# Deformable Image Registration for Use in Radiotherapy

Using GPU Acceleration

Karsten Østergaard Noe

PhD Dissertation


Department of Computer Science
Aarhus University
Denmark

# Deformable Image Registration for Use in Radiotherapy

Using GPU Acceleration

A Dissertation
Presented to the Faculty of Science
of Aarhus University
in Partial Fulfilment of the Requirements for the
PhD Degree

by
Karsten Østergaard Noe
October 26, 2009

# Summary

This dissertation investigates methods for deformable image registration (DIR) in radiation therapy. DIR is the process of finding a point to point correspondence map between positions in one medical scan and positions in another scan. This is a necessary prerequisite for accurately evaluating the accumulated dose from a number of radiation therapy treatments. Both registration methods based only on image intensities and methods using mesh based elastic simulation of organs are treated. The intensity based methods investigated are the viscous-fluid registration method and two methods based on optical flow estimation. Two new elastic models for registration of meshes derived from organ segmentation are presented:

- A model for registration of solid organs like the prostate or liver. The biomechanical model used for this registration method is based on a non-linear elastic finite element model and driven by forces derived from a Euclidean distance field as well as forces working normal to the organ surface.

- A model for registration of the bladder. Here a 2D parameterisation is created of one of the organ surfaces. Vertices from a surface mesh of the other organ segmentation are restricted to stay in the 3D positions described in this parameterisation. After an initial projection into the 2D space, the positions of vertices are optimised based on relaxing a spring-mass model and landmark point matching.

Ways of combining intensity based methods with mesh based methods will be proposed.

Graphics processing units (GPUs) have in recent years been successfully used for accelerating a variety of computational problems, and the use of GPU computations plays a key role in reducing the computation time of the presented methods.

# Resumé

I denne afhandling udforskes metoder til deform billedregistrering til brug i forbindelse med strålebehandling af kræftpatienter. Formålet med billedregistrering er givet to medicinske scanninger at beregne en beskrivelse af, hvilke positioner i den ene scanning, der svarer til hvilke positioner i den anden scanning. En sådan registrering er nødvendig for at kunne give et nøjagtigt estimat af den samlede fordeling af stråledosis, som patienten har modtaget fra en række strålebehandlinger.

Der behandles både registreringsmetoder, som alene baserer registreringen på billedintensiteter samt metoder, der simulerer organdeformationer v.h.a. en elastisk materialemodel. Her repræsenteres organer enten som voluminer bestående af tetraeder eller som overflader bestående af trekanter.

De intensitetsbaserede metoder, som undersøges, er viscous-fluid-registreringsmetoden samt to metoder, der baseres på bestemmelse af optical flow. Herudover præsenteres to nye metoder til elastisk registrering, som baserer sig på tetraede- eller trekants-repræsentationer af segmenterede organer:

- En metode, der egner sig til registrering af fyldte organer såsom prostata eller leveren. Den anvendte biomekaniske model baseres på en ikke-lineær elastisk elementmetode. Registreringen drives af kræfter, som beregnes fra Euklidiske afstandsfelter, samt kræfter, der virker i normalretningen på overfladen.

- En metode til registrering af blærer. I denne model skabes en to-dimensionel parametrisering af overfladen fra den ene segmentering. En trekantsrepræsentaton af overfladen på den anden segmentering simuleres som en elastisk membran. Knuderne i denne bevæges rundt på overfladen af den første segmentering ved hjælp af parametriseringen. Herved minimeres den potentielle energi i membranen samtidig med, at prædefinerede punktkorrespondancer bringes til at stemme overens.

Der fremlægges ideer til, hvorledes de intensitetsbaserede metoder og metoder, som baseres på tetraede- eller trekants-repræsentationer, kan bringes til at arbejde sammen.

I de senere år er processorerne på grafikkort med succes blevet anvendt til at nedsætte beregningstiden på en lang række beregningsproblemer. I denne afhandling anvendes grafikprocessorer til at reducere kørselstiden for de præsenterede registreringsmetoder.

# Acknowledgements

Thanks to Ole Østerby for friendly guidance and thorough proofreading, Thomas Sangild Sørensen for many valuable technical discussions and practical advice, and Kari Tanderup for always being ready for answering question about clinical issues. Also thanks to Cai Grau for helping me become part of the research group on clinical oncology in Aarhus.

Thanks to Brian Bunch Kristensen and Jens Rimestad for cooperation on the implementation of the isosurface stuffing algorithm and the TLED FEM. Also thanks to Zeike Taylor for helping me debug. Thanks to Jesper Mosegaard for cooperation on the physics engine, Peter Trier for letting me borrow code for optimising rigid alignment of point sets, and Baudouin Denis de Senneville for cooperation on the optical flow methods. Also thanks to David Atkinson for making me feel welcome during my stay at the Centre for Medical Image Computing at the University College London.

I would also like to thank my clinical research partners Kari Tanderup and Jacob Lindegaard on the clinical case of intracavitary BT for treatment of cervical cancer, and Ulrik Vindelev Elstrøm on the head and neck cone beam CT case.

The work presented in this dissertation was partially funded by Varian Medical Systems, Inc, Palo Alto, CA, USA. Also the Danish Cancer Society supported the work with travel grants. Finally the project was supported by CIRRO - The Lundbeck Foundation Center for Interventional Research in Radiation Oncology and The Danish Council for Strategic Research.

*Karsten Noe,*
*Århus, October 26, 2009.*

# Contents

# Publications during the Ph.D.

### Journal papers

*Acceleration and validation of optical flow based deformable registration for image-guided radiotherapy.* K.Ø. Noe, B.D. de Senneville, U.V. Elstrøm, K. Tanderup, T.S. Sørensen. Acta Oncologica 2008; 47(7):1286-1293.

*Accelerating the Non-equispaced Fast Fourier Transform on Commodity Graphics Hardware.* T.S. Sørensen, T. Schaeffter, K.Ø. Noe, M.S. Hansen. IEEE Transactions on Medical Imaging 2008; 27(4):538-547.

### Conference full papers

*GPU accelerated Viscous-fluid deformable registration for radiotherapy.* K.Ø. Noe, K. Tanderup, J.C. Lindegaard, C. Grau, T.S. Sørensen. 16th Medicine Meets Virtual Reality Conference, Long Beach, USA. Stud. Health. Technol. Inform. 2008;132:327-32.

*An optimised multi-baseline approach for on-line MR-temperature monitoring on commodity graphics hardware.* B.D de Senneville, K.Ø. Noe, M. Ries, M. Pedersen, C. Moonen, T.S. Sørensen. Fifth IEEE International Symposium on Biomedical Imaging, Paris, France 2008: 1513-1516.

### Conference short papers

*A framework for shape matching in deformable image registration.* K.Ø. Noe, J. Mosegaard, K. Tanderup, T.S. Sørensen. 16th Medicine Meets Virtual Reality, Long Beach, USA. Stud. Health. Technol. Inform. 2008;132:333-35.

*The Visible Ear Surgery Simulator.* P. Trier, K.Ø. Noe, M.S. Sørensen, J. Mosegaard. 16th Medicine Meets Virtual Reality, Long Beach, USA. Stud. Health. Technol. Inform. 2008;132:523-525.

These publications can be found on the accompanying CDROM.

# Abbreviations

| | |
|---|---|
| API | application programming interface |
| ART | adaptive radiation therapy |
| BCC | body centered cubic |
| BT | brachytherapy |
| BVH | bounding volume hierarchy |
| CBCT | cone beam computed tomography |
| CPU | central processing unit |
| CT | computed tomography |
| DIR | deformable image registration |
| DRR | digitally reconstructed radiograph |
| EBRT | external beam radiotherapy |
| FFD | free-form deformation |
| GPGPU | general purpose computations on the GPU |
| GPU | graphics processing unit |
| GRV | geometrically resolved view |
| IGRT | image guided radiation therapy |
| IMRT | intensity modulated radiation therapy |
| ITK | Insight Segmentation and Registration Toolkit |
| MI | mutual information |
| MRI | magnetic resonance imaging |
| NCC | normalised correlation coefficient |
| NFG | normalised gradient field |
| OAR | organ at risk |
| PCA | principal component analysis |
| PDE | partial differential equation |
| PET | positron emission spectroscopy |
| RBF | radial basis function |
| ROI | region of interest |
| RT | radiotherapy |
| SDK | software development kit |
| SIMD | single instruction multiple data |
| SIMT | single instruction multiple threads |
| SSD | sum of squared differences |
| TLED | total Lagrangian explicit dynamic |
| TPS | thin plate spline |
| TRE | target registration error |

# Part I

# Background

# Chapter 1

# Introduction

The subject of this dissertation is methods for *deformable image registration* (DIR) in radiotherapy. It focuses on registration methods based purely on image intensities and methods relying on explicitly simulating elastic behaviour using a mesh representation of each organ.

## The structure of this dissertation

The dissertation is divided into four parts:

1. **Part I:** The purpose of this introductory part is to introduce the reader to the problem domain of the dissertation as well as the background tools used to address the problems faced. First an introduction to image guided radiotherapy will be given, motivating the need for deformable image registration. In particular the use cases will be presented which motivate the registration methods focused on in this dissertation. Following this the reader will be introduced to existing techniques for deformable registration. The introductory part will end with a short introduction to computation based on graphics programming units (GPUs).

2. **Part II** is about using GPU computation for accelerating image registration based entirely on image intensities. In particular work on accelerating the so-called viscous-fluid registration method and two methods based on optical flow estimation will be presented. The achieved reduction in processing time makes it practically feasible to perform a clinical evaluation of the methods.

3. **Part III** involves registration of mesh representations of organs. For solid organs like the prostate or liver a method is presented which uses an elastic finite element model to ensure a physically plausible registration between different organ deformations. Furthermore a method will be presented for registering surfaces of the bladder. This model is based on an elastic membrane model and a spherical parameterisation of organs.

4. In **part IV** the dissertation is concluded and a discussion is made about how to combine the methods presented in parts II and III to allow mesh

based registration to serve as extra guidance in cases where intensity based registration does not adequately succeed.

Parts II and III constitute the primary technical contributions of the dissertation. The presented registration methods will be validated on a "proof of concept" level. A full scale clinical evaluation is important future work.

Besides the work described in this dissertation, the author has been part of two other research projects. One is concerned with GPU acceleration of a non-Cartesian fast Fourier transform for use in reconstruction of magnetic resonance images [130], and the other is on the development of a surgical simulator for ear surgery [139]. The referenced publications can be found on the accompanying CDROM.

On this CDROM a number of movies are also found which further illustrate techniques and registration results presented in chapters 7 and 8. References to these movies will be made with a red number in curly braces like this: {1}.

# Chapter 2

## Image guided radiotherapy

*Radiation therapy* or *radiotherapy* is an essential part of the treatment of cancer diseases, which works by destroying the genetic material of cancer cells with ionising radiation. The radiation can be administered as *external beam radiotherapy* (EBRT) where a planned radiation dose is delivered from an external device. A technique called *intensity modulated radiation therapy* (IMRT) in which the radiation is shielded using a number of moving collimator leaves (see figure 2.1) can be used for conforming the radiation dose to a particular *target volume*. This conformation is based on a *CT planning image* that has under-

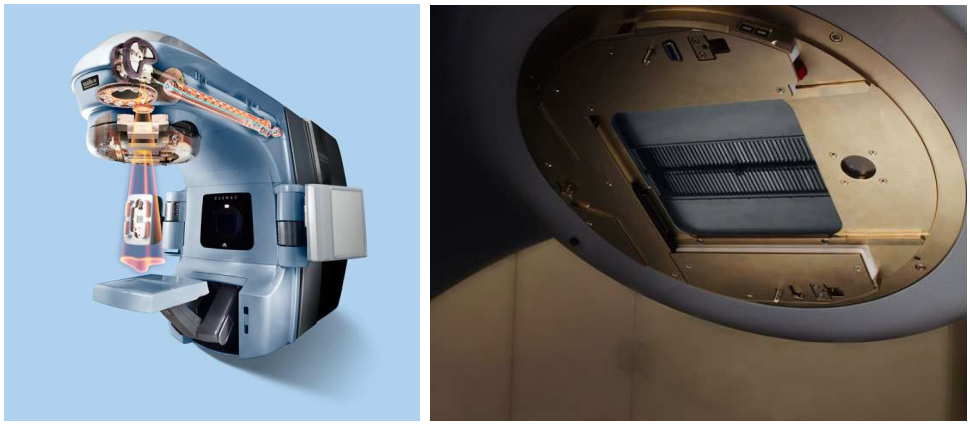

Figure 2.1: **Left:** A linear accelerator. The gantry rotates around the patient and delivers the planned 3D dose distribution. **Right:** Multileaf collimators work by shifting in and out of the beam during treatment allowing precise conformation of dose. Both images from `http://www.varian.com`

gone a (usually manual) *segmentation*. In this dissertation the word image is used to denote both 2D images and 3D data sets consisting of a number of 2D slices. Segmentation is the process of partitioning an image into a number of parts (organs and tumour volumes). This is done by contouring segments on the slices of the planning image. During *dose planning* the target volumes and *organs at risk* (OAR) to be spared are specified along with a prescribed dose, and based on this the planning software calculates the physical trajectory of the gantry and movement of the collimator leaves in a process called *inverse*

*dose calculation*. A variable number of treatment sessions, called *fractions*, are delivered.

Alternatively or supplementary to EBRT a technique called *brachytherapy* (BT) can be used in which a small source of radiation is placed inside the patients body. In so-called intracavitary BT an applicator designed for the particular cancer site is inserted in naturally occuring body cavities - see figure 2.2. The brachytherapy applicator has a number of tubes in which the radioactive

Figure 2.2: Photo of a brachytherapy applicator used for treatment of cervical cancer. The leftmost part is inserted in the cervix and via the tubes seen on the right a small radioactive source can be inserted.

source can be inserted and pulled out automatically using a machine called an *afterloader*. The administered dose is regulated by adjusting the dwell time during which the source is positioned at different dwell positions in the tubes. BT planning can be done on MR images which give a better contrast between different types of soft tissues. BT treatment is characterised by giving a very localised dose with steep dose gradients (compare the two parts of figure 2.3).

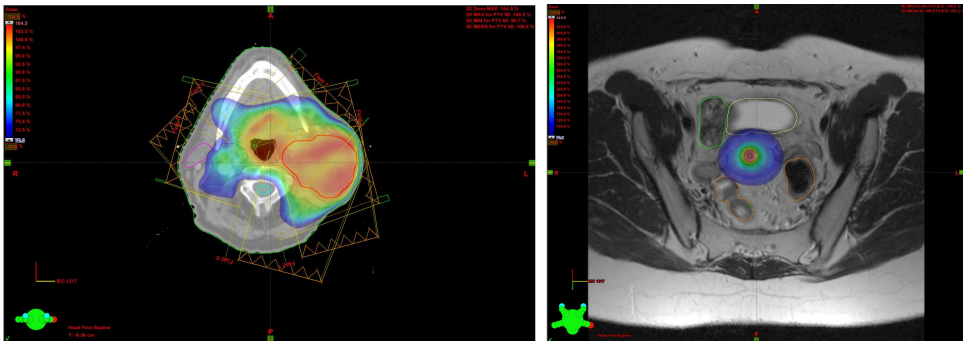

Figure 2.3: **Left:** Dose plan for a head and neck tumour computed by inverse dose calculations based on prescribed dose constraints to the segmented volumes. Coloured overlay shows the planned dose. The computed fields of radiation are also shown. **Right:** Dose plan for brachytherapy of cervical cancer. The delivered dose is very localised to the vicinity of the applicator.

Radiotherapy can cause severe side-effects, so the radiation dose should always be conformed to the diseased volume as well as possible. However, organ and tissue deformations between radiotherapy sessions represent a significant

challenge to this dose conformation. Margins are added to the tumour volume to ensure that the entire tumour receives sufficient dose in each treatment session. Currently, huge efforts are put into minimising this margin by applying different imaging modalities to guide the planning and delivery of radiation (IGRT: *image guided radiotherapy*).

Recent technological advances have made it possible to perform CT imaging in the treatment room in connection with each radiotherapy session using one of the technologies *cone beam computed tomography* (CBCT) [136] or *CT-on-rails* [7]. The device used in CT-on-rails is a conventional CT scanner which is located near the accelerator used for EBRT. Without leaving the couch the patient can be moved back and forth between accelerator and CT scanner. In CBCT the imaging device is mounted on the accelerator arm. Instead of a thin slice of radiation as used in a conventional CT scanner, a cone of radiation is sent through the patient. This causes a larger contribution of scattered radiation which results in more noise. An advantage of CBCT is that the patient does not need to be moved and is scanned in the frame of reference of the treatment beam. Besides these two technologies, increased access to magnetic resonance imaging (MRI), conventional computed tomography (CT), and positron emission tomography (PET) scanners has made it possible to perform repetitive scans during radiotherapy treatment. The repetitive acquisitions provide valuable information about organ deformation and movement over time as well as tumour shrinkage, which can potentially be used for ongoing dynamic dose optimisation of the treatment. To draw full advantage of this, advanced methods of image analysis are required to handle organ deformations.

## 2.1 Image registration in radiotherapy

Aligning images rigidly allows some changes in images to be easily detected. However such an alignment does not model changes from e.g. organ deformation, patient weight loss, or tumour shrinkage. It is possible to take such changes into account using *deformable image registration* (DIR) which is a method for finding the mapping between points in one image and the corresponding point in another image. DIR has the perspective of being widely integrated into many different steps of the radiotherapy process. The tasks of planning, delivery and evaluation of radiotherapy can all be improved by taking organ deformation into account. Use of image registration in IGRT can be split in the following categories:

***intra-patient registration*** is registration of images of a single patient. Uses can again be divided:

- ***inter-fractional registration*** is used for matching an image acquired at one treatment fraction to an image from another fraction or to the planning image. Possible uses include improving patient positioning, evaluating organ motion relative to bones, and enabling pointwise accumulation of dose.

***intra-fractional registration*** is for matching images acquired during a single treatment fraction. The use case here is online tracking of organ movement. An example could be tracking respiratory motion.

***inter-patient registration*** is used for matching images from different patients. This can be used for generating a model of organ motion across a population of patients. Another use of inter-patient registration is matching an image to an *atlas* i.e. an "average" of images acquired from a number of patients, thereby allowing information to be transferred from the atlas to the newly acquired image.

The process of combining information from two images after these have been registered is called *data fusion*. A particular use of data transfer between images is the propagation of contours from the planning image or an atlas to a newly acquired image [48] [154].

In this dissertation focus us put on intra-patient inter-fractional registration in connection with cervical cancer and head/neck cancer respectively as presented in section 2.5. For an introduction to other uses of image registration in RT we refer to the comprehensive survey paper by Sarrut [120]. In addition to inter-fractional registration for dose accumulation his survey covers data fusion in connection with MR spectroscopic imaging and PET acquisition as well as compensation for breathing deformation and inter-patient registration for atlas generation. For a broader introduction we refer to the survey by Kessler [66].

## 2.2 Aligning a series of images to a common reference system

By registering all 3D acquisitions from a series of treatments to the same reference data set, a geometrically resolved view (GRV [19]) can be generated making it possible to calculate the accumulated dose distribution (see below) and to evaluate tumour growth/regression [83].

### 2.2.1 Addition of dose distributions

Assume that we have a series of $M$ image acquisitions, each corresponding to a treatment fraction. Every image $i$ has a *dose matrix* $D_i(\mathbf{x})$ assigned to it. This dose matrix can originate from either an EBRT or BT dose planning, and is a description of the distribution of radiation dose given in the fraction corresponding to image $i$. Oncologists would like to know the accumulated dose distribution from multiple radiation treatments. In the accumulation process, deformation of the organs between fractions must be taken into account. The role of DIR is to generate the spatial transformations $\mathbf{h}_{ij}, \quad i, j \in \{1, 2, ..., M\}$. The expression $\mathbf{h}_{ij}(\mathbf{x})$ evaluates to the position in the source image $I_i$ that corresponds to the position $\mathbf{x}$ in the reference image $I_j$.

We also need to define a reference system $R \in \{1, 2, ..., M\}$. Given this we can add the dose distributions in the following way [28]:

$$D_{Total}(\mathbf{x}) = D_R(\mathbf{x}) + \sum_{i=1, i\neq R}^{M} D_i(\mathbf{h}_{iR}(\mathbf{x})) \tag{2.1}$$

This means that for each treatment we use the registration method to describe how the internal organs have been deformed compared to the reference image. By transforming the dose distribution using the transformation found by registration, the dose distribution is transferred to the reference system. Since the dose distributions for all treatments are transferred to the same reference system, they can be summed to find the accumulated dose distribution. Note that this model does not take into account biological effects of dose fractionation - for a more advanced accumulation model see e.g. [12].

#### Calculating the dose distribution for each fraction

For a course of EBRT treatments dose planning is typically done only once based on the planning CT. An assumption is made that the patient morphology will be the same for all fractions. However, in practise changes will occur due to e.g. weight loss or tumour shrinkage. Since the tissue at fraction $i$ has undergone a deformation compared to the planning scan, the dose distribution calculated during dose planning does not correctly correspond to the image acquired at this fraction. Therefore it is required to recalculate the dose distribution based on the same treatment parameters but on the new image[1]. For EBRT this recalculation must be done on CT images (or CBCT - see section 2.5.2). The CT modality measures the attenuation of radiation as the radiation goes through the body, which is exactly what we wish to compute in the dose calculation.

For BT dose gradients are so steep and localised that we may base the dose calculation on MR acquisitions (dose delivered is primarily a function of the distance from the source). In MR-guided brachytherapy an MR image is acquired in connection with each fraction and used for planning. The treatment is also here based on an assumption that only minor changes occur in the time between image acquisition and treatment.

## 2.3 Adaptive radiotherapy

In *adaptive radiotherapy* (ART) the idea is to use image feedback from the first $N$ fractions to reoptimise the treatment plan for the following fractions [12] [151] [83]. For now it is only realistic to achieve image guided adaptation off-line (i.e. using image information acquired during a fraction for improving following fraction), but a full on-line approach is also theoretically possible in which the treatment plan for a fraction is changed based on information from the same fraction.

[1] For some cancer sites recalculation may not be necessary [12]

## 2.4 Time requirements

The feasibility of introducing deformable registration into different steps of radiotherapy treatment planning depends on the computation time. *Pre-treatment* planning and *post-treatment* evaluation of radiotherapy are tasks that run over hours or days, but the registration time should still be kept reasonable low. A registration time of hours for instance will make a method unsuitable for clinical use.

When DIR is to be used in connection with the actual delivery the timing is critical, since the patient will be in the treatment room during the process of deformable registration. Considering the comfort of the patient, the accuracy of the treatment, and the patient flow, the registration time should be kept as low as possible.

## 2.5 Clinical cases

I will briefly describe two clinical cases that have been the motivation for choice of registration algorithms to be investigated. One of these is the use of intracavitary BT for treatment of cervical cancer. The other clinical case is registration between the planning CT and CBCT images acquired during head and neck cancer EBRT treatment fractions. For both cases the need for deformable registration will be motivated. It is out of scope of this dissertation to do a thorough evaluation of the proposed registration methods for the use cases below. Instead evaluation will be done on a "proof of concept"-level, demonstrating the methods on a few images from the cases.

### 2.5.1 Cervical cancer treatment - dose estimation

In Aarhus the standard treatment of locally advanced cervical cancer is a combination of EBRT and BT. The BT is based on a modern system of 3D image guidance using MR images. A 3D MR image is acquired in connection with every BT treatment fraction. On each such image, the target volume and organs at risk are delineated, and the position of the brachytherapy applicator is determined. Subsequently the treatment is planned. Knowing the dwell positions and dwell times it is possible to compute the delivered dose for each voxel of the acquired MR image.

We wish to be able to register all the MR images to the same (e.g. the first) image, which will allow us to estimate the accumulated dose of all fractions. One use of this information would be to relate delivered doses at different positions to side effects observed after a treatment course and the rate of survival. Another use of an improved assessment of dose (especially to highly mobile organs like the sigmoid or the bladder) would be to use the accumulated dose for improved dose optimisation. Having determined a low accumulated dose to an OAR, it might be possible to increase the dose delivered to the diseased tissue. On the other hand if a high accumulated dose to an OAR is observed, it is important to spare this tissue if possible. Figure 2.4 shows an example where the sigmoid colon has undergone large deformations between fractions.

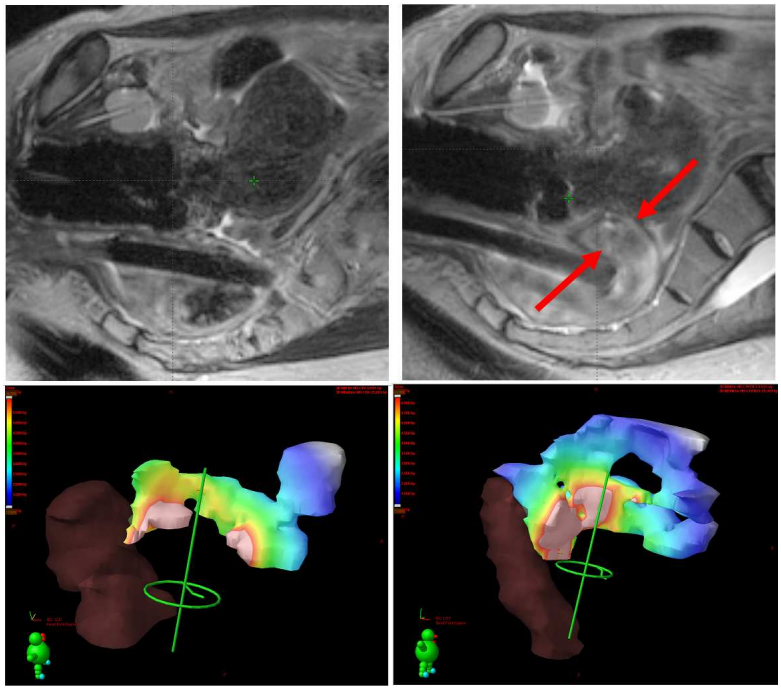

Figure 2.4: MR images acquired in connection with 1.st (top left) and 3.rd (top right) BT fractions. Below are 3D reconstructions of organ delineations in the respective images. The rectum is shown in brown and the sigmoid colon is coloured according to the radiation received. Positioning of the BT applicator is also shown. Arrows in the top right image highlight that the sigmoid has moved in between the rectum and the uterus very near applicator. From [134].

As an example of what extra information a reliable registration can give we can consider the so called $D_{0.1cc}$ for an OAR. This number describes the minimum dose given to the 0.1 $cm^3$ of most irradiated volume, and gives an impression of "hot spot" radiation to the organ. If computed on multiple images independently, worst case calculations assume that the hot spots occur at the same positions and thus the $D_{0.1cc}$ values should be added. However, given a reliable registration of images we can get a more accurate impression of hot spot radiation. If the highly radiated spots occur at different positions, higher doses may be allowable in future fractions.

### 2.5.2 Head and neck cancer treatment

In connection with EBRT of head and neck cancer patients in Aarhus, CBCT images are routinely acquired. These CBCT images contain artifacts and deviations from the Hounsfield Units (the quantity measured in the CT modality) of common scanners due to differences in scattered radiation - see figure 2.5. There is some controversy about the effect on dose calculations of the differences in Hounsfield units. Ding et al. report that the discrepancies in Hounsfield units do not give rise to major errors in dose calculations [38]. However, Yang

et al. find that dose calculation performed on CBCT images is not acceptable in the face of organ motion during acquisition [153]. Although this latter study is not on head and neck images, we would like to apply their approach of using DIR for mapping Hounsfield units from conventional CT images to CBCT images, which will make it possible to quantify the impact of using CBCT instead of CT for estimating dose distribution. It would be very useful to be able to do automated registration of CT and CBCT images. Besides elimination of image artifacts from the CBCT acquisitions it will enable tracking of accumulated dose to organs at risk (which can deviate from the planned doses due to tumour shrinkage/weight loss). Finally such a registration will also allow auto segmentation of structures of interest by contour propagation from the planning image.

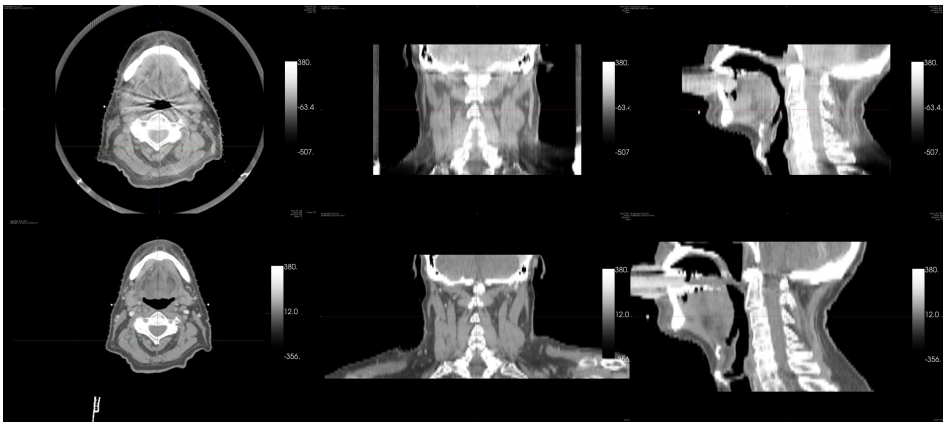

Figure 2.5: Images for comparison of conventional CT versus CBCT on images from a head and neck cancer patient. **Top:** axial, coronal and sagittal views of CBCT images. **Bottom:** axial, coronal and sagittal views of conventional CT images

# Chapter 3

## An introduction to image registration

The problem of registering medical images has been the subject of a huge number of previous research projects. These have resulted in a variety of different registration algorithms. In medical image processing in general there are three typical uses for 3D registration methods:

- Tracking of deformable objects in a series of medical scans (e.g. CT or MRI).
- Matching of images from different patients (inter-patient registration).
- Multi-modal registration which means matching images of the same patient acquired by different imaging technologies.

The registration problems treated in this dissertation are mainly belonging to the first of these uses[1]. This means that our task is to find a mapping between two images of the same patient acquired at different times describing the trajectory of each physical point.

*Deformable registration* is an ill-posed problem because there is generally no unique solution to a registration problem. Usually image registration is formulated as an optimisation problem. Following [22] image registration can be defined as finding the functions $h$ and $g$ in the following mapping between two 3D images $I_1$ and $I_2$:

$$I_2(x, y, z) = g(I_1(h(x, y, z))) \tag{3.1}$$

where $I_1$ is called the *source image* and $I_2$ is called the *reference image*. The images $I_1$ and $I_2$ can be thought of as $\mathbb{R}^3 \rightarrow \mathbb{R}$ mappings from 3D coordinates to image intensities.

The function $g$ is called an *intensity mapping function* that accounts for a difference in image intensities of the same object in $I_1$ and $I_2$. In other words it is used to describe so-called *photometric differences*. Difference in intensities is addressed in work on multimodal registration (see section 3.1) and in work using a more explicit model for intensity differences [108] [32].

[1]However, registration of cone beam CT and conventional CT may also be considered a multimodal registration problem.

The function $h$ is used to describe *geometric differences*. It is a *spatial 3D transformation* that describes the mapping between the spatial coordinates $(x, y, z)$ in the reference image and coordinates $(x', y', z')$ in the source image so that $(x', y', z') = h(x, y, z)$. These transformations take different forms depending on the registration method used.

Registration methods can be based on information derived from image intensities or from landmark information (such as contours or points) placed on the images. Also hybrid models are possible using a combination of intensities and landmarks.

## 3.1 Similarity measures

Before we can evaluate the quality of a registration result or even perform the registration we need to define what we mean by a pair of images being "more identical" after registration. It is usually not the case that images are perfectly matched as described by equation 3.1. Instead a *similarity measure* is defined, and the optimal registration is the one that features a transformation which minimises this measure. In the following I will introduce a number of the most popular similarity metrics.

### 3.1.1 Sum of squared differences

Perhaps the most widely used similarity metric is the sum of squared differences (SSD) measure defined as:

$$\mathcal{D}_{SSD}[I_S, I_R; h] = \iiint_{\Omega} |I_S(h(\mathbf{x})) - I_R(\mathbf{x})|^2 dV \tag{3.2}$$

The notation $\mathcal{D}_{SSD}[I_S, I_R; h]$ should be understood as "the SSD measure of similarity between the transformed source image $I_S$ and the reference image $I_R$ given the transformation $h$". The advantages with this metric is that it is simple and intuitive. Also it is fast to compute (it is O(n)). The main problem is that it requires images to be matched to be of the same modality (e.g. both must be MRI images acquired using the same protocol).

### 3.1.2 Cross Correlation

Another well known similarity metric is the *normalized cross correlation* (NCC) metric which can be written as:

$$\mathcal{D}_{NCC}[I_S, I_R; h] = \frac{\iiint_{\Omega} I_S(h(\mathbf{x}))I_R(\mathbf{x})dV}{\sqrt{\iiint_{\Omega} I_S^2(h(\mathbf{x}))dV \iiint_{\Omega} I_R^2(\mathbf{x})dV}} \tag{3.3}$$

This metric has the advantage that it has a reduced dependence on linear scaling of image intensities. This means that two images can be registered even though one is brighter than the other.

### 3.1.3 Mutual Information

For multimodal registration the most commonly used similarity metric the *mutual information* (MI) metric. It was proposed independently by Viola and Wells [143] and Maes et al. [86]. The metric is based on information theory and is thus a statistical measure of the correspondence between images.

The most intuitive way of expressing the MI measure between two images $I_1$ and $I_2$ is

$$\mathcal{D}_{MI}[I_1, I_2] = H(I_2) - H(I_2|I_1) \tag{3.4}$$

where the Shannon entropy $H$ is defined as:

$$H(I) = \sum_i \left[ p(i) \log_2 \frac{1}{p(i)} \right] \tag{3.5}$$

where $i$ runs over all intensity levels. The second part of (3.4) is the entropy involving the conditional probability that a pixel has intensity $i_2$ in $I_2$ given the information that the corresponding pixel in $I_1$ has intensity $i_1$.

The MI metric can also be written as

$$\mathcal{D}_{MI}[I_1, I_2] = \sum_{i_1, i_2} \left[ p(i_1, i_2) \log_2 \frac{p(i_1, i_2)}{p(i_1)p(i_2)} \right] \tag{3.6}$$

which is easier to compute. Here $p(i_1, i_2)$ is the probability of intensites being $i_1$ in $I_1$ and $i_2$ in $I_2$ at corresponding pixel positions.

### 3.1.4 Similarity measures based on normalised gradient fields

In [50] Haber and Modersitzki proposed basing a similarity measure suited for multimodal registration on *normalised gradient fields*. The idea is to consider two images as similar if intensity changes occur at the same position in both images. The image gradients are normalised and regularised which in 3D can be described as follows:

$$n_\varepsilon(I, \mathbf{x}) = \frac{\nabla I(\mathbf{x})}{||\nabla I(\mathbf{x})||_\varepsilon}, \quad \text{where} \quad ||\mathbf{x}||_\varepsilon = \sqrt{x_1^2 + x_2^2 + x_3^2 + \varepsilon^2} \tag{3.7}$$

Here $\varepsilon$ is a parameter thresholding the scale of intensity changes that we are interested in. An automatic choice for $\varepsilon$ can be computed based on image intensity gradients [50]. Based on the regularised normalised gradient field two similarity measures can be formulated:

$$\mathcal{D}_{NGFc}[I_1, I_2] = \frac{1}{2} \int_\Omega d^c(I_1, I_2, \mathbf{x}) \mathrm{d}\mathbf{x}, \quad d^c(I_1, I_2, \mathbf{x}) = ||n_\varepsilon(I_2, \mathbf{x}) \times n_\varepsilon(I_1, \mathbf{x})||^2 \tag{3.8}$$

$$\mathcal{D}_{NGFd}[I_1, I_2] = -\frac{1}{2} \int_\Omega d^d(I_1, I_2, \mathbf{x}) \mathrm{d}\mathbf{x}, \quad d^d(I_1, I_2, \mathbf{x}) = \langle n_\varepsilon(I_2, \mathbf{x}), n_\varepsilon(I_1, \mathbf{x}) \rangle^2 \tag{3.9}$$

The cross product in (3.8) means that this measure is related to the square of the sine of the angle between normalised gradient vectors which we wish to

minimise. The inner product in (3.9) is related to the cosine of the angle, so here we wish to maximise the square of the cosine.

## 3.2 Transformation functions

Methods for medical image registration can be divided into those using parametric transformation functions and those using non-parametric ones. The parametric methods are characterised by featuring a transformation function that is described by a quite limited number of parameters. In contrast to this non-parametric methods typically feature a transformation function that is based on a vector per voxel describing the displacement of the point represented by this voxel. This is converted to a continuous function by interpolation. In the following subsections we will look at some examples of commonly used transformation functions.

### 3.2.1 Parametric transformation functions

A registration method based on a parametric transformation function is usually formulated as a minimisation problem in which an optimal set of parameters must explicitly be found that minimises the chosen similarity measure.

Often *landmark information* (also called *features*) is used in connection with parametric transformation functions. Typically we wish to construct a smooth mapping based on a collection of $N$ point landmarks $\mathbf{p}_i$. These are to be matched by the transformation to another collection of reference landmarks $\mathbf{q}_i$, where the points $\mathbf{p}_i$ and $\mathbf{q}_i$ are corresponding points.

#### Global transformation

In registration problems where the volumes to be registered are translated and/or rotated with respect to each other, a rigid function suffices. A simple use would be the minimisation of the squared Euclidean distances between two paired sets of points subject to a rigid transformation function. A little more generality is obtained when using affine transformations, which have the property of preserving parallel lines and can describe e.g. shear and scaling. Global transformation functions can also be based on high order polynomials or Fourier series.

The matching of point features in source and reference images can be done manually by a medical expert based on fiducial markers placed before image acquisition or image features extracted from images after scanning. When using fiducial markers point matching is sometimes possible to do automatically using a method like the iterative closest point (ICP) algorithm [156].

#### Transformations based on radial basis functions

A more powerful way to describe the geometric transformation is creating a global function based on a set of *radial basis functions* (RBF), which are functions depending only on the distance between two points. *Thin plate splines*

(TPS) are such radial basis functions that are derived from minimisation of a smoothness measure based on the partial derivatives of the transformation [14]. For intuition one can think of a thin metal plate that is being bent by point constraints. This plate will arrange itself in a configuration in which the bending is smoothly distributed.

A number of other basis functions for RBF-based transformations have been proposed for image registration including *multiquadratics*, *inverse multiquadratics*, *elastic body splines* (EBS), and Gaussian functions. The reader is referred to e.g. [120, section 3.1] for references. Fornefett et al. used *Wendland functions* which have similar shape as a Gaussian function but compact support [44]. This means that the deformation resulting from matching a landmark only influences the transformation in the vicinity of this landmark. Kohlrausch et al. developed a variant of EBS which are based on a Gaussian. These are called *Gaussian elastic body splines* (GEBS) [72]. Like EBS, GEBS come from an analytic solution of the Navier equation which will be presented in section 3.3.3. In chapter 8 shifted log functions [89] will be used for interpolating surface displacements from organ surface to its interior.

An interesting example of the use of feature matching and RBF interpolation in registration for radiotherapy is described in a recent paper by Osorio et al. [105]. Here feature matching and TPS approximation is solved simultaneously using an iterative method called thin plate spline robust point matching. Point features are automatically generated from triangulated surface representations of organs. Their model allows registration of multiple organs simultaneously and inclusion of extra user specified point or line features.

