國立交通大學

資訊科學與工程研究所

碩士論文

快速準確的微分同構對稱非剛性腦部磁振
造影影像對位演算法

A Fast and Accurate Algorithm for Diffeomorphic and
Symmetric Non-rigid Registration of Brain Magnetic
Resonance Images

研究生：林奭彥

指導教授：陳永昇　教授

中華民國 101 年 12 月

快速準確的微分同構對稱非剛性腦部磁振造影影像對位演算法
A Fast and Accurate Algorithm for Diffeomorphic and Symmetric Non-rigid Registration of Brain Magnetic Resonance Images

研究生：林爽彥　　Student：Shih-Yen Lin

指導教授：陳永昇　　Advisor：Yong-Sheng Chen

國立交通大學
資訊科學與工程研究所
碩士論文

A Thesis
Submitted to Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Computer Science

December 2012

Hsinchu, Taiwan, Republic of China

中華民國 101 年 12 月

A Fast and Accurate Algorithm for Diffeomorphic and Symmetric Non-rigid Registration of Brain Magnetic Resonance Images

A thesis presented

by

Shih-Yen Lin

to

Institute of Computer Science and Engineering

College of Computer Science

in partial fulfillment of the requirements

for the degree of

Master

in the subject of

Computer Science

National Chiao Tung University

Hsinchu, Taiwan

2012

A Fast and Accurate Algorithm for Diffeomorphic and Symmetric Non-rigid Registration of Brain Magnetic Resonance Images

摘要

於本論文中，我們提出了一個快速、精確、對稱且微分同構的核磁共振影像(MRI)對位演算法。我們利用一個對數－歐幾里德的架構以建立微分同構的模型，在此模型中，微分同構的李代數是由不隨時間變化的速度場表示，此速度場的模型是由多個具緊支撐的 Wendland 徑向基底函數的線性組合構成。最佳化所用的目標函數是由一有對稱性質的相關比及經過權重後的拉普拉斯算子模型組成。我們使用一具區域性及貪婪演算法性質的最佳化架構，藉以增進演算法的速度。在此架構中，我們使用一具對稱性的單純形演算法，來分別且逐一地找出各個徑向基函數的係數。為了在與仿射對位的結果合併時仍能保有整體對稱性，我們設計出一個應用“中途空間”概念的架構。藉由此架構，如果使用具對稱性的仿射對位演算法，則能確保整體對位流程也具對稱性。我們使用一個階層式的架構以增加演算法的速度及準確度，在此架構中，徑向基底函數是以由粗略至精細的順序逐一地被部署及估計。我們利用 LPBA40 數據集的 40 個 T1-權重 MRI 影像的共 1560 對影像對的對位以驗證本論文提出的演算法。經由驗證可得知此演算法完全滿足微分同構的性質，且對稱性的誤差也小於體素寬度。為了驗證準確度，我們利用 Klein 等人於 2010 年提出的驗證架構評估本演算法，並與其他 14 種對位演算法進行比較。驗證結果顯示本演算法的中位目標重疊值高於全部 14 個演算法。另外，在使用 5 層規模級別時，本演算法較 14 種演算法中所有具微分同構性質者快速。

Abstract

A fast symmetric and diffeomorphic non-rigid registration algorithm for magnetic resonance images (MRIs) is proposed in this work. A log-Euclidean framework is used to model diffeomorphisms, in which the Lie Algebra of the diffeomorphism is modeled by time-invariant velocity fields. The velocity fields are modeled using linear combinations of compactly-supported Wendland radial basis functions. A symmetric correlation ratio combined with a weighted Laplacian model is used as the objective function for optimization. We used a greedy local optimization scheme to increase the speed of the algorithm. In this setup, a symmetric downhill simplex method is used to estimate the coefficient of each radial basis function separately and consecutively. To incorporate the result of initial affine registration while maintaining overall symmetry, a framework utilizing the concept of "halfway space" is devised. This framework can ensure overall symmetry if the affine registration algorithm is symmetric. To increase the speed and accuracy, we used a hierarchical framework in which the RBFs are deployed and estimated in a coarse-to-fine manner. The proposed algorithm was evaluated using the results of 1560 pairwise registrations of 40 T1-weighted MRIs in LPBA40 dataset. According to the evaluation results, the proposed algorithm is completely diffeomorphic and has sub-voxel accuracy in terms of symmetry. The accuracy of the proposed algorithm was evaluated and compared with 14 registration methods using the evaluation framework by Klein et al., 2010. The median target overlap of the proposed algorithm using LPBA40 dataset is higher than all 14 registration methods. In addition, the proposed algorithm is faster than all diffeomorphic registration methods in the comparison when using 5 scale levels.

致謝

我很幸運能夠在BSP實驗室渡過我的碩士生涯。一路走來，陳永昇老師不只在學業的指導讓我受益良多，也在生活上給予我們諸多關心和照顧，是個不可多得的好老師。另外實驗室的氣氛也相當和樂與團結。最要感謝的是同組的予涵,兩年間無數次的討論，才讓我得到了如今的論文成果。另外，我也很謝謝泰成、旭龍、佩綺這些同屆一起同甘共苦的同學們，以及其他的實驗室夥伴們。另外也謝謝鍾孝文老師、陳麗芬老師以及王才沛老師，老師們給的建議以及指正讓我的論文內容更臻詳實。最後，也謝謝一路支持我的家人。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Backgrounds

In a nutshell, image registration is a procedure that warps an image toward another such that these two images will match one another as well as possible under a certain criterion. It is an important operation for the analysis of brain magnetic resonance images (MRI), and, more specifically, the central procedure of the spatial normalization. Spatial normalization involves moving multiple images to a single reference frame for further analysis. It enables the researchers to analyze the intra-subject and inter-subject anatomical variation on the same basis. As a result, spatial normalization has been used in a large number of studies for various purposes such as the investigation of various brain diseases such as dementia [9], schizophrenia [16] and epilepsy [25].

Registration can be categorized into two types: Affine (or global) registration and non-rigid (or local) registration. Affine registrations ($\Phi^{(g)}$) transform images globally by modelling the transformations as affine transformation matrices ($\left[\Phi^{(g)}(x)\ 1\right]^T = T\,[x\ 1]^T$). Affine registrations are simple but can only roughly match two images. In contrast, non-rigid registrations model transformations as mapping functions ($\Phi : \Re^3 \rightarrow \Re^3$). Non-rigid registrations can account for local deformations and can hence achieve better correspondence between two images. However, non-rigid registration is in general a more difficult task. The speed and accuracy of non-rigid registrations are likely to be affected if the images in the image pair are very different from one another. In practice, most non-rigid registration algorithms entail affine registrations in pre-processing steps for the sake of stability:

$$\Phi = \Phi^{(l)} \circ \Phi^{(g)} \tag{1.1}$$

Our work mainly focused on the non-rigid registration.

In this thesis, the two images in the image pair used in the registration is referred to as the source image ($I_s : \Re^3 \rightarrow \Re$) and the target image ($I_t : \Re^3 \rightarrow \Re$) respectively, in which the source image denotes the image to be warped and the target image is the static reference image. The registration process is the estimation of the optimal mapping such that the warped source image will match the target image best under a certain cost function

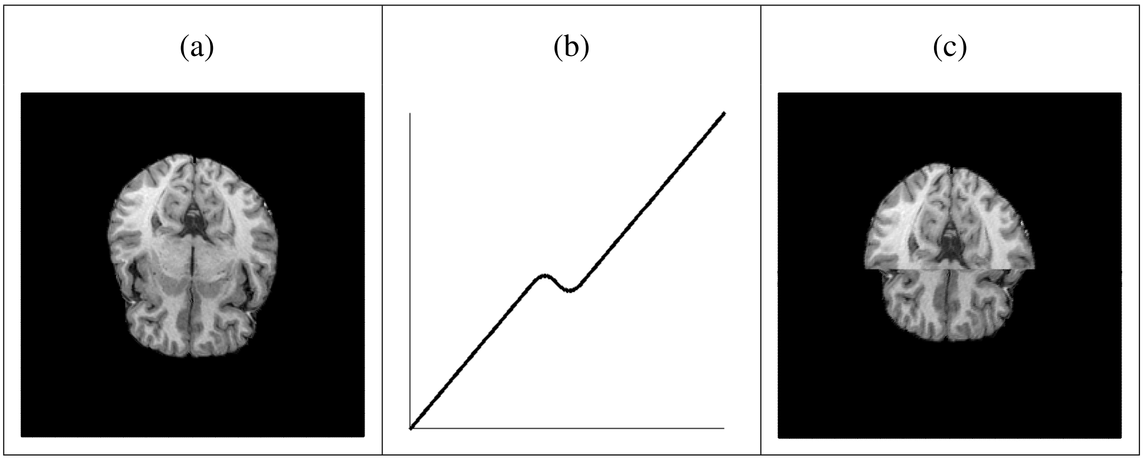

Figure 1.1: An example of folding. (a) The image before transformation. (b) The mapping function along y-axis (non-injective). (c) The image after transformation. The center part of the image is folded together, causing unrecoverable volume loss.

(or objective function):

$$\bar{\Phi} = \arg\min_{\Phi} E\left(I_s, I_t, \Phi\right) \tag{1.2}$$

1.2 Small-Deformation Registration Frameworks

According to Ashburner [7], registration methods can be divided into two general categories: Small-deformation frameworks and large-deformation (or diffeomorphic) frameworks. Most conventional registration algorithms use a small-deformation framework, in which transformations are encoded in the form of displacement fields (d): $\Phi\left(x\right) = x + d\left(x\right)$. Registration algorithms using such framework include FNIRT [2], SPM5 normalization algorithm and BIRT [22]. Registration algorithms using this type of framework assume transformations to be Euclidean and allows transformations to be processed by Euclidean operations. The main advantage of this type of framework is its simplicity and efficiency in terms of the calculation of the mapping function, since the relationship between the mapping function and the displacement field can be established through a single vector addition. However, it does not naturally guarantee a one-to-one mapping and, in other words, the existence of the inverse transformation. This means that the topology of the image is not necessarily preserved after transformation in this type of framework. Loss of topology leads to artifacts such as folding (see Figure 1.1), which is physically impossible and also causes unrecoverable loss of volume after transformation as well. In the light of this problem, some small-deformation methods attempted to ensure the preservation of topology by imposing regularization constraints [31]. Another major drawback of small-deformation based methods is that the inverse transformation in this type of framework cannot be found in a straightforward manner. This may be a fatal flaw for applications such as the construction of brain templates [20], in which the informations of inverse transformations is necessary. Some works tackled this problem by estimating forward and inverse transform simultaneously [27, 30].

1.3 Diffeomorphic Registration frameworks

The development of diffeomorphic registration frameworks has provided a more elegant solution to these limitations of small-deformation frameworks [7, 8, 10, 14, 38]. Diffeomorphic frameworks represent the mapping using the well-defined mathematical structure of diffeomorphism. A diffeomorphism is a smooth and invertible function whose inverse is also smooth and invertible. This type of framework ensures one-to-one relationship of both the mapping function and its inverse, and thus guarantees to preserve local structures after transformation. Moreover, inverse transformations (Φ^{-1}) in diffeomorphic frameworks are readily available once the forward transformations (Φ) is found (detailed explanations are given in Section 2). Diffeomorphisms form a Lie Group under composition of functions, and, as a result, the composition of multiple diffeomorphic transformations will also be diffeomorphic. Same as Lie Groups, the space of diffeomorphisms is also a manifold.

1.4 Symmetric Registration

With guaranteed invertibility and readily available inverse transformations, diffeomorphic registration facilitates the imposition of another important property: symmetry (or inverse-consistency). Symmetric registrations can produce consistent result when the source and the target image are interchanged. Let $\bar{\Phi}_{B\to A}$ denote the estimated transformation obtained by registering image A to image B, fully symmetric registrations will satisfy the following:

$$\forall A, B, \bar{\Phi}_{B\to A} = \left(\bar{\Phi}_{A\to B}\right)^{-1} \tag{1.3}$$

This property is crucial especially for applications in which pairwise registrations are involved. If a registration algorithm is asymmetric, the transformation of registering image A to image B will be inconsistent to that of registering image B to image A. This means the result of the registration somehow depends on the choice of which image in the image pair is to be warped onto one another. This dependency may introduce bias and algorithm-induced artifacts in subsequent analysis. Reuter et al. [27] pointed out that the bias caused by asymmetric registrations in longitudinal image processing may lead to incorrect results and potentially flawed interpretation of outcome measures. Artifacts induced by asymmetric registrations have also been reported in various works [37, 41]. Using symmetric registrations can prevent this kind of problem and generate more valid analysis result.

Some symmetric registration methods using the small-deformation framework has been proposed [27, 36]. The method in [27] transforms two images in both directions simultaneously in order to achieve symmetry. However, these small-deformation-based methods merely approximate symmetry by including penalties in the optimization algorithm. On the other hand, diffeomorphic frameworks provide the well-formed mathematical model of diffeomorphism. Using this type of framework, the design of symmetric registration algorithms can be greatly simplified, and the accuracy in terms of the symmetry can be improved as well.

1.5 Motivation

Despite all the advantages of diffeomorphic frameworks, the computation of diffeomorphic transformations are rather computationally expensive when compared to simple displacement-field models in small-deformation frameworks. For example, registrations using DARTEL [7] usually take more than an hour, while some small-deformation based methods such as BIRT only need a few minutes to achieve similar accuracy. Therefore, our goal in this work is to design a diffeomorphic and symmetric registration algorithm which is free of the typical long computation time of diffeomorphic registration algorithms and hopefully also surpasses other diffeomorphic registration algorithms in other aspects.

1.6 Thesis Overview

The remaining part of this thesis is organized as follows. In Chapter 2, some background knowledge and related works are briefly explained, which includes the concepts of diffeomorphisms, Lie Algebra and basis functions. Chapter 3 gives a detailed description of the proposed registration algorithm. Chapter 4 explains various implementation details involved in our work. In Chapter 5, the results and the evaluations of the proposed algorithm in different aspects are given. Chapter 6 discusses various issues related to the proposed algorithm, which includes discussions on the result of evaluation as well as the potential flaws of the proposed algorithm and their possible solutions. Finally, the concluding remarks and our future goals are given in Chapter 7.

Chapter 2

Background Knowledge and Related Works

Before giving a detailed description of our algorithm, some prerequisite background knowledge related to our work are explained in this chapter. Diffeomorphisms and Lie Algebra are two key elements to most diffeomorphic registration methods. The theories as well as the implementations of these two terms are briefly explained in this chapter. Additionally, the role of basis functions in registration algorithms is also discussed. Previous works related to these topics are also included in this chapter.

2.1 Diffeomorphisms and Lie Algebra

As mentioned, diffeomorphisms forms a manifold of Lie Group under composition of functions. In other words, a diffeomorphism only resembles an Euclidean space in a local scale. The non-Euclidean property of diffeomorphisms becomes apparent when the deformation is large. As a result, it cannot be directly modeled using a simple Euclidean vector field (e.g. a displacement field as in the small-deformation framework). However, diffeomorphisms can be implicitly modelled as Euclidean vector fields by using the concept of Lie algebra. By doing so, operations on the manifold of diffeomorphisms can be simplified to simple Euclidean operations, and the property of diffeomorphism can be also preserved after operations. As a result, the concept of Lie Algebra was utilized in most diffeomorphic registration algorithms.

For a Lie Group (G), its Lie Algebra (g) is the Euclidean tangent space at the identity element. The conversion between Lie Groups and Lie Algebras is defined as the exponential and the logarithm operators:

$$\exp(\cdot) : g \rightarrow G \tag{2.1}$$

$$\log(\cdot) = \exp^{-1}(\cdot) : G \rightarrow g \tag{2.2}$$

To be simple, the logarithm operator projects a point on a manifold of Lie Group to a Euclidean tangent space, and the exponential operator projects a point on the Euclidean tangent space back to the Lie group. In our case, a diffeomorphism (Φ), which belongs to a Lie Group G, can be modeled by its Lie algebra (V), which is an Euclidean vector field:

$$V = \log(\Phi) \tag{2.3}$$

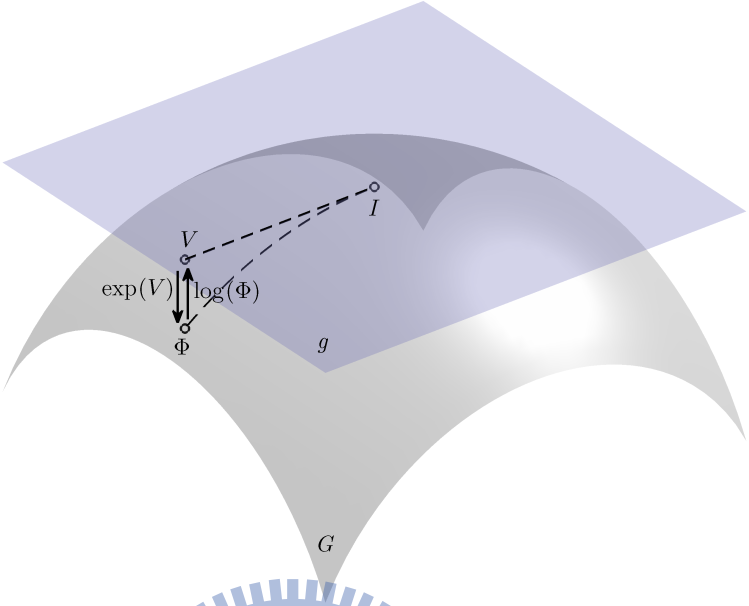

Figure 2.1: A simple illustration of the relationship between the Lie Group of diffeomorphisms and its Lie Algebra. A diffeomorphism (Φ) is a point on the Lie Group of diffeomorphisms (The gray curved surface). The logarithm operator projects a diffeomorphism to a point in the Euclidean space of Lie Algebra (The purple plane). Conversely, the exponential operator projects a point in the Lie Algebra back to the Lie Group.

Conversely, a diffeomorphism can be found through exponentiation of its Lie algebra:

$$\Phi = \exp\left(V\right) \tag{2.4}$$

A simple illustration is given in Figure 2.1. Since a Lie algebra is an Euclidean space, operations of diffeomorphisms can be simplified to simple Euclidean operations. For instance, combinations of diffeomorphisms can be done through simple vector additions:

$$\left(\exp aV\right)\left(\exp bV\right) = \exp\left(a+b\right)V \tag{2.5}$$

And the inverse transformation of a diffeomorphism is simplified to the negation of its Lie Algebra:

$$\left(\exp V\right)^{-1} = \exp\left(-V\right) \tag{2.6}$$

This simplicity for combination and inversion of transformations is a highly desirable property for applications where operations of transformations are needed.

2.2 Models of Diffeomorphisms

The concept of Lie algebra is used in many diffeomorphic registration algorithms, although the definitions of the Lie Algebra of diffeomorphism vary. Christensen et al. [14] and Beg et al. [10] modelled the Lie Algebra of a diffeomorphism as a time-dependent velocity field ($V(t, \cdot)$), and the exponential operator is modeled as the flow (φ) of this velocity field at a unit time:

$$\exp(V) = \varphi_V(\cdot, 1) \tag{2.7}$$

$\varphi_V(\cdot, t)$ denotes the flow of the partial differential equation (PDE) $\dot{x}(t) = V(t, x(t))$ at time t.

This model was used in many diffeomorphic registration algorithms. In the work by Christensen et al. [13, 14], diffeomorphisms were modeled as a highly viscous fluid, and the registrations were done by solving PDE's derived from continuum mechanics for deformable bodies. Beg et al. [10] proposed the LDDMM (larger deformation diffeomorphic metric mapping) algorithm, which finds the optimal velocity fields that will minimize the geodesic distance on the manifold of diffeomorphisms.

