Journal of Biomedical Informatics 48 (2014) 137–147

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Foldover-free shape deformation for biomedicine

Hongchuan Yu [a], Jian J. Zhang [a,*], Tong-Yee Lee [b]

[a] National Center for Computer Animation, Bournemouth University, Poole, UK
[b] Dept. of Computer Science and Information Engineering, National Cheng-Kung University, Taiwan

ARTICLE INFO

Article history:
Received 29 July 2013
Accepted 16 December 2013
Available online 25 December 2013

Keywords:
Shape deformation
Foldover
One-to-one mapping

ABSTRACT

Shape deformation as a fundamental geometric operation underpins a wide range of applications, from geometric modelling, medical imaging to biomechanics. In medical imaging, for example, to quantify the difference between two corresponding images, 2D or 3D, one needs to find the deformation between both images. However, such deformations, particularly deforming complex volume datasets, are prone to the problem of foldover, i.e. during deformation, the required property of one-to-one mapping no longer holds for some points. Despite numerous research efforts, the construction of a mathematically robust foldover-free solution subject to positional constraints remains open. In this paper, we address this challenge by developing a radial basis function-based deformation method. In particular we formulate an effective iterative mechanism which ensures the foldover-free property is satisfied all the time. The experimental results suggest that the resulting deformations meet the internal positional constraints. In addition to radial basis functions, this iterative mechanism can also be incorporated into other deformation approaches, e.g. B-spline based FFDs, to develop different deformable approaches for various applications.

1. Introduction

Shape deformation is an active topic of research in geometric modelling, biomechanics, and computational anatomy, which involves analysis and manipulation of 2D and 3D datasets. For instance, in computational anatomy, deformation techniques are usually deployed for accurate alignment of patient brain scans with a stereotactic atlas in order to quantify the morphology of anatomical structures. This is to understand the variability of brain anatomy by comparing and contrasting volumetric measurements between different patients, e.g. identifying the morphological elements that characterise the attributes of respective groups [2]. Moreover, high dimensional deformation techniques are useful not only in 3D volume data analysis, but also 4D flow MR Imaging. For example, 3D deformation fields can incorporate prior probability density functions from training sample sets into volume dataset segmentation and registration algorithms [22]. 4D-MRI further measures and visualises the temporal evolution of complex flow patterns in a 3D volume [5]. In biomechanics, deformation methods are used to mimic the static and dynamic behaviour of the objects inside the human body through the musculoskeletal model. This model is usually applied to the simulation of continuously deformable organs and tissues, such as muscles, ligaments and tendons [13]. The deformation of the tissue shape plays an important role in computing the muscles and joint forces using this model [15].

However, all the above mentioned techniques do encounter the same difficulty when the amount of deformation is large, which means that unexpected foldovers are likely to appear in the resulting deformations. In computational anatomy, this implies that the mapping between the source and target datasets is not injective. This is a tricky issue for many applications where large deformation is inevitable, such as landmark matching via large deformation [3,11,30]. Despite numerous effects, anomalies, including irregular edges, spikes and jumps, still appear in image warping [7]. In biomechanics, foldover will violate the constraint of topology preservation. One obvious case is that an improper deformation may cause an intersection between muscles and bones when simulating the human limb movement. In geometric modelling, foldovers often lead to self-intersection of the deformed surface. In general, there are two kinds of self-intersections, global intersection (e.g. intersection of different regions of a surface) and local self-intersection (e.g. adjacent faces/pixels overlapping). This paper focuses on the latter. The objective of our research is to develop a mechanism to ensure that large deformation does not result in mesh anomalies, i.e. the resulting datasets are foldover-free.

Our work here is primarily concerned with robust foldover-free deformations for high dimensional datasets. We first derive a general numerical condition, called the *foldover-free condition*, which guarantees no foldover occurs for an arbitrary dimensional dataset.

* Corresponding author.
E-mail addresses: hyu@bournemouth.ac.uk (H. Yu), jzhang@bournemouth.ac.uk (J.J. Zhang), tonylee@mail.ncku.edu.tw (T.-Y. Lee).

1532-0464/$ - see front matter Crown Copyright © 2013 Published by Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jbi.2013.12.011

Our main contribution is to formulate a generalised iterative deformation mechanism by incorporating this condition. We then employ radial basis functions (RBFs) to estimate the iterative step length. The second contribution of this paper is that the proposed iterative mechanism can guarantee the transformations locally invertible. Finally, the proposed iterative mechanism is further extended to other deformation schemes, e.g. B-spline based FFDs. This is the third contribution of this paper. Experiments will further validate the effectiveness and robustness of the proposed iterative algorithm in handling large deformations, some of which are topologically complex.

The paper is organised as follows. Section 2 briefly reviews the related work. Section 3 gives an overview of our foldover-free shape deformation method. In Section 4, we further describe the foldover-free iterative mechanism in detail. Applications and discussions are given in Section 5. The conclusions and future work are presented in Section 6.