### Transformations based a grid of control points

Another approach to parameterising a transformation using basis functions is to base the transformation on a number of control points arranged in a regular grid. An often used technique is cubic B-splines, which are defined using the four basis functions [118]:

$$\begin{aligned} B_0(u) &= (1-u)^3/6 \\ B_1(u) &= (3u^3 - 6u^2 + 4)/6 \\ B_2(u) &= (-3u^3 + 3u^2 + 3u + 1)/6 \\ B_3(u) &= u^3/6 \end{aligned} \tag{3.10}$$

Using these it is possible to develop a transformation function which is locally controlled using an $n_x n_y n_z$ grid of control points $\phi_{i,j,k}$ with uniform spacing:

$$h(\mathbf{x}) = \sum_{l=0}^{3}\sum_{m=0}^{3}\sum_{n=0}^{3} B_l(u)B_m(v)B_n(w)\phi_{i+l,j+m,k+n} \tag{3.11}$$

where $i = \lfloor x_1/n_x \rfloor - 1$, $j = \lfloor x_2/n_y \rfloor - 1$, $k = \lfloor x_3/n_z \rfloor - 1$ , $u = x_1/n_x - \lfloor x_1/n_x \rfloor$, $v = x_2/n_y - \lfloor x_2/n_y \rfloor$, and $w = x_3/n_z - \lfloor x_3/n_z \rfloor$.

This technique is called *free form deformation* (FFD)[2]. That the transformation is locally controlled means that when a control point is moved the points in the vicinity are transformed. Also the compact support of the B-Splines means that when evaluating the effect of moving a control point, only the vicinity of this point needs to be considered. In [118] a cubic B-spline FFD transformation approach is applied in registration used in mammography for breast cancer. This registration is based on MR images and using MI as similarity measure for creation of external forces. The internal energy expression is composed of elastic, stiff and continuous terms. Here an atlas based term is also used. The FFD approach combined with a MI similarity metric has been used by multiple authors in the field of radiotherapy see e.g. [90] [122] [153]. Another set of basis functions that can be used for FFD are Bernstein polynomials used for Bézier curves [131].

#### Mesh based models

Some models for image registration are based on dividing the entire image into polygons (2D) or polyhedra (3D), where the subdivision follows boundaries in the images (usually the subdivision is finer near boundaries too). This can be used for *finite element analysis* [115] [52] [51] or spring-mass based registration [91] [125] (see section 8.2).

A number of authors have based their registration approach on using organ segmentations for creating a mesh of points connected by triangles (organ surface), tetrahedra or hexahedra (entire organ volume). This approach is called *model based registration* by some authors. Usually an elastic model is used. Further discussion of mesh based registration is postponed until the 'related work' sections of chapters 7 and 8.

### 3.2.2 Non-parametric transformation functions

As mentioned above non-parametric transformations are typically described by a field consisting of a displacement vector per voxel of the reference image. A continuous transformation function is defined by interpolation between these vectors. The grid based displacement vector field representation constitutes a vast number of degrees of freedom.

### 3.2.3 Topology of the Transformations

A desirable quality of a transformation function is that it is homeomorphic. This means it is continuous, one-to-one and onto. A property of homeomorphic transformations is that they have a unique and continuous inverse transformation.

That the transformations are continuous ensures that adjacent structures are still adjacent after the transformation. A continuous transformation concatenated with another continuous transformation results in a third continuous transformation.

[2]Note that some authors use the term free form deformation for non-parametric transformations

To check whether a transformation $h$ is a homeomorphism it is sufficient to check that the Jacobian of $h(\mathbf{x})$ is positive for all $\vec{x} \in \Omega$. For $h(\mathbf{x}) = [h_1(x_1, x_2, x_3), h_2(x_1, x_2, x_3), h_3(x_1, x_2, x_3)]^T$ the Jacobian $J$ is defined as:

$$J(\mathbf{x}) = det(\nabla h|_{\mathbf{x}}) = \begin{vmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \frac{\partial h_1}{\partial x_3} \\ \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial x_2} & \frac{\partial h_2}{\partial x_3} \\ \frac{\partial h_3}{\partial x_1} & \frac{\partial h_3}{\partial x_2} & \frac{\partial h_3}{\partial x_3} \end{vmatrix} \tag{3.12}$$

Checking whether or not the Jacobian is greater than zero can be used to determine if the transformation is locally one-to-one in each point $\mathbf{x}$. This is a prerequisite for the transformation to be globally one-to-one.

## 3.3 Optimisation methodologies

Deformable image registration is in general an ill-posed problem. For example, there can be many fields of displacement vectors in a nonparametric registration resulting in the same deformed image and thereby the same cost value as calculated by the chosen similarity measure. Therefore the similarity metric is usually combined with a regularisation term.

For parametric transformations the regularisation is often achieved using a combination of a regularisation energy term on the parameters and the properties of the parameterisation function itself. Other transformations (like RBFs) function as interpolators and work by providing a smooth interpolation of prescribed displacements (the matching of landmarks).

For non-parametric methods the smoothness of the resulting transformation relies entirely on the regularisation chosen. For some non-parametric methods the regularisation imposed is an implicit result of a search strategy instead of a term included in the cost function to optimise.

### 3.3.1 Hierarchical approaches

Most practical implementations of image registration methods utilise some kind of Hierarchical *coarse-to-fine* approach. Several possible approaches exist [77]:

**multiresolution approaches:** The deformation is first approximated on low resolution versions of the images to be registered. The result of this coarse registration is then used as a starting point for a registration at a higher resolution. This continues until the deformation has been approximated at the highest resolution. A multiresolution strategy enables us to systematically handle modes of deformation at different scales. By finding a minimising transformation at a low resolution first we have a better chance of avoiding local minima at a higher resolution. Often the downsampling of images is combined with an image filter like a Gaussian filter.

**Gaussian scale space:** Involves the convolution of the input images using Gaussian filter with successively increasing standard deviation (while preserving image resolution). Again the idea is to do initial registration on

images with few details, and then iteratively increasing the amount of detail.

**increasing complexity of the transformation function:** this is especially well suited for transformation functions which consist of a number of basis functions like a Fourier series, where the first terms describe large scale image information and further terms describe increasing levels of detail. Here the initial level of registration can estimate coefficients of the first terms which can then be fixed while further levels estimate the next coefficients.

**increasing complexity of registration methods:** This entails consecutively using different registration methods. For example most deformable registration methods require an initial global (e.g. rigid or affine) registration to be made.

### 3.3.2 Methods for parametric registration

For parametric methods a number of numerical methods can be used for optimisation of the cost function. Among the most popular are the gradient descent (GD), conjugate gradients (CG), the Levenberg-Marquardt algorithm (LMA), the Broyden-Fletcher-Goldfarb-Shanno (BFGS) and its limited-memory counterpart (L-BFGS). As the optimisation performed by the registration methods in this dissertation are either physically inspired gradient descent or derived from the calculus of variations, the difference between the optimisation methods mentioned above will not be treated here. Instead the reader is referred to [95]. A key ingredient in efficent optimisation of a cost function is how efficient it is to compute the derivative of the cost function with respect to each of its parameters. If these derivatives cannot be found analytically they may be estimated using finite difference approximations.

### 3.3.3 Methods for non-parametric registration

Where transformations in parametric approaches to some extent are regularised by the continuous nature the parametric functions, regularisation is crucial when using non-parametric transformations. In this section some examples will be given of non-parametric registration methods.

#### Elastic matching using linear elastostatics

A method for registration using a model of a linear elastic continuum was presented in [4]. This model is based on finding an equilibrium between external forces applied to the elastic continuum and internal forces that arise from elastic properties being modeled. The external forces applied are derived based on local similarity of voxels in the images. The internal elastic forces are based on a model in which small local elastic deformations are assumed, and where a linear relationship between the deformation and restoring forces is also assumed. An external force $\mathbf{b}$ is applied. This leads to the *Navier equation* which is a partial differential equation (PDE) used for regularisation of the registration:

$$\mu\nabla^2\mathbf{u}(\mathbf{x}) + (\lambda+\mu)\nabla\left(\nabla\cdot\mathbf{u}(\mathbf{x})\right) + \mathbf{b}(\mathbf{x}) = \mathbf{0} \tag{3.13}$$

where the vector $\mathbf{u}$ is the displacement vector used in the description of the spatial transformation, and $\lambda$ and $\mu$ are material constants. This equation must be solved for every voxel in the image volume. Christensen and Johnson used the linear elastic model combined with a Fourier series parameterisation of the transformation for developing an inversely consistent registration [26].

### Demons

The so-called demons registration method was introduced by Thirion in [138]. Optical flow (see section 3.3.3) is used to find a driving force at each point based on the intensity gradient of the image. The allowed transformations are described using a vector field where each voxel has an associated deformation vector describing where this voxel is mapped to in the reference image. To regularise the flow a Gaussian filter is used. The voxel based demon method is described in detail in [137]. In [144] this method is validated on a suite of CT images from multiple cancer sites. In this work they achieved a speedup of the registration by 40 % by reformulating the driving forces in a way that uses gradient information from both source and reference images.

### Viscous-fluid registration

A registration method designed to handle large geometric displacements between two images is the *viscous-fluid registration* method by Christensen et al. [27] [29]. We will investigate this method further in chapter 5 The general idea in this method is to use a motion model that is derived from continuum physics and describes the motion of a viscous fluid for regularising the registration process. Using this model homeomorphic mappings can be achieved even for image registrations that require large deformations to be described. The driving force in the viscous-fluid registration is a *body force* vector field that is derived on the basis of image intensities finding the local direction of steepest decrease of an SSD similarity measure.

The method is very time consuming because it requires an iterative solution of a partial differential equation (PDE) and in each iteration another PDE must be solved to find a vector field of velocities. In [27] they obtained registration times in the order of 9-10 hours for a 128x128x100 volume on a so-called massively parallel (MasPar) supercomputer. They used the numerical method of successive over-relaxation (SOR) to solve for the velocity vector field. In [16] and [18] Bro-Nielsen et al. present a method that speeds up the calculation of the velocity vector field using a convolution filter based on an eigenfunction basis of a linear operator that (when applied to velocities) describes a relationship between velocities and body forces. A comparison is made between the demons method and the viscous-fluid registration method using Bro-Nielsen's filter for finding the velocities. Bro-Nielsen points out that driving forces are very similar and both methods use explicit Euler time integration. Thus the

main difference is the use of a Gaussian filter for regularisation instead of his filter for solving for velocities. He proceeds to show that the Gaussian filter is a low-order approximation to his filter. In [146] registration time of Bro-Nielsens method is compared to other methods for solving the viscous-fluid PDE also taking the accuracy of the solution into consideration. Here it is noted that due to the need for a considerable size of Bro-Nielsen filter to get an accurate solution, the numerical method of successive over relaxation (SOR) is faster in some cases. A fast implementation is achieved in [35] through the use of a full multi-grid solver for solving the viscous-fluid PDE and use of this solution method to also achieve a multiresolution implementation.

In [28] the viscous-fluid registration method is extended to include the use of landmark information. A hybrid model is presented in which regions of interest are converted to binary volumes. These volumes are included when body forces are calculated which makes it easier to guarantee that important structures in the images are matched. This hybrid model is used to match images from patients being treated for cervical cancer. For some patient material the method does not completely succeed in registering the images due to ill-posed deformations of organs. To remedy this, a purely landmark based approach called *fluid-landmark registration* [30] is used as a preprocessing step where matching is done on the basis of strategically placed points.

### HAMMER

An advanced registration method that has caught some attention, especially for brain registration, is the "hierarchical attribute matching mechanism for elastic registration" (HAMMER) proposed by Shen et al. [124]. Besides intensities and edge information, registration here is based on a segmentation of images into different types of tissue. This is used as input for a characterisation of voxels based on *geometric moment invariants* which is a way of characterising geometric properties of objects. The method is hierarchical both in terms of a multiresolution implementation and an iterative increase in the number of driving voxels used for point matching between images. Local minima are avoided by using driving vectors with the most distinct sets of attributes first and gradually adding points until all voxels are used.

### Optical flow based registration methods

The process of estimating optical flow means finding a quantitative measure of how image intensity information has changed between two images. Technically both images are regarded as part of one mathematical function where spatial changes have occurred in the time between acquisitions transforming one image into the other. The optical flow is a vector field consisting of the changes in space coordinates. These vectors can be thought of as 'optical velocity' vectors showing the direction of image intensity flow. Most optical flow estimation methods originate from the computer vision discipline, and as a result many methods are only formulated in two dimensions. Furthermore occlusion of objects behind other objects in a 2D frame of video is a difficult issue. This

problem does not exists in the context of medical image registration.

A well known method for estimating optical flow is the Horn and Schunck algorithm [56]. Here the optical flow field is found by minimising a cost function which consists of an intensity term and a term penalising non-smooth flow fields. The optimisation is based on the calculus of variations. Because the Horn and Schunck method performs a global optimisation it is able to produce very smooth transformations. The method was used for estimating intrathoracic tumour motion by Guerrero et al. [48].

In [152] an invertibility cost term is added to the Horn and Schunck method. This is used for obtaining inverse consistent registration (that is registration of source image to the reference is the same as the inverse transformation of the reference to the source). In this paper Yang et al. also formulate an inverse consistent version of the Demons algorithm.

A different approach than the global optimisation performed by Horn and Schunck was taken by Lucas and Kanade [84]. Here an assumption of constant flow in a window around a pixel being considered is made. This leads to an overdetermined system which can be solved by the least squares method. The Lucas and Kanade method leads to a registration result which is of a more local nature - that is information about displacements at edges does not propagate through areas of uniform intensity.

The two optical flow estimation methods mentioned above as well as the Demons method by Thirion have become the basis of a wealth of variants. In chapter 6 we focus on two optical flow based methods for deformable image registration; a 3D version of the Horn and Schunck method and the extension of this algorithm proposed by Cornelius and Kanade [32] to handle intensity changes that do not arise as a direct consequence of geometric motion, i.e. intensity variation due to physical properties of the acquisitions themselves.

A survey on methods for optical flow estimation is out of scope here. The reader is instead referred the survey by Beuchemin and Barron [10] and to the papers by Barron et al. [8] and McCane et al. [92] in which a comparison of optical flow methods are made in a computer vision context.

#### Other variational methods

Other groups have also used the calculus of variations in their work on image registration. E.g. Lu et al. used calculus of variations to represent the minimisation of their registration cost function as a set of elliptic partial differential equations [82], and validate the method on lung and prostate CT images. Zhang et al. used variational methods for registration of head and neck images [154]

## 3.4 Concluding remark

Images of different anatomical sites of the human body and different imaging modalities pose different problems to the registration method. Registration of pelvic MR images is something completely different from registering e.g. CT brain images. No registration method exists that is suitable for all applications, and the choice of registration method for a particular problem must be based

on which assumptions can be made about the images to be registered and the nature of the deformation expected.

# Chapter 4

# Using graphics hardware for scientific computing

## 4.1 Introduction

The *graphics processing unit* (GPU) of modern graphics cards provides a high ratio between computational power and cost making it an attractive platform for general computations. Modern graphics cards have a highly parallelised architecture that enables multiple calculations to be processed simultaneously leading to very fast processing rates. The key to utilising this parallelisability is the formulation of a numerical problem in a SIMD (single instruction multiple data) fashion. This means that the same calculations must be performed on a large amount of data elements. The SIMD architecture means that the GPU is well suited for accelerating e.g. numerical computations formulated on a regular grid like the voxels of a medical 3D image.

When we started using GPU processing power for accelerating image registration the only way to do this was to access the hardware through a graphics library like OpenGL or DirectX. This was done by mapping general computations onto concepts from computer graphics (section 4.2). During this Ph.D.-project the field of *general purpose computation on the GPU* (GPGPU) has matured enormously lead by the involvement of the major graphics hardware producers. The leading of the emerged frameworks for GPGPU is called CUDA. The earliest work presented in this dissertation is based on GPGPU through the OpenGL graphics API and a major part of the later work is based on CUDA. As a consequence GPGPU through both OpenGL and through CUDA will be presented. The chapter will end with a short survey of existing work on GPU accelerated image registration. Parts of the chapter come from [130].

## 4.2 GPGPU through graphics drivers

The first approach to programming the GPU for general computations was to be well aware of the fact that graphics cards and the accompanying software layers are developed to render computer graphics and map the required computations and memory model onto graphical concepts. In this approach GPU processing power for non-graphical applications was utilised through standard (OpenGL or

DirectX) graphics drivers. A major limitation to this way of using the GPU for calculations is that each thread of computation has a predetermined memory location where its resulting value will be stored, and it is not possible to output values to other memory locations. We say that it is possible to *gather* but not *scatter* data.

As will be presented in chapter 5 we have implemented the viscous-fluid registration method in graphics hardware through the use of ordinary 3D graphics drivers. In this section a very short introduction to conventional 3D graphics rendering will be given followed by an introduction to how this can be used for GPGPU. First the general notion of a graphics pipeline is introduced together with the fixed function pipeline, the standard implementation of the graphics pipeline. After this some programmability is introduced, which enables us to perform more general computations than just rendering graphics.

### 4.2.1 The fixed function pipeline

Traditionally the rendering of 3-dimensional objects into an image on a graphics card was done in a so-called *fixed function pipeline*, in which the application programmer had very little control over the data processing done on the GPU. For an overview of the graphics pipeline see figure 4.1, where the relationship between the three stages *application stage*, *geometry stage*, and *rasterisation stage* can be seen.

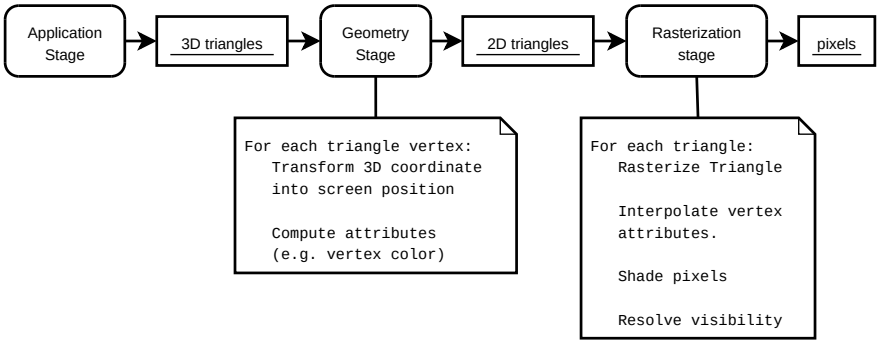


Figure 4.1: A general overview of a classical graphics hardware architecture.

In the application stage input for the graphics card is generated depending on what is to be rendered. The main input to the GPU consists of points (so-called vertices) and directions in 3D space (both represented as vectors) and textures. The input points can be used to specify vertices of polygons or lines to be drawn. Complex models of 3-dimensional objects can be made from hundreds or thousands of such polygons. The polygons are divided into triangles.

In the geometry stage the 3D points are projected from 3D space to 2D space to find out which pixels on the output image correspond to the input points. Also some per vertex attributes are calculated. For example the colour at the vertex is calculated. These attributes are to be interpolated over the triangles. The image being produced by the GPU consists of pixels. In the rasterisation

stage, the projected triangles are rasterized. The term *rasterisation* means filling in the triangles and colouring the pixels inside the triangle edges.

Often one or more textures are mapped onto the rendered triangles. Textures are images which are stored in *texture memory* on the graphics hardware. For each vertex of a polygon, texture coordinates can be specified to indicate a position in the texture to be mapped onto the vertex. Usually texture coordinates are specified in relative coordinates, in which the entire texture surface maps to the quadrilateral $[0, 1]^2$ The texture coordinates are interpolated over the triangle, which results in the image being stretched to match the projected vertices with these positions. The texture colours read from the interpolated position in the texture is combined with the interpolated colour to result in a final output colour.

### 4.2.2 Programmability on the GPU

With the introduction of *vertex programs* and *fragment programs*, programmers of 3D computer graphics gained a lot more power over the computations on the GPU. Vertex programs are pieces of code to be executed for each vertex, and fragment programs are pieces of code to be executed once for each *fragment*, which can be described as "candidate pixels". In the fragment program the colour of a pixel is calculated, but the fragment may later be discarded because it is determined to be non-visible.

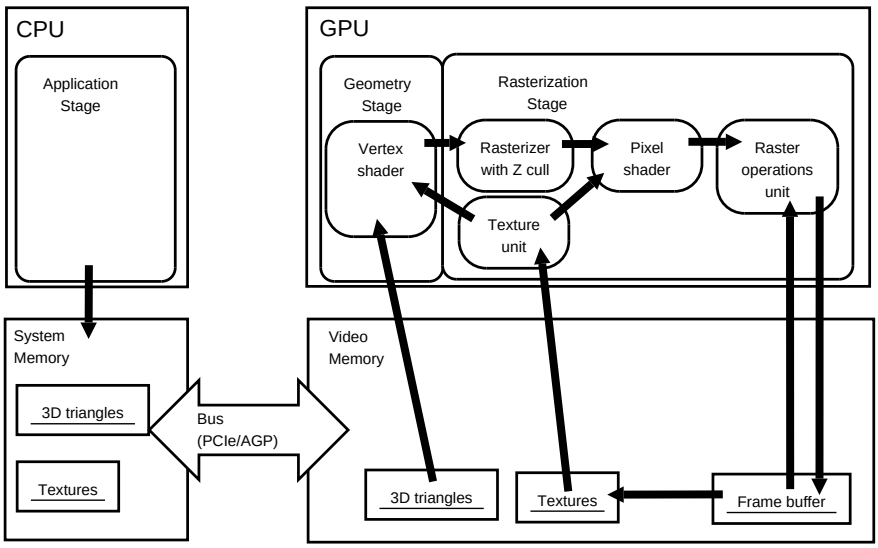


Figure 4.2: An overview of the GeForce 6 architecture which was the target platform of the OpenGL based implementation of the viscous-fluid registration method

The Nvidia GeForce 6 hardware platform which was the target architecture for the OpenGL based implementation of the viscous-fluid registration method (chapter 5) is outlined in figure 4.2. This generation of hardware supports shader model 3.0 which allows long vertex and fragment programs with advanced control structure. The geometry stage consists of a vertex shader that

is responsible for executing vertex programs. The rasterisation stage consists of a rasterizer, which fills out triangles, a pixel shader, which runs fragment programs, and a *raster operations unit*(ROP) which is responsible for blending semitransparent pixels into the frame buffer or simply writing opaque colours. A so-called *depth buffer* is maintained and used to check whether or not a pixel is visible. As can be seen on the figure, both vertex programs and fragment programs can read from texture via the *texture unit*. If fragment and/or vertex programs are not specified, a default (fixed function) implementation will be run.

### 4.2.3 Doing general computations on the GPU

The computational capacity of modern graphics hardware can be utilised for general computations by adopting a so-called *stream programming model*, in which fragment programs are used as computational kernels operating on streams of fragments stored in textures. This can be done through one of the graphics rendering APIs OpenGL or DirectX.

When one would loop over all data points in a CPU based implementation of a numerical calculation, the same is achieved in a GPU based version through graphics drivers by drawing a polygon or a line that covers the pixels corresponding to the data values that we wish to process. In this way rasterisation serves to invoke computation and the fragment program runs on all fragments in parallel.

Iterative computations can be achieved by using multiple render passes. Feedback from one iteration to the next is achieved by storing the resulting pixels from one rendering pass in a texture which can be used as input in a subsequent pass. By designating *texture coordinates* to vertices we can keep track of positions in textures to be read from in a fragment program. These coordinates are interpolated throughout fragments of each triangle and in this way the fragment program knows the relationship between the position in the grid of the data element being processed and the corresponding texture position. Of course textures containing results from other calculations can also be used as input.

#### Mapping data to textures

In many scientific computing applications we need to perform calculations on three-dimensional arrays. There are multiple ways of representing 3D arrays on the graphics card. The most commonly used option is to represent them as "flat 3D textures" [109], which are essentially two-dimensional textures on which the slices of 3D arrays are placed next to each other. Contrary to the alternative of using a dedicated 3D texture this method requires address translation to be made from 3D grid positions to 2D texture coordinates. The advantage with the flat representation is that it is possible to update the entire grid in a single render pass, something which is not possible for 3D textures.

**Different types of kernels**

Naturally, the types of computation performed in fragment program kernels depend on the GPGPU problem being solved. A simple use of a fragment program is to implement a map operation in which a function is evaluated on each element in a stream, returning a stream of the resulting values. In general the value returned from fragment programs depend on multiple values read from one or more textures. Each of these textures are uploaded from the CPU main memory or result from computations from earlier rendering passes. Kernels can also have other functionality than map operations. One possibility is to use them as filters, in which only a subset of the input stream is output. Another important use of kernels is *reduction operations*, in which multiple stream elements are combined into one. By running multiple passes reduction operations allow e.g. summation of all elements or location of the largest element in parallel without write conflicts.

## 4.3 Non-graphics abstractions for GPGPU

Following the interest in GPGPU the major graphics hardware producers have seen a new market in the use of their hardware. As a consequence of this the hardware vendors Nvidia and ATI[1] each released a more dedicated framework for doing scientific computing on graphics hardware. These frameworks are called CUDA and CTM respectively. These removed some of the previous limitations for doing scientific computations on the GPU. As a consequence of the new programming interfaces operating through graphics rendering APIs is no longer necessary.

ATI released *Close To Metal (CTM)* [3] which is an API allowing programmers to directly access the memory and processor of ATIs graphics hardware. We have used CTM for implementing a non-Cartesian fast Fourier transform (NFFT) [130]. Although more powerful than using OpenGL or DirectX, CTM was still cumbersome to use because of the low level programming needed. CTM was followed by the *ATI Stream SDK* which supplies higher level abstractions to the programmer.

Shortly after CTM was released the *Compute Unified Device Architecture* (CUDA) [103] was released by Nvidia. CUDA quickly became popular because of its powerful C-based abstractions which allowed the programmer to concentrate on the data parallel formulation of a computational problem instead of struggling with hardware related issues. As CUDA is the language of choice for most of the implementations presented in this dissertation, it will be described in more details in section 4.4.

*Brook* is an extension of ANSI C that implements a streaming computational model with abstractions for streams and kernels. Also reduction operations are supported. These allow for instance a summation of all elements in a stream. The first GPU based implementation of Brook is called BrookGPU [24]. While presenting the user with the Brook compiler and runtime systems, BrookGPU

[1] Now owned by AMD

can use both OpenGL, DirectX, and CTM as computational backend. In the ATI Stream SDK kernels are written in the Brook+ language which is an implementation of Brook with some enhancements.

With the high level abstractions the GPU can now be used for computation through high-level C-like programming languages without knowledge of graphics programming in general. In the CUDA and CTM frameworks scatter operations have been made possible, giving new possibilities when using the newest generation of graphics hardware for general computations. In the CUDA framework we furthermore have access to an amount of very fast memory shared between a number of kernels. Although some restrictions do apply in the two APIs, one can consider the GPU as a multiprocessor where each individual processor can read from and write to a shared memory pool.

A recent framework for GPGPU programming is the *open computing language* (OpenCL), which excels at being efficiently executable on both GPUs and CPUs. It is based on a C-based syntax and many of the central concepts in the framework is similar to ones in CUDA. With the project being supported by the major hardware producers Apple, AMD, Intel and Nvidia it has a good chance of becoming popular.

## 4.4 A closer look at CUDA

In this section an introduction to CUDA will be given. First the computational and memory concepts of the CUDA architecture are introduced followed by a look at the hardware realisation of this architecture. Finally we will look at how to program CUDA enabled devices and optimise computations for a larger throughput.

### 4.4.1 Kernels in CUDA

In CUDA kernels are written in C[2]. CUDA kernels have no return value but are allowed to gather and scatter values arbitrarily in GPU memory.

#### Thread hierarchies and kernel invocations

In CUDA the single instruction multiple data (SIMD) terminology has been replaced with the term *single instruction multiple threads (SIMT)*. Kernels are executed $N$ times in parallel in $N$ individual *CUDA threads* by providing an *execution configuration*. This configuration specifies the dimensions of 1-, 2- or 3-dimensional *thread blocks* which are again arranged in a specified 1- or 2-dimensional *grid* of thread blocks - see figure 4.3. Each thread has access to its position in the thread block called its *thread id* as well as the position of this block in the grid of blocks, called its *block id*. This information is used for determining which data elements correspond to each thread.

The configuration of threads into blocks has opened new possibilities for designing massively parallel algorithms. As will be described below threads

[2]Support for other high-level languages is planned in future versions of CUDA.

in a block can cooperate through a pool of *shared memory* with no latency. Furthermore threads in a block can be synchronised at specified synchronisation points. This is practical for example when all threads in a block cooperate on writing values in shared memory and all shared values must be written before these can be processed.

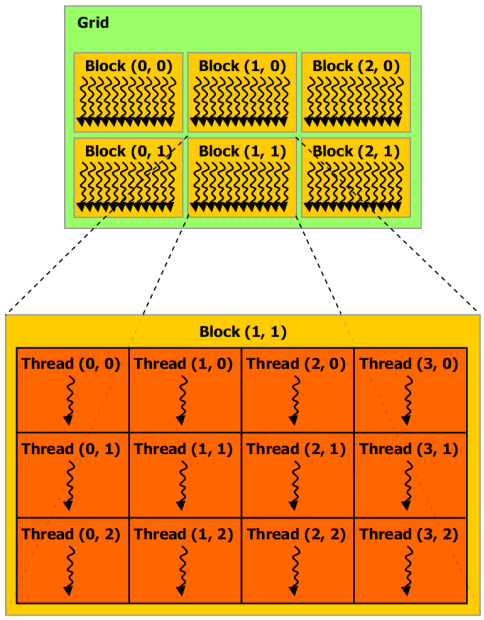


Figure 4.3: The configuration of 2D thread blocks in a 2D grid of blocks. From [103]

### 4.4.2 Memory model

In CUDA terms the main memory accessible from the CPU is called *host memory* and memory positioned on the hardware device supporting CUDA is called *device memory*. CUDA kernels have access to a number of device memory spaces. As the use of these spaces depends on latency of memory operations, the physical implementations on the hardware platforms supporting CUDA is important and this will be included in the presentation below.

***global memory*** consists of portions of device memory. A considerable latency is associated with accessing global memory. This memory can be accessed across multiple thread blocks in a single thread invocation (although we have no guarantees about avoidance of write conflicts). Also this memory is persistent over multiple kernel invocations.

***local memory*** is per-thread only memory which is used when a kernel cannot be compiled into only using GPU registers for computations. This can be because too few registers are available or because indexed memory access is needed which is not possible with registers. This memory is as slow to access as global memory.

***shared memory*** is a pool of memory which is shared between the threads in a block. Shared memory is on-chip memory and thus as fast as registers.

***constant memory*** contains values that can only be written to from code on the host side (code running on the CPU) through a runtime API. Constant memory is cached meaning that repeated access to a constant memory element is sped up considerably.

***texture memory*** is a remnant of the graphics concepts previously used for GPGPU. Texture memory can be configured in one-, two-, or three-dimensional arrays of various types of values. This memory is cached and depending on the configuration and data type, interpolation and boundary conditions can be handled in hardware.

### 4.4.3 Multiprocessor hardware

CUDA enabled hardware devices contain a number of *streaming multiprocessors* (SMs) which each contain eight scalar processors and a SIMT instruction unit (illustrated in figure 4.4). As mentioned each multiprocessor also contains on-chip shared memory. The thread hierarchy model maps to this hardware as one multiprocessor is responsible for the execution of all threads in a block, running each thread on its own scalar processor.

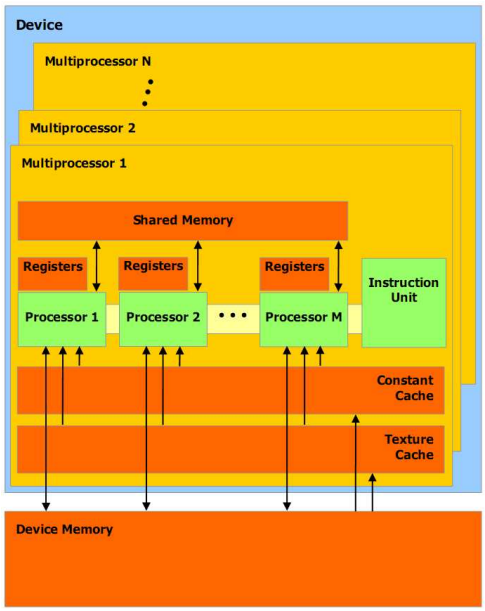


Figure 4.4: SIMT multiprocessors with on-chip shared memory. From [103]

### 4.4.4 Extensions to the C programming language

To allow programmers to execute parallelised code in CUDA a number of keywords have been added to the C programming language. Some of these allow specification of whether a C function is to be compiled for the host computer

architecture, the CUDA device, or both. Others are used for specification of which memory space to be used for storing a variable.

CUDA also provides a runtime library consisting of both host and device components as well as components available in both. The host runtime component provides functionality for management of CUDA devices, allocation and management of memory, invocation of device kernels, and interoperability with OpenGL/DirectX graphics APIs for visualisation of results.

The device runtime components provides fast but less accurate versions of common mathematical functions as well as the `__syncthreads()` function which allows synchronisation of all threads *in a block*. Besides this the library provides functions for reading from textures, atomic functions (read-modify-write operation without interference from other threads) and *warp vote functions*. The result of these vote functions called `__all` and `__any` is the same for all threads in a warp and based on the arguments given in all threads. The common component consists of built-in vector types and a number of functions from the standard C library.

Extra libraries provide access to GPU-based computational resources for common computational tasks including the CUBLAS library for linear algebra operations and the CUFFT library for fast Fourier transformations.

As a final comment, an invaluable advantage of CUDA compared to the old OpenGL based approach is the ability to compile device code to host code running in an emulation mode. This allows much more efficient debugging.

## 4.5 Optimising performance

An important concept when analysing the performance of a GPU-based algorithm is its arithmetic intensity. This is defined as the "amount of computational work" that is performed per off-chip (global) memory access. Applications with high arithmetic intensity are most likely compute bound while a low arithmetic intensity is an indication of a memory bound algorithm. To illustrate why this concept is important, figure 4.5 shows the execution time of six programs with different arithmetic intensities as a function of the number of instructions. The data making up the figure was obtained using the GPU benchmark suite GPUBench [23]. The figure is comprised of six subtests that perform one to six memory cache accesses each (solid lines). Notice the horizontal line segments. They show that for each memory access, a number of "free" computations can be made without influencing the overall execution time if they are independent of the result of the prior memory fetches. Only as the diagonal part of the graph is reached, there is a cost associated with issuing additional instructions.

From the figure, we can predict the execution time of an application consisting entirely of memory reads by following the dashed line. Notice that the slope of this line is much steeper than the slope of the diagonal solid line, which constitutes the border between a memory bound and a truly compute bound application. An application with an arithmetic intensity that places it between the dashed line and the solid diagonal line is memory bound, while an appli-

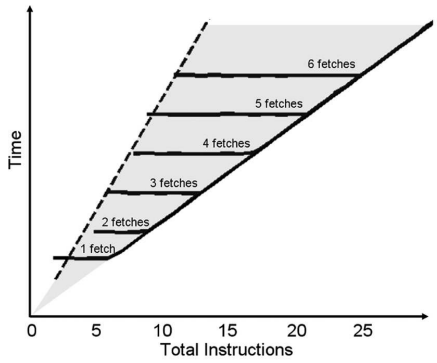


Figure 4.5: Memory access costs as a function of the number of shader instructions (solid lines). Number of memory fetches is varied from 1 to 6. Data was obtained using GPUBench [23] on an ATI Radeon X1800 XT.

cation with an arithmetic intensity that places it on (or close to) the diagonal would be compute bound. The performance of a compute bound application will grow with the rapid increase in arithmetic performance from each new generation of GPUs, whereas a memory bound application will increase speed as a function of memory bandwidth, which unfortunately is much slower growing. Whenever possible, one should use on-chip memory (e.g., registers or dedicated fast (shared) memory) to avoid the cost of expensive global memory fetches. As it is often impossible to interleave all memory fetches with computations independent of their results, the GPU APIs handle several concurrent threads on each processor. When a thread requires memory access on which the subsequent instruction relies, the processor will switch to a different thread to avoid being idle. Once the data arrives it can again switch back to the original thread. The number of concurrent threads is determined by the amount of on-chip storage for each processor. The reader is referred to [109] for further discussion of this topic.

To summarise the discussion above, it is crucial to minimise the overall number of global memory accesses for a given algorithm in order to obtain maximum performance. Ideally, each memory access should be interleaved with computations independent of its result to hide its cost completely. When at all possible, one should use on-chip memory such as registers for computations as there is no penalising cost associated with accessing these.

Specifically for CUDA additional guidelines are given in [103] on how to increase performance through optimising data layout and memory access patterns. Optimal strategies differ between access to global and shared memory. Another target for optimisation is the execution configuration, which must configured so that enough blocks run in parallel to utilise the available processing power.

## 4.6 Related work on image registration on the GPU

The earliest examples of work on using the computational power of a GPU to accelerate image registration involves using graphics hardware through OpenGL for creating *digitally reconstructed radiographs* (DRRs) which are essentially (2D) X-ray images constructed from a (3D) CT scan. This can be used for doing a rigid 3D-to-2D registration based on an X-ray image taken at the time of treatment and a pre-operative CT scan [76] [70]. Khamene et al. use a DRR approach in radiotherapy for registering 2 portal images to a planning CT [68] [67]. These first implementations only used the GPU for creation of the DRR. Ino et al. used GPGPU for DRR creation as well as evaluating the NCC similarity and gradients used for optimisation [57]. Kubias et al. base their GPU DRR registration on a similarity measure which is a combination of 8 known similarity measures all computed in parallel on the GPU [74]. Finally Grabner et al. have also implemented DRR-based 3D-to-2D registration on a GPU through OpenGL [45]. In this work *automatic differentiation* was utilized, which means that no finite differences are needed.

Soza et al. used graphics hardware for accelerating image deformation by Bézier curves used in an FFD-based registration approach [131]. Using a similar approach Levin et al. used graphics hardware for accelerating image deformation based on thin plate splines [78]. Hastreiter et al. based their registration transformation on piecewise linear functions and adaptive subdivisions of the grid of displacement vectors approximating a non-linear transformation [55]. In this work they use OpenGL functionality for accelerating the deformation of the source image prior to computation of mutual information.

In [132] Strzodka et al. presented a GPGPU based implementation of the 2D *regularised gradient flow* registration method [31]. In this work the DirectX API was utilised. Later this work was extended to 3D by Köhn et al. [69].

Vetter et al. [142] used OpenGL for the implementation of a deformable multi-modal registration method. Here the similarity measure based on mutual information was supplemented with the statistical measure *Kullback-Leibler divergence* between the current joint intensity distribution and prior information about the joint intensity distribution learned from earlier registrations [49].

The demons based method described in section 3.3.3 has been implemented on the GPU by Sharp et al. [123]. In their work graphics hardware was also used for accelerating the reconstruction of cone beam CT (CBCT) images. Their work was implemented in Brook. Courty et al. also published a GPGPU-based implementation of demons using recursive Gaussian filtering when regularising displacements [33]. Finally the Demons method was also accelerated on the GPU by Muyan-Özçelik et al. [98] [119]. In this work CUDA was used. Using shared memory allows for a 10 % faster registration than the Brook implementation by Sharp et al. [123] on the same hardware.

