,m} \left[ \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \frac{\partial \boldsymbol{P}_{f,m}}{\partial \boldsymbol{I}_m} \frac{1}{\boldsymbol{P}_{f,m}} \frac{\partial \boldsymbol{P}_k}{\partial \boldsymbol{I}_m} \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \right] \\ &+ \sum_{m} \left[ \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \frac{\partial \boldsymbol{P}_m}{\partial \boldsymbol{I}_m} \frac{1}{\boldsymbol{P}_m} \frac{\partial \boldsymbol{P}_m}{\partial \boldsymbol{I}_m} \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \right]\end{aligned} \tag{4.24}$$

(Note that the fixed image intensity distribution may be considered fixed in the B-spline Parzen windowed formulation.) This approximation will be called $\mathcal{H}_{TU} D_{MI}$ since Thévenaz and Unser first proposed it [185, 186].

This second (blue) term of Equation 4.23 is actually quite large, and its importance can be understood using the following argument. The entropy of the fixed image can be considered to remain fixed, and the entropy of the moving image can be considered to remain very nearly fixed, at least locally. Therefore the shape of the mutual information cost function is almost completely determined by the shape of the joint entropy function. Noting that an outer product of a matrix with itself is always positive definite, and that since all the entries of $\boldsymbol{P}$ are less than one, and therefore their logarithms are all negative, the first term in Equation 4.23 (black) is a negative definite matrix, and the second term (blue) is positive definite. The negative mutual information cost function, however, forms a non-convex funnel shape – *i.e.,* its Hessian must be saddle shaped (except possibly very close to the optimum). Therefore neither of these terms alone can accurately describe the shape of the function. This is not simply a problem of scaling, but of the reversal of curvature in at least one direction if one of these terms is ignored.

It is argued in [70, 185] that this second term is unimportant because it disappears when identical images are in exact registration. However, this argument is flawed. Consider that for the entropy calculation, bins of the probability distribution which contain zero contribute nothing to the total entropy, nor to its derivatives. They may therefore be ignored. Then when two identical images are in perfect registration there is thus an exact bin to bin match between the joint probability distribution, and the moving image intensity probability distribution. In this situation, the entropy of the

joint distribution cancels exactly with the entropy of the moving image distribution, *and so do all its derivatives.* The Hessian would be a zero matrix. The MI function, in this particular case, becomes a sharp point at perfect registration, and the derivatives at the point are not well defined.

Fortunately, the derivative does not usually disappear at the minimum for MI image registration problems. A number of factors, including the blurring caused by the B-spline Parzen windows, differences in intensity distributions between the images and noise in either image prevent the joint probability distribution from precisely matching the moving image intensity distribution in practice. However, these factors also prevent the second term from disappearing, so the argument still should not be used.

The approximation that is truly equivalent to the one that is made in the squared difference case is to drop only the third (red) term of Equation 4.23. Adding the entropies of the joint and moving image intensities gives:

$$\begin{aligned}\mathcal{H}_{GN}D_{MI} = &-\sum_{f,m}\left[\frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}\frac{\partial \boldsymbol{P}_{f,m}}{\partial \boldsymbol{I}_m}\frac{1}{\boldsymbol{P}_{f,m}}\frac{\partial \boldsymbol{P}_{f,m}}{\partial \boldsymbol{I}_m}\frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} + \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}\frac{\partial^2 \boldsymbol{P}_{f,m}}{\partial I_m^2}\frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}\log(\boldsymbol{P}_{f,m})\right] \\ &+\sum_{m}\left[\frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}\frac{\partial \boldsymbol{P}_m}{\partial I_m}\frac{1}{\boldsymbol{P}_m}\frac{\partial \boldsymbol{P}_m}{\partial \boldsymbol{I}_m}\frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} + \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}\frac{\partial^2 \boldsymbol{P}_m}{\partial \boldsymbol{I}_m^2}\frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}\log(\boldsymbol{P}_m)\right]\end{aligned} \tag{4.25}$$

The second and fourth terms (blue, these correspond to the blue second term in Equation 4.23) are tractable, although somewhat difficult. Since $\boldsymbol{P}_{f,m}$ is not known during the loop over pixels, the $\frac{\partial \boldsymbol{P}_{f,m}}{\partial \boldsymbol{I}_m}$ component of the second term must be kept, and then reduced when $P$ is known. For a transformation with $\sharp(\boldsymbol{\phi})$ parameters, this is an object of size $|P| \times \sharp(\boldsymbol{\phi}) \times (\sharp(\boldsymbol{\phi}) + 1) \times 0.5$ (note that it is symmetric, so only a triangular matrix of terms are needed). For reasonably small numbers of parameters this is feasible, although it does make the MI approximate Hessian more computationally expensive than those for MSD and NCC.

As in the case of MSD, the MI Gauss-Newton Hessian approximation can be tested on a function that can be exactly computed. MI is intended to be used when there is a complex (*i.e.,* non-linear) intensity relationship between the images, so to simulate

that, $f(x)$ is changed to:

$$f(x) = (\sin\sqrt{x + 200})^3; \tag{4.26}$$

and proceed as before. This new function is shown for five different positions of the "warp" parameter, $p$, in Figure 4.14.

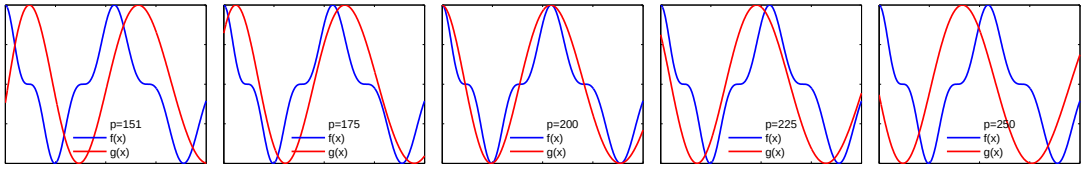
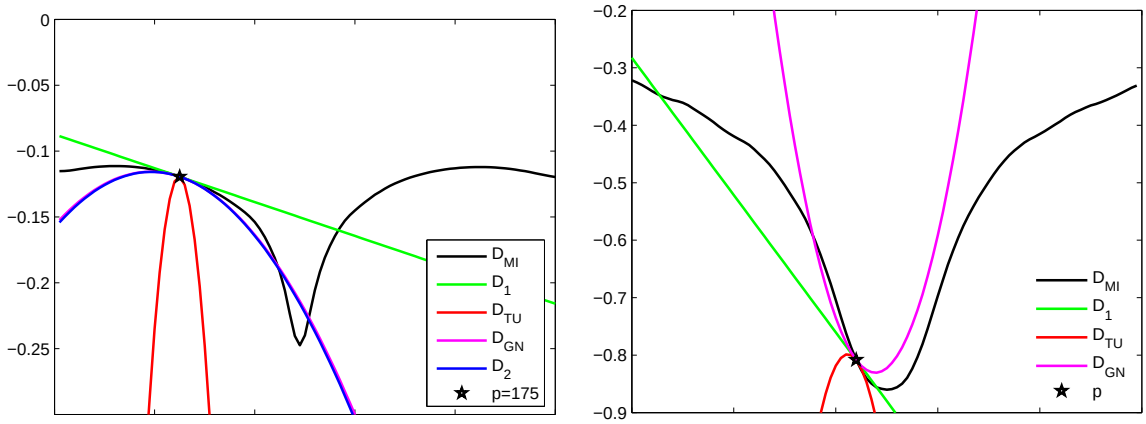



**Figure 4.14** The simple functions used for testing the MI Gauss-Newton approximation. The fixed and moving one-dimensional images are shown for values of the parameter, $p$, ranging from 151 to 250. Correct registration of the functions occurs at $p = 200$.

Both the Gauss-Newton approximation in Equation 4.25 and the one used in [70, 185], Equation 4.24 are tested. The results are shown visually in Figure 4.15a A numerical examination, using Equation 4.10, gives the result that the approximations using the Gauss-Newton approximate Hessian of Equation 4.25, $\mathcal{H}_{GN}D_{MI}$, and the full second order Taylor series have identical errors of 27.3%. While much larger than the NCC or MSD cases, this simply indicates that the 3rd order and higher terms in the function are quite large, and that very little is lost by making the Gauss-Newton approximation. The approximation using the gradient only has an error of 38.2%, but using $\mathcal{H}_{TU}D_{MI}$ has an approximation error of 548%.

The values of $D_{MI}$ were computed between the images shown in Figure 4.1a and 4.1c along a line through the 2D rigid parameter space. The gradient, and each of the two Gauss-Newton approximations were collected at the point shown and used to approximate the function. The image difference measure and these approximations are shown in Figure 4.15b. Note that the generalized Gauss-Newton approximation ($D_{GN}$, magenta) has positive curvature and approximates the MI function well. The shorter approximation ($D_{TU}$, red) has negative curvature and is a poor approximation to the function.

As in the case of the NCC measure, to test the effectiveness of these Hessian approximations two simulated registration experiments using the 2D TPS and 2D




(a) Calculated (b) Real

**Figure 4.15** The actual function, $D_{MI}$, and approximations using the full Taylor series approximation, $D_2$, generalized Gauss-Newton Hessian approximation, $D_{GN}$, the Thévenaz-Unser approximation to the Hessian, $D_{TU}$, and a linear approximation from the gradient, $D_1$. The calculated case (left) shows the comparison on a function whose Hessian can be calculated exactly. The real case (right) shows the approximations generated from a real image.

B-spline transforms were performed. The sagittal brain image warped by the synthetic 2D deformation was used (Figures 4.1 and 4.4b). No scaling factor was used. Figure 4.16 graphically reports the results of these registrations in terms of running time, mTRE, number of function evaluations and failure rate. These were tested for statistical significance using the framework described in Section 3.2.1.

There is a statistically significant difference in optimizer performance between the two Hessian approximations. Many fewer iterations are required when using the generalized Gauss-Newton approximation, $\mathcal{H}_{GN}D_{MI}$. Note the timing results carefully, however. Despite the fact that many fewer iterations were required with the generalized Gauss-Newton Hessian, it was not efficient to use it in these cases due to the computational cost of computing the Hessian. This issue is discussed more thoroughly in Section 4.6.

Using the alternative shorter truncation, $\mathcal{H}_{TU}D_{MI}$, the performance is not appreciably different from simply using gradient descent. This shorter truncation confers no detectable benefit on the optimization process. Therefore, it is concluded that the

generalized Gauss-Newton approach,, $\mathcal{H}_{GN}D_{MI}$, is the correct Hessian approximation to use in Newton-Raphson optimizers and it will be used in all further experiments.

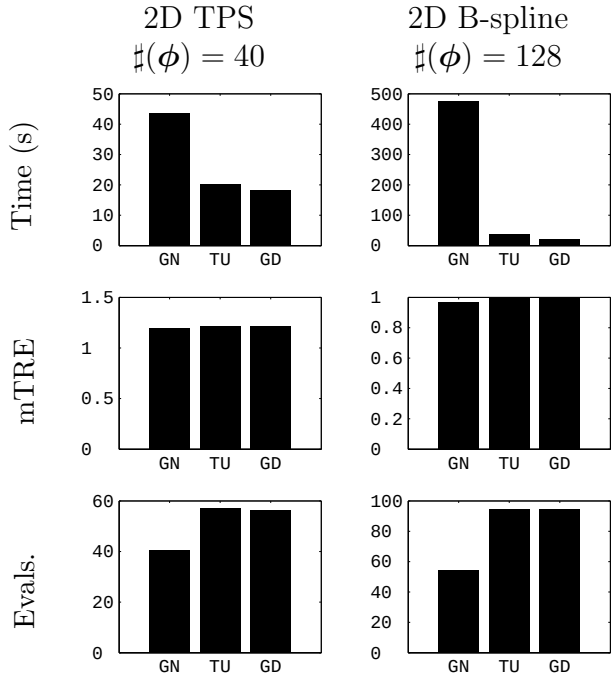



**Figure 4.16** Registration performance using a Newton-Raphson optimization technique with both approximate Hessian calculation methods for the MI measure. Results are also compared against using gradient descent. The two Hessian approximations, $\mathcal{H}_{GN}D_{MI}$ and $\mathcal{H}_{TU}D_{MI}$, give very different results. The shorter truncation of the Taylor series, $\mathcal{H}_{TU}D_{MI}$, does not result in significantly different performance from gradient descent. The longer, generalized Gauss-Newton approximation $\mathcal{H}_{GN}D_{MI}$ is preferred. (The mTRE is given in mm.)

### 4.4.3 Summary

In this section, a generalization of the Gauss-Newton Hessian approximation for non least squares functions has been derived. The generalization applies to any cost function that can be expressed as a composition of functions, $D(\boldsymbol{I}(\boldsymbol{\phi}))$, where the inner function $\boldsymbol{I}$ is a large dimensional vector. This approximate Hessian has been explicitly worked out for the NCC and MI cost functions, and shown to provide a local approximation to the cost function very nearly as good as the full second order Tay-

lor series. Image registration experiments confirm it is effective for Newton-Raphson optimization in image registration problems.

Recall from the introduction to this chapter that solutions to two issues were necessary before the performance of different optimization algorithms could be properly compared. The following section deals with the second of these issues, that of parameter scaling.

## 4.5 Parameter Scaling

To get the best performance from an optimization algorithm, it is generally recommended that the problem be "well-scaled" [59, 80, 85, 151, 192]. It is known that parameter scaling is important in image registration (*e.g.,* [26, 109, 194]), particularly for gradient descent optimization, but methods of choosing these scale factors have remained ad-hoc. This section examines what the effect of scaling is, and presents a new method for choosing an optimal scale factor.

There are two major reasons why parameter scaling is important. First, scaling the variables also affects the meaning of the stopping criteria. The stopping criteria are commonly based on the change in the parameters, $|\boldsymbol{\phi}_{(n)} - \boldsymbol{\phi}_{(n-1)}|$, the change in the objective function, $|D(\boldsymbol{\phi}_{(n)}) - D(\boldsymbol{\phi}_{(n-1)})|$, and or the gradient magnitude $||\nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}_{(n)})||$. If the problem is far more sensitive to one parameter than another, the final result will be nearly completely determined by the performance in that one parameter. For example, an optimization step of 0.1 pixels is fairly small and a registration algorithm could probably safely terminate when it reaches this step size. On the other hand, a step of 0.1 radians of rotation is quite large, and it would probably be premature to terminate the algorithm at this step size. This can lead to erratic performance, where the algorithm stops far too soon, or runs far too long depending on how well that one parameter is resolved in a particular instance. As a guideline, then, scaling should make the expected ranges of the parameters be approximately the same. In fact, Gill *et al.* [85] recommends that all the parameter values should be approximately one. If the natural expression of the problem does not meet these requirements, a linear rescaling can be applied to obtain better performance.

Second, and more importantly, scaling the problem variables affects the direction

of the gradient, which, not surprisingly, has a significant effect on gradient descent type algorithms. A linear rescaling of the parameters, $\boldsymbol{\phi}(\boldsymbol{\theta}) = \boldsymbol{S} \cdot \boldsymbol{\theta}$, where $\boldsymbol{S}$ is a positive definite scaling matrix, changes the optimization problem to be optimizing over a new set of parameters, $\boldsymbol{\theta}$, and the gradient of the objective function $D(\boldsymbol{\theta})$ becomes

$$\nabla_{\boldsymbol{\theta}} D(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \boldsymbol{\phi}(\boldsymbol{\theta}) \cdot \nabla D(\boldsymbol{\phi}) = \boldsymbol{S}^T \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) \tag{4.27}$$

and the Hessian becomes

$$\mathcal{H}_{\boldsymbol{\theta}} D(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \boldsymbol{\phi}(\boldsymbol{\theta}) \cdot \mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) \cdot \nabla_{\boldsymbol{\theta}} \boldsymbol{\phi}(\boldsymbol{\theta})^T = \boldsymbol{S}^T \cdot \mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) \cdot \boldsymbol{S} \tag{4.28}$$

A unit step in the gradient direction is now different,

$$\Delta \boldsymbol{\phi} = \boldsymbol{S} \Delta \boldsymbol{\theta} = \boldsymbol{S} \frac{\boldsymbol{S}^T \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi})}{||\boldsymbol{S}^T \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi})||},$$

which has a very significant effect on the performance of pure gradient descent optimizers. The performance of gradient descent methods will improve, the lower the condition number (the ratio of largest to smallest eigenvalues) of the Hessian becomes [151]. The search directions of quasi-Newton algorithms will also be affected by scaling, although, as they build up their estimate of the Hessian, the effect will gradually be reduced.

The Newton-Raphson step (Equation 3.13) remains unchanged, as the scaling matrices cancel,

$$\begin{aligned} \Delta \boldsymbol{\phi} = \boldsymbol{S} \Delta \boldsymbol{\theta} &= \boldsymbol{S} \left[ \boldsymbol{S}^T \cdot \mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) \cdot \boldsymbol{S} \right]^{-1} \boldsymbol{S}^T \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) && (4.29) \\ &= \boldsymbol{S} \boldsymbol{S}^{-1} \cdot \mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi})^{-1} \cdot \boldsymbol{S}^{T^{-1}} \boldsymbol{S}^T \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}) && (4.30) \\ &= \mathcal{H}_{\boldsymbol{\phi}} D(\boldsymbol{\phi})^{-1} \cdot \nabla_{\boldsymbol{\phi}} D(\boldsymbol{\phi}). && (4.31) \end{aligned}$$

However, they are not entirely immune to the effect of parameter scaling. For trust-region methods the shape of the trust-region is changed by the scaling, and Conn *et al.* [59] emphasize the importance of scaling for good performance. A trust-region with inappropriate scaling will lead to the internal model that is formed being trusted too well in some directions, and not well enough in others. At best, this will lead

to slow convergence while at worst it may cause algorithm failure [59]. The effect on line searches is mainly to change the size of the starting steps along the line which are used to bracket the minimum. In image registration problems, this can lead to the optimizer jumping completely outside the quasiconvex range of transformations in the first few steps, which usually leads to algorithm failure.

For practical reasons diagonal scaling matrices are generally used, and the remainder of the discussion will be restricted to diagonal matrices. A diagonal $\boldsymbol{S}$ scales each parameter independently, rather than linearly combining them. This is efficient in both time and space, and also makes the scaled parameters easier to understand intuitively.

### 4.5.1 An Automatic Method For Scaling

Given that there are numerous reasons why proper scaling of an optimization problem is desirable, what characteristics determine a good scaling? The technical considerations described in the previous section indicate that the scaling should: (1) reduce the condition number of the Hessian, and (2) somehow make equivalent the accuracies of different parameters. Ideally, it should be possible to determine the scaling by only inspecting the fixed image. During the image registration process, the fixed image is constant, but the moving image is changing. If the scaling is dependent on the moving image it might have to be changed every iteration. Furthermore, many practical registration problems have the form of matching many data images to a common reference image. A scaling method that can be computed from the reference image will be easier to apply in this situation.

To address the first criterion, consider that the need for controlling the spectrum or condition number of a matrix arises frequently in the solution of linear systems, *i.e.,* $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, and is usually addressed through *preconditioning* [16]. That is, both sides of the equation can be multiplied by a preconditioning matrix, $\boldsymbol{S}$. If well chosen, this matrix will improve the spectral properties of $\boldsymbol{A}$, rendering the solution much easier [16]. That is, instead of solving,

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{4.32}$$

one solves,

$$\boldsymbol{S}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{S}\boldsymbol{b}, \tag{4.33}$$

If the preconditioner(s) can be chosen so that the resulting linear system is so much easier to solve that the cost savings is greater than the cost of computing and using the preconditioner, then this will be a more efficient way to solve the system [16]. The Newton-Raphson step involves the solution of just such a linear system. Applying a scaling to each parameter is equivalent to preconditioning the Hessian with a diagonal matrix. Therefore, a good diagonal preconditioner for the Hessian may provide a good scaling for the image registration problem.

Consider the approximate Hessians of each of the image difference measures discussed in this thesis, MSD (Equation 4.5), NCC (Equation 4.17) and MI (Equation 4.25). Each of their Hessians has an important term, $\boldsymbol{T}$, of the form

$$\boldsymbol{T} = \gamma \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}}^T \frac{\partial \boldsymbol{I}_m}{\partial \boldsymbol{\phi}} \tag{4.34}$$

where $\gamma$ is an arbitrary positive scalar multiplier. For MSD and NCC this term is clearly the dominant term. For MI, it is not dominant, but it is the main positive definite component. This means that this term is responsible for the largest eigenvalue of the Hessian. The MI image difference measure forms a funnel shape around the minimum (Fig. 4.15). Its curvature is negative in a radial direction from the minimum and positive orthogonal to that. It is this positive curvature – which must come from the positive definite term – that primarily controls the behavior of the optimizer. Therefore, a preconditioner for matrices of this form should provide a good set of scaling factors.

Dennis and Wolkowicz [68] showed that the optimal diagonal preconditioner for matrices of the form $\boldsymbol{A}^T\boldsymbol{A}$ (such as the term $\boldsymbol{T}$) is a diagonal matrix with terms equal to

$$\boldsymbol{S}_{ii} = \frac{1}{\sqrt{\sum_j \boldsymbol{A}_{ij}^2}}$$

Each $\boldsymbol{S}_{ii}$ is the reciprocal square root of the corresponding diagonal element in the matrix $\boldsymbol{A}^T\boldsymbol{A}$. This is also known as the Jacobi preconditioner [16].

However, the term $\boldsymbol{T}$ in Equation 4.34 is a function of the moving image, and for

practical reasons, it is desirable to avoid using the moving image to derive the scales. Note that each diagonal element of $\boldsymbol{T}$ is a summation of squared terms

$$\boldsymbol{T}_{ii} = \gamma \sum_p \left[ \sum_d \frac{\partial \boldsymbol{I}_{m_p}(\boldsymbol{W})}{\partial \boldsymbol{W}_d} \frac{\partial \boldsymbol{W}_d(\boldsymbol{x}_p, \boldsymbol{\phi})}{\partial \boldsymbol{\phi}_i} \right]^2 \tag{4.35}$$

where $p$ indexes all the pixels, and $d$ indexes the dimensions of the image space. Without needing to know the image itself, suppose that the image derivatives are evenly spatially distributed throughout the image with some average value $\alpha$. Then this term becomes

$$\boldsymbol{T}_{ii} = \gamma \sum_p \left[ \sum_d \alpha \frac{\partial \boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi})}{\partial \boldsymbol{\phi}_i} \right]^2 = \alpha^2 \gamma \sum_p \left[ \sum_d \frac{\partial \boldsymbol{W}_d(\boldsymbol{x}_p, \boldsymbol{\phi})}{\partial \boldsymbol{\phi}_i} \right]^2 \tag{4.36}$$

The global factor, $\alpha^2\gamma$, can be ignored, as scalar multipliers do not affect the condition number, and a Jacobi preconditioner for $\boldsymbol{T}$ can be computed as

$$\boldsymbol{S}_{ii} = \frac{1}{\sqrt{\sum_p \sum_d \frac{\partial \boldsymbol{W}_d(\boldsymbol{x}_p, \boldsymbol{\phi})}{\partial \boldsymbol{\phi}_i} \frac{\partial \boldsymbol{W}_d(\boldsymbol{x}_p, \boldsymbol{\phi})}{\partial \boldsymbol{\phi}_i}}}. \tag{4.37}$$

Since only a very basic assumption about the moving image has been made, and no actual moving image components are used in the calculation, this scaling matrix can be computed using only the geometry of the problem. In other words, only the Jacobian matrix of the transformation and the coordinate lattice of the pixels of the fixed image are required.

The preconditioning properties of this matrix should reduce the condition number, but recall that this was not the only criterion for a good set of scaling factors. This scaling matrix also addresses the second criterion. Consider that $\sum_d \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{\phi}_p} \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{\phi}_p}$ approximates the square of the distance moved by a pixel under a shift of one unit in the transformation parameters. Thus each element of $\boldsymbol{S}_p$ is the reciprocal of the root mean square distance moved by a pixel under a unit change in that parameter. Scaling by this factor thus makes the parameters roughly equivalent in terms of the average motion they induce on pixels in the image.

In cases where only certain areas of the fixed image are relevant – such as when

the moving image is smaller, when large parts are masked out, or when there are large blank areas – then treating all pixels equally may not be appropriate. In such cases, the assumption made earlier that image gradients are evenly distributed is false. A more reasonable approach is in these cases would be to mask out the unused areas, and compute the scale factors using the pixels that remain.

Scaling the parameters by the matrix, $\boldsymbol{S}$, should improve optimization performance so long as it effectively preconditions the Hessian by reducing its condition number. This provides a means of determining when this scaling method can be expected to work well, and when it may not. In Figure 4.17, the Hessians for the case of registering the image in Figure 4.1 with itself have been computed. The condition number of each Hessian has been determined both before and after scaling by this matrix computed for the problem. In all cases the condition number of the Hessian is improved by a diagonal scaling. The improvement is very dramatic for the matrix based transforms. For the thin plate spline transform, the condition number was not overly large to start with, and is not changed much by this scaling. The scaling does not have much effect in this case because the Hessian is very dense. The Hessian of the B-spline transform is extremely sparse. Its condition number both before and after scaling is extremely large, which indicates the matrix is nearly singular. Based on these results, scaling with this factor can be expected to be particularly important for the matrix based transforms, and less important, although still having some effect, on the deformable transforms.

### 4.5.2 Testing the Scale Factors

To test the effectiveness of this set of parameter scaling factors it is necessary to compare it against other possible scaling factors. A constant multiplier has no effect, so to compare against other factors the relative ratios between elements of $\boldsymbol{S}$ must be changed. In other words, the ratios $\frac{\boldsymbol{S}_{ii}}{\boldsymbol{S}_{jj}}$ must be changed. To do this, alternate

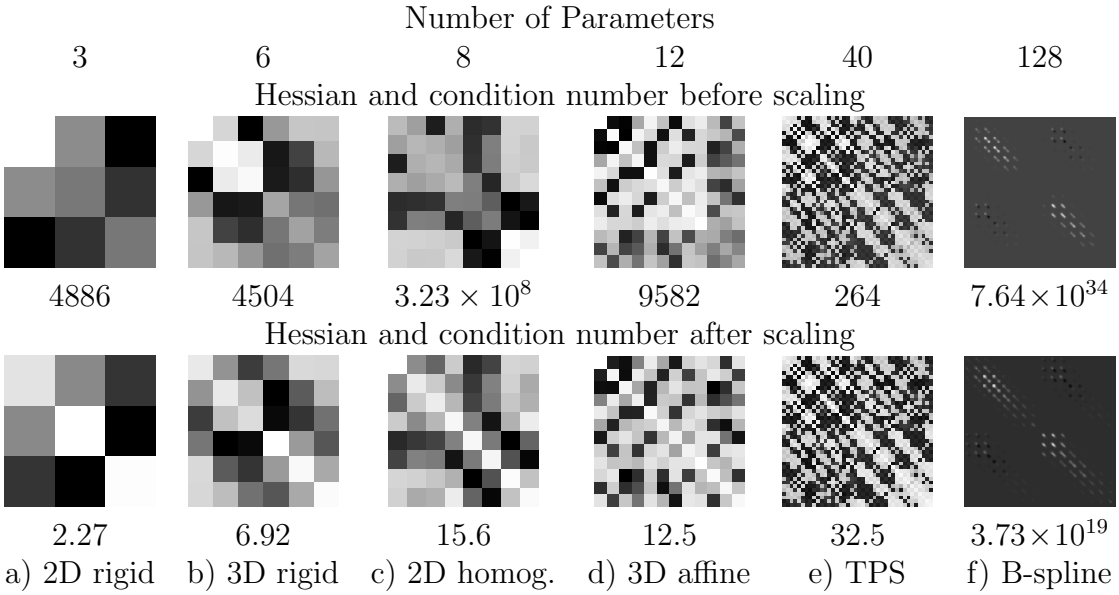



**Figure 4.17** These images show the Hessians and their condition numbers for various image transformations both before and after scaling with the matrix, $\boldsymbol{S}$. The Hessian values have been logarithmically scaled to fit in the range of image intensities. Note how the condition numbers have been reduced significantly by the parameter scaling. This appears as a stronger, and more even, diagonal components after scaling for the matrix based transforms (a-d). The density of the off-diagonal elements in the TPS transform (e), and the near singularity of the B-spline Hessian (f) reduce the effectiveness of scaling for these cases. The intensities of the pre and post scaling Hessian images are scaled differently and should not be considered comparable.

scaling matrices are defined using the following formula:

$$\hat{\boldsymbol{S}(f)} = \begin{cases} \boldsymbol{S}_i(1-f)+f & \text{if } f<0 \text{ and } \boldsymbol{S}_i \geq 1 \\ 1/\left(\frac{1}{\boldsymbol{S}_i}(1-f)+f\right) & \text{if } f<0 \text{ and } \boldsymbol{S}_i < 1 \\ \boldsymbol{S}_i \cdot \frac{1}{1+f} + \frac{f}{1+f} & \text{if } f \geq 0 \text{ and } \boldsymbol{S}_i \geq 1 \\ 1/\left(\frac{1}{\boldsymbol{S}_i} \cdot \frac{1}{1+f} + \frac{f}{1+f}\right) & \text{if } f \geq 0 \text{ and } \boldsymbol{S}_i < 1 \end{cases} \tag{4.38}$$

Here the $\boldsymbol{S}_i$ are the scale factors. This formula has the effect of altering the ratios between elements of a set of numbers, as shown in Table 4.5. For these experiments $f = \pm 9$ was used, which closely approximates multiplying or dividing the larger scale numbers by 10. In the following, the scale factors computed with Equation 4.37 will be referred to as case x1, those where $f = 9$ as case x10, and those where $f = -9$ as case x0.1. The case of not using any scale factor at all (*i.e.,* $\boldsymbol{S}$ is the identity matrix)is also tested, and is referred to as case NSCL.

| Case | $f$ | $\hat{\boldsymbol{S}}$ | | | | | | |
|---|---|---|---|---|---|---|---|---|
| x10 | -9 | 9991 | 991 | 91 | 1 | 0.011 | 0.001 | 0.0001 |
| x1 | 0 | 1000 | 100 | 10 | 1 | 0.1 | 0.01 | 0.001 |
| x0.1 | 9 | 100.9 | 10.9 | 1.9 | 1 | 0.53 | 0.092 | 0.0099 |
| NSCL | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Table 4.5** Effect of different parameters $f$ on a set of scale factors. The parameter $f$ changes the ratios between elements of the scaling matrix $\boldsymbol{S}$. The use of $f = \pm 9$ corresponds to approximately multiplying and dividing by 10 for the larger elements of $\boldsymbol{S}$. The NSCL case is to use no parameter scaling, *i.e.,* $\boldsymbol{S}_{ii} = 1$.

All the synthetic registration problems were tested using all four scaling cases (x10,x1,x0.1,NSCL) for three image difference measures (MSD, NCC and MI). Results for all three measures were extremely similar, and lead to identical conclusions, so only the results for the MI measure are presented here. Figure 4.18 shows the results of the registrations with the matrix-based transformations (*i.e.,* 2D and 3D rigid transformations, 2D homographies and 3D affine transforms). For each of the matrix based transformations investigated, all five optimizers being considered were applied. (The optimizers are: downhill simplex (DS), Powell's (P), gradient descent (GD), lim-

ited memory Broyden-Fletcher-Goldfarb-Shanno (BFGS) and Newton-Raphson (N).)

Note that in all the graphs, lower numbers indicate better performance. Each case (transform/optimizer) is represented by four bars. The bars are, in order from left to right, cases x10, x1 (the optimal scaling), x0.1 and NSCL. Therefore, a roughly "$\cup$" shaped graph indicates better performance using the scaling factors of Equation 4.37, while a "$\cap$" shaped graph indicates poorer performance with these scaling factors.

The effect of the scaling is most visible in the failure rate graph (bottom row, Figure 4.18). For all algorithms if no scaling factor is used, the algorithms are extremely unreliable. (Recall that the definition of failure is completing registration further than 5 units away from the correct position in mTRE, or aborting in an error state.) Between scaling factors x10, x1 and x0.1 for all algorithms except Powell's, the use of the scaling factor given by Equation 4.37 (case x1) yields strongly better results. All of the other performance criteria also deteriorate when scaling factors other than x1 are used, but the effect is not as striking. One reason that the effect is diminished in the other graphs is that failed runs are rejected when these statistics are computed.

The gradient descent algorithm is the most sensitive to the problem scaling, which is not surprising since its search direction is directly affected by the parameter scaling. The Newton-Raphson and downhill simplex algorithms are somewhat sensitive, while the BFGS and Powell method are the least. The greater robustness of the BFGS and Powell algorithms may be explained by the fact that they use line-searches. Scaling affects the line search in two ways. A poorly scaled problem may cause the optimizer to jump out of the capture range of the minimum on its first few steps, or the optimizer may stop too early due to the misinterpretation of the stopping criteria. If these issues are avoided, the line search will probably complete successfully. Overall, the results are unequivocal. Algorithm performance for optimization of matrix based transforms is significantly improved by using the scaling factors computed by Equation 4.37.

The running time and number of evaluations required for the registrations with deformable transformations (*i.e.,* the 2D and 3D thin plate splines and B-splines) are shown in Figure 4.19. The computational cost of using downhill simplex or Powell's algorithm on these transformations, and that of using the Newton-Raphson algorithm on the largest dimensional case, was too high, so these cases were not examined. Again results for all the image difference measures are extremely similar, so only those for

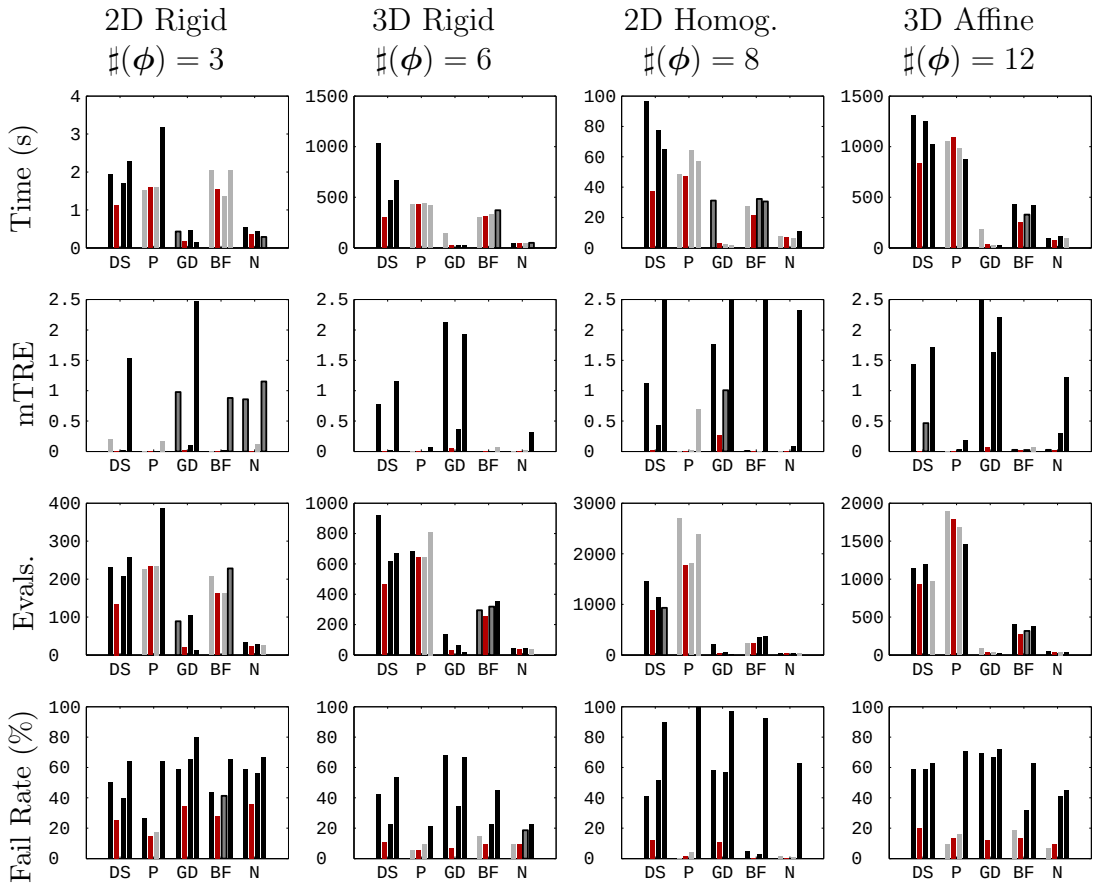



**Figure 4.18** Comparison of registration results for transforms with different numbers of parameters using the mutual information cost function. The optimizers are identified as DS – downhill simplex, P – Powell's, GD – gradient descent, BFGS – Broyden-Fletcher-Goldfarb-Shanno, N – Newton-Raphson. For each optimizer, the bars represent, from left to right, the result using the scale factors x10, x1, x0.1 and NSCL. Case x1, the optimal scale factor, is identified in red. If the results of the other cases differ in a statistically significant way from case x1 they are shown in black. Ambiguous results (see Section 3.2.1) are shown with black outline, and results that are not significantly different from case x1 are shown in gray. The datasets used were: 2D Rigid – Sagittal Brain Slice; 3D Rigid and Affine – Brainweb; 2D Homography – Agra Fort. mTRE is in mm for all cases except the homography, where it is in pixels. Note that the vertical scale is not the same in all cases.

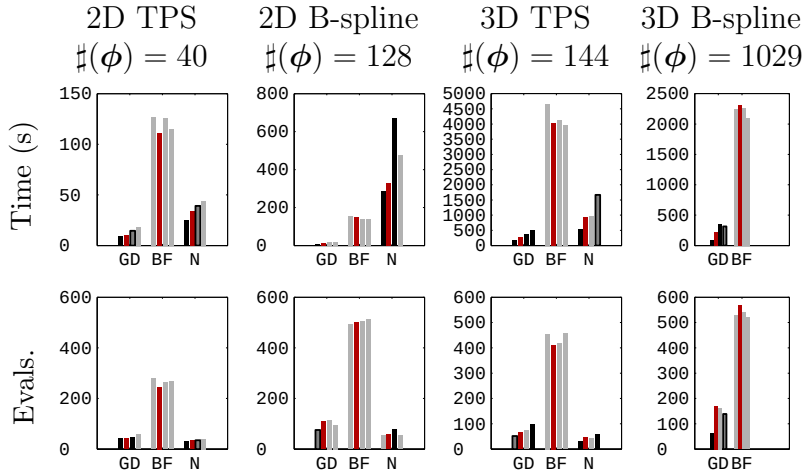



**Figure 4.19** Comparison of registration results for different scale factors for transforms with different numbers of parameters using the mutual information cost function. The optimizers are identified as GD –gradient descent, BFGS – Broyden-Fletcher-Goldfarb-Shanno, N – Newton-Raphson. For each optimizer, the bars represent, from left to right, the result using the scale factors x10, x1, x0.1 and NSCL. Case x1, the optimal scale factor, is identified in red. If the results of the other cases differ in a statistically significant way from case x1 they are shown in black. Ambiguous results (see Section 3.2.1) are shown with black outline, and results that are not significantly different from case x1 are shown in gray. The datasets used were: 2D TPS and B-spline – Sagittal Brain Slice; 3D TPS and B-spline Deformable Phantom. Note that the vertical scale is not the same in all cases.

mutual information are shown. As the effect on the Hessians shown in Figure 4.17 suggested, the effect of scaling on the deformable transformations is much more subtle than in the matrix transform case. The results suggest that the gradient descent optimization is somewhat slower for the x0.1 and NSCL case. Many differences are not statistically significant, however, so it is difficult to draw a firm conclusion. The case of the 2D B-spline registrations with the Newton-Raphson algorithm shows a large, statistically significant increase in time required for the x0.1 case, but as the rest do not show significant differences, this may be anomalous.

The accuracy results for these registrations are reported as comparisons of the distributions of errors of points in a $50 \times 50$ grid across the image. The differences between the error distributions can best be described as subtle. One graph which typifies the overall results is shown in Figure 4.20. The remainder have been placed in Appendix D.

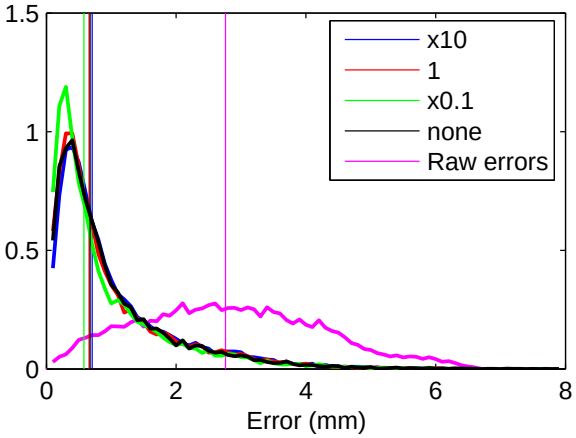



**Figure 4.20** Distributions of errors remaining after registration of the deformed Sagittal Brain Image using the 2D B-spline transform with the gradient descent optimizer. The image difference measure used was mutual information. This result is selected as being typical of the results of this experiment. The graph shows the distribution of errors over a $50 \times 50$ grid of points in the image extent. Point errors for all 25 runs have been combined into one histogram for each set of scale factors. Vertical lines show the median error. With all scale factors, reasonably successful registration has been achieved, as shown by the distinct leftward shift in the distribution. Case x0.1 produces a slightly, but statistically significantly smaller median error.

From a visual inspection of these graphs and the registration results, it is difficult if not impossible to draw a conclusion about the effectiveness of the scaling factors. However, when a statistical testing method is applied, a pattern does emerge. The distributions of errors can be checked to see if they have a statistically significant difference using a rank sum test (see Section 3.2.2). If a significant difference exists, the median error is compared and the configuration with the lowest median error is considered superior. Using this comparison technique, the scaling factors can be ranked in the following order: x0.1, x1, NSCL and x10. This ranking was consistent across optimizer and image difference measure. To be clear this does not mean that case x0.1 had significantly lower median error in every case, but in a majority of cases.

In certain cases, it was noted that the registration made a small proportion of the point error shift to the *right*. That means that the error on these points actually increased as a result of the registration. This appeared to affect mainly points on the very edge of the image, and the effect appeared to be strongest in the case of x10, then x1, x0.1 and smallest for no scale factor. Using the scaling factors appears to make certain parts of the image register more accurately and certain other parts less accurately.

### 4.5.3 Experiments on Realistic Data

To confirm these results on realistic data, 2D and 3D rigid registrations were performed using the Axial Brain Slices and Vertebra datasets described previously. These datasets are multimodal, so only mutual information was used to register them. Results are shown in Figure 4.21. As in the simulated registration cases, the failure rate is most affected by the scaling factor. The other criteria, running time, mTRE and number of evaluations are, for nearly all cases, significantly better when using the scaling factors of case x1. The exception is Powell's algorithm, which runs a small but significant amount faster for case x10. However, its mTRE and failure rate are slightly worse for this case.

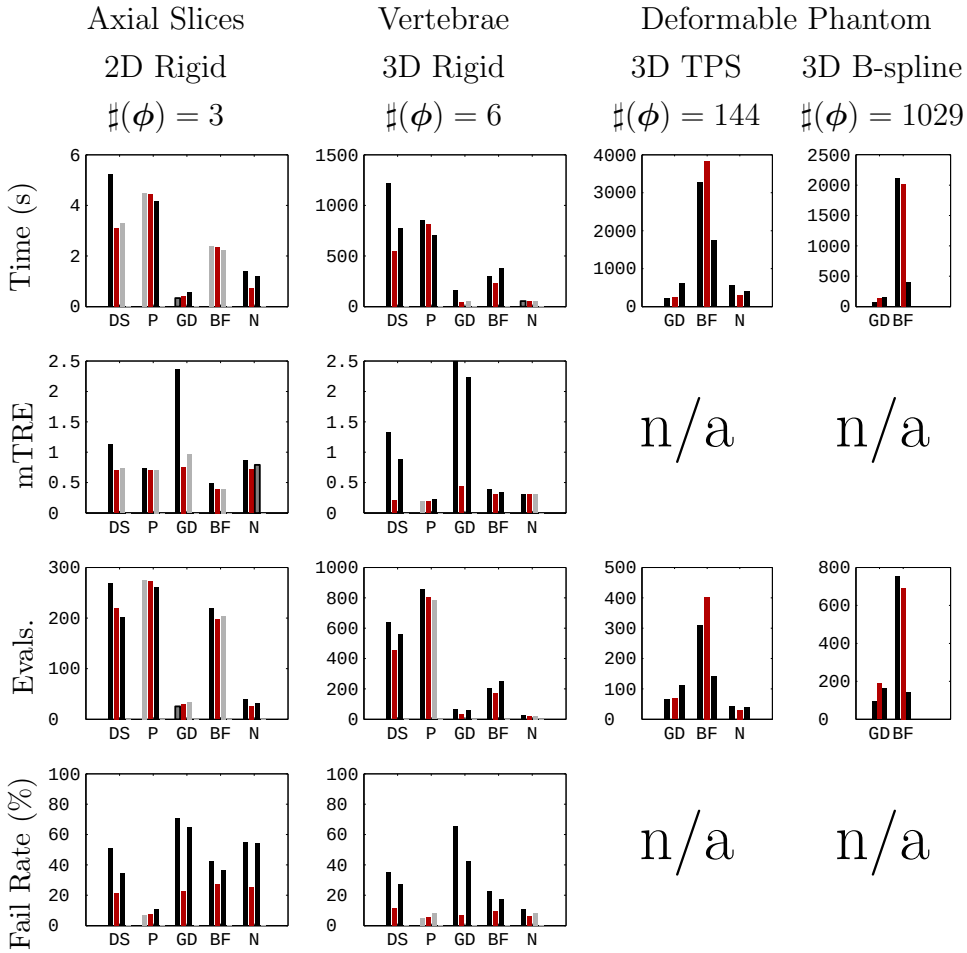



**Figure 4.21** Comparison of registration results for realistic registration problems with different scale factors. Due to its poor performance the NSCL case was not used for these experiments. The optimizers are identified as DS – downhill simplex, P – Powell's, GD – gradient descent, BFGS – Broyden-Fletcher-Goldfarb-Shanno, N – Newton-Raphson. For each optimizer, the bars represent, from left to right, the result using the scale factors x10, x1, and x0.1. Case x1, the optimal scale factor, is identified in red. If the results of the other cases differ in a statistically significant way from case x1 they are shown in black. Ambiguous results (see Section 3.2.1) are shown with black outline, and results that are not significantly different from case x1 are shown in gray. mTRE is in mm for all cases. mTRE is analyzed separately for the deformable transform case, and failure rate does not apply.

