UNIVERSITÉ DE NICE – SOPHIA-ANTIPOLIS
UFR Sciences

École Doctorale de Sciences et Technologies
de l'Information et de la Communication

T H È S E

pour obtenir le titre de

Docteur en Sciences

de l'Université de Nice Sophia-Antipolis

Spécialité : Informatique

presentée et soutenue par

Radu-Constantin STEFANESCU

Parallel nonlinear registration of medical images with a priori information on anatomy and pathology

Thèse dirigée par Nicholas AYACHE
et préparée à l'INRIA Sophia-Antipolis, projet EPIDAURE

Soutenue le 23 mars 2005

Jury

Christian	**BARILLOT**	Directeur de recherches	Président
Nicholas	**AYACHE**	Directeur de recherches	Directeur
Xavier	**PENNEC**	Chargé de recherches	Co-directeur
Derek	**HILL**	Professeur	Rapporteurs
Dirk	**VANDERMEULEN**	Professeur	
Pierre-Yves	**BONDIAU**	Médecin	Invité

Contents

Chapter 1

Introduction

This thesis deals with the registration of medical images, and particularly images coming from different subjects. Intersubject registration is a fundamental problem in medical image analysis. It facilitates anatomical comparisons and studies, and provides a general method to segment almost any medical image by using an anatomical atlas. If we have an "atlas" image and its segmentation, then any subject image of the same part of the body can be segmented by registering it with the atlas and then propagating the atlas segmentation in the subject geometry.

Atlas to subject registration is a difficult problem. Although we register images that represent the same anatomical structures (mostly the brain throughout this thesis), the anatomy of different individuals is generally different. The problem is further complicated if one of the images comes from a patient with a pathology. This case is commonly encountered in atlas to subject registration, when the atlas image represents a healthy anatomy, and the subject is a clinical patient. In neuroscience, this is also sometimes the case, even if the subject is supposed to be healthy.

Throughout this thesis, we follow a multidisciplinary approach to registration, and address the issue regarding both the quality of the result, and the integration of the registration software into the real clinical world. We therefore also tackle the problems of computation time, and take the first steps towards the integration with clinical visualization systems.

1.1 Context

The two clinical applications that are addressed in this thesis involve the segmentation of various structures in the human brain. Both applications are involved in

pre-operative planning, and the general methods are similar. However, their practical requirements may slightly differ, particularly regarding the size and localization of the structures to segment.

1.1.1 Conformal brain radiotherapy

The treatment of cerebral tumor may involve surgery, radiotherapy, or chemotherapy. The radiotherapy using one or more irradiating beams must satisfy two goals: the destruction of the tumor and the preservation of healthy structures. Special care must be taken in order to preserve "high-risk" anatomical structures that are vital for the patient's life or living comfort: eyes, optical tracts, brain stem, etc.

Optimizing the position and the shape of the beam requires an accurate localization of different structures (tumors and high risk structures). Different techniques using photon beams allow to achieve both goals:

- Many beams may be used.
- The intensity of the beams may be modulated during irradiation. This technique is called "Intensity Modulated Conformal Radiotherapy".
- The shapes and positions of the beams may be modified, depending on the lesions to irradiate and the healthy structures to avoid. This technique is called "3D Conformal Radiotherapy".

When localizing these structures, the required level of accuracy is dependent on the structure: we are looking for an accuracy of about 1mm for small structures such the optical tracts, and we can tolerate a lower accuracy (of about 2mm) for larger structures, like the brain stem.

The quality of the treatment is evaluated by quantifying the amount of radiation delivered to the lesion and to high risk organs. In order to achieve a satisfying level of safety, an extreme accuracy is required at each step of the treatment chain: data acquisition, segmentation of structures, definition of the beams, and control of the patient's position during irradiation. The risk in case of failure is high: parts of the tumor tissue may survive the therapy with lethal consequences, and the irradiation of healthy structures may induce loss of senses and/or movement ability, or even death. Since the clinical act must be very accurate, it requires a large amount of time. Various software tools may help the clinician at different steps of the procedure:

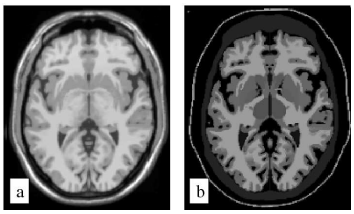


Figure 1.1: A slice of the 3D anatomical atlas from the Centre Antoine Lacassagne (Bondiau et al. (2004)). The T1-MR image (a) has been manually labeled (b).

1. The acquired images are transfered to the "treatment planning system", and they are rigidly registered. This registration step is multimodal, since the images are generally acquired using different modalities.
2. The structures of interest are segmented.
3. The positions of the beams are computed, using information provided by the first two steps.
4. The tumor is irradiated.

Traditionally, the segmentation of brain structures is manual and each structure has to be delineated in each slice of a 3-D image (e.g. MRI). The treatment team spends a significant amount of time to delimit the structures of interest (eyes, optical tracts, etc.) with the precision requested for the conformal radiotherapy. An automatic segmentation algorithm of all the critical structures in a patient image is then an invaluable tool for radiotherapy, and its main requirement is a precise delineation of the structures of interest. We are investigating this application in close collaboration with the *Centre Antoine Lacassagne* hospital (33 Avenue de Valombrose, 06189 Nice cedex 2, France).

In order to extract the structures of interest in a specific patient's image, Bondiau et al. (2004) built a numerical reference atlas of all these structures. The segmentation approach consists in using matching techniques to warp this atlas onto one patient's image. The atlas (Fig. 3.2b) was manually labeled from an artificial MR image (obtained from the Brainweb[1], see Fig. 3.2a). The first step is a rigid matching between the atlas image and the patient MRI's (usually T1, T2 and T1 injected). The recovered transformation is refined using non-rigid registration, and then applied to the labels of the atlas in order to obtain a segmentation of the patient image.

The nonlinear registration step is challenging: First, due to its multi-subject nature, this registration problem is generally difficult. The topology of the brain, the

[1] `http://www.bic.mni.mcgill.ca/brainweb/`

shape of the ventricles, the number and shape of the sulci vary strongly from one individual to another. The ideal transformation is smooth in some places, and has fine details in others. Thus, not only do we have to deal with the ambiguity of the structures to match, but we also have to take into account the large variability of the deformations between the two brains. Second, an issue that arises in our case is the presence of pathologies in the patient image, such as tumors or surgical resections. These structures have no equivalent in the atlas. They usually lead the non-rigid registration to important errors, especially around the pathology which is the area of interest for radiotherapy. Finally, a software running inside the clinical environment has to fulfill some requirements linked to the practical organization of hospital work. One of these is the computation time, which should be low when compared to the duration of the pre-operative planning itself. In order to simplify the problem, the atlas image was chosen to be of the same modality as the patient images, which allows us to perform a (simpler and faster) monomodal registration.

1.1.2 Stereotactic neurosurgery for Deep Brain Stimulation in Parkinsonian patients

Deep Brain Stimulation, the second clinical application we consider in this thesis, is a procedure that greatly reduces disabling symptoms in patients of Parkinson's disease: walking problems, tremor, rigidity, slow movement. It uses a battery-powered device called *neurostimulator*, which electrically stimulates specific areas in the brain that are responsible for the above-mentioned symptoms, like the subthalamic nuclei. In essence, this device works similarly to a pacemaker controlling the beat of the heart. The neurostimulator has three parts: the electrode, the extension, and the neurostimulator itself. The neurostimulator device is about the size of a watch (Fig. 1.2) and is normally implanted near the clavicle. The electrode (a thin wire) is inserted into the skull and implanted in the target inside the brain. The extension is a wire passed under the skin connecting the electrode to the neurostimulator. Once in place, the neurostimulator sends electrical signals through the extension wire and to the electrode within the brain.

The introduction of the electrode inside the brain is performed through surgery. At the La Pitié Salpêtrière hospital in Paris, a metallic non-magnetic stereotactic frame guides the electrodes' insertion (Fig. 1.3). This frame, fixed on the patient's head and visible from different modalities (T1-weighted and T2-weighted MR images, and CT-scans), is used as a geometrical referential. The target is first located on pre-operative images, and then the path of the electrodes is planned through the parenchyma in order to avoid high risk structures. At the beginning of the procedure, the surgeon performs two holes in the skull in order to access the two hemispheres. The area surrounding the subthalamic nuclei is explored systematically with the electrode, in order to precisely determine the target location.

Figure 1.2: The Medtronic® Kinetra neurostimulator (bottom-left) and its Access Therapy Controller (top-right), allowing its control by a physician. The implanted stimulator is about the size of a watch. Image courtesy of Medtronic, Inc.

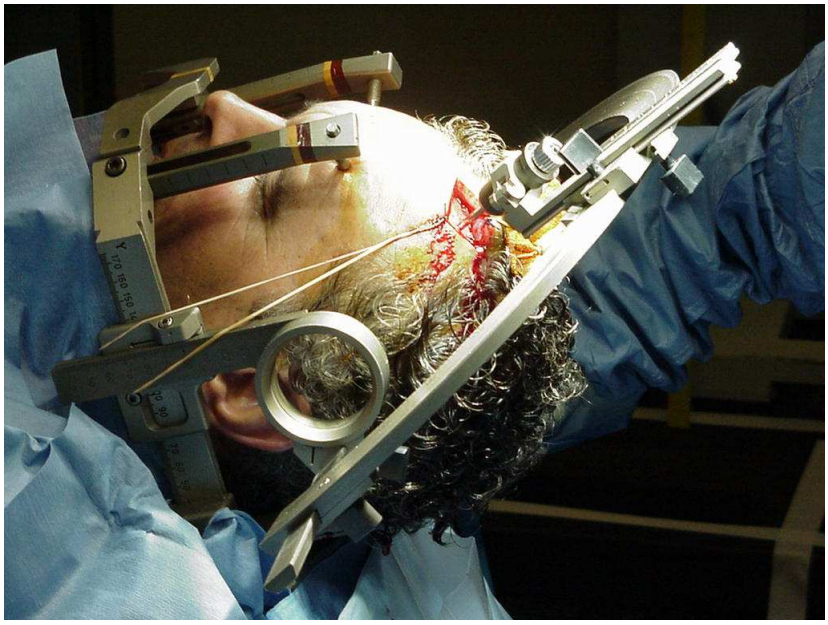


Figure 1.3: Stereotactic implantation of electrodes in the subthalamic nuclei for the Parkinson's disease at the La Pitié Salpêtrière hospital (Paris, France). The stereotactic frames fixed on the skull is used as a referential and as a guiding system for the electrodes. The image is courtesy of Medtronic, Inc.

One of the most difficult parts of the intervention is the accurate determination of the target. Like in the previous application, the patient image is registered with an anatomical atlas, and segmented structures in the atlas are deformed towards the patient's geometry. The positions of the patient's subthalamic nuclei are estimated by transforming their positions from the atlas. The requirements for the registration algorithm are similar to the ones imposed by the conformal brain radiotherapy planning. The registration is multi-subject, which, once again, means that the degree of smoothness of the transformations is highly variable. Furthermore, since the structures we are looking for are small, a high degree of regularity is required for their contours. The pre-operative planning being done in a single day, the registration should only take a minimal amount of time.

A requirement of this application is that the algorithm works not only for the subthalamic nuclei, but also for the other central grey nuclei. This may allow to generalize the usage of the algorithm for other nearby anatomical structures. The accuracy we are looking for is of about 2mm.

1.1.3 Discussion on the clinical requirements

The two clinical applications presented in this section have common characteristics. First, they require the segmentation of structures in the human brain. Second, the chosen approach is for both the registration of the patient image with an anatomical atlas. This registration is multisubject, so the algorithm will have to deal with the ambiguity of matching two brains.

However, each application has a specific challenge. On one hand, patients that must undergo conformal brain radiotherapy have either tumors or surgical resections inside the brain. These pathological structures have no correspondent in the atlas, and there is a danger to match "pathological voxels" inside them with some "normal voxel" in the atlas (belonging to a normal tissue). This would make the registration locally fail. On the other hand, the segmentation of the central grey nuclei must be very accurate, and respectful of the anatomy: the contour provided by the algorithm should have the strict shape of the corresponding anatomical structure. Since noise or partial volume effect tend to influence the registration, it is difficult to ensure both a high accuracy, and a good regularity of the segmented structures.

Regarding the practical use, both applications have a time constraint. Since the algorithm is to be run inside the clinical environment, its computation time should always be kept low with respect to the duration of the clinical act itself. We believe that 5 minutes is a reasonable time limit.

1.2 Manuscript organization

Chapter 2 presents the two clinical applications that led to the research presented in this thesis, and the way multisubject nonlinear registration represents a solution to them. A brief review of currently existing nonlinear registration methods shows that a certain class of algorithms (called "pair-and-smooth") is appropriate for solving our problems. However, currently existing algorithms of this class do not completely suit our purpose.

In this framework, Chapter 3 introduces a registration algorithm which addresses the issues listed in the previous chapter: robustness with respect to image noise, and the ability to incorporate a priori knowledge about the anatomy and pathology. A fast implementation based on an implicit numerical scheme is also presented, as well as illustrations on two clinical applications.

In Chapter 4, we refine the modeling of anatomical a priori information with the introduction of anisotropy. This evolution enables a regularization dependent not only on the localization, but also on the direction. This idea is applied to the preservation of the surface coherence of anatomical structures.

Chapter 5 addresses the computation time issue. In order to reduce execution time and enable a better integration inside the clinical environment, we present a parallel implementation of the algorithm on a cluster of networked personal computers, and also describe novel parallelization methods for some commonly used image processing algorithms.

The effort to integrate the algorithm into a clinical image processing system is continued in Chapter 6. In order to enable the use of a graphical user interface and thus the embedding of the registration software into a clinical interactive processing pipeline, the registration software has been implemented as a grid service, callable by a graphical client.

Finally, Chapter 7 provides a summary and discussion of our contributions, as well as hopefully insightful ideas to further extend this work.

Chapter 2

Nonlinear registration: the pair-and-smooth approach

This chapter briefly overviews existing registration techniques in view of our clinical context (Section 1.1). It reveals that a specific class of registration algorithms called "pair-and-smooth" exhibits some interesting features for our clinical applications. However, some open issues have to be solved.

2.1 A brief overview of nonlinear registration

The world of medical image registration contains many different methods which may potentially fulfill our requirements. This section reviews several algorithms that we found to be the most representative for the techniques they use, without aiming at being exhaustive. For complete information, we refer the reader to a survey of registration algorithms (e.g. Maintz and Viergever (1998); Hajnal et al. (2001); Pluim et al. (2003); Modersitzki (2004)). We classify them along two axes: the image features they consider for matching, and the dimensionality of their transformation space.

2.1.1 Feature space

Based on their feature space, two classes of registration algorithms have emerged. Some algorithms, that we call "geometric", attempt to extract some higher-level geometric (often anatomical) structures in the image, and then they match these structures. Others, called "iconic", use the intensities of all image voxels in order to estimate a transformation.

Geometric registration

The dimensionality of the geometric structures used as features varies. Points, curves or surfaces have been segmented and matched, and different methods to extrapolate this correspondence to the entire image have emerged.

Thirion (1996a) uses differential operators in order to compute crest lines of the brain. Based on a randomly chosen starting "seed" inside the brain, he uses the Marching Lines algorithm in order to follow the line to the neighboring locations. On the resulting lines, he estimates points where the curvature of the line is locally extremal. The algorithm then estimates the rigid transformation that best superimposes the extracted points in the two images.

Chui et al. (1999) use robust point matching techniques to register cortical structures in brain images. Rohr et al. (2001) use 3D extensions of 2D corner operations (Rohr (1997)) in a semi-automatic methods that estimates landmarks inside brain images, typically on the skull or on the surface of the ventricles. In their method, the user selects a region of interest, and corner operators are applied inside it in order to estimate landmarks. After the user confirmation, these points are matched, and the correspondences are injected into an interpolating thin-plate splines model (Bookstein (1989); Evans et al. (1991)).

Collins et al. (1996), followed by Collins et al. (1998), add the information provided by the brain sulcal lines to an iconic registration algorithm (see Section 2.1.1). The sulci segmentation method, called SEAL (Sulcal Extraction and Automatic Labeling) uses an active model similar to a snake, which is submitted to a set of forces derived from the local image characteristics (Le Goualher et al. (1997)). The extracted sulci are matched using a chamfer distance function as a similarity criterion (Sandor and Leahy (1995)). Vaillant and Davatzikos (1999) integrated sulcal information into a registration algorithm based on surface matching.

Sulcal lines were also integrated into iconic registration algorithms by Hellier and Barillot (2001) (see also Hellier and Barillot (2003)) and Cachier et al. (2001). In the last approach, sulci are segmented in four steps: non-uniform bias field correction, segmentation of grey matter/CSF, homotopic skeletonization and splitting into simple surfaces. An algorithm based on a neural network trained on a manually-labeled set (Riviere et al. (2000)) is then used in order to label the sulci. The sulcal lines are inserted as an additional term into a registration algorithm based on a variational approach (see the iconic registration section below).

Ferrant et al. (2000) first segment the external surface of the brain and the surface of the ventricles. An active surface algorithm deforms the segmented borders in one image into the same borders from the other image. The surface is modeled as an elastic membrane, deformed by image-driven forces. The resulting displacement on the borders is then inserted, based on a finite element method developed

by Cohen and Cohen (1991), into a linearly elastic finite element model which computes a displacement for each node of a volumetric mesh that covers the entire brain. Surface matching has also been used by Thompson and Toga (1996) to drive the registration of brain images.

Liu et al. (2004) perform multi-subject registration by matching the grey matter/white matter surfaces of the two brains and coupling this information to a volumetric mapping (Shen and Davatzikos (2002)). Ganser et al. (2004) propose an electronic Talairach and Tournoux atlas (Talairach and Tournoux (1988, 1993)). The authors interactively segmented 49 anatomical structures (ventricles, thalamus, putamen, etc.) inside the atlas MR image. Four classes of structures are also segmented in the patient image: cortex, ventricles, tumors and sulci. After an initial registration step using the Talairach referential, the authors use three separate matching algorithms for different types of structures (cortex, ventricles, tumors) in order to refine the deformation in a radial basis functions framework.

All the methods described in this section have two aspects in common: 1) They depend on an initial segmentation of "relevant" structures in the images. 2) They require the correct matching of these structures to perform the global registration of the images. These methods have the advantage of being potentially able to achieve a very good accuracy on the structures. However, for our two applications, they present three drawbacks:

1. Segmentation is by itself a difficult problem. Few automatic methods exist, and, like in the case of many other classes of algorithms (including registration), most of them are not 100% reliable. Whereas a minor, localized segmentation error would go unnoticed, problems in the initial segmentation may become acute if the segmentation is used in order to match the entire image. In this case, the shapes of supposedly corresponding structures may differ in an anatomically incoherent manner, with unpredictable results on the quality of the final registration.

2. The matched features do not cover the entire image. The accuracy of the correspondence field they provide tends to be valid only in their neighborhoods, and results generally degrade in regions that are far away from them. Furthermore, the relative position of different structures inside the brain tends to vary in a nonlinear way from one subject to another (e.g. achieving a perfect registration of the ventricles and the cortex does not guarantee a good registration of the putamen).

3. In both applications, the structures we want to segment sometimes exhibit a low contrast, and therefore cannot be used as features. Contrasted structures are generally far away, so their correspondence provides inaccurate information (as explained at item 2).

Let us therefore examine a second class of methods that estimate correspondences in two images, which have the advantage of using the entire information available.

Iconic registration

These methods rely on the optimization of a *similarity criterion* between the intensities of the two images. The registration is performed by optimizing this criterion with respect to the transformation. So far, many similarity criteria have been proposed. This section reviews some of the most widely used. Let us first define some notations that will be used in this section and throughout the manuscript:

- I and J are the two images to register. Since we usually work in 3D, the images are defined as functions $\mathbb{R}^3 \longrightarrow \mathbb{R}$. Remark that images are discrete and have a finite spatial support. By considering their discrete nature, and N_x, N_y and N_z the number of voxels along the x, y and z axes, then an image is a function defined on $\Omega = [1...N_x] \times [1...N_y] \times [1...N_z]$ with values in $\mathbb{R}$.

- $p \in \mathbb{R}^3$ is a 3D point. For a given image I, $I(p)$ is the intensity of the image at this point. If $p \in \mathbb{Z}^3$ (we call it a voxel), its intensity can be directly recovered from the 3D array that represents the image. Otherwise, its value has to be interpolated from the values of its neighbors. For a good review of existing interpolation methods, the reader should consult Lehmann et al. (1999) or Thévenaz et al. (2000).

- T is a transformation (mapping) that links some point p in I to its correspondent q in J. For simplicity, $T : \mathbb{R}^3 \longrightarrow \mathbb{R}^3$. U is the corresponding displacement field ($U : \mathbb{R}^3 \longrightarrow \mathbb{R}^3$), defined as $U(p) = T(p) - p$. Applying a transformation T to an image J yields an image J' where $J'(p) = J(T(p))$. In this case, we say that $J' = J \circ T$. We extend this notation to the displacement field: $J' = J \circ U$ if and only if $J'(p) = J(p + U(p))$ for all points p.

- One can "compose" two displacement fields by composing their associated transformations. Let U_1 and U_2 be two displacement fields and T_1 and T_2 their associated transformations. If $T_1(p) = p + U_1(p)$ and $T_2(p) = p + U_2(p))$, then $(T_1 \circ T_2)(p) = T_1(T_2(p))$. Therefore, $(U_1 \circ U_2)(p) = U_1(p + U_2(p)) + U_2(p)$.

Sum of squared distances (SSD) This simple similarity criterion makes the assumption that two corresponding voxels have (almost) the same intensity. For some

point $p \in \Omega_I$, $T(p)$ is found by minimizing the function $(I(p) - J(T(p)))^2$. By integrating over the entire image, the criterion to minimize is

$$SSD(I, J, T) = \|I - J \circ T\|^2 = \int_{p \in \Omega} (I(p) - J(T(p)))^2 \, dp$$

A widely used method to minimize the SSD is the optical flow (Horn and Schunk (1981)), which computes the displacement U as an iterative minimization

$$U \leftarrow U + \frac{I - J \circ U}{\|\nabla I\|^2} \nabla I$$

This criterion is relevant only if the two images where acquired using the same modality. Even so, its validity may be doubtful, especially if one image is affected by a bias. Differences in the tuning of the acquisition equipment used for the two images also partially invalidate this criterion. Thus, other similarity criteria may be necessary.

Correlation coefficient (CC) The correlation coefficient has been widely used as a robust similarity criterion (Keriven and Faugeras (1998); Netsch et al. (2001); Ourselin et al. (2000)). It is relatively constrained, and it measures a global affine dependency between the intensities of the two images:

$$CC(I, J, T) = \|I - a \, J \circ T - b\|^2 = min_{a,b} \int_p (I(p) - a \, J(T(p)) - b)^2 \, dp$$

In order to avoiding a minimum of the criterion when the overlap of the two images is small, Roche et al. (2000) proposed its renormalization to $CC(I, J, T) = \|I - a \, J \circ T - b\|^2 / Var[I]$, where $Var[I]$ is the variance of the intensity of the image I. Let us also denote by $E[I]$ the mean intensity of the image I. The estimation of the affine parameters a and b leads to the following criterion to maximize:

$$CC(I, J, T) = \frac{E[I] \, E[J \circ T] - E[I \, J \circ T]}{\sqrt{(E[I^2] - E[I]^2) \; (E[(J \circ T)^2] - E[J \circ T]^2)}}$$

A local variant of this criterion has been introduced by Cachier and Pennec (2000). By defining the local average of an image around a point p as $E_p[I] = G_p * I$, where G_p is a Gaussian centered on p, and by integrated over p, we get

$$LCC(I, J, T) = \int_p \frac{G_p * [(I - G_p * I) \; (J \circ T - G_p * (J \circ T))]}{\sqrt{[G_p * (I - G_p * I)] \; [G_p * (J \circ T - G_p * (J \circ T))]}}$$

The local criterion allows for the affine dependency measured by the correlation coefficient to change depending on the position in space.

Correlation ratio (CR) The correlation ratio was introduced by Roche et al. (1998) as a similarity measure for multimodal registration and extended to the deformable case by Guimond et al. (2001a). It generalizes the affine dependency encountered in the correlation coefficient to an arbitrary functional relationship

$$CR(I, J, T) = min_f \frac{\|I - f(J \circ T)\|^2}{Var[I]}$$

Here, f is a function that maps the intensities of the images I and J.

The Woods criterion (W) When registering MR and PET images, Woods et al. (1993) assumed that the range of values in the PET image corresponding to a particular value in the MR image should be minimized. If Ω_j is the subregion of the overlapping image space Ω where image intensities in image $J \circ T$ are j, and μ_j and σ_j are, respectively, the average intensity and the standard deviation in image I of the subregion Ω_j, then Woods criterion is

$$W(I, J, T) = \frac{1}{\|\Omega\|} \sum_{\Omega_j} \|\Omega_j\| \frac{\sigma_j}{\mu_j}$$

In the equation above, $\|\Omega\|$ denotes the number of voxels inside the region Ω.

Joint entropy (JE) In the discrete case, for the two image to match at each optimization step (I and $J \circ T$), the spatially correlated presence of, respectively, the x value in image I, and the y value in the image $J \circ T$ provides a quantity of information equal to $-\log_2 p(I = x, J \circ T = y)$ (where $p(I = x, J \circ T = y)$ is the probability). The average quantity of information is called the *joint entropy* of the two images $\mathcal{H}(I, J) = -\sum_y \sum_x p(I = x, J \circ T = y) \log_2 p(I = x, J \circ T = y)$. In its continuous version, the joint entropy has the expression $\mathcal{H}(I, J) = -\int_{x,y \in \mathbb{R}} f_{I,J \circ T}(x, y) \ln f_{I,J \circ T}(x, y)\, dx\, dy$ (where $f_{I,J \circ T}$ is the joint distribution of the intensities of the images I and $J \circ T$). This criterion was used for multimodal registration (Collignon et al. (1995); Studholme et al. (1995)):

$$JE(I, J, T) = \mathcal{H}(I, J \circ T)$$

The *Shannon entropy* is defined (similarly to the joint entropy) for a discrete random variable as $\mathcal{H}(I) = -\sum_x p(I = x) \log_2 p(I = x)$. Its continuous version (called *differential entropy*) has the expression $\mathcal{H}(I) = -\int_{x \in \mathbb{R}} f_I(x) \ln f_I(x)\, dx$ (where f_I is the distribution of the intensity of the image I).

Mutual information (MI) Mutual information is also used as a similarity criterion for multimodal registration (Collignon et al. (1995); Viola (1995); Viola and

Wells (1997)). The criterion to maximize measures the distance between the joint distribution of the two image ($f_{I,J\circ T}(x,y)$) and their joint distribution in case of independence ($f_I(x)\ f_{J\circ T}(x)$):

$$\begin{aligned} MI(I,J,T) &= -\int_{x,y\in\mathbb{R}} f_{I,J\circ T}(x,y)\log_2\frac{f_{I,J\circ T}(x,y)}{f_I(x)\ f_{J\circ T}(x)} \\ &= \mathcal{H}(I)+\mathcal{H}(J\circ T)-\mathcal{H}(I,J\circ T) \end{aligned}$$

This criterion is in widespread use (e.g. Hermosillo et al. (2002); D'Agostino et al. (2003)), since it does not require any assumption on the type of correlation between the intensities of the two images.

Normalized mutual information (NMI) This criterion is a normalized version of the previous one (Studholme et al. (1999)):

$$NMI(I,J,T)=\frac{\mathcal{H}(I)+\mathcal{H}(J\circ T)}{\mathcal{H}(I,J\circ T)}$$

One advantage of this normalized similarity measure is that it avoids the possibility of the mutual information increasing when the overlap region between the two images (I and $J\circ T$) decreases. This measure was used for various registration applications (e.g. Rueckert et al. (1999); Denton et al. (2000); Rohlfing and Maurer (2001)).

Entropy correlation coefficient (ECC) Another way to normalize the mutual information was proposed by Maes et al. (1997); Collignon (1998):

$$ECC(I,J,T)=\frac{2\left(\mathcal{H}(I)+\mathcal{H}(J\circ T)-\mathcal{H}(I,J\circ T)\right)}{\mathcal{H}(I)+\mathcal{H}(J\circ T)}$$

This measure is related to the NMI ($ECC=2-2/NMI$).

Discussion In atlas to subject registration, the atlas image is chosen to be of the same modality as the patient images (see Section 1.1). This simplifies the similarity criterion, and generally allows us to use the SSD. In practice, this "monomodal" registration problem is more difficult: the parameters of the acquisition device may change, leading to different levels of intensities for the same anatomical structure in different images; and the images may be affected by bias. In such situations, more powerful criteria, such as the global or local correlation coefficient are necessary. For simplicity, we illustrate the algorithm presented in this thesis using the SSD, but it can be replaced if needed with more powerful similarity metrics. As for our applications we are only interested in monomodal registration, we do not expect this to have any influence on the results.

2.1.2 Transformation space

The model of transformation determines the quality of the result in an essential manner. The number of degrees of freedom that are allowed for the transformation ranges from the rigid transformation (6 degrees of freedom) to the dense displacement field (one 3D vector per voxel). This section provides a brief survey of the most wide-spread transformation models.

Rigid / affine transformation

These are the simplest models. The rigid transformation assumes that the two image present the same objects in the same state, but in different positions. Therefore, the transformation is modeled through a rotation/translation matrix (6 degrees of freedom). The affine transformation is a generalization of the rigid one: the rotation is replaced by a general linear transformation matrix, thus adding shear and scaling and increasing the number of degrees of freedom to 12. Rigid registration is the most appropriate for non-deformable objects (e.g. the head) if the two images belong to the same subject, and no surgical intervention has occurred between their acquisition times. Affine registration is the simplest form of deformable registration, and it can be used if deformations are very low. A large amount of work has been dedicated to recovering rigid and affine transformations (e.g. Woods et al. (1993); Collignon et al. (1994); Hill et al. (1994); Collignon et al. (1995); Maes et al. (1997); Viola and Wells (1997); Bro-Nielsen (1997); Ourselin et al. (2000, 2001)), and computation times are generally low, which makes them well suited for clinical use (Ourselin et al. (2002)).

Parametric deformations

The deformation is modeled using a tunable number of parameters, larger than the 12 parameters of an affine transformation, and generally much lower than one vector per voxel.

In order to achieve a fast implementation of elasticity-based deformation models (previously introduced by Broit (1981); Bajscy and Broit (1982); Bajcsy and Kovačič (1989) as described in the next section), some authors (Gee et al. (1993); Ferrant et al. (2002); Rexilius et al. (2001)) represent the deformations by the position of the nodes of a tetrahedrization of the image space, and provide an interpolation method for the rest of the image. This technique was extensively used for brain-shift detection (Ferrant et al. (2002); Rexilius et al. (2001); Clatz et al. (2003)). It also admits very efficient parallel implementations (Warfield et al. (2002); Sermesant et al. (2003)).

Another widely used model of transformation is based on splines. Some authors used B-splines (Rueckert et al. (1999); Dornier et al. (2003); Hufnagel et al. (2005)) in order to model relatively smooth deformations. Others (Evans et al. (1991); Rohr et al. (2001)) used interpolating thin-plate splines (Bookstein (1989)), which have a bending energy that can control the smoothness of the deformation. The main disadvantage of this kind of method is that the computation time tends to become very large when recovering high resolution deformations. One potential progress may come from the use of Non-Uniform Rational B-Splines (Piegl (1991)), which allow for unevenly spaced control points. This may enable a multi-resolution modeling of the transformation, able to recover fine deformations in certain areas (by using many control points), and smoother (and easy to compute) ones in others (by using less control points).

Rohde et al. (2003) proposed a multiresolution transformation based on Radial Basis Functions. They select, on a multi-resolution grid, the points that best optimize the normalized mutual information in order to drive a radial basis transformation model. Thanks to the local support of the radial basis functions, the similarity criterion is only computed for a relatively small number of voxels. A similar threshold on the gradient of the cost function has previously been used by Collins and Evans (1997). The multi-resolution framework allows a spatially adaptive description of the transformation. Its effective precision is limited to the final grid resolution (smaller details cannot be modeled), which has to remain rather coarse in order to achieve reasonable computation times (reported by Rohde et al. (2003) to be of up to 3 hours for typical images and a final grid of size 17x17x15). However, as the authors predict, this time may be significantly decreased by a parallel implementation.

Locally rigid or affine transformation have been used as a mean to reduce the number of degrees of freedom. Hellier et al. (2001) use a multi-resolution approach in which they divide the image space into "uniform resolution" cubes. Inside each cube, an affine transformation is used. Pitiot et al. (2003) also estimate locally affine transformations. Each voxel is warped by one of several affine transformations, and a clustering algorithm is used to decide which affine transformation drives each voxel. Arsigny et al. (2003) propose a locally rigid registration algorithm in which the displacement is obtained by integrating a weighted combination of rigid or affine transformations.

