Correspondence-Less Non-Rigid Registration of Triangular Surface Meshes

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
{santazs,kato}@inf.u-szeged.hu

Abstract

A novel correspondence-less approach is proposed to find a thin plate spline map between a pair of deformable 3D objects represented by triangular surface meshes. The proposed method works without landmark extraction and feature correspondences. The aligning transformation is found simply by solving a system of nonlinear equations. Each equation is generated by integrating a nonlinear function over the object's domains. We derive recursive formulas for the efficient computation of these integrals. Based on a series of comparative tests on a large synthetic dataset, our triangular mesh-based algorithm outperforms state of the art methods both in terms of computing time and accuracy. The applicability of the proposed approach has been demonstrated on the registration of 3D lung CT volumes.

1. Introduction

A wide range of application areas, such as registration of 3D laser scans [22] or volumetric medical images [16], requires the alignment of deformable objects [11, 19]. 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 [10]. Herein, we deal with the latter representation, which typically originate from interpolation and approximation theory. A broadly used class of such parametric models are splines, in particular thin plate splines (TPS) [2, 24]. TPS models are quite useful whenever a parametric free-form registration is required but the underlying physical model of the object deformation is unknown or too complex. Furthermore, TPS models can be extended to include various regularizations, such as the bending energy) [2].

From a methodological point of view, we can differentiate *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 [11, 19, 22, 16, 17].

Most of these approaches are focusing on point set registration. In [17], 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 [11], 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 cases, however, the goal is the accurate alignment of whole volumetric objects of several megavoxels (*e.g*. medical applications). Therefore object level registration methods are needed.

Several other correspondence-less approach could be found, focusing on solving the related partial matching or shape recognition problems [14, 4, 3]. These approaches could handle large scope of deformations, but usually they do not use classical parametric transformation models.

While elastic registration of planar shapes has been addressed by many researchers [1, 8], the extension of 2D methods for 3D object registration is far from trivial. Herein, we propose a general TPS framework for aligning 3D deformable objects. The basic idea of the method is inspired by [8]: an overdetermined system of nonlinear equations is constructed by integrating a set of nonlinear functions over the object volumes, which is then solved in the least squares sense. However, the procedure we develop here to construct the equations is substantially different: in [8], the integrals are evaluated in terms of 2D pixels, which is relatively straightforward to extend to 3D vox-

els. Unfortunately, the computational complexity of such a naive extension is extremely high for practical applications. In this paper, we will show that complexity can be drastically reduced when the input objects are represented as triangular surface meshes. Triangular surface meshes are typically produced by *e.g.* stereo reconstruction methods [23, 12] but they can be easily extracted from volume images [21] as well. The performance and robustness of the proposed approach has been quantitatively evaluated on a synthetic dataset, and it has also been successfully applied to the alignment of segmented lung CT images.

2. Thin plate spline registration framework

Thin plate splines (TPS) [2, 24] are commonly used as parametric models for elastic deformations using radial basis functions. In the three dimensional space, a TPS transformation $\varphi : \mathbb{R}^3 \to \mathbb{R}^3$ can be decomposed as three coordinate functions $\varphi(\mathbf{x}) = [\varphi_1(\mathbf{x}), \varphi_2(\mathbf{x}), \varphi_3(\mathbf{x})]^T, \forall \varphi_i(\mathbf{x}) : \mathbb{R}^3 \to \mathbb{R}$. Given a set of control points $c_k \in \mathbb{R}^3$ and associated mapping coefficients $a_{ij}, w_{ki} \in \mathbb{R}$ with $i = 1, \ldots, 3, j = 1, \ldots, 4$ and $k = 1, \ldots, K$, the TPS functions are

$$\varphi_i(\mathbf{x}) = a_{i1}x_1 + a_{i2}x_2 + a_{i3}x_3 + a_{i4} + \sum_{k=1}^{K} w_{ki}\|c_k - \mathbf{x}\|. \tag{1}$$

Note that in 3D the radial basis functions the Euclidean norms $\|c_k - \mathbf{x}\|$ [10]. The number of parameters are $N = 3(K + 4)$ consisting of 12 affine parameters a_{ij} and 3 local coefficients ω_{ki} for each of the K control points c_k. The local parameters are also required to satisfy the following additional constraints [24], ensuring that the TPS at infinity behaves according to its affine term:

$$\sum_{k=1}^{K} w_{ki} = 0 \quad \text{and} \quad \sum_{k=1}^{K} c_{k_j} w_{ki} = 0, \quad i, j = 1, 2, 3. \tag{2}$$

When correspondences are available, the exact mapping of the control points are also known which, using (1), provides constraints on the unknown parameters. Thus in classical correspondence based approaches, control points are placed at extracted point matches, and the deformation at other positions is interpolated by the TPS. Therefore in such cases, a TPS can be regarded as an optimal *interpolating* function whose parameters are usually recovered via a complex optimization procedure [2, 24].