The deformable phantom datasets 4.7 were registered using the MSD cost function

as they are of the same modality. Both the 3D TPS and 3D B-spline transformations were used. The timing results for the registration of the deformable phantom objects show an interesting pattern (Figure 4.21). The gradient descent method runs significantly slower for case x0.1, while the BFGS method runs significantly faster. This is correlated with improved accuracy performance in this case.

For this dataset, there is no gold standard transformation to compare to. However, a comparison for accuracy can be performed using the distances between points in the segmented mesh representation of the balloon, as described in Section 4.2.4. The distribution of these errors can be compared as it was for the previous experiments, and it is found that, similar to the synthetic registration cases, case x0.1 gives better overall performance. This means that using the scaling factors of case x0.1 for the BFGS method yields a significant improvement in both speed and accuracy for that optimizer. Figure 4.22 shows one example which typifies the results. The remaining graphs have been placed in Appendix D.

### 4.5.4 Discussion

The testing of the parameter scaling factors has yielded unequivocal results on the matrix based transforms. All optimizers perform as well or better when using the scaling factor defined by Equation 4.37, *i.e.,* case x1. In the following experiments, these scale factors will be used. For the deformable transformations, the effect of parameter scaling is quite subtle. This is consistent with the limited improvement to the Hessian condition number in these cases shown previously. It was found using a set of scaling factors with a less extreme range than the ones calculated by Equation 4.37 – specifically those of case x0.1 – gave a small, but consistent improvement in median accuracy. Accordingly, the scaling factors of case x0.1 will be used for the following deformable registration experiments.

It was not possible to exactly identify the cause of the better performance of case x0.1. By checking the effect of this set of scale factors on the Hessian it was ascertained that it was *not* due to it being a better preconditioner. The scale factors of case x0.1 do not improve the condition numbers of the Hessians as much as the optimal preconditioner, case x1. It is conjectured that the observed effect is due to the somewhat different nature of these registration problems as compared to the matrix transform

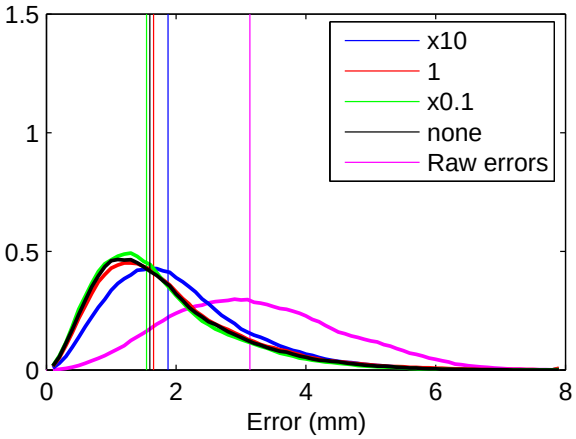



**Figure 4.22** Distributions of errors remaining after registration of the Deformable Phantom, Case partially inflated to fully inflated using the 3D B-spline transform with the gradient descent optimizer. The image difference measure used was MSD. This result is selected as being typical of the results of this experiment. The graph shows the distribution of distances between points in a warped mesh of the balloon part of the phantom, and the mesh of the balloon created from the target volume. Vertical lines show the median error. With all scale factors, reasonably successful registration has been achieved, as shown by the distinct leftward shift in the distribution. Case x0.1 produces a slightly, but statistically significantly smaller median error.

case. The parameters of deformable transforms naturally affect different parts of the image unequally. The scaling factors become very small (*i.e.,* $<< 1$), making the RMS effect of a unit shift equal over the entire image. However, this does not conform to the nature of the problem. For instance, the parameters for the padding control points of the B-splines have very small effects on the image. Their scale factors become very small, allowing these parameters to show very large variations – deviations equating to thousands of pixels of deformation – during the optimization. The net deformation does not seem to be seriously affected, but deformations this large are very unrealistic. This is also consistent with the observation that a small proportion of points seem to be made worse by the registration process. The best solution may be to apply a regularization term, to keep these terms within reasonable bounds for the problem.

Now that the approximate Hessian calculation, and the selection of the scale factors has been addressed, it is possible to configure each optimization algorithm well for each

problem. This allows a fair comparison between them to be performed. Optimizer performance is investigated in the following section.

## 4.6 Optimizer Selection

The optimization algorithm plays a key role in performing efficient and reliable image registration, and selecting the right one can greatly improve performance. However, there is a lack of reasoned guidelines for selecting optimizers, with previous studies focusing on very specific registration problems. When selecting an optimizer, there is a tradeoff between the input that is required, and the number of function evaluations that the optimizer will require to converge. While optimizers that take more input information can be expected to converge faster in terms of number of iterations, producing that additional information (*i.e.,* gradients, Hessians) has a cost. It may be that computing the additional information takes so much time that nothing is gained by having fewer iterations.

It is also important to consider how fast convergence can take place when the optimizer is started near the optimum. This issue becomes particularly important in a multiresolution context, where for the second and subsequent levels, it is reasonable to assume that the optimization will be started fairly close to the true optimum. An optimization algorithm that cannot stop quickly when appropriate may waste a lot of computation in this case. The importance of this issue for optimizer selection was discussed at length by Maes *et al.* [137] where it has a strong influence on the final recommendations of the paper.

For image registration problems, the cost function being evaluated is computationally expensive. As a result, the time required to perform the optimization is almost completely determined by the number of function evaluations that it will require. This premise lies behind much of the work in this chapter, and to justify it, consider the following simple example.

If the Newton-Raphson algorithm was implemented naively, it would require the matrix inversion of the Hessian. This would be the most computationally intense part of the optimizer internal steps, and actually inverting the matrix would be a particularly inefficient approach to solving for the Newton-Raphson step. Nevertheless, a test

of the speed of inverting random matrices using the ITK library on a 3.2Ghz Pentium IV revealed that inverting $100 \times 100$ matrices required an average of 0.04 seconds, and inverting $250 \times 250$ matrices an average of 0.66 seconds. Note (see also Figure 4.26) that computing the Hessian even on a very small image with 100 parameters required approximately 10 seconds, and with 180 parameters required over 20 seconds, on the same machine. The matrix inversion or other steps involved in the optimizers internal processing are simply not the computational bottleneck in the process.

### 4.6.1 Estimating Optimizer Performance

Estimating the number of function evaluations required by an algorithm is an essential step in selecting the right one. Unfortunately, estimating how many function evaluations will be required for successful completion of a particular optimization algorithm is not an easy task. The convergence of optimization algorithms has been heavily studied [80, 85, 131, 151, 158, 192], but these results do not lead directly to an answer for this question. Nevertheless, although exact bounds are not possible, this section attempts to derive reasonable guesses for the average complexity of carrying out the optimization.

Convergence of optimization algorithms is described in two ways in the literature [59, 151, 192]. The first way is that, for certain algorithms, the number of steps required to converge to an exact solution on a quadratic function can be determined. The second way is to describe the rate of convergence of the sequence of iterates generated by the algorithm. For example, the rate of convergence is described as *Q-linear* if [59]

$$\exists N | \forall n > N : \quad ||\boldsymbol{\phi}_{(n+1)} - \boldsymbol{\phi}_{opt}|| \leq c||\boldsymbol{\phi}_{(n)} - \boldsymbol{\phi}_{opt}||; \quad 0 < c < 1 \tag{4.39}$$

The property of linear convergence can be used to establish a rough estimate of the number of steps required for an algorithm to complete. It is easy to see that while this property holds the number of steps required to reduce the error by a ratio, $r$, is $\log_c(r)$. For the algorithm to complete, the residual error must drop to some acceptable level. If it is assumed that the error in the parameters is roughly equal in each parameter, then the starting error increases as the square root of the number of parameters.

Therefore, the number of steps required for an algorithm with the linear convergence property to complete can be estimated to be $O(\log(\sharp(\boldsymbol{\phi})))$, where $\sharp(\boldsymbol{\phi})$ is the number of parameters.

Both of these types of convergence estimates do not permit any solid predictions about how an optimizer will perform in practice. The first type of convergence result does not directly apply because the functions of interest are rarely quadratic. If the cost function is analytic then it may be considered quadratic in a close vicinity of the minimum. However exactly how close is necessary is difficult to say. The second type of convergence result is asymptotic in nature – the property is only guaranteed to hold for large enough $N$. It is entirely possible that the $N$ where it holds is far larger than any number of iterations that will be reached in a realistic problem.

However, bearing these caveats in mind, it is possible to use this information about the convergence properties of each algorithm to make some approximate guesses about how many evaluations will be required for each algorithm to complete. The notation $\tilde{O}(n)$ will be used to denote these guesses, and to indicate that they do not have the formal bounding property of the $O(n)$ notation. In what follows, let $N = \sharp(\boldsymbol{\phi})$, *i.e.,* the number of parameters.

The Nelder-Mead Downhill Simplex [149] algorithm requires only function values as input. It is not even guaranteed to converge, although in practice it usually does [131]. Its rate of convergence is therefore difficult to define, although it is reported to be "slow" [131]. Let the estimate be that it is somewhat worse than Powell's algorithm (described below), requiring $\tilde{O}(N^q); q > 1$ iterations. Note that each iteration of downhill simplex may require $O(N)$ function evaluations, as the shrink operation requires $N + 3$ function evaluations.

Powell's method [156] also only requires function values as input and is widely known for converging in $N$ steps on a $N$-dimensional quadratic [131, 192]. Thus it can be directly estimated that it will require $\tilde{O}(N)$ iterations to converge. However like the downhill simplex method it should be noted that each of those $N$ iterations contains within it $N$ line searches [192], so the number of function evaluations required is $\tilde{O}(N^2)$.

Gradient descent, as its name implies, requires both function values and gradients as input. Gradient descent methods simply take steps in the direction opposite the

gradient. The step size may be chosen by a line search, or, as is more common, some sort of heuristic is used to adapt the step size [59, 192]. The convergence rate of gradient descent is linear, that is, it follows Equation 4.39 [176]. Based on the reasoning about Equation 4.39, the convergence can be estimated to require $\tilde{O}(\log(N))$ steps.

Quasi-Newton methods require both function values and gradients as input. At each step, they examine the change in the gradient and update an internal model of the function Hessian. This internal Hessian is used to compute the next step according to the Newton step. The first few steps of quasi-Newton methods are therefore a lot like gradient descent, but their performance approaches that of a Newton-Raphson type method as they continue. Their rate of convergence is better than linear [59, 192]. The number of steps required will be estimated as $\tilde{O}(\log(N)^r)$, where $0 < r \leq 1$

Newton-Raphson methods require function values, gradients and Hessians as input, but will converge in a single step on a quadratic function. The number of steps required to converge can thus be estimated as $\tilde{O}(1)$.

If started exactly at the optimum, neither the downhill simplex, nor Powell's method can identify the point as a minimum until they have gone through their full cycle. That is, the downhill simplex must shrink its simplex down to its minimum size, and Powell's method must perform a full battery of line searches. The remaining methods, however, have access to the gradient information. Because the gradient is zero at a minimum, these methods can stop after a single step if needed. As previously mentioned, this is an important issue in a multiresolution approach.

In general, the optimization algorithms that use more input information (*i.e.,* gradients, Hessians) converge faster. However, there is a tradeoff here, because that additional information has to be computed. The complexity of evaluating the image difference measure, its gradient or its Hessian is dependent on the transform being used. For an $N$-dimensional transformation, evaluating the cost function requires $O(N)$ operations per pixel. (It might first appear that this should be $O(1)$, but it is important to consider that the calculation must presumably access each parameter at least once.) The complexity of calculating of the difference measure gradient can vary. For the transformations used in this work, it ranges between $O(N)$ and $O(N^2)$ per pixel. Even in the case where the function evaluation has the same complexity with

or without the gradient, the gradient calculation can be expected to take longer. The calculation of the Hessian requires the calculation of $N^2$ elements so it is necessarily an $O(N^2)$ per pixel operation.

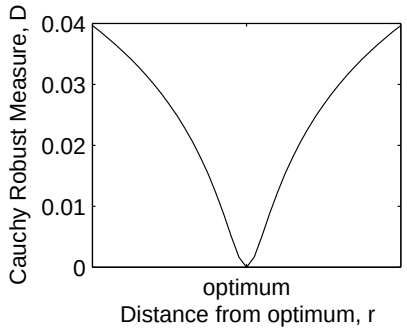
All these estimated complexities, are summarized in Table 4.6. From the preceding discussion it is clear that these must be taken as estimates, not as true bounds, but nevertheless several useful deductions can be made. As the number of parameters increases, the downhill simplex and Powell methods will prove to be the slowest, even though they require the least computationally expensive input. It can also be expected that a quasi-Newton method should outperform an equivalent gradient descent method as the number of parameters increases. Finally, whether a Newton-Raphson method will outperform a gradient descent or quasi-Newton method will depend on the complexity of evaluating the gradient. For transformations where the evaluation of the gradient is an $O(N^2)$ operation, Newton-Raphson methods may prove to be more competitive. For transformations where the evaluation of the gradient is an $O(N)$ operation, there may be a set of parameter sizes where the Newton-Raphson method is competitive, but as $N$ increases, the cost of computing the Hessian will quickly become prohibitive. The following section investigates these complexities on simple problems.

| Algorithm | Minimum Evals | Steps Required | Evals Per Step | Cost per Eval | Total Complexity |
|---|---|---|---|---|---|
| Simplex | $O(N)$ | $O(N^p); p > 1$ | $O(N)$ | $O(NP)$ | $O(N^q P); q > 3$ |
| Powell | $O(N)$ | $O(N)$ | $O(N)$ | $O(NP)$ | $O(N^3 P)$ |
| Grad. Descent | 1 | $O(\log(N))$ | $O(1)$ | $O(NP)$–$O(N^2 P)$ | $O(N \log(N) P)$–$O(N^2 \log(N) P)$ |
| Quasi-Newton | 1 | $O(\log(N)^r)$; $0 < r \leq 1$ | $O(1)$ | $O(NP)$–$O(N^2 P)$ | $O(N \log(N)^r P)$–$O(N^2 \log(N)^r P)$ $0 < r < 1$ |
| Newton-Raphson | 1 | $O(1)$ | $O(1)$ | $O(N^2 P)$ | $O(N^2 P)$ |

**Table 4.6** Computational tradeoffs in different optimization algorithms, here $N$ is the number of parameters, and $P$ is the number of pixels.

### 4.6.2 Exploring Optimizer Efficiency

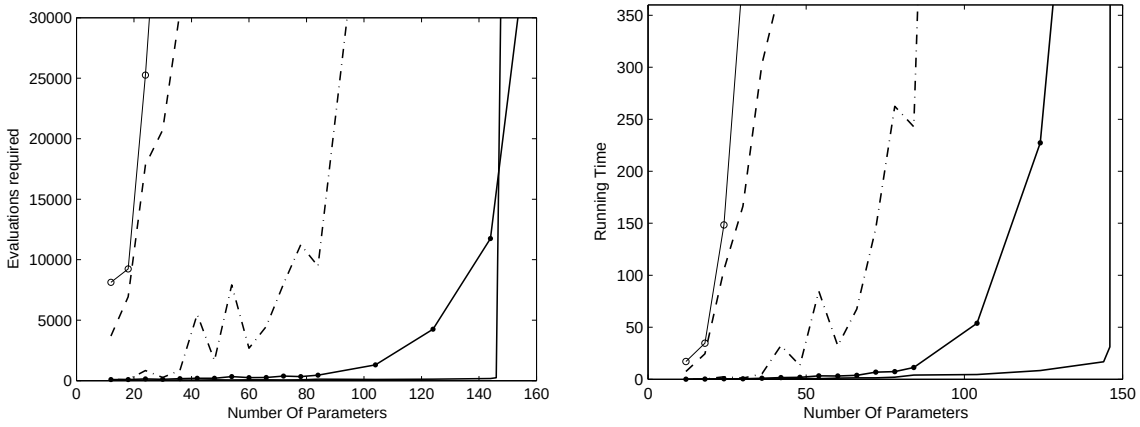
The analysis in Table 4.6 is first investigated by performing a number of optimizations using different numbers of parameters on a simple cost function. For this test, a series of 2D thin plate spline transforms with progressively more points were defined. (This method is described in Appendix C.) These give rise to a series of transforms with progressively more parameters. The parameters of these transforms were perturbed with uniformly distributed noise with a range of $\pm 1.5mm$. This range was chosen so that there was no danger of folds or tears occurring in the transform. The cost function was a Cauchy robust distance measure [179] on the positions of a grid of points evenly spaced over the image extent. This distance measure (see Figure 4.23) was chosen to avoid using a perfectly quadratic measure. It has negative curvature over part of its range, similar to MI.




$$D = \sum_i b^2 \log\left(1 + \left(\frac{r_i(\boldsymbol{\phi})^2}{b}\right)^2\right)$$

**Figure 4.23** The Cauchy robust cost function [179]. The figure on the left illustrates the shape of the Cauchy robust cost function (Equation on right). Note that the shape is quasiconvex, but not convex, like the MI image difference measure. Here $i$ indexes the grid of points, and $r_i(\boldsymbol{\phi})$ is the distance moved by the transformed point. The parameter, $b$, was set to 0.1 for this test.

The number of evaluations and running time required for each optimizer to converge to a solution was recorded, and the results are shown in Figure 4.24. The amount of evaluations, and therefore also of time, required for the downhill simplex and Powell methods grows so rapidly that it is difficult to tell if it is linear or quadratic in nature. In any case, these optimizers rapidly become impractical and are probably completely out of the question for transforms with much more than 10 parameters.




(a) (b)

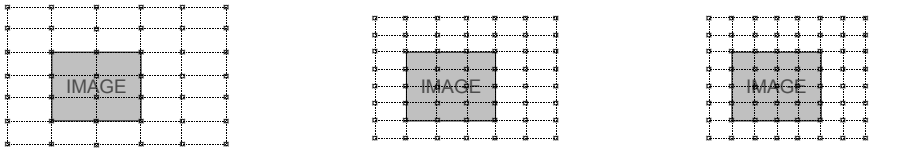
**Figure 4.24** Number of evaluations and time required for optimizing a simple cost function as the number of parameters changes. a) Number of evaluations required for convergence for various numbers of parameters, similar starting positions. b) Time required for convergence.

D.H. Simplex; Powell; Grad. Desc.; LBFGS; Newton;

The number of iterations for gradient descent and the quasi-Newton method have a very apparent upward curve. This is in contrast to the anticipated rate according to Table 4.6 which would have been logarithmic. This theoretical rate of convergence, however, only holds close to the minimum. It is apparent from Figure 4.24 that even this simple problem, which does not start too far from the minimum, is still outside the region where the theoretical rate holds. Of these two methods the gradient descent method clearly requires more iterations than the BFGS method. The Newton-Raphson method requires very few iterations to converge, except for the very last case where its performance has suddenly started to degrade. Examination of the time graph (Figure 4.24b), however, shows that the time required for this method is rising noticeably toward the end. This is due to the greater cost of computing the Hessian, indicating that the cost per iteration must also be considered to get a true picture.

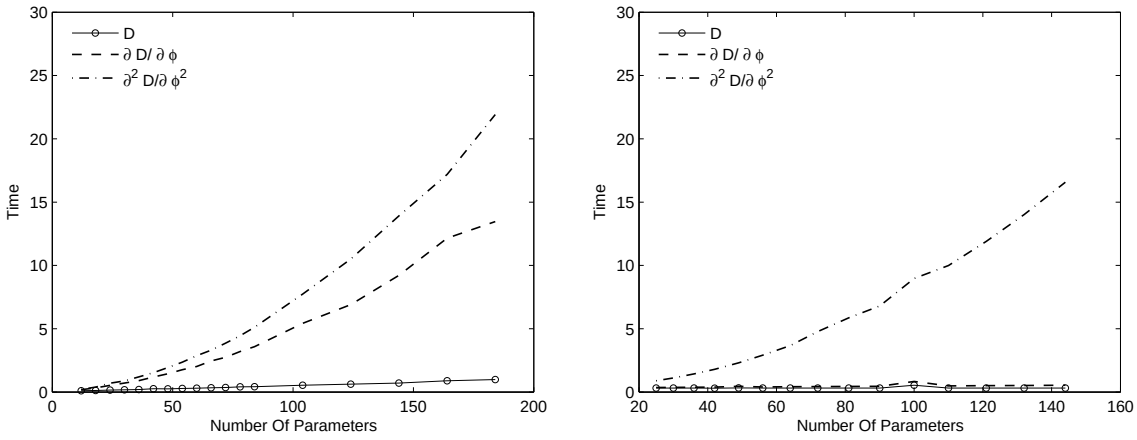
To evaluate the rate of growth of the per-iteration cost, the MSD measure was evaluated between the image in Figure 4.1 and a warped version of itself. The average time required to perform 10 evaluations of the function alone, with its gradient, or with its gradient and Hessian was recorded at different number of parameters. The

measurements were taken on each of the TPS transforms described above and the timing results are shown in Figure 4.26a. A sequence of B-spline transformations with an increasing number of parameters was also defined. These were defined by successively increasing each size of the dimension of the B-spline grid as shown in Figure 4.25. The timing results for similar evaluations using the B-spline transforms are shown in Figure 4.26b.




(a) 3x4 grid ($N = 84$) (b) 4x5 grid ($N = 112$) (c) 5x5 grid ($N = 128$)

**Figure 4.25** B-splines with gradually increasing numbers of parameters, $N$




(a) Thin plate spline (b) B-spline

**Figure 4.26** Time required to compute $D$, $\nabla D$ and $\mathcal{H}D$ on a $240 \times 256$ image. a) Thin plate spline b) B-spline. Note the difference in complexity of computing the gradient.

As the analysis predicted, the time complexity of computing the function alone is roughly linear in the number of parameters. In the case of the TPS transform, the times required for computation of the gradient and Hessian curve upward. The graphs are consistent with the complexities being quadratic in the number of parameters as

found in Section 4.6.1. The cost of computing the Hessian is still more than the cost of computing the gradient for all cases tested, but the cost of computing the gradient rises significantly as the number of parameters increases. However, for the B-spline transform, the complexity of computing the gradient is linear in the number of parameters. For this case, shown in Figure 4.26b, it is clear that for large numbers of parameters it is much more efficient to compute the gradient alone, rather than to compute the Hessian. Therefore significant differences in efficiency between the two approaches can be expected as the number of parameters rises.

The number of evaluations required for the optimizer to converge when started exactly at the optimum was also measured(Figure 4.27). Because the gradient magnitude is used as one of the stopping criteria, gradient descent, quasi-Newton and Newton-Raphson methods all stop after one evaluation. As expected, the evaluations required for the Powell and downhill simplex methods are linear in the number of parameters.

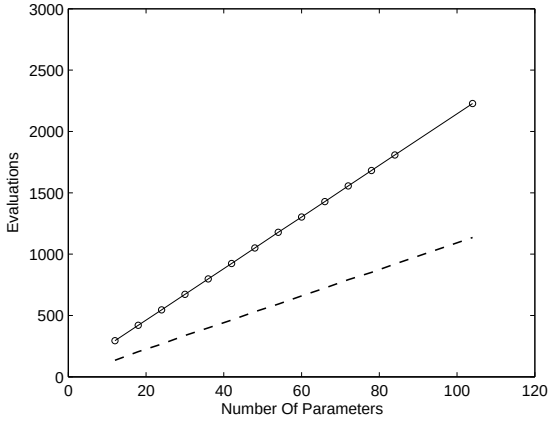



**Figure 4.27** Number of evaluations required to converge when started at the minimum. For gradient based methods only one evaluation is required.

D.H. Simplex; Powell;

These results mostly confirm the expectations from the approximate analysis summarized in Table 4.6. One major exception is that the number of steps required for convergence with the gradient descent and quasi-Newton optimizers rises at a polynomial, rather than logarithmic rate with respect to the number of parameters. The estimate of the rate of convergence was derived assuming the optimization algorithm

to be in the range where the asymptotic convergence rate holds. This assumption clearly does not hold, even on this simple problem. This may mean that there are ranges of parameter vector sizes for which the Newton method is superior, even for transformations where the gradient calculation is efficient. In the following section, these questions will be addressed with experiments on synthetic and real registration problems.

### 4.6.3 Image Registration Tests

Each of the five algorithms being discussed (downhill simplex, Powell's, gradient descent, BFGS, and Newton-Raphson) were tested by running numerous image registrations from different starting positions. As described in Section 3.2.1 the performance of the algorithms was measured by recording the time required, the positional accuracy, the number of function evaluations required and the failure rate. Pairwise statistical significance testing of the results was performed to confirm that the differences detected are meaningful. All runs were performed on a 3.6GHz Intel Xeon using a single threaded implementation.

Based on the results of Section 4.3 the tests reported here use Hessians computed using Equation 4.16 for NCC (the shorter approximation), and Equation 4.25 for MI (the generalized Gauss-Newton approximation). Similarly, based on the conclusions of Section 4.5, the scale factors of Equation 4.37 (*i.e.,* case x1) have been used for matrix based transforms, and the scale factors with internal ratio reduced, *i.e.,* case x0.1, have been used for registrations of deformable transforms. As in Section 4.5 synthetic image registration problems have been run on a range of transforms of different complexities. (In fact, this analysis is performed on the same result data, but focusing on the x1/x0.1 cases to compare the different optimizers.)

### 4.6.4 Optimizer Comparison

Figure 4.28 shows the results for registrations of the matrix based transforms using the MI cost function. Observe that there is a clear progression in number of evaluations required as the number of parameters, $\sharp(\boldsymbol{\phi})$, increases. For the Powell and downhill simplex algorithms, this progression is very rapid. Despite these evaluations being

relatively cheap computationally, since no gradient is required, the methods are much slower than other approaches for all but the simplest transform.

Based on statistical significance testing, the following overall conclusions could be drawn for the matrix based transforms

- For all cost functions, Powell required the largest number of evaluations, followed by downhill simplex, followed by BFGS.
- For MSD and NCC, gradient descent required a statistically significantly larger number of evaluations than the Newton-Raphson method. For MI, there was no statistically significant difference. This is likely due to the negative curvature present in the MI function which makes the Newton-Raphson approach less effective.
- For most cases, gradient descent was significantly less accurate than all other methods. It was only showed significantly more accuracy than another method on the rigid 2D transforms of the Sagittal Brain Slice.

The relationship between time required and number of evaluations for gradient descent, BFGS and Newton is somewhat difficult to see on Figure 4.28. It also differs between the different cost functions, MSD, NCC and MI. Therefore, Figure 4.29 shows the time and evaluation results for only these three optimizers. The Newton-Raphson method requires an equal or lesser number of evaluations than gradient descent, but this does not always translate into the corresponding time savings. This is due to the relatively high cost of computing the Hessian. For MI, which has a computationally expensive Hessian, the Newton-Raphson method is always slower than gradient descent. However, it is worth noting that the failure rate of the Newton-Raphson method on the homography cases is significantly better than gradient descent. (These results are confirmed by the results shown in Chapter 7, Figures 7.3 and 7.4).

The analysis in Section 4.6.1, and the preliminary experiments in Section 4.6.2 suggested that the BFGS method should require fewer function evaluations than gradient descent. This is only the case for the homographies; for all other transforms, BFGS required significantly more evaluations (and therefore more time). This is possibly due to its use of line-search as opposed to a trust-region approach, which could

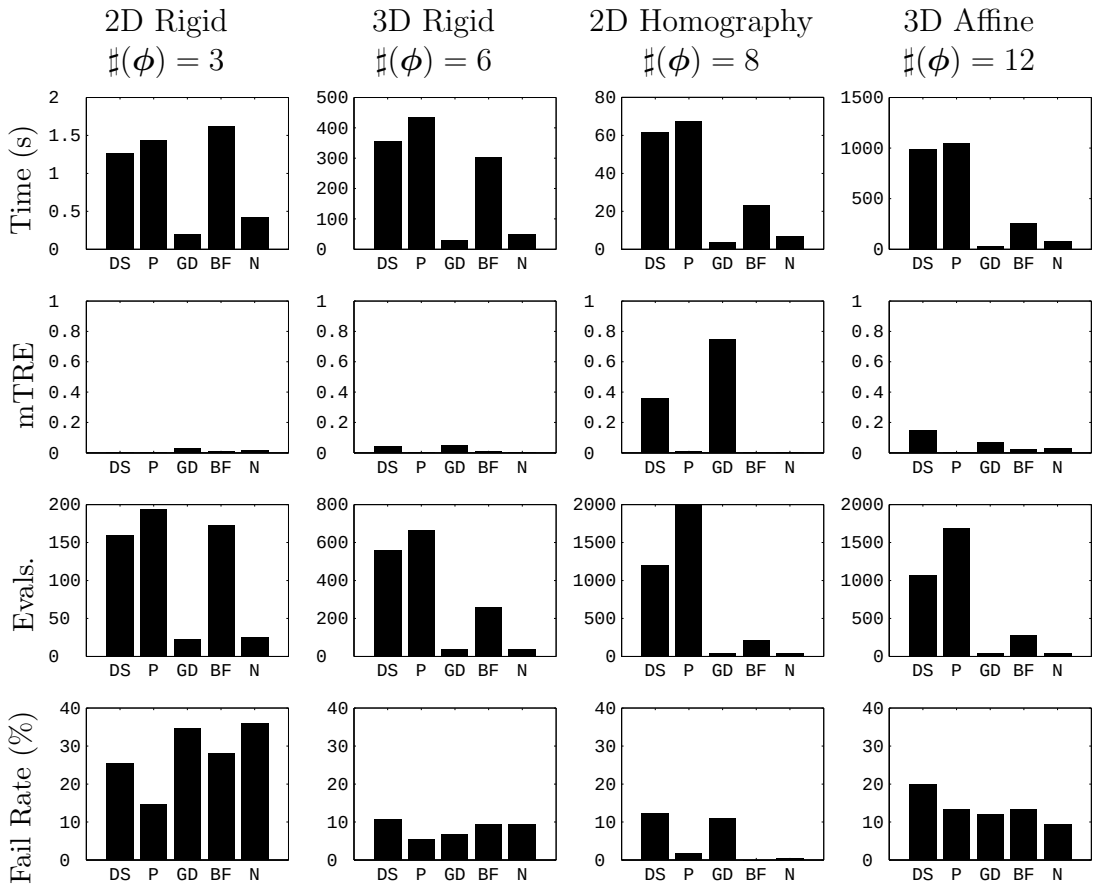



**Figure 4.28** Comparison of registration results for transforms with different numbers of parameters using the mutual information cost function. The optimizers are identified as DS – downhill simplex, P – Powell's, GD – gradient descent, BFGS – Broyden-Fletcher-Goldfarb-Shanno, N – Newton-Raphson. mTRE is in mm for all cases except the homography, where it is in pixels. The datasets used were: 2D Rigid – Sagittal Brain Slice; 3D Rigid and Affine – Brainweb; 2D Homography – Agra Fort. Note that the vertical scale is not the same in all cases.

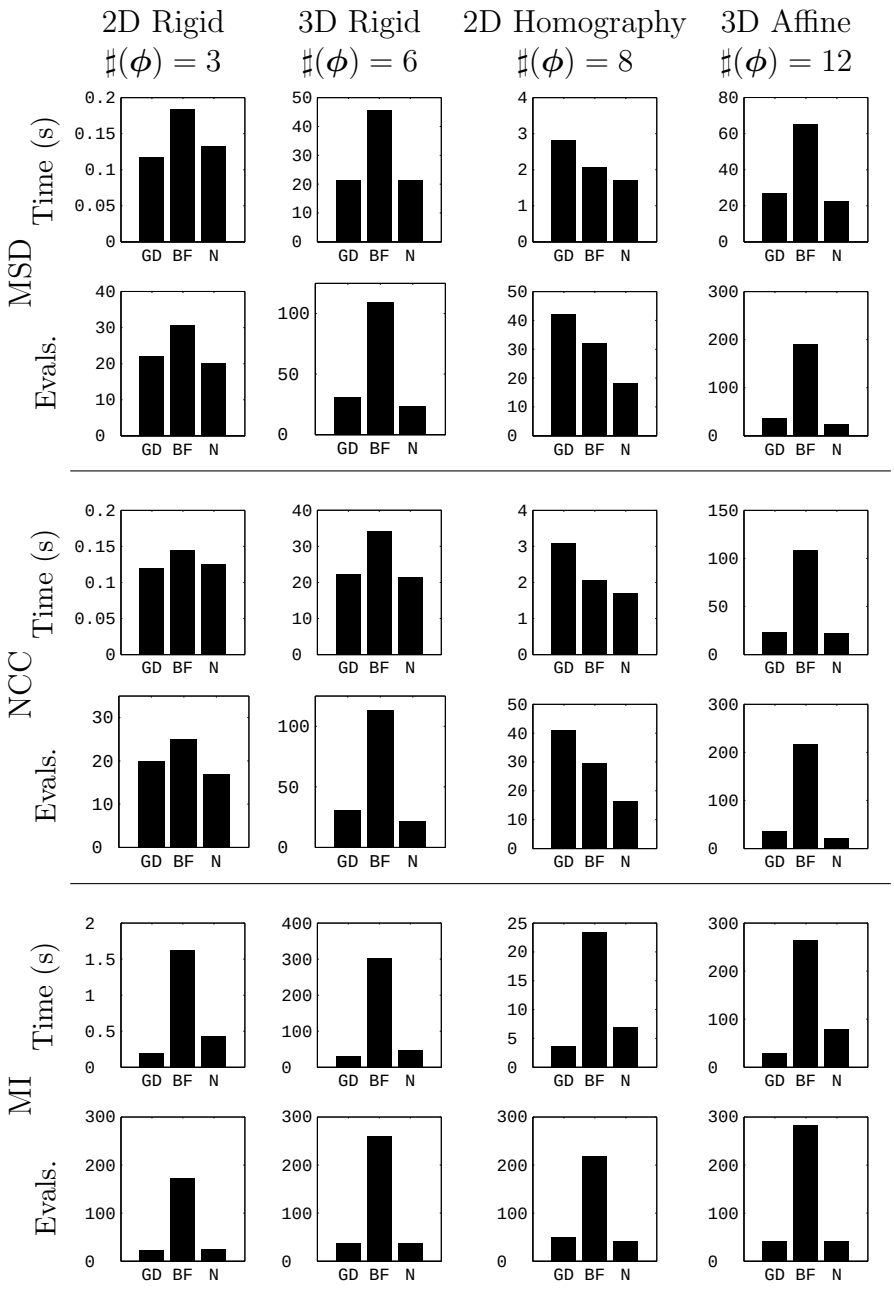



**Figure 4.29** Comparison of time and number of evaluations for matrix based transforms with different numbers of parameters shown for all cost functions on matrix based transforms. Optimizers shown are the Newton-Raphson (N), Broyden-Fletcher-Goldfarb-Shanno (BFGS) and gradient descent (GD). Note that the vertical scale is not the same in all cases. The datasets used were: 2D Rigid – Sagittal Brain Slice; 3D Rigid and Affine – Brainweb; 2D Homography – Agra Fort

require more evaluations. However, note that on the simple function of Section 4.6.2, BFGS used far fewer iterations. It seems to be a characteristic of image registration problems that makes the BFGS method use more iterations.

It is clear from the results, that as the number of parameters increases, the downhill simplex and Powell methods become impractical. Therefore they were not investigated for the deformable registrations. The Newton-Raphson algorithm also required too much time for the 1029 parameter 3D B-spline case and so only was applied to the three smaller transforms. The time and number of evaluations for the registrations with deformable transforms are shown in Figure 4.30. Once again, the results are presented for all three image difference measures investigated to show the differences. Several patterns are apparent. For MSD and NCC, for the 2D transforms which have relatively lower numbers of parameters, the BFGS optimizer requires fewer evaluations, consistent with the analysis in Section 4.6.1. However, in all other cases it requires far more evaluations than gradient descent. This is surprising, and may be partly due to the line search being inefficient. The Newton-Raphson optimizer consistently requires fewer iterations than the gradient descent method, but this only translates into a cost savings for the TPS transforms. This is because computing the gradient has the same computational complexity as computing the Hessian for these transforms. The savings in number of iterations is worthwhile. For B-splines, however, the gradient is relatively cheap, and despite the savings in iterations it is not efficient to compute the Hessian.

The accuracy of the deformable transformations was again evaluated using histograms of the residual error. These are shown in Figure 4.31. As in the case of the scaling experiments, the differences are subtle. These distributions of errors were compared run by run with a rank sum test for significance. When the difference between runs was significant, the algorithm with the lower median error was considered superior. For a majority of cases, BFGS and Newton had higher accuracy using this criterion than gradient descent. However, as can be seen in Figure 4.31, they are only slightly more accurate. In fact, although the numbers and statistical tests indicate the superiority of the BFGS and Newton-Raphson methods, this is only apparent on the 2D B-spline case of Figure 4.31. The 3D B-spline case of Figure 4.31 is interesting because it shows a bulge at the right for the gradient descent method. This indicates

that a certain number of points are being made worse by the registration process. As discussed in Section 4.5.4, explicit regularization may be the best way to approach this issue.

For thin plate spline transforms, when using an MSD or NCC cost function, the Newton-Raphson approach would seem to be a clear choice with both superior time performance, and equivalent or superior accuracy. For the B-spline transforms, the BFGS method seems to give slightly more accuracy, but at a much larger computational expense. It is probable that adjusting the stopping criteria for the gradient descent measure could enhance its accuracy without much additional cost. The gradient descent method is thus the recommended choice for optimizing B-splines transformations, and for optimizing TPS transformations with the MI image difference measure.

That these accuracy differences are small, and probably inconsequential, is also illustrated by Figures 4.32 and 4.33. These figures show difference images between the registered results for one run of each of the simulated deformable transforms. Despite the large scale of the printing, objective differences between the optimization algorithms are extremely hard to detect. The results for the BFGS may be marginally visually better (see especially Figures 4.32g and 4.33g), which agrees with the statistical testing results. Whether the additional computational expense is worth this marginal improvement is debatable.

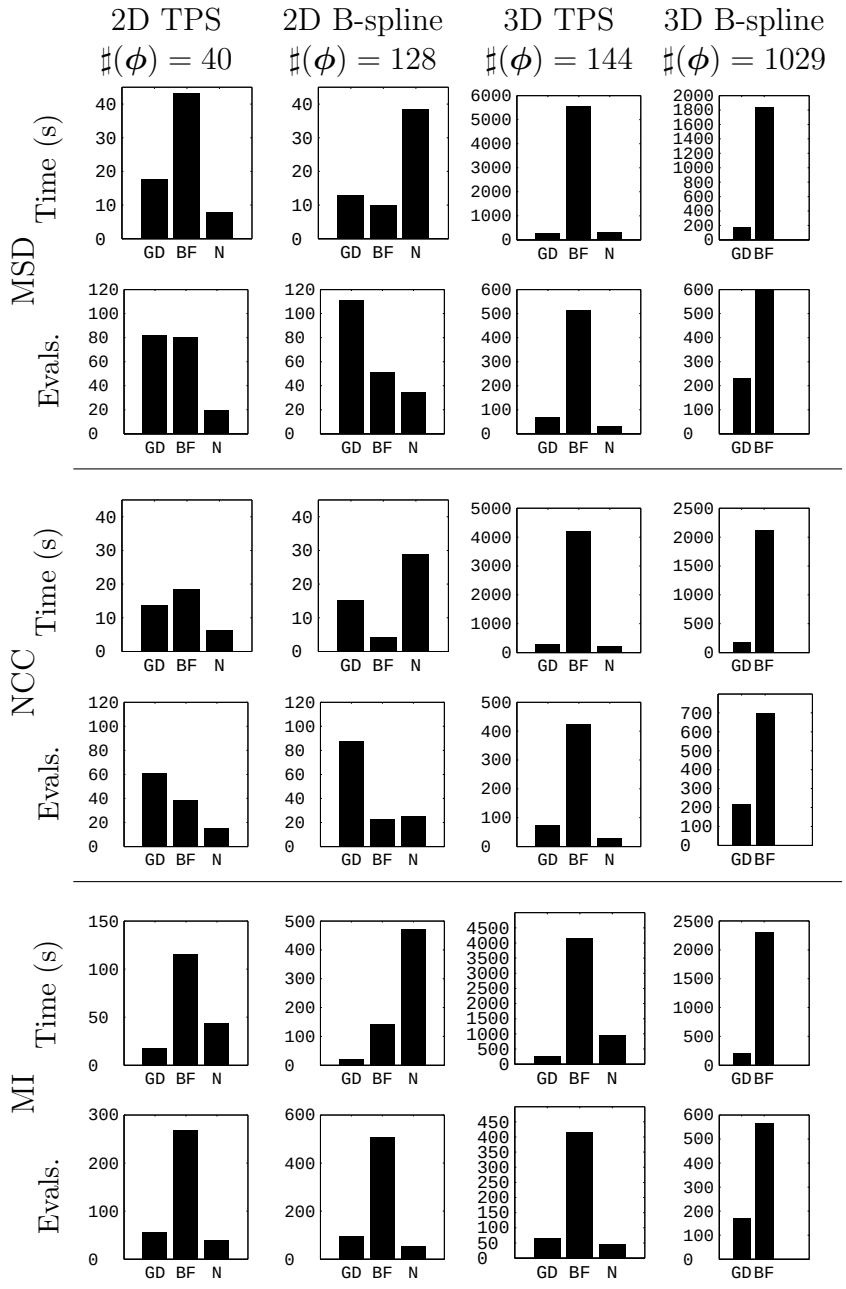
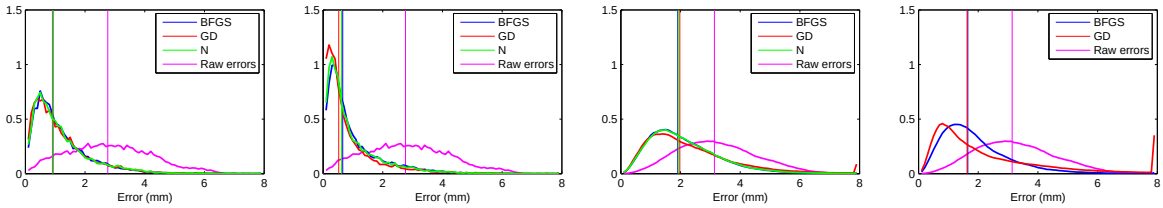


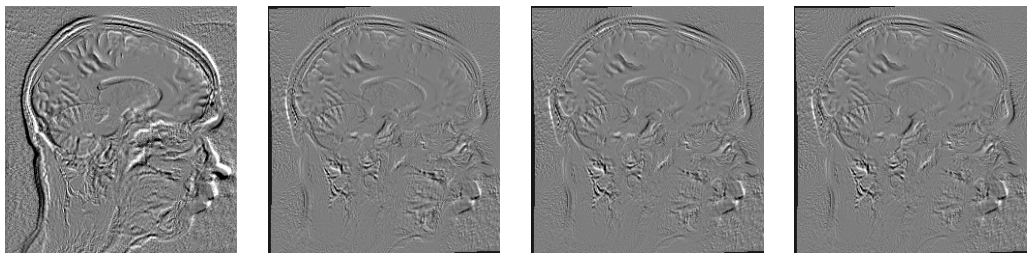

**Figure 4.30** Comparison of time and number of evaluations for deformable transforms with different numbers of parameters showing all cost functions on matrix based transforms. Optimizers shown are the Newton-Raphson (N), Broyden-Fletcher-Goldfarb-Shanno (BFGS) and gradient descent (GD). Note that the vertical scale is not the same in all cases. Datasets used were: 2D TPS and B-spline – Sagittal Brain Slice; 3D TPS and B-spline Deformable Phantom.




(a) 2D TPS (b) 2D B-spline (c) 3D TPS (d) 3D B-spline

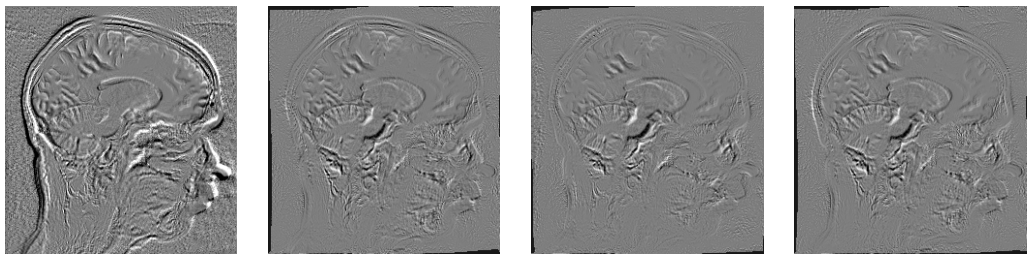
**Figure 4.31** Error distributions for results of deformable registration using the MI cost function. The graph shows the distribution of errors over a $50 \times 50$ grid ($50 \times 50 \times 50$ in 3D) of points in the image extent. Point errors for all 25 runs have been combined into one histogram for each optimizer. Vertical lines show the median error. All optimizers, achieved a reasonably successful registration, as shown by the distinct leftward shift in the distributions. Datasets used are: 2D TPS and B-spline – Sagittal Brain Slice; 3D TPS and B-spline Deformable Phantom.

Results using 2D TPS transformation



(a) Pre-registration (b) GD (c) BFGS (d) N

Results using 2D B-spline transformation



(e) Pre-registration (f) GD (g) BFGS (h) N

**Figure 4.32** Difference images before and after 2D registration with TPS transforms (top row) and B-spline transforms (bottom row). Although some significant differences between algorithms were detected, visually the differences are minor. The results for the BFGS algorithm are slightly better visually, which agrees with the numerical results.