Li et al. have implemented the *Levenberg-Marquardt* algorithm which is a non-linear least squares optimisation method on the GPU through OpenGL [79]. This is used for optimising the SSD similarity measure subject to a Sobolev smoothing term based on a finite element discretisation. Also using OpenGL Rehman et al. used the GPU for solving the *optimal mass transport* problem

formulated as a first order PDE [117]. This problem was originally formulated as a minimisation of transport cost associated with moving a pile of soil from one site to another. This translates to image registration as finding the optimal transformation in terms of transformation displacements moving image intensities from one image to another. Modat et al. have used GPU computation through CUDA to realise a fast implementation of the FFD approach with MI as the similarity measure [94]. A parallel-friendly calculation of gradients from the MI measure was used.

## 4.7 Concluding remarks

Many numerical problems can be elegantly formulated in a SIMD manner. This is especially true for many 3D image processing tasks, where a voxel constitutes a natural unit for data parallelism. The large number of voxels in 3D images means that it is possible to keep all computational units occupied simultaneously, allowing highly efficient implementations. In this dissertation we have used GPU computation as a tool for enabling registrations to be made in a time frame which makes validation studies of the registration methods practical and which will be suitable for potential future clinical use.

# Part II

# Intensity based image registration

# Chapter 5

## GPU accelerated viscous-fluid registration

This chapter presents the implementation of the viscous-fluid registration method by Christensen et al. [28] on the GPU. As mentioned in section 3.3.3 a drawback with Christensen's method has been very long computation times. We will demonstrate that it is possible to implement the viscous-fluid registration method in a way that utilises the computational power of modern graphics hardware and hereby achieve significantly faster registration times. The prime motivation for using a mainstream graphics card is that it constitutes a very cost effective way of obtaining a high amount of computational power. The chapter is based on [101] and [102].

### 5.1 Input data

In our implementation of the viscous-fluid registration method both voxel intensities and *regions of interest* (ROIs) that are manually contoured can be used to calculate the driving potential. To describe this, an image set $T_i(\mathbf{x})$ is defined as in [28] for every image study $i$:

$$T_i(\mathbf{x}) = \begin{bmatrix} T_{i0}(\mathbf{x}) \\ T_{i1}(\mathbf{x}) \\ T_{i2}(\mathbf{x}) \\ \vdots \\ T_{iN}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \text{Gray scale (zero for } \mathbf{x} \text{ in masks)} \\ \text{Binary mask for organ 1} \\ \text{Binary mask for organ 2} \\ \vdots \\ \text{Binary mask for organ N} \end{bmatrix}$$

where $\mathbf{x} \in \Omega$

See figure 5.1 for an illustration. This way of representing images means that every image $T_i(\mathbf{x})$ contains $N+1$ parts. The first part is the gray scale intensities in the range from 0 to 1. The last $N$ parts are binary masks. Each of these are 1 for $\mathbf{x}$ that are inside the corresponding contoured ROI and 0 for other $\mathbf{x}$. The vector $\mathbf{x}$ is a position in the scanned volume $\Omega$.

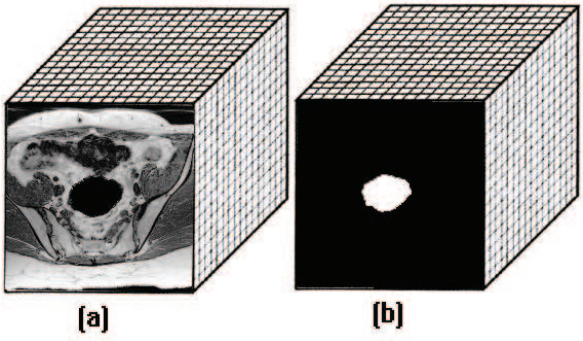


Figure 5.1: Illustration of the representation of images acquired in brachytherapy treatment. Grid-representation of the image can be seen in (a) and (b). (a) Grey-scale intensities with voxels that are inside regions of interest set to 0 (black colour). (b) Binary mask representing a uterus ROI. Voxels inside this ROI are set to 1 (white) and voxels outside are set to 0 (black). If more than one region of interest had been used, more of the binary masks would have been added.

## 5.2 Frame of reference

The description of a transformation between image study $i$ and $j$ is denoted $h_{ij}$. It is based on an *Eulerian frame of reference*, where values are associated with fixed spatial positions in the reference volume through which the "particles" of the source image move. In other words, the particles are tracked with respect to the final coordinates. Informally this means that for each voxel point in the grid that forms the resulting image of the registration, a vector is stored pointing back to the point in the source image, that has been deformed to this final coordinate. When using an Eulerian frame of reference, we are sure that every voxel in the resulting image is accounted for.

This frame of reference is in contrast with a *Langrangian frame of reference* in which the positions of the particles are tracked with respect to the source coordinate system. Here we are not sure that the positions of the particles being tracked correspond to pixel coordinates in the reference coordinate system.

The vectors pointing from the resulting image (described in the reference system) back to the source image constitute a displacement field $\mathbf{u}$ for each registration. The relationship between $\mathbf{u}$ and $h_{ij}$ is as follows:

$$h_{ij}(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\mathbf{x}) \tag{5.1}$$

By applying the material derivative to find the time rate of change of $\mathbf{u}$ we can find an explicit Euler equation for time integration:

$$\mathbf{u}(\mathbf{x}, t+\delta) = \mathbf{u}(\mathbf{x}, t) + \delta\mathbf{v}(\mathbf{x}, t) - \delta\sum_{i=1}^{3} v_i(\mathbf{x}, t)\frac{\partial\mathbf{u}(\mathbf{x}, t)}{\partial x_i} \tag{5.2}$$

where $\mathbf{v}$ is a velocity vector ($v_i$ denotes the $i$th component of $\mathbf{v}$) and $\delta$ is a small time step. This can be used to iteratively find the displacement field using a velocity field in every step. In the next section how to find this velocity field will be explained.

## 5.3 Motion model

Equation (5.2) provides us with a way of iteratively updating the displacement field using small time steps $\delta$. Between each update, a field of velocity vectors that describe the motion of particles must be calculated. These velocity vectors are calculated from a motion model derived from continuum mechanics describing the motion of a viscous fluid. The motivation for using this model is the need to be able to describe very complex local deformations while still allowing large deformations. The viscous-fluid model is used to perform a regularisation to ensure continuity and smoothness of the transformation. It is described by the following PDE of velocities [27]:

$$\mu\nabla^2\mathbf{v}(\mathbf{x},t) + (\lambda+\mu)\nabla\left(\nabla\cdot\mathbf{v}(\mathbf{x},t)\right) + \mathbf{b}(\mathbf{x},\mathbf{h}_{iR}(\mathbf{x},t))) = \mathbf{0} \tag{5.3}$$

where $\nabla^2$ is the Laplacian operator, $\nabla$ is the gradient operator, and $\nabla\cdot\mathbf{v}$ is the divergence of $\mathbf{v}$. The material constants $\lambda$ and $\mu$ describe the viscosity of the fluid. $\mathbf{v}(\mathbf{x},t)$ is the velocity of the "particles" of the image being deformed. The vector $\mathbf{b}(\mathbf{x},h_{iR}(\mathbf{x},t)))$ is the *body force* that is used as driving potential in the registration process. How to find the body forces will be explained in the next section.

Despite the regularisation performed, the viscous-fluid model allows large deformations by penalising large velocities instead of large displacements. Where the linear elastic deformation model assumes that the restoring force is proportional to linear deformation, the viscous-fluid model instead assumes that the restoring force is proportional to the *velocity* of the motion (compare eq. 5.3 with eq. 3.13), and large displacements are relaxed over time.

## 5.4 Driving forces

The driving potential in the fluid registration model (eq. 5.3) is a field of body forces $\mathbf{b}(\mathbf{x},h_{iR}(\mathbf{x},t)))$. The body forces are designed to minimize the following SSD similarity measure [28]:

$$\mathcal{D}[T_i,T_R;h_{iR}] = \sum_{k=0}^{N}\alpha_k\iiint_\Omega |T_{ik}(h_{iR}(\mathbf{x})) - T_{Rk}(\mathbf{x})|^2 dV \tag{5.4}$$

where $R$ denotes the reference image. The $\alpha_k$ values are weights describing the importance of the entries of $T$.

The body forces are found by minimising the Gâteaux derivative of the similarity metric resulting in [28]:

$$\mathbf{b}(\mathbf{x}, h_{iR}(\mathbf{x},t)) = -\sum_{k=0}^{N} \alpha_k (T_{ik}(h_{iR}(\mathbf{x},t)) - T_{Rk}(\mathbf{x})) \cdot \nabla T_{ik}|_{h_{iR}(\mathbf{x},t)} \tag{5.5}$$

The expression $\nabla T_{ik}|_{h_{iR}(\mathbf{x},t)}$ means the evaluation of the gradient *in the Lagrangian reference frame* of $T_{ik}$ at position $h_{iR}(\mathbf{x},t)$. From eq. 5.5 a direction of the force is found in which the deformation is locally pushed towards a lower value of the similarity metric.

## 5.5 GPU based implementation

In this section we show that the viscous-fluid registration method described in the previous sections can be implemented in a way that exploits the computational capabilities of a modern GPU.

The following subsections describe the discretisations used to evaluate the above equations from a grid based description of the vector fields involved and our mapping of these grid computations to graphics hardware.

### 5.5.1 Discretisation and algorithm

To solve the equations in the viscous-fluid registration method numerically on a spatial grid, suitable discretisations need to be made. In this work finite difference approximations are utilized.

By rewriting the expression in (5.3) and applying second order finite difference approximations, a system of linear equations on the form $A\mathbf{v} = -\mathbf{b}$ can be derived that describes an approximation to (5.3) on a 3D grid . Here the vectors $\mathbf{v}$ and $\mathbf{b}$ are lexicographical orderings of the components of all $\mathbf{v}$- and $\mathbf{b}$-vectors in the grid, and $A$ is a sparse, banded matrix. To find an approximate solution to this, we use the numerical method of Jacobi iteration. For each grid point (corresponding to a voxel in the scanned images) the corresponding velocity vector is found iteratively in a number of passes where the following update rule is used:

$$v_i^{(n)} = \frac{-b_i - \sum_{j \neq i} a_{ij} v_j^{(n-1)}}{a_{ii}} \tag{5.6}$$

In each pass a more accurate approximation to the correct solution is found by applying the update rule to velocity vectors from the result of the previous pass. At each grid point velocity vectors at all 26 neighbours in the grid need to be accessed for this update rule to be evaluated. Points on the boundary of the grid are handled by *sliding boundary* conditions allowing vectors to point along the boundary edge but not into or out of it. Second order finite difference approximations are also used for evaluating (5.2) and (5.5) on the grid. Here trilinear interpolation is used for looking up values in between grid points.

The method of Jacobi iteration is chosen because the update of each grid point in an iteration is independent on the updated value of all other grid points.

This makes it suitable for a massively parallelised implementation on graphics hardware.

When updating displacement vectors from the velocity field, it is not sufficient to just use a discretised version of (5.2) since singularities will arise after a number of iterations. To prevent this from happening, it is necessary to perform *regridding* [29] when the Jacobian gets near zero at any point. During regridding the deformation currently described by the displacement field is applied to the source image. The result of this is made the new source image used in the subsequent registration iterations.

An overview of the method can be seen in algorithm 1. Prior to the viscous-fluid registration an automated rigid registration is performed to account for a global displacement of images w.r.t. each other. This is currently done using the Insight Segmentation and Registration Toolkit[1].

Inspired by [29] a *perturbation field* is calculated in algorithm 1 for each time step. This field is used for checking the Jacobian without updating the displacement field, and it is also used to find a suitable time step $\delta$. The vectors $\mathbf{P}$ in the pertubation field is computed by:

$$\mathbf{P}(\mathbf{x},t) = \mathbf{v}(\mathbf{x},t) - \sum_{i=1}^{3} v_i(\mathbf{x},t)\frac{\partial \mathbf{u}(\mathbf{x},t)}{\partial x_i} \tag{5.7}$$

Notice that the Euler update rule in equation 5.2 can be written using the displacement field in the following way:

$$\mathbf{u}(\mathbf{x},t+\delta) = u(\mathbf{x},t) + \delta\mathbf{P}(\mathbf{x},t) \tag{5.8}$$

The step size $\delta$ is found by finding the largest vector in the perturbation field and it is chosen so the maximum displacement is at a fixed value of 0.5 voxels.

### 5.5.2 GPU implementation via OpenGL

Mapping of the required computations to a GPU is done using the fragment processor, where computations are performed on a per-voxel granularity. The individual steps of algorithm 1 are implemented as fragment programs. Each program is responsible for evaluating the results of one of the steps at the designated grid point corresponding to a single pixel. The parallelised execution of the fragment programs on the GPU allows for a substantial acceleration of the calculations. For an overview of textures and fragment programs used in the OpenGL based implementation, see figure 5.2. Numerical values used in the registration are stored in graphics memory in 2D textures using the flat 3D representation described in section 4.2.3. For storing vector values, 32 bit float RGB textures are used. These are textures with a Red, Green, and Blue colour component at each pixel. Each component is an IEEE floating point value with 32 bit precision. For storing the voxel intensities and regions of interest, 8 bit RGBA textures are used. These textures contain only a single byte for each colour component and are used for storing both the voxel intensities of the

[1] See `http://www.itk.org/`

**Algorithm 1**: ViscousFluidRegistration($S, R$): The viscous-fluid registration method (at a single resolution).

**input** : Source image $S$, Reference image $R$
**output**: Resulting displacement field
Let all vector fields be reset to zero vectors
$t \leftarrow 0$
**repeat**
  Calculate body forces
  Solve for instantaneous velocities
  Calculate the perturbation field
  From this perturbation field calculate a step size $\delta$
  **if** *regridding is needed* **then**
    Perform regridding
  **else**
    Update displacements using step size $\delta$
  $t \leftarrow t + 1$
**until** $t >= t_{max}$ *or* $\delta > \delta_{max}$ ;
**return** *displacement field*

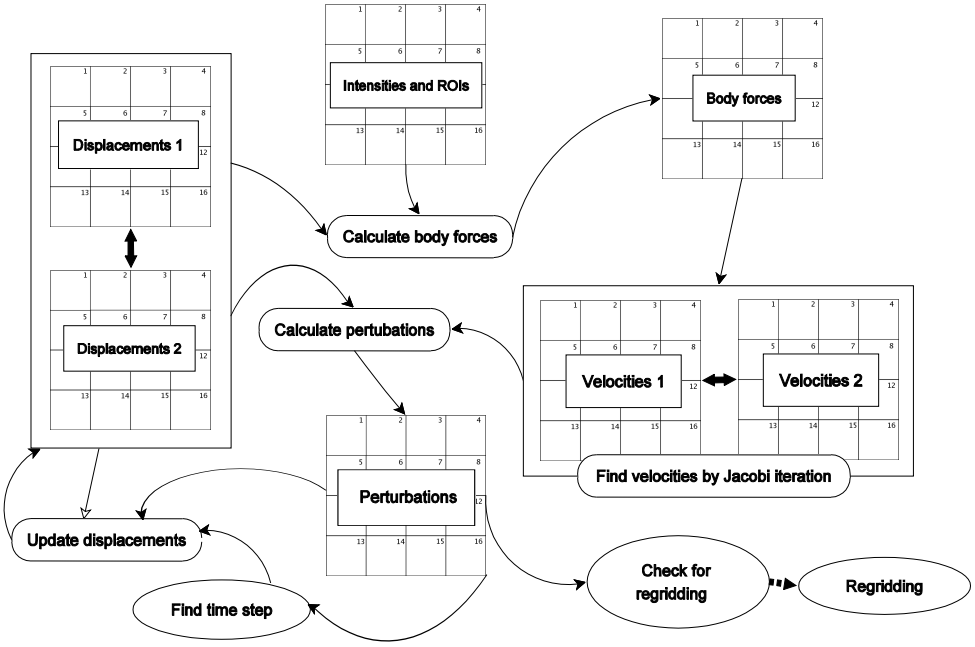


Figure 5.2: Dependencies between data textures and fragment programs in the OpenGL based implementation of the viscous-fluid registration method. Squares represent textures (subdivision represents a flat 3D representation containing 16 slices). Rounded rectangles represent fragment programs. Ellipses represent steps running on the CPU

source and reference images and the binary volumes of the regions of interest. In the red colour component, the intensity of the corresponding voxel of the source

image is stored. Equivalently the intensity of the reference volume is stored in the blue component. The green and alpha components are used to store the boolean values from the regions of interest of the source and reference images respectively. As the regions of interest are defined not to overlap, a unique byte value is designated to each region of interest, and this byte is stored in the texture. The regions of the source and target images that correspond to the same organ are given the same value.

We have used two approaches to converting from 3D grid positions to 2D texture coordinates. Values at arbitrary positions need to be looked up when calculating body forces. These positions are found through a look-up in the texture containing displacement vectors. To handle this the fragment program is given information about the number of slices to the left and right in the current row of slices in the texture. This information is used to make a local displacement without knowledge of the global position in the texture. In the fragment programs implementing the remaining steps of the viscous-fluid method, it is known which slices are needed in the computations. Therefore the corners of these slices are simply specified as vertex attributes when drawing the quadrilateral that is used to invoke computation of a slice. In this way coordinates are interpolated in hardware and the fragment program can know the texture coordinates of all neighbouring values.

Two textures are used for containing velocity vectors because the Jacobi iteration kernel uses one texture for reading values from the previous iteration and the other texture for storing the updated values. In this way values are read and written back and forth between the two textures for the desired number of iterations. A similar approach must be used when updating the displacement vectors.

## 5.6 Registration results

In this section the results of accelerating the viscous fluid registration method are presented. Timing experiments were performed in which the time required for registering images from radiation therapy on the GPU was compared to registration time on an identical CPU based reference implementation. Here "identical" means that this CPU version also uses Jacobi iteration for solving the viscous-fluid PDE. All optimisation flags were turned on in the compiler. However no hand optimisation was done on the CPU based code. This results in a comparison which is somewhat unfair because we have not taken advantage of the more general memory model in our CPU implementation. For a serial architecture several efficient methods for solving systems of linear equations exist. Also we do not pre-calculate image gradients. However, although the CPU implementation is not optimal the results of our timings can still be used as guide to the order of magnitude of the obtained speedup.

The timing experiments were performed on an Intel Core2 6400 running at 2.13GHz with 2 gigabytes of system memory and a Geforce8800GTX GPU. The registered image set was of resolution 256x256x32 with a single ROI specified. 100 Jacobi iterations were used per time step. The results can be seen in table

| Step | CPU | GPU | Speedup |
|---|---|---|---|
| Calculate forces | 5.62 | 0.02 | 236.6 |
| Solve for velocities | 86.31 | 0.54 | 161.2 |
| Perturbation field + update displacements | 2.49 | 0.13 | 18.7 |
| Evaluate Jacobian | 1.37 | 0.12 | 11.0 |
| Sum (ignoring regridding cost) | 95.79 | 0.82 | 117.2 |
| Average iteration (with regridding) | 99.04 | 1.14 | 87.1 |

Table 5.1: Timings of the different steps of the viscous-fluid registration method on a CPU and a GPU based implementation. Times are reported in seconds per iteration.

5.1. In case the evaluation of the Jacobian results in a regridding step this amounted to 4.72 sec. and 2.86 sec. of additional work on the CPU and GPU respectively. The average execution time per iteration in the bottom row of the table reflects the additional cost of regridding.

To demonstrate the feasibility of using the registration method for radiotherapy 3D images, the method has been applied to both MRI and CT images acquired in connection with radiotherapy treatment.

### 5.6.1 CT registration experiments

To investigate the ability of our implementation of the viscous fluid registration method to handle the CT image modality, registration experiments were performed on pairwise combinations of three 3D CT images acquired in connection with the treatment of a head and neck cancer patient. Slices of two images can be seen in figure 5.3 (upper left and lower left parts). Too large deformations exist in these images for a rigid registration alone to be sufficient. In the images depicted here, a coarse rigid registration was performed manually before the viscous-fluid method was applied. A total of 170 viscous-fluid time steps were used to register the two images of dimension 256x256x64. The result of the registration is illustrated in the upper and lower right parts of the figure. The registration method handles these CT images quite well. As can be seen in figure 5.3 the algorithm is successful in describing the difference in tumor size (arrow). In general the few head and neck CT registration experiments performed gave encouraging results.

### 5.6.2 MRI Registration experiments

We have also tested our implementation on magnetic resonance images. This was done by registering a number of 3D MRI images from cervical cancer patients.

In most cases the viscous-fluid registration was capable of handling the cervical cancer images reasonably well. However some images could not be registered by this method without the use of pre-segmented regions of interest (more on this below) and some registrations failed despite the use of ROIs. An example of a successful registration can be seen in figure 5.4. An automatic rigid

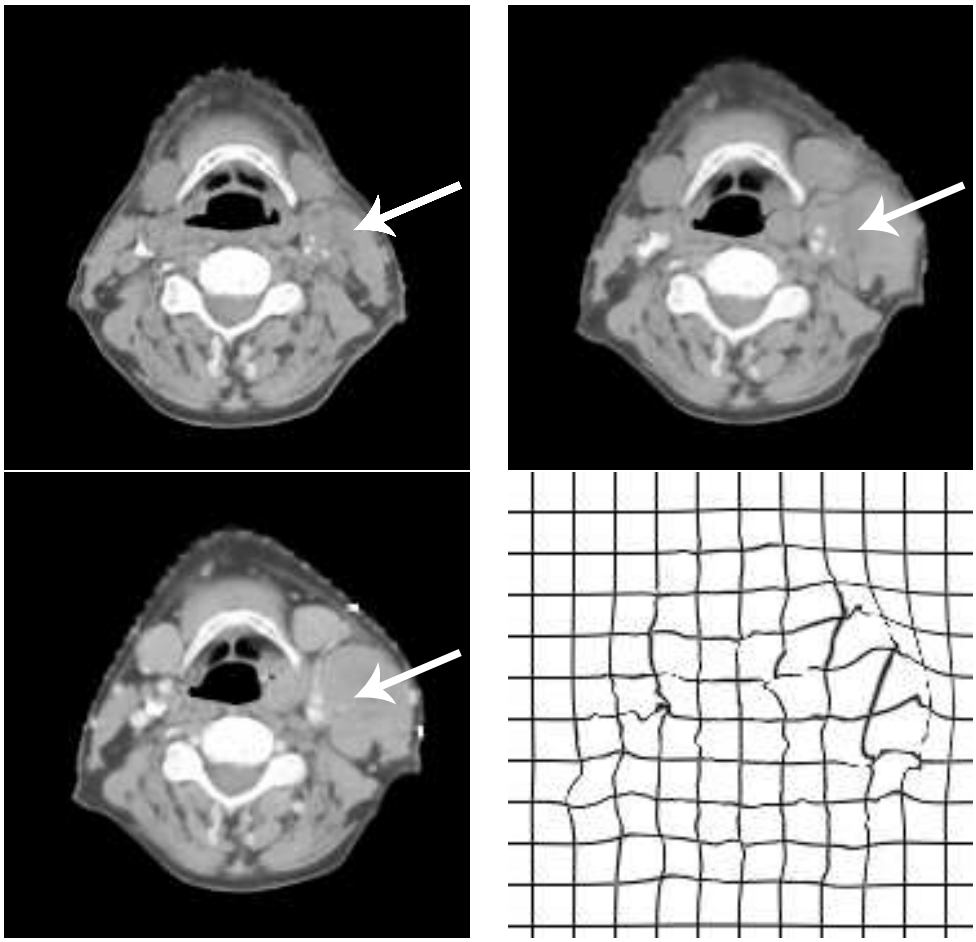

Figure 5.3: Images depicting the results of registering two CT images acquired in connection with the treatment of a head and neck cancer patient. All images depict the same slice. The upper left part depicts the undeformed source image followed by the deformed source image (upper right). The lower left part shows the reference image followed by an image displaying the result of applying the transformation to a rectilinear grid.

registration was performed based on image intensities before the deformable registration was started. 130 time steps were used for the shown registration. It can be seen that the difference in bladder positioning is nicely described by the method even without the use of regions of interest. In figure 5.4 the deformable registration has successfully described a difference in angling of the pelvic bones on the two images. This can not be seen on the deformed grid because this only displays in-plane deformations.

The cervical cancer MR images pose a more difficult task for the registration method than the above head/neck CT image because of larger organ motion and since the assumption of an identity intensity mapping is often violated due to differences in intestine content. An example of this can be seen on the left hand side images of figure 5.5. For registering these 256x256x32 images 110 time

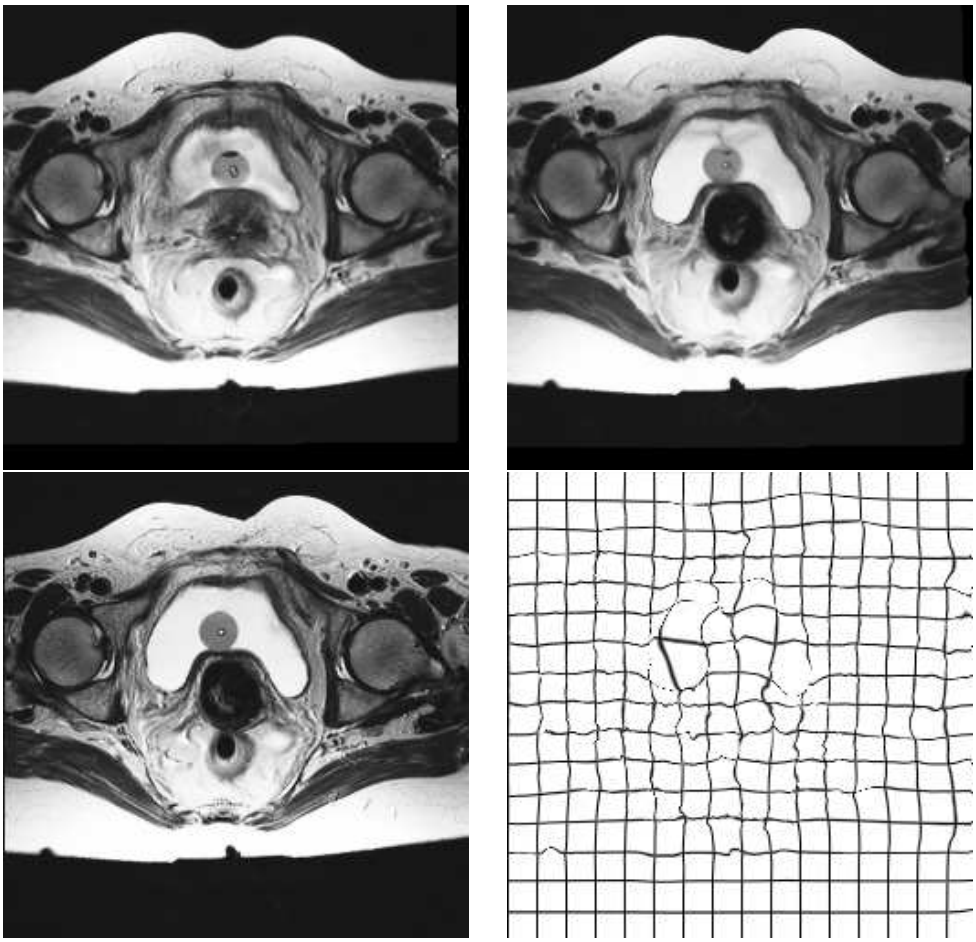

Figure 5.4: Images depicting the results of registering two MR images acquired in connection with the treatment of a cervical cancer patient. For a description of the different parts of the figure, see the caption of figure 5.3.

steps were used. Looking at the results on the right hand side of figure 5.5 it can be seen that the uterus has been wrongly expanded to fill out a surrounding piece of intestine with a similar intensity (arrow). In this example the use of a region of interest describing the position of the uterus will be advantageous because the algorithm can use this prior information to distinguish between organs.

The clinical registration experiments above can be hard to evaluate because no gold standard exists describing the "correct" transformation. With the purpose of introducing a correct registration result, a number of experiments have been carried out in which images acquired by MRI in connection with radiotherapy treatment have been deformed using mathematical functions. In the example demonstrated here, rotational displacements around the centre are applied. How large an angle the rotation is has a sine dependence to the distance from the center. Images depicting the result of registering an image from a brachytherapy treatment to a version transformed with the function in

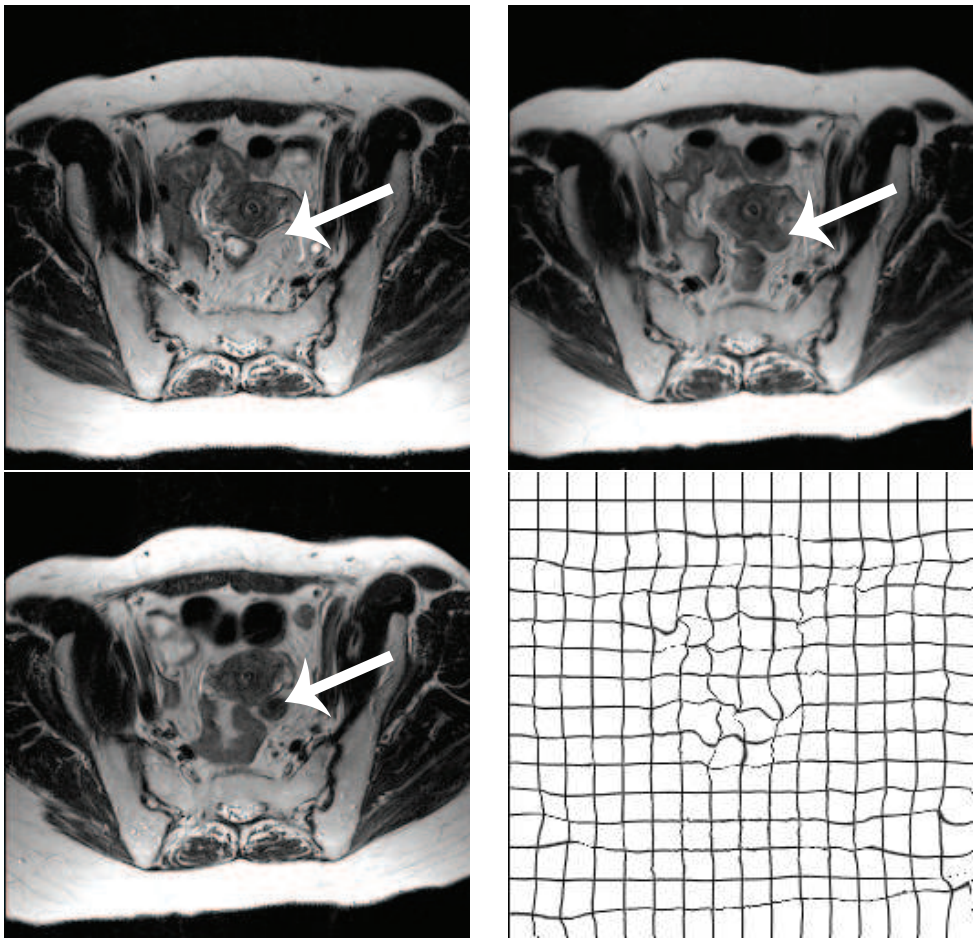

Figure 5.5: Images depicting the results of registering two MR images acquired in connection with the treatment of a cervical cancer patient. This is a difficult registration task due to differences in intestine content. Note that the uterus has been wrongly expanded to fill out a surrounding piece of intestine with a similar intensity (arrow). For a description of the different parts of the figure, see the caption of figure 5.3.

the wave experiment can be seen in figure 5.6. Besides the source, transformed source, and reference images, we also show an image describing the mathematical function used to transform a grid (on the bottom left). On the bottom right we show the transformation found by registration. In the middle the found displacements are subtracted from the mathematical transformation. This can be used to see if the registration algorithm finds the same transformation as was defined mathematically. If this is the case, a rectilinear grid should emerge. It can be seen that the registration method handles the wave experiment rather well in most parts of the image. However some parts of the registration result show a large degree of local deformation resulting in a "pinching" of parts of the image. Note that this is not obvious from a quick look at the image resulting from the registration. Besides the pinching deviations between the deforming

function and the registration result also occur near image borders where our boundary conditions prevent the method from finding a match.

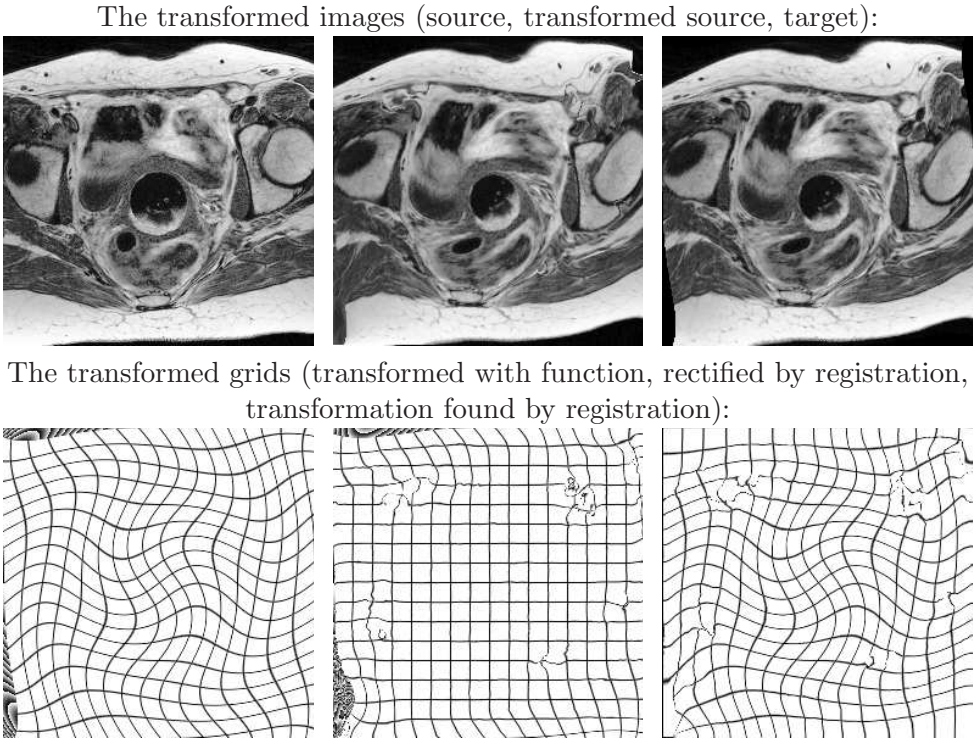


Figure 5.6: Images depicting the result of registering an image from a brachytherapy treatment to a mathematically transformed version of that image. On the top row the source image, transformed source image, and reference image is found. Here slice 9 of 16 is shown. On the bottom row: to the left the mathematical function is used to transform a grid. On the right side the transformation found by registration is shown. In the middle the found displacements are subtracted from the mathematical transformation.

## 5.7 Discussion

In this section we discuss a number of issues that one needs to be aware of when using the viscous-fluid method. We also discuss of the use of a GPU for accelerating the method.

For the method to work well, intensities need to be consistent meaning that the same part of an organ in two images need to be the of same (or very similar) intensity. For CT scans the assumption of intensity preservation is generally met, but in connection with MRI, it is required to use the same protocol and placement of coils. Due to the choice of similarity measure between images, the viscous-fluid method with its current implementation of driving forces cannot be used for multimodality registration (image fusion). In the face of images of different modality, the only way of using our implementation is to

base registration on binary masks alone which is not optimal. Ways to allow the viscous fluid motion model to be used for multimodal registration would be to base the driving forces on mutual information [34] or normalised gradient fields [50]

As a result of the way driving forces are calculated in the viscous-fluid registration, it does a better job at describing expansion rather than rotation. In practice this can give problems when an organ (e.g. the uterus scanned in connection with cervical cancer treatment) is bent in one image and straight in another. In this case it can happen that some parts of the source image are incorrectly compressed while other parts are incorrectly expanded to fill out the positioning of the organ in the reference image leading to an inadequate registration. An example of this can be seen in figure 5.7 which is taken from [28].

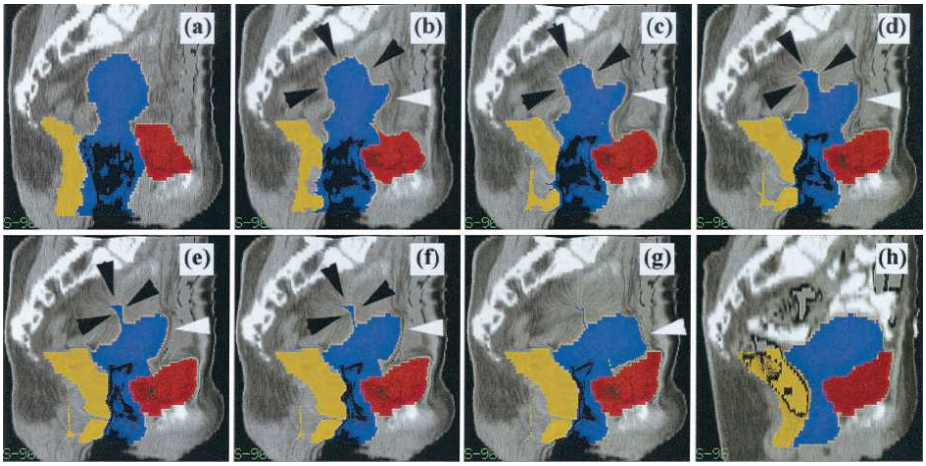


Figure 5.7: Illustration showing an erroneous registration of an upright positioned uterus (blue shape) to a bent uterus. Several steps in the registration are shown. It can be seen that some parts of the source image (top left) are incorrectly compressed while other parts are incorrectly expanded to fill out the positioning of the organ in the reference image (bottom right) From [28] courtesy of Gary E. Christensen. Reprinted with permission from Elsevier

The significant speedup of registration obtained by porting the algorithm to being GPU accelerated can partly be explained by the superior number of floating point operations per second (Flops) of the GPU compared to the CPU. An Nvidia Geforce 8800GTX GPU is capable of producing approximately 500 GFlops [103]. This must be compared to the theoretical maximum throughput of 7.4 GFlops for a 3.8 GHz Intel Xeon CPU. In general the development in computer graphics hardware works in our favour. The capabilities in terms of computing and memory access is steeply increasing. This development is expected to continue in the coming years. Also the possibility of using multiple GPUs in a single PC has the potential of decreasing computation times even further.

However the number of Flops achievable on the two architectures is not

sufficient information for determining the achievable speedup when porting to the GPU. As mentioned in section 4.5 what must also be taken into account is the ratio between the number of machine instructions and the number of memory accesses - this is sometimes called the *arithmetic intensity* [109]. In the viscous-fluid implementation - especially in the Jacobi iteration steps - the computational intensity is not high. This means that the limiting factor in the calculation is here not the number of Flops but rather the number of gigabytes per second of memory access.

The viscous-fluid registration method discretised using finite difference approximations is very well suited for being GPU accelerated through OpenGL drivers. One of the reasons for this is that in all steps except the calculation of body forces, all texture coordinates for looking up values can be determined by interpolation between texture coordinates specified as vertex attributes. This enables the use of hardware interpolation and cache friendly memory access instead of random access [129].