2. Related work

Image registration is to determine the correspondence between the source and target images. Herein, shape deformation is described as the transformation between the source and target domain. For medical imaging, the mapping is from 3D to 3D in general, and is nonlinear (or nonrigid) in practice. A desirable property is the preservation of the topology of anatomical structures, which is closely related to the property of one-to-one mapping of the transformation. There is an extensive literature that addresses both the theoretical aspects of nonrigid image registration and practical implementations. For a detailed review, the reader is referred to [10]. Herein, we focus on a subcategory, the basis function expansions. The transformations can be based on radial basis functions [23], B-splines [19] or wavelets [21]. They are smooth and can effectively handle large deformations. Additionally, from a numerical perspective, radial basis functions have a small number of degrees of freedom and can provide fast closed-form solutions. However, these traditional transformations are not invertible in general, i.e. they do not preserve the topology of an object. Recent works in computational anatomy [11,3] have presented the registration approaches based on diffeomorphic nonlinear transformations for one-to-one differential mappings. As applying viscous fluid flow models, diffeomorphic mappings can deal with large deformations. However, the challenge is to solve complex partial differential equations that integrate the velocity field over time along a geodesic path. The implementation thus involves a large number of degrees of freedom, resulting in high computational complexity. This makes such methods unattractive for clinical applications. How to overcome the non-invertibility of the traditional deformation approaches is becoming an active research issue.

In biomechanics, one often analyses and predicates the forces, stresses and strains that occur inside the human body during movement using a musculoskeletal model. For example, to model the limbs, muscles cannot be simply represented as a set of straight lines from the origin to insertion, as the complex morphology makes the muscles warp around passive structures, the bones. Computing the muscle forces had to take into account the bone shapes [15,6]. To estimate muscle parameters, the muscle boundaries of the quantitative CT scans were first manually traced at each vertebral level, then the 3D models of the individual muscles were reconstructed accordingly for parameter estimation in [1]. One challenge in their research is how to prevent the disappearance of existing or new structures during deformation. In another example of biomechanics research, a similar musculoskeletal model is developed for motion analysis, such as clinical gait analysis. Gait is a well-defined motion type that is a repetitive sequence occurring from heelstrike to heelstrike. Several biomechanical models of the lower limb have been developed for the computation of joint angles, joint forces and moments [16]. They have been applied to the investigation of treatment planning that addresses orthopaedic pathology and prevents injury. Similar to [1], they firstly measured a set of markers on the body and then retarget a standard 3D anatomy model accordingly for musculoskeletal modelling. It is essential to ensure the deformed 3D anatomy model is topology preserved.

Another important application of shape deformation is in CAD field. [8] gives a survey of spatial deformation methods. We only recap several papers here that address the self-intersection issue. [24] noted the “space-tearing” phenomenon and proposed the Simple Constrained Deformation (SCoDef), which assigns each constrained point an independent region of influence so that the deformation is bounded by a union of these spherical regions. Further work, e.g. Sweepers [25], Warp Sculpting [26], Blendeforming [27] and Vector-Field Based Deformation [28], also adopted the similar idea as SCoDef. Essentially, shape deformation requires that transformations must be invertible. Unfortunately, B-spline based FFD transformations are not invertible in general. Invertibility of transformations is the intrinsic property causing foldovers. Thus, it is meaningful to seek the invertible transformations for shape deformation. The distinct advantage of our proposed algorithm is to guarantee the local invertibility of large deformations. Additionally, many methods of shape deformation also suffer from undesired volume changes, even resulting in local self-intersection [29]. Shape- and volume-preserving deformation techniques are therefore receiving more and more attentions.

3. Algorithm overview

Considering a given dataset S in R^n, a transformation U is a one-to-one mapping which maps the points $X \in S$ into the individual corresponding points U of another domain Ω in R^n with arbitrary m constraint point pairs $(X_i^* \leftrightarrow U_i^*)$, i.e.

$$\begin{cases} \quad X \in S \rightarrow U(X) \in \Omega \\ \text{subject to } U(X_i^*) = U_i^*, \quad i = 1, \dots, m \end{cases}.$$

Our algorithm is developed based on radial basis functions (RBFs). This is due to many well behaved properties of the RBF scheme, e.g. mesh-free, C^2 continuous and ability to handle large deformations. In particular the RBF scheme is very suitable for the implementation of the positional constraint based deformation, since the radial basis functions simplify the multidimensional representation of the deformation through a radial distance metric. In contrast, B-spline expansions conventionally use the linear combinations of univariate forms to construct multidimensional basis functions. Extending from the bivariate to a trivariate case, there is a proliferation in control points that makes B-spline based FFDs difficult in practice [8]. Therefore, using RBFs will not lead to any new control points both for triangular meshes and tetrahedral ones.