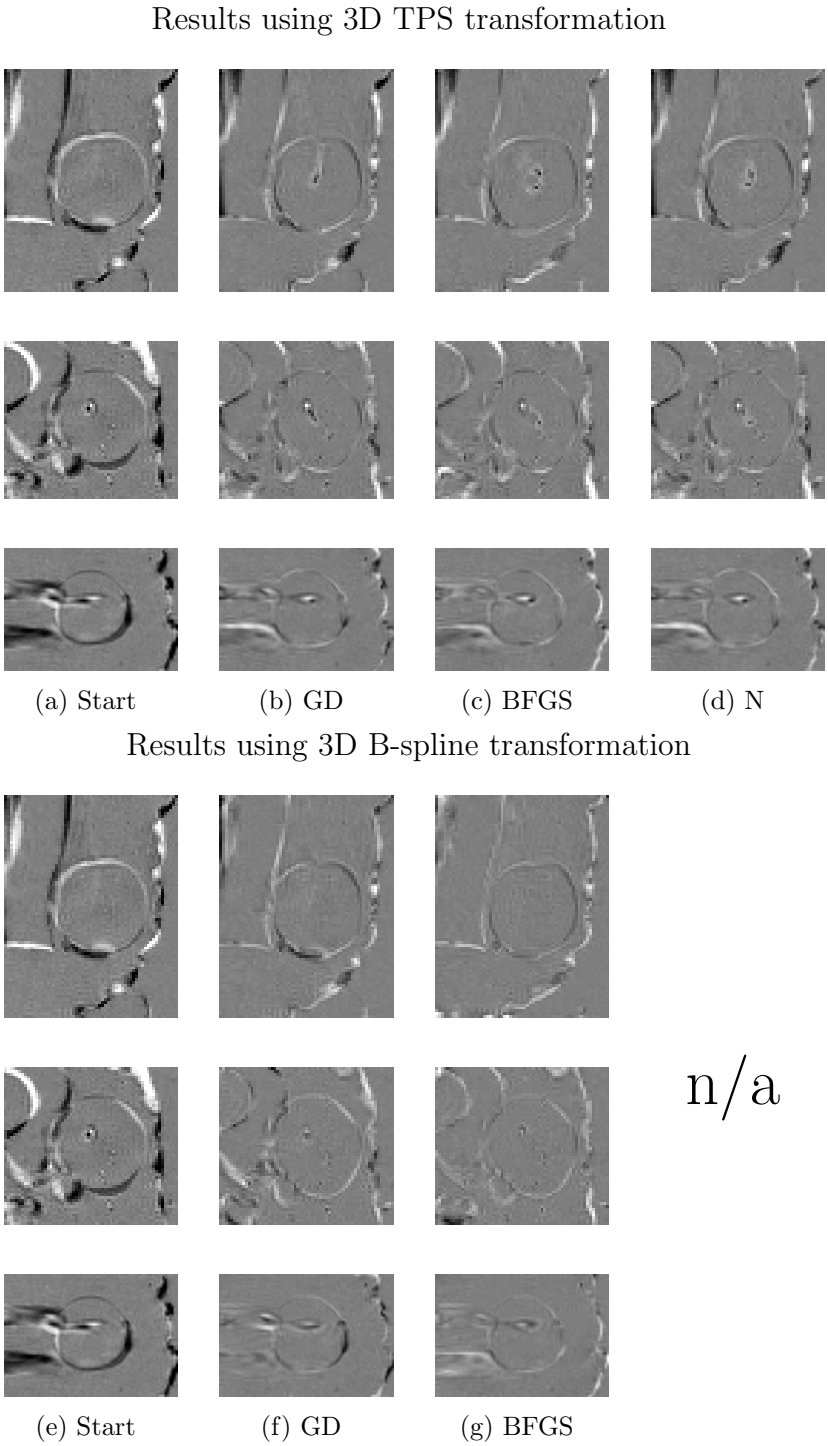



**Figure 4.33** Cross sections through difference images before and after 3D deformable registration with TPS transforms (top row) and B-spline transforms (bottom row). Although some statistically significant differences between algorithms were detected, visually the differences are minor. The results for the BFGS algorithm are slightly better visually, which agrees with the numerical results. The Newton-Raphson algorithm was too slow to use with in the 3D B-spline case.

### 4.6.5 Realistic Experiments

The preceding experiments on synthetic registration problems have strongly suggested certain conclusions. To confirm these results on realistic data, 2D and 3D rigid registrations using the Axial Brain Slices and Vertebra datasets described previously were examined. These datasets are multimodal, so only mutual information was used to register them. The results for registrations using all 5 of the algorithms under test are shown in the leftmost two columns of Figure 4.34. As in the simulated registrations, downhill simplex and Powell's algorithm require by far the most iterations and the most time. The BFGS algorithm requires significantly more iterations than either gradient descent or the Newton-Raphson method which causes it to take more time. This is consistent with the results on the simulated registration problems, but differs from the expectations of the analysis in Section 4.6.1 and the simple experiments in Section 4.6.2.

Registrations were also performed between the different cases of the deformable phantom object. The time required and number of iterations is shown in the rightmost two columns of Figure 4.34. There are not enough samples to safely draw conclusions about statistical significance, but the results are in agreement with the results of the simulated experiments. In this case, there is no gold standard transformation to compare to. However, a comparison for accuracy can be performed using the distances between points in the segmented mesh representation of the balloon, as described in Section 4.2.4. The distribution of these errors can be compared as it was for the previous experiments, and these results are shown graphically in Figure 4.35. By using the statistical comparison technique to discern differences, for the TPS transform gradient descent significantly outperformed Newton in Cases 1, 2 and 4, and for the B-spline case BFGS significantly outperformed gradient descent in cases 2, 3 and 4. It is interesting that the results for TPS are opposite to those reached in the simulated cases. It is important to note that the differences in median, while statistically significant, are very small.

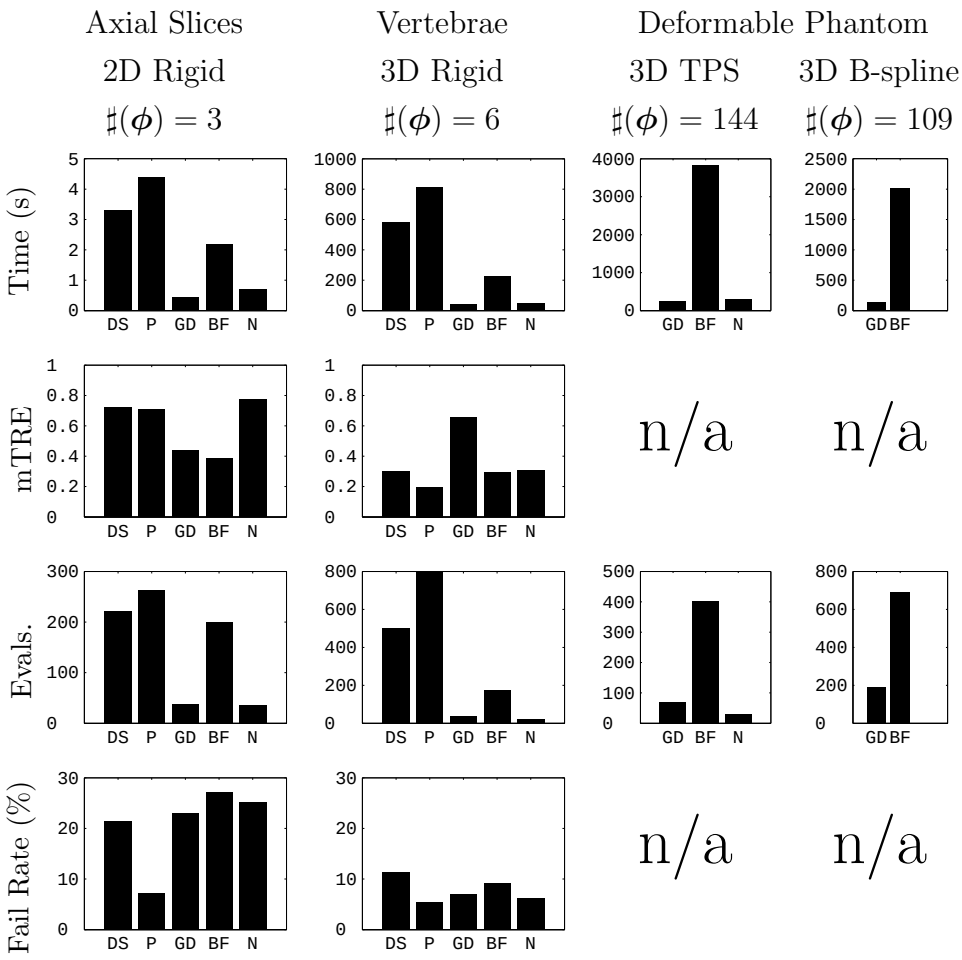



**Figure 4.34** Optimizer comparison on realistic datasets. Top row, rigid 2D registrations of axial slices. Second row, rigid 3D registration of Vertebra dataset. Bottom two rows report time and number of evaluations for deformable registration of the phantom dataset. Accuracy is described using histograms of point error, and there were no failures for this case.

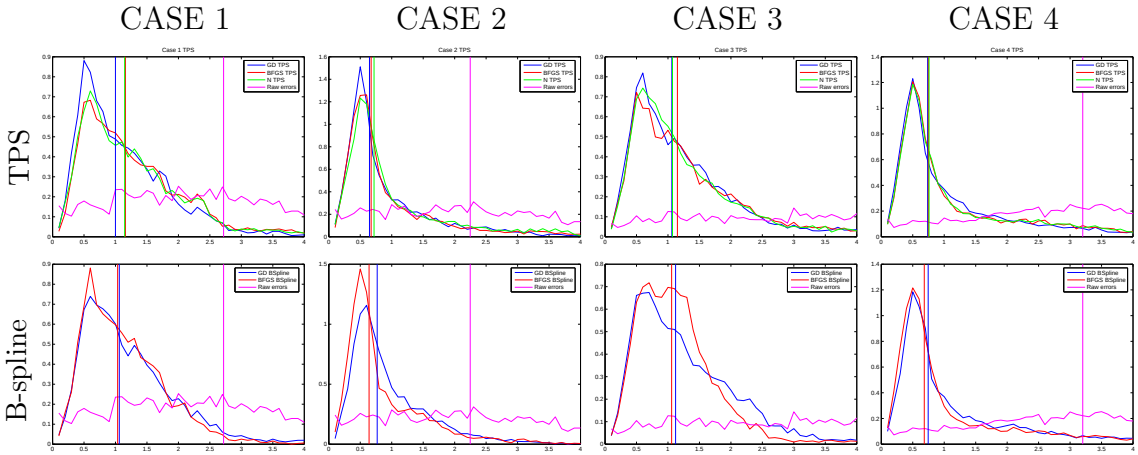



**Figure 4.35** Distributions of errors for registrations of deformable phantom volumes. Top row, registration using 3D thin plate spline. Second row, registration using 3D B-spline. Cases are (1) partially inflated to deflated, (2) partially inflated to fully inflated, (3) fully inflated to deflated, (4) fully inflated to partially inflated. For case 3, the median error for the raw data is off the graph, at 5.19mm.

### 4.6.6 Summary

The differences between optimizer performance has been examined for the MSD, NCC and MI image difference measures across a number of different transforms. The results show very clearly that the Powell and downhill simplex algorithms are slower than others, and that this slowness rapidly becomes more severe as the number of parameters increases. These algorithms are not an ideal choice in terms of efficiency for any case. Powell's algorithm, however, proved to be very robust to issues of scaling, and generally performed very reliably.

The differences between the gradient descent, BFGS and Newton method are highly dependent on the image difference measure, and the number of parameters in the transform. The BFGS algorithm requires more evaluations than the gradient descent method in nearly all cases, which is unexpected based on the analysis and preliminary experiments. Some of this may be due to differences between the line search approach of the BFGS algorithm and the trust-region approach of the gradient descent one. However, these differences should have affected the preliminary experiment as well. It is conjectured that the apparent difficulty that the BFGS method

encounters may be due to some violation of the assumptions that it is based on. The BFGS method assumes that the function is smooth and the Hessian is positive definite. Image difference measures are always a bit noisy. The cases where the BFGS method takes a particularly large number of iterations to complete are when using the MI measure, and for the 3D B-spline transforms. The MI function is saddle-shaped, meaning its Hessian is indefinite, while the B-spline transforms have a very sparse Hessian which is nearly singular. Similar, somewhat disappointing, performance is shown for the BFGS method in Chapter 7 when working with inverse compositional MI.

Whether Newton-Raphson or gradient descent would be the better method to choose depends on both the type of transform and the image difference measure. The Newton-Raphson method requires fewer, but more computationally expensive iterations. When computing the gradient is computationally expensive, such as with the TPS transforms, then the Newton-Raphson approach is quite efficient. When computing the gradient is computationally cheap, then the savings in numbers of iterations is outweighed by the cost of calculating the Hessian, and the gradient descent method is more effective. These effects are strongest for the MSD and NCC cost functions; the Newton-Raphson method is less effective on MI, due to the negative curvature in the cost function. It is worth noting that the Newton-Raphson approach seems to be significantly more reliable for the homography, which may be due to the non-linear nature of this transform.

## 4.7 Conclusions

This chapter has addressed the issue of optimizer selection for efficient image registration in three parts. First, in order for any comparison of Newton-Raphson algorithms to be valid, it was necessary to compute valid approximations of the image difference measure Hessian. Second, as parameter scaling has a large effect on optimizer performance, a means of consistently and reliably scaling the parameters was necessary. Finally, in a context where these issues were resolved, a valid comparision could be made between algorithms.

Section 4.3 examined how the Hessian may be approximated for image registra-

tion problems. It was shown that the Gauss-Newton approach to approximating the Hessian relies on the cost function having the form $D(I(\boldsymbol{\phi}))$ where $I$ is a large dimension vector. This clearly applies to the image registration problem. In such a case, the Taylor series expansion has two second order terms, and the Gauss-Newton approximation is to drop the one that has second order derivatives of the innermost function, $I$. This technique was applied to derive approximate Hessians for the NCC and MI image difference measures. The MI approximate Hessian was shown to be superior to existing approximations in the literature.

While it is often recommended that optimization problems be well-scaled, no principled method of determining these scale factors for image registration problems existed. A method for automatically determining these scale factors from the problem geometry alone was derived based on an optimal diagonal preconditioner. Experiments showed that these scaling factors are important for optimizer performance, and that for matrix based transforms, using the scaling factors given by Equation 4.37 resulted in clearly superior performance in terms of failure rate, with time required and mTRE also showing better results.

For the deformable transformations the effect of scaling the parameters was much more subtle. The experiments shown that a set of scaling factors modified to have less extreme ratios between terms was more effective on these cases. It is conjectured that this difference occurs because the optimal scaling factors allow very large variations in certain parameters of the deformable transformation. No explicit regularization was applied to the problems explored here, and applying a regularization technique to keep these parameters from getting too large might be an appropriate way to address this issue.

Section 4.6 compared five different optimization algorithms across a range of image difference measures and transformations. The overall conclusions are that the downhill simplex and Powell algorithms are too inefficient for any but the smallest registration problems. Choosing between the gradient descent, BFGS and Newton-Raphson techniques, however, depends on the specific details of the registration problem.

For the NCC and MSD cost functions, the Newton-Raphson algorithm requires fewer iterations and is as reliable as either gradient descent or BFGS for moderate numbers of parameters. However, for the MI cost function, the Newton-Raphson

algorithm is less effective, probably due to the negative curvature of the MI cost function. Nevertheless, for moderate numbers of parameters (see, *e.g.,* Figure 4.30), the Newton-Raphson algorithm requires fewer iterations for MI as well. Whether this savings in evaluations is worthwhile depends entirely on the computational expense of evaluating the gradient. Even for the TPS transform, which has an expensive, $O(N^2)$, gradient calculation, the Newton-Raphson method proves slower than the gradient descent method. It is worth pointing out, however, that for the registrations of 2D homographies using MI the Newton-Raphson method was significantly more reliable than the gradient descent method, and faster than the BFGS method.

The BFGS method required a larger number of iterations than the preliminary analysis suggested. This makes the gradient descent method quite attractive in comparison. However, overall the gradient descent method seems to give slightly less accuracy than either BFGS or the Newton-Raphson method. As the difference in accuracy is small, however, the gradient descent method may still be the approach of choice, particularly in a time-sensitive application. It is probably possible to achieve better accuracy with gradient descent by adjusting the stopping criteria, although that was not explored here.

### 4.7.1 Future Directions

There are several aspects of this research that would be interesting to explore further. Both trust-region and line search approaches were used, but the experimental setup does not allow any conclusions about their relative merit to be drawn. It would be interesting to directly examine the difference in performance between them.

It was presumed in this work that the complexity of computing the Hessian was $O(N^2)$. However, if the Hessian is sparse, this complexity may actually be lower. Changing this complexity will change the balance between the gradient descent, Newton-Raphson and BFGS methods, and sparse-Hessian based approaches could lead to efficient Newton-Raphson optimization of B-spline transforms.

The performance of the BFGS optimization algorithm was poorer than expected, requiring a relatively large number of iterations. It is conjectured that one reason for this could be indefiniteness of the Hessian. Certain trust-region quasi-Newton approaches, such as the symmetric rank-one update are reputed to be more effective

on problems of this kind [151]. However, implementation of trust-region quasi-Newton methods is not a trivial matter [151], and more research would be needed in this area.

Finally, it is clear from this work that different optimization approaches have different advantages and disadvantages. Ultimately the ideal solution for image registration problems may be to combine approaches. This could be done simply by using different optimizers at different levels of the multiresolution pyramid, or in a more sophisticated way, by actually alternating Newton, gradient descent or BFGS steps at different points in the optimization. For example, based on the relative strengths and weaknesses of the gradient descent and B-spline methods, one could imagine doing a number of gradient descent steps to approach the solution of a B-spline transform, followed by a short period of BFGS line-searches to improve the accuracy.

# Chapter 5

# Anytime Algorithms for Faster Registration

Recall that the direct image registration problem is expressed as the minimization of an image difference measure, $D$, defined between the fixed image, and a moving image that is being warped to match it. As discussed at length in the previous chapter, the optimization procedure requires repeated evaluation of this measure. One way to achieve computational efficiency is to use only some of the pixels for the registration process. However, using only some of the image can lead to problems with accuracy and reliability. This chapter focuses on the question of *how many* of the pixels should be used.

The image registration problem clearly faces a tradeoff between computation time and accuracy. In the field of artificial intelligence, the process of intelligently dealing with tradeoffs such as this one is referred to as *deliberation control*. This chapter explores the idea of mapping these techniques developed in the field of real-time artificial intelligence, onto the image registration problem. Deliberation control methods rely on two key components: algorithms that support partial evaluation, and knowledge about how those algorithms perform after different amounts of computation. A class of algorithms supporting partial evaluation are the *anytime algorithms* [65, 106], which provide a solution when run for any length of time. The solution quality is guaranteed to improve with the amount of computation performed. This chapter proposes the use of a deliberation control framework using anytime algorithms to arrive

at a principled solution to the speed vs. accuracy trade-off in this problem. The first step is an off-line training process to learn the properties of the difference measure under consideration, in terms of accuracy vs. computation time, by analyzing image pairs for which the transformation parameters are known. Given a new pair of images to align, this knowledge is used to determine the number of pixels that need to be considered at each step of the optimization. This chapter explores the effectiveness of this approach using two common difference measures, mean squared difference and mutual information, and a gradient descent optimizer. The algorithm has been tested on several types of images: images of everyday scenes, multi-modal medical images and earth observation data (*i.e.,* Landsat and Radarsat images). In all cases, using a deliberation control approach is faster than computing the transformation using all the image data and gives more reliable results than simply performing the optimization using an arbitrary, fixed, percentage of the pixels.

The remainder of this chapter is organized as follows. The following section reviews previous work on performing image registration with only partial data, and on methods of deliberation control using anytime algorithms. The details of how deliberation control has been implemented in the context of image alignment are then given in Section 5.2. Finally, Sections 5.3 and 5.4 describe the experimental setup, results and conclusions.

### Publications

The majority of this chapter is based on work published in:

[39] Rupert Brooks, Tal Arbel, and Doina Precup. Anytime similarity measures for faster alignment. *Computer Vision and Image Understanding*, 110(3):378–89, 2008.

[38] Rupert Brooks, Tal Arbel, and Doina Precup. Fast image alignment using anytime algorithms. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI2007)*, pages 2078–2083, Hyderabad, India, January 2007.

## 5.1 Previous Work

That only some of the pixels are necessary to compute the image difference measure was realized early on by Barnea and Silverman, who proposed the Sequential Similarity Detection Algorithms (SSDA) [15]. They were interested in the problem of finding the best translational match between a model image patch and an image of interest. As was the state of the art at the time, they solved for this with an exhaustive search over possible integer translation positions. They defined a threshold such that if the image difference measure exceeded this threshold, it was certain to be a bad match and could be rejected without further computation. (At about the same time, Nagel and Rosenfeld [148] suggested something similar involving pixel selection, which will be discussed in the next chapter.)

Defining this threshold requires some kind of bound on the image difference measure, and this work did not deal at all with the use of derivatives for image measure calculation. Nevertheless, the idea of using only some of the pixels was popular, many authors (*e.g.,* [108, 124, 181, 185]) applied it in somewhat ad-hoc ways by using some fixed percentage of the image data. For example, the ITK manual [109] recommends using about 30% of the pixels for mutual information registration, a number presumably arrived at by experiment. Recently, Klein *et al.* [119, 120] have investigated the performance of various optimization algorithms under different levels of regular subsampling of the image while solving for complex deformations modeled by B-Splines.

This recent work has neglected to consider that different levels of computation may be required at different times in the optimization. It could be argued that the original SSDA [15] has this property, but instead of an arbitrary percentage of pixels, they propose an arbitrary cutoff on the difference measure value. This is fundamentally a branch and bound search, and does not generalize well to iterative optimization. The issue of whether different types of images require different levels of calculation has also been ignored.

### 5.1.1 Deliberation Control with Anytime Algorithms

In many artificial intelligence tasks, *e.g.,* planning, the quality of the solution obtained depends on the amount of time spent in computations. Hence, trade-offs are

necessary between the cost of sub-optimal solutions and the cost of spending time doing further computation. This process, called *deliberation control*, has been investigated in the context of real-time artificial intelligence and a number of approaches have been proposed [107]. In order for such an approach to be practical, it must be possible to partially evaluate solutions to the underlying problem and the relationship between the amount of computation performed and the quality of the solution must be understood.

A class of algorithms supporting partial evaluation are the *anytime algorithms* [65, 106], which provide a solution when run for any length of time. The solution quality is guaranteed to improve with the amount of computation performed. Deliberation control strategies using anytime algorithms have been applied to both theoretical and practical problems including robot control [195], solution of decision problems [152], and shape extraction in image processing [125].

Interestingly, Fischer and Niemann [77] proposed an "any–time" approach to high level control of image parsing, but they seem to have coined the term independently. They propose to perform iterative optimization of multiple hypothesis fits independently in parallel, taking the best after a certain cutoff is reached. The work presented here is based on the formal concept of anytime algorithms proposed in [65, 106], and deals with the computation level at each evaluation of the difference measure, $D$.

## 5.2 Deliberation Control in Image Registration

To formulate an effective deliberation control strategy using anytime algorithms it is necessary to have meta-level knowledge of their performance as a function of the amount of computation performed [65, 106]. This knowledge is stored in a *performance profile*. Performance profiles may be based on theoretical knowledge of the algorithm, on empirical testing of its performance at different computation levels, or a combination of the two. In any case, the decision to continue computation will be based on an estimate of the accuracy of the current result, and an estimate of the potential improvement if the algorithm continues to run [65, 92, 106, 126].

The simplest type of performance profile is a static one, which predicts accuracy as a function of the amount of computation completed. However, for the problem of

interest, this is equivalent to simply using a fixed, arbitrary percentage of the pixels. If feedback about the current run of the algorithm is available, it can be incorporated in a more sophisticated approach. A *dynamic performance profile* [92, 126] uses feedback to estimate the accuracy as the algorithm progresses. It is described by two functions: $\hat{a} = P_{fwd}(f, p)$, maps the percentage of computation completed, $p$, and the feedback parameter, $f$, to an expected accuracy, $\hat{a}$. The other, $\hat{p} = P_{rev}(f, a)$, maps $a$ and $f$ to the expected percentage of computation required, $\hat{p}$. Conceptually, these two functions are inverses. However, both have to be maintained in general, to facilitate the decision making. A controller can use these functions to gradually increase the amount of computation performed, until the estimated accuracy is adequate for the task.

As mentioned in Chapter 1, the most computationally intensive part of image alignment is the repeated evaluation of the difference measure $D$ and its gradient $\nabla D$. The optimization algorithm needs this information in order to take a step in parameter space towards the optimal setting. Note that the calculation only has to be accurate enough to ensure that the next step is correct; determining these values exactly is not necessary. Therefore, it is proposed to implement image difference measures and their gradients as anytime algorithms, and to learn performance profiles describing their accuracy at different levels of computation.

### 5.2.1 Anytime Image Difference Measures

For an image difference measure to act as an anytime algorithm, its implementation must be able to support partial evaluation, as well as to continue an interrupted calculation efficiently. To achieve this, each image difference measure, $D(\boldsymbol{\phi})$, is redefined as a function $D(\boldsymbol{\phi}, p)$ of both the parameters, $\boldsymbol{\phi}$, and the percentage of pixels to be used, $p$. To avoid biasing the computation towards one area of the image, the pixels are processed in a random order. Anytime versions of the popular mean squared difference and mutual information image difference measures have been implemented here, although the approach is general and can be applied to other difference measures.

### Mean Squared Difference

Recall from Section 3.1.2 that the mean squared difference $D_{MSD}$ is simply an average of the squared differences in intensity between corresponding pixels. Thus it is very easy to redefine it as an anytime measure, by simply averaging over the first $pN$ randomly selected pixels.

$$D_{MSD}(\boldsymbol{\phi}, p) = -\frac{1}{\lfloor pN \rfloor} \sum_{i=1}^{\lfloor pN \rfloor} (I_f(\boldsymbol{X}_i) - I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}))^2$$

where $N$ is the total number of pixels in the image. The gradient of this measure is also easy to compute as an anytime algorithm:

$$\nabla_{\boldsymbol{\phi}} D_{MSD}(\boldsymbol{\phi}, p) = \frac{2}{\lfloor pN \rfloor} \sum_{i=1}^{\lfloor pN \rfloor} [(I_f(\boldsymbol{X}_i) - I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})))$$
$$\cdot \nabla_{\boldsymbol{W}} I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})) \nabla_{\boldsymbol{\phi}} \boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})]$$

Note that because the difference measure and its derivative consist of simple sums, it is easy to add more pixels to the computation, and both the difference measure and the gradient can be updated incrementally in the usual fashion.

### Mutual Information

As described in Section 3.1.4, the mutual information implementation used here is the one proposed by Thévenaz and Unser [185], which relies on a B-spline windowed representation of the joint probability distribution of the intensity levels in the two images. This measure is implemented by keeping an unnormalized joint histogram of the intensities in both images, $\boldsymbol{P}_{kl}$, and a set of derivatives of this joint histogram, one for each transformation parameter. The anytime implementation also maintains these arrays. Specifically the probability distribution is given by

$$\boldsymbol{P}_{kl}(\boldsymbol{\phi}, p) = \sum_{i=1}^{\lfloor pN \rfloor} \beta_0 \left( k, \left\lceil \frac{I_f(\boldsymbol{X}_i) - b_{f_0}}{d_f} \right\rceil \right) \beta_3 \left( l, \left\lceil \frac{I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})) - b_{m_0}}{d_m} \right\rceil \right)$$

which is simply Equation 3.5 modified to be over the first $pN$ pixels. The normalization factor of $\boldsymbol{P}_{kl}$ at a particular computation level $p$ is:

$$\alpha(\boldsymbol{\phi}, p) = \sum_{k=1}^{K} \sum_{l=1}^{L} \boldsymbol{P}_{kl}(\boldsymbol{\phi}, p) = \lfloor pN \rfloor$$

Similarly, the gradient information is kept in tables of the form

$$\nabla_{\boldsymbol{\phi}} P_{kl}(\boldsymbol{\phi}, p) = \sum_{i=1}^{\lfloor pN \rfloor} \frac{\partial \beta_3 \left( l - \frac{I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})) - b_{m_0}}{d_m} \right)}{\partial I_{m_i}} \cdot \frac{\partial I_{m_i}}{\partial \boldsymbol{\phi}}. \tag{5.1}$$

which is Equation 3.7 extended to support partial evaluation controlled by the level $p$. When needed, these tables can be normalized using the factor $\gamma = (d_m \lfloor pN \rfloor)^{-1}$.

The algorithm maintains the unnormalized probability distribution, and its unnormalized partial derivatives, as described above, which can be easily updated when more pixels are added, because the table entries are simple sums. The image difference measure and its gradient are then computed as needed from this table using

$$D_{MI}(\boldsymbol{\phi}, p) = \sum_{k=1}^{K} \sum_{l=1}^{L} \boldsymbol{P}_{kl}(\boldsymbol{\phi}, p) \log \frac{\boldsymbol{P}_{kl}(\boldsymbol{\phi}, p)}{\left(\sum_{k'} \boldsymbol{P}_{k'l}(\boldsymbol{\phi}, p)\right) \left(\sum_{l'} \boldsymbol{P}_{kl'}(\boldsymbol{\phi}, p)\right)}$$

and

$$\nabla_{\boldsymbol{\phi}, p} D_{MI}(\boldsymbol{\phi}, p) = \sum_{k}^{K} \sum_{l}^{L} \nabla_{\boldsymbol{\phi}, p} \boldsymbol{P}_{kl}(\boldsymbol{\phi}, p) \log \frac{\boldsymbol{P}_{kl}(\boldsymbol{\phi}, p)}{\sum_{k'}^{K} \boldsymbol{P}_{k'l}(\boldsymbol{\phi}, p)}$$

which are just simple modifications of Equations 3.6 and 3.8.

Besides the property of being able to be stopped and restarted, anytime algorithms must also produce a better result the longer they are allowed to run. Both of these measures and their derivatives may be considered anytime algorithms, as the bound on the maximum error decreases monotonically as the percentage of computation increases. A proof of this is presented for the MSD and MI measures in Appendix E. It should be straightforward to extend the proof to other cost functions if needed.

### 5.2.2 Performance Profiles

Exploiting the partial evaluation possibilities of the anytime difference measures requires dynamic performance profiles describing their expected accuracy at different computation levels and feedback values. These profiles are created off-line, before the registrations are performed. They are created by investigating the properties of the gradient of the image difference measure on various image pairs which are characteristic of the type of data to be aligned. By "characteristic" consider that most of the applications considered involve the repeated registration of pairs of images which come from similar instruments being used in similar contexts. For example, an application might be the registration of multiple photos into a mosaic, in which case, the images in question would be images of outdoor scenes. Perhaps a more compelling example application of this algorithm would be for the registration of intraoperative multimodal imagery during surgery. In this context the images in question could be corresponding MRI and ultrasound pairs and it would be necessary to register them so that the current position of the patient, which is being mapped by the ultrasound, is brought into alignment with the usually higher resolution preoperative data. In either case, prior to using the algorithm, typical examples of image pairs to be registered would be obtained, and a performance profile generated from them.

The property of the image difference measure which must be measured by this profile depends on how it is being used by the optimization algorithm. The implementation described in this thesis uses a simple steepest descent optimizer (described more fully in Section 5.3.2). Carter [50] has analyzed a similar class of optimizers and has proven their convergence using the following measure of relative error:

$$\epsilon = \frac{||\nabla_{true} D - \nabla_{measured} D||}{||\nabla_{true} D||}, \tag{5.2}$$

where $\nabla_{true} D$ is the correct value of the gradient of $D$ and $\nabla_{measured} D$ is the value actually computed, which contains some error. This definition is adopted for the performance profiles.

A dynamic performance profile requires a feedback parameter which indicates the progress of a particular calculation run. Based on the Equation 5.2 the gradient magnitude is an ideal candidate. Given a set of samples of gradients computed at

different computation levels at different points throughout the parameter space, a performance profile can be created as a table mapping computation level and gradient magnitude to accuracy. To construct such a table the gradient was sampled at different computation levels at many points in the transformation space. For each computation level, $p$, the gradient magnitudes were grouped into bins and the expected accuracy of the gradient, $\bar{E}_p$, was computed for each bin as follows:

$$\bar{E}_p = 1 - \sum_i \frac{||\nabla_{100\%} D(\boldsymbol{\phi}_i) - \nabla_p D(\boldsymbol{\phi}_i)||}{||\nabla_{100\%} D(\boldsymbol{\phi}_i)||}. \tag{5.3}$$

Here $\nabla_{100\%}(\boldsymbol{\phi}_i)$ is the gradient computed using all the pixels and $\nabla_p(\boldsymbol{\phi}_i)$ is the gradient computed at computation level $p$. These correspond to $\nabla_{true} D$ and $\nabla_{measured} D$ in Equation 5.2, respectively. The $\boldsymbol{\phi}_i$ are the sampled points in the transform space.

When a new pair of images is to be registered, the optimizer uses this table to progressively increase the amount of computation performed until the estimated accuracy reaches a criterion for acceptability. The analysis in [50] indicates that significant computational gains can be made, with a small ($\epsilon \approx 10\%$) reduction in accuracy. Therefore the optimizer seeks an expected accuracy of 90% for each gradient that it computes.

A simple example of how the table can be used to control computation is shown in Figure 5.1. This table can act as a performance profile where two functions, $\hat{a} = P_{fwd}(f, p)$ and $\hat{p} = P_{rev}(f, a)$ are implemented through simple lookup. For example, suppose an optimizer requires an accuracy of 98%. On an initial probe, the feedback parameter, $f$, is 0.55. By examining the third row applying to $0.4 < f \leq 0.6$, one predicts that with $p = 8\%$, the desired accuracy level will be obtained (arrow 1). After performing 8% of the computation, however, suppose $f$ is now 0.1. Thus the accuracy is only 93% (arrow 2) meaning more computation is required. The required $p$ is now estimated as 16% (arrow 3), and so on. In the image registration case, the parameter $p$ will be the percentage of pixels processed in the image. The feedback parameter is the magnitude of the gradient.

Percentage Completed ($p$)

Feedback ($f$)

| | 0.5% | 1% | 2% | 3% | 4% | 6% | 8% | 16% | 32% | 64% | 100% |
|---|---|---|---|---|---|---|---|---|---|---|---|
| f≤0.2 | 31% | 53% | 63% | 76% | 89% | 91% | 93% | 98% | 99% | 100% | 100% |
| 0.2<f≤0.4 | 55% | 71% | 88% | 90% | 91% | 93% | 95% | 99% | 100% | 100% | 100% |
| 0.4<f≤0.6 | 72% | 88% | 90% | 93% | 95% | 97% | 98% | 99% | 100% | 100% | 100% |
| 0.6<f | 80% | 89% | 92% | 96% | 99% | 100% | 100% | 100% | 100% | 100% | 100% |



**Figure 5.1** Dynamic performance profile example: Values in the table represent the expected accuracy of the results.

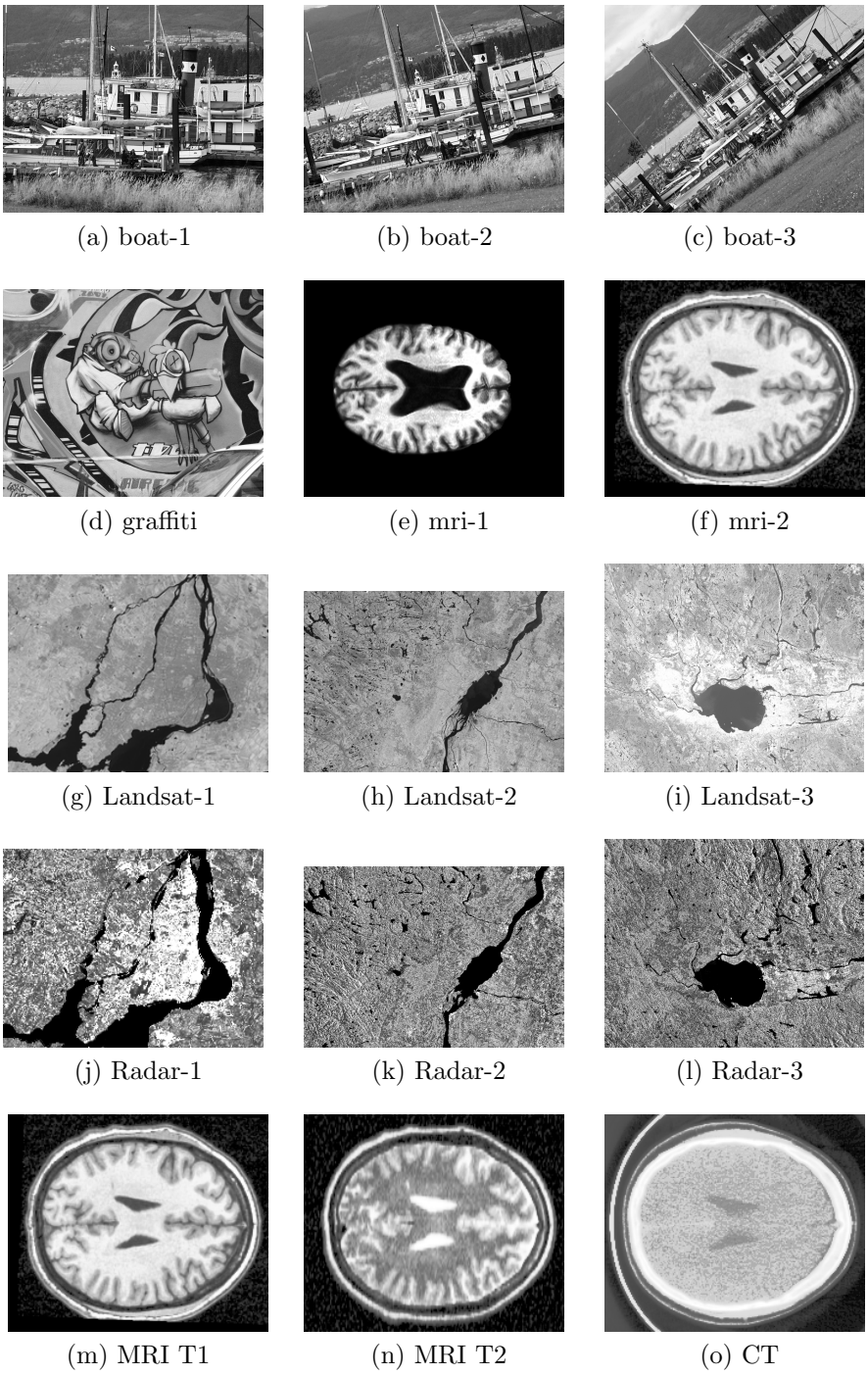
## 5.3 Experiments

To test the anytime algorithm approach, a number of performance profiles were generated off-line, and alignments were performed on a different set of images on-line during testing. Four classes of images (shown in Figure 5.2) with at least two image pairs each were used in the testing process. The first image class consisted of typical digital photos (DP) (images a-d) Both a) and d) were self-aligned and a) was aligned affinely against several images of the same scene taken from different camera positions (images b, c) using both difference measures[1]. The second class of images (M1) (images e, f) were slices from T1-weighted magnetic resonance imaging (MRI) volumes which were self-registered using the $D_{MSD}$ measure. The third class of images (EO) are patches from georeferenced, orthorectified Landsat 7 and Radarsat imagery that were registered to each other using the mutual information measure (rows 3 and 4)[2]. The final class of images (M2) are slices from previously registered volumes in different medical imaging modalities, including T1 and T2 weighted MRI, and computed tomography (CT) (last row). These images were aligned to each other using the mutual information measure[3].

[1]Images 5.2a, 5.2b, 5.2c, and 5.2d from K. Mikolajczyk `http://www.inrialpes.fr/lear/people/Mikolajczyk/`.

[2]Landsat and Radarsat images (5.2g, 5.2h, 5.2i, 5.2j, 5.2k, 5.2l) from Natural Resources Canada `http://geogratis.gc.ca`.

[3]Medical images (5.2e, 5.2f, 5.2m, 5.2n, and 5.2o) courtesy Montreal Neurological Institute.



(a) boat-1 (b) boat-2 (c) boat-3

(d) graffiti (e) mri-1 (f) mri-2

(g) Landsat-1 (h) Landsat-2 (i) Landsat-3

(j) Radar-1 (k) Radar-2 (l) Radar-3

(m) MRI T1 (n) MRI T2 (o) CT

**Figure 5.2** Images used for anytime registration experiments

### 5.3.1 Generating Performance Profiles

A performance profile was generated offline for each combination of image class and difference measure that was to be tested. These profiles were constructed using training image pairs of the same imaging modalities as the ones to be aligned. As these performance profiles should generalize to images of different sizes and intensity ranges, the image intensities are normalized to a range of 0 to 1 before calculation and the optimal scale factor developed in Section 4.5 was used to scale the gradients. This scaling has the effect of making unit changes in the parameters cause equal RMS pixel shifts in the image, thus normalizing for the sizes of the images.

In order to produce a reliable performance profile, it is important to sample the possible values of the gradient fairly evenly over its range of possible magnitudes. However, the magnitudes of the gradient do not tend to have an even distribution. In general, the large values of $\nabla D$ are found far from the correct registration parameters, while smaller values of $\nabla D$ are clustered around the true parameters. Therefore, in order to get a reasonably even sampling, two sets of samples of $\nabla D$ were computed. The first set was computed at 2000 points randomly distributed in the rigid transform space over the fairly wide range of (angle $= \pm 60°; \Delta x, \Delta y = \pm 60$ pixels). A second set of samples was also taken at 2000 random points in a much smaller range: (angle $= \pm 1°; \Delta x, \Delta y = \pm 1$ pixels). In each case, the value of $\nabla D$ was computed at 12 different levels of computation: $p \in \{$ 0.01%, 0.02%, 0.05%, 0.1%, 0.2%, 0.5%, 0.7%, 1%, 2%, 5%, 7%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 100%$\}$

With these samples, the profiles were constructed using the method described in Section 5.2.2. The resulting performance profiles are shown graphically in Figure 5.3. Note that each profile has a roughly similar shape. At large values of the feedback parameter, $||\nabla_{\phi} D||$, little computation is needed to get a good result. For smaller values, however, progressively more computation is needed. This agrees with intuition; since the image noise level remains constant, small values are progressively harder to measure. It is the curved shape of these graphs that allows us to realize important performance gains. Simply selecting a constant fraction of pixels to use would inevitably be too many for some feedback values and too few for others.

Despite their basic similarity, however, there are important differences between the profiles. For example, note that for both difference measures, the medical images

require a much greater percentage of computation for an accurate result for a given gradient magnitude. The alignment tests, discussed below, reveal that these images require more computation to align successfully. Also note that the $D_{MI}$ performance profiles for both geographic data and medical images indicate that no less than 30% of the pixels will ever be used for these classes of images using this method.

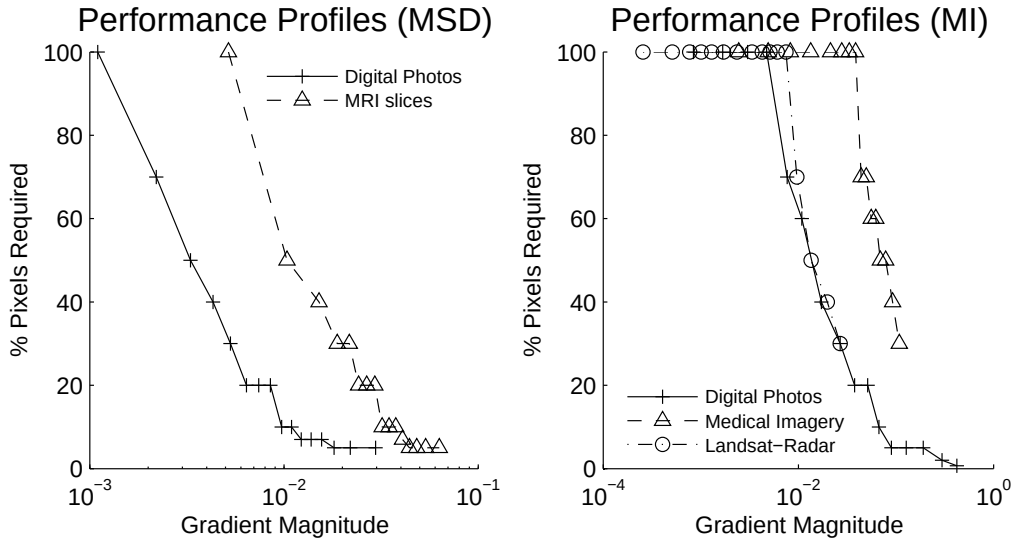



**Figure 5.3** Performance profiles: The percentage of computation required to achieve $E_p = 90\%$. The graph on the left is for the mean squared difference measure (MSD) and the one on the right is for the mutual information measure (MI).

### 5.3.2 Implementation

As described in Section 3.1 the Insight Toolkit library has been used to implement an image registration software framework. This provides a strong baseline implementation for comparison purposes. In the usual implementation, the cost function is calculated using a loop over all the pixels in the fixed image. Extending this to support anytime computation is reasonably simple. Instead of discarding the intermediate variables described in Sections 5.2.1 and 5.2.1 they are cached. If the cost function is evaluated again, to a greater level of accuracy, the accumulation over pixels is continued where it left off.

The optimizer used was the simple but effective gradient descent optimizer used in many of the examples in [109], and previously discussed in Section 3.1.10. This algorithm is shown in Algorithm 1. It works by taking steps of a constant length in the direction opposite the difference measure gradient. At each step, the gradient is compared to the last gradient evaluated. If the gradient has changed direction by more than 90°, then the step size is halved. The algorithm stops when either the step size, or the gradient magnitude drops too low.

As discussed in Chapter 4, the gradient descent optimization can work very well provided the objective function is well-scaled. In this work, in all cases the objective function was scaled by the optimal scaling given by Equation 4.37.

To adapt the optimizer for use with an anytime measure, an additional loop is inserted in place of steps 3 and 4 in Algorithm 1 (see Algorithm 4 steps 3-10). First, the gradient is computed using some small number of pixels, to get an estimate of its magnitude. Then, the accuracy of the current gradient, and the amount of computation required to get the desired accuracy, are estimated using the performance profile. If the accuracy is not yet sufficient, the calculation is continued. Once the required estimated accuracy is achieved, the algorithm continues as the original.

### 5.3.3 Registration Tests

The experimental procedure follows the basic framework described in Section 3.2: Three sets of 20 random starting positions were created. Each set was at a different effective distance from the identity transform in order to test the algorithms over the capture range of the optimizer. For each combination of image, difference measure and algorithm, the true transform was composed with these starting positions, and the result was used to initialize the alignment process. Alignments were performed using the standard approach (labeled GD100 in the graphs), the standard approach using only a specified percentage of pixels (labeled GDx, if $x$% of the pixels is used) and the anytime approach (AGD). The computational efficiency was measured in terms of running time. Each reported time was obtained on a 1.9GHz AMD Athlon machine with 3GB of RAM.

