Elastic Registration of 3D Deformable Objects

Zsolt Sánta and Zoltan Kato
Image Processing and Computer Graphics Dept., University of Szeged
H-6701 Szeged, PO. Box 652., Hungary, Fax: +36 62 546-397
Email: {santazs, kato}@inf.u-szeged.hu

***Abstract*—A novel correspondence-less approach is proposed to find a non-linear aligning transformation between a pair of deformable 3D objects. Herein, we consider a polynomial deformation model, but our framework can be easily adapted to other common deformations. The basic idea of the proposed method is to set up a system of nonlinear equations whose solution directly provides the parameters of the aligning transformation. Each equation is generated by integrating a nonlinear function over the object's domains. Thus the number of equations is determined by the number of adopted nonlinear functions yielding a flexible mechanism to generate sufficiently many equations. While classical approaches would establish correspondences between the shapes, our method works without landmarks. The efficiency of the proposed approach has been demonstrated on a large synthetic dataset as well as in the context of medical image registration.**

I. Introduction

Registration of deformable objects [1], [2] has many application areas such as 3D laser scan registration [3] or volumetric medical image alignment [4]. When registering a pair of objects, first we have to characterize the possible deformations. From this point of view, registration techniques can be classified into two main categories: physical model-based and parametric or functional representation [5]. Herein, we deal with the latter representation, which typically originate from interpolation and approximation theory. A broad class of such deformations are polynomial. Unlike spline-based transformations *e.g.* thin plate splines [6], [7], which are typically based on interpolation, polynomial models are approximating the underlying deformation. Furthermore, the control points of interpolating thin plate spline models are placed at extracted point matches and they usually include various regularizations, such as the bending energy [6]. On the other hand, polynomial deformations are governed by fewer parameters and are acting *globally* on the shapes, hence regularization is not needed. Moreover, many non-polynomial transformation can be approximated by a polynomial one *e.g.* via a Taylor expansion [8].

From an algorithmic viewpoint, registration methods can be divided into two main categories: *landmark-based* and *area-based* (or *feature-less*) approaches. Landmark-based methods are challenged by the correspondence problem, which is particularly difficult to solve in the case of non-linear deformations. Area-based approaches are typically relying on the availability of rich radiometric information which is used to construct a similarity measure based on some kind of intensity correlations. The aligning transformation is then found by maximizing similarity between the objects, which usually yields a complex non-linear optimization procedure. In many cases, reliable radiometric information may not be available (*e.g.* range data or multimodal medical images), therefore purely shape-based methods are particularly interesting [1], [2], [3], [4], [9].

Most of these approaches are focusing on point set registration. In [9], a probabilistic model is proposed where a Gaussian mixture with centroids corresponding to the first set is fit to the second point set by maximizing the likelihood. In [1], both point sets are represented as a Gaussian mixture model and then the L2 distance of the two mixtures is minimized. While point-based registration has the advantage of tolerating rather high occlusions, they are typically inefficient for large point sets. In many applications, however, the goal is the accurate alignment of whole volumetric objects of several megavoxels. Therefore object level registration methods are needed *e.g.* in medical imaging.

While elastic registration of planar shapes has been addressed by many researchers [10], [8], the extension of 2D methods for 3D object registration is far from trivial. Herein, we propose a correspondence-less approach for aligning 3D deformable objects subject to a polynomial deformation. The basic idea is to construct a system of nonlinear equations by integrating a set of nonlinear functions over the object volumes and then solve it by classical *Levenberg-Marquardt* algorithm. We have conducted a series of quantitative tests on a synthetic dataset to demonstrate the performance and robustness of the proposed approach. The method has been successfully applied to the alignment of segmented lung CT images.

II. Polynomial registration framework

A broadly used class of deformations is the polynomial family. In the three dimensional case, the deformation field $\pi : \mathbb{R}^3 \to \mathbb{R}^3$, $\pi(\mathbf{x}) = [\pi_1(\mathbf{x}), \pi_2(\mathbf{x}), \pi_3(\mathbf{x})]$ is given by three polynomial functions $\pi_i : \mathbb{R}^3 \to \mathbb{R}$. Without loss of generality, we can assume that $d = deg(\pi_1) = deg(\pi_2) = deg(\pi_3)$:

$$\pi_1(\mathbf{x}) = \sum_{i=0}^{d} \sum_{j=0}^{d-i} \sum_{k=0}^{d-i-j} a_{ijk} x_1^i x_2^j x_3^k$$

$$\pi_2(\mathbf{x}) = \sum_{i=0}^{d} \sum_{j=0}^{d-i} \sum_{k=0}^{d-i-j} b_{ijk} x_1^i x_2^j x_3^k$$

$$\pi_3(\mathbf{x}) = \sum_{i=0}^{d} \sum_{j=0}^{d-i} \sum_{k=0}^{d-i-j} c_{ijk} x_1^i x_2^j x_3^k.$$