A simpler model was also proposed [3, 7]. These works modelled transformations using one-parameter subgroups of diffeomorphisms, which can be modelled as flows of time-independent velocity fields ($V(\cdot)$) at a unit time. The flows involved in the exponentiation in these works are simplified to finding a the solution of a stationary PDE ($\dot{x}(t) = V(x)$). An example is given in Figure 2.2. This model greatly alleviated the computational burdens when calculating diffeomorphisms, since diffeomorphisms can be generated through estimating a single time-invariant velocity fields instead of a series of velocity fields at different time.

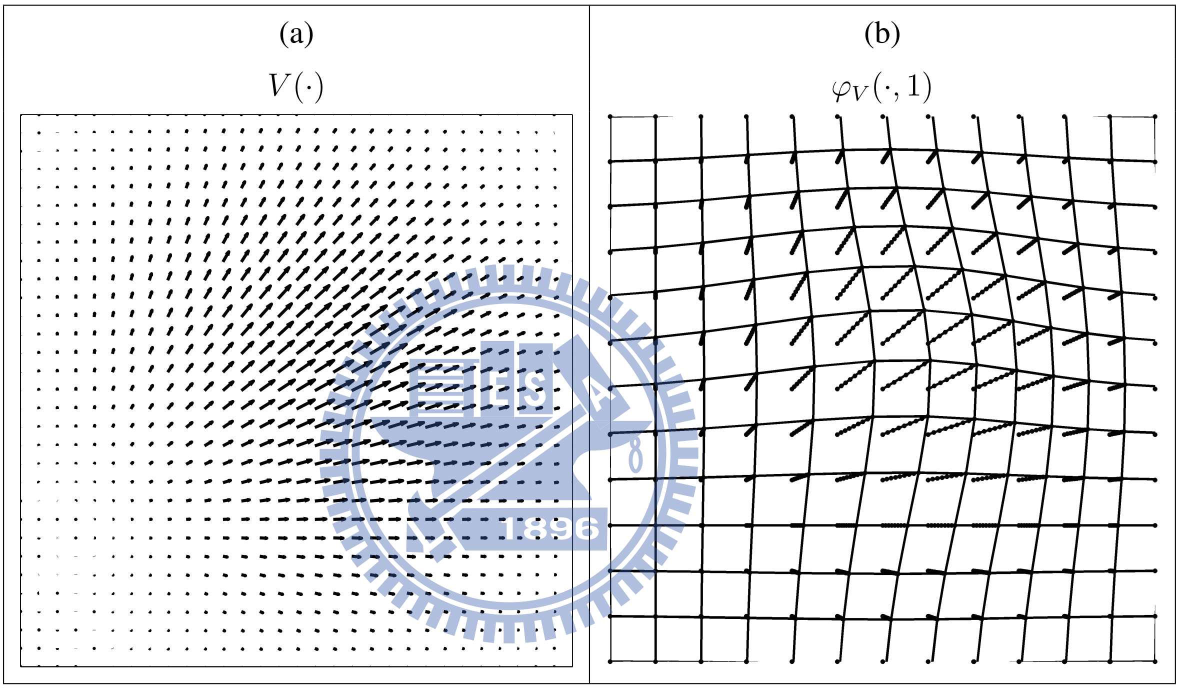

Figure 2.2: A diffeomorphism is obtained by calculating the flow of its velocity field. (a) A velocity field. (b)The resultant diffeomorphism is the flow of this velocity field at a unit time.

2.3 Basis Functions

Basis functions have been used as the models of local deformations in many registration algorithms, regardless of the type of framework used. In these works, mapping functions (either in the form of displacement fields as in small-deformation frameworks or velocity fields as in diffeomorphic frameworks) are estimated by linear combinations of basis functions:

$$u(x) = \sum_i \alpha_i \rho_i(x) \tag{2.8}$$

Where α is a vector coefficient and ρ_i is the i^{th} basis function. u can be either a displacement field or a velocity field, which depends on the type of framework used. For registration approaches based on the small-deformation framework, basis functions are used to describe the displacement fields. On the other hand, basis functions are used to estimate velocity fields in diffeomorphic registration approaches. For example, Ashburner [7] parametrizes the velocity fields using combinations of B-spline basis functions. These basis functions can either be deployed regularly [22] or according to other mechanisms (e.g. around misregistered regions [31] or on user-identified landmarks [21]).

The use of basis functions greatly reduces the parameters to be estimated. For example, for a $256 \times 256 \times 128$ MR image, there are a total of 25 million parameters if no basis function is used. If the mapping function is described by basis functions on regular grids with width W, the number of parameters can be reduced by a ratio of W^3. Many types of basis functions have been used in previous works on registration. This includes thin-plate splines, Gaussian, inverse-multiquadratics [33] , multiquadratics [21], wavelets [1], discrete cosine transform [4], B-splines [32], and Wendland's RBFs [22, 31].

Chapter 3

Methods

3.1 Model of Diffeomorphism

In our work, a framework similar to that of Ashburner [7] and Arsigny et al. [3] is used. A diffeomorphism (Φ) is implicitly parametrized by its Lie Algebra, which is an time-independent velocity field (denoted as V):

$$V = log(\Phi) \tag{3.1}$$

Or, conversely, a diffeomorphism is the exponential map of its velocity field:

$$\Phi = exp(V) = log^{-1}(V) \tag{3.2}$$

The exponential operator proposed by Arsigny et al. is defined as the flow of the PDE $\dot{x}(t) = V(x)$ at time 1:

$$\exp(V) = \varphi_V(\cdot, 1) \tag{3.3}$$

φ_V denotes the flow of the velocity field V. The diffeomorphism is obtained through the exponentiation of its Lie Algebra (i.e. velocity field). Details of calculation of flows are further discussed in Chapter 4.

3.2 Radial Basis Functions

Recalling (2.8), the velocity fields can be modelled as linear combinations of basis functions. In the proposed algorithm, the velocity field is estimated using the linear combination of radial basis functions (RBF):

$$V(x) = \sum_i \alpha_i \rho_i(x) \tag{3.4}$$

Where:

$$\rho_i(x) = \Psi_{S_i}(\|x - c_i\|) \tag{3.5}$$

α is the vector coefficient representing the orientation and the magnitude. $\Psi_S(\|x - c\|)$ is a radial basis function centered at c with support extent s. In this work, a type of the Wendland Ψ-functions [40], $\Psi_{3,1}$, is used as the radial basis function. Wendland Ψ-functions

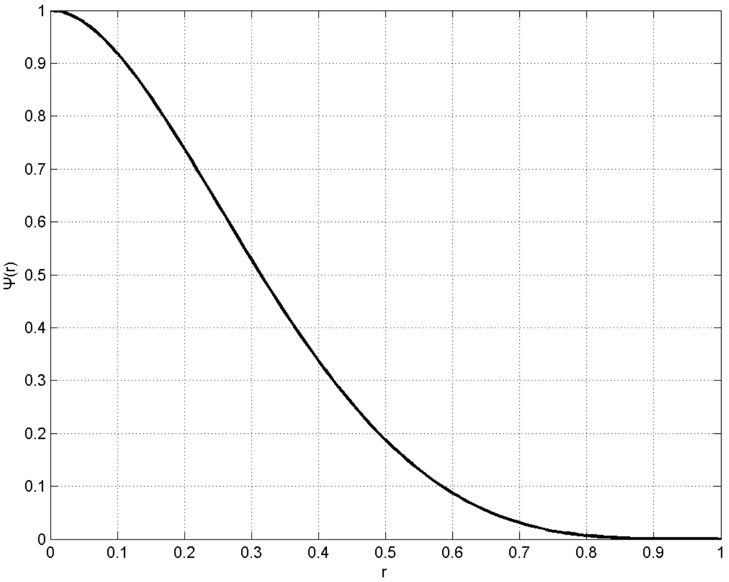

Figure 3.1: Wendland's RBF ($\Psi(r)$)

(shown in Figure 3.1) is a piecewise polynomial and compact supported radial basis function:

$$\Psi_{3,1}(r) = \begin{cases} (1-r)^4(4r+1) & \text{if } 0 \leq r < 1 \\ 0 & \text{if } r =\geq 1 \end{cases} \tag{3.6}$$

The support extent of Wendland Ψ-functions is restricted to a sphere with unit length, meaning the RBF $\Psi_{3,1}(r)$ is always zero when r is larger than 1. An additional parameter S can be used to specify variable support extents:

$$\Psi_S(r) = \Psi(\frac{r}{S}) \tag{3.7}$$

The characteristics of Wendland Ψ-functions are highly beneficial for the speed of the proposed algorithm in several aspects. First, piecewise polynomial RBFs can be calculated much faster than other types of RBFs such as Gaussian since only calculations of polynomials are involved (while Gaussian requires numerical calculations of exponentials). Second, the support of a Wendland's RBF is confined to a certain local region. This means that adding a Wendland's RBF to the velocity field will only affect a certain local region. In contrast, adding a RBF without compact support (e.g. Gaussian) to the velocity will effect the whole velocity field, meaning all points in the velocity field have to be included in the calculation. As a result, the use of Wendland's RBF can greatly reduce the time of summing basis functions or optimizing the vector coefficient of a certain RBF.

3.3 The Objective Function

Generally, the objective function for the registration algorithm is a measure of how good the parameter set (v) is for registering a dataset (D). The maximization of the objective function can be formulated as the maximum a posteriori probability (MAP) estimate:

$$v_{ML} = \arg\max_{v} p(v|D) \tag{3.8}$$

Where the objective function is in the form of the posterior probability:

$$p(v|D) = \frac{p(D|v)p(v)}{p(D)} \tag{3.9}$$

$p(D|v)$ is the likelihood, which indicates the probability of the data given the parameters. $p(v)$ is the prior probability. $p(D)$ is the probability of the data, which is a constant. Normally, the probabilities are represented in the form of their logarithms (or negative logarithms) in order to simplify the calculation. In this work, the probabilities are represented by their negative logarithms:

$$-\log p(v|D) = -\log p(D|v) - \log p(v) + \log p(D) \tag{3.10}$$

The negative logarithm of the probabilities can also be thought of as the energy of the transformation. The constant term log $p(D)$ is neglected since it does not affect the result. We rewrite the objective function as follows:

$$E(I_s, I_t, \exp(V)) = E_1(I_s, I_t, \exp(V)) + \lambda E_2(\exp(V)) \tag{3.11}$$

E_1 is the likelihood term and E_2 is the prior term. The likelihood term is a measure which indicates the similarity (or dissimilarity) between the warped source image and the target image. The prior term regularizes the transformation to conform to a given prior knowledge. λ is a weight (usually user-specified) indicating the comparative importance of the prior term to the overall objective function. The best set of parameters is the one which has the minimum energy:

$$v_{ML} = \arg\min_{v} E(I_s, I_t, \exp(V)) \tag{3.12}$$

The definitions of the likelihood and the prior term vary between different registration approaches. The rest of this section will focus on different implementations of these terms in previous works as well as the choice of these terms in the proposed algorithm.

3.3.1 The Likelihood Term

Likelihood term, which is often referred to as the similarity measure, is an intensity-based measure which indicates the similarity between two images. To choose a suitable similarity measure for the MRI registration algorithm, some characteristics of MRIs should be taken into considerations. First, there is a considerable variability of the brightness and the contrast among MRIs acquired from different scanners (see Figure 3.2). Second, there is a universal phenomenon of so-called inhomogeneity (or non-uniformity) among MRIs (see Figure 3.3). Due to the some technical factors, the intensity of a certain tissue may vary between local regions. As a result, correction of inhomogeneity is an indispensable preprocessing step for most registration algorithms. Third, MRIs acquired using different pulse sequences have distinct properties and intensity modalities. For images of different modalities, the intensity charastics of tissues could be totally different (see Figure 3.4). For example, the white matter has higher intensity than the gray matter T1-weighted images but has lower intensity in the gray matter in T2-weighted images. In some cases of research, researchers need to incorporate the results from images of different intensity modalities (e.g. T1-weighted MRIs and CT images) through registration. Since the intensity relationship between images of different modalities is even more complex, the similarity measure used in this kind of multi-modal registration needs to be chosen carefully.

The similarity measure used in the MRI registration algorithm should be able to account for these attributes of MRIs. Various types of similarity measures have been used to account for the differences between images. Similarity measures can be categorized into

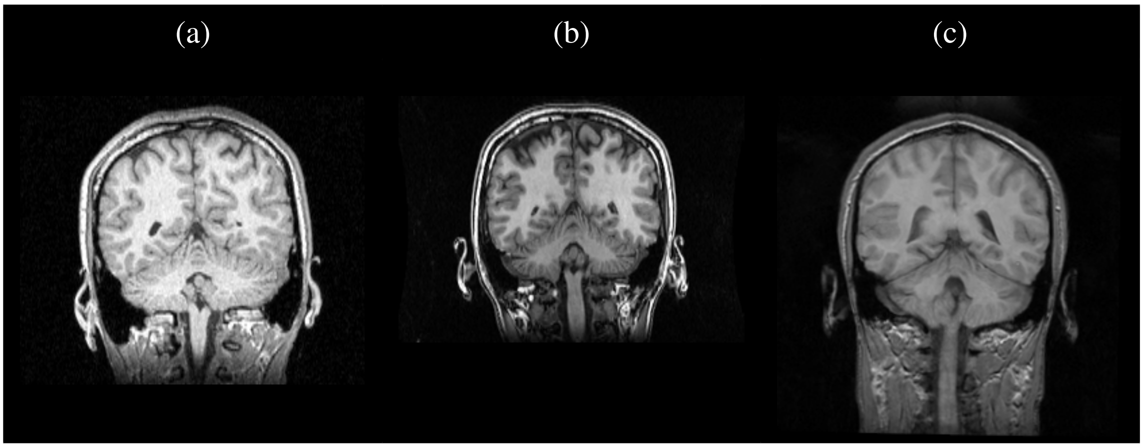

Figure 3.2: T1-weighted MR images acquired from different scanners present quite different intensity properties. MR images (a), (b), and (c) were scanned on a Bruker MedSpec S300 3T system, GE Signa EXCITE 1.5T system, and Siemens Magnetom 1.5T system, respectively.

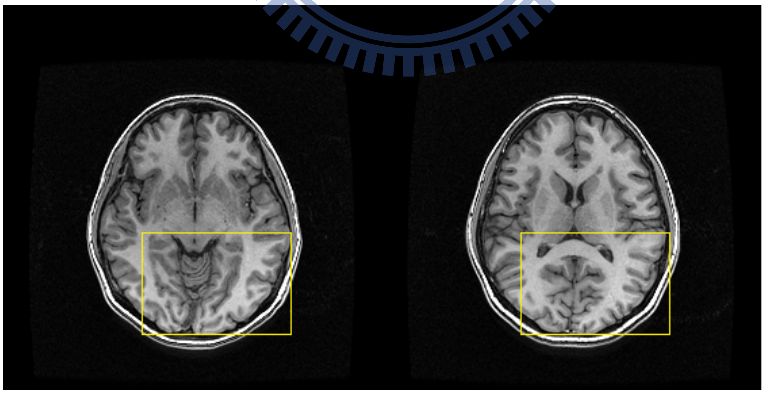

Figure 3.3: Intensity inhomogeneity of T1-weighted MR images. The right-posterior tissues highlighted in yellow boxes show obviously higher intensity compared to other regions.

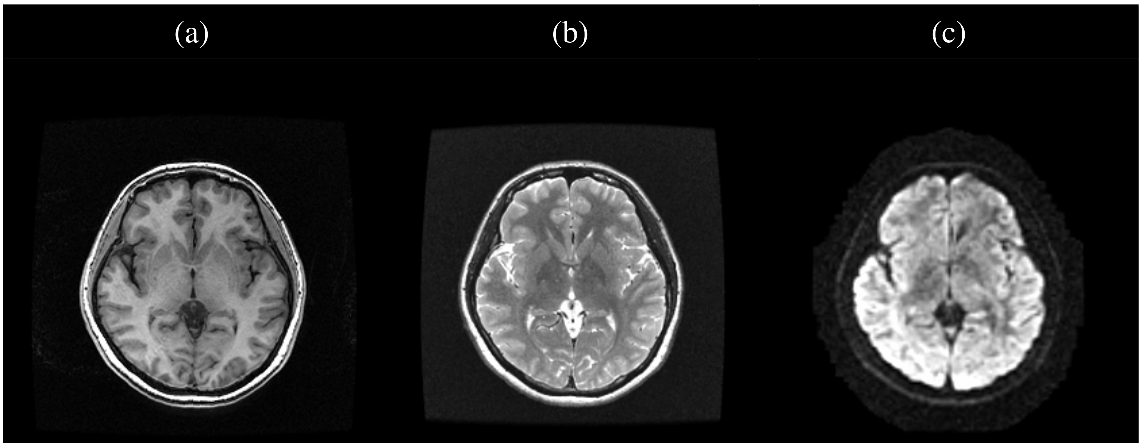

Figure 3.4: MR images acquired using different pulse sequences. The intensity characteristics of tissues vary between different modalities. (a) T1-weighted MR image. (b) T2-weighted MR image. (c) Diffusion weighted image.

four types of hypothesis: Identity relationship, affine relationship, functional relationship and statistical relationship (For details, see [28]). Similarity measures assuming identity relationships (e.g. sum of squared intensity differences) cannot account for variable brightness and contrast or the inhomogeneity of MRIs. Measures assuming affine relationships (e.g. cross-correlation) prove to perform decently for uni-modal registrations of MRI [8]. Measures assuming functional relationship (e.g. correlation ratio [29]) and statistical relationship (e.g. mutual information) are more well-formed statistically. These two kinds of measures are robust against the variable brightness and contrast of MRIs. Additionally, they are also fairly suitable for multi-modal registration.

In our work, a symmetric version of correlation ratio (CR) is used as the likelihood term. Correlation ratio is a similarity measure which assumes that the intensities of perfectly matched images conform to a certain functional relationship (i.e. $I_t \approx f(I_s)$). Intuitively speaking, for voxels with same (or similar) intensity in the target image, CR assumes the corresponding points of these voxels in a perfectly matched source image should have a certain uniform intensity (see Figure 3.5a). Conversely, intensity dispersion of these corresponding points is interpreted as the dissimilarity between the source image and the target image (see Figure 3.5b). The similarity can thus be quantified by measuring the degree of intensity dispersion of these corresponding points. Comparing to other conventional similarity measures that assume identity or affine relationships, CR is a more robust similarity measure against the variations of intensity and contrast as well as different modalities.

The definition of CR is given in the following. First, the intensity range of the target image is divided into N_B bins: $B_i, i = 1, , N_B$. Given an evaluation region Ω with N voxels, X_i denotes the voxels in the region Ω of the source image whose corresponding points in the target image belong to the i^{th} intensity bin, B_i:

$$X_i = \{x | x \in \Omega, I_t(\Phi(x)) \in B_i\} \tag{3.13}$$

We denote N_i as the number of voxels in X_i. The correlation ratio is formulated as:

$$\mathrm{CR}(I_s, I_t, \Phi) = 1 - \frac{\sum_{i=1}^{N_B} N_i \, \mathrm{Var}(I_s(X_i))}{N \, \mathrm{Var}(I_s(\Omega))} \tag{3.14}$$

Figure 3.6 shows a real example of two T1-weighted MR images. In Figure 3.6a, a badly-matched image pair has no clear functional relationship and thus large variance in each bin. On the other hand, Figure 3.6b shows the intensity relationship of a registered image pair, and we can see a clearer intensity relationship and hence smaller variances.