As discussed in detail in Section 3.2, in addition to measuring any speed increases the quality and reliability of the results must also be determined. The quality of a

**Algorithm 4** Anytime Steepest Descent Optimizer

**Set** start position $\boldsymbol{\phi}_0$; iteration counter $n = 0$; scaling matrix $\boldsymbol{S}$; starting step size $s$; accuracy level $a$; starting percentage of pixels $p_{start}$
**Repeat**
  **Set** anytime probe counter $m = 0$,starting computation level $p_{(m)} = p_{start}$
  **Repeat**
    **Compute** the gradient at the current position and computation level, $\nabla D_{(n,m)}(\boldsymbol{\phi}_{(n)}, p_{(m)})$
    **Compute** the scaled gradient, $\nabla D^* = \boldsymbol{S} \times \nabla D_{(n,m)}(\boldsymbol{\phi}_{(n)})$
    **Compute** the estimated accuracy of this gradient $\hat{a}_{(m)} = P_{fwd}(||\nabla D^*||), p_{(m)})$
    **Compute** the estimated amount of computation required $p_{(m)} = P_{ref}(||\nabla D^*||), a)$
    **Set** $m = m + 1$;
  **Until** $\hat{a}_{(m)} > a$
  **If** the scaled gradient has changed direction by more than 90° **then**
    **Set** $s = \frac{s}{2}$  // Reduce the step size
  **End if**
  **Compute** the update to the parameters, $\Delta\boldsymbol{\phi}_{(n)} = \frac{\nabla D^*}{||\nabla D^*||} \cdot s$
  **Set** $\boldsymbol{\phi}_{i+1)} = \boldsymbol{\phi}_{(n)} + \Delta\boldsymbol{\phi}_{(n)}$ and $n = n + 1$
**Until** the convergence criteria are reached

result was measured by the mean target registration error (mTRE), and the reliability of each method was measured by the number of failed alignments. All of these experimental observations were tested for statistical significance using a paired t-test for the run times and mTRE, and a McNemar test for the failure rates. (See Section 3.2.1 for more detail on these tests.)

| Image Pair | mTRE (pixels) | | | | Run time (s) | | | | Failure Rate (%) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | GD100 | GD50 | GD30 | AGD | GD100 | GD50 | GD30 | AGD | GD100 | GD50 | GD30 | AGD |
| a)-a) (DP) | 0.02 | 0.02 | 0.01 | 0.01 | 48.0 | **24.3** | **15.2** | **31.8** | 1.7% | 3.3% | 1.7% | 1.7% |
| d)-d) (DP) | 0.03 | 0.02 | 0.02 | 0.03 | 37.7 | **20.6** | **13.1** | **28.2** | 0.0% | 0.0% | 1.7% | 1.7% |
| a)-b) (DP) | 0.39 | 0.38 | 0.39 | 0.39 | 106.0 | *40.7* | **22.0** | *35.8* | 8.3% | 5.0% | 8.3% | 5.0% |
| a)-c) (DP) | 0.26 | 0.26 | 0.26 | 0.25 | 59.1 | **36.8** | **21.1** | **27.7** | 5.0% | 3.3% | 8.3% | 6.7% |
| e)-e) (M1) | 0.05 | 0.03 | 0.03 | 0.05 | 4.6 | **2.6** | **1.7** | 4.5 | 0.0% | 0.0% | 0.0% | 0.0% |
| f)-f) (M1) | 0.08 | *0.04* | 0.05 | 0.09 | 2.9 | **1.5** | **1.0** | *2.6* | 3.3% | 1.7% | 1.7% | 1.7% |
| MSD-Avg | 0.13 | *0.12* | *0.12* | 0.13 | 42.0 | **20.5** | **12.0** | **21.5** | 3.1% | 2.2% | 3.6% | 2.8% |

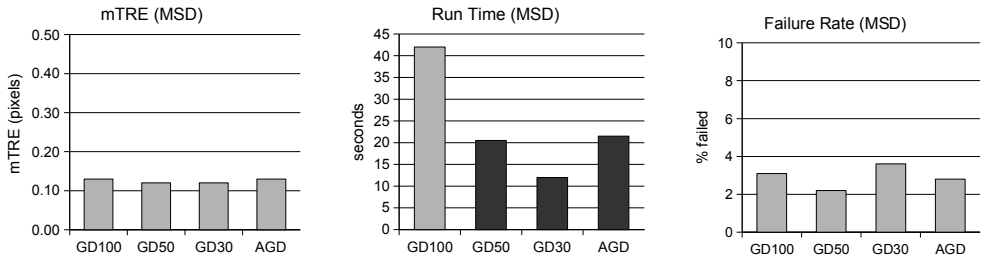
(a) Results for the MSD measure

| Image Pair | mTRE (pixels) | | | | Run time (s) | | | | Failure Rate (%) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | GD100 | GD50 | GD30 | AGD | GD100 | GD50 | GD30 | AGD | GD100 | GD50 | GD30 | AGD |
| m)-o) (M2) | 0.54 | 0.76 | 0.83 | 0.68 | 8.7 | **4.4** | **3.3** | 8.3 | 13.3% | *26.7%* | **46.7%** | 16.7% |
| n)-o) (M2) | 0.90 | *1.35* | *1.47* | 1.08 | 9.5 | **4.8** | **2.4** | 8.8 | 23.3% | *38.3%* | **46.7%** | 20.0% |
| m)-n) (M2) | 0.10 | 0.12 | **0.16** | 0.11 | 6.0 | **3.5** | **2.2** | **5.3** | 3.3% | **31.7%** | **36.7%** | 5.0% |
| g)-j) (EO) | 1.68 | *1.99* | 1.98 | 1.90 | 13.6 | **7.1** | **4.0** | **7.7** | 28.3% | **45.0%** | **55.0%** | 35.0% |
| h)-k) (EO) | 1.31 | 1.30 | 1.30 | 1.32 | 42.9 | **25.5** | **16.2** | 38.9 | 13.3% | *30.0%* | **51.7%** | 18.3% |
| i)-l) (EO) | 0.07 | 0.07 | **0.08** | 0.07 | 72.4 | **37.4** | **24.1** | **50.3** | 6.7% | *21.7%* | **41.7%** | 6.7% |
| a)-a) (DP) | 0.01 | 0.02 | 0.02 | 0.01 | 115.3 | **62.9** | **37.5** | **14.7** | 5.0% | 5.0% | 8.3% | 5.0% |
| d)-d) (DP) | 0.07 | 0.02 | 0.02 | 0.04 | 110.6 | **52.4** | **32.7** | **19.8** | 0.0% | 5.0% | 1.7% | 0.0% |
| MI-avg | 0.41 | *0.48* | **0.50** | 0.45 | 64.5 | **33.0** | **20.3** | **18.5** | 11.7% | **25.4%** | **36.0%** | 13.3% |

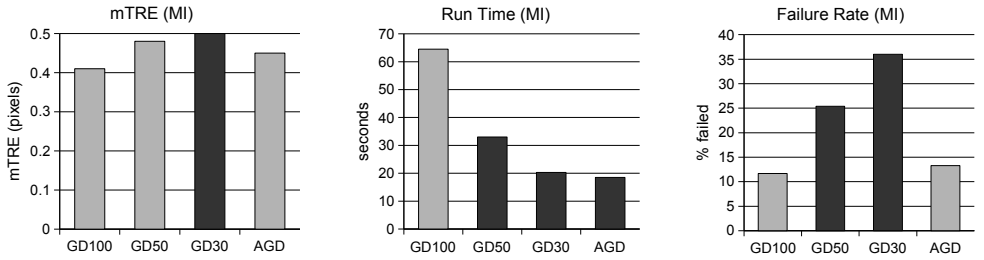
(b) Results for the MI measure

**Table 5.1** Experimental Results: Results by image pair, and combined for each measure (bottom rows). Algorithms: GDX – standard algorithm, using X% of pixels; AGD – Anytime algorithm. Entries shown in plain font are not significantly different from GD100; those in *italic* are ambiguous (see text) and items in **bold** significantly differ from GD100.

The experimental results are reported in Table 5.1a (for MSD) and 5.1a (for MI). The tables show the average runtime of each method for each image pair, as well as for all runs combined. They also show the failure rate and the mTRE. The combined results for each difference measure are also shown graphically in Figure 5.4. For the MSD measure, all the algorithms under test show some improvement in speed, without significantly affecting failure rate or mTRE. For this measure, there is little to distinguish the anytime method from simply reducing the number of pixels. However, the results for the MI measure highlight the advantages of the anytime method. The



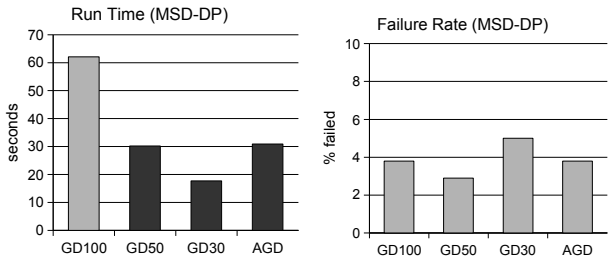

(a) Graphical summary of results for the MSD Measure



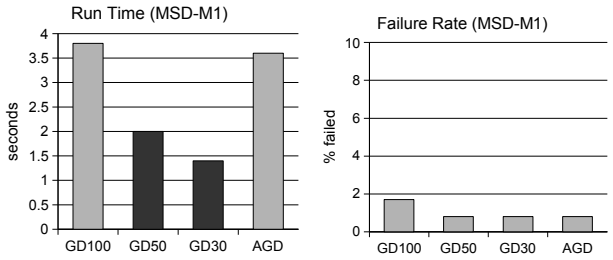

(b) Graphical summary of results for the MI Measure

**Figure 5.4** Graphical summary of complete results. The graphs show mTRE, running time and failure rate for all runs combined. Results that are statistically significantly different from the unmodified algorithm are shown with darker shading.

quality of the results obtained by simply reducing the number of pixels by a percentage is very variable, and frequently involves a statistically significant loss of quality or reliability. The anytime method delivers significantly faster times without sacrificing either positional accuracy, or the failure rate. The overall results (bottom row, Table 5.1b and Figure 5.4b) show that the anytime method significantly outperformed simply reducing the number of pixels.



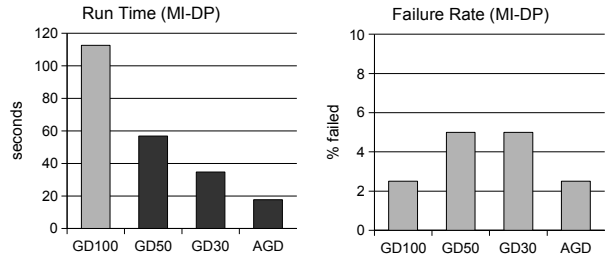

(a) Results for the MSD measure on digital photo images (DP)



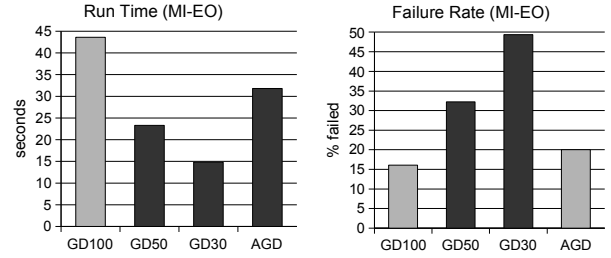

(b) Results for the MSD measure on medical images (M1)

**Figure 5.5** Graphical summary of results for MSD measure by image type. The graphs show running time and failure rate combined by image class (Digital Photos – DP; Medical images – M1). Results that are statistically significantly different from the unmodified algorithm are shown with darker shading. The mTRE is omitted as the differences between the algorithms are minor.

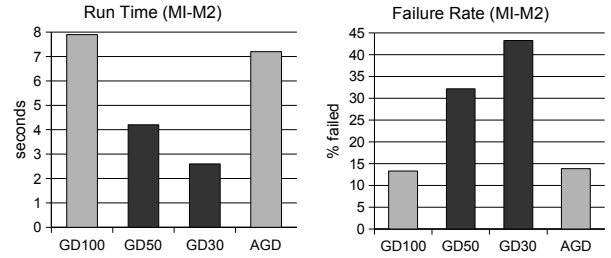
The results can be further analyzed by grouping the images by their type. Results grouped by image type are shown in Figures 5.5 and 5.6. The results for the MI measure (Figure 5.6), in particular, show that an important advantage of the anytime approach is its adaptability. Note that simply choosing an arbitrary percentage of pixels to process gives very different results depending on the image class. In contrast, the anytime approach was able to align each of these image types without




(a) Results for the MI measure on digital photo images (DP)




(b) Results for the MI measure on earth observation images (EO)




(c) Results for the MI measure on medical images (M2)

**Figure 5.6** Graphical summary of results for MI measure by image type. The graphs show running time and failure rate combined by image class (Digital Photos – DP; Medical images – M2; Earth Observation images – EO). Results that are statistically significantly different from the unmodified algorithm are shown with darker shading. The mTRE is omitted as the differences between the algorithms are minor.

significantly increasing the failure rate. The digital photos required only a little more than 10% of the original running time (Figure 5.6a), and the failure rate did not significantly increase for any of the approaches. However, the earth observation images (Figure 5.6b) seem to be somewhat more difficult to align. When the number of pixels is reduced by an arbitrary percentage, the failure rate increases significantly. The anytime algorithm achieved a significant speedup, but it is not as striking as the digital photo case. Finally, in the case of the multimodal medical images (Figure 5.6c), the anytime approach was not able to achieve a statistically significant speedup. It seems that in this case more pixels are inherently required to successfully align the images. Reducing the number of pixels used to an arbitrary level causes a severe jump in the failure rate, more than doubling it when only 30% of the pixels are used. The anytime method has adapted to that requirement and maintains a low failure rate by increasing the amount of computation performed.

## 5.4 Conclusions and Future Work

This chapter proposed the use of deliberation control methods in order to improve the efficiency of computer vision applications. Such methods were implemented for the image registration problem and showed a significant improvement in speed without degrading the quality of the results. In certain cases, the deliberation control approach significantly outperforms a simple reduction in the number of pixels because it can selectively use more pixels when needed.

Even when the performance gains are limited, a major advantage of this approach is that the number of pixels used is determined using a training process. The results show that arbitrarily selecting a percentage of the image data to use for alignment will lead to very different results on different classes of images. This method gives a principled way to determining how much of the image data needs to be processed to achieve reasonable results.

The key elements of this approach are (1) that the amount of computation to do should vary during the optimization process, and not be a global parameter of the entire registration, and (2) that a formal and reliable method is needed to learn the performance of the algorithm at varying levels of computation on the problem of

interest. While the dynamic performance profile proved to be a successful method for doing this in this work, it is not the only way to accomplish this. For example, there is interesting work presented in [1] which models the performance of a VLSI circuit layout algorithm with a probabilistic model of the performance profile. This work also applies the anytime algorithm approach to control of the parameters of stochastic optimization algorithms. While this clearly goes well beyond the scope of this thesis, it would be interesting to apply these or similar more sophisticated deliberation control methods to this problem.

# Chapter 6

# Pixel Selection Revisited

As discussed in Chapter 2, direct parameterized image registration is often framed as the optimization of an image difference measure, $D$, which is computed between the fixed image, and the warped, moving image. The computation of the difference measure is clearly the most expensive part of this process. To save computation time, $D$ can be computed using only some of the pixels in the image. In Chapter 5, the question of how many pixels to use was addressed. However, if only some of the pixels are to be used, it is natural to ask if some pixels are more helpful than others. A widely used guideline is that pixels of high derivative are most useful for the image registration process. However, as it is noted in [66] this may cause some performance degradation.

This chapter formally addresses this performance degradation and investigates the claim that selecting a maximally informative set of pixels is better than simply selecting them randomly. Scale-space theory tells us that the notion of the derivative of a sampled signal can only be understood in the context of some scale. It is shown that the approach to pixel selection exemplified by [66] requires that the image derivatives in question be computed at a scale appropriate for the amount of inaccuracy in the transformation parameters. If this is not taken into account, then under some conditions, the use of these pixel selection methods can lead to a deterioration in performance when compared to randomly selecting an equal number of pixels.

The following section reviews previous work on pixel selection for image registration. The approach of Dellaert and Collins [66] is particularly relevant to this

discussion, and is reviewed in detail in Section 6.2. Section 6.3 analyzes the idea of pixel selection in depth, and proposes improved alternatives to their pixel selection criterion. As part of the analysis, the effect of derivative scale on the pixel selection criteria is identified. Finally, Section 6.4 presents experiments that demonstrate that when pixel selection is used the capture radius of the image difference measure, and therefore the performance of the optimizer, is dependent on the scale of the derivative used to compute the selection criterion.

### Publications

The majority of this chapter is based on work published in:

[33] Rupert Brooks and Tal Arbel. The importance of scale when selecting pixels for image registration. In *Proceedings of the 4th Canadian Conference on Computer and Robot Vision (CRV2007)*, pages 235–242, Montreal, Canada, May 2007.

## 6.1 Previous Work

This question of which pixels to use for image matching was first addressed by Nagel and Rosenfeld [148] in the context of template matching. They showed that for the mean absolute difference (MAD) image difference measure, $D_{MAD}$, defined as

$$D_{MAD} = \frac{1}{N}\sum_{i=1}^{N} |I_f(\boldsymbol{X}_i) - I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})| \tag{6.1}$$

the pixels furthest from the expected value of an average pixel could contribute the most to the sum and were therefore the ones to examine first. (Here $N$ is the number of pixels, $I_f(\boldsymbol{x})$ and $I_m(\boldsymbol{x})$ are the fixed and moving images respectively, $\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi})$, is the warp, controlled by the parameters, $\boldsymbol{\phi}$, and the $\boldsymbol{X}_i$ are the pixel positions in the fixed image.) Other means of bounding the possible resulting values of the image difference measure with partial data have been proposed (*e.g.,* [69]). These methods can be incorporated into branch and bound search approaches.

Most iterative image registration methods are sensitive to the image gradient. This concept has its roots in the computation of optical flow, where the regions of smooth image content were found to have an undetermined flow field [17, 105]. This undetermined area had to be filled in with regularization techniques. This has given rise to a "common wisdom" in the field that better registration performance can be achieved by using the pixels of high gradient (*e.g.,* [124, 181]). This type of approach was formally analyzed by Dellaert and Collins [66] in the context of fast template-based tracking. They developed a criterion for pixel selection based on both the transform Jacobian, and the image derivative. However, they noted that there may be some degradation in performance. Buenaposada and Baumela [44] refined the selection criteria somewhat, but they did not address the issue of performance degradation at all. Certain problem specific approaches also exist. For example, Huang *et al.* [108] speed up the registration of cardiac ultrasound and MR images by using only those pixels in the ultrasound which contain information relevant to the registration. It is difficult to see how to generalize that technique to other modalities.

Shortly after the material in this chapter was published, Benhimane [23] put forward an interesting proposal to deal with the same issues in the context of template based tracking. The motivation for that work was similar to this. Optimization algorithms make an implicit assumption that the warped image can be approximated by a linear or quadratic Taylor series approximation around the starting transformation. Benhimane's approach [23] is to simulate many possible transformations of the image, and determine to what degree various pixels agree with that assumption and so can be used to infer the warp parameters. He selects pixels that agree with that assumption, and rejects pixels that do not. However, he does not deal explicitly with the role of derivative scale in determining how well the pixels agree with the Taylor series approximation.

## 6.2 Pixel Selection for Faster Registration

In [66], the pixel selection problem is analyzed in terms of which single pixel, $\boldsymbol{I}_i$, will provide the most new information about the transformation. They consider a least

squares image difference measure,

$$D_{MSD}(\boldsymbol{\phi}) = \frac{1}{2}\sum_i (I_f(\boldsymbol{X}_i) - I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi})))^2, \tag{6.2}$$

and ask which pixel will most reduce the amount of uncertainty in the parameters of interest. Using a well known argument, given a good enough initial guess of the parameters, $\boldsymbol{\phi}_0$, the minimum of Equation 6.2 can be found by iterating

$$\boldsymbol{\phi}_{(n+1)} = \boldsymbol{\phi}_{(n)} - \left[\boldsymbol{J}^T\boldsymbol{J}\right]^{-1}\boldsymbol{J}^T(I_f(\boldsymbol{X}) - I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))) \tag{6.3}$$

where $\boldsymbol{J}$ is the derivative of $I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))$ with respect to $\boldsymbol{\phi}$:

$$\boldsymbol{J} = \frac{\partial \boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi})}{\partial \boldsymbol{\phi}} \cdot \frac{\partial I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))}{\partial \boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi})} \tag{6.4}$$

Assuming a Gaussian prior distribution with mean 0 and covariance $\boldsymbol{C}_{init}$ on $\boldsymbol{\phi}$, then the covariance matrix of the final parameters $\boldsymbol{C}_{final}$ may be estimated using the law of propagation of variance,

$$\boldsymbol{C}_{final} = (\sigma^{-2}\boldsymbol{J}^T\boldsymbol{J} + \boldsymbol{C}_{init}^{-1})^{-1}, \tag{6.5}$$

where $\sigma$ is the expected noise on each pixel. The contribution of a single pixel, $i$, may be calculated by keeping only its row in $\boldsymbol{J}$, thus

$$\boldsymbol{C}_{final_i} = (\sigma^{-2}\boldsymbol{J}_i^T\boldsymbol{J}_i + \boldsymbol{C}_{init}^{-1})^{-1} \tag{6.6}$$

where $\boldsymbol{J}_i$ is the row of $\boldsymbol{J}$ corresponding to pixel $i$. In [66] it is proposed that the best pixel is the one that minimizes the expected trace of $\boldsymbol{C}_{final}$. In the rest of the chapter, this criterion is referred to as $Tr(\boldsymbol{C})$.

To select a set of pixels by truly minimizing their expected final covariance matrix is intractable, since the number of possible combinations is exponential in the number of pixels. However, in [66] it is demonstrated that simply selecting a random set of pixels from the best x% of individual pixels works very well, provided that care is taken to avoid clustering.

This method for pixel selection is reported to work, and seems to support the notion that pixels of high derivative are relevant to image registration. However, this chapter argues that further elaboration of this idea is needed. The criterion used for pixel selection has not been well explored. New criteria are proposed that are both more theoretically justified and more computationally efficient. Furthermore, the optimization process is sensitive to the scale of the derivative used, an issue that has been ignored up to this point.

## 6.3 Refinements to Pixel Selection Methods

### 6.3.1 Pixel Selection Criteria

The criterion used by Dellaert and Collins [66] for selecting pixels is to minimize the trace of the estimated covariance matrix. The trace of a matrix is the sum of the squares of its eigenvalues, which for a covariance matrix, is equivalent to the sum of the squares of the principal radii of its confidence hyperellipsoid. Thus, this measure will tend to penalize resulting covariance matrices that have one or more long axes.

Recent work on observation selection such as [64] argues convincingly that the mutual information between an observation and the parameters of interest is the optimal observation selection criterion. While mutual information between arbitrary distributions may be difficult to compute, note that if a Gaussian distribution is assumed for both the observations and the parameters of interest, then minimizing the determinant of the expected resulting covariance matrix corresponds to maximizing the mutual information [64]. Interestingly this has been proposed as a criterion for observation selection much earlier, for example in [202], where it was used for selecting the next camera position in an active vision system. This criterion may also be more intuitively satisfying, as it corresponds to the volume of the confidence ellipsoid. Since any given pixel cannot fully determine the final solution the resulting matrices will be nearly singular. Thus, most of the confidence ellipsoids generated will have at least one very long axis. As several observations are being combined, it also makes more intuitive sense to minimize the volume of each hyperellipsoid, rather than its length. Thus there are compelling reasons to use the determinant of the expected covariance matrix, rather than its trace. In the following, this criterion will be referred to as

$Det(\boldsymbol{C})$.

Computational issues are also important for any selection technique. After all, the driving reason for selecting pixels in these applications is to improve performance. At first glance $Tr(\boldsymbol{C})$ may appear to require less computation, but this is not the case. Computing the trace of the covariance matrix requires the following operations (where $n$ is the number of parameters):

1. A vector outer product $O(n^2)$
2. A matrix summation $O(n^2)$
3. A matrix inversion[1] $O(n^2 \log(n))$
4. A matrix trace $O(n)$

Because the determinant of a matrix is the reciprocal of the determinant of its inverse, computing the determinant of the covariance matrix requires marginally fewer operations.

1. A vector outer product $O(n^2)$
2. A matrix summation $O(n^2)$
3. A matrix determinant[1] $O(n^2 \log(n))$
4. A division $O(1)$

Thus there is no computational advantage to using $Tr(\boldsymbol{C})$. Furthermore, matrix inversions are often numerically unstable, which may make $Tr(\boldsymbol{C})$ more difficult to implement.

There are, however, reasonable approximations to both these criteria that require substantially less computation. Consider that the matrix created by the vector outer product generally has a strong diagonal component. Both the trace and the determinant of the inverse of a diagonal matrix, $\boldsymbol{M}$, can be easily calculated. Specifically,

[1]While classic matrix inversion methods have a complexity of $O(n^3)$, recent developments in matrix algorithms have pushed the theoretical complexity of determinants and inverses near $O(n^2 \log(n))$ [193].

the determinant is the product of reciprocals of the diagonal elements,

$$Det(\boldsymbol{M}^{-1}) = \prod_{i=1}^{n} \boldsymbol{M}_{i,i}^{-1} \tag{6.7}$$

and the trace is the reciprocal of the harmonic sum of the diagonal elements.

$$Tr(\boldsymbol{M}^{-1}) = \sum_{i=1}^{n} \boldsymbol{M}_{i,i}^{-1} \tag{6.8}$$

When the matrix $\boldsymbol{J}^T\boldsymbol{J}$ is generally strongly diagonal, these are reasonable approximations of the preceding criteria. This is true for matrix based transforms, see Figure 4.17. Note that they can be computed directly from $\boldsymbol{J}$, avoiding the outer product entirely, as follows

$$Det(\boldsymbol{C}_i) \approx \prod_{j=1}^{n} \boldsymbol{J}_{i_j}^{-2} \tag{6.9}$$

$$Tr(\boldsymbol{C}_i) \approx \sum_{j=1}^{n} \boldsymbol{J}_{i_j}^{-2} \tag{6.10}$$

where $\boldsymbol{C}_i$ is the covariance matrix corresponding to the $i$-th pixel, and $\boldsymbol{J}_{i_j}$ is the $j$-th element of the $i$-th row of $\boldsymbol{J}$. Each of these requires only $O(n)$ operations to compute. In the following the criteria of Equation 6.9 will be identified as as $\prod \boldsymbol{J}_i^{-2}$ and of Equation 6.10 as $\sum \boldsymbol{J}_i^{-2}$.

Thus it is proposed that the approximation to the determinant, $\prod \boldsymbol{J}_i^{-2}$, is the best criteria to use. It is a reasonable approximation to the mutual information between the observed pixel and the transformation parameters, and can be computed in $O(n)$ time.

### 6.3.2 The Importance of Scale

Dellaert and Collins [66] observed that one problem with their method is that it can reduce the capture radius of the optimization, but they did not analyze how serious this problem was, or how it could be controlled. The capture radius refers to the distance away from the true transformation that the optimization process may be

started and can still be expected to converge to the correct solution. In conceptual terms, the function to be optimized forms a "pit" around the true minimum. Within this pit, an optimization algorithm can "slide down" to the solution, but outside this region it may fail to converge to the solution.

This problem arises because of the nature of the pixel selection method. Observe that any of these pixel selection criteria is determined completely by the row of $\boldsymbol{J}$ that corresponds to the pixel in question. In turn, $\boldsymbol{J}$ is computed from the image derivatives. However, the derivative of a sampled signal is not a clearly defined quantity. It is dependent on the aperture through which the signal is viewed, or in other words, on the scale at which the signal is viewed [184].

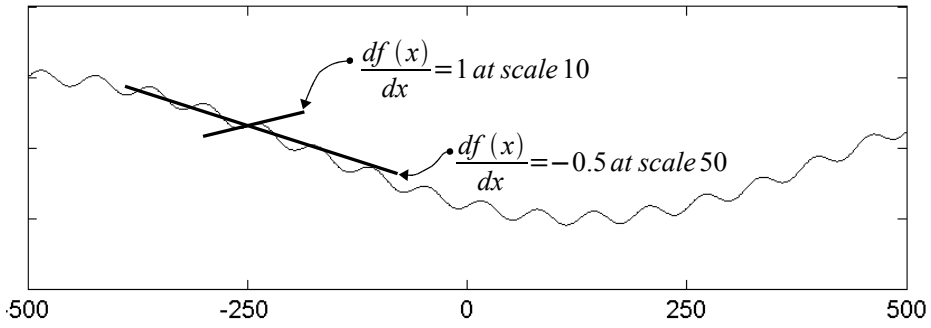



**Figure 6.1** A derivative varies with scale.

Figure 6.1 illustrates this property, and also shows that the scale of a derivative can be viewed as determining the range over which the linear approximation to the signal will be valid. All the components of this framework rely on this linear approximation being valid to work properly. The law of propagation of variance, used to develop Equation 6.6 holds in the linear case and relies on the linearization of the cost function to be valid. All optimization algorithms used in this thesis rely in one way or another on the second order Taylor series being a close approximation to the cost function to work properly. If the minimum being searched for is outside the valid range of all the derivatives being used to compute the search direction, it is unlikely that an optimization method that relies on these derivatives will find it.

It is hypothesized that using one of these pixel selection methods may improve performance, provided that the starting estimate of the transformation is not "too far"

from the correct one. The definition of "too far" in this case, will be directly related to the scale of the derivative used. To quantify this idea requires a measure of the distance between transformations. The one proposed by Van de Kraats *et al.* [190] is used, as discussed in Section 3.2. This measures the distance between two transformations by computing the mean difference in position of a set of corresponding points warped by each transformation. Based on this, it is proposed that the scale of the derivative should be equal or greater to the expected distance of the starting position from the true solution or the optimization may not converge correctly.

## 6.4 Experiments

The experimental verification of this proposal has three parts. First the effect of each of the pixel selection methods in Section 6.3.1 on two common image difference measures is examined, and their running times are compared. Secondly, the role of the scale of the derivative when pixel selection using the method of [66] is applied with the derivative computed at different scales is investigated. Finally, these hypotheses are tested to see if they have the expected effects on actual image registrations by registering a number of images and observing the performance of each approach.

### 6.4.1 Implementation

As discussed in Section 3.1, an image registration system has been implemented using the Insight Toolkit [109]. To this system, the ability to select a set of pixels and use only that selected set in the image difference measure computation has been added. In the following experiments, two image difference measures have been used, the mean squared difference (MSD), described in Section 3.1.2, and the mutual information, described in Section 3.1.4. (The Mean Squared Difference measure was used for the original analysis in [66]).

To eliminate the possibility that the results could be skewed by quirks of the optimization algorithm, two different optimization algorithms were used. The first is the simple gradient descent optimizer from the ITK library [109] described in Algorithm 1 which corresponds to the type of optimizer expected when the original analysis in [66] was done. The second optimizer was Powell's method [156] (see Section 3.1.10) which

does not require the derivative of the cost function. Using a derivative free approach provides an interesting confirmation that these results will generalize to a wide variety of optimization algorithms.

### 6.4.2 Pixel Selection Criteria

Scores are assigned to pixels as described in Section 6.3.1. Once each pixel has been assigned a score, however, the actual selection must introduce some randomness, to avoid clustering of the pixels [66]. All the pixels are ranked by their score and grouped into 5% blocks. Pixels are then chosen randomly from the best 5%, followed by choosing from the next 5% and so on.

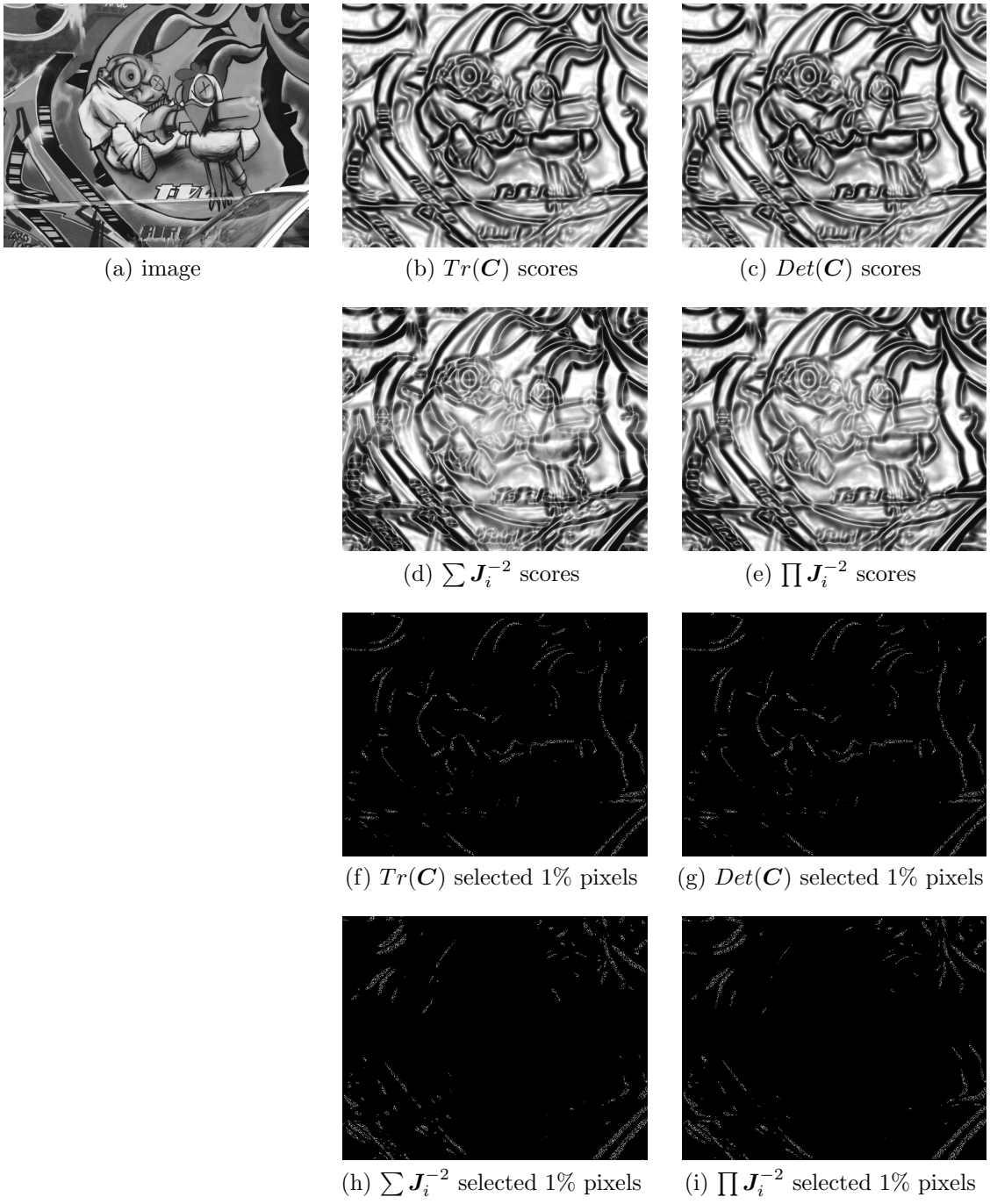
Each of these selection methods requires a prior estimate of the accuracy of the transformation. In the absence of other information, it was desirable to assign equal accuracies to each element. However, a standard deviation of one radian in rotation represents a very different level of accuracy than one pixel on a translational component. With projective transformations, the effects of some parameters are as much as four orders of magnitude larger than others. The optimal scale factor described in Section 4.5 was used to linearly rescale all the parameters so that unit step in the parameter space caused an average movement of one pixel width in the image space. In effect, each transformation $\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi})$ was reparameterized as $\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi} = \boldsymbol{M} \cdot \boldsymbol{\psi})$ where $\boldsymbol{M}$ is a diagonal matrix equal to

$$\boldsymbol{M}_{jj} = \sqrt{\sum_i \boldsymbol{J}_{ij}^2}. \tag{6.11}$$

Then the prior covariance matrix was set to the identity.

Figure 6.2 shows the resulting pixel scores and a selection of 1% of pixels for a typical digital camera image[2], undergoing a projective transformation. While there are some differences between each pixel scoring method, the selected pixel sets are quite similar in each case. The main visual difference is that the diagonal approximations, $\prod \boldsymbol{J}_i^{-2}$ and $\sum \boldsymbol{J}_i^{-2}$ are somewhat faded in the center. This is due to certain components of $J$, particularly the rotation and elation parameters, being relatively

[2]Image from sample images provided by K. Mikolajczyk `http://www.inrialpes.fr/lear/people/Mikolajczyk/`.



(a) image (b) $Tr(\boldsymbol{C})$ scores (c) $Det(\boldsymbol{C})$ scores

(d) $\sum \boldsymbol{J}_i^{-2}$ scores (e) $\prod \boldsymbol{J}_i^{-2}$ scores

(f) $Tr(\boldsymbol{C})$ selected 1% pixels (g) $Det(\boldsymbol{C})$ selected 1% pixels

(h) $\sum \boldsymbol{J}_i^{-2}$ selected 1% pixels (i) $\prod \boldsymbol{J}_i^{-2}$ selected 1% pixels

**Figure 6.2** Pixel scores and selected pixels for each selection criterion. For images b-e, darker is better. For images f-i, selected 1% of pixels are shown in white.

larger towards the edge of the image, an effect that is more pronounced when the off-diagonal elements are not considered.

The main argument for recommending the use of $\prod \boldsymbol{J}_i^{-2}$ as a selection criterion, is that it is faster to compute. The computation times to generate each of these pixel selections are shown in Table 6.1. This criterion is significantly faster to compute. The original criterion of [66], $Tr(\boldsymbol{C})$, is remarkably slow, although this may be partly due to inefficiencies in the matrix library used[3].

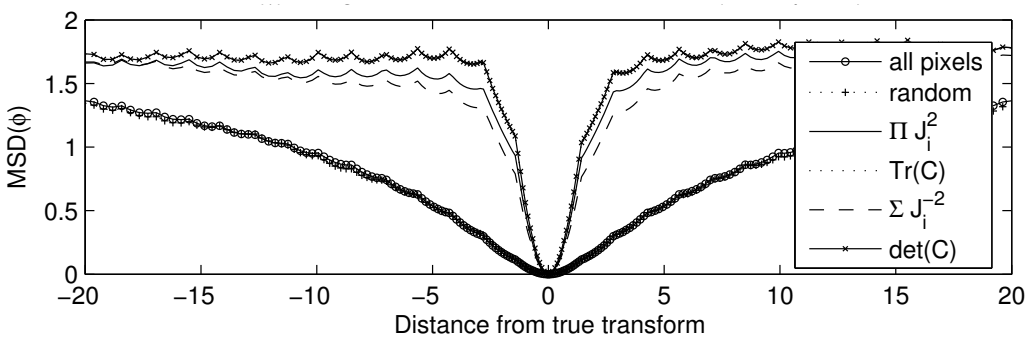
| $Tr(\boldsymbol{C})$ | $Det(\boldsymbol{C})$ | $\sum \boldsymbol{J}_i^{-2}$ | $\prod \boldsymbol{J}_i^{-2}$ |
|---|---|---|---|
| 34.3 s | 7.40 s | 2.25 s | 1.84 s |

**Table 6.1** Computation times to generate the pixel selections using each criterion.

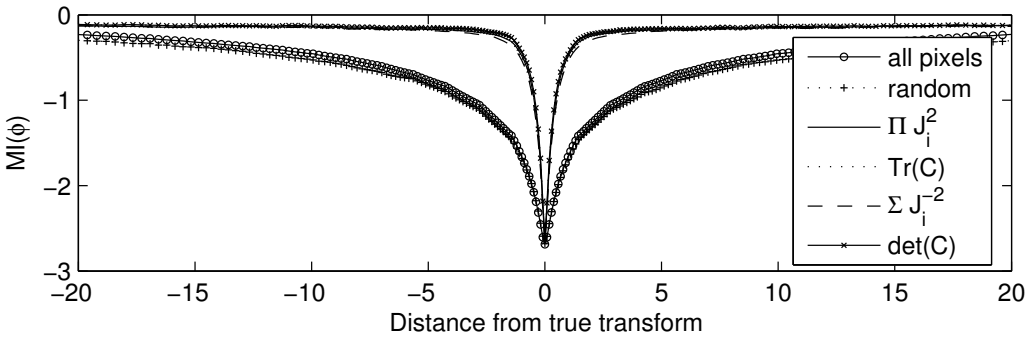
To further investigate the differences between each approach for pixel selection, the value of the image difference measure was computed between an image and itself at numerous points in the transformation parameter space along a vector passing through the parameters representing the identity transformation. The resulting difference measure values for a transformation consisting of translation only are shown in Figure 6.3. Note that the "ripples" in the cost function are due to interpolation artifacts.

For each of these calculations, 1% of the pixels were used and the derivative scale was 5 pixels. The different criteria give extremely similar results, as could be anticipated from the fact that the pixels chosen are extremely similar in Figure 6.2. In all cases, the cost function has a much steeper pit near the optimum when using pixel selection, but the capture radius is much smaller than when pixels are selected randomly. This narrowness of the capture radius is less pronounced if the transformation is more complex than just a translation. For example, Figure 6.4 shows the same cost functions, but here the vector through the transformation space has an angular component. As a rotation of the image moves some pixels more than others, the effect is to smooth the cost function somewhat. However, the valley in the cost function remains much sharper than if the pixels are chosen randomly. Indeed, if the pixels

[3]The matrix was inverted using the VXL library `http://vxl.sourceforge.net`, which defaults to using singular value decomposition for inversion.




(a) Mean Squared Difference values using different pixel selection criteria (1% of pixels)



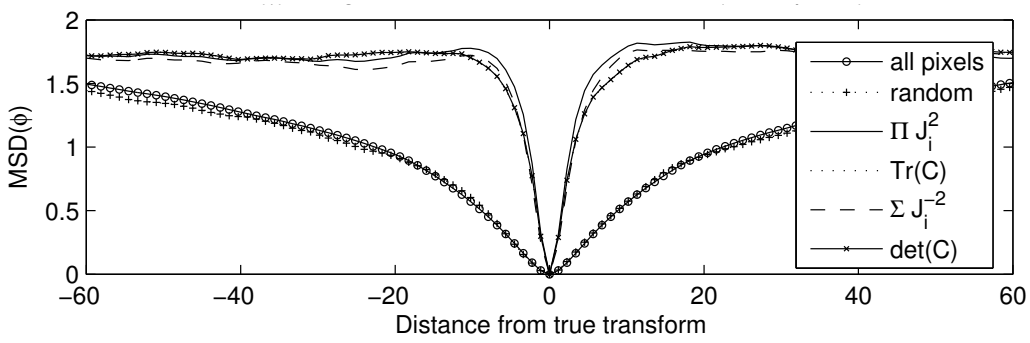

(b) Mutual Information values using different pixel selection criteria (1% of pixels)

**Figure 6.3** Image difference measure values for pixels selected with different criteria. Case 1: Translation

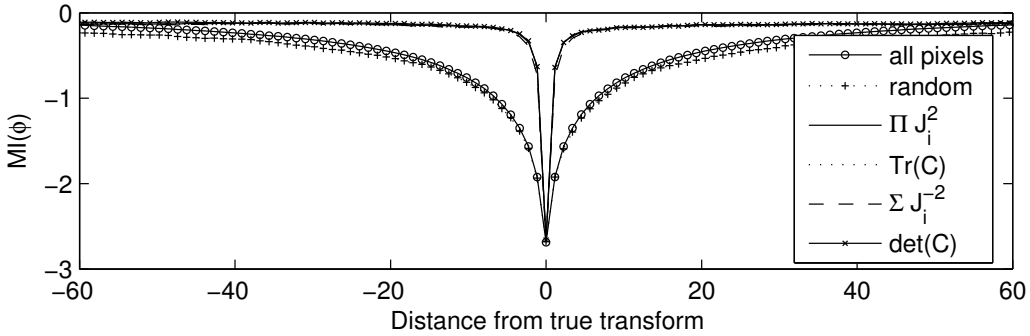
are chosen randomly, the cost function has exactly the same shape as when using all the pixels, and is merely somewhat noisier.

### 6.4.3 Role of Derivative Scale

The second set of experiments addresses the shape of the cost function as the scale of the derivative is varied. Once again, the cost function has been computed for a series of points in the transformation space, which pass through the origin. Figure 6.5 shows this for the original selection method in [66], at derivative scales of 1, 5 and 10 pixels. As hypothesized, when the scale of the derivative increases, the cost function tends to flatten out and approach the shape of the cost function using randomly selected pixels. The capture radius clearly increases with the scale of the derivative, therefore it is essential to consider how large the initial misregistration of the images will be




(a) Mean Squared Difference values using different pixel selection criteria (1% of pixels)




(b) Mutual Information values using different pixel selection criteria (1% of pixels)

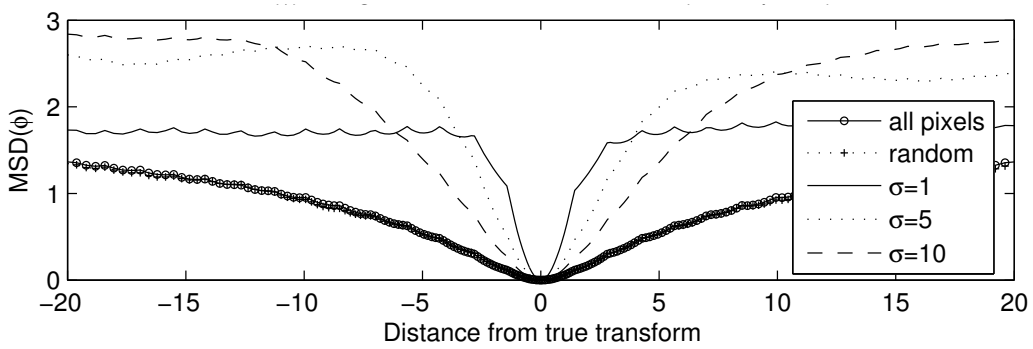
**Figure 6.4** Image difference measure values for pixels selected with different criteria. Case 2: Translation + Rotation

when a pixel selection approach is used.