However, we are interested in solving the TPS registration problem without correspondences. Therefore, as in [8], a TPS is a parametric model to *approximate* the true deformation. Control points can be placed *e.g.* on a uniform grid in order to capture local deformations everywhere. Obviously, a finer grid would allow a more refined approximation of the deformation field at the price of an increased number of free parameters. Denoting the coordinates of the *template* volume $\mathcal{F}_t$ and *observation* $\mathcal{F}_o$ by $\mathbf{x} = [x_1, x_2, x_3]^T$ and $\mathbf{y} = [y_1, y_2, y_3]^T$, respectively, the points are related by φ:

$$\varphi(\mathbf{x}) = \mathbf{y} \tag{3}$$

Since individual point correspondences are not available, let us integrate both sides of (3) over the object volumes:

$$\int_{\mathcal{F}_o} \mathbf{y} d\mathbf{y} = \int_{\varphi(\mathcal{F}_t)} \mathbf{z} d\mathbf{z}, \tag{4}$$

where $\varphi(\mathcal{F}_t)$ corresponds to the domain of the transformed *template* image. When objects are represented as a set of foreground voxels, the generation of the transformed volume would be computationally expensive. In order to avoid the generation of $\varphi(\mathcal{F}_t)$, in the 2D case Domokos *et al.* [8] proposed to use integral transformation

$$\int_{\mathcal{F}_o} \mathbf{y} d\mathbf{y} = \int_{\mathcal{F}_t} \varphi(\mathbf{x}) |J_\varphi(\mathbf{x})| d\mathbf{x}, \tag{5}$$

which involves the Jacobian determinant of φ, composed of the partial derivatives of the transformation. While the naive extension of this idea to 3D is fairly straightforward, its complexity is still quite high due to the large number of voxels. In the next section, we will show that when 3D objects are represented as triangulated surface meshes, the computational complexity of $\varphi(\mathcal{F}_t)$ can be drastically reduced by computing the integrals in (4) via recursive formulas.

Of course, (4) provides three equations only (one for each coordinate), but any useful TPS model has much more than three parameters. To generate sufficiently many equations, we will extend the 2D solution [8] and apply a set of non-linear functions $\{\omega_i\}_{i=1}^{\ell}$ to both sides of (3) yielding a system of ℓ nonlinear equations:

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

Since each ω_i generates one equation and we have N unknowns, $\ell \geq N$ must hold. The least squares solution of the above system directly provides the parameters of the aligning TPS transformation.

3. Computing the integrals over meshes

In the following, we will propose an efficient computational scheme for the integrals in (6). For that purpose, let us denote the triangular surface meshes of the *template* and *observation* objects by $T_\triangle$ and $O_\triangle$ and the volumes enclosed by these meshes by $F_t^\triangle$ and $F_o^\triangle$, respectively.

First, let us observe that the transformation φ acting on the mesh $T_\triangle$ generates a new surface mesh by transforming the vertices of the triangles:

$$\varphi(T_\triangle) = \{(\varphi(A), \varphi(B), \varphi(C)) \mid (A, B, C) \in T_\triangle\} \quad (7)$$

The volume enclosed by this transformed surface mesh $\varphi(T_\triangle)$ is denoted by $\mathcal{D}_\varphi$. Since $\mathcal{F}_t \approx F_t^\triangle$ and $\mathcal{F}_o \approx F_o^\triangle$, the integrals from (4) can be approximated as

$$\int_{\mathcal{F}_o} \mathbf{y} d\mathbf{y} \approx \int_{F_o^\triangle} \mathbf{y} d\mathbf{y} \quad (8)$$

$$\int_{\varphi(\mathcal{F}_t)} \mathbf{z} d\mathbf{z} \approx \int_{\mathcal{D}_\varphi} \mathbf{x} d\mathbf{x}. \quad (9)$$

Assuming that every triangle is oriented consistently counter-clockwise when seen from the exterior of the object, the integrals over the volumes of (8) and (9) can be computed exactly as the sums of signed integrals over properly generated tetrahedrons:

$$\int_{F_o^\triangle} \mathbf{y} d\mathbf{y} = \sum_{o \in O_\triangle} \operatorname{sgn}(\operatorname{vol}(\mathcal{T}_o)) \int_{\mathcal{T}_o} \mathbf{y} d\mathbf{y} \quad (10)$$

$$\int_{\mathcal{D}_\varphi} \mathbf{z} d\mathbf{z} = \sum_{\pi \in \varphi(T_\triangle)} \operatorname{sgn}(\operatorname{vol}(\mathcal{T}_\pi)) \int_{\mathcal{T}_\pi} \mathbf{z} d\mathbf{z} \quad (11)$$

