A Log-Euclidean Polyaffine Framework for Locally Rigid or Affine Registration

Vincent Arsigny[1], Olivier Commowick[1,2],
Xavier Pennec[1], and Nicholas Ayache[1]

[1] INRIA Sophia - Epidaure Project, 2004 Route des Lucioles BP 93
06902 Sophia Antipolis Cedex, France
Vincent.Arsigny@Polytechnique.org
[2] DOSISoft S.A., 45 Avenue Carnot, 94 230 Cachan, France

Abstract. In this article, we focus on the parameterization of non-rigid geometrical deformations with a small number of flexible degrees of freedom . In previous work, we proposed a general framework called *polyaffine* to parameterize deformations with a small number of rigid or affine components, while guaranteeing the invertibility of global deformations. However, this framework lacks some important properties: the inverse of a polyaffine transformation is not polyaffine in general, and the polyaffine fusion of affine components is not invariant with respect to a change of coordinate system. We present here a novel general framework, called *Log-Euclidean polyaffine*, which overcomes these defects. We also detail a simple algorithm, the *Fast Polyaffine Transform*, which allows to compute very efficiently Log-Euclidean polyaffine transformations and their inverses on a regular grid. The results presented here on real 3D locally affine registration suggest that our novel framework provides a general and efficient way of fusing local rigid or affine deformations into a global invertible transformation without introducing artifacts, independently of the way local deformations are first estimated.

1 Introduction

The registration of medical images is in general a difficult problem, and numerous methods and tools have been already devised to address this task [9]. Still currently, much effort continues to be devoted to finding adequate measures of similarity, relevant parameterizations of geometrical deformations, efficient optimization methods, or realistic mechanical models of deformations, depending on the precise type of registration considered.

In this article, we focus on the parameterization of non-rigid geometrical deformations with a small number of flexible degrees of freedom . This type of parameterization is particularly well-adapted for example to the registration of articulated structures [11] and to the registration of histological slices [12, 3]. After a *global* affine (or rigid) alignment, this sort of parameterization also allows a finer *local* registration with *very smooth* transformations [5, 10, 6, 13].

In [3], we parameterized deformations with a small number of *rigid or affine components*, which can model smoothly a large variety of local deformations.

J.P.W. Pluim, B. Likar, and F.A. Gerritsen (Eds.): WBIR 2006, LNCS 4057, pp. 120–127, 2006.

We provided a general framework to fuse these components into a global transformation, called *polyrigid* or *polyaffine*, whose *invertibility* is guaranteed. However, this framework lacks some important properties: the inverse of a polyaffine transformation is not polyaffine in general, and the polyaffine fusion of affine components is not invariant with respect to a change of coordinate system (i.e. is not *affine-invariant*). Here, we present a novel general framework to fuse rigid or affine components, called *Log-Euclidean polyaffine*, which overcomes these defects and yields transformations which can be very efficiently computed.

The sequel of this article is organized as follows. In Section 2, we present the Log-Euclidean polyaffine framework and its intuitive properties. Then, we present the *Fast Polyaffine Transform* (FPT), which allows to compute very efficiently Log-Euclidean polyaffine transformations (LEPTs) and their inverses on a regular grid. Finally, we apply the FPT to a real 3D example, where affine components are estimated with the algorithm of [5]. *Without introducing artifacts*, our novel fusion ensures the invertibility of the global transformation.

2 A Log-Euclidean Polyaffine Framework

Before presenting our novel polyaffine framework, let us briefly recall our framework for locally affine registration and the original polyaffine framework, described in [3].

Locally Affine or Rigid Transformations. Following the seminal work of [8], we parameterize locally affine (or rigid) transformations by N affine (or rigid) *components*. Each component i consists of an affine transformation $T_i = (M_i, t_i)$ (M_i and t_i are the linear and translation parts) and of a non-negative *weight function* $w_i(x)$, such that the influence of the i^{th} component at point x is proportional to $w_i(x)$. Here, we assume that the weights are *normalized*: that for all x, $\sum_{i=1}^{N} w_i(x) = 1$.

Direct Fusion of Components. To obtain a global transformation from several components, the classical approach [14], called here *direct fusion*, simply consists in averaging the associated displacements according to the weights:

$$T(x) = \sum_{i=1}^{N} w_i(x) T_i(x). \tag{1}$$

The transformation obtained using (1) is smooth, but although each component is invertible, the resulting global transformation is *not invertible* in general.

