Parametric non-rigid registration using a stationary velocity field

Marc Modat, Pankaj Daga, M. Jorge Cardoso, Sebastien Ourselin,
Centre for Medical Image Computing,
University College London, WC1 6BT, UK
m.modat@ucl.ac.uk

Gerard R. Ridgway, John Ashburner
Wellcome Trust Centre for Neuroimaging,
12 Queen Square, London, WC1N 3BG, UK

Abstract

The Free-Form Deformation (FFD) algorithm is a widely-used approach for non-rigid registration. Modifications have previously been proposed to ensure topology preservation and invertibility within this framework. However, in practice, none of these yield the inverse transformation itself, and one loses the parsimonious B-spline parameterisation.

We present a novel log-Euclidean FFD approach, in which a spline model of a stationary velocity field is exponentiated to yield a diffeomorphism, using an efficient scaling-and-squaring algorithm. The log-Euclidean framework allows easy computation of a consistent inverse transformation, and offers advantages in group-wise atlas building and statistical analysis. We optimise the Normalised Mutual Information plus a regularisation term based on the Jacobian determinant of the velocity field.

The proposed method has been assessed against the conventional FFD using T1-weighted magnetic resonance brain images, following a published protocol with an openly available data-set (MGH10) to enable comparison with many other algorithms. The proposed method performed similarly to the state of the art.

Diffeomorphic frameworks for non-rigid registration lead to one-to-one transformations, which preserve topology and ensure invertibility. These features are of great interest as they enable plausible biomedical analysis of volume, shape or rate of change over time. They are, for example, the core of morphometric studies where correspondence between subjects is crucial. Whereas, due to the limited amount of change between time points, longitudinal changes are usually captured using small-deformation based algorithms, differences between patients are larger and include complex shape variability. Several diffeomorphic methods have been presented in the last few years: notably the Large Deformation Diffeomorphic Metric Mapping (LDDMM) algorithm [4], Dartel [3], the diffeomorphic demons [21] and Hernandez *et al.* [6] method. All these algorithms have in common that they are based on the structure of the infinite-dimensional group of diffeomorphisms (analogous to finite-dimensional Lie groups), in that compositions of diffeomorphisms are guaranteed to remain in the group, i.e. are also diffeomorphic [14]. In theory, the diffeomorphic formulation assumes continuous space and time; however, in practice, the implementation is based on a relatively small number of discrete time integration steps, and is spatially sampled at discrete voxels. The LDDMM method differs from the others as it can cover a larger range of diffeomorphisms due to its parametrisation being variable in time. However, this is achieved with significantly higher computation time and memory requirement.

The Free-Form Deformation (FFD) algorithm [17] is a well-known and established method which has been found to perform well for inter-subject registration [9], but which is not guaranteed to yield invertible transformations. One appealing aspect of the conventional FFD algorithm is its parametric nature, in which a relatively low dimensional set of degrees of freedom needs to be optimised (and stored, and potentially analysed), in the form of a lattice of control points for a cubic B-spline model. This more parsimonious parametrisation might have advantages for statistical analysis or shape modelling [16].

Rohlfing *et al.* [13] and Sdika [18] modified the FFD to ensure invertibility by using the Jacobian determinant respectively as a penalty term or as a constraint. However, these methods do not have the flexibility of diffeomorphic approaches to generate large deformations by integrating flows over time [4]. Rohlfing *et al.* also rely on finite differencing for the gradient of the objective function, which may reduce accuracy and makes the computation expensive since it requires extra resampling of sub-volumes of the image and multiple objective function evaluations.

Rueckert *et al.* [15] used compositions of spatially constrained transformations to yield large deformation diffeomorphisms. However, they found the method slightly worsened performance, perhaps because the individual transformations were very strongly constrained, meaning that a large number were required to be composed, potentially increasing errors.

Importantly, these three techniques for obtaining invertible free form deformations do not benefit from some elegant theoretical and practical aspects of the diffeomorphism group setting. For example, while they ensure invertibility in theory, in practice they do not easily yield the inverse transformations. Rueckert *et al.*'s method enables the computation of the inverse transformation only if each of the many composed small deformations are kept so that they can be inverted and composed in reverse order. In contrast, methods that integrate flows can simply integrate them in reverse to obtain a consistent inverse [3]. Similarly, Riemannian interpolation, extrapolation and averaging procedures that account for the non-Euclidean structure of the group of diffeomorphisms should be superior to their simpler equivalents, as argued by Joshi *et al.* [7] regarding the estimation of group-wise average templates.