Unlike the previous applications of RBFs [9,20], in our algorithm, the RBF scheme is used to compute the displacement of the point's coordinates, that is, the deformation fields,

$$\Delta u_i = P_i(X) + \sum_j^m \lambda_{ij} \phi(\|X - C_j\|), \quad i = 1, \dots, n \tag{1}$$

where the displacement is represented as a vector of $\Delta U = (\Delta u_1, \dots, \Delta u_n)^T$, RBF coefficients denote λ_{ij}, $C_j \in R^n$ represents the constraint points in Ω, and $P_i(X)$ is an affine transformation, i.e. $P_i(X) = a_{i0} + \sum_{k=1}^n a_{ik} x_k$ (herein, $a_{ik}, k = 0, 1, \dots n$ are affine coefficients), n stands for the dimensionality of the datasets. Our intention is to introduce foldover-free constraints through the deformation fields of Eq. (1).

Although there are various forms of radial basis functions, we utilise the thin plate spline (TPS) as the basis functions ϕ here rather than others, e.g. multiquadric interpolation (MQ). This is because for the large and global deformation, TPS performances better than MQ [23]. The deformed U is obtained by updating $U(X) = X + \Delta U$. For next iteration, let $X \leftarrow U(X)$.

Our deformation algorithm is summarised as below. The proposed algorithm is executed iteratively, and the superscript j stands for the iteration index below. m points in the dataset are constrained to the specified positions: $C_i^{(0)}$ denotes the initial positions, and their destination locations are denoted as $C_i^*, i = 1, \ldots m$. The iterative mechanism will be described in detail in the following section.

```
(1) Input: Initial dataset $S^{(0)}$ and a set of user-specified
    constraint point pairs $(C_i^{(0)}, C_i^*), i = 1, \ldots, m$.
(2) Loop: estimate the scaling factor $\delta$ by Eq. (6) (see
    Section 4) based on the configuration of the current
    $m$ constraint point pairs $(C_i^{(j)}, C_i^*), i = 1, \ldots, m$.
(3) Computing the current constraint points' displacements
    by $\Delta C_i^{(j+1)} = \delta(C_i^* - C_i^{(j)})$, such that $C_i^{(j+1)} = C_i^{(j)} + \Delta C_i^{(j+1)}$.
(4) Computing the displacements of points on $S^{(j)}$ by
    Eq. (A2) (see Appendix A) and updating $S^{(j)} \rightarrow S^{(j+1)}$.
(5) End Loop until all $C_i^{(j)}$ reach $C_i^*$.
```

Fig. 1 illustrates our RBF based deformation algorithm foldover-free under large deformation. For the sake of visualisation, our method is performed on a 2D mesh. Moreover, for some extreme deformations, the final deformed mesh may not always converge to the most ideal positions by our approach (see Fig. 1d). This may be because either the RBF scheme is not flexible enough or the foldover-free condition prevents the constrained points from reaching the desired locations. To deal with the extreme deformations, a simple strategy is to subdivide meshes (or tetrahedral ones), so as to make the mesh more flexible as shown in Fig. 1e.

4. Foldover-free iterative mechanism

From a mathematical point of view, a "foldover-free" deformation gives a "one-to-one" mapping between the original surfaces (or datasets) and their target domains. We will first formulate a foldover-free condition, and then explain our iterative mechanism in a general form.

4.1. Foldover-free condition

The goal of our work is to develop a foldover-free deformation approach with a set of the positional constraints. It requires that the mapping $U(X)$ be globally univalent or "globally one-to-one", that is, the topology or the connection relationship between any pair of vertices in the datasets should remain unchanged before and after the deformation. Mathematically it means the determinant of the Jacobian matrix must be positive everywhere,

$$\det(\nabla U) > 0. \tag{2}$$

According to the Gershgorin circle theorem, a sufficient condition of satisfying Eq. (2) can be given as,

$$\frac{\partial u_i}{\partial x_i} > \sum_{j=1, j \neq i}^{n} \left| \frac{\partial u_i}{\partial x_j} \right|, \tag{3}$$

where $i = 1, \ldots, n$. The geometric meaning of Eq. (3) is that the vectors $\partial u_i / \partial(x_1, \ldots, x_n)$ are linearly independent of each other. For 2D scenarios, this implies that the included angle of the vectors is less than π. For 3D scenarios, such three vectors should not be co-planar. Eq. (3) is called the foldover-free condition.

4.2. Iterative framework based on RBFs

Our deformation algorithm employs an iterative framework and the displacements of vertices are determined considering the condition of Eq. (3), which eliminates foldovers. To this end, we need to limit the length of each displacement vector by introducing a scaling factor δ. Herein we firstly illustrate the geometric meaning of the sufficient condition Eq. (3) as below, and then show the scaling factor δ as the iterative steplength.

