

Diffeomorphic Registration of Diffusion Tensor Images with Exact Finite-Strain Differential

B.T. Thomas Yeo Tom Vercauteren Pierre Fillard
Xavier Pennec Polina Golland Nicholas Ayache Olivier Clatz

Abstract

In this paper, we propose an algorithm for the diffeomorphic non-linear registration of diffusion tensor images. Previous diffusion tensor registration algorithms using full tensor information suffer from difficulties in computing the differential of the Finite Strain tensor reorientation strategy and the gradient of the objective function. In contrast, we borrow results from the pose estimation literature in computer vision to derive an analytical gradient of the registration objective function. By leveraging on the closed-form gradient and the velocity field representation of one parameter subgroups of diffeomorphisms, the resulting registration algorithm is diffeomorphic and fast. Implemented under the Insight Toolkit (ITK) framework, registration of a pair of 128x128x60 diffusion tensor volumes takes 15 minutes, faster than many non-linear scalar image registration algorithms in the literature. We contrast the algorithm with a classic alternative that does not take into account the reorientation in the gradient computation and show that using the exact gradient achieves significantly better registration at the cost of being a few times slower. Registration is performed and evaluated on a set of 10 diffusion tensor brain images, using both Euclidean and Log-Euclidean interpolation and both Euclidean and Log-Euclidean Sum of Squares Difference similarity measure.

Index Terms

This research is funded by the INRIA “associated teams” program *CompuTumor*: <http://www-sop.inria.fr/asclepios/projects/boston/>. B.T. Thomas Yeo is funded by the Agency for Science, Technology and Research, Singapore. He did this work while at INRIA. DTI data courtesy of Denis Ducreux, MD, PhD, Bicêtre Hospital, Paris.

B.T. Thomas Yeo and Polina Golland are with the department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, USA. Tom Vercauteren is with Mauna Loa Technologies, Paris, France. Pierre Fillard, Xavier Pennec, Nicholas Ayache and Olivier Clatz are with the Asclepios Group, INRIA, Sophia Antipolis, France.

Contacts: ythomas@csail.mit.edu; tom.vercauteren@sophia.inria.fr; pierre.fillard@sophia.inria.fr;
xavier.pennec@sophia.inria.fr; polina@csail.mit.edu; nicholas.ayache@sophia.inria.fr; olivier.clatz@sophia.inria.fr

Diffusion Tensor Imaging, Registration, Diffeomorphisms, Finite Strain Differential

I. INTRODUCTION

Diffusion Tensor Imaging (DTI) non-invasively measures the diffusion of water in *in-vivo* biological tissues [6]. The diffusion is anisotropic in tissues such as cerebral white matter. DTI is therefore a powerful imaging modality for studying white matter structures in the brain. The rate and anisotropy of diffusion at each pixel of a diffusion tensor image is summarized by a rank 2 symmetric positive definite tensor. This is in contrast to scalar values in traditional magnetic resonance images. The eigenvectors of the tensor correspond to the three principal directions of diffusion while the eigenvalues measure the rate of diffusion in these directions.

To study the variability or similarity of white matter structures across a population or to track white matter changes of a single subject through time, registration is necessary to establish correspondences across different diffusion tensor (DT) images. Registration can be simplistically thought of as warping one image to match another. For scalar images, such a warp can for example be defined by a deformation field and an interpolation scheme. For DT images however, one also needs to define a tensor reorientation scheme. Reorientation of tensors is necessary to warp a tensor image consistent with the anatomy [2]. There are two commonly used reorientation strategies: the Finite Strain (FS) reorientation and the Preservation of Principal Directions (PPD) reorientation. Their empirical performances are very similar [15], [31].

Many DTI registration algorithms have been proposed [1], [11], [16], [20], [21], [24], [31], [32]. Because the reorientation strategies cause the computation of the gradient of the registration objective function to be non-trivial [31], many of these registration techniques use scalar values or features that are invariant to image transformations. This includes the use of fractional anisotropy [21] and fibers [32]. Leemans *et al.* [20] use mutual information to affinely align the diffusion weighted images from which the DT images are estimated from.

Alexander and Gee [1] perform elastic registration of tensor images by reorientating the tensors after each iteration using PPD reorientation. The reorientation is not taken into account in the objective function. Cao *et al.* [11] propose a diffeomorphic registration of tensor images using PPD reorientation. The diffeomorphism is parameterized by a non-stationary velocity field under the Large Deformation Diffeomorphic Metric Mapping (LDDMM) framework [7]. An exact gradient of the PPD orientation is computed by a clever analytical reformulation of the PPD reorientation algorithm.

For a general transformation, such as defined by B-splines or non-parametric free form displacement field, the FS reorientation [2] is defined by the rotation component of the deformation field. This rotation is given by the polar decomposition of the Jacobian of the deformation field using principles of continuum mechanics. Furthermore, the rotation produced by the polar decomposition of the Jacobian is the closest orthogonal operator to the Jacobian under any unitary invariant norm [18]. However, the polar decomposition requires computing the square root of a positive definite matrix, defined by the replacement of the eigenvalues of the original matrix with their square roots. The dependence of the rotation matrix on the Jacobian of deformation is therefore complicated and the gradient of any objective function incorporating the reorientation is hard to compute.

Zhang *et al.* [30], [31] propose and validate a novel piecewise local affine registration algorithm to register tensor images using FS reorientation. The tensor image is divided into uniform regions and optimal affine transformations are then found for each of these regions. This is possible because the transformation considered is affine. Therefore, the rotation component of the deformation needs *not* be computed. Instead, since rotation is already explicitly optimized in affine registration, the gradient due to FS reorientation can be easily computed. These piecewise affine transformations are fused together to generate a smooth warp field. The algorithm is iterated in a multiscale fashion with smaller uniform regions. However, it is unclear how much of the optimality is lost through the fusion of these optimal piecewise affine transformations.

In this paper, we borrow results from the pose estimation literature in computer vision to compute the analytical differential of the rotation matrix with respect to the Jacobian of the displacement field. We propose a diffeomorphic DTI registration algorithm, which extends the recently introduced diffeomorphic demons registration of scalar images [27] to tensor images. The availability of the exact analytical gradient allows us to utilize the Gauss-Newton method for optimization. The resulting DT registration is fast, taking about 15 minutes. This is comparable to the non-linear registration of scalar images whose runtime might range from a couple of minutes to hours.

The diffeomorphic demons registration algorithm is an extension of the popular demons algorithm [25]. It guarantees that the transformation is diffeomorphic. This diffeomorphism is parameterized by a composition of deformations, each of which is parametrized by a stationary velocity field. Such a representation is not unlike that used by the Large Deformable Diffeomorphic Metric Mapping (LDDMM) framework [7]. However, unlike LDDMM, the diffeomorphic demons algorithm does not seek a geodesic of the Lie group of diffeomorphism. One can therefore consider diffeomorphic demons to be part of the small deformation framework: at each iteration, the diffeomorphic demons algorithm seeks the best diffeomorphism to

be composed with the current transformation. The restriction of each diffeomorphism update to be a one parameter subgroup of diffeomorphism results in a much faster algorithm than the typical algorithms under the LDDMM framework or even algorithms that parameterize diffeomorphism by stationary velocity field [5], [17].

We also propose a simpler and faster algorithm that ignores the reorientation during the gradient computation. Reorientation is done after each iteration. This faster algorithm is therefore a diffeomorphic variant of the one proposed by Alexander and Gee [1] with Gauss-Newton optimization. We compare the two algorithms and show that using the exact gradient results in significantly better registration at the cost of computation time. In particular, we show that when using Log-Euclidean interpolation [3] and Log-Euclidean Sum of Squares Difference (LOG-SSD) similarity measure with a small deformation regularization, the resulting registration with the exact gradient not only has a lower Log-Euclidean Sum of Squares Difference (LOG-SSD) but also a lower Euclidean Sum of Squares Difference (EUC-SSD). This is true for the entire spectrum of deformation penalties. The same holds when using Euclidean interpolation and EUC-SSD in the registration objective function. The use of Euclidean interpolation and EUC-SSD in warping and comparing tensor images is quite common [1], [2], [31]. More recently, Log-Euclidean interpolation and metrics have also become popular [3], [14]. A second experiment shows that the exact gradient is also able to recover randomly generated deformation fields significantly better than when using the approximate gradient that ignores reorientation.

We emphasize that there is no theoretical guarantees that using the true gradient will lead to a better solution. After all, the registration problem is non-convex and any solution we find is a local optimum. In practice, the experiments show that taking reorientation into account does significantly improve the registration results. We believe that the reorientation of tensors provides an additional degree of freedom, in the sense that local rotation of tensors does not incur deformation penalty. Taking the reorientation into account therefore allows us to match two tensor images more easily. On the other hand, the reorientation also provides an additional constraint. The registration algorithm cannot arbitrarily pull in a far-away region for matching because this induces the reorientation of tensors in other regions (think of the famous ‘C’ example in large deformation fluid registration [12]). This additional constraint acts as a further regularization, leading to a better solution.

This paper is organized as follows. The next section describes the computation of the FS differential. We then give an overview of the diffeomorphic demons algorithm in section III. We extend the diffeomorphic demons to tensor images in section IV using the exact FS differential. We also propose a simpler and faster algorithm that ignores the reorientation during the gradient computation. In section V, we compare

the two algorithms with a set of 10 DT brain images.

To summarize, our contributions are:

- 1) Derivation of the exact Finite Strain (FS) differential.
- 2) Incorporation of the FS differential into a fast diffeomorphic DT image registration algorithm. We emphasize that the FS differential is useful, even if one were to use a different model of deformation or a different similarity metric.
- 3) Demonstration that the use of the exact gradient leads to better registration, in the sense that for any deformation energy, we obtain lower LOG-SSD and EUC-SSD, regardless of whether we use LOG-SSD or EUC-SSD in the objective function. We also show that the exact gradient is able to recover randomly generated deformation fields significantly better than when using an approximated gradient that ignores reorientation.
- 4) Our implementation allows for Euclidean interpolation and EUC-SSD metric, as well as, Log-Euclidean interpolation and LOG-SSD metric.