For the special case of diffeomorphisms in one-parameter subgroups, one can exponentiate stationary velocity fields using an efficient scaling-and-squaring approach [2]. Furthermore, log-Euclidean analysis of diffeomorphisms via their velocity fields greatly simplifies computation of averages or variability [2]. Directly modelling discretised versions of these continuous velocity fields leads to particularly efficient implementations [3, 21].

The cited diffeomorphic algorithms [4, 3, 21, 6] use a smoothing approach or a regularizer to ensure a one-to-one mapping. The smoothing or regularisation parameter is thus of great importance; under-regularisation can lead to folded transformations while over-regularisation can prevent precise alignment. We propose to take advantage of the spline's closed-form expression for the Jacobian matrix to penalize our transformation model. Using the squared log of the Jacobian determinant and its analytical derivative, we ensure that the transformation is unfolded for any penalty term weight.

We propose that the merits of Arsigny's log-Euclidean framework and the FFD's parametric nature can be combined; in this work we model a stationary velocity field in a B-spline basis, and efficiently approximate the scaling-and-squaring approach in the lower-dimensional space of the FFD control points before refining the transformation in the high-dimensional space of the reference image. As emphasised already, this automatically leads to a consistent inverse transformation, and enables a simple procedure for building group-wise (log-Euclidean) average templates. The appeal of the FFD is retained, along with its computational efficiency and ease of parallelisation [10].

Finally, by optimising the velocity field to maximise the Normalised Mutual Information (NMI), we obtain one of relatively few diffeomorphic implementations suitable for multi-modal fusion applications. This is of interest for example for EPI- (echo planar imaging) distortion correction [20].

In the next section, we present the details of the proposed method: the fast free-form diffeomorphic deformation (F$_3$D^2). The experimentation follows the Klein *et al.* study [9] enabling direct comparison to several other algorithms. The final section will discuss the method and its merits with respect to other approaches.

1. Method

1.1. Overview of Free-Form Deformation

A floating image F is transformed into a warped image $F(\mathbf{T})$ to maximise its similarity with a reference image R. The transformation $\mathbf{T}$ is computed through a cubic B-spline interpolation from a lattice of control points $\{\mu^{\xi}_{i,j,k}\}$ overlaid on R. Indices i, j, k correspond to the points along the x-, y- and z-axis and ξ denotes the x-, y- or z-component. The spacing between control points along each axis is denoted as δ_{ξ}. The cubic B-spline function is represented by β^3. Thus $\mathbf{T}^{\xi}(\vec{x}) = \vec{x}^{\xi} +$

$$\sum_{i,j,k} \beta^3\left(\frac{x}{\delta_x} - i\right)\beta^3\left(\frac{y}{\delta_y} - j\right)\beta^3\left(\frac{z}{\delta_z} - k\right)\mu^{\xi}_{ijk}. \quad (1)$$

The control points are adjusted to maximise an objective function that balances a measure of image similarity with a smoothness-favouring penalty term. The similarity between $F(\mathbf{T})$ and R is assessed with the NMI [19], an entropy-based measure that aims to quantify the amount of information that one image has about the other. The penalty term is usually the bending-energy[17], which enforces smoothness but does not guarantee one-to-one mappings. The FFD algorithm is not able to handle large deformations such as the classical circle to c case, illustrated in Figure 1.

1.2. Scaling-and-squaring on a control point lattice

In F$_3$D^2 control points are used to parametrize a stationary velocity field rather than a displacement field; the final transformation is then computed through exponentiation.