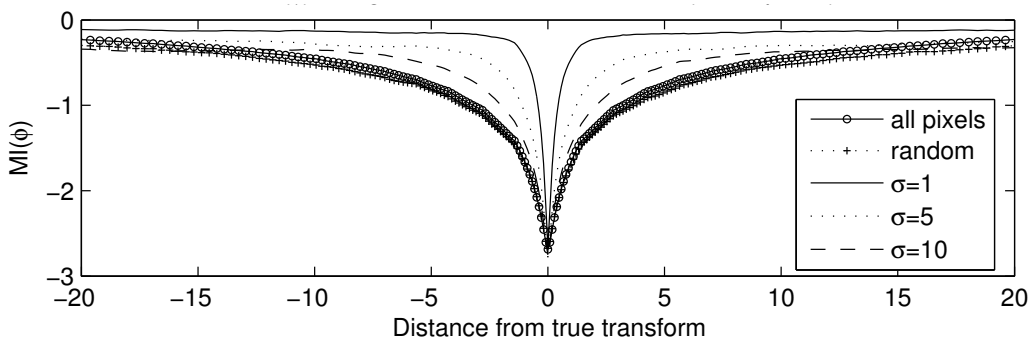
### 6.4.4 Image Alignments

To test these hypotheses under real image registration conditions, a number of typical image registration examples were performed using randomly selected pixels, pixels selected using the original method, and the proposed simplification. One hundred randomly generated transformations were used as starting positions. Each starting position was composed with the true transform, and the result used as a starting position for the registration. A run was considered to have failed if either the optimization process aborted in an error condition, or the final transformation parameters were a distance of more than five pixels from the true parameters.

When registration was successful, accuracy of the different methods is comparable.




(a) Mean Squared Difference values using different derivative scales (1% of pixels)




(b) Mutual Information values using different derivative scales (1% of pixels)

**Figure 6.5** Image difference measure values for pixels selected using different derivative scales.

The main variation between the different approaches is their rate of failure. The following figure (6.7) shows the cumulative number of failures when the transformations are considered in order, starting closest to the distance from the identity transformation. As all the approaches were tested from the same starting positions, the results are directly comparable.

For the MSD measure, the graffiti image was registered using a projective transformation to another image of the same scene, taken from a different camera position (shown in Figure 6.6). Figure 6.7a shows the results of the experiment. The graphs are the cumulative number of failures moving through the list of transformations from the smallest (in terms of distance from the identity) to the largest. As hypothesized, when using pixel selection, the algorithm is more prone to failure the further the starting position is from the true optimum. Furthermore, the range over which the optimizer

(a) Graffiti I (b) Graffiti II

**Figure 6.6** The graffiti image pair.

can converge to the solution tends to increase with the scale of the derivative.

Figure 6.7b shows similar results, but using the approximate criterion, $\prod \boldsymbol{J}^{-2}$. As expected from the results of the previous experiment, these results are quite similar to those for the $Tr(\boldsymbol{C})$ method.

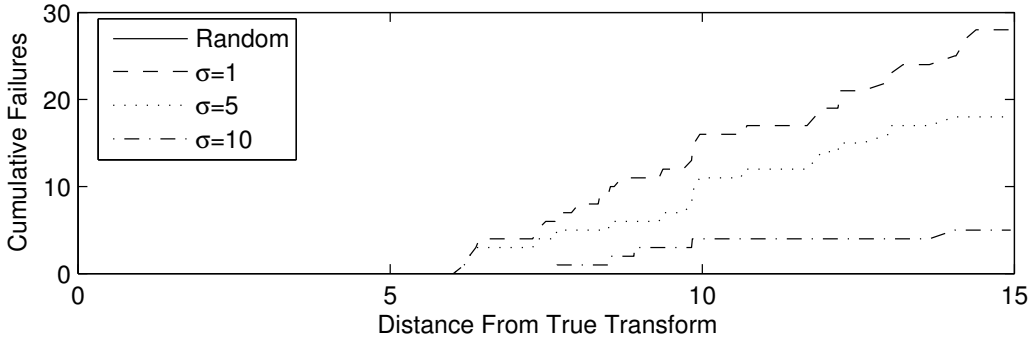
Finally, Figure 6.7c shows the results when Powell's optimization method was used, with the $Tr(\boldsymbol{C})$ criterion. These results show the influence of the scale of the derivative even more strikingly than the results with the gradient descent optimizer.

For the MI measure, slices extracted from previously registered medical imaging volumes[4] were used. The first group of images ({6.8a, 6.8b, 6.8c}) are 3D Xray (3DRX), Computed Tomography (CT) and Magnetic Resonance (MR) images of a cadaver spine, and the second group are Magnetic Resonance images of a human brain using Proton Density (PD), T1-Weighted (T1) and T2-weighted (T2) imaging sequences. These images were registered to each other using a rigid transformation.

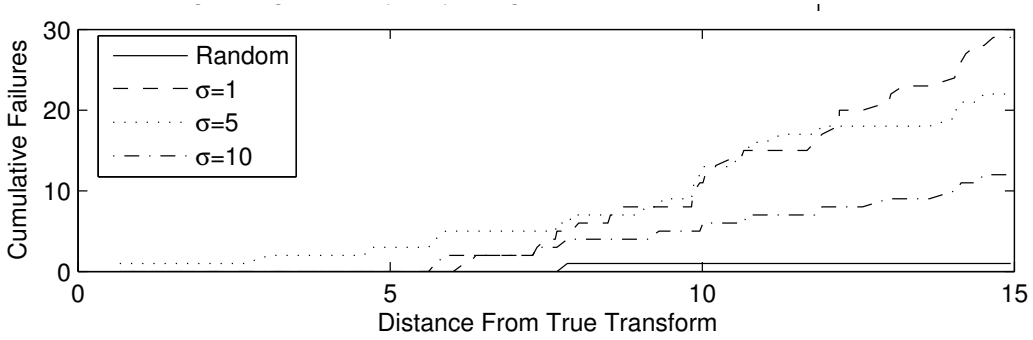
As the images in each group ({6.8a, 6.8b, 6.8c} and {6.8d, 6.8e, 6.8f}) are the same size, the results can be combined by group. The results using a gradient descent optimizer, and the $Tr(\boldsymbol{C})$ criterion, are shown in Figure 6.9. Results for the other selection criteria, and Powell's optimizer, are extremely similar, and have been omitted for brevity.

The main point of the work presented in this chapter was to question the conventional wisdom that pixels should be selected using some property based on their derivative. Overall, there appears to be a significantly higher failure rate when using

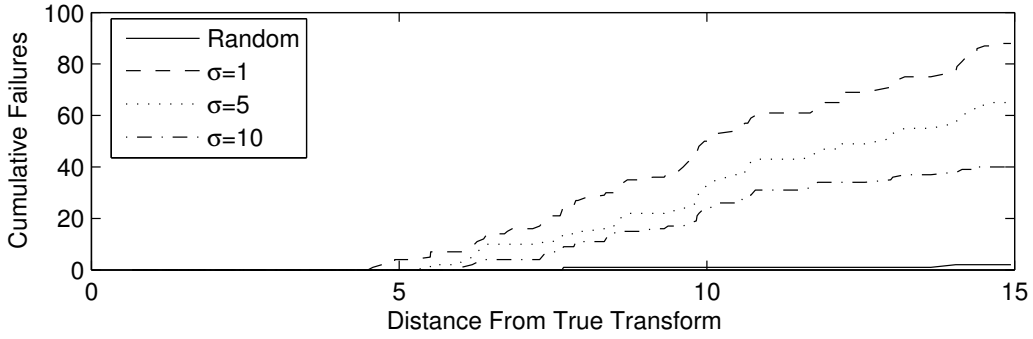
[4]Images {6.8a, 6.8b, 6.8c} from the datasets described in [190] and images {6.8d, 6.8e, 6.8f} are courtesy Montreal Neurological Institute.




(a) Registering Graffiti b) to a) using gradient descent and the $Tr(\boldsymbol{C})$ criterion



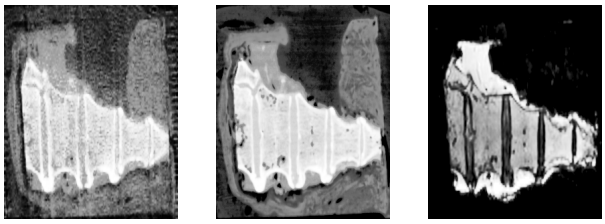

(b) Registering Graffiti a) to b) using gradient descent and the $\prod \boldsymbol{J}^{-2}$ criterion




(c) Registrations of Graffiti images using Powell's method and the $Tr(\boldsymbol{C})$ criterion

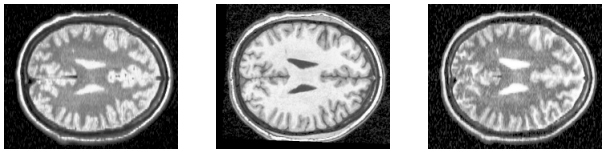
**Figure 6.7** Cumulative failures of registration of the graffiti image pair with gradient descent (a, b) and Powell's method (c). $\sigma$ is the scale of the derivative.

MI Group 1



(a) 3DRX (b) CT (c) MR

MI Group 2

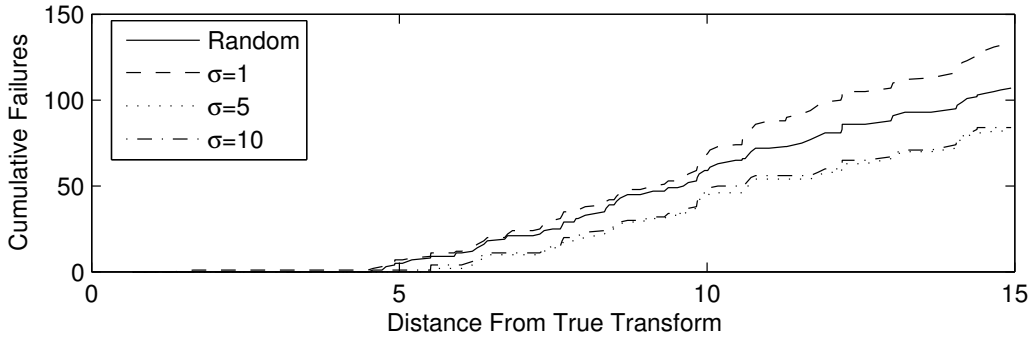


(d) PD (e) T1 (f) T2

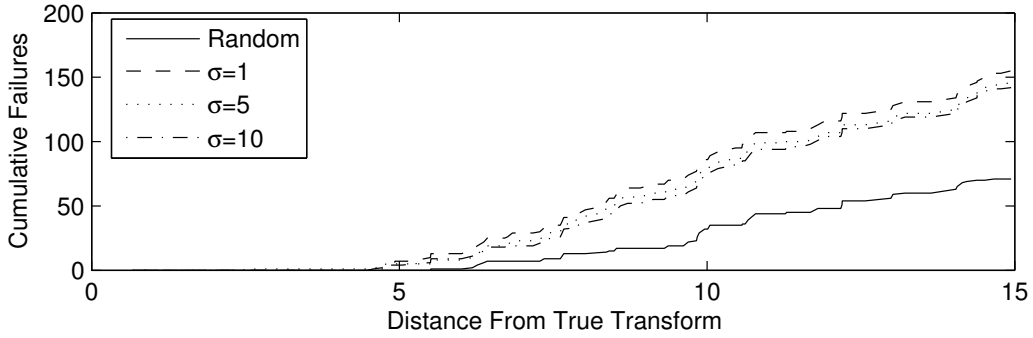
**Figure 6.8** Images used for MI registrations.

pixel selection approaches, and one may ask if it is worthwhile to select pixels at all. These experiments show that using pixel selection methods is very sensitive to the scale of the derivative used to compute the selection criteria, and for many cases, selecting pixels randomly is as good or better.

However, under certain conditions, for example, in the case of MI Group 2 (see the 3rd row of Table 6.2), the use of selected pixels does result in superior performance when compared with using randomly selected pixels. Furthermore, the timing data indicates that, when the conditions of the registration problem make it appropriate, there may be additional computational benefits to using pixel selection methods. Table 6.2 shows the mean run times and mean number of iterations for each scale of selection, considering the cases where (1) the process starts close to the optimum (*i.e.,* within a distance of five pixels for the MSD case, and five mm in the MI cases) and (2) both the original and pixel selection methods succeeded. The registration process runs slightly faster when pixel selection is used with MI. This may be due to requiring fewer iterations of the optimization algorithm than are required by a random selection of pixels.




(a) Results for Image Set 1 {6.8a, 6.8b, 6.8c} using gradient descent and $Tr(\boldsymbol{C})$ criterion




(b) Results for Image Set 2 {6.8d, 6.8e, 6.8f} using gradient descent and $Tr(\boldsymbol{C})$ criterion

**Figure 6.9** Cumulative failures for registration of MI groups 1 and 2

## 6.5 Conclusions

This chapter analyzed the use of pixel selection methods to speed up direct image registration. The $\prod \boldsymbol{J}_i^{-2}$ criterion for selecting optimal pixels was proposed and shown to be better justified and faster than the criterion currently used.

The widely accepted notion that image registration performance can be improved by computing the image difference measure on a selected subset of pixels (*i.e.,* rather than a random subset) has been tested. It is found that this is only true when very particular limitations on the starting positions of the optimization can be met. Specifically, this work examines the important role of the scale of the derivative in the analysis, a point that has been generally overlooked in the literature. When using a pixel selection method, the capture range of the optimization algorithm will be reduced to be approximately the same size as the derivative scale. Should these conditions be met, pixel selection approaches may give additional speed advantages

| Image set | Random | | $\sigma = 1$ | | $\sigma = 5$ | | $\sigma = 10$ | |
|---|---|---|---|---|---|---|---|---|
| | time | iter | time | iter | time | iter | time | iter |
| Graf. (MSD) | 1.58 | 15.4 | 2.12 | 18.2 | 2.13 | 17.1 | 2.14 | 20.4 |
| MI 1 | 1.91 | 15.1 | 1.33 | 14.8 | 1.56 | 17.0 | 1.53 | 15.4 |
| MI 2 | 1.16 | 13.7 | 0.92 | 12.9 | 1.03 | 12.9 | 0.92 | 12.8 |

**Table 6.2** Mean time and number of iterations for selected registrations. The selected cases are where (1) the process starts close to the optimum (*i.e.,* within a distance of five pixels for the MSD case, and five mm in the MI cases) and (2) both the original and pixel selection methods succeeded. The registration process runs slightly faster when pixel selection is used with MI.

superior to those achievable by simply selecting pixels randomly.

It is interesting to note that the method proposed in [23] does not deal explicitly with the issue of scale, or address whether selected sets of pixels should be different for large or small motions. In the tracking context, good convergence properties are more important than precision. The work presented in this chapter suggests that a different set of pixels might be necessary for precise registration, as opposed to the set required for convergence from a far starting position. It would be interesting in future to explore this aspect of the problem.

# Chapter 7

# Generalizing Inverse Compositional and ESM Image Alignment

As discussed in previous chapters, the image registration problem is usually formulated between a pair of images, such that one image is held fixed, and the other is warped by the transformation. The problem can then be expressed as the optimization of some image difference measure over the transformation parameters. This optimization problem has some special characteristics. One is that it is easy to imagine interchanging the roles of the images. Instead of warping the moving image, the reference image could be inverse warped by an equivalent amount. Another special characteristic of this problem is that the optimization step can be done either additively, by adding an update to the warp parameters, or compositionally, by composing the warp parameters. (A compositional step is equivalent to warping the image with one set of parameters, and then warping the result with another set.)

The special structure of the registration optimization problem has been exploited by a number of authors [11, 12, 22, 90, 114, 139, 140, 172] to develop efficient image registration algorithms. Briefly, the gains arise from computing the derivative of the image difference measure with respect to the parameters of the warp of the fixed image. In the case of the inverse compositional method, for example, computing the derivative in this way saves computation time since much of the calculation does not change and can be cached from step to step. However, all of these gains have come at the expense of greater difficulty in implementation, or limiting the domain of application. In this

chapter, it is shown that the fixed image transformation parameters can be viewed as a different parameterization of the moving image transformation parameters. Thus derivatives computed relative to one parameterization can be converted to derivatives relative to another parameterization by using the familiar chain rule. This allows these efficient image registration methods to be generalized to a much wider range of cost functions and optimizers than they are currently being used with. Two of the most successful efficient registration techniques of this type, the inverse compositional method [11, 12], and the efficient second order method [22, 139, 140], are generalized in this way.

This chapter begins with a discussion of the previous work that lead up to the development of the two methods that will be examined in detail. Each of these methods is then discussed in depth, in Sections 7.2, and 7.3. The generalized approach which applies to both of these algorithms is described in Section 7.4. This is followed by a discussion of the implementation in Section 7.4 and the experiments in Section 7.6. The experiments show that the generalized IC method is just as fast as the original, and in certain cases proves much more reliable. The generalized ESM method shows an improvement in reliability over the classical method, rather than an increase in speed. Finally, the overall conclusions are discussed in Section 7.7.

### Publications

The majority of this chapter is based on work being reviewed for publication as:

[34] Rupert Brooks and Tal Arbel. Generalizing inverse compositional and ESM image alignment. *International Journal of Computer Vision*, 2008. SUBMITTED FOR PUBLICATION.

Early work on this problem was published as:

[32] Rupert Brooks and Tal Arbel. Generalizing inverse compositional image alignment. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR2006)*, volume 2, pages 1200–1203, Hong Kong, August 2006.

## 7.1 Previous Work

The image registration problem has two special characteristics which have allowed the development of efficiency improvements. First, a warp of the moving image can be considered equivalent to an inverse warp of the fixed image. Therefore computing a warp for the fixed image is just as valid as warping the moving image. Thus the image registration problem can be generalized to:

$$\begin{bmatrix} \boldsymbol{\phi}_{f_{opt}} \\ \boldsymbol{\phi}_{m_{opt}} \end{bmatrix} = \underset{\boldsymbol{\phi}_f, \boldsymbol{\phi}_m}{\operatorname{argmin}} \left( D(I_f(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}_f)), I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}_m))) \right), \tag{7.1}$$

where $\boldsymbol{\phi}_f$ is the parameter vector of a warp of the fixed image, and $\boldsymbol{\phi}_m$ is the parameter vector of a warp of the moving image. This optimization has many solutions, as there are an infinite number of possible pairs of $\boldsymbol{\phi}_f, \boldsymbol{\phi}_m$ that are equivalent.

Second, many warps that are of interest can be both composed and inverted. Thus it is possible to imagine applying a warp update step to the fixed or moving warp parameters by either adding it to the existing parameters, or by composing it with the existing parameters.

As a result of these two special characteristics there are four ways the update step can be combined with the current warp parameters [11, 12]. While some of these had been used by various authors for efficient registration, Baker and Matthews [11, 12] were the first to identify and categorize them all based on these properties of the image registration problem. In what follows, the terminology and classification proposed in [11, 12] will be used to describe these earlier methods.

In the classical, or *forward additive*, approach [18, 136] the cost function is expressed as it has been throughout this thesis so far, namely,

$$D(\boldsymbol{\phi}_{(n+1)}) = D(I_f, I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}_{(n)} + \Delta\boldsymbol{\phi}_{(n)}))), \tag{7.2}$$

and it is optimized in the usual way based on additive steps, *i.e.*, $\boldsymbol{\phi}_{(n+1)} = \boldsymbol{\phi}_{(n)} + \Delta\boldsymbol{\phi}_{(n)}$. This is a standard unconstrained optimization problem and a wide range of optimization algorithms may be applied to it.

The first efficient alternative exploiting the symmetries of the problem was the *inverse additive* approach proposed by Hager and Belhumeur [90]. They assumed

that the images were approximately aligned, so $\boldsymbol{I}_f \approx I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}))$. The image gradient $\nabla_{\boldsymbol{X}} I_m$, can then be approximated from the gradient of the fixed image, as follows

$$\nabla_{\boldsymbol{X}} \boldsymbol{I}_m \approx \nabla_{\boldsymbol{X}} \boldsymbol{I}_f \left[ \frac{\partial \boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi})}{\partial \boldsymbol{X}} \right]^{-1}$$

They could then develop an efficient update step by replacing $\nabla_x I_m$ with this approximation. However, this approach was only efficient if the warp Jacobian matrix could be factorized and this placed limiting restriction on the warps that could be used.

Shum and Szeliski [172] proposed the *forward compositional* approach. In this approach, the cost function was expressed as

$$D(\boldsymbol{\phi}_{(n+1)}) = D(\boldsymbol{I}_f, I_m(\boldsymbol{W}(\boldsymbol{W}(\boldsymbol{X}, \Delta\boldsymbol{\phi}_{(n)}), \boldsymbol{\phi}_{(n)}))),$$

and the update has to be performed compositionally, *i.e.,* $\boldsymbol{\phi}_{(n+1)} = \boldsymbol{\phi}_{(n)} \circ \Delta\boldsymbol{\phi}_{(n)}$. This approach provided some efficiency gains because at each step the current $\Delta\boldsymbol{\phi}$ is zero, and so the Jacobian, $\frac{\partial \boldsymbol{W}(\boldsymbol{X}, \Delta\boldsymbol{\phi})}{\partial \Delta\boldsymbol{\phi}}$ can be precomputed and cached. However, the gradient of the warped moving image still has to be computed at each step.

In 2001, Baker and Matthews [11] identified and named these three approaches, and proposed a fourth. This fourth approach, the *inverse compositional* approach, is shown to be the most efficient and general of all. In [12], they analyze all these approaches in depth. They show that they are all equivalent to first order, and therefore all of them can be expected to converge. They also shown that the inverse compositional approach is more efficient, and less restrictive than the inverse additive or forward compositional methods. In the following section, the inverse compositional method will be described in detail.

## 7.2 Inverse Compositional Method

Baker and Matthews [12] observed that an update step $\Delta\boldsymbol{\phi}_f$ for the fixed image can be computed efficiently when $\boldsymbol{\phi}_f$ is kept at the identity warp. Because of the equivalence between the transformation of the fixed and moving images, this update of $\boldsymbol{\phi}_f$ is

equivalent to updating $\boldsymbol{\phi}_m$ using

$$\boldsymbol{\phi}_{m_{(n+1)}} = \boldsymbol{\phi}_{m_{(n)}} \circ \left[\Delta\boldsymbol{\phi}_{f_{(n)}}^{-1}\right]$$

This computes the update to $\boldsymbol{\phi}_m$ by *inverting* and *composing* the update step, $\Delta\boldsymbol{\phi}_f$, with the current $\boldsymbol{\phi}_m$ (see Algorithm 5).

**Algorithm 5** Inverse Compositional Image Alignment

**Set** start position $\boldsymbol{\phi}_{m_0}$; $\boldsymbol{\phi}_f = 0$; iteration counter $n = 0$
**Compute** the derivative of the fixed image with respect to the warp parameters, $\boldsymbol{J}_f^T = \nabla_{\boldsymbol{\phi}_f} I_f(\boldsymbol{\phi}_f)$
**Repeat**
  **Compute** $D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$,$\nabla_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$, and/or $\mathcal{H}_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$ as needed by the specific optimization technique.
  **Compute** update step $\Delta\boldsymbol{\phi}_f$ from the results of the previous step.
  **Set** $\boldsymbol{\phi}_{m_{(n+1)}} = \boldsymbol{\phi}_{m_{(n)}} \circ \Delta\boldsymbol{\phi}_{f_{(n)}}^{-1}$ and $n = n + 1$
**Until** convergence criteria reached

The IC method is significantly faster than the forward additive method because the search is always performed around the zero warp of the fixed image, and therefore the Jacobian of the fixed image with respect to the parameters, $\boldsymbol{J}_f = \nabla_{\boldsymbol{\phi}_f} I_f(\boldsymbol{\phi}_f)$, and the Gauss-Newton approximation to the Hessian, $\mathcal{H}_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_m) = \boldsymbol{J}_f^T \boldsymbol{J}_f$, may be precomputed. Clearly, this relies on composition and inversion being valid operations on warps – *i.e.,* that the warps form a group. This is true for the matrix based transformations (Section 3.1.6) and it has been shown in [143] that, even with approximate inverse and composition operators, these relationships will work.

This idea has been applied in several contexts, including active appearance models [143], and 3D medical data [2, 13]. Extensions to the method have incorporated intensity changes into the cost function [19] and application of the approach to mutual information [32, 71]. However, in all cases, the update step must be compositional. This poses a problem because most widely available optimization methods are additive. Thus, using the IC method requires modifications to the optimization algorithm. In previous iterations of the work presented in this chapter, the method was extended to general optimizers and cost functions, but this extension still required careful interaction with the optimizer update step. The advantages of the IC method all relate

to how efficiently it allows computation of $\nabla_{\boldsymbol{\phi}} D$, and the requirements for changes to the optimization algorithm only increase the difficulty of its implementation, and limit its applicability.

## 7.3 Efficient Second Order Minimization

While Baker and Matthews identified the different ways in which the update step could be computed, they did not consider combining these update steps. A method that does this the Efficient Second order Minimization (ESM) which was proposed by Malis and Benhimane [22, 139, 140] for tracking and visual servoing. This method does not speed up the computation of $D$. In fact, it may slow it down as more computation is done at each step. Instead, it achieves its efficiency by reducing the number of optimization steps required to find the optimum.

The ESM method (Algorithm 6) is based on the observation that due to the symmetry of the image registration problem, the Gauss-Newton approximation to the Hessian could be calculated with respect to either $\boldsymbol{\phi}_f$ or $\boldsymbol{\phi}_m$. The update step from each would be slightly different. They show that the approximate Hessian given by the average of the two approaches is a superior approximation to the one from either approach alone. Thus they use an update step of

$$\Delta\boldsymbol{\phi}_{m_{(n)}} = -(\bar{\boldsymbol{J}}^T\bar{\boldsymbol{J}})^{-1}\bar{\boldsymbol{J}}^T\left[\boldsymbol{I}_f - I_m(\boldsymbol{\phi}_{m_{(n)}})\right] \tag{7.3}$$

where $\bar{\boldsymbol{J}}$ is,

$$\bar{J} = \frac{\boldsymbol{J}_m - \boldsymbol{J}_f}{2} = \nabla_{\boldsymbol{\phi}_m} I_m(\boldsymbol{\phi}_m) - \nabla_{\boldsymbol{\phi}_f} I_f(\boldsymbol{\phi}_f), \tag{7.4}$$

the average of the Jacobians computed with respect to either set of transform parameters. A similar insight has been proposed in [114], without addressing the generality of Equation 7.4.

As originally presented, the ESM method could only be applied where a Gauss-Newton approximation is being used. However, it is this author's opinion that the deeper insight of the ESM method is that the update steps computed using the fixed image, and the one using the moving image are not identical, and that the combination of both may prove superior to either one.

**Algorithm 6** Efficient Second Order Image Alignment

**Set** start position $\boldsymbol{\phi}_{m_0}$; $\boldsymbol{\phi}_f = 0$; iteration counter $n = 0$
**Compute** $\boldsymbol{J}_f = \nabla_{\boldsymbol{\phi}_f} I_f(\boldsymbol{\phi}_f)$
**Repeat**
  **Compute** $D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$, $\boldsymbol{J}_m = \nabla_{\boldsymbol{\phi}_m} I_m(\boldsymbol{\phi}_m)$, $\nabla_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$, and $\nabla_{\boldsymbol{\phi}_m} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$.
  **Compute** $\bar{\boldsymbol{J}}$ using Equation 7.4.
  **Compute** update step $\Delta\boldsymbol{\phi}_{m_{(n)}}$ based on Equation 7.3
  **Set** $\boldsymbol{\phi}_{m_{(n+1)}} = \boldsymbol{\phi}_{m_{(n)}} \circ \Delta\boldsymbol{\phi}_{m_{(n)}}$ and $n = n + 1$
**Until** convergence criteria reached

A more serious limitation is that, as discussed in [22, 139, 140], the ESM method can only be applied in very specific circumstances. For it to work, combining the two Jacobians using Equation 7.4 must be a meaningful thing to do. This implies that the warp must have the property that $\boldsymbol{\phi}^{-1} = -\boldsymbol{\phi}$. This is not true in general. However, it is true in the neighborhood of the identity warp, when the warp forms a Lie group and is parameterized using its exponential map [140]. Thus, they must express the warp using that parameterization, and compute the $\boldsymbol{J}$'s for both the fixed and moving image around the identity warp. This also requires the use of the forward compositional approach for $\boldsymbol{J}_m$. This limits the applicability of this approach. It may be undesirable to use the exponential map parameterization, as it requires an evaluation of a matrix exponential, which can be difficult [146]. Furthermore, while moving along the exponential map is a geodesic on the transformation manifold, it generally induces a relatively curved path in the image space, which can make the optimization more difficult.

In the following section, it is shown that the symmetry in the image registration problem means that the moving image warp can be considered as a smooth function of the fixed image warp. As a result, gradients with respect to one set of parameters can be converted to gradients with respect to the other. This allows us to generalize both methods to use additive optimizers, and any parameterization of the transformations that allows composition and inversion.

## 7.4 Generalized Approach to IC and ESM

The IC and ESM methods for efficient image alignment have special characteristics which make their widespread adoption difficult. The IC approach maps update steps from the space of $\boldsymbol{\phi}_f$ to the space of $\boldsymbol{\phi}_m$ using inversion and composition. This requires customization of any optimization algorithm intended to be used. The ESM approach combines the forward and inverse Gauss-Newton steps, but to do so relies on the transformation having the property that $\boldsymbol{\phi}^{-1} = -\boldsymbol{\phi}$. This is undesirable for similar reasons: such parameterizations are not universally available, and may present implementation difficulties.

Note that both the IC and ESM approaches rely on the special equivalence that exists in the pairwise image registration problem. Specifically,

$$D(I_f(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}_f)), I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}_m))), \tag{7.5}$$

is equivalent to

$$D(I_f(\boldsymbol{X}), I_m(\boldsymbol{W}(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}_f^{-1}), \boldsymbol{\phi}_m))). \tag{7.6}$$

When the composition and inversion operations are smooth and differentiable – something that is implied already by the existing requirements to use either of these methods – then there is a one-to-one differentiable mapping between warps of the fixed image and warps of the moving image. At a particular step $n$, the moving image warp can be expressed as a function of the fixed image warp:

$$\boldsymbol{\phi}_{m_{(n+1)}}(\boldsymbol{\phi}_{f_{(n)}}) = \boldsymbol{\phi}_{m_{(n)}} \circ \boldsymbol{\phi}_{f_{(n)}}^{-1},$$

which reparameterizes the cost function in Equation 7.2 in terms of $\boldsymbol{\phi}_f$:

$$D(I_f(\boldsymbol{X}), I_m(\boldsymbol{W}(\boldsymbol{X}, \boldsymbol{\phi}_{m_{(n)}}(\boldsymbol{\phi}_{f_{(n)}})))).$$

The inverse compositional method provides an efficient way of computing $\frac{\partial D}{\partial \boldsymbol{\phi}_f}$, but standard optimization methods require the derivative with respect to $\boldsymbol{\phi}_m$, $\frac{\partial D}{\partial \boldsymbol{\phi}_m}$.

Expressing $\boldsymbol{\phi}_f$ as a function of $\boldsymbol{\phi}_m$, and applying the chain rule yields:

$$\frac{\partial D}{\partial \boldsymbol{\phi}_m} = \frac{\partial D}{\partial \boldsymbol{\phi}_f} \cdot \frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m}. \tag{7.7}$$

The Gauss-Newton approximation to the Hessian is the sum of the outer products of the gradient of each pixel with respect to the transformation parameters.

$$\mathcal{H}_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_m) \approxeq \boldsymbol{J}_f^T \boldsymbol{J}_f = \sum_i \frac{\partial \boldsymbol{I}_{f_i}}{\partial \boldsymbol{\phi}_f}^T \cdot \frac{\partial \boldsymbol{I}_{f_i}}{\partial \boldsymbol{\phi}_f}, \tag{7.8}$$

where $\boldsymbol{I}_{f_i}$ is pixel $i$ of $\boldsymbol{I}_f$. Each of these partial derivatives can be converted using the chain rule as above,

$$\frac{\partial \boldsymbol{I}_{m_i}}{\partial \boldsymbol{\phi}_m} = \frac{\partial \boldsymbol{I}_{f_i}}{\partial \boldsymbol{\phi}_f} \cdot \frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m},$$

giving,

$$\mathcal{H}_{\boldsymbol{\phi}_m} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_m) \approx \sum_i \frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m}^T \frac{\partial \boldsymbol{I}_{f_i}}{\partial \boldsymbol{\phi}_f}^T \cdot \frac{\partial \boldsymbol{I}_{f_i}}{\partial \boldsymbol{\phi}_f} \frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m},$$

which simplifies to

$$\mathcal{H}_{\boldsymbol{\phi}_m} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_m) \approx \frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m}^T \sum_i \left[ \frac{\partial \boldsymbol{I}_{f_i}}{\partial \boldsymbol{\phi}_f}^T \cdot \frac{\partial \boldsymbol{I}_{f_i}}{\partial \boldsymbol{\phi}_f} \right] \frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m}. \tag{7.9}$$

Thus, it is possible to convert the gradient and Hessian of the image difference measure with respect to the fixed image warp parameters to be derivatives with respect to the moving image warp parameters as long as it is possible to compute the Jacobian matrix, $\frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m}$.

### 7.4.1 Generalized Inverse Compositional Alignment

The proposal for generalizing the IC algorithm is to use these methods to convert the derivative and Gauss-Newton approximation to the Hessian from being with respect to the fixed image warp parameters to being with respect to the moving image warp parameters. These converted derivatives can be provided to any standard optimization technique, without requiring any special modifications of the optimization algorithm. In particular, the update step is now additive, which permits using standard additive

optimizers. The resulting generalized algorithm is shown in Algorithm 7.

**Algorithm 7** Generalized Inverse Compositional Image Alignment

**Set** start position $\boldsymbol{\phi}_{m_{(0)}}$; $\boldsymbol{\phi}_f = 0$; iteration counter $n = 0$
**Compute** $\boldsymbol{J}_f = \nabla_{\boldsymbol{\phi}_f} I_f(\boldsymbol{\phi}_f)$
**Repeat**
  **Compute** $D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$, $\nabla_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$ and $\mathcal{H}_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$ as required
  **Convert** $\nabla_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$ to $\nabla_{\boldsymbol{\phi}_m} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$ and $\mathcal{H}_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_m)$ to $\mathcal{H}_{\boldsymbol{\phi}_m} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$ using Equations 7.7, 7.9
  **Compute** update step $\Delta\boldsymbol{\phi}_m$ using the chosen optimization method.
  **Set** $\boldsymbol{\phi}_{m_{(n+1)}} = \boldsymbol{\phi}_{m_{(n)}} + \Delta\boldsymbol{\phi}_{m_{(n)}}$ and $n = n + 1$. Note that this is an additive step.
**Until** convergence criteria reached

One may ask how this generalization will affect the steps taken by the optimization process. The steps taken by this algorithm will differ from the original IC algorithm. The original algorithm – when not using a single Gauss-Newton step – moves along a straight line in the $\boldsymbol{\phi}_f$ space. This line is generally curved if it is converted into the $\boldsymbol{\phi}_m$ space. The generalized approach moves along a straight line in $\boldsymbol{\phi}_m$ space. This line is tangent to the curve produced by the original algorithm (See Fig. 7.1).

### 7.4.2 Generalized ESM

The ESM method can also be generalized using Equations 7.7–7.9. Any derivative with respect to $\boldsymbol{\phi}_f$ can be converted to the space of $\boldsymbol{\phi}_m$ before averaging it with the corresponding derivative computed relative to $\boldsymbol{\phi}_m$[1]. It is no longer necessary to restrict the parameterization so that Equation 7.4 is valid: any parameterization where the IC method will work is suitable. This leads to a generalized ESM algorithm shown in Algorithm 8.

It is more difficult to compare the steps taken by the generalized ESM algorithm to the ones taken by the original algorithm. In the exponential map based parameterizations used by the original implementation [22, 139, 140], $\frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m} = -\boldsymbol{I}$ and the steps are identical to those taken by Algorithm 6. In the more general cases to which it can now be applied the different parameterizations give rise to different derivatives,

[1]However, the name "Efficient Second Order Method" may become a misnomer when it is generalized in this way, as it is only a second order approximation when the Jacobian pseudoinverse is being used.

**Algorithm 8** Generalized "ESM" Image Alignment

**Set** start position $\boldsymbol{\phi}_{m_0}$; $\boldsymbol{\phi}_f = 0$; iteration counter $n = 0$
**Compute** $\boldsymbol{J}_f = \nabla_{\boldsymbol{\phi}_f} I_f(\boldsymbol{\phi}_f)$
**Repeat**
  **Compute** $D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$, $\nabla_{\boldsymbol{\phi}_{m_{(n)}}} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$, $\nabla_{\boldsymbol{\phi}_f} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$, $\nabla_{\boldsymbol{\phi}_m} D(\boldsymbol{\phi}_f, \boldsymbol{\phi}_{m_{(n)}})$ and $\boldsymbol{J}_m(\boldsymbol{\phi}_{m_{(n)}})$ as required
  **Convert** $\boldsymbol{J}_f$ to the space of $J_m$ using Equation 7.9 and average with $\boldsymbol{J}_m$ to form $\bar{\boldsymbol{J}}$.
  **Compute** compute update step $\Delta\boldsymbol{\phi}_{m_{(n)}}$ using the chosen optimization algorithm.

  **Set** $\boldsymbol{\phi}_{m_{(n+1)}} = \boldsymbol{\phi}_{m_{(n)}} + \Delta\boldsymbol{\phi}^{-1}_{m(n)}$ and $n = n + 1$
**Until** convergence criteria reached

different Gauss-Newton hyperplane approximations (in Fig. 4.8) and thus different steps. The importance of this generalization is that the ESM method can now be applied in a much wider set of contexts than before.

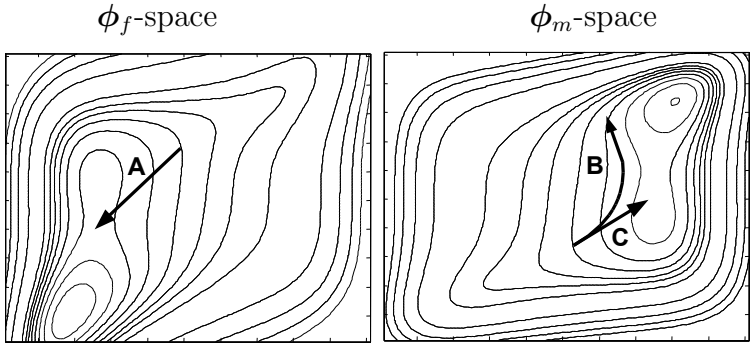



**Figure 7.1** The cost function with respect to the fixed image parameters (left) and the moving image parameters (right) is related with a non-linear mapping. Thus a linear movement (A) of the fixed image parameters, $\boldsymbol{\phi}_f$, can be transformed to a (generally curved) movement (B) of the moving image parameters, $\boldsymbol{\phi}_m$. With the proposed method, it is the tangent to the curve at its beginning that is mapped, and the motion is linear (C) in the moving image parameter space.

The $\frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m}$ matrix is key to the implementation of both methods. Appendix F contains analytical evaluations of this matrix for 2D homographies, and 3D rigid transformations. The generalized approaches rely on the calculation of the $\frac{\partial \boldsymbol{\phi}_f}{\partial \boldsymbol{\phi}_m}$ matrix at each iteration, followed by a matrix-vector multiplication in the case of the

derivative, and two matrix multiplications in the case of the approximate Hessian. These operations are of order $O(n^3)$, where $n$ is the number of parameters. For typical image alignment problems in 2 and 3 dimensions, the warp typically has tens of parameters while the images typically have thousands of pixels. Thus these operations are not the computational bottleneck in the process. In addition, the original IC method included an inversion, and a composition of the warp parameters, which are operations of equivalent complexity. Therefore, the generalized methods retain the advantages of the IC and ESM methods, which are, respectively, faster and more accurate calculation of the update step. The generalized methods avoid the restrictions on optimization technique and transform parameterization that affected the original methods, without incurring a greater computational cost.

## 7.5 Implementation

To assess the effectiveness of the generalized approach, it has been implemented in the standard multiscale image registration framework (Section 3.1). Each generalized approach described above has been implemented, along with the classical, forward additive method. Each these has been implemented for three widely used image difference measures (MSD, NCC and MI), and three widely used optimizers (gradient descent, BFGS, and Newton-Raphson). For comparison purposes, an original IC method which takes steps compositionally in the parameter space has also been implemented. For the case of the mean squared difference measure and the Newton-Raphson optimizer, this implementation of the original IC method is nearly identical to the method described in [12], which provides a meaningful base-line comparison for this work. For the other difference measure and optimizer contexts, this original IC implementation is not precisely comparable to other published work, as it uses a different optimization strategy compared to [32] and a different MI formulation than [71]. Nevertheless, it provides an interesting check that this generalization has not lost critical elements of the method.

### 7.5.1 Image Difference Measures

To use an image difference measure in an IC or ESM context it must be possible to compute the derivative and Hessian with respect to $\boldsymbol{\phi}_f$ rather than $\boldsymbol{\phi}_m$. For the MSD and NCC measures, this is straightforward. The expressions for these measures can be found in Sections 3.1.2 and 3.1.3. Note that in both cases, the roles of the fixed and moving images are totally symmetric; thus, obtaining $\frac{\partial D}{\partial \boldsymbol{\phi}_f}$ and $\frac{\partial^2 D}{\partial \boldsymbol{\phi}_f^2}$ is easily done by reversing the roles of $\boldsymbol{I}_f$ and $\boldsymbol{I}_m$ in the expressions for each measure.

#### Mutual Information

The mutual information implementation used in this thesis is based on the efficient formulation of [185]. As described in Section 3.1.4 this method accumulates a Parzen-windowed model of the joint distribution,

$$\boldsymbol{P}_{kl}(\boldsymbol{\phi}_f, \boldsymbol{\phi}_m) = \sum_{i=1}^{N} \frac{1}{N} \beta_0 \left( k - \frac{I_f(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}_f)) - b_{f_0}}{d_f} \right) \beta_3 \left( l - \frac{I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}_m)) - b_{m_0}}{d_m} \right) \tag{7.10}$$

and its derivatives

$$\nabla_{\boldsymbol{\phi}} \boldsymbol{P}_{klp}(\boldsymbol{\phi}_f, \boldsymbol{\phi}_m) = \sum_{i=1}^{N} \frac{1}{d_m N} \frac{\partial \beta_3 \left( l - \frac{I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}_m)) - b_{m_0}}{d_m} \right)}{\partial I_{m_i}} \cdot \frac{\partial I_{m_i}}{\partial \boldsymbol{\phi}}. \tag{7.11}$$

over all the pixels.

Unfortunately, this formulation is asymmetric. Because a 3rd order B-spline is differentiable, the gradient of the joint histogram with respect to $\boldsymbol{\phi}_m$, $\nabla_{\boldsymbol{\phi}} \boldsymbol{P}_{kl}(\boldsymbol{\phi})$, can be easily computed. The derivative with respect to $\boldsymbol{\phi}_f$ however, involves the derivative of a zero-th order B-spline, which is undefined.

To overcome this issue, the MI implementation was modified to compute this gradient by reversing the roles of the reference and the fixed image for the gradient part of the calculation only. That is, an additional joint distribution,

$$\boldsymbol{Q}_{kl}(\boldsymbol{\phi}_f, \boldsymbol{\phi}_m) = \sum_{i=1}^{N} \frac{1}{N} \beta_0 \left( k - \frac{I_m(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}_m)) - b_{f_0}}{d_f} \right) \beta_3 \left( l - \frac{I_f(\boldsymbol{W}(\boldsymbol{X}_i, \boldsymbol{\phi}_f)) - b_{m_0}}{d_m} \right) \tag{7.12}$$

is accumulated, along with its derivatives in terms of $\boldsymbol{\phi}_f$ instead of $\boldsymbol{\phi}_m$. Then $D_{MI}$ is computed in the original way, but its gradient and Hessian are computed by substituting $\boldsymbol{Q}_{kl}$ for $\boldsymbol{P}_{kl}$, and swapping the roles of the summation indexes $k$ and $l$. This has the effect of increasing the computational load somewhat to compute the derivatives with respect to $\boldsymbol{\phi}_f$. Computational savings are still obtained, but they are not as large as they would be in a symmetric formulation of MI.

### 7.5.2 Optimization

The registrations have been performed using three optimizers: a gradient descent, a Newton-Raphson method, and a quasi-Newton method, which are broadly representative of the types of optimization algorithms being used in image registration currently.

Gradient descent methods, despite having a number of known problems, are still widely used due to their simplicity. All such methods operate by taking steps in the direction of the gradient, and the methods are mainly differentiated by how the sizes of these steps are chosen. A trust-region approach has been used to adapting the step size in the gradient descent method, as described in Section 3.1.10, and Algorithm 2.

Quasi-Newton methods are widely used when the Hessian is unavailable. They work by building up an approximation to the Hessian from the sequence of function gradients that they receive. Of these methods, the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update is generally considered to be superior [151, 192]. In this work, the limited memory BFGS implementation described by [48] was used. Briefly, the approach approximates the inverse Hessian from a limited length sequence of the gradients that it has received. At each direction, it selects a search direction based on this estimate, and performs a line search in that direction. Complete details can be found in [48, 207]. The original IC algorithm does not support the quasi-Newton method, and so quasi-Newton experiments with the original IC method were not performed.

Newton-Raphson optimization methods have been most widely used with the inverse-compositional approach, including Gauss-Newton methods in [12], and Levenberg-Marquardt methods in [71]. Trust-region Newton-Raphson methods are closely related to the Levenberg-Marquardt method, but are somewhat more recent and general [59]. For the second order model, solving the trust-region subproblem is more

complex. The Steihaug-Toint algorithm [59, p. 205] was used for this purpose. Algorithm 3 describes the trust-region Newton-Raphson approach in detail. Details of the Steihaug-Toint solution can be found in [59].

In the experiments that follow, the generalized method and the original method have been applied to a variety of datasets both synthetic and real. Not all image difference measures or optimizers are appropriate for every context, but each of the measures has been tested with each of the optimizers in at least one context.

## 7.6 Experiments

The algorithms were tested by running numerous image registrations from different starting positions. Each algorithm was applied using a multiresolution approach, with the number of levels dependent on the size of the image. Scaling of the optimization parameters is important for any optimization algorithm. The parameters have been scaled using the optimal scaling factors described in Section 4.5. Each optimizer used similar stopping criteria, specifically, that the minimal scaled step size should be 0.005 units, and the minimal scaled gradient magnitude should be 0.05 units. A large bound (400) was placed on the number of iterations and reaching that bound was considered a failure. This was an extremely rare occurrence.