II. FINITE STRAIN DIFFERENTIAL

Deforming a tensor image by a transformation s involves tensor interpolation followed by tensor reorientation [2]. To compute a deformed tensor at a voxel n , one first interpolates the tensor to get the interpolated tensor $T(n)$. Interpolation can for example be done using Euclidean interpolation [2], Log-Euclidean interpolation [3], affine-invariant framework [22] or Geodesic-Loxodromes [19]. In this work, we will focus on Euclidean and Log-Euclidean interpolation since they are currently the most commonly used interpolation scheme for tensors and because they are computationally simple. The Finite Strain (FS) differential we compute in this section is related to tensor reorientation, and the discussion is therefore independent of the interpolation strategy.

More formally, the transformation s sends a point p to the point $s(p)$. Let $u \triangleq s - I$ be the displacement field associated with the transformation s . Then

$$s(p) = p + u \tag{1}$$

The ‘ $-$ ’ in $u \triangleq s - I$ is used in a loose sense and is meant to convey that u is the transformation s without the identity part of the transform I . Similarly, we denote $s = I + u$. Note that even for parametric representation of transformations, such as splines, one can always derive the equivalent displacement field representation.

According to the FS tensor reorientation strategy [2] for nonlinear deformation, one first computes the rotation component of the deformation at the n -th pixel:

$$R(n) = (J(n)J(n)^T)^{-\frac{1}{2}}J(n) \quad (2)$$

where $J(n)$ is the Jacobian of the spatial transform s at the n -th pixel:

$$J(n) = \begin{pmatrix} \frac{\partial s_x(n)}{\partial x} & \frac{\partial s_x(n)}{\partial y} & \frac{\partial s_x(n)}{\partial z} \\ \frac{\partial s_y(n)}{\partial x} & \frac{\partial s_y(n)}{\partial y} & \frac{\partial s_y(n)}{\partial z} \\ \frac{\partial s_z(n)}{\partial x} & \frac{\partial s_z(n)}{\partial y} & \frac{\partial s_z(n)}{\partial z} \end{pmatrix} = I + \begin{pmatrix} \frac{\partial u_x(n)}{\partial x} & \frac{\partial u_x(n)}{\partial y} & \frac{\partial u_x(n)}{\partial z} \\ \frac{\partial u_y(n)}{\partial x} & \frac{\partial u_y(n)}{\partial y} & \frac{\partial u_y(n)}{\partial z} \\ \frac{\partial u_z(n)}{\partial x} & \frac{\partial u_z(n)}{\partial y} & \frac{\partial u_z(n)}{\partial z} \end{pmatrix} \quad (3)$$

u_x, u_y, u_z are the displacement field in the x, y and z directions. $R(n)$ is known as the polar decomposition of the matrix $J(n)$ and is therefore the function of the displacement field u in the neighborhood of n . Notice that under identity transformation, i.e., zero displacements, $J(n) = I$ and $R(n) = I$.

The interpolated tensor $T(n)$ is then reoriented, resulting in the final tensor $T'(n)$:

$$T'(n) = R^T(n)T(n)R(n) \quad (4)$$

For registration using the FS strategy, it is therefore necessary to compute the differential of rotation R with respect to the transformation s , and thus by chain rule, with respect to Jacobian J . Denoting $S = (JJ^T)^{\frac{1}{2}}$ and using the results of pose estimation literature [13], we get the 3x3 matrix (see Appendix A):

$$dR = -R \left[R^T (\text{tr}(S)I - S)^{-1} R \sum_i (R^T)_i \otimes (dJ^T)_i \right]^{\oplus} \quad (5)$$

where \otimes denotes the 3D cross product, $(\cdot)_i$ denotes the i -th column of (\cdot) and \oplus is the operator defined as

$$m^{\oplus} = ([m_1, m_2, m_3]^T)^{\oplus} \triangleq \begin{pmatrix} 0 & -m_3 & m_2 \\ m_3 & 0 & -m_1 \\ -m_2 & m_1 & 0 \end{pmatrix} \quad (6)$$

Let J_{ij} be the ij -th component of J . Eq. (5) tells us the variation of rotation R in terms of the components of Jacobian J . In other words, $\frac{\partial R}{\partial J_{ij}}$ is simply dR from Eq. (5) by setting the matrix dJ to 0, except for $(dJ)_{ij}$ which is set to 1.

III. BACKGROUND ON DIFFEOMORPHISM

In this section, we discuss the diffeomorphic extension [27] of Thirion's demons algorithm [25]. We will also discuss numerical issues related to representing diffeomorphism by velocity fields and optimization methods we will be using in this paper.

A. Diffeomorphic Demons

We consider the modified demons objective function [10] for registering a moving scalar image M to a fixed scalar image F :

$$E(c, s) = \|\Sigma^{-1}(F - M \circ c)\|^2 + \frac{1}{\sigma_x^2} \text{dist}(s, c) + \frac{1}{\sigma_T} \text{Reg}(s) \quad (7)$$

where s is the non-parametric dense spatial transformation to be optimized, c is an auxiliary vector field and $\|\cdot\|$ denotes the L_2 -norm of a vector (or vector field depending on the context). We can think of the fixed image F and warped moving image $M \circ c$ as a one dimensional vector of length N voxels. Σ is a $N \times N$ diagonal matrix that defines how much variability one observes at a particular voxel. This allows a fast and simple optimization procedure by alternately optimizing the first two terms and the last two terms of Eq.(7). Typically, $\text{dist}(c, s) = \|c - s\|^2$ and $\text{Reg}(s) = \|\nabla s\|^2$. However the regularization can be modified to handle fluid-like constraints.

For the demons algorithm and its variants, the objective function is optimized over the complete space of non-parametric spatial transformations [10], [23], [25], [28]. This non-parametric spatial transformation is usually represented by a displacement field. Unfortunately, the resulting deformation might not be diffeomorphic. Instead, Vercauteren *et al.* [27] optimize over a composition of deformations, each of which is parametrized by a stationary velocity field. At each iteration, the diffeomorphic demons algorithm seeks the best diffeomorphism (represented by the stationary velocity v) to be composed with the current transformation.

In this case, v is an element of the Lie algebra g and $\exp(v)$ is the diffeomorphism associated with the velocity field v . The operator $\exp(\cdot)$ is the group exponential relating the Lie Group G to its associated Lie algebra g . More formally, let $\Phi_{tv}(x_0)$ be the solution at time t to the following Ordinary Differential Equation (ODE) with stationary velocity field:

$$\frac{dx}{dt} = v(x) \quad \text{with initial condition} \quad x(0) = x_0 \quad (8)$$

We define

$$\exp(v)(x) \triangleq \Phi_v(x) \triangleq w(x) \quad (9)$$

An image $M \circ \exp(v)$ is therefore a deformed version of image M by transforming the coordinate system of M by $\exp(v)$: the point x in the deformed coordinate system corresponds to the point $\Phi_v(x)$ in the old coordinate system.

We can summarize the diffeomorphic demons algorithm [26], [27] as follows:

- 1) Choose a starting spatial transformation $s^{(0)}$ (represented by a displacement field)
- 2) Iterate until convergence:
 - (i) Given $s^{(i)}$, compute a stationary velocity field update $v^{(i+1)}$ by minimizing the first two terms of Eq. (7):

$$v^{(i+1)} = \underset{v}{\operatorname{argmin}} \left\| \Sigma^{-1}(F - M \circ s^{(i)} \circ \exp(v)) \right\|^2 + \frac{1}{\sigma_x^2} \operatorname{dist}(s^{(i)}, s^{(i)} \circ \exp(v)) \quad (10)$$

where v is an element of the Lie algebra g associated with the Lie group.

- (ii) If a fluid like regularization is used, let $v^{(i+1)} \leftarrow K_{\text{fluid}} \star v^{(i+1)}$. The convolution kernel will typically be Gaussian.
- (iii) Let $c^{(i+1)} \leftarrow s^{(i)} \circ \exp(v^{(i+1)})$
- (iv) If a diffusion-like regularization is used, let $s^{(i+1)} \leftarrow I + K_{\text{diff}} \star (c^{(i+1)} - I)$ (else let $s^{(i+1)} \leftarrow c^{(i+1)}$). The convolution kernel will also typically be Gaussian. Once again, the ‘-’ and ‘+’ are used in a loose sense.

Steps 2(ii) to 2(iv) essentially optimize the last two terms of Eq. (7). For detailed discussion of using convolution kernels to achieve elastic and fluid regularization, see [9], [10].

B. Numerical Issues in Velocity Field Representations

While v and $\Phi_v(x) = \exp(v)(x)$ are technically defined on the entire continuous image domain, in practice, v and u are represented by vector fields defined on discrete points of the image, such as at each pixel [25], [26], [27] or control points [5], [7]. From the theories of ODEs [8], we know that the integral curves $u = \exp(v)$ (or trajectories) of a velocity field $v(x, t)$ exist and are unique if $v(x, t)$ is Lipschitz continuous in x and continuous in t . Uniqueness means that the trajectories do not cross, implying that the deformation is invertible. Furthermore, we know from the theories of ODEs that a C^r continuous velocity field v produces a C^r continuous deformation field $\Phi_{tv}(x)$. Therefore, a sufficiently smooth velocity field results in a diffeomorphic transformation.

Since the velocity field v is stationary in the case of the one parameter subgroup of diffeomorphism, v is clearly continuous (and in fact C^∞) in t . A smooth interpolation of v is continuous in the spatial domain and is Lipschitz continuous if we consider a compact domain (which holds since we only consider images that are closed and bounded).