Displacement fields

Displacement fields lie at the extreme side of the transformation spectrum. For each point in one image, the displacement field contains a (3D) vector that specifies its relative position (displacement) after the motion. However, the number of degrees of freedom provided by displacement fields is too large for most applications. We

would like the recovered displacement to be continuous and smooth, much like in the case of real physical deformations. This is necessary if we want an anatomically meaningful transformation: the position of different tissues is not inverted, and there are neither tissue cuts nor tissue contractions. Therefore, the displacement field is generally "regularized" in order to enforce its smoothness.

The first registration algorithms using displacement fields (Broit (1981), Bajscy and Broit (1982)) simulated real world elastic deformations. They were later extended to a multiresolution framework by Bajcsy and Kovačič (1989). The regularization may also be done by convolving the displacement with a Gaussian (Thirion (1998)), or by applying more complex filters (Bro-Nielsen and Gramkow (1996); Hermosillo et al. (2002); Tchumperle (2002)).

The main drawback of the model is the large memory space required by the transformation. However, it accumulates two advantages that make it extremely well suited for inter-subject registration:

1. It can represent a nonlinear transformation with an arbitrarily large number of degrees of freedom (1 displacement vector per image voxel). Thus, it can potentially best capture the fine differences that exist between two brains. As we will see in Chapters 3 and 4, a displacement field can very well represent transformations that are smooth in some areas, and present fine details in others.

2. Algorithms controlling the local level of regularity of the deformation are computationally very efficient (Chapter 3). Furthermore, these algorithms can be easily implemented on parallel computers, which leads to low computation times (Chapter 5).

The above reasons led us to use a dense displacement field in order to represent deformations. Section 2.2 reviews some of the currently existing registration algorithms based on dense displacement fields.

2.1.3 Registration of images containing pathologies

Up to know, we have assumed that there was an anatomically meaningful correspondence between all the voxels of the two images. When registering patient images containing brain tumors with an anatomical atlas, voxels located inside the tumor have no correspondence in the atlas (which is generally an image coming from a healthy subject). Numerous methods and tools have been already devised to address non-rigid registration (Maintz and Viergever (1998)), but much fewer deal with this kind of pathological abnormalities.

Kyriacou et al. (1999) used a biomechanical modeling of the brain and tumor based on non-linear elasticity. In the case of multi-subject patient/atlas registration, elastic models are of low relevancy, since the transformation to recover does not correspond to a physical deformation. Furthermore, Christensen et al. (1996) showed that a fluid component is needed in order to recover large deformations, such as the ones implied by multi-subject registration. Therefore, algorithms based entirely on elastic deformations may not be able to recover the differences between two different subjects.

Dawant et al. (2002) proposed to artificially introduce the pathology in the atlas. A first non-rigid registration between patient and atlas yields an initial deformation that is used to implant a "pathology seed" inside the atlas. This deformation is then refined by non-rigidly registering the subject image with the seeded atlas. Later, BachCuadra et al. (2004) added a tumor growth model to the algorithm. In this framework, the main problem is the first registration, which can easily fail, especially if the pathology is located closely to the brain border or the ventricles.

Recently, Liu et al. (2004) modified the HAMMER algorithm (Shen and Davatzikos (2002)) in order to use statistical interpolation in "non-corresponding regions" (supposed to be inside the tumor). This enables the algorithm to register tumor-diseased images with an atlas. Such an analysis may presumably be unreliable for multisubject registration, where it is hard to distinguish between regions that "do not correspond" because they are pathological, and other regions where the anatomy in the two images is just a little different.

2.1.4 Our approach

In summary, in our opinion the algorithm requirements are the following:

1. In order to avoid tedious pre-segmentation steps to which the algorithm may potentially be very sensible, the similarity of the two images should be estimated by directly using the voxel intensities as features. Therefore, the algorithm will constantly try to **"pair"** each voxel from one image with some voxel from the other.

2. A dense displacement field guarantees that the transformation can be potentially represented at it maximal level of detail all over the image space. This field undergoes a regularization (**"smoothing"**) step which controls the local level of detail/regularity. This choice provides a flexible transformation model, while still giving the ability to estimate it in a multiresolution manner. As described in the following chapters, the algorithms allowing to locally control the level of resolution are very fast. Furthermore, since a

displacement is explicitly described for each voxel, this approach may potentially enable the introduction of geometric information at a low cost (Cachier et al. (2001)).

Following a classification of registration algorithm presented by Cachier (2002), we call this class of algorithms "pair-and-smooth".

2.2 The pair-and-smooth approach

Numerous pair-and-smooth algorithms have been proposed in the last decade, and the technique is relatively mature. This section performs a brief synthesis of the most important algorithms that will serve as a basis for the developments proposed in this thesis.

2.2.1 A summary of existing pair-and-smooth methods

In order to ensure the invertibility of the recovered transformation, Christensen et al. (1996) proposed a nonlinear registration algorithm based on a physical fluid dynamics model of deformation (Christensen et al. (1994a,b)). The algorithm uses three displacement fields, which are all initialized to 0: the final displacement U, and two intermediate ones (u and v). The displacement is estimated through a four-step iterative process:

1. Perform a gradient descent step on the SSD similarity criterion between I and $J \circ U$ in order to recover a small correction field u.
2. Compute the velocity field v using the Navier-Stokes equation[1] with u as the external force
$$\mu \Delta v + (\lambda + \mu) \nabla (\nabla \cdot v) + u = 0$$
3. Compute the associated perturbation R in the Eulerian framework (where τ is the time step, and ∇U is the Jacobian matrix field of U).
$$R = \tau (v + \nabla U^T \, v)$$
4. Compute the Jacobian of the perturbed field $U + R$. If the Jacobian is close to being negative in some places, then the updated field $U + R$ is in danger of not being invertible. If so, update the displacement field through composition as $U \leftarrow U \circ (\tau v)$ and reinitialize v to 0. Otherwise, update it through explicit Euler integration: $U \leftarrow U + R$.

[1] The model has been simplified by neglecting the pressure gradient and the inertial terms.

5. Go to step 1.

The procedure which ensures that the recovered transformation is a diffeomorphism is called "regridding": the time integration is performed in the Eulerian framework; and if the Jacobian of the transformation is in danger of becoming negative, the non-invertibility is avoided by using composition rather than explicit Euler integration. Although the invertibility of the transformation is ensured, the model has two drawbacks. First, although the recovered displacement field is a diffeomorphism, the purely fluid model of deformation (allowing very large deformations) may locally bring the transformation very close to singularity: a Jacobian that, although positive, is very close to 0 (a large volume contraction) or very large (a large volume expansion). This amounts to huge volume contractions or expansions. Second, there are two reasons for which the algorithm is extremely slow:

1. The Navier-Stokes equation is solved through successive overrelaxation (Christensen et al. (1994a)), which computes an accurately fluid model at the expense of a large computation time.
2. The method requires at each iteration the computation of the Jacobian matrix of the displacement field, which is computationally very expensive.

Hence, Christensen et al. (1994a) report computation times that are as large as 2 hours on a 128×128 MasPar computer and 1.4 days for a MIPS R8000 processor for a couple of $128 \times 128 \times 100$ images. Bro-Nielsen and Gramkow (1996) propose a faster version of the algorithm by implementing the fluid model as a convolution filter rather than by successive overrelaxation. This has reportedly decreased the computation time by an order of magnitude.

Lester et al. (1999) extended the fluid registration algorithm proposed by Christensen et al. (1994a) to variable viscosity. They model the image space as a inhomogeneous fluid, whose viscosity varies spatially. The method uses the successive over-relaxation method to solve the model, which leads to a computation time penalty. However, as we will see below, being able to model inhomogeneous deformations is particularly important in multi-subject registration.

Another way to guarantee the invertibility of the transformation is to simultaneously compute the transformation and its inverse. This method, developped by Christensen (1999) (see also Christensen and Johnson (2001) and Johnson and Christensen (2002)) solves a problem commonly encountered in nonrigid registration: the transformation that deforms the image I into the image J is not the inverse of the transformation that deforms the image J into the image I. In practice, this algorithm requires large computation times (e.g. one hour for 256×320 2D images), and is therefore incompatible with our clinical requirements.

The *demons' algorithm*, proposed by Thirion (1996b), has the advantage of a much lower computation time, while being an approximation of the method proposed by Christensen et al. (1994a), as shown by Bro-Nielsen and Gramkow (1996). This algorithm has been relatively widely used so far (Bricault et al. (1998); Webb et al. (1999); Prima et al. (1998); Dawant et al. (2002); BachCuadra et al. (2004)). It optimizes the similarity criterion through a four step alternated minimization:

1. A correction field u is estimated using a renormalized version of the optical flow
$$u = \frac{I - J \circ U}{\|\nabla I\|^2 + \alpha(I - J \circ U)^2} \nabla I$$

2. The correction field u is regularized using a convolution with a Gaussian.

3. The current displacement field is composed with the correction field $U \leftarrow u \circ U$.

This approach eliminates the two performance bottlenecks of Christensen's fluid model: the computation of the Jacobian and the resolution of the Navier-Stokes equation. Furthermore, each iteration of the algorithm is computed in a linear time with respect to the image size, which was not the case before. However, as shown, the model does not present any mathematical guarantees of convergence towards a minimum: the estimation of the correction field decreases the regularity, and the Gaussian filter decreases the similarity. Therefore, Pennec et al. (1999) and later Modersitzki (2004) put the algorithm in a variational framework and showed that it minimizes a global energy. Although different, their methods propose to solve the registration problem by minimizing energy functions close to the one below:

$$E = \underbrace{SSD(I, J \circ U)}_{\text{similarity}} + \gamma \underbrace{\int_\Omega \|\nabla U\|^2}_{\text{regularity}} \tag{2.1}$$

under Neumann boundary conditions[2]. In this formulation, the optical flow has been replaced by a normalized gradient descent on the SSD similarity criterion, whereas the regularity is enforced by decreasing the gradient of the deformation field. The similarity term (pushing towards better similarity) and the regularity term (pushing towards a more regular transformation) are linked through a weight γ.

[2]The Neumann boundary conditions specify that, on the domain border, the gradient of U is perpendicular on the normal to the border ($\nabla U^T \cdot \vec{n}$, where $\vec{n} \perp \partial\Omega$).

Cachier et al. (2003) introduced auxiliary variables (Cohen (1996)) to formalize the alternate minimization of the demons' algorithm, while preserving its computational efficiency. The energy in Equation 2.1 is reformulated as

$$E = \underbrace{SSD(I, J \circ C)}_{\text{similarity}} + \sigma \underbrace{\int_\Omega \|C - U\|^2}_{\text{link}} + \gamma \underbrace{\int_\Omega \|\nabla U\|^2}_{\text{regularity}}$$

As in the previous algorithm, one can remark the presence of similarity and regularity terms. However, the presence of the middle link term separates the two variable to optimize (C and U) and guarantees that the total energy diminishes at each gradient descent step. This link term $\int \|C - U\|^2$ enforces the resemblance between the auxiliary variable C and the displacement field U. The optimization is performed in two steps. The first part of the criterion ($SSD(I, J \circ C) + \sigma \int \|C - U\|^2$) is minimized through gradient descent with respect to C (U being fixed). Then, the second part ($\sigma \int \|C - U\|^2 + \gamma \int \|\nabla U\|^2$) is minimized with respect to U (C being fixed). This is done by convolution with a Gaussian.

In order to deal with inhomogeneous transformations, Hermosillo et al. (2002) regularize the transformation using anisotropic diffusion, based on the local image intensities as done by Nagel and Enkelmann (1986) (also see Alvarez et al. (2000)). They also derive some multi-modal similarity criteria (mutual information, correlation coefficient and correlation ratio), in order to optimize them through a fast, gradient-based technique. Chefd'hotel et al. (2002) proposed a similar algorithm, but using a Gaussian regularization, and showed that it preserves the diffeomorphism property on the recovered transformation. A pair and smooth fluid registration algorithm based on the mutual information and using the convolution with a Gaussian for regularization has also been proposed by D'Agostino et al. (2003).

2.2.2 A simple pair and smooth algorithm

This section synthesizes the prototype of a pair-and-smooth registration algorithm, and provides the basis of the method proposed in this manuscript. In order to clarify the description of our algorithm, our presentation will follow the derivation of the demons algorithm of Thirion (1998) as presented in Pennec et al. (1999). The description shows its main weaknesses that we will address in this thesis.

The demons algorithm alternates the maximization of the similarity and the regularity criteria. The similarity is maximized through a gradient descent that we describe below, while the regularity is enforced by convolving the deformation field with a stationary Gaussian. The algorithm can use different similarity criteria, such as the Smallest Squared Distance (SSD) (Pennec et al. (1999)) or the Local Correlation Coefficient (Cachier and Pennec (2000)). Since the SSD criterion is

commonly accepted as a reliable method for monomodal registration, we will use it as an example throughout this chapter. As we already saw, more powerful similarity metrics for multi-modal registration are described in (Roche et al. (1998, 2000); Cachier and Pennec (2000); Guimond et al. (2001b); Hermosillo et al. (2002)).

The gradient descent step can be described as follows: given the current value of the deformation U, the goal is to find a *small additive correction* u that minimizes the chosen similarity criterion. A first order Taylor expansion leads to:

$$SSD(I, J \circ (U+u)) = \int \left[I(p) - (J \circ (U+u))(p)\right]^2 dp \tag{2.2}$$

As by definition $\int f(p)^\top \, u(p) \, dp$ is the dot product of the vector functions f and u, we get by identification:

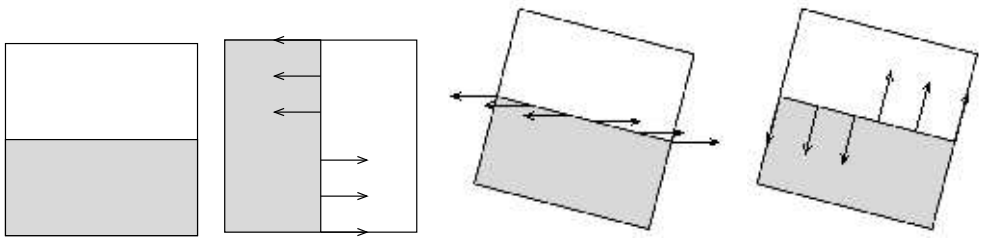
$$\nabla SSD = 2\left[(J \circ U)(p) - I(p)\right] \, \left[(\nabla J) \circ U\right](p)$$

From the Taylor expansion, we can see that the criterion is minimized if we update the current displacement field U^n at iteration n by adding a small fraction ϵ of the gradient $u^n = -\epsilon \cdot \nabla SSD$ to obtain $U^{n+1} = U^n + u^n$. This first order gradient descent is usually the evolution equation used in PDE approaches (Hermosillo et al. (2002)). The demons algorithm corresponds to a slightly more complex second order gradient descent scheme where the gradient is renormalized using an approximation of the second order derivative of the SSD criterion (Pennec et al. (1999)).

This method is efficient, but it suffers from a major drawback: The additive correction scheme presented above does not re-compute the gradient of the source image at each optimization step, but rather resamples it. In Section 2.2.3, we argue that this additive scheme becomes invalid with large local rotations, and it may lead to a non-invertible transformation. Therefore, we propose a "compositive scheme" that tackles all types of displacements and, furthermore, ensures the invertibility of the recovered transformation.

2.2.3 Recovering large deformations

The additive formulation of the Taylor expansion of the SSD leads to a gradient descent proportional to the resampled gradient of the source image ∇J, without changing its direction. In real cases, where the deformation contains large local rotations, the direction of the gradient of the criterion will slowly become parallel to the contours of the resampled image $J \circ U$, thereby diminishing its efficiency at each iteration. Figure 2.1 presents an example of a 90° rotation of a simple image. At the beginning of the registration, both schemes evolve in similar manners (Fig. 2.1b). However, as the rotation is gradually recovered, the correction field yielded



(a) Target image I.

(b) Source image J.

(c) Resampled source image $J \circ U$ for an intermediate correction field (additive scheme).

(d) Resampled source image $J \circ U$ for an intermediate correction field (compositive scheme).

Figure 2.1: Recovering a 90° rotation of a simple 2D image (Fig. a) using the SSD criterion and two correction schemes (additive and compositive). The arrows indicate the correction field at a certain iteration. At the beginning, the two schemes exhibit similar behaviors (Fig. b). However, as the rotation is being estimated, the additive scheme yields corrections which tend to be parallel to the contour lines, thereby lowering its efficiency (Fig. c). By updating the direction of the correction, the compositive scheme maintains its efficiency to recover rotations (Fig. d).

by the additive scheme (Fig. 2.1c) gets increasingly parallel to the contour lines (i.e. perpendicular to the gradient of the image). As the motion perpendicular to the gradient of the image is not detectable (aperture problem), the corresponding correction field component does not provide useful information, and the efficiency of the correction decreases (Fig. 2.1c).

Remark A good quantifier of the efficiency of the correction field is the cosine of the angle between the deformation force and the local image gradient: if this angle is 0 (i.e. they are parallel), the force is fully efficient; the force diminishes as this angle increases.

Following Trouvé (1998), Miller and Younes (2001) and Chefd'hotel et al. (2002), we replace the addition of the correction and displacement fields $U^{n+1} = U^n + u^n$ by the composition of the corresponding transformations. If Id is the identity transformation, this corresponds to

$$Id + U^{n+1} = (Id + U^n) \circ (Id + u^n) = Id + U^n \circ (Id + u^n) + u^n.$$

Thus, denoting by $(U \circ u)(p) = U(p + u(p)) + u(p)$ the result of the transformation composition on the displacement fields, we end up with the update equation

$U^{n+1} = U^n \circ u^n$. One can remark that a first order Taylor expansion of $(U \circ u)$ yields:

$$(U \circ u)(p) = U(p+u(p)) + u(p) = U(p) + \nabla U(p)^T u(p) + u(p) + O(\|u\|^2) \quad (2.3)$$

Therefore, the two schemes are equivalent if the displacement field is locally constant ($\nabla U \approx 0$), which corresponds to a (local) translation.

The goal is now to find the correction field u that minimizes $SSD(I, J \circ (U \circ u))$. Taking $J' = J \circ U$, we came back to the additive formulation with $U' = 0$ since $(0 \circ u) = u$. Therefore, we have:

$$SSD(I, J \circ (U \circ u)) = SSD(I, J' \circ (0 + u))) \quad (2.4)$$

Thus, the major difference with the additive scheme (Eq. 2.2) lies in that we now take the gradient of the resampled image $\nabla(J \circ U)$ instead of the resampled gradient of the original image $(\nabla J) \circ U$. To summarize, the usual additive scheme consists in computing the gradient of the SSD at the current displacement field U^n and then to update the displacement using a fraction of this gradient:

$$U^{n+1} = U^n - \epsilon.2\,[J \circ U^n - I]\ [(\nabla J) \circ U^n] \quad (2.5)$$

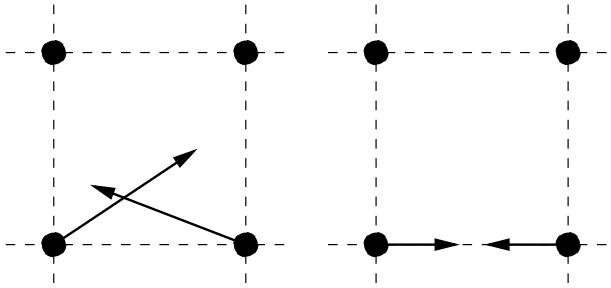
The equivalent compositive scheme is

$$U^{n+1} = U^n \circ (-\epsilon.2\,[J \circ U^n - I]\ [\nabla(J \circ U^n)]) \quad (2.6)$$

Remark The fraction ϵ of the gradient that is taken at each optimization step is an algorithm parameter. Section 3.2.1 will show a method to automatically tune it.

If we go back to our square example, one can notice in Figure 2.1d the effect of the gradient of the deformed source image being updated with the compositive scheme: at each iteration, the correction is perpendicular to the contour lines, which increases the convergence rate. This simple example translates into the general case if we consider the two small images as being merely local details of larger ones. Large local rotations, such as the one presented here, occur rather frequently, especially in multi-subject registration, and they are difficult to recover by considering an additive correction field.

Guaranteeing invertibility The compositive scheme has an additional advantage. If the correction u is a homeomorphism at each step, then the total displacement U is also a homeomorphism. What is interesting is that this property still holds in practice with the discrete composition of displacement fields using trilinear interpolation. Moreover, guaranteeing that u is invertible is fairly easy: it suffices to limit its norm to half of the voxel size. Figure 2.2 presents four voxels



(a) At each iteration, too large correction vectors may yield an uninvertible transformation.

(b) By limiting the correction vectors at each iteration, the recovered transformation is invertible.

Figure 2.2: Guaranteeing the invertibility of the displacement field.

of a two-dimensional image on a regular grid, and the displacement vectors of the two lower voxels. If the norms of these vectors are large, they can potentially cross and cause foldings in the resampled image (Fig. 2.2a). Invertibility of the displacement field can be easily ensured by limiting the norm of these vectors to half of the voxels size (Fig. 2.2b). This way, even in the worst case scenario, vectors cannot cross and the displacement field is always invertible. This avoids the computation of the Jacobian at each iteration, as it is done by Christensen et al. (1996).

Compositive scheme vs. regridding: execution time comparison This section illustrated the compositive scheme with the SSD criterion, but this may be easily generalized to other similarity criteria. One may also compare this scheme with the *regridding* method used by Christensen et al. (1996), and described in Section 2.2.1. Table 2.1 shows a comparative analysis of the operations required by each of the three schemes. Assuming a computation time similar for a scalar filtering and for a scalar resampling (we basically have to sweep the memory for the image values), we can see that the compositive scheme is only 25% slower than the additive one, and it takes one third of the time required by a regridding scheme. However, its accuracy is equivalent to the one of the regridding.

2.3 Discussion

This chapter presented possible existing solutions to the atlas to patient registration problem in view of the clinical requirements presented in Section 1.1. Both

Scheme	Scalar filtering	Scalar resampling
Additive	0	3
Compositive	3	1
Regridding	9	3

Table 2.1: Comparative analysis of the computational complexity of additive, compositive and regridding schemes. The compositive scheme is three times faster than the regridding, while the accuracy is similar.

applications require a nonlinear registration algorithm that has to fulfill several conditions.

First, the algorithm has to register a subject image with an atlas. In order to simplify the problem, the atlas image was chosen to be of the same modality as the patient ones. Therefore, we found existing similarity criteria, such as the SSD or the correlation coefficient to be very efficient. Furthermore, even with a dense displacement field, an invertibility guarantee can be achieved rather easily, with a low computation time. However, the registration is multisubject, which means that we will have to address some other issues:

- Sometimes, we know that deformations are highly inhomogeneous in certain areas, and exhibit a large spatial coherence in others. In Chapter 3, we argue that the regularization with a uniform Gaussian is not adapted to recover such deformations, and that elastic models are computationally expensive to solve. Thus, we propose a faster regularization method that is compatible with inhomogeneous deformations and has low computation times.
- The above algorithm, like many others, takes all voxels into account in an equal manner. Based on the observation that some areas in images contribute more relevant information than others, we describe in Chapter 3 a method that weights the local influence of the correction field in the registration based on a priori information about the local reliability of the similarity criterion at each voxel. This method can be adapted to deal with images of patients with a brain cancer that contain large, contrasted lesions which have no correspondence in the atlas. Therefore, they tend to misguide the registration algorithm. In Chapter 3 we propose a method to ignore lesions, and only concentrate on reliable information.

Second, in certain cases, we are only interested in the surfaces that delimit structures of interest. We want these surfaces to remain regular after deformation, and give the rest of the image the freedom to deform. In Chapter 4, we present an anisotropic regularization method that enforces the regularity of specified surfaces

during the deformation process. The presented anisotropic constraints can be potentially extended to other type of structures, such as white matter fiber bundles or sulci.

Last, the algorithm should take a low execution time, in order to be useful to physicians. Since the problem is complex, and we do not want to take the risk of lowering the computation time at the expense of the results' quality, Chapters 5 and 6 propose a parallel implementation on a low cost parallel machine.

Chapter 3

Fast fluid registration with local elastic constraints

This chapter provides some solutions for the problems problems described in Chapter 2, arising in multisubject registration. First, we modify the demons registration energy (Eq. 2.1) in order to account for tissues with space-varying deformability, and we describe an efficient resolution method. Then, we tackle the problem of the relative importance of the information in various regions in the images. In order to achieve reasonable computation times, we describe a fast numerical scheme that allows to achieve both goals. We present the results of the method on the two applications described in Chapter 2: the registration of Parkinsonian and tumor-diseased brain images with an atlas.

3.1 Method

3.1.1 Non-stationary "elastic" regularization based on anatomy

Real deformations are often highly inhomogeneous. When regularizing with a stationary Gaussian filtering, the amount of regularization is given by the standard deviation of the filter. This value is either too large, preventing the retrieval of fine details, or too small, which yields a noisy deformation field in smooth areas. The solution we propose is to use a priori information about the variability of the different structures in the images to locally adapt the level of regularization.

A modification of the energy function for non-stationary regularization

A model that allows us to specify the local degree of regularization is elasticity. However, the model is complex and solving it can be computationally intensive. Furthermore, in the multi-subject case, an elastic constraint on the deformation that links the objects in the two images has no physical justification.

We have seen in the previous chapter (Section 2.2) that the demons algorithm corresponds to minimizing the following energy (where U_α is the displacement along the direction $\alpha \in x, y, z$):

$$E = \underbrace{SSD(I, J \circ U)}_{\text{similarity term}} + \underbrace{\gamma \sum_{\alpha \in \{x,y,z\}} \int \|\nabla U_\alpha\|^2}_{\text{regularity term}} \tag{3.1}$$

where SSD is the similarity criterion, and U is the displacement field. The second (regularity) term tends to keep local variations of the displacement low, which imposes a strong correlation of the displacement of neighbor voxels. Optimizing the regularity term amounts to convolving the displacement field with a Gaussian. We modified this criterion in order to weight the local spatial correlation (regularization) of the displacement more in some places than in others. If $d(x, y, z)$ is a spatial field that measures the local importance for the displacements of neighbor voxels to be correlated, then the energy to minimize becomes

$$E = SSD(I, J \circ U) + \gamma \sum_{\alpha \in \{x,y,z\}} \int d(x, y, z) \ \|\nabla U_\alpha\|^2 \tag{3.2}$$

Minimizing the regularity criterion

The new regularity criterion for some direction α has the formulation

$$Reg(U_\alpha) = \int_\Omega d \ \langle \nabla U_\alpha, \nabla U_\alpha \rangle$$

By expanding $Reg(U_\alpha + h)$, we get

$$\begin{aligned} Reg(U_\alpha + h) &= \int_\Omega d \ \langle \nabla U_\alpha + \nabla h, \nabla U_\alpha + \nabla h \rangle \\ &= Reg(U_\alpha) + 2 \int_\Omega \langle d \, \nabla U_\alpha, \nabla h \rangle + \mathcal{O}((\nabla h)^2) \end{aligned}$$

To compute the gradient of the regularity criterion, we need to relate $\int_\Omega \langle d\,\nabla U_\alpha, \nabla h\rangle$ to a term involving only h and not its gradient (namely $-\int_\Omega h\,\mathrm{div}(d\,\nabla U_\alpha)$). To achieve that, let us apply the Gauss-Ostrogradsky divergence theorem:

$$\begin{aligned}\int_{\partial\Omega} \langle h\ (d\,\nabla U_\alpha)\,, \vec{n}\rangle &= \int_\Omega \mathrm{div}\,(h\ (d\,\nabla U_\alpha)) \\ &= \int_\Omega \langle \nabla h, d\,\nabla U_\alpha\rangle + \int_\Omega h\,\mathrm{div}\,(d\,\nabla U_\alpha)\end{aligned}$$

where $\partial\Omega$ is the domain boundary, and $\vec{n}$ is its normal. The last term was obtained by the standard differentiation of a product: $\mathrm{div}(h\,\vec{v}) = \langle \nabla h, \vec{v}\rangle + h\,\mathrm{div}(\vec{v})$. By assuming the Neumann boundary conditions ($\langle \nabla U_\alpha, \vec{n}\rangle = 0$ on the border $\partial\Omega$ of Ω), the left part of the above expression becomes null (hd is a scalar expression that can be taken out of the dot product). Thus, we are left with the sought result:

$$\int_\Omega \langle d\,\nabla U_\alpha, \nabla h\rangle = -\int_\Omega h\,\mathrm{div}\,(d\,\nabla U_\alpha)$$

This means that

$$Reg(U_\alpha + h) - Reg(U_\alpha) = -2\int_\Omega h\,\mathrm{div}(d\,\nabla U_\alpha) + \mathcal{O}(\nabla h^2)$$

By definition, the gradient field ∇Reg exists if we have for any perturbation h the equality

$$Reg(U_\alpha + h) = Reg(U_\alpha) + \int_\Omega h\,\nabla Reg + \mathcal{O}(h^2)$$

By identification, we get

$$\nabla Reg = -2\,\mathrm{div}(d\,\nabla U_\alpha)$$

Therefore, a gradient descent on the regularity criterion $\frac{\partial U_\alpha}{\partial t}(p) = -\nabla Reg(p)$ leads to the *nonstationary heat equation*

$$\frac{\partial U_\alpha}{\partial t}(p) = -\nabla Reg(p) = \mathrm{div}(d\,\nabla U_\alpha)(p) \tag{3.3}$$

Discussion

The scalar field d gives the *cost of irregularity*. If the diffusion field is constant, Equation 3.3 describes a linear diffusion filter, which can be implemented using a convolution with a stationary Gaussian. The larger d is, the more important is the diffusion. By analogy with mechanical deformations, we can call d a *stiffness field*:

to fit our purpose of tolerating large deformations in certain areas and small deformations in others, it suffices to give d large values in areas where we expect little deformations and small values in areas that may contain large ones. In this case, d also encodes the anatomical knowledge that we have on the deformability of the tissues. Currently, d is estimated by using region-based segmentation algorithms. Since d expresses the local stiffness of the target image I, we only need to segment this image to compute it. However, unlike feature-based registration that needs an accurate segmentation of the structures to match, our algorithm only needs a fuzzy segmentation. In Section 3.2.1 we will show a method to automatically estimate the d field for T1 MRI images of the brain.

The proposed regularization method has a double heuristic motivation. One one hand, if two points belong to a region that does not changes from one image to another, their displacements are more correlated than if they belong to a region that changes a lot. On the other hand, the closer the two points are to each other, the more their displacements are correlated. Therefore, we use a local regularization: If a region exhibits a low variation between two different images, then we want it too deform less and behave in a stiffer manner. On the contrary, highly variable regions should be able to deform more, and therefore be assigned a lower stiffness. The choice of a local Gaussian smoothing has a triple motivation:

1. Smoothing by a Gaussian (which is equivalent to our diffusion equation with a locally constant stiffness) is not far from the simulation of an elastic displacement and it can model it with a reasonable approximation (Bro-Nielsen (1996)).

2. The strength of the regularity constraint is described by the local value of the stiffness field d. By tuning this, we can regularize inhomogeneous deformations that alternate smooth and highly varying regions.

3. As we will see below, solving a diffusion equation is fast and easily implementable on a parallel system.

3.1.2 Confidence-based weighting of the correction field

The images we register are inevitably noisy. This noise has a strong impact on the gradient of the similarity criterion, especially in intensity uniform areas where the local signal to noise ratio is low. When regularizing the displacement, these spurious values tend to weaken the well defined displacement "forces" located at edges. The goal of this section is to filter out these unreliable values from the raw correction field. In this section we assume that the noise that affects the correction field follows a Gaussian model.

We use a method inspired by the image-guided anisotropic diffusion (Weickert (1997)): once the correction field u is computed, its components u_α along each axis are filtered using a nonstationary diffusion PDE:

$$\frac{\partial u_\alpha}{\partial t}(p) = \text{div}\left[(1 - k(p))\nabla u_\alpha(p)\right], \text{where } k(p) \in [0, 1] \tag{3.4}$$

Like in the previous section, if k is a spatially constant field, the above equation amounts to a Gaussian diffusion, which was previously used in similar conditions (Cachier et al. (2003) and D'Agostino et al. (2003)). If k varies spatially, it measures the local degree of smoothing applied to u_α. For $k(p) = 1$, the local displacement $u_\alpha(p)$ will be locally unaffected by the above partial differential equation. In case of smaller values for $k(p)$, the correction is locally smoothed. This feature is particularly well adapted to our problem. In areas where the signal to noise level is low, we smooth the correction field, thereby attenuating the effects of noise. The k field measures the *local confidence* in the similarity criterion.