The key point here is the creation of the tetrahedrons: For

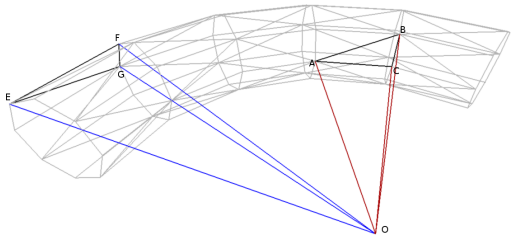

Figure 1. Tetrahedrons generated from a triangular mesh of a torus: The blue tetrahedron has positive signed volume and the red one has negative.

each triangle $o = (A, B, C)$, let us create a tetrahedron $\mathcal{T}_o = (A, B, C, 0)$ defined by the vertices of the corresponding triangle o and the origin 0. Thus for all triangle, we generate a tetrahedron with the fourth vertex shared by all of these tetrahedrons (see Fig. 1). Although the choice of the fourth vertex is arbitrary, setting it to the origin will greatly simplify our computations. The computation of the signed volume of such a tetrahedron $\mathcal{T}_o$ is straightforward:

$$\operatorname{vol}(\mathcal{T}_o) = \frac{1}{6} \begin{vmatrix} A_1 & B_1 & C_1 \\ A_2 & B_2 & C_2 \\ A_3 & B_3 & C_3 \end{vmatrix}, \quad (12)$$

where $A, B, C \in \mathbb{R}^3$ are the vertices of the corresponding triangle o. The sign is important because it describes the orientation of triangles seen from the origin for non-convex shapes. For example in Fig. 1, there is a torus given by its triangular mesh. The volumes of the tetrahedrons generated by the two marked triangles has different signs, caused by the different vertex order.

Then from (10) and (11), we get the following approximation for our basic equation (4):

$$\sum_{o \in O_\triangle} \operatorname{sgn}(\operatorname{vol}(T_o)) \int_{\mathcal{T}_o} \mathbf{y} d\mathbf{y} \approx \sum_{\pi \in \varphi(T_\triangle)} \operatorname{sgn}(\operatorname{vol}(\mathcal{T}_\pi)) \int_{\mathcal{T}_\pi} \mathbf{z} d\mathbf{z} \quad (13)$$

How to proceed when an ω_i function is acting on both sides of the equation? In general, we cannot compute the integrals in a straightforward way for an arbitrary function. However, it is possible to derive a similar closed form formula for the integrals as before, when ω_i are polynomials. In particular, if $\omega_i(\mathbf{x}) = x_1^{n_i} x_2^{m_i} x_3^{o_i}$, where $\{(n_i, m_i, o_i)\}_{i=1}^{\ell} = \{(a, b, c) \mid a + b + c = O\}$ and $O \in \{0, \ldots, O_{max}\}$, then (6) becomes:

$$\sum_{o \in O_\triangle} \operatorname{sgn}(\operatorname{vol}(T_o)) \int_{\mathcal{T}_o} y_1^{m_i} y_2^{n_i} y_3^{o_i} d\mathbf{y} \approx \sum_{\pi \in \varphi(T_\triangle)} \operatorname{sgn}(\operatorname{vol}(\mathcal{T}_\pi)) \int_{\mathcal{T}_\pi} z_1^{m_i} z_2^{n_i} z_3^{o_i} d\mathbf{z} \quad (14)$$

Note that the integrands are simply various geometric moments of tetrahedrons $\mathcal{T}_o$ and $\mathcal{T}_\pi$, which can be efficiently computed by applying the methods proposed in [13, 20].

Let us first define trinomial coefficients as

$$(a, b, c) = \frac{(a + b + c)!}{a!b!c!}.$$

The integral over a tetrahedron $\mathcal{T}_o$ can then be written as [13]

$$\int_{\mathcal{T}_o} y_1^{m_i} y_2^{n_i} y_3^{o_i} d\mathbf{y} = 6 \frac{|\operatorname{vol}(\mathcal{T}_o)| m_i! n_i! o_i!}{(m_i + n_i + o_i + 3)!} S_{m_i n_i o_i}(\mathcal{T}_o) \quad (15)$$

with

$$S_{m_i n_i o_i} = \sum_{a_1 + a_2 + a_3 = m_i} \sum_{b_1 + b_2 + b_3 = n_i} \sum_{c_1 + c_2 + c_3 = o_i} (a_1, b_1, c_1)(a_2, b_2, c_2)(a_3, b_3, c_3) \times A_1^{a_1} A_2^{b_1} A_3^{c_1} B_1^{a_2} B_2^{b_2} B_3^{c_2} C_1^{a_3} C_2^{b_3} C_3^{c_3} \quad (16)$$

where $A, B, C \in \mathbb{R}^3$ are the vertices of the corresponding triangle o. The computational complexity of the i^{th} integral would be $\mathcal{O}(M_i^9)$, where $M_i = m_i + n_i + o_i$ is the order of the moments. However, the summations in (16) can be rearranged using the following recursive formulas [20]:

$$C_{ijk}(\mathbf{p}) = (i, j, k) p_1^i p_2^j p_3^k \quad (17)$$

$$D_{abc}(\mathbf{p},\mathbf{q}) = \sum_{i=0}^{a}\sum_{j=0}^{b}\sum_{k=0}^{c} C_{ijk}(\mathbf{p})C_{a-i,b-j,c-k}(\mathbf{q}) \quad (18)$$