To compute the final deformation of an image, we have to estimate $\exp(v)$ at least at the set of image grid points. For example, we can compute $\exp(v)$ by numerically integrating the smoothly interpolated velocity field v with Euler integration, such as that in [7]. In this case, the estimate becomes arbitrarily

close to the true $\exp(v)$ as the number of integration time steps increases. With a sufficiently large number of integration steps, we expect the estimate to be invertible and the resulting transformation to be diffeomorphic.

The parameterization of diffeomorphism by stationary velocity field is made popular by the use of the fast “scaling and squaring” approach [4] in computing $\exp(v)$. Instead of Euler integration, the “scaling and squaring” approach works by multiple composition of displacement fields:

$$\begin{aligned}
\Phi_{\frac{1}{2^N}v}(x) &= x + \frac{1}{2^N}v(x) \\
\Phi_{\frac{1}{2^{N-1}}v}(x) &= \Phi_{\frac{1}{2^N}v}(x) \circ \Phi_{\frac{1}{2^N}v}(x) \\
&\vdots \\
\Phi_v(x) &= \Phi_{\frac{1}{2}v}(x) \circ \Phi_{\frac{1}{2}v}(x)
\end{aligned} \tag{11}$$

While this method is correct in the continuous case, in the discrete case, composition of the displacement field requires interpolation of displacement fields, introducing errors in the process. In particular, suppose $\Phi_{t_0v}(x)$ and $\Phi_{2t_0v}(x)$ are the true trajectories found by performing an accurate Euler integration up to time t_0 and $2t_0$ respectively. Then, there does not exist a trivial interpolation scheme, so that $\Phi_{2t_0v}(x) = \Phi_{t_0v}(x) \circ \Phi_{t_0v}(x)$. In practice however, it is widely reported that “scaling and squaring” tends to preserve invertibility even with rather large deformation [5], [27]. In this work, we will use trilinear interpolation because it is fast. We find that in practice, the transformation is indeed diffeomorphic. Technically speaking, since we use linear interpolation for the displacement field, the transformation is only homeomorphic rather than diffeomorphic. However, we will follow the convention of [4], [5], [27] who call their transformation diffeomorphic even though they are technically homeomorphic.

C. Gauss-Newton Nonlinear Least-Squares Optimization

We now focus on the optimization of step 2(i) of the diffeomorphic demons algorithm. We first note that the objective function in step 2(i) can be written in a non-linear least-squares form:

$$E_s(v) = \left\| \begin{bmatrix} \Sigma^{-1}(F - M \circ s \circ \exp(v)) \\ \frac{1}{\sigma_x} \exp(v) \end{bmatrix} \right\|^2 \tag{12}$$

$$= \left\| \begin{bmatrix} \varphi^1(s \circ \exp(v)) \\ \varphi^2(s \circ \exp(v)) \end{bmatrix} \right\|^2 \tag{13}$$

$$= \|\varphi(s \circ \exp(v))\|^2 \tag{14}$$

where we have made the choice that $\text{dist}(s, s \circ \exp(v)) = \|s^{-1} \circ s \circ \exp(v)\|^2 = \|\exp(v)\|^2 = \|u\|^2$, where $u = \exp(v) - I$. Here, we are “subtracting” out the identity transformation so that identity transformation corresponds to no penalty. One can rewrite the above as follows:

$$E_s(v) = \left\| \varphi(s) + \begin{bmatrix} D_s^{\varphi^1}(0) \\ D_s^{\varphi^2}(0) \end{bmatrix} v + O(\|v\|^2) \right\|^2 \quad (15)$$

To interpret the above equation, for 3-dim images with N voxels, let v be a $3N \times 1$ vector: $\{v_x(1), v_y(1), v_z(1), \dots, v_x(N), v_y(N), v_z(N)\}$. Then, $D_s^{\varphi^1}(0)$ is a $N \times 3N$ block diagonal matrix, where the n -th block corresponds to a 1×3 matrix $[D_s^{\varphi^1}(0)]_n = \frac{\partial \varphi_n^1(s \circ \exp(v))}{\partial v(n)} \Big|_{v(n)=0}$ where $\varphi_n^1(s \circ \exp(v)) = \Sigma^{-1}(n)(F(n) - M \circ s \circ \exp(v)(n))$ is the n -th component of φ^1 . Using the chain rule, we get:

$$[D_s^{\varphi^1}(0)]_n = \frac{\partial \varphi_n^1(s \circ \exp(v))}{\partial \exp(v)(n)} \cdot \frac{\partial \exp(v)(n)}{\partial v(n)} \Big|_{v(n)=0} \quad (16)$$

$$= \frac{\partial \varphi_n^1(s \circ \exp(v))}{\partial \exp(v)(n)} \Big|_{v(n)=0} \quad (17)$$

$$= \frac{\partial \varphi_n^1(s \circ w)}{\partial w(n)} \Big|_{w(n)=n} \quad (18)$$

$$= -\Sigma^{-1}(n) \nabla(M \circ s)(n) \quad (19)$$

where $w(n) = \exp(v)(n) = \Phi_v(n)$ is the transformation of voxel n . In Eq. 17, we made use of the fact that the differential of the exponential map is the identity. $\nabla(M \circ s)(n)$ is the spatial derivative of the image intensity at voxel n of the warped moving image $M \circ s$: $\{\frac{\partial M_n}{\partial x}, \frac{\partial M_n}{\partial y}, \frac{\partial M_n}{\partial z}\}$. Therefore, we denote $-\Sigma^{-1}(\nabla(M \circ s))$ to be $D_s^{\varphi^1}(0)$.

Similarly, $D_s^{\varphi^2}(0)$ is equal to $\frac{1}{\sigma_x} I$ where I is a $3N \times 3N$ identity matrix. By ignoring the $O(\|v\|^2)$ term within the norm in Eq. (15), we end up with the classic linear least squares problem, which can be solved via the normal equations. This is the Gauss-Newton optimization method. In particular, Eq. (12) can then be re-written as:

$$E_s(v) \approx \left\| \begin{bmatrix} \Sigma^{-1}(F - M \circ s) \\ 0 \end{bmatrix} + \begin{bmatrix} -\Sigma^{-1}(\nabla(M \circ s)) \\ \frac{1}{\sigma_x} I \end{bmatrix} v \right\|^2 \quad (20)$$

$$= \|b - Av\|^2 \quad (21)$$

which is a linear least squares problem. Despite the size of the matrices, it is easy to solve the resulting linear system $Av = b$ since we can consider each pixel separately. In fact, with the help of the Sherman-Morrison matrix inversion lemma, no matrix inversion is even needed to invert the resulting block diagonal $3N \times 3N$ matrix $A^T A$ [26].

We note that in the original demons algorithm [25], $\nabla(M \circ s)$ is replaced by ∇F . This can be justified by the fact that at the optimum, the gradient of the warped moving image is the same as the fixed image.

IV. TENSOR IMAGE REGISTRATION

A. Diffeomorphic Demons for Vector Images

We define a vector image to be an image with a vector of intensities at each voxel. We can treat a vector image like a scalar image in the sense that each vector component is independent of the other components. Deformation of a vector image works just like a scalar image, by treating each component of the vector separately. We extend the diffeomorphic demons registration of scalar images to vector images. The discussion here will be used for computing update steps when ignoring tensor reorientation in section IV-D.

It is fairly straightforward to re-derive the results from the previous section for vector images. Let K be the dimension of the intensity vector at each voxel. For convenience, we define F_n to be the $K \times 1$ intensity vector $\{F_n(1), \dots, F_n(K)\}$ of the n -th voxel and F to be the $NK \times 1$ vector $F = \{F_1, \dots, F_N\}$. Then the diffeomorphic demons algorithm from the previous section applies exactly to vector images except $\nabla(M \circ s)$ is now a sparse $NK \times 3N$ block diagonal matrix, where each block is $K \times 3$. In particular, the n -th block corresponds to:

$$\begin{pmatrix} \frac{\partial(M \circ s)_n(1)}{\partial x} & \frac{\partial(M \circ s)_n(1)}{\partial y} & \frac{\partial(M \circ s)_n(1)}{\partial z} \\ \vdots & \vdots & \vdots \\ \frac{\partial(M \circ s)_n(K)}{\partial x} & \frac{\partial(M \circ s)_n(K)}{\partial y} & \frac{\partial(M \circ s)_n(K)}{\partial z} \end{pmatrix} \quad (22)$$

The resulting least squares linear system $Av = b$ is slightly harder to solve than before. For each pixel n , we have to solve a 3×3 linear system for speed vector update $v(n)$.

B. Diffeomorphic Demons for Tensor Images

A DT image is different from a vector image because of the additional structure present in a tensor. In particular, the space of symmetric positive definite matrices (tensor) is not a vector space. When deforming a DT image, reorientation is also necessary. We extend the diffeomorphic demons registration of scalar images to tensor images.

In this work, we use the modified demons objective function (Eq. (7)) and the Finite Strain (FS)

reorientation strategy in our registration. Then Eq. (12) becomes

$$E_s(v) = \left\| \begin{bmatrix} \Sigma^{-1} [F - R^T (M \circ s \circ \exp(v)) R] \\ \frac{1}{\sigma_x} \exp(v) \end{bmatrix} \right\|^2 \quad (23)$$

$$= \left\| \begin{bmatrix} \varphi^1(s \circ \exp(v)) \\ \varphi^2(s \circ \exp(v)) \end{bmatrix} \right\|^2 \quad (24)$$

$$= \|\varphi(s \circ \exp(v))\|^2 \quad (25)$$

Here, $[F - R^T (M \circ s \circ \exp(v)) R]$ is the Euclidean Sum of Squares Difference (EUC-SSD) between the tensor images. In particular, F can be seen as a $9N \times 1$ vector by “rasterizing” the 3×3 rank 2 tensor at each voxel into a column vector. $M \circ s \circ \exp(v)$ should be interpreted as the interpolated tensor image. In practice, since the tensor is symmetric, we can get away with using a $6N \times 1$ vector and by adjusting the weights of some of the entries of Σ^{-1} by $\sqrt{2}$. Each interpolated tensor is then reoriented using the rotation matrix R of each pixel and “rasterized” into a column vector. Note that R is implicitly dependent on the transformation $s \circ \exp(v)$. $[F - R^T (M \circ s \circ \exp(v)) R]$ then computes the SSD between each tensor of the fixed image and the corresponding reoriented and interpolated tensor in the warped moving image (by treating each tensor as a vector) and summing the SSD for all voxels.