The regions of the 1st order partial derivatives in R^n defined by Eq. (3) can be further described as follows, $i = 1, \ldots, n$,

$$\Omega_i(\delta) = \left\{ \begin{array}{l} (r_1, \ldots, 1 + r_i, \ldots, r_n) : |r_j| \leqslant \left| M_{x_j} K^{-1} \begin{pmatrix} \Delta c_i \\ 0 \end{pmatrix} \right| \delta, \;\; j = 1, \ldots, n \\ 1 + r_i > \sum\limits_{j=1, j \neq i}^{n} |r_j| \end{array} \right\}, \tag{4}$$

where $(r_1, \ldots, 1 + r_i, \ldots, r_n)$ denotes a point in $\Omega_i(\delta)$, $\Delta c_i = (\Delta x_i^1, \ldots, \Delta x_i^m)^T$ denotes as the displacements of all the constraint points, K is a symmetric matrix filled with the radial basis functions $\phi_{kj} = \phi(\|C_k - C_j\|), k, j = 1, \ldots, m$ and the constraint points' coordinates, $M(X) = (\phi(\|X - C_1\|), \ldots, \phi(\|X - C_m\|), x_1, \ldots, x_n, 1)$ and M_{x_j} denote the partial derivatives of $M(X)$ (For details, refer to Appendix A). The scaling factor δ is not a constant, and depends on the displacements of the constraint points. For simplicity, assume that the vectors $\{\Delta c_i\}$ have the same distribution. The regions $\Omega_i(\delta)$ would then have the same size in terms of Eq. (4). In an n-dimensional Euclidean space, the condition of Eq. (3) can be described as,

$$\sum_{j=1}^{n} \left(M_{x_j} K^{-1} \begin{pmatrix} \Delta c_i \\ 0 \end{pmatrix} \right)^2 \delta^2 < \frac{1}{n}, \tag{5}$$

for the region $\Omega_i(\delta)$ and $i = 1, \ldots, n$.

To further illustrate the geometric meaning of Eq. (4), we consider both 2D and 3D scenarios here. When $n = 2$, Eq. (4) is expressed as,

$$\left\{ \begin{array}{l} \Omega_1(\delta) = \left\{ (1 + r_1, r_2) : |r_1| \leqslant \left| \mathbf{M}_x K^{-1} \begin{pmatrix} \Delta c_1 \\ 0 \end{pmatrix} \right| \delta, |r_2| \leqslant \left| \mathbf{M}_y K^{-1} \begin{pmatrix} \Delta c_1 \\ 0 \end{pmatrix} \right| \delta, 1 + r_1 > |r_2| \right\} \\ \Omega_2(\delta) = \left\{ (r_1, 1 + r_2) : |r_1| \leqslant \left| \mathbf{M}_x K^{-1} \begin{pmatrix} \Delta c_2 \\ 0 \end{pmatrix} \right| \delta, |r_2| \leqslant \left| \mathbf{M}_y K^{-1} \begin{pmatrix} \Delta c_2 \\ 0 \end{pmatrix} \right| \delta, 1 + r_2 > |r_1| \right\} \end{array} \right. .$$