The exponential of a smooth vector field $\mathbf{V}$ is ensured to be a diffeomorphism [14], and can be computed using an Euler integration approach that consists of successive compositions of deformation fields with small time steps. This computation can be efficiently approximated using a scaling-and-squaring approach as shown by Arsigny *et al.*

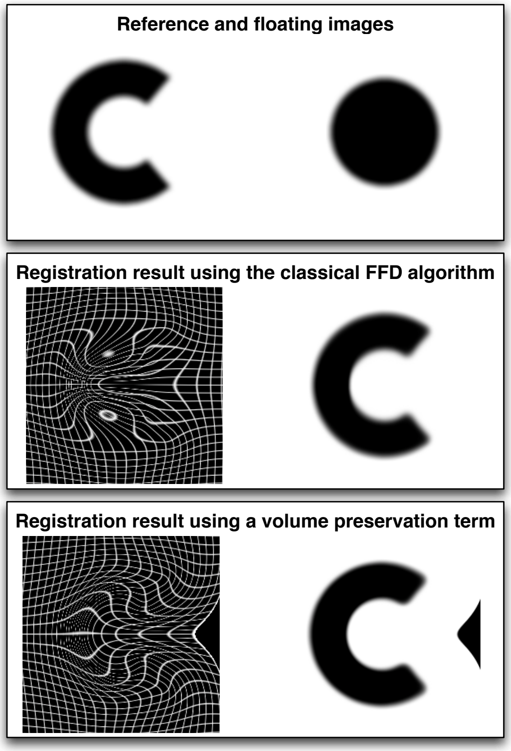
Reference and floating images
Registration result using the classical FFD algorithm
Registration result using a volume preservation term

Figure 1. Circle to c registration using the FFD. (Top) the reference and floating images. (Middle) registration result where folding has occurred. (Bottom) registration result using a volume preserving penalty term; note that deformation is propagated to the image border.

[2] and employed in [3, 21, 6]. The scaling-and-squaring approach involves an initial scaling of the velocity field $\mathbf{V}$ to obtain a small deformation field $\mathbf{u}$. The composition of the resultant field with itself several times approximates its integration over a large number of steps and leading to the final deformation $\mathbf{U}$. We report the reader to Asburner [3], figures 4 and 5 for a scaling-and-squaring illustration.

In our approach, we approximate the final deformation field $\mathbf{U}$ using the low-resolution space defined by the control point lattice, enabling greatly improved efficiency. The stationary velocity field, $\{\vec{\mu}\}$, is considered only at the control points' initial positions leading to an initial deformation field $\mathbf{u}_l$. The final deformation field in the control point space $\mathbf{U}_l$, is computed using squaring of the initial field through a cubic spline interpolation method. Using the final control point based deformation field, we can then compute an approximated field $\tilde{\mathbf{U}}$ defined in the reference image space. Table 1 presents inverse-consistency errors for the high- and low-resolution squaring method. For each configuration of image size and control point spacing we generated 100 random initial velocity fields. We computed all the deformation fields $\mathbf{U}$ and $\tilde{\mathbf{U}}$ as well as their inverses, as described in subsection 1.3. We then composed, denoted by $\circ$, both transformations and computed the mean euclidean distance to an identity transformation. Note that $\mathbf{U} \circ \mathbf{U}^{-1}$ and $\mathbf{U}^{-1} \circ \mathbf{U}$ gave similar inverse consistency errors. Consistency errors have been averaged for the different image size as the error did not change with different image dimension. All deformation fields have been generated using 6 squaring steps. Table 1 also reports the speed-up ratio between the computation of $\mathbf{U}$ and $\tilde{\mathbf{U}}$ (single-threaded implementation on a 3GHz Xeon).

Table 1. Comparison of the high- and low-resolution squaring. Tests have been performed using different (cubic) image sizes and control point spacings (δ, in voxels).

	$\delta = \mathbf{2.5}$	$\delta = \mathbf{5}$	$\delta = \mathbf{10}$
Error($\mathbf{U} \circ \mathbf{U}^{-1}$)	0.0229	0.0149	0.0080
Error($\tilde{\mathbf{U}} \circ \tilde{\mathbf{U}}^{-1}$)	0.0440	0.0856	0.1694
Speed-up ratio (64^3 image)	2.1	3.7	6.6
Speed-up ratio (128^3 image)	2.2	4.0	6.9
Speed-up ratio (256^3 image)	2.5	5.0	8.2

It can be noticed from table 1 that the difference between the proposed approximation and the dense scaling-and-squaring approach is not negligible. In order to merge efficiency and accuracy, we therefore used a 2-step approach where $\{\vec{\mu}\}$ were first optimised using the scaling-and-squaring approach in the control point space. After convergence, the scaling-and-squaring approach was then performed in the dense reference image space until second convergence. It can also be noticed from the table that the speed-up ratio is not proportional to the ratio of voxels to control points. This is because we used linear interpolation to perform squaring of the dense deformation field, while we used cubic spline interpolation for squaring in the control point space.