Eq. (23) can also be interpreted as the LOG-SSD between tensors if F and M are the Log-Euclidean transforms of the original tensor images. This is done by converting each tensor T in the original image to a log-tensor $\log(T)$. Note that $\log(T)$ is simply a symmetric matrix [3]. $M \circ s \circ \exp(v)$ is then the interpolated log-tensor image and $R^T (M \circ s \circ \exp(v)) R$ is the interpolated and reoriented log-tensor image. This works because $\log(R^T T R) = R^T \log(T) R$ for any rotation matrix R . Therefore, reorienting a tensor followed by Log-Euclidean transformation is the same as Log-Euclidean transforming a tensor followed by reorientation. This is convenient since we can perform a one time Log-Euclidean transformation of the tensor images into log-tensor images before registration and then treat the resulting log-tensors as tensors.

The differential of φ^2 denoted as $D_s^{\varphi^2}(0)$ is the $3N \times 3N$ matrix $\frac{1}{\sigma_x} I$. On the other hand, $D_s^{\varphi^1}(0)$ is a sparse $9N \times 3N$ matrix. One can interpret $D_s^{\varphi^1}(0)$ as $N \times N$ blocks of 9×3 matrices. In particular, the (n, j) -th block $[D_s^{\varphi^1}(0)]_{nj}$ is equal to $\frac{\partial \varphi_n^1(s \circ w)}{\partial v(j)}(v(j) = 0)$, where we remind the readers that $w = \exp(v) = \Phi_v$.

Using the chain rule, the fact that the differential of the exponential map is the identity and the product

rule, we get

$$[D_s^{\varphi^1}(0)]_{nj} = \left. \frac{\partial \varphi_n^1(s \circ w)}{\partial v(j)} \right|_{v(j)=0} \quad (26)$$

$$= \sum_{k=1}^N \left. \frac{\partial \varphi_n^1(s \circ w)}{\partial w(k)} \frac{\partial w(k)}{\partial v(j)} \right|_{v(j)=0} \quad (27)$$

$$= \left. \frac{\partial \varphi_n^1(s \circ w)}{\partial w(j)} \right|_{w(j)=j} \quad (28)$$

$$= -\Sigma^{-1}(n) \left[\frac{\partial R^T(n)}{\partial w(j)} (M \circ s \circ w(n)) R(n) \right. \\ \left. + R^T(n) \frac{\partial (M \circ s \circ w(n))}{\partial w(j)} R(n) \right. \\ \left. + R^T(n) (M \circ s \circ w(n)) \frac{\partial R(n)}{\partial w(j)} \right] \Bigg|_{w(j)=j} \quad (29)$$

Recall that $R(n)$ is a function of the Jacobian of displacement field $J(n)$ at the n -th pixel and that Eq. (3) gives an analytical expression of $J(n)$. In practice, $J(n)$ is defined numerically using finite central difference as

$$J(n) = \begin{pmatrix} \frac{s \circ w_x(n_{x+}) - s \circ w_x(n_{x-})}{2\Delta x} & \frac{s \circ w_x(n_{y+}) - s \circ w_x(n_{y-})}{2\Delta y} & \frac{s \circ w_x(n_{z+}) - s \circ w_x(n_{z-})}{2\Delta z} \\ \frac{s \circ w_y(n_{x+}) - s \circ w_y(n_{x-})}{2\Delta x} & \frac{s \circ w_y(n_{y+}) - s \circ w_y(n_{y-})}{2\Delta y} & \frac{s \circ w_y(n_{z+}) - s \circ w_y(n_{z-})}{2\Delta z} \\ \frac{s \circ w_z(n_{x+}) - s \circ w_z(n_{x-})}{2\Delta x} & \frac{s \circ w_z(n_{y+}) - s \circ w_z(n_{y-})}{2\Delta y} & \frac{s \circ w_z(n_{z+}) - s \circ w_z(n_{z-})}{2\Delta z} \end{pmatrix} \quad (30)$$

where $\{n_{x-}, n_{x+}, n_{y-}, n_{y+}, n_{z-}, n_{z+}\}$ are the neighbors of voxel n in the x , y and z directions respectively. Therefore $w_y(n_{x+})$ denotes the y -coordinate of n_{x+} after transformation $w(n_{x+})$ and $w_y(n_{x-})$ denotes the y -coordinate of n_{x-} after transformation $w(n_{x-})$. Δx , Δy and Δz are the voxel spacings in the x , y and z directions respectively.

Therefore,

$$[D_s^{\varphi^1}(0)]_{nn} = -\Sigma^{-1}(n) R^T(n) \nabla (M \circ s)(n) R(n) \quad (31)$$

and for neighbors j of voxel n , we get

$$[D_s^{\varphi^1}(0)]_{nj} = -\Sigma^{-1}(n) \left[\frac{\partial R^T(n)}{\partial w(j)} (M \circ s)(n) R(n) + R^T(n) (M \circ s)(n) \frac{\partial R(n)}{\partial w(j)} \right] \Bigg|_{w(j)=j} \quad (32)$$

Note that the first and second terms in the expression above are transpose of each other. Using the differential of R (Eq. (5)) and the expression of J (Eq. (30)), we can compute $\frac{\partial R(n)}{\partial w(j)}$ using the chain rule. For completeness, $\frac{\partial R(n)}{\partial w(j)}$ are derived in Appendix B.

In summary, we have computed the full gradient of our objective function:

$$D_s^{\varphi}(0) = \begin{bmatrix} D_s^{\varphi^1}(0) \\ \frac{1}{\sigma_x^2} I \end{bmatrix} \quad (33)$$

where $\frac{1}{\sigma_x^2}I$ is a $3N \times 3N$ identity matrix, while $D_s^{\varphi^1}(0)$ is a sparse $9N \times 3N$ matrix.

C. Optimization: Gauss-Newton Method

From the previous sections, we can now write

$$E_s(v) \approx \left\| \begin{bmatrix} \Sigma^{-1}(F - R^T(M \circ s)R) \\ 0 \end{bmatrix} + \begin{bmatrix} D_s^{\varphi^1}(0) \\ \frac{1}{\sigma_x}I \end{bmatrix} v \right\|^2 \quad (34)$$

$$= \|b - Av\|^2 \quad (35)$$

The resulting least-squares problem is harder to solve than before, since the linear systems of equations cannot be separated into a per voxel basis. In practice, we solve the linear systems of equations using the free Gmm++, a generic C++ template library for sparse matrices. At the finest resolution, solving the sparse linear system requires about 60 seconds. This is the bottleneck of the algorithm. However, due to the fast convergence of Gauss-Newton method, we typically only need to solve the linear systems 10 times per multi-resolution level. The resulting registration takes about 15 minutes.

We note that the efficiency of the Demons algorithm comes from the division of the optimization into 2 phases: optimization of the similarity measure and optimization of the regularization term. This avoids the need to solve a non-separable system of linear equations when considering the 2 phases together. Because of the reorientation in tensor registration, we have to solve a sparse system of linear equations anyway. In this case, we can incorporate the optimization of the regularization term together with the optimization of the similarity measure without loss of efficiency. In this work, we keep the two phases separate to allow for fair comparison with the case of ignoring the reorientation of tensors in the gradient computation (see section IV-D) by using almost the same implementation. Any improvement must then clearly come from the use of the true gradient and not from using a one-phase optimization scheme versus a two-phase optimization scheme.

D. Classical Alternative: Ignoring the Reorientation of Tensors

Previous work [1] performs tensor registration by not including the reorientation in the gradient computation, but reorienting the tensors after each iteration using the current estimated displacement field. To illustrate the utility of the true gradient, we adapt our algorithm to ignore the reorientation part of the objective function in the gradient computation. In particular, we can emulate the Gauss-Newton optimization from previously by setting $[D_s^{\varphi^1}(0)]_{nj} = 0$ and $[D_s^{\varphi^1}(0)]_{nn} = -\Sigma^{-1}\nabla(R^T(n)(M \circ s)R(n))$, effectively ignoring the effects of the displacement field of a voxel on the reorientation of its neighbors.

Note that $[D_s^{\varphi^1}(0)]_{nn}$ is slightly different from before. At each iteration, we therefore treat the tensor like a vector, except when deforming the moving image. The resulting least-squares problem degenerates to that of section IV-A. The algorithm is thus much faster since we only need to invert a 3x3 matrix per voxel at each iteration. Registration only takes a few minutes.

V. EXPERIMENTS AND DISCUSSIONS

In our experiments, we use 10 DT images (128x128x60, 25 gradient directions). These images are kindly contributed by Denis Ducreux, M.D., Ph.D., Bicêtre Hospital, Paris, France. A foreground mask for each of the 10 DT images is computed. When computing the SSD objective function, only voxels which are considered foreground in both the fixed image and warped moving image are included. As a result, the final SSD might depend on the number of overlapped foreground voxels between the fixed image and warped moving image. To correctly compare the results of two different algorithms, we consider the average SSD, which we call the Mean Square Error (MSE).