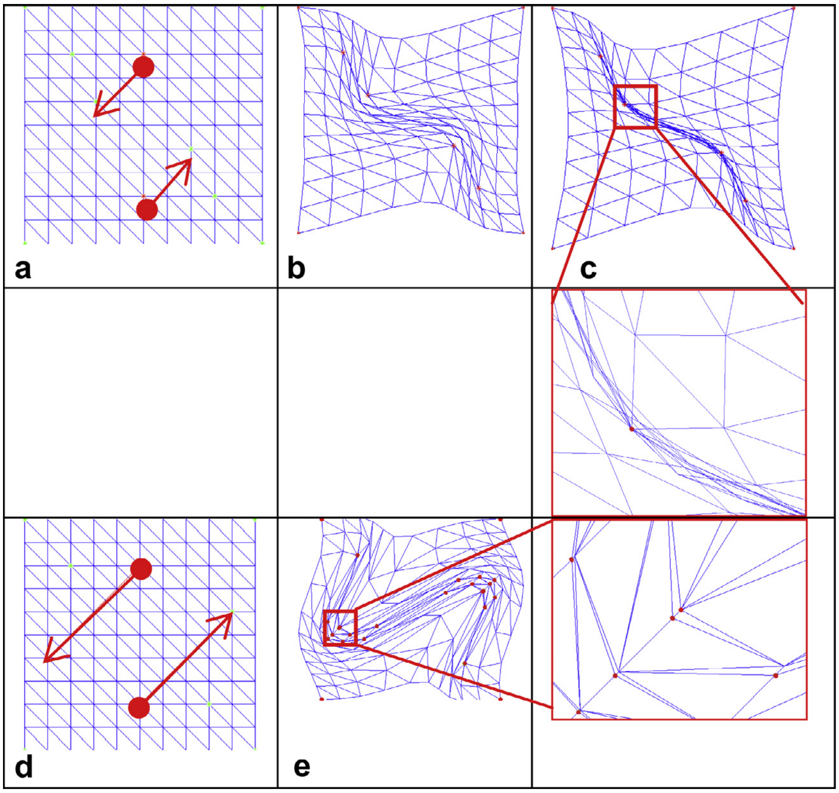

Fig. 1. Illustration of our method performing on a 2D mesh with large deformation. The red lines in (a) show the two constrained points are required to move to their individual destinations in the deformed domain, (b) shows the result by our method, while (c) showing the result by the non-iterative RBF scheme. It can be observed that triangles are folding. Moreover, (d) shows these two constrained points are required to move farther than that in (a). Applying the subdivision strategy, (e) shows the result by our method. The red points in (e) indicate the new vertices generated by the subdivision process. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

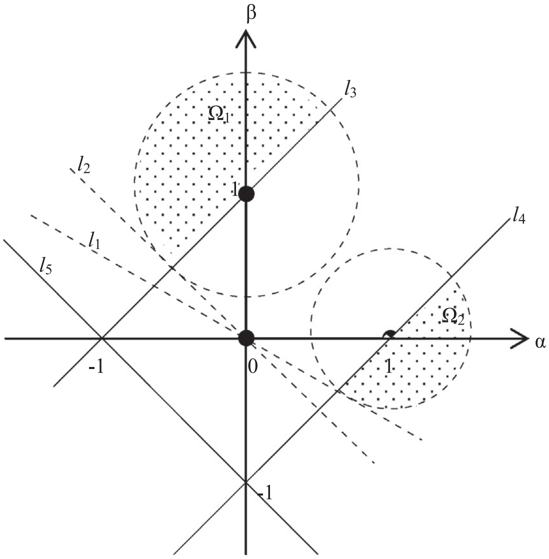

Fig. 2. Illustration of the condition of Eq. (5) in 2D scenarios. The dashed lines denote the undetermined boundaries.

Fig. 2 illustrates the regions of $\Omega_1(\delta)$, $\Omega_2(\delta)$. The dashed line is used to highlight these undetermined boundaries. The condition of Eq. (3) implies that vectors $\partial u/\partial(x,y)$, $\partial v/\partial(x,y)$ should be linearly independent of each other. Fig. 2 intuitively illustrates this concept by five lines: l_1, l_2, l_3, l_4, l_5. For example, r_1 and r_2 should be above line l_3 or under line l_4, and above line l_5, so that the linear independence can be guaranteed. This can be achieved by the scaling factor δ in Eq. (4). Moreover, for simplicity, assume that the vectors $\Delta c_1, \Delta c_2$ have the same distribution. The regions $\Omega_1(\delta)$, $\Omega_2(\delta)$ would then have the same size in terms of Eq. (4). This will lead to the overlap of straight lines l_1 and l_2 and form a new dividing line, which is $r_1+r_2=0$ in Fig. 2. Line $r_1+r_2=0$ guarantees that the included angle is less than π. Consequently, the 2D form of the condition Eq. (5) can be re-formulated as,

$$\begin{cases} \left(M_xK^{-1}\begin{pmatrix}\Delta c_1\\ 0\end{pmatrix}\right)^2\delta^2+\left(M_yK^{-1}\begin{pmatrix}\Delta c_1\\ 0\end{pmatrix}\right)^2\delta^2<\frac{1}{2} \\ \left(M_xK^{-1}\begin{pmatrix}\Delta c_2\\ 0\end{pmatrix}\right)^2\delta^2+\left(M_yK^{-1}\begin{pmatrix}\Delta c_2\\ 0\end{pmatrix}\right)^2\delta^2<\frac{1}{2} \end{cases}.$$

When $n=3$, Eq. (4) is given by,

$$\Omega_1(\delta)=\left\{\begin{array}{l}(1+r_1,r_2,r_3):|r_1|\leqslant\left|M_xK^{-1}\begin{pmatrix}\Delta c_1\\ 0\end{pmatrix}\right|\delta,\\ |r_2|\leqslant\left|M_yK^{-1}\begin{pmatrix}\Delta c_1\\ 0\end{pmatrix}\right|\delta,\\ |r_3|\leqslant\left|M_zK^{-1}\begin{pmatrix}\Delta c_1\\ 0\end{pmatrix}\right|\delta, 1+r_1>|r_2|+|r_3|\end{array}\right\},$$

$$\Omega_2(\delta)=\left\{\begin{array}{l}(r_1,1+r_2,r_3):|r_1|\leqslant\left|M_xK^{-1}\begin{pmatrix}\Delta c_2\\ 0\end{pmatrix}\right|\delta,\\ |r_2|\leqslant\left|M_yK^{-1}\begin{pmatrix}\Delta c_2\\ 0\end{pmatrix}\right|\delta,\\ |r_3|\leqslant\left|M_zK^{-1}\begin{pmatrix}\Delta c_2\\ 0\end{pmatrix}\right|\delta, 1+r_2>|r_1|+|r_3|\end{array}\right\},$$