The transformation has a total of $k = (d+3)(d+2)(d+1)/2$ parameters. Denoting the coordinates of the *template* and the *observation* by $\mathbf{x} = [x_1, x_2, x_3]^T \in \mathbb{R}^3$ and $\mathbf{y} = [y_1, y_2, y_3]^T \in \mathbb{R}^3$, respectively, the following relation holds between points of the two objects:

$$\pi(\mathbf{x}) = \mathbf{y} \quad \Leftrightarrow \quad \mathbf{x} = \pi^{-1}(\mathbf{y}). \tag{1}$$

Since individual point correspondences are not available, let us integrate out individual point matches of (1) over the foreground regions $\mathcal{F}_t$ and $\mathcal{F}_o$ of the *template* and the *observation*:

$$\int_{\mathcal{F}_o} \mathbf{y} d\mathbf{y} = \int_{\mathcal{F}_t} \pi(\mathbf{x}) |J_\pi(\mathbf{x})| d\mathbf{x}, \tag{2}$$

where $|J_\pi(\mathbf{x})| : \mathbb{R}^3 \to \mathbb{R}$ is the Jacobian determinant of the transformation composed of the following partial derivatives:

$$\frac{\partial \pi_1}{\partial x_1} = \sum_{i=1}^{d} \sum_{j=0}^{d-i} \sum_{k=0}^{d-i-j} i a_{ijk} x_1^{i-1} x_2^j x_3^k \tag{3}$$

$$\frac{\partial \pi_1}{\partial x_2} = \sum_{j=1}^{d} \sum_{i=0}^{d-j} \sum_{k=0}^{d-i-j} j a_{ijk} x_1^i x_2^{j-1} x_3^k \tag{4}$$

$$\frac{\partial \pi_1}{\partial x_3} = \sum_{k=1}^{d} \sum_{i=0}^{d-k} \sum_{j=0}^{d-i-k} k a_{ijk} x_1^i x_2^j x_3^{k-1}. \tag{5}$$

For π_2 and π_3, we get a similar formula. Note that (2) is a system of three equations. Unfortunately, in most cases the number of the parameters of π is much higher than three, therefore we need to generate more equations. For that purpose, observe that applying a properly chosen $\omega_i : \mathbb{R}^3 \to \mathbb{R}$ function to both sides of (1) yields the following form of (2):

$$\int_{\mathcal{F}_o} \omega_i(\mathbf{y}) d\mathbf{y} = \int_{\mathcal{F}_t} \omega_i(\pi(\mathbf{x})) |J_\pi(\mathbf{x})| d\mathbf{x}, \qquad i = 1, \ldots, \ell. \tag{6}$$

The basic idea of the proposed approach is to generate a sufficient number of independent equations by making use of a set of non-linear functions $\{\omega_i\}_{i=1}^{\ell}$. Since we need at least k equations and each ω_i generates one equation, $\ell \geq k$ must hold.

III. Numerical implementation

In practice, we only have a digital representation of the continuous objects $\mathcal{F}_t$ and $\mathcal{F}_o$, hence the equations of (6) are only approximately valid. In addition, objects extracted from images are subject to various segmentation errors, which again introduces errors in the equations. Therefore in practice, an overdetermined system is constructed, *i.e.* $\ell > k$, which is then solved in the least-squares sense via a *Levenberg-Marquardt* algorithm.

Since the system is solved by minimizing the algebraic error, proper normalization is critical for numerical stability. For that purpose, the *template* and *observation* coordinates are normalized into $[-0.5, 0.5]$ by applying normalizing transformations N_o and N_t, respectively:

$$N_o = \begin{pmatrix} o_1 & 0 & 0 & -o_1 O_1 \\ 0 & o_2 & 0 & -o_2 O_2 \\ 0 & 0 & o_3 & -o_3 O_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{7}$$

$$N_t = \begin{pmatrix} t_1 & 0 & 0 & -t_1 T_1 \\ 0 & t_2 & 0 & -t_2 T_2 \\ 0 & 0 & t_3 & -t_3 T_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{8}$$

where T_i, O_i $(i = 1, \ldots, 3)$ are the centroids of the template and the observation, respectively and t_i, o_i $(i = 1, \ldots, 3)$ are the scale factors. This will basically transform the input objects into a unit sphere centered at the origin.

Moreover the range of the ω_i functions should also be normalized into $[-1, 1]$ in order to ensure a balanced contribution of the equations to the algebraic error. This can be achieved by dividing the integrals with the maximal magnitude of the integral over the unit sphere containing the objects [8]:

$$N_i = \int_{\|\mathbf{x}\| \leq \frac{\sqrt{3}}{2}} |\omega_i(\mathbf{x})| d\mathbf{x} \tag{9}$$

The discretized form of (6) using the above normalizations is thus

$$\frac{|N_o|}{N_i} \sum_{\mathbf{y} \in F_o} \omega_i(\mathbf{y}) = \frac{|N_t|}{N_i} \sum_{\mathbf{x} \in F_t} \omega(\pi(\mathbf{x})) |J_\pi(\mathbf{x})| \tag{10}$$

where F_o and F_t are the set of the normalized voxel coordinates of the *observation* and *template* respectively; and $|N_o|$ and $|N_t|$ are the determinants of the corresponding normalizing linear transformations.