## 5.8 Conclusion and further comments

In this chapter it has been demonstrated that modern graphics hardware can be used to significantly decrease the time required for registration of medical 3D images using the viscous-fluid registration method. This was done by formulating the required computations in fragment programs executable on a GPU.

Images from the female pelvic region which is one of the clinical focus areas in this dissertation feature very large geometric discrepancies due to high mobility of soft organs like the uterus, intestines and bladder. Furthermore differences in bladder and intestine content can introduce large differences in image intensity in which case an assumption of identity photometric transformation breaks down.

As mentioned in chapter 3 the hybrid viscous-fluid model of [28] (intensities plus binary volumes) has previously been used to register images from cervical cancer patients. In cases where the method does not completely succeed in registering the images due to ill-posed deformations of organs the *fluid landmark registration* [30] was used. In our initial work on the cervical cancer case we adopted the hybrid viscous-fluid approach (chapter 5). However we found it desirable to avoid the fluid landmark step. For this reason focus shifted towards including physical organ simulations in the registration - more on this in chapters 7 and 8.

While gaining experience with the viscous-fluid method, we saw quite a few cases in which the registration looked acceptable when evaluated on a comparison between the transformed image and the reference image, but a closer inspection of the development of the transformation revealed physical impossibilities. An example of this is the growth of an intestine from the uterus seen in figure 5.5. We also found that even quite small differences in intensities caused by changes in position of the coil during MR acquisition could cause a compression or expansion of parts of the image (usually the fatty tissue below the skin)

so that image intensity was smeared out. The latter of the mentioned problems might be alleviated by the use of MI or NGF as the similarity measure. However the development of impossible transformations that were not easily detected from looking at the registered image made us uncomfortable with the method, and we began looking for alternative methods. While working on a GPU implementation of the 2D Horn & Schunck optical flow algorithm for a project on MR temperature monitoring [37] we were impressed with the performance of the method on 2D MR data. This lead us in the direction of the work by Cornelius & Kanade on including an intensity deviation term in the optical flow estimation, which we extended to 3D. We implemented these optical flow estimation methods on the GPU, which will be presented in chapter 6.

# Chapter 6

# Optical flow registration

## 6.1 Introduction

In this chapter we evaluate two fully automated intensity based deformable registration methods driven by the concept of optical flow [56], [32]. These methods were chosen based on our previous experience with registration of 2D MRI for online MR temperature monitoring [37]. We extended the models from 2D to 3D. In literature the Horn and Schunck method has previously been used in 3D (see e.g. [48]). However, to our knowledge the Cornelius and Kanade method has previously only been presented for applications in 2D. As a preliminary validation study we report the result of using one of these methods for registering 4DCT lung acquisitions and the other for registering head and neck CBCT and conventional CT acquisitions. To compute such complex registrations in a clinically acceptable time frame, we implemented the two algorithms in parallel on a commodity graphics processing unit (GPU). The methods used in the study presented here are easily parallelisable making them ideal for GPU implementation. Furthermore, the running times of these methods are relatively short even in a non-accelerated version meaning that a significant acceleration will allow us to do deformable registration in very short time frames. The chapter is partially based on [100].

## 6.2 The Horn and Schunck model

The Horn and Schunck method is based on an assumption of preserved image intensity between the two 3D images to be registered. This means that it only works for registering images of the same modality and only for images with consistent grey values when multiple image sets are compared. It is also assumed that the deformation is smooth. This is in general a valid assumption for soft tissue deformation. The Horn and Schunck method is thus suitable for registering successive CT images due to the reproducibility of Hounsfield Units for this modality. Denoting the image intensity by $I$ we can write the assumption of intensity preservation as:

$$I(x, y, z, t) = I(x + \delta x, y + \delta y, z + \delta z, t + \delta t) \tag{6.1}$$

where $\delta x$, $\delta y$, and $\delta z$ are the displacements occurring between times $t$ and $t+\delta t$. Assuming small displacements the right hand side of (6.1) can be expanded as a Taylor series resulting in:

$$I(x+\delta x, y+\delta y, z+\delta z, t+\delta t) = I(x,y,z,t)+\frac{\partial I}{\partial x}\delta x+\frac{\partial I}{\partial y}\delta y+\frac{\partial I}{\partial z}\delta z+\frac{\partial I}{\partial t}\delta t+\text{H.O.T.} \tag{6.2}$$

where H.O.T. means higher order terms, which we ignore. Combining (6.1) and (6.2) we get

$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial z}\delta z + \frac{\partial I}{\partial t}\delta t = 0. \tag{6.3}$$

Dividing by $\delta t$ results in:

$$\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial z}\frac{\delta z}{\delta t} + \frac{\partial I}{\partial t}\frac{\delta t}{\delta t} = 0 \tag{6.4}$$

and taking the differential limit we get

$$\frac{\partial I}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} + \frac{\partial I}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t} + \frac{\partial I}{\partial z}\frac{\mathrm{d}z}{\mathrm{d}t} + \frac{\partial I}{\partial t} = 0. \tag{6.5}$$

We now denote the three velocity components by $u = \frac{\mathrm{d}x}{\mathrm{d}t}$, $v = \frac{\mathrm{d}y}{\mathrm{d}t}$, and $w = \frac{\mathrm{d}z}{\mathrm{d}t}$ (think of these as the displacements occurring in the time between two image acquisitions). This means that (6.5) can be written as:

$$I_x u + I_y v + I_z w + I_t = 0 \tag{6.6}$$

where the $x$, $y$, $z$, or $t$ in subscript denotes differentiation.

In practise deviations from intensity preservation will occur. Therefore we introduce an energy term $\mathscr{E}_b$ that we wish to minimise:

$$\mathscr{E}_b = I_x u + I_y v + I_z w + I_t. \tag{6.7}$$

In the minimisation of 6.7 we wish to ensure that the field of velocity vectors is smooth. Thus we introduce a constraint energy term $\mathscr{E}_c$:

$$\begin{aligned} \mathscr{E}_c^2 &= ||\nabla u||^2 + ||\nabla v||^2 + ||\nabla w||^2 \\ &= u_x^2 + u_y^2 + u_z^2 + v_x^2 + v_y^2 + v_z^2 + w_x^2 + w_y^2 + w_z^2 \end{aligned} \tag{6.8}$$

and combine the two energy terms in an energy functional $E$, which is evaluated over the entire image domain:

$$E = \iiint \left(\mathscr{E}_b^2 + \alpha^2 \mathscr{E}_c^2\right) \mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z = \iiint f\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \tag{6.9}$$

where $\alpha$ is a weighting factor.

A necessary condition for a velocity field to be a minimizer of $E$ is that it satisfies the *Euler-Lagrange equations* (see e.g [73]) given by:

$$\frac{\partial f}{\partial u} - \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial u_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial u_y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial f}{\partial u_z}\right) = 0 \tag{6.10}$$

$$\frac{\partial f}{\partial v} - \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial v_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial v_y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial f}{\partial v_z}\right) = 0 \tag{6.11}$$

$$\frac{\partial f}{\partial w} - \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial w_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial w_y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial f}{\partial w_z}\right) = 0 \tag{6.12}$$

Evaluating the terms in (6.10) results in:

$$\frac{\partial f}{\partial u} = 2I_x\left(I_x u + I_y v + I_z w + I_t\right), \tag{6.13}$$

$$\frac{\partial}{\partial x}\left(\frac{\partial f}{\partial u_x}\right) = \frac{\partial}{\partial x}\left(2\alpha^2 u_x\right) = 2\alpha^2 u_{xx}, \tag{6.14}$$

$$\frac{\partial}{\partial y}\left(\frac{\partial f}{\partial u_y}\right) = 2\alpha^2 u_{yy}, \tag{6.15}$$

$$\frac{\partial}{\partial z}\left(\frac{\partial f}{\partial u_z}\right) = 2\alpha^2 u_{zz} \tag{6.16}$$

Eliminating the factor 2 and introducing the Laplacian $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$ the Euler-Lagrange equation (6.10) for $u$ becomes:

$$I_x\left(I_x u + I_y v + I_z w + I_t\right) - \alpha^2 \nabla^2 u = 0 \tag{6.17}$$

Similarly the expressions for $v$ and $w$ become:

$$I_y\left(I_x u + I_y v + I_z w + I_t\right) - \alpha^2 \nabla^2 v = 0 \tag{6.18}$$

$$I_z\left(I_x u + I_y v + I_z w + I_t\right) - \alpha^2 \nabla^2 w = 0 \tag{6.19}$$

We now assume we are representing the velocities in a Cartesian grid and use finite difference approximations when evaluating the Laplacians. Then we approximate the Laplacian of $u$ by $\nabla^2 u \approx \bar{u} - u$ where $\bar{u}$ is the mean of $u$ in the $3 \times 3 \times 3$ neighbourhood of the current grid position (excluding the current grid position itself). Weighting all neighbors equally this approximation is proportional to $\nabla^2 u$ and we can account for the difference from $\nabla^2 u$ using the factor $\alpha$. Likewise the approximations $\nabla^2 v \approx \bar{v} - v$ and $\nabla^2 w \approx \bar{w} - w$ are used. Inserting this in (6.17)-(6.19) gives:

$$I_x\left(I_x u + I_y v + I_z w + I_t\right) - \alpha^2\left(\bar{u} - u\right) = 0 \tag{6.20}$$

$$I_y\left(I_x u + I_y v + I_z w + I_t\right) - \alpha^2\left(\bar{v} - v\right) = 0 \tag{6.21}$$

$$I_z\left(I_x u + I_y v + I_z w + I_t\right) - \alpha^2\left(\bar{w} - w\right) = 0 \tag{6.22}$$

This can be written on matrix-form like this:

$$\begin{bmatrix} {I_x}^2+\alpha^2 & I_xI_y & I_xI_z \\ I_xI_y & {I_y}^2+\alpha^2 & I_yI_z \\ I_xI_z & I_yI_z & {I_z}^2+\alpha^2 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} -I_xI_t+\alpha^2\bar{u} \\ -I_yI_t+\alpha^2\bar{v} \\ -I_zI_t+\alpha^2\bar{w} \end{bmatrix} \tag{6.23}$$

Use of Cramer's rule results in (after some calculations):

$$\begin{aligned} (I_x^2+I_y^2+I_z^2+\alpha^2)u &= I_y^2\bar{u}+I_z^2\bar{u}-I_xI_t+\alpha^2\bar{u}-I_xI_y\bar{v}-I_xI_z\bar{w} && (6.24)\\ (I_x^2+I_y^2+I_z^2+\alpha^2)v &= I_x^2\bar{v}+I_z^2\bar{v}-I_yI_t+\alpha^2\bar{v}-I_xI_y\bar{u}-I_yI_z\bar{w} && (6.25)\\ (I_x^2+I_y^2+I_z^2+\alpha^2)w &= I_x^2\bar{w}+I_y^2\bar{w}-I_zI_t+\alpha^2\bar{w}-I_xI_z\bar{u}-I_yI_z\bar{v} && (6.26) \end{aligned}$$

which can be rewritten as

$$\begin{aligned} (I_x^2+I_y^2+I_z^2+\alpha^2)(u-\bar{u}) &= -I_x[I_x\bar{u}+I_y\bar{v}+I_z\bar{w}+I_t] && (6.27)\\ (I_x^2+I_y^2+I_z^2+\alpha^2)(v-\bar{v}) &= -I_y[I_x\bar{u}+I_y\bar{v}+I_z\bar{w}+I_t] && (6.28)\\ (I_x^2+I_y^2+I_z^2+\alpha^2)(w-\bar{w}) &= -I_z[I_x\bar{u}+I_y\bar{v}+I_z\bar{w}+I_t]. && (6.29) \end{aligned}$$

We are now ready to solve the system (6.23) by Jacobi iteration (see e.g. [71, section 4.6]) using the following update rules:

$$u^{(k+1)} = \bar{u}^{(k)} - I_x[I_x\bar{u}^{(k)}+I_y\bar{v}^{(k)}+I_z\bar{w}^{(k)}+I_t]/(I_x^2+I_y^2+I_z^2+\alpha^2) \tag{6.30}$$

$$v^{(k+1)} = \bar{v}^{(k)} - I_y[I_x\bar{u}^{(k)}+I_y\bar{v}^{(k)}+I_z\bar{w}^{(k)}+I_t]/(I_x^2+I_y^2+I_z^2+\alpha^2) \tag{6.31}$$

$$w^{(k+1)} = \bar{w}^{(k)} - I_z[I_x\bar{u}^{(k)}+I_y\bar{v}^{(k)}+I_z\bar{w}^{(k)}+I_t]/(I_x^2+I_y^2+I_z^2+\alpha^2) \tag{6.32}$$

where we approximate $I_x$, $I_y$, and $I_z$ as the mean of two central difference approximations (one for the source image and one for the reference image). For $I_t$ we use a forward difference approximation.

## 6.3 The Cornelius and Kanade model

When registering MR to MR or CBCT to CT the assumption of intensity preservation is no longer valid. In the CBCT modality intensity in Hounsfield units are affected by the larger contribution from x-ray scatter. Also the design of the detector and the image reconstruction algorithm used has an impact on Hounsfield unit reproducibility for CBCT. To facilitate handling of intensity differences, Cornelius and Kanade extended the original algorithm thus enabling it to tolerate some deviation from that assumption [32]. In their work it was further assumed that the non-motion-related intensity differences are smoothly varying in space. Here we extend the method to three dimensions.

The function $E$ to be minimised in the Cornelius and Kanade method is on the following form:

$$E = \iiint \left(\mathscr{E}_b^2+\alpha^2\mathscr{E}_c^2+\beta^2\mathscr{E}_I^2\right)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z = \iiint f\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \tag{6.33}$$

where $\mathscr{E}_c$ is as in the previous section and $\mathscr{E}_I$ will be introduced below. The term $\mathscr{E}_b$ is introduced as a result of acknowledging that changes in intensity may have other causes than motion. Isolating the intensity difference from (6.2) results in

$$I(x+\delta x, y+\delta y, z+\delta z, t+\delta t) - I(x,y,z,t) = \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial z}\delta z + \frac{\partial I}{\partial t}\delta t. \quad (6.34)$$

Dividing by $\delta t$ results in:

$$\frac{I(x+\delta x, y+\delta y, z+\delta z, t+\delta t) - I(x,y,z,t)}{\delta t} = \frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial z}\frac{\delta z}{\delta t} + \frac{\partial I}{\partial t}\frac{\delta t}{\delta t} \quad (6.35)$$

and taking the differential limit we get

$$\frac{\mathrm{d}I}{\mathrm{d}t} = \frac{\partial I}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} + \frac{\partial I}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t} + \frac{\partial I}{\partial z}\frac{\mathrm{d}z}{\mathrm{d}t} + \frac{\partial I}{\partial t}. \quad (6.36)$$

We now let the term $\mathscr{E}_b$ denote the difference between left and right hand side of this expression:

$$\mathscr{E}_b = \frac{\mathrm{d}I}{\mathrm{d}t} - \frac{\partial I}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} - \frac{\partial I}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t} - \frac{\partial I}{\partial z}\frac{\mathrm{d}z}{\mathrm{d}t} - \frac{\partial I}{\partial t} \quad (6.37)$$

or, using the shorthand notation $B = \frac{\mathrm{d}I}{\mathrm{d}t}$:

$$\mathscr{E}_b = B - I_x u - I_y v - I_z w - I_t \quad (6.38)$$

The idea in the Cornelius and Kanade method is to solve for $B$ as well as for the displacement field, thereby allowing compensation for deviations from intensity preservation. The new energy term $\mathscr{E}_I$ penalises large derivatives in the field of $B$ values:

$$\mathscr{E}_I^2 = B_x^2 + B_y^2 + B_z^2 \quad (6.39)$$

Besides the Euler-Lagrange equations (6.10)-(6.12) we now also have an Euler-Lagrange equation in $B$:

$$\frac{\partial f}{\partial B} - \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial B_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial B_y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial f}{\partial B_z}\right) = 0 \quad (6.40)$$

From the four Euler-Lagrange equations we get:

$$I_x\left(-B + I_x u + I_y v + I_z w + I_t\right) - \alpha^2\nabla^2 u = 0 \quad (6.41)$$
$$I_y\left(-B + I_x u + I_y v + I_z w + I_t\right) - \alpha^2\nabla^2 v = 0 \quad (6.42)$$
$$I_z\left(-B + I_x u + I_y v + I_z w + I_t\right) - \alpha^2\nabla^2 w = 0 \quad (6.43)$$
$$-B + I_x u + I_y v + I_z w + I_t + \beta^2\nabla^2 B = 0 \quad (6.44)$$

Introducing approximations to the Laplacians as above, the linear system of equations becomes:

$$\begin{bmatrix} {I_x}^2+\alpha^2 & I_xI_y & I_xI_z & -I_x \\ I_xI_y & {I_y}^2+\alpha^2 & I_yI_z & -I_y \\ I_xI_z & I_yI_z & {I_z}^2+\alpha^2 & -I_z \\ I_x & I_y & I_z & -1-\beta^2 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ B \end{bmatrix} = \begin{bmatrix} -I_xI_t+\alpha^2\bar{u} \\ -I_yI_t+\alpha^2\bar{v} \\ -I_zI_t+\alpha^2\bar{w} \\ -I_t-\beta^2\bar{B} \end{bmatrix} \quad (6.45)$$

Use of Cramer's rule gives:

$$\begin{aligned}
&(-\beta^2I_x^2-\beta^2I_y^2-\beta^2I_z^2-\alpha^2-\alpha^2\beta^2)u \\
&\quad = \beta^2I_xI_t-\beta^2I_y^2\bar{u}-\beta^2I_z^2\bar{u}\ -\alpha^2\bar{u}-\alpha^2\beta^2\bar{u}+\beta^2I_xI_y\bar{v}+\beta^2I_xI_z\bar{w}-\beta^2I_x\bar{B} \\
&(-\beta^2I_x^2-\beta^2I_y^2-\beta^2I_z^2-\alpha^2-\alpha^2\beta^2)v \\
&\quad = -\beta^2I_x^2\bar{v}+\beta^2I_yI_t-\beta^2I_z^2\bar{v}\ -\alpha^2\bar{v}-\alpha^2\beta^2\bar{v}+\beta^2I_yI_z\bar{w}-\beta^2I_y\bar{B}+\beta^2I_xI_y\bar{u} \\
&(-\beta^2I_x^2-\beta^2I_y^2-\beta^2I_z^2-\alpha^2-\alpha^2\beta^2)w \\
&\quad = -\beta^2I_x^2\bar{w}-\beta^2I_y^2\bar{w}+\beta^2I_zI_t\ -\alpha^2\bar{w}-\alpha^2\beta^2\bar{w}-\beta^2I_z\bar{B}+\beta^2I_yI_z\bar{v}+\beta^2I_xI_z\bar{u} \\
&(-\beta^2I_x^2-\beta^2I_y^2-\beta^2I_z^2-\alpha^2-\alpha^2\beta^2)B \\
&\quad = -\beta^2I_x^2\bar{B}-\beta^2I_y^2\bar{B}-\beta^2I_z^2\bar{B}\ -\alpha^2I_t-\alpha^2\beta^2\bar{B}-\alpha^2I_z\bar{w}-\alpha^2I_y\bar{v}-\alpha^2I_x\bar{u}
\end{aligned}$$

Letting $C=\beta^2I_x^2+\beta^2I_y^2+\beta^2I_z^2+\alpha^2+\alpha^2\beta^2$ this can be written as

$$\begin{aligned}
C(u-\bar{u}) &= -\beta^2I_x(I_x\bar{u}+I_y\bar{v}+I_z\bar{w}+I_t-\bar{B}) \\
C(v-\bar{v}) &= -\beta^2I_y(I_x\bar{u}+I_y\bar{v}+I_z\bar{w}+I_t-\bar{B}) \\
C(w-\bar{w}) &= -\beta^2I_z(I_x\bar{u}+I_y\bar{v}+I_z\bar{w}+I_t-\bar{B}) \\
C(B-\bar{B}) &= \alpha^2(I_x\bar{u}+I_y\bar{v}+I_z\bar{w}+I_t-\bar{B})
\end{aligned} \quad (6.46)$$

which results in the following Jacobi iteration update rules:

$$\begin{aligned}
u^{(k+1)} &= \bar{u}^{(k)}-\beta^2I_x(I_x\bar{u}^{(k)}+I_y\bar{v}^{(k)}+I_z\bar{w}^{(k)}+I_t-\bar{B}^{(k)})/C \\
v^{(k+1)} &= \bar{v}^{(k)}-\beta^2I_y(I_x\bar{u}^{(k)}+I_y\bar{v}^{(k)}+I_z\bar{w}^{(k)}+I_t-\bar{B}^{(k)})/C \\
w^{(k+1)} &= \bar{w}^{(k)}-\beta^2I_z(I_x\bar{u}^{(k)}+I_y\bar{v}^{(k)}+I_z\bar{w}^{(k)}+I_t-\bar{B}^{(k)})/C \\
B^{(k+1)} &= \bar{B}^{(k)}+\alpha^2(I_x\bar{u}^{(k)}+I_y\bar{v}^{(k)}+I_z\bar{w}^{(k)}+I_t-\bar{B}^{(k)})/C
\end{aligned} \quad (6.47)$$

## 6.4 GPU based implementations

The numerical method of Jacobi iteration is very suitable for an SIMT implementation to be run on a GPU. This is due to the fact that during the computation of each Jacobi iteration only information from the previous iteration is needed at each grid point. Thus write conflicts are avoided. The downside to Jacobi iteration is a slower convergence than e.g. Gauss-Seidel (or variants such as successive overrelaxation). Furthermore, more memory is used because values of the solution vector are needed in both iteration $k$ for reading and $k+1$ for writing. This means that twice the memory is consumed. The amount of memory available in the graphics cards we have used can be a constraint when

registering large data sets. Therefore our GPU implementation of the Horn & Schunck method as well as the Cornelius & Kanade method can be configured to use a method which can be characterised as something in between Jacobi iteration and Gauss-Seidel.

Using shared memory copies of the required values from the solution vector of the $n$-th iteration needed by any of the threads in each three dimensional block of threads are read. For this the threads work together, each copying a small part of the input data. To ensure all values are read before further computation a (per block) thread synchronisation is performed. The copied values are subsequently used for a step of the Jacobi iteration. For the grid points covered by the part of a block bordering towards another block we need to read values that are to be updated by another block. We have no guarantees about the order of execution of blocks so we do not know which of the read values belong to iteration $k$ or have been updated to iteration $k + 1$. At grid points interior to a block a normal Jacobi iteration scheme is followed. Using per-block Jacobi iteration allows us to halve the memory used for storing values of the solution vector.

Possible termination criteria for the Jacobi iteration are to use a fixed number of iterations or find the largest change in an optical flow velocity/displacement vector and terminate the iterations once the norm of this vector is below a certain threshold. We use the latter approach but avoid the cost of a reduction operation by allocating a Boolean variable in global memory which is set to false before every iteration. Those of the threads responsible for Jacobi iteration that change their displacement vector by more than the threshold it will write true in the variable. This means that if the Boolean variable is false after an iteration we have met the termination criterion.

### 6.4.1 Multi-resolution approach

To speed up the registration and to avoid local minima our implementation uses a multiresolution approach. A number $n$ is chosen which specifies the number of resolutions. In each resolution the input images are downsampled by a factor 2 in all directions[1] followed by a recursive call to the multiresolution method. When the recursive call returns the resulting displacement field is upsampled to the current resolution and the input image is transformed using the upsampled transformation field. The deformed image is then fed as input to the optical flow estimation method. The resulting displacement field is used to transform the field from the recursive call and the resulting field is returned. The recursion ends once the lowest resolution is reached. The approach can be seen in pseudocode in algorithm 2.

Again we have focused on memory consumption. After computing spatial and time derivatives in each resolution the input images can be swapped out to host memory as only the derivatives are needed in the Jacobi iteration. This introduces an overhead in time consumption but allows registration of larger data sets.

[1] In cases where the $x$ and $y$-dimensions were more than twice the $z$-dimension(the number of slices), the images was not downsampled in the $z$-direction.

**Algorithm 2**: MultiResolutionRegistration($S, R, n$): The multiresolution method used for our optical flow registration implementations. The function OpticalFlowEstimation($S_t, R$) called is either the Horn and Schunck or the Cornelius and Kanade method.

**input** : Source image $S$, Reference image $R$, resolution number $n$
**output**: Resulting displacement field
**if** $n > 1$ **then**
  $S_d \leftarrow$ DownSample($S$)
  $R_d \leftarrow$ DownSample($R$)
  $D \leftarrow$ MultiResolutionRegistration($S_d, R_d, n-1$)
  $D_u \leftarrow$ UpSample($D$)
  $S_t \leftarrow S$ transformed using $D_u$
  $D_n \leftarrow$ OpticalFlowEstimation($S_t, R$)
  $D_t \leftarrow D_u$ transformed using $D_n$ (composition of transformations)
  **return** $D_t$
**else**
  $D_n \leftarrow$ OpticalFlowEstimation($S_d, R_d$)
  **return** $D_n$

## 6.5 Registration validation studies

### 6.5.1 Image material

The Horn and Schunck registration has been evaluated on the POPI-model which is a 4D thorax virtual phantom [141]. It consists of 10 CT data sets of resolution 482×360×141 which have been acquired at different breathing phases during a single breathing cycle. The images were acquired at the Léon Bérard Cancer Center, Lyon, France. In each data set corresponding to a breathing phase 41 landmark points have been manually identified, and these points are used for our validation. The voxel spacing of the acquisitions is 0.98×0.98×2.0 mm$^3$. The image material used for validating the Cornelius and Kanade method is a series of 6 CBCT images of a head and neck cancer patient and a conventional planning CT image acquired at the Department of Oncology, Aarhus University Hospital. The scans have been conducted weekly during the treatment course starting at the first fraction. The CBCT images are of dimensions 512×512×51 with a voxel spacing of 0.47×0.47×3.0 mm$^3$, while the conventional CT image is of dimensions 512×512×55 with a voxel spacing of 0.78×0.78×3.0 mm$^3$. Validation of bone alignment is based on 6 landmark points in each 3D data set. These points have been manually positioned prior to registration at easily identifiable positions on the bony anatomy of the cervical vertebrae and the base of skull representing clinically relevant match points. Positioning of these points is illustrated in figure 6.1.

Three series of registration experiments have been carried out:

1. CT to CT registration using the Horn and Schunck registration method: Following the convention from the POPI initiative all images from the 4D

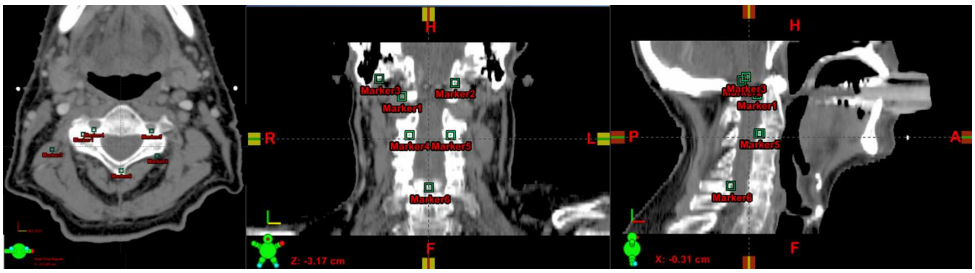


Figure 6.1: Visualisation of the positioning of the landmark points on the images acquired from the head and neck cancer patient.

data set (at time phases numbered 0, 2, 3, 4, 5, 6, 7, 8 and 9) have been registered to the reference image at phase 1.

2. CBCT to CBCT registration using the Cornelius and Kanade registration method: The CBCT images numbered 2 to 6 have been registered to CBCT image 1.

3. CBCT to CT registration using the Cornelius and Kanade registration method: The CBCT images have been registered to the planning CT image.

For the Horn and Schunck method a reference CPU based implementation has also been implemented allowing us to compare the computation times between the CPU and GPU versions.

### Image preprocessing

The CBCT images were processed using the curvature preserving GREYC-storation image denoising filter [140]. As this filter works in two dimensions, an in-house program has been used that simply filters each image slice independently. The effect of the filter is to remove noise (and in some cases artifacts from the CBCT reconstruction) while preserving the edge contrast between different kinds of morphology. See figure 6.2 for an example. The registration methods we present in this paper are designed to estimate the detailed deformation of morphology. If there is global displacement of patient position (that is translation and/or rotation) between two images it is necessary to do a rigid alignment of the images before the deformable registration in order to supply the method with a suitable starting point for estimation of organ deformation. The rigid registration method we use is based on the Insight Registration and Segmentation Toolkit (ITK). The measure used to compare images is based on mutual information. Input images are filtered using a threshold filter so that only bone morphology is included in the rigid registration. Again a multi-resolution approach is taken. For validation study 1 no image preprocessing was required as the images were already rigidly aligned. In validation studies 2 and 3, a rigid registration was required. A bounding box corresponding to the physical extent of CBCT image 1 has been cut out of the planning CT image and resampled to the same resolution as the CBCT images.

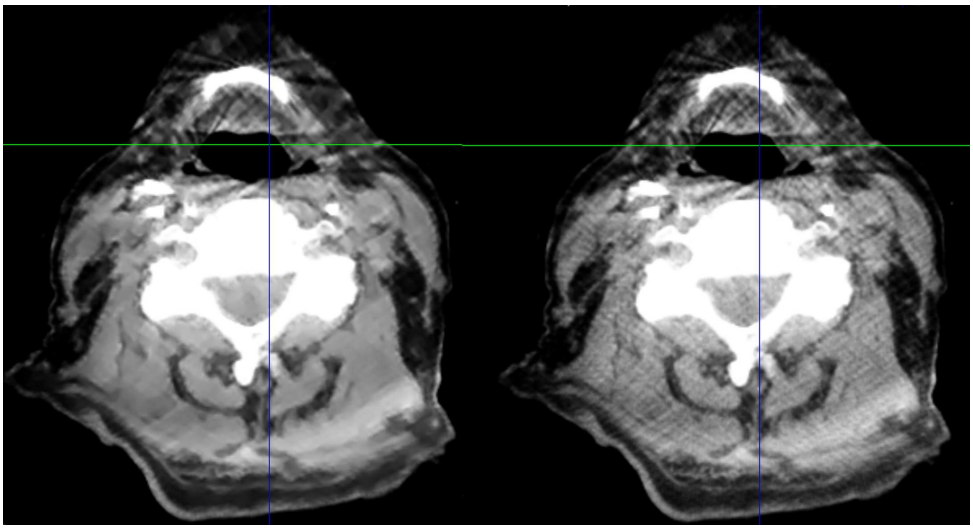

Figure 6.2: The effect of applying the GREYCstoration filter to a head and neck CBCT image. The unfiltered image is the one on the right. The level of noise has been reduced without blurring the image. Window-level settings have been set to emphasize the difference between the two images.

### 6.5.2 Results

When registering images from the POPI 4DCT data set the 768 megabytes of onboard memory available on the Nvidia geforce 8800GTX used allowed us to do full registration without using the memory conserving version of the Jacobi iteration scheme described in section 6.4. However when registering the CBCT images we used the new scheme. We have not observed any noticeable difference in registration result achieved when exchanging the normal Jacobi iteration with the per-block variant.

#### Validation study 1: Registration of the POPI 4DCT data set

The registration accuracy, evaluated on the target registration error (TRE) of landmark positions, is summarized in table 6.1. The distances are calculated as the Euclidian length of 3D vectors. Original average landmark distance was 3.5 mm $\pm$ 2.0 mm. After registration, this average distance was equal to 1.1 mm $\pm$ 0.6 mm.

A visualisation of the registration result can be seen in figure 6.3. In this visualisation the source image is shown in a reddish colour while the reference image is shown in a bluish colour. Where the images align a grey scale image emerges. In the unregistered case on the left blue and red areas can clearly be seen indicating that the morphology is not aligned. In the registered case to the right these coloured areas have almost disappeared indicating that the images have been successfully registered.

| Registration (src./ref.) | Mean original distance / mm | Max. original. distance / mm | Mean TRE / mm | Max. TRE / mm |
|---|---|---|---|---|
| 0/1 | 0.5 ± 0.5 | 2.4 | 0.7 ± 0.3 | 1.4 |
| 2/1 | 0.5 ± 0.6 | 2.6 | 0.7 ± 0.4 | 1.7 |
| 3/1 | 2.2 ± 1.8 | 6.6 | 1.3 ± 0.8 | 3.4 |
| 4/1 | 4.5 ± 2.5 | 10.0 | 1.2 ± 0.5 | 2.7 |
| 5/1 | 6.0 ± 2.9 | 12.1 | 1.3 ± 0.7 | 3.6 |
| 6/1 | 6.5 ± 3.3 | 14.0 | 1.2 ± 0.6 | 3.2 |
| 7/1 | 5.5 ± 3.0 | 14.0 | 1.3 ± 0.6 | 2.8 |
| 8/1 | 3.8 ± 1.6 | 6.2 | 1.0 ± 0.5 | 2.7 |
| 9/1 | 2.1 ± 1.0 | 4.5 | 0.9 ± 0.6 | 2.5 |
| Average | 3.5 ± 2.0 | 8.0 | 1.1 ± 0.6 | 2.7 |

Table 6.1: Target registration error (TRE) compared to original distances of landmark points in the POPI data set.

| Images | Before registration | | After rigid registration | | After deformable reg. | |
|---|---|---|---|---|---|---|
| src/ref | mean/mm | max/mm | mean/mm | max/mm | mean/mm | max/mm |
| 2/1 | 4.3 ± 1.3 | 6.4 | 1.7 ± 1.1 | 3.5 | 1.6 ± 0.4 | 1.9 |
| 3/1 | 4.2 ± 1.6 | 6.5 | 2.5 ± 1.1 | 3.7 | 1.8 ± 1.0 | 3.4 |
| 4/1 | 5.2 ± 1.1 | 6.6 | 1.8 ± 1.2 | 3.9 | 1.4 ± 1.0 | 3.1 |
| 5/1 | 6.5 ± 0.8 | 8.0 | 1.4 ± 0.9 | 2.7 | 1.4 ± 0.6 | 2.1 |
| 6/1 | 7.2 ± 0.9 | 8.6 | 1.7 ± 0.7 | 2.8 | 1.7 ± 0.8 | 3.0 |
| Average | 5.8 ± 1.1 | 7.2 | 1.8 ± 1.0 | 3.3 | 1.6 ± 0.8 | 2.7 |

Table 6.2: Evaluation of distances of bony landmarks for the CBCT-to-CBCT registration.

**Validation study 2: CBCT to CBCT registration**

In table 6.2 the alignment error of the landmark points positioned on the bony anatomy is summarized both before registration, after the rigid registration, and after the deformable registration. Original average landmark distance was 5.8 mm ± 1.1 mm. After the rigid registration, this average distance was equal to 1.8 mm ± 1.0 mm and after the deformable registration it was 1.6 mm ± 0.8 mm.

The result of registering CBCT image 3 to CBCT image 1 is visualized in figure 6.4. The images depicting the results of the rigid registration show an acceptable alignment of most bony anatomy, but it can be seen that the soft tissue and the area surrounding the oral cavity is not aligned. Improved alignment is obtained as a result of the deformable image registration as shown in the visualisation. In figure 6.5 a visualisation of the computed transformation can be seen showing that the deformation is smooth.

**Validation study 3: CBCT to planning CT registration**

The average alignment errors after the CBCT to CT registrations are found in table 6.3. After the rigid registration, this average distance was equal to 2.2

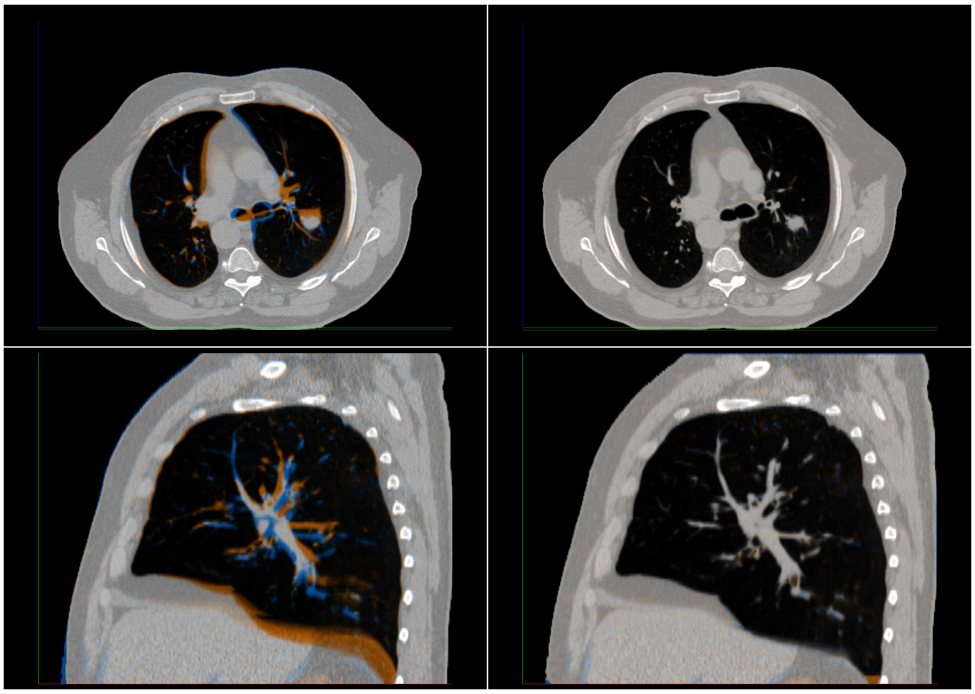

Figure 6.3: Differences between source and reference image before (left) and after (right) registration of an image from the POPI data set. A saggital slice and an axial slice are shown before and after registration. The source image is shown in a reddish colour while the reference image is shown in a bluish colour giving a gray scale image where the images align.

mm ± 0.6 mm and after the deformable registration it was 1.8 mm ± 0.6 mm.

The result of registering CBCT image 6 to the planning CT image can be seen in figure 6.6 . Again it can be seen that even though the skull and spine is aligned after the rigid registration, deformable registration is needed to account for changes in jaw positioning and deformation of soft tissue.

### Time consumption

On an Intel Core 2 CPU at 2.4 GHz the Horn and Schunck registration used for each 3D image in the POPI dataset in validation study 1 takes 30 minutes. On an Nvidia Geforce 8800GTX GPU in the same machine each registration takes 37 seconds, making the GPU version 48.6 times faster. For the 3D Cornelius and Kanade method we did not write a CPU reference implementation. Therefore we do not know the exact difference in processing time between CPU and GPU. However we expect the acceleration of this method to be somewhat smaller than for the Horn and Schunck method due to a less efficient use of shared memory on the GPU. Each 3D registration of the CBCT images in studies 2 and 3 using the Cornelius and Kanade method takes 64 seconds.