Points that are reliable landmarks in the source image are those where the neighborhood has a characteristic pattern that cannot be produced by noise. The confidence can therefore be taken as a measure of the local intensity variability in the source image, such as the local variance or gradient. This type of measure is static, since it only has to be computed once, at the beginning of the algorithm. The expression we used is derived from Weickert (2000) in the case of non-stationary image diffusion:

$$k(p) = \exp\left(\frac{-c}{\left(\frac{\|\nabla J(p)\|}{\lambda}\right)^4}\right) \tag{3.5}$$

The confidence described in the above equations is close to 1 for large image gradients, and to 0 in uniform areas. λ is a contrast parameter that discriminates low contrast regions (which are mainly diffused) from high contrast ones (which preserve the edges in the deformation field), and c is a scalar parameter usually taken around 3.3 (see Weickert (2000)). We show in Section 3.2.1 how these parameters are tuned.

The confidence field k can also be used to encode some anatomical knowledge: in certain applications, especially in difficult multi-subject cases, one might choose to deliberately ignore certain aspects of the images, which could make the registration fail (e.g. tumors, marker-enhanced regions). In such applications one can impose the confidence to be null in areas that are considered irrelevant for the registration.

This formulation of the confidence can be understood as being a type of *soft feature-based registration*. Indeed, depending on the method used to estimate correspondences, registration algorithms are classified into two main categories. Intensity-based algorithms treat images as sets of voxels characterized by their intensity

which contribute in equal measure to the final transformation. Feature-based registration consists in estimating correspondences by matching geometric structures in the images, such as surfaces or lines. Our method is somewhere in-between. Voxels do not have the same weights in the computation of the correspondences. The ones which are on the edges of significant structures have a larger weight than the others. By using a kind of fuzzy features extraction, our algorithm is able to take into account the structures visible in the images, without needing generally error-prone binary segmentations.

Similar approaches have been used in image registration. Ourselin et al. (2000) estimate the correspondences in the case of rigid and affine registration using block-matching. Only the blocks in the source image that have a variance larger than a certain threshold are taken into account. The Nagel-Enkelmann operator (Nagel and Enkelmann (1986)) used in Alvarez et al. (2000) and Hermosillo et al. (2002) also uses a measure of the local variation of the source image to weight an elastic-type matching.

3.1.3 Fluid vs. elastic regularization: a viscoelastic model

Our algorithm has two levels of regularization. The first one acts on the correction field, which can be seen as the velocity field. Thus, its regularization can be interpreted as a non-stationary *viscous fluid* constraint. On the contrary, the regularization of the displacement field can be understood as imposing a kind of *elastic* behavior. Consequently, the combination of the two regularizations ends up in a sort of *viscoelastic* movement. Our terminology is based on the following classification of rheological behaviors of all materials:

- Viscous materials: in a purely viscous material all energy added is dissipated into heat.
- Elastic materials: in a purely elastic material all energy added is stored in the material.
- Viscoelastic materials: a viscoelastic material exhibits viscous as well as elastic behavior. Typical examples of viscoelastic materials are bread dough, polymer melts and artificial or natural gels.

Let us include our coherence-based regularization of the correction field (Eq. 3.4) in the global energy function of the regularization (Eq. 3.2). In this case, we consider the correction field u as being the temporal derivative of the displacement field U. Moreover, since we update at each step U by composing it with u, we

will see the correction field as the Eulerian derivative of the displacement $u = \frac{\partial}{\partial t}U(p,t)$. The updated energy function is

$$E = \underbrace{SSD(I, J \circ U)}_{\text{similarity}} + \underbrace{\beta \sum_{\alpha \in \{x,y,z\}} \int [1 - k(x,y,z)] \left\| \nabla \frac{\partial U_\alpha}{\partial t} \right\|^2}_{\text{fluid regularization}}$$
$$+ \underbrace{\gamma \sum_{\alpha \in \{x,y,z\}} \int d(x,y,z) \ \|\nabla U_\alpha\|^2}_{\text{elastic regularization}}$$

One may ask why this energy function has two level of regularization. In Christensen et al. (1996), it is argued that, in order to recover large deformations, a fluid model of deformation is necessary. If we do not take into account the elastic regularization, our model of deformation can be seen as a fluid one. However, the purely fluid model has a drawback: it does not preserve the anatomical coherence of images. Indeed, since there is no constraint on the displacement field itself, the model allows too large local volume expansions or contractions. Mathematically, this translates into values of the Jacobian of the transformation that are either very large or very close to 0. This poses a problem with partial volumes and non-corresponding structures (e.g. in brain imaging, partial cerebro-spinal fluid/white matter voxels being transformed into grey nuclei, or structures contracting and eventually vanishing). Furthermore, when image noise generates spurious gradients in otherwise uniform areas, strong artifacts appear in the deformation field inside these regions. These side-effects occur due to the unconstrained nature of the fluid model when optimizing the similarity (see Section 3.2.3).

The proposed method also bears some resemblance to the variable viscosity registration proposed by Lester et al. (1999). The main difference resides in the numerical resolution. Whereas Lester et al. (1999) follow a strict mechanical model and use a complex and time-consuming method such a the successive over-relaxation to solve it, we believe that in multi-subject registration mechanical simulation is not particularly adapted: we do not deal with real world deformations. Instead, we use a more time-efficient approach based on a simpler inhomogeneous regularization (non-stationary diffusion) that offers the same functionality (inhomogeneous deformations) and a low computation time (see Section 3.1.4).

Finally, we experimentally observed that a small amount of elastic regularization was sufficient to allow a huge decrease of the fluid regularization, leading to a much faster convergence of the algorithm.

3.1.4 Numerical implementation

As we have seen, both the confidence-based filtering (Equation 3.4) and the regularization (Equation 3.3) can be described using non-stationary diffusion PDE's on scalar fields v (the x, y and z components of U and u). In this section we tackle the problem of efficiently solving such equations in the isotropic case. A similar method has been used in Fischer and Modersitzki (1999) and Modersitzki (2004).

Explicit, implicit and semi-implicit schemes

The simplest way to solve a scalar diffusion equation such as

$$\frac{\partial v}{\partial t} = \text{div}(d\,\nabla v)$$

on a discrete image v is to compute the derivatives using finite differences and then reformulate the problem using a matrix vector multiplication. Considering the image v as a big one dimensional vector of size N containing successively all its voxels (e.g. $N = dim_x \times dim_y \times dim_z$ for a 3D image), the derivatives can be encoded into a big $N \times N$ matrix A, so that we get the *explicit scheme*:

$$\frac{v^{t+\Delta t} - v^t}{\Delta t} = A^t\, v^t.$$

where t is the time and Δt is the time step. The Laplacian matrix A^t depends on the time t if the diffusion field d is time varying.

For this explicit scheme, all the variables on the right side are known at time t, and the resolution is simply $v^{t+\Delta t} = v^t + \Delta t\, A^t\, v^t$. However, such an approach is very slow, since the time step has to be very small in order to avoid divergence (Weickert et al. (1998)). This drawback can be avoided by solving the *implicit scheme*, which contains on the right side only the variables at the time $t + \Delta t$:

$$\frac{v^{t+\Delta t} - v^t}{\Delta t} = A^{t+\Delta t}\, v^{t+\Delta t}$$

This scheme is guaranteed to be stable for all values of Δt. However, it is difficult to solve (since we do not know $A^{t+\Delta t}$), and therefore a *semi-implicit scheme* is often preferred:

$$\frac{v^{t+\Delta t} - v^t}{\Delta t} = A^t\, v^{t+\Delta t}$$

This amounts to solving the equation

$$v^{t+\Delta t} = (\mathbf{I_N} - \Delta t\, A^t)^{-1}\, v^t$$

Appendix A.1 shows a fast algorithm that solves the above linear system in linear time (w.r.t. to the image size). For more information on the discretization of the equation itself, the reader should look at Appendix A.2. Appendix A.3 presents the implementation of the Neumann and Dirichlet boundary conditions.

The semi-implicit scheme in 3D: AOS

In the three-dimensional case, a problem arises: the matrix to invert is no longer tridiagonal, which leads to a much higher computation time. In order to address this problem, Weickert et al. (1998) introduced the *Additive Operator Scheme* (AOS), which makes the resolution of the PDE separable, thereby simplifying the computations. If the filtering operator is separable, we can consider the Laplacian operator A to be the sum of its projections on the three axes $A = \sum_{\alpha \in \{x,y,z\}} A_\alpha$. Therefore

$$v^{t+\Delta t} = \left(\mathbf{I_N} - \Delta t \sum_{\alpha \in \{x,y,z\}} A_\alpha \right)^{-1} v^t$$

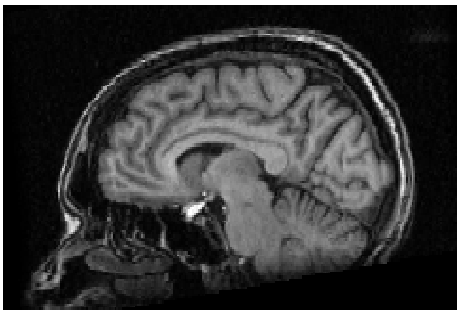
In the case of non-stationary diffusion, he then used the following approximation, justified by a Taylor expansion of both members:

$$v^{t+\Delta t} = \left(\mathbf{I_N} - \Delta t \sum_{\alpha \in \{x,y,z\}} A_\alpha \right)^{-1} v^t = \frac{1}{3} \sum_{\alpha \in \{x,y,z\}} (\mathbf{I_N} - 3\,\Delta t\, A_\alpha)^{-1}\, v^t + \mathcal{O}(\Delta t^2)$$

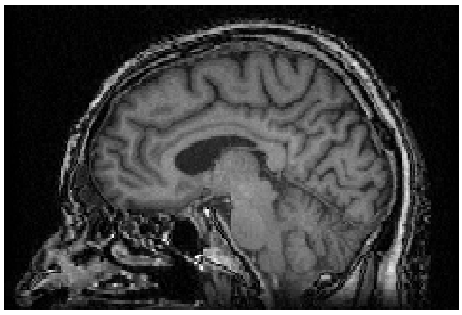
This reduces the 3D diffusion to three 1D ones, thus replacing the inversion of a non-tridiagonal matrix with three inversions of tridiagonal ones. In practice, we obtain in our iterative optimization algorithm an equivalent computational load for one AOS regularization step and for the computation of the gradient of the similarity criterion. Appendix A.4 describes in more detail the practical implementation of the Additive Operator Scheme.

3.2 The registration of brain MRI's from Parkinsonian patients

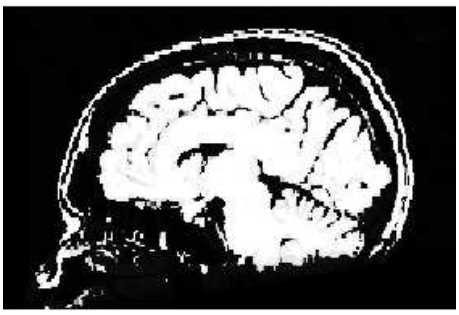
We tested our algorithm by registering 10 3D T1-weighted MRI images of Parkinsonian patients, such as the ones presented in Figure 3.1. They were all acquired using the IR-FSPGR (3D acquisition, Inversion Recovery, Fast Spoiled Gradient Echo) protocol and a field strength of 1.5T. These images were acquired preoperatively under stereotactic conditions, in order to select optimal targets for deep brain stimulation. All images have the same sizes $256 \times 256 \times 124$. In order to eliminate large displacements that do not reflect anatomical differences, image couples were affinely registered before the non-rigid registration.



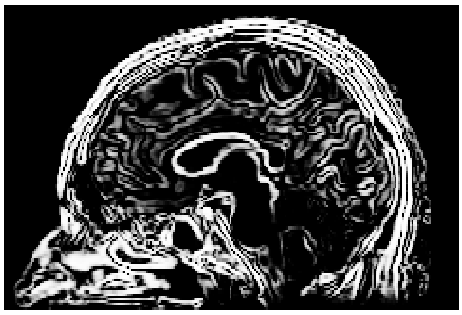
(a) Target image (sagittal view).



(b) Source image (sagittal view).



(c) Stiffness information.



(d) Confidence.

Figure 3.1: Registering two T1-MRI images of different subjects. The four images present the same sagittal slice of the target and source images, and the stiffness and confidence fields. The images are courtesy of Pr. D. Dormont (Neuro-radiology Dept., Pitié-Salpétrière Hospital, Paris, France).

3.2.1 Parameter tuning

Confidence field

We compute the confidence as a function of source image gradient, as described by Equation 3.5. The values of c and λ are parameters of the algorithm (the way to tune their values is detailed below in this section). Figure 3.1d presents a slice of the computed confidence field (k). Its values give the amount of smoothing of the incremental correction field. In places where these values are low, the correction field will be smoother (remark that the diffusion is weighted by $1 - k$), thereby making these regions count less in the registration. At each iteration, the confidence field is resampled into the deformed geometry.

Discussion The standard deviation of the gradient used to compute the confidence is the same as the standard deviation of the gradient used to compute the deformation force (1.0). This way, we preserve this force field in important areas on edges, and smooth it in areas where it could be affected by noise.

Stiffness field

In brain images, the shapes of structures like ventricles or gyri are highly varying. A common problem with non-rigid registration algorithms that use a uniform regularization is their inability to properly deform the ventricles. In our algorithm, the regularization allows the use of a higher level of regularization in certain areas than in others. For choosing the local level of regularization inside a structure, a good reference would be the relative variability of the structure (normalized by its size). Computing such a measure is a difficult problem. Our experience showed that a good choice is to use a level of regularization three times larger within the brain than in the fluid-dominated areas (inside the cerebro-spinal fluid and image background). Achieving a fuzzy segmentation of these areas for T1-MRI images of healthy subjects is rather straightforward, since a simple thresholding gives rather good results. However, we wanted a more general segmentation method, able to take into account other modalities, and also brains with pathologies. Thus, we considered classification algorithms.

In these experiments, we used the fuzzy k-means algorithm (Bezdek (1981); de-Gruijter and McBratney (1988)) to classify the images into five classes: image background, cerebro-spinal fluid (CSF), grey matter (GM), white matter (WM) and fat. If $P_{back}(p)$, $P_{csf}(p)$, $P_{gm}(p)$, $P_{wm}(p)$ and $P_{fat}(p)$ are the fuzzy memberships at a voxel p for respectively, the image background, CSF, grey matter, white

matter and fat classes, we compute the stiffness field (Fig. 3.1c) as:

$$d(p) = P_{gm}(p) + P_{wm}(p) + P_{fat}(p)$$

As an input, the classification algorithm needs initial estimates of the average values of the classes. These *protocol parameters* are easily specified by the user: thanks to the graphical interface we have developed, the user visualizes the images and interactively determines the initial intensity values for each tissue class. They are used as input parameters (five for each image to register) of the fuzzy k-means algorithm. The fuzziness index[1] was fixed to 2.

Time steps

The result of the registration depends on the similarity gradient descent fraction ϵ, the two diffusion (elastic and fluid) time steps, and the parameters c and λ from Equation 3.5. Manually tuning these parameters can be a tedious task, since regularization and similarity have different units. Our solution is to provide a normalization of the intensities before registration, as follows: From the fuzzy segmentation that allowed us to compute the stiffness field, we take the average intensity of the white matter μ_{wm} as a reference level, and then apply the following intensity correction:

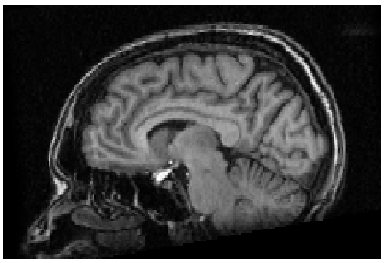
$$I_{new} = \frac{K}{\mu_{wm}} I_{old}, \text{ where } K \text{ is a known constant giving the final intensities.}$$

We have experimentally noticed that the normalization procedure described below for T1-MRI brain images significantly decreases the sensitivity of the algorithm with respect to these parameters. Once the algorithm parameters are tuned for a certain value of K, the user does not have to change their values significantly between two experiments. In fact, all the experiments presented in this chapter were done using the same values of the parameters ($K = 256$, $c = 3.3$, $\lambda = 200$, $\Delta t = 0.2$, $\epsilon = 0.0005$).

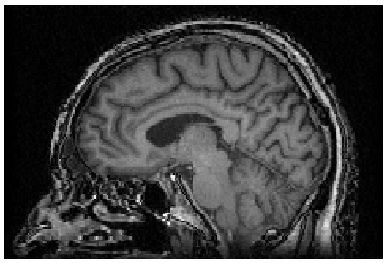
3.2.2 Results

The algorithm was run in parallel (see Chapter 5) on a cluster of 15 2GHz Pentium IV personal computers, linked together through a 1GB/s Ethernet network. For these images of size $256 \times 256 \times 124$, the computation time was 5 minutes, 11 seconds. For comparison, the same registration on a single machine takes 1 hour. Figure 3.2 presents a first registration experiment: large anatomical differences are well recovered by the algorithm, while keeping the transformation invertible and smooth.

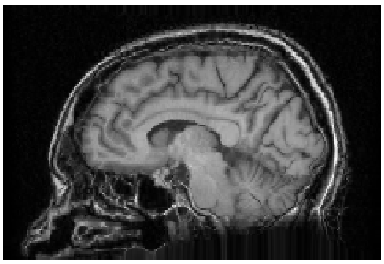
[1] The fuzziness index represents the degree of fuzziness of the classification and it ranges from 1 to ∞. A fuzziness index of 1 corresponds to a hard partitioning.



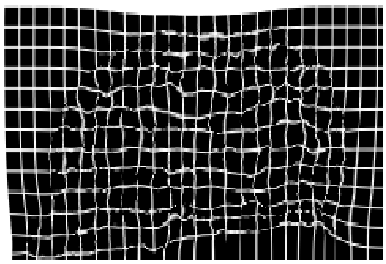
(a) Target image (sagittal view).



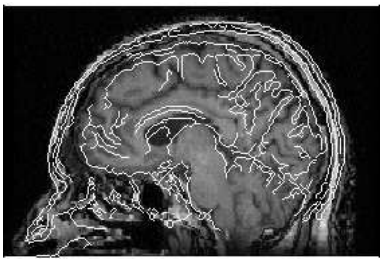
(b) Source image (sagittal view).



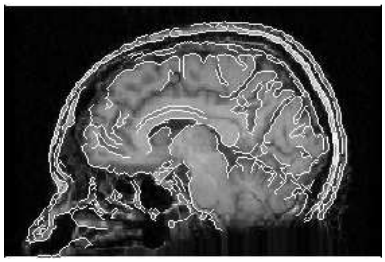
(c) Resampled source image after registration (compare to image in Fig. a).



(d) The deformation field (applied to a regular grid).



(e) Target contours, superimposed on the source image before registration.



(f) Target contours, superimposed on the resampled source image after registration.

Figure 3.2: Registration experiment: Even if the brains presented in the source (Fig. b) and target (Fig. a) images are anatomically rather different, the resampled image after registration (Fig. c) is very close to the target image (Fig. a). The algorithm is able to recover very well the shapes of the ventricles and the major sulci. The recovered displacement field (Fig. d) is smooth. The differences are also presented by superposing contours from the target on the source and resampled images (see, respectively, Fig. e and f).

Figures 3.3 and 3.4 present a second and third experiment, using images of different patients. The result in Figure 3.4 is remarkable in the fact that the algorithm was able to successfully recover the very large difference that exists in the shape of the ventricles.

3.2.3 Discussion: fluid vs. elastic registration

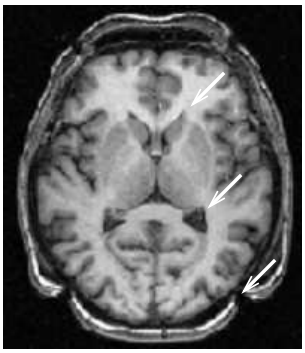
In order to verify the assertions made in Section 3.1.3, we ran a fluid-only registration algorithm, i.e. with no elastic constraints. By comparing the resampled source image obtained with our visco-elastic registration algorithm with the result of a purely fluid algorithm, one can see that the former better minimizes the similarity criterion (Fig. 3.5). This can be explained by the fact that the fluid framework authorizes deformations that are much larger that in the viscoelastic case (Fig. 3.6). As a consequence, the Jacobian of the transformation obtained through fluid registration has much more extreme values than in the viscoelastic case, reflecting much larger volume contractions and expansions (Fig. 3.6).

Our primary objective is not to optimize a similarity criterion, but to recover anatomically meaningful deformations. We have therefore inverted the resulted deformation field, and applied it to the target image[2] (Fig. 3.7). If the registration result is valid, we should obtain a resampled image that is close to the source image. Figure 3.7 shows that this is not the case with the fluid registration: beside large resampling artifacts, the inverted transformation has almost completely eliminated the putamens of the brain (i.e. reduced to a fraction of a voxel). This leads us to conclude that, in the general case, fluid registration does not sufficiently preserve the anatomical coherency of the imaged organs. Figure 3.7 also shows that these artifacts do not occur with viscoelastic registration.

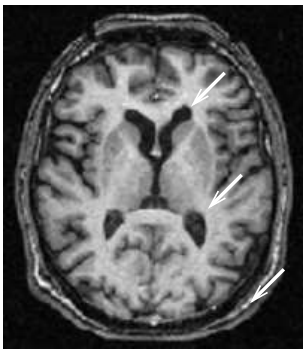
Contrary to the fluid algorithm, an elastic regularization may prevent algorithms from recovering large deformations, but it is able to enforce an a priori constraint on the shapes in the resampled result image. Figure 3.8 presents the result of an elastic registration, which can be compared with the result of our visco-elastic algorithm. By comparing with the registration target, one can notice that, by eliminating the fluid regularization through confidence-based weighting of the correction field, the algorithm is less able to recover the large shape difference of the ventricles. A similar result was obtained with a uniform confidence equal to 1.0.

We believe that this is due to the noise in the image intensities, generating "random" values of the correction field in uniform intensity areas: a stationary smoothing tends to average out to zero the field in these areas, and also reduces the correct

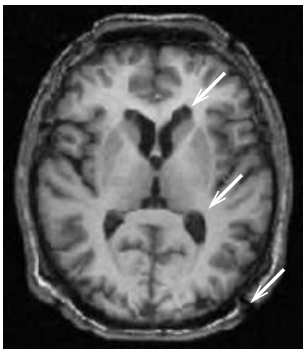
[2]Such an inversion of the deformation field is needed, for instance to propagate atlas labels to patient images when these images are registered into the geometry of the atlas.



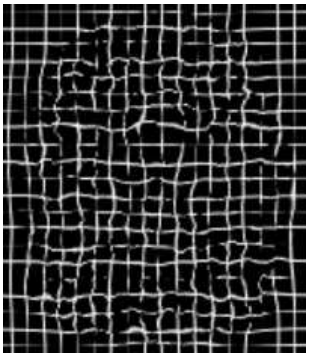
(a) Source image (axial view).



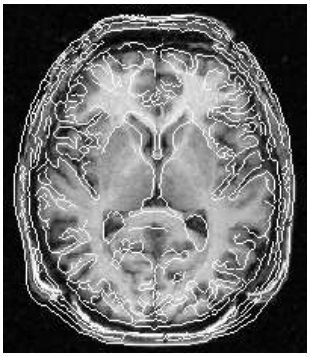
(b) Target image (axial view).



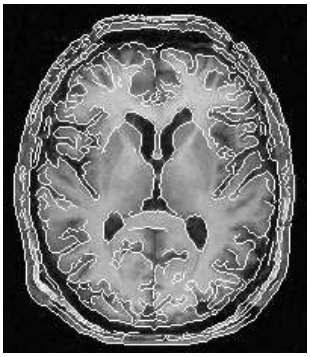
(c) Resampled source image after registration (axial view).



(d) Deformation.

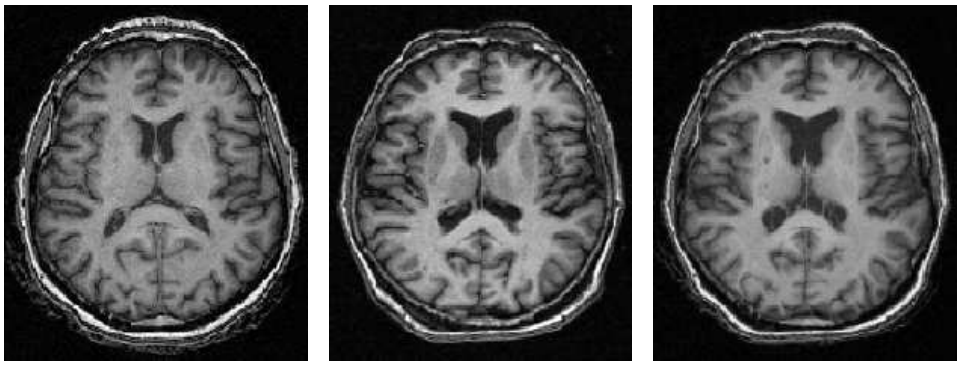


(e) Contours before registration.



(f) Contours after registration.

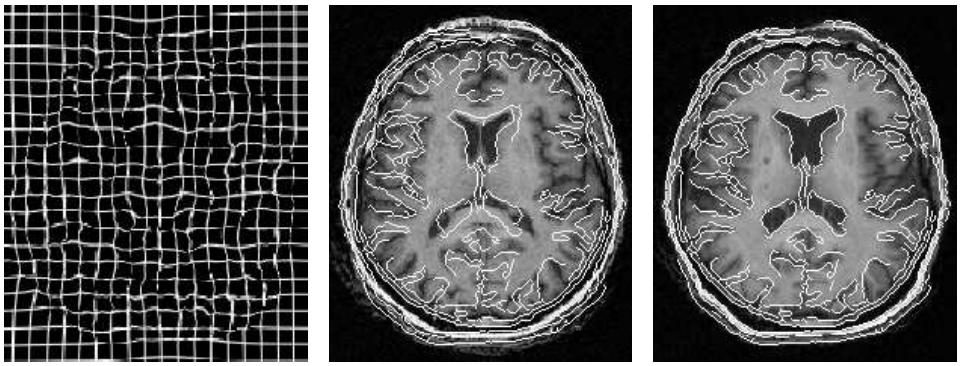
Figure 3.3: Second experiment. The figures show the same axial view for all images. The upper and middle white arrows underline two parts of the ventricles where the source and target images are particularly different. The lower white arrow points to a hole in the skull skin in the source image (Fig. a). This hole is caused by surgery and is not present in the target image (Fig. b). However, after registration, the hole was preserved in the resampled image (Fig. c). We believe that this is the right behavior, since the hole can be consider a part of the image anatomy. The transformation (Fig. d) is invertible and smooth. Figures e and f allow to examine more closely the quality of the result, by comparing target contours superposed on the source image before and after registration (Fig. e and f).



(a) Source image (axial view).

(b) Target image (axial view).

(c) Resampled source image after registration (axial view).

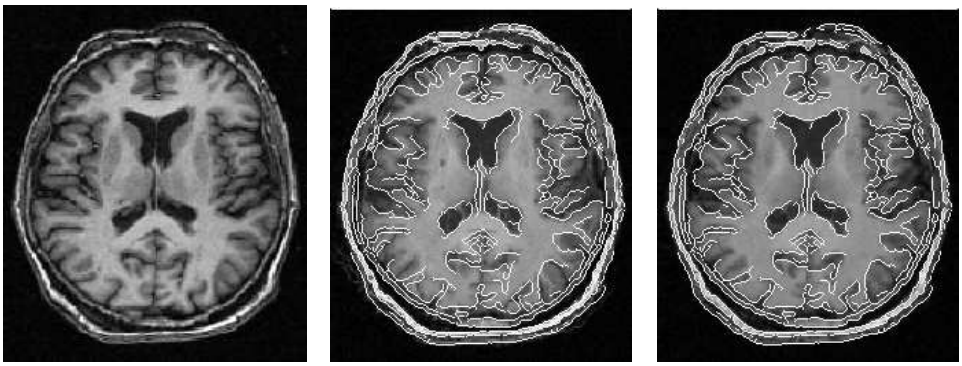


(d) Deformation.

(e) Contours before registration.

(f) Contours after registration.

Figure 3.4: Third experiment: the algorithm is able to compensate very large variations of the shape of the ventricles. The resampled image (Fig. c) is very close to the target image (Fig. b), despite very large initial anatomical differences (Fig. e). The transformation is smooth and invertible (Fig. d).



(a) Target image.

(b) Resampled image after viscoelastic registration. $SSD = 2.7 \cdot 10^6$

(c) Resampled image after fluid registration. $SSD = 1.3 \cdot 10^6$

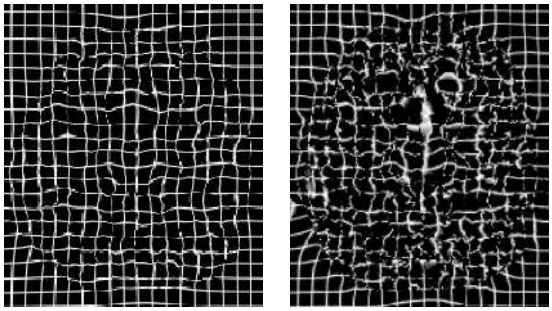
Figure 3.5: Fluid vs. viscoelastic registration: the resampled image obtained by fluid registration (Fig. c) is slightly closer to the target (Fig. a) than the resampled image obtained through viscoelastic registration (Fig. b). The similarity criterion (namely the SSD) has close values in both case.

displacements around the edges. Thus, we observe an excessive adhesiveness of uniform regions that prevent the edges from achieving their complete displacement. On the contrary, a non-stationary regularization "extrapolates" the displacement of the edges to the unreliable uniform areas, leading to a faster and more accurate convergence. This effect is confirmed by a 10% increase in the number of iterations if no fluid regularization is used or if it is stationary.

The solution we propose in this chapter is hybrid between elastic and fluid registration. It composes the correction field with the displacement rather than adding it, and the regularization of the correction follows a fluid model. However, we choose to perform a selective elastic regularization in the areas where, due to anatomical reasons, the displacement of neighboring voxels should be coherent. This enables us to inject in the algorithm some a priori information to constrain the deformations.

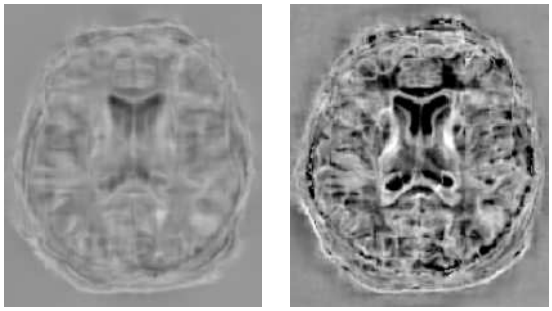
3.3 Pathology-aware registration for 3D conformal radiotherapy planning

Warping a digital atlas toward a patient image allows the simultaneous segmentation of several structures. This is of great interest for cerebral images, since the



(a) The transformation obtained through viscoelastic deformation is smooth.

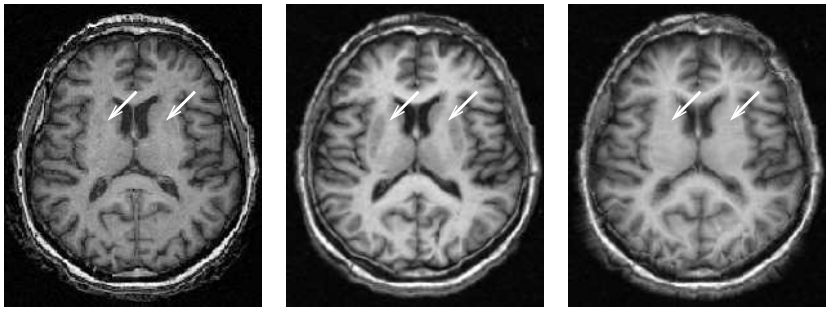
(b) The fluid deformation field is rather irregular.



(c) Viscoelastic registration: the Jacobian has more moderate values ($||\log J|| < 3.24$).

(d) The Jacobian of the transformation has extreme values. ($||\log J|| < 20.46$).

Figure 3.6: Fluid vs. viscoelastic registration: fluid registration results in a displacement field (Fig. b) that is less smooth that the one obtained through viscoelastic registration (Fig. a). This leads to extreme values of the Jacobian of the fluid transformation (Fig. d), reflecting large volume contractions and expansions. For comparison, the Jacobian of the transformation obtained through viscoelastic registration indicates much lower volume contractions/expansions, which reflect in more moderate values of its logarithm (Fig. c).