Although CR is a powerful similarity measure, it is not symmetric. To achieve symmetry while preserving the advantages of CR, a symmetric version of CR inspired by Lau et al. [19] is used. For image A and B, the symmetric correlation ratio (SCR) is formulated as follows:

$$\mathrm{SCR}(A, B, \Phi) = \mathrm{SCR}(B, A, \Phi^{-1}) = \frac{(\mathrm{CR}(A, B, \Phi) + \mathrm{CR}(B, A, \Phi^{-1}))}{2} \tag{3.15}$$

The proposed symmetric correlation ratio is similar to that of Lau et al. [19] except for the normalization term. The SCR in our work is normalized to the range of $[0\ 1]$. Using the SCR, the likelihood term can be formulated as follows:

$$E_1(I_s, I_t, \Phi) = \mathrm{SCR}(I_s, I_t, \Phi) \tag{3.16}$$

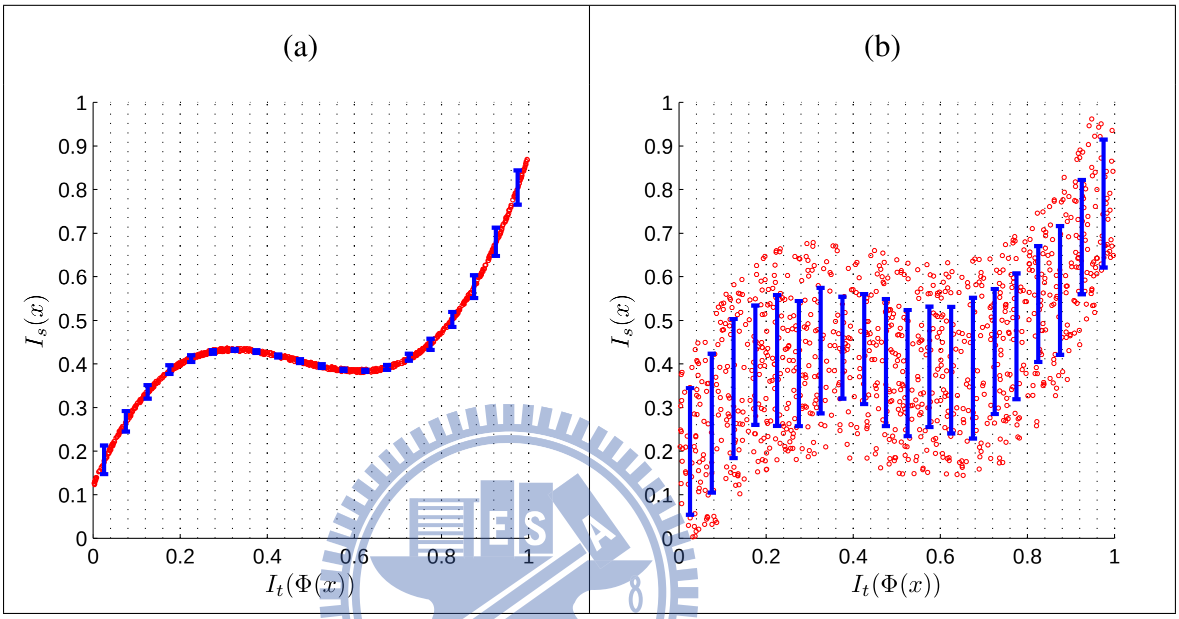

Figure 3.5: An illustration of CR. Red circles represent the intensity correspondence between the target image and the warpped source image at each point. For each point, its X coordinate indicates its intensity in the target image and its Y coordinate indicates the intensity of its corresponding point in the warped source image. The vertical dotted lines represent the borders of intensity bins, and the blue bar in each bin indicates the range of standard deviation for all points within that bin. (a) Similar image pair leads to smaller variance within each bin, which indicates that the intensity of the image pair conforms to a certain functional relationship. (b) Dissimilar image pair has larger variance within each bin.

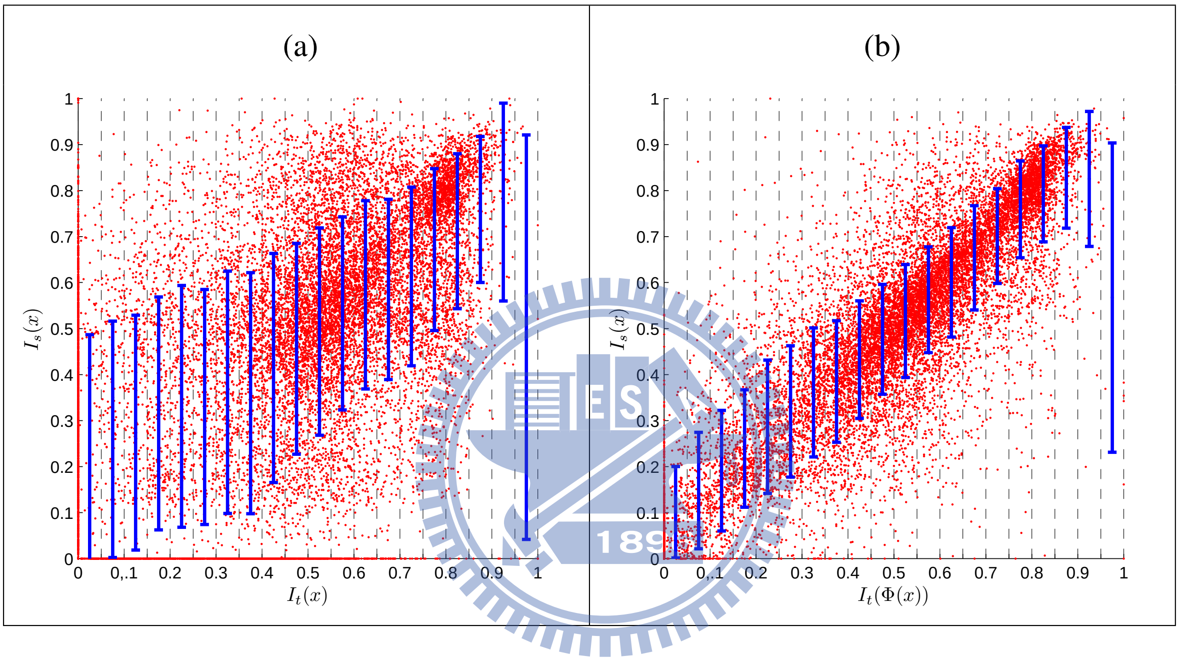

Figure 3.6: An example of CR using a real image pair. (a)The unmatched image pair does not has clear intensity relationships, leading to larger intensity variance in each bin. (b)A better-matched image pair has a clearer intensity relationship and smaller intensity dispersion in each bin.

When calculating the correlation ratio, one should note that the width and the number of bins are important factors pertaining to the performance and the validity of CR. The performance of CR may drop significantly if the bin width is either too large or too small. Additionally, the choice of the evaluation region is also important. To increase the efficiency, the evaluation region should be as compact as possible. Also, the calculation of $\mathrm{CR}(A, B, \Phi)$ and $\mathrm{CR}(B, A, \Phi^{-1})$ involves different evaluation regions. For now we simply denote the evaluation region $\mathrm{CR}(A, B, \Phi)$ as Ω and the evaluation region for $\mathrm{CR}(B, A, \Phi^{-1})$ as Ω^{inv}. Selection of the optimal bin width and the evaluation regions is further discussed in Chapter 4.

3.3.2 The Prior Term

The prior term is a measure which indicates how probable the transformation function is. This term can ensure the transformation to be realistic according to a certain prior knowledge such as smoothness of deformation or preservation of topology. Various types of prior terms have been used in previous works of registrations. This includes membrane energy, bending energy and linear-elastic energy [5].

The prior term used in our work is the membrane energy of the velocity field, which is also known as the Laplacian model:

$$E_{Laplacian}(V) = \frac{1}{\Omega}\iiint [(\frac{\partial V}{\partial x})^2 + (\frac{\partial V}{\partial y})^2 + (\frac{\partial V}{\partial z})^2] dx\, dy\, dz \tag{3.17}$$

And:

$$\lambda E_2(\exp(V)) = \lambda E_{Laplacian}(V) \tag{3.18}$$

In here, Ω is the volume involved in the estimation. In our case, V is the support of the current RBF. The membrane energy is term that favors smoother deformation. This prior term was also used in other works [7, 22]. Unlike Liu et al. [22], in which the membrane energy was used to regularize the displacement field, this term was used to regularize the velocity field in the proposed algorithm. The user-specified weight (λ) controls how much the prior term effects the optimization. Large weight leads to results with smoother deformations and lower similarity measures. The registration results using small weight have higher similarity but may be unrealistic. In our work, we set the value of λ empirically to 0.05 according to experiments using T1-MRIs.

3.4 Optimization

This section is divided into two parts. The first part explains how we designed the optimization problem as separate optimization problems in different local regions. The second part describes the symmetric optimization algorithm used in our work.

3.4.1 Local Optimization Scheme

In most registration approaches, coefficients of all basis functions are estimated simultaneously. This involves optimization in an extremely high-dimensional parameter space. Even with the use of basis functions to reduce the number of parameters, the number of parameters to be estimated can still easily reach millions. The curse of dimensionality arises when finding the parameter set in such extremely high-dimensional search space, causing the typical high time complexity of registration algorithms. In the light of this problem, Rohde et al. [31] proposed a greedy approach which optimizes the basis functions separately. In their work, 8 RBFs are applied to each identified region of mis-registration, and the vector coefficients of these 8 RBFs are estimated simultaneously. By doing so, the high-dimensional optimization problem is reduced to a sequence of 24-parameter optimization problems. Liu et al. [22] further elevated this idea by optimizing all regularly deployed basis functions separately, which results in a sequence of 3-parameter optimization problems. The results of these works have showed that separating the optimization of parameters yields significant improvement in term of the speed while maintaining decently high accuracy.

A similar greedy approach to Rohde et al. and Liu et al. is used in our work. In the proposed algorithm, a cumulative velocity field V_k is used to store the sum of all previously estimated RBFs from the start to the current step k:

$$V_k = \sum_{i=1}^{k} \alpha_i \rho_i(x) \tag{3.19}$$

Each RBF is estimated such that the objective function will be minimized when this RBF is added to the cumulative velocity field:

$$\alpha_i = \arg\min_{\alpha_i} E(I_s, I_t, \exp(V_{i-1} + \alpha_i \rho_i(x))) \tag{3.20}$$

The estimated RBF is then added to the cumulative velocity field to form a new one:

$$V_i \leftarrow V_{i-1} + \alpha_i \rho_i(x) \tag{3.21}$$

Repeating these steps through all N RBFs yields the velocity field representing the overall transformation:

$$\Phi = \exp(V_N) \tag{3.22}$$

Since RBFs are estimated sequentially, the order of RBFs in this optimization scheme may affect the accuracy of the final result. In our work, the order of RBFs is given according to the distance of each RBF from the brain center such that RBFs closer to the brain center are estimated before those farther from the brain center. This design is based on the fact that the cerebral cortex has higher structural complexity and anatomical variability than the structures near the center of the brain (e.g. corpus callosum). As a result, registering tissues near the brain surface is more difficult than registering tissues around the brain center. In the light of this fact, RBFs are designed to be estimated in the ascending order of their distances from the brain center. Registering around the brain center first may warp the cortex area to better initial positions, potentially leading to better estimation and thus higher accuracy.

3.4.2 Optimization Algorithm

The symmetry of the objective function alone still does not guarantee the symmetry of the whole registration algorithm. To make the registration algorithm symmetric, all elements in the algorithm must be unbiased toward the order within the image pair, i.e., which image in the image pair is the target image and which image is the source image. Therefore, a symmetric optimization algorithm is necessary in our work.

The optimization algorithm used in our work is a modified version of the downhill simplex method. Downhill simplex method [26] is an optimization algorithm that is also used by BIRT. This method is efficient but is rather unstable in terms of the initial positions of the simplex points. A slight change in the initial positions of simplex points may generate a very different result. Furthermore, the original downhill simplex method initializes the simplex points in a random manner. As a result, BIRT cannot reproduce the same registration result in different trials using the same pair of image. This makes BIRT an unstable algorithm. To solve this problem, we use the gradient of the objective function at the origin of the orientation coefficient ($\alpha = [0\ 0\ 0]^T$) to initialize the simplex points. By doing so, we can ensure the symmetry of the optimization algorithm. An example is shown in Figure 3.7. According to the result of our expreiment, this symmetric downhill simplex method has similar speed and accuracy to the original downhill simplex method using random initial simplex points.

3.5 Incorporation of Affine Registrations

In this work, we mainly focused on designing a symmetric non-rigid registration algorithm. However, a symmetric affine registration algorithm is also necessary for a complete symmetric registration framework, since a registration process typically consists of an affine registration followed by a non-rigid registration. Therefore, it is important for the proposed algorithm to incorporate the results of symmetric affine registrations.

We must point out that it is extremely crucial to choose a proper manner to incorporate the result of a symmetry affine registration. Improper collaboration with affine registration result will lead to asymmetry of the overall registration framework even if both the affine and non-rigid registration algorithms are symmetric. However, few previous researches has covered this issue as to combine the results of affine registration while maintaining symmetry.

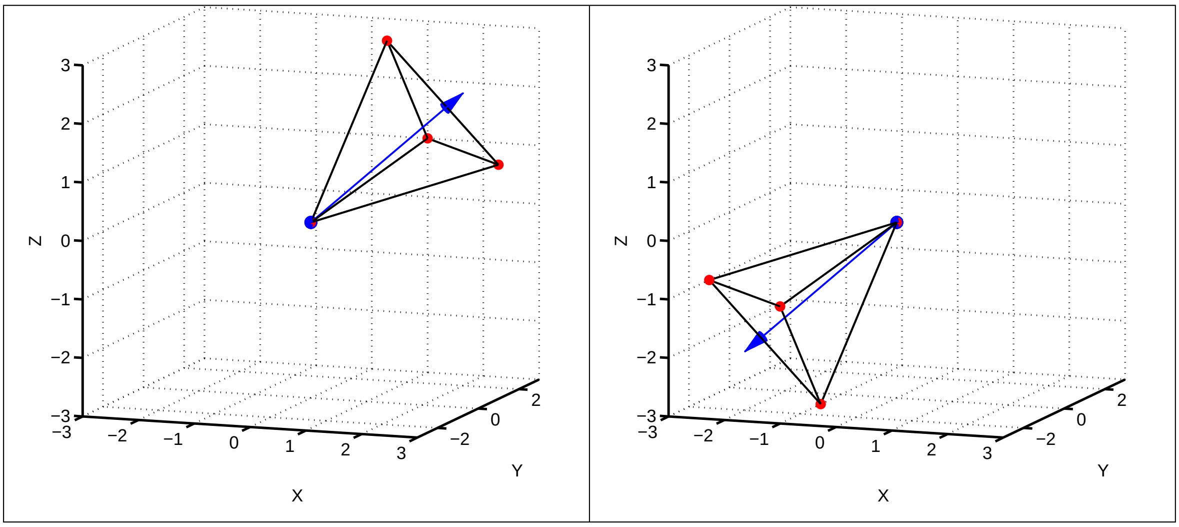

Figure 3.7: An illustration of symmetric initialization of downhill simplex method. The initial simplex points are determined by the gradient direction at the origin of the orientation coefficient.

To understand this problem in a more intuitive perspective, we can view the registration process as a processing pipeline which consists of a sequence of compositions of affine transformations and non-rigid transformations. Inversion of a registration result can be seen as a result from the reverse of the original processing pipeline. To achieve symmetry for the overall registration (provided that all registration algorithms are symmetric), the processing pipeline must be symmetric as well, i.e. the reversed pipeline must be equivalent to the original pipeline.

We give an example in the following to show how a asymmetric processing pipeline of framework affects the symmetry of the final results. For registration form image A to image B, we denote $\Phi^{(l)}_{A\rightarrow B}$ and $\Phi^{(g)}_{A\rightarrow B}$ as the resultant mapping function of non-rigid and affine registration respectively. We assume that these registration algorithms are perfectly symmetric, i.e. , $\Phi^{(g)}_{B\rightarrow A} = (\Phi^{(g)}_{A\rightarrow B})^{-1}$ and $\Phi^{(l)}_{B\rightarrow A} = (\Phi^{(l)}_{A\rightarrow B})^{-1}$. Typically, the registration of a pair of images involves an affine registration followed by a non-rigid registration:

$$\Phi_{A\rightarrow B} = \Phi^{(l)}_{A\rightarrow B} \circ \Phi^{(g)}_{A\rightarrow B} \tag{3.23}$$

This conventional framework is asymmetric. The inversion of the backward registration (registering image B to image A) results in a transformation which consists of a non-rigid transformation followed by a affine transformation:

$$(\Phi_{B\to A})^{-1} = (\Phi^{(l)}_{B\to A}\circ\Phi^{(g)}_{B\to A})^{-1} = (\Phi^{(g)}_{B\to A})^{-1}\circ(\Phi^{(l)}_{B\to A})^{-1} = \Phi^{(g)}_{A\to B}\circ\Phi^{(l)}_{A\to B} \tag{3.24}$$

This inversion is not equivalent to the pipeline of the forward registration ($\Phi_{A\to B} = \Phi^{(l)}_{A\to B}\circ \Phi^{(g)}_{A\to B}$). Instead, it belongs to a totally different processing pipeline than the processing pipeline of a registration. As a result, it does not impose symmetry to the final results even if all registration algorithms are symmetric.

To ensure the overall symmetry when incorporating with the result of symmetric affine registrations, a framework inspired by the concept of ”half-way space” [27] is used. This concept involves warping both two images toward one another by half-way simultaneously instead of warping one image onto the other. For a registration framework using half-way space, it estimates the transformation such that the objective function will be minimized after the two images are warped by its square root and the inversion of its square root respectively:

$$\Phi = \arg\min_{\Phi} E(\Phi^{0.5}(I_s), \Phi^{-0.5}(I_t), I) \tag{3.25}$$

I without lower index denotes an identity mapping function. The overall transformation consists of an affine transformation followed by a non-rigid transformation. Combining these two steps of registration yields:

$$\begin{cases} \Phi^{(g)} = \arg\min_{\Phi^{(g)}} E((\Phi^{(g)})^{0.5}(I_s), (\Phi^{(g)})^{-0.5}(I_t), I) \\ \Phi^{(l)} = \arg\min_{\Phi^{(l)}} E((\Phi^{(l)})^{0.5}\circ(\Phi^{(g)})^{0.5}(I_s), (\Phi^{(l)})^{-0.5}\circ(\Phi^{(g)})^{-0.5}(I_t), I) \end{cases} \tag{3.26}$$

Figure 3.8a gives an intuitive illustration of ((3.26)). For an affine transformation, the square root of the transformation function can be obtained using the square root of the transformation matrix ($[(\Phi^{(g)})^{0.5}\ \ 1]^T = T^{0.5}\,[x\ \ 1]^T$). And for a non-rigid diffeomorphic transformation, the square root of a diffeomorphism can be found by dividing its Lie Algebra (which is the velocity field in our case) by two ($(\exp V)^{0.5} = \exp(0.5V)$).

In practice, the simultaneous warping of two images can be simplified into only the warping of the source image based on the hypothesis of $E(\Phi(I_s), \Gamma(I_t)) \approx E(\Gamma^{-1}(\Phi(I_s)), I_t)$:

$$\begin{cases} \Phi^{(g)} = \arg\min_{\Phi^{(g)}} E(I_s, I_t, \Phi^{(g)}) \\ \Phi^{(l)} = \arg\min_{\Phi^{(l)}} E(I_s, I_t, (\Phi^{(g)})^{0.5} \circ \Phi^{(l)} \circ (\Phi^{(g)})^{0.5}) \end{cases} \tag{3.27}$$

And the overall registration result will be $(\Phi^{(g)})^{0.5} \circ \Phi^{(l)} \circ (\Phi^{(g)})^{0.5}$. A simple illustration of this framework is shown in Figure 3.8b. This processing pipeline is symmetric and can thus ensure the overall symmetry of the registration framework:

$$\begin{aligned} & (\Phi_{B\to A})^{-1} \\ = \; & ((\Phi^{(g)}_{B\to A})^{0.5} \circ \Phi^{(l)}_{B\to A} \circ (\Phi^{(g)}_{B\to A})^{0.5}))^{-1} \\ = \; & (\Phi^{(g)}_{B\to A})^{-0.5} \circ (\Phi^{(l)}_{B\to A})^{-1} \circ (\Phi^{(g)}_{B\to A})^{-0.5}) \\ = \; & (\Phi^{(g)}_{A\to B})^{0.5} \circ \Phi^{(l)}_{A\to B} \circ (\Phi^{(g)}_{A\to B})^{0.5} \\ = \; & \Phi_{A\to B} \end{aligned} \tag{3.28}$$

Therefore, the optimization formula in (3.20) can be re-formulated as follows:

$$\alpha_i = \arg\min_{\alpha_i} E(I_s, I_t, (\Phi^{(g)})^{0.5} \circ \exp(V_{i-1} + \alpha_i \rho_i) \circ (\Phi^{(g)})^{0.5}) \tag{3.29}$$

In addition to maintaining symmetry, this framework also benefits the registration greatly in other aspects. First and foremost, generating warped images involves re-sampling from the original image, which inevitably comes with some extent of degradation of image quality. As a result, using affine matrix yields better accuracy of the registration result than directly using affine-transformed images due to better image quality. Our experiment has shown that using initial affine matrices instead of affine-transformed images does yield improvement in the accuracy of registrations. In addition, if affine matrices are used, the final warped images will have better quality since only a single time of re-sampling is involved instead of two.