Then (16) can be rewritten as

$$S_{m_i n_i o_i} = \sum_{a=0}^{m_i}\sum_{b=0}^{n_i}\sum_{c=0}^{o_i} C_{abc}(A)D_{m_i-a,n_i-b,o_i-c}(B,C) \quad (19)$$

As a result, the complexity reduces to $\mathcal{O}(M_i^6)$ [20]. A further reduction to $\mathcal{O}(M_i^3)$ can be achieved by applying the results of [13] for rearranging (17), (18), and (19) as

$$C_{ijk}(\mathbf{p}) = p_1 C_{i-1,j,k}(\mathbf{p}) + p_2 C_{i,j-1,k}(\mathbf{p}) + p_3 C_{i,j,k-1} \quad (20)$$

$$D_{ijk}(\mathbf{p},\mathbf{q}) = \begin{cases} 0, & \text{if } l<0 \text{ for some } l \in \{i,j,k\} \\ 1, & \text{if } l=0 \text{ for all } l \in \{i,j,k\} \\ p_1 D_{i-1,j,k}(\mathbf{p},\mathbf{q}) + p_2 D_{i,j-1,k}(\mathbf{p},\mathbf{q}) + \\ \quad p_3 D_{i,j,k-1}(\mathbf{p},\mathbf{q}) + C_{ijk}(\mathbf{q}), & \text{otherwise} \end{cases} \quad (21)$$

$$S_{ijk}(\mathcal{T}_o) = \begin{cases} 0, & \text{if } l<0 \text{ for some } l \in \{i,j,k\} \\ 1, & \text{if } l=0 \text{ for all } l \in \{i,j,k\} \\ A_1 S_{i-1,j,k}(\mathcal{T}_o) + A_2 S_{i,j-1,k}(\mathcal{T}_o) + \\ \quad A_3 S_{i,j,k-1}(\mathcal{T}_o) + D_{ijk}(B,C), & \text{otherwise} \end{cases} \quad (22)$$

Since all of the above formulas are recursive, we can store the results of lower order polynomials to compute the higher order ones. Hence the complexity of computing all of the integrals for $i = 1,\ldots,\ell$ over one tetrahedron will be $\mathcal{O}(M^3)$, where M is the maximal degree of polynomials from the $\{\omega_i\}_{i=1}^{\ell}$ set.

4. Numerical implementation

As we presented in the previous section the equations of (6) are only approximately valid, due to the discrete representation of triangular surfaces. In addition, objects extracted from images are subject to various segmentation errors, which again introduces errors in (6). Therefore, an overdetermined system is constructed, *i.e.* $\ell > N$ in (6), 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_t and N_o, respectively. This will basically transform the input objects into a unit cube centered at the origin. Furthermore, after normalization the origin becomes the centroid of the objects, therefore the fourth point of each tetrahedron, described in the previous section, will be the centroid of the objects, which also increases numerical stability.

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. Following [8], this can be achieved by dividing the integrals with the maximal magnitude of the integral over the unit sphere containing the objects:

$$N_i = \int_{\|\mathbf{x}\| \le \frac{\sqrt{3}}{2}} |\omega_i(\mathbf{x})| d\mathbf{x} \quad (23)$$

We thus obtain the following system of equations

$$\frac{1}{N_i}\int_{N_o(\mathcal{F}_o)} \omega_i(\mathbf{y}) d\mathbf{y} = \frac{1}{N_i}\int_{N_t(\varphi(\mathcal{F}_t))} \omega(\mathbf{z}) d\mathbf{z}, \quad (24)$$

where $i = 1,\ldots,\ell$. Using the triangular surface mesh representations, we get the following approximation of our system:

$$\frac{1}{N_i}\sum_{o \in N_o(O_\triangle)} \operatorname{sgn}(\operatorname{vol}(T_o)) \int_{\mathcal{T}_o} y_1^{m_i} y_2^{n_i} y_3^{o_i} d\mathbf{y} \approx \frac{1}{N_i}\sum_{\pi \in \varphi(N_t(T_\triangle))} \operatorname{sgn}(\operatorname{vol}(\mathcal{T}_\pi)) \int_{\mathcal{T}_\pi} z_1^{m_i} z_2^{n_i} z_3^{o_i} d\mathbf{z}, \quad (25)$$

where $i = 1\ldots\ell$ and the integrals over tetrahedrons can be computed via (15) using the recursive formulas (20)–(22). The overall computational complexity of Algorithm 1 is thus $\mathcal{O}(M^3(|O_\triangle| + |T_\triangle| I))$, where I is the number of function calls executed by the *Levenberg-Marquardt* solver.