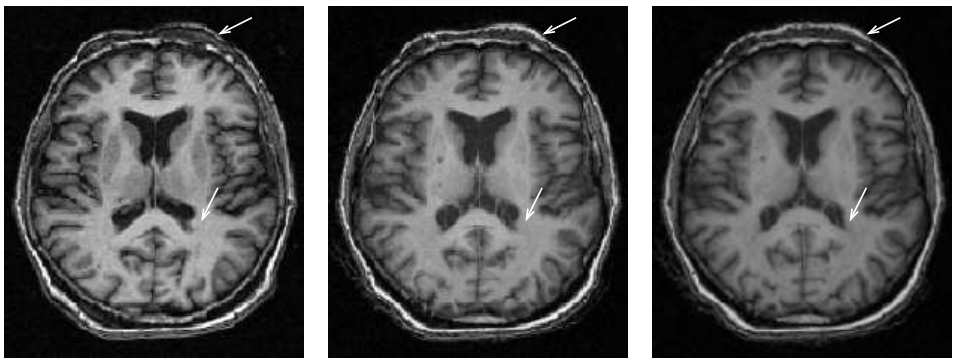


(a) Source image J. The putamen is barely visible (see white arrows).

(b) Viscoelastic $I \circ T^{-1}$: the putamen stays in place.

(c) Fluid $I \circ T^{-1}$ exhibits resampling artifacts; the putamen disapeared.

Figure 3.7: Fluid vs. viscoelastic registration: the transformation was inverted and applied to the target image. Ideally, the resampled image should be identical to the source (Fig. a). However, this is far from being the case in fluid registration: a large partial volume effect appeared around the ventricles, and the putamen simply disappeared, as shown by the white arrows (Fig. c). For comparison, these problems do not occur with viscoelastic registration (Fig. b).



(a) Registration target.

(b) Result using the confidence-weighted elastic regularization.

(c) Result using a uniform fluid regularization.

Figure 3.8: Elastic registration: without fluid regularization, the algorithm is less able to recover the large shape difference between the two brains. White arrows show a lower ability of the elastic algorithm to recover deformations around the ventricles and the skull.

brain contains a large number of small but important structures (optical nerves, grey nuclei, etc.). As we saw in Chapter 1, one important application is the conformal radiotherapy of cerebral tumors, where a precise delineation of all these structures is required. However, in this case, the variability induced by the tumor or a surgical resection, that are not present in the digital atlas, prevents an accurate registration between the atlas and the patient images. Since our registration method allows to locally control the amount of regularization, we are able to explicitly introduce those areas in the warping process.

3.3.1 Pathology segmentation

In order to obtain a priori information on the tumor and the surgical resection to guide the atlas registration, we have to segment these regions in the patient's brain. In experiments, we have used methods developed by Commowick (2003) for automatically delineating respectively the surgical resection and the tumor. The segmentation algorithm mainly uses the information of the joint histogram of T1 and T2 MR images of the patient and a region-based labeling to compute a binary mask of the pathology. The entire method, producing a binary mask of the tumor, is summarized in Appendix B.

3.3.2 Using a-priori anatomical information about the patient

As for every registration algorithm, our method assumes that the chosen similarity metric describes a meaningful correspondence between the two images. This assumption is of course violated when the patient image contains additional structures, such as tumors or resections. Since there is no correspondent in the atlas to each "pathological" voxel in the patient image, we remove the influence of these voxels from the similarity metric, or more efficiently set their confidence $k(p)$ to zero. As a consequence, the correspondences in this area will be determined by interpolation from non-pathological neighboring voxels, for which correspondences can be reliably estimated.

When performing the confidence-weighted filtering of unreliable matches, we assign a null confidence to each voxel inside the pathology. Since we specify the confidence inside the source image, we use the patient image as the source. After registration, we inverse the transformation in order to resample the atlas labels in the geometry of the subject image.

3.3.3 Experimental results

Our test dataset contains 22 T1-weighted MR images of different patients. After a preliminary rigid registration, the images are resampled into the atlas geometry and their sizes are $256 \times 256 \times 60$.

The pathology segmentation takes between 1 and 3 minute, and the non-rigid registration takes about 4 minutes on a cluster of 15 personal computers (2GHz Pentium IV processors, 1GB/s network), which amounts to a total computation time of 5 to 10 minutes (see Chapter 5). The whole database has been processed. Results have been visually inspected by a radiotherapist, and appear satisfactory.

Figure 3.9 shows the atlas used for the registration, and its segmentation. The atlas has been registered with a patient image presenting a large tumor. The pathology has been automatically segmented. If the tumor is not taken into account in the non-rigid registration, the displacement field is biased by the tumor. This results in an erroneous segmentation of the right lenticular nucleus and lateral ventricle. Taking into consideration the pathology ends up in an interpolated displacement field in the tumor area. Therefore, the correspondences around the right lenticular nucleus and lateral ventricle are no longer biased, which leads to a better segmentation.

In Figure 3.10, we present an example where the patient brain has a large surgical resection, that we segmented using the algorithm of Appendix B. A simple non-rigid registration is not able to follow the contour of the cerebellum. If we use the resection segmentation in our algorithm, the segmentation is the cerebellum is largely improved.

3.3.4 Numerical quantification

Testing the quality of a registration algorithm for a given application is an open problem. However, testing the quality of a segmentation is more straightforward when expert segmentations are available. We therefore chose to evaluate the resulting segmentation rather than the registration itself.

We tested the proposed algorithm by registering six patient images towards our atlas. For each patient image, the atlas segmentation of the brain stem was deformed into the patient geometry, and compared to manual segmentations performed by seven clinical experts. The STAPLE (Simultaneous Truth and Performance Level Estimation) algorithm proposed by Warfield et al. (2004a) was used to numerically evaluate the specificity and sensitivity of the automatic segmentations of the brain stem with respect to the expert segmentations. The purpose is to verify that the specificity and sensitivity offered by the atlas registration algorithm are at least as

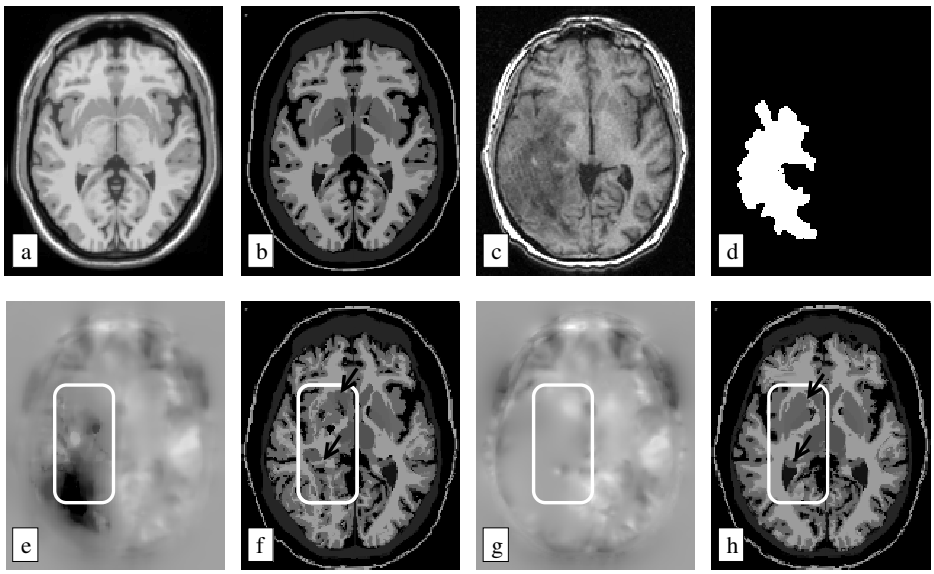


Figure 3.9: Segmentation of a patient image containing a large tumor. (a) Atlas MRI. (b) Atlas segmentation. (c) Patient image, with large tumor. (d) Tumor segmentation. (e) Recovered deformation field without taking the tumor into account: the registration is influenced by the tumor. (f) Resampled atlas labels without using the tumor information: the segmentation fails locally (see black arrows) due to the tumor. (g) Recovered deformation field when taking the tumor into account: if the tumor is taken into account, the displacement field is interpolated inside the tumor area. (h) Resampled atlas labels when using the tumor information: the introduction of the tumor information leads to a more realistic segmentation.

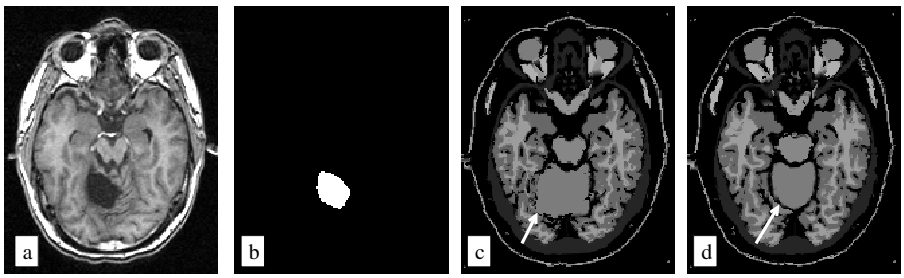


Figure 3.10: Segmentation of a patient image showing a brain tumor resection. (a) Patient image. (b) Resection segmentation. (c) Result produced by a simple registration, unaware of the resection. (d) Result produced by our algorithm, exhibiting a more realistic segmentation of the cerebellum (see white arrows).

Table 3.1: The reliability of the seven experts and of our atlas registration algorithm. The table shows the specificity and sensitivity of each expert / algorithm, and their ranks. Higher specificity / sensitivity and lower ranks are better.

	Specificity	Specificity rank	Sensitivity	Sensitivity rank
Expert 1	0,841	4	0,941	3
Expert 2	0,852	3	0,859	7
Expert 3	0,789	6	0,905	5
Expert 4	0,325	8	0,988	1
Expert 5	0,693	7	0,981	2
Expert 6	0,860	2	0,870	6
Expert 7	0,900	1	0,909	4
Average expert	**0,751**		**0,922**	
Our algorithm	**0,813**	5	**0,839**	8

good as the ones achieved by the clinical experts. Furthermore, we would like the results to be reproducible, so that we can provide a reasonable guarantee of result.

The sensitivity and specificity measures achieved by seven experts and one registration algorithm on six patients represent a large amount of information. Thus, we chose to present two indicators that seem more appropriate to us:

- The **reliability** of an expert or algorithm represents its average specificity / sensitivity on a population of patients.
- The **unpredictability** of an expert or algorithm represents the standard deviation of the specificity / sensitivity on a population of patients. Discussions with physicians showed us that this indicator is important in practical use: a lower unpredictability means that the worst case scenario of the algorithm failing completely is less likely to happen[3].

Table 3.1 presents the reliability of the experts and the registration algorithm. It shows that our algorithms has a slightly higher specificity and a slightly lower sensitivity than the average expert. Both the specificity and sensitivity of the algorithm are comparable with the one achieved by the experts.

Table 3.2 on the next page shows the unpredictability of the experts and of our algorithm. Again, the unpredictability of the algorithm is comparable with the one of the experts, and is lower than the average expert.

[3] Of course, an algorithm with a bad reliability and a good predictability is not desirable: it would be a "predictably bad" algorithm.

Table 3.2: The unpredictability of the seven experts and of our atlas registration algorithm. The table shows the unpredictability of the specificity and the sensitivity of each expert / algorithm, and their ranks. Lower unpredictability on the specificity / sensitivity and lower ranks are better.

	Specificity	Specificity rank	Sensitivity	Sensitivity rank
Expert 1	0,107	6	0,016	2
Expert 2	0,108	7	0,148	8
Expert 3	0,129	8	0,051	6
Expert 4	0,090	3	0,013	1
Expert 5	0,103	5	0,029	3
Expert 6	0,093	4	0,084	7
Expert 7	0,039	1	0,039	5
Average expert	**0,096**		**0,054**	
Our algorithm	**0,067**	2	**0,034**	4

The statistics presented above show that the proposed automatic registration method gives results of a quality that is comparable with the one achieved by clinical experts. Furthermore, the results are reproducible from one patient to another.

The measures presented in this section do not concern the distance between the contours drawn by our algorithm and the real contours. At this point, no measures where performed about this distance. However, visual inspection showed that we are under the 2mm limit required by the application.

3.3.5 Discussion

In this section, we described a non-rigid atlas to subject registration algorithm aimed at automating a brain image segmentation method for conformal radiotherapy. The main difficulty consists in the unpredictable and huge variability introduced either by the tumor or the surgical resection in the patient image, that has no correspondent in the digital atlas. These additional structures introduce false matches in the transformation, and result in a local failure of the registration around the pathology, that may also lead to errors in the neighborhood because of the regularization. Our method is based on segmenting the pathology and reducing the confidence of the voxels inside the pathology. Lowering the confidence in areas containing pathologies enables us to compute the matches by interpolation in these areas.

Results show an improvement of the segmentation in the pathology area. In the

near future, we will validate this method by comparing segmentations produced by algorithm to ones produced by clinical experts. We expect that our grid implementation, presented in Chapter 6, will enable the clinical user to transparently use the computing facilities of our laboratory. We believe this will be very helpful during the validation. Another future improvement is the use of our segmentation as an a priori for the pathology segmentation. An iteration between registration and segmentation will probably result in additional accuracy gains.

Chapter 4

Anisotropic regularization

The previous chapter introduced anatomy-based stiffness information into the registration. In a fluid algorithm, able to estimate very large deformation, the application may impose some limitations in order to locally preserve anatomical coherency. Until now, the stiffness has been non-stationary (position dependent), but isotropic: the proposed algorithm cannot take into account direction-dependent elastic constraints. This chapter attempts to bring such information into the registration process.

4.1 Introducing anisotropic regularization

4.1.1 Method

So far, during the optimization process, the current estimate of the displacement field was continuously regularized with a nonstationary diffusion using the heat equation

$$\frac{\partial U}{\partial t} = \text{div}(\mathbf{d}\ \nabla U)$$

The space-varying $\mathbf{d}$ field represents the local stiffness of the tissue: at some point p, the bigger $\mathbf{d}(p)$ is, the less are local deformations authorized. In a multisubject framework, the "stiffness" is rather virtual, since the registration does not recover a mechanical deformation.

Here, we attempt to go one step further, and incorporate directional information into $\mathbf{d}$. If we replace this scalar field by a tensor one (denoted as $\mathbf{D}$), the elastic

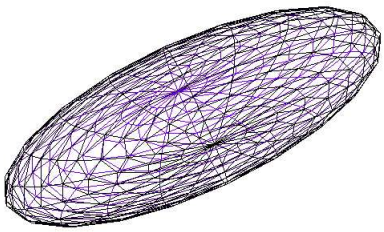


Figure 4.1: Graphical representation of a 3×3 tensor: a 3D ellipsoid. Its orientation is given by the eigenvectors' coordinate system, while the corresponding eigenvalues define the size along these axes.

regularization can become direction-dependent. If $\mathbf{D}(p)$ is the tensor stiffness at point p, its singular value decomposition is

$$\mathbf{D}(p) = (e_1; e_2; e_3) \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} (e_1; e_2; e_3)^T \tag{4.1}$$

where $e_1 = \begin{pmatrix} e_1^x & e_1^y & e_1^z \end{pmatrix}^T$, $e_2 = \begin{pmatrix} e_2^x & e_2^y & e_2^z \end{pmatrix}^T$ and $e_3 = \begin{pmatrix} e_3^x & e_3^y & e_3^z \end{pmatrix}^T$ are the eigenvectors associated to the eigenvalues λ_1, λ_2 and λ_3. The three eigenvectors represent a coordinate system in the 3D space: they have a norm equal to 1 and are all perpendicular on each other. Of course, eigenvectors and eigenvalues are position dependent. Throughout this chapter, we will use for 3×3 tensors a common graphical representation: the 3D ellipsoid. Its orientation is given by the tensor's eigenvectors (which are all perpendicular to each other), and its sizes along the three axes is given by the corresponding eigenvalues.

In an anisotropic diffusion framework, the three eigenvalues represent the amount of diffusion along the axes represented by their corresponding eigenvectors. We can distinguish several levels of complexity:

- The isotropic regularization described in Chapter 3 can be described as a special case of anisotropic diffusion: all three eigenvalues are equal to the local scalar stiffness, which is equivalent to the diffusion tensor

$$\mathbf{D}(p) = \begin{pmatrix} \mathbf{d}(p) & 0 & 0 \\ 0 & \mathbf{d}(p) & 0 \\ 0 & 0 & \mathbf{d}(p) \end{pmatrix}$$

  This tensor can describe isotropically deformable objects, such as illustrated in Chapter 3.

- One may also perform a regularization that is different (but uncorrelated) along each axis of the image space, leading to a scalar diffusion $\mathbf{d}_x(p)$, $\mathbf{d}_y(p)$ and $\mathbf{d}_z(p)$ along, respectively, the x, y and z axes. This leads to the following diffusion tensor:

$$\mathbf{D}(p) = \begin{pmatrix} \mathbf{d}_x(p) & 0 & 0 \\ 0 & \mathbf{d}_y(p) & 0 \\ 0 & 0 & \mathbf{d}_z(p) \end{pmatrix}$$

  This direction-dependent diffusion is simple to implement, since the associated linear operator is separable along three axes. However, since the eigenvectors are fixed to be the axes of the coordinate system, this tensor does not carry enough physical information: there is no reason why in the brain (and, more generally, in natural objects) structures should be specifically oriented along the x, y and z axes only.

- In the general case, the three eigenvectors represent a system of coordinates that is different in each point, and the associated eigenvalues are the diffusivities along the axes of the system (according to Eq. 4.1). Anisotropic stiffness is useful if we want to specifically regularize 1D or 2D imaged structures. For instance, when registering brain images, it may be important to ensure the consistency of sulcal lines: voxels lying on a relatively smooth sulcal line should be warped so that their final positions describe a smooth sulcal line. We will therefore consider for each voxel on a sulcal line a tensor whose eigenvalue along the local tangent to the line is much larger than the two eigenvalues along two axes perpendicular on that tangent. In other cases, we will have to register images containing anatomical structures with segmented surfaces. We would like these surfaces to be deformed in a smooth manner, while giving the algorithm the freedom to deform the image space anywhere else. We leave the tuning of the diffusion tensor for the next section.

In Section 4.2, we will use this diffusion tensor in order to preserve the shapes of the structures of interest during the deformation.

4.1.2 Numerical implementation

After replacing the scalar stiffness field d with a tensor field D, the regularization step consists in solving the following anisotropic diffusion equation for each component U_α of the displacement field

$$\frac{\partial U_\alpha}{\partial t} = \text{div}(D\ \nabla U_\alpha)$$

The Additive Operator Scheme, as presented in Section 3.1.4, is no longer valid for anisotropic diffusion: the right hand term of the above equation can no longer be separated into 3 terms, each containing the derivatives of U_α along the directions x, y and z. Equivalent schemes have been proposed for anisotropic diffusion (Weickert (1998); Mrázek and Navara (2001)), but they separate the diffusion along many directions.

For simplicity, we decided to implement our anisotropic diffusion equation using an explicit scheme, discretized using finite differences. Although this scheme imposes small time steps, its computation time can be considerably diminished, since a parallel implementation is straightforward.

4.2 A thick membrane model

Commonly, anatomical atlases such as the one used for the segmentation of the central grey nuclei in deep brain stimulation planning, contain an atlas MRI, and the surfaces of the structures of interest which were delineated by experts. The displacement of these surfaces must be kept regular, while still allowing them to deform.

In Chapter 3, we showed that fluid registration can recover very large deformations, but the estimated deformation does not sufficiently preserve the anatomical coherency. In this section, we benefit from the fact that we are only interested by the surfaces of the structures which are segmented in the atlas, and we are less interested about the anatomical coherency anywhere else. We want the surfaces to move freely, be highly deformable, but also highly regular, in order for their shapes to be anatomically correct. Therefore, we use as a starting point a fluid registration algorithm, which is highly deformable, and in which we introduce a minimum level of elastic regularization in order to keep surfaces regular. In essence, the idea is to model the deformation of closed membranes (representing the surfaces of the structures of interest) in a fluid environment. The fluid can deform freely, both inside and outside the membrane. However, the deformation of the membrane should stay smooth, in order to preserve the shape of the anatomical structure it represents.

Remark We authorize large deformations (a fluid behavior) inside each membrane. This amounts to the interior of the structures of interest deforming freely. Therefore, the anatomical coherency is no longer enforced inside these structures. This is not really annoying if, for the clinical application, we do not use knowledge about eventual subdivisions of the structures of interest. In particular, this is the case for the planning of the implantation of electrodes for deep brain stimulation, where we are interested in pinpointing the various grey nuclei.

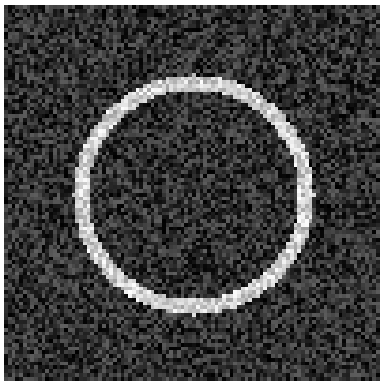
4.2.1 Problem presentation on a synthetic example

Figure 4.2 presents a simple example: we want to regularize a sphere surface, and in the same time minimize the influence of black values surrounding the sphere on the intensities of the sphere voxels. This situation is encountered on the displacement field, where we would like to regularize the surface of a segmented structure, but diminish the influence of surrounding areas on it.

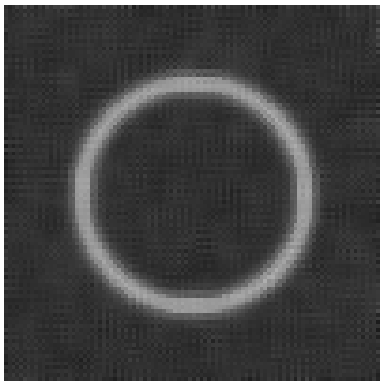
By applying the kind of regularization that is performed in Chapter 3 (strong and uniform inside brain tissues), the white borders are affected by values in smooth areas, and therefore degraded. The figure also shows the result of an isotropic diffusion, which does not regularize the displacements of voxels inside and outside the structure. However, since the borders are thin and contain relatively few voxels, the diffusion still takes too much into account points surrounding them when regularizing the displacement of the structure borders. The signal is therefore still degraded.

A technique commonly used in image denoising is the coherence enhancing anisotropic diffusion. The image is diffused everywhere, except in areas that have strong gradients. On these "contours", the image is only diffused in the directions perpendicular to the image gradient, thereby avoiding to smooth out the imaged objects. If u is the (scalar) image, the structure tensor field is defined as $S = G_\sigma * (\nabla u \nabla u^T)$, where G_σ is the Gaussian of standard deviation σ. The eigenvector corresponding to the largest eigenvalue of the structure tensor gives the direction of the largest intensity variation. In order to enhance the coherency of the contours, the diffusion tensor D near the edges is built with a small eigenvalue along this largest intensity variation direction. The other eigenvalues (in the plane locally tangent to the surface) are fairly large. In areas that are not on edges, all eigenvalues are large, which ensures a large regularization. The process then consists in anisotropically diffusing the image with the structure tensor $\frac{\partial u}{\partial t} = \mathrm{div}(D \nabla u)$. For a review of currently existing anisotropic diffusion techniques, we refer the reader to consult Weickert (1997).

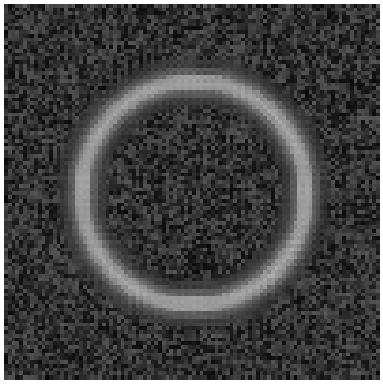
The solution we adopt to obtain a regular sphere without lowering its intensity is to anisotropically diffuse the sphere image. The diffusion tensor (Fig. 4.3) is small and isotropic inside and outside the sphere. On the surface of the sphere, the tensors exhibit a large anisotropy: Eigenvalues corresponding to the two eigenvectors in the plane locally tangent to the sphere are large, thus ensuring that the image intensity on the sphere is regularized. The third eigenvalue, corresponding to the normal to the sphere surface is small, thereby preventing the image intensity of the sphere to be lowered by the influence of surrounding black voxels.



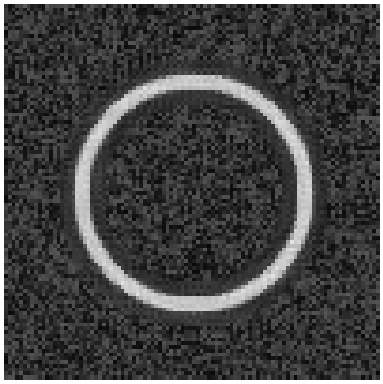
(a) A slice of a white spherical thick surface on black background, affected by 30% noise.



(b) The sphere image was uniformly diffused, leading to a degradation of the intensities of sphere voxels.



(c) Only the thick surface was isotropically diffused. The signal is still degraded.



(d) Only the thick spherical surface was diffused, in the tangent plane. The signal is not degraded.

Figure 4.2: Regularizing an image of a thick spherical surface (a). By performing a uniform diffusion (b), the intensity of the points on the spherical surface is degraded by surrounding values. Even if we isotropically regularize only the voxels that lie near the sphere surface (c), the intensity of these voxels is still lowered. Anisotropic diffusion weighted by the tensor field proposed in Section 4.2.2 regularizes the sphere surface (d), without lowering the intensity on it. Note: in order to make the sphere visible in the images, the sphere surface is 6 voxels thick.

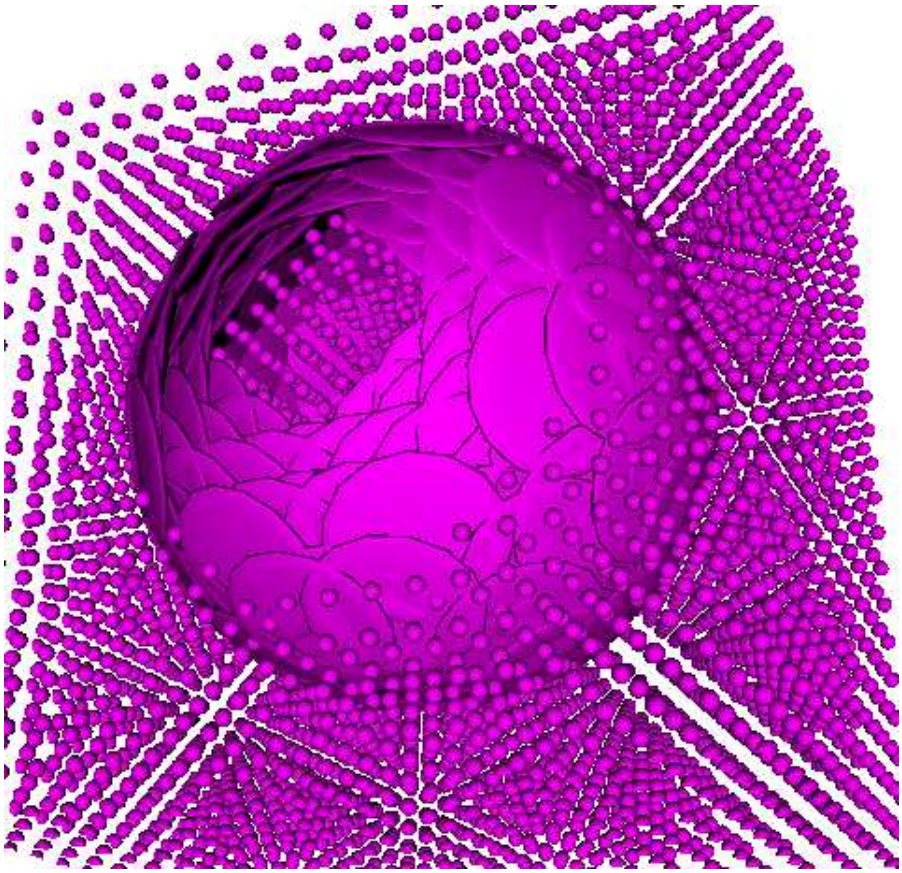


Figure 4.3: Diffusion tensors generated from the sphere structure borders. On these borders, they are large, rather flat and parallel to the surface. They are small and isotropic anywhere else. The image was “cut” on the visible side, in order to allow a view on the tensors inside the sphere.

Algorithm 1 Regularizing structure borders: computation of the diffusion tensor. This algorithm makes references to Algorithm 2 on the facing page.

for each voxel p **do**
  if p belongs to some surface $\mathcal{S}$ (according to Algorithm 2) **then**
    n = normal to the surface of $\mathcal{S}$ in p (according to Algorithm 2)
    e_2 and e_3 = two unit vectors perpendicular on n and on each other
    λ_R = the "small" (residual) diffusion
    λ_S = the (large) amount of diffusion on the surface
    the diffusion tensor has eigenvalues $(\lambda_R, \lambda_S, \lambda_S)$ and eigenvectors (n, e_2, e_3) (Eq. 4.1)
  else
    the diffusion tensor is isotropic with all eigenvalues equal to λ_R
  end if
end for

4.2.2 Regularizing structure surfaces

We want to ensure that the displacements of the surfaces of the structures of interest deform in a regular manner, while the remainder of the image deforms freely. We model the surfaces of these structures as elastic membranes in a fluid environment. In order to let these surfaces behave as close as possible as membranes (in particular to let them slide over each other), we regularize the displacement of the surfaces of interest only in their tangent plane. In doing this, we proceed in a manner that is somewhat similar to the coherence enhancing anisotropic diffusion, but with three major differences: First, our diffusion tensor is estimated based not on image edges, but on the structure surfaces. Second, the tensor is used to regularize the displacement field, rather than an image. In our application, we assume that the regularization of the three displacement coordinates is independent and regularized using the same tensor. Third, like in the sphere example, we perform a very small diffusion in areas that are not on the surfaces, rather than a large one. This will make our algorithm to behave in a very fluid manner everywhere except the structure surfaces. Therefore, the tensor field is computed using the simple algorithm 1.

Let us apply this method to the registration of an image of a Parkinsonian patient with the segmented anatomical atlas. Figure 4.4 presents the atlas T1-MR image with superimposed structures. These structures were manually delineated on histological sections, and then rigidly registered in the MRI geometry. The structures were only segmented in the left hemisphere of the brain (using radiological image alignment conventions, the left hemisphere is on the right side of the image in axial and coronal slices). In order to segment the other hemisphere, the patient image is mirrored w.r.t. the sagittal plane.