3.6 A Hierarchical Framework

In our work, a hierarchical framework is used to increase the accuracy and stability. The resolutions of the images and the support extents of RBFs are divided in a coarse-to-fine manner. In the initial step, transformations are initially roughly estimated using

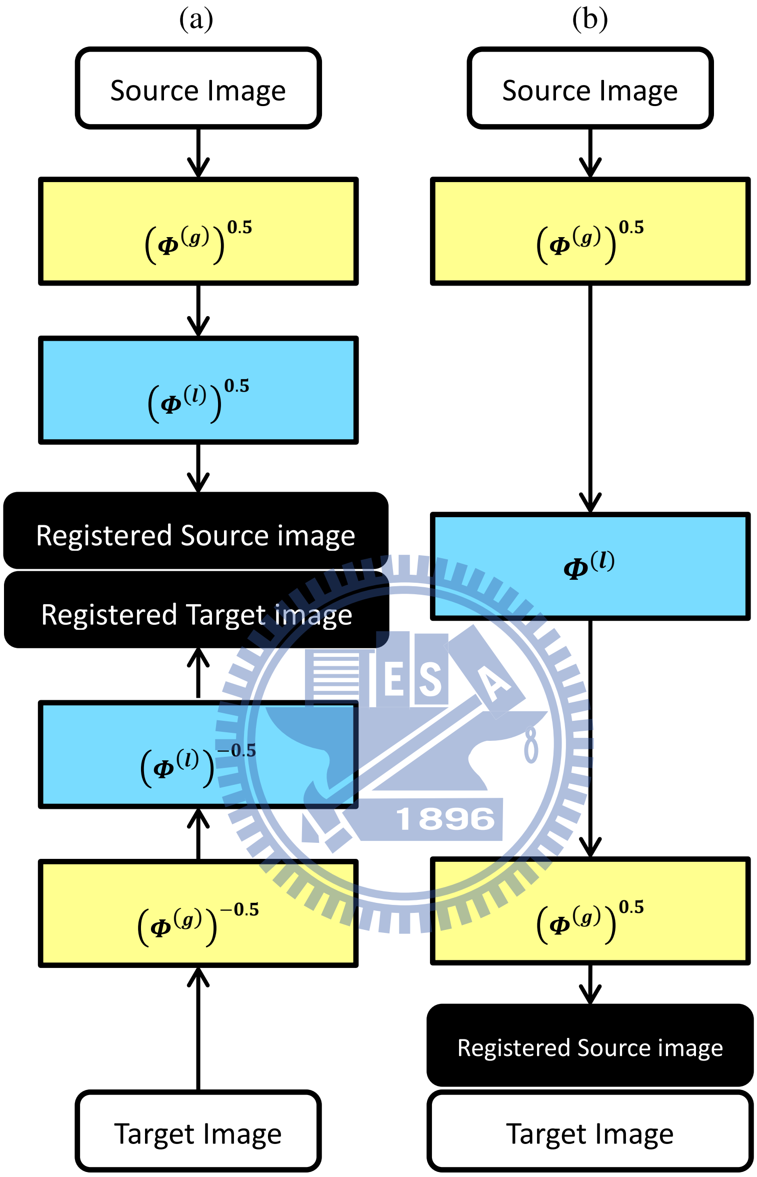

Figure 3.8: An illustration of the proposed symmetric framework for incorporating affine registration.(a) This framework is equivalent to transforming both images simultaneously during each step of registration. (b) The simplified framework which involves only the transformation of the source image.

low-resolution images and few RBFs with wider support extents. These coarse estimations are then gradually refined in later steps through increasingly detailed estimations (by using higher-resolution images and more RBFs with smaller support extents). A similar framework is used in other previous works [22, 31].

Applying the concept of hierarchical framework to (3.4), the overall velocity field can be formulated as follows:

$$\log(\Phi) = V_{L,K_j} = \sum_{j=1}^{L} \sum_{i=1}^{K_j} \alpha_{j,i}\rho_{j,i}(x) \tag{3.30}$$

Where:

$$\rho_{j,i}(x) = \Psi_{S_{j,i}}(\|x - c_{j,i}\|) \tag{3.31}$$

L specifies the total number of scale level, and K_j denotes the number of local diffeomorphisms at level j. $V_{l,m}$ is the cumulative sum of all RBFs from the start to the i^{th} RBF at level m:

$$V_{l,m} = \sum_{j=1}^{l-1} \sum_{i=1}^{K_j} \alpha_{j,i}\rho_{j,i}(x) + \sum_{i=1}^{m} \alpha_{l,i}\rho_{l,i}(x) \tag{3.32}$$

The center and the support extent of each RBF in each level are specified by the following manner. First, the algorithm finds a minimum bounding cube of the union of the brain region of source and the target image ($B_s \cup B_t$), so that such bounding cube encompasses all brain tissues when applied to each of the images. Brain regions is generally given by the brain masks from the preprocessing of skull-stripping (In our experiment, we simply used the skull-stripped brain images as the brain masks, i.e. $B_s = \{x|I_s(x) > 0\}$, $B_t = \{x|I_t(x) > 0\}$, since skull-stripped images intrinsically contains the information of their brain masks). The width of such bounding cube is denoted as W. In each scale level j, this minimum bounding cube is divided into 8^{j-1} equally-volumed cubes with width $\frac{W}{2^{j-1}}$, and a RBF is deployed at the center of each cube. The support extent of each RBF at each level is set to be proportional to the edge length of their corresponding cube with a fixed ratio r, i.e., $\forall i, S_{j,i} = \frac{rW}{2^{j-1}}$. r must be chosen in a way such that the support extent of every RBF completely covers its corresponding cube. Larger r results in larger overlaps between different RBFs and higher accuracy, since small-scaled local deformations can propagate each other to represent large deformations.

Notice that in some circumstances the evaluation point set of a certain weighted RBF may not contain any brain tissue in neither of the source nor the target image. In these cases, the estimated weighted RBF will not provide any valid information. So it is better to skip the estimation when no brain tissue is in the both $B_s(\Omega)$ nor $B_t(\Omega)$. In the proposed algorithm, the estimation of a local diffeomorphism is skipped if its evaluation point set (of both the forward and inverse transformation) does not intersect with the union of brain regions $B_s \cup B_t$. Also, brain images are re-sampled hierarchically using pyramid representation. The number of levels of the image pyramid (p) is a user-specified parameter. For each scale level l, the downsampling factor (M) of the image used for estimation is:

$$M = 2^{p-l+1} \tag{3.33}$$

Along with variable support extents, the number of the voxels within the support of $\rho_{j,i}$ is approximately:

$$\forall i, card(supp(\rho_{j,i})) \approx \frac{4}{3}\pi r W 2^{-p} \tag{3.34}$$

According to (3.34), the number of voxels in the support of any RBF is independent to the scale level. As a result, using down-sampled images can increase the speed of optimizations, and the stability can be increased by this multi-resolution set-up as well.

The vector coefficient of each RBF is estimated through the optimization algorithm consecutively. For an RBF $\rho_{j,i}(x)$, the optimization estimates the optimal vector coefficient $\alpha_{j,i}$ that lead to minimum objective function when this weighted RBF is added to the cumulative velocity field, $V_{j,i-1}$:

$$\alpha_{j,i} = \arg\min_{\alpha_{j,i}} E(I_s, I_t, (\Phi^{(g)})^{0.5} \circ \exp(V_{j,i-1} + \alpha_{j,i}\rho_{j,i}) \circ (\Phi^{(g)})^{0.5}) \tag{3.35}$$

After optimization, the estimated weighted RBF is then added to the cumulative velocity field:

$$V_{j,i} \leftarrow V_{j,i-1} + \alpha_{j,i}\rho_{j,i} \tag{3.36}$$

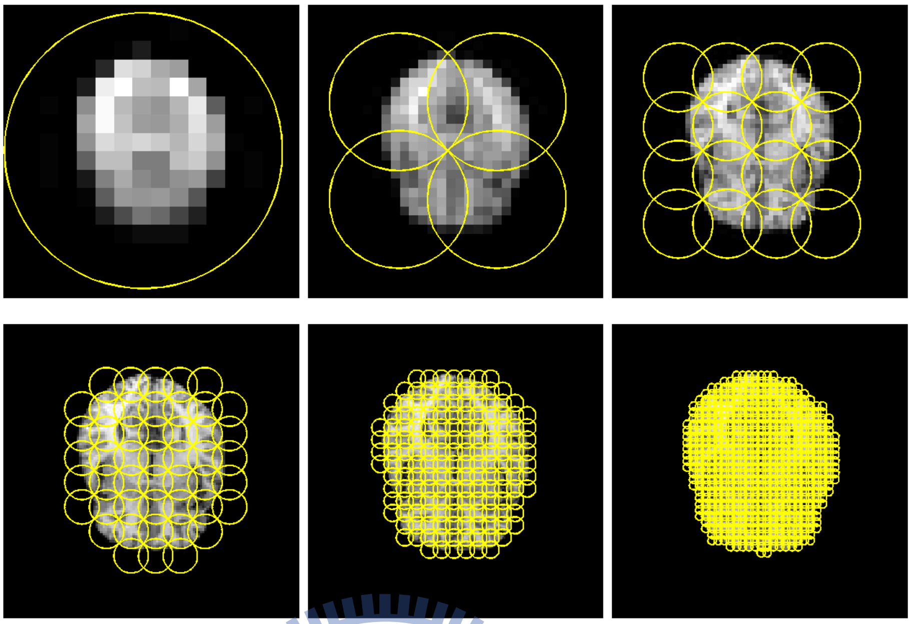

Figure 3.9: An illustration of the proposed hierarchical framework. The yellow circle represents the supports of RBFs. In this framework, the transformation is first roughly estimated using low-resolution images and large RBFs. As the level increases, the transformation is gradually refined by using more detailed images and smaller RBFs. The RBFs are placed along grid points. The estimation of the coefficient of an RBF is skipped if the evaluation point set of this RBF does not contain any brain tissues.

RBFs are estimated and added to the cumulative velocity field consecutively from lower level to higher level. In addition, RBFs in the same level are estimated in the ascending order of the distance from the brain center, as described in Section 3.4.1. This is repeated until the user-specified scale level is reached. The proposed hierarchical framework is illustrated in Figure 3.9.

The algorithm is summarized in Table 3.1.

Table 3.1: Summary of proposed algorithm

```
$V \leftarrow 0$
Find the minimum bounding cube of $B_s \cup B_t$ and its width $W$
**For** scale level $j = 1 : L$
    Calculate RBF support extent in level j:$\forall i$, $S_{j,i} = \frac{rW}{2^{j-1}}$
    **For** RBF number $i = 1 : K_j$
        Find the evaluation point set $\Omega_{j,i}$ and $\Omega_{j,i}^{\text{inv}}$
        **If** $(B_s \cup B_t) \cap (\Omega_{j,i} \cup \Omega_{j,i}^{\text{inv}}) \neq \emptyset :$
            $\alpha_{j,i} = \arg\min_{\alpha_{j,i}} E(I_s, I_t(\Phi^{(g)})^{0.5} \circ \exp(V + \alpha_{j,i}\rho_{j,i}) \circ (\Phi^{(g)})^{0.5})$
            $V \leftarrow V + \alpha_{j,i}\rho_{j,i}$
The result of non-rigid registration is: $\Phi^{(l)} = \exp(V)$
The result of the overall registration is: $\Phi = (\Phi^{(g)})^{0.5} \circ \Phi^{(l)} \circ (\Phi^{(g)})^{0.5}$
```

Chapter 4

Implementation Issues

There are some issues in detail that should be treated carefully when implementing the algorithm. For the readability of this thesis, these detailed issues were discussed in this separate section.

4.1 Solving Partial Differential Equations

The exponentiation of velocity field involves solving the PDE of the flow numerically. In general, the numerical solution of a PDE is found by incrementing the result of many small consecutive time steps of length h. For example, the simplest way of solving PDEs, the Euler method, is as follows (for simplicity, we denote $x^{(t)}$ as the location of x at time t):

$$\begin{aligned} x^{(0)} &\leftarrow x \\ x^{(t+h)} &= x^{(t)} + hV(t, x^{(t)}) + O(h^2) \\ \varphi_V(x,1) &\leftarrow x_1 \end{aligned} \tag{4.1}$$

The velocity field in the equation above has two variables which indicates time and location respectively. Since the velocity field in our case is time-invariant, the first variable indicating time can be ignored. The second term in the last equation in (4.1) indicates the numerical error. It indicates how rapidly the error drops when the step size decreases. Larger power in the error term is more desirable since the error decreases more quickly when using smaller step sizes.

The $O(h^2)$ error term of the Euler method makes it impractical in applications with strong demands for speed and accuracy. Many other methods can solve PDEs with higher accuracies than the Euler method. Our implementation uses the modified midpoint method, which can achieve $O(h^3)$ error term with only $n+1$ steps. Euler method and fourth-order Runge-Kutta method are also available in our algorithm. The modified midpoint method is as follows:

$$\begin{aligned} x^{(0)} &\leftarrow x \\ x^{(h)} &= x^{(0)} + hV(t, x^{(0)}) \\ x^{(t+h)} &= x^{(t-h)} + hV(t, x^{(t)}) + O(h^3) \\ \varphi_V(x,1) &\leftarrow \tfrac{1}{2}(x_1 + x^{(1-h)} + hV(1, x_1)) \end{aligned} \tag{4.2}$$

Fourth-order Runge-Kutta method can achiever $O(h^5)$ with $4n$ steps. It is formulated as follows:

$$\begin{aligned}
x^{(0)} &\leftarrow x \\
k_1^{(t)} &= hV(t, x^{(t)}) \\
k_2^{(t)} &= hV(t, x^{(t)} + \tfrac{k_1}{2}) \\
k_3^{(t)} &= hV(t, x^{(t)} + \tfrac{k_2}{2}) \\
k_4^{(t)} &= hV(t, x^{(t)} + k_3) \\
x^{(t+h)} &= x^{(t)} + \tfrac{k_1}{6} + \tfrac{k_2}{3} + \tfrac{k_3}{3} + \tfrac{k_4}{6} + O(h^5) \\
\varphi_V(x, 1) &\leftarrow x_1
\end{aligned} \tag{4.3}$$

Smaller step size leads to higher accuracy while increases the computation time. By default, our implementation uses 16-time-step modified midpoint method to solve the PDE. This proved to be adequate to achieve sub-voxel accuracy of symmetry in the proposed algorithms.

4.2 Re-sampling Algorithms

Image re-sampling is a frequently-used operation which greatly affects the quality of the result. All images in our implementation are only discrete approximations of continuous scalar fields and, as a result, finding the intensity value on a certain point usually involves some form of re-sampling (since, in most cases, this point does not fall on the lattice points with specified intensity values). Similarly, the flow fields as well as the mapping functions in our implementation are only discrete approximations of continuous vector fields. As a result, the implementation of registration algorithms usually entails a large number of re-sampling. For example, each step of calculation of the flow involves finding the velocity at a given point through re-sampling of the flow field. Also, the calculation of CR requires the intensity of each warped lattice point (Recalling (3.13) and (3.14)). The image intensity at each of these points are calculated through image re-sampling since they usually do not fall on the fixed lattice points. In addition, after the registration is done, the warped source image is generated through re-sampling from the original source image. This dependency of our algorithm on re-sampling implies that the speed and accuracy of our algorithm is also highly dependent on the choice of the re-sampling algorithm. Our goal is to choose a re-sampling algorithm which is as accurate as possible while not degrading the speed of our algorithm too much.

Four kinds of re-sampling algorithms were implemented and tested in our work. This includes nearest-neighbor interpolation, trilinear interpolation, tricubic interpolation and sinc interpolation. These interpolation will be explained and discussed in following subsections.

4.2.1 Nearest Neighbor Interpolation

Nearest-neighbor interpolation is an algorithm which choose the intensity of the nearest voxel to the point to be resampled as its estimated intensity. In here, the actual image intensity at fixed lattice points is denoted as $I(\cdot)$, and the estimated image intensity obtained through nearest neighbor is denoted as $I_{nn}(\cdot)$. Assuming the point to be re-sampled is at location $u \in \Re^3$, and the location of the nearest voxel of u is denoted as $\mathrm{NN}(u)$, the nearest neighbor interpolation can be formulated as:

$$\begin{aligned} \mathrm{NN}(u) &= \arg\min_{x \in \Omega} \|x - u\|_2 \\ I_{nn}(u) &= I(\mathrm{NN}(u)) \end{aligned} \tag{4.4}$$

This re-sampling algorithm can preserve the original voxel intensities and, due to its simplicity, has the highest speed among all interpolation algorithms. However, the quality of the images degrade dramatically after re-sampling when using this algorithm (see Figure 4.2b). As a result, nearest neighbor algorithm is seldom used in re-sampling MR images as well as flow fields and mapping functions. However, it is very useful when warping the label images since it can preserve the original label numbers.

4.2.2 Trilinear Interpolation

Trilinear interpolation involves consecutive linear interpolations along each dimension using the $2^3 = 8$ adjacent voxels surrounding the point to be resampled. For a point to be resampled $u = [x, y, z]^T$, we denote x_0 as the X coordinate of the closest X-grid below x, and x_1 as the X coordinate of the closest X-grid above x. y_0, y_1, z_0 and z_1 are all defined in similar manners. The 8 adjacent voxels are denoted in the form of $u_{a,b,c} = [x_a, y_b, z_c]^T$, where $a, b, c \in \{0, 1\}$ (e.g. $u_{0,1,0} = [x_0, y_1, z_0]^T$, $u_{1,0,1} = [x_1, y_0, z_1]^T$). The interpolated intensity at point u using trilinear interpolation is denoted as $\bar{I}_l(u)$. trilinear interpolation involves consecutive linear interpolation along X,Y and Z dimension. First, the interpolation is done along X dimension to obtain the interpolated intensity on $u_{b,c} = [x, y_b, z_c]^T, b, c \in \{0, 1\}$:

$$\forall b, c \in \{0, 1\}, \bar{I}_l(u_{b,c}) = (1 - w_x)I(u_{0,b,c}) + w_x I(u_{1,b,c}) \tag{4.5}$$

where:

$$w_x = \frac{(x - x_0)}{(x_1 - x_0)} \tag{4.6}$$

Then, the interpolation is done along Y dimension to obtain the interpolated intensity on $u_c = [x, y, z_c]^T, c \in \{0, 1\}$:

$$\forall c \in \{0, 1\}, \bar{I}_l(u_c) = (1 - w_y)I(u_{0,c}) + w_y I(u_{1,c}) \tag{4.7}$$

where:

$$w_y = \frac{(y - y_0)}{(y_1 - y_0)} \tag{4.8}$$

Finally, the interpolation along Z dimension yield the interpolated intensity on point u:

$$\bar{I}_l(u) = (1 - w_z)I(u_0) + w_z I(u_1) \tag{4.9}$$

where:

$$w_z = \frac{(z - z_0)}{(z_1 - z_0)} \tag{4.10}$$

An illustration of trilinear interpolation is shown in Figure 4.1a. Trilinear interpolation is a commonly used re-sampling method due to its simplicity and speed. However, it comes with larger degradation in image quality when compared with other more complex interpolation algorithms. Since trilinear interpolation involves the calculation of the weighted average of intensities, re-sampled images using this interpolation method are all ”blurred” in some measure (see Figure 4.2c). This drawback make trilinear interpolation less favorable if the image quality is highly-demanded or multiple times of consecutive image re-sampling are necessary.