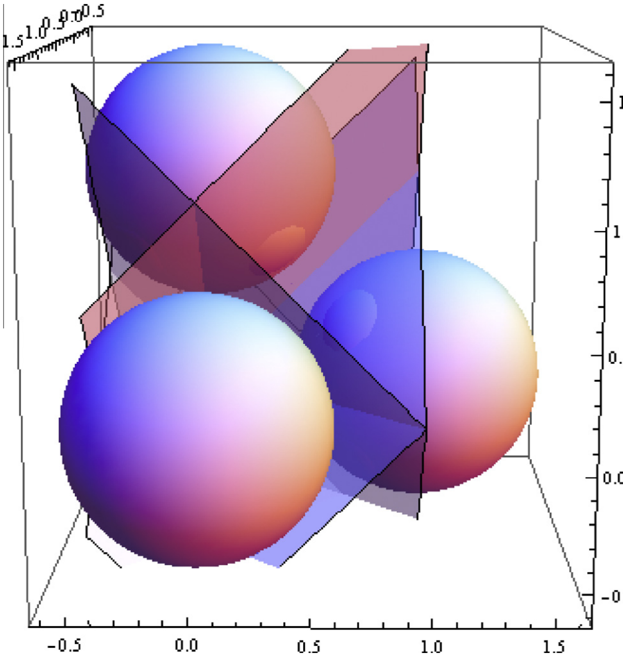

Fig. 3. The illustration of the condition of Eq. (5) in 3D scenarios.

$$\Omega_3(\delta)=\left\{\begin{array}{l}(r_1,r_2,1+r_3):|r_1|\leqslant\left|M_xK^{-1}\begin{pmatrix}\Delta c_3\\0\end{pmatrix}\right|\delta,\\|r_2|\leqslant\left|M_yK^{-1}\begin{pmatrix}\Delta c_3\\0\end{pmatrix}\right|\delta,\\|r_3|\leqslant\left|M_zK^{-1}\begin{pmatrix}\Delta c_3\\0\end{pmatrix}\right|\delta,1+r_3>|r_2|+|r_1|\end{array}\right\}.$$

Fig. 3 shows the regions of $\Omega_1(\delta)$, $\Omega_2(\delta)$, $\Omega_3(\delta)$. Taking the condition of Eq. (3) into account, we hope to point out that the dividing plane of $r_1 + r_2 + r_3 = 0$ guarantees that the determinant of Jacobian is greater than zero. Similar to the 2D scenarios, the 3D form of the condition Eq. (5) is expressed as,

$$\left\{\begin{array}{l}\left(M_xK^{-1}\begin{pmatrix}\Delta c_1\\0\end{pmatrix}\right)^2\delta^2+\left(M_yK^{-1}\begin{pmatrix}\Delta c_1\\0\end{pmatrix}\right)^2\delta^2+\left(M_zK^{-1}\begin{pmatrix}\Delta c_1\\0\end{pmatrix}\right)^2\delta^2<\frac{1}{3}\\\left(M_xK^{-1}\begin{pmatrix}\Delta c_2\\0\end{pmatrix}\right)^2\delta^2+\left(M_yK^{-1}\begin{pmatrix}\Delta c_2\\0\end{pmatrix}\right)^2\delta^2+\left(M_zK^{-1}\begin{pmatrix}\Delta c_2\\0\end{pmatrix}\right)^2\delta^2<\frac{1}{3}\\\left(M_xK^{-1}\begin{pmatrix}\Delta c_3\\0\end{pmatrix}\right)^2\delta^2+\left(M_yK^{-1}\begin{pmatrix}\Delta c_3\\0\end{pmatrix}\right)^2\delta^2+\left(M_zK^{-1}\begin{pmatrix}\Delta c_3\\0\end{pmatrix}\right)^2\delta^2<\frac{1}{3}\end{array}\right..$$

In terms of Eq. (5), every region can provide their individual estimations of the scaling factor. For the entire dataset S, it is straightforward to select the minimum as the estimation of δ as below,

$$\delta=\min_{X\in S}\delta(X) \tag{6}$$

where

$$\delta(X)=\min_{i=1}^{n}\left\{\frac{1}{\sqrt{n}}\left(\sum_{j=1}^{n}\left(M_{x_j}K^{-1}\begin{pmatrix}\Delta c_i\\0\end{pmatrix}\right)^2\right)^{-1/2}\right\}.$$

Note that vectors Δc_i, $i = 1, \ldots, n$ are the differences of the current constraint point positions and their individual destinations' positions. Scaling factor δ depends on the configuration of the current constraint points. For any constraint point C_i, its displacement needs to be scaled by δ iteratively to approximate its destination C_i^*. The configuration of all the current constraint points may be defined as the current state of the dataset. When the constraint points' positions are updated, the state is changed accordingly. Thus, δ can also be regarded as the iterative step length.