1.3. The inverse transformation

A diffeomorphic transformation $\mathbf{T}$ implies invertibility in a sense that $\mathbf{T}^{-1}$ is defined. However, not all diffeomorphic algorithms explicitly provide the $\mathbf{T}^{-1}$ transformation.

Since our transformation is based on an exponential map, we know that $\exp(-\mathbf{V}) = (\exp \mathbf{V})^{-1}$ [14]. It it thus possible, using the scaling-and-squaring approach and $\{-\vec{v}_{i,j,k}\}$ as a stationary velocity field, to estimate $\mathbf{T}^{-1}$. The same technique could be applied in both the voxel and the control point space. To assess the influence of the number of squaring steps on the inverse consistency, we generated 100 stationary velocity fields that we integrated using different numbers of steps. We report in table 2 the inverse consistency errors using a 256^3 image size and a 5 voxel control point spacing. The inverse consistency error decreases as the number of squaring step increases, is at a plateau from 6 to 9 steps, then increases due to limited numerical precision. Table 2 has been generated using double precision but similar results are obtained with single precision.

Table 2. Inverse consistency error using different squaring step number.

Step number	Error($\mathbf{U} \circ \mathbf{U}^{-1}$)	Error($\mathbf{U} \circ \mathbf{U}^{-1}$)
2	0.0390	0.0389
3	0.0243	0.0244
4	0.0184	0.0184
5	0.0161	0.0161
6	0.0153	0.0153
7	0.0150	0.0150
8	0.0149	0.0149
9	0.0154	0.0154
10	0.0170	0.0170
11	0.0221	0.0220
12	0.0338	0.0338

1.4. Optimisation procedure

Within the FFD framework, the objective function to maximise consists of a similarity measure (NMI) and one or several penalty terms P which enforce a smooth deformation. The penalty terms we used are the bending-energy, $\mathcal{P}_{\text{BE}}$, the divergence of the velocity field, $\mathcal{P}_{\text{div}}$, and similarly to Rohlfing *et al.* [13] a term $\mathcal{P}_{\text{Jac}}$ based on the Jacobian determinant:

$$\mathcal{P}_{\text{Jac}} = \log^2(\det(\text{Jac}(\mathbf{T}))). \tag{2}$$

In order to generate a diffeomorphism, one must ensure that the scaled velocity field has positive Jacobian determinant. Our approach enables us to analytically compute the Jacobian determinant and its gradient from the B-spline parametrisation in the initial dense deformation field. By using $\mathcal{P}_{\text{Jac}}$ as a penalty term we guarantee diffeomorphism.

The objective function $\mathcal{O}(R, F; \vec{\mathbf{v}})$ to maximise is thus:

$$\begin{aligned} \mathcal{O}(R, F; \vec{\mathbf{v}}) &= \alpha \times \text{NMI} \\ &- \beta \times \mathcal{P}_{\text{BE}} \\ &- \gamma \times \mathcal{P}_{\text{Jac}} \\ &- \epsilon \times \mathcal{P}_{\text{div}}, \end{aligned} \tag{3}$$

The trade-off values α, β, γ and ϵ are chosen to sum to 1.

In order to maximise $\mathcal{O}(R, F; \vec{\mathbf{v}})$, the derivatives of the NMI and the derivative of the penalty terms have to be computed. In the classical FFD framework that assumes a small deformation model, the NMI gradient is computed using R and $F(\mathbf{T})$. Typically, the derivatives of a joint intensity distribution are computed, from which the derivatives of the marginal and joint entropies are assessed before obtaining the gradients of the NMI for each control point. With a stationary velocity field parametrisation, the gradient must be appropriately transformed to account for the exponentiation. We first compute the similarity measure gradient between R and $F(\mathbf{T})$ and then use a scaling-and-squaring approach to compute the gradient as follows:

- Compute the voxel-wise NMI gradient $\mathbf{G}$ between R and $F(\mathbf{T})$
- Compute the initial deformation field $\mathbf{u}_{-1/2^n}$ by negating and scaling the velocity field.
- Compute for each voxel the Jacobian matrix $\mathbf{J}_{-1/2^n}$ from $\mathbf{u}_{-1/2^n}$
- Update the gradient: $\mathbf{G}(\vec{x}) = \mathbf{G}(\vec{x}) + \mathbf{u}_{-1/2^n}(\mathbf{G}(\vec{x})) \times \mathbf{J}_{-1/2^n}(\vec{x})^T$
- Perform similarly for each subsequent squaring step.
- Extract the node-wise NMI gradient from the obtained voxel-wise gradient, as in [10].