In this paper, we perform pairwise registration between pairs of the 10 DT images. via a standard multiresolution optimization, by smoothing and downsampling the data for initial registration and using the resulting registration from a coarser resolution to initialize the registration of a finer resolution. We find that the algorithms are able to recover global deformations.

A. Qualitative Evaluation

We first qualitatively compare the registration results of the exact FS gradient and the approximated gradient. Figure 1 shows the registration of a pair of subjects. Figure 1(a) and figure 1(b) show the fixed and moving images respectively. Figure 1(c) and figure 1(d) show the output of the registration algorithm when using the exact FS gradient and approximated gradient respectively. Log-Euclidean interpolation and LOG-SSD are used in the objective function. Figure 1(e) shows the LOG-SSD (computed pixelwise) attained by the exact FS gradient minus LOG-SSD (computed pixelwise) attained by the approximated gradient. Black region implies exact FS gradient is performing better in the region. White region implies approximated gradient is performing better in the region. 59% of the foreground pixels attain better LOG-SSD using the exact FS gradient than using the approximate gradient, even though the amount of deformation as reflected in the harmonic energy is the same.

We note that the percentage of improved voxels is an underestimation because in general the exact FS gradient results in more foreground overlap between the fixed and moving images. In this particular pair

of images, the distribution of improved voxels appears to be diffuse across the brain. This is in contrast to the pair of subjects in figure 2.

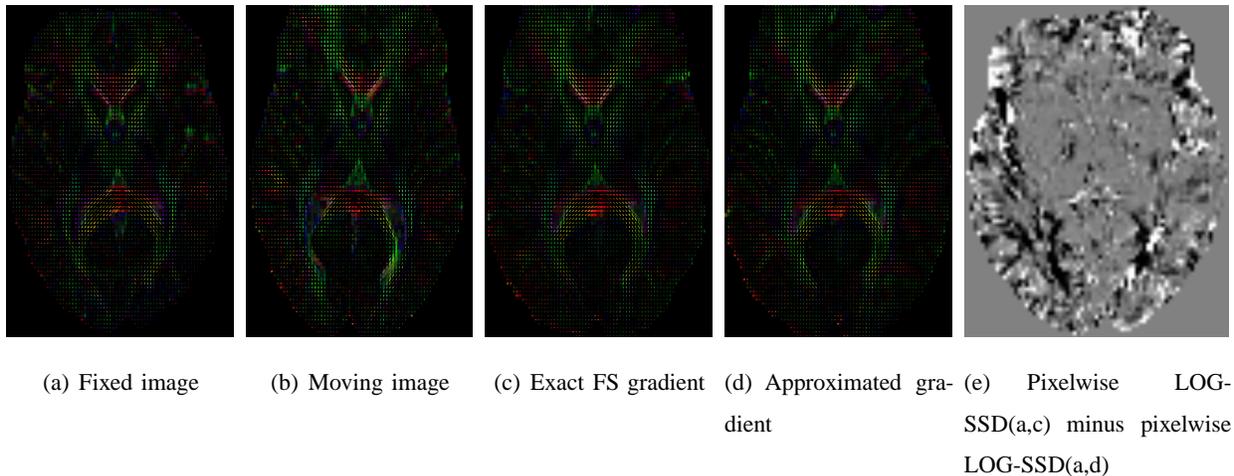


Fig. 1. Qualitative comparison between the use of exact FS gradient and approximated gradient for registering a pair of subjects. Log-Euclidean interpolation and LOG-SSD are used in the objective function. (a) Fixed Image (b) Moving Image (c) Registration result from using exact FS gradient. LOG-SSD = 0.39. Harmonic Energy = 0.16 (d) Registration result from using approximated gradient. LOG-SSD = 0.45. Harmonic Energy = 0.16 (e) Pixelwise LOG-SSD between image (a) and (c) minus pixelwise LOG-SSD between image (a) and (d). Black region implies exact FS gradient is performing better in the region. White region implies approximated gradient is performing better in the region. 59% of the foreground pixels attain better LOG-SSD using the exact FS gradient than using the approximate gradient, even though the amount of deformation as reflected in the harmonic energy is the same. The distribution of improved voxels appears to be diffuse across the brain. This is in contrast to the pair of subjects in figure 2.

Figure 2 shows the registration of a second pair of subjects. Once again, Log-Euclidean interpolation and LOG-SSD are used in the objective function. 56% of the foreground pixels attain better LOG-SSD using the exact FS gradient than using the approximate gradient even though there is more deformation when using the approximated gradient, as reflected in the harmonic energy. As shown in figure 2(e-i), the corpus callosum appears to be significantly better aligned when the exact FS gradient is used.

B. Quantitative Evaluation I

To more quantitatively compare the performance of the exact FS gradient, the approximated gradient and the fixed image gradient, we will consider pairwise registration of the 10 DT images. Since our registration is not symmetric between the fixed and moving images, there are 90 possible pairwise registration. We randomly select 40 pairs of images for pairwise registration. From our experiments,

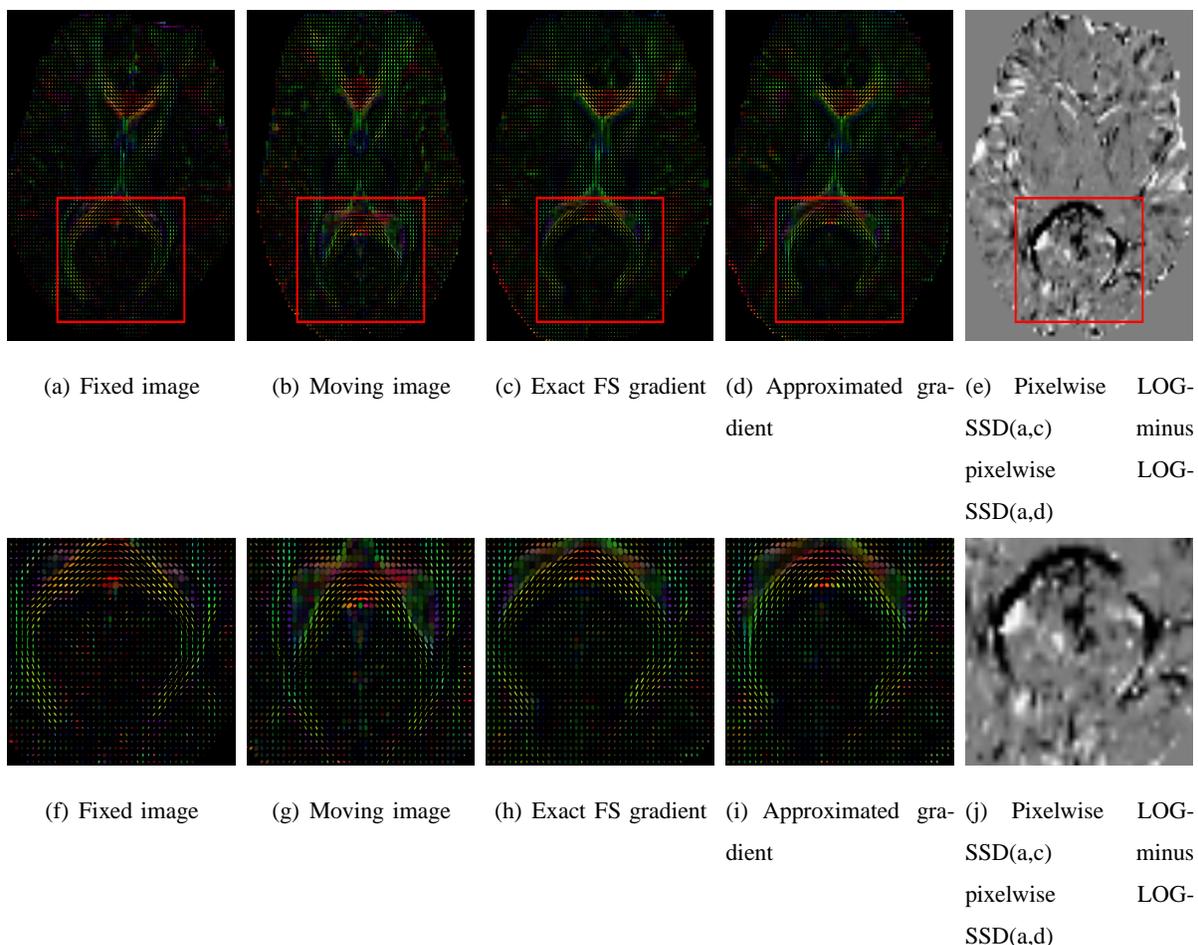


Fig. 2. Qualitative comparison between the use of exact FS gradient and approximated gradient for registering a second pair of subjects. Log-Euclidean interpolation and LOG-SSD are used in the objective function. (a) Fixed Image (b) Moving Image (c) Registration result from using exact FS gradient. LOG-SSD = 0.54. Harmonic Energy = 0.15 (d) Registration result from using approximated gradient. LOG-SSD = 0.49. Harmonic Energy = 0.12 (e) Pixelwise LOG-SSD between image (a) and (c) minus pixelwise LOG-SSD between image (a) and (d). Black region implies exact FS gradient is performing better in the region. White region implies approximated gradient is performing better in the region. 56% of the foreground pixels attain better LOG-SSD using the exact FS gradient than using the approximate gradient, even though there is more deformation when using the approximated gradient, as reflected in the harmonic energy. In this case, the corpus callosum appears to be significantly better aligned when the exact FS gradient is used. (f-j) zooms into the red box bounding the corpus callosum in (a-e).

we find that the statistics we compute appear to converge after about 30 pairwise registrations, hence 40 pairwise registrations appear sufficient for our purpose.

As implied by previous literature [10], it does not make sense to compare two registration algorithms with a fixed tradeoff between the similarity measure and regularization, especially when the two algorithms use different similarity measures and/or regularizations. Furthermore, one needs to be careful with the

tradeoff selection for optimal performance [29] in a given application.