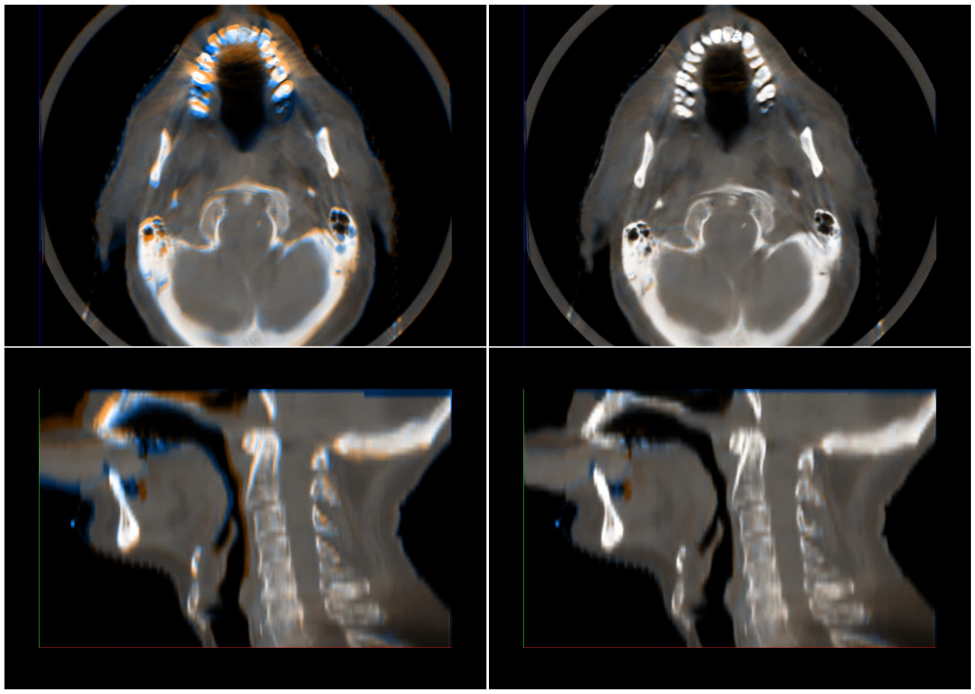

Figure 6.4: Red/blue visualisation of the difference between the rigid registration (left) and the deformable registration (right) of CBCT image 3 to CBCT image 1. A saggital slice and an axial slice are shown for each registration.

| | After rigid registration | | After deformable registration | |
|---|---|---|---|---|
| Source image | mean / mm | max / mm | mean / mm | max / mm |
| CBCT 1 | 2.2 ± 1.0 | 3.7 | 1.4 ± 0.6 | 2.7 |
| CBCT 2 | 1.7 ± 0.4 | 2.2 | 1.4 ± 0.5 | 1.9 |
| CBCT 3 | 1.9 ± 0.5 | 2.9 | 1.7 ± 0.3 | 2.2 |
| CBCT 4 | 2.8 ± 0.6 | 4.1 | 2.7 ± 1.1 | 4.9 |
| CBCT 5 | 2.0 ± 0.3 | 2.5 | 1.7 ± 0.5 | 2.6 |
| CBCT 6 | 2.3 ± 0.7 | 3.7 | 2.0 ± 0.6 | 3.3 |
| Average | 2.2 ± 0.6 | 3.2 | 1.8 ± 0.6 | 3.0 |

Table 6.3: Evaluation of distances of bony landmarks for the CBCT-to-CT registration.

## 6.6 Discussion

When specifying landmark points the limiting factor on the accuracy is often the slice thickness because a point may be between two slices which makes it hard to identify. In the light of this we consider the registration accuracy results in validation study 1 very acceptable as the mean landmark error is well below the slice thickness. This accuracy is comparable to results for the Demons algorithm previously reported in the POPI initiative [140]. In validation study 2 and 3 we used landmark points to track the registration of clinically relevant points on the cervical vertebrae and the base of skull. For this preliminary

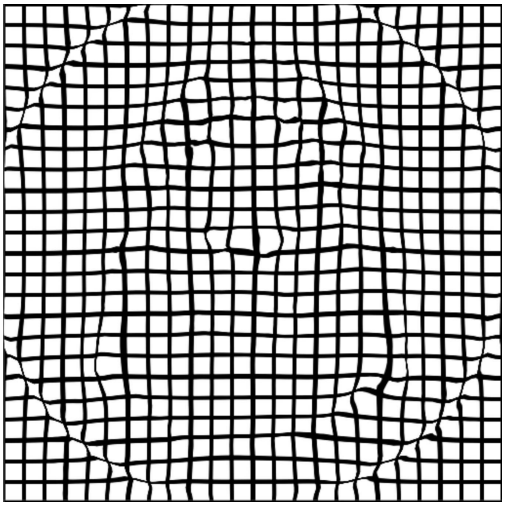

Figure 6.5: Illustration of the transformation applied to the axial slice shown in figure 6.4. The transform is used to deform a rectilinear grid with a grid spacing of 10 mm. Only the in-plane deformation can be seen.

study we did not perform a dedicated evaluation of the error in landmark point identification but we estimate that the error in each landmark point position may be as high as 2.0 mm. The magnitude of this error is mainly due to the slice thickness of 3 mm. It is interesting to note that although the rigid registration was done on bone morphology the mean landmark error *on bone morphology* is reduced by the deformable registration. In all registrations the mean error is smaller than the slice thickness. As demonstrated in figures 6.4 and 6.6 a rigid registration is not sufficient to describe the geometrical difference between the images. However these geometrical differences have been substantially reduced by the deformable registration.

Based on these studies we are optimistic that the Cornelius and Kanade method is suitable for registering head and neck CBCT images from a series of radiotherapy treatments to the planning image. Hopefully this will allow us to compensate for the unreliable Hounsfield units by using the inverse transformation of the one found in validation study 3 to map the Hounsfield units from the planning CT to each CBCT. This will make it possible to evaluate the doses delivered in the treatment fraction corresponding to each CBCT acquisition by doing a dose calculation on the corrected CBCT image. This has previously been suggested by Yang et al. [153]. When all CBCT images are registered to the same geometrical reference system, it will then be possible to evaluate the actual accumulated dose from a series of fractions for comparison with the planned doses. This can be done by deforming the dose distributions from each CBCT to the geometrical reference frame constituted by the planning CT [104].

Furthermore an accurate registration makes it possible to do automatic segmentation by transferring segmentations from the planning CT. Currently a larger head and neck CBCT registration study is being planned in which landmarks in bone as well as soft tissue will be used for accuracy evaluation.

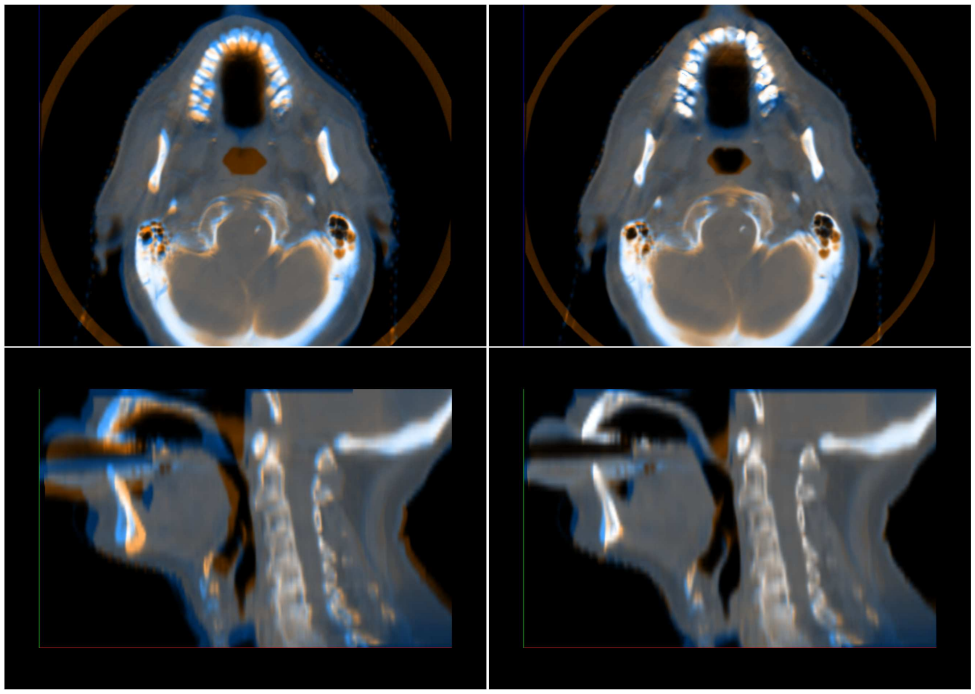

Figure 6.6: Red/blue visualisation of the difference between the rigid registration (left) and the deformable registration (right) of CBCT image 6 to the planning CT image. A saggital slice and an axial slice are shown for each registration.

Also a smaller slice thickness will be used allowing us to position landmarks more accurately.

In the pursuit of online IGRT, the performance of the required image processing in a sufficiently short time frame constitutes a huge technical challenge. Using the GPU has led to a very significant reduction of the registration time. The explanation of this reduction in processing time must be found in the parallelized architecture of the GPU. An acceleration in the magnitude presented here is not only possible for the Horn and Schunck method but should be attainable for other registration methods which lend themselves to being split into a large number of independent calculations. By splitting the images into blocks to be registered, is it also possible to distribute computations onto multiple GPUs. This does however introduce an overhead from the memory synchronisation needed at block boundaries so whether or not this would speed up the registration significantly is unclear. So far we have shown that using GPUs the mentioned registration methods can be accelerated to a level which is acceptable for use in an online setting in which the deformable registration is done while the patient is still on the treatment couch. This is the first step towards online dose plan adjustment. The processing power of GPUs can be utilized not only for registration as presented here, but for many of the compute intensive imaging tasks in IGRT making it an ideal and cost-efficient tool, which can help us getting further towards online IGRT.

# Part III

# Biomechanically based registration

# Chapter 7

## Biomechanical volumetric mesh registration

In this chapter we present our work on creating a fully automatic algorithm for registration of organs represented as a volumetric mesh. Registration of the organ surface is driven by force terms based on a distance field representation of pre-segmented source and reference shapes. Registration of internal morphology is achieved using a non-linear elastic biomechanical finite element model. The method is evaluated on phantom data based on fiducial marker accuracy and prostate data obtained in vivo based on inverse consistency of transformations. Furthermore we investigate the applicability of the method for registering female pelvic organs. The parallel nature of the method allows an efficient implementation on a GPU and as a result the method is very fast. The presented work is the continuation of work previously published [99] in which we used a simpler elastic model [88] for regularisation of the organ deformation. A key feature of the method is that the user does not need to specify boundary conditions (surface point correspondences) prior to the finite element analysis. Instead the boundary matches are found as an integrated part of the analysis. The proposed method has many potential uses in image guided radiotherapy (IGRT) which relies on registration to account for organ deformation between treatment sessions. Also, since the proposed method relies on pre-segmented data, it can be used for multimodal registration of e.g. prostate in the context of functional imaging.

### 7.1 Related work

#### 7.1.1 Registration using elastic finite element models

The *finite element method* (FEM) is an often used approach for solving differential equations numerically over a 2D area or 3D volume. The basic idea in finite element analysis (FEA) is to discretise the domain into "elements" spanned by vertex points. This can e.g. be triangles in 2D or tetrahedra in 3D. Values are associated with the vertices and interpolated inside elements using basis functions. The differential equation is rewritten into an integral form called its weak formulation [6]. By using (typically linear) element functions defined both globally and locally over each element it is possible to integrate over space and time. This is done either explicitly or implicitly by formulating a (typically sparse)

system of equations, making it possible to solve for a solution to the differential equation at discretised vertex positions. In the kinds of deformable models we are interested in the differential equation is typically a linear [17] [145] or non-linear [93] elastic model derived from continuum physics. Linear models feature an assumption of a linear relation between stress (a measure of force per unit area) and strain (a measure of deformation), whereas non-linear models feature a non-linear stress-strain relationship. The cost of the more advanced non-linear model is a higher computational burden.

### Finite element modelling of meshes spanning an entire image

Some authors have used elastic FEA as regularisation term in intensity based registration. In [52] a linear elastic deformable FEM is applied to the entire image domain, and the image is divided into subvolumes which are given different material properties. These are connected by appropriate boundary conditions. This technique is used for registering images used in preoperative planning of brain surgery. In subsequent work the linear elastic model is coupled with a model of incompressible fluids via appropriate boundary conditions [51]. Peckar et al. have used linear elastic FEA for finding a smooth transformation that satisfies a number of point constraints. Schnabel et al. [121] have also used a finite element model in connection with intensity based registration, but for creation of images to be registered. The idea is generating realistic registration problems with a known displacement mapping so that accuracy of existing registration methods can be evaluated.

### Finite element modelling of mesh representations of organs

Yan et al. uses FEA of for quantifying organ motion based on volumetric meshes retrieved from segmentation of CT images. This was used for dose accumulation [150]. Ferrant et al. used active surfaces and distance fields for initial surface matching of brain meshes and FEA for subsequent computation of interior mesh vertex displacements [40]. This model was also used for prostate registration [11]. Brock et al. also use FEA for estimating interior vertex displacements after initial boundary vertex projections [21] [19] and have also in earlier work used a linear elastic model for investigating the impact of breathing deformation on the delivered dose to a liver tumour [20]. They use a pre-computed surface correspondence map based on curvature correspondences as boundary conditions to the simulation.

Liang and Yan note that the accuracy of such techniques are limited by the surface projection algorithm [80]. To alleviate this problem they solve for interior and boundary vertices simultaneously by applying extra constraints ensuring that boundary vertices of the deforming organ stay on a triangle mesh describing the surface of the reference organ. Their model requires initial user defined boundary landmark correspondences.

In most work on FEM in image registration a linear elastic tissue deformation model has been applied. For large deformations however, especially those involving rotation, this model induces significant errors [110]. Consequently we

have chosen a non-linear model.

Recently Choi et al. published an iterative mesh fitting method which bears many similarities with our method [25]. They define a surface distance term for driving the registration. Unlike our approach their method uses a linear elastic FEM to displace meshes. In order to handle large deformations they utilise a technique called stiffness warping to account for the rotational part of the deformation. Zhang et al. avoid manual definition of landmarks by employing a finite element contact impact analysis [155]. This allows a model of a lung to slide in the pleural cavity. In their FEA the compute the deformation in multiple steps because their linear elastic model is not able to handle the large deformations of the lung in one step. Also Haker et al. use an iterative FEM [54]. Based on landmarks they project both source and reference organs onto a sphere using an active surface model. The sphere has been triangulated into a tetrahedral mesh, and guided by a few control landmarks, a point correspondence is found for mesh vertices. These are used as boundary conditions for a volumetric finite element analysis. Their work is based on theory in conformal mapping[1] [1] [53]. Conformal mapping was also used by Warfield et al., who also extended the linear elastic model with inhomogeneous material characteristics which allows fibres to be modelled [145].

### 7.1.2 Distance fields in image registration

Euclidean *distance fields* (or *distance maps*) are used in an $N$-dimensional space to track the (Euclidean) distance to the closest point of an object in this space. In 3D the field is used to describe the distance to a 2D surface. Both unsigned and signed distance fields can be used. The latter features a negative distance when inside an object. An extensive survey of applications of Euclidean distance fields have been presented by Jones et al. [59].

Besides the already mentioned work by Ferrant et al. [40] and Choi et al. [25], a number of groups working on medical image registration have included the use of Euclidean distance fields in their work. In [87] tissue classification in reference CT images is performed based on fixed Hounsfield units and distance fields to material boundaries are computed. In the source image bones are represented as a point cloud. Based on an SSD measure of distance values bones are rigidly registered to the reference image. This is used for tracking of wrist bones. Marai et al. also minimised a cost function of distance field terms to recover rigid body transformations [87]. In [106] an SSD of distance field based similarity metric is constructed for shape matching. Here the transformation consists of both a rigid term and a term for local deformable corrections. Xiao et al. use distance fields for rigid registration of surfaces that only partially overlap [147].

## 7.2 Shape representation

Prior to the registration, data from a three-dimensional imaging source has been segmented and segmented organs in both source and reference datasets

[1] That the mapping is conformal means that it is bijective and that angles are preserved.

are represented as binary volumes. These binary images are initially smoothed using a Gaussian filter. Next, Euclidean distance fields $\phi : \mathbb{R}^3 \rightarrow \mathbb{R}$ are generated for both datasets by using the implementation of Danielssons method [36] found in the Insight Segmentation and Registration Toolkit (ITK).

The distance fields are subsequently used as input to an algorithm which produces tetrahedral meshes. Before mesh generation the distance field is further smoothed using the fourth order level set anisotropic diffusion smoothing filter implementation in ITK. This filter smoothes the surface while preserving sharp features. The quality of the smoothing (and thus parameters to the algorithm) has been evaluated by visual inspection of a few generated meshes and parameters have been kept constant throughout all registration experiments. The deviation of the surface from original contours introduced by filtering has not been investigated.

Based on a distance field we use an in-house implementation of the *isosurface stuffing* mesh generation algorithm [75] for producing a tetrahedral mesh. This meshing method guarantees strict upper and lower bounds on the dihedral angles (angle between planes spanned by the sides of a tetrahedron) in the generated tetrahedra which increases the stability of the used FEM. The method operates on a body centered cubic (BCC) grid which consists of two regular uniform grids, one shifted by a half grid distance in all three spatial directions (figure 7.1 top left). All grid points are marked as either inside, outside or on the surface. This can be easily evaluated using the distance field. Furthermore all axis aligned or diagonal lines between grid points are checked for intersections with the surface. At intersections *cut points* are introduced. A so-called warping step evaluates the distance between cuts and BCC grid points, and if too close removes the cut points while moving the grid points onto the shape surface. Finally tetrahedra are generated based on a set of stencils covering all possible combinations of grid point and cut points on the grid (figure 7.1 bottom left). Two examples of meshes generated with this method can be seen in figure 7.1 right. Our implementation fills the mesh with tetrahedra of uniform size, but in [75] the method is extended to produce graded meshes with large tetrahedra inside and small tetrahedra near the boundary.

## 7.3 External driving forces

As mentioned, the objective of the registration is to find a transformation of mesh vertices such that the surface vertices of the source body is mapped to the surface of the reference body while interior vertices are at an equilibrium. We propose an iterative algorithm driven by forces computed near the boundary of the source shape. These forces, concatenated in the vector $\mathbf{R}$, are defined for each source mesh vertex $i$ by

$$\mathbf{R}_i = \gamma(\mathbf{A}_i + \beta\mathbf{B}_i) \tag{7.1}$$

where $\beta$ is constant throughout the registration process and $\gamma$ is iteratively increased until the surface match is achieved.

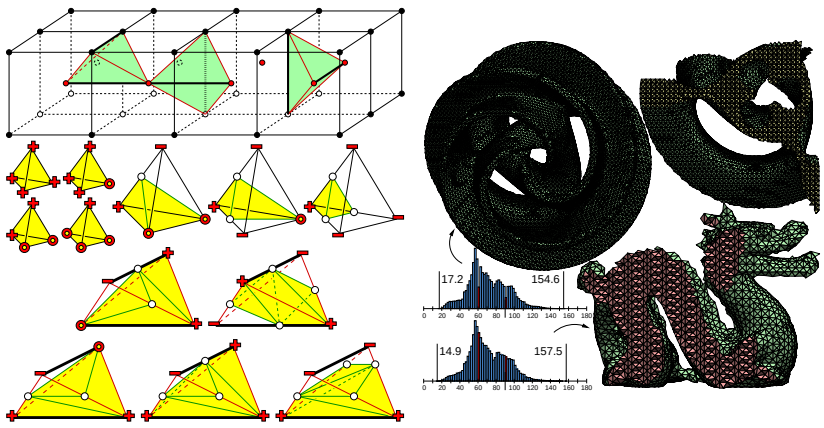


Figure 7.1: The isosurface stuffing method for generation of a tetrahedral mesh. **Top left:** The body centered cubic (BCC) grid allows 3-dimensional space to be spanned by tetrahedra. **Bottom left:** Stencils for mesh generation. Plus means that a vertex is inside the input shape, minus means it is outside, and zeroes are on the surface. White circles are cut-points. The yellow tetrahedra are those generated for the shown combinations. **Right:** Examples of generated meshes along with histograms of dihedral angles. Note that red bars have been scaled down by a factor of 20. All three illustrations from [75] courtesy of François Labelle and Jonathan Shewchuk

The force term $\mathbf{A}$ is based on the distance fields that were pre-computed before mesh generation of both the source ($\phi_S$) and reference ($\phi_R$) organ shapes. It applies to all mesh vertices less than $\tau$ millimetres from the source shape surface and is given by

$$\mathbf{A}_i = \left(\phi_S(\mathbf{p}_i^o) - \phi_R(\mathbf{p}_i^d)\right) \nabla\phi_R|_{\mathbf{p}_i^d} \tag{7.2}$$

where $\mathbf{p}_i^o$ denotes the initial position of particle $i$ (in the source shape) and $\mathbf{p}_i^d$ denotes the position of particle $i$ in the current configuration of the iterative registration process. The expression $\phi_R|_{\mathbf{p}_i^d}$ evaluates the reference surface distance at the deformed position of particle $i$. Intuitively Eq. (7.2) provides force vectors in the direction of steepest descent of the distance field calculated from the reference shape - i.e. towards the reference volume. The forces are weighted with the difference between each particle's current distance to the reference shape and the particle's original distance from the surface in the source shape. $\mathbf{A}_i$ vanishes as particle $i$ finds it "desired" distance to the reference surface. Equation (7.2) can be found from the Gâteaux derivative of the metric $\mathcal{D}[\phi_R, \phi_S; \mathbf{p}^d] = \sum_{i\in\mathcal{S}_\tau} \left(\phi_S(\mathbf{p}_i^o) - \phi_R(\mathbf{p}_i^d)\right)^2$ on the distance fields $\phi_S$ and $\phi_R$. Here $\mathbf{p}^d$ denotes the $3N$-dimensional vector of all components of the $\mathbf{p}_i^d$ vectors, and $\mathcal{S}_\tau$ denotes the set of vertex indices for vertices less than $\tau$ millimetres from the source surface.

The force term $\mathbf{B}$ is working normal to the current configuration of the source surface and is calculated for all boundary vertices as the mean of all normal vectors from adjacent surface triangles :

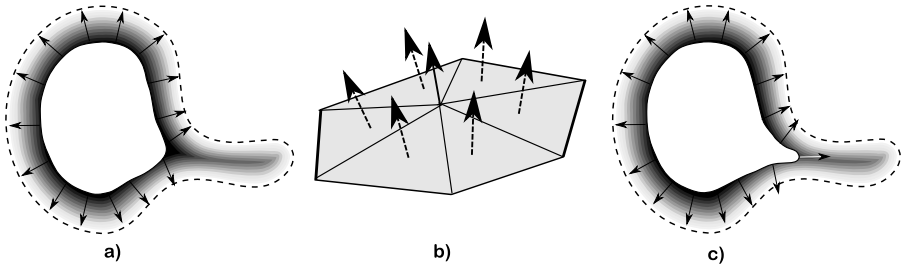


Figure 7.2: Illustration of the forces driving the source shape surface vertices towards the surface of the reference shape. a) 2D illustration showing the reference surface (dashed line), distance field (intensity gradient), source shape (white area), and the direction of forces derived from force term **A**. b) a surface patch consisting of 6 surface triangles surrounding a vertex. A normal vector (dashed line) is calculated per triangle, and the mean vector is used in force term **B**. c) 2D illustration of **B** forces showing the same components as in a). Notice how this term allows the source boundary to move into narrow extrusions of the reference surface.

$$\mathbf{B}_i = \left(\phi_S(\mathbf{p}_i^o) - \phi_R(\mathbf{p}_i^d)\right) \frac{1}{N_i} \sum_{\mathbf{f} \in \mathcal{T}_i} (\mathbf{f}_2 - \mathbf{f}_1) \times (\mathbf{f}_3 - \mathbf{f}_1) \tag{7.3}$$

where $\mathcal{T}_i$ denotes the set of $N_i$ surface triangles which include vertex $i$. Each surface triangle $\mathbf{f}$ consists of vertex points $\mathbf{f}_1$, $\mathbf{f}_2$ and $\mathbf{f}_3$ - see figure 7.2

## 7.4 Non-linear elastic model

In each iteration a non-linear elastic FEM is used to displace the source mesh based on the "boundary" forces **R** computed above. This step is based on the work by Miller et al. defining a total Lagrangian explicit dynamics (TLED) FEM formulation [93]. An efficient parallel implementation has been described in detail by Taylor et. al [135]. A summary of their model is provided below.

The TLED FEM on a tetrahedral mesh leads to the standard equations of equilibrium, $\mathbf{M}\ddot{\mathbf{U}} + \mathbf{D}\dot{\mathbf{U}} + \mathbf{K}(\mathbf{U})\mathbf{U} = \mathbf{R}$, where **M** is a mass matrix, **D** is a damping matrix, $\mathbf{K(U)}$ is a stiffness matrix, **R** are the external forces, and **U** is the nodal displacements. Notice that **K** is a function of **U**. In this work we omit the dynamic terms $\mathbf{M}\ddot{\mathbf{U}}$ and $\mathbf{D}\dot{\mathbf{U}}$. Hence, for a given load of external forces we search for a configuration of nodal displacements **U** in which $\mathbf{K}(\mathbf{U})\mathbf{U} = \mathbf{R}(\mathbf{U})$. Note that in our case **R** also depends on **U** according to equations (7.1)-(7.3).

The TLED FEM method describes shape deformation and displacements with respect to the initial undeformed geometry. Element function derivatives precalculated from the initial mesh configuration are used in each iteration when calculating interior forces (stress and strain) reacting to an exterior load. With such formulation continuous recalculation of spatial derivatives of the element functions is avoided. Deformation is measured in terms of the *deformation gradient* tensor ${}^t_0\mathbf{X}$ which contains elements ${}^t_0X_{ij} = \frac{\partial^t x_i}{\partial^0 x_j}$. In words,

the deformation gradient describes the degree of displacement of material point coordinates ${}^t\mathbf{x}$ with respect to the initial position ${}^0\mathbf{x}$. By convention a left superscript describes at which configuration of the body a quantity is measured and left subscript describes the configuration the measurement is made with respect to [9]. The deformation gradient can be calculated from the vertex displacements and precomputed element shape function derivatives [93], [135]. Given linear tetrahedral element shape functions $h$ the deformation gradient can be calculated by ${}^t_0\mathbf{X} = {}^t\mathbf{u}_e^T \; {}_0\partial\mathbf{h}_x + \mathbf{I}$. Here ${}^t\mathbf{u}_e$ is a matrix of displacements of vertices in the element, ${}_0\partial\mathbf{h}_x$ is a matrix of the derivatives of the shape functions in the element, and $\mathbf{I}$ is the identity matrix.

The internal forces $\mathbf{K}(\mathbf{U})\mathbf{U}$ are calculated as a sum of elemental forces $\mathbf{K}(\mathbf{U})\mathbf{U} = \mathbf{F}(\mathbf{U}) = \sum_e {}^t\tilde{\mathbf{F}}^{(e)}$ where $e$ denotes the individual elements. For linear tetrahedral elements ${}^t\tilde{\mathbf{F}}$ can be evaluated as

$$ {}^t\tilde{\mathbf{F}} = {}^0V \; {}^t_0\mathbf{B}_L^T \; {}^t_0\hat{\mathbf{S}} \tag{7.4} $$

where ${}^0V$ is the volume of the undeformed tetrahedron, ${}_0\mathbf{B}_L$ is the strain-displacement matrix, which describes the relationship between nodal displacements and strain in an element, and ${}^t_0\hat{\mathbf{S}} = [{}^t_0S_{11} \; {}^t_0S_{22} \; {}^t_0S_{33} \; {}^t_0S_{12} \; {}^t_0S_{23} \; {}^t_0S_{13}]^T$ is the second Piola-Kirchhoff stress on vector form. As the element deforms ${}^t_0\mathbf{B}_L$ can be found by transforming the initial strain displacement matrix ${}^t_0\mathbf{B}_{L0}$ using the transformation gradient :

$$ {}^t_0\mathbf{B}_L^{(a)} = {}_0\mathbf{B}_{L0}^{(a)} \; {}^t_0\mathbf{X}^T \tag{7.5} $$

where $a$ ranges over the four vertices of element e. ${}_0\mathbf{B}_{L0}^{(a)}$ is given by:

$$ {}_0\mathbf{B}_{L0}^{(a)} = \begin{bmatrix} {}_0h_{a,1} & 0 & 0 \\ 0 & {}_0h_{a,2} & 0 \\ 0 & 0 & {}_0h_{a,3} \\ {}_0h_{a,2} & {}_0h_{a,1} & 0 \\ 0 & {}_0h_{a,3} & {}_0h_{a,2} \\ {}_0h_{a,3} & 0 & {}_0h_{a,1} \end{bmatrix} \quad (a = 1, 2, 3, 4), \quad \text{where } {}_0h_{a,i} = \frac{\partial h_a}{\partial x_i} \tag{7.6} $$

For the second Piola-Kirchhoff stress ${}^t_0\hat{\mathbf{S}}$ in (7.4) we have a choice of constitutive equation. We use a hyperelastic neo-Hookean model given by

$$ {}^t_0S_{ij} = \mu(\delta_{ij} - {}^t_0C_{ij}^{-1}) + \lambda {}^tJ({}^tJ - 1){}^t_0C_{ij}^{-1} \tag{7.7} $$

Here $\mu$ and $\lambda$ are the Lamé constants, $\delta_{ij}$ is Kronecker's delta, ${}^tJ = \det({}^t_0\mathbf{X})$, and ${}^t_0\mathbf{C}$ is the Right Cauchy-Green deformation tensor calculated by ${}^t_0\mathbf{C} = {}^t_0\mathbf{X}^T \; {}^t_0\mathbf{X}$.

## 7.5 Enforcing a rigid motion

Besides the elastic FEM, a rigid motion model has also been included in the registration method. This model is based on the work by Müller et al. on meshless deformation based on shape matching [97]. The basic idea is to fit

the system of particles $\mathbf{p}_i^d$ to a rigid translation and rotation. Thus we wish to minimise

$$\sum_i m_i \left( \mathbf{R}(\mathbf{p}_i^o - \mathbf{p}_{cm}^o) - (\mathbf{p}_i^d - \mathbf{p}_{cm}^d) \right)^2 \tag{7.8}$$

Here $m_i$ is the mass of particle $i$, $\mathbf{R}$ is a $3 \times 3$ rotation matrix, and $\mathbf{p}_{cm}^o$ and $\mathbf{p}_{cm}^d$ are the centres of mass of the original (undeformed) particle system and the deformed particle system respectively:

$$\mathbf{p}_{cm}^o = \frac{\sum_i m_i \mathbf{p}_i^o}{\sum_i m_i} \qquad \mathbf{p}_{cm}^d = \frac{\sum_i m_i \mathbf{p}_i^d}{\sum_i m_i} \tag{7.9}$$

In the following we use the following notation for positions relative to the centre of mass: $\mathbf{r}_i = \mathbf{p}_i^d - \mathbf{p}_{cm}^d$ and $\mathbf{q}_i = \mathbf{p}_i^o - \mathbf{p}_{cm}^o$ To find $\mathbf{R}$ we first look at the simpler problem of finding a general transformation matrix $\mathbf{M}$ that minimises $\sum_i m_i(\mathbf{A}\mathbf{q}_i - \mathbf{r}_i)^2$. For a matrix $\mathbf{M}$ to be a minimiser of this expression the derivatives of the expression with respect to all coefficients of $\mathbf{M}$ must be zero. This results in the following transformation [97]:

$$\mathbf{M} = \left( \sum_i m_i \mathbf{r}_i \mathbf{q}_i^T \right) \left( \sum_i m_i \mathbf{q}_i \mathbf{q}_i^T \right)^{-1} = \mathbf{M}_{rq} \mathbf{M}_{qq} \tag{7.10}$$

The symmetric matrix $\mathbf{M}_{qq}$ only contains scaling, i.e. no rotation. Consequently $\mathbf{R}$ can be found as the rotational part of $\mathbf{M}_{rq}$. This can be found by doing a polar decomposition of $\mathbf{M}_{rq}$ into the matrix product $\mathbf{M}_{rq} = \mathbf{RS}$, where $\mathbf{R}$ is the rotational part and $\mathbf{S}$ is the symmetric part. Having determined $\mathbf{R}$ all particles can be restricted to the rigid transformation of the original positions by changing their position to

$$\mathbf{p}_{i,rigid}^d = \mathbf{R}(\mathbf{p}_i^o - \mathbf{p}_{cm}^o) + \mathbf{p}_{cm}^d \tag{7.11}$$

## 7.6 Registration algorithm

An overview of the registration algorithm is given in algorithm 3. The search for a source mesh deformation in which the source and reference surfaces are matched while interior vertices are at equilibrium is based on a gradient descent type approach. In each iteration of the registration algorithm the total force, $\mathbf{T}(\mathbf{U})$, acting on the source shape is calculated as $\mathbf{T}(\mathbf{U}) = \mathbf{R}(\mathbf{U}) + \mathbf{F}(\mathbf{U})$. Each entry in this vector is used to update displacements ($\mathbf{U}_i \leftarrow \mathbf{U}_i + \delta \cdot \mathbf{T}_i$) with a maximum displacement change of $\Delta_{U,max}$. Note that algorithm 3 does not cover the rigid part of the registration. When the registration is restricted to rigid motion as explained in section 7.5 the interior forces are not computed. Instead all particles are just iteratively displaced along their external force vectors and in each step subsequently fitted to a rigid transformation.

The scaling factor $\gamma$ (Eq. (7.1)) is iteratively increased by multiplication with constant factor $\Delta_\gamma > 1$ whenever $\mathbf{T}(\mathbf{U})$ is approaching an equilibrium, i.e. when $\max(||\mathbf{T}_i||) < T_1$. The algorithm terminates when all interior vertices are

at rest and increasing $\gamma$ does not increase $\max(|\mathbf{R}_i|)$ at which point all surface vertices of the source mesh have been registered to the surface of the reference mesh. This leads to the termination criterion $\max(|\mathbf{T}_i|) < T_2$.

**Algorithm 3**: Overview of our FEM-based registration algorithm

```
Let source organ shape be represented as a tetrahedral mesh with
original vertex distances φ_S(p_i^o) known
Let reference organ shape be represented as a Euclidian distance field φ_R
Let β, τ, λ, and μ be known and constant
Let all vertex displacements U be initialized to zero
γ ← γ_0
repeat
    Calculate external forces R(U) from (7.2), (7.3) and (7.1)
    Evaluate internal forces F(U) using the FEM
    Calculate resulting force T(U) = R(U) + F(U)
    Update displacements U_i ← U_i + δ · T_i, with a maximum
    displacement change of Δ_{U,max}
    if max(|T_i|) < T_1 then
        γ ← γ · Δ_γ
until max(|T_i|) < T_2 ;
```

### 7.6.1 Implementation

Compute intensive parts of the proposed registration method was implemented on the GPU using the programming framework CUDA. The GPU implementation of the TLED algorithm {1} is similar to the implementation presented by Taylor et al. [135]. As the internal forces in the FEM are calculated per element the calculations in the TLED model can be massively parallelised. When accumulating forces from elementwise internal force computation, care must be taken to avoid write conflicts. Our solution is to associate four write indices to each tetrahedron which enable us to know where to store the force contributions in elementwise kernels. A per-vertex kernel is subsequently responsible for addition of force contributions.

For evaluation of the expressions of the type $\max(|\mathbf{T}_i|) < T$, a kernel has been written which inspects a $\mathbf{T}_i$ value and stores the value `false` in a fixed global memory position if $|\mathbf{T}_i| > T$. Before invocation of this kernel on all force vectors, `true` is written to the particular memory position, and if it is still `true` after all kernels have terminated the criterion $\max(|\mathbf{T}_i|) < T$ is fulfilled.

The computation of an optimal rotation matrix in the rigid transformation model has been implemented on the CPU. For the polar decomposition the "newmat" matrix library[2] was used.

[2]newmat can be found at `http://www.robertnz.net/nm_intro.htm`

## 7.7 Evaluation

As an investigation of the versatility of the registration method, a number of registration experiments have been conducted. Below the result of some of these experiments will be presented. The running times reported below are measured on an NVIDIA Quadro FX 5600 graphics card in a computer with an Intel core 2 6400 CPU.

### 7.7.1 Data sets

#### Modelling wax phantom

To evaluate the performance of the proposed registration method when registering meshes of an object that has undergone a bending motion, we performed experiments using a modelling wax cylinder phantom. The phantom was CT-scanned in three configurations: **(a)** straight **(b)** bent approximately 45 degrees and **(c)** bent 90 degrees. The phantom surface was manually delineated. Seven lead markers were positioned on the surface of the phantom for evaluation purposes - see figure 7.3.

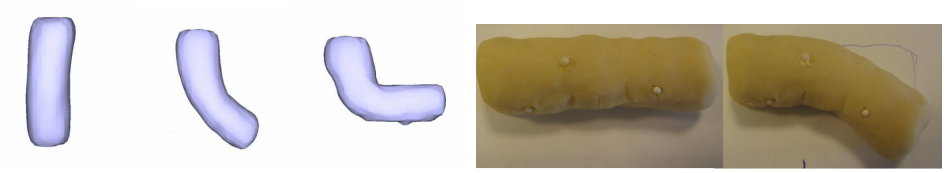

Figure 7.3: The wax cylinder used in the validation experiments. Left: three different configurations (1) to (3) of the vax cylinder (reconstructed surface). Right: photographs of the phantom.

The following parameters were used for the registration algorithm: $\beta = 0.01$, $\gamma_0 = 1000$, $\lambda = 5$ kPa, $\mu = 10$ kPa, $\Delta_\gamma = 1.03$, $\Delta_{U,max} = 0.005$ mm, $\tau = 5$ mm, $\delta = 0.01$, $T_1 = 30$ kN, and $T_2 = 3$ kN.

#### Prostate data set

To evaluate the algorithm on actual morphology acquired in vivo, we registered two manually delineated prostates from successive MRI scans of a prostate cancer patient. As no ground truth about the deformation between the two configurations **(a)** and **(b)** of the organ is available, an evaluation was performed based on inverse consistency of the resulting transformation. For prostate registration parameters were set as above with the following exceptions: $\lambda = 0.33$ MPa, $\mu = 3.3$ kPa, and $\Delta_\gamma = 1.07$.

#### Cervical cancer BT MR image

We have tested the proposed registration method on organs delineated on two MR images that have been acquired in connection with cervical cancer brachytherapy treatment - see figure 7.4. Parameters used were identical to the

ones used on the prostate data set. The meshes will be referred to as configuration **(a)** and **(b)** respectively. Before creation of meshes the two images were rigidly aligned based on a manual landmark match on the pelvic bone.

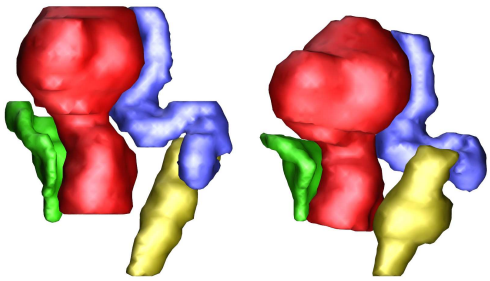

Figure 7.4: Visualisation of tetrahedral meshes **(a)** and **(b)** generated from delineations of organs on two MR images acquired in connection with brachytherapy of cervical cancer. Green: bladder, red: uterus, blue: sigmoid colon, and yellow: rectum.