Figure 4.5 presents a detail of the atlas MRI. There are two interesting aspects in

Algorithm 2 Testing if a point p belongs to a surface $\mathcal{S}$ and, if yes, computes the normal n to the surface. This algorithm is not optimal, since it does not allow to determine thin surfaces.

let $\mathcal{M}$ be a binary mask of the interior of S
let $grad(\mathcal{M})$ be the gradient image of $\mathcal{M}$, and $\|grad(\mathcal{M})\|$ its norm
let $gmax$ be a user supplied gradient threshold
if $\|grad(\mathcal{M})\| > gmax$ **then**
    p belongs to the surface $\mathcal{S}$
    $n = grad(\mathcal{M}) / \|grad(\mathcal{M})\|$
else
    p does not belong to the surface $\mathcal{S}$
end if

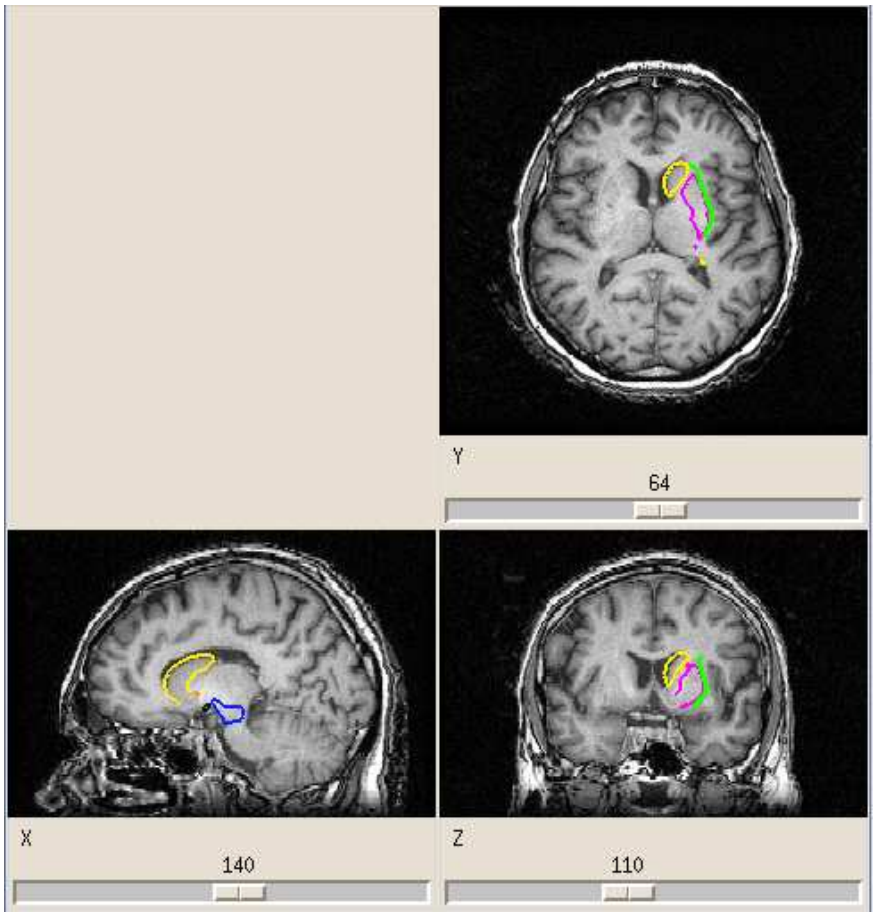
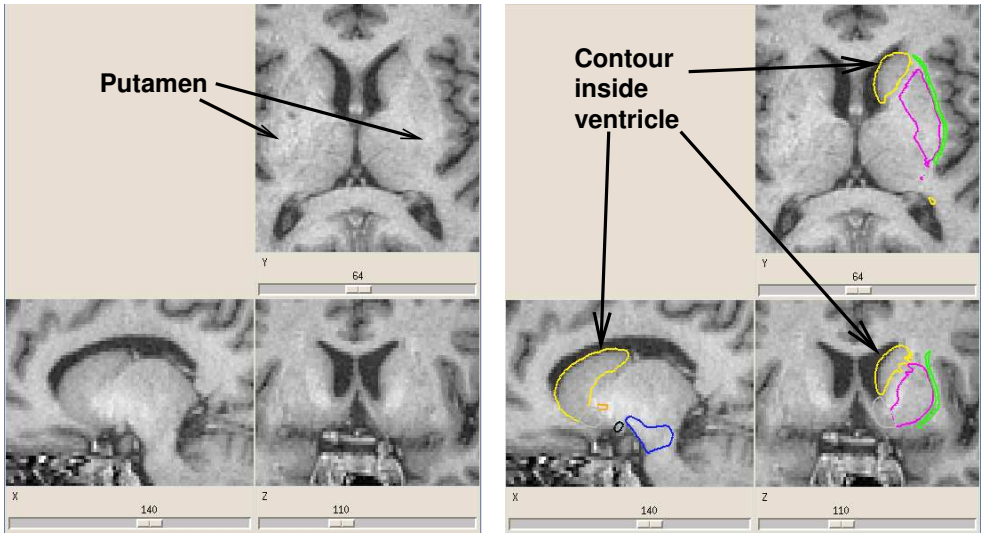


Figure 4.4: Atlas T1-MR image with some superimposed structures.



(a) Atlas detail: some borders of the grey nuclei (e.g. putamen) are barely visible.

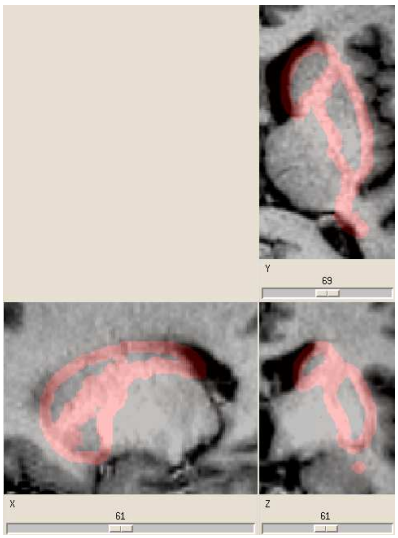
(b) Atlas detail with some structures: due to a missregistration of histological sections towards atlas, some borders of the nucleus caudatus are inside the ventricle.

Figure 4.5: A detail of the atlas MRI exhibits some atlas issues: the edges of some structures (e.g. the putamen) are barely visible (left); contours yielded by expert segmentation on histological sections are slightly misplaced due to a minor rigid misregistration between the histological sections and the MRI (right).

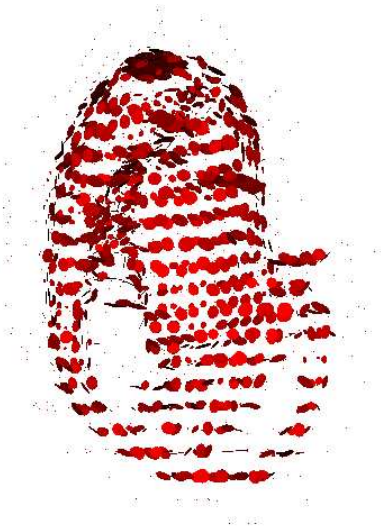
this figure: first, the atlas MRI is a post-mortem one, with a very low white matter/grey matter contrast, and thus the borders of the grey nuclei are sometimes not distinguishable; second, the rigid registration between the MRI and the stack of histological sections was not always perfect, which makes the borders of the nucleus caudatus slightly penetrate inside the ventricle. We are currently addressing these issues.

In the following experiment we only used two structures: the nucleus caudatus and the putamen. A tensor field was computed based on the borders of these two structures according to Algorithm 1 (Fig. 4.6). As in the synthetic example, on the surfaces of the structures tensors are quasi-planar and parallel to the surface. They are isotropic and very small anywhere else. For the planar tensors, the two "large" eigenvalues are about 100 times larger that the small one.

Figure 4.7 shows a comparison between the target image (atlas) and the resampled source (patient) image. The two images are very close, which is a characteristic of fluid registration. However, another comparison with fluid registration (Fig.

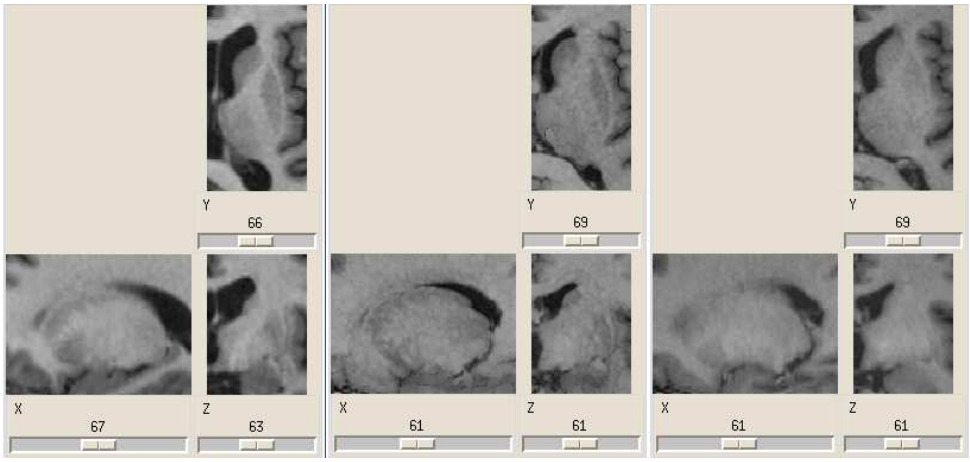


(a) Norm of the tensor field superimposed on a crop of the target image.



(b) Ellipsoid representation of the tensor field. The visualization represents one tensor every six voxels.

Figure 4.6: Tensor field computed based on the borders of the nucleus caudatus and the putamen. On structure borders, tensors are quasi-planar and parallel to the surface (the two "large" eigenvalues are about 100 times larger that the small one). They are isotropic very small anywhere else.



(a) Zoom of the patient image (used as the source image in the registration).

(b) Corresponding zoom of the resampled image (to be compared to the atlas image (c)).

(c) Zoom of the atlas (target image in the registration).

Figure 4.7: Registration result using the proposed anisotropic regularization method on the surfaces of the putamen and the nucleus caudatus: After registration, the patient image (a) was deformed into the geometry of the atlas. This resampled image (b) is very close to the target atlas image (c) despite the source image (a) being very different.

4.8) shows that the introduction of our regularization method greatly improves the regularity of segmented surfaces after deformation.

4.2.3 Discussion

The regularization method presented in this section introduces in the fluid registration a minimum level of regularization in order to preserve the coherency of the surfaces of desired structures. This is done by modeling these surfaces as elastic membranes, and let the remainder of the image (interior and exterior of the surfaces) behave like a viscous fluid. In order to achieve this effect, we replace the isotropic regularization by an anisotropic one: close to the structure surfaces, the diffusion is high, anisotropic and parallel to the surface; it is isotropic and very small anywhere else. The main advantage of this method is that it gives to the structures the ability to deform and to move with respect to each other, while still keeping their surfaces regular. This allows us to cumulate advantages of fluid and

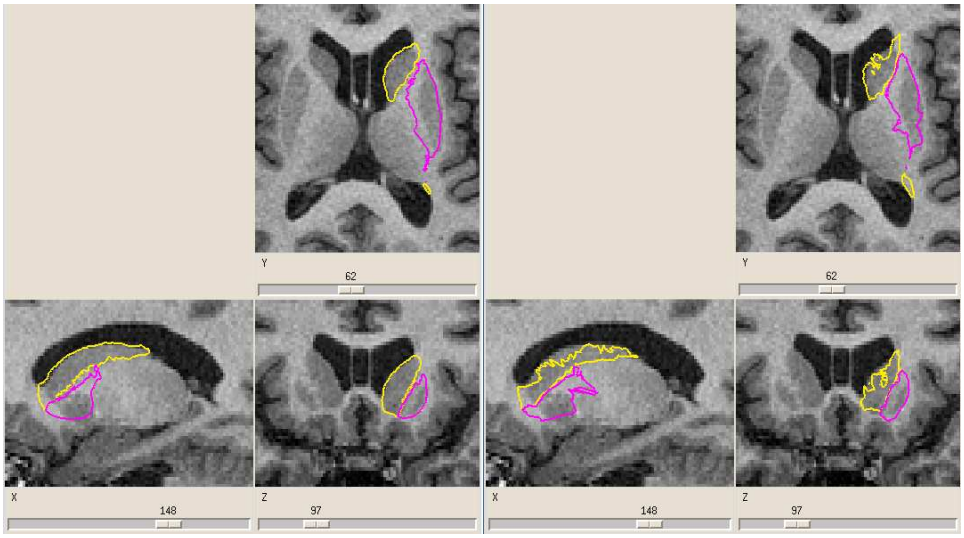


Figure 4.8: The segmentation of the putamen and nucleus caudatus. The contours (a) are smooth compared to the ones obtained through fluid registration (b).

elastic registration: the resampled image is very close to the target (as in fluid registration), and the deformed surfaces are regular (as in elastic registration).

Our experiment underlined some issues remaining to be solved in our approach. The main problem comes from the relative lack of contrast of the central grey nuclei: the intensities in the two images do not seem to provide enough information on the borders of some nuclei. The lack of contrast is particularly acute in the atlas image which was acquired post-mortem, and thus exhibits an exceptionally low white matter / grey matter contrast. Future work will have to concentrate on increasing the contrast of the grey nuclei. We envision two possible sources of additional information that can be used to match these nuclei:

1. One could use the information of the segmented structures in the atlas in order to compensate for the lack of contrast. This may allow to artificially enhance the contrast of the atlas MRI near the surfaces.

2. If more than one image modality is available for both the atlas and the patient image, a classification method similar to the one presented by Dugas-Phocion et al. (2004) (based on a multi-modality EM algorithm) could be used in order to obtain fuzzy maps of white and grey matter. We believe that these maps provide more reliable information than the raw image intensity.

Chapter 5

Parallelization

Previous chapters described a non-rigid registration algorithm that is able to realistically model the deformations of non-homogeneous organs, such as the brain. In order to estimate more realistic deformations of these organs, the model uses a priori knowledge about their deformability. The algorithm uses a dense transformation model that is able to describe fine local deformations. The semi-implicit Additive Operator Scheme used to perform the non-stationary Gaussian regularization insures a relatively low computation time (several tens of minutes, depending on the image size). However, in hospitals, time is expensive in human and financial terms. Ideally, the registration should only add a time that is small with respect to the total amount of time required by the medical intervention. After discussions with physicians, we estimated that the computation should take no longer than five minutes. This would bring the computation time down to a "human time"[1] of a few minutes.

The purpose of this chapter is to present a method that accelerates the algorithm without compromising on its accuracy or its robustness. We want to drastically reduce the computation time while executing rigorously the same algorithm that was described in the previous chapters. Since there are no miracle ways to make a microprocessor execute the same computations in much less time, we used a parallel computer with several microprocessors to perform the registration.

This chapter begins by a quick presentation of parallel computing and how we can make the best use of it in order to solve our registration problem (Section 5.1.1). In Section 5.2, we provide a global view of the parallel implementation. Following sections detail the parallelization of the algorithm.

[1] We think a good estimate of how much a user wants to spend waiting for a program to finish is given by the duration of a coffee break. Writing an e-mail is another reliable estimate :-)

5.1 A brief overview of parallel computing

5.1.1 Parallel architectures

Technology divides parallel computers into two main classes, depending on the way the microprocessors synchronize with each other:

- In *shared memory computers*, all microprocessors have a random access to the same internal memory. They share a common memory space, in which any processor can access data in a transparent manner, based only on the memory address.
- *Distributed memory computers* contain microprocessors that have their own private memory. No processor can have a direct access to the memory of any other processor, and there is no common address space. Instead, the processors are connected by a communication network. Synchronization is done by communication: one processor *sends* data, while another one *receives* it. The functional unit containing the processor, its memory and the network interface is commonly called a *node*[2], or simply a *processor*.

The two technologies have their own advantages and drawbacks. Since they have a common address space, shared memory computers are easier to program: all processors have access to all the variables in the program. However, it is difficult for a single memory chip to simultaneously provide data to many processors. This tends to make shared memory computers inefficient for memory intensive applications like ours. Technological solutions to avoid this *memory bottleneck* exist, but they are financially very expensive. On the contrary, distributed memory computers are more efficient for memory intensive applications: each memory only "talks" to its own processor. However, synchronization between processors is much harder to manage, as there are no shared variables and the programmer has to use explicit data communications between processors. These communications are easier to describe if the algorithm has a regular structure: it uses large vectors of data which are processed in a similar manner.

The registration algorithm described in the previous chapter uses a lot of memory, but is rather regular. The algorithmic operations performed in order to estimate the transformation at each point are identical[3]. This enables us to use a distributed memory computer. The machine that we used is a widespread model: a *cluster of workstations*. It consists of several workstations (they can be PC's, Mac's or Unix workstations) linked together through a Local Area Network (LAN). This setup is common in laboratories and hospitals.

[2] One can imagine a distributed memory computer as a graph where a network edge links each pair of computational nodes.

[3] With minor exceptions on the borders.

5.1.2 The Single-Program-Multiple-Data programming model

Fortunately, the programmer of a distributed memory parallel computer does not have to explicitly describe the actions of each of its nodes. In the Single-Program-Multiple-Data model, all nodes execute the same program, but their behavior changes depending on the local values of internal variables. One important variable is the *rank* of the node, an integer that uniquely identifies each node inside a cluster. For simplicity, we consider the rank to be in the range $0 \ldots number\ of\ nodes - 1$. We call the node with rank 0 the *master* of the cluster. Although nothing distinguishes this processor from its fellows, it is by convention the one that performs some special operations, like console or file input and output.

Processors can communicate the contents of internal variables. The communication is done through *messages* that are *sent* by one processors and *received* by another. Send and receive operations must correspond to each other. Sometimes a node has to send the contents of a variable to all the other nodes. We call this operation a *broadcast*. For instance, it is common for the master node to read the input data from a file and then broadcast it to the other nodes.

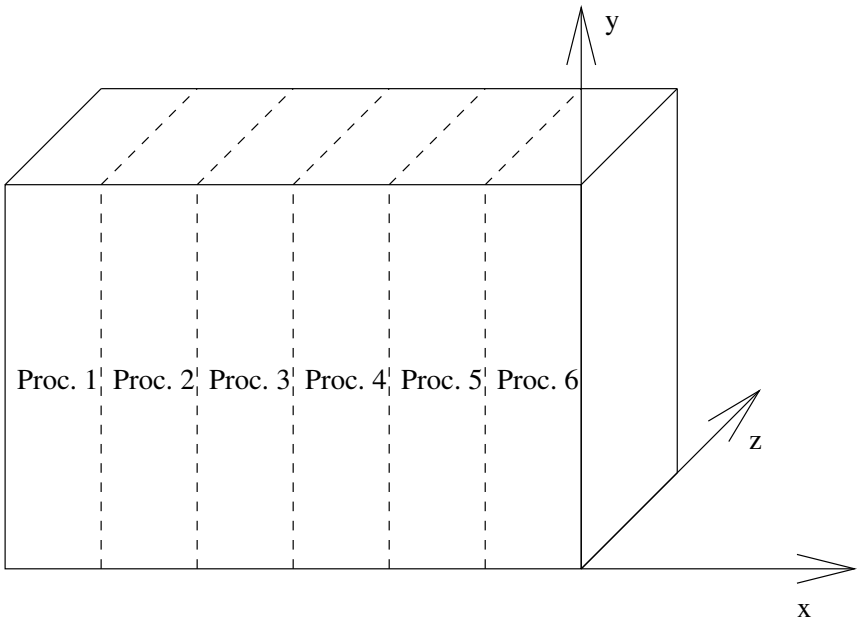
Many times the internal variables are large arrays. If parallelism is achieved by having each processor use or compute a part of the elements of such an array, it is interesting to distribute the array to the nodes. The manner in which the elements are distributed is called the *data decomposition*. If initially all elements of the array are on the same node, they must be *scattered* to the other nodes in order to obtain a distributed array. Conversely, the elements of a distributed array can be *gathered* to obtain a conventional array inside the memory of a node.

5.2 The parallel structure of the algorithm

Perhaps the most important thing in creating a parallel algorithm is to establish a data decomposition of the various data arrays that the algorithm uses. This determines in an essential manner the pattern of communication between the nodes. At the beginning of the algorithm, the spatial support of the transformation is cut into subdomains, consisting in parallel stacks of slices, that are distributed to the available processors (Fig. 5.1). Since the processing nodes only compute the displacement field at the voxels of their own subdomain, the correction field is decomposed in the same manner as the displacement one.

Thanks to their relatively small size, the two input images may be maintained in the memory of each node. Since at each iteration the source image is resampled with an a priori unlimited displacement field, the source image has to be entirely memorized by each node. However, if the SSD similarity criterion is used, a processor

Figure 5.1: The data decomposition of the displacement and correction fields with 6 processors.



computing a certain vector $u(p)$ of the correction field at a point p only needs the value of the target image $I(p)$ in the same point p. In this particular case, the target image can be distributed.

From an algorithmic point of view, we can distinguish 3 logical parts in our method: the computation of the gradient of the similarity criterion; the composition of the displacement and correction fields; the Gaussian filtering (required to compute the gradient of the similarity criterion); and the AOS-based regularizations.

The estimation of the gradient of the SSD at one voxel requires the resampling of the source image and the computation of the gradient of the resulting image. The resampling can be done independently in each subdomain since processors know both images entirely and the transformation of their subdomain.

5.3 Parallel composition of displacement fields

As we have seen in Chapter 2, once a small correction field has been estimated, it is composed with the current displacement field. This raises an issue in the parallel implementation: both the correction and displacement field are distributed

among the processors. The trivial solution is to gather the two fields on processor 0, make it perform the composition, and then scatter the composed field back to the processors. This algorithm is however inefficient, for two reasons:

1. Each of the fields is fairly large. For instance, if the size of the images to register is a typical $256 \times 256 \times 120$, the total size of the two fields is 180MB.

2. Even when using a simple trilinear resampling, the composition itself is a demanding computation, which consists in more than 70 floating point operations for each voxel.

Therefore, we would like to perform the composition in parallel, without redistributing data. Let us recall that the composition of the two fields is

$$(U \circ u)(p) = U(p + u(p)) + u(p)$$

In our data decomposition, a processor owning the voxel p has to access the correction field u at the voxel p, and displacement field U at the voxel $p + u(p)$. If the sizes of the correction vectors were unlimited, the communication pattern would be very complex, as each processor would have to explicitly request parts of the deformation field that are owned by the other processors. However, we guaranteed in Section 2.2.3 the invertibility of the recovered transformation by limiting the size (in voxels) of each component of the correction field at each iteration to the interval $(-0.5, 0.5)$. This implies that, in order to compute $U(p+u(p))$ using a trilinear resampling, a processor only has to know the displacement field U *in points that are neighbors of* p *on the grid*. The parallel composition of the two fields is straightforward:

1. Each processors communicates to its neighbors the displacement vectors of U lying on the one-voxel-wide borders of its domain. At the end of this communication step, if a processor has a point p in its domain, it also knows the displacement field in all neighbors of p.

2. As the components of $u(p)$ are limited in size to 0.5, each processor can now compute $U(p + u(p)) + u(p)$ for all points p of its domain using a trilinear resampling.

5.4 Parallel Gaussian smoothing

Although the regularization is implemented using an AOS-based diffusion, Gaussian filtering is still needed by the algorithm in order to compute the gradient required by the derivative[4] of the similarity criterion $\nabla SSD = (J \circ T - I)\nabla(J \circ T)$. Indeed, when computing this derivative of the similarity criterion, it is important to avoid local minima. Since the derivative of an image containing noise is affected by that noise, we need to compute a smooth gradient $\nabla(J \circ T)$ of the source image. Therefore, we have to parallelize an implementation of the Gaussian filtering.

There are at least two ways to reduce the computation time when convolving a three-dimensional image with a three-dimensional Gaussian. First, one can take advantage of the *separability* of the three-dimensional Gaussian[5]: convolving an image with it amounts to successively convolving the image with three one-dimensional Gaussians with the same standard deviation. If we consider a Gaussian of standard deviation σ as approximately null outside the $[-3\sigma, 3\sigma]$ interval, the corresponding Gaussian filter consists in convolving the image with a cubical kernel of size $(6\sigma)^3$. If the voxels are of size $1 \times 1 \times 1$, this amounts to $(6\sigma)^3$ multiplications for each voxel. This high dependence of the computation time on the standard deviation of the Gaussian, hence on the level of regularization, is highly penalizing. However, a one dimensional Gaussian filter with the same standard deviation consists in convolving the image with a linear kernel of size 6σ. By taking into account that the image is filtered three times with one-dimensional kernels, this amounts to 18σ multiplications for each voxel, and therefore a linear variation of the computational complexity with the amount of regularization.

Second, Deriche (1992) proposed to replace the convolution with a Gaussian kernel by a recursive approximation, which achieves a computation time for each voxel that is *constant with respect to the width of the Gaussian*. In this section, we begin by recalling this algorithm, propose two parallel implementations, and finish by a comparative analysis of their respective algorithmic complexities.

5.4.1 The sequential recursive algorithm

In this section, we recall the recursive implementation of the Gaussian filtering, as established by Deriche (1992). It will serve as a basis to establish the parallel algorithm. The main advantage of this algorithm is that, unlike the convolution, the computation time does not depend on the standard deviation of the Gaussian. This time depends linearly on the image size.

[4] In practice, we convolve with the derivative of the Gaussian, but the parallel filtering algorithm is the same.

[5] This property is valid for any number of dimensions.

In one dimension, the Gaussian can be represented by a *non-causal* filter[6]. Since a non-causal filter cannot be implemented in a recursive manner, the algorithm consists in applying one *causal*[7] and one *anti-causal*[8] filter: assuming $h(j)$ to be the impulse response of the Gaussian, it is split into a causal part

$$h_+(j) = \begin{cases} h(j) & \text{if } j \geq 0 \\ 0 & \text{if } j < 0 \end{cases}$$

and an anti-causal part

$$h_-(j) = \begin{cases} 0 & \text{if } j \geq 0 \\ h(j) & \text{if } j < 0 \end{cases}$$

By minimizing the mean square error between the result of fourth order recursive filters, and the result of equivalent Gaussian filterings, the algorithm pre-computes the coefficients α_j, β_j, γ_j and η_j of two fourth order recursive filters, depending on the required standard deviation of the Gaussian. Then, the fourth order causal filter (5.1) is applied forwards and the corresponding anti-causal filter (5.2) is applied backwards.

$$out_i^+ = \sum_{j=0}^{j<5} \alpha_j in_{i-j} + \sum_{j=1}^{j<5} \beta_j out_{i-j}^+ \tag{5.1}$$

$$out_i^- = \sum_{j=0}^{j<5} \gamma_j in_{i+j} + \sum_{j=1}^{j<5} \eta_j out_{i+j}^- \tag{5.2}$$

The filtered version of the signal is obtained by taking the sum $out^+ + out^-$ of the signal filtered by (5.1) and (5.2). In more dimensions, one takes advantage of the separability property of the Gaussian, and successively filters the image along all directions. When applying the filter to some image in a certain direction, the image is decomposed into lines along that direction. Throughout this chapter we will call the lines along the recursive filtering direction *scanlines*. Each scanline is considered as an one-dimensional signal that is filtered independently of the others using the forwards and backwards scheme (Algorithm 3).

[6] A filter whose output value at the current time depends not only on the input values at past moments, but also on the input values at future instants.

[7] The output values of a causal filter depend only on the input values at past moments.

[8] Conversely, the output values depend only on the input values at future moments.

Algorithm 3 Recursive Gaussian filtering.

compute the coefficients α_j, β_j, γ_j and η_j depending on σ
for each direction $d \in \{x, y, z\}$ **do**
  for each scanline l along d **do**
    // filter forwards
    for each point i in l in *increasing* order **do**
      //Eq.(5.1)
      $out_i^+ := \sum_{j=0}^{j<5} \alpha_j l_{i-j} + \sum_{j=1}^{j<5} \beta_j out_{i-j}^+$
    end for
    // filter backwards
    for each point i in l in *decreasing* order **do**
      //Eq.(5.2)
      $out_i^- := \sum_{j=0}^{j<5} \gamma_j l_{i+j} + \sum_{j=1}^{j<5} \eta_j out_{i+j}^-$
    end for
    // write the result back into the line l
    for each point i in l **do**
      $l_i := out_i^+ + out_i^-$
    end for
  end for
end for

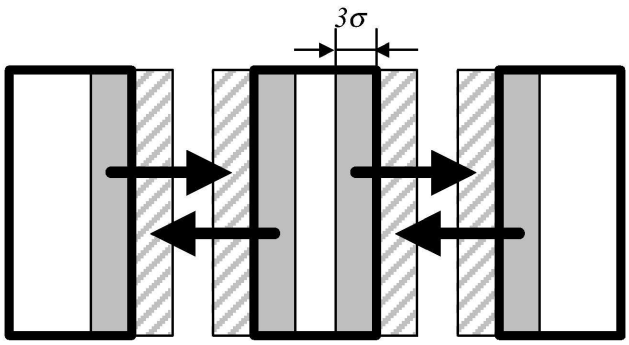
5.4.2 The border sending algorithm

We present below the case of a scalar image. For a vector field, the operation has to be repeated for each of its components. The Gaussian filter is separable, so convolving a three dimensional image with an isotropic Gaussian is equivalent to successively convolving the image with a one-dimensional Gaussian along each axis. By adopting a block decomposition *along one axis only*, the filtering along two directions can be done within each block without any communication. For filtering along the decomposition axis, one may benefit from the exponential decay of the Gaussian: a good approximation is to consider the Gaussian as null outside its $[\mu - 3\sigma, \mu + 3\sigma]$ interval (where we denoted with μ the Gaussian's mean and with σ its standard deviation). Therefore, when convolving a one-dimensional signal with a Gaussian, we can consider that the value of the filtered signal in some point p depends only of the values of the points of the initial signal within $p \pm 3\sigma$. This leads to the following simple algorithm (Figure 5.2):

1. Each process sends its borders of width 3σ (in gray in Figure 5.2) to its neighbors.

2. Each process receives the neighbors' borders (in dashed grey) and adds them to its own domain, obtaining an extended domain.

Figure 5.2: The send-borders algorithm: Each process sends its 3σ-wide borders to its neighbors and then filters the enlarged domain.



3. Each node recursively filters its own extended domain and then throws away the received borders, in order to obtain a domain the size of the initial one. For efficiency reasons, the convolution with a Gaussian is approximated inside each domain by the sequential recursive filter.

This algorithm has two drawbacks: First, the produced results are not rigorously correct since the value of a Gaussian is not perfectly null outside the 3σ interval. Using larger borders will make the parallel algorithm less efficient. Second, each process has to apply the filter to a domain that is larger than its own. And finally, the amount of data sent through the network is proportional to the filter's standard deviation. This is penalizing if the desired level of regularization is high.

5.4.3 A pipeline algorithm

An alternative is to directly parallelize the fourth order recursive implementation of the Gaussian, as follows. Let us consider the lines of the image along the block decomposition direction: they can be filtered independently of each other. Due to the recursive nature of the filter, computing the value of one point depends on the filtered version of the previous point when filtering forwards, and of the following point when filtering backwards. This means that the filtering of one single line cannot be done in parallel. However, different processors can deal with their parts of different lines simultaneously (Algorithm 4, Figure 5.3): At step 1, the left process begins processing its part of the first scanline. Meanwhile, processors 2 and 3 wait. Once processor 1 finished, it can pass on to processor 2 the contents of the 4 points (since the order of the filter is 4) that processor 2 needs in order to process its first point of its part of the first scanline. Process 1 filters its part of the second scanline

Algorithm 4 The pipeline parallel Gaussian recursive filter.

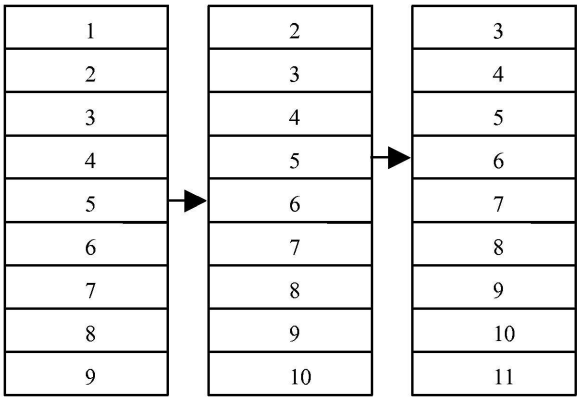
for each direction d **do**
    / filter forwards */*
    for each scanline l along d **do**
        if not the first processor along d **then**
            receive from the preceding processor along d its already-filtered last 4 points
        end if
        filter forwards the line l
        if not the last processor along d **then**
            send the last 4 points to the succeeding processor along d
        end if
    end for
    / filter backwards */*
    for each scanline l along d **do**
        if not the last processor along d **then**
            receive from the succeeding processor along d its already-filtered first 4 points
        end if
        filter backwards the line l
        if not the first processor along d **then**
            send the first 4 points to the preceding processor along d
        end if
    end for
end for

while process 2 filters its part of the first scanline and process 3 does nothing. At the end of this step, process 1 passes the last 4 points of its part of the second scanline on to process 2, while the latter one sends the last 4 points of its part of the first scanline to the third process. This way, all the processes work simultaneously without filtering one scanline in parallel. The "process pipeline" however takes a number of steps equal to the number of processes before working at its full capacity. The full acceleration is achieved if the number of lines is much larger then the number of processes, which is usually true in a cluster of workstations.

5.4.4 Performance analysis

Two methods to achieve parallel recursive Gaussian filtering were proposed. We now quantify the algorithmic complexity of each of them: computational complexity, network usage and maximal acceleration (when the communication time is null). We assume below that we are filtering an image of size $N_x \times N_y \times N_z$ using M processors. The filter has a standard deviation σ. We assume that computing

Figure 5.3: The pipeline parallel filtering of a 2D image of 10 lines with 3 processors. Inside each line, the step at which it is processed is given. At the end of step 5, processor 1 has just finished filtering its part of the fifth line, and is sending the last 4 points to processor 2. Meanwhile, processor 2 has just finished filtering its part of the fourth line and is sending its last 4 points to processor 3 who has just finished filtering its part of the third scanline. At step 6, the three processors will filter their parts of lines 6, 5 and 4.



the recursive filter (Equations (5.1) and (5.2)) for each point is done in a time t. This time is the same for the sequential method and the two parallel algorithms.

Computational complexity In the first method, each process filters its own domain plus the margins that the neighboring nodes sent. So the computation time is $N_y N_z \left(\frac{N_x}{M} + 6\sigma\right) \cdot t$. The second method does not filter borders, but there is a pipeline filling and emptying penalty. Therefore the computation time in this case is $\frac{N_x N_y N_z}{M} t + (M-1)\frac{N_x}{M} t$.

Network usage Another important quantifier of the algorithmic efficiency is the total amount of data sent through the network. In the first algorithm, each processor sends 3σ-wide borders to its neighbor, so the total amount of data sent by each processor is $6N_y N_z \sigma$. As we saw before, the amount of data sent by the second algorithm does not depend on σ, each processor sending only the last 4 points (original data and filter result). The total amount of data for this algorithm is $16N_y N_z$.