The resulting $\hat{\pi}^*$ transformation is acting between the normalized objects, hence we have to denormalize it. In our experiments, we register the template image to the observation:

$$\begin{array}{ccc} \mathcal{F}_t & \xrightarrow{\hat{\pi}} & \mathcal{F}_o \\ \downarrow & & \downarrow \\ N_t(\mathcal{F}_t) & \xrightarrow{\hat{\pi}^*} & N_o(\mathcal{F}_o) \end{array} \qquad \hat{\pi} = N_o^{-1} \circ \hat{\pi}^* \circ N_t \tag{11}$$

therefore to obtain $\hat{\pi}$ from $\hat{\pi}^*$, we have to use N_t and the inverse of N_o:

$$N_o^{-1} = \begin{pmatrix} 1/o_1 & 0 & 0 & O_1 \\ 0 & 1/o_2 & 0 & O_2 \\ 0 & 0 & 1/o_3 & O_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{12}$$

To denormalize the solution $\hat{\pi}^*$, we have to combine N_t with the polynomial base functions, multiply all parameters with the corresponding scale factor of N_o^{-1} and then add the corresponding translation terms of N_o^{-1} to the parameters of the zero order terms:

$$\hat{\pi}_1(\mathbf{x}) = O_1 + \frac{1}{o_1} \sum_{i=0}^{d} \sum_{j=0}^{d-i} \sum_{k=0}^{d-i-j} a^*_{ijk} ((x_1 - T_1)t_1)^i ((x_2 - T_2)t_2)^j ((x_3 - T_3)t_3)^k, \tag{13}$$

where a^*_{ijk} are the parameters of the normalized solution ($\hat{\pi}_2$ and $\hat{\pi}_3$ can be obtained similarly). Algorithm 1 shows the main steps of the proposed method. The computational complexity is

$$\mathcal{O}(\ell|F_o| + I|F_t|(\ell + k + 9J)), \tag{14}$$

where ℓ is the number of ω functions, k and $9J$ are the number of the required operations for computing the transformation function and the Jacobian determinant on each coordinate, respectively. I denotes the number of function calls executed by the *Levenberg-Marquardt* solver.

Algorithm 1 Pseudo code of the proposed algorithm

Input: *template* and *observation* voxels

Output: The transformation parameters of $\hat{\pi}$

1: Choose a set of $\ell > k$ nonlinear functions $\{\omega_i\}_{i=1}^{\ell}$, and for each ω_i compute the normalizing constant N_i using (9).
2: Compute the normalizing transformations N_o and N_t as in (7) and (8), which maps voxel coordinates into $[-0.5, 0.5]$.
3: Construct the system of equations (10).
4: Find a least-squares solution of the system using the *Levenberg-Marquardt* algorithm. Use the identity transformation for initialization.
5: Denormalize the solution $\hat{\pi}^*$ using (13), which provides the parameters of the aligning transformation.

A. Choice of the ω functions

Theoretically, any nonlinear ω_i function is applicable in (6) as long as it is integrable and rich enough. Herein, however, we will adopt simple power functions of the form $\omega_i(\mathbf{x}) = x_1^{n_i} x_2^{m_i} x_3^{o_i}$ as it makes (10) a polynomial equation, which is computationally more favorable. The system of (10) for $\ell > k$ thus becomes:

$$\frac{|N_o|}{N_i} \sum_{\mathbf{y} \in F_o} y_1^{m_i} y_2^{n_i} y_3^{o_i} = \frac{|N_t|}{N_i} \sum_{\mathbf{x} \in F_t} \pi_1(\mathbf{x})^{m_i} \pi_2(\mathbf{x})^{n_i} \pi_3(\mathbf{x})^{o_i} |J_\pi(\mathbf{x})| \tag{15}$$

The left hand side is clearly polynomial. On the right hand side, the product of the powered transformation functions are polynomials too and so as the Jacobian determinant of the transformation (5), hence their product will also be polynomial of degree $d_i = d(n_i + m_i + o_i) + 3(d-1)$:

$$\pi_1^{n_i} \pi_2^{m_i} \pi_3^{o_1} |J_\pi(\mathbf{x})| = \sum_{q=0}^{d_i} \sum_{r=0}^{d_i - q} \sum_{s=0}^{d_i - q - r} g_{iqrs} x_1^q x_2^r x_3^s, \tag{16}$$

where d is the degree of the transformation polynomial and, according to the Multinomial theorem [11], g_{iqrs} is also a polynomial of the parameters of the transformation. Furthermore, the polynomials g_{iqrs} do not contain image coordinates, hence they are independent from the objects. Substituting back into the right hand side of (15), we get

$$\frac{|N_t|}{N_i} \sum_{\mathbf{x} \in F_t} \pi_1^{n_i}(\mathbf{x}) \pi_2^{m_i}(\mathbf{x}) \pi_3^{o_i}(\mathbf{x}) |J_\pi(\mathbf{x})| = \frac{|N_t|}{N_i} \sum_{q=0}^{d_i} \sum_{r=0}^{d_i - q} \sum_{s=0}^{d_i - q - r} g_{iqrs} \sum_{\mathbf{x} \in F_t} x_1^q x_2^r x_3^s. \tag{17}$$