### 7.7.2 Results

#### Modelling wax phantom registration

A mesh consisting of 6034 tetrahedra was constructed from configuration **(a)** of the phantom cylinder and registered to configurations **(b)** and **(c)**. The two registrations were completed in 9 seconds and 17 seconds respectively. The results are shown in figure 7.5. The marker positions were expressed in barycentric coordinates on an associated surface triangle and tracked to the final deformed configuration of the mesh. For each marker a registration error was calculated as the length of the difference vector between the registered marker position

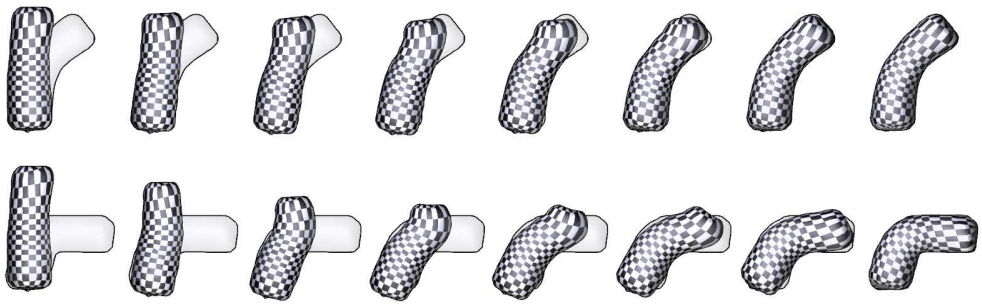

Figure 7.5: Iterative registration of a modelling wax cylinder from a straight configuration to two configurations with different degrees of bending (top and bottom respectively). The leftmost images depict the initial source mesh configurations with the reference mesh overlaid transparently. The rightmost images depict the final deformation. Intermediate steps in between show every 1000th iteration (top) or every 2000th iteration (bottom). A chequerboard pattern has been mapped to the surface for easier movement tracking {2}{3}{4}.

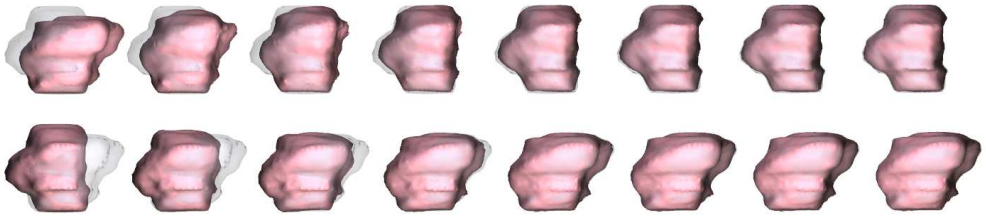

Figure 7.6: Two-way registration of two prostate meshes obtained from successive MRI scans of a prostate cancer patient. The leftmost images depict the initial configurations of the source mesh with the reference mesh overlaid transparently. The rightmost images depict the final deformation. Intermediate steps in between show every 1000th iteration {5}.

and the corresponding marker positions in the reference mesh. For the registration from configuration **(a)** to configuration **(b)** the initial error was 14.2 mm ± 12.7 mm (average ± std. dev.). After registration the error was reduced to 1.33 mm ± 1.39 mm. When registering configuration **a** to configuration **c** the error was reduced from 26.9 mm ± 24.8 mm to 2.03 mm ± 1.03 mm.

### 7.7.3 Prostate registration

From the prostate delineations in two MRI datasets meshes were created consisting of 49005 **(a)** and 50697 **(b)** tetrahedra respectively. Two registrations were made: from **(a)** to **(b)** and **(b)** to **(a)**. The running times of the two registration were 20 seconds and 28 seconds respectively. The results are shown in figure 7.6. For 5000 randomly selected points inside prostate configuration **(a)** registration consistency errors were evaluated by transforming the point from configuration **(a)** to configuration **(b)** and back. The average inverse consistency errors were calculated as the average length of each vector between the initial and final points. The average obtained errors were 0.955 mm ± 0.469 mm (max = 2.62 mm). This should be seen relative to the registration displacements of these same points: 6.25 mm ± 2.83 mm ( max = 16.4 mm ). The distribution of the inverse consistency errors can be seen in figure 7.7. As the inverse consistency error can be biased, it can only be used as an indication of accuracy. To adequately evaluate the performance of the proposed method for prostate registration, a larger study must be made involving known markers that can be used for validation.

### 7.7.4 Registration of female pelvic organs

In the following registration of organs from repeated MR acquisitions of the female pelvis is validated by visual inspection of the final result as well as the movement of the organ surface during registration. This is far from sufficient for a thorough examination of accuracy and/or robustness of the method but serves as an initial exploration of possible uses for registration of pelvic organs.

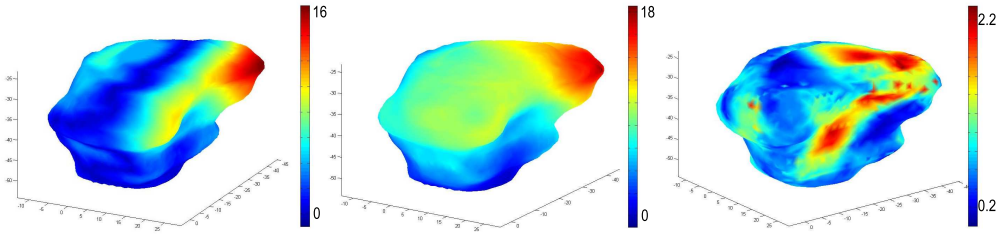


Figure 7.7: Inverse consistency errors. Left: distance from source shape surface to surface of reference shape. Middle: Magnitude of deformation found by the registration. Right: Distribution of consistency errors 2 mm below surface. All measures are in millimetres.

### Registering the uterus

Registration of the uterus from configuration **(a)** to **(b)** can be seen in figure 7.8. Based on visual inspection of the resulting registered surface, we find the registration quite successful. However for this particular case a rigid registration might be sufficient to achieve sufficient accuracy. The registration took approximately 1 minute for a mesh size of 4529 vertices and 21107 tetrahedra.

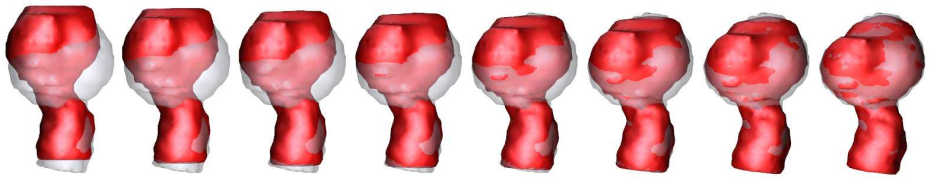

Figure 7.8: Registration of the uterus from configuration **(a)** to configuration **(b)**. The source shape is shown in red and the reference shape in transparent grey. Every 3000th iteration is shown.

### Registering the sigmoid colon

The sigmoid colon may deform to a degree where a rigid registration isn't sufficient. With regards to the sigmoid this case is quite well posed because the delineation produces a tubular structure. This is in general not the case as the sigmoid may twist so much that it is hard to automatically identify a tubular structure from the contours. Motion of the organ surface during registration from **(a)** to **(b)** can be seen in figure 7.9. Notice that the source surface intersects itself during registration. This is of course not acceptable. In this case the self-intersection resolves itself during further registration. The problem of self-intersections will be discussed further in section 7.8. Initiating the registration with a rigid registration (as described in section 7.5) solves the problem of self-intersection in this particular case (see figure 7.10). The registration took approximately 1 minute for a mesh consisting of 3443 vertices and 14564 tetrahedra when using only the elastic model. Using the rigid model

for the initial match saves 20 seconds of that time.

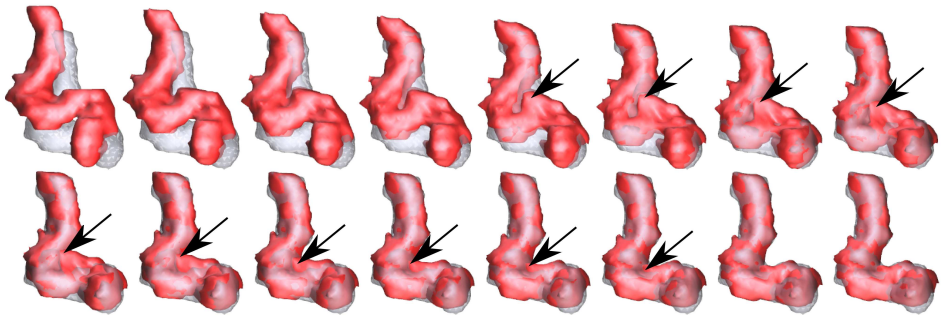

Figure 7.9: Registration of the sigmoid colon from configuration **(a)** to configuration **(b)**. The source shape is shown in red and the reference shape in transparent grey. Every 3000th iteration is shown. Arrows indicate where self intersection occurs.

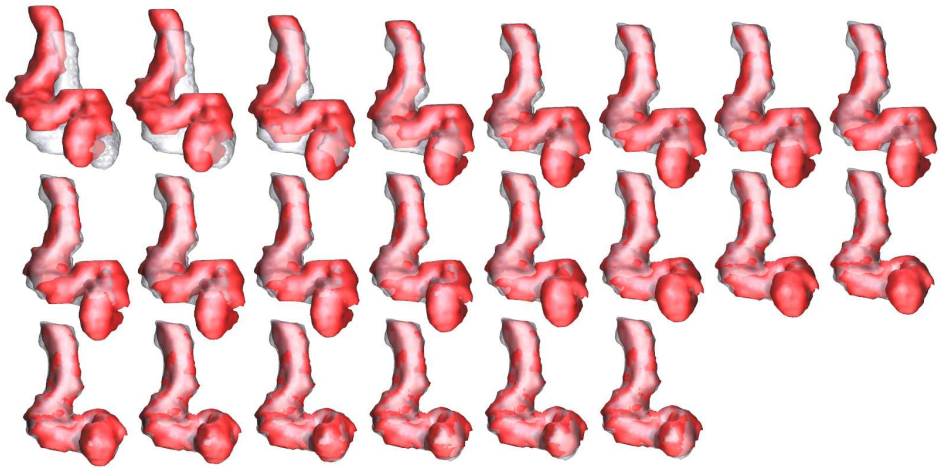

Figure 7.10: Registration of the sigmoid from configuration **(a)** to configuration **(b)** using first a rigid motion model followed by the non-linear elastic FEM. The source shape is shown in red and the reference shape in transparent grey. Every 200th iteration is shown for the rigid motion (the first 8 steps) and every 3000th iteration for the deformable part.

### Registering the rectum

For the rectum difference in filling means that configuration **(b)** is wider than **(a)**. As we do not wish to register the rectum content, meshes are generated only for a wall of 6 mm or less below the surface. The value of 6 mm is a compromise chosen as a consequence of our implementation of the mesh generation method which needs the wall size to be at least double the BCC grid distance. Having a thinner wall requires a smaller tetrahedron size and thus longer registration time.

Initial attempts of registering the rectum from **(a)** to **(b)** without initial rigid motion were unsuccessful and as a consequence of this the motion was restricted to be rigid until the shapes were estimated to be rigidly aligned by visual inspection. After this the deformable registration proceeded as in the previous registration experiments. The registration is visualised in figure 7.11. Despite the rigid part the registration still fails because the bladder wall is attracted to the wrong side of the reference surface. Registration of the rectum from **(b)** to **(a)** is shown in figure 7.12. This time the registration is more successful.

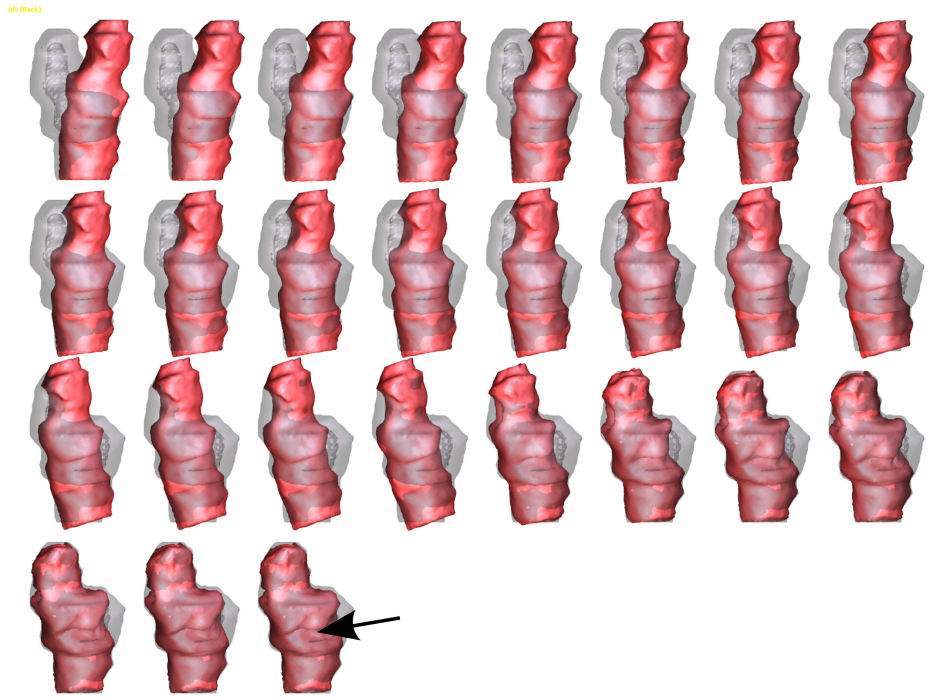

Figure 7.11: Registration of the rectum from configuration **(a)** to configuration **(b)**. The source shape is shown in red and the reference shape in transparent grey. An arrow indicates where self intersection occurs. The arrow indicates an area of the rectum wall that is attracted towards a wrong destination. Every 3000th iteration is shown for the FEM based registration and every 200th iteration for the rigid part (the first 19 steps shown).

### Registering the bladder

For the bladder meshes shown in figure 7.4 our registration method does not work well. In this case the main problem is our mesh generation method which is not well suited for generating meshes for this delineation without connecting opposite bladder walls. In trial runs of the method on other bladder delineations where the bladder volume changed a lot we often ran into problems with either self-intersections (registering from high to low volume) or that too large forces were needed making the system unstable (registering from low to high volume).

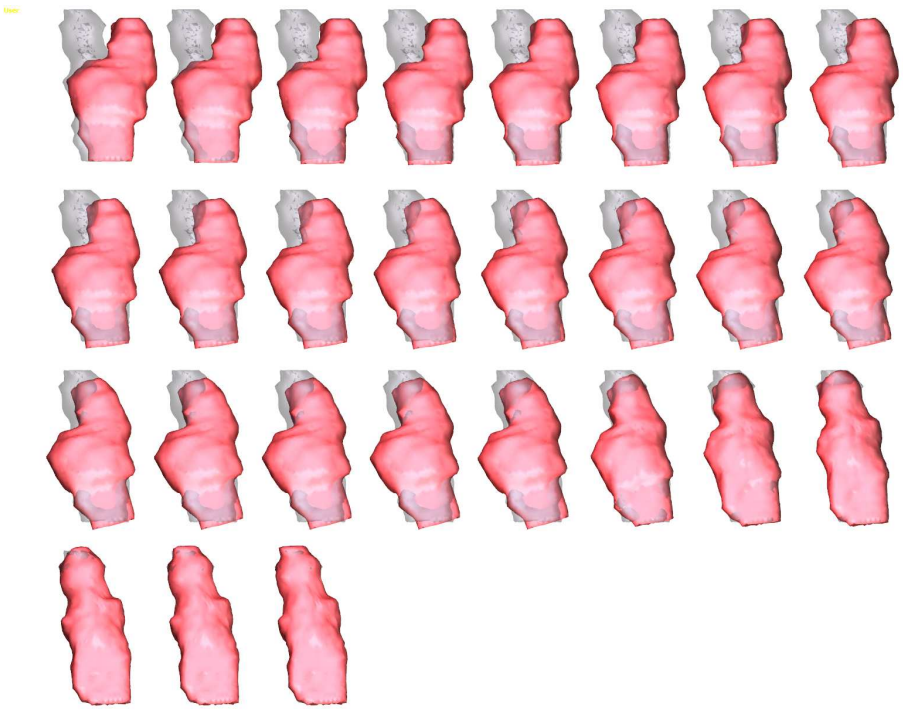

Figure 7.12: Registration of the rectum from configuration **(b)** to configuration **(a)**. The source shape is shown in red and the reference shape in transparent grey. Every 3000th iteration is shown for the FEM based registration and every 200th iteration for the rigid part (the first 20 steps shown).

In the latter case the risk exists that a tetrahedron will be inverted causing the registration to "explode".

## 7.8 Discussion

Considering the accumulation of inherent errors such as contouring errors, finite image resolution etc., we are quite satisfied with the accuracy achieved in the phantom experiment. With respect to the prostate registration we find a mean inverse consistency error of less than 1 mm very acceptable as well. However as the ground truth in the latter experiment is unknown, this does not necessarily imply that the actual error is this low. Clinical studies including patients with implanted fiducial markers or other consistent anatomical landmarks within the segmented organs are needed to further evaluate the registration accuracy. The magnitude of errors introduced in the contouring process has not been investigated for any of the reported registration experiments.

The proposed registration method includes a number of free parameters that must be defined initially. The algorithm is quite robust w.r.t. the choice of most of these parameters. Note however that when different types of tissue are deformed, it is necessary not only to change the material parameters $\lambda$ and $\mu$ but also to adjust $\Delta_\gamma$ which governs rate of increase of driving forces during

the registration process.

By incorporating a physically consistent deformation model for the transformation of vertex positions, it is possible to specify measured material properties for the morphology being registered. When using different parameters for e.g. tumour tissue and surrounding healthy tissue, we can furthermore register non-homogeneous materials. Again, it is left for further studies to investigate this effect on the registration accuracy.

The introduction of the force term **B** was motivated by the inability of the distance field term to lead the surface nodes into narrow boundary concavities. An alternative to introducing the **B** term would be using the *gradient vector flow* approach of Xu and Prince [149]. Based on a diffusion of gradient vectors, their approach creates a vector field which points towards the surface but also points into narrow concavities.

As exemplified with the above rectum and bladder cases our method does not in general work with organs that empty and fill, i.e. cases where we model organ walls as "solid" structures. Based on our experience using the registration method so far we have identified three ways that the proposed method can fail:

1. Too large volume difference between organ shape requires stretching or compression of tetrahedra to an extent that induces a risk of inverted tetrahedra.

2. The organ surface deforms to intersect itself during registration.

3. The organ surface is attracted towards a part of the surface that we know is not correct.

The first of these issues is perhaps the most problematic as it is linked to the discretisation in the biomechanical model. Using larger tetrahedra would increase the stability but would restrict us from having high detail in our organ meshes. Possibly a graded mesh generation (different tetrahedron sizes inside the organ) would alleviate the problem somewhat. At present the conclusion must be that the proposed method is only suitable for registering organs that do not change volume too much.

Standard ways of avoiding self-intersection in physical modelling is to include a collision handling step in the simulation loop. A possible approach is described by Fisher and Lin in [43]. Here distance fields are continually deformed according to object deformation which allows for easy collision detection.

As our force calculations only use information from the vicinity of a particular vertex it is not straightforward to determine whether a particle is attracted towards the "right" surface. When registering solid objects with little volume change this is usually not a problem as the regularisation performed by the biomechanical model will ensure that opposite surfaces in the source mesh will also eventually be mapped to opposite surfaces in the reference configuration. For hollow objects the problem can be alleviated by weighting the inflation force term **B** higher. A solution we are considering is to allow a user to intervene in the registration by interactively selecting landmark positions on source and reference surfaces to guide the registration. The philosophy here would be that

the user is not required to give exact point correspondences but just specify extra forces to aid the registration. Like the force terms **A** and **B** these forces would then slowly disappear as surface match is approached.

Due to the nature of the proposed algorithm it can easily be parallelised and implemented e.g. on a GPU to achieve high performance. We find execution times in the order of 20-60 seconds acceptable for potential clinical use. Due to the parameter $\Delta_{U,max}$, which limits the magnitude of displacement changes in each iteration, the running time is highly dependent on the initial magnitude of deformation between source and reference shapes. Thus registration time does vary from case to case.

The registration time reported above does not include the time spent on calculation of distance fields, which in the currently used implementation can be significant. Fortunately GPU based distance field computation has been investigated by several parties which have resulted in e.g. the rasterisation based methods found in [133] and [39].

The problems faced when attempting to use the method presented in this chapter for bladder registration has inspired us to develop a new method for registration of the bladder, which instead of volumetric modelling of the bladder wall is based on a membrane model. This method will be presented in the next chapter.

# Chapter 8

## Surface membrane registration

### 8.1 Introduction

The finite element based elastic registration method introduced in the previous chapter is not well suited for cases where the organ being registered undergoes large changes in volume. Such cases occur especially when registering organs that empty and fill like intestines or the bladder. In these situations we are however not very interested in registration of interior morphology. As e.g. bladder content present in one image will be gone in another, a point matching of this filling to positions in an empty bladder does not make much sense. What we are interested in registering in this case is the bladder wall.

As the bladder can drastically change its shape between two image aquisitions we cannot rely on e.g. curvature correspondences for creating a bladder surface mapping. In this chapter a method for creating a match of surface morphology between an organ segmented in two images is presented. This method is based on minimising a surface membrane energy. Furthermore it is possible to guide the registration by specifying landmark point correspondences.

### 8.2 Related work

The registration of organ surfaces has attracted the attention of multiple authors in the field of radiotherapy. Also, many of the methods we will refer to in this section have been developed for registration of cortical structures in the brain.

Kaus et al. [63] propose an automated surface registration model in which a rigid registration of a triangle mesh to a binary image is followed by a non-rigid deformable registration model that is inspired by the active contour models of Kass et al. [61]. This model has been applied for registration of lung, liver and prostate [62], where displacement of organ surface vertices is interpolated to the interior using radial basis functions.

Many methods for finding point correspondences involve creating a mapping from the object surface and onto a sphere. This is always possible if the organ surface is a *genus zero surface* which means that it must be closed, smooth, and have no self-intersections or handles. Brechbühler et al. set up a constrained

optimisation problem for mapping a cuberille[1] onto a sphere [15]. To get a sensible initial mapping for optimisation they simulate a heat conduction between a hot and a cold pole. After optimisation the spherical parameterisation is expanded as a series of spherical harmonic functions. Kelemen et al. utilised this approach for creating a model for brain segmentation based on principal component analysis [64]. The work by Brechbühler et al. was improved by Quicken et al. who solved the parameterisation problem for general triangulated surfaces using a multiresolution approach [114]. Also Price and Moore have extended the approach of Brechbühler et al. to triangular surfaces [113]. After the spherical parameterisation and creation of the spherical harmonic series expansion the source triangle mesh sphere is rotated to align with the reference. This is done by matching the first order ellipsoids from the series expansion guided by a few point landmarks. Fischl et al. were among the first to perform brain registration in spherical space [42]. Alignment of folding patterns of the cortical surface is performed by minimising the squared difference in convexity between a new subject and an average convexity of a population previously registered. The measure of convexity is formulated in the spherical space. Their spherical parameterisation [41] was based on minimising squared changes in distances between each vertex in a triangular mesh and a neighbourhood of vertices around it. Also triangle flipping was penalised using a term calculated from the oriented area of triangle faces. Their approach is somewhat similar to the internal energy minimisation we propose in this chapter. Before projection to a sphere and minimisation of the above terms, the cortex is "inflated" while the surface is held together using a spring-mass system (more on this below).

Based on the work on creating a conformal mapping between any genus zero surface and a sphere [1] [53], Haker et al. map two delineations of a prostate to a sphere for registration [54]. Using the same method Xiong et al. register bladder surfaces [148]. Given three point matches, the transformation between two conformal mappings is uniquely determined. Gu et al. present another method for creating conformal mappings [47] and use this for registering meshes of brains guided by landmark curves.

In the current work we utilise a 2D grid representation of an object surface. In such a grid each grid point contains a 3-dimensional vector describing a position on the object surface. This kind of representation was introduced by Gu et al. who call these *geometry images* [46]. Their method for creating geometry images was based on cutting a network of edges between triangles and mapping the cut onto the image boundary. Joshi et al. used a method similar to this for mapping a brain surface onto a unit square [60]. In this parameterised space brain surfaces were registered based on landmark lines and linear elasticity. Praun and Hoppe present another method for creating geometry images [112]. They create a spherical parameterisation by minimising a stretch based measure and subsequently map this parameterisation onto a polyhedron that is unwrapped to form a square.

As will be presented later we use a method similar to that of Fischl et al. [41]

[1] a cuberille is a surface consisting of square faces separating the foreground and background voxels of a binary grid

for projecting a closed surface onto a sphere. The method is physically motivated and simulates the "inflation" of a sphere inside an elastic membrane. The membrane model used is a *spring-mass system* [13] which is a simple model for animation of deformable objects which is used in many contexts - e.g. surgical simulation [96] and computer games [58]. The model is based on point masses connected by massless springs. When springs are stretched or contracted a restoring spring force is generated which is proportional to the change in length. The idea of using a spring-mass system in the process of creating a spherical parameterisation was first suggested by Kent et al. [65]. Matuszewski et al. have used spring-mass systems for image registration [91]. Here the external forces driving the deformation were based on the cross-correlation similarity measure. Söhn et al. have also used a spring-mass model in image registration [127]. Here the role of the spring-mass system was regularising the displacements of a set of "featurelets" that each were guided by anatomical features in a subvolume of the image surrounding the featurelet.

## 8.3 An overview of the proposed method

In the following we assume that the organs to be registered have been contoured in both source and reference images. These segmentations are furthermore assumed to be represented as triangular surface meshes. A standard way to achieve this is to rasterise segmentation contours into a binary mask and use the marching cubes method for creating the triangle mesh [81]. Another possibility is applying contour stitching techniques [5]. We have used our implementation of the isosurface stuffing method (see section 7.2) for generating tetrahedral meshes [75] and retrieved a triangle based surface mesh by simply taking those triangle faces from the tetrahedra which have all three corners on the organ surface.

The main idea in the surface registration method presented in this chapter is to parameterise the surface of the reference organ in a 2D space. This results in a *position mapping* between $(u, v)$ coordinates in this two-dimensional space and $(x', y', z')$ coordinates in physical space. The $(u, v)$ space is represented as an ordinary two-dimensional grid (a geometry image) allowing us to quickly look up a position in physical space from a $(u, v)$ coordinate and also allowing simple linear interpolation to be made. From the geometry image derivatives of position with respect to the parameters $u$ and $v$ are computed and stored in similar maps. As an initial step both the source and reference surfaces are projected onto a sphere. As will be described below this is done by simulating a sphere inflation inside the respective surface while modelling the surface as a rubber membrane. Positions on the spheres are mapped to (u,v)-coordinates using polar coordinates. From the $(u, v)$-mapped reference triangle mesh a geometry image is generated. The $(u, v)$-mapping of source mesh vertices found by sphere inflation is used as an initial mapping into the geometry image.
Registering the source surface to the reference surface then amounts to optimising the mapping from vertices in the triangular mesh representing the source surface into the $(u, v)$-space representing the reference surface. The optimal

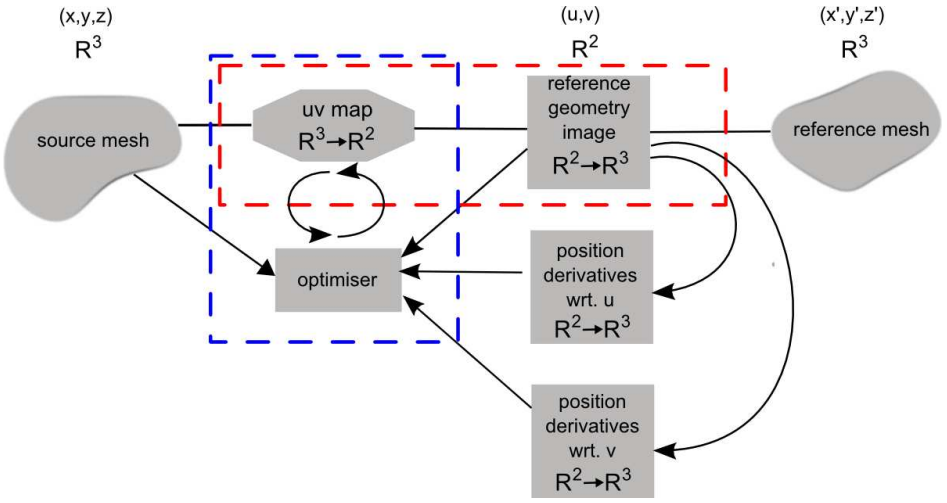


Figure 8.1: An overview of the presented surface matching method. The red box contains the components constituting the transformation between source and reference shape. The blue box emphasises the optimisation part.

mapping sought for is minimising the potential elastic energy in a membrane model of the source surface while the vertices of this surface are constrained to move on the reference surface. For an overview see figure 8.1. The entire transformation (red box) between shapes is constituted by the reference geometry image and the $(u, v)$-map from source mesh vertices to $(u, v)$-positions in the geometry image. The object of the optimisation (blue box) is to optimise this $(u, v)$-mapping. For this the source mesh of original positions, the geometry image, and derivatives of the geometry image with respect to $u$ and $v$ are needed.

The membrane model used in the current work is a spring-mass system. Energy minimisation is performed iteratively in a gradient descent-like manner. For each source mesh vertex a force which leads to a lower potential energy is computed in 3D space and projected onto the tangent space of the reference surface at the current $(u, v)$-position. The projected force vector is then used to move the $(u, v)$-coordinates of the vertex. Besides the membrane energy minimisation, landmark point matches can be specified in 3D space and included in the optimisation via a landmark matching force term.

Our main contribution in this work is the idea of using geometry images for constraining the positions of source vertices while formulating the energy term to optimise in Cartesian 3D space. To our knowledge ours is the only method based on this approach. We believe the approach is more intuitive than the optimisation of stretch or curvature based measures in the parameterised 2D space used in methods mentioned in the previous section. Furthermore our approach allows us constrain surface particles during minimisation of internal energy in any elastic model as long as the minimisation can be performed by iterative displacement of vertex positions.

## 8.4 Creating surface maps

The 2D parameterisation of an organ surface is based on projecting the triangular mesh representing the organ onto a sphere and parameterising based on spherical coordinates. As all mesh vertices are on the sphere surface the radial component $r$ of the spherical mapping is constant and thus redundant, creating a representation of surface points in a two-dimensional space. As illustrated in figure 8.2 the azimuthal angle $\phi \in [0, 2\pi]$ is mapped to $u \in [0, 1]$ and the elevation angle $\theta \in [-\pi, \pi]$ is mapped to $v \in [0, 1]$.

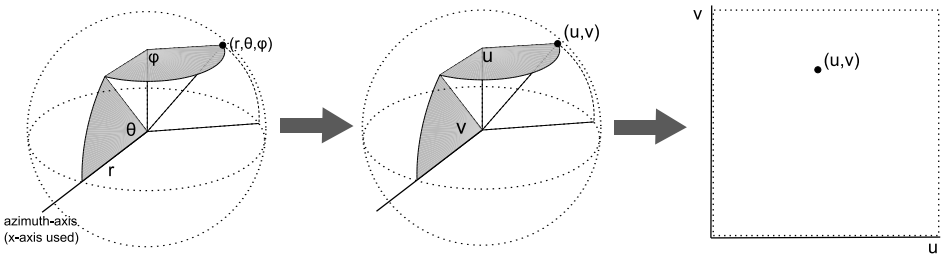


Figure 8.2: Going from spherical coordinates to a two-dimensional space

### 8.4.1 Projecting organ surface onto a sphere

As discussed multiple methods exist for mapping a surface mesh onto a sphere. We have chosen to use a physically motivated model in which a sphere is "inflated" inside the organ and surface vertices are restricted to stay outside this sphere. This choice is mostly motivated by the fact that we had already implemented this model at an earlier time. A spring-mass system is applied on the organ surface to ensure that surface vertices stay connected during the sphere inflation. *Verlet integration* [58] is used for evolving the model in time. The sphere inflation is continued until all surface vertices are positioned at a distance from the sphere centre corresponding to the sphere radius - see figure 8.3.

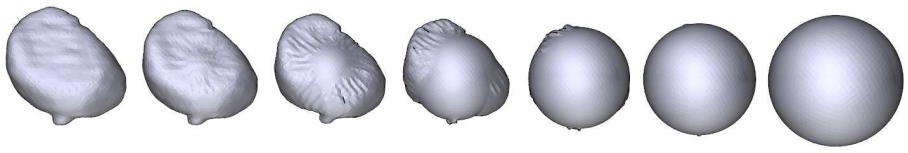

Figure 8.3: Projection of a bladder surface onto a sphere. The sphere radius is increased until all vertices are positioned on the sphere surface {6}.

Having subtracted the sphere centre from a surface vertex position and normalised in the radial direction so that the position is mapped to the unit sphere, the coordinates $u_i$ and $v_i$ for vertex $i$ are computed from sphere surface coordinate $(\bar{x}_i, \bar{y}_i, \bar{z}_i, )$ by:

$$u_i = \frac{\arctan(\bar{x}_i/\bar{y}_i)}{2\pi} + 1/2 \quad (8.1)$$

$$v_i = \frac{\arccos(\bar{z}_i)}{\pi} \quad (8.2)$$

### 8.4.2 Creating the geometry image

The surface projection described above creates a mapping from surface vertices to a position$(u, v)$ in the parameterised space. We want to create a map allowing fast determination of which physical point $(x, y, z)$ on the organ surface corresponds to a given parameter coordinate $(u, v)$. As described below our approach is to sample the $(u, v)$ space $[0, 1] \times [0, 1]$ in a regular grid. This allows us to evaluate any $(u, v)$ position by doing bilinear interpolation between sampled values.

It is not straight-forward to create the geometry image by simply rasterising surface triangles into the $(u, v)$ map because the space is non-linear - especially near the poles of the sphere (see figure 8.4 right). Inspired by work on ray tracing [126] and collision detection [85] we instead send a ray from the sphere centre along the line determined by the angles corresponding to a particular combination of $u$ and $v$ and compute which triangle this ray will intersect - see figure 8.4 left. Ray-triangle intersection testing is accelerated using a *bounding volume hierarchy* (BVH) [126, chapter 9]. In this technique 3D space is partitioned in a tree of volumes (we use spheres) each bounding an entire subtree. Intersection testing is done by traversing the tree, avoiding intersection test on triangles inside spheres determined not to be hit by the ray. It is furthermore possible to avoid traversing the BVH in many cases, as rays corresponding to neighbouring positions in the map often hit the same or neighbouring triangles [85]. Once a ray-triangle intersection is determined, barycentric coordinates are computed and used to interpolate between the original positions of the surface triangle vertices. Due to the singularities near the poles, when $v = 0$ or $v = 1$ all values for $u$ will result in a mapping to the same position (the position of the pole). To avoid over-sampling geometry mapped near the poles, no rays are sent for $v < 0.005$ or $v > 9.995$. Instead the interval $v \in [0.005, 9.995]$ is stretched to cover $v \in [0, 1]$ in the geometry image.

Following the creation of the geometry image, second order finite difference approximations are used to compute derivatives of the 3D position with respect to $u$ and $v$ respectively. Looking up neighbouring values when evaluating the approximations is handled as illustrated in figure 8.5. In the event that the $u$ component is outside the interval $[0, 1]$ it can be transformed into the interval by "wrapping" as shown in the figure. If $v \notin [0, 1]$ the $u$ coordinate must be mirrored around the edge and the $v$ coordinate is transformed by $v_{new} = 3/2 - v$. These precautions reflect the spherical coordinate system forming the basis of our $(u, v)$ map and allow us to extend the mapping to a periodic, continuous function of $\mathbb{R}^2$. Note that because we have pre-calculated our $u$ and $v$ derivatives in the case where $u \in [0, 1]$ and $v \in [0, 1]$ we must take this into account when looking up derivatives in the case $v \notin [0, 1]$. This is done by negating the

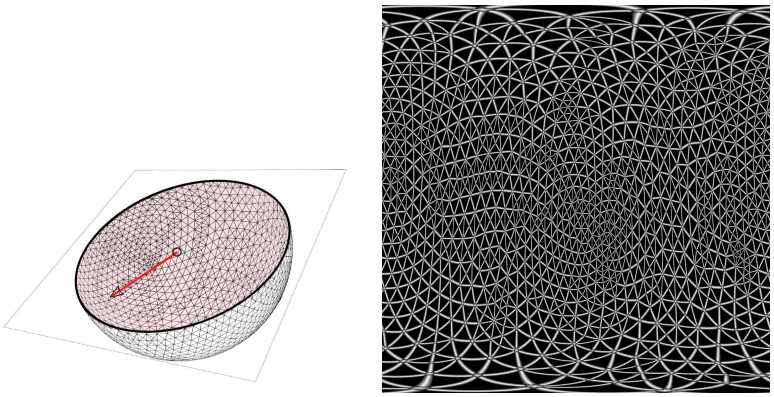

Figure 8.4: **Left:** Casting rays from the sphere centre to evaluate the boundary position corresponding a given $(u, v)$ coordinate (half of the sphere has been removed for illustrative purposes). **Right:** Visualisation of how the triangles of a bladder surface mesh are positioned after a projection onto the $(u, v)$ space

derivative because $u$ and $v$ axes are inverted (see figure 8.5).

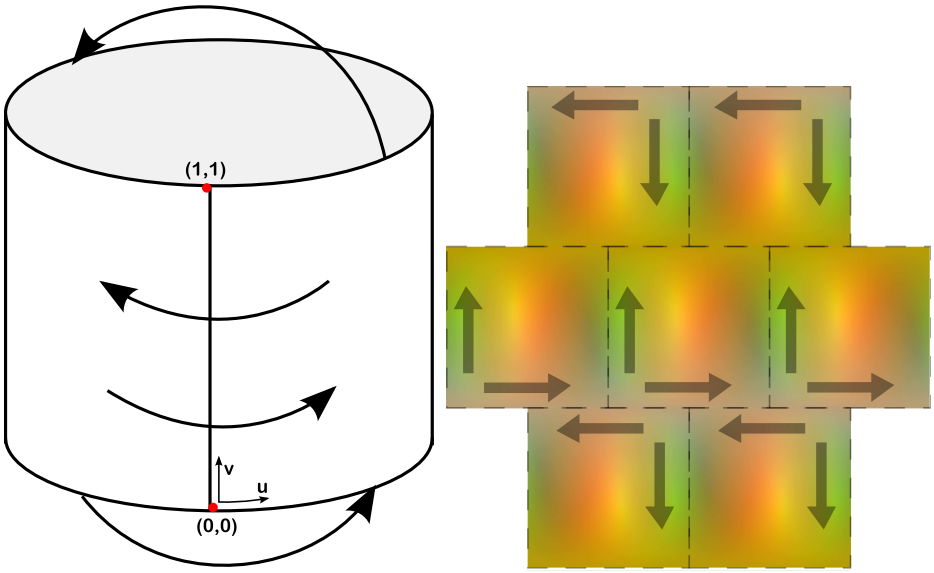


Figure 8.5: **Left:** Handling of boundaries in the $(u, v)$ maps achieving continuity. **Right:** The position map tiles as shown. Arrows indicate direction of $u$ and $v$ axes. Colours in the map encode positions, showing $x$,$y$, and $z$-components in the red, green, and blue color components respectively.

## 8.5 Optimising vertex $(u, v)$ mapping

As result of the above we now have a 2D map in which any position corresponds to a position on the surface of the reference organ. Let us denote this map by

$\mathbf{p}(u,v):\ \mathbb{R}^2 \mapsto \mathbb{R}^3$. Furthermore we have mappings from $(u,v)$ coordinates to derivatives in $u$: $\mathbf{d}_u(u,v):\ \mathbb{R}^2 \mapsto \mathbb{R}^3$ and $v$: $\mathbf{d}_v(u,v):\ \mathbb{R}^2 \mapsto \mathbb{R}^3$.