In this paper, even though we are considering algorithms with the same similarity measure and regularization (and effectively the same implementation) but different optimization schemes, we find that for a fixed-size smoothing kernel, using the exact FS differential tends to converge to a solution of lower harmonic energy, i.e., a smoother transformation field. We define the harmonic energy to be the average over all pixels of the squared Frobenius norm of the Jacobian of the displacement field. Here, the Jacobian exclude the identity in Eq. (3).

Smaller harmonic energy implies a smoother deformation. This provides some evidence that the reorientation provides additional constraint on the registration problem. Therefore, to properly compare the algorithms, we consider smoothing kernels of sizes: $\{0.5, 0.6, \dots, 1.9, 2.0\}$. Note that bigger kernel sizes lead to more smoothing and thus lower harmonic energy. In particular, we perform the following experiment:

For each pair of subjects

For each kernel size

- i) Run the diffeomorphic demons registration algorithm using Euclidean interpolation and EUC-SSD using
 - (a) Exact FS gradient.
 - (b) Approximate gradient by ignoring reorientation.
 - (c) Fixed image gradient. This is the gradient proposed in Thirion's original demons algorithm [25].
- ii) Repeat (i) using Log-Euclidean interpolation and LOG-SSD.

For each pair of subjects, we therefore obtain a set of MSE with corresponding harmonic energies. We note that the harmonic energies and MSE across different pair of subjects are different. To average across trials, we linearly interpolate the MSE over a fixed set of harmonic energies for each pair of subjects. We can then compute the mean as well as the standard error of MSE across trials for a particular harmonic energy.

Exact vs Approximated Gradient. Figure 3(a) and 3(b) show the differences between the EUC-MSE (and the LOG-MSE) of the exact FS gradient and the approximate gradient when using Euclidean interpolation and EUC-SSD in the registration objective function. Figure 3(c) and 3(d) show the differences between the EUC-MSE (and the LOG-MSE) of the exact FS gradient and approximate gradient when using Log-Euclidean interpolation and LOG-SSD in the registration objective function. A negative values

imply a lower MSE for a given harmonic energy. The error bars show the standard error in MSE across different pairs of subjects, thus conveying the confidence of the results. In particular, we see that using the exact FS gradient yields a lower EUC-MSE and LOG-MSE over the entire spectrum of harmonic energies.

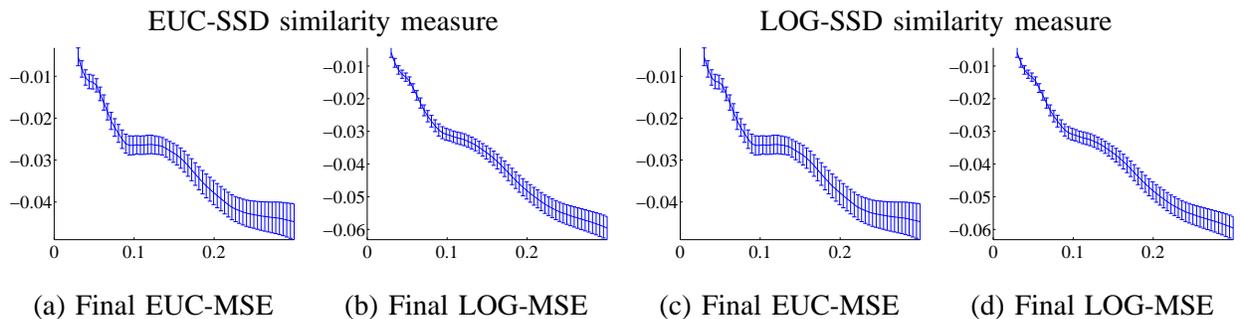


Fig. 3. Comparison of exact FS gradient and approximated gradient over an entire spectrum of harmonic energy (x-axis). Harmonic energy is increased by decreasing the size of the smoothing kernel. Y-axis corresponds to differences in MSE. The error bars show the standard error in MSE across different pairs of subjects. Negative values imply the FS gradient is obtaining lower MSE.

Exact vs Fixed Image Gradient. Figure 4 compares the registration results of the exact FS gradient with the fixed image gradient. Like before, we see that using the exact FS gradient yields a lower EUC-MSE and LOG-MSE over the entire spectrum of harmonic energies

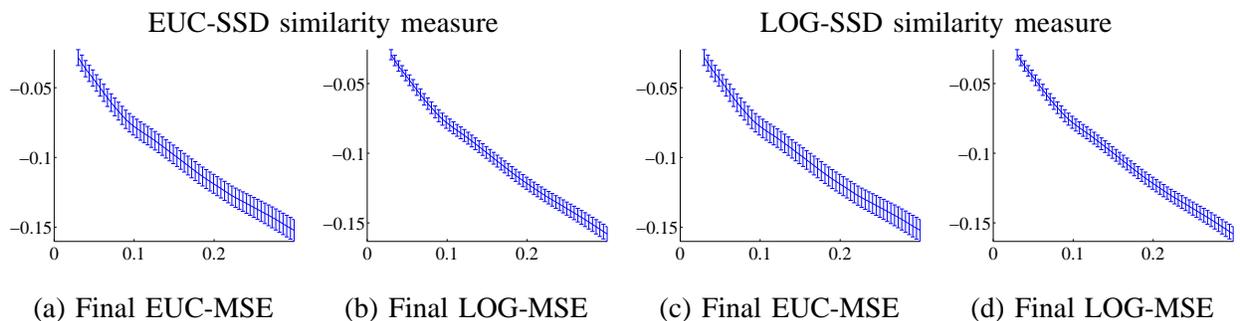


Fig. 4. Comparison of exact FS gradient and fixed image gradient over an entire spectrum of harmonic energy (x-axis). Harmonic energy is increased by decreasing the size of the smoothing kernel. Y-axis corresponds to differences in MSE. The error bars show the standard error in MSE across different pairs of subjects. Negative values imply the FS gradient is obtaining lower MSE.

Further Discussion. We emphasize that the improvements persist in figures 3(b,c) and 4(b,c) even though a different similarity measure from the original objective function is used. From figure 3 and

figure 4, we see that there is improvement over the entire range of harmonic energies and the improvement is statistically significant (one-sided paired-samples t-test p-value less than 10^{-5} for the entire range of harmonic energies).

Since Figures 3 and 4 only display the MSE differences, it is unclear whether the improvement is big. In figure 5, we plot the average MSE (over the 40 pair-wise registrations) with respect to the harmonic energies. We see that using the exact FS gradient gives the best results followed by the approximate gradient and the fixed image gradient. The amount of improvement increases as the harmonic energies increase. In our experiments, a harmonic energy of 0.3 corresponds to severe distortion (pushing the limits of the numerical stability of scaling and squaring), while a harmonic energy of 0.03 corresponds to very smooth warps. Previous literature, such as [29], suggests that extreme distortion causes overfitting, while extremely smooth warps might result in insufficient fitting. Only a concrete application can inform us the optimal amount of distortion and is the subject of future studies. For now, we assume a “safe” range to assess the different gradients to be between harmonic energies 0.1 and 0.2. From the values in figure 5, we conclude that the exact FS gradient provides an improvement of between 5 to 10 percent over the approximate gradient in this “safe” range of harmonic energies.

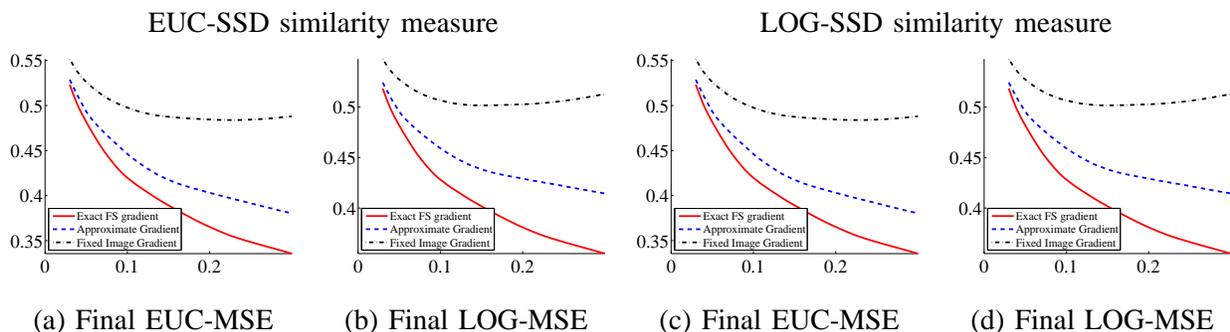


Fig. 5. Comparison of exact FS gradient and fixed image gradient over an entire spectrum of harmonic energy (x-axis). Harmonic energy is increased by decreasing the size of the smoothing kernel. Y-axis corresponds to differences in MSE. The error bars show the standard error in MSE across different pairs of subjects. Negative values imply the FS gradient is obtaining lower MSE.

Since Gauss-Newton optimization allows the use of “big steps” in the optimization, this might cause the approximated gradient to be more sensitive to the reorientation. Another optimization method such as conjugate gradient might improve the results of using the approximated gradient, since it allows the algorithm to take “smaller steps” and reorientate after each “small step”. However, from optimization theory and from our experience, Gauss-Newton method requires much fewer iterations to converge than

conjugate gradient. Furthermore, conjugate gradient requires a line search, resulting in many function evaluations. Function evaluations are quite expensive in our case, because of the need to reorientate and perform “scaling and squaring” of the velocity field. On the other hand, we find that in practice, line search is not necessary with Gauss-Newton optimization.