Finally, let us investigate the number of messages sent. For the first algorithm, each processor sends only two messages containing the two borders. For the second algorithm, the number of messages is larger ($2N_y N_z$), which can make the

Table 5.1: Theoretical performances of the two parallel recursive Gaussian filtering algorithms.

Quantifier	Borders sending algo 1	Pipeline algo 2
Computation time	$N_y N_z \cdot \left(\frac{N_x}{M} + 6\sigma\right) \cdot t$	$\frac{N_x N_y N_z}{M} \cdot t + (M-1)L\frac{N_x}{M}t$
Amount of data sent	$6N_y N_z \sigma$	$16N_y N_z$
Number of messages	2	$\frac{2N_y N_z}{L}$
Maximal acceleration	$\frac{M}{1+\frac{6\sigma M}{N_x}}$	$\frac{M}{1+\frac{L(M-1)}{N_y N_z}}$

algorithm inefficient. However, a trade-off between the number of messages to send and the maximum parallel acceleration can be made as follows: Rather than sending the last four points of a single line at a time, one can send the last four points of several lines in a single message at the cost of an increase of the time necessary to fill the pipeline. If we denote this number of lines with L, the number of messages for the second algorithm becomes $\frac{2N_y N_z}{L}$ and the computation time becomes $\frac{N_x N_y N_z}{M} \cdot t + (M-1)L\frac{N_x}{M}t$.

Maximal acceleration Even if we consider a null communication time, the acceleration is not linear in either of the cases. For the first algorithm, one must take into account the additional time needed to filter the two received borders. Therefore, the acceleration with M processors is:

$$A_1(M) = \frac{N_x N_y N_z \cdot t}{N_y N_z \cdot \left(\frac{N_x}{M} + 6\sigma\right) \cdot t} = \frac{M}{1 + \frac{6\sigma M}{N_x}}$$

Notice that for one and two processors, the law above is not true: For one processor no borders are added ($A_1(1) = 1$), whereas for two processors, only one border is added per processor ($A_1(2) = 2/\left(1 + \frac{6\sigma}{N_x}\right)$).

For the second algorithm, we must take into account the time necessary for filling and emptying the pipeline:

$$A_2(M) = \frac{N_x N_y N_z \cdot t}{(M-1)L\frac{N_x}{M}t + \frac{N_x N_y N_z}{M}t} = \frac{M}{1 + \frac{L(M-1)}{N_y N_z}}$$

Each of the two algorithms can be the most efficient in different situations. The first algorithm is efficient if the standard deviation of the Gaussian is low and the connection network has a high latency. The second one is more efficient in sparing processor time and network bandwidth, especially if the standard deviation of the filter's Gaussian is high. The drawback of sending many messages can be dealt

with by tuning the L parameter (number of lines sent in one message). In practice, L is determined experimentally, as a function of the network latency: it has high values when the latency is high, while low latency networks can tolerate small L's, thereby improving the maximal acceleration. The second algorithm has another advantage: The minimum width of a processor domain is 4 points, whereas in the case of the first algorithm the minimal width is 6σ. This enables our pipeline recursive filtering algorithm to properly work with a much higher number of processors. Another advantage of the second algorithm is, off course, the fact that it is more accurate. In our experiments we chose to use the pipeline recursive filtering rather than the borders sending algorithm.

5.5 Parallel Additive Operator Scheme

In our registration algorithm, the regularization consists in solving a heat equation. In order to speed up the computation, we use the Additive Operator Scheme (described in Section 3.1.4). There are already existing implementations of the AOS on distributed memory parallel computers. Bruhn et al. (2002) propose an implementation that requires clusters connected through high performance (hence expensive) networks. At each step, the image that is being filtered is redistributed, requiring all processors to communicate to each other. This implementation is not adapted to clusters of PC's connected through low cost networks.

As we saw in Section 3.1.4, the AOS scheme is separable. If we consider the diffusion equation ($\partial U/\partial t = \text{div}(d\ \nabla U)$), the right hand term can be separated into three terms, each of them depending only on the derivatives of U along one of the directions x, y and z. This transforms the semi-implicit scheme of the diffusion equation ($\mathbf{v}^{t+\Delta t} = (\mathbf{I_N} - \Delta t\ A^t)^{-1}\ \mathbf{v}^t = B^{-1}\mathbf{v}^t$) into one that implies only the inversion of tridiagonal matrices: $\mathbf{v}^{t+\Delta t} = \frac{1}{3}B_x^{-1}\mathbf{v}^{t+\Delta t} + \frac{1}{3}B_y^{-1}\mathbf{v}^{t+\Delta t} + \frac{1}{3}B_z^{-1}\mathbf{v}^{t+\Delta t}$. The matrices B_x, B_y and B_z are either tridiagonal or they can be transformed into tridiagonal matrices by properly reordering the elements of the vector $\mathbf{v}$. Along a given axis, the diffusion can be independently performed on each line (we call *line* the succession of the voxels encountered in the image when only one coordinate parameter does vary). Since we cut the support of our transformation into subdomains that are parallel stacks of slices (say perpendicular to the x direction), there are two directions for which all processors work entirely in parallel (the y and z axis), without any need for communication. We are left with the x direction, perpendicular to the domain decomposition, and along which all the lines are distributed among the processors. Let us describe the inversion of the matrix B_x. A line $\mathbf{w}$ along the x axis can be seen as a projection of size dim_x of our previous "big image vector" $\mathbf{v}$. Let $B_\mathbf{W}$ be the corresponding projection (of size $dim_x \times dim_x$) of the matrix B_x. For each line $\mathbf{w}$, we have to solve $\mathbf{w}^{t+\Delta t} = B_\mathbf{W}^{-1}\ \mathbf{w}^t$. Since the matrix $B_\mathbf{W}$ is tridiagonal, its inversion can be done

using the Thomas algorithm (see Appendix A.1). The main remaining difficulty is that no parallel processing is possible *on a single* line along that axis due to the recursive nature of the Thomas algorithm. However, we have *several* lines to process. Furthermore, the Thomas algorithm is a first order recursive filter with one causal and one anticausal part. Algorithmically, this is very similar to the structure of the recursive Gaussian filter (described in Section 5.4.1). We will therefore apply a pipeline parallelization method, such as the one we used for the recursive Gaussian filter (Section 5.4.3).

A rigorous pseudo-code description of the entire algorithm is given in Algorithm 5. The goal is to invert in parallel M tridiagonal matrices $B^0, B^1, \ldots, B^{M-1}$ of size $N \times N$, on P processors. If we equally distribute each line to all processors, each processor p is responsible for processing the components $\alpha_{p\frac{N}{P}+1,\ldots,(p+1)\frac{N}{P}}$, $\beta_{p\frac{N}{P}+1,\ldots,(p+1)\frac{N}{P}}$, and $\gamma_{p\frac{N}{P}+1,\ldots,(p+1)\frac{N}{P}}$. It also memorizes a part of the elements of the matrices L and R: the elements of the vectors l, r and m with the same indices. In order to minimize the total number of messages, we fuse in the algorithm below the loops that compute the LR decomposition and the forward substitution step.

5.6 Parallel performance

5.6.1 Execution time comparison: stationary vs. non-stationary regularization

In Chapter 3, we replaced the stationary regularization (through Gaussian filtering) by a nonstationary diffusion PDE. Unfortunately, the fast implementation of the stationary Gaussian (Section 5.4.1) can not be used in the case where the degree of regularity varies spatially. A simple option would consist in implementing our regularization method through convolution, and locally adapt the standard deviation of the Gaussian kernel. However, this would imply very large computation times. The AOS-based filter we chose has the advantage of being recursive and requiring a very low number of operations. In this section, we evaluate the potential loss of time implied by the evolution from the stationary to the nonstationary regularization. Four factors are important when evaluating the execution time of a parallel algorithm: the computational complexity, the amount of memory used, the total quantity of data communicated through the network and the total number of messages.

In Table 5.2, we compare our parallel implementation of the AOS-based regularization method with the parallel implementation of the recursive Gaussian filter. The recursive implementation of the Gaussian is a fourth order recursive filter, whereas the AOS scheme is in essence a first order recursive filter. Since the two parallel

Algorithm 5 Parallel Additive Operator Scheme

$fi := p\frac{N}{P} + 1$ the first index memorized by processor p
$li := (p+1)\frac{N}{P}$ the last index memorized by processor p
// Fused LR decomposition and forward substitution steps
for each matrix $j \in [0, M-1]$ **do**
  if $p = 0$ **then**
    $m_1^j := \alpha_1^j$ //$fi = 1$
    $r_1^j := \beta_1^j$
    $y_1^j := d_1^j$
  else
    receive m_{fi-1}^j, β_{fi-1}^j and y_{fi-1}^j from processor $p-1$
    $l_{fi}^j := \gamma_{fi}^j / m_{fi-1}^j$
    $m_{fi}^j := \alpha_{fi}^j - l_{fi}^j \beta_{fi-1}^j$
    $r_{fi}^j := \beta_{fi}^j$
    $y_{fi}^j := d_{fi}^j - l_{fi}^j y_{fi-1}^j$
  end if
  for $i := fi + 1$ to li do
    $l_i^j := \gamma_i^j / m_{i-1}^j$
    $m_i^j := \alpha_i^j - l_i^j \beta_{i-1}^j$
    $r_i^j := \beta_i^j$
    $y_i^j := d_i^j - l_i^j y_{i-1}^j$
  end for
  if $p \neq P-1$ **then**
    send m_{li}^j, β_{li}^j, and y_{li}^j to processor $p+1$
  end if
end for
// Backward substitution
for each matrix $j \in [0, M-1]$ **do**
  if $p = P-1$ **then**
    $u_N^j := y_N^j / m_N^j$ //$li = N$
  else
    receive u_{li+1}^j from processor $p+1$
    $u_{li}^j := \left(y_{li}^j - \beta_{li}^j u_{li+1}^j\right) / m_{li}^j$
  end if
  for $i := li - 1$ down to fi **do**
    $u_i^j := \left(y_i^j - \beta_i^j u_{i+1}^j\right) / m_i^j$
  end for
  if $p \neq 0$ **then**
    send u_{fi}^j to processor $p-1$
  end if
end for

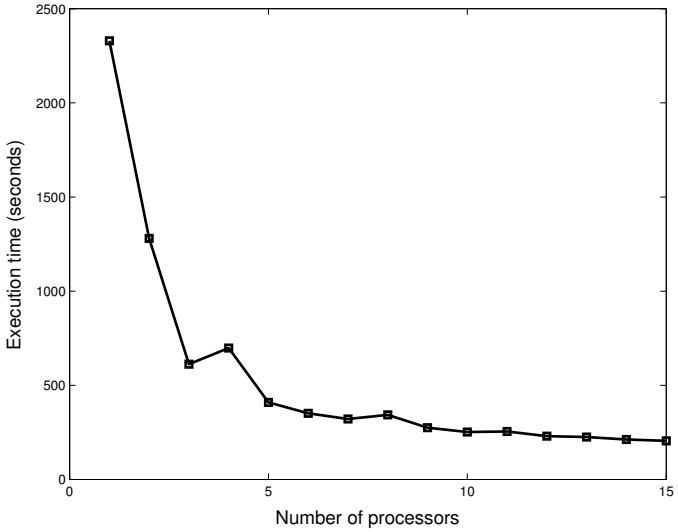
Table 5.2: A performance comparison of the parallel AOS algorithm with respect to the parallel recursive Gaussian filtering. Bold values indicate indicate gains in computations, memory need or communications. The values take into account the fact that the value of each voxel of the transformation is a three-component vector, which triples the computational complexity, the amount of communicated data and the memory needed.

	AOS	Gaussian
Computation (operations/voxel)	**24**	90
Memory (numbers/voxel)	27	**9**
Communications (numbers/line)	**12**	45
Number of messages per line	2	2

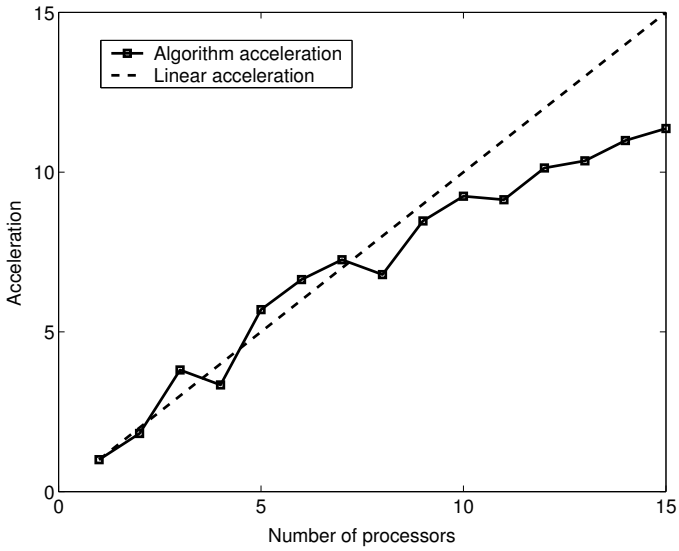
algorithms use the same communication pattern, the first order AOS filter sends the same number of messages as the fourth order Gaussian filter, but each message is about four times smaller. The AOS filter also needs a lower number of arithmetical operations. From Table 5.2, one can see that the first order filter performing the AOS non-stationary diffusion is in every aspect at least as good as the fourth order Gaussian filter, except for memory use. However, since the memory used by the two filters is distributed over the different workstations, the extra memory use does not represent a problem.

5.6.2 Parallel acceleration of the entire algorithm

The computation times of the complete registration algorithm and the parallel acceleration are presented in Figure 5.4, for two images of size $256 \times 256 \times 120$. By using 15 2 GHz Pentium IV processors, we obtained an acceleration of 11 (reducing the computation time from 40min to 3min30). This can be justified by the fact that some parts of the algorithm (creation of the image pyramids) are still sequential. We believe that the minimal time 3min30 makes the algorithm suitable for clinical use. Notice that, for some configurations, the acceleration that was obtained was larger than the number of processors. We link this fact to the performance of the machines' memory. Since the algorithm uses large quantities of memory, cache misses are rather frequent when run sequentially. When spatial blocks are smaller, cache misses occur much less often, which largely improves the algorithm performances.



(a) Execution time graph.



(b) Acceleration graph (real vs. theoretical).

Figure 5.4: Performance graphs for the entire parallel algorithm. The algorithm acceleration (continuous curve, fig. b) is sometimes larger than the linear one (dotted curve, fig. b). We believe that this is due to the memory cache: if the size of the local data on each processor is smaller, the memory cache becomes more efficient, and the processor spends much less time parsing the memory. The minimal time (fig. a) is of 3min30.

5.7 Conclusion

In order to lower the computation time and make the algorithm more suited to clinical use, we proposed in this chapter a parallel implementation on a cluster of personal computers. This hardware platform has a low financial cost, and can be purchased by many laboratories and hospitals. The parallel algorithm is based on a data decomposition of a displacement field, and at each step a processor only communicates with at most two other processors (no costly one-to-many or many-to-many communications). The result is a very good parallel acceleration, even when using a relatively large number of processors. This enables the algorithm to take good advantage of inexpensive but powerful medium-size clusters.

Although it mainly concerns the implementation of the algorithm on a particular hardware platform (a cluster of PCs), this chapter describes some theoretical contributions regarding the parallel computing for image processing. As such, we described novel parallel implementations for two algorithms which are already very fast: the recursive filter for Gaussian smoothing, and the Additive Operator Scheme for solving diffusion PDE's. This enables us to achieve a low computation time (3 minutes 30), which in our opinion fits our initial purpose: execute the algorithm in an amount of time that a clinical user can spend waiting for the results.

Chapter 6

Grid implementation

The software described so far is able to register two images in a reasonable amount of time. However, its usability remains rather low. The user must log into a cluster, upload the input images, run the program through a command line, wait for the results, download and visualize them. Each of these steps must be performed separately by the user. This is clearly not the way things are supposed to be in a clinical environment. In hospitals, physicians are busy treating patients and they have very little time left for hacking into complex computer systems.

A common way to ease the usage of a software is to embed it into a Graphical User Interface (GUI). In our case, we would like our GUI to have the following features:

- open and visualize the input images;
- manually set the parameters, and be able to save and load a parameter set;
- start the registration;
- visualize in real time an intermediate result;
- if necessary, stop the process;
- visualize and save the final result.

For a sequential program, building a GUI around it is not an especially difficult job, and there are well established technologies that transform this task into an interesting engineering problem. However, a setup containing a parallel computer can be more complex. In order to maximize the computer power and minimize costs, manufacturers build machines which, even if based on workstation technology, have somewhat special designs. For instance, the nodes of the cluster may

lack all the components which are not absolutely required in order to compute, like the graphics card or hard-disk. The direct consequence is their inability to display graphics. Furthermore, due to their complexity, clusters have special administration needs that require permanent professional handling. This places them outside the clinical environment, in the hands of system administrators inside a data center. For us, there is a direct consequence: neither of the cluster's nodes can display the user interface. An external visualization workstation, placed in the clinical environment, should be in charge of all user interaction. The purpose of this chapter is to present a system that connects the visualization module, running on a clinical workstation, with the registration module, running on a cluster placed in a data center. Some important additional issues, like speed and security, will also be addressed.

6.1 System overview

Our system is composed of two interacting modules: the registration software and the graphics user interface. The registration module performs the computation. It is run on a parallel computer that usually needs special conditions to operate, such as air-conditioned rooms. Therefore, such a machine cannot lie next to the user and is more likely to find a suitable place as a shared resource in a data center than in an operating or pre-operative planning room. The GUI provides the interaction between the registration module and the user. It also deals with the visualization of the three dimensional images involved. For visualization performance reasons, the GUI must be executed on the user's computer. The computers running the two components of our system must be connected through a network. Due to the different conditions in which these two computers must operate, such a network may be a long distance one. Moreover, it may be difficult to install additional wires in the operating room for intra-operative registration. Thus, we want to enable our service to run through a wireless, hence low bandwidth, network interface.

The purpose of our work is to implement the registration as a grid service able to connect the clinically useful user interface to a computational center. This interaction has to be sufficiently refined to allow the user to dynamically manipulate the state of the registration software.

The problem of building grid services to control medical imaging algorithms (such as registration or segmentation) has been addressed by many research efforts. Burns et al. (2004b) propose a universal framework (named IXI - Information eXtraction from Images), which allows to use in an uniform manner a wide range of medical imaging software (also see Burns et al. (2004a)). The IXI system provides a centralized access to medical imaging services located all over the world, allowing users to use and compare the efficiency of many different algorithms.

For neuroimaging, a similar goal is targeted by the Neurobase project (Barillot et al. (2004)). In order to facilitate the building of functional cerebral maps (both under normal and pathological conditions), this project allows to federate multiple sources of medical data and algorithms under an uniform access framework.

The specific problem of grid-enabling a registration algorithm for clinical usage was addressed by Ino et al. (2003), who used a high-end cluster connected to the visualization workstation through a high-bandwidth Wide Area Network (WAN). We believe that the infrastructure used in this work is not representative of the one available in most hospitals. Lower end to average clusters and low bandwidth WAN's are in our opinion more realistic hardware platforms for any medical grid service. Moreover, this paper does not address the interactive usability issue that we believe essential in a clinical environment.

In this chapter, we present a system that enables the interactive use of parallel registration software running on a remotely located cluster. The presentation emphasizes the mechanisms used to connect our user interface to the grid service, and the security and usability concerns that are raised by such a system.

6.2 Requirements

To summarize, we want to build a registration grid service implemented on a cluster of PC's on the (computation) server side, and a graphical interface able to control it on the user side. In order to connect them, we need a communication library that fulfills several conditions:

1. The communication library has to be as flexible as possible. If the users makes two requests A and B to the GUI, and request A is transmitted to the grid service before request B, it is important that no constraint be imposed to the responses of the two request. The response to B may arrive before or after the response to A, and one or both responses may be never sent at all.

2. Communications have to pass through firewalls.

3. The access to the grid service should be secured. A public key authentication such as RSA fulfills our needs regarding access permissions to the grid service.

4. Communications should be encrypted. Since confidential medical data is transmitted through a WAN that is not always under the hospital's full control, this data should be encrypted.

5. The whole system should preserve the anonymity constraints imposed by current regulations (Herveg and Poullet (2003)). Image files usually contain information about the subject's identity. This information should never make its way to the non-controlled WAN. The solution is to send only the data necessary for the registration: the image sizes and the image intensity data.

6. In order to maximize the accessibility to the grid service, the communication library should spare bandwidth. Therefore, each message transmitted through the WAN should be compressed before sending.

6.3 Technological choice

During the recent years, a vast amount of work has been invested in distributed computing environments providing standardized and secure communications between remotely located computers. We review below some of these environments in view of our application.

6.3.1 Client/server

Several standardized communication methods between a client and a server have emerged. Recently, older standards, such as RPC and CORBA, have been replaced by protocols based on XML, which ensure the interoperability between implementations. The SOAP protocol, used by Web Services (WS) and the Open Grid Services Infrastructure (OGSI) (Tuecke et al. (2003); Foster et al. (2002)) has lately become the communication standard for client/server distributed systems.

Generally, the server interface is described in a dedicated interface description language (IDL in CORBA, WSDL in Web Services, GWSDL in OGSI), and the communication interface is generated automatically. The main advantage of such a system is that, when the client sends a service request to the server, the identification of the request and the decoding of its parameters is done in a transparent manner. We feel that several aspects of client/server systems makes them unsuited for our task:

First, a client cannot simultaneously communicate with more than one server. This means that when the server is a parallel software, the client would have to communicate exclusively with the master node. However, when the master node receives a request, the identification of the request is done in a transparent manner, outside the user's control. Thus, further action would have to be taken in order to forward the request type to the other nodes. From this point of view, client/server would not bring additional functionality to the system.

The client/server system has another major drawback in our case. In this model, the server's sole purpose is to answer requests from the client. It cannot send requests to the client, and it remains idle between processing two successive requests. In our system, this constraint imposed on the server is too hard.

6.3.2 Message passing

This model, used by basic networking protocols (TCP/IP) and low level grid layers (Globus I/O), and generally adopted by computation libraries (MPI, PVM), provides flexible high-performance communications. In TCP/IP, the user has to explicitly describe the data transfers and the message encoding. Higher level libraries (MPI, PVM) facilitate the message encoding and also provide collective communications (e.g. broadcast). However, these libraries were designed for non-secured communication between the nodes of a parallel computer. Furthermore, they are generally unable to pass through firewalls.

We prefer the message passing communication model, since it does not impose any constraints on the possible actions of the service. Our goal is to create a message passing library that can provide secured communications and is compatible with firewalls.

6.4 Method

We designed a message passing library that is able to transmit messages back and forth between the GUI and each of the nodes running the grid service. Thanks to compression and encryption, the transmission of messages is fast and secure.

This section provides a resume of the method. More implementation details can be found in Appendix C.

6.4.1 Communications

In our program, we distinguish between computation messages (exclusively exchanged between the nodes of the cluster) and control messages (exchanged between the parallel program and the GUI). Computation messages are directly communicated using MPI's point-to-point or collective communication subroutines, and they were explained in Chapter 5. Control messages are programmed as C++ classes containing a message tag that identifies the message, and a message handler describing the action to execute upon reception. Each message has two methods,

pack and unpack, that describe the way the message can be packed into a buffer for sending, or unpacked from a received buffer.

Control messages are sent through communication channels, that encapsulate the communication protocols. From a programmer's point of view, these messages are C++ classes that provide:

- packing and unpacking basic data types (integer and real numbers and arrays) into buffers. These functions are called by the message's pack and unpack procedures upon the sending and the receipt of the message.
- sending and receiving the packed buffer.

Until now, we have implemented two channels:

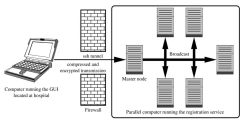
- The long distance channel, currently based on TCP/IP, is used for communicating between the GUI and the master node. For performance reasons, this channel transparently compresses the messages before sending. It also takes care of the endianism difference that might exist between the machines running the GUI and the cluster.
- The local channel, based on MPI, is used to forward the messages received by the master node from the GUI to the other nodes of the cluster.

The communication of a message using a channel is done as follows: the message tag and the message itself are packed into a buffer of the appropriate size and then sent. Upon receipt, the message tag is decoded and an object of the appropriate message class is instantiated. The message is unpacked and its associated message handler is invoked.

In the case where the GUI sends a message to the cluster, the message is first sent to the master node using the long distance channel, and then distributed by the master node to the entire cluster using the local channel (see Figure 6.1). The message handler is subsequently invoked in parallel by all the nodes of the cluster. The sending of a message by the service to the GUI is performed by the master node of the cluster.

The architecture we have presented here has the advantage of being modular. Adding message types or communication channel can be easily done by inheriting the appropriate C++ class.

Figure 6.1: The visualization workstation communicates with the registration module running on a cluster



6.4.2 Security issues

Both the GUI and the grid service are isolated from the rest of the world by firewalls. Regarding the long distance channel, three security issues are important:

- Communications must be able to pass through the two firewalls.
- The user must be authenticated before being allowed to use the service.
- Communications should be encrypted.

We use SSH tunneling in order to fulfill these three requirements. If a SSH tunnel is not created the registration service is not visible by the user. In order to create the SSH tunnel, the user has to authenticate using a user name and a password delivered by the service administrator. The tunnel also provides the encryption of the data flow between the GUI and the service.

Despite all these security mechanisms, a possible anonymity breach can come from the facial reconstruction that can be done on some MRI images. The solution would be to modify the images before transmission through the network in a way that does not affect registration but makes facial reconstruction impossible. We have not tackled this problem.

6.5 Results

The final system is the following: After user authentication, the GUI enables the loading and comparative visualization of the images to register. A user interface button enables the user to contact the grid service and start the registration. During the process, the user is informed in real time about the status of the algorithm (small size messages). An intermediate result image (several megabytes of data) is regularly sent to the GUI which displays it. At any time the user has the option to

abort the registration. Upon termination of the algorithm, the registration result is sent back to the GUI.

We have tested our system by running the GUI on a 600MHz Pentium III laptop equipped with a wireless network interface. The grid service runs on 15 2GHz Pentium IV PC's linked through a 1GB/s Ethernet network and protected by a firewall running on a 2GHz Pentium IV PC. All systems run the Linux operating system. The resolution of the input images is 256x256x124, and their size is 47.2MB. After compression, they reach a size of only 3.3MB. The transmission of the input images takes 39 seconds, including the additional compression and decompression times. For comparison, if compression is not used, the transmission takes 2 minutes and 26 seconds. During the execution of the algorithm, each update of the intermediate result takes 18 seconds. The extra time took by the transmission of images through the network is of about 2 minutes, thus bringing the total registration time to about 7 minutes. However, thanks to the user interface and the grid service, the registration software becomes usable in the clinical environment. Thus, we believe that the two extra minutes in computation time are tolerable.

We also tested our system in a worst case scenario by using a home Asymmetric DSL connection between the user interface and the grid service. The maximum transfer rate is 16kB/s from the GUI to the service and 64kB/s from the service to the GUI. The image upload time is about 3min30, and the result update time is about 1min30. In this extreme case, the total registration time reaches ten minutes, including image transfer time. However, this proves that our non-rigid registration service can be used from almost everywhere with a very modest equipment.

6.6 Discussion

We have presented a technology that potentially enables a clinician to transparently use a computationally intensive but powerful non-rigid registration algorithm run by a parallel computer physically located at a large distance. The system combines the speed and precision provided by the parallel computer to the ease of use of the physician's workstation. A graphics interface enables the user to supervise the execution of the algorithm in real time. In the future, we will modify the system so that physicians will be able to use their clinical expertise to guide the registration. We will add an integrated authentication, the automatic creation of the ssh tunnel and the remote execution of the service by the user. A test in a clinical setup is also in preparation.

Our non-rigid registration service opens up numerous advanced medical image analysis applications to the clinical practice, such as the usage of brain atlases or image guided therapy. However, while medical imaging algorithmic issues have

mostly been dealt with, grid management problems still have to be addressed. The most important problem is the standardization. Indeed, while the long distance TCP/IP channel fulfills its functionality requirements, its integration with standard security and authentication mechanisms and discovery services is not ensured. These aspects have been addressed in existing grid environments, such as the Globus Toolkit. In the future, we intend to replace the TCP/IP-based long distance channel with one based on Globus I/O. This would also enable us to replace SSL certificates and SSH tunneling with the similar but more standardized functionality provided by the Globus Security Infrastructure.

Chapter 7

Conclusion

Contributions can be divided in three classes: first, on the introduction of a priori information into multi-subject nonlinear registration methods; second, on parallel and grid computing methods applied to medical image registration; and third, on the application of the proposed methods to concrete clinical problems. These contributions gave rise to 3 publications in international journals (Stefanescu et al. (2004b,c, 2005)), 3 articles in international peer-reviewed conferences (Stefanescu et al. (2003a, 2004a); Ourselin et al. (2002)), as well as other conference articles (Stefanescu et al. (2003b, 2004d)).

7.1 Methodological contributions to nonlinear registration

One of the goals of this thesis was to incorporate a priori knowledge about the anatomy or the possible pathology of imaged objects into the registration algorithm. The manuscript proposed a "pair-and-smooth" approach that uses a dense displacement field to describe the transformation, which ensures that it can recover deformations at the voxel resolution. The algorithm alternates the estimation of an incremental correction field (based on a similarity criterion) and two level of regularization (fluid and elastic). The whole process guarantees that the recovered transformation is invertible. The contributions of the thesis are based on three assumptions. First, we consider that not all voxels in images contribute equally relevant information to the registration. Voxels close to contours give more reliable displacements than voxels in smooth areas. Second, the imaged objects do not deform homogeneously. And finally, some structures in one image, especially

pathological ones, do not have correspondents in the other. The anatomical modeling is performed through the usage of two fields: a confidence field on the intensity matches, which weights the importance of the different voxels in the registration process; and a possibly anisotropic nonstationary stiffness field, which finely tunes the amount of deformability in the various regions of the images.

Discriminating important voxels We proposed in Section 3.1.2 a method to limit the influence of unreliable voxels in the registration. Once the incremental correction field has been computed, a nonstationary diffusion is used to smooth the incremental corrections that were computed based on irrelevant image voxels. This creates an interpolation effect in those regions of the displacement field. Experiments showed that this improves the results, and also diminishes the computation time. The computation time is low, thanks to the usage of the fast semi-implicit Additive Operator Scheme. This contribution was presented at the MICCAI 2003 conference (Stefanescu et al. (2003a)) and published in Medical Image Analysis (Stefanescu et al. (2004c)).

Registering images with pathologies When attempting to segment real patient images by registering them with an anatomical atlas for radiotherapy planning, we encountered the following problem: the tumor or the possible surgical resection present in the patient image has no correspondent in the atlas. Therefore, false correspondences are estimated for points inside the pathology, which leads to a locally erroneous registration. The solution described in Section 3.3 consists in giving the pathological voxels a low weight in the registration. Results show that this tends to interpolate the displacement field inside the tumor from its values outside it, which prevents potential distortions caused by the pathology. This result was presented at MICCAI 2004 (Stefanescu et al. (2004a)).

Object deformability is position and direction-dependent Usually, pair and smooth algorithms perform isotropic and uniform regularization. We introduced in Chapter 3 an elastic regularization of the displacement field, which is weighted by a "stiffness" field estimating the local deformability of tissues. The regularization is implemented as a nonstationary diffusion equation and solved using the AOS. This realizes a quite good and very fast approximation of an elastic behavior. In Chapter 4, further information is incorporated into the registration. We show how directional (tensor) stiffness information can be embedded in the registration by replacing the isotropic nonstationary regularization with an anisotropic one. This is used in order to maintain the regularity of specific surfaces in atlas to subject registration. This contribution was presented at the MICCAI 2003 conference (Stefanescu et al. (2003a)) and published in Medical Image Analysis (Stefanescu et al. (2004c)).

7.2 Parallel and grid computing for medical image registration

Long computation times may prevent an algorithm from being clinically functional. They become prohibitive if the processing chain in which the algorithm is embedded contains interactivity. Discussions with physicians suggested that a good integration means that: the computation time should be of a few minutes; and it should integrate well into currently existing clinical setups. Moreover, the required hardware should not be more expensive than common medical equipment. In order to facilitate the clinical integration of our registration algorithm, we proposed a parallel implementation which ensures a low execution time. This implementation contains theoretical contributions to parallel computing for image processing. The use of grid computing methods allows a large flexibility concerning the cost and localization of the computing resources.

Parallel computing to reduce computation times Chapter 5 presented a parallel implementation of the registration algorithm on an inexpensive cluster of personal computers. We proposed not only a practical implementation that reduces the computation time to only a few minutes, but also theoretical contributions to parallel computing applied to image processing: our pipeline parallelization strategy of the Gaussian filter and the Additive Operator Schemes can be generalized to other recursive filters, which are usually optimal sequential filters. The results were presented at the HealthGrid 2003 conference (Stefanescu et al. (2003b)) and published in Parallel Processing Letters (Stefanescu et al. (2004b)).