As mentioned in the previous section, TPS has to satisfy the additional constrains of (2). Since these constraints are linear in the parameters of the transformation, the system (25) can be solved as a constrained least-square problem with linear constraints.

5. Experimental results

In our experiments we used a TPS model with 64 control points placed on a uniform grid, yielding a total of 204 parameters. A system of 220 equations were generated by the set $\omega_i(\mathbf{x}) = x_1^{m_i} x_2^{n_i} x_3^{o_i}$, where $(m_i, n_i, o_i)_{i=1}^{220} = \{(a,b,c) \mid a+b+c = O, O = 1,\ldots,9\}$.

The algorithms have been implemented in `C++` using the *Levenberg-Marquardt* implementation `levmar` of

Algorithm 1 Pseudo code of the proposed algorithm

Input: *template* and *observation* triangular surface meshes
Output: The transformation parameters of φ

1: Choose O_{max} such that $\ell > N$ for the resulting nonlinear function set $\{\omega_i\}_{i=1}^{\ell}$, and for each ω_i compute the normalizing constant N_i using (23).
2: Compute the normalizing transformations N_t and N_o which maps vertex coordinates into $[-0.5, 0.5]$.
3: Construct the system of equations (25) using the recursive formulas (20)–(22).
4: Add constraints from (2).
5: Find a constrained least-squares solution of the system using the *Levenberg-Marquardt* algorithm initialized with the identity transformation.
6: Unnormalizing the solution gives the parameters of the aligning transformation.

Lourakis [15]. 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 Dice coefficient of the aligned objects:

$$\delta = \frac{|F_r \triangle F_o|}{|F_r| + |F_o|} \cdot 100\%, \tag{26}$$

where F_o and F_r denote the set of foreground voxels of the *observation* and *registered* objects respectively.

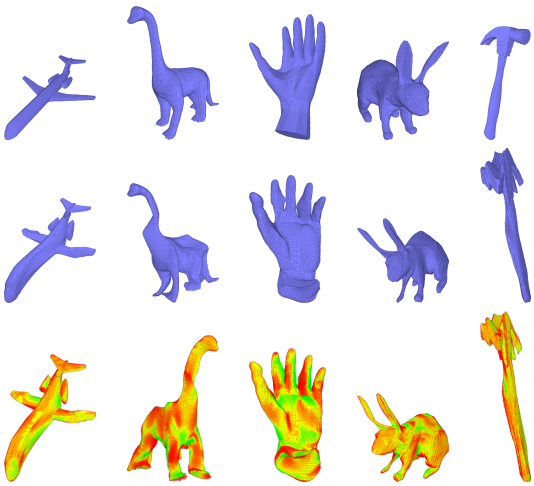

Figure 2. Some results on the synthetic database: First and second row contains the *template* and *observation* objects, respectively. In the last row, *registered* objects are overlayed, overlapping voxels shown in yellow and non-overlapping ones in red and green.

In order to quantitatively evaluate the performance of the proposed method, a synthetic database of 750 deformed objects of size $0.1 - 2.5$ megavoxels has been created. Deformations were generated based on thin plate splines with 16, 32, and 64 control points placed on corresponding grids. Random translations with elements from $[-0.2, 0.2]$ were applied to the control points yielding a random free-form deformation of the objects. In order to preserve the topology of the input object, a diffeomorphic transformation were generated. To achieve this the Jacobian of the generated transformation has been computed and it has been accepted only if it's Jacobian were positive everywhere over the object [10]. Some examples of our registration results on this dataset can be seen in Fig. 2.

The synthetic observations as well as registered objects were generated by the following procedure using 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/` [18]. Note that this triangularization is completely independent of the other triangularizations used as input for Algorithm 1. In particular, the triangular mesh used for the transformation had much higher resolution than what was used in the registration process.

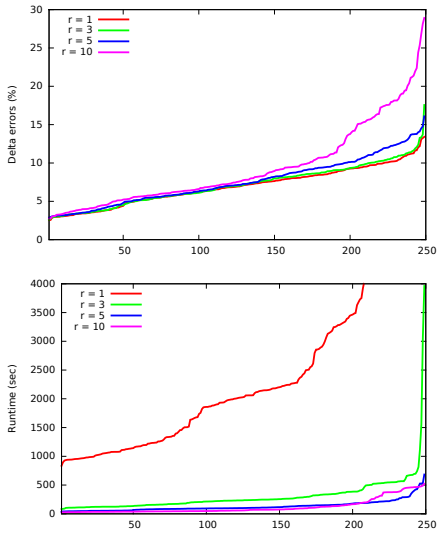

Figure 3. Comparison of registration error and computing time with various r values for the Delaunay sphere in surface mesh extraction.