Previous Polyaffine Framework. We proposed in [3] to average displacements *infinitesimally*. The resulting global transformation is obtained by integrating an Ordinary Differential Equation (ODE), called polyaffine, which is computationally more expensive, but guarantees the invertibility of global deformations. To define a polyaffine ODE, this approach relies on *principal logarithms* of the *linear parts* M_i of the transformations T_i. However, as mentioned in the introduction, this framework lacks some important and desirable properties.

Logarithm of an Rigid or Affine Transformation. The key idea of our novel approach is to use the logarithms of the transformations *themselves*. In 3D, the logarithm of an affine (or rigid) transformation T is given in homogeneous coordinates by a 4x4 matrix which is simply the matrix logarithm of the 4x4 matrix representing T [1]:

$$\log(T) = \log \begin{pmatrix} M & t \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} L & v \\ 0 & 0 \end{pmatrix}.$$

This logarithm is well-defined if and only if none of the eigenvalues of M are non-positive real numbers. See [4] for more details and an efficient numerical algorithm to compute matrix logarithms. Intuitively, this constraint only excludes affine transformations *very far from the identity*, which we did not observe at all in our registration experiments. In particular, for rigid components, this only imposes that (local) rotations be stricly below π radians. For a discussion of this limitation, see [1]. In the following, we assume that the logarithms of our affine transformations are well-defined.

Log-Euclidean Polyaffine Transformations. Let (T_i) be N affine (or rigid) transformations, and let $(\log(T_i))$ be their logarithms. Using these logarithms, one can fuse the T_i infinitesimally according to the weights $w_i(x)$ with a *stationary* (or *autonomous*) ODE. In homogeneous coordinates, this ODE is the following:

$$\dot{x} = V(x) \stackrel{def}{=} \sum_i w_i(x) \log(T_i).x. \tag{2}$$

The solutions of (2) are always well-defined for all time. The proof is extremely similar to that given in [3]. The value at a point x of the Log-Euclidean polyaffine transformation (LEPT) defined by (2) is given by integrating (2) between time 0 and 1 with x as initial condition. This novel framework is called *Log-Euclidean*, because when the weights $w_i(x)$ do not depend on x, the resulting LEPT is simply the affine (or rigid) transformation equal to $\exp(\sum_i w_i \log(T_i))$, i.e. the Log-Euclidean mean of the components, similarly as in our work on tensors [2].

Remarkable Properties. The stationarity of (2) yields particularly nice and intuitive properties, conveniently expressed in terms of *flow*. At an instant s, the flow $T(s,.)$ of (2) is the mapping which gives the way the ambient space is deformed by the integration of (2) during s units of time. It is always invertible and smooth (as well as its inverse) [16], i.e. it is a *diffeomorphism*.

A classical property of the flow is the following: it is a *one-parameter subgroup* of diffeomorphisms, i.e. $T(s,.) \circ T(t,.) = T(s+t,.)$. Here, it is also a one-parameter subgroup of *LEPTs*. This means that $T(s,.)$ is the s^{th} power of the Log-Euclidean polyaffine transformation defined by $T(1,.)$. In particular, we have the intuitive properties that the *inverse* of $T(1,.)$ (resp. its square root) is simply $T(-1,.)$ (resp. $T(1/2,.)$), i.e. the LEPT with *identical* weights but whose affine transformations are the *inverses* (resp. square roots) of the original ones. Last but not least, (2) is *affine-invariant*: the Log-Euclidean polyaffine fusion does not depend on the current coordinate system. For more details, see [1].

3 Fast Polyaffine Transform

The remarkable (and novel) properties of the Log-Euclidean polyaffine framework allow fast computations of LEPTs. We propose here a very efficient algorithm to evaluate a Log-Euclidean polyaffine transformation on a regular grid.

Method Overview. Surprisingly, our fast algorithm generalizes a method widely used to compute *matrix exponentials* to the non-linear case. The basic idea of this method, called 'Scaling and Squaring', is that for a square matrix M, we have: $\exp(M) = \exp(\frac{M}{2^N})^{2^N}$. Since the matrix exponential is much simpler to compute for matrices *close to zero*, for example using Padé approximants, one can compute very accurately $\exp\left(\frac{M}{2^N}\right)$ and obtain $\exp(M)$ by squaring recursively N times the result [7]. In the non-linear case, since the flow $T(s,.)$ of (2) is a one-parameter subgroup, we also have:

$$T(1,.) = T\left(\frac{1}{2^N},.\right)^{2^N}, \tag{3}$$

which means that what the deformation observed at time 1 (i.e., the LEPT) results of 2^N times the composition of the small deformations observed at time $\frac{1}{2^N}$. Therefore, one can generalize the 'Scaling and Squaring' method to LEPTs in a straightforward way. This method, called the 'Fast Polyaffine Transform' (FPT), follows the same three steps as in the matrix case:

1. **Scaling step:** divide $V(x)$ (speed vectors of (2)) by a factor 2^N, so that $V(x)/2^N$ is close enough to zero (according to the level of accuracy desired).
2. **Exponentiation step:** $T\left(\frac{1}{2^N},.\right)$ is computed with a numerical scheme.
3. **Squaring step:** using (3), N recursive squarings of $T\left(\frac{1}{2^N},.\right)$ yield an accurate estimation of $T(1,.)$ (only N *compositions of mappings* are used).