Note that the above two formulas are mathematically equivalent, but they have different computational complexities when equations are solved via an iterative least-squares method. Since the algebraic error has to be evaluated at each iteration, the above formulas have to be computed several times for each iteration step with the actual values of the unknowns. The formula on the left hand side contains fewer terms but, since these terms are within a sum over voxels $\mathbf{x} \in F_t$, the object voxels need to be visited each time the formula gets evaluated. On the other hand, the formula (17) contains much more terms, but the polynomials g_{iqrs} of the unknowns are separated from the sum over voxels $\mathbf{x} \in F_t$, therefore the latter sums can be precomputed and object voxels are not needed anymore for the solver. The computational complexity of the original formula is given by (14). Let us now have a closer look at the complexity of the formula (17).

Denote the number of terms in a three dimensional full polynomial by $\gamma : \mathbb{R} \to \mathbb{R}$, $\gamma(d) = (d+3)(d+2)(d+1)/6$, where d is the degree of the polynomial. Since the inner sum

$$\sum_{\mathbf{x} \in F_t} x_1^q x_2^r x_3^s \tag{18}$$

in (17) do not depend on the unknowns, they can be precomputed as sums of various powers of coordinates, which are simply various geometric moments of the objects. This can be done for all ω_i functions in $\mathcal{O}(|F_t|\gamma(d_M))$ time, where $d_M = \max\{d_i\}$, because the higher order polynomials already contain the lower orders.

Now, examine the g_{iqrs} polynomials. In the worst case, they are full polynomials of the transformation parameters with a degree of $n_i + m_i + o_i + 1$, determined by the degree of the power functions and the Jacobian determinant. The number of these terms for the i^{th} equation is $\gamma(d_i)$, therefore the total number of operations is $\mathcal{O}(\gamma(d_i)\gamma(n_i+m_i+o_i+1))$, assuming that one term of q_{iqrs} can be computed in constant time. The overall computation time of (17) is thus

$$\mathcal{O}(\ell|F_o| + \gamma(d_M)|F_t| + I \sum_{i=1}^{\ell} \gamma(d_i)\gamma(n_i + m_i + o_i + 1)) \tag{19}$$

where I is the number of function calls executed by the *Levenberg-Marquardt* solver.

Which one of the two formulas should be used to achieve optimal computational complexity? There is no generic answer to this question. Since the left hand side of the equations are always computed only once and ℓ is basically determined by the adopted deformation model, the choice between the two formulas for a particular transformation is determined by the