In the first experiment, we evaluated the sensitivity of the mesh-based algorithm (see Algorithm 1) with respect to the mesh resolution. For that purpose, the triangular mesh extractor algorithms from CGAL library [21] were used where the resolution of the triangular mesh can be controlled by the maximal radius r of the corresponding Delaunay sphere. Using our synthetic dataset, triangular meshes has been created for $r \in \{1, 3, 5, 6, 10\}$. Registration results are presented in Fig. 3 for each of these r values. Obviously, the resolution of the triangular mesh affects the computation

time. On the other hand, the computational time can be significantly reduced by decreasing r at the price of a slightly lower registration accuracy. We found, that $r = 5$ has the best quality over time ratio.

Method	Runtime (min)			$\delta(\%)$		
	m	μ	σ	m	μ	σ
Proposed	1.7	1.38	1.22	6.44	6.12	2.52
Voxel based	31.42	59.77	67.74	7.19	7.05	2.3

Table 1. Results on 750 synthetic images (m – median, μ – mean and σ – standard deviation).

In Table 1, we compared the registration quality and computing times of the naive voxel-based extension of [8] and the proposed mesh-based algorithm for $r = 5$. While the registration errors are of similar magnitude, the computing times show an almost 20 times speed-up for the mesh-based algorithm. This is not surprising, as the mesh-based algorithm works only with triangle vertices whose number is less than 9000, whereas the voxel-based method has to deal with several hundred thousand voxels. It is more interesting, that the mesh-based algorithm also outperforms the voxel-based one in terms of alignment accuracy (see also Fig. 5). This is due to the way these numerical schemes approximate the continuous integrals in (24). In the voxel-based case, approximation error is due to 1) the discretization error on the object's surface (inner voxels are not producing such errors) and 2) the Jacobian is implicitly assumed to be constant within each voxel (including the inner ones!). However, the mesh-based algorithm computes the exact (continuous) integrals over each tetrahedron, therefore the only source of the approximation error is the difference between the true object surface and its approximating triangular mesh.

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 10%, 20% and 30% of the original object volume (see sample slices in Fig. 4). The plots in Fig. 4 show the quantitative evaluation of the alignment error δ on 125 objects. Considering that a $\delta < 10\%$ corresponds to a visually good alignment, our approach is quite robust up to as high as 20% surface error. The robustness test in [8] showed that the method does not tolerate occlusions well, which remains true in 3D as well. Essentially, the algorithm will find a TPS which aligns the template with the occluded observation.

We have also compared our results to the point-based registration frameworks in [11] (GMMREG) and [17] (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

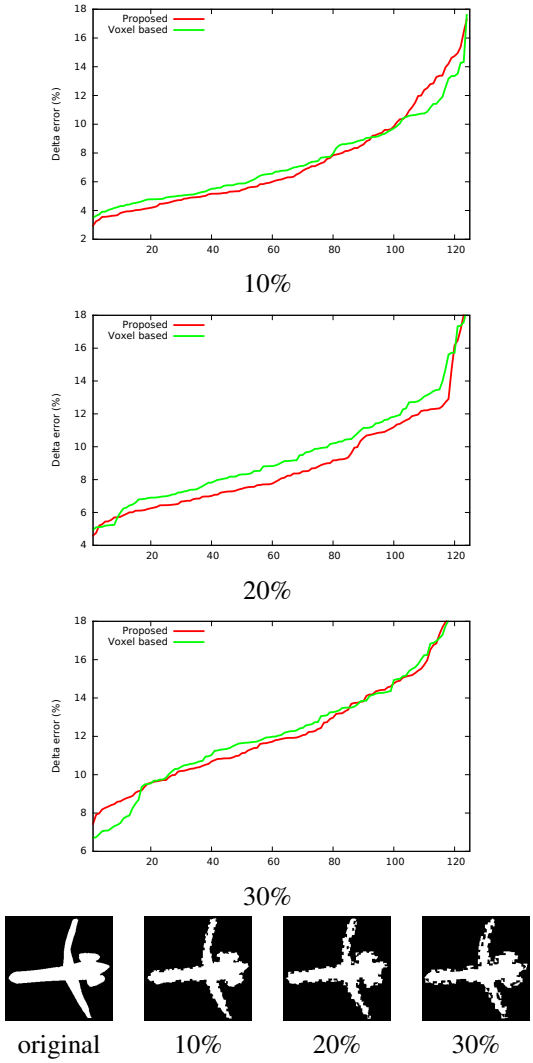

Figure 4. Robustness test results for various degree of synthetically generated surface errors. For each test, samples of surface errors on a voxel slice are shown.