Grid methods to integrate into the clinical setup Despite their low cost, clusters of PC's are still somewhat incompatible with a clinical usage. Administration and maintenance costs may incite hospitals to delocalize and share the computing power. In the same time, registration has to be integrated into currently existing data processing systems, which supposes a total control of the algorithm. Since we are dealing with images, this can only be done through a graphical user interface. By using grid computing methods, Chapter 6 presented a system that enables a visualization workstation in the clinical environment to control the registration software running on a distant cluster and interact with it in real time. The system allows a great flexibility: It has been tested with a cluster hundreds of kilometers away from the user, through a standard network as well as using a home DSL link. The technology was demonstrated at HealthGrid 2004 (Stefanescu et al. (2004d)) and also described in an article published in Methods of Information in Medicine (Stefanescu et al. (2005)).

7.3 Integration into two clinical applications

The registration algorithm was tested for two different clinical applications: the segmentation of high risk organs for the planning of conformal brain radiotherapy, and the localization of the central grey nuclei for the planning of the implantation of electrodes required by deep brain stimulations. Each application was developed in collaboration with one clinical and one industrial partner. In both cases, the algorithm has been (or is about to be) evaluated by clinical experts.

Conformal brain radiotherapy In order to segment high risk structures and avoid their irradiation during the radiotherapy, patient images are registered towards a labeled anatomical atlas. After an initial affine registration, we applied our nonlinear registration algorithm and subsequently deformed the atlas structures towards the patient geometry. Preliminary measures of the specificity and sensitivity of the segmentation produced by our atlas to subject registration algorithm show that these segmentation are of a quality comparable to the one achieved by clinical experts. This quality is furthermore reproducible from one patient to another.

Furthermore, quantitative comparison on a 22 patients database has been performed by an independent clinical expert from the *Centre Antoine Lacassagne*, and our algorithm has been shown to register the brain stem more accurately than both an affine registration and another elastic registration algorithm. As a consequence, our nonlinear registration algorithm was integrated into a prototype of a planning system for conformal brain radiotherapy developed by DosiSoft S.A. The atlas-based segmentation system, including our nonlinear registration algorithm, will soon be clinically validated at the *Institut Gustave Roussy*.

Deep brain stimulation The purpose of the second application is to automatically localize the central grey nuclei, in order to guide the implantation inside them of electrodes linked to a neurostimulator. The central grey nuclei are segmented by registering patient images with an anatomical atlas. After an initial affine registration, the patient and atlas images are nonlinearly registered using the algorithm described in this thesis. The expert segmentations of the atlas are subsequently deformed into the patient's geometry. The grey nuclei being located very near to the highly deformable ventricles, our anisotropic diffusion regularization allows to better retrieve the motion and to obtain a better segmentation. The validation of the system is a future work. We believe that the most clinically significant method to quantify the validity of the method is to compare the predicted positions of the implantation targets with their real positions, determined during the intervention. We expect to perform such a validation in the next months using per-operative data from the La Pitié Salpêtrière Hospital.

7.4 Future work

The research presented in this thesis opens the way to several research paths that need to be explored.

Numerical validation The validation method was mainly application-based, through the visual inspection and quantification of the result (deformed images and atlas structures) by a clinical expert. However, a more general validation would be desirable, in order to better understand the strengths and weaknesses of the algorithm, and compare it with other methods. Since there is little chance to obtain a realistic ground truth by simulating intersubject differences, we intend to validate the algorithm be examining the atlas segmentations deformed into the patients geometry. We are currently testing the Simultaneous Truth and Performance Level Estimation method proposed by Warfield et al. (2004b). Another possible track is the integration of the algorithm into a currently existing validation framework, such as the one proposed by Hellier et al. (2003).

Simultaneous usage of different similarity criteria In our approach, a single similarity criterion (usually the smallest square distance or the local correlation coefficient) is used to register a pair of images. However, numerous other measures have been proposed. While none of them can claim supremacy over the others, some criterion may prove to be more efficient than others in order to register certain structures. For instance, regions in images where intensities correspond very well may be registered with the SSD criterion, while in other areas a multi-modal measure may be necessary.

Multi-channel registration Some structures (such as the brain's central grey nuclei) exhibit a low contrast in some modalities, such as the T1-MRI. However, many clinical protocols include the simultaneous acquisition of different modalities, each of them being relevant for some anatomical structures. A similarity criterion that takes into account not two images, but two sets of images acquired using different modalities may provide more information about lowly contrasted structures. Different similarity criteria may be used for each one of these modalities.

Bringing more clinical expertise into the registration Of course, we would love to be able to report perfect results. However, the quality of registration is not only dependent on the intensities of voxels, but also on the clinical interpretation of the image contents. Concerning the later, human expertise is invaluable. Ideally,

the user should be able to locally correct the registration process, and thus avoid ambiguity and local minima. Eventually, these correction may be "learned" by the algorithm which could subsequently avoid the error. Thanks to its short computation times, the proposed algorithm can potentially be modified in order to be used in an interactive manner.

Long distance displacement correlation In our current regularization method, the displacement of a voxel has an influence only on the displacement of its neighbors, and the amount of influence a voxel can have on others decreases with the distance. It would be interesting to be to be able to impose a kind of correlation between the displacements of voxels that are far away from each other. This would enable to use more general statistical information about the displacement of different regions inside an object.

Building dynamic atlases Hill et al. (2002) proposed to replace traditional brain atlases, aiming to compute average brains representative of large segments of the population, with a dynamic atlas, customized to a specific subject: databases located in different clinical centers provide images coming from subjects which have the same characteristics (age, sex, etc.). The average is then dynamically built based on these selected images. A fast intersubject registration algorithm, such as the one proposed in this thesis may prove very helpful, both for the construction and the usage of such a dynamic atlas.

Appendix A

Numerical implementation

A.1 Solving the diffusion equation in 1-D

In our algorithm, the regularization consists in solving a non-stationary heat equation. In the one-dimension case, this can be done using the semi-implicit scheme $v^{t+\Delta t} = (\mathbf{I_N} - \Delta t A^t)^{-1} v^t$ (where A is the linear operator corresponding to a finite difference discretization, as described in the following section A.2). Since we use finite differences, the value $v_i^{t+\Delta t}$ at each point of index i depends only on $v_{i-1}^{t+\Delta t}$, $v_i^{t+\Delta t}$ and $v_{i+1}^{t+\Delta t}$. Therefore, the matrix A^t is tridiagonal, and so is $\mathbf{I_N} - \Delta t\ A^t$ (see Appendix A.2). The inversion of a tridiagonal matrix can be achieved using the Thomas algorithm (Press et al. (1993)), which consists in a LR decomposition followed by forward and backward substitution steps, as below. For simplicity, we rename the variables in our equation: $B = \mathbf{I_N} - \Delta t\ A^t$, $u = v^{t+\Delta t}$, $d = v^t$. The following first order recursive algorithm operates in linear time:

Given a tridiagonal matrix B, the purpose is to solve the linear system

$$B\ u = d$$

where B is the tri-diagonal matrix

$$B = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \gamma_2 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \gamma_{N-1} & \alpha_{N-1} & \beta_{N-1} \\ & & & \gamma_N & \alpha_N \end{pmatrix} \tag{A.1}$$

The first step is the *LR decomposition* $B = LR$ with L being a lower bidiagonal matrix and R an upper bidiagonal matrix. If we denote the L and R matrices as

Algorithm 6 LR decomposition

$m_1 := \alpha_1$
$r_1 := \beta_1$
for $i := 2, 3, \ldots, N$:
  $l_i := \gamma_i / m_{i-1}$
  $m_i := \alpha_i - l_i \, \beta_{i-1}$
  $r_i := \beta_i$

Algorithm 7 Resolution of a tridiagonal system. Forward substitution.

$y_1 := d_1$
for $i := 2, 3, \ldots, N$:
  $y_i := d_i - l_i \, y_{i-1}$

$$
\begin{aligned}
L &= \begin{pmatrix} 1 & & & \\ l_2 & 1 & & \\ & \ddots & \ddots & \\ & & l_N & 1 \end{pmatrix} \\
R &= \begin{pmatrix} m_1 & r_1 & & \\ & \ddots & \ddots & \\ & & m_{N-1} & r_{N-1} \\ & & & m_N \end{pmatrix}
\end{aligned}
$$

then Algorithm 6 gives the decomposition method. The resolution of the system is done in two steps: a forward (Algorithm 7) and a backward (Algorithm 8) substitution.

A.2 From diffusion to the linear operator in one dimension

If v and d are vectors (one-dimensional arrays) and h is the grid size, the finite difference discretization of the heat equation in a point p of the grid is

Algorithm 8 Resolution of a tridiagonal system. Backward substitution.

$u_N := y_N / m_N$
for $i := N-1, N-2, \ldots, 1$:
  $u_i := \left(y_i - \beta_i \, u_{i+1} \right) / m_i$

$$\nabla(d\nabla v)\,|_p = \nabla d \nabla u + d\frac{\mathbf{d}^2 v}{\mathbf{d}x^2} \approx \frac{d_{p+1}-d_{p-1}}{2h}\cdot\frac{u_{p+1}-u_{p-1}}{2h} + d_p\frac{v_{p+1}+v_{p-1}-2v_p}{h^2}$$

Thus, the corresponding semi-implicit scheme is

$$\begin{aligned}\frac{v_p^{t+1}-v_p^t}{\Delta t} &= \frac{1}{h^2}\left(d_p + \frac{d_{p+1}-d_{p-1}}{4}\right)v_{p+1} + \\ &+\frac{1}{h^2}\left(d_p - \frac{d_{p+1}-d_{p-1}}{4}\right)v_{p-1} - \\ &-\frac{2}{h^2}d_p v_p\end{aligned} \tag{A.2}$$

which leads to an $N \times N$ matrix whose p^{th} line corresponds to point p:

Column	1	$\ldots$	$p-2$	$p-1$	p	$p+1$	$p+2$	$\ldots$	N
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
Value	0	$\ldots$	0	$\frac{d_p+\frac{d_{p+1}-d_{p-1}}{4}}{h^2}$	$-2\frac{d_p}{h^2}$	$\frac{d_p-\frac{d_{p+1}-d_{p-1}}{4}}{h^2}$	0	$\ldots$	0

This amounts to the following values for the α, β and γ elements from Eq. A.1:

$$\begin{aligned}\alpha_p &= 1 + 2\,\Delta t\,\frac{d_p}{h^2} \\ \beta_p &= -\Delta t\,\frac{d_p - \frac{d_{p+1}-d_{p-1}}{4}}{h^2} \\ \gamma_p &= -\Delta t\,\frac{d_p + \frac{d_{p+1}-d_{p-1}}{4}}{h^2}\end{aligned}$$

Remark If the stiffness field d is relatively smooth smooth ($\|(d_{p+1}-d_{p-1})/4\| \ll \|d_p\|$), and the above numerical scheme approximates the one of the Laplacian $\frac{\partial v}{\partial t} = d\Delta v$:

$$\frac{v_p^{t+1}-v_p^t}{\Delta t} = \frac{1}{h^2}\left(d_p v_{p+1} + d_p v_{p-1} - 2d_p u_p\right)$$

This approximation stands in regions where d is relatively large. If we give d large values in areas that are "anatomically relevant" and low ones in less interesting regions, we can use this approximation. For historical reasons, this approximation is currently used in the current version of the registration software. In the rest of this appendix, we will also use this simplifying approximation.

Practical implementation In practice, the resolution of the tridiagonal linear system on a single processor computer was performed using the LAPACK (Anderson et al. (1990)) routine `dgtsv`.

A.3 Boundary conditions

In this section, we also address the case where the derivatives of the stiffness field (∇d) are small w.r.t. to the values of the field (d). There are two special cases, at the first and at the last point. They correspond to the border conditions. When the diffusion equation is applied to a displacement or correction field, we are mainly interested by two situations:

1) Dirichlet boundary conditions

The image objects are "attached" to the borders of the image. Numerically, this amounts to extending with zeros the displacement/correction field outside the image support. This way, the diffusion ensures that the values of the field next to the borders are always very close to 0. For the first point, the discretization from Eq. A.2 becomes

$$\frac{v_1^{t+1} - v_1^t}{\Delta t} = \frac{d_1}{h^2}\, 0 - 2\,\frac{d_1}{h^2}\, v_1^{t+1} + \frac{d_1}{h^2}\, v_2^{t+1}$$

For the last point, the discrete scheme becomes

$$\frac{v_N^{t+1} - v_N^t}{\Delta t} = \frac{d_N}{h^2}\, v_{N-1}^{t+1} - 2\,\frac{d_N}{h^2}\, v_N^{t+1} + \frac{d_N}{h^2}\, 0$$

This leads to the following values for the first and last line of the matrix B in Eq. A.1:

$$\begin{aligned} \alpha_1 &= -2\,\frac{d_1}{h^2} \\ \beta_1 &= \frac{d_1}{h^2} \\ \alpha_N &= -2\,\frac{d_N}{h^2} \\ \gamma_N &= \frac{d_N}{h^2} \end{aligned}$$

2) Neumann boundary conditions

The imaged objects can move freely in the plane tangent to the boundary surface. In order to achieve this effect, we extend the field outside its domain with the value of the closest point on the border. For the first point, this leads to

$$\frac{v_1^{t+1} - v_1^t}{\Delta t} = \frac{d_1}{h^2}\, v_1^{t+1} - 2\,\frac{d_1}{h^2}\, v_1^{t+1} + \frac{d_1}{h^2}\, v_2^{t+1}$$

whereas for the last point the discrete scheme becomes

$$\frac{v_N^{t+1} - v_N^t}{\Delta t} = \frac{d_N}{h^2}\, v_{N-1}^{t+1} - 2\,\frac{d_N}{h^2}\, v_N^{t+1} + \frac{d_N}{h^2}\, v_N^{t+1}$$

By summing in the first equation the two terms containing v_1^{t+1}, and in the second equation the two terms containing v_N^{t+1}, we get:

$$\begin{aligned} \alpha_1 &= -\frac{d_1}{h^2} \\ \beta_1 &= \frac{d_1}{h^2} \\ \alpha_N &= -\frac{d_N}{h^2} \\ \gamma_N &= \frac{d_N}{h^2} \end{aligned}$$

A.4 Implementing the AOS in three dimensions

By neglecting the derivatives of d, the diffusion equation in 3D

$$\frac{\partial v}{\partial t} = \text{div}(d\,\nabla v)$$

becomes

$$\frac{\partial v}{\partial t} = d\,\Delta v = d\,\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right)$$

By denoting with $v[x, y, z]$ the value of the volume v at point (x, y, z), the discretization of the above equation is

$$\begin{aligned}\frac{v^{t+1}[x,y,z]-v^t[x,y,z]}{\Delta t} &= d[x,y,z]\left(\frac{v^t[x+1,y,z]-2v^t[x,y,z]+v^t[x-1,y,z]}{h_x^2}+\right.\\ &+ \frac{v^t[x,y+1,z]-2v^t[x,y,z]+v^t[x,y-1,z]}{h_y^2}+\\ &+ \left.\frac{v^t[x,y,z+1]-2v^t[x,y,z]+v^t[x,y,z-1]}{h_z^2}\right)\end{aligned}$$

In Section A.2 we have seen that the Additive Operator Scheme consists in separating the linear operator above along the directions x, y, and z:

$$\begin{aligned}\frac{v^{t+1}[x,y,z]-v^t[x,y,z]}{\Delta t} &= d[x,y,z]\left(\frac{v^t[x+1,y,z]-2v^t[x,y,z]+v^t[x-1,y,z]}{h_x^2}\right)+\\ &+ d[x,y,z]\left(\frac{v^t[x,y+1,z]-2v^t[x,y,z]+v^t[x,y-1,z]}{h_y^2}\right)+\\ &+ d[x,y,z]\left(\frac{v^t[x,y,z+1]-2v^t[x,y,z]+v^t[x,y,z-1]}{h_z^2}\right)\end{aligned}$$

Therefore, the three operators A_x, A_y and A_z (along the x, y, and z directions) correspond, respectively, to the following one-dimensional schemes:

$$\begin{aligned}A_x: \quad \frac{v^{t+1}[x,y,z]-v^t[x,y,z]}{\Delta t} &= \frac{d[x,y,z]}{h_x^2}\left(v^t[x+1,y,z]-2v^t[x,y,z]+v^t[x-1,y,z]\right)\\ A_y: \quad \frac{v^{t+1}[x,y,z]-v^t[x,y,z]}{\Delta t} &= \frac{d[x,y,z]}{h_y^2}\left(v^t[x,y+1,z]-2v^t[x,y,z]+v^t[x,y-1,z]\right)\\ A_z: \quad \frac{v^{t+1}[x,y,z]-v^t[x,y,z]}{\Delta t} &= \frac{d[x,y,z]}{h_z^2}\left(v^t[x,y,z+1]-2v^t[x,y,z]+v^t[x,y,z-1]\right)\end{aligned}$$

The inversion algorithm for tridiagonal matrices described in Sections A.1 and A.2 is applied in order to separately solve the three tridiagonal systems.

Appendix B

Segmentation of pathology in tumor-diseased brains

B.1 Segmentation of a surgical resection

A surgical resection corresponds to an absence of matter in the considered region, filled with CSF, and possibly connected with the ventricles. Its shape is more spherical than the other structures of the CSF, and is composed of only one big connected component. These are the basic properties that we exploit for delineating the resection.

First, we extract all structures behaving like CSF in the joint MR T1 and T2 histogram (low signal in T1 and high signal in T2) by fitting a 2D Gaussian on the corresponding area of the histogram. Selecting all the voxels whose joint intensity is statistically compatible gives us an oversized segmentation of CSF which still contains structures like the eyes and the ventricles. The eyes are quite easy to remove since they appear as two isolated connected components. To select them, we robustly register an atlas with an affine transformation, and remove the connected components that have an intersection with the eyes of the atlas. To separate the ventricles from the surgical resection, we use a region labeling algorithm based on a skeletonization by influence zone (SKIZ) Soille (1999). As this labeling is sensitive to narrowings in a connected component, it easily classifies the surgical resection and the ventricle as different regions. The regions that intersect the ventricles of the atlas are removed as above.

Finally, we have to select the surgical resection region among remaining structures. The sulci are relatively small with respect to a surgical resection and thus easy to remove. The main problem comes from the possible presence of a CSF component

between the brain and the skull due to brain shift during the surgical operation. The volume of this component may be quite large, but its shape is mostly flat. Thus, we compute a distance map in each remaining CSF connected component, and select the one that has the largest inscribed ball radius.

B.2 Delineation of the tumor

Delineating a tumor is a hard task due to the multiple appearances it may have in the image. The tumor may generate an edema at its frontiers, and contain a necrotic center. The tumor tissues and the edema usually appear like partial volume (CSF and grey matter) intensities, while the necrosis resembles the CSF.

Traditional Expectation-Maximization algorithms (Leemput et al. (1999)) fail to provide good results because of the presence of these tissues. An alternative is to consider tumor intensities as outliers in this mixture of Gaussians, or to add some specific classes to model the tumor and edema intensities (Moon et al. (2002)). As this was often not sufficient, some anatomical knowledge was added, either by combining geometric priors given by the non-rigid registration of an atlas to a tissue classification (Kaus et al. (2001)), or by using Markov Random Fields (Kapur (1999)). Other methods include region growing from a region of interest delineated by one of the preceding methods using level-sets methods (Ho et al. (2002)).

All these methods end up in very complex algorithm as they attempt to segment all the tissues. In our case, we are only interested in the tumor segmentation, so that we could rely on a very simple mathematical morphology scheme as we developed in the previous section.

We fit this time a mixture of two Gaussians to the selected region of the joint T1 an T2 intensity histogram: one for the necrotic part of the tumor (which appear like CSF), and a second one for the tumor tissues and its edema (resembling partial volume CSF/grey matter). We obtain an oversized segmentation where we need to remove structures like the sulci or the ventricles without removing interesting parts. Indeed, we now have CSF and grey matter partial volume voxels, and the necrotic part of the tumor can be near a region containing CSF. The ventricles and the eyes are removed like before. Then the remaining part of the segmentation is labeled into SKIZ zones. Each region is then compared with an a priori statistical atlas of the CSF to compute the mean probability of belonging to the CSF. A threshold on this probability allows us to remove the CSF structures like the ventricles or the sulci. In each of these two steps we also compute a distance map to the CSF of the statistical atlas in each region to avoid removing regions containing voxels too far from the expected CSF.

Appendix C

ClusterConnect: a library for controlling a parallel program from a graphics interface

We have grouped our software interface allowing to coordinate a graphics interface running on a visualization workstation to control a parallel software running on a distantly located cluster into a library called ClusterConnect.

C.1 Structure

Messages, channels, message tags, collective operations

As stated above, the purpose of this library is the transmission of messages between a visualization workstation (VW) and the nodes of a cluster. The communication should fulfill several conditions:

- It must be fast in order to allow numerous large messages.
- It must be secure, since messages may potentially pass through an unsecured wide area network (WAN).
- Since we want to allow long-distance communications, messages must be able to pass through firewalls.
- The software interface should be as simple as possible.

C.2 Software architecture

The main mission of the library is the ability to send a message through a communication channel. Hence, the two main notions, represented by the C++ classes Message and CommChannel (communication channel).

C.2.1 Communication channels

Communication channels are able to send and receive primitive objects, such as numbers. This means that 1) they have to manage the "low level" interface of communication: sending and receiving a flow of bytes stored in a data buffer; 2) they provide a method to encode numbers into the buffer.

Class hierarchy

Figure C.1 shows the class hierarchy of implemented communication channels. The base class `CommChannel` serves as a common interface between them. It defines the following methods:

- `packInt` and `packFloat`: encode a sequence of integers or reals into the communication buffer.
- `unpackInt` and `unpackFloat`: decode a sequence of integers or reals from the buffer.
- `sizeInt` and `sizeFloat`: return the size of the buffer required to encode a given number of integers or reals. Depending on the encoding method, this size can vary from the one occupied by the numbers in the memory of the computer.
- `send` and `receive`: transmission of the data buffer.

Remark The `unpackInt` and `unpackFloat` functions need to now in advance the number of numbers they have to recover. Thus, the transmission of a vector occurs as:

```
// communicating an integer vector vec of size n
Sender:
```

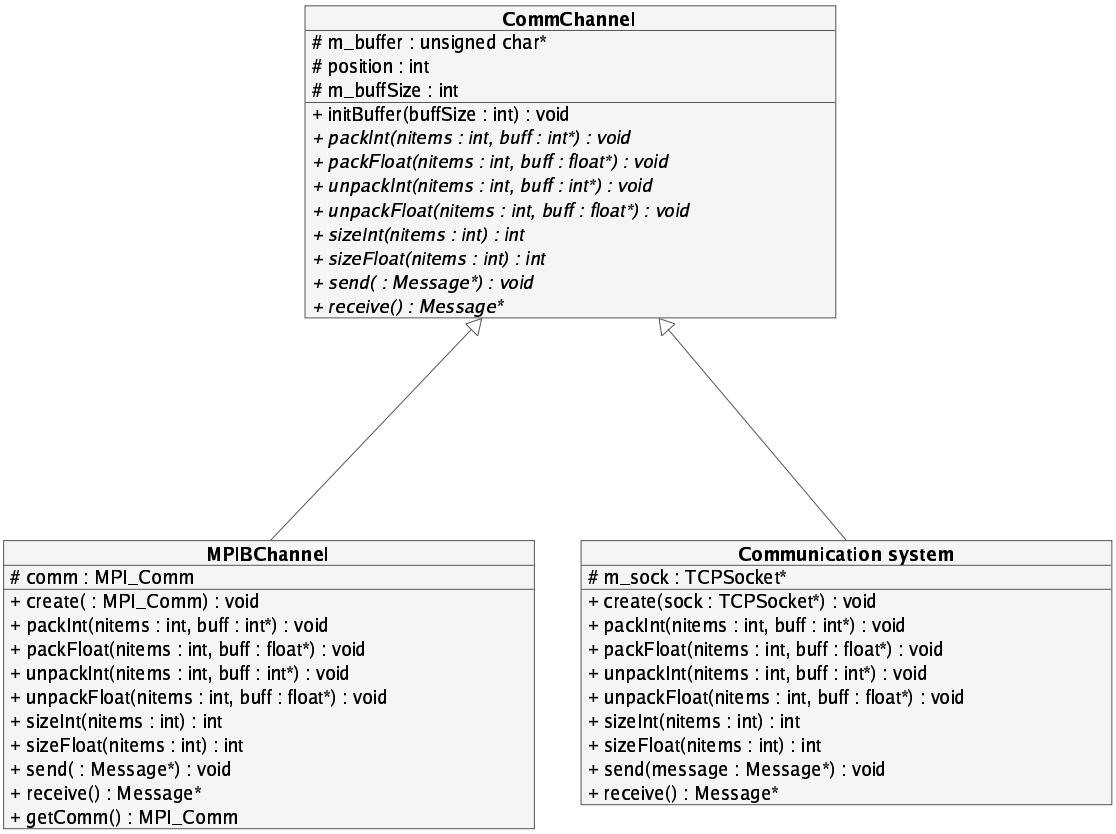


Figure C.1: Class diagram for the channel hierarchy.

```
        packInt(1, n)
        packInt(n, vec)
    Receiver:
        unpackInt(1, n)
        vec = allocate_vector_of_size(n)
        unpackInt(n, vec)
```

The MPI broadcast channel This channel allows the master node to broadcast a message to the others. The communication itself is done using the MPI function `MPI_Bcast`, which enables the broadcast of simple objects, such as numbers.

The TCP/IP channel It communicates the data buffer through TCP/IP sockets. Since this channel may have to send data through slow links, the information is first

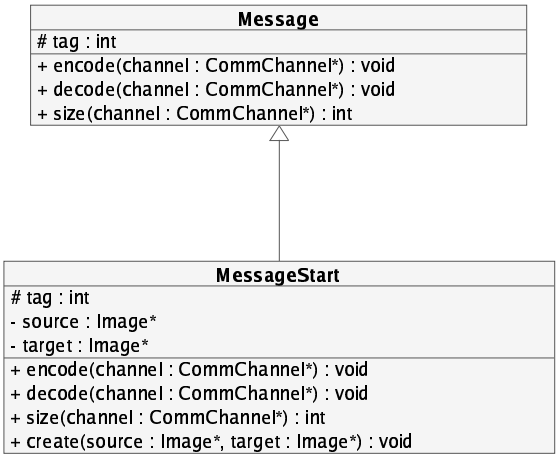


Figure C.2: Class diagram for the message hierarchy.

compressed before sending, and transparently decompressed at destination. The compression/decompression is performed using the zlib library[1], which employs an algorithm derived from Ziv and Lempel (1977).

C.2.2 Messages

A message is an object containing the data that needs to be transmitted. In general, its contents consists of integer or real numbers or vectors. The base class `Message` provides a common interface for all messages. A message also contains the algorithm that packs (function `encode`) or unpacks (function `decode`) it from a communication channel's data buffer. In order to pack or unpack itself to/from the channel's buffer, it uses the desired channel's `packInt`, `packFloat`, `unpackInt` and `unpackFloat` methods. Using the channel's `sizeInt` and `sizeFloat` functions, the `size` function reports the size of the data buffer needed to encode the message.

Other message classes are derived from the base class. Figure C.2 shows the example of `MessageStart`, used by the GUI to request de beginning of a registration job. The data it carries consists (in a simplified vision) in the two input images for the registration (source and target). The GUI creates one such message, passing it the two images. The `encode`, `decode` and `size` functions are appropriately implemented in order to carry the two images.

[1] http://www.gzip.org/

Algorithm 9 Receiving a message

1. receive an integer n=the size of the message
2. receive a buffer buf of size n (in bytes)
3. receive the tag of the message, representing its type
4. use the $MessageFactory$ to build an object m of the type corresponding to the tag
5. use the $MessageHandlerFactory$ to build an appropriate $MessageHandler$ for the message tag
6. invoke the $handle$ method of the $MessageHandler$, and pass it the message m as a parameter

C.3 Receiving a message

The difficulty in receiving a message consists in the fact that not all messages contain the same data (see Algorithm 9). Each class representing a message has a static integer tag. Identifying a message at its receipt is done through its tag, which is sent through the communication channel prior to the message itself. Upon receipt of the tag and a data buffer containing the packed message, a message factory is invoked to create a message object of the type given by the tag. It then calls the `decode` method of the message which reads the data from the buffer. The message handler is then invoked.

C.4 GUI to cluster communication

Due to the possible isolation of the visualization workstation from the cluster by a firewall[2], the GUI cannot directly broadcast a message to all the nodes of the cluster: it can only communicate with the master node. Thus, the message has to be sent first to the master node, which forwards it to the rest of the cluster. The system is further complexified by the fact that the communications between, first, the GUI and the master, and second, the master and the cluster, do not use the same communication protocol.

The algorithm is described in Figure C.3. A message of an appropriate type is first created and filled with appropriate data by the GUI (step 1) and a “send” order is

[2]This situation is present in our test system, where a firewall filters the network traffic between the visualization workstation and the computers of the cluster.

issued (step 2). The TCP/IP channel (class `TCPChannel`) is used to pack the message into a buffer (step 3), which is communicated (along with the message tag) to the master node (step 4) through the TCP/IP channel. Upon its receipt, the message is unpacked by the master (step 5) as described in Section C.3. Since the message will be forwarded to the other nodes using an MPI broadcast primitive (instead of TCP/IP), it is repacked using the MPI broadcast channel (class `MPIBChannel`) (step 6) and sent (step 7). At the end, all nodes receive the message which they decode as in Section C.3 (step 8).

The algorithm has been described here using the TCP/IP protocol to communicate between the visualization workstation and the master node. However, the architecture is not dependent on the usage of TCP/IP, and this protocol can be replaced with a higher performance one. Valid alternatives are Globus I/O and HTTP, but we didn't test either of them.

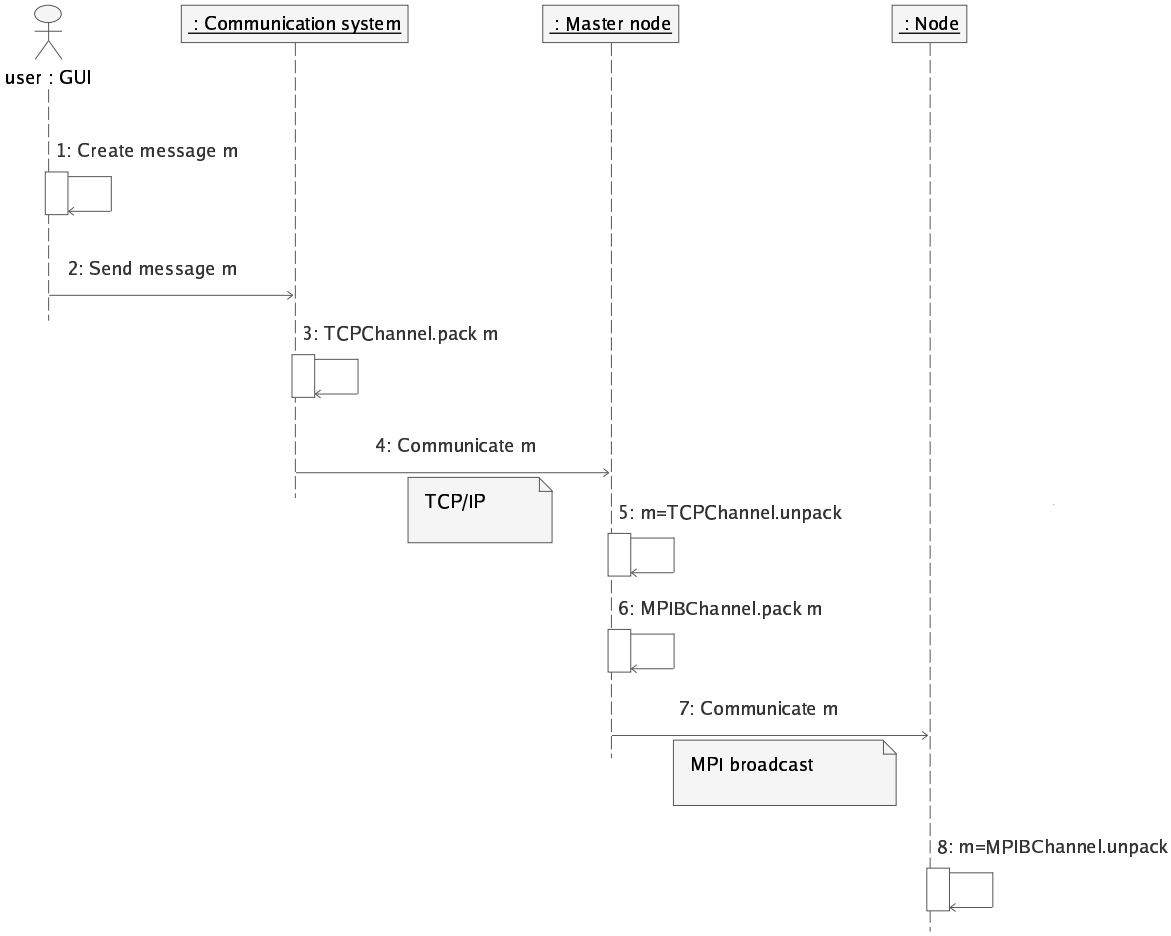


Figure C.3: The GUI broadcasts a message to all the processors of the cluster. The step number is indicated on the left side of the step description.