The performance of the algorithms was measured, and tested for statistically significant differences using the testing approach described in Section 3.2. Specifically, the experiments recorded the time required, the positional accuracy, the number of function evaluations required and the failure rate. All runs were performed on 2.2GHz Sun Opteron 875 processors using a single threaded implementation. Run time is the time required to initialize and run the optimization, but does not include the time to load the data from disk.

It is worth considering just how much speed up could possibly be achieved by this method. Each evaluation of the cost function also requires the calculation of its derivative and possibly its Hessian. The calculation of the derivative requires the computation of the derivative at each pixel, and the summation of these values. The IC method speeds up the computation of the derivative at each pixel (the per-pixel derivative). For Hessian based methods, the IC method allows caching of the Hessian,

so there are some further savings.

There are at least two major ways to implement the calculation of $\frac{\partial D}{\partial \phi_m}$ in the forward additive approach. One, used by [12], warps the image and computes the derivative on this warped image at each iteration. The other approach is to compute the derivative of the image once, cache it, and warp this derivative at each evaluation. The second approach is much faster, and is what was used here for comparison.

To determine just how much speed up could possibly be achieved under the best possible conditions, the Callgrind profiler [200] was used to determine approximately what percentage of the computation is spent on evaluating the per-pixel derivative. This forms an upper bound on the speedup achievable by the IC method for gradient based methods. This upper bound cannot ever be achieved in practice because some time must be spent computing the derivative once, and time must still be spent to retrieve the data from memory. Nevertheless, it provides a guide for evaluating the effectiveness of an IC implementation. The results of the timings are shown in Table 7.1 and indicate that it may be possible to speed up the time by roughly a third, with the improvement for the MI metric likely to be a little less than the others. Some additional improvement may be possible for the Hessian based methods, as there is no longer any need even to sum up the computed derivatives once the Hessian is cached.

| Measure | MSD | NCC | MI |
|---|---|---|---|
| Percent time | 38% | 36% | 29% |

**Table 7.1** Percentage of registration time spent computing the per-pixel derivative. This is a upper bound on the speedup achievable using the inverse compositional method.

Four sets of experiments with these algorithms are presented. One of the only ways to have a true gold standard for accuracy is to warp images by a known transformation and try to recover it. Experiments 1 and 2 utilize images synthetically warped by known, random transforms. Experiment 1 consists of 2D images, warped by a homography, and Experiment 2 consists of 3D image volumes warped by a rigid 3D transformation. These transforms are then recovered with these algorithms and the accuracy is evaluated. While synthetically warped data provides insight into the true accuracy of the algorithms, working with real data often provides more insight about the reliability of an image registration method. Experiments 3 and 4 present

experiments on real cases similar to the simulated cases in experiments 1 and 2.

### 7.6.1 Experiment Set 1: Synthetic Homographies

(a) Original (b) Original cropped to ROI (c) Example of warped image

**Figure 7.2** Example of input data for Experiment 1.

When two images are taken from a perspective camera that is rotating about its optical center, or when a moving perspective camera takes multiple images of a planar scene, those images are related by a homography. Not surprisingly, the homography is one of the most commonly used transforms in computer vision applications. Ten synthetic homographies were generated and used to warp the image shown in Figure 7.2a. The central part of the image was then cropped out and used for the registration tests. This cropping process avoided having the edge of the image in the region to be registered, which could have skewed the results. The starting image, and an example of the one of the warped input images are shown in Fig. 7.2. All work was done with greyscale versions of the images.

For the first part of this experiment, each warped image was registered to the original from 80 starting positions at distances from 2.6 to 81 pixels in mTRE. All registrations were attempted in both directions, that is, both choices for which image was fixed and which was moving were tried. Four multiresolution levels were used, and, for speed reasons, the cost functions were evaluated using a randomly chosen 30% of the pixels. Each of the three cost functions under investigation was used to register the data. The results are shown in Figure 7.3.

The results shown in Figure 7.3 a) show a clear ranking in terms of time required to complete with ESM being the slowest, followed by the classical, forward additive, algorithm, and an approximate tie between the generalized IC method and the original

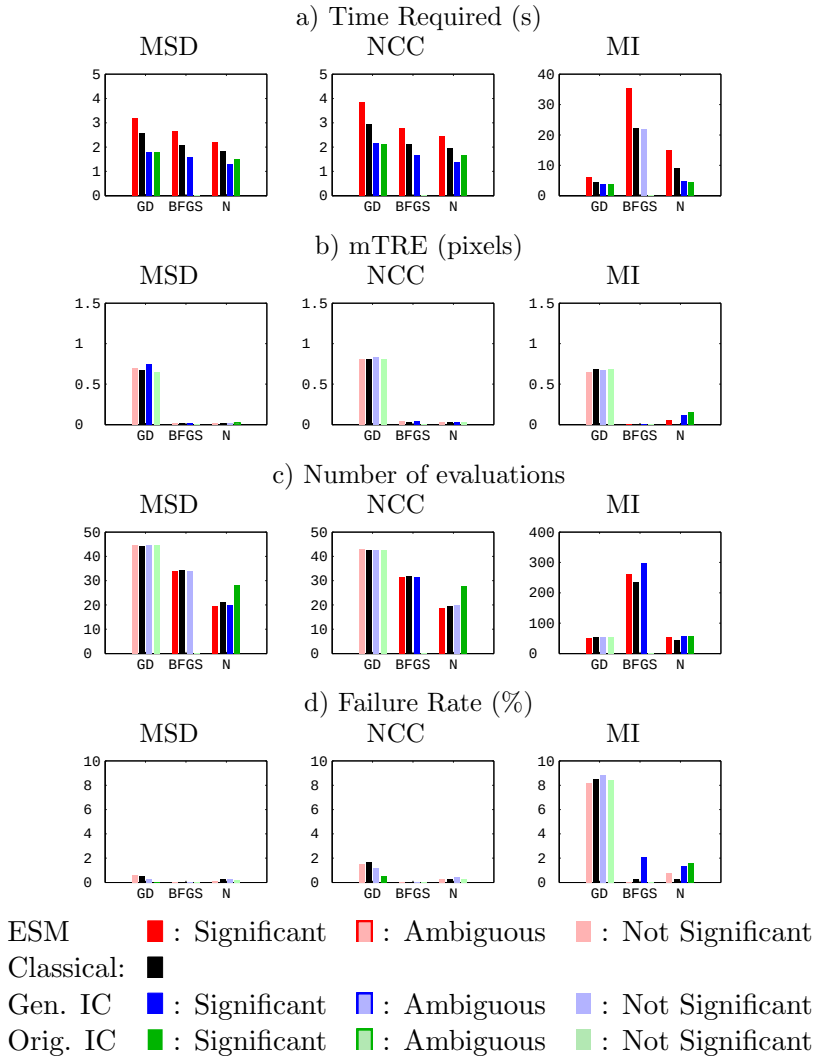



**Figure 7.3** Experimental results for synthetic homographies, case 1: Starting normal distances from the true position. The optimization algorithms are Gradient Descent (GD), Broyden-Fletcher-Goldfarb-Shanno (BFGS) and Newton-Raphson (N). For each algorithm, bars are shown for the following methods from left to right: 1: ESM (Red) 2: Classical Forwards Additive (Black) 3: Generalized IC (Blue) and 4: Original IC (Green). Dark bars are statistically significantly different from classical, light bars are not significantly different and outline bars are ambiguous (see Section 3.2.1). No original IC method was implemented for the BFGS optimizer.

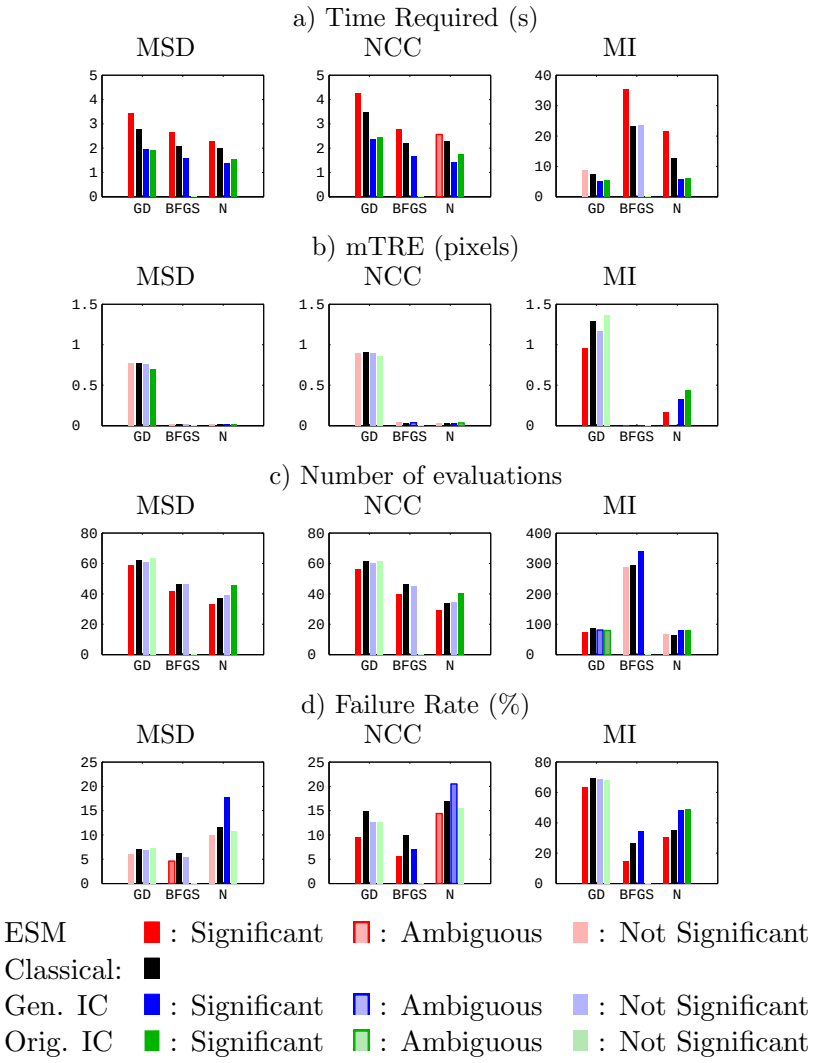



**Figure 7.4** Experimental results for synthetic homographies, case 2: Starting far from the correct transform. The optimization algorithms are Gradient Descent (GD), Broyden-Fletcher-Goldfarb-Shanno (BFGS) and Newton-Raphson (N). For each algorithm, bars are shown for the following methods from left to right: 1: ESM (Red) 2: Classical Forwards Additive (Black) 3: Generalized IC (Blue) and 4: Original IC (Green). Dark bars are statistically significantly different from classical, light bars are not significantly different and outline bars are ambiguous (see Section 3.2.1). No original IC method was implemented for the BFGS optimizer.

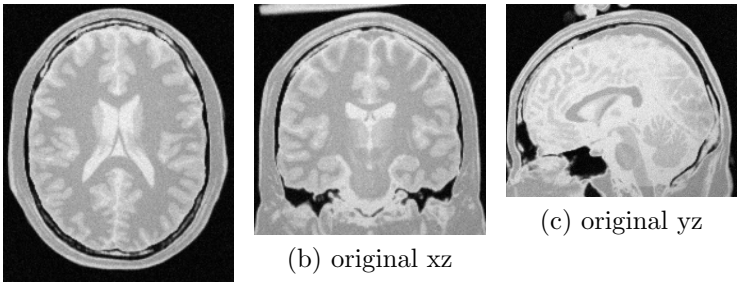
IC method. This implies that the new approach is generalizing the original IC method, without losing any of its speed advantages. The one exception is that the IC BFGS method for MI does not show a speed advantage. Upon examining the number of evaluations, it was determined that this is because it has required more evaluations to complete. It is the author's opinion that this effect occurs because of the way the gradient was formulated. Due to the asymmetry of the problem, the IC gradient is slightly less accurate than the forward gradient. The line search used in the BFGS algorithm forms a cubic model of the function using the gradient, and is sensitive to this. This does not seem to affect the other optimizers at all.

Other than the issues with the BFGS method, the differences in accuracy (as measured by mTRE) and number of evaluations are very small. Nevertheless, several of the bars are colored as statistically significant. Upon investigation of the data, this occurs because the IC method has a tendency to stop slightly early. Because the tendency is consistently biased one way, even if small, the pairwise statistical tests can pick it up. Its effect on the overall results is very minor.

While the ESM method does not have any speed advantage for this problem configuration, it does show a reliability advantage over the classical method. Direct image registration using this approach is dependent on the *capture radius* of the optimizer being used. The capture radius is the distance (in parameter space) from the true position that the optimizer can be started and still be expected to converge. The results of this experiment shown in Figure 7.3 suggest that the ESM may improve the capture radius and thus give more reliable results. In this experiment, all the failure rates are small and it is difficult to draw conclusions. For this reason, another experiment was required to test this conjecture. A second set of similar experiments (Figure 7.4) were performed starting far from the correct registration. For these experiments, each image was registered from 50 starting positions with mTRE ranging from 36 to 176 pixels. Not surprisingly, failure rates in all cases are higher, and this allows us to see that the failure rate for the ESM method is significantly lower than for the other methods. Intuitively, this is because the optimizer chooses the average of the directions calculated by the IC and forward methods. If only one of the two methods is outside its capture radius, the other may provide sufficient information for the optimizer to converge.

Similar to the work of [71], this experiment shows that the IC methods have a slightly smaller capture radius. This cannot be only due to the inclusion or exclusion of certain pixels in the Hessian matrix (as thought by [71]) since the issue is also present in optimizers that do not use the Hessian.

These results also show clear differences between the optimizers. The BFGS optimizer and Newton-Raphson optimizer are generally more accurate and require fewer iterations than the gradient descent optimizer on the MSD and NCC metrics. On the MI metric, the BFGS optimizer takes more iterations than the other two, for all methods. This is probably due to the negative curvature in the MI function. The MI function is saddle shaped, meaning its Hessian is not positive definite, but the BFGS update method enforces a constraint that the Hessian be positive definite. It is probable that this mismatch between the algorithm assumptions and the cost function properties is causing this optimizer to require more iterations. This is similar to the results found in Section 4.6.



(a) original xy (b) original xz (c) original yz

(d) Warped xy (e) Warped xz (f) Warped yz

**Figure 7.5** Slices through the center of the Brainweb volume used. Top row, original volume, bottom row, warped volume.

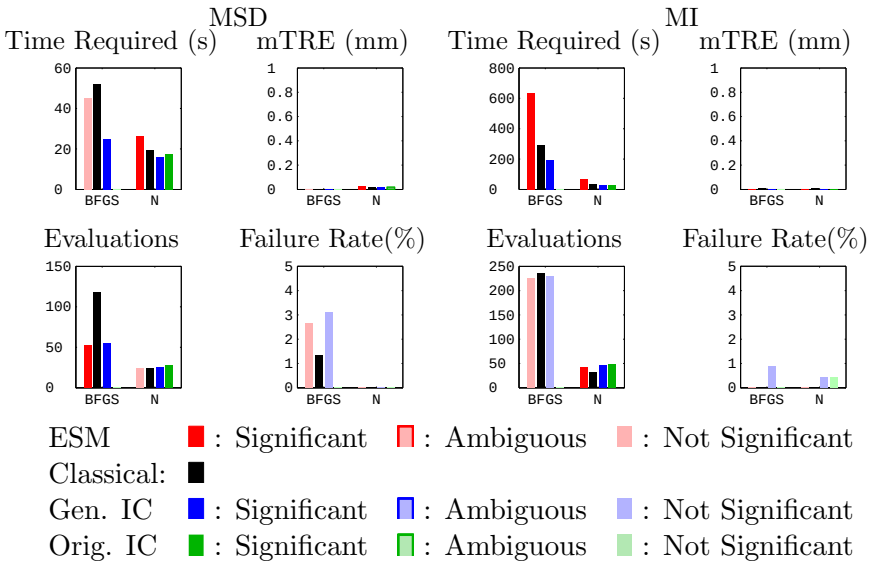



**Figure 7.6** Experimental results for synthetic rigid 3D transforms for MSD (left) and MI (right). The optimization algorithms are Broyden-Fletcher-Goldfarb-Shanno (BFGS) and Newton-Raphson (N). For each algorithm, bars are shown for the following methods from left to right: 1: ESM (Red) 2: Classical Forwards Additive (Black) 3: Generalized IC (Blue) and 4: Original IC (Green). Dark bars are statistically significantly different from classical, light bars are not significantly different and outline bars are ambiguous (see Section 3.2.1). No original IC method was implemented for the BFGS optimizer.

### 7.6.2 Experiment Set 2: Synthetic 3D Rigid Transformations

Only limited work with the inverse compositional method on 3D images has been reported [2, 13], although its efficiency gains are particularly relevant to this problem due to the size of the images. In this experiment, a synthetic Magnetic Resonance Imaging (MRI) volume generated using the Brainweb [123] software was used. Five random rigid transformations were created, and the original image was warped by each of them. Registration was then performed from 45 starting positions ranging from 2 to 43 mm in mTRE.

As the brain image (illustrated in Figure 7.5) was surrounded by a black background, edge effects were considered to be small, and a cropping step was not performed as it was in the previous experiment. Three multiresolution levels were used in the optimization, and, as in the previous experiments, a random 30% of the image

voxels were used to calculate the measures. To save time and space, and because MSD and MI are the most widely used measures in medical imaging applications, these experiments were not performed with NCC. Also, as the results for gradient descent in the previous experiment were consistent with the other methods, but it was slower and less accurate than BFGS or Newton-Raphson, it was omitted from these experiments.

Overall, the results shown in Figure 7.6 are entirely consistent with the results of experiment 1 using synthetic homographies. The generalized IC approach is consistently faster than the classical approach, even for the BFGS optimizer in this case. This is interesting to demonstrate, as the mapping $\boldsymbol{\phi}_f(\boldsymbol{\phi}_m)$ is much more non-linear in this case than in the previous case (see Appendix F). This means that the difference between the path explored by the generalized method and the original IC method (see Fig. 7.1) is more significant in experiment 2. That the generalized approach works still achieves speed gains here is a sign of its effectiveness.

There is a clear, and statistically significant difference in the time required for the algorithms to run, with ESM being slower than the classical, forward additive, method, and with both IC methods being faster. The differences in evaluations and mTRE are tiny, but in certain cases appear significant. As in the previous experiment, this is due to a slight but consistent bias – in this case the IC method is occasionally taking an additional iteration to complete, and gaining a marginal (on the order of 0.01 pixels) amount of accuracy. Such a tiny difference does not have any practical impact. The failure rates on this problem were very low and no statistically significant differences between algorithms were detected. Overall, the generalized IC method is faster than the classical method, without increasing the failure rate.

### 7.6.3 Experiment Set 3: Real Homographies

In this experiment, 8 image pairs taken with a camera being rotated on its optical center were acquired. Under these conditions, the transformation between the images can be modeled by a homography, and this is often done to allow panoramic stitching of the images. A form of ground truth was created using the Autostitch software [42], and manually checked for accuracy. However, this ground truth can only be considered a *silver standard* – an independent comparison, but not absolutely accurate.

**Figure 7.7** A typical image pair for Experiment 3

These images present a moderately difficult registration problem. One reason is that the amount of overlap is limited. Furthermore, no correction for lens distortion was made; thus, the true transformation from image to image is not a perfect homography. Nevertheless, these conditions are realistic, and make the problem more interesting. Figure 7.7 shows a typical image pair from the set.

For this experiment, the images were registered from 20 starting positions ranging from approximately 3 to 19 pixels in mTRE. All registrations were attempted in both directions, that is, both choices for which image was fixed and which was moving were tried. Due to the significant lighting changes between images in any given pair, MSD was an ineffective measure, and was not used for these experiments. As in experiment 1, four multiresolution levels were used, and a random 30% of the pixels were taken to compute the measures. All registrations were performed on greyscale versions of the images.

For brevity, the results for all images have been combined in the graphs presented in Figure 7.8. The images are of different sizes and characters, so these numbers must be interpreted in that context. (The statistical significance comparisons are pairwise, however, and are therefore valid.) As in the previous two experiments, there was little interesting difference in the number of evaluations and the mTRE, so these results have not been presented graphically. It is worth noting that the mTRE was higher

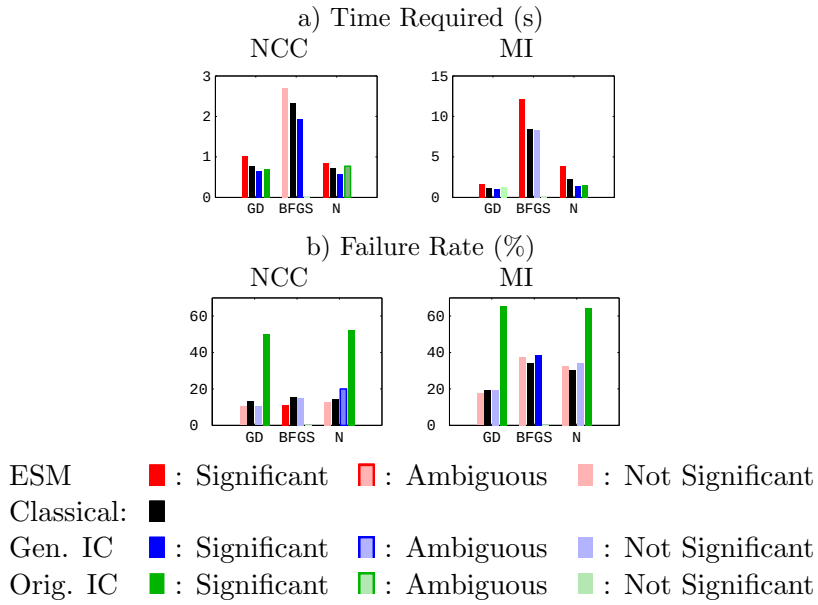



**Figure 7.8** Experimental results for registrations of real homographies using NCC (left) and MI (right). The optimization algorithms are Gradient Descent (GD), Broyden-Fletcher-Goldfarb-Shanno (BFGS) and Newton-Raphson (N). For each algorithm, bars are shown for the following methods from left to right: 1: ESM (Red) 2: Classical Forwards Additive (Black) 3: Generalized IC (Blue) and 4: Original IC (Green). Dark bars are statistically significantly different from classical, light bars are not significantly different and outline bars are ambiguous (see Section 3.2.1). No original IC method was implemented for the BFGS optimizer.

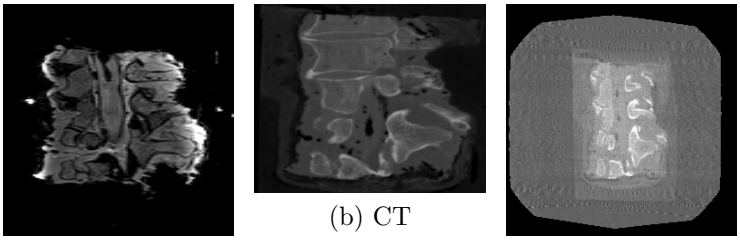
than in the synthetic examples in all cases, averaging about 2 pixels. This is due to the inexact nature of the ground truth standard, lever effects due to large areas (*i.e.,* sky) without overlap, and residual lens distortions in the images.

Once again, there is a clear and statistically significant ranking in terms of speed, with the generalized ESM approach being the slowest, followed by the classic forward additive approach, with the generalized IC approach being the fastest in all cases except the BFGS optimizer for MI. The original IC method does not perform as well in terms of speed on these images, and in the case of Newton-Raphson optimization of NCC is actually slower than the classical method (see Figure 7.8 a)). This slow performance is most likely related to its even poorer performance in terms of reliability.

The generalized IC algorithm performed as well in terms of failure rate as the clas-

sical, forward additive, algorithm, with only the Newton-Raphson, NCC case showing an ambiguous difference in Figure 7.8 b). However, the original IC method proved very unreliable on these images, showing a striking increase in failure rate compared to all other algorithms. It is the author's opinion that this was due to residual lens distortions invalidating the compositional update of the original algorithm. As shown in Figure 7.1, the compositional update traces out a curved path in the moving image parameter space. The validity of this path is determined by the validity of the transformation model. Converting the derivative, on the other hand, only relies on the validity of the tangent space local to the current point on the manifold of transformations. These results indicate that this is a more robust approach.

### 7.6.4 Experiment Set 4: Real Rigid 3D Transformations



(a) MR (b) CT (c) 3DRX

**Figure 7.9** Slices through MR, CT and 3DRX images of cadaver vertebrae provided for registration testing in [190]

In [190], several 3D volumes in different modalities intended for use in testing of registration algorithms are described. These have been registered independently and this provides a silver standard for registration. The datasets consist of volumetric images of cadaver vertebrae obtained with magnetic resonance (MR), computed tomography (CT), and 3D X-Ray (3DRX) imaging techniques. Two triplets of registered images are provided. Figure 7.9 shows slices through the center of the volumes for one of the sets of images, the other is similar. Mask images are also provided by [190] to remove the extraneous surrounding area from the registration.

The original images were provided at resolutions of $1 \times 0.75 \times 0.75$mm for MR, $0.3 \times 0.49 \times 0.3$mm for CT and $1 \times 1 \times 1$mm for the 3DRX dataset. To make the dataset

sizes and resolutions comparable, the CT data was resampled to a $0.75 \times 0.75 \times 0.75$mm grid.

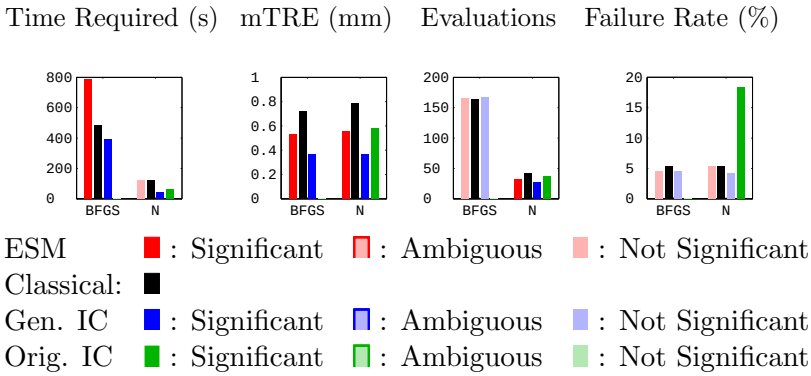



**Figure 7.10** Experimental results for real 3D rigid registrations with mutual information. The optimization algorithms are Broyden-Fletcher-Goldfarb-Shanno (BFGS) and Newton-Raphson (N). For each algorithm, bars are shown for the following methods from left to right: 1: ESM (Red) 2: Classical Forwards Additive (Black) 3: Generalized IC (Blue) and 4: Original IC (Green). Dark bars are statistically significantly different from classical, light bars are not significantly different and outline bars are ambiguous (see Section 3.2.1). No original IC method was implemented for the BFGS optimizer.

As the images are of greatly different modalities, MI is the only appropriate measure to use for registration. As in experiment 2, gradient descent was considered too slow a method to use on these large datasets, so only the BFGS and Newton-Raphson methods were used. The images were registered from starting positions ranging from 4 to 9 mm in mTRE from the true registration solution. Two multiresolution levels were used in the registration, and in this case 100% of the available pixels were used.

The combined results for all 6 image pairs are presented in Figure 7.10. Like the previous experiment, there are some variations in image size and character, and the combined results should be interpreted with that in mind. The timing results are consistent with the previous experiments, with the generalized IC method being fastest. However, in this case the ESM method proves to be much slower than the other methods. Upon examination of the data, it was determined that this is occurring because the optimizer has a tendency to move from the first to the second multiresolution level too early.

In this particular case, the inverse compositional and ESM methods both show a statistically significant improvement in accuracy over the classical, forward additive approach. However, as this was not the case for other datasets, it is the author's opinion that this is not a general trend, and is most likely due to differences in the noise levels in the gradients of this particular dataset. Finally, as in the case of the real homography experiment, the failure rate of the original inverse compositional method is significantly higher on this data.

### 7.6.5 Discussion

Overall, the results of each separate experiment are consistent. In each case, the generalized inverse compositional method runs faster than the classic, forward additive method, and the generalized ESM method runs a bit slower. No significant drop in the number of evaluations required when using the generalized ESM was observed. This is most likely related to the curvature of the path through the image domain, as discussed in Section 7.3. However, an increase in reliability over the other algorithms when starting far from the correct transform (see Figure 7.4) was observed.

In all cases, the generalized IC method is equivalent or better than the original IC method in terms of both speed and reliability. Its major advantage, is the ease with which new optimization methods, or existing optimization libraries can be integrated with it. There may be a slight decrease in reliability far from the correct transform when using either the generalized or the original IC method, which is consistent with results found by other authors (*e.g.,* [71]) for the original IC method. However, in the real homography case, the original IC method proved very unreliable. It is the author's opinion that this is due to the transforms in question not truly being homographies due to residual lens distortion.

Table 7.2 summarizes the overall speed up obtained for each experiment, and for all experiments combined. Overall, the generalized method obtains speed improvements quite near the maximum possible. The benefit is most pronounced for the MSD metric, and somewhat less pronounced for the NCC metric – which is consistent with the relative proportion of time (Table 7.1) spent computing the derivative in each.

The time improvements for the MI measure are also significant. For the Newton-Raphson method they are particularly large, which is due to the MI Hessian requiring

| Measure | | MSD | NCC | MI |
|---|---|---|---|---|
| 1a: Syn. Hom. | GD | 29.6% | 27.1% | 14.2% |
| 1a: Syn. Hom. | BFGS | 24% | 21.5% | 0.8% |
| 1a: Syn. Hom. | N | 29.3% | 30% | 45.3% |
| 1b: Syn. Hom. | GD | 29.7% | 31.4% | 31.5% |
| 1b: Syn. Hom. | BFGS | 23.6% | 23.6% | -2.1% |
| 1b: Syn. Hom. | N | 30.6% | 37.1% | 53.3% |
| 2: Syn. R. 3D | BFGS | 52% | – | 33.3% |
| 2: Syn. R. 3D | N | 17.5% | – | 21.2% |
| 3: Homog. | GD | – | 16.1% | 10.7% |
| 3: Homog. | BFGS | – | 17.4% | 1.5% |
| 3: Homog. | N | – | 19.2% | 39.5% |
| 4: Rigid 3D | BFGS | – | – | 19.5% |
| 4: Rigid 3D | N | – | – | 63.6% |
| Total: | GD | 29.6% | 24.9% | 18.8% |
| Total: | BFGS | 33.2% | 20.8% | 8.8% |
| Total: | N | 25.8% | 28.8% | 44.6% |
| Total: | ALL | 29.5% | 24.8% | 24.9% |

**Table 7.2** Overall speedup found when using the generalized IC method. These show the percentage speedup ($\frac{Time_{STD}-Time_{IC}}{Time_{STD}} \times 100\%$) when compared to the classical (forward additive) method. The totals are averages of these percentages.

more computation than the MSD or NCC Hessians. For the gradient based optimizers, however, the speed up is not as near the theoretical maximum as for MSD and NCC. This is due to three reasons. One being the different balance of computational burden – MI spends less time proportionally computing the gradient than the other two methods and thus has less to gain from the inverse compositional approach. The second reason is that the asymmetric implementation requires some extra computation to compute $\nabla_{\boldsymbol{\phi}_f} D_{MI}$. Finally, the BFGS optimizer had some difficulty optimizing the IC MI due, again, to the asymmetry. An extension of the symmetric MILK (Mutual Information Lucas-Kanade) implementation of [71] would likely provide some further efficiency gains.

From the results, it seems clear that the advantages of the generalized ESM method in this context, are not its speed but its greater reliability. Table 7.3 shows the change in failure rate for each experiment, and the overall total. The results for experiment 1b, which was designed to be particularly difficult, show significantly more reliable

| Measure | | MSD | NCC | MI |
|---|---|---|---|---|
| 1a: Syn. Hom. | GD | 0.1% (11.1%) | -0.2% (-11.1%) | -0.3% (-3.9%) |
| 1a: Syn. Hom. | BFGS | 0.0% (0.0%) | 0.0% (0.0%) | -0.2% (-100.0%) |
| 1a: Syn. Hom. | N | -0.2% (-60.0%) | -0.1% (-20.0%) | 0.5% (200.0%) |
| 1b: Syn. Hom. | GD | -1.0% (-14.3%) | -5.4% (-36.0%) | -5.6% (-8.0%) |
| 1b: Syn. Hom. | BFGS | -1.6% (-25.8%) | -4.2% (-42.4%) | -12.1% (-45.3%) |
| 1b: Syn. Hom. | N | -1.8% (-15.4%) | -2.6% (-15.3%) | -5.0% (-14.1%) |
| 2: Syn. R. 3D | BFGS | 1.3% (100.0%) | – (–) | 0.0% (0.0%) |
| 2: Syn. R. 3D | N | 0.0% (0.0%) | – (–) | 0.0% (0.0%) |
| 3: Homog. | GD | – (–) | -2.8% (-21.4%) | -1.9% (-9.7%) |
| 3: Homog. | BFGS | – (–) | -4.4% (-28.6%) | 3.1% (9.1%) |
| 3: Homog. | N | – (–) | -1.6% (-10.9%) | 2.5% (8.2%) |
| 4: Rigid 3D | BFGS | – (–) | – (–) | -0.8% (-15.4%) |
| 4: Rigid 3D | N | – (–) | – (–) | 0.0% (0.0%) |
| Total: | GD | -0.5% (-13.2%) | -2.8% (-28.3%) | -2.6% (-8.0%) |
| Total: | BFGS | -0.1% (-4.0%) | -2.9% (-34.5%) | -1.6% (-14.4%) |
| Total: | N | -0.7% (-17.5%) | -1.4% (-13.2%) | -0.4% (-2.8%) |
| Total: | ALL | -0.4% (-11.8%) | -2.3% (-24.0%) | -1.3% (-7.6%) |

**Table 7.3** Change in failure rates found when using the generalized ESM method. These show the change in failure rate percentage ($Fail_{ESM} - Fail_{STD}$) when compared to the classical (forward additive) method. The percentage change of the failure rate ($\frac{Fail_{ESM} - Fail_{STD}}{Fail_{STD}} \times 100\%$) is shown in parentheses. The totals are computed by first averaging the failure rates giving each experiment equal weight, and then computing the change.

results when using ESM. The improvement is somewhat hidden in the overall results (bottom rows, Table 7.3) because most experiments were feasible for all the algorithms under test, and thus the greater reliability of generalized ESM never came into play.

## 7.7 Conclusions

In this chapter, generalizations of the popular inverse compositional (IC) and efficient second order methods (ESM) for image registration have been presented. These generalizations are based on the insight that these methods can be viewed as different ways of computing the derivative and Hessian of the image difference measure. The inverse compositional method provided us with an efficient way to compute $\nabla_{\boldsymbol{\phi}_f} D$ and $\mathcal{H}_{\boldsymbol{\phi}_f} D$ when what standard optimizers need is $\nabla_{\boldsymbol{\phi}_m} D$ and $\mathcal{H}_{\boldsymbol{\phi}_m} D$. ESM, on the

other hand, computes more accurate estimates of $\nabla_{\boldsymbol{\phi}} D$ and $\mathcal{H}_{\boldsymbol{\phi}} D$ by using both those with respect to $\boldsymbol{\phi}_f$ and those with respect to $\boldsymbol{\phi}_m$. Both methods previously placed significant restrictions on how the rest of the image registration problem had to be formulated. The generalization shows that the chain rule can be used to convert the derivatives from one variable to another. The advantages of these methods can then be obtained in more general image registration contexts.

These approaches have been explored using three popular image similarity measures, the Mean Squared Difference (MSD), Normalized Cross-Correlation (NCC), and Mutual Information (MI), and three typical optimizers, gradient descent (GD), quasi-Newton (BFGS) and Newton-Raphson (N) methods. The IC method can achieve a maximal speed up of about 1/3 and the generalized method achieves nearly that for the MSD and NCC image difference measures on these optimizers. The speedup is less, but still significant, for the MI measure. This measure was more difficult mainly due to its asymmetric formulation.

It was found that the generalized ESM method does not improve results in terms of speed, but does improve the reliability of the optimization. This can be understood intuitively as the ESM method combines estimates of the optimization step from both the inverse and forward approaches. This estimate is, particularly in difficult cases, a better estimate on average than either estimate alone.

A further advantage of the generalization presented in this thesis, is that the various approaches can be easily combined, which is something of interest to address in future work. It is probable that using a few of the relatively expensive ESM derivative calculations early in the optimization process would improve the optimizers' capture radius, while using IC derivative calculations later in the optimization process would improve the computational efficiency. In this way the generalization can allow one to get the best of both worlds.

# Chapter 8

# Conclusions and Future Directions

In this thesis I have examined methods for efficient and reliable direct, parameterized image registration. Image registration is a key component of a wide variety of applications. For example, this thesis has built directly on image registration work that was originally applied to tracking (*e.g.,* [12, 66]), medical data fusion (*e.g.,* [75, 164]), and visual servoing [22, 139, 140]. Direct registration techniques also form important parts of applications in remote sensing (*e.g.,* [55, 204]), superresolution (*e.g.,* [98, 111]) and video coding (*e.g.,* [72]). While all applications can benefit greater efficiency and reliability, certain time sensitive applications, such as intraoperative image guidance (*e.g.,* [5, 54, 160]) can particularly benefit from fast and reliable approaches.

Recall that direct parameterized image registration works by defining an image difference measure between two images to be registered. The mapping between the spaces of the two images is expressed as a parameterized function and one image is warped to match the other. Registration can then be expressed as an optimization problem, that of finding the warp parameters that minimize the difference between the two images. The main computational burden in solving this problem lies in the calculation of the image difference measure, $D$. To achieve faster registration, either $D$ has to be evaluated fewer times, or it has to be evaluated more quickly, or both. To obtain equivalent results with fewer evaluations of the image difference measure requires selecting the optimization methods most appropriate to the problem. Quicker evaluation can be achieved either by using only some of the image data, or by precalculating and caching parts of the measure. In this thesis, I have examined all of these

possible approaches. Throughout, I have taken the philosophy of being as general as possible rather than focusing on a specific application. Each problem has been approached from a theoretical perspective first, but these theoretical developments have been backed up in each case by extensive experiments on a wide range of both 2D and 3D images. As a result, the conclusions of this thesis are applicable to a wide range of problems in both computer vision and medical imaging.

## 8.1 Discussion

In Chapter 4 I examined the choice and tuning of optimization algorithms for the image registration problem. To perform a valid comparison between algorithms, it was first necessary to address two issues: (1) the calculation of approximate Hessians for image difference measures, and (2) the scaling of the parameters of the transformation function. Although their importance was known, previous work had left these issues without well-analyzed solutions.

For image registration problems, the true Hessian is impractical to calculate. However, for least-squares problems (*i.e.,* for the mean squared image difference measure) the Gauss-Newton approximation has proved very useful [12, 136]. I have shown in Section 4.3 that the Gauss-Newton Hessian approximation idea can be extended to non least squares cost functions, specifically to normalized correlation and mutual information. For mutual information (MI) in particular this new approximation truncates the Taylor series in a different place than previous work [70, 185, 186]. I have shown theoretically and experimentally that the approximation I have presented for the MI Hessian is significantly more accurate.

The scaling of the transformation parameters is known to have a significant effect on the performance of optimization algorithms. However, previous examinations of optimization in registration had treated this issue in a problem specific way. I have examined the issue of parameter scaling and presented a method in Section 4.5 for automatically determining an optimal set of scale factors for the parameters of any registration problem. Experiments show that using these scaling factors gives superior optimization performance on image registration problems using matrix based transforms. For deformable transforms, the experimental results were more subtle. I found

that using a set of scale factors with a somewhat compressed range provided better performance on these problems. I am unable to conclusively state why this occurred, but I conjecture that it is related to the fact that these problems were approached using implicit regularization only. With the scale factors I had originally proposed on this problem the optimization tended to explore very unlikely sets of parameter values. It is my opinion that applying an explicit regularization technique is probably the best way to address this issue.

With scaling and the Hessian properly calculated, it was possible to make a meaningful comparison of the performance of different optimization algorithms. While studies that examine optimizer performance on a particular image registration problem are useful, it was my intention to determine guidelines for optimizer selection and use that could be generalized to a wide range of image registration problems in both computer vision and medical imaging. The analysis in Section 4.6.1 showed that the performance of optimization algorithms depends greatly on the number of parameters in the transformation used.

I examined the performance of five algorithms – the downhill simplex, Powell's, gradient descent, BFGS and Newton-Raphson algorithms – on synthetic and realistic image registration problems ranging from 3 to 1029 parameters. The results immediately make clear that the downhill simplex and Powell's algorithms are not an efficient choice for any but the smallest of image registration problems. This is true despite the fact that they require evaluation of the cost function only, which is cheaper than the gradient and Hessian evaluations required by other methods. The relative performance of the remaining methods – gradient descent, BFGS and Newton-Raphson – is highly dependent on the specific characteristics of the image registration problem.

The theoretical analysis, and experiments on a simple problem implied that the BFGS quasi-Newton method should require fewer iterations than the gradient descent method, particularly as the number of parameters increased. However, the experiments on image registration problems gave opposite results. For several transformations with both large and small numbers of parameters, the BFGS method required significantly more function evaluations, and therefore more time, than gradient descent although it does yield a slight improvement in precision. It also showed somewhat poorer performance optimizing mutual information. It is my opinion that the

marginal improvement in precision on the high-dimensional deformable transforms is not worth the significant additional computational burden that this method required. I suspect that the poorer performance of the BFGS algorithm is due to violations of its basic assumptions inherent in the image registration problem, particularly the quasiconvexity of the cost function.

The Newton-Raphson algorithm, on the other hand, behaved as expected based on a theoretical analysis. It required fewer function evaluations than either of the other methods (although this effect is less pronounced for MI). However, these evaluations are expensive. Thus it is most appropriate for use on transformations such as the thin plate spline (TPS), which have very expensive gradient calculations, and it gives superior results for a middle range of parameter sizes.

As a brief overall guideline, for both small ($\leq 6$) and large ($\geq 150$) parameter problems, the gradient descent method is recommended. In this middle range of numbers of parameters, if the gradient has complexity $O(N)$ to compute, then the BFGS method is appropriate, while if the gradient complexity is $O(N^2)$, then the Newton-Raphson method is likely superior. For mutual information which has an indefinite, and computationally expensive Hessian, the gradient descent or BFGS methods are probably preferable to Newton-Raphson.

The work of Chapter 4 addressed mainly the types of efficiency gains which involve reducing the evaluation of $D$. Chapter 5 addresses speeding up the evaluation of $D$ by using only some of the pixels. I examined the question of *how many* pixels should be used to compute the cost function. While the idea of using only some of the pixels is an old one, this was the first time a formal deliberation control framework has been applied to this problem. Deliberation control methods require algorithms which can be used for partial evaluation – such as the anytime algorithms – and a model of how the quality of the results varies as the level of computation is changed. Image difference measures are formulated as a calculation looping over all the pixels, so they lend themselves easily to formulation as anytime algorithms. The magnitude of the image difference measure gradient was used as a feedback parameter, and a dynamic performance profile was used to model the quality versus calculation level characteristics of these functions. This approach has the distinct advantage of adapting the computation level at every evaluation of the cost function. The obvious competing

technique to this adaptive approach is simply to use some constant percentage of the pixels. The comparative experiments in Section 5.3.3 show that adapting the amount of computation at each evaluation of the cost function allows the anytime approach to make performance gains where possible, without the loss of reliability which occurs with an arbitrary cutoff on computation. Because the performance profiles are learned for various classes of images, the technique can also adapt to the properties of the images being examined. For example, the registration of T1-weighted and T2-weighted magnetic resonance images is likely to require rather different numbers of pixels than the registration of computed tomography and ultrasound images.

It has been stated by a number of authors (*e.g.,* [124, 181]) that using pixels of high derivative is more effective for image registration. This idea was formalized in the framework of [66] where pixels are chosen based on a greedy approach to reducing the size of the estimated covariance matrix of the parameters. However, some performance degradation had been reported, without being examined in detail. In Chapter 6, I examined the reasons for this performance degradation. The optimization techniques used in image registration rely on the Taylor series of the image difference function being a valid approximation. The range over which this approximation is valid is dependent on the scale of the image derivatives used to compute it, an issue that had previously been ignored in the literature on pixel selection. It is shown that when pixels are selected using derivative based methods the capture range of the optimization is proportional to the scale of the derivative used to select the pixels. I also examined the pixel selection criterion of [66] and developed both a more formally valid criterion which approximates more closely the mutual information between observation and parameter, and approximate criteria that are faster to compute.

The inverse compositional (IC) algorithm [12] and the efficient second order (ESM) algorithm [22, 139, 140] are efficient image registration methods which make use of the special symmetry inherent in the registration problem. Specifically, there is an equivalence between warping the moving image one way, and warping the fixed image in an equal and opposite way. The inverse compositional method computes an update step to the warp for the fixed image, and composes the inverse of that warp with the current moving image warp. This achieves fast registration because the image derivative and the difference measure Hessian can be precomputed and cached.

However, all previous implementations have required the optimization algorithm to makes steps compositionally in the parameter space. This complicates the implementation, because most standard optimization algorithms are designed to use additive update steps. The ESM algorithm computes updates to both this fixed image warp and the moving image warp and combines them in a final update step. Current implementations were restricted because it was necessary to be able to add together the fixed and moving warp parameters, something that was only possible for specific parameterizations based on the exponential map of the Lie group of transformations.

I have shown in Chapter 7 that so long as the warp has differentiable composition and inversion operations – a restriction that was already placed on the warps by the use of these methods – then we can view the moving image warp as a continuous, differentiable function of the fixed image warp. As a result, the derivative with respect to the fixed image warp can be converted to be with respect to the fixed image warp by a simple application of the chain rule for derivatives. By converting the derivative in this way, both algorithms can be generalized so that the inverse compositional algorithm no longer requires a compositional step, and the ESM algorithm no longer requires special parameterizations. Experimental results show that the generalized inverse compositional algorithm performs at least as well as the original, and in certain cases performs much better. The generalized ESM algorithm does not provide a speed gain, but does provide a gain in reliability. This occurs because using the generalized ESM approach has the effect of widening the capture radius of the cost function.

Each of the approaches I have developed can be applied, alone or in combination, to design efficient and reliable direct image registration solutions. As I have avoided application specific approaches, the efficiency gains of these methods should be applicable to any of a wide range of applications.