So far we have outlined the proposed iterative deformation mechanism and estimated the scaling factor. The displacements of the constraint points are adaptively changed based on the estimated scaling factor δ of Eq. (6). We also hope to point out that Eq. (6) is a sufficient condition, which means that there might be scenarios where Eq. (6) is not satisfied, but still is foldover-free. There is a simple explanation, that is, the scaling factor is estimated by the regions defined on continuous domains. Compared to continuous domains, the discrete settings (e.g. triangle mesh) are too sparse. Thus, Eq. (6) seems to be overstrict to the discrete settings. What can be guaranteed however is by satisfying Eq. (6), we can both eliminate all possible cases of foldovers and preserve mesh topology.

4.3. Extension to B-spline scheme

B-spline based FFD approaches have been widely applied to various geometric modelling applications. We can extend the iterative deformation mechanism to include the B-spline scheme. Without loss of generality, a 2D cubic B-spline function is represented by a patch as follows,

$$U(X)=X+\sum_{i=0}^{3}\sum_{j=0}^{3}B_i(w)B_j(v)\Delta C_{ij}, \tag{7}$$

where $0 < w$, $v < 1$, B_i, B_j denote the basis functions, and ΔC_{ij} denotes the displacement vector of the control points. According to Eq. (7), the deformation field X is updated by $X \leftarrow U(X)$ in an iterative way. We can deduce the scaling factor δ as,

$$\delta=\min\left\{\begin{array}{l}\frac{1}{\sqrt{2}}\left(\begin{array}{c}\left(\frac{B_w(\Delta c_1)}{B_w(c_1)}+\frac{B_v(\Delta c_1)}{B_v(c_1)}\right)^2+\\\left(\frac{B_w(\Delta c_1)}{B_w(c_2)}+\frac{B_v(\Delta c_1)}{B_v(c_2)}\right)^2\end{array}\right)^{-1/2},\\\frac{1}{\sqrt{2}}\left(\begin{array}{c}\left(\frac{B_w(\Delta c_2)}{B_w(c_1)}+\frac{B_v(\Delta c_2)}{B_v(c_1)}\right)^2+\\\left(\frac{B_w(\Delta c_2)}{B_w(c_2)}+\frac{B_v(\Delta c_2)}{B_v(c_2)}\right)^2\end{array}\right)^{-1/2}\end{array}\right\}. \tag{8}$$

(For the details of computation, refer to Appendix B please.) For a 3D cubic B-spline scheme, we can yield an analogous estimation of δ, as well as Eq. (8).

5. Applications

For simplicity, in our implementations, our deformation algorithm presented in Section 3 employs the RBF scheme rather than the B-spline one. This is because extending the B-spline scheme from the bivariate to trivariate case will involve a large number of control points. To highlight the advantage of the proposed iterative deformation mechanism, we perform the deformation separately by our iterative algorithm (i.e. with constraint the scaling factor) and the non-iterative RBF scheme (i.e. without constraint of the scaling factor) in the following.

This section will first illustrate some interesting properties of the proposed algorithm and then address some important applications on medical imaging and biomechanics.

5.1. Numerical properties

In general, to estimate the scaling factor δ, it is required to test all the points X within a dataset. However, our observation suggests that the constraint points always hold larger displacements than the others at each iteration. Thus, it is sufficient to estimate the δ only by testing the constraint points instead of all the points.

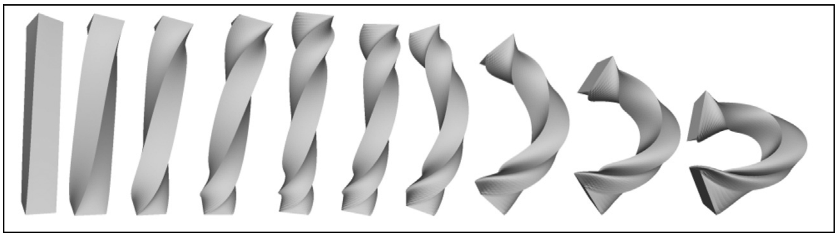

Fig. 4. Illustration of bending and twisting a bar by our method. For large bending, we firstly placed 5 constraint points on the skeleton of the bar. Several bending states of the bar are generated.