Bibliography

Alvarez, L., Weickert, J., Sánchez, J., 2000. Reliable estimation of dense optical flow fields with large displacements. International Journal of Computer Vision 39 (1), 41–56.

Anderson, E., Bai, Z., Bischof, C., Demmel, J., Dongarra, J., DuCroz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D., 1990. LAPACK: A Portable Linear Algebra Library for High-Performance Computers. Tech. Rep. UT-CS-90-105, University of Tennessee.
URL http://www.netlib.org/lapack/lawns/lawn20.ps

Arsigny, V., Pennec, X., Ayache, N., 2003. Polyrigid and polyaffine transformations: A new class of diffeomorphisms for locally rigid or affine registration. In: Ellis, R. E., Peters, T. M. (Eds.), Proceedings of Medical Image Computing and Computer Assisted Intervention. Vol. 2879 of LNCS. Springer Verlag, Montreal, pp. 829–837.

BachCuadra, M., Pollo, C., Bardera, A., Cuisenaire, O., Villemure, J.-G., Thiran, J.-P., 2004. Atlas-based segmentation of pathological mr brain images using a model of lesion growth. IEEE Transactions on Medical Imaging 23 (10), 1301–1314.

Bajcsy, R., Kovačič, S., 1989. Multiresolution elastic matching. Computer Vision, Graphics and Image Processing 46, 1–21.

Bajscy, R., Broit, C., 1982. Matching of deformed images. In: Proc. of 6th International Conference on Pattern Recognition.

Barillot, C., Valabrègue, R., Matsumoto, J., Aubry, F., Benali, H., Cointepas, Y., Dameron, O., Dojat, M., Duchesnay, E., Gibaud, B., Kinkingne'hun, S., Papadopoulos, D., Pélégrini-Issac, M., Simon, E., 2004. NeuroBase: Management of Distributed and Heterogeneous Information Sources in Neuroimaging. In: Dojat, M., Gibaud, M. (Eds.), DiDaMIC'04 Workshop. Satellite of the MICCAI conference (St-Malo, 26-29 September 2004). Rennes, pp. 85–94.
URL http://www.irisa.fr/visages/demo/Neurobase/Didamic/index.html

Bezdek, J., 1981. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York.

Bondiau, P., Malandain, G., Commowick, O., Marcy, P., Chanalet, S., Ayache, N., 2004. Atlas-based automatic segmentation of mr images: Validation study on the brainstem in radiotherapy. In: 90th Scientific Assembly and Annual Meeting of the Radiological Society of North America (RSNA). Chicago.

Bookstein, F., 1989. Principal warps: Thin-plate splines and the decomposition of deformations. IEEE Transactions on Pattern Analysis and Machine Intellingence 11 (6), 567–585.

Bricault, I., Ferretti, G., Cinquin, P., 1998. Registration of real and CT-derived virtual bronchoscopic imag es to assist transbronchial biopsy. Transactions in Medical Imaging 17 (5), 703–714.

Bro-Nielsen, M., 1996. Medical image registration and surgery simulation. Ph.D. thesis, IMM-DTU.

Bro-Nielsen, M., 1997. Rigid registration of CT, MR and cryosection images using a GLCM framework. In: Proceedings of CVRMed/MRCAS'97. Vol. 1205 of LNCS. Springer, pp. 171–180.

Bro-Nielsen, M., Gramkow, C., 1996. Fast fluid registration of medical images. In: Proceedings of Visualization in Biomedical Computing (VBC'96). Vol. 1131 of LNCS. pp. 267–276.

Broit, C., 1981. Optimal registration of deformed images. Ph.D. thesis, University of Pennsylvania.

Bruhn, A., Jacob, T., Fischer, M., Kohlberger, T., Weickert, J., Brüning, U., Schnörr, C., 2002. Designing 3d nonlinear diffusion filters for high performance cluster computing. In: LNCS. Vol. 2449. pp. 396–399.

Burns, M., Rowland, A., Rueckert, D., Hajnal, J., Hill, D., 2004a. A grid infrastructure for image registration and segmentation. In: Proceedings of the UK e-Science All Hands Meeting 2004, 31st August - 3rd September, Nottingham UK.
URL http://www.allhands.org.uk/2004/proceedings/papers/268.pdf

Burns, M., Rowland, A. L., Rueckert, D., Hajnal, J., Leung, K., Hill, D., Vickers, J., 2004b. Information extraction from images (ixi) grid services for medical imaging. In: Proc of the Workshop on DIstributed DAtabases and processing » in Medical Image Computing (DIDAMIC'04), September 30th, IRISA, Rennes.
URL http://www.irisa.fr/visages/demo/Neurobase/Didamic/index.html

Cachier, P., 2002. Recalage non rigide d'images medicales volumiques - contribution aux approches iconiques et geometriques. Ph.D. thesis, Ecole Centrale des Arts et Manufactures.

Cachier, P., Bardinet, E., Dormont, D., Pennec, X., Ayache, N., 2003. Iconic Feature Based Nonrigid Registration: The PASHA Algorithm. CVIU — Special Issue on Nonrigid Registration 89 (2–3), 272–298.

Cachier, P., Mangin, J.-F., Pennec, X., Rivière, D., Papadopoulos-Orfanos, D., Régis, J., Ayache, N., 2001. Multisubject Non-Rigid Registration of Brain MRI using Intensity and Geometric Features. In: Niessen, W., Viergever, M. (Eds.), 4th Int. Conf. on Medical Image Computing and Computer-Assisted Intervention (MICCAI'01). Vol. 2208 of LNCS. Utrecht, The Netherlands, pp. 734–742.

Cachier, P., Pennec, X., 2000. 3D non-rigid registration by gradient descent on a gaussian-windowed similarity measure using convolutions. In: Proc. of IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA'00). IEEE Computer society, Hilton Head Island, South Carolina, USA, pp. 182–189.

Chefd'hotel, C., Hermosillo, G., Faugeras, O., 2002. Flows of diffeomorphisms for multimodal image registration. In: Proceedings of IEEE International Symposium on Biomedical Imaging. IEEE Computer Society, pp. 753–756.

Christensen, G., Joshi, S., Miller, M., 1994a. 3D brain mapping using a deformable neuroanatomy. Physics in Medicine and Biology 39 (609–618).

Christensen, G., Miller, M., Vannier, M., 1994b. A 3D deformable magnetic resonance textbook based on elasticity. In: Proceedings of the American Association for Artificial Inteligence, Symposium: Applications of Computer Vision in Medical Image Processing.

Christensen, G., Rabitt, R., Miller, M., 1996. Deformable templates using large deformation kinetics. IEEE Transactions on Image Processing 5 (10), 1435–1447.

Christensen, G. E., 1999. Consistent linear-elastic transformations for image matching. In: IPMI '99: Proceedings of the 16th International Conference on Information Processing in Medical Imaging. Springer-Verlag, London, UK, pp. 224–237.

Christensen, G. E., Johnson, H. J., 2001. Consistent image registration. IEEE Transactions on Medical Imaging 20 (7).

Chui, H., Rambo, J., Duncan, J., Schultz, R., Rangarajan, A., 1999. Registration of cortical anatomical structures via robust 3D point matching. In: Proc. Information Processing in Medical Imaging. Vol. 1613 of Lecture Notes in Computer Science. Springer.

Clatz, O., Delingette, H., Bardinet, E., Dormont, D., Ayache, N., 2003. Patient specific biomechanical model of the brain: Application to parkinson's disease procedure. In: Ayache, N., Delingette, H. (Eds.), International Symposium on

Surgery Simulation and Soft Tissue Modeling (IS4TM'03). Vol. 2673 of Lecture Notes in Computer Science. INRIA Sophia Antipolis, Springer-Verlag, Juan-les-Pins, France, pp. 321–331.

Cohen, L., 1996. Auxiliary variables and two-step iterative algorithms in computer vision problems. Journal of Mathematical Imaging and Vision 6 (1), 61–86.

Cohen, L., Cohen, I., 1991. Finite element methods for active contour models and balloons for 2d and 3d images. IEEE Tranactions on Pattern Analysis and Machine Intelligence 15 (11), 1131–1147.

Collignon, A., 1998. Multi-modality medical image registration by maximization of mutual information. Ph.D. thesis, Catholic University of Leuven, Leuven, Belgium.

Collignon, A., Maes, F., Delaere, D., Vandermeulen, D., Suetens, P., Marchal, G., 1995. Automated multimodality image registration using information theory. In: Bizais, Y., Barillot, C. (Eds.), Proceedings of Information Processing in Medical Imaging. Kluwer Academic Publishers, pp. 263–274.

Collignon, A., Vandermeulen, D., Suetens, P., Marchal, G., Baert, A., Oosterlinck, A., 1994. Automatic registration of 3D images of the brain based on fuzzy objects. In: Loew, M. H. (Ed.), Medical Imaging: Image Processing. Vol. 2167. SPIE Press, pp. 162–175.

Collins, D., Evans, A., 1997. Animal: Validation and applications of nonlinear registration-based segmentation. International Journal of Pattern Recognition and Artificial Intelligence 11 (8), 1271–1294.

Collins, D., LeGoualher, G., Evans, A., 1998. Non-linear cerebral registration with sulcal constraints. In: Proceedings of Medical Image Computing and Computer Assisted Intervention. Vol. 1496 of LNCS. Springer, pp. 974–984.

Collins, D. L., Goualher, G. L., Venugopal, R., Caramanos, A., Evans, A. C., Barillot, C., 1996. Cortical constraints for non-linear cortical registration. In: VBC '96: Proceedings of the 4th International Conference on Visualization in Biomedical Computing. Springer-Verlag, London, UK, pp. 307–316.

Commowick, O., 2003. Utilisation d'atlas anatomiques numériques pour la cancérologie. Master's thesis, DEA Mathématiques, Vision, Apprentissage, École Normale Supérieure Cachan.

D'Agostino, E., Maes, F., Vandermeulen, D., Suetens, P., 2003. A viscous fluid model for multimodal non-rigid image registration using mutual information. Medical Image Analysis 7 (4), 565–575.

Dawant, B., Hartmann, S., Pan, S., Gadamsetty, S., 2002. Brain atlas deformation in the presence of small and large space-occupying tumors. Computer Aided Surgery 7, 1–10.

deGruijter, J., McBratney, A., 1988. Classification and Related Methods of Data Analysis. Elsevier Science, Amsterdam, Ch. A modified fuzzy k means for predictive classification, pp. 97–104.

Denton, E., Holden, M., Christ, E., Jarosz, J., Russell-Jones, D., Goodey, J., Cox, T., Hill, D., 2000. The identification of cerebral volume changes in treated growth hormone-deficient adults using serial 3D MR image processing. Journal of Computer Assisted Tomography 24 (1), 139–145.

Deriche, R., 1992. Recursively implementing the gaussian and its derivatives. In: Proc. Second International Conference On Image Processing. pp. 263–267.

Dornier, C., Ivancevic, M., Thévenaz, P., Unser, M., Vallée, J., July 10-16, 2003. Accurate MR cardiac perfusion analysis by using a multiresolution B-splines registration algorithm. In: Proceedings of the Eleventh Scientific Meeting of the International Society for Magnetic Resonance in Medicine (ISMRM'03). Toronto ON, Canada, p. 700.

Dugas-Phocion, G., González Ballester, M. A., Malandain, G., Lebrun, C., Ayache, N., 2004. Improved em-based tissue segmentation and partial volume effect quantification in multi-sequence brain MRI. In: Proc. of MICCAI'04. Lecture Notes in Computer Science. Springer, Saint-Malo, France.

Evans, A., Dai, W., Collins, L., Neelin, P., Marrett, S., 1991. Warping of a computerized 3-D atlas to match brain image volumes for quantitative neuroanatomical and functional analysis. In: Loew, M. (Ed.), Proceedings of the International Society of Optical Engineering: Medical Imaging V. Vol. 1445 of SPIE. pp. 236–246.

Ferrant, M., Nabavi, A., Macq, B., Black, P., Jolesz, F., Kikinis, R., Warfield, S., 2002. Serial registration of intraoperative MR images of the brain. Medical Image Analysis 6 (4), 337–359.

Ferrant, M., Warfield, S., Nabavi, A., Jolesz, F., Kikinis, R., 2000. Registration of 3D intraoperative mr images of the brain using a finite element biomechanical model. In: Proceedings of Medical Image Computing and Computer Assisted Intervention. Vol. 1935 of LNCS. pp. 19–28.

Fischer, B., Modersitzki, J., 1999. Fast inversion of matrices arising in image processing. Numerical Algorithms 22, 1–11.

Foster, I., Kesselman, C., Nick, J., Tuecke, S., 2002. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum.

Ganser, K., Dickhaus, H., Metzner, R., Wirtz, C., 2004. A deformable digital brain atlas system according to Talairach and Tournoux. Medical Image Analysis 8, 3–22.

Gee, J., Reivich, M., Bajcsy, R., 1993. Elastically deforming 3D atlas to match anatomical brain images. Journal of Computer-Assisted Tomography 17 (2).

Guimond, A., Roche, A., Ayache, N., Meunier, J., 2001a. Multimodal Brain Warping Using the Demons Algorithm and Adaptative Intensity Corrections. IEEE Transaction on Medical Imaging 20 (1), 58–69.

Guimond, A., Roche, A., Ayache, N., Meunier, J., 2001b. Multimodal Brain Warping Using the Demons Algorithm and Adaptative Intensity Corrections. IEEE Transaction on Medical Imaging 20 (1), 58–69.

Hajnal, J. V., Hill, D. L. G., Hawkes, D. J. (Eds.), 2001. Medical Image Registration. CRC Press.

Hellier, P., Barillot, C., 2001. Cooperation between local and global approaches to register brain images. In: IPMI '01: Proceedings of the 17th International Conference on Information Processing in Medical Imaging. Springer-Verlag, London, UK, pp. 315–328.

Hellier, P., Barillot, C., February 2003. Coupling dense and landmark-based approaches for non rigid registration. IEEE Transactions on Medical Imaging 22 (2), 217–227.

Hellier, P., Barillot, C., Corouge, I., Gibaud, B., Le Goualher, G., Collins, D., Evans, A., Malandain, G., Ayache, N., Christensen, G., Johnson, H., 2003. Retrospective evaluation of inter-subject brain registration. IEEE Transactions on Medical Imaging 22 (9), 1120–1130.

Hellier, P., Barillot, C., Mémin, E., Pérez, P., 2001. Hierarchical estimation of a dense deformation field for 3-d robust registration. IEEE Transactions on Medical Imaging 20 (5), 388–402.

Hermosillo, G., Chefd'hotel, C., Faugeras, O., 2002. Variational methods in multimodal image matching. IJCV 50(3), 329–343.

Herveg, J., Poullet, Y., 2003. Directive 95/46 and the Use of Grid Technologies in the Healthcare Sector: Selected Legal Issues. In: Norager, S. (Ed.), Proc. of HealthGrid'03. European Commission, DG Information Society, Lyon.

Hill, D. L. G., Hajnal, J. V., Rueckert, D., Smith, S. M., Hartkens, T., McLeish, K., 2002. A dynamic brain atlas. In: Dohi, T., Kikinis, R. (Eds.), Proc. of Fifth Int. Conf. on Medical Image Computing and Computer-Assisted Intervention (MICCAI '02). Vol. 2488 of LNCS. Springer, Tokyo, pp. 532–539.

Hill, D. L. G., Studholme, C., Hawkes, D. J., 1994. Voxel similarity measures for automated image registration. In: Robb, R. A. (Ed.), Proceedings of Visualization in Biomedical Computing. Vol. 2359. SPIE Press, pp. 205–216.

Ho, S., Bullitt, E., Gerig, G., 2002. Level set evolution with region competition: Automatic 3-d segmentation of brain tumors. In: Proc. 16th Int Conf on Pattern Recognition ICPR 2002. pp. 532–535.

Horn, B. K. P., Schunk, B. G., 1981. Determining optical flow. Artificial Intelligence 17, 185–203.

Hufnagel, H., Pennec, X., Malandain, G., Handels, H., Ayache, N., 2005. Nonlinear 2d and 3d registration using block-matching and b-splines. In: Bildverarbeitung fuer die Medizin 2005. Deutsches Krebsforschungszentrum, Heidelberg, Germany.

Ino, F., Ooyama, K., Kawasaki, Y., Takeuchi, A., Mizutani, Y., Masumoto, J., Sato, Y., Sugano, N., Nishii, T., Miki, H., Yoshikawa, H., Yonenobu, K., Tamura, S., Ochi, T., Hagihara, K., 2003. A high performance computing service over the internet for nonrigid image registration. In: Elsevier Science (Ed.), Proceedings of Computer Assisted Radiology and Surgery 17th International Congress and Exhibition (CARS 2003). Vol. 1256 of International Congress Series. pp. 193–199.

Johnson, H. J., Christensen, G. E., 2002. Consistent landmark and intensity-based image registration. IEEE Trans. Medical Imaging 5 (21).

Kapur, T., 1999. Model based three dimensional medical image segmentation. Ph.D Thesis, Massachusetts Institute of Technology.

Kaus, M. R., Warfield, S. K., Nabavi, A., Black, P. M., Jolesz, F. A., Kikinis, R., 2001. Automated segmentation of mr images of brain tumors. Radiology 218 (2), 586–591.

Keriven, R., Faugeras, O., 1998. Variational priciples, surface evolution, PDE's, level-set methods and stereo problem. IEEE Transactions on Medical Imaging 7 (3), 336–344.

Kyriacou, S., Davatzikos, C., Zinreich, S., Bryan, R., 1999. Nonlinear elastic registration of brain images with tumor pathology using a biomechanical model. IEEE Trans. Med. Imaging 18 (7), 580–592.

Le Goualher, G., Barillot, C., Bizais, Y., 1997. Three-dimensional segmentation and representation of cortical sulci using active ribbons. International Journal of Pattern Recognition and Artificial Intelligence 11 (8), 1295–1315.

Leemput, K. V., Maes, F., Vandermeulen, D., Suetens, P., 1999. Automated model-based tissue classification of MR images of the brain. IEEE transactions on medical imaging 18 (10), 897–908.

Lehmann, T. M., Gonner, C., Spitzer, K., 1999. Survey: interpolation methods in medical image processing. IEEE Transactions on Medical Imaging 18 (11), 1049–1075.

Lester, H., Arridge, S. R., Jansons, K. M., Lemieux, L., Hajnal, J. V., Oatridge, A., 1999. Non-linear registration with the variable viscosity fluid algorithm. In: Proceedings of IPMI. pp. 238–251.

Liu, T., Shen, D., Davatzikos, C., 2004. Deformable registration of tumor-diseased brain images. In: Barillot, C., Haynor, D., Hellier, P. (Eds.), Proceeding of Medical Image Computing and Computer Assisted Intervention. Vol. 3216 of LNCS. Springer, pp. 720–728.

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

Maintz, J., Viergever, M., 1998. A survey of medical image registration. Medical Image Analysis 2 (1), 1–36.

Miller, M., Younes, L., 2001. Group actions, homeomorphisms, and matching: A general framework. International Journal of Computer Vision 41 (1/2), 61–84.

Modersitzki, J., 2004. Numerical Methods for Image Registration. Oxford University Press.

Moon, N., van Leemput, K., Bullitt, E., Gerig, G., 2002. Automatic brain and tumor segmentation. In: MICCAI. pp. 372–379.

Mrázek, P., Navara, M., 2001. Consistent positive directional splitting of anisotropic diffusion. In: Likar, B. (Ed.), Proc. of Computer Vision Winter Workshop. Bled, Slovenia, pp. 37–48.

Nagel, H.-H., Enkelmann, W., 1986. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. IEEE Transactions on Pattern Analysis and Machine Intelligence 8, 565–593.

Netsch, T., Rosch, P., van Muiswinkel, A., Weese, J., 2001. Towards real-time multi-modality 3d medical image registration. In: Proceedings of the 8th International Conference on Computer Vision. IEEE Computer Society, pp. 718–725.

Ourselin, S., Roche, A., Prima, S., Ayache, N., 2000. Block Matching: A General Framework to Improve Robustness of Rigid Registration of Medical Images. In: DiGioia, A., Delp, S. (Eds.), Third International Conference on Medical Robotics, Imaging And Computer Assisted Surgery (MICCAI 2000). Vol. 1935 of Lectures Notes in Computer Science. Springer, Pittsburgh, Pennsylvanie USA, pp. 557–566.

Ourselin, S., Roche, A., Subsol, G., Pennec, X., Ayache, N., 2001. Reconstructing a 3D Structure from Serial Histological Sections. Image and Vision Computing 19 (1-2), 25–31.

Ourselin, S., Stefanescu, R., Pennec, X., 2002. Robust registration of multi-modal images: towards real-time clinical applications. In: Dohi, T., Kikinis, R. (Eds.), Medical Image Computing and Computer-Assisted Intervention (MICCAI'02). Vol. 2489 of LNCS. Springer, Tokyo, pp. 140–147.

Pennec, X., Cachier, P., Ayache, N., 1999. Understanding the "demon's algorithm": 3D non-rigid registration by gradient descent. In: Proc. of MICCAI 1999. Vol. 1679 of LNCS. pp. 597–605.

Piegl, L., 1991. On NURBS: A survey. IEEE Computer Graphics and Applications 11 (1), 55–71.

Pitiot, A., Malandain, G., Bardinet, E., Thompson, P., 2003. Piecewise affine registration of biological images. In: Gee, J., Maintz, J. A., Vannier, M. W. (Eds.), Second International Workshop on Biomedical Image Registration WBIR'03. Vol. 2717 of Lecture Notes in Computer Science. Springer-Verlag, Philadelphia, PA, USA, pp. 91–101, also research report INRIA RR-4866.

Pluim, J. P., Maintz, J. B. A., Viergever, M. A., 2003. Mutual-information-based registration of medical images: a survey. IEEE Transactions on Medical Imaging 22 (8), 986–1004.

Press, W., Flannery, B., Teukolsky, S., Vetterling, W., 1993. Numerical Recipes in C : The Art of Scientific Computing. Cambridge University Press.

Prima, S., Thirion, J.-P., Subsol, G., Roberts, N., 1998. Automatic Analysis of Normal Brain Dissymmetry of Males and Females in MR Images. In: Proc. of MICCAI'98. Vol. LNCS 1496. pp. 770–779.

Rexilius, J., Warfield, S., Guttmann, C., Wei, X., Benson, R., Wolfson, L., Shenton, M., Handels, H., Kikinis, R., 2001. A novel nonrigid registration algorithm and applications. In: Proceedings of MICCAI'01.

Riviere, D., Mangin, J.-F., Papadopoulos, D., Martinez, J.-M., Frouin, V., Regis, J., 2000. Automatic recognition of cortical sulci using a congregation of neural networks. In: Delp, S., DiGioia, A., Jaramaz, B. (Eds.), Proceeding of Medical Image Computing and Computer Assisted Intervention. Vol. 1935 of LNCS. Springer, pp. 40–49.

Roche, A., Malandain, G., Ayache, N., 2000. Unifying Maximum Likelihood Approaches in Medical Image Registration. International Journal of Imaging Systems and Technology: Special Issue on 3D Imaging 11 (1), 71–80.

Roche, A., Malandain, G., Pennec, X., Ayache, N., 1998. The correlation ratio as a new similarity measure for multimodal image registration. In: Proc. of First Int. Conf. on Medical Image Computing and Computer-Assisted Intervention (MICCAI'98). Vol. 1496 of LNCS. Springer Verlag, Cambridge, USA, pp. 1115–1124.

Rohde, G., Aldroubi, A., Dawant, B., 2003. The adaptive bases algorithm for intensity based nonrigid image registration. Special issue on image registration. IEEE Transactions on Medical Imaging 22, 1470–1479.

Rohlfing, T., Maurer, C., 2001. Intensity-based non-rigid registration using adaptive multilevel free-form deformation with an incompressibility constraint. In: Springer (Ed.), Proc. of MICCAI'01. Vol. 2208 of LNCS. pp. 111–119.

Rohr, K., 1997. On 3D differential operators for detecting point landmarks. Image and Vision Computing 15 (3), 219–233.

Rohr, K., Stiehl, H., Sprengel, R., Buzug, T., Weese, J., Kuhn, M., 2001. Landmark-based elastic registration using approximating thin-plate splines. IEEE Transactions on Medical Imaging 20 (6), 526–534.

Rueckert, D., Sonoda, L., Hayes, C., Hill, D., Leach, M., Hawkes, D., 1999. Non-rigid registration using free-form deformations: Application to breast MR images. IEEE Transactions on Medical Imaging 18 (8), 712–712.

Sandor, S., Leahy, R., 1995. Towards automated labeling of the cerebral cortex using a deformable atlas. In: Bizais, Y., Barillot, C., DiPaola, R. (Eds.), Proceedings of Information Processing in Medical Imaging. Kluwer, pp. 127–138.

Sermesant, M., Clatz, O., Li, Z., Lantéri, S., Delingette, H., Ayache, N., 2003. A parallel implementation of non-rigid registration using a volumetric biomechanical model. In: Gee, J., Maintz, J. A., Vannier, M. W. (Eds.), Second International Workshop on Biomedical Image Registration WBIR'03. Vol. 2717 of Lecture Notes in Computer Science. Springer-Verlag, Philadelphia, PA, USA, pp. 398–407.

Shen, D., Davatzikos, C., 2002. Hammer: Hierarchical attribute matching mechanism for elastic registration. IEEE Transactions on Medical Imaging 21 (11), 1421–1439.

Soille, P., 1999. Morphological image analysis : principles and applications. Springer.

Stefanescu, R., Commowick, O., Malandain, G., Bondiau, P.-Y., Ayache, N., Pennec, X., 2004a. Non-rigid atlas to subject registration with pathologies for conformal brain radiotherapy. In: Barillot, C., Haynor, D., Hellier, P. (Eds.), Proc. of the 7th Int. Conf on Medical Image Computing and Computer-Assisted Intervention - MICCAI 2004. Vol. 3216 of LNCS. Springer Verlag, Saint-Malo, France, pp. 704–711.

Stefanescu, R., Pennec, X., Ayache, N., November 2003a. Grid enabled non-rigid registration with a dense transformation and a priori information. In: Ellis, R. E., Peters, T. M. (Eds.), Proc. of MICCAI'03, Part II. Vol. 2879 of LNCS. Springer Verlag, Montreal, pp. 804–811.

Stefanescu, R., Pennec, X., Ayache, N., January 2003b. Parallel non-rigid registration on a cluster of workstations. In: Norager, S. (Ed.), Proc. of HealthGrid'03. European Commission, DG Information Society, Lyon.

Stefanescu, R., Pennec, X., Ayache, N., 2004b. Grid-enabled non-rigid registration of medical images. Parallel Processing Letters 14 (2), 197–216.

Stefanescu, R., Pennec, X., Ayache, N., 2004c. Grid powered nonlinear image registration with locally adaptive regularization. Medical Image Analysis 8 (3), 325–342, mICCAI 2003 Special Issue.

Stefanescu, R., Pennec, X., Ayache, N., 2004d. A grid service for the interactive use of a parallel non-rigid registration algorithm. In: Proc. of HealthGrid 2004. European Commission, DG Information Society, Clermont-Ferrand.
URL `http://clermont2004.healthgrid.org`

Stefanescu, R., Pennec, X., Ayache, N., 2005. A grid service for the interactive use of a parallel non-rigid registration algorithm of medical images. Methods of Information in Medicine 44 (1).

Studholme, C., Hill, D. L., Hawkes, D. J., 1995. Multiresolution voxel similarity measures for MR-PET registration. In: Bizais, Y. (Ed.), Proceedings of Information Processing in Medical Imaging. Kluwer, pp. 287–298.

Studholme, C., Hill, D. L. G., Hawkes, D. J., 1999. An overlap invariant entropy measure of 3D medical image alignment. Pattern Recognition 32, 71–86.

Talairach, J., Tournoux, P., 1988. Co-planar Stereotaxic Atlas of the Human Brain. Georg Thieme Verlag, Stuttgart.

Talairach, J., Tournoux, P., 1993. Referentially Oriented Cerebral MRI Anatomy. Georg Thieme Verlag, Stuttgart, New-York.

Tchumperle, D., 2002. PDE-based regularization of multivalued images and applications. Ph.D. thesis, University of Nice – Sophia-Antipolis.

Thévenaz, P., Blu, T., Unser, M., 2000. Interpolation revisited. IEEE Transactions on Medical Imaging 19 (7), 739–758.

Thirion, J.-P., 1996a. New feature points based on geometric invariants for 3D image registration. International Journal of Computer Vision 18 (2), 121–137.

Thirion, J.-P., 1996b. Non-rigid matching using demons. In: Computer Vision and Pattern Recognition, CVPR'96. San Francisco, California, USA.

Thirion, J.-P., 1998. Image matching as a diffusion process: an analogy with Maxwell's demons. Medical Image Analysis 2 (3), 243–260.

Thompson, P., Toga, A. W., 1996. A surface-based technique for warping three-dimensional images of the brain. IEEE Trans. on Medical Imaging 15 (4).

Trouvé, A., 1998. Diffeomorphisms groups and pattern matching in image analysis. International Journal of Computer Vision 28 (3), 213–221.

Tuecke, S., Czajkowski, K., Foster, I., Frey, J., Graham, S., Kesselman, C., Maguire, T., Sandholm, T., Vanderbilt, P., Snelling, D., 2003. Open Grid Services Infrastructure (OGSI) Version 1.0. Global Grid Forum Draft Recommendation.

Vaillant, M., Davatzikos, C., 1999. Hierarchical matching of cortical features for deformable brain image registration. In: IPMI '99: Proceedings of the 16th International Conference on Information Processing in Medical Imaging. Springer-Verlag, London, UK, pp. 182–195.

Viola, P., 1995. Alignment by maximization of mutual information. Ph.D. thesis, Massachusetts Institute of Technology, Artificial Intelligence Laboratory.

Viola, P., Wells, W., 1997. Alignment by maximization of mutual information. The International Journal of Computer Vision 24 (2), 137–154.

Warfield, S., Zou, K., Wells, W., 2004a. Simultaneous truth and performance level estimation (STAPLE): An algorithm for the validation of image segmentation. IEEE Transactions on Medical Imaging 23 (7).

Warfield, S., Zou, K., Wells, W., 2004b. Simultaneous truth and performance level estimation (STAPLE): An algorithm for the validation of image segmentation. IEEE Transactions on Medical Imaging 23 (7), 903–921.

Warfield, S. K., Talos, F., Tei, A., Bharatha, A., Nabavi, A., Ferrant, M., Black, P. M., Jolesz, F. A., Kikinis, R., 2002. Real-time registration of volumetric brain mri by biomechanical simulation of deformation during image guided neurosurgery. Computing and Visualization in Science.

Webb, J., Guimond, A., Roberts, N., Eldridge, P., Chadwick, D., Meunier, J., Thirion, J.-P., 1999. Automatic detection of hippocampal atrophy on magnetic resonnance images. Magnetic Resonance Imaging 17.

Weickert, J., 1997. A review of nonlinear diffusion filtering. In: Scale-Space Theory in Computer Vision. Vol. 1252 of Lecture Notes in Computer Science. Springer Verlag, pp. 3–28.

Weickert, J., 1998. Anisotropic diffusion in image processing. European Consortium for Mathematics in Industry. Teubner Verlag.

Weickert, J., 2000. Applications of nonlinear diffusion in image processing and computer vision. Acta Mathematica Universitatis Comenianae 70 (1), 33–50.

Weickert, J., Haar, B., Viergever, R., 1998. Efficient and reliable schemes for nonlinear diffusion filtering. IEEE Transactions on Image Processing 7, 398–410.

Woods, R. P., Mazziotta, J. C., Cherry, S. R., 1993. MRI-PET registration with automated algorithm. Journal of Computer Assisted Tomography 17, 536–546.

Ziv, J., Lempel, A., 1977. A universal algorithm for sequential data compression. IEEE Transactions on Information Theory 23 (3), 337–343.