Using a stationary velocity field parametrisation, we are able to compute the inverse which enables us to use a symmetric formulation of the similarity measure.

$$\text{NMI} = \frac{H(R) + H(F(\mathbf{T}))}{H(R, F(\mathbf{T}))} + \frac{H(F) + H(R(\mathbf{T}^{-1}))}{H(F, R(\mathbf{T}^{-1}))} \tag{4}$$

where $H(R)$, $H(F)$, $H(F(\mathbf{T}))$ and $H(R(\mathbf{T}^{-1}))$ are the marginal entropy and $H(R, F(\mathbf{T}))$ and $H(F, T(\mathbf{T}^{-1}))$ are the joint entropies of pair of images. These entropy values are computed from a joint histogram, which in our implementation is based on a Parzen window approach where a cubic B-spline is used to define the window.

The penalty terms are computed here directly on the velocity field, analogous to their computation on the deformation field in the classical FFD. We refer the reader to [10] for an efficient analytical computation of the bending-energy term. Rohlfing *et al.* [13] used an optimisation based on a symmetric difference evaluation of the objective function. Novelly here, for efficiency and accuracy, we use the analytical derivative of the objective function. The Jacobian based penalty term can be differentiated with regards to each degree of freedom $v^{\xi}_{i,j,k}$, as:

$$\begin{aligned} \frac{\partial \mathcal{P}_{\text{Jac}}}{\partial v^{\xi}_{i,j,k}} &= 2 \times \log\left(\det(\text{Jac}(\mathbf{T}))\right) \\ &\times \ \text{Tr}\left(\text{Jac}^{-1}(\mathbf{T}) \frac{\partial \text{Jac}(\mathbf{T})}{\partial v^{\xi}_{i,j,k}}\right), \end{aligned} \tag{5}$$

where Tr(.) is the trace operator. For efficiency, one can compute the gradient only at the control point positions.

The set of all derivatives $\partial \mathcal{O}(R, F; \vec{\mathbf{v}}) / \partial v^{\xi}_{i,j,k}\}$ are then used to perform a conjugate gradient ascent, updating all degrees of freedom concurrently. The penalty terms prevent any folded solution from being accepted in the optimisation. However, since the optimisation is performed concurrently for every degree of freedom, a folded voxel could prematurely stop the line-ascent process. Karaçali and Davatzikos [8] proposed to locally unfold areas with negative Jacobian determinants. However, their method is based on a non-parametric deformation model and is thus not directly suitable for a cubic B-spline deformation model. Andersson *et al.* use this approach to unfold deformation fields[1] in their FNIRT software (http://www.fmrib.ox.ac.uk/fsl/fnirt/index.html). Instead, we propose to take advantage again of the analytical definition of the Jacobian matrices. We locally use the gradient of the Jacobian determinant of the folded voxel in order to increase their values. This is done by modifying the velocity field values along this Jacobian determinant gradient line. This approach has the advantage to be performed directly on the transformation parameters $\{\vec{\mu}\}$ and does not require approximation by going to a dense deformation field and back to the B-spline parameters. We applied this *unfolding step* each time a negative Jacobian determinant is produced while performing the gradient ascent.

2. Experiments and results

The proposed method has been assessed against the fast free-form deformation [10] (F_3D, part of the NiftyReg package) that directly optimises the control point final positions. We performed evaluation by following the strategy used in Klein *et al.* [9] to allow direct comparison.