(within the given Matlab framework). The input of these algorithms were the vertices of the extracted triangular surface meshes, using the same extraction technique as in the previous tests. The mesh resolution was controlled by the maximal radius r of the corresponding Delaunay sphere. For the best GMMREG set both the average and median runtimes were 30 min, for the lower resolutions both the average and median runtimes were 3 min. In Fig. 5, quantitative results for $r \in \{2.5, 4.5, 6.5\}$ indicate that these methods provide inferior alignments.

5.1. Medical application

Lung alignment is a crucial task in lung cancer diagnosis [5]. 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

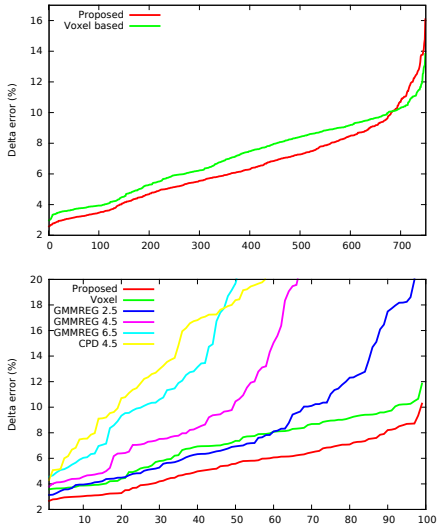

Figure 5. Comparison of voxel-based and triangular mesh-based results. The Delaunay sphere in surface mesh extraction was $r = 5$. On the bottom, comparison with GMMREG [11] and CPD [17]

.

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

We successfully applied the proposed approach to align 3D lung CT scans. Since the triangular mesh based-approach is more accurate and faster, the input voxel volumes were first transformed into a triangular mesh using $r = 5$, and then the TPS parameters were recovered by Algorithm 1. Promising results were obtained on the available 8 image pairs with a median δ error of 5.41% (the mean and standard deviation were 5.83% and 2.09%, respectively).

In medical applications, however, it is necessary to align the inner part of the objects too. For that purpose, the TPS model needs to be regularized and hence we were looking for a solution, which not only solves the system of equations but also minimizes the bending energy:

$$E_{bending} = \lambda \int_{\mathcal{F}_t} \left\{ \left(\frac{\partial^2\phi}{\partial x_1^2}\right)^2 + \left(\frac{\partial^2\phi}{\partial x_2^2}\right)^2 + \left(\frac{\partial^2\phi}{\partial x_3^2}\right)^2 + 2\left(\frac{\partial^2\phi}{\partial x_1 \partial x_2}\right)^2 + 2\left(\frac{\partial^2\phi}{\partial x_2 \partial x_3}\right)^2 + 2\left(\frac{\partial^2\phi}{\partial x_3 \partial x_1}\right)^2 \right\} d\mathbf{x},$$

where $\lambda > 0$ is an application dependent parameter ($\lambda = 10^{-8}$ in our experiments). In Algorithm 1, this function is added to the algebraic error of the system of equations, which are then minimized simultaneously. Some of our results are presented in Fig. 6, where we also show the achieved inner 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.

In many medical image registration problem, one is looking for a smooth deformation, thus a diffeomorphic solution is needed. It is a well known fact, that thin plate splines may not be diffeomorphic without further regularization. For example in the landmark based approaches, an energy function, composed by the derivatives of the displacement field, is minimized [7] or a flow of diffeomorphisms is defined, and using its velocity field and a linear differential operator the deformation energy is minimized [9, 6] to achieve a diffeomorphic transformation. In out approach, using the bending energy minimization provided sufficient constraint to obtain diffeomorphic solutions. This finding has been experimentally confirmed on the lung dataset by computing the Jacobian of the obtained transformations for all of the input voxels: the values were positive in every cases.

6. Conclusion

We have proposed a novel TPS registration method which works without established correspondences to register two objects represented by their triangular surface meshes. The basic idea is to set-up a system of nonlinear equations whose solution directly provides the parameters of the aligning transformation. An efficient numerical scheme were proposed for triangular mesh representation. The efficiency and robustness of the proposed approach have been demonstrated on a large synthetic dataset, and the mesh-based algorithm proved to be the most efficient in both accuracy and computational complexity. Our method compares favorably to two recent 3D matching algorithms [11, 17]. Finally, the algorithm achieved promising results in aligning lung CT images, which demonstrates the usefulness of the method in real life applications.

We also remark, that our numerical approach can be easily applied in the 2D case, using a polygonal representation of the planar shapes: For each edge of the polygons and a common point for all edges, a triangle can be generated, which gives a two dimensional triangular grid, for which similar recursive formulas could be derived as in Section 3.

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 the European Social Fund through project FuturICT.hu (grant no.: TMOP-4.2.2.C-11/1/KONV-2012-0013). Lung images provided by *Mediso Ltd., Budapest, Hungary*.