C. Quantitative Evaluation II

We perform a second set of experiments to recover randomly generated synthetic warps. Given a DT image, we first generate a set of random warps by sampling a random velocity at each voxel location from an independent and identically distributed (I.I.D.) gaussian. The foreground mask is then used to remove the velocity field from the background voxels. The resulting velocity field is smoothed spatially with a gaussian filter. We compute the resulting displacement field by “scaling and squaring”. This displacement field is used to warp the given DT image using Log-Euclidean interpolation and FS reorientation. I.I.D. gaussian noise is added to the warped DT image. We note that these random synthetic warps follow a stationary velocity field deformation model of [4], [5], [17] and is *different* from our deformation model. In this work, we are considering composition of diffeomorphisms parameterized by stationary velocity field: $\Phi_{v_1} \circ \dots \circ \Phi_{v_n}$. In general, there does not exist a velocity field u , such that $\Phi_u = \Phi_{v_1} \circ \dots \circ \Phi_{v_n}$.

We take a single DT image and generate 40 sets of random warps. We obtain an average displacement of 9.7mm over the foreground voxels. The average harmonic energy is 0.16, and the average sum of squares energy of the warps is 116mm^2 . We then perform pairwise registration between the DT image and the warped DT image using LOG-SSD. Once again, we consider a wide range of smoothing kernel sizes. For each registration, we compute the SSD between the ground truth random warps and the deformation field we obtain from the registration algorithm averaged over all the foreground voxels. We also compute the registration error in mm between the random warps and the estimated deformation field averaged over all the foreground voxels. Note that under the identity transformation, the average SSD will be 116mm^2 and the average registration error will be 9.7mm.

Figure 6(a) shows the differences between the SSD of the exact FS gradient and the approximated gradient. Figure 6(b) shows the differences between the SSD of the exact FS gradient and the fixed image gradient. Negative values imply that the exact FS gradient is achieving lower SSD. The error bars show the standard error in SSD, thus conveying the confidence of the results. In particular, we see that using the exact FS gradient yields the lowest SSD over the entire spectrum of harmonic energies. Similarly, Figure 6(c) and 6(d) show that the exact FS gradient achieves the smallest registration error than both the approximated and fixed image gradient.

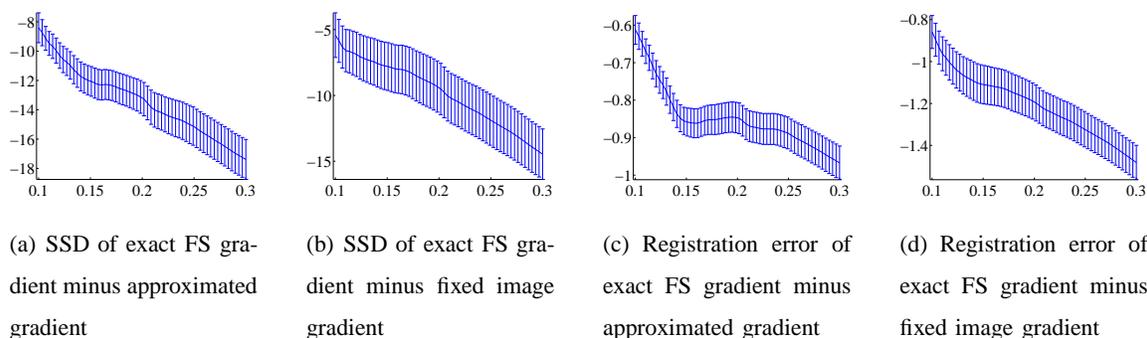


Fig. 6. Comparison of exact FS gradient, approximated gradient and fixed image gradient over an entire spectrum of harmonic energy (x-axis). Harmonic energy is increased by decreasing the size of the smoothing kernel. Y-axes of (a,b) correspond to differences in SSD. Y-axes of (c,d) correspond to differences in registration error in mm. The error bars show the standard error in SSD or registration error across different trials. Negative values in (a,b) imply that the exact FS gradient is obtaining lower SSD. Negative values in (c,d) imply that the exact FS gradient is achieving lower registration error.

To get a better idea of the magnitude of improvement, figure 7(a) shows the SSD (averaged over 40 trials) of the three gradients we are considering. Figure 7(b) shows the average registration error. Once again, the exact FS gradient obtains the lowest SSD and registration error. Interestingly, the fixed image gradient achieves a lower SSD but a higher registration error than the approximated gradient. This suggests that the approximated gradient suffers relatively big errors in some anatomical regions which are amplified by the quadratic penalty when using SSD. These errors dominate the average SSD. On the other hand, the registration error measures the absolute difference between the random warps and the estimated deformation field, so there is no extra weight on failed registration.

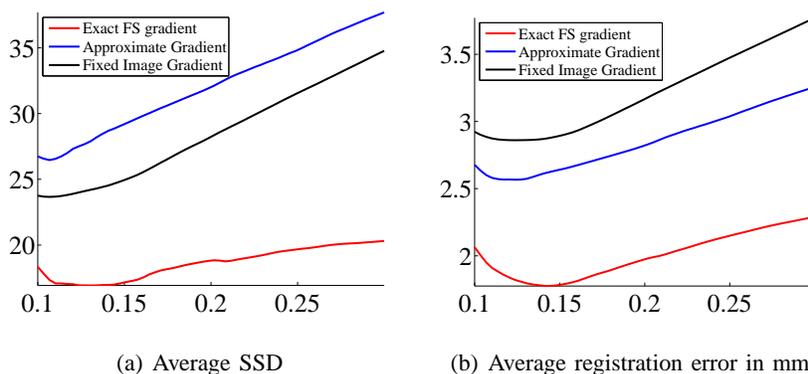


Fig. 7. (a) Average SSD (y-axis) over an entire spectrum of harmonic energy (x-axis). (b) Average registration error (y-axis) over an entire spectrum of harmonic energy (x-axis). Harmonic energy is increased by decreasing the size of the smoothing kernel.

From the plots, we see that the exact FS gradient is able to recover the ground truth warps up to 1.78mm or 18.3% error with respect to the average 9.7mm random warps. The approximated gradient achieves 2.56mm or 26.4% error. Finally, the fixed image gradient achieves 2.86mm or 29.4% error. Therefore, the exact FS gradient achieves an average of 31% and 38% reduction in registration error compared with the approximate gradient and fixed image gradient respectively.

Another interesting fact is that the best registration occurs when the kernel size is such that the harmonic energy is about 0.14, which is close to the average harmonic energy of the synthetic warps reported earlier.

VI. CONCLUSION

In this work, we derive the exact differential of the FS reorientation. We propose a fast diffeomorphic DT image registration algorithm using the exact FS differential. We show that the use of the exact differential improve EUC-SSD and LOG-SSD by 5 to 10 percent over an entire spectrum of harmonic energies. The improvements persist even if we use a different similarity measure from the objective function we optimize. In a second experiment, the exact differential reduces registration error on a set of randomly generated warps by as much as 31% compared with the approximated gradient which ignores reorientation.

APPENDIX A

FINITE STRAIN DIFFERENTIAL

In [13], the differential of the matrix $r = A(A^T A)^{-\frac{1}{2}}$ is computed, where $A = YX^T$ and Y and X are $3 \times n$ matrices. In the context of [13], X are the measured coordinates of a set of labeled points and Y are their measured positions after rigid body motion. X and Y can be used to estimate the rotation component of the rigid motion r using the least-squares estimate $r = A(A^T A)^{-\frac{1}{2}}$. Finding the differential dr in terms X and Y therefore allows the error analysis of the estimate r when the measurements X and Y are noisy.

From [13], we know that $dr r^T = -r dr^T$. Defining $\delta r \triangleq dr r^T$, we have

$$\delta r \triangleq dr r^T = -r dr^T \quad (36)$$

Note that δr is a skew symmetric matrix, and therefore takes the form

$$\delta r = \begin{pmatrix} 0 & -m_3 & m_2 \\ m_3 & 0 & -m_1 \\ -m_2 & m_1 & 0 \end{pmatrix} \triangleq m^\oplus \quad (37)$$

Denote $\text{vec}(\delta r) \triangleq (m_1, m_2, m_3)^T$ and $S \triangleq (A^T A)^{\frac{1}{2}}$. Then, the major result of [13] is that

$$\text{vec}(\delta r) = r(\text{tr}(S)I - S)^{-1}r^T \left(\sum_i (rX)_i \otimes dY_i + (rdX)_i \otimes Y_i \right) \quad (38)$$

where $(\cdot)_i$ denote the i -th column of (\cdot) respectively and \otimes denotes cross product.