Numerical Scheme for the Exponentiation Step. Integrating an ODE during a very short interval of time (short with respect to the smoothness of the solution) is quite simple. Generalizing ideas of [3], we use in this article a second-order scheme, called the *affine exponentiation scheme* (A.S.), which is *exact* in the case of a single affine component. It writes in homogeneous coordinates:

$$T\left(\frac{1}{2^N},x\right)_{\text{A.S.}} \overset{def}{=} \sum_{i=1}^{N} w_i(x).\exp\left(\frac{1}{2^N}\log\left(T_i\right)\right).x.$$

This choice of scheme comes from our numerical experiments [1] which show that this numerical scheme is on average approximately 40% more accurate than the first-order explicit scheme, with a similar simplicity and computational cost.

Computational Cost. An integration of (2) between times 0 and 1 with a time-step of 2^{-N} is performed in only N steps, and not in 2^N steps as with

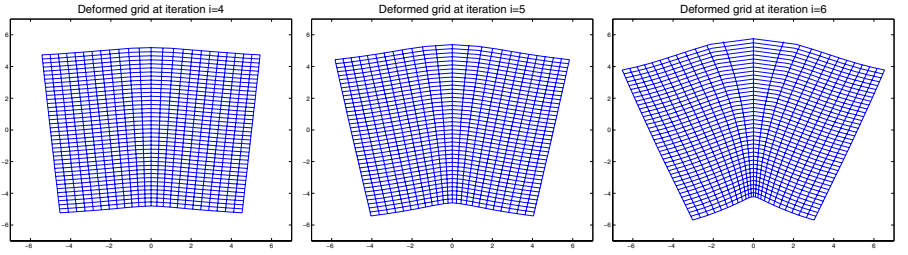

Fig. 1. Fast polyaffine transform for two rotations: three last squaring steps. 6 squarings are used in this experiment. Note how the deformation is initially very small, and increases exponentially. The relative accuracy of the estimation of the polyaffine transformation is on average of 0.21%, and the maximal error is below 3.2%.

methods using fixed time-steps. This is somehow comparable with the computational gain obtained by using the Fast Fourier Transform. Interestingly, after the exponentiation step, only N compositions between transformations are needed, which is an operation based on interpolation techniques. In this work, we use bi- and tri-linear interpolations, which are simple and guarantee a continuous interpolation of our transformation.

Synthetic Experiments. We measure the accuracy of our results by computing the relative difference with respect to an accurate estimation of the continuous transformation, obtained by a classical integration (i.e., with fixed time step, here 2^{-8}) of (2) for each voxel of the grid, which has 50×40 pixels. Numerical errors at the boundary of the regular grid are drastically reduced here by adding extra points to the grid so that it contains the boundary of the original grid deformed by direct fusion.

Fig. 1 displays the last 3 squaring steps of a typical FPT, using two rotations of opposite angles of 0.63 radians, (normalized) Gaussian weights ($\sigma = 5$) and a scaling of 2^6 (i.e., 6 squarings). Errors are low: the relative accuracy of the resulting estimation of the polyaffine transformation is on average 0.21% (instead of approximately 0.6% without an enlarged grid), and the maximal relative error is below 3.2% (instead of 11% wihtout an enlarged grid).

Inversion with the FPT. The inverse of a LEPT is simply (and intuitively) the LEPT with the same weights and with inverted affine transformations. Therefore, it can also be computed using the FPT. The accuracy of the inversion is evaluated via the composition of the estimation of the original LEPT and of its inverse by FPT, which should be close to the identity. Fig. 2 shows the evolution of this accuracy when the number of squarings varies, in our example of fusion between two rotations. We thus see that an excellent quality of inversion can be achieved using a small number of squarings, typically 7. The maximal relative error converges below 2% and the mean relative error is of the order of 0.2%. Similar results were obtained in [1] on other examples.