Using the geometry image we can move source vertices around in the $(u,v)$ space and are thus implicitly moving the vertices on the reference organ surface. The initial mapping generated by sphere inflation is not scale preserving, so we need to shift source vertex from its initial $(u,v)$ position to a position in which the system is at rest as defined by some elastic model. In the following a spring-mass model is used, but the parameterisation presented here can be incorporated into any model that relies on iterative displacement of vertices for optimisation.

### 8.5.1 The spring-mass model

We are interested in distributing the vertices of the source organ surface mesh over the surface of the reference organ. This is done by searching for a point distribution minimising the potential energy in a spring-mass system. Each vertex is connected to a number of neighbours by Hookean springs (springs that generate a restoring force that is linearly proportional to the displacement from a rest length). Denoting the number of vertices by $M$ and the number of neighbours of vertex $i$ by $N_i$, the energy we wish to minimise can be written as

$$E = \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{N_i} k_{ij}(c_{ij} - o_{ij})^2 \tag{8.3}$$

where $c_{ij}$ is the distance between vertices $i$ and $j$ in the transformed configuration, $k_{ij}$ are spring constants, and $o_{ij}$ is the distance in the original (untransformed) state. The current neighbour distance (spring length) between $i$ and $j$ can be found as $c_{ij} = ||\mathbf{p}(u_j,v_j) - \mathbf{p}(u_i,v_i)||$. The original (rest) length $o_{ij}$ can be pre-computed from the initial mesh.

One strategy for minimisation of the energy in (8.3) is to sum force contributions from all neighbour springs to a vertex and displacing the vertex in the direction of the resulting force. How this displacement is done in $(u,v)$-space will be described below. In $(x,y,z)$ space the resulting force for vertex $i$ is calculated by:

$$\mathbf{f}_i = \sum_{j=1}^{N_i} k_{ij}(c_{ij} - o_{ij})\frac{\mathbf{p}(u_i,v_i) - \mathbf{p}(u_j,v_j)}{c_{ij}} \tag{8.4}$$

Our experience with this strategy is that there is a high risk that triangle flipping will occur (that one or more triangles get inverted, facing their original back side outwards). As a consequence we instead adopted another strategy for calculating a force on vertex $i$. We find the neighbour $m$ where the ratio $\frac{||\mathbf{p}(u_m,v_m)-\mathbf{p}(u_i,v_i)||}{c_{im}}$ is largest and calculate the force as:

$$\mathbf{f}_i = k_{im}(c_{im} - o_{im})\frac{\mathbf{p}(u_i,v_i) - \mathbf{p}(u_m,v_m)}{c_{im}} \tag{8.5}$$

Concerning what connectivity to use for the spring-mass system, there are multiple choices. I have used the set of vertices reachable from vertex $i$ by following

one, two, or three triangle edges (see figure 8.6 for an illustration of the one- or two-step neighbourhoods).

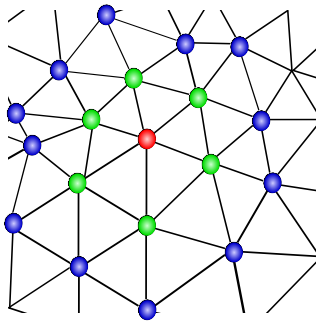

Figure 8.6: The neighbourhood of vertex $i$ shown in red consists of all vertices reachable by following one or two triangle edges (shown in green or blue respectively).

### 8.5.2 Landmark matching forces

As a supplement to the membrane energy minimisation, landmark correspondences can be specified as a way of adding prior knowledge into the registration process. Landmarks are positioned in physical 3D space and need not correspond to any vertex in the source or reference meshes. In figure 8.7 our strategy for landmark matching is illustrated. The red sphere represents landmark number $k$ at position $\mathbf{q}_k^{src}$ in source $(x, y, z)$ space and the yellow sphere represents the corresponding landmark at position $\mathbf{q}_k^{ref}$ in reference $(x', y', z')$ space. The source landmark influences those source vertices that are positioned closer than $r_l$ to $\mathbf{q}_k^{src}$ *in the undeformed source configuration* (the sphere of influence is illustrated in grey). Denote distance between source position of landmark $k$ and original position of source vertex $i$ by $l_{ik}^o$ , and the distance between reference position of landmark $k$ and the deformed position of source vertex $i$ by $l_{ik}^d = ||\mathbf{q}_k^{ref} - \mathbf{p}(u_i, v_i)||$. For each of the influenced vertices a landmark matching force $\mathbf{g}_{ik}$ is computed as:

$$\mathbf{g}_{ik} = w(l_{ik}^d, l_{ik}^o) \frac{\mathbf{q}_k^{ref} - \mathbf{p}(u_i, v_i)}{l_{ik}^d}, \tag{8.6}$$

where the weighting function $w$ is given by

$$w(a, b) = \begin{cases} 0 & \text{if } a < b \\ (a - b)K_w & \text{if } 0 \leq a - b \leq 1 \\ K_w & \text{if } a - b > 1 \end{cases} \tag{8.7}$$

Here $K_w$ is a weighting constant that should be relative to the elasticity of the membrane material. In figure 8.7 the influenced particles are blue or yellow. The blue vertices are further away from the destination landmark than they were

from the source landmark originally and thus attraction forces are calculated from (8.6) and (8.7). The black particle is closer to the landmark than it originally was (compare lengths A and B), and no force is calculated. Thus landmark matching relies purely on attraction and not on repulsion.

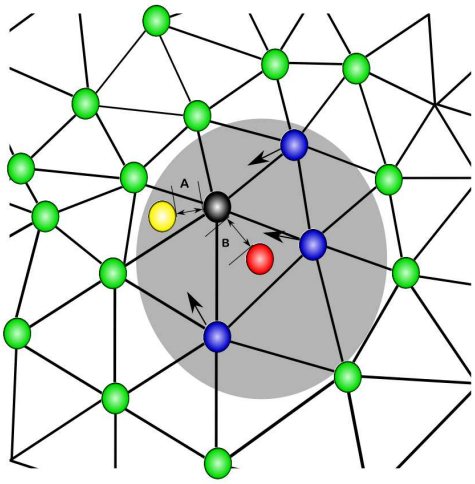


Figure 8.7: Illustration of forces responsible for matching corresponding landmarks from the source and reference surfaces. See text for an explanation.

### 8.5.3 Optimisation procedure

From (8.5) we compute a direction to displace vertex $i$ in physical $(x', y', z')$ space using a step size $\delta$. To relate this displacement $\mathbf{t}_i$ $(= \delta \mathbf{f}_i)$ to displacement $\mathbf{s}_i$ in $(u, v)$ space, it is projected onto the derivative vectors $\mathbf{d}_u$ and $\mathbf{d}_v$:

$$\mathbf{s}_i = \begin{pmatrix} \frac{\mathbf{t}_i \cdot \mathbf{d}_u(u_i, v_i)}{||\mathbf{d}_u(u_i, v_i)||^2} \\ \frac{\mathbf{t}_i \cdot \mathbf{d}_v(u_i, v_i)}{||\mathbf{d}_v(u_i, v_i)||^2} \end{pmatrix} \tag{8.8}$$

The denominator found in the components of $\mathbf{s}_i$ penalises large derivative vectors thus compensating for the scale distortion of the geometry image. The vector $\mathbf{s}_i$ is used to displace the $(u_i, v_i)$ coordinate. An overview of the optimisation procedure can be found in pseudo code in algorithm 4.

## 8.6 Interpolating surface registration to interior

As we would like to be able to transfer deformation of the registered organ to a volumetric vector field, we must decide how to estimate deformation of the interior based on surface displacements. To define a transformation of organ interior we follow the approach presented in [89]. Here *radial basis functions* (RBFs) are used for interpolating the computed displacements of the source

**Algorithm 4**: Overview of the optimisation of the mapping of vertex positions from source surface mesh to reference surface

**input** : Triangle mesh of source surface.
Triangle mesh of reference surface.
Step size $\delta$, landmark matching weight $K_w$, landmark influence radius $r_l$ and spring constants $k_{ij}$
**output**: The set of optimised $(u, v)$ coordinates for vertices $i = 1 \dots M$
Compute geometry image $\mathbf{p}(u, v)$ of the reference surface (section 8.4)
Compute initial $(u, v)$ coordinates for all source vertices (section 8.4)
generate neighbourhoods of all source mesh vertices
**repeat**
  **for** *all source vertices* $i = 1 \dots M$ **do**
    Evaluate spring force $\mathbf{f}_i$ using (8.5)
    Compute position displacement $\mathbf{t}_i = \delta \mathbf{f}_i$
    Project $\mathbf{t}_i$ onto position derivatives using (8.8) to get $\mathbf{s}_i$
    Displace the $(u_i, v_i)$ coordinates by adding $\mathbf{s}_i$
**until** *Potential energy E (8.3) and landmark position errors have converged* ;
**return** $(u, v)$ *coordinates*

organ surface to the interior. In the following this technique is described both for general interpolation and in particular for interpolating displacements. As our triangular meshes also have description of the interior based as a tetrahedral mesh, we have a choice of evaluating the interpolation function at vertices in the tetrahedral mesh or at all positions in a uniform grid.

### 8.6.1 Interpolation by radial basis functions

Assume that the value of a scalar valued function $F : \mathbb{R}^3 \to \mathbb{R}$ is known in $M$ distinct discrete points $\mathbf{x}_i$ in three dimensional space. Then RBFs provide a means for creating a smooth interpolation function of $F$ in the whole domain of $\mathbb{R}^3$. This function is written as a sum of $M$ evaluations of a radial basis function $g(r_i) : \mathbb{R} \to \mathbb{R}$. Here $r_i$ is the distance between the point $\mathbf{x} = (x, y, z)$ to be evaluated and $\mathbf{x}_i$:

$$F(\mathbf{x}) = \sum_{i=1}^{M} a_i g(||\mathbf{x} - \mathbf{x}_i||) + c_0 + c_1 x + c_2 y + c_3 z, \quad \mathbf{x} = (x, y, z) \tag{8.9}$$

where $a_i$ are scalar coefficients. The last four terms constitute a first degree polynomial with coefficients $c_0$ to $c_3$. This describes an affine transformation which cannot be realised by the radial basis functions alone. From the $M$ known function values $F(x_i, y_i, z_i) = F_i$ we can assemble a system of $M + 4$ linear equations:

$$\mathbf{GA} = \mathbf{F} \tag{8.10}$$

where $\mathbf{F} = (F_1, F_2, \ldots, F_M, 0, 0, 0, 0)$, $\mathbf{A} = (a_1, a_2, \ldots, a_M, c_0, c_1, c_2, c_3)$ and $\mathbf{G}$ is an $(M+4) \times (M+4)$ matrix :

$$\mathbf{G} = \begin{bmatrix} g_{11} & g_{12} & \bullet & \bullet & \bullet & g_{1M} & 1 & x_1 & y_1 & z_1 \\ g_{21} & g_{22} & \bullet & \bullet & \bullet & g_{2M} & 1 & x_2 & y_2 & z_2 \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ g_{M1} & g_{M2} & \bullet & \bullet & \bullet & g_{MM} & 1 & x_M & y_M & z_M \\ 1 & 1 & \bullet & \bullet & \bullet & 1 & 0 & 0 & 0 & 0 \\ x_1 & x_2 & \bullet & \bullet & \bullet & x_M & 0 & 0 & 0 & 0 \\ y_1 & y_2 & \bullet & \bullet & \bullet & y_M & 0 & 0 & 0 & 0 \\ z_1 & z_2 & \bullet & \bullet & \bullet & z_M & 0 & 0 & 0 & 0 \end{bmatrix} \tag{8.11}$$

Here $g_{ij} = g(||\mathbf{x}_i - \mathbf{x}_j||)$. A number of choices for $g$ will result in a unique solution of the system [2]. We use the *shifted log function*:

$$g(t) = \sqrt{log(t^2 + k^2)}, \quad k^2 \geq 1 \tag{8.12}$$

with $k = 1$. Solving (8.10) for $\mathbf{A}$ gives us the coefficients to use in (8.9) when interpolating between known values.

### 8.6.2 Interpolating displacements

The RBF interpolation method can be used for interpolating displacements to the interior of the registered organ. Let the initial positions of surface vertices be denoted $\mathbf{x}_i = (x_i, y_i, z_i)$ and the displacements of these vertices as computed in the registration by $\mathbf{u}_i = (u_i^x, u_i^y, u_i^z)$. Then three systems are set up

$$\mathbf{G}\mathbf{A}_x = (u_1^x, u_2^x, \ldots, u_M^x, 0, 0, 0, 0)^T \tag{8.13}$$
$$\mathbf{G}\mathbf{A}_y = (u_1^y, u_2^y, \ldots, u_M^y, 0, 0, 0, 0)^T \tag{8.14}$$
$$\mathbf{G}\mathbf{A}_z = (u_1^z, u_2^z, \ldots, u_M^z, 0, 0, 0, 0)^T \tag{8.15}$$

where $\mathbf{G}$ is found from (8.11). Solving for $\mathbf{A}_x$, $\mathbf{A}_y$, and $\mathbf{A}_z$ involves a single matrix inversion and three matrix-vector multiplications and gives us the coefficients for interpolating displacements in all three directions by the expression (8.9) {7}.

## 8.7 Evaluation

In the following we present the initial evaluation of the proposed surface registration method. The results of four registration experiments are reported:

1. Registration of a modelling wax cylinder from straight configuration **(a)** to a 45 degree bend **(b)**.

2. Registration of a modelling wax cylinder from straight configuration **(a)** to a 90 degree bend **(c)**.

3. Registration of a repeated delineation of a bladder showing a large change in volume

4. Registration of a repeated delineation of a bladder doing a bending motion

For all registration experiments the same spring constant of $k = 10$ N/m was used for all springs, and the landmark matching constant was also set to $K_w = k$. A landmark influence radius of $r_l = 2.0$ cm and a step size of $\delta = 2.5$ was used. Unless otherwise stated the 2-step neighbourhood has been used for spring generation. The generated geometry images had a resolution of $400 \times 400$. Convergence of the optimisation was established manually by inspecting a visualisation of the surface registration as well as the potential energy and landmark error distance.

### 8.7.1 Modelling wax cylinder registration

The modelling wax phantom data set used for testing the registration method is the same as presented in section 7.7.1. The triangular meshes being registered consist of the surface triangles of the tetrahedral meshes used there. To guide the registration four landmarks have been placed on the surface of each mesh. The positioning of these points can be seen in figure 8.8. The mesh of configuration **(a)** consists 1434 triangles, configuration **(b)** of 1380 triangles, and configuration **(c)** of 1438 triangles.

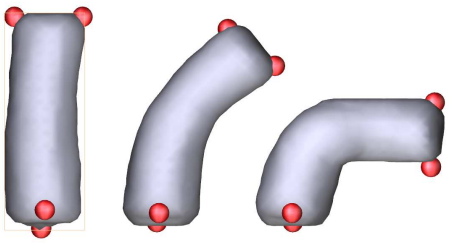

Figure 8.8: Positioning of landmarks used to constrain the surface registration of the modelling wax cylinder. The shown meshes are configurations **(a)**, **(b)**, and **(c)** respectively

#### Registering configuration (a) to (b)

For the registration of cylinder **(a)** to **(b)** the entire registration process (including parameterisation of both objects and geometry map generation) took 1.5 minutes. Figure 8.9 shows the combined landmark matching and spring relaxation process. It can be seen that during optimisation the surface is displaced over the top end of the cylinder. Visual comparison of the optimized surface match and the undeformed source shape confirms that the surface has been shifted into place. Figure 8.10 shows linear interpolation between the source shape and the deformed source both before and after the landmark matching and minimisation of potential energy. It can be seen that before optimisation

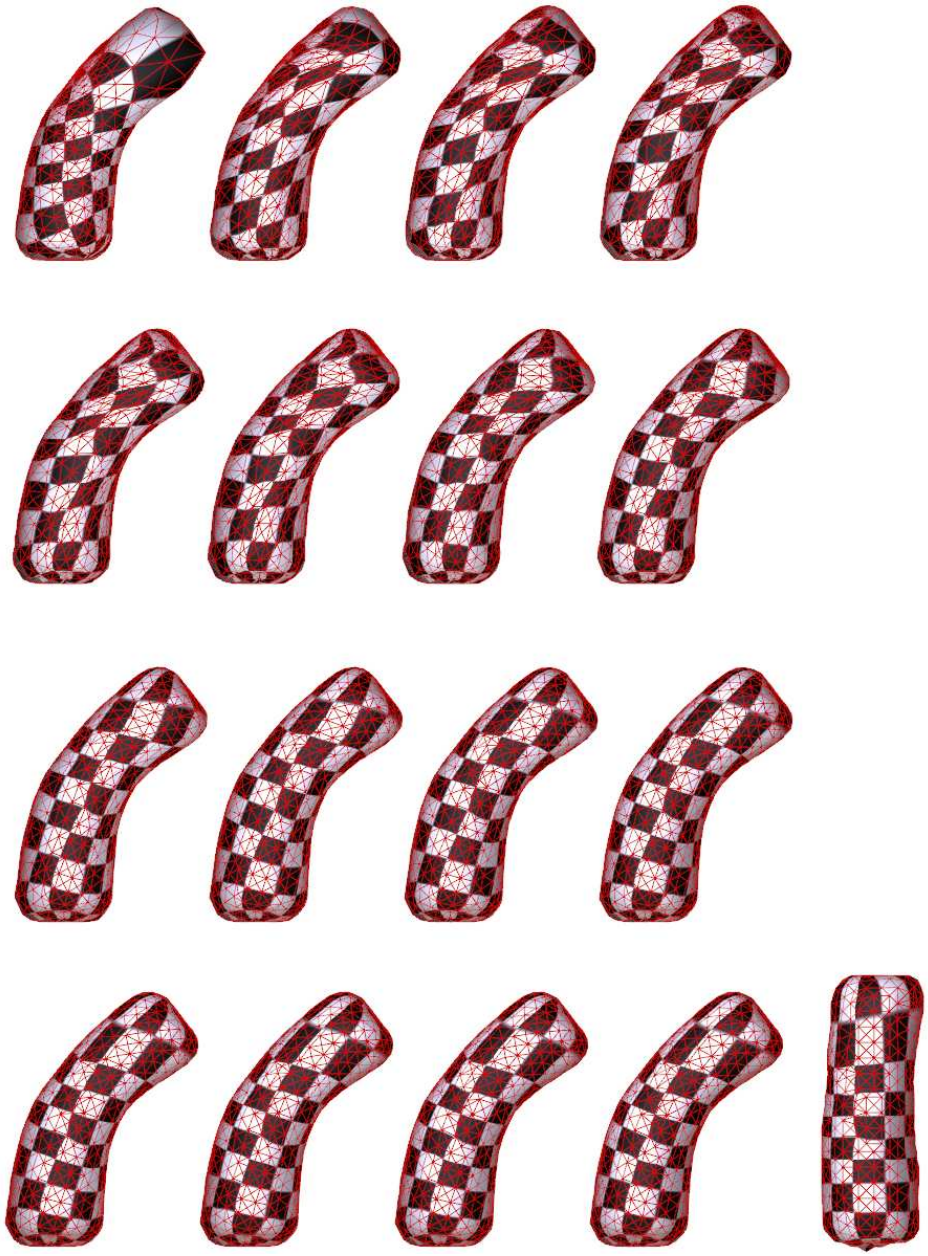

Figure 8.9: Matching landmark constraints and minimising membrane energy during the registration of wax cylinder configuration **(a)** to **(b)**. The surface is visualised for each 200 iterations. The triangle mesh is shown in red and a chequerboard texture is shown to emphasise membrane stretching {9}. The chequerboard layout on the final result must be compared with the layout on the undeformed source cylinder (shown on the right).

the top end grows from the side of the straight cylinder and after optimisation the surface registration correctly describes a bending motion.

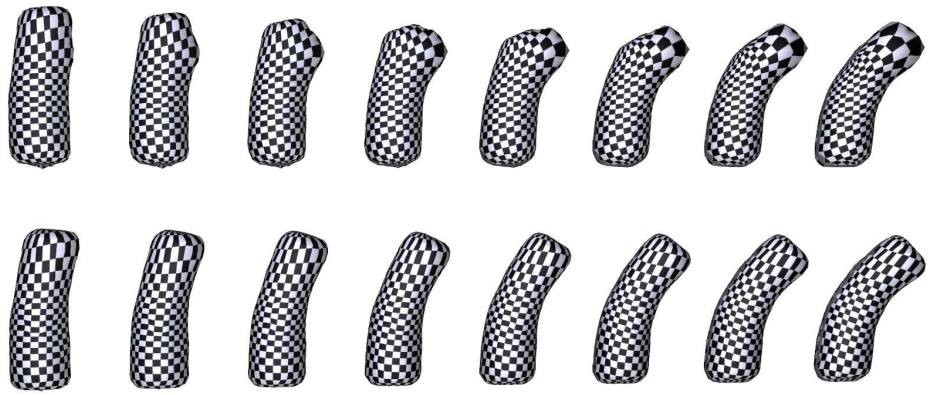

Figure 8.10: Registration from **(a)** to **(b)**. Linear interpolation between the source and deformed source shape (left to right). **Top:** after the initial projection {8}. **Bottom:** after constrained energy minimisation. {10}

#### Registering configuration a to c

Attempts to perform registration of cylinder configuration **(a)** to **(c)** using the 2-step neighbourhood resulted in a flipping of a few of the surface triangles. For this reason a 3-step neighbourhood was used instead. This induced a higher computational burden meaning that the entire registration (2979 iterations) took 20 minutes. Linear interpolation of the undeformed source to the initial projection and optimised projection respectively can be seen in figure 8.11. Again it can be seen that the failure of the initial projection to describe the bending motion is alleviated during registration.

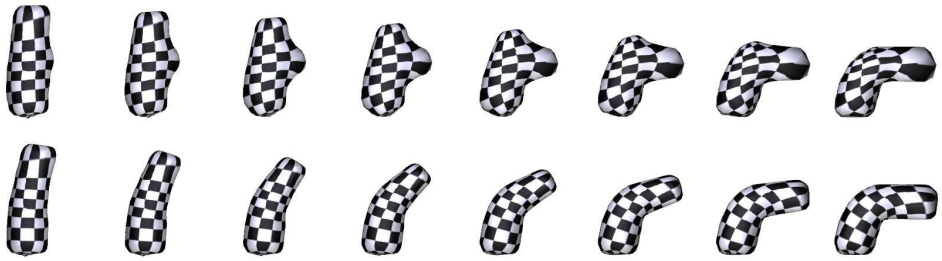

Figure 8.11: Registration from **(a)** to **c)**. Linear interpolation between the source and deformed source shape (left to right). **Top:** after the initial projection. **Bottom:** after constrained energy minimisation.

### 8.7.2 Bladder registrations

As mentioned two cases were used when investigating the use of the proposed surface registration method on bladders. These cases can be seen in figure 8.12.

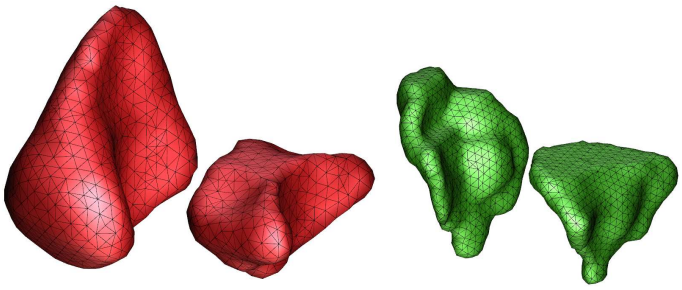

Figure 8.12: The two cases of bladder registration investigated: a case involving a large difference in bladder volume (red) and a case involving a bending motion (green). Black lines show the surface triangulation.

### Volume changing bladder registration

For the bladder showing a change in volume (registering the high-volume bladder to low-volume bladder) the initial projection performed by sphere inflation introduced less distortion than in the wax cylinder registrations. However, as figure 8.13 shows, the surface was stretched in the left and right sides of the bladder. This was relaxed by energy minimisation. Guided by MR images two landmarks were positioned next to the base of the urethra. The low volume bladder mesh consists of 1034 triangles while the high volume bladder mesh consists of 2002 triangles. Figure 8.14 shows the interpolation between the undeformed source and the optimised registration. The registration took approximately 3 min.

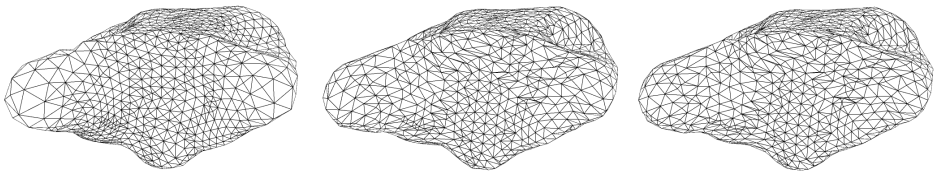

Figure 8.13: Three steps in the minimisation of potential membrane energy after initial surface projection in the high-to-low-volume bladder registration.

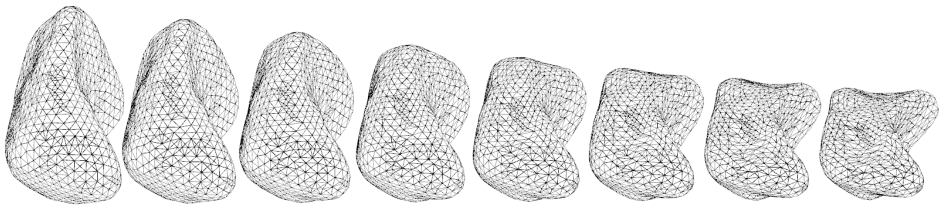

Figure 8.14: Interpolation between high and low volume configurations in the high-to-low-volume bladder registration. Seen from the side compared to figure 8.13

### Bending bladder registration

For the bladder case involving a bending motion we again used 2 landmarks placed near the base of the urethra. The entire registration took three and a half minutes (3000 optimisation iterations). The source mesh consisted of 2480 triangles. From figure 8.15 the difference can be seen in surface projection before and after the optimisation process. In the initial projection (top row) the base of the bladder is registered to another part of the bladder, while this is corrected after optimisation.

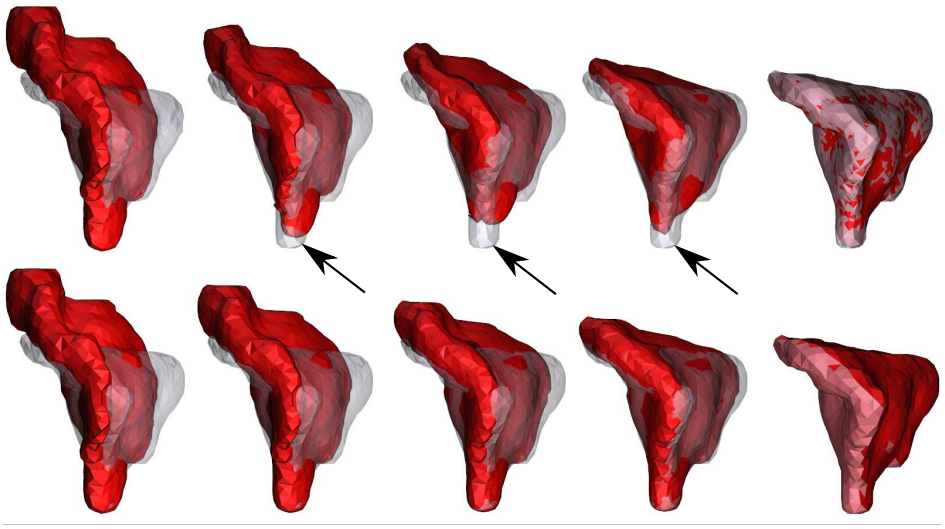

Figure 8.15: **Top:** Linear interpolation between source and deformed source after initial projection {11}. **Bottom:** Linear interpolation between source and deformed source after constrained membrane energy minimisation.{12}{13}

## 8.8 Conclusion and discussion

In this chapter a method was presented for optimising the surface to surface mapping of an organ that has been delineated in images from repeated 3D acquisition. Specifically the intended use for the method is registration of bladders. The key concept is the use of geometry images of the reference surface for restricting source vertices to stay on the surface while the elastic potential energy of a membrane model is minimised. Non-trivial boundary conditions ensure continuity even as the $(u, v)$ coordinates of a vertex is displaced to be outside the geometry image domain.

Using spring-mass systems it is hard to accurately simulate the behaviour of a real-life elastic material. The modelled behaviour is highly dependent on the topology of the mesh used, and in particular the determination of suitable spring constants is non-trivial. However, the registration results shown in the previous section seem physically plausible. Problem caused by triangle flipping might be avoided by introducing an triangle area based penalty as in [41]. As

mentioned the use of geometry images in the way described here is not restricted to spring-mass systems but is also applicable for e.g. finite element analysis of membrane behaviour, as long as the minimisation of potential energy is based on iterative displacement of vertex positions.

For interpolating the transformation of the surface to the organ interior we proposed using radial basis functions. This choice of method is based on the use case of bladder registration where we are not too concerned with the transformation of the interior but are mainly concerned with achieving a smooth transformation. However, once a surface to surface mapping has been obtained, this mapping could constitute the boundary condition for a volumetric finite element analysis of interior displacements as proposed in multiple references in section 7.1.1.

A note is in order about the projection of forces onto position gradients from the geometry image. This technique puts an upper limit on the size of surface triangles relative to the curvature of the described reference shape. As illustrated in figure 8.16 (**a** and **b**) too large distances between vertices may cause projected spring forces to work in a wrong direction. An evaluation of the effect on potential energy of moving along the projected force vectors may help us detect these cases. The same problems exist for the landmark matching

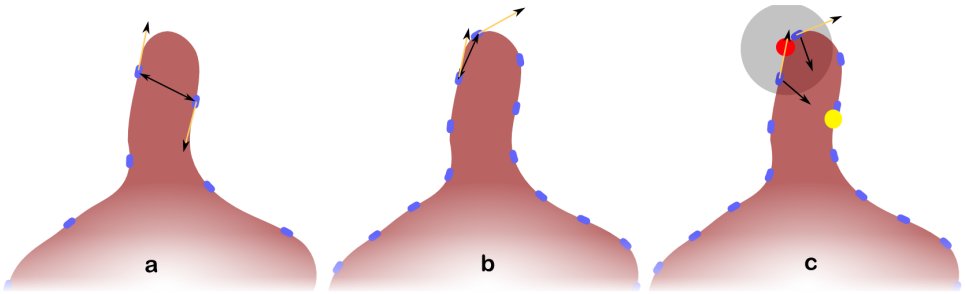


Figure 8.16: **a** : Too low mesh resolution compared to the curvature of the surface. The blue dots represent surface vertices in a 2D analogy. Black arrows are forces to be projected onto surface tangent vectors (orange arrows). The direction of force between two vertices is shown. For the leftmost of the two vertices a projection of this force onto the tangent vector will work in the opposite direction than intended. **b** : Here this problem is avoided because of a higher mesh resolution. **c**: The problem exists for landmark matching too. Two vertices influenced by the red source landmark are attracted towards the corresponding yellow reference landmark. When projecting the resulting forces onto the surface tangent they will work in a counterproductive direction.

with the proposed method. If the source and reference landmarks are too far away after the initial projection we risk facing the problem shown in figure 8.16 (**c**), where the projection of landmark matching forces work in the wrong direction. For this reason it is desirable that the landmarks are approximately matched already during the projection - e.g. by finding a best-fit rotation of inflated spheres possibly followed by interpolation in spherical 2D space.

The surface matching method presented in this chapter has been designed to be straightforward to parallelise in a SIMD fashion. All computations in the initial surface projection and subsequent optimisation are performed in an iterative, independent fashion. Thus to us it will be an obvious continuation of our work to accelerate the method using GPU-based computation. Earlier work shows that there is great potential for acceleration of spring-mass computations using GPUs [96] [116], and as demonstrated in the previous chapter more physically accurate models based on FEA can also be accelerated on this platform. The most important future direction however is evaluation of the method on a larger collection of bladder delineations to validate the accuracy and robustness of the approach.

# Part IV

# Conclusion and future work

# Chapter 9

## Future work

In the preceding chapters a number of suggestions have been made about future improvements of the presented methods. In this section some of these suggestions will be given a somewhat more elaborate discussion. Furthermore other possible future directions will be mentioned - specifically a joint framework for intensity and mesh based registration.

### 9.1 Combining mesh based and intensity based registration

As exemplified in figure 5.7 many existing intensity based registration methods relying on local gradients of a similarity measure have difficulties in accurately describing significant rotation and bending of entities (e.g. organs) between two images. A common problem in this case is to ensure that the resulting registration is physically plausible, i.e. that the registration describes the actual bending/rotation occurring rather than just introducing expansion in some areas and shrinkage in others.

During the work presented in chapters 5–8 our scope has been directed towards the future development of a general framework for deformable image registration of two 3D images. The goal of this framework is to achieve the following:

1. to take advantage of as much prior information as possible, i.e. when registering images in which organs have already been segmented, we wish to take this information into account. Also we wish to include information from auto segmentation if possible (of e.g. bone structures in CT images). Finally we wish to support using information about the physical properties of our segmented volumes - e.g. that bones are rigid and that the bladder is likely to change its volume.

2. to combine registration based on segmented shapes and registration of image intensities outside shapes.

3. to support structural connections between shapes - e.g. that vertebra in the neck are connected by discs, or that the rectum is connected to the sigmoid colon.

In the work implemented so far we have focused on fast deformable registration of meshes that remedies some of the disadvantages of intensity based methods. We would like to use the result of these methods to guide intensity based registration. Originally our idea was to register all shapes and intensities simultaneously via exchange of forces between an elastic biomechanical model and physically motivated intensity based image registration like the viscous-fluid model. This idea was inspired by the inclusion of regions of interest in the body force term in chapter 5 (proposed by Christensen et al. in [28]). However, for the time being we have narrowed our focus somewhat and plan to first register tetrahedral meshes of segmented organs and subsequently use the resulting deformation maps as prior information in intensity based registration. In this chapter we present our thoughts and ideas on unifying the registration methods presented in this dissertation.

### 9.1.1 Frames of reference

One important problem in combining intensity based registration with mesh based registration is the difference in reference frames. Intensity based registration typically uses an Eulerian frame of reference in which displacements are tracked with respect to voxel positions that are *fixed in the reference image*. In contrast model based registration relies on a Lagrangian frame of reference where the displacements are tracked with respect to initial vertex positions that are *fixed in the source image*.

To efficiently use displacements from a mesh-based model in an intensity based setting we need to create a regular grid of displacements corresponding to the voxels of the intensity based model. We propose to produce such a displacement grid by rasterising displacements from the mesh model by interpolating vertex displacements inside tetrahedra. There are two possible approaches for this:

1. For each tetrahedron we write a displacement vector to all grid positions that are inside the tetrahedron in its original (undeformed) configuration. The vectors written are found by interpolation of vertex displacements. For this to be consistent with the Eulerian frame of reference of the displacement grid, the model based registration must be performed in the "opposite" direction of the intensity based registration - i.e. the mesh must be generated from a segmentation of the reference image and registered to the configuration of the source image.

2. For each tetrahedron we write an interpolated displacement vector to grid positions inside the deformed tetrahedron. The displacement vectors written must be interpolated between vectors pointing from the displaced vertex positions to the original ones.

An advantage of the former approach is that the position to write in the displacement vector grid for a point inside the mesh does not change during registration. Then for grid positions in the displacement field we can store information about which tetrahedron covers the position along with the weights needed

to interpolate between displacements of corners in the tetrahedron allowing us to efficiently look up displacements from the mesh. Taking the latter approach we cannot register intensities and mesh models simultaneously but must first finish the model based registration and subsequently use this information in intensity based registration.

### 9.1.2 Combining optical flow estimation with model-based registration

To combine optical flow estimation with model-based registration we plan to use displacements from pre-registered meshes as spatial boundary conditions for the Jacobi iterations used to solve for the optimal displacement field. The idea is to use the normal Jacobi iteration kernel for voxel positions outside organ binary masks and write displacements from biomechanical registration for voxel positions inside organs. To ensure numerical stability it will probably be necessary to ensure that we avoid large initial discontinuities in the displacement field (in the first Jacobi iteration). There are multiple ways of achieving this:

- By incremental increase of the displacements from organ registrations. This can be done by linearly interpolating between zero vectors and the computed displacements.
- By creating a smooth displacement field covering the entire image and containing the precomputed displacements. This can e.g. be done by extrapolating using radial basis functions. This displacement field would then serve as the initial values in the optimisation. To avoid introducing large misregistrations in this initial guess, it would probably be beneficial to include rigid registration of bones in the RBF extrapolation of the initial field.

An alternative to using radial basis functions in the latter option is the elastic deformation approach using prescribed displacements proposed by Peckar et al. [107]. Here point displacements are used as hard constraints, and the method computes a unique displacement field satisfying the constraints and deforming the remaining image based on linear elasticity. Because an incremental method is used, large deformations can also be handled.

Instead of using point correspondences Postelnicu et al. extrapolate a triangle mesh surface registration to the entire 3D image domain [111]. This is done based on an FEA and linear elasticity. Subsequently the registration is refined for the 3D image domain using intensity based optical flow (similarly to what we propose). A similar approach has been taken by Joshi et al. who also extended a brain surface registration to the entire image volume and refined using an inverse consistent linear elastic intensity driven registration [60].

### 9.1.3 Combining viscous-fluid registration with model-based registration

In the viscous-fluid registration method the registration is driven by physically motivated force terms. Hence we can compute an attraction force driving each

voxel inside binary masks towards the "target" positions known from the pre-computed displacement field. This technique is somewhat similar to the one used by Christensen et al. [28], but by pre-registering we can hopefully avoid "pinching problems" like the one depicted in figure 5.7 (page 51).

### 9.1.4 About the need for segmentation in both images

A drawback with the methods presented in chapters 7 and 8 is the need for organ segmentation in both source and reference images. With the membrane model it seems there is no way around this as the segmentation of the reference image is needed for creating the geometry images, which is an essential part of the method. However, the volumetric biomechanical model of chapter 7 could potentially be guided by other means than distance fields. For moderate deformation mutual information might work either using the local force approach of Crum et al. [34] or using the featurelet approach of Söhn et al. [127]. Here a choice can be made about what to compare to the reference image: a binary mask of the current organ configuration or a deformed source image. Also normalised gradient field might serve as a matching component with both these choices. Using the deformed source approach we might even use NCC or SSD as similarity measures.

Driving the elastic model using one or more of these intensity based measures will essentially turn the registration into an intensity based method with mesh-based regularisation. In this context the difference in reference frames described above is actually an advantage as the organ segmentation needs to be done in the *reference image* of the purely intensity based method, which for our use will most often be the first (planning) image, where segmentation is already made. If such a registration is sufficiently accurate it will serve as an auto segmentation of the new image.