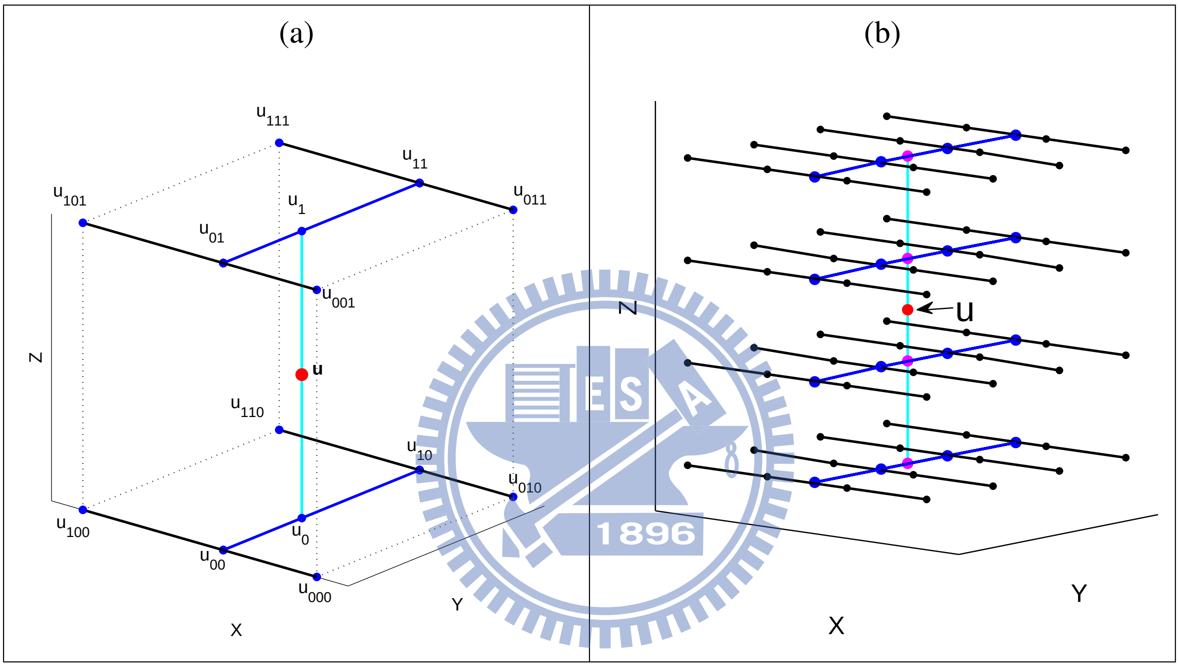

Figure 4.1: An illustration of trilinear and tricubic interpolation. Both interpolation methods involved interpolating along each dimension sequentially. (a) Trinilear interpolation. (b) Tricubic interpolation.

4.2.3 Tricubic Interpolation

Similar to trilinear interpolation, tricubic interpolation involves consecutive cubic interpolation along X,Y and Z dimension. Unlike trilinear interpolation, which requires only adjacent lattice points from the closest two grids in each dimension, tricubic interpolation needs four closest grids along each dimension in order to calculate the re-sampled intensity. In here, x_0 are x_1 are defined in the same manner as in trilinear interpolation, while x_{-1} and x_2 are denoted as the second closest X-grid below and above u, respectively. y_{-1}, y_2, y_{-1} and y_2 are all defined in similar manners. Tricubic interpolation require $4^3 = 64$ lattice points, which are denoted as $u_{a,b,c} = [x_a, y_b, z_c]^T$, where $a, b, c \in \{-1, 0, 1, 2\}$. Using these lattice points, tricubic interpolation applies single-dimensional cubic interpolation along X, Y and Z dimension consecutively. In the one-dimensional case, given I_{-1}, I_0, I_1 and I_2 as the intensity value at -1, 0, 1 and 2 respectively, the the cubic interpolation at point $x \in [0, 1]$ is formulated as :

$$\underset{x}{\mathrm{CINT}}\left(I_{-1}, I_0, I_1, I_2\right) = \frac{1}{2}\begin{bmatrix} -x^3+2x^2-x \\ 3x^3-5x^2+2 \\ -3x^3+4x^2+x \\ x^3-x^2 \end{bmatrix}^T \begin{bmatrix} I_{-1} \\ I_0 \\ I_1 \\ I_2 \end{bmatrix} \tag{4.11}$$

Tricubic interpolation can thus be formulated as:

$$\forall b, c \in \{-1, 0, 1, 2\} :$$
$$\begin{cases} u_{b,c} \equiv [x, y_b, z_c]^T, \bar{I}_c(u_{b,c}) = \mathrm{CINT}_{\left(\frac{x_1-x}{x_1-x_0}\right)}\left(I(u_{-1,b,c}), I(u_{0,b,c}), I(u_{1,b,c}), I(u_{2,b,c})\right) \\ u_c \equiv [x, y, z_c]^T, \bar{I}_c(u_c) = \mathrm{CINT}_{\left(\frac{y_1-y}{y_1-y_0}\right)}\left(\bar{I}_c(u_{-1,c}), \bar{I}_c(u_{0,c}), \bar{I}_c(u_{1,c}), \bar{I}_c(u_{2,c})\right) \\ \bar{I}_c(u) = \mathrm{CINT}_{\left(\frac{z_1-z}{z_1-z_0}\right)}\left(\bar{I}_c(u_{-1}), \bar{I}_c(u_0), \bar{I}_c(u_1), \bar{I}_c(u_2)\right) \end{cases} \tag{4.12}$$

An illustration of tricubic interpolation is shown in Figure 4.1b. Tricubic interpolation yields higher accuracy and less degradation in image quality when compared with trilinear interpolation. The re-sampled image using tricubic interpolation has less blurring artifact than that using trilinear interpolation (see Figure 4.2d).

4.2.4 Sinc Interpolation

Sinc interpolation is an interpolation method which re-samples the image in Fourier space. Sinc interpolation involves convolution of the image with a normalized sinc function. In practice, for the sake of speed, the sinc function is usually modulated by a Hanning window to limit the number of sample points used in interpolation. When using N nearest voxels (denoted as $u_i, 1 \leq i \leq N$) to the point to be re-sampled, the sinc interpolation is formulated as:

$$I_{sinc}(u) = \sum_{i=1}^{N} \left(\frac{I(u_i)sinc(d_i)Hann(d_i))}{\sum_{j=1}^{N}(sinc(d_j)Hann(d_j))} \right) \tag{4.13}$$

where d_i denotes the distance from the i^{th} voxel to the point to be re-sampled:

$$d_i = \|u_i - u\|_2 \tag{4.14}$$

$sinc(\cdot)$ denotes the normalized sinc function:

$$sinc(x) = \frac{sinc(\pi x)}{x} \tag{4.15}$$

and $Hann(\cdot)$ denotes the Hanning window function:

$$Hann(x) = 0.5 \left(1 - cos\left(\frac{2\pi x}{N-1}\right)\right) \tag{4.16}$$

The number of nearest voxels is a user-specified parameter which directly affects than accuracy and the speed of the interpolation. Empirically, we set $N = 20$ as the default number in our experiments. Sinc interpolation can generate re-sampling images with much higher quality than that from trilinear interpolation and tricubic interpolation. As can be seen in Figure 4.2e, the re-sampled image using sinc interpolation is nearly as clear as the original image, while the images re-sampled using trilinear interpolation and tricubic interpolation all have some degree of quality deterioration. However, the high quality of the sinc interpolation comes with the price of much longer computation time. For a $256 \times 256 \times 124$ T-1 MR image, resampling using sinc interpolation take slightly more than a minute while it takes only a few seconds when using trilinear or tricubic interpolation. As a result, using sinc interpolation during the optimization process will result in a long computation time for the registration process.

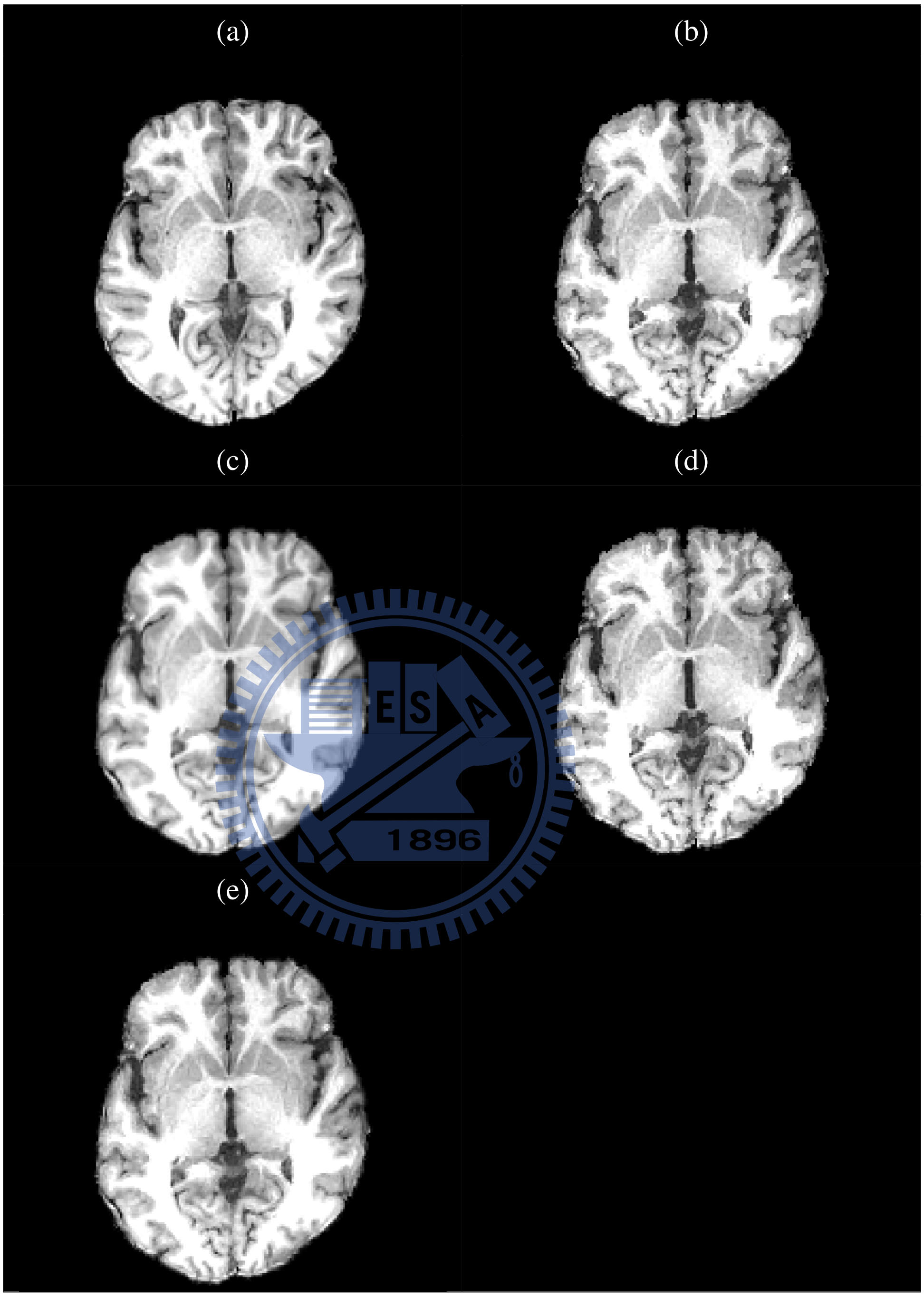

Figure 4.2: Comparison between different interpolation methods. (a) Image before interpolation (b) Nearest-neighbor interpolation (c) Trilinear interpolation (d) Tricubic interpolation (e) Sinc interpolation using 20 nearest sample points.

Although theoretically sinc and tricubic interpolation can generate re-sampled images with less distortion, in some cases the simple trilinear interpolation is more favorable even in terms of accuracy. This is especially true when re-sampling velocity fields and mapping funcitons. While the complex sinc and tricubic interpolation have less blurring artefacts, they may tend to overfit the approximated value. And since the velocity fields and mapping functions for the transformation is smooth by nature, blurring them in fact causes little harm. On the contrary, over-fitting of the velocity fields or mapping functions caused by the sinc and tricubic interpolation may induce larger error than that of the blurring effect caused by trilinear interpolation. In addition, when calculating the objective function, using blurred images surprisingly has its advantage. Since using blurred image smooths the implicit objective function, the optimization process is less likely to be trapped in a local optimum. As a result, all re-samplings involved in the optimization process are implemented using trilinear interpolation, and we only uses complex interpolation methods as options for generating warped images.

4.3 The Evaluation Point Set

There is an issue that should be treated carefully when calculating the correlation ratio. Since calculating the flow of diffeomorphism is very computationally expensive, large evaluation point set will result in an extremely slow program. Therefore, it is better to keep the evaluation point set as compact as possible. After estimating the coefficient of a RBF, say, the i^{th} RBF, it will be added with V_{i-1} to form V_i. Since the Wendland RBFs have compact supports, the difference between $\exp V_{i-1}$ and $\exp V_i$ will also be limited to a certain local region. Therefore, the best choice of evaluation point set will be the set of points whose diffeomorphic mapping is affected by adding $\alpha_i \Psi_i$ to the cumulative velocity field V_{i-1}. Including points which are unaffected by this local diffeomorphisms will only add unnecessary time cost to the algorithm.

When $i = 1$, with no previously accumulated velocity field, the evaluation point set is simply the support of the 1^{st} RBF ($supp(\Psi_1)$). But for the cases of $i > 1$, this point set are no longer limited to $supp(\Psi_1)$ since the previously accumulated velocity field V_{i-1} has to be taken into considerations. Recalling $\Phi_V(\cdot) = \varphi_V(\cdot, 1)$, a point x will be affected by the changes within $supp(\Psi_i)$ if and only if it has passed through $supp(\Psi_i)$ at any time t between $[0\,1]$ when moving along the velocity field. The evaluation point set, Ω_i , can be formulated as follows (For simplicity, the pre-calculated affine transformation is not included in the formulation):

$$\Omega_i = \{x \mid \exists\, t, 0 \le t \le 1, \varphi_{V_{i-1}}(x,t) \in supp(\Psi_i)\} \tag{4.17}$$

And since the inverse transform also needs to be evaluated, we formulate the evaluation point set for the inverse transform at step i as Ω_i^{inv}:

$$\Omega_i^{\text{inv}} = \{x \mid \exists\, t, 0 \le t \le 1, \varphi_{-V_{i-1}}(x,t) \in supp(\Psi_i)\} \tag{4.18}$$

However, it is impractical calculating the flow of all voxels in the image just to find this tiny portion of points. A more efficient way of finding the evaluation point set is through backward propagation. Instead of calculating the flows of all voxels and determine if they has passed through $supp(\Psi^{(i)})$, the evaluation point set in our work is found by tracing backward from the points in $supp(\Psi^{(i)})$ (see Figure 4.3):

$$\Omega_i = \bigcup_{x \in supp(v_i)} \bigcup_{-1 \le t \le 0} Q(\varphi_{V_{i-1}}(x,t)) \tag{4.19}$$

Similarly, the evaluation point set for the inverse transformation is:

$$\Omega_i^{\text{inv}} = \bigcup_{x \in supp(v_i)} \bigcup_{-1 \le t \le 0} Q(\varphi_{-V_{i-1}}(x,t)) \tag{4.20}$$

$Q(\cdot)$ is the quantization function that forces the evaluation points to voxel centers in current image resolution. Using this method, only the flows of the points within $supp(\Psi_i)$ need to be calculated to find the evaluation point set.

4.4 The Selection of Bin Width

The choice of the bin widths for the correlation ratio is extremely important since improper choice of bin width directly affects the performance and the validity of the correlation ratio. If the bin width is too small, the voxels in each bin may be too few to calculate

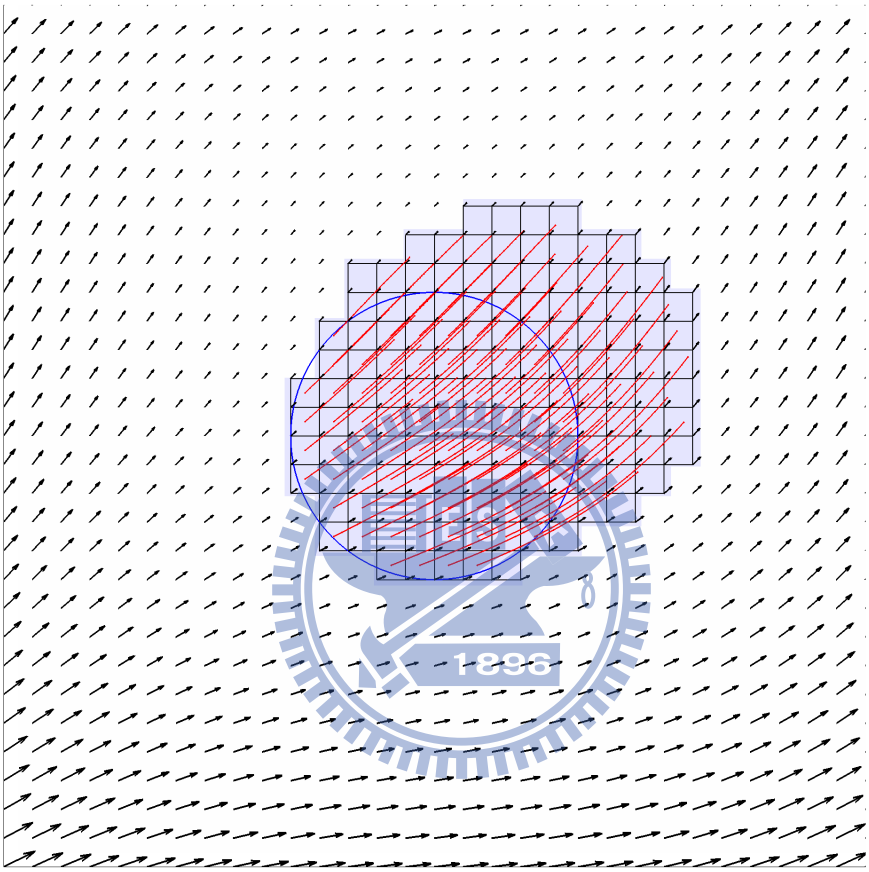

Figure 4.3: An illustration of finding the evaluation point set. The process involves tracking backwards from all voxel in the current RBF support (blue circle). A voxel is put into the evaluation point set (light blue squares) if any flow trajectory(red line) passes through that voxel.

variance correctly. And if the bin width is too large, multiple tissues may be included in a single bin. For a uni-model PDF with n sample points and estimated standard deviation $\bar{\sigma}$, Scott [34] stated that its optimal bin width is:

$$h_n = 3.49\bar{\sigma}n^{-1/3} \tag{4.21}$$

More sophisticated methods for finding optimal bin size are proposed by Wand [39] . However, due to the consideration of speed, we use the simpler estimation proposed by Scott [34]. In our algorithm, the optimal bin sizes are estimated in every step of local optimization using the evaluation point sets. The optimal bin sizes for the source and target image are estimated separately, since the source and target image may have different intensity ranges. The result shows that this mechanism of bin width selection can produce decent results.

Chapter 5

Results

In this section, numerous aspects of performance for the proposed registration algorithm are evaluated, which includes diffeomorphism, symmetry, speed and accuracy. The proposed algorithm proves to be diffeomorphic and symmetric according to the results of registration. The speed and the accuracy of the algorithm are also evaluated and compared with other methods using the frameworks provided by Klein et al. [18].