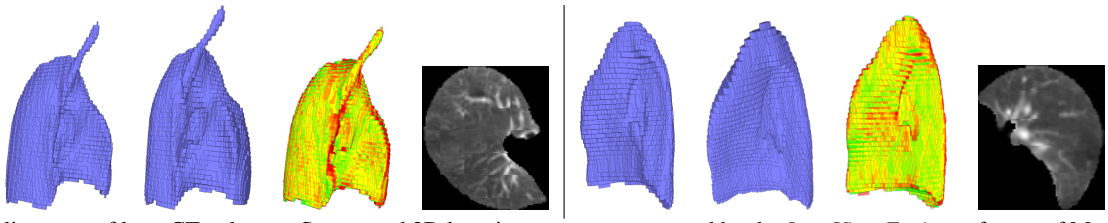

Figure 6. Alignment of lung CT volumes. Segmented 3D lung images were generated by the *InterView Fusion* software of Mediso Ltd.

References

[1] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509–522, 2002. 1

[2] F. L. Bookstein. Principal warps: Thin-Plate Splines and the Decomposition of deformations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(6):567–585, 1989. 1, 2

[3] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences of the United States of America*, 103(5):1168–1172, Jan. 2006. 1

[4] M. M. Bronstein and A. M. Bronstein. Shape recognition with spectral distances. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):1065–1071, 2011. 1

[5] 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*, 82:1016–1020, 2006. 6

[6] V. Camion and L. Younes. Geodesic interpolating splines. In *IN Energy Minimization Methods for Computer Vision and Pattern Recognition*, pages 513–527. Springer, 2001. 7

[7] G. Christensen. Consistent linear-elastic transformations for image matching. In *Proceedings of Information Processing in Medical Imaging*, pages 224–237. Springer-Verlag, 1999. 7

[8] C. Domokos, J. Nemeth, and Z. Kato. Nonlinear Shape Registration without Correspondences. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(5):943–958, 2012. 1, 2, 4, 6

[9] H. Guo, A. Rangarajan, and S. Joshi. Diffeomorphic point matching. In N. Paragios, Y. Chen, and O. Faugeras, editors, *Handbook of Mathematical Models in Computer Vision*, pages 205–219. Springer US, 2006. 7

[10] M. Holden. A review of geometric transformations for nonrigid body registration. *IEEE Trans. Med. Imaging*, 27(1):111–128, 2008. 1, 2, 5

[11] B. Jian and B. C. Vemuri. Robust point set registration using Gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1633–1645, Aug. 2011. 1, 6, 7

[12] G. Jin, S.-J. Lee, J. K. Hahn, S. Bielamowicz, R. Mittal, and R. Walsh. 3d surface reconstruction and registration for image guided medialization laryngoplasty. In *Proceedings of the Second international conference on Advances in Visual Computing - Volume Part I*, ISVC'06, pages 761–770, Berlin, Heidelberg, 2006. Springer-Verlag. 2

[13] P. Koehl. Fast Recursive Computation of 3D Geometric Moments from Surface Meshes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(PrePrints), 2012. 3, 4

[14] Y. Lipman and I. Daubechies. Conformal wasserstein distances: Comparing surfaces in polynomial time. *Advances in Mathematics*, 227(3):1047 – 1077, 2011. 1

[15] 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.]. 5

[16] F. Michel, M. M. Bronstein, A. M. Bronstein, and N. Paragios. Boosted metric learning for 3D multi-modal deformable registration. In *International Symposium on Biomedical Imaging: From Nano to Macro*, pages 1209–1214, Chicago, Illinois, USA, Mar. 2011. IEEE, IEEE. 1

[17] A. Myronenko, X. B. Song, and M. Á. Carreira-Perpiñán. Non-rigid point set registration: Coherent point drift. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Proceedings of the Annual Conference on Neural Information Processing Systems*, pages 1009–1016, Vancouver, British Columbia, Canada, Dec. 2006. MIT Press. 1, 6, 7

[18] F. S. Nooruddin and G. Turk. Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):191 – 205, Apr. 2003. 5

[19] C. Papazov and D. Burschka. Deformable 3D shape registration based on local similarity transforms. *Computer Graphics Forum*, 30(5):1493–1502, Aug. 2011. 1

[20] J. M. Pozo, M. C. Villa-Uriol, and A. F. Frangi. Efficient 3D Geometric and Zernike Moments Computation from Unstructured Surface Meshes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(3):471–484, Mar. 2011. 3, 4

[21] L. Rineau and M. Yvinec. 3D surface mesh generation. In *CGAL User and Reference Manual*. CGAL Editorial Board, 4.0 edition, 2012. 2, 5

[22] 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*, pages 1558–1565, Kyoto, Japan, Oct. 2009. IEEE, IEEE. 1

[23] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 519–528. IEEE Computer Society, 2006. 2

[24] L. Zagorchev and A. Goshtasby. A comparative study of transformation functions for nonrigid image registration. *Image Processing, IEEE Transactions on*, 15(3):529 –538, march 2006. 1, 2