Recall that we are interested in dR , where $R = (JJ^T)^{-\frac{1}{2}}J$. By setting $J \triangleq A^T$ and $S \triangleq (A^T A)^{\frac{1}{2}} = (JJ^T)^{\frac{1}{2}}$, we get $R = r^T$ and $dR = dr^T$. Therefore,

$$\delta r \stackrel{(36)}{=} -rdR \quad (39)$$

Using $I = r^T r$, we get

$$dR \stackrel{(39)}{=} -r^T \delta r = -R\delta r \quad (40)$$

By setting, $X = I$ and so $A = YI^T = J^T$, we finally obtain

$$\begin{aligned} dR &\stackrel{(40)}{=} -R\delta r \\ &\stackrel{(38)}{=} -R \left[R^T (\text{tr}(S)I - S)^{-1} R \sum_i (R^T)_i \otimes (dJ^T)_i \right]^{\oplus} \end{aligned} \quad (41)$$

APPENDIX B

ROTATION DERIVATIVES

For completeness, we will now derive the expressions for $\frac{\partial R(n)}{\partial w(j)}(w(j) = j)$, where j are the neighboring voxels of voxel n . Recall that $\{n_{x-}, n_{x+}, n_{y-}, n_{y+}, n_{z-}, n_{z+}\}$ are the neighbors of voxel n in the x , y and z directions respectively. u_x, u_y, u_z are the components of the displacement field in the x , y and z directions. For convenience, we denote $\frac{\partial R(n)}{\partial w(j)}(w(j) = j) = \left\{ \frac{\partial R(n)}{\partial w_x(j)}, \frac{\partial R(n)}{\partial w_y(j)}, \frac{\partial R(n)}{\partial w_z(j)} \right\} = \left\{ \frac{\partial R(n)}{\partial w_k(j)} \right\}$. Using chain rule, we have

$$\frac{\partial R(n)}{\partial w_k(n_{x+})} = \sum_m \sum_{ij} \frac{\partial R(n)}{\partial J_{ij}(n)} \frac{\partial J_{ij}(n)}{\partial (s \circ w_m(n_{x+}))} \frac{\partial (s \circ w_m(n_{x+}))}{\partial w_k(n_{x+})} (w_k(n_{x+}) = n_{x+}) \quad (42)$$

$$= \sum_m \frac{\partial R(n)}{\partial J_{m1}(n)} \frac{\partial J_{m1}(n)}{\partial (s \circ w_m(n_{x+}))} \frac{\partial (s \circ w_m(n_{x+}))}{\partial w_k(n_{x+})} (w_k(n_{x+}) = n_{x+}) \quad (43)$$

$$= \frac{1}{2\Delta x} \sum_m \frac{\partial R(n)}{\partial J_{m1}(n)} J_{mk}(n_{x+}) \quad (44)$$

The second and third equalities come from evaluating $\frac{\partial J_{ij}(n)}{\partial(\text{so}w_m(n_{x+}))}$, which are mostly zeros. Notice that $J_{m1}(n)$ and $J_{mk}(n_{x+})$ are evaluated at two different voxels. Similarly, we have

$$\begin{aligned}
\frac{\partial R(n)}{\partial w_k(n_{x-})} &= -\frac{1}{2\Delta x} \sum_m \frac{\partial R(n)}{\partial J_{m1}(n)} J_{mk}(n_{x-}) \\
\frac{\partial R(n)}{\partial w_k(n_{y+})} &= \frac{1}{2\Delta y} \sum_m \frac{\partial R(n)}{\partial J_{m2}(n)} J_{mk}(n_{y+}) \\
\frac{\partial R(n)}{\partial w_k(n_{y-})} &= -\frac{1}{2\Delta y} \sum_m \frac{\partial R(n)}{\partial J_{m2}(n)} J_{mk}(n_{y-}) \\
\frac{\partial R(n)}{\partial w_k(n_{z+})} &= \frac{1}{2\Delta z} \sum_m \frac{\partial R(n)}{\partial J_{m3}(n)} J_{mk}(n_{z+}) \\
\frac{\partial R(n)}{\partial w_k(n_{z-})} &= -\frac{1}{2\Delta z} \sum_m \frac{\partial R(n)}{\partial J_{m3}(n)} J_{mk}(n_{z-})
\end{aligned} \tag{45}$$

ACKNOWLEDGMENTS

The authors would like to thank

REFERENCES

- [1] D. Alexander and J. Gee. Elastic Matching of Diffusion Tensor Images. *Computer Vision and Image Understanding*, 77:233–250, 2000.
- [2] D. Alexander, C. Pierpaoli, and J. Gee. Spatial Transformations of Diffusion Tensor Magnetic Resonance Images. *IEEE Transactions on Medical Imaging*, 20(11):1131–1139, 2001.
- [3] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Log-Euclidean Metrics for Fast and Simple Calculus on Diffusion Tensors. *Magnetic Resonance in Medicine*, 56(2):411–421, 2006.
- [4] Vincent Arsigny, Olivier Commowick, Xavier Pennec, and Nicholas Ayache. A Log-Euclidean framework for statistics on diffeomorphisms. volume 4190, pages 924–931, 2006.
- [5] J. Ashburner. A Fast Diffeomorphic Image Registration Algorithm. *NeuroImage*, 38:95–113, 2007.
- [6] P. Basser and S. Pajevic. Estimation of the Effective Self-Diffusion Tensor from the NMR Spin Echo. *Journal of Magnetic Resonance*, 103(3):247–254, 1994.
- [7] M. Beg, M. Miller, A. Troune, and L. Younes. Computing Large Deformation Metric Mappings via Geodesic Flows of Diffeomorphisms. *International Journal of Computer Vision*, 61(2):139–157, 2005.
- [8] G. Birkhoff and G. Rota. *Ordinary Differential Equations*. John Wiley and Sons Inc, 1978.
- [9] M. Bro-Nielsen and C. Gramkow. Fast Fluid Registration of Medical Images. *Proc. Visualization in Biomedical Computing*, 1131:267–276, 1996.
- [10] P. Cachier, E. Bardinet, D. Dormont, X. Pennec, and N. Ayache. Iconic Feature Based Non-Rigid Registration: The PASHA algorithm. *Compute Vision and Image Understanding*, 89(2-3):272–298, 2003.
- [11] Y. Cao, M.I. Miller, S. Mori, R.L. Winslow, and L. Younes. Diffeomorphic Matching of Diffusin Tensor Images. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Workshop on Mathematical Methods in Biomedical Image Analysis*, 2006.

- [12] G. Christensen, R. Rabbit, and M. Miller. Deformable Templates Using Large Deformation Kinematics. *IEEE Transactions on Image Processing*, 5(10):1435–1447, 1996.
- [13] L. Dorst. First Order Error Propagation of the Procrustes Method for 3D Attitude Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2):221–229, 2005.
- [14] P. Fillard, X. Pennec, V. Arsigny, and N. Ayache. Clinical DT-MRI Estimation, Smoothing, and Fiber Tracking with Log-Euclidean Metrics. *IEEE Transactions on Medical Imaging*, 26(11):1472–1482, 2007.
- [15] J. Gee and D. Alexander. Diffusion Tensor Image Registration. In *Welckert and Hagen: Visualization and Image Processing of Tensor Fields*, 2005.
- [16] A. Guimond, C. Guttman, S. Warfield, and C-F. Westin. Deformable Registration of DT-MRI data Based on Transformation Invariant Tensor Characteristics. *Proceedings of IEEE International Symposium on Biomedical Imaging*, 2002.
- [17] M. Hernadez, M. Bossa, and S. Olmos. Registration of Anatomical Images Using Geodesic Paths of Diffeomorphisms Parameterized with Stationary Velocity Fields. *MMBIA*, 2007.
- [18] J. Keller. Closest Unitary, Orthogonal and Hermitian Operators to a Given Operator. *Mathematics Magazine*, 48(4):192–197, 1975.
- [19] G. Kindlmann, R. Estepar, M. Niethammer, S. Haker, and C. Westin. Geodesic-Loxodromes for Diffusion Tensor Interpolation and Difference Measurement. *MICCAI*, 2007.
- [20] A. Leemans, J. Sijbers, S. Backer, E. Vandervliet, and P. Parizel. Affine Coregistration of Diffusion Tensor Magnetic Resonance Images Using Mutual Information. *ACVIS*, 2005.
- [21] H-J. Park, M. Kubicki, M. Shenton, A. Guimond, R. McCarley, S. Maier, R. Kikinis, F. Jolesz, and C-F. Westin. Spatial Normalization of Diffusion Tensor MRI using Multiple Channels. *NeuroImage*, 20(4):1995–2009, 2003.
- [22] X. Pennec, P. Fillar, and N. Ayache. A Riemannian Framework for Tensor Computing. *International Journal of Computer Vision*, 66:41–66, 2005.
- [23] P. Rogelj and S. Kovacic. Symmetric Image Registration. *Medical Image Analysis*, 10(3):484–493, 2006.
- [24] J. Ruiz-Alzola, C-F. Westin, S. Warfield, C. Alberola, S. Maier, and R. Kikinis. Nonrigid Registration of 3D Tensor Medical Data. *Medical Image Analysis*, 6:143–161, 2002.
- [25] J. Thirion. Image Matching as a Diffusion Process: an Analogy with Maxwell’s Demons. *Medical Image Analysis*, 2(3):243–260, 1998.
- [26] T. Vercauteren, X. Pennec, E. Malis, A. Perchant, and N. Ayache. Insight into Efficient Image Registration Techniques and the Demons Algorithm. *IPMI*, 2007.
- [27] T. Vercauteren, X. Pennec, A. Perchant, and N. Ayache. Non-parametric Diffeomorphic Image Registration with the Demons Registration. *MICCAI*, 2007.
- [28] H. Wang, L. Dong, J. O’Daniel, R. Mohan, A. Garden, K. Ang, D. Kuban, M. Bonnen, J. Chang, and R. Cheung. Validation of an Accelerated ‘Demons’ Algorithm for Deformable Image Registration in Radiation Therapy. *Physics in Medicine and Biology*, 50(12), 2005.
- [29] B.T.T. Yeo, M. Sabuncu, R. Desikan, B. Fischl, and P. Golland. Effects of Registration Regularization and Atlas Sharpness on Segmentation Accuracy. *MICCAI*, pages 683–691, 2007.
- [30] H. Zhang, B. Avants, P. Yuschkevich, J. Woo, S. Wang, L. McCluskey, L. Elman, E. Melhem, and J. Gee. High-Dimensional Spatial Normalization of Diffusion Tensor Images Improves the Detection of White matter Differences: An Example Study Using Amyotrophic Lateral Sclerosis. *IEEE Transactions on Medical Imaging*, 26(11):1585–1597, 2007.

- [31] H. Zhang, P. Yushkevich, D. Alexander, and J. Gee. Deformable Registration of Diffusion Tensor MR Images with Explicit Orientation Optimization. *Medical Image Analysis*, 10(5):764–785, 2006.
- [32] U. Ziyang, M. Sabuncu, L. Donnell, and C-F. Westin. Nonlinear Registration of Diffusion MR Images Based on Fiber Bundles. *MICCAI*, 2007.