5.1 Data and Registration

The evaluations in this section are done using the LPBA40 dataset provided by the Laboratory of Neuro Imaging (LONI) at UCLA [35]. LBPA40 contains $256 \times 256 \times 124$ T1-wieghted brain MR images of 40 healthy and normal volunteers with manually delineated labels. These 40 brain images were skull-stripped in the preprocessing according to the protocols in Shattuck et al. [35]. These skull-stripped images were registered to the MNI305 atlas [15] using rigid-body transformations to correct head tilts and alignment. 56 different structures were labeled manually for each brain. These images then underwent non-uniformity correction.

The registration process for each image pair consists of a 12-parameter affine registration (FLIRT) followed by the proposed non-rigid registration. The registration process is done on all $P_{40,2} = 1560$ source-target pairs. The registration process was run on a Linux workstation with 16 Genuine Intel CPU cores and 48 GB main memory. An example of the registration result is given in Figure 5.2.

5.2 Evaluation of Diffeomorphism

First and foremost, we need to prove that the proposed algorithm conforms to the property of diffeomorphism and symmetry. Evaluation of diffeomorphism can be done using

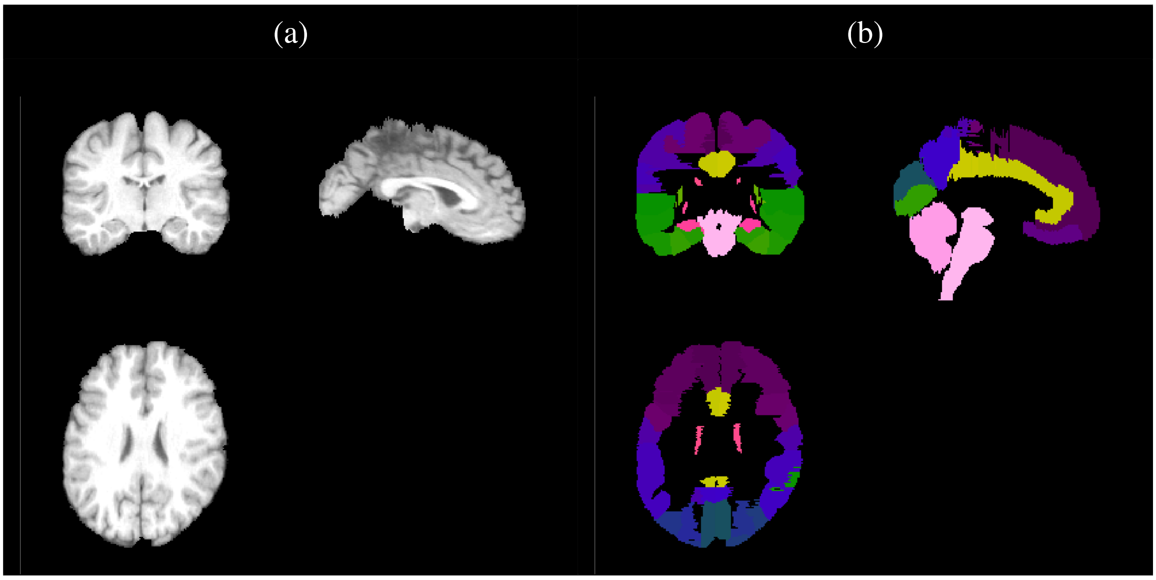

Figure 5.1: An example of LPBA40 dataset. (a) T1-weighted image (b) Labels of 56 structures

the determinants of the Jacobian matrices of the transformation functions:

$$\det J_{\Phi}(x) = \det(\nabla\Phi^{T} \circ x) = \begin{vmatrix} \frac{\partial\Phi_1(x)}{\partial x_1(x)} & \frac{\partial\Phi_1(x)}{\partial x_2(x)} & \frac{\partial\Phi_1(x)}{\partial x_3(x)} \\ \frac{\partial\Phi_2(x)}{\partial x_1(x)} & \frac{\partial\Phi_2(x)}{\partial x_2(x)} & \frac{\partial\Phi_2(x)}{\partial x_3(x)} \\ \frac{\partial\Phi_3(x)}{\partial x_1(x)} & \frac{\partial\Phi_3(x)}{\partial x_2(x)} & \frac{\partial\Phi_3(x)}{\partial x_3(x)} \end{vmatrix} \tag{5.1}$$

This measure indicates the volume ratio before and after transformations. A region with non-positive determinants indicates that the one-to-one mapping is lost [7]. For a mapping function, the error measure in terms of diffeomorphism can defined by counting the the ratio of volume with non-positive determinants among the whole image space (D). These measures are named as volume loss (VL):

$$\mathrm{VL}\,(\Phi) = \frac{card(\{x|x \in D, \det J_{\Phi}(x) \leq 0\})}{card(D)} \tag{5.2}$$

This measure is calculated for all 1560 registered source-target pairs. Table 5.1 shows the average and the standard deviation of 1560 VL values. The result indicates that non-positive determinant value is non-existent. According to this evaluation, the proposed algorithm is fully diffeomorphic at least for all 1560 registrations using LPBA40 dataset.

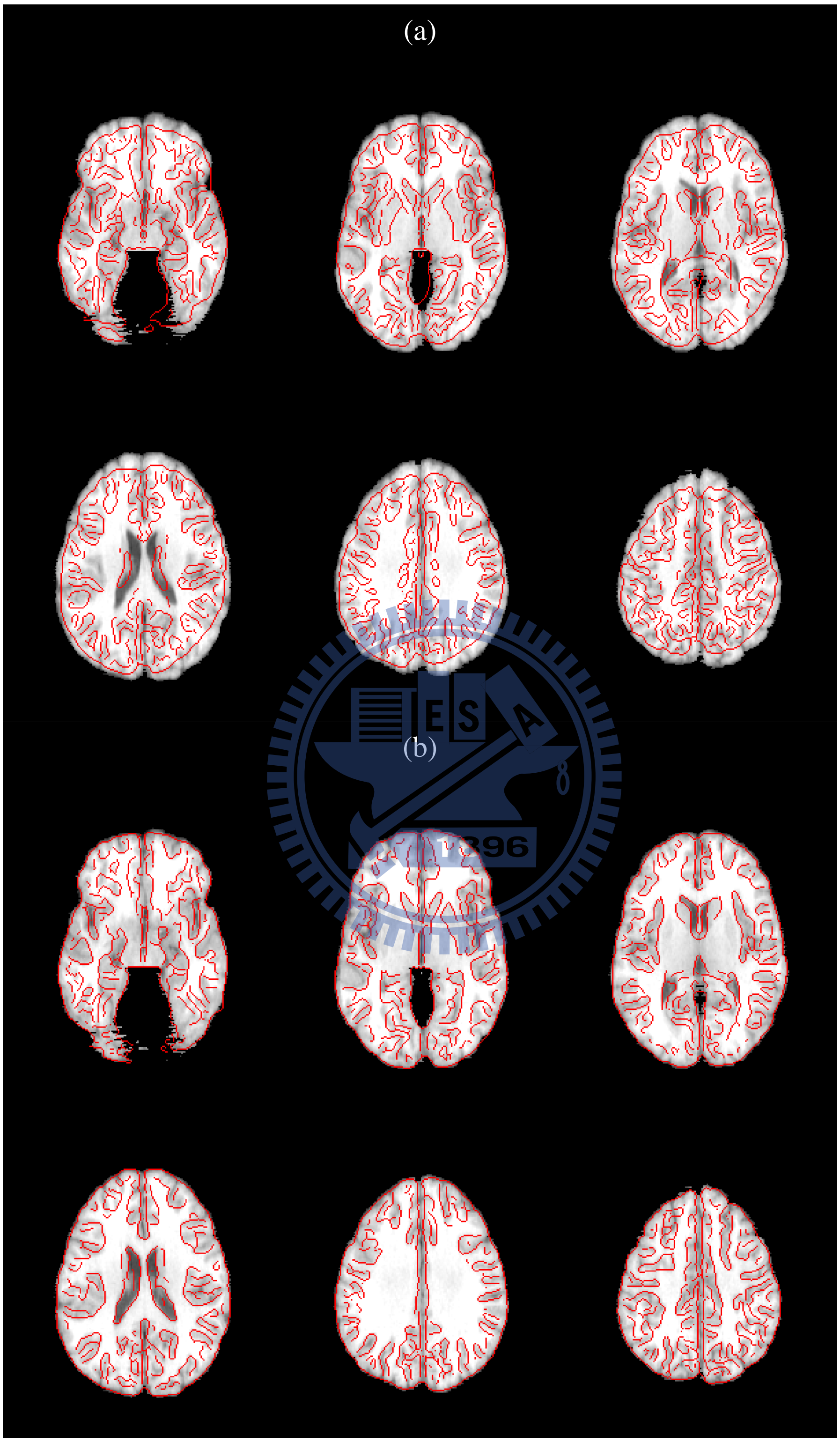

Figure 5.2: An example of registration result. The red lines are the edges of the source image. (a) Before registration (b) After registration

Table 5.1: Volume Loss values for the proposed algorithm. These values are acquired by averaging the result of 1560 registrations of LPBA40 dataset. All 0's are exact values.

number of scale levels	1 levels	2 levels	3 levels	4 levels	5 levels	6 levels
VL (averaged)	0	0	0	0	0	0
VL (std. dev.)	0	0	0	0	0	0

5.3 Evaluation of Symmetry

As mentioned, perfectly symmetric registration algorithms would yield:

$$\forall A, B, \bar{\Phi}_{B\to A} = \left(\bar{\Phi}_{A\to B}\right)^{-1} \tag{5.3}$$

Or, in another form:

$$\forall A, B, \bar{\Phi}_{A\to B} \circ \bar{\Phi}_{B\to A} = I \tag{5.4}$$

where I is an identity function. (5.4) can be rewritten in the form of logarithms:

$$\forall A, B, \log(\bar{\Phi}_{A\to B}) + \log(\bar{\Phi}_{B\to A}) = 0 \tag{5.5}$$

The ”0” here symbolizes a zero function. According to this equation, for a perfectly symmetric registration algorithm, the sum of the velocity fields for forward and backward registration should be zero for all points, i.e. the velocity field form the forward registration should be the exact negation of that from the backward registration.

In this evaluation, we used two different error measures of symmetry which are derived from (5.4) and (5.5) respectively. From (5.4), we define the residual for a point x :

$$r(A, B)(x) = \|(\bar{\Phi}_{A\to B} \circ \bar{\Phi}_{B\to A} - I)(x)\|_2 \tag{5.6}$$

For a image pair, its mean residual is the average residual of all points in the image space:

$$r^{(avg)}(A, B) = \operatorname*{avg}_{x\in I}\{r(A, B)(x)\} \tag{5.7}$$

And the maximum residual is the maximum single-voxel residual within the image space:

$$r^{(max)}(A, B) = \max_{x\in I}\{r(A, B)(x)\} \tag{5.8}$$

The error measure for symmetry derived from (5.5) is the inconsistency of velocity fields (ICV) of forward and backward registrations. This measure is calculated by the absolute sum of two velocity fields in each locations:

$$ICV(A,B)(x) = \|V_{A\to B}(x) + V_{B\to A}(x)\| \tag{5.9}$$

Similar to the residual terms, inconsistency of velocity fields can be divided into two types: averaged inconsistency of velocity fields and maximum inconsistency of velocity fields:

$$ICV^{(avg)}(A,B) = \operatorname*{avg}_{x\in I}\{ICV(A,B)(x)\} \tag{5.10}$$

$$ICV^{(max)}(A,B) = \max_{x\in I}\{ICV(A,B)(x)\} \tag{5.11}$$

In reality, perfect symmetry is difficult to achieve due to numerical errors. In theory, a velocity field belongs to a infinite dimensional vector space, which requires infinite amount of memory to store. As a result, the velocity field in our implementation is only a finite dimensional approximation for a fixed lattice. This means that the velocity in each step of calculating flows is only an approximation through interpolation of discrete grid-points. Other than the velocity field, the flows in our work are also finite-dimensional approximations, since a flow theoretically should consists of infinite number of time steps. Also, the mapping function representing the transformation are represented using fixed lattices. As a result, the function composition for calculating $\bar{\Phi}_{A\to B} \circ \bar{\Phi}_{B\to A}$ also involves resampling of the transformation functions, which also induces some extents of numerical errors. Due to these reasons, even if the estimated velocity fields for forward registration and backward registration are perfectly consistent, one cannot guarantee the residual to be zero. As a result, the interpretations of errors from these two kinds of error measures are somehow different. ICV indicates how well the registration algorithm itself conforms to the property of symmetry, and, on the other hand, the residuals better reflects the real error of symmetry in real-world applications.

A useful threshold of tolerance for residuals is the voxel width of the input images. For a symmetric registration method, the maximum residual should be always below the width of voxel for all registration results. With the cost of more memory spaces and computation time, the error for symmetry can be reduced by using more time steps when calculating flows can using velocity fields and mapping functions with higher resolutions. In this evaluation, the calculation of flow involves 16-step modified midpoint methods. The resolutions of velocity fields and mapping functions are the same as that of the input images.

In general, a registration process consists of an affine registration followed by the proposed non-rigid registration, and the symmetry of the overall registration entails the symmetry of the affine registration algorithm used as well as the symmetry of the proposed non-rigid registration algorithm. In this evaluation, we only focused on the the symmetry of the proposed algorithm given a perfectly symmetric affine registration algorithm. Since FLIRT is asymmetric, the symmetry of affine registrations is achieved only in a simulative manner. For each pair of forward affine registration (e.g. $\Phi^{(g)}_{A\rightarrow B}$) and backward affine registration (e.g. $\Phi^{(g)}_{B\rightarrow A}$), the transformation of the backward affine registration is estimated directly by the inverse transformation of the forward affine registration ($\Phi^{(g)}_{B\rightarrow A} \equiv (\Phi^{(g)}_{A\rightarrow B})^{-1}$). By this manner, the forward and inverse registration use consistent results from a single affine registration and thus can achieve simulated symmetry.

In this evaluation, all $\binom{40}{2} = 780$ pairs of forward and inverse registrations in LPBA40 data are used to evaluate the symmetry of the proposed algorithm. The result of evaluation is shown in Table 5.2. The result shows that the mean residual is far below the width of the voxels, and the largest residual among all results is still within the width of the voxels. Furthermore, the ICV is always zero in all registrations. This implies that the proposed algorithm is perfectly symmetric in estimating velocity fields. The error in the residual term is solely due to the finite-dimensional approximations of mapping functions and the finite-step approximations of flows. As a result, the residuals is guaranteed to reduce when using more accurate (i.e. more steps) calculation of flows and mapping functions in higher resolutions. Nevertheless, according to our experiments, sub-voxel accuracy can be achieved simply using the same resolution of input images and 16-step flow calculations.

Table 5.2: Error of symmetry for the proposed algorithm. These values are acquired by averaging the result of 780 pairs of forward and inverse registrations of LPBA40 dataset. The error measures used are (a) residual (b) inconsistency of velocity fields. All 0's are exact values.

(a)

number of scale levels	$r^{(avg)}$ (mm)			$r^{(max)}$ (mm)		
	avg.	std. dev.	max.	avg.	std. dev.	max.
1 levels	0.0002	0.0001	0.0005	0.0024	0.0013	0.0105
2 levels	0.0004	0.0001	0.0008	0.0114	0.0049	0.0394
3 levels	0.0014	0.0002	0.0022	0.0393	0.0179	0.1673
4 levels	0.0047	0.0006	0.0074	0.0747	0.0242	0.2747
5 levels	0.0125	0.00111	0.0153	0.2676	0.0780	0.8471
6 levels	0.0200	0.00129	0.0228	0.3802	0.0744	0.8503

(b)

number of scale levels	$ICV^{(avg)}$			$ICV^{(max)}$		
	avg.	std. dev.	max.	avg.	std. dev.	max.
1 levels	0	0	0	0	0	0
2 levels	0	0	0	0	0	0
3 levels	0	0	0	0	0	0
4 levels	0	0	0	0	0	0
5 levels	0	0	0	0	0	0
6 levels	0	0	0	0	0	0

5.4 Evaluation of Accuracy

The purpose of this section is to provide a objective comparison of accuracy between the proposed algorithm and other registration methods. To do this, the criterion of comparison must be chosen carefully. Improper choice of comparison criterion may lead to result with poor validity. For example, choosing the similarity measure used in any registration method to be compared will lead to biased comparison results since this similarity measure would favor the methods which uses it for optimization. Another example of improper comparison criterion is CR. CR is extremely biased towards blurred images since it is generally based on measuring intensity variances. A extreme case is when comparing a completely blank image with a brain image. In this case, the CR value would be 1 since the variance of the blank image is always zero. This makes CR unsuitable for comparisons between different registration methods. Since different registration methods might use different re-sampling methods or different times of re-sampling, the result of comparison might be dominated by the image quality if CR is used as the criterion for comparison.

To give a objective comparison, the evaluation of accuracy was done using the procedure proposed by Klein et al. [18]. The evaluation framework by Klein et al. [18] is generally based on manually-marked labels. For a given dataset of N images, the procedure by Klein et al. registers all $P_{40,2}$ permutations of source-target image pairs. And accuracy of each registered image pair is evaluated through the overlap agreement of labelled brain structures between the target image and the transformed source image. The target overlap (TO) is used as the overlap agreement measure in our evaluation. It measures the intersection of a similarly labelled region r in the source image S and target image T divided by the volume of the region in T. The volume is calculated by the number of voxels:

$$TO_r = \frac{|S_r \cap T_r|}{|T_r|} \tag{5.12}$$

And the target overlap of a given registration result is calculated by summing over the whole set of labelled regions:

$$TO = \frac{\sum_r |S_r \cap T_r|}{\sum_r |T_r|} \tag{5.13}$$

Distributions of the target overlap values of all 1560 registration results of LPBA40 data are shown in Figure 5.3. The median value for level-5 results (registration using 5 RBF scale levels) and level-6 results are 0.731 and 0.742 respectively, while the highest median TO value reported by Klein et al. [18] is approximately 0.725. However, a recent work by Ashburner et al. [6] has reported a median TO value of 0.753. It suggests that although our algorithm is superior than all 14 methods listed in [18], the method by Ashburner et al. [6] still has higher accuracy.

The mean TO value in each region is visualized in Figure 5.4. The regional TO values of the proposed algorithm are compared with that of other methods listed in the work of Klein et al. [18] are shown in Figure 5.5.

5.5 Evaluation of Speed

The speed for our registration algorithm depends on the number of scale level used in the registration. Increasing the total number of scale levels will increase the accuracy with the cost of longer computation time. For 256 × 256 × 124 T1-MRI images of LPBA40 dataset, more then 5 scale levels are required for usable result. The evaluation of speed was done using registration results of 100 randomly chosen LPBA40 image pairs. This 100 registrations are done sequentially in order to reflect the true computational costs (since, in practice, large amounts of registrations are usually parallelized to reduce the total computational cost). The averaged computation time of 100 5-level registrations is approximately 4 minutes. When using 6 scale levels, the averaged computation time increases for nearly 7 times to about 30 minutes. The speed performance of the proposed algorithm was compared with the reported speed of the diffeomorphic registration algorithms listed in the work of Klein et al. [18]. These algorithms includes SyN [8], DARTEL [7], JRD-fluid [11], Diffeomorphic Demons [38] and SICLE [12]. Since the speed performances reported in the work of Klein et al. was derived using similar datasets and computing environments to that of our evaluations, this comparison still gives adequate validity.