The MGH10[1] database consists of 10 MRI acquired at the MGH/MIT/HMS Athinoula A. Martinos Center for Biomedical Imaging using a 3T Siemens scanner. Subject ages range from 22 to 29. Each brain has been manually parcellated into 73 regions of interest. Each of the 10 brain images have been registered to the 9 remaining images. The registrations were performed using the F_3D the proposed method first without any approximation and then using the approximation described in section 1.2. Registrations were performed after inhomegeneity correction and only voxels within the brain mask were considered for the optimisation process. It resulted in a total of 90 × 3 registrations, transformations from which were then used to propagate regions of interest between subject. We then used a segmentation overlap measure as a measure of registration accuracy. For every registration, the control point spacing was set to 2.5 voxels along each axis. This value was chosen to reproduce the parameters used with IRTK[2] [17]. The bending-energy weight was set to 1% for all registrations. The Jacobian based constraint and the divergence were both set to 10% for the F_3D^2 implementation. These trade-off values were defined empirically using visual assessment on other images. The scaling-and-squaring used 6 steps. Similarly to [9] we used the target overlap (TO) (eq. 6) to assess overlap between the gold standard segmentation (GS) and the propagated (PS) segmentations.

$$\text{TO} = \frac{\text{GS} \cap \text{PS}}{\text{GS}} \tag{6}$$

Mean results are presented in figure 2 and are directly comparable to those obtained by Klein *et al.* [9]. For comparison, we also report the TO values using only affine registration. The three approaches lead to similar results and are also similar to the best results reported in [9]. As expected, no folding occurred with the proposed F_3D^2 approaches, while F_3D generated some negative Jacobian values.

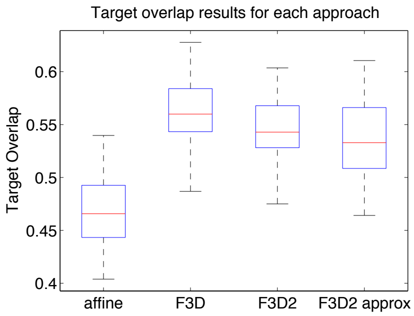

Figure 2. Segmentation propagation results for the MGH10 data.

All CPU-based registrations have been performed on a computer cluster with processors ranging from 1.8 to 3GHz. Using single-threaded implementation[3], the mean computation time were 4.7(0.8), 64.7(21.1) and 15.7(4.9) minutes for F_3D, F_3D^2 and F_3D^2 with approximation.

3. Discussion

We presented an implementation of the log-Euclidean method in the FFD framework. The method takes advantage of the scaling-and-squaring approach to efficiently obtain the deformation field. The stationary velocity field parametrisation enabled the optimisation of the NMI using a symmetric approach where forward and backward transformation are considered.

The proposed framework is guaranteed to avoid folding by using an analytical implementation of a Jacobian-based penalty term. Non-parametric formulations for non-rigid registration [4, 3, 21, 6] ensure diffeomorphism using a smoothing parameter or regularisation of the velocity field with a small enough time-step. However, if the amount of regularisation is too low or the time-step too high, folding can still occur. Using F_3D^2 the continuous property of the spline and the Jacobian-based penalty term ensure unfolded deformation for any control point spacing value and any penalty term trade-off values.

[1] Data information can be found here: http://speechlab.bu.edu/imaging.php

[2] http://www.doc.ic.ac.uk/∼dr/software/

[3] Note that the released implementations handle multi-core CPU architecture.

An efficient implementation of a directly invertible deformation would be helpful for several applications: groupwise registration of a large dataset for phenotyping using segmentation propagation; joint analysis of volumes and surfaces, since the latter are typically transformed by forward-mapping their vertex coordinates, instead of the backward-mapping more common for volumetric warping; or the inter-subject transformation of within-subject deformation or motion fields [12].

De Craene *et al.* [5] use Euler integration to follow the path of each voxel through a non-stationary velocity field over time. In their cardiac application, they have a large number of time-points (around 18 images per cardiac cycle), but do not specify the number or spacing of the control points in time (in space they have an initial grid of $3 \times 5 \times 5$ which is refined twice). With only two time-points, and a small number of control points in time, the velocity field would become approximately stationary in time, as ours is. However, two important differences remain. Firstly, by following just the points starting at the original control point locations, we have a far less computationally expensive algorithm, but potentially at the cost of modelling large deformations less accurately. The second difference is their use of evenly spaced Euler time-steps, instead of the scaling-and-squaring approach. The scaling-and-squaring approach for matrix exponentiation has been thoroughly studied and found to be superior to more naive implementations [11]. However, the exponentiation (integration) of velocity fields has been less thoroughly studied; for example, Arsigny [2] did not compare the method to any simpler approaches, so it remains possible that uniform Euler integration could be preferable. We intend to investigate both of these aspects in future work.