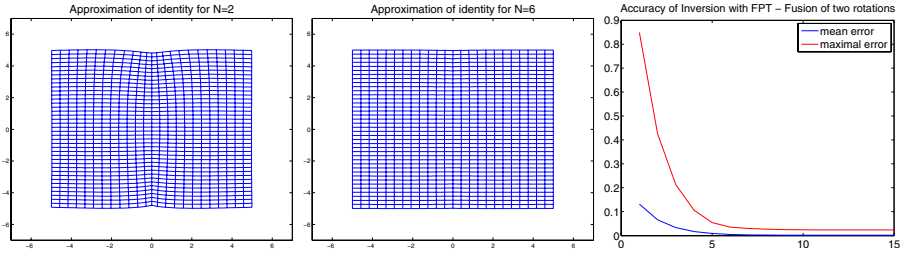

Fig. 2. Inversion with the FPT. From left to right: the regular grid deformed by the composition between the FPT of the LEPT and the FPT of its inverse, first with 2 squarings then 6 squarings. **On the right:** evolution of the relative accuracy when the number of squarings varies. An excellent accuracy is already achieved with 6 squarings, and the mean and maximal relative errors converge toward 0.2%and 2%.

4 Application to Locally Affine 3D Registration

Let us now consider a real 3D example of locally affine registration, between an atlas of $216 \times 180 \times 180$ voxels and a T_1-weighted MR image, with the multi-resolution and robust block-matching algorithm described in [5], without regularization. 7 structures of interest are considered: eyes (1 affine component each), cerebellum (2 components), brain stem (2 components), optic chiasm (1 component), 1 supplementary component (set to the identity) elsewhere. Weight functions are defined in the atlas geometry using mathematical morphology and a smoothing kernel in a preliminary step [5].

LEPTs as a Post-Processing Tool. To obtain short computation times (typically 10 minutes), our locally affine registration algorithm estimates affine components using the *direct fusion*. The FPT is used in a *final step* to ensure

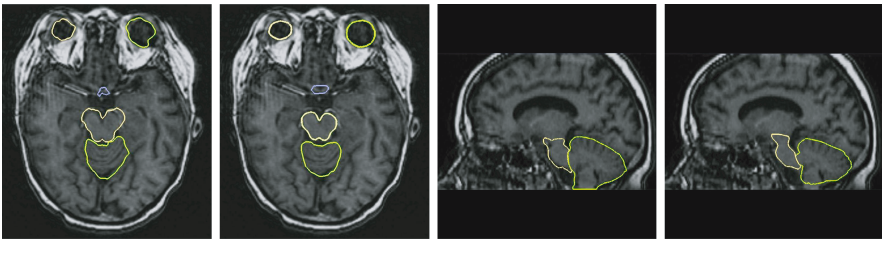

Fig. 3. Locally affine vs. dense-transformation: smoothness of deformations. The contours of our structures of interest (eyes, brain stem, cerebellum, optic chiasm) are displayed on the subject and are obtained by deforming those of the atlas using the dense transformation of [15] and using our locally affine framework. **From left to right**: axial slice, with first dense and then locally affine deformations; sagittal slice, with again dense and then locally affine deformations. Note how smoother contours are in the locally affine case, although both accuracies are comparable.

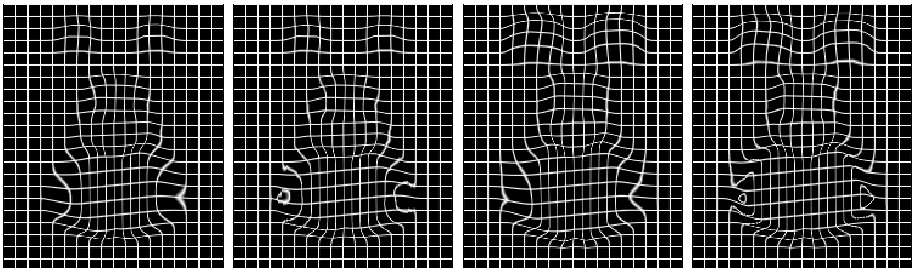

Fig. 4. Singularity removal with LEPTs. A 3D regular grid is deformed with the locally affine transformation obtained with the algorithm of [5], two slices are displayed. **From left to right:** polyaffine fusion and direct fusion (for the first slice) and then again polyaffine fusion and direct fusion (second slice). Note how the singularities of the direct fusion disappear with LEPTs. Remarkably, this is obtained without introducing any artifacts: outside singularities, both fusions yield very close results.

the invertibility of the final transformation, as well as to compute its inverse. A typical result of this registration procedure is illustrated by Fig. 3, which shows that the locally affine registration, with much smoother deformations, has an accuracy in the structures of interest which is comparable to the dense transformation case of [15].