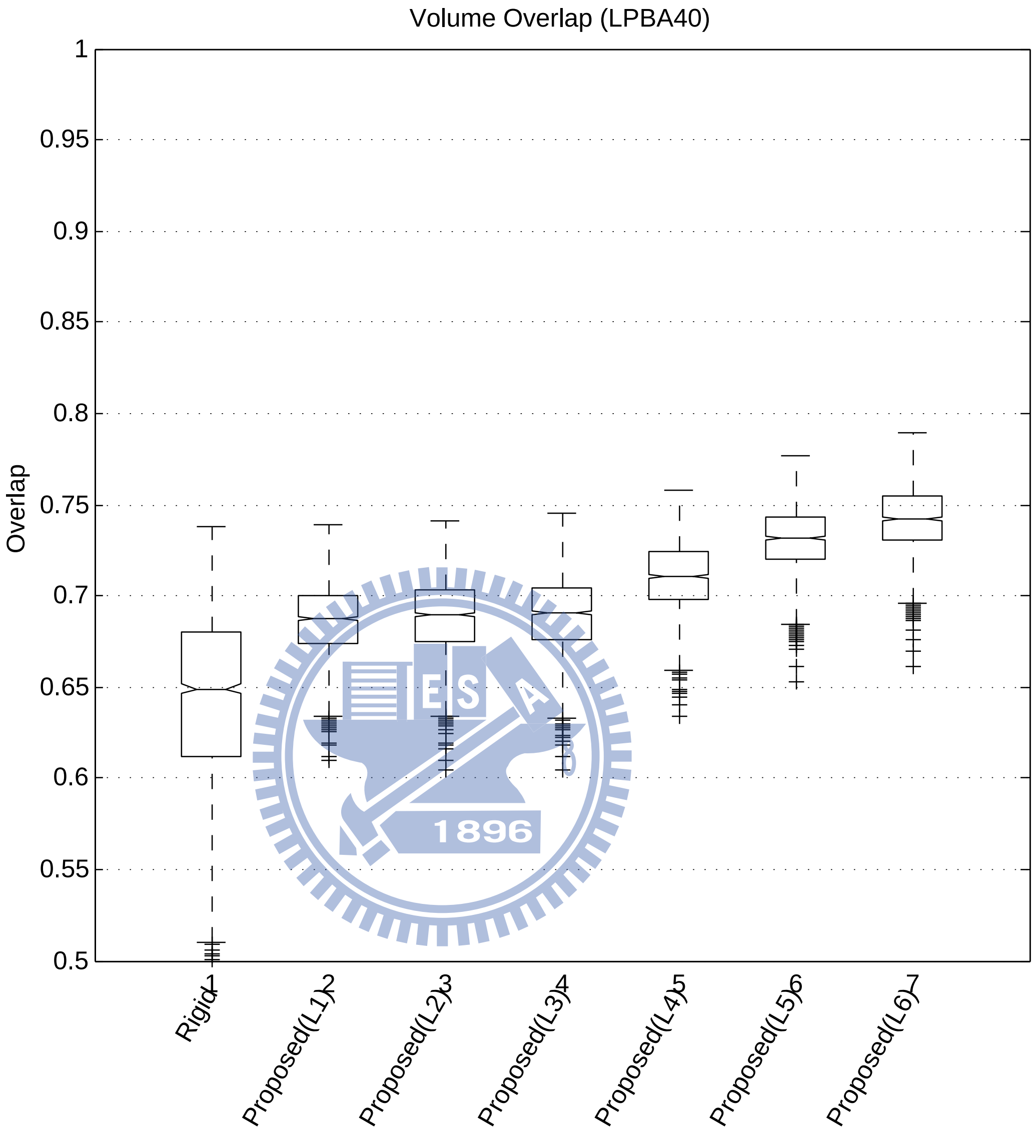

Figure 5.3: TO value for LPBA40 dataset. The line in the middle of each box represents the median TO value for all 1560 registrations, and each box represents the interquartile range. Whiskers at each end of the box extends to the most extreme value within 2.5 times the interquartile range from the median value. Values beyond 2.5 times the interquartile range from the median value are marked as outliers (+).

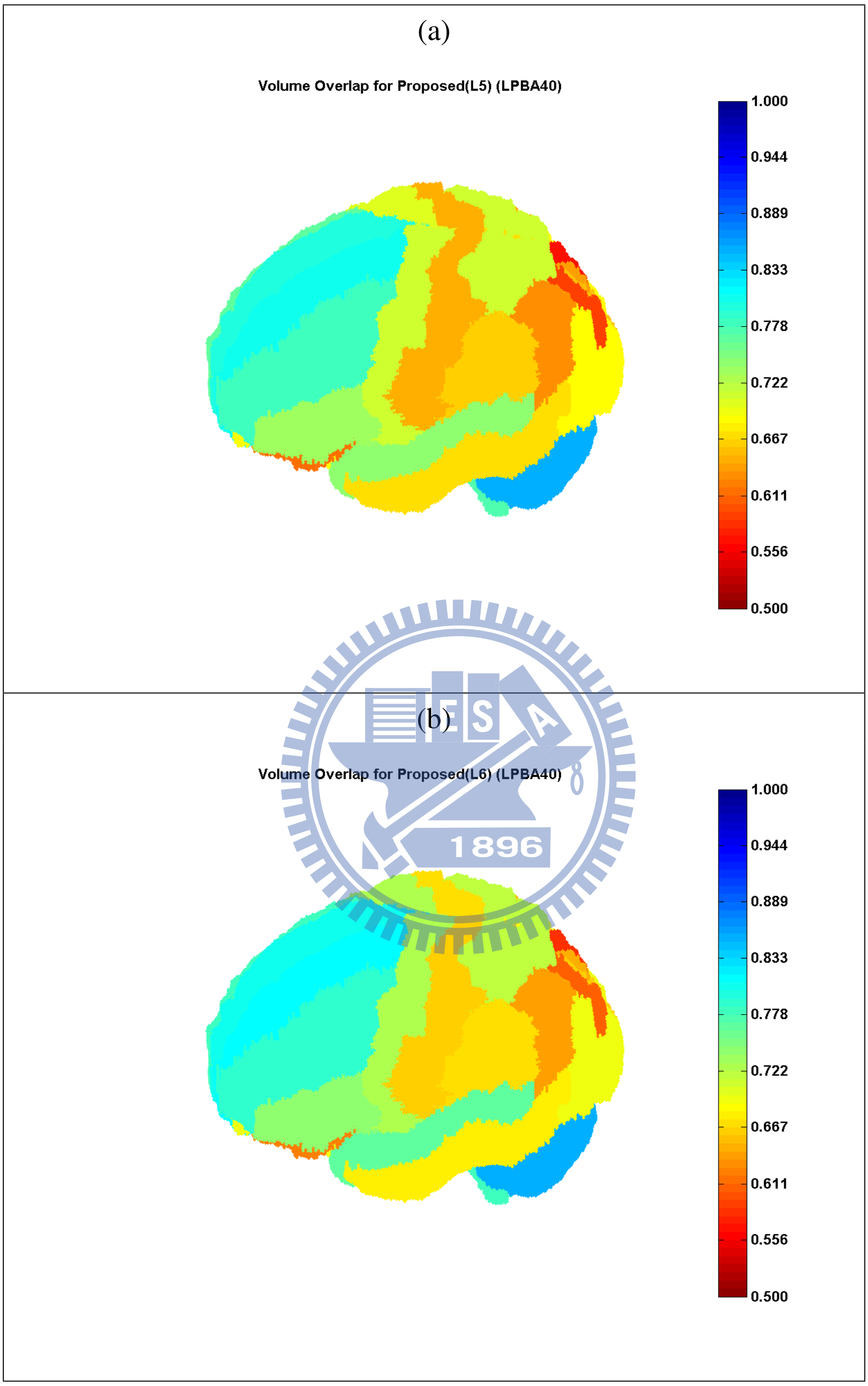

Figure 5.4: Visualization of mean TO values for each brain region. (a) Using 5 RBF levels (b) Using 6 RBF levels

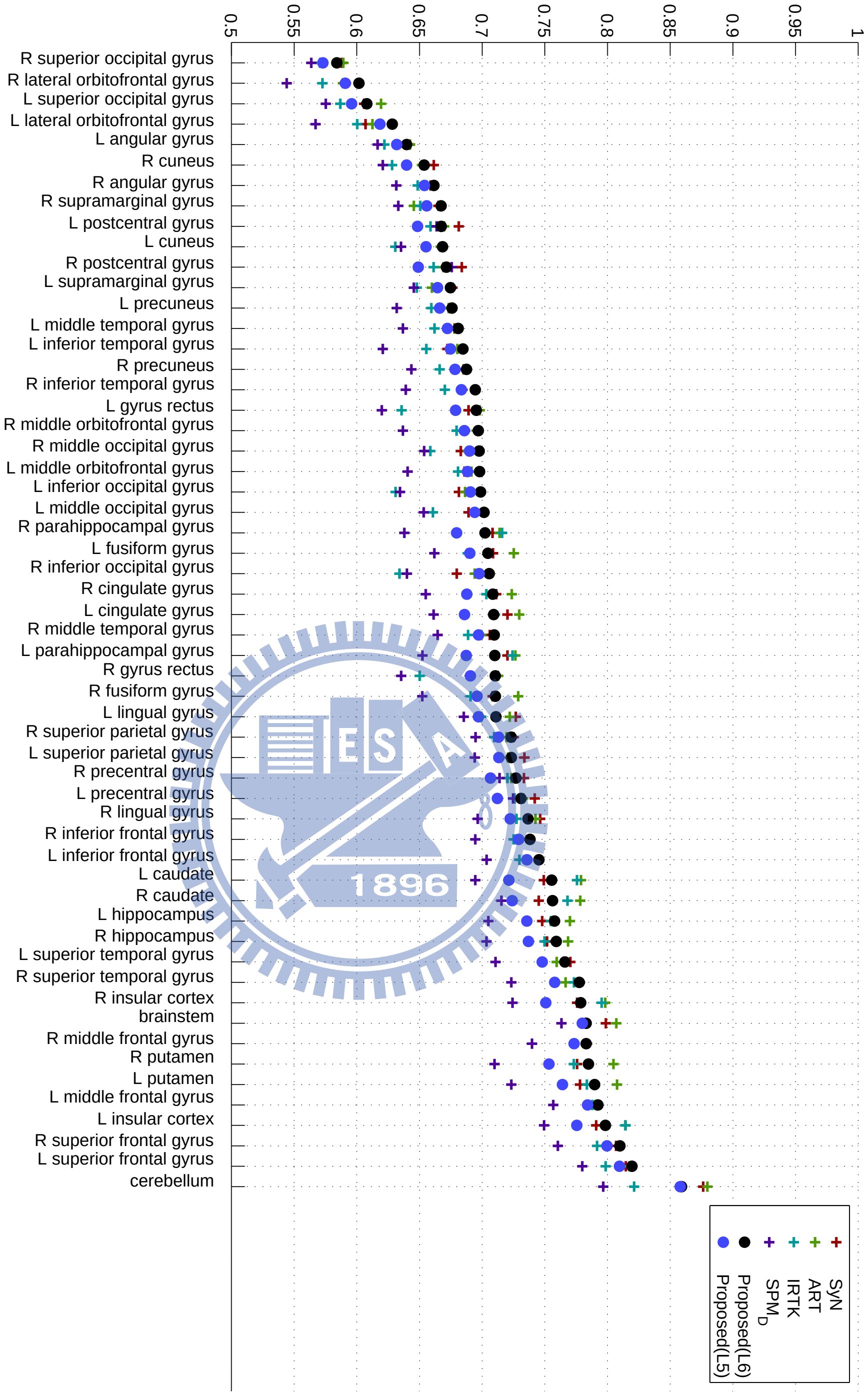

Figure 5.5: Plot of mean TO value in each region for different methods. Only methods ranked higher than 3 are listed. This plot is visualized in similar manner as in Ashburner et al., 2011.

Table 5.3 shows the comparison between the speed of the proposed algorithms the and the diffeomorphic registration algorithms listed in the work of Klein et al.. In similar computing environments, the proposed algorithm is faster than all diffeomorphic algorithms listed when using 5 scale levels. The proposed algorithm is slower than JRD-fluid and Diffeomorphic Demons when using 6 scale levels. However, accuracy should be also taken into considerations. When using more than 5 scale levels, the proposed algorithm is more accurate than all algorithms listed while having higher speed than the highest-ranked diffeomorphic algorithm in the work of Klein et al..

Multiple registration tasks can be parallelized to further reduce the total computation time. In our evaluation, the registration processes are run on a worksation with 16 CPU cores with a maximum of 14 parallel tasks allowed, and registering all 1560 image pairs takes approximately 12 hours when using 5 scale levels. When using 6 scale levels, 1560 registrations take about 60 hours.

Table 5.3: Comparison of speed with other diffeomorphic registration methods. All statistic data (including run time and mean rank) of other methods are cited from Klein et al., 2009. Lower mean rank number indicates higher accuracy. This comparison is rough but still valid since our experiments were run on similar computing environment as in Klein et al., 2009.

Algorithm	Mean rank	Run time(min)
Proposed(level 5)	n/a	∼4.5
Proposed(level 6)	n/a	∼25
SyN	1.00	77(15.1)
ART	1.00	20(1.6)
DARTEL	1.88	71.8(6.3)
JRD-fluid	2.50	17.1(1.0)
Diffeomorphic Demons	3.00	8.7(1.2)
SICLE	> 4	33.5(6.6)

Chapter 6

Discussion

6.1 Validity of Evaluation Results

The first thing we would like to emphasize is that one should remain conservative to the result of the comparison of the proposed algorithm with the reported results in the work of Klein et al. [18]. Although Klein et al. [18] has provided an accurate and convenient evaluation framework for nonlinear registration algorithms, there seems to be several inconsistencies between the result reported by Klein et al. [18] and those by Ashburner et al. [6] and our work. First, according to the rigidly-registered LPBA40 dataset provided by Klein et al., the median TO value after 9-parameter rigid registration (using FLIRT) for LPBA40 is 0.639, while the median TO value for FLIRT registration reported by Klein et al. is about 0.60. The second inconsistency is between the reported result of DARTEL by Klein et al. ($\sim$0.71) and that by Ashburner et al. [6] (0.753). Another anomaly of the result in the work by Klein et al. [18] is that the TO values in some cases deteriorate dramatically after non-rigid registration. This phenomenon is most apparent for the result of DARTEL and SPM's Normalize. Furthermore, the median TO value of SPM's Normalize is even lower than that of rigid registration (FLIRT).

Instead of concluding that these registration algorithms are inferior, we conjecture that this phenomenon is more probably due to trivial technical issues when using the algorithms (e.g. the choice of parameters, the format of the images). For example, one image in the LPBA40 dataset has a different intensity range than other images. Some registration algorithms clearly perform badly when registering images with different intensity ranges (e.g. BIRT). This assumption may explain the inconsistent results between different works. The higher performance of DARTEL reported by Ashburner et al. is more possible to be the actual performance when all technical issues are eliminated. On the other hand, the validity of the results reported by Klein et al. becomes questionable if the performances of registration algorithms are likely to be lowered due to technical issues. Therefore, it is better to use more than one evaluation frameworks to produce more credible results of evaluations.

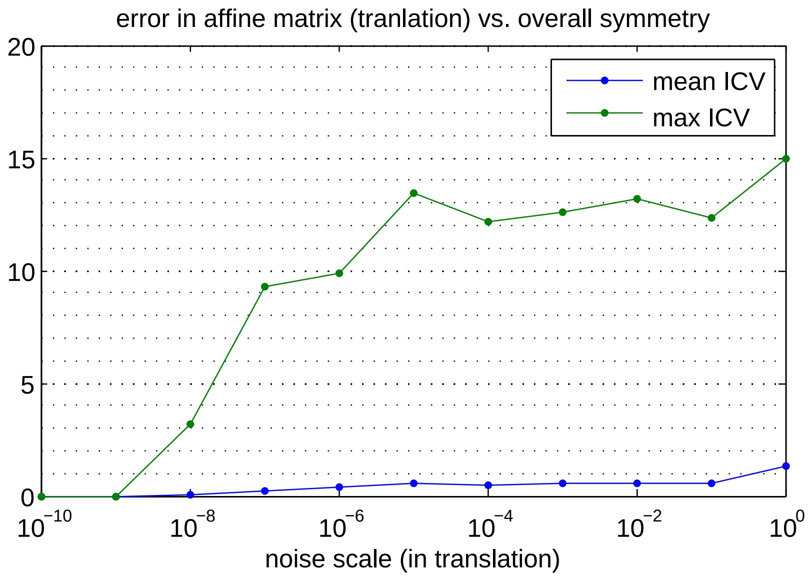

Figure 6.1: Poor robustness of symmetry of the proposed algorithm. Errors as small as 10^{-8} can lead to total loss of symmetry.

6.2 Robustness of Symmetry

We have shown that our algorithm is symmetric by itself and can also achieve symmetry when incorporating with a symmetric affine registration. However, the symmetry of the proposed algorithm is not robust against the error of symmetry in initial affine registrations. Our experiment has shown that minor errors in symmetry in affine registration will result in a total deviation from symmetry after non-rigid registration. In our experiment, an image pair is registered by proposed algorithm for 10 trials. In each trial, random noises are added to the translation components of the affine matrix. Figure 6.1 shows the relationship between the noise in the affine matrix and the averaged resultant symmetry. As can be seen, error as small as 10^{-7} in the affine transformation matrix can lead to complete loss of symmetry. This instability is extremely undesirable. If the transformation matrix is stored using single-precision floating-point format, The numerical error induced by matrix inversion may already exceeds 10^{-7} in some cases. In this situation, even if the affine registration method is perfectly symmetric, the overall symmetry may still be lost solely due to some numerical errors.

One possible cause for this instability is the optimization algorithm as well as how the optimization problem is defined in our work. Downhill simplex method is an fast optimization method but it is rather unstable in terms of the choice of the initial simplex points. In addition, the optimization problem in the proposed algorithm is modelled as a sequence of lower-dimensional optimization problems. This means that each optimization problem is dependent on previous ones. This sequential optimization set-up added with the effect of the instability of downhill simplex method may cause small errors in the affine transformation matrix to eventually accumulate to scale that seriously compromises the symmetry of the results. For example, registrations using 5 scale levels involve about 2000 optimizations of RBF coefficients, while registrations using 6 scale levels involves about 10000 optimizations. Since each RBF overlaps with its adjacent RBFs, the error of symmetry induced during the optimization of this RBF will be propagated to its adjacent RBFs, causing even larger errors when optimizing the coefficients of its adjacent RBFs. After repeating this error propagation for thousands of optimizations, the symmetry would be completely lost.

A possible solution is to use other optimization methods which are more stable than downhill simplex method, but this may compromise the speed of our algorithm. Another possible solution is to reduce the extent of error accumulation through reducing the long ”chains” of dependencies in the proposed sequential optimization framework. In our original framework, each optimization of RBF coefficient is dependent to its last previous optimization. As a result, the extent of error is proportional to be number of RBFs in the optimization ($\epsilon = O(n)$). However, we can divide the RBFs in each scale level into sets of non-overlapping RBFs. Since each RBF in a set does not overlap with other RBFs in the same set, the optimizations of the RBFs in the same set are all independent from each other. As a result, optimizing all RBFs in a non-overlapping RBF set yields no error propagation. If the optimization in each level is done set-by-set, the length of the dependencies will reduce from n to $log(n)$. The extent of accumulated error can thus be reduced to $O(log(n))$.

The second possible cause to the instability may come from the ”rugged” objective function caused by improper binning methods. This phenomenon has been reported by Maes et al. [24]. In the work by Maes et al., different binned approaches for building joint histograms were compared. A common binning approach is to use trilinear interpolation to find the re-sampled intensity of a point using its adjacent voxels, and then find the corresponding bin of the re-sampled intensity. However, trilinear interpolation may introduce new intensity values which are originally not present, which may result in unpredictable behaviors in the marginal distribution. And Maes et al. shows that the MI function using this binning method has many noises and local maxima around the optimal solution, which potentially damages the accuracy of optimizations. This phenomenon may also exist in the proposed algorithm, since the calculation of CR in our work also involves binning trilinear-interpolated intensity values. Noisy objective functions may be one of the causes for the extremely low robustness against the error of symmetry in initial affine registrations.