## 8.2 Summary of Contributions

In summary, in this thesis I have presented a number of specific contributions that can be directly applied to a wide range of practical image registration problems.

- For the use of a Newton-Raphson type iteration, a Hessian, or approximate Hessian is required. I have shown how the Gauss-Newton approximation to the

Hessian may be generalized to non-least squares cost functions. Experimental results indicate that this approximation is effective, and in particular, that this approximation of the MI Hessian is more effective than previous approaches.

- The scaling of the parameters of the optimization problem is very important for optimizer performance, but previous methods for determining the scaling factors have been ad-hoc. I have presented a method for computing optimal scaling factors from the geometric configuration of the problem alone. Experimental results show that using these scaling factors greatly enhances optimizer performance on matrix based transforms.

- Different optimization algorithms are appropriate for different registration problems. I have analyzed optimizer performance both theoretically and experimentally, and have made recommendations that can be generalized to a wide range of different optimization problems. It is my hope that these recommendations will help future designers make principled, rather than ad-hoc, choices of optimization algorithms for registration problems.

- It is clear that image registration can be made more efficient by using only some of the pixels to calculate the image difference measure is clear. However, previous work had not addressed the tradeoff between the speed gains of using this approach, and the potential loss of accuracy. In this thesis, I have presented the first approach that uses a formal deliberation control framework to manage the computation level at each evaluation of the cost function. Experiments have shown that this principled, adaptive approach can achieve speed gains without degrading accuracy and reliability.

- Many authors have recommended the heuristic of using pixels of high derivative for image registration, and more formal schemes have been proposed (*e.g.,* by Dellaert and Collins [66]). However, this previous work has ignored the important issue of scale. I have shown that the scale of the derivative used to select pixels has an important effect on the capture radius of the image difference measure. The capture radius will be roughly equivalent to the scale of the derivative used to select pixels, thus the scale should be chosen based on the uncertainty

in the transformation parameters.

- The IC and ESM efficient alignment methods relied on special characteristics of the image registration problem to achieve efficiency, but they imposed restrictions on the optimization and parameterization used. I have shown that each method can be viewed as a way to calculate the derivative with respect to a new set of parameters. As these new parameters are continuous, differentiable functions of the old, it is possible to transform them using the chain rule, removing the need for the restrictions previously imposed. The experiments show that generalized methods based on this transformation of the derivative perform well, and in fact the generalized inverse compositional method outperforms the original in terms of reliability in a number of cases.

## 8.3 Future Directions

In Chapter 4 I compared optimization algorithms and made recommendations regarding which algorithm would perform in a superior way on different problems. However, there are a vast number of optimization approaches and it was impossible to address all aspects of this problem. I will now discuss a number of interesting open problems for further research.

The results of my research suggest that trust-region approaches may have a speed advantage while line-search approaches may have a reliability advantage, but more research would be needed to definitively answer this question. It is also my opinion that trust-region quasi-Newton algorithms could perform very well on the image registration problem, particularly as they are reputed to be more robust to indefinite Hessians [151]. Finally, in the context of this research, there was not sufficient time to explore the use of stochastic optimization algorithms. Klein *et al.* [120] found a stochastic algorithm, the Robbins-Monro method, to be the most efficient approach of the ones they tested. There are stochastic analogs for all the algorithms tested here and the work of [1], in the VLSI domain, suggests that their stochastic nature could be integrated with a deliberation control approach.

Beyond the simple selection of the algorithm, the results of Chapter 4 suggest that combinations of optimization techniques may prove most effective. For instance,

a robust, but slow algorithm may be appropriate at the top level of a multiresolution pyramid, while less robust, but faster algorithms could be applied lower down in the pyramid. Alternatively, the results of the experiments of Section 4.6 suggest that it may be more effective to apply gradient descent methods early in a deformable registration for speed, and some BFGS iterations later for accuracy.

Chapter 5 showed that my specific implementation of a deliberation control framework yielded efficiency gains on image registration problems without losing reliability. However, in general what is needed for deliberation control is algorithms that support partial evaluation, and a model of how their accuracy changes with the amount of computation. While the model I used here – that of dynamic performance profiles – provided interesting results, it is not the only model possible. It would be interesting to explore other ways of modeling the relationship between computation and accuracy in the image registration context.

The insight presented in Chapter 6, that the effectiveness of a particular set of pixels for image registration purposes is dependent on the scale at which they are chosen, explains a number of observations about the image registration process. However, I did not use this insight to develop an algorithm for fast registration. A method has recently been proposed [23] which selects pixels based on how well they adhere to a particular Taylor series model of the cost function around the point of registration, but the method does not explicitly deal with the matter of scale. It would be interesting to explore the effect of the scale of pixel selection in the context of this new pixel selection method.

Finally the generalization of the inverse compositional and ESM methods presented here are developed under the assumption that the transformations in question form a group. The inverse compositional method has been shown to work in cases where the transformations only *approximately* form a group [143]. It would be interesting to explore the extension of my generalized approach to cases like this. It could prove particularly useful for warps such as the thin plate spline (TPS). Recall from Section 4.6.1 that the calculation of the Jacobian of the TPS is time consuming, requiring $O(N^2)$ operations. A generalized inverse compositional approach to the TPS would allow the Jacobian matrix to be precomputed. This would make warps with very large numbers of parameters with the thin plate spline feasible.

While each of the components of this thesis presents interesting opportunities for further research, the components presented here are also ready to be used. I carried out this research on test image registration problems that were quite well defined and bounded, which was critical in order to get objective, statistically testable results. However, now that the methods proposed here have been tested in a controlled context, it will be interesting to exploit this knowledge in practical image registration problems.

It was problems in image guided neurosurgery that originally motivated my interest in this domain. These problems are characterized by using multimodal imagery and requiring both rigid and curved transformations. Naturally, this application requires methods that are both efficient and reliable. I believe that some of the most interesting future challenges will lie in applying theory to engineering solutions that directly benefit society. I list below some related work which represents a beginning of the process of connecting this research with real clinical practice.

### Related work

The work in the following papers does not directly appear in this thesis, but is closely related to this research.

[40] Rupert Brooks, D. Louis Collins, Xavier Morandi, and Tal Arbel. Deformable ultrasound registration without reconstruction. In *Proceedings of the 11th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI'08)*, pages 1023–1031, New York, USA. 2008.

[6] Michel Audette, Rupert Brooks, Robert Funnell, Gero Strauss, and Tal Arbel. Piecewise affine initialized spline-based patient-specific registration of a high-resolution ear model for surgical guidance. In *MICCAI Workshop on Image Guidance and Computer Assistance for Soft-Tissue Interventions*, New York, USA. 2008.

# Appendix A

# Mathematical Glossary

### Symbols

Following the usual standard, multidimensional objects (vectors, matrices and tensors) will be shown in bold font, *i.e.,* $\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi})$. The following table summarizes the most commonly used mathematical symbols in this thesis.

| | |
|---|---|
| $\circ$ | The composition operator |
| $\sharp(\boldsymbol{x})$ | The dimension of the vector, $x$ |
| $\lfloor x \rfloor$ | The floor operator; the largest integer less than $x$ |
| $\lceil x \rceil$ | The ceiling operator; the smallest integer greater than $x$ |
| $\beta_0, \beta_1, ...$ | B-splines of order 0,1,... |
| $D$ | The image difference measure; cost function for optimization |
| $\boldsymbol{\phi}$ | The parameters of the image warping function |
| $\mathcal{H}_{\boldsymbol{x}} f(\boldsymbol{x})$ | The Hessian matrix of second derivatives with respect to $\boldsymbol{x}$ of the function $f(\boldsymbol{x})$ |
| $\boldsymbol{I}_f$ | The fixed image as a set of pixels. |
| $I_f(\boldsymbol{x})$ | The fixed image as a scalar function of $\boldsymbol{x} \in \mathbb{R}^d$ |
| $\boldsymbol{I}_m$ | The moving image as a set of pixels. |
| $I_m(\boldsymbol{x})$ | The moving image as a scalar function of $\boldsymbol{x} \in \mathbb{R}^d$ |
| $\boldsymbol{J}$ | A Jacobian image. Considering an image warped by some parameters, $I(W(x, \boldsymbol{\phi}))$, then $\boldsymbol{J} = \frac{\partial \boldsymbol{I}}{\partial \boldsymbol{W}} \cdot \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{\phi}}$ |
| $\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$ | The gradient with respect to $\boldsymbol{x}$ of the function $f(\boldsymbol{x})$ |

| | |
|---|---|
| $\mathbb{R}$ | The real numbers |
| $\boldsymbol{W}(\boldsymbol{x}, \boldsymbol{\phi})$ | The transformation, or warp. It maps the point $x$ to a new position, and is controlled by the warp parameters, $\boldsymbol{\phi}$ |
| $\boldsymbol{x}$ | A point in space. |
| $\boldsymbol{X}$ | A set of points in space. |

### Subscripts

| | |
|---|---|
| $\boldsymbol{X}_{(n)}$ | $n$ is an iteration counter. Iteration counters are shown in parentheses. |
| $\boldsymbol{X}_i$ | $i$ is an index to an element of the object, $X$ |

### Images

An image is a scalar function defined over a space, $I(\boldsymbol{x})$; $x \in \mathbb{R}^d$. An image can also be considered as a set of pixels, in which case this function is evaluated at (usually a regular lattice of) points in the space. To represent this set of points, an upper case $\boldsymbol{X}$ will be used. So $I(\boldsymbol{X})$ is the scalar image function evaluated at the set of points, $\boldsymbol{X}$. For brevity, this may be written $\boldsymbol{I}$ where being bold indicates that this is a multidimensional object, rather than a scalar function. Indexing a particular pixel can then be written either $\boldsymbol{I}_i$ or $I(\boldsymbol{X}_i)$.

### Derivatives

If $f(\boldsymbol{x})$ is a scalar function of a vector argument, then $\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}$ has $\sharp(\boldsymbol{x})$ elements. The following convention is used:

$$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}$$

is a row vector, while

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$$

is a column vector. So $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \left[\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right]^T$. The convention can be extended to vector functions of vector variables, so:

$$\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x} = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_1(\boldsymbol{x})}{\partial x_2} & \cdots \\ \frac{\partial f_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_2(\boldsymbol{x})}{\partial x_2} & \cdots \\ \vdots & \vdots & \vdots \end{bmatrix}$$

# Appendix B

# Kernel Splines

The material in this Appendix is based on the work previously published as:

[36] Rupert Brooks and Tal Arbel. Improvements to the itk::KernelTransform and subclasses. *Insight Journal*, March 2007. DSpace handle `http://hdl.handle.net/1926/494`.

Deformable transformations based on a small set of matched points are extremely useful in medical imaging. While the deformation at each point is explicitly defined by the correspondence, the deformation between points must be interpolated. There are a family of spline based approaches which interpolate a smooth deformation field, of which the most widely known is the thin plate spline [31, 94]. In the ITK, this family of transformations is referred to as *kernel* transformations, because they can be expressed as linear combinations of radially symmetric kernel functions centered on each point.

In order to use the kernel splines in the ITK registration framework certain modifications to the existing implementation were required. This appendix describes the implementation which was also published in [36].

## B.1 Kernel Transforms

We begin with two sets of $n$ corresponding landmark points, in a space with *dim* dimensions. We will refer to these as the source landmarks, $\boldsymbol{P}_S = \{\mathbf{p}_{S_i}\}$, and the

target landmarks, $\boldsymbol{P}_T = \{\mathbf{p}_{T_i}\}$. A displacement vector, $\mathbf{d}_i = \mathbf{p}_{T_i} - \mathbf{p}_{S_i}$, can be defined at each point, and these vectors can be grouped into one long vector, $\mathbf{D}$, as follows:

$$\mathbf{D} = \begin{bmatrix} \mathbf{d}_1^T & \mathbf{d}_2^T & ... & \mathbf{d}_n^T \end{bmatrix}. \tag{B.1}$$

The kernel function, $\boldsymbol{g}(\mathbf{x})$ maps $dim$-dimensional vectors onto $dim \times dim$ symmetric matrices.

In this model, a deformation field, $\mathbf{F}(\mathbf{x})$, is viewed as a set of $dim$ independent functions of spatial position,$\mathbf{x}$, $\mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), ..., f_{dim}(\mathbf{x})]^T$. Thus there is a separate, independent spline for the displacement in each coordinate. Each such spline is considered to be a combination of a linear (affine) transformation, plus a weighted combination of kernel functions centered at each point.

$$\mathbf{F}(\mathbf{x}) = \boldsymbol{C}^T \cdot \boldsymbol{G}(\mathbf{x}) + \boldsymbol{A} \cdot \mathbf{x} + \mathbf{b} \tag{B.2}$$

where $C^T$ is a $dim \times n$ vector of weights, $\boldsymbol{G}$ is the $dim \times dim \cdot n$ matrix made by stacking the kernel functions centered at each landmark point, evaluated at the point of interest, and $\boldsymbol{A}$ and $\mathbf{b}$ are a linear transformation in the coordinates. Thus the complete spline is defined by $n \cdot dim + dim^2 + dim$ parameters.

For $n$ landmark points the value of the displacement field at each point provides $dim \cdot n$ constraints. There are fewer equations than unknowns so this leaves the system underdetermined. A further constraint is applied by requiring that the deformation should flatten out to an affine transformation far from all the landmarks. This flatness constraint can be developed into the set of linear equations [94]

$$\boldsymbol{P}_S^T \cdot C = 0, \tag{B.3}$$

where $\boldsymbol{P}_S$ is the source landmark coordinates arranged in a matrix, as follows:

$$\boldsymbol{P}_S = \begin{bmatrix} \mathbf{p}_{S_{1_1}} I & \dots & \mathbf{p}_{S_{1_{dim}}} \boldsymbol{I} & \boldsymbol{I} \\ \vdots & \dots & \vdots & \vdots \\ \mathbf{p}_{S_{n_1}} I & \dots & \mathbf{p}_{S_{n_{dim}}} \boldsymbol{I} & \boldsymbol{I} \end{bmatrix} \tag{B.4}$$

This provides enough additional equations to make the system have full rank.

Combining these components in the following way

$$\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{D} \\ \mathbf{0} \end{bmatrix} \tag{B.5}$$

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{K} & \boldsymbol{P}_S \\ \boldsymbol{P}_S^T & \mathbf{0} \end{bmatrix} \tag{B.6}$$

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{C} \\ \mathbf{a}_1^T \\ \vdots \\ \mathbf{a}_{dim}^T \\ \mathbf{b}^T \end{bmatrix} \tag{B.7}$$

$$\tag{B.8}$$

where the $\mathbf{a}_i$ are the rows of $\boldsymbol{A}$, and $\boldsymbol{K}$ is the matrix of kernel functions at each point combination,

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{g}(\mathbf{p}_{S_1} - \mathbf{p}_{S_1}) & \cdots & \boldsymbol{g}(\mathbf{p}_{S_1} - \mathbf{p}_{S_n}) \\ \vdots & \cdots & \vdots \\ \boldsymbol{g}(\mathbf{p}_{S_n} - \mathbf{p}_{S_1}) & \cdots & \boldsymbol{g}(\mathbf{p}_{S_n} - \mathbf{p}_{S_n}) \end{bmatrix} \tag{B.9}$$

it is possible to write a set of linear equations for the parameters of each spline, $\boldsymbol{L} \cdot \boldsymbol{W} = \boldsymbol{Y}$, and solve for the weights [31, 63, 94], $\boldsymbol{W}$, as

$$\boldsymbol{W} = \boldsymbol{L}^{-1} \cdot \boldsymbol{Y} \tag{B.10}$$

The classic spline interpolation can be relaxed to allow for some misfit at the landmark points, by adding a multiple of the identity to the matrix $\boldsymbol{K}$ [177]. That is, $\boldsymbol{L}$ becomes

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{K} + \lambda I & \boldsymbol{P}_S \\ \boldsymbol{P}_S^T & 0 \end{bmatrix} \tag{B.11}$$

## B.2 Problems with the existing implementation

The ITK system contained a kernel transform implementation which was suitable for warping images, but did not integrate well into the ITK registration framework for

two reasons. Primarily, the Jacobian of the transformation was not implemented. Indeed, the code only contained this comment in the Jacobian method:

```
// TODO
// The Jacobian should be computable in terms of the matrices
// used to Transform points...
```

Without a Jacobian, gradient based optimizers cannot be used, and approaches such as Powell's method or Amoeba (Downhill Simplex) are not well suited for transformations with this many parameters.

A more serious problem was that the implementation required that the weights, $\boldsymbol{W}$ needed to be recomputed each time the parameters were set. This is extremely computationally inefficient when the class is used in an optimization framework. This requirement arose because the source landmark positions were considered to be the moving parameters. From the development in section B.1 it is clear that changing the locations of $\boldsymbol{P}_S$ requires recomputing $\boldsymbol{K}$ and therefore $\boldsymbol{L}$ and $\boldsymbol{W}$. Furthermore, there is not a clear way to take the derivative of $\mathbf{F}(\mathbf{x})$ with respect to the source parameter positions, which is most likely why the Jacobian was not completed.

## B.3 Changes

We reversed the role of the fixed and moving parameters, so that the `SetParameters()` method updates the *target* landmarks rather than the source landmarks. In this case, $\boldsymbol{L}^{-1}$ does not change when the moving parameters are updated, so it can be precomputed and cached. On each update of the parameters, $\boldsymbol{Y}$ needs to be updated, and the weights $\boldsymbol{W}$ can be found through a matrix multiplication via Equation B.10. This way, the parameters can be changed repeatedly in the optimization framework at much less computational cost.

Furthermore, with this formulation, the Jacobian of the transformation becomes

easy to calculate.

$$\mathbf{F}(\mathbf{x}) = \left[ \begin{array}{cccc} \boldsymbol{G}(\mathbf{x}) & \mathbf{x}_1\boldsymbol{I} & \ldots & \mathbf{x}_{dim}\boldsymbol{I} \quad \boldsymbol{I} \end{array} \right] \cdot \boldsymbol{W} \tag{B.12}$$

$$\mathbf{F}(\mathbf{x}) = \left[ \begin{array}{cccc} \boldsymbol{G}(\mathbf{x}) & \mathbf{x}_1\boldsymbol{I} & \ldots & \mathbf{x}_{dim}\boldsymbol{I} \quad \boldsymbol{I} \end{array} \right] \cdot \boldsymbol{L}^{-1} \cdot \boldsymbol{Y} \tag{B.13}$$

$$\frac{\partial \mathbf{F}(\mathbf{x})}{\partial \boldsymbol{\phi}} = \left[ \begin{array}{cccc} \boldsymbol{G}(\mathbf{x}) & \mathbf{x}_1\boldsymbol{I} & \ldots & \mathbf{x}_{dim}\boldsymbol{I} \quad \boldsymbol{I} \end{array} \right] \cdot \boldsymbol{L}^{-1} \cdot \frac{\partial \boldsymbol{Y}(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}} \tag{B.14}$$

$$\frac{\partial \mathbf{F}(\mathbf{x})}{\partial \boldsymbol{\phi}} = \left[ \begin{array}{cccc} \boldsymbol{G}(\mathbf{x}) & \mathbf{x}_1\boldsymbol{I} & \ldots & \mathbf{x}_{dim}\boldsymbol{I} \quad \boldsymbol{I} \end{array} \right] \cdot \boldsymbol{L}^{-1} \cdot \left[ \begin{array}{c} \boldsymbol{Z} \\ 0 \end{array} \right] \tag{B.15}$$

where $\boldsymbol{Z}$ is a negative $n \cdot d$ diagonal matrix with all diagonal entries $-1$.

Conceptually, this implementation supports the idea that the positions of the source landmarks (or fixed parameters) are the positions of landmark points in the coordinate system of the fixed image. The position of the target landmarks, (or moving parameters) will then be the corresponding positions of these points in the moving images. (The previous approach corresponded to the idea that the fixed parameters were the positions of landmarks in the target image, and the moving parameters were the locations of these landmarks in the coordinate system of the fixed image.)

# Appendix C

# Generating Meshes for Thin Plate Splines

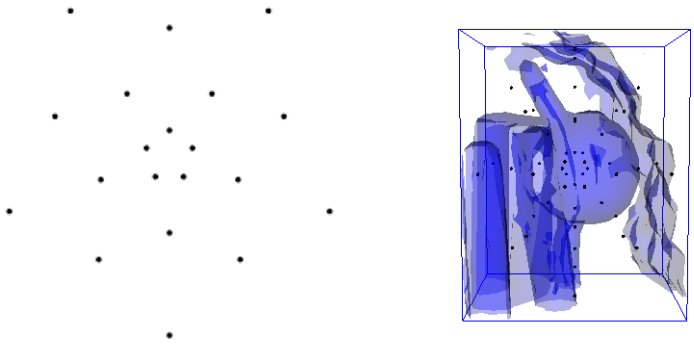
Kernel splines, including thin plate splines, need to be defined based on sets of matched landmarks. Rather than manually matching landmarks, it can be useful to generate sets of points relatively evenly spaced through an image which can be used to define a kernel spline over the image. Two algorithms for generating evenly distributed points were used in the experiments reported in Chapter 4. They give a set of points that cover the space well, and are relatively unlikely to generate non-diffeomorphic transformations.

## C.1 Circular and Cylindrical Meshes

The first algorithm generates a set of points which are laid out as a set of staggered concentric rings in 2D, and as a set of staggered concentric cylinders in 3D. This algorithm was used to generate the landmarks for the thin plate spline transforms used for the registration experiments in Chapter 4. The algorithm is described in Algorithm 9, and images of the point positions are in Figure C.1

**Algorithm 9** Creating a set of staggered circular (2D) or cylindrical (3D) points. For the 2D case, the zlevels are set to 1

**Set** number of rings, $N_r$, points per ring, $N_p$, maximum radius, $R$, zlevels $N_z$, and Z size $Z$
**Set** angular step $\Delta\theta = \frac{2\pi}{N_p}$
**Set** z step $\Delta z = \frac{Z}{N_z - 1}$
**For** $z = 0$ to $N_z - 1$ **Do**
  **For** $i = 1$ to $N_r$ **Do**
    **Set** radius $r = i \cdot \frac{R}{N_r}$, offset $o = \frac{\Delta\theta}{2} \cdot \mod(i + z, 2)$
    **For** $j = 0$ to $N_p - 1$ **Do**
      Add point at $r, \Delta\theta * j + o, z$ where $r$,$\theta$ are in polar coordinates
    **End for**
  **End for**
**End for**



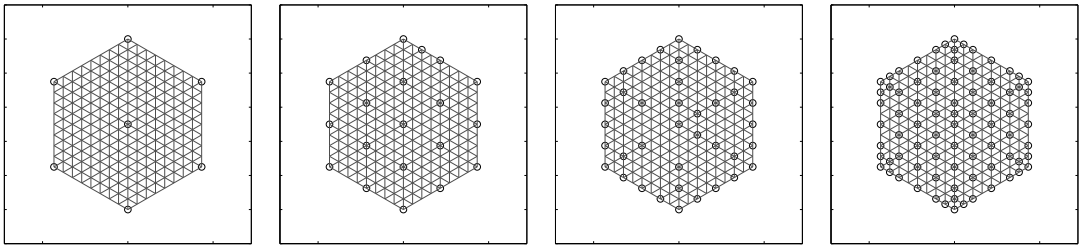
(a) 2D TPS point layout (b) 3D TPS point layout

**Figure C.1** Point layout used for TPS transforms in Chapter 4

## C.2 Subdividing a Triangular Mesh

This method was used to create a mesh where the number of points could be controlled very finely for the simple optimization timing experiments in Section 4.6.2. The intention is to avoid long skinny triangles in the Delaunay triangulation of the point set, as these would be prone to collapsing, resulting in a non-diffeomorphic transform. The idea is to generate a point set where points can be added one by one from a list, and any configuration of these points will be suitable for use as landmarks in a TPS.

Such a mesh can be generated by a recursive algorithm. At each level, $n$, the set of points is all the points at the last level, $n-1$, plus all the midpoints of the edges in their Delaunay triangulation. At level 0, a set of seven points is used, a center point surrounded by a hexagon. If points are added from the set in order, they will tend to densify the space evenly, and, since the Delaunay triangulation is used, long skinny triangles are avoided. Figure C.2 illustrates this idea.



(a) 7 points (b) 20 points (c) 40 points (d) 80 points

**Figure C.2** Different size sets of landmarks generated using the Delaunay based technique. The lines show the 3rd level mesh, which was used as a base for this example. The 7 point case shows the zeroth level. The 20 point case is all the points from level 1 and below, plus one point from level 2. The 40 and 80 point case show how adding the points in order of creation gradually fills in the space without clustering of points.

# Appendix D

# Extra graphs for comparison of deformable registrations

This appendix contains certain graphs which were considered too repetitive to be presented individually in Chapter 4

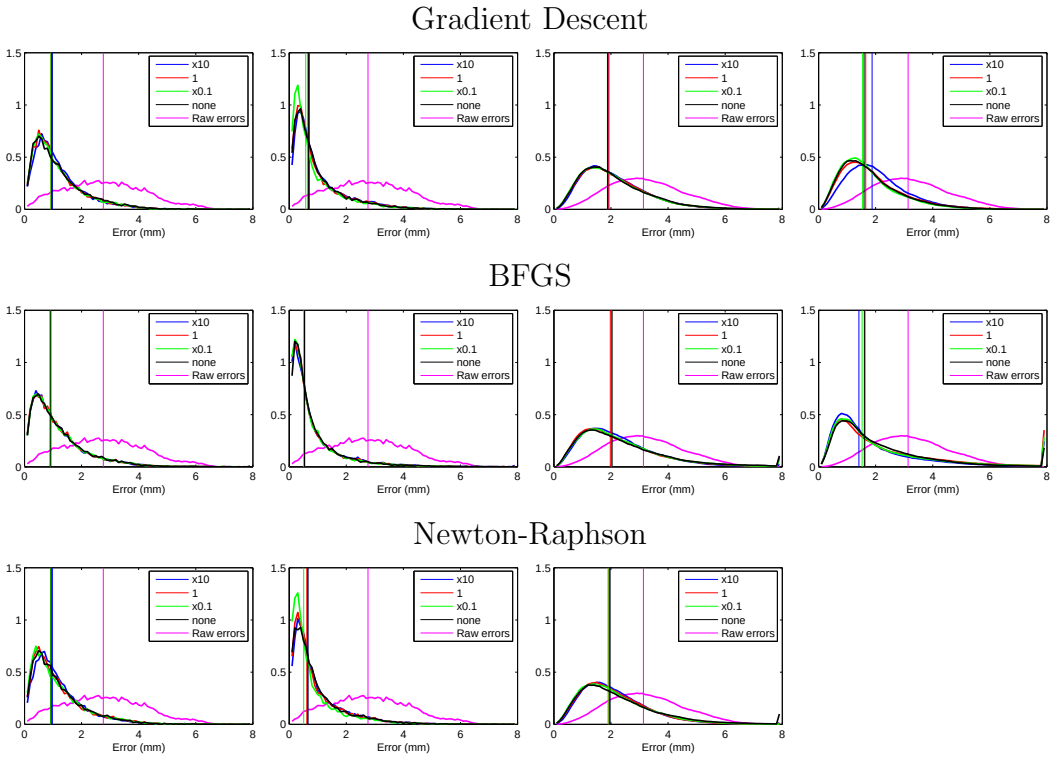



**Figure D.1** Distributions of errors remaining after registration of the deformed Sagittal Brain Image using the 2D B-spline and TPS transforms, and the synthetically deformed Phantom image using the 3D B-spline and TPS transforms. The image difference measure used was mutual information, results for other cost functions are similar. The graphs show the distribution of errors over a $50 \times 50$ grid ($50 \times 50 \times 50$ grid in 3D) of points in the image extent. Point errors for all 25 runs have been combined into one histogram for each set of scale factors. Vertical lines show the median error. With all scale factors, reasonably successful registration has been achieved, as shown by the distinct leftward shift in the distribution. Overall case 0.1x produces a slightly, but statistically significantly smaller median error. The 2D B-spline registration with gradient descent also appears as Figure D.1

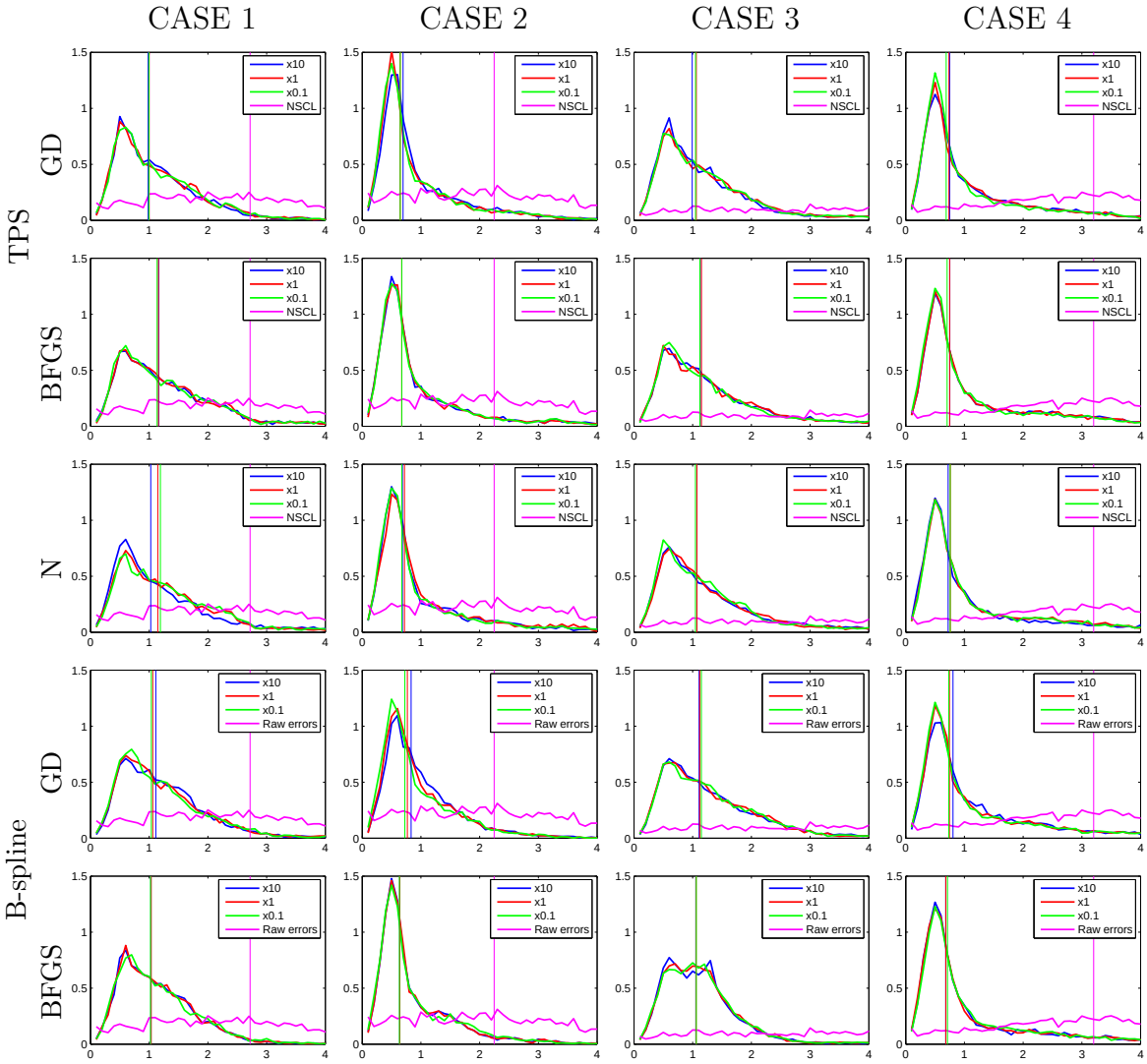



**Figure D.2** The graphs show the distribution of errors over a $50 \times 50 \times 50$ grid of points evenly spaced over the image extent. The image difference measure used was MSD. This result is selected as being typical of the results of this experiment. The graph shows the distribution of distances between points in a warped mesh of the balloon part of the phantom, and the mesh of the balloon created from the target volume. The vertical lines show the median of the error. Cases are (1) partially inflated to deflated, (2) partially inflated to fully inflated, (3) fully inflated to deflated, (4) fully inflated to partially inflated. For case 3, the median error for the raw data is off the graph, at 5.19mm.

# Appendix E

# Proof of the Boundedness of Image Difference Measure Errors

To be considered an anytime algorithm, the quality of the answer reported by the algorithm must increase as the amount of computation performed increases. In this section it is shown that the maximum error decreases as the number of pixels processed increases for the anytime formulations of the Mean Squared Difference (MSD) and Mutual Information (MI) measures. The result when all pixels are processed is considered to be the correct result, and the error, $|E_p|$ to be the difference in magnitude between the result reported at a particular amount of calculation and this correct result. One assumption is required: that the pixel values in the images and the derivatives of the transformation, $\nabla W(\mathbf{x})$, are bounded.

First consider a simple mean of $N$ values, $V_i$,

$$M = \frac{1}{N}\left(V_1 + V_2 + V_3 + ... + V_N\right),$$

where the $V_i$ are bounded. That is there exists $V_{min}$ and $V_{max}$ such that $\forall V_i : V_{min} \leq V_i \leq V_{max}$. Let $M(p)$ be the mean computed using the first $p$ values, and $M(N)$ be the mean computed using all the values.

**Theorem:** The absolute value of the difference between a mean computed with $p$ values, and that computed with all $N$ values is bounded, and this bound decreases as

$p$ increases. That is

$$\forall q > p : \max_{V_i} |M(q) - M(N)| \leq \max_{V_i} |M(p) - M(N)| \tag{E.1}$$

**Proof:** The maximum difference will occur when all of the values used to compute $M(i)$ have one extreme value, and all the rest have the other. Without loss of generality, assume that the first $p$ values have the value $V_{max}$, and the rest have the value $V_{min}$. Then the maximum possible error is:

$$\begin{aligned}
\max_{V_i} |M(p) - M(N)| &= \frac{1}{p} p V_{max} - \frac{1}{N} p V_{max} - \frac{1}{N}(N-p) V_{min} \\
&= \frac{1}{N} \left[ (N-p)\, V_{max} - (N-p)\, V_{min} \right] \\
&= \frac{N-p}{N} [V_{max} - V_{min}]
\end{aligned}$$

which is bounded and decreases as $p$ increases. Note that equality in Relation E.1 is only achieved if all the $V_i$ are equal, and the error is zero at all computation levels. □

As the pixel values in the images of interest are bounded, their differences and derivatives are also bounded. The MSD and its derivatives are means of bounded terms, so the above theorem applies directly. Note that the derivative of the MSD includes the derivative of the transformation in its values. This is why the assumption that $\nabla W(\mathbf{x})$ is bounded is necessary.

For the MI measure, the joint distribution and its derivatives are averages of bounded terms, so the theorem holds for those internal parameters. However, to show that the error on the MI and its derivative is bounded, the mapping from these intermediate values to the final ones must also be considered. Noting that a log of a product is the sum of the logs:

$$A \log \left( \frac{B}{C \cdot D} \right) = A \log(B) - A \log(C) - A \log(D)$$

Thus the MI measure and its derivative can be expressed as a sum of terms of the form:

$$M_i(p) \log(M_j(p))$$

where the $M_i(p)$ and $M_j(p)$ are means having the bounded error property described above. Furthermore, the terms inside the log are constrained to be between 0 and 1, and terms including a log of zero will be cancelled out. (See [61, p. 18]). Considering a single term, let $E_i(p)$ be the difference between $M_i(p)$ and $M_i(N)$. Then

$$\begin{aligned}
& |M_i(p)\log(M_j(p)) - M_i(N)\log(M_j(N))| \\
= \; & |(M_i(N) - E_i(p))\log(M_j(p)) - M_i(N)\log(M_j(N))| \\
= \; & |M_i(N)(\log(M_j(p)) - \log(M_j(N))) - E_i(p)\log(M_j(p))| \\
= \; & \left| M_i(N)\left(\log\left(\frac{M_j(p)}{M_j(N)}\right)\right) - E_i(p)\log(M_j(p))\right| \\
\leq \; & \left| M_i(N)\left(\log\left(\frac{M_j(p)}{M_j(N)}\right)\right)\right| + |E_i(p)\log(M_j(p))|
\end{aligned}$$

The second term is clearly never greater than $E_i(p)$ and is therefore bounded and decreases as $p$ increases. For the first term, note that the maximum difference will occur when $M_j(p) = 1$ and the remaining contributions to the mean are zero, so that $M_j(N) = \frac{N-p}{N}$. Therefore,

$$\left| M_i(N)\left(\log\left(\frac{M_j(p)}{M_j(N)}\right)\right)\right| \leq \left|\log\left(\frac{N}{p}\right)\right|$$

The right hand side is clearly bounded and decreases as $p$ increases. It can be concluded that since the maximum absolute error on each term in the sum is bounded and decreasing, that the maximum absolute error on the entire sum is also bounded and decreasing. Therefore, the maximum absolute error in the MI measure and its derivative is bounded and this bound decreases as the number of pixels used increases.

# Appendix F

# Jacobian matrices of fixed/moving warp parameters

In this section, it is shown how the $\frac{\partial \phi_f}{\partial \phi_m}$ can be computed for the transforms used in Chapter 7. This involves a number of multidimensional objects which can be considered as matrices, and as vectorizations of these matrices. To be clear in matters of dimensionality Harshman's matrix notation [95] is used. In this notation, all array dimensions are represented as subscripts following the matrix name. Adjacent matrices with the same subscript indicates a contraction (multiplication and summation) along the corresponding dimension. (This is also known as the Einstein summation convention.) For example,

$$A_{IJ}B_{I'J} = \sum_{j=1}^{J} A_{ij}B_{i'j} = A \cdot B^T.$$

Note that $I$ and $I'$ are not the same index and thus no summation is performed on them. Indexes that are grouped with parentheses represent the vectorization of those two dimensions into one longer dimension. For example,

$$\text{if } \; A_{IJ} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \; \text{ then } \; A_{(IJ)} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}.$$

To avoid confusion between subscripts for summation, and modifiers to the names of variables, a modified variable name such as $\boldsymbol{\phi}_m$ will be written as $\underline{\boldsymbol{\phi}_m}_P$ in this notation. For brevity, when no summation or multiplication is involved, such as when a vector is the argument of a function, the subscript will be omitted (*e. g.* $M_{IJ}(\boldsymbol{\phi})$). Finally, $\delta_{IJK}$ represents the identity matrix, that is

$$\delta_{ijk} = \begin{cases} 1 & \text{when } i = j = k \\ 0 & \text{otherwise.} \end{cases}$$

Several transforms in common use in image registration problems have a matrix transform of homogeneous coordinates at their core. That is, the transformation is carried out using the following equation.

$$\begin{bmatrix} \lambda \hat{x}_D \\ \lambda \end{bmatrix}_I = T_{IJ} M_{JK}(\boldsymbol{\phi}) T^{-1}_{KL} \cdot \begin{bmatrix} x_D \\ 1 \end{bmatrix}_L , \tag{F.1}$$

where $x_D$ are the original coordinates, $\hat{x}_D$ are the transformed coordinates, $T_{IJ}$ is a translation that manages the effective center of the transform, and $M_{JK}(\boldsymbol{\phi})$ is a function returning the homogeneous transformation matrix corresponding to the parameters, $\boldsymbol{\phi}$. The transforms are composed through matrix multiplication, and inverted by matrix inversion. It is necessary to find the parameters of the fixed transform $\underline{\boldsymbol{\phi}_f}$ as a function of the parameters of the moving transform, $\underline{\boldsymbol{\phi}_m}$. Using the matrix operations yields,

$$\begin{aligned} \underline{\boldsymbol{\phi}_f}_P(\underline{\boldsymbol{\phi}_m}) &= \underline{\boldsymbol{\phi}_m}^{-1} \circ \underline{\hat{\boldsymbol{\phi}}_m} \\ &= \boldsymbol{\phi}_P(M_{IJ}(\underline{\boldsymbol{\phi}_m}) M_{JK}(\underline{\hat{\boldsymbol{\phi}}_m})) \end{aligned}$$

where $M_{IJ}(\boldsymbol{\phi})$ is the matrix that corresponds to a set of parameters, $\boldsymbol{\phi}$; $\underline{\hat{\boldsymbol{\phi}}_m}$ is the starting transform, and $\boldsymbol{\phi}_P(M)$ is the function that returns the transformation parameters for a given matrix, $M$. So long as the transform center does not change, it may be ignored for this discussion, since

$$\left[T_{IJ} M_{JK} T^{-1}_{KL}\right]^{-1} \left[T_{LM} N_{MN} T^{-1}_{NP}\right] = \left[T_{IJ} M^{-1}_{JK} N_{KN} T^{-1}_{NP}\right] .$$

Then the derivative of interest is

$$\frac{\partial \underline{\boldsymbol{\phi}_f}_P}{\partial \underline{\boldsymbol{\phi}_m}_Q} = \frac{\partial \underline{\boldsymbol{\phi}_f}_P}{\partial M_{(IJ)}(\underline{\boldsymbol{\phi}_m})} \frac{\partial M_{(IJ)}(\underline{\boldsymbol{\phi}_m})}{\partial \underline{\boldsymbol{\phi}_m}_Q}$$

Noting that the derivative of the inverse of a matrix function of a scalar parameter $a$ is:

$$\frac{\partial M_{(IL)}(a)^{-1}}{\partial a} = M_{IJ}(a)^{-1} \frac{\partial M_{JK}(a)}{\partial a} M_{KL}(a)^{-1}$$

and that

$$M_{IK}(\underline{\boldsymbol{\phi}_f}) = M_{IJ}(\underline{\boldsymbol{\phi}_m})^{-1} M_{JK}(\underline{\hat{\boldsymbol{\phi}}_m}),$$

results in

$$\begin{aligned} \frac{\partial M_{(IM)}(\underline{\boldsymbol{\phi}_f})}{\partial \underline{\boldsymbol{\phi}_m}_Q} &= M_{IJ}(\underline{\boldsymbol{\phi}_m})^{-1} \cdot \frac{\partial M_{JK}(\underline{\boldsymbol{\phi}_m})}{\partial \underline{\boldsymbol{\phi}_m}_Q} \\ &\cdot \quad M_{KL}(\underline{\boldsymbol{\phi}_m})^{-1} \cdot M_{LM}(\underline{\hat{\boldsymbol{\phi}}_m}) \end{aligned} \tag{F.2}$$

Note the indexes carefully – the final result is a matrix of size $(IJ) \times Q$ formed by stacking up the vectorized results of $Q$ multiplications of 2D matrices of size $I \times J$ . Therefore to compute $\frac{\partial \underline{\boldsymbol{\phi}_f}_P}{\partial \underline{\boldsymbol{\phi}_m}_Q}$ for a matrix based transform it is only required to know $\frac{\partial \boldsymbol{\phi}_P}{\partial M_{(IJ)}}$ and $\frac{\partial M_{(IJ)}}{\partial \boldsymbol{\phi}_P}$ for its particular parameterization.

## F.1 Homogeneous transform

The parameterization of the homography used here is one commonly used in computer vision (*e. g.* [12]), shown here for the 2D case. (The 3D case – not used in this paper – can easily be developed by adding columns and rows corresponding to the additional dimension.)

$$M(\boldsymbol{\phi}) = \begin{bmatrix} \boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \boldsymbol{\phi}_5 \\ \boldsymbol{\phi}_3 & \boldsymbol{\phi}_4 & \boldsymbol{\phi}_6 \\ \boldsymbol{\phi}_7 & \boldsymbol{\phi}_8 & 1 \end{bmatrix}.$$

There is a linear relationship between $M_{IJ}$ and $\boldsymbol{\phi}_P$ which can be represented as a matrix equation

$$cM_{IJ}(\boldsymbol{\phi}) = A_{(IJ)P}\boldsymbol{\phi}_P = \begin{bmatrix} \delta_{DD'} & 0_{DD'} & 0_D & 0_D & 0_D & 0_{DD'} \\ 0_{D'} & 0_{D'} & 0 & 1 & 0 & 0_{D'} \\ 0_{DD'} & \delta_{DD'} & 0_D & 0_D & 0_D & 0_{DD'} \\ 0_{D'} & 0_{D'} & 0 & 0 & 1 & 0_{D'} \\ 0_{DD'} & 0_{DD'} & 0_D & 0_D & 0_D & \delta_{DD'} \\ 0_{D'} & 0_{D'} & 0 & 0 & 0 & 0_{D'} \end{bmatrix}_{(IJ)P} \cdot\boldsymbol{\phi}_P \tag{F.3}$$

The subscripts $D$,$(D')$ will consistently be used to indicate that there are a number of rows (columns) equal to the space dimension.

To do the reverse and obtain $\boldsymbol{\phi}$ from a matrix, the matrix must first be scaled so that its lower right element is one. Then the inverse of Equation F.3 is clearly

$$\boldsymbol{\phi}_P(M) = \frac{1}{M_{3,3}} A^+_{P(IJ)} M_{(IJ)}$$

where $A^+_{P(IJ)}$ is the pseudoinverse of $A_{(IJ)P}$, so

$$\boldsymbol{\phi}_P(M) = \begin{bmatrix} \delta_{DD'} & 0_D & 0_{DD'} & 0_D & 0_{DD'} & 0_D \\ 0_{DD'} & 0_D & \delta_{DD'} & 0_D & 0_{DD'} & 0_D \\ 0_{D'} & 1 & 0_{D'} & 0 & 0_{D'} & 0 \\ 0_{D'} & 0 & 0_{D'} & 1 & 0_{D'} & 0 \\ 0_{DD'} & 0_D & 0_{DD'} & 0_D & \delta_{DD'} & 0_D \end{bmatrix}_{(IJ)P} \frac{M_{(IJ)}}{M_{3,3}} \quad .$$

Then,

$$\frac{\partial \boldsymbol{\phi}_P}{\partial M_{(IJ)}} = \frac{1}{M_{3,3}} A_{(IJ)P} + \frac{\partial M_{3,3}}{\partial M_{(IJ)}} A^+_{P(IJ)} M_{(IJ)}$$

and

$$\frac{\partial \boldsymbol{\phi}_P}{\partial M_{(IJ)}} = \frac{1}{M_{3,3}} \cdot \begin{bmatrix} \delta_{DD'} & 0_D & 0_{DD'} & 0_D & 0_{DD'} & \frac{-1}{M_{3,3}} M_{\frac{1,2}{D}} \\ 0_{DD'} & 0_D & \delta_{DD'} & 0_D & 0_{DD'} & \frac{-1}{M_{3,3}} M_{\frac{4,5}{D}} \\ 0_{D'} & 1 & 0_{D'} & 0 & 0_{D'} & \frac{-1}{M_{3,3}} M_3 \\ 0_{D'} & 0 & 0_{D'} & 1 & 0_{D'} & \frac{-1}{M_{3,3}} M_4 \\ 0_{DD'} & 0_D & 0_{DD'} & 0_D & \delta_{DD'} & \frac{-1}{M_{3,3}} M_{\frac{7,8}{D}} \end{bmatrix}_{(IJ)P}$$

The necessary derivative, $\frac{\partial \underline{\boldsymbol{\phi} f}_P}{\partial \underline{\boldsymbol{\phi} m}_Q}$ can now be calculated using Equation F.2.