References

[1] J. L. R. Andersson, M. Jenkinson, and S. M. Smith. Nonlinear registration, aka spatial normalization: FMRIB technical report TR07JA2. Technical report, FMRIB Analysis Group of the University of Oxford, Oxford, UK, 2007.

[2] V. Arsigny, O. Commowick, X. Pennec, and N. Ayache. A Log-Euclidean framework for statistics on diffeomorphisms. In *Proc. MICCAI'06*, volume 4190, page 924, 2006.

[3] J. Ashburner. A fast diffeomorphic image registration algorithm. *Neuroimage*, 38(1):95–113, 2007.

[4] M. Beg, M. Miller, A. Trouvé, and L. Younes. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *Int. J. Comput. Vis.*, 61(2):139–157, 2005.

[5] M. De Craene, G. Piella, N. Duchateau, E. Silva, A. Doltra, H. Gao, J. D'hooge, O. Camara, J. Brugada, M. Sitges, and A. Frangi. Temporal diffeomorphic free-form deformation for strain quantification in 3D-US images. In *Proc. MICCAI'10*, pages 1–8, 2010.

[6] M. Hernandez, M. Bossa, and S. Olmos. Registration of anatomical images using paths of diffeomorphisms parameterized with stationary vector field flows. *Int. J. Comput. Vis.*, 85:291–306, 2009.

[7] S. Joshi, B. Davis, M. Jomier, and G. Gerig. Unbiased diffeomorphic atlas construction for computational anatomy. *Neuroimage*, 23:151–160, 2004.

[8] B. Karaçali and C. Davatzikos. Estimating topology preserving and smooth displacement fields. *IEEE Trans. Med. Imag.*, 23(7):868–80, 2004.

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

[10] M. Modat, G. R. Ridgway, Z. Taylor, M. Lehmann, J. Barnes, D. Hawkes, N. C. Fox, and S. Ourselin. Fast free-form deformation using graphics processing units. *Comput. Methods Programs Biomed.*, 98(3):278–84, 2010.

[11] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[12] A. Rao, R. Chandrashekara, G. Sanchez-Ortiz, R. Mohiaddin, P. Aljabar, J. Hajnal, B. Puri, and D. Rueckert. Spatial transformation of motion and deformation fields using nonrigid registration. *IEEE Trans. Med. Imag.*, 23(9):1065–1076, 2004.

[13] T. Rohlfing, C. Maurer, D. Bluemke, and M. Jacobs. Volume-preserving nonrigid registration of MR breast images using free-form deformation with an incompressibility constraint. *IEEE Trans. Med. Imag.*, 22(6):730–741, 2003.

[14] W. Rossmann. *Lie groups: an introduction through linear groups*. Oxford University Press, USA, 2002.

[15] D. Rueckert, P. Aljabar, R. Heckemann, J. Hajnal, and A. Hammers. Diffeomorphic registration using B-splines. In *Proc. MICCAI'06*, volume 4191, pages 702–709, 2006.

[16] D. Rueckert, A. F. Frangi, and J. A. Schnabel. Automatic construction of 3-D statistical deformation models of the brain using nonrigid registration. *IEEE Trans. Med. Imag.*, 22(8):1014–25, 2003.

[17] 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 Trans. Med. Imag.*, 18:712–721, 1999.

[18] M. Sdika. A fast nonrigid image registration with constraints on the Jacobian using large scale constrained optimization. *IEEE Trans. Med. Imag.*, 27(2):271–281, 2008.

[19] C. Studholme, D. Hill, and D. Hawkes. An overlap invariant entropy measure of 3D medical image alignment. *Pattern Recognition*, 32(1):71–86, 1999.

[20] R. Tao, P. Fletcher, S. Gerber, and R. Whitaker. A variational image-based approach to the correction of susceptibility artifacts in the alignment of diffusion weighted and structural MRI. In *Proc. IPMI'09*, 2009.

[21] T. Vercauteren, X. Pennec, A. Perchant, and N. Ayache. Diffeomorphic demons: Efficient non-parametric image registration. *Neuroimage*, 45(1S1):61–72, 2009.