## 9.2 Multi-organ registration

As mentioned above we would like to support structural connections between organs in a biomechanical multi-organ registration setting. We hypothesise that the extra prior information included in this approach will allow more physically plausible results. Brock et al. use a multi-organ approach in which organ connections are modelled by the use of "tied connections" in which two organs share common vertices [21]. A similar approach could be taken in our method. As mentioned in section 7.8 we would also like to include a general intersection handling method in the biomechanical registration, to handle both the problem of self-intersection and to allow organs to interact in a physically plausible way. We believe this will increase the robustness of the registration and may open new applications.

## 9.3 Volumetric mesh registration using geometry images

The approach of using geometry images for constraining vertices to a reference surface can also be used in other models than the membrane registration of chapter 8. We have already mentioned the possibility of using the result of membrane based registration as fixed boundary conditions for a subsequent volumetric elastic FEA. Another approach would be to apply the geometry image as sliding boundary conditions during the volumetric FEA thus combining the appproaches of chapters 7 and 8. The idea is to first create a mapping between the source and reference configurations that when interpolated to the interior of the organ is smooth and regular enough that no tetrahedron in our FEM is inverted. Then, using the force projection approach presented in section 8.5.3, the interior energy of the volumetric FEM is minimised while restricting boundary vertices to stay in the parameterised $(u, v)$-space of the reference geometry image.

## 9.4 Principal component analysis

A potential use for the two mesh based organ registration methods is *principal component analysis* (PCA), which has been proposed for creating patient-individual organ motion/deformation models for use in radiotherapy [128]. Using PCA the motion and deformation of organs can be statistically modelled as a linear combination of so-called eigenmodes, which are the most dominant components of the motion/deformation. Weighting different modes differently it is possible to interpolate and even extrapolate new organ geometries from a number of initial organ configurations. This can be used to analyse how large margins on the radiated field are needed in an EBRT treatment. The PCA approach requires information about surface-to-surface point correspondence, which can be found with the proposed methods.

## 9.5 Clinical validation

The evaluation of methods presented in this dissertation has been on a "proof-of-concept" level. As mentioned several times in the preceding chapters, a clinical validation is needed for determining the accuracy and robustness of the methods.

# Chapter 10

## Summary and conclusions

Deformable image registration techniques able to account for displacement and deformation of organs in a series of medical images acquired in connection with fractions of radiotherapy is a key component in the efforts to improve the treatment guided by image data. In this dissertation five registration methods have been examined as a potential tool in image guided radiotherapy. Three of these methods were already known intensity based registration methods. These have been implemented in a massively parallel setting using GPU computations. As a result computation times were reduced considerably.

The first method was the viscous-fluid registration method, which was made suitable for SIMD parallelisation by using Jacobi iteration for solving the viscous-fluid PDE in each time step. As a result of GPU based acceleration, the running time was reduced to a time frame that makes it feasible to initiate a clinical evaluation of the method. Our initial experience with the method tells us that the high degree of flexibility in the resulting spatial transformations makes it possible to perform registrations involving both large deformations and locally detailed deformation. However, its ability to register such cases also makes it difficult to evaluate the results without an inspection of the deformation field. An example of a misregistration that might go unnoticed was shown on figure 5.5 (page 49).

The remaining two GPU accelerated intensity based methods were optical flow estimation methods that had been extended from 2D to 3D. Use of the Horn and Schunck method was presented for CT-to-CT registration, while the Cornelius and Kanade method was used for registration of CT-to-CBCT as well as CBCT-to-CBCT. The initial evaluation was encouraging, but an extensive clinical validation is needed to fully determine the applicability of the methods. The running time of the two optical flow estimation methods was reduced to a level that makes it feasible to register a newly acquired head and neck CBCT image to a planning CT image while the patient is still on the treatment couch.

The fourth registration method was a new method based on a biomechanical model of an elastic solid and assumed that a segmentation of an organ had been made in both source and reference image. A non-linear elastic finite element model was used to model interior forces as a tetrahedral mesh representation of the organ in the source image was deformed toward the configuration of the organ in the reference image. This deformation was driven by forces based on a

distance field representation of the reference surface and forces working normal to the source shape. An advantage with this method is that finding surface correspondences is coupled with modelling of the interior elastic model, which means that unlike some previous methods, we do not need to pre-compute boundary conditions before FEA. All parts of this registration method had been formulated in a way that made a SIMT formulation possible, paving the way for a GPU implementation. A rigid model was included to allow rigid alignment to be made prior to deformable modelling. An initial test of the model was made. From the results we concluded that registration of solid, volume preserving organs seems generally feasible. Further validation studies are needed to validate this claim. Changes in the method must be made for it to be able to handle organs that empty and fill.

Finally a novel method for deformable surface registration was presented. The deforming source surface was modelled as an elastic membrane. Using a geometry image to represent a two-dimensional parameterisation of the reference surface, the source surface vertices were constrained to stay on the reference surface. Subject to this constraint the potential energy in the membrane was minimised using iterative displacement of source vertices. The displacements were calculated in 3D space based on spring force directions and projected onto the tangent plane of the parameterised surface, thus resulting in a displacement in the parameterised 2D space. A spring-mass system was used for modelling the membrane. Furthermore the presented model allowed inclusion of user specified landmark correspondences which was matched using a driving force approach. The intended use of the membrane model is for bladder registration.

Our focus on registration algorithms was motivated by two clinical cases. One was registration of MR images acquired in connection with intracavitary brachytherapy for treatment of cervical cancer and the other was registration of daily head and neck CBCT images to the planning CT image. The performance of the Cornelius and Kanade method on head and neck CBCT images was encouraging. Future validation will show if the method is robust enough to enable e.g. auto-segmentation by contour propagation. Initial testing has revealed that none of the three intensity based methods considered were able to consistently register all of the pelvis MR images tested in the BT case. This has been the starting point for the work on mesh based organ registration, and a future aim is to combine these organ registrations with an intensity based method outside organs. We presented methods suitable for registering the uterus and the bladder, but our methods need modifications to be able to consistently handle registration of the rectum and the sigmoid colon.

As elaborated on in the previous chapter, there are several possible future directions that will enable the presented methods to be combined. At present however, the methods work individually for a limited set of test problems. A next step is clinical validation of the general robustness and accuracy of the methods for a larger set of registration problems.

# Bibliography


[1] S. Angenent, S. Haker, A. Tannenbaum, and R. Kikinis. On the Laplace-Beltrami operator and brain surface flattening. *IEEE Transactions on Medical Imaging*, 18(8):700–711, Aug. 1999.

[2] N. Arad, N. Dyn, D. Reisfeld, and Y. Yeshurun. Image warping by radial basis functions: applications to facial expressions. *CVGIP: Graph. Models Image Process.*, 56(2):161–172, 1994.

[3] ATI Researcher Relations. ATI CTM Guide. Technical report, ATI, `http://ati.amd.com/companyinfo/researcher/documents.html`, 2006.

[4] R. Bajcsy and S. Kovacic. Multiresolution Elastic Matching. *Computer Vision, Graphics, and Image Processing*, 46:1–21, 1988.

[5] G. Barequet and M. Sharir. Piecewise-Linear Interpolation between Polygonal Slices. *Computer Vision and Image Understanding*, 63(2):251 – 272, 1996.

[6] A. Barker and J. Reffstrup. *The finite element method for partial differential equations.* Technical University of Denmark, 1998.

[7] J. L. Barker, A. S. Garden, K. K. Ang, J. C. O'Daniel, H. Wang, L. E. Court, W. H. Morrison, D. I. Rosenthal, K. S. C. Chao, S. L. Tucker, R. Mohan, and L. Dong. Quantification of volumetric and geometric changes occurring during fractionated radiotherapy for head-and-neck cancer using an integrated CT/linear accelerator system. *International Journal of Radiation Oncology*Biology*Physics*, 59(4):960 – 970, 2004.

[8] J. L. Barron, D. J. Fleet, S. S. Beauchemin, and T. A. Burkitt. Performance Of Optical Flow Techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[9] K.-J. Bathe. *Finite Element Procedures.* Prentice-Hall, 1996.

[10] S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM Comput. Surv.*, 27(3):433–466, 1995.

[11] A. Bharatha, M. Hirose, N. Hata, S. K. Warfield, M. Ferrant, K. H. Zou, E. Suarez-Santana, J. Ruiz-Alzola, A. D'Amico, R. A. Cormack, R. Kikinis, F. A. Jolesz, and C. M. C. Tempany. Evaluation of three-dimensional

finite element-based deformable registration of pre- and intraoperative prostate imaging. *Medical Physics*, 28:2551–2560, Dec. 2001.

[12] M. Birkner, D. Yan, M. Alber, J. Liang, and F. Nüsslin. Adapting inverse planning to patient and organ geometrical variation: algorithm and implementation. *Medical Physics*, 30(10):2822–2831, 2003.

[13] R. Blickhan. The spring-mass model for running and hopping. *J Biomech*, 22(11-12):1217–1227, 1989.

[14] F. L. Bookstein. Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(6):567–585, 1989.

[15] C. Brechbühler, G. Gerig, and O. Kübler. Parametrization of closed surfaces for 3-D shape description. *Comput. Vis. Image Underst.*, 61(2):154–170, 1995.

[16] M. Bro-Nielsen. *Medical Image Registration and Surgery Simulation*. PhD thesis, 1997.

[17] M. Bro-Nielsen and S. Cotin. Real-time Volumetric Deformable Models for Surgery Simulation using Finite Elements and Condensation. In *Computer Graphics Forum*, volume 5, pages 57–66, 1996.

[18] M. Bro-Nielsen and C. Gramkow. Fast Fluid Registration of Medical Images. *Lecture Notes In Computer Science, Spinger*, vol 1131:Visualization in Biomedical Computing:265–276, 1996.

[19] K. K. Brock, L. A. Dawson, M. B. Sharpe, D. J. Moseley, and D. A. Jaffray. Feasibility of a novel deformable image registration technique to facilitate classification, targeting, and monitoring of tumor and normal tissue. *Int J Radiat Oncol Biol Phys*, 64(4):1245–1254, Mar 2006.

[20] K. K. Brock, D. L. McShan, R. K. T. Haken, S. J. Hollister, L. A. Dawson, and J. M. Balter. Inclusion of organ deformation in dose calculations. *Medical Physics*, 30(3):290–295, 2003.

[21] K. K. Brock, M. B. Sharpe, L. A. Dawson, S. M. Kim, and D. A. Jaffray. Accuracy of finite element model-based multi-organ deformable image registration. *Medical Physics*, 32(6):1647–1659, 2005.

[22] L. G. Brown. A Survey of Image Registration Techniques. *ACM Computing Surveys*, 24, December 1992.

[23] I. Buck, K. Fatahalian, and P. Hanrahan. GPUBench: Evaluating GPU performance for numerical and scientific application. In *ACM Workshop General Purpose Computing Graphics Processors, Los Angeles,CA*, pages C–20, 2004.

[24] I. Buck, T. Foley, D. Horn, J. Sugerman, K. Fatahalian, M. Houston, and P. Hanrahan. Brook for GPUs: stream computing on graphics hardware. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 777–786, New York, NY, USA, 2004. ACM.

[25] J. Choi and A. Szymczak. Fitting solid meshes to animated surfaces using linear elasticity. *ACM Trans. Graph.*, 28(1):1–10, 2009.

[26] G. Christensen and H. Johnson. Consistent image registration. *IEEE Transactions on Medical Imaging*, 20(7):568–582, July 2001.

[27] G. E. Christensen. *Deformable Shape Models for Anatomy.* Doctoral thesis, Sener Institute, Washington University, 1994.

[28] G. E. Christensen, B. Carlson, K. S. C. Chao, P. Yin, P. W. Grigsby, K. Nguyen, J. F. Dempsey, F. A. Lerma, K. T. Bae, M. W. Vannier, and J. F. Williamson. Image-based dose planning of intracavitary brachytherapy: registration of serial imaging studies using deformable anatomic templates. *Int. J. Radiation Oncology Biol. Phys*, Vol. 51:1:227–243, 2001.

[29] G. E. Christensen, R. D. Rabbitt, and M. I. Miller. Deformable Templates Using Large Deformation Kinematics. *IEEE Transactions on Image Processing*, 5:1435–1447, 1996.

[30] G. E. Christensen, P. Yin, M. W. Vannier, K. S. C. Chao, J. F. Dempsey, and J. F. Williamson. Large-Deformation Image Registration Using Fluid Landmarks. In *SSIAI '00: Proceedings of the 4th IEEE Southwest Symposium on Image Analysis and Interpretation*, page 269, Washington, DC, USA, 2000. IEEE Computer Society.

[31] U. Clarenz, M. Droske, and M. Rumpf. Towards Fast Non-Rigid Registration. In *Inverse Problems, Image Analysis and Medical Imaging, AMS Special Session Interaction of Inverse Problems and Image Analysis*, pages 67–84. AMS, 2002.

[32] N. Cornelius and T. Kanade. Adapting optical-flow to measure object motion in reflectance and X-ray image sequences. In *Proc. of the ACM SIGGRAPH/SIGART interdisciplinary workshop on Motion: representation and perception*, pages 145–153, New York, NY, USA, 1986. Elsevier North-Holland, Inc.

[33] N. Courty and P. Hellier. Accelerating 3D Non-Rigid Registration using Graphics Hardware. In *International Journal of Image and Graphics*, volume 8, pages 81–98, 2008.

[34] W. R. Crum, D. L. Hill, and D. J. Hawkes. Information theoretic similarity measures in non-rigid registration. *Inf. Process. Med. Imaging.*, 18:378–387, 2003.

[35] W. R. Crum, C. Tanner, and D. J. Hawkes. Anisotropic multi-scale fluid registration: evaluation in magnetic resonance breast imaging. *Phys Med Biol*, 50(21):5153–5174, Nov 2005.

[36] P. Danielsson. Euclidean Distance Mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[37] B. de Senneville, K. Noe, M. Ries, M. Pedersen, C. Moonen, and T. Sorensen. An optimised multi-baseline approach for on-line MR-temperature monitoring on commodity graphics hardware. *5th IEEE International Symposium on Biomedical Imaging (ISBI): From Nano to Macro*, pages 1513–1516, 2008.

[38] G. X. Ding, D. M. Duggan, C. W. Coffey, M. Deeley, D. E. Hallahan, A. Cmelak, and A. Malcolm. A study on adaptive IMRT treatment planning using kV cone-beam CT. *Radiotherapy and Oncology*, 85(1):116 – 125, 2007.

[39] K. Erleben and H. Dohlmann. Scan Conversion of Signed Distance Fields. In *Proceedings fra den 15. Danske Konference i Mønstergenkendelse og Billedanalyse*, pages 81–91, 2006.

[40] M. Ferrant, S. K. Warfield, A. Nabavi, F. A. Jolesz, and R. Kikinis. Registration of 3D intraoperative MR images of the brain using a finite element biomechanical model. In *IEEE Transactions on Medical Imaging*, pages 1384–1397. Springer-Verlag, 2001.

[41] B. Fischl, M. I. Sereno, and A. M. Dale. Cortical Surface-Based Analysis: II: Inflation, Flattening, and a Surface-Based Coordinate System. *NeuroImage*, 9(2):195 – 207, 1999.

[42] B. Fischl, M. I. Sereno, R. B. H. Tootell, and A. M. Dale. High-Resolution Intersubject Averaging and a Coordinate System for the Cortical Surface. *Hum. Brain Mapp*, 8:272–284, 1999.

[43] S. Fisher and M. C. Lin. Deformed distance fields for simulation of non-penetrating flexible bodies. In *Computer Animation and Simulation*, 2001.

[44] M. Fornefett, K. Rohr, and H. S. Stiehl. Radial Basis Functions with Compact Support for Elastic Registration of Medical Images. *Image and Vision Computing*, 19:87–96, 2001.

[45] M. Grabner, T. Pock, T. Gross, and B. Kainz. Automatic Differentiation for GPU-Accelerated 2D/3D Registration. In C. H. Bischof, H. M. Bücker, P. D. Hovland, U. Naumann, and J. Utke, editors, *Advances in Automatic Differentiation*, pages 259–269. Springer, 2008.

[46] X. Gu, S. J. Gortler, and H. Hoppe. Geometry images. *ACM Trans. Graph.*, 21(3):355–361, 2002.

[47] X. Gu, Y. Wang, T. Chan, P. Thompson, and S.-T. Yau. Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE Transactions on Medical Imaging*, 23(8):949–958, Aug. 2004.

[48] T. Guerrero, G. Zhang, T.-C. Huang, and K.-P. Lin. Intrathoracic tumour motion estimation from CT imaging using the 3D optical flow method. *Physics in Medicine and Biology*, 49(17):4147, 2004.

[49] C. Guetter, C. Xu, F. Sauer, and J. Hornegger. Learning based non-rigid multi-modal image registration using Kullback-Leibler divergence. *Int. Conf. Med Image. Comput. Comput. Assist. Interv.*, 8(Pt 2):255–62, 2005.

[50] E. Haber and J. Modersitzki. Beyond Mutual Information: A simple and robust alternative. In H. Meinzer, H. Handels, A. Horsch, and T. Tolxdorff, editors, *Bildverarbeitung für die Medizin 2005*, pages 350–354. Springer, 2005.

[51] A. Hagemann, K. Rohr, and H. S. Stiehl. Coupling of fluid and elastic models for biomechanical simulations of brain deformations using FEM. *Medical Image Analysis*, 6(4):375 – 388, 2002.

[52] A. Hagemann, K. Rohr, H. S. Stiehl, U. Spetzger, and J. M. Gilsbach. Biomechanical Modeling of the Human Head for Physically Based, Non-rigid Image Registration. *IEEE transactions on medical imaging*, 18:875–882, 1999.

[53] S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro, and M. Halle. Conformal surface parameterization for texture mapping. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):181–189, Apr-Jun 2000.

[54] S. Haker, S. K. Warfield, and C. M. Tempany. Landmark-Guided Surface Matching and Volumetric Warping for Improved Prostate Biopsy Targeting and Guidance. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2004*, pages 853–861, 2004.

[55] P. Hastreiter, C. Rezk-Salama, G. Soza, M. Bauer, G. Greiner, R. Fahlbusch, O. Ganslandt, and C. Nimsky. Strategies for brain shift evaluation. *Medical Image Analysis*, 8(4):447 – 464, 2004.

[56] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–204, 1981.

[57] F. Ino, J. Gomita, Y. Kawasaki, and K. Hagihara. A GPGPU Approach for Accelerating 2-D/3-D Rigid Registration of Medical Images. In *Springer Lecture Notes in Computer Science, 4330: Parallel and Distributed Processing and Applications*, pages 939–950, 2006.

[58] T. Jakobsen. Advanced Character Physics. In *Game Developers Conference proceedings*, 2001.

[59] M. W. Jones, J. A. Bærentzen, and M. Sramek. 3D Distance Fields: A Survey of Techniques and Applications. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):581–599, 2006.

[60] A. Joshi, D. Shattuck, P. Thompson, and R. Leahy. Surface-Constrained Volumetric Brain Registration Using Harmonic Mappings. *IEEE Transactions on Medical Imaging*, 26(12):1657–1669, Dec. 2007.

[61] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, 1:321–331, 1988.

[62] M. R. Kaus, K. K. Brock, V. Pekar, L. A. Dawson, A. M. Nichol, and D. A. Jaffray. Assessment of a Model-Based Deformable Image Registration Approach for Radiation Therapy Planning. *International Journal of Radiation Oncology*Biology*Physics*, 68(2):572 – 580, 2007.

[63] M. R. Kaus, V. Pekar, C. Lorenz, R. Truyen, S. Lobregt, J. Richolt, and J. Weese. Automated 3D PDM Construction Using Deformable Models. *Computer Vision, IEEE International Conference on*, 1:566, 2001.

[64] A. Kelemen, G. Szekely, and G. Gerig. Elastic model-based segmentation of 3-D neuroradiological data sets. *IEEE Transactions on Medical Imaging*, 18(10):828–839, Oct. 1999.

[65] J. R. Kent, W. E. Carlson, and R. E. Parent. Shape transformation for polyhedral objects. *SIGGRAPH Comput. Graph.*, 26(2):47–54, 1992.

[66] M. L. Kessler. Image registration and data fusion in radiation therapy. *The British journal of radiology*, 79(Spec No 1):99–108, 2006.

[67] A. Khamene, P. Bloch, W. Wein, M. Svatos, and F. Sauer. Automatic registration of portal images and volumetric CT for patient positioning in radiation therapy. *Medical Image Analysis*, 10:96–112, 2006.

[68] A. Khamene, R. Chisu, W. Wein, N. Navab, and F. Sauer. A Novel Projection Based Approach for Medical Image Registration. In *Proceedings of Third International Workshop on Biomedical Image Registration (WBIR)*, 2006.

[69] A. Köhn, J. Drexl, F. Ritter, M. König, and H. Peitgen. GPU Accelerated Image Registration in Two and Three Dimensions. In *Bildverarbeitung für die Medizin 2006*, pages 261–265, 2006.

[70] K. Kim, S. Park, H. Hong, and Y. G. Shin. Fast 2D-3D registration using GPU-based preprocessing. In *Proceedings of 7th International Workshop on Enterprise networking and Computing in Healthcare Industry (HEALTHCOM)*, pages 139–143, June 2005.

[71] D. Kincaid and W. Cheney. *Numerical Analysis: Mathematics of Scientific Computing*. Brooks/Cole, 2002.

[72] J. Kohlrausch, K. Rohr, and H. S. Stiehl. A New Class of Elastic Body Splines for Nonrigid Registration of Medical Images. *J. Math. Imaging Vis.*, 23(3):253–280, 2005.

[73] L. Komzsik. *Applied Calculus of Variations for Engineers*. CRC Press, 2009.

[74] A. Kubias, F. Deinzer, T. Feldmann, D. Paulus, B. Schreiber, and T. Brunner. 2D/3D image registration on the GPU. *Pattern Recognition and Image Analysis*, 18(3):381– 389, 2008.

[75] F. Labelle and J. R. Shewchuk. Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 57, New York, 2007. ACM.

[76] D. LaRose. *Iterative X-ray/CT Registration Using Accelerated Volume Rendering*. PhD thesis, Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, 2001.

[77] H. Lester and S. R. Arridge. A survey of hierarchical non-linear medical image registration. *Pattern Recognition*, 32(1):129–149, January 1999.

[78] D. Levin, D. Dey, and P. J. Slomka. Acceleration of 3D, nonlinear warping using standard video graphics hardware: implementation and initial validation. *Computerized Medical Imaging and Graphics*, 28(8):471 – 483, 2004.

[79] B. Li, A. Young, and B. Cowan. GPU Accelerated Non-rigid Registration for the Evaluation of Cardiac Function. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2008*, pages 880–887, 2008.

[80] J. Liang and D. Yan. Reducing uncertainties in volumetric image based deformable organ registration. *Med. Phys.*, 30:2116–2122, 2003.

[81] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169, New York, 1987. ACM Press.

[82] W. Lu, M.-L. Chen, G. H. Olivera, K. J. Ruchala, and T. R. Mackie. Fast free-form deformable registration via calculus of variations. *Physics in Medicine and Biology*, 49:3067–3087, July 2004.

[83] W. Lu, G. H. Olivera, Q. Chen, K. J. Ruchala, J. Haimerl, S. L. Meeks, K. M. Langen, and P. A. Kupelian. Deformable registration of the planning image (kVCT) and the daily images (MVCT) for adaptive radiation therapy. *Phys. Med. Biol*, 51:4357–4374, 2006.

[84] B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision (DARPA). In *Proceedings of the 1981 DARPA Image Understanding Workshop*, pages 121–130, 1981.

[85] A. Maciel and S. De. An efficient dynamic point algorithm for line-based collision detection in real time virtual environments involving haptics. *Comput. Animat. Virtual Worlds*, 19(2):151–163, 2008.

[86] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. Multimodality image registration by maximization of mutual information. *IEEE Transactions on Medical Imaging*, 16(2):187–198, 1997.

[87] G. E. Marai, D. H. Laidlaw, and J. J. Crisco. Super-Resolution Registration Using Tissue-Classified Distance Fields. *IEEE transactions on medical imaging*, 25:177–187, 2006.

[88] M. Marchal, E. Promayon, and J. Troccaz. Simulating Prostate Surgical Procedures with a Discrete Soft Tissue Model. In *Workshop in Virtual Reality and Physical Simulation*, 2006.

[89] G. K. Matsopoulos, N. A. Mouravliansky, P. A. Asvestas, K. K. Delibasis, and V. Kouloulias. Thoracic non-rigid registration combining self-organizing maps and radial basis functions. *Medical Image Analysis*, 9(3):237 – 254, 2005.

[90] D. Mattes, D. R. Haynor, H. Vesselle, T. K. Lewellen, and W. Eubank. PET-CT Image Registration in the Chest Using Free-form Deformations. *IEEE transactions on medical imaging*, 22:120–128, 2003.

[91] B. Matuszewski, J. Shen, and L.-K. Shark. Elastic image matching with embedded rigid structures using spring-mass system. In *Proceedings of the International Conference on Image Processing, 2003.*, volume 2, pages 937–340, 2003.

[92] B. McCane, K. Novins, D. Crannitch, and B. Galvin. On benchmarking optical flow. *Comput. Vis. Image Underst.*, 84(1):126–143, 2001.

[93] C. Miller, G. Joldes, D. Lance, and A. Wittek. Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. *Communications in Numerical Methods in Engineering*, 23(2):121–134, 2006.

[94] M. Modat, G. R. Ridgway, Z. A. Taylor, D. J. Hawkes, N. C. Fox, and S. Ourselin. A parallel-friendly normalized mutual information gradient for free-form registration. In J. P. W. Pluim and B. M. Dawant, editors, *SPIE Proceedings 7259: Medical Imaging 2009: Image Processing*, volume 7259. SPIE, 2009.

[95] J. Modersitzki. *Numerical Methods for Image Registration*. Oxford University Press, 2004.

[96] J. Mosegaard, P. Herborg, and T. S. Sørensen. A GPU accelerated spring mass system for surgical simulation. *Proceedings of Medicine Meets Virtual Reality 13. Studies in Health Technology and Informatics*, 111:342–348, January 2005.

[97] M. Müller, B. Heidelberger, M. Teschner, and M. Gross. Meshless deformations based on shape matching. *ACM Trans. Graph.*, 24(3):471–478, 2005.

[98] P. Muyan-Özcelik, J. Owens, J. Xia, and S. Samant. Fast Deformable Registration on the GPU: A CUDA Implementation of Demons. In *International Conference on Computational Sciences and Its Applications. ICCSA '08.*, pages 223–233, 2008.

[99] K. Ø. Noe, J. Mosegaard, K. Tanderup, and T. S. Sørensen. A framework for shape matching in deformable image registration. *Stud. Health. Technol. Inform.*, 132:333–335, 2008.

[100] K. Ø. Noe, B. D. D. Senneville, U. V. Elstrøm, K. Tanderup, and T. S. Sørensen. Acceleration and validation of optical flow based deformable registration for image-guided radiotherapy. *Acta Oncologica*, 47(7):1286–1293, 2008.

[101] K. Ø. Noe, K. Tanderup, J. C. Lindegaard, C. Grau, and T. S. Sørensen. GPU Accelerated Viscous-fluid Deformable Registration for Radiotherapy. Technical report, Dept. of Computer Science DAIMI PB-583, University of Aarhus, 2007.

[102] K. Ø. Noe, K. Tanderup, J. C. Lindegaard, C. Grau, and T. S. Sørensen. GPU accelerated viscous-fluid deformable registration for radiotherapy. *Stud. Health. Technol. Inform.*, 132:327–332, 2008.

[103] NVIDIA Corporation. *NVIDIA CUDA Programming guide*. `http://www.nvidia.com/object/cuda_develop.html`, 2009.

[104] J. C. O'Daniel, A. S. Garden, D. L. Schwartz, H. Wang, K. K. Ang, A. Ahamad, D. I. Rosenthal, W. H. Morrison, J. A. Asper, L. Zhang, S.-M. Tung, R. Mohan, and L. Dong. Parotid Gland Dose in Intensity-Modulated Radiotherapy for Head and Neck Cancer: Is What You Plan What You Get? *International Journal of Radiation Oncology*Biology*Physics*, 69(4):1290 – 1296, 2007.

[105] E. M. V. Osorio, M. S. Hoogeman, L. Bondar, P. C. Levendag, and B. J. M. Heijmen. A novel flexible framework with automatic feature correspondence optimization for nonrigid registration in radiotherapy. *Medical Physics*, 36(7):2848–2859, 2009.

[106] N. Paragios, M. Rousson, and V. Ramesha. Non-rigid registration using distance functions. *Computer Vision and Image Understanding*, pages 142–165, 2003.

[107] W. Peckar, C. Schnörr, K. Rohr, and H. S. Stiehl. Parameter-Free Elastic Deformation Approach for 2D and 3D Registration Using Prescribed Displacements. *J. Math. Imaging Vis.*, 10(2):143–162, 1999.

[108] S. Periaswamy and H. Farid. Elastic Registration in the Presence of Intensity Variations. *IEEE Transactions on Medical Imaging*, 22:865–874, 2003.

[109] M. Pharr. *GPU Gems 2 - Programming Techniques for High-Performance Graphics and General-Purpose Computation*. Nvidia Corporation, 2004.

[110] G. Picinbono, H. Delingette, and N. Ayache. Nonlinear and anisotropic elastic soft tissue models for medical simulation. *IEEE International Conference on Robotics and Automation (ICRA), 2001. Proceedings*, 2:1370–1375, 2001.

[111] G. Postelnicu, L. Zollei, and B. Fischl. Combined Volumetric and Surface Registration. *IEEE Transactions on Medical Imaging*, 28(4):508–522, 2009.

[112] E. Praun and H. Hoppe. Spherical parametrization and remeshing. *ACM Trans. Graph.*, 22(3):340–349, 2003.

[113] G. J. Price and C. J. Moore. A method to calculate coverage probability from uncertainties in radiotherapy via a statistical shape model. *Physics in Medicine and Biology*, 52(7):1947, 2007.

[114] M. Quicken, C. Brechbühler, J. Hug, H. Blattmann, and G. Szekely. Parameterization of closed surfaces for parametric surface description. In *IEEE Conference on Computer Vision and Pattern Recognition, 2000. Proceedings*, volume 1, pages 354–360, 2000.

[115] R. D. Rabbitt, J. A. Weiss, G. E. Christensen, and M. I. Miller. Mapping of hyperelastic deformable templates using the finite element method. volume 2573, pages 252–265. SPIE, 1995.

[116] A. Rasmusson, J. Mosegaard, and T. S. Sørensen. Exploring Parallel Algorithms for Volumetric Mass-Spring-Damper Models in CUDA. In *Lecture Notes In Computer Science; Vol. 5104: Proceedings of the 4th international symposium on Biomedical Simulation*, volume 5104, pages 49–58, 2008.

[117] T. Rehman, E. Haber, G. Pryor, J. Melonakos, and A. Tannenbaum. 3D nonrigid registration via optimal mass transport on the GPU. *Medical Image Analysis*, In Press, Corrected Proof:–, 2008.

[118] 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*, 18:712–712, 1999.

[119] S. S. Samant, J. Xia, P. Muyan-Özcelik, and J. D. Owens. High performance computing for deformable image registration: Towards a new paradigm in adaptive radiotherapy. *Medical Physics*, 35(8):3546–3553, 2008.

[120] D. Sarrut. Deformable registration for image-guided radiation therapy. *Zeitschrift fur medizinische Physik*, 16:285–97, 2006.

[121] J. Schnabel, C. Tanner, A. Castellano-Smith, A. Degenhard, M. Leach, D. Hose, D. Hill, and D. Hawkes. Validation of nonrigid image registration using finite-element methods: application to breast MR images. *IEEE Transactions on Medical Imaging*, 22(2):238–247, Feb. 2003.

[122] E. Schreibmann, G. T. Chen, and L. Xing. Image interpolation in 4D CT using a BSpline deformable registration model. *International Journal of Radiation Oncology*Biology*Physics*, 64(5):1537 – 1550, 2006.

[123] G. C. Sharp, N. Kandasamy, H. Singh, and M. Folkert. GPU-based streaming architectures for fast cone-beam CT image reconstruction and demons deformable registration. *Physics in Medicine and Biology*, 52:5771–5783, 2007.

[124] D. Shen and C. Davatzikos. HAMMER: hierarchical attribute matching mechanism for elastic registration. *IEEE Transactions on Medical Imaging*, 21(11):1421–1439, Nov. 2002.

[125] J. Shen, B. J. Matuszewski, L. Shark, and J. Moore. Deformable Image Registration using Spring Mass System. In *ICIP 2003. Proceedings*, 2003.

[126] P. Shirley and R. K. Morley. *Realistic Ray Tracing*. A. K. Peters, Ltd., Natick, MA, USA, 2003.

[127] M. Söhn, M. Birkner, Y. Chi, J. Wang, D. Yan, B. Berger, and M. Alber. Model-Independent, Multi-Modality Deformable Image Registration by Local Matching of Anatomical Features and Minimization of Elastic Energy. *Medical Physics*, 35(3):866–878, 2008.

[128] M. Söhn, M. Birkner, D. Yan, and M. Alber. Modelling individual geometric variation based on dominant eigenmodes of organ deformation: implementation and evaluation. *Physics in Medicine and Biology*, 50:5893–5908, Dec. 2005.

[129] T. Sørensen and J. Mosegaard. An Introduction to GPU Accelerated Surgical Simulation. *3rd Symposium on Biomedical Simulation. Zurich, Switzerland*, pages 93–104, 2006.

[130] T. S. Sørensen, T. Schaeffter, K. Ø. Noe, and M. S. Hansen. Accelerating the Non-equispaced Fast Fourier Transform on Commodity Graphics Hardware. *IEEE Transactions on Medical Imaging*, 27(4):538–547, 2008.

[131] G. Soza, M. Bauer, P. Hastreiter, C. Nimsky, and G. Greiner. Non-rigid Registration with Use of Hardware-Based 3D Bézier Functions. In *MICCAI 2002: Proceedings of the 5th International Conference on Medical Image Computing and Computer-Assisted Intervention Part II*, pages 549–556, London, UK, 2002. Springer-Verlag.

[132] R. Strzodka, M. Droske, and M. Rumpf. Image Registration by a Regularized Gradient Flow - A Streaming Implementation in DX9 Graphics Hardware. *Computing*, 73(4):373–389, 2004.

[133] A. Sud, M. A. Otaduy, and D. Manocha. DiFi: Fast 3D Distance Field Computation Using Graphics Hardware. *EUROGRAPHICS*, 23, 2004.

[134] K. Tanderup. *Optimisation of treatment planning and delivery in 3D brachytherapy.* PhD thesis, Faculty of Health Sciences, University of Aarhus, 2007.

[135] Z. A. Taylor, M. Cheng, and S. Ourselin. High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. *IEEE Transactions on Medical Imaging*, 27(5):650–663, May 2008.

[136] C. Thilmann, S. Nill, T. Tucking, B. Hesse, L. Dietrich, B. Rhein, P. Haering, U. Oelfke, J. Debus, and P. Huber. Correction of Patient Positioning Errors based on In-Line Cone Beam CTs: Clinical Implementation and First Experiences. *International Journal of Radiation Oncology*Biology*Physics*, 63:550–551, 2006.

[137] J. P. Thirion. Fast Non-Rigid Matching of 3D Medical Images. *Rapports de recherche - INRIA-RR - 2547*, 2547, 1995.

[138] J.-P. Thirion. Non-rigid matching using demons. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR '96*, pages 245–251, 1996.

[139] P. Trier, K. Ø. Noe, M. S. Sørensen, and J. Mosegaard. The visible ear surgery simulator. *Stud. Health. Technol. Inform.*, 132:523–525, 2008.

[140] D. Tschumperlé. Fast Anisotropic Smoothing of Multi-Valued Images using Curvature-Preserving PDE's. *Int. J. Comput. Vision*, 68(1):65–82, 2006.

[141] J. Vandemeulebroucke, D. Sarrut, and P. Clarysse. The POPI-model, a point-validated pixel-based breathing thorax model. In *XVth International Conference on the Use of Computers in Radiation Therapy (ICCR), Toronto, Canada.*, 2007.

[142] C. Vetter, C. Guetter, C. Xu, and R. Westermann. Non-rigid multi-modal registration on the GPU. volume 6512, page 651228. SPIE, 2007.

[143] P. Viola and W. M. Wells III. Alignment by Maximization of Mutual Information. *International Journal of Computer Vision*, 24:137–154, 1997.

[144] H. Wang, L. Dong, J. O'Daniel, R. Mohan, A. S. Garden, K. K. Ang, D. A. Kuban, M. Bonnen, J. Y. Chang, and R. Cheung. Validation of an Accelerated 'Demons' Algorithm for Deformable Image Registration in Radiation Therapy. *Phys. Med. Biol.*, vol. 50:pp. 2887–2905, 2005.

[145] S. K. Warfield, S. J. Haker, I.-F. Talos, C. A. Kemper, N. Weisenfeld, A. U. Mewes, D. Goldberg-Zimring, K. H. Zou, C.-F. Westin, W. M. Wells, C. M. Tempany, A. Golby, P. M. Black, F. A. Jolesz, and R. Kikinis. Capturing intraoperative deformations: research experience at Brigham and Women's hospital. *Medical Image Analysis*, 9(2):145 – 162, 2005.

[146] G. Wollny and F. Kruggel. Computational cost of nonrigid registration algorithms based on fluid dynamics [MRI time series application]. *IEEE Transactions on Medical Imaging*, 21(8):946–952, Aug. 2002.

[147] G. Xiao, S. Ong, and K. Foong. 3D registration of partially overlapping surfaces using a volumetric approach. *Image and Vision Computing*, 25(6):934 – 944, 2007.

[148] L. Xiong, A. Viswanathan, A. J. Stewart, S. Haker, C. M. Tempany, L. M. Chin, and R. A. Cormack. Deformable structure registration of bladder through surface mapping. *Medical Physics*, 33(6):1848–1856, 2006.

[149] C. Xu and J. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7(3):359–369, 1998.

[150] D. Yan, D. A. Jaffray, and J. W. Wong. A model to accumulate fractionated dose in a deforming organ. *International Journal of Radiation Oncology*Biology*Physics*, 44(3):665 – 675, 1999.

[151] D. Yan, D. Lockman, A. Martinez, J. Wong, D. Brabbins, F. Vicini, J. Liang, and L. Kestin. Computed Tomography Guided Management of Interfractional Patient Variation. *Seminars in Radiation Oncology*, 15(3):168 – 179, 2005.

[152] D. Yang, H. Li, D. A. Low, J. O. Deasy, and I. El Naqa. A fast inverse consistent deformable image registration method based on symmetric optical flow computation. *Physics in Medicine and Biology*, 53:6143–6165, Nov. 2008.

[153] Y. Yang, E. Schreibmann, T. Li, C. Wang, and L. Xing. Evaluation of on-board kV cone beam CT (CBCT)-based dose calculation. *Physics in Medicine and Biology*, 52:685–705, Feb. 2007.

[154] T. Zhang, Y. Chi, E. Meldolesi, and D. Yan. Automatic Delineation of On-Line Head-And-Neck Computed Tomography Images: Toward On-Line Adaptive Radiotherapy. *International Journal of Radiation Oncology*Biology*Physics*, 68(2):522 – 530, 2007.

[155] T. Zhang, N. P. Orton, T. R. Mackie, and B. R. Paliwal. Technical note: A novel boundary condition using contact elements for finite element based deformable image registration. *Medical Physics*, 31(9):2412–2415, 2004.

[156] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13:119–152, 1994.

# Index