Here, the scaling used in 2^8 and the FPT is computed in $40s$ on a Pentium4 Xeon™2.8 GHz on a $216 \times 180 \times 180$ regular grid. As shown by Fig. 4, the direct fusion of components estimated by [5] can lead to singularities, which is not the case when the FPT is used. Remarkably, both fusions are very close *outside* of regions with singularities. This means that no artifacts are introduced by the FPT, which justifies *a posteriori* the estimation of affine components with the (faster) direct fusion.

5 Conclusion and Perspectives

In this work, we have presented a novel framework to fuse rigid or affine components into a global transformation, called *Log-Euclidean polyaffine*. Similarly to the previous polyaffine framework of [3], it guarantees the *invertibility* of the result. However, contrary to the previous framework, this is achieved with very intuitive properties: for example the inverse of a LEPT is a LEPT with identical weights and inverted affine components. Moreover, this novel fusion is *affine-invariant*, i.e. does not depend on the choice of coordinate system. We have also shown that remarkably, and contrary to previous polyaffine transformations, the specific properties of LEPTs allow their fast computations on regular grids, with an algorithm called the 'Fast Polyaffine Transform', whose efficiency is somehow comparable to that of the Fast Fourier Transform.

In the example of locally affine 3D registration presented here, we use LEPTs in a final step to fuse the affine components estimated during the algorithm of

[5]. With the FPT, this is done very efficiently. Remarkably, the novel fusion is *very close* to the direct fusion in regions without singularities. This suggests that our novel framework provides a general and efficient way of fusing local rigid or affine deformations into a global invertible transformation without introducing artifacts, *independently* of the way local affine deformations are first estimated.

References

1. V. Arsigny, O. Commowick, X. Pennec, and N. Ayache. A fast and Log-Euclidean polyaffine framework for locally affine registration. Research report RR-5865, INRIA, March 2006.
2. V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Fast and simple calculus on tensors in the Log-Euclidean framework. In *MICCAI (1)*, pages 115–122, 2005.
3. V. Arsigny, X. Pennec, and N. Ayache. Polyrigid and polyaffine transformations: a novel geometrical tool to deal with non-rigid deformations - application to the registration of histological slices. *Med. Im. Anal.*, 9(6):507–523, December 2005.
4. S. Hun Cheng, N. J. Higham, C. S. Kenney, and A. J. Laub. Approximating the logarithm of a matrix to specified accuracy. *SIAM J. Matrix Anal. Appl.*, 22(4):1112–1125, 2001.
5. O. Commowick, V. Arsigny, J. Costa, G. Malandain, and N. Ayache. An efficient multi-affine framework for the registration of anatomical structures. In *Proceedings of ISBI'2006*. IEEE, 2006. To appear.
6. A. Cuzol, P. Hellier, and E. Mémin. A novel parametric method for non-rigid image registration. In *Proc. of IPMI'05*, number 3565 in LNCS, pages 456–467, 2005.
7. N. J. Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM J. Matrix Anal. Appl.*, 26(4):1179–1193, 2005.
8. J.A. Little, D.L.G. Hill, and D.J. Hawkes. Deformations incorpotationg rigid structures. *CVIU*, 66(2):223–232, May 1996.
9. J.B.A. Maintz and M.A Viergever. A survey of medical registration. *Medical image analysis*, 2(1):1–36, 1998.
10. R. Narayanan, J.A. Fessler, H. Park, and C.R. Meyer. Diffeomorphic nonlinear transformations: A local parametric approach for image registration. In *Proceedings of IPMI'05*, volume 3565 of LNCS, pages 174–185, 2005.
11. X. Papademetris, D.P. Dione, L.W. Dobrucki, L.H. Staib, and A.J. Sinusas. Articulated rigid registration for serial lower-limb mouse imaging. In *MICCAI'05 (2)*, pages 919–926, 2005.
12. A. Pitiot, E. Bardinet, P.M. Thompson, and G. Malandain. Piecewise affine registration of biological images for volume reconstruction. *Med. Im. Anal.*, 2005. accepted for publication.
13. D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes. Non-rigid registration using free-form deformations: Application to breast MR images. *IEEE Trans. Medecal Imaging*, 18(8):712–721, 1999.
14. D. Sheppard. A two-dimensionnal interpolation function for irregularly spaced data. In *23rd National Conference of the ACM*, pages 517–524, 1968.
15. R. Stefanescu, X. Pennec, and N. Ayache. Grid powered nonlinear image registration with locally adaptive regularization. *Med. Im. Anal.*, 8(3):325–342, September 2004.
16. M. Tenenbaum and H. Pollard. *Ordinary Differential Equations*. Dover, 1985.