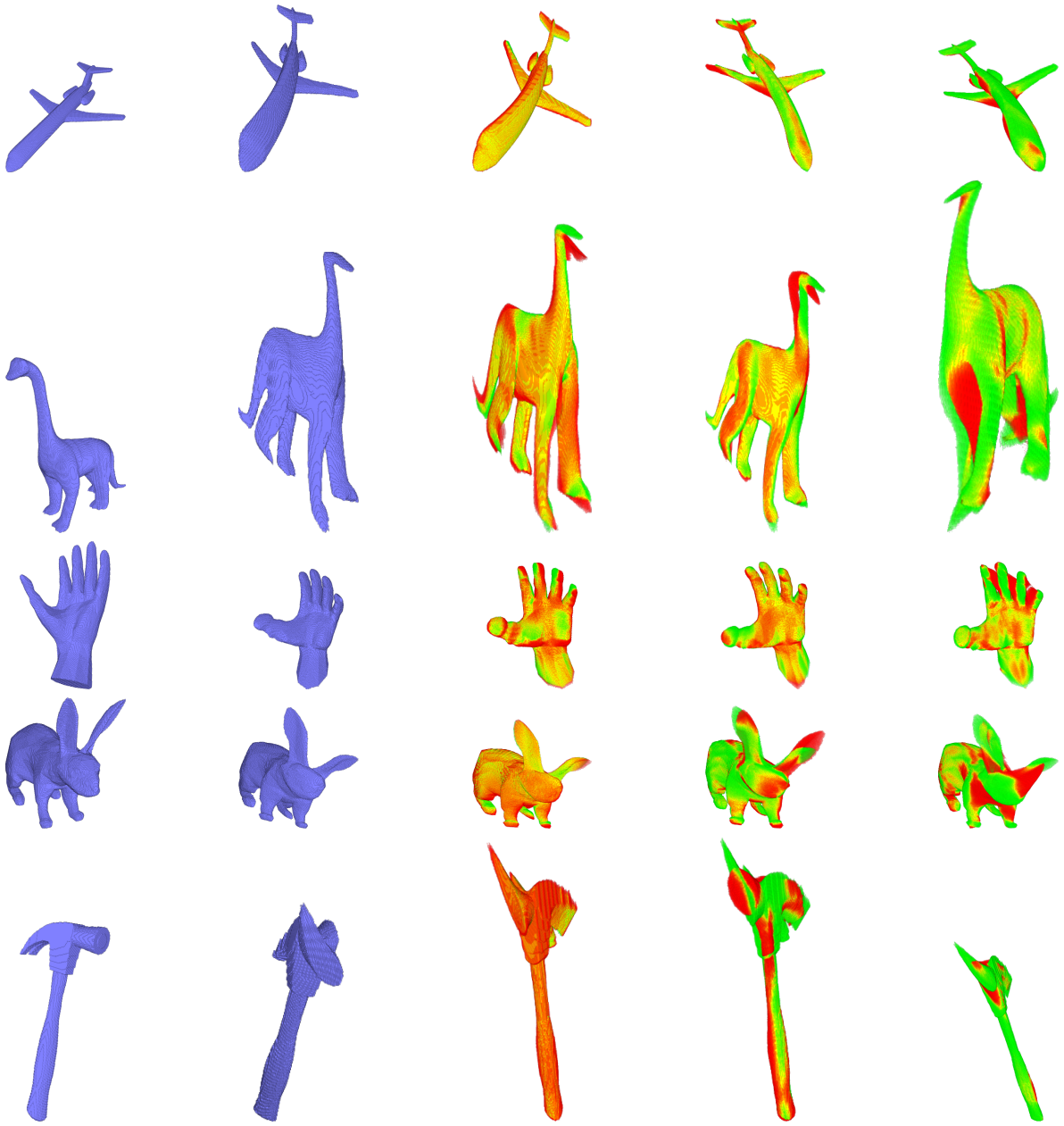

Fig. 1. Some results on the synthetic images. *Observation* and *registered* objects were overlayed, overlapping voxels are shown in yellow, while non-overlapping ones in red and green.

size of the *template* object $|F_t|$ and the number of function calls I.

As an example, we will now analyze the complexity of a third order polynomial deformation (*i.e.* $d = 3$) model used also in our experiments. To generate sufficiently many equations, we need at least $k = 60$ ω_i functions. Therefore let us choose power functions with maximal degree of 3:

$$\{(n_i, m_i, o_i)\}_{i=1}^{64} = \{(a, b, c) \mid 0 \leq a, b, c \leq 3\} \tag{20}$$

generating a total of 64 equations. Using the formulas above,

$$\begin{aligned} d_M &= 33 \\ \gamma(d_M) &= 7,140 \\ \sum_{i=1}^{64} \gamma(d_i)\gamma(n_i + m_i + o_i + 1) &= 16,453,488 \end{aligned}$$

In our experiments the average number of function calls was 380 for this deformation model and ω_i set. Note that the number of function calls is the same for both formulas, since they evaluate the same entity but in different ways. Hence, the

computational complexity of the formula (17) will be

$$\mathcal{O}(64|F_o| + 7,140|F_t| + 380 \cdot 16,453,488), \quad (21)$$

while the complexity of the original formula is

$$\mathcal{O}(64|F_o| + 380(64 + 60 + 90)|F_t|). \quad (22)$$

We thus conclude, that it is worth to use the formula (17) when

$$|F_t| \geq 84,286. \quad (23)$$

Runtime (min)			$\delta(\%)$		
m	μ	σ	m	μ	σ
13.96	6.85	17.84	7.02	6.06	4.58

TABLE I
RESULTS ON 550 SYNTHETIC IMAGES (M – MEDIAN, μ – MEAN AND σ – STANDARD DEVIATION).

IV. EXPERIMENTAL RESULTS

In our experiments, we used a third order polynomial deformation model (*i.e.* $d = 3$), which has a total of $k = (d+3)(d+2)(d+1)/2 = 60$ parameters. A system of 64 equations were generated by the set $\omega_i(\mathbf{x}) = x_1^{n_i} x_2^{m_i} x_3^{o_i}$, where $\{(n_i, m_i, o_i)\}_{i=1}^{64} = \{(a, b, c) \mid 0 \leq a, b, c \leq 3\}$.

The algorithm has been implemented in `C++` using the `levmar` library written by M.I.A. Lourakis [12]. All tests were ran under a Linux system running on a virtualized Core i5 3.1 GHz architecture. The registration error has been quantitatively evaluated based on the absolute difference of the aligned objects:

$$\delta = \frac{|\mathcal{F}_r \triangle \mathcal{F}_o|}{|\mathcal{F}_r| + |\mathcal{F}_o|} \cdot 100\%, \quad (24)$$

where $\mathcal{F}_o$ and $\mathcal{F}_r$ denote the set of foreground voxels of the *observation* and *registered* objects respectively.