## F.2 Rigid 3D transform

A rigid 3D transform can also be represented as a matrix. However, the rigid transforms form a subgroup of the matrix transforms, and consequently cannot be parameterized in terms of matrix elements. This renders the derivative quite non-linear in the parameters.

Here the 3D case is developed, for an Euler angle parameterization of 3D rigid transforms, which is used in the thesis. Euler angle parameterizations present flaws when large rotations (near 90°) are present, but that is not the case for the problems of interest in this thesis. The matrix for a given set of parameters is given by

$$A_{IJ}(\boldsymbol{\phi}) = \begin{bmatrix} R_{\mathcal{Z}}(\boldsymbol{\phi}_3) R_{\mathcal{X}}(\boldsymbol{\phi}_1) R_{\mathcal{Y}}(\boldsymbol{\phi}_2) & \boldsymbol{\phi}_{4,5,6} \\ 0_{D'} & 1 \end{bmatrix}_{IJ}$$

where $R_\alpha(\phi)$ is a rotation matrix around the $\alpha$ axis by an angle $\phi$.

For brevity in what follows, define $\mathrm{s}_1 = \sin(\boldsymbol{\phi}_1)$, $\mathrm{c}_1 = \cos(\boldsymbol{\phi}_1)$ and so forth. The

matrix $A_{IJ}(\boldsymbol{\phi})$ can be expanded as

$$A_{IJ}(\boldsymbol{\phi}) = \begin{bmatrix} \mathrm{c}_3\mathrm{c}_2 - \mathrm{s}_3\mathrm{s}_1\mathrm{s}_2 & -\mathrm{s}_3\mathrm{c}_1 & \mathrm{c}_3\mathrm{s}_2 + \mathrm{s}_3\mathrm{s}_1\mathrm{c}_2 & \boldsymbol{\phi}_4 \\ \mathrm{s}_3\mathrm{c}_2 + \mathrm{c}_3\mathrm{s}_1\mathrm{s}_2 & \mathrm{c}_3\mathrm{c}_1 & \mathrm{s}_3\mathrm{s}_2 - \mathrm{c}_3\mathrm{s}_1\mathrm{c}_2 & \boldsymbol{\phi}_5 \\ -\mathrm{c}_1\mathrm{s}_2 & \mathrm{s}_1 & \mathrm{c}_1\mathrm{c}_2 & \boldsymbol{\phi}_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{IJ}.$$

If $\mathrm{c}_1$ is small then

$$\boldsymbol{\phi}_P(A) = \begin{bmatrix} \sin^{-1}(A_{3,2}) \\ \tan^{-1}(\frac{A_{3,3}}{\cos(\sin^{-1}(A_{3,2}))}, \frac{-A_{3,1}}{\cos(\sin^{-1}(A_{3,2}))}) \\ \tan^{-1}(\frac{A_{2,2}}{\cos(\sin^{-1}(A_{3,2}))}, \frac{-A_{1,2}}{\cos(\sin^{-1}(A_{3,2}))}) \\ A_{1,4} \\ A_{2,4} \\ A_{3,4} \end{bmatrix}; \tag{F.4}$$

otherwise,

$$\boldsymbol{\phi}_P(A) = \begin{bmatrix} \sin^{-1}(A_{3,2}) \\ \tan^{-1}(A_{1,1}, A_{2,1}) \\ 0 \\ A_{1,4} \\ A_{2,4} \\ A_{3,4} \end{bmatrix} \tag{F.5}$$

(The criterion that $|A_{3,2}| < 0.99999999875$ has been used to choose between Equations F.4 and F.4.)

Here $\tan^{-1}(c, s)$ is the inverse tangent function defined for the full circle, where $c$ and $s$ are the cosine and sine of the angle in question:

$$\tan^{-1}(c, s) = \begin{cases} \sin^{-1}(s) & \forall \quad |c| > |s|, c > 0 \\ \pi - \sin^{-1}(s) & \forall \quad |c| > |s|, c < 0 \\ \cos^{-1}(c) & \forall \quad |c| < |s|, s > 0 \\ -\cos^{-1}(c) & \forall \quad |c| < |s|, s < 0 \end{cases}$$

The derivative of $\tan^{-1}(c, s)$, where $c$ and $s$ are dependent functions of some other

variable, $x$, is

$$\frac{\partial \tan^{-1}(c(x), s(x))}{\partial x} = \begin{cases} \frac{1}{c(x)} \frac{\partial c(x)}{\partial x} & \forall \quad |c(x)| > |s(x)| \\ \frac{-1}{s(x)} \frac{\partial s(x)}{\partial x} & \forall \quad |c(x)| < |s(x)| \end{cases}$$

The derivatives of $A(\boldsymbol{\phi})$ and $\boldsymbol{\phi}(A)$ can now be computed

$$\frac{\partial A_{(IJ)}(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}_P} = \begin{bmatrix} -\mathrm{s}_3\mathrm{s}_2\mathrm{c}_1 & -\mathrm{c}_3\mathrm{s}_2 - \mathrm{s}_3\mathrm{s}_1\mathrm{c}_2 & -\mathrm{s}_3\mathrm{c}_2 - \mathrm{c}_3\mathrm{s}_1\mathrm{s}_2 & 0 & 0 & 0 \\ \mathrm{s}_3\mathrm{s}_1 & 0 & -\mathrm{c}_3\mathrm{c}_1 & 0 & 0 & 0 \\ \mathrm{s}_3\mathrm{c}_1\mathrm{c}_2 & \mathrm{c}_3\mathrm{c}_2 - \mathrm{s}_3\mathrm{s}_1\mathrm{s}_2 & -\mathrm{s}_3\mathrm{s}_2 + \mathrm{c}_3\mathrm{s}_1\mathrm{c}_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \mathrm{c}_3\mathrm{s}_2\mathrm{c}_1 & -\mathrm{s}_3\mathrm{s}_2 + \mathrm{c}_3\mathrm{s}_1\mathrm{c}_2 & \mathrm{c}_3\mathrm{c}_2 - \mathrm{s}_3\mathrm{s}_1\mathrm{s}_2 & 0 & 0 & 0 \\ -\mathrm{c}_3\mathrm{s}_1 & 0 & -\mathrm{s}_3\mathrm{c}_1 & 0 & 0 & 0 \\ -\mathrm{c}_3\mathrm{c}_1\mathrm{c}_2 & \mathrm{s}_3\mathrm{c}_2 + \mathrm{c}_3\mathrm{s}_1\mathrm{s}_2 & \mathrm{c}_3\mathrm{s}_2 + \mathrm{s}_3\mathrm{s}_1\mathrm{c}_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \mathrm{s}_1\mathrm{s}_2 & -\mathrm{c}_1\mathrm{c}_2 & 0 & 0 & 0 & 0 \\ \mathrm{c}_1 & 0 & 0 & 0 & 0 & 0 \\ -\mathrm{c}_2\mathrm{s}_1 & -\mathrm{c}_1\mathrm{s}_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(IJ)P} \tag{F.6}$$

Let $\lambda$ and $\bar{\lambda}$ be indicator functions where $\lambda_i = 1$ iff $|c_i| \geq |s_i|$ and $\bar{\lambda}_i = (1 - \lambda_i)$.

$$\frac{\partial \boldsymbol{\phi}_P(A)}{\partial A_{(IJ)}}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-\lambda_3}{\mathrm{c}_1\mathrm{c}_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-\bar{\lambda}_3}{\mathrm{c}_1\mathrm{s}_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & \frac{-\lambda_2}{\mathrm{c}_1\mathrm{c}_2} & 0 & 0 & 0 & 0 \\ \frac{1}{\mathrm{c}_1} & p & q & 0 & 0 & 0 \\ 0 & \frac{-\bar{\lambda}_2}{\mathrm{c}_1\mathrm{s}_2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{P(IJ)} \tag{F.7}$$

Where

$$p = \frac{\lambda_2 A_{3,1}\mathrm{s}_1}{\mathrm{c}_2|\mathrm{c}_1^3|} + \frac{\bar{\lambda}_2 A_{3,3}\mathrm{s}_1}{\mathrm{s}_2|\mathrm{c}_1^3|} \quad \text{and} \quad q = \frac{\lambda_3 A_{1,2}\mathrm{s}_1}{\mathrm{c}_3|\mathrm{c}_1^3|} + \frac{\bar{\lambda}_3 A_{2,2}\mathrm{s}_1}{\mathrm{s}_3|\mathrm{c}_1^3|}$$

Equation F.4 is valid as long as the angles involved do not approach 90 degrees. However, this is not a concern in this work since Equation F.7 is only ever evaluated at $\boldsymbol{\phi}_f = 0$. At this point, all the cosine terms become one, and the sine terms become

zero, allowing the simplification of Equation F.7 to

$$\frac{\partial \boldsymbol{\phi}_P(A)}{\partial A_{(IJ)}}^T = \left[\begin{array}{cccccc} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \left[\begin{array}{cccccc} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \end{array}\right] \left[\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right]_{P(IJ)} .$$

As in the previous case, this allows the calculation of $\frac{\partial \underline{\boldsymbol{\phi} f}_P}{\partial \underline{\boldsymbol{\phi} m}_Q}$ using Equation F.2.

# References

[1] S. Aine, P. P. Chakrabarti, and R. Kumar. An automated meta-level control framework for optimizing the quality–time tradeoff of VLSI algorithms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26 (11):1997–2008, 2007.

[2] A. Andreopoulos and J. K. Tsotsos. A novel algorithm for fitting 3D active appearance models: Applications to cardiac MRI segmentation. In *Proceedings of the 14th Scandinavian Conference on Image Analysis*, Joensuu, Finland, 2005.

[3] P. E. Anuta. Digital registration of multispectral imagery. *SPIE Journal*, 7(6): 168–175, 1969.

[4] P. E. Anuta. Spatial registration of multispectral and multitemporal digital imagery using fast Fourier transform techniques. *IEEE Transactions on Geoscience Electronics*, 8(4):353–368, 1970.

[5] T. Arbel, X. Morandi, R. M. Comeau, and D. L. Collins. Automatic nonlinear MRI-ultrasound registration for the correction of intra-operative brain deformations. In W. Niessen and M. A. Viergever, editors, *Proceedings of the Conference for Medical Image Computing and Computer Assisted Intervention - MICCAI 2001*, pages 913–922. Springer-Verlag, Utrecht, Netherlands, October 2001.

[6] M. Audette, R. Brooks, R. Funnell, G. Strauss, and T. Arbel. Piecewise affine initialized spline-based patient-specific registration of a high-resolution ear model for surgical guidance. In *MICCAI Workshop on Image Guidance and Computer Assistance for Soft-Tissue Interventions*, 2008.

[7] S. Aylward, J. Jomier, S. Barre, B. Davis, and L. Ibanez. Optimizing ITKs registration methods for multi-processor, shared-memory systems. *Insight Journal*, 2007. ISC/NA-MIC Workshop on Open Science at MICCAI 2007 Issue `http://hdl.handle.net/1926/566`.

[8] R. Bajcsy and C. Broit. Matching of deformed images. In *Proceedings of the International Conference on Pattern Recognition, (ICPR'82)*, pages 351–353, 1982.

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

[10] A. Baker. *Matrix Groups: An Introduction to Lie Group Theory*. Springer, 2002.

[11] S. Baker and I. Matthews. Equivalence and efficiency of image alignment algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1090–1097, 2001.

[12] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unified framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[13] S. Baker, R. Patil, K. M. Cheung, and I. Matthews. Lucas-Kanade 20 years on: Part 5. Technical Report CMU-RI-TR-04-64, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2004.

[14] S. K. Balci, P. Golland, and W. M. Wells III. Non-rigid groupwise registration using B-spline deformation model. *Insight Journal - 2007 MICCAI Open Science Workshop*, 2007.

[15] E. I. Barnea and H. F. Silverman. A class of algorithms for fast digital image registration. *IEEE Transactions on Computers*, C-21:179–186, 1972.

[16] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*. SIAM, Philadelphia, PA, 1994.

[17] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[18] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 659–663, 1977.

[19] A. Bartoli. Groupwise geometric and photometric direct image registration. In *Proceedings of the 17th British Machine Vision Conference*, volume 1, pages 157–166, Edinburgh, UK, September 2006.

[20] A. Bartoli and A. Zisserman. Direct estimation of non-rigid registrations. In *Proceedings of the 15th British Machine Vision Conference*, pages 899–908, 2004.

[21] A. Bartoli, M. Perriollat, and S. Chambon. Generalized thin-plate spline warps. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'07)*, 2007.

[22] S. Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots Systems*, Sendai, Japan, October 2004.

[23] S. Benhimane, A. Ladikos, V. Lepetit, and N. Navab. Linear and quadratic subsets for template-based tracking. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'07)*, 2007.

[24] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proceedings of the European Conference on Computer Vision (ECCV'92)*, pages 237–252, 1992.

[25] J. L. Bernon, D. Mariano-Goulart, M. Zanca, and M. Rossi. A fast 3-d multimodality registration algorithm for human brain. In *Proceedings of the International Conference on Image Processing (ICIP99)*, volume 3, pages 446–448, Kobe, Japan, 1999.

[26] J. L. Bernon, V. Boudousq, J. F. Rohmer, M. Fourcade, M. Zanca, M. Rossi, and D. Mariano-Goulart. A comparative study of Powell's and downhill simplex algorithms for a fast multimodal surface matching in brain imaging. *Computerized Medical Imaging and Graphics*, 24(4):287–297, 2001.

[27] K. K. Bhatia, J. Hajnal, A. Hammers, and D. Rueckert. Similarity metrics for groupwise non-rigid registration. In N. Ayache, S. Ourselin, and A. Maeder, editors, *Proceedings of the 10th International Conference on Medical Image Computing and Computer Aided Intervention (MICCAI 2007), Vol. 2*, number 4792 in Lecture Notes in Computer Science, pages 544–552, Brisbane, Australia, October 2007. Springer.

[28] N. Bićanić and K. H. Johnson. Who was '-Raphson'? *International Journal for Numerical Methods in Engineering*, 14(1):148–152, 1979.

[29] M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *Proceedings of the European Conference on Computer Vision (ECCV'96)*, volume 1, pages 329–342, 1996. URL `citeseer.nj.nec.com/black96eigentracking.html`.

[30] J. Bloomenthal and J. Rokne. Homogeneous coordinates. *The Visual Computer*, 11(1):15–26, 1994.

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

[32] R. Brooks and T. Arbel. Generalizing inverse compositional image alignment. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR2006)*, volume 2, pages 1200–1203, Hong Kong, August 2006.

[33] R. Brooks and T. Arbel. The importance of scale when selecting pixels for image registration. In *Proceedings of the 4th Canadian Conference on Computer and Robot Vision (CRV2007)*, pages 235–242, Montreal, Canada, May 2007.

[34] R. Brooks and T. Arbel. Generalizing inverse compositional and ESM image alignment. *International Journal of Computer Vision*, 2008. SUBMITTED FOR PUBLICATION.

[35] R. Brooks and T. Arbel. A homogeneous transform class for the ITK. *Insight Journal*, March 2007. DSpace handle `http://hdl.handle.net/1926/493`.

[36] R. Brooks and T. Arbel. Improvements to the itk::KernelTransform and subclasses. *Insight Journal*, March 2007. DSpace handle `http://hdl.handle.net/1926/494`.

[37] R. Brooks, D. L. Collins, and T. Arbel. Scaling angles and distances to maximize efficiency of image registration. Short paper presented at the 8th International Conference on Medical Image Computing and Computer Aided Intervention (MICCAI 2005), October 2005. available online `http://www.ia.unc.edu/MICCAI2005/ShortPapers/`.

[38] R. Brooks, T. Arbel, and D. Precup. Fast image alignment using anytime algorithms. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI2007)*, pages 2078–2083, Hyderabad, India, January 2007.

[39] R. Brooks, T. Arbel, and D. Precup. Anytime similarity measures for faster alignment. *Computer Vision and Image Understanding*, 110(3):378–389, 2008.

[40] R. Brooks, D. L. Collins, X. Morandi, and T. Arbel. Deformable ultrasound registration without reconstruction. In D. Metaxas, L. Axel, G. Szekely, and G. Fichtinger, editors, *Proceedings of the 11th International Conference on Medical Image Computing and Computer Aided Intervention (MICCAI 2008), Vol.*

*2*, number 5242 in Lecture Notes in Computer Science, pages 1023–1031, New York, USA, September 2008. Springer.

[41] L. G. Brown. A survey of image registration techniques. *ACM Computing Surveys*, 24(4):325–376, December 1992.

[42] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 74(1):59–73, 2007.

[43] C. G. Broyden. The convergence of a class of double-rank minimization algorithms. *Journal of the Institute of Mathematics and Its Applications*, 6:76–90, 1970.

[44] J. M. Buenaposada and L. Baumela. Speeding up SSD planar tracking by pixel selection. In *Proceedings of the International Conference on Image Processing (ICIP'02)*, pages 565–569, 2002.

[45] D. J. Burr. A dynamic model for image registration. In *Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing*, pages 17–24, Chicago, IL, USA, 1979.

[46] D. J. Burr. A dynamic model for image registration. *Computer Graphics and Image Processing*, 15:102–112, 1981.

[47] R. Burtch. History of photogrammetry. Course Notes for Survey Engineering 340, Ferris University, 2008. URL `http://www.ferris.edu/faculty/burtchr/sure340/notes/History.pdf`. Accessed April 23, 2008.

[48] R. H. Byrd, P. Lu, and J. Nocedal. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific and Statistical Computing*, 16(5):1190–1208, 1995.

[49] P. Cachier, E. Bardinet, D. Dormont, X. Pennec, and N. Ayache. Iconic feature based nonrigid registration: The PASHA algorithm. *Computer Vision and Image Understanding*, 89(2-3):272–298, 2003.

[50] R. G. Carter. Numerical experience with a class of algorithms for nonlinear optimization using inexact function and gradient information. *SIAM Journal of Scientific Computing*, 14(2):368–388, 1993.

[51] Y. Chen, R. R. Brooks, S. S. Iyengar, N. S. V. Rao, and J. Barhen. Efficient global optimization for image registration. *IEEE Transactions on Data and Knowledge Engineering*, 14(1):79–92, 2002.

[52] G. E. Christensen and H. Johnson. Consistent image registration. *IEEE Transactions on Medical Imaging*, 20(7):568–582, 2001.

[53] G. E. Christensen, R. D. Rabbitt, and M. I. Miller. Deformable templates using large deformation kinematics. *IEEE Transactions on Image Processing*, 5(10): 1435–1447, 1996.

[54] O. Clatz, H. Delingette, I. Talos, A. J. Golby, R. Kikinis, F. A. Jolesz, N. Ayache, and S. K. Warfield. Robust nonrigid registration to capture brain shift from intraoperative MRI. *IEEE Transactions on Medical Imaging*, 24(11):1417–1427, 2005.

[55] A. A. Cole-Rhodes, K. L. Johnson, J. Le Moigne, and I. Zavorin. Multiresolution registration of remote sensing imagery by optimization of mutual information using a stochastic gradient. *IEEE Transactions on Image Processing*, 12(12): 1495–1511, 2003.

[56] A. Collignon, F. Maes, D. Delaere, D. Vandermeulen, P. Suetens, and G. Marchal. Automated multimodality image registration using information theory. In Y. Bizais, C. Barillot, and R. Di Paola, editors, *Proceedings of the 14th International Conference on Information Processing in Medical Imaging (IPMI-95)*, pages 263–274, Ile de Berder, France, June 1995. Kluwer Academic.

[57] A. Collignon, D. Vandermeulen, P. Suetens, and G. Marchal. 3D multimodality medical image registration using feature space clustering. In N. Ayache, editor, *Computer Vision, Virtual Reality, and Robotics in Medicine*, volume 905 of *Lecture Notes In Computer Science*, pages 195–204, 1995.

[58] D. L. Collins and A. C. Evans. ANIMAL: Validation and applications of nonlinear registration-based segmentation. *International Journal of Pattern Recognition and Artificial Intelligence*, 11(8):1271–1294, 1997.

[59] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust-Region Methods*. Society for Industrial and Applied Mathematics and Mathematical Programming Society, 2000.

[60] P. Corke. *Visual Control of Robots: High Performance Visual Servoing*. Wiley, 1996.

[61] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, NY, USA, 1991.

[62] W. R. Crum, T. Hartkens, and D. L. G. Hill. Non-rigid image registration: theory and practice. *British Journal of Radiology*, 77(Special Issue):S140–S153, 2004.

[63] M. H. Davis, A. Khotanzad, D. P. Flamig, and S. E. Harms. A physics-based coordinate transformation for 3D image matching. *IEEE Transactions on Medical Imaging*, 16(3):317–328, 1997.

[64] A. J. Davison. Active search for real-time vision. In *Proceedings of the International Conference on Computer Vision (ICCV'05)*, 2005.

[65] T. Dean and M. S. Boddy. An analysis of time-dependent planning. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 49–54, St. Paul, Minnesota, USA, August 1988. AAAI Press.

[66] F. Dellaert and R. Collins. Fast image-based tracking by selective pixel integration. Presented at the Workshop on Frame-Rate Vision, 6th International Conference on Computer Vision (ICCV-99), September 1999. Available online `http://www.ri.cmu.edu/pubs/pub_3195.html`.

[67] J. Dengler. Local motion estimation with the dynamic pyramid. In *Pyramidal Systems for Computer Vision*, pages 289–297. Springer-Verlag, 1986.

[68] J. E. Dennis, Jr. and H. Wolkowicz. Sizing and least-change secant methods. *SIAM Journal of Numerical Analysis*, 30(5):1291–1314, 1993.

[69] L. di Stefano, S. Mattoccia, and F. Tombari. An algorithm for efficient and exhaustive template matching. In *Proceedings of the International Conference on Image Analysis and Recognition*, pages 408–415, Porto, Portugal, 2004.

[70] N. Dowson and R. Bowden. A unifying framework for mutual information methods for use in non-linear optimisation. In A. Leonardis, H. Bischof, and A. Prinz, editors, *Proceedings of the European Conference on Computer Vision (ECCV 2006)*, volume 3951 of *Lecture Notes in Computer Science*, pages 365–378. Springer-Verlag, 2006.

[71] N. Dowson and R. Bowden. Mutual information for Lucas-Kanade tracking (MILK): An inverse compositional formulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1):180–185, 2008.

[72] F. Dufaux and J. Konrad. Efficient, robust, and fast global motion estimation for video coding. *IEEE Transactions on Image Processing*, 9(3):497–501, 2000.

[73] R. M. Endlich, D. E. Wolf, D. J. Hall, and A. E. Brain. Use of a pattern recognition technique for determining cloud motions from sequences of satellite photographs. *Journal of Applied Meteorology*, 10(1):105–117, 1971.

[74] W. Enkelmann. Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences. *Computer Vision, Graphics, and Image Processing*, 43(2):150–177, 1988.

[75] A. C. Evans, W. Dai, D. L. Collins, P. Neelin, and S. Marrett. Warping of a computerized 3D atlas to match brain image volumes for quantitative neuroanatomical and functional analysis. In *Proceedings of SPIE – Volume 1445 Medical Imaging V: Image Processing*, pages 236–246, 1991.

[76] A. C. Evans, D. L. Collins, S. R. Mills, E. D. Brown, R. L. Kelly, and T. M. Peters. 3D statistical neuroanatomical models from 305 MRI volumes. In *Proceedings of the IEEE-Nuclear Science Symposium and Medical Imaging Conference*, pages 1813–1817, 1993.

[77] V. Fischer and H. Niemann. A parallel any-time control algorithm for image understanding. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 1, pages 141–145, 25-29 Aug 1996. doi: 10.1109/ICPR.1996.546007.

[78] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, C-22(1):67–92, 1973.

[79] R. Fletcher. A new approach to variable metric algorithms. *Computer Journal*, 13:317–322, 1970.

[80] R. Fletcher. *Practical Methods Of Optimization*. John Wiley & Sons, New York, NY, USA, 1987.

[81] R. Floca and H. Dickhaus. A flexible registration and evaluation engine (f.r.e.e.). *Computer Methods and Programs in Biomedicine*, 87(2):81–92, 2007.

[82] C. F. Gauss. *Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientium*. Dover, 1809. English translation by C. H. Davis, reprinted 1963.

[83] V. Gay-Bellile, A. Bartoli, and P. Sayd. Direct estimation of non-rigid registrations with image-based self-occlusion reasoning. In *Proceedings of the International Conference on Computer Vision (ICCV'07)*, 2007.

[84] J. J. Gibson. *Perception of the Visual World*. Houghton Mifflin, Boston, 1950.

[85] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press., 1981.

[86] D. Goldfarb. A family of variable metric updates derived by variational means. *Mathematics of Computation*, 24:23–26, 1970.

[87] A. Goshtasby. Piecewise linear mapping function for image registration. *Pattern Recognition*, 19(6):459–466, 1986.

[88] A. Goshtasby. Piecewise cubic mapping functions for image registration. *Pattern Recognition*, 20(5):525–533, 1987.

[89] A. Goshtasby, L. Staib, C. Studholme, and D. Terzopoulos. Nonrigid image registration: Guest editors introduction. *Computer Vision and Image Understanding*, 89:109–113, 2003.

[90] G. D. Hager and P. N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.

[91] J. V. Hajnal, D. L. G. Hill, and D. J. Hawkes, editors. *Medical Image Registration*. CRC Press, New York, NY, USA, 2001.

[92] E. A. Hansen and S. Zilberstein. Monitoring and control of anytime algorithms: A dynamic programming approach. *Artificial Intelligence*, 126:5–41, 2001.

[93] A. J. Hanson. *Visualizing Quaternions*. Morgan Kaufmann, 2006.

[94] R. L. Harder and R. N. Desmarais. Interpolation using surface splines. *Journal of Aircraft*, 9(2):189–191, 1972.

[95] R. A. Harshman. An index formalism that generalizes the capabilities of matrix notation and algebra to n-way arrays. *Journal of Chemometrics*, 15(9):689–714, 2001.

[96] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2003.

[97] N. Hata, A. Nabavi, W. M. Wells III, S. K. Warfield, R. Kikinis, P. Black, and F. A. Jolesz. Three-dimensional optical flow method for measurement of volumetric brain deformation from intraoperative MR images. *Journal of Computer Assisted Tomography*, 24(4):531–538, 2000.

[98] Y. He, K. H. Yap, L. Chen, and L. P. Chau. A nonlinear least square technique for simultaneous image registration and super-resolution. *IEEE Transactions on Image Processing*, 16(11):2830–2841, 2007.

[99] D. L. G. Hill, C. Studholme, and D. J. Hawkes. Voxel similarity measures for automated image registration. In R. A. Robb, editor, *Proceedings of SPIE – Volume 2359 Visualization in Biomedical Computing*, pages 205–216, 1994.

[100] 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.

[101] G. L. Hobrough. Automatic stereo plotting. *Photogrammetric Engineering*, 25 (5):763–769, 1959.

[102] G. L. Hobrough. Automatic stereoplotting system and method, 1961. U.S. Patent No. 3,145,303.

[103] G. L. Hobrough and G. A. Wood. Automatic image registration. *International Archives of Photogrammetry*, 15(4):17 pages, 1965.

[104] A. Holder, editor. *Mathematical Programming Glossary*. INFORMS Computing Society, `http://glossary.computing.society.informs.org`, 2006–07. Originally authored by H. J. Greenberg, 1999-2006.

[105] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[106] E. Horvitz. Reasoning about beliefs and actions under computational resource constraints. In *Proceedings of the 3rd Workshop on Uncertainty in Artificial Intelligence (UAI-87)*, pages 429–444, Seattle, WA, USA, July 1987. Elsevier Science.

[107] E. Horvitz and S. Zilberstein. Editorial: Computational tradeoffs under bounded resources. *Artificial Intelligence*, 126(1-2):1–4, 2001.

[108] X. Huang, N. A. Hill, J. Ren, and T. M. Peters. Rapid registration of multimodal images using a reduced number of voxels. In Kevin R. Cleary and Jr. Galloway, Robert L., editors, *Proceedings of SPIE Medical Imaging 2006: Visualization, Image-Guided Procedures and Display*, volume 6141 614116-1, 2006.

[109] L. Ibanez, W. Schroeder, L. Ng, and J. Cates. *The ITK Software Guide: ITK V2.0*. Kitware Inc, Clifton Park, NY, USA, 2005. `http://www.itk.org`.

[110] M. Irani and P. Anandan. All about direct methods. In *Proceedings of the International Workshop on Vision Algorithms*, Corfu, Greece, 1999.

[111] M. Irani and S. Peleg. Improving resolution by image registration. *CVGIP:Gr aphical Models and Image Processing*, 53:231–239, 1991.

[112] S. Joshi, B. Davis, M. Jomier, and G. Gerig. Unbiased diffeomorphic atlas construction for computational anatomy. *NeuroImage*, 23:S151–S160, 2004.

[113] K. Kanatani, Y. Shimizu, N. Ohta, M. J. Brooks, W. Chojnacki, and A. van den Hengel. Fundamental matrix from optical flow: optimal computation and reliability evaluation. *Journal of Electronic Imaging*, 9(2):194–202, 2000.

[114] Y. Keller and A. Averbuch. Fast motion estimation using bi-directional gradient methods. *IEEE Transactions on Image Processing*, 13(8):1042–1054, 2004.

[115] Kitware, Inc. itk::LBFGSBOptimizer. Source code of ITK Toolkit, Version 3.4. Online documentation: `http://www.itk.org/Doxygen34/html/classitk_1_1LBFGSBOptimizer.html`, 2007.

[116] Kitware, Inc. itk::PowellOptimizer. Source code of ITK Toolkit, Version 3.4. Online documentation: `http://www.itk.org/Doxygen34/html/classitk_1_1PowellOptimizer.html`, 2007.

[117] Kitware, Inc. itk::RegularStepGradientDescentOptimizer. Source code of ITK Toolkit, Version 3.4. Online documentation: `http://www.itk.org/Doxygen34/html/classitk_1_1RegularStepGradientDescentOptimizer.html`, 2007.

[118] Kitware, Inc. itk::AmoebaOptimizer. Source code of ITK Toolkit, Version 3.4. Online documentation: `http://www.itk.org/Doxygen34/html/classitk_1_1AmoebaOptimizer.html`, 2007.

[119] S. Klein, M. Staring, and J. P. W. Pluim. A comparison of acceleration techniques for nonrigid medical image registration. In *Proceedings of the 3rd International Conference on Biomedical Image Registration*, number 4057 in Lecture Notes in Computer Science. Springer, 2006.

[120] S. Klein, M. Staring, and J. P. W. Pluim. Evaluation of optimization methods for nonrigid medical image registration using mutual information and B-splines. *IEEE Transactions on Image Processing*, 16(12):2879–2890, 2007.

[121] A. Köhn, J. Drexl, F. Ritter, M. König, and H. O. Peitgen. GPU accelerated image registration in two and three dimensions. In *Bildverarbeitung für die Medizin 2006*, pages 261–265, 2006.

[122] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. In *Proceedings of the IEEE International Conference on Cybernetics and Society*, pages 163–165, 1975.

[123] R. K. S. Kwan, A. C. Evans, and G. B. Pike. MRI simulation-based evaluation of image-processing and classification methods. *IEEE Transactions on Medical Imaging*, 18(11):1085–1097, 1999.

[124] J. Kybic and M. Unser. Fast parametric elastic image registration. *IEEE Transactions on Image Processing*, 12(11):1427–1442, 2003.

[125] W. W. Kywe, D. Fujiwara, and K. Murakami. Scheduling of image processing using anytime algorithm for real-time system. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR2006)*, volume 3, pages 1095–1098, Hong Kong, China, August 2006.

[126] K. Larson and T. Sandholm. Using performance profile trees to improve deliberation control. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-04)*, pages 73–79, San Jose, CA, USA, July 2004. AAAI Press.

[127] S. Lavallée, R. Szeliski, and L. Brune. Anatomy-based registration of three-dimensional medical images, range images, x-ray projection and three-dimensional models using octree-splines. In R. H. Taylor, S. Lavallée, G. C. Burdea, and R. Mösges, editors, *Computer Integrated Surgery: Technology and Clinical Applications*, pages 115–144. MIT Press, 1996.

[128] J. A. Leese, C. S. Novak, and B. B. Clark. An automated technique for obtaining cloud motion from geosynchronous satellite data using cross correlation. *Journal of Applied Meteorology*, 10(1):118–132, 1971.

[129] M. M. J. Letteboer, P. Willems, M. A. Viergever, and W. J. Niessen. Brain shift estimation in image-guided neurosurgery using 3D ultrasound. *IEEE Transactions on Biomedical Engineering*, 52(2), 2004.

[130] K. Levenberg. A method for the solution of certain problems in least squares. *Quarterly of Applied Mathematics*, 2:164–168, 1944.

[131] R. M. Lewis, V. Torczon, and M. W. Trosset. *Numerical Analysis 2000 Volume 4 - Nonlinear Equations and Optimisation*, chapter Direct Search Methods: Then and Now, pages 191–207. Elsevier, 2001.

[132] T. L. Liu and H. T. Chen. Real-time tracking using trust-region methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):397–402, 2004.

[133] D. Loeckx, F. Maes, D. Vandermeulen, and P. Suetens. Nonrigid image registration using free-form deformations with a local rigidity constraint. In C. Barillot, D. R. Haynor, and P. Hellier, editors, *Proceedings of the 7th International Conference on Medical Image Computing and Computer Aided Intervention (MICCAI 2004), Vol. 1*, number 3216 in Lecture Notes in Computer Science, pages 639–646, Saint-Malo, France, September 2004. Springer.

[134] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.

[135] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision (ICCV'99)*, pages 1150–1157, 1999.

[136] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of Image Understanding Workshop*, pages 121–130, 1981.

[137] F. Maes, D. Vandermeulen, and P. Suetens. Comparative evaluation of multiresolution optimization strategies for multimodality image registration by maximization of mutual information. *Medical Image Analysis*, 3(4):373–386, 1999.

[138] J. B. A. Maintz and M. A. Viergever. A survey of medical image registration. *Medical Image Analysis*, 2(1):1–36, 1998.

[139] E. Malis. Improving vision-based control using efficient second-order minimization techniques. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA04)*, pages 1843–1848, New Orleans, LA, USA, April 2004.

[140] E. Malis and S. Benhimane. A unified approach to visual tracking and servoing. *Robotics and Autonomous Systems*, 52(1):39–52, 2005.

[141] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal of Applied Mathematics*, 11:431–441, 1963.

[142] D. Mattes, D. R. Haynor, H. Vesselle, T. Lewellen, and W. Eubank. PET-CT image registration in the chest using free-form deformations. *IEEE Transactions on Medical Imaging*, 22(1):120–128, 2003.

[143] I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135–164, November 2004.

[144] K. I. M. McKinnon. Convergence of the Nelder-Mead simplex method to a nonstationary point. *SIAM Journal on Optimization*, 9:148–158, 1999.

[145] J. Modersitzki. *Numerical Methods for Image Registration*. Numerical Mathematics and Scientific Computation. Oxford University Press, 2004.

[146] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty five years later. *SIAM Review*, 45(1):3–49, 2003.

[147] K. I. Mori, M. Kidode, and H. Asada. An iterative prediction and correction method for automatic stereocomparison. *Computer Graphics and Image Processing*, 2(3-4):393–401, 1973.

[148] R. Nagel and A. Rosenfeld. Ordered search techniques in template matching. *Proceedings of the IEEE*, 60(2):242–244, 1972.

[149] A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[150] I. Newton. *De Analysi per Aequationes Infinitas*. 1669. As described and translated in Bićanić and Johnson, 1979.

[151] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006.

[152] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 1025–1030, Acapulco, Mexico, August 2003. Morgan Kaufman.

[153] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. Interpolation artefacts in mutual information-based image registration. *Computer Vision and Image Understanding*, 77(2):211–232, 2000.

[154] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. Mutual information matching in multiresolution contexts. *Image and Vision Computing*, 19:45–52, 2001.

[155] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. Mutual-information based registration of medical images: A survey. *IEEE Transactions on Medical Imaging*, 22(8):986–1004, 2003.

[156] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer Journal*, 7:155–162, 1964.

[157] W. K. Pratt. Correlation techniques of image registration. *IEEE Transactions on Aerospace Electronic Systems*, 10(3):353–358, 1974.

[158] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1992. URL `http://www.nr.com`.

[159] J. Raphson. *Analysis Aequationum Universalis Seu ad Aequationes Algebraicas Resolvendas Methodus Generalis, et Expedita, ex Nova Infinitarum Serierum Doctrina, Deducta ac Demonstrata*. London, 1690. As described and translated in Bićanić and Johnson, 1979.

[160] I. Reinertsen, M. Descoteaux, K. Siddiqi, and D. L. Collins. Validation of vessel-based registration for correction of brain shift. *Medical Image Analysis*, 11(4): 374–388, 2007.

[161] A. Roche, G. Malandain, N. Ayache, and S. Prima. Towards a better comprehension of similarity measures used in medical image registration. In C. J. Taylor and A. C. F. Colchester, editors, *Proceedings of the 2nd International conference on Medical Image Computing and Computer Assisted Intervention (MICCAI'99)*, number 1679 in Lecture Notes in Computer Science, Cambridge, UK, 1999. Springer.

[162] T. Rohlfing and C. R. Maurer, Jr. Nonrigid 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.

[163] T. Rohlfing, C. R. Maurer, Jr., D. A. Bluemke, and M. A. Jacobs. Volume-preserving nonrigid registration of MR breast images using free-form deformation with an incompressibility constraint. *IEEE Transactions on Medical Imaging*, 22(6):730–741, 2003.

[164] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes. Nonrigid registration using free-form deformations: Application to breast MR images. *IEEE Transactions on Medical Imaging*, 18(8):712–721, 1999.

[165] D. Rueckert, A. F. Frangi, and J. A. Schnabel. Automatic construction of 3D statistical deformation models of the brain using nonrigid registration. *IEEE Transactions on Medical Imaging*, 22(8):1014–1025, 2003.

[166] D. Ruijters, B. M. ter Haar-Romeny, and P. Suetens. Efficient GPU-accelerated elastic image registration. In *Proceedings of the 6th IASTED International Conference on Biomedical Engineering*, pages 419–424, Innsbruck, Austria, 2008.

[167] O. Salvado and D. L. Wilson. Removal of local and biased global maxima in intensity-based registration. *Medical Image Analysis*, 11(2):183–196, 2007.

[168] O. Scherzer, editor. *Mathematical Models for Registration and Applications to Medical Imaging*. Number 10 in Mathematics in Industry. Springer, Berlin, 2006.

[169] D. F. Shanno. Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, 24:647–656, 1970.

[170] D. Shen and C. Davatzikos. HAMMER: Hierarchical attribute matching mechanism for elastic registration. *IEEE Transactions on Medical Imaging*, 21(11): 1421–1439, 2002.

[171] D. J. Sheskin. *Handbook of parametric and nonparametric statistical procedures*. Chapman and Hall/CRC, New York, NY, USA, 2nd edition, 2000.

[172] H. Y. Shum and R. Szeliski. Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision*, 36(2):101–130, February 2000.

[173] P. Y. Simard, Y. A. Le Cun, J. S. Denker, and B. Victorri. Transformation invariance in pattern recognition: Tangent distance and propagation. *International Journal of Imaging Systems and Technology*, 11(3):181–197, 2000.

[174] M. V. Simkin and V. P. Roychowdhury. Read before you cite! *Complex Systems*, 14(3):269–274, 2003.

[175] M. V. Simkin and V. P. Roychowdhury. Do copied citations create renowned papers? *Annals of Improbable Research*, 11(1):24–27, 2005.

[176] J. C. Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control*. Wiley-Interscience, Hoboken, New Jersey, USA, 2003.

[177] R. Sprengel, K. Rohr, and H. S. Steihl. Thin-plate spline approximation for image registration. In *Proceedings of the 18th International Conference of the IEEE Engineering in Medicine and Biology Society*, 1996.

[178] M. Staring. Contributions to the normalized correlation and the mean squares metric. *Insight Journal*, January-June 2006. January-June 2006 Issue `http://hdl.handle.net/1926/190`.

[179] C. V. Stewart. Robust parameter estimation in computer vision. *SIAM Review*, 41(3):513–537, 1999.

[180] C. Studholme, D. L. G. Hill, and D. Hawkes. Multiresolution voxel similarity measures for MR-PET registration. In Y. Bizais, C. Barillot, and R. Di Paola, editors, *Proceedings of the Conference on Information Processing in Medical Imaging (IPMI'95)*, pages 287–298, 1995.

[181] R. Szeliski. Image alignment and stitching: A tutorial. Technical Report MSR-TR-2004-92, Microsoft Research, December 2004.

[182] R. Szeliski and J. Coughlan. Spline-based image registration. *International Journal of Computer Vision*, 22(3):199–218, 1997.

[183] R. Szeliski and H. Y. Shum. Creating full view panoramic image mosaics and environment maps. *Computer Graphics*, 31(SIGGRAPH'97 Proceedings):251–258, 1997.

[184] B. M. ter Haar Romeny. *Front-End Vision and Multi-Scale Image Analysis*. Kluwer Academic Publisher, 2003.

[185] P. Thévenaz and M. Unser. Optimization of mutual information for multiresolution image registration. *IEEE Transactions on Image Processing*, 9(12): 2083–2099, 2000.

[186] P. Thevenaz and M. Unser. An efficient mutual information optimizer for multiresolution image registration. In *Proceedings of the International Conference on Image Processing (ICIP 98)*, 1998.

[187] A. W. Toga, editor. *Brain Warping*. Academic Press, 1999.

[188] A. W. Toga and P. M. Thompson. The role of image registration in brain mapping. *Image and Vision Computing*, 19(1-2):3–24, 2001.

[189] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing: Part I–theory. *IEEE Transactions on Signal Processing*, 41(2):821–833, 1993.

[190] E. B. van de Kraats, G. P. Penney, D. Tomazevic, T. van Walsum, and W. J. Niessen. Standardized Evaluation Methodology for 2D–3D Registration. *IEEE Transactions on Medical Imaging*, 24(9):1177–1190, 2005.

[191] M. Čapek and L. Poušek. Biomedical volume alignment using an efficient optimization method and fast data resampling. In *Proceedings of the 3rd IEEE International Symposium on Signal Processing and Information Technology (ISSPIT 2003)*, pages 483–486, 2003.

[192] P. Venkataraman. *Applied Optimization with MATLAB Programming*. John Wiley and Sons, Canada, 2001.

[193] G. Villard. Computation of the inverse and determinant of a matrix. In F. Chyzak, editor, *Algorithms Seminar 2001-2002*, pages 29–32. INRIA, 2002. Available online `http://algo.inria.fr/seminars`.

[194] P. Viola and W. M. Wells III. Alignment by maximization of mutual information. In *Proceedings of the 5th IEEE International Conference on Computer Vision (ICCV1995)*, pages 16–23, Cambridge, MA, USA, June 1995. IEEE Computer Society.

[195] N. Vlassis, R. K. Elhorst, and J. R. Kok. Anytime algorithms for multiagent decision making using coordination graphs. In *Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics*, volume 1, pages 953–957, October 2004.

[196] M. P. Wachowiak and T. M. Peters. High performance derivative-free optimization applied to biomedical image registration. In I. Kotsireas and D. Stacey, editors, *Proceedings of the 19th International Symposium on High Performance Computing Systems and Applications (HPCS'05)*, pages 50–56, Guelph, Ontario, Canada, 2005. IEEE Press.

[197] M. P. Wachowiak and T. M. Peters. High-performance medical image registration using new optimization techniques. *IEEE Transactions on Information Technology in Biomedicine*, 10(2):344–353, 2006.

[198] F. Wang and B. C. Vemuri. Non-rigid multi-modal image registration using cross-cumulative residual entropy. *International Journal of Computer Vision*, 74(2):201–215, 2007.

[199] J. Weickert, A. Bruhn, T. Brox, and N. Papenberg. *A Survey on Variational Optic Flow Methods for Small Displacements*, pages 103–138. In Scherzer [168], 2006.

[200] J. Weidendorfer, M. Kowarschik, and C. Trinitis. A tool suite for simulation based analysis of memory access behavior. In *Proceedings of the 4th International Conference on Computational Science (ICCS 2004)*, Krakow, Poland, 2004. URL `http://valgrind.org`.

[201] E. W. Weisstein. Diffeomorphism. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/Diffeomorphism.html.

[202] P. Whaite and F. P. Ferrie. Autonomous exploration: Driven by uncertainty. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):193–205, 1997.

[203] B. Widrow. The rubber mask technique, parts I and II. *Pattern Recognition*, 5 (3):175–211, 1973.

[204] R. P. Wildes, D. Hirvonen, S. Hsu, R. Kumar, W. Lehman, B. Matei, and W. Zhao. Video georegistration: Algorithm and quantitative evaluation. In *Proceedings of the 8th IEEE International Conference on Computer Vision (ICCV2001)*, volume 2, pages 343–350, Vancouver, Canada, July 2001. IEEE Computer Society.

[205] R. P. Woods, J. C. Mazziotta, and S. R. Cherry. MRI-PET registration with automated algorithm. *Journal of Computer Assisted Tomography*, 17(4):536–546, 1993.

[206] T. J. Ypma. Historical development of the Newton-Raphson method. *SIAM Review*, 37(4):531–551, 1995.

[207] C. Zhu, R. H. Byrd, and J. Nocedal. L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization. *ACM Transactions on Mathematical Software*, 23(4):550–560, 1997.

[208] B. Zitova and J. Flusser. Image registration methods: A survey. *Image and Vision Computing*, 21(11):977–1000, 2003.