The solution of this problem is also proposed in the same work by Maes et al. [24]. Maes et al. proposed to use trilinear partial volume distribution (PV) as the binning method. This method does not interpolate the intensity value directly. In contrast, it uses interpolation as a way to find the ”membership” of the adjacent voxels. After the membership for each voxel is found, the corresponding bin for each voxel is added by the real-numbered membership value. Since the value of the each bin is added by a fuzzy membership instead of discrete one-or-zero count, the resultant MI function using PV is much smoother toward the change of transformation. By utilizing this concept, we may modify the binning method and the variance calculation in CR to make it compatible to fuzzy membership values. This smoother objective function may help to increase the robustness of the proposed symmetric framework.

6.3 Effect of Basis Functions on Accuracy

As mentioned in Chapter 3, we used Wendland’s RBF to model the velocity field. The compact support of Wendland’s RBF can greatly reduce the evaluation region and thus increase the speed of the proposed algorithm. However, it has some specific disadvantages.

Unlike basis functions such as cubic spline, Wendland's RBFs are independent and does not incorporate with each other when being combined to form the overall approximation of a certain function. As a result, the velocity fields modelled by Wendland's RBFs tend to produce rugged artifacts. Figure 6.2 illustrates artifacts produced when approximating a smooth function ($f(x) = 1$) using Wendland's RBFs. In this figure, the distance between the centers of two adjacent RBFs is 0.3 times the support extent of RBFs. This RBF distribution is denser than that in our implementation (~ 0.33), and still produces clearly visible artifacts. Figure 6.3 shows an example of the resultant warping in the form of displacement field. The "grains" of RBFs are clearly visible even around the center of the volume where the displacements should be small. The results of these "grainy" mapping function is the artifacts in the warped images. An example is shown in Figure 6.4. In this example, some tissues in the cerebellum are warped in a unreasonable manner. This type of artifact may reduce the validity of the registration result, and this phenomenon may also exist among other kinds of RBFs due to their similar characteristics. In our work, the solution to this problem was to use a proper weight for the prior term in order to limit to scale of this kind of artifacts. However, this solution is only a trade-off between validity and accuracy. Increasing the weight for the prior term also in some way compromises the accuracy of the registration result. In contrast, using other basis functions such as cubic spline can achieve the same level of accuracy with smoother velocity field. In other words, given the same weight of regularization, even higher accuracy may be achieved if cubic splines are used as the basis functions. However, basis functions such as cubic splines entails much more computation time, which is yet another problem to be solved.

6.4 Similarity Measure

This part focuses on the discussion of limitations the similarity measure used in our work. As mentioned, CR is a robust similarity measure suitable for multi-modal registration. However, CR has a limitation which can greatly affect the performance of the proposed algorithm. First, the calculation of CR needs a sufficient amount of sample points in order to generate valid estimations of variances. This is contradictory to the philosophy of the

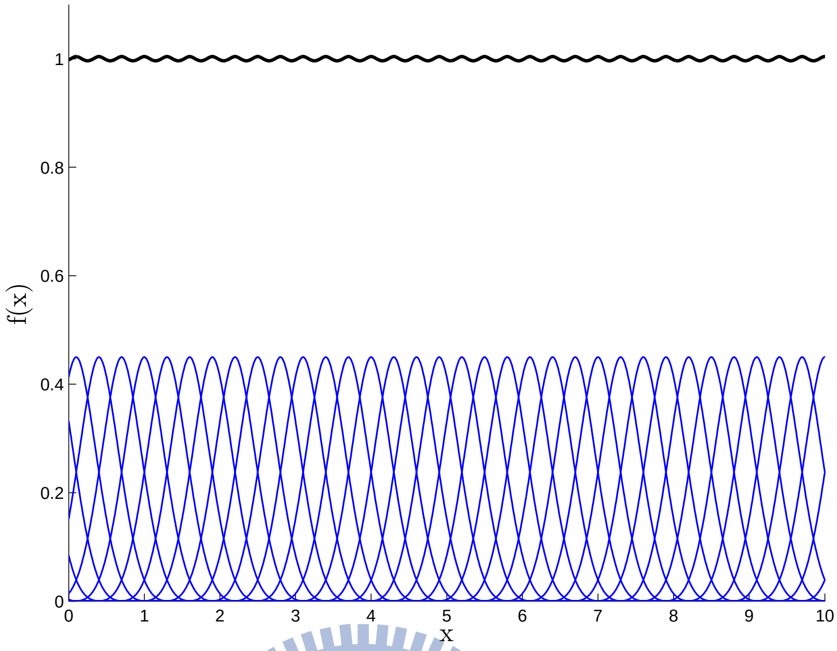

Figure 6.2: An example of artifacts produced by RBF approximation. The black wave-like line symbolizes the the approximation of $f(x) = 1$ using Wendland's RBFs (blue line) with support extents of 1 and spacings of 0.3 between each other.

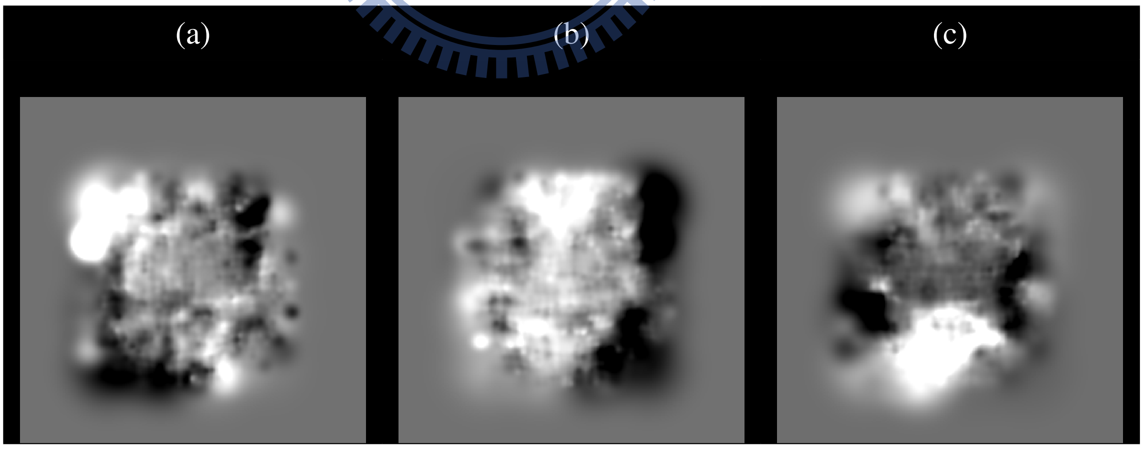

Figure 6.3: An example of resultant mapping function visualized in the form of displacement field. (a) displacements in X-direction. (b) displacements in Y-direction. (c) displacements in Z-direction.

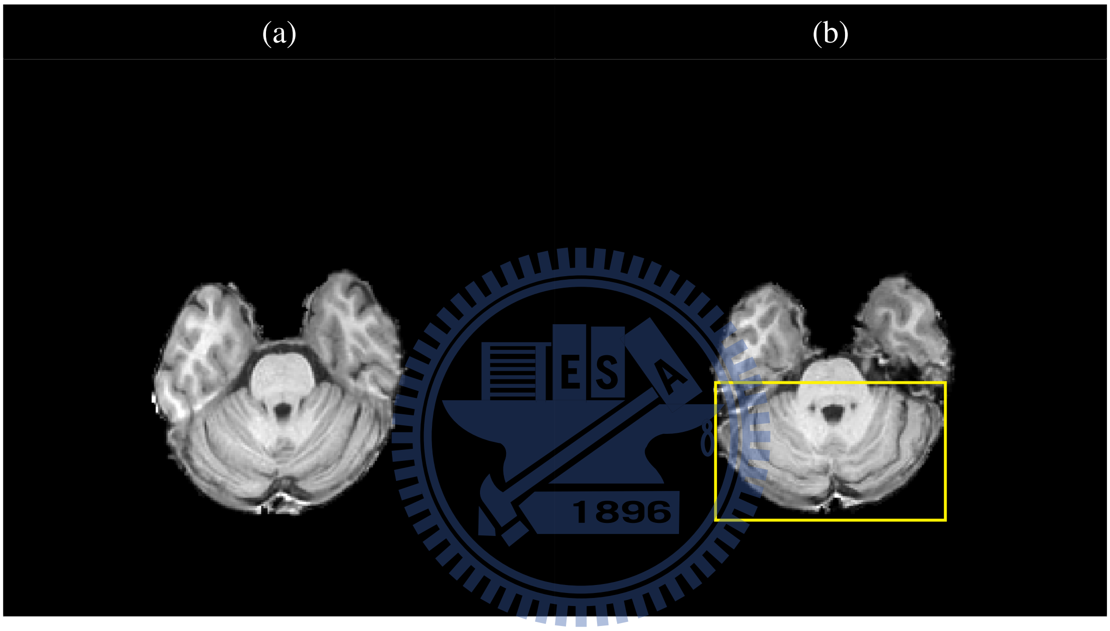

Figure 6.4: Artifacts in a registered image. Artifacts are clearly visible in some parts of the cerebellum (the yellow box). (a)The image before non-rigid registration. (b)The image after non-rigid registration.

proposed algorithm that the set of sample points should be as compact as possible in order to increase the speed. This forms a limitation when we want to use smaller down-sampled images to increase the speed. One possible solution to this problem is using other similarity measures such as cross-correlation. Cross-correlation (CC) is a similarity measure which is robust against the variability of brightness and contrast as well as inhomogeneity. It can be correctly computed using relatively few sample points when compared with CR. The only disadvantage of CC is it can only handle uni-modal registrations. But when an application only involves uni-modal registrations, CC may be a better similarity measure of choice since it imposes stronger constraints to the intensity relationship between source and target image.

The second limitation of CR is its flexibility in terms of handling multi-modal registrations. Although CR can handle most multi-modal registration schemes through its hypothesis of functional relationship of intensity, we must point out that there are still cases in which this assumption fails, that is, CR cannot handle all types of multi-modal registrations. For example, there is no functional intensity relationship between a T1-weighted MR image and a diffusion weighted image. In a T1-weighted MR image, the intensity of white matters is generally uniform throughout the image (ignoring the factors such as intensity non-uniformity). In a diffusion-weighted image, however, the intensity of white matters varies tremendously due to different directions of the fibers. In such cases when no functional relationship exists, similarity measures imposing more general assumptions are more preferable (e.g. mutual information).

Another limitation of the similarity measure used in our work is that the similarity is determined solely by the intensity of the images, and no anatomical knowledge is incorporated. One possible way to remove this limitation is by incorporating anatomical information in the similarity measure. For example, in the work by Hellier et al. [17], sulcal patterns are extracted and used as constraints for non-rigid brain registration. Liu et al. [23] proposed a hybrid registration method for cortical surfaces. It consists of a volumetric warping followed by an attribute-based surface warping. Incorporating anatomical information such as the cortical surfaces during the registrations may result in better alignment of tissues. This may be a promising direction for the improvement of the proposed algorithm.

Chapter 7

Conclusion

In this work, we have developed an efficient non-rigid registration algorithm which is symmetric and diffeomorphic. In the proposed algorithm, diffeomorphism is ensured by using the log-Euclidean framework. An symmetric correlation ratio combined with weighted Laplacian model is used as the objective function. To increase efficiency, we used a greedy local optimization approach based on radial basis functions. A hierarchical framework is used to further improve the speed and accuracy. The proposed algorithm can incorporate results of symmetric affine registrations without losing overall symmetry. The performance of the proposed algorithm was evaluated using the result of pairwise registration of LPBA40 dataset. The result of evaluation shows that the proposed algorithm is fully diffeomorphic and has sub-voxel accuracy in terms of symmetry. The proposed algorithm is faster than most diffeomorphic registration methods while maintaining high accuracy. According to the evaluation framework by Klein et al. [18], the proposed algorithm is more accurate than all 14 non-rigid registration algorithms listed by Klein et al. [18], and is faster than the top-ranked diffeomorphic registration algorithm. Although the result of the evaluation appears to be promising, more evaluations are still needed before any solid conclusion is made. Future works include further improvements in terms of accuracy and more evaluations based on different datasets and evaluation methods.

Bibliography

[1] Y. Amit. A nonlinear variational problem for image matching. *SIAM Journal on Scientific Computing*, 15(1):207–224, 1994.

[2] J.L.R. Andersson, M. Jenkinson, S. Smith, and J. Andersson. Non-linear optimisation. fMRIb technical report tr07ja1, 2007.

[3] Vincent Arsigny, Olivier Commowick, Xavier Pennec, and Nicholas Ayache. A log-euclidean framework for statistics on diffeomorphisms. In *Medical Image Computing and Computer-Assisted Intervention - Miccai 2006, Pt 1*, volume 4190, pages 924–931. 2006.

[4] J. Ashburner and K.J. Friston. Nonlinear spatial normalization using basis functions. *Human brain mapping*, 7(4):254–266, 1999.

[5] J. Ashburner and K.J. Friston. Spatial normalization using basis functions. *Human brain function*, pages 655–672, 2003.

[6] J. Ashburner and K.J. Friston. Diffeomorphic registration using geodesic shooting and gauss-newton optimisation. *NeuroImage*, 2011.

[7] John Ashburner. A fast diffeomorphic image registration algorithm. *NeuroImage*, 38(1):95–113, 2007.

[8] BB Avants, CL Epstein, M. Grossman, and JC Gee. Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain. *Medical image analysis*, 12(1):26–41, 2008.

[9] JC Baron, G. Chetelat, B. Desgranges, G. Perchey, B. Landeau, V. De La Sayette, and F. Eustache. In vivo mapping of gray matter loss with voxel-based morphometry in mild alzheimer's disease. *NeuroImage*, 14(2):298–309, 2001.

[10] M.F. Beg, M.I. Miller, A. Trouv, and L. Younes. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *International Journal of Computer Vision*, 61(2):139–157, 2005.

[11] M.C. Chiang, R.A. Dutton, K.M. Hayashi, AW Toga, OL Lopez, HJ Aizenstein, JT Becker, and PM Thompson. Fluid registration of medical images using jensen-renyi divergence reveals 3d profile of brain atrophy in hiv/aids. pages 193–196. IEEE, 2006. 3rd IEEE International Symposium on Biomedical Imaging: Nano to Macro.

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

[13] G.E. Christensen, R.D. Rabbitt, and M.I. Miller. 3d brain mapping using a deformable neuroanatomy. *Physics in Medicine and Biology*, 39:609, 1994.

[14] G.E. Christensen, R.D. Rabbitt, and M.I. Miller. Deformable templates using large deformation kinematics. *IEEE Transactions on Image Processing*, 5(10):1435–1447, 1996.

[15] A.C. Evans, D.L. Collins, SR Mills, ED Brown, RL Kelly, and TM Peters. 3d statistical neuroanatomical models from 305 MRI volumes. pages 1813–1817 vol. 3. IEEE, 1993. Nuclear Science Symposium and Medical Imaging Conference, 1993., 1993 IEEE Conference Record.

[16] A. Fornito, M. Ycel, J. Patti, SJ Wood, and C. Pantelis. Mapping grey matter reductions in schizophrenia: an anatomical likelihood estimation analysis of voxel-based morphometry studies. *Schizophrenia research*, 108(1-3):104–113, 2009.

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

[18] A. Klein, J. Andersson, B.A. Ardekani, J. Ashburner, B. Avants, M.C. Chiang, G.E. Christensen, D.L. Collins, J. Gee, and P. Hellier. Evaluation of 14 nonlinear deformation algorithms applied to human brain MRI registration. *NeuroImage*, 46(3):786–802, 2009.

[19] Y.H. Lau, M. Braun, and B.F. Hutton. Non-rigid image registration using a median-filtered coarse-to-fine displacement field and a symmetric correlation ratio. *Physics in Medicine and Biology*, 46:1297, 2001.

[20] Kuo-Wei Lee. *Construction of Customized Brain Template from Magnetic Resonance Images*. Master's thesis, 2011.

[21] JA Little, DLG Hill, and DJ Hawkes. Deformations incorporating rigid structures. pages 104–113. IEEE, 1996. Proceedings of the Workshop on Mathematical Methods in Biomedical Image Analysis.

[22] Jia-Xiu Liu, Yong-Sheng Chen, and Li-Fen Chen. Fast and accurate registration techniques for affine and nonrigid alignment of mr brain images. *Annals of Biomedical Engineering*, 38(1):138–157, 2010.

[23] T. Liu, D. Shen, and C. Davatzikos. Deformable registration of cortical structures via hybrid volumetric and surface warping. *NeuroImage*, 22(4):1790–1801, 2004.

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

[25] M. Merschhemke, TN Mitchell, SL Free, A. Hammers, L. Kinton, A. Siddiqui, J. Stevens, B. Kendall, HJ Meencke, and JS Duncan. Quantitative MRI detects abnormalities in relatives of patients with epilepsy and malformations of cortical development. *NeuroImage*, 18(3):642–649, 2003.

[26] W.H. Press and SA Teukolsky. Wtv: Numerical recipes in c: The art of scientific computing, 1992.

[27] Martin Reuter, H. Diana Rosas, and Bruce Fischl. Highly accurate inverse consistent registration: A robust approach. *NeuroImage*, 53(4):1181–1196, 2010.

[28] A. Roche, G. Malandain, N. Ayache, and S. Prima. Towards a better comprehension of similarity measures used in medical image registration. pages 555–566. Springer, 1999. Medical Image Computing and Computer-Assisted Intervention - ICCAI'99.

[29] A. Roche, G. Malandain, X. Pennec, and N. Ayache. The correlation ratio as a new similarity measure for multimodal image registration. *Medical Image Computing and Computer-Assisted Interventation - MICCAI'98*, pages 1115–1124, 1998.

[30] P. Rogelj and S. Kovai. Symmetric image registration. *Medical image analysis*, 10(3):484–493, 2006.

[31] G.K. Rohde, A. Aldroubi, and B.M. Dawant. The adaptive bases algorithm for intensity-based nonrigid image registration. *IEEE Transactions on Medical Imaging*, 22(11):1470–1479, 2003.

[32] D. Rueckert, L.I. Sonoda, C. Hayes, D.L.G. Hill, M.O. Leach, and D.J. Hawkes. Nonrigid registration using free-form deformations: application to breast mr images. *IEEE Transactions on Medical Imaging*, 18(8):712–721, 1999.

[33] D. Ruprecht, R. Nagel, and H. Mller. Spatial free-form deformation with scattered data interpolation methods. *Computers and graphics*, 19(1):63–71, 1995.

[34] D.W. Scott. On optimal and data-based histograms. *Biometrika*, 66(3):605–610, 1979.

[35] D.W. Shattuck, M. Mirza, V. Adisetiyo, C. Hojatkashani, G. Salamon, K.L. Narr, R.A. Poldrack, R.M. Bilder, and A.W. Toga. Construction of a 3d probabilistic atlas of human cortical structures. *NeuroImage*, 39(3):1064–1080, 2008.

[36] J.P. Thirion. Image matching as a diffusion process: an analogy with maxwell's demons. *Medical image analysis*, 2(3):243–260, 1998.

[37] W.K. Thompson and D. Holland. Bias in tensor based morphometry stat-roi measures may result in unrealistic power estimates. *NeuroImage*, 57(1):1, 2011.

[38] Tom Vercauteren, Xavier Pennec, Aymeric Perchant, and Nicholas Ayache. Diffeomorphic demons: Efficient non-parametric image registration. *NeuroImage*, 45(1):S61–S72, 2009.

[39] MP Wand. Data-based choice of histogram bin width. *American Statistician*, pages 59–64, 1997.

[40] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in computational Mathematics*, 4(1):389–396, 1995.

[41] P.A. Yushkevich, B.B. Avants, S.R. Das, J. Pluta, M. Altinay, and C. Craige. Bias in estimation of hippocampal atrophy using deformation-based morphometry arises from asymmetric global normalization: An illustration in adni 3 t MRI data. *NeuroImage*, 50(2):434–445, 2010.