In order to quantitatively evaluate the performance of the proposed method, a synthetic database of 500 deformed objects has been created. Observations have been generated by applying second and third order polynomial deformations to different *template* objects. The transformation parameters were randomly picked from the following intervals: $a_{11}, b_{21}, c_{31} \in [0.5; 1.5]$, $a_{21}, a_{31}, b_{11}, b_{31}, c_{11}, c_{21} \in [-0.25; 0.25]$ and all other parameters are from $[-0.5; 0.5]$. Note that $a_{00} = b_{00} = c_{00} = 0$ (*i.e.* no translations), because initial normalization would remove any larger translations. Some of our results are presented in Fig. 1, where *observation* and *registered* objects were overlayed, overlapping voxels are shown in yellow, while non-overlapping ones in red and green. The statistics of our test are described in Table I.

The registered objects, as well as synthetic observations, were generated by the following procedure (in Matlab): First, a smooth triangular mesh has been created from the object's surface using Matlab's internal `isosurface` function. Then the transformation was applied to the vertices yielding the transformed triangular mesh, from which the final voxelized object was obtained by the `binvox` program available from http://www.cs.princeton.edu/~min/binvox/ [13].

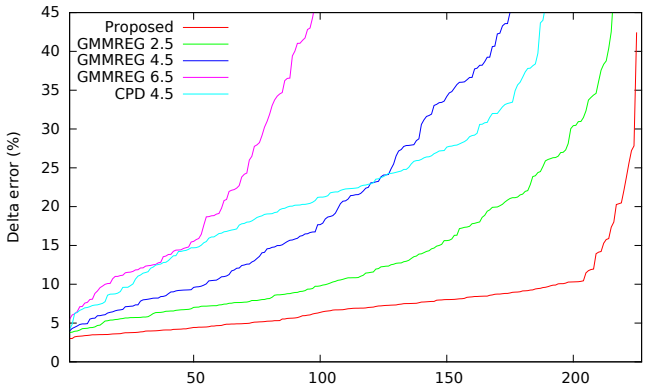

Fig. 2. Comparison with GMMREG [1] and CPD [9].

We have compared our results to the point-based registration frameworks in [1] (GMMREG) and [9] (CPD). We used the `C++` implementation of these methods available from http://code.google.com/p/gmmreg and set the parameters to their default values. At a first stage, we used all of the surface voxels as the input point set to the algorithm. For these sets consisting of about 0.5 megavoxel, the algorithms were running more than 12 hours without finding the transformation. Therefore the size of the point sets was reduced by using the vertices of an approximating triangular surface with various resolutions [14]. The mesh resolution was controlled by the maximal radius r of the corresponding Delaunay sphere. In Fig. 2, quantitative results for $r = \{2.5, 4.5, 6.5\}$ indicate that these methods provide inferior alignments.

In practice, segmentation never produces perfect shapes, therefore robustness against segmentation errors was also evaluated on simulated data: we randomly added or removed squares uniformly around the boundary of each slice of the observations yielding a surface error of 15%, 22% and 30% of the original object volume (see sample slices in Fig. 4). Fig. 3 shows the quantitative evaluation of the alignment error δ on more than 115 objects and the same test for the best GMMREG set (where $r = 2.5$). Considering that a $\delta < 10\%$ corresponds to a visually good alignment, our approach is quite robust up to as high as 22% surface error.

A. Medical application

Lung alignment is a crucial task in lung cancer diagnosis [15]. During the treatment, changes in the tumor size are determined by comparing *follow-up* PET/CT scans which are taken at regular intervals depending on the treatment and the size of the tumor. Due to respiratory motion, the lung is subject to a nonlinear deformation between such *follow-up*s, hence it is hard to automatically find correspondences. A common practice is to determine corresponding regions by hand, but this makes the procedure time consuming and

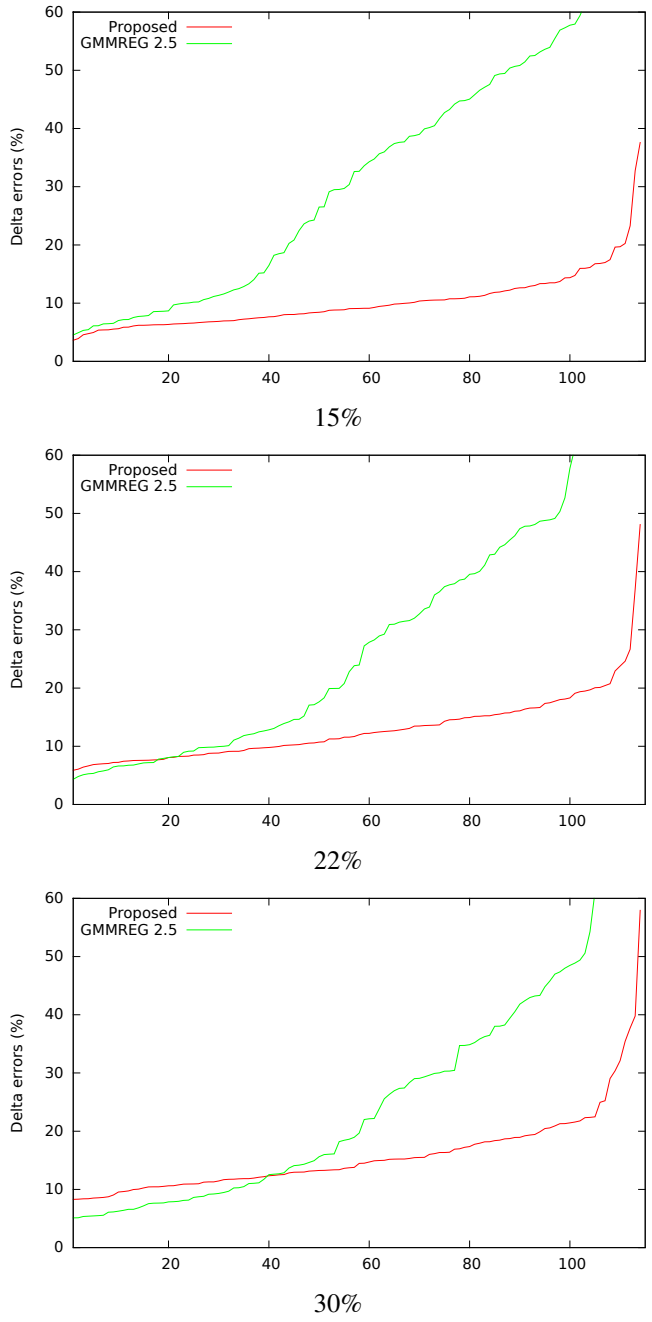

Fig. 3. Robustness test comparison with the best GMMREG set ($r = 2.5$).

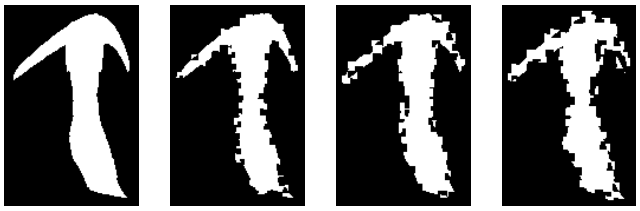

Fig. 4. Sample surface errors on a slice.

the obtained alignments may not be accurate enough for measuring changes.

Our algorithm has been successfully applied to align 3D lung CT scans. The polynomial model proved to be a good approximation of the underlying physical deformation. Promising results were obtained on the available 8 image pairs with a median δ error of 8.41% (the mean and standard deviation were 7.99% and 3.03%, respectively). Some of these results are presented in Fig. 5, where we also show the achieved alignment on grayscale slices of the original lung CT images. For these slices, the original and transformed images were combined as an 8×8 checkerboard pattern.

V. Conclusions

We have proposed a novel elastic registration method which works without established correspondences. The basic idea is to set-up a system of non-linear equations whose solution directly provides the parameters of the aligning transformation. Herein, we considered a polynomial deformation model, but other diffeomorphism can also be used by approximating it via a Taylor expansion. The computational complexity has been analyzed for two alternative forms of the equations and optimal choice between these computational schemes has also been discussed. The efficiency and robustness of the proposed approach have been demonstrated on a large synthetic dataset. Our method compares favorably to two recent 3D matching algorithms [1], [9]. Finally, the algorithm achieved promising results in aligning lung CT images.

Acknowledgment

This research was partially supported by the grant CNK80370 of the National Innovation Office (NIH) & the Hungarian Scientific Research Fund (OTKA); the European Union and co-financed by the European Regional Development Fund within the project TÁMOP-4.2.1/B-09/1/KONV-2010-0005. Lung images provided by *Mediso Ltd., Budapest, Hungary*.

References

[1] B. Jian and B. C. Vemuri, "Robust point set registration using Gaussian mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1633–1645, Aug. 2011. [Online]. Available: http://gmmreg.googlecode.com

[2] C. Papazov and D. Burschka, "Deformable 3D shape registration based on local similarity transforms," *Computer Graphics Forum*, vol. 30, no. 5, pp. 1493–1502, Aug. 2011.

[3] R. Sagawa, K. Akasaka, Y. Yagi, H. Hamer, and L. Van Gool, "Elastic convolved ICP for the registration of deformable objects," in *Proceedings of International Conference on Computer Vision*, IEEE. Kyoto, Japan: IEEE, Oct. 2009, pp. 1558–1565.

[4] F. Michel, M. M. Bronstein, A. M. Bronstein, and N. Paragios, "Boosted metric learning for 3D multi-modal deformable registration," in *Proceedings of International Symposium on Biomedical Imaging: From Nano to Macro*, IEEE. Chicago, Illinois, USA: IEEE, Mar. 2011, pp. 1209–1214.

[5] M. Holden, "A review of geometric transformations for nonrigid body registration," *IEEE Transactions on Medical Imaging*, vol. 27, no. 1, pp. 111–128, 2008.

[6] F. L. Bookstein, "Principal warps: Thin-Plate Splines and the Decomposition of deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 6, pp. 567–585, 1989.

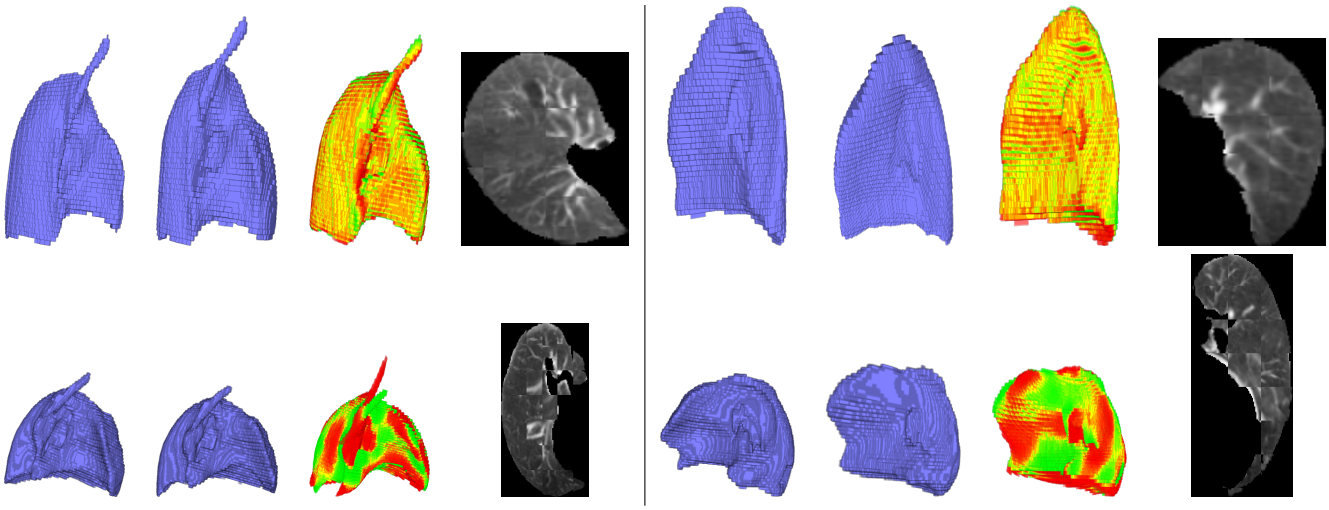

Fig. 5. Alignment of lung CT volumes. Segmented 3D lung images were generated by the *InterView Fusion* software of Mediso Ltd.

[7] L. Zagorchev and A. Goshtasby, "A comparative study of transformation functions for nonrigid image registration," *IEEE Transactions on Image Processing*, vol. 15, no. 3, pp. 529 –538, march 2006.

[8] C. Domokos, J. Nemeth, and Z. Kato, "Nonlinear Shape Registration without Correspondences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 943–958, 2012.

[9] A. Myronenko, X. B. Song, and M. Á. Carreira-Perpiñán, "Non-rigid point set registration: Coherent point drift," in *Proceedings of the Annual Conference on Neural Information Processing Systems*, B. Schölkopf, J. C. Platt, and T. Hoffman, Eds. Vancouver, British Columbia, Canada: MIT Press, Dec. 2006, pp. 1009–1016.

[10] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape context," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, April 2002.

[11] R. Merris, *Combinatorics*, ser. Wiley Series in Discrete Mathematics and Optimization. John Wiley, 2003. [Online]. Available: http://books.google.hu/books?id=OM3CP4i58b4C

[12] M. Lourakis, "levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++," [web page] http://www.ics.forth.gr/~lourakis/levmar/, Jul. 2004, [Accessed on 14 Jun. 2012.].

[13] F. S. Nooruddin and G. Turk, "Simplification and repair of polygonal models using volumetric techniques," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 2, pp. 191 – 205, Apr. 2003.

[14] Q. Fang and D. Boas, "Tetrahedral mesh generation from volumetric binary and gray-scale images," in *Proceedings of International Symposium on Biomedical Imaging: From Nano to Macro*, IEEE. Boston, Massachusetts, USA: IEEE, Jun. 2009, pp. 1142–1145.

[15] A. S. Bryant and R. J. Cerfolio, "The maximum standardized uptake values on integrated FDG-PET/CT is useful in differentiating benign from malignant pulmonary nodules," *The Annals of Thoracic Surgery*, vol. 82, pp. 1016–1020, 2006. [Online]. Available: http://ats.ctsnetjournals.org/cgi/content/abstract/82/3/1016