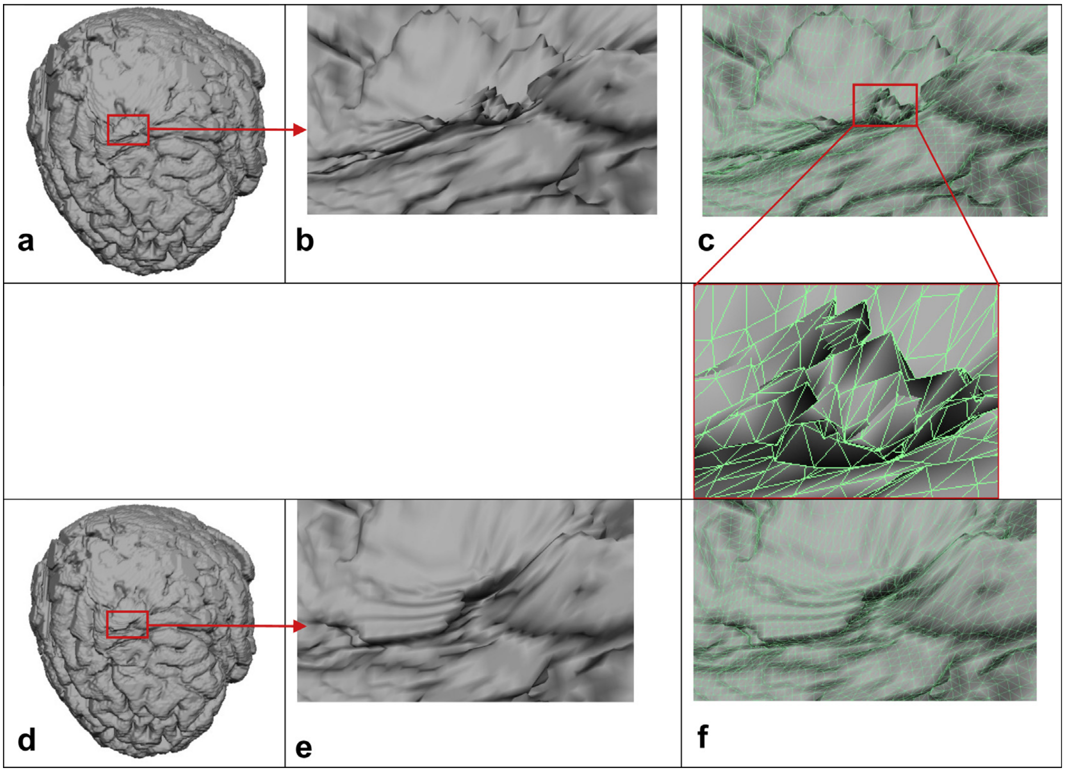

Fig. 5. Illustration of the foldovers on 3D volume data. Applying 3D FFDs to the gyrus of brain volume data, and the resulting deformation in (a) the whole deformed brain gyrus; (b) the local detail of foldovers; (c) the isosurface with mesh corresponding to (b). For comparison, the results by our approach are shown in (d)–(f). Note that there are 350 constraint points in (a) while only 28 constraint points in (d).

Volume preservation is one of the desired properties for shape deformation. [18] proposed to firstly embed triangular meshes into tetrahedral ones and then perform some deformation approaches directly on the tetrahedral meshes. However, this usually leads to a large computational burden. Since the RBF scheme is mesh-free, our deformation algorithm will not incur extra computational cost. To illustrate the effectiveness of our algorithm, Fig. 4 shows some extreme deformation examples, i.e. a bar is firstly twisted by 360° and then bent by 150°. Notice that taking into account the volume, it is natural to place some constrained points manually along the skeleton of an object, e.g. 5 constrained points on the skeletons of the bar. This is similar to the ideas of the sketch-based editing system [14] and the skeleton-curve deformation method [24]. The user does not necessarily need to place the constraint

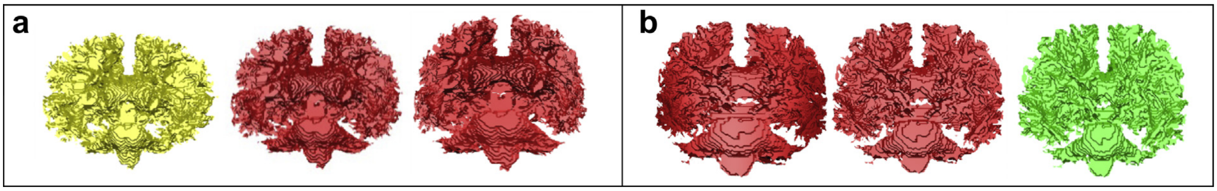

Fig. 6. Simulation of a child brain white matter development. (a) The deformation from the source (1st model, a child brain white matter) toward the target (3rd model in (b), an adult brain white matter); (b) shows the deformation from the target (3rd model) toward the source (1st model in (a)). The 3rd model of (a) and the 1st model of (b) are the most similar to each other. We did not further deform the source model to the target, since further deformation would result in perceptible distortion. Herein, there are 40 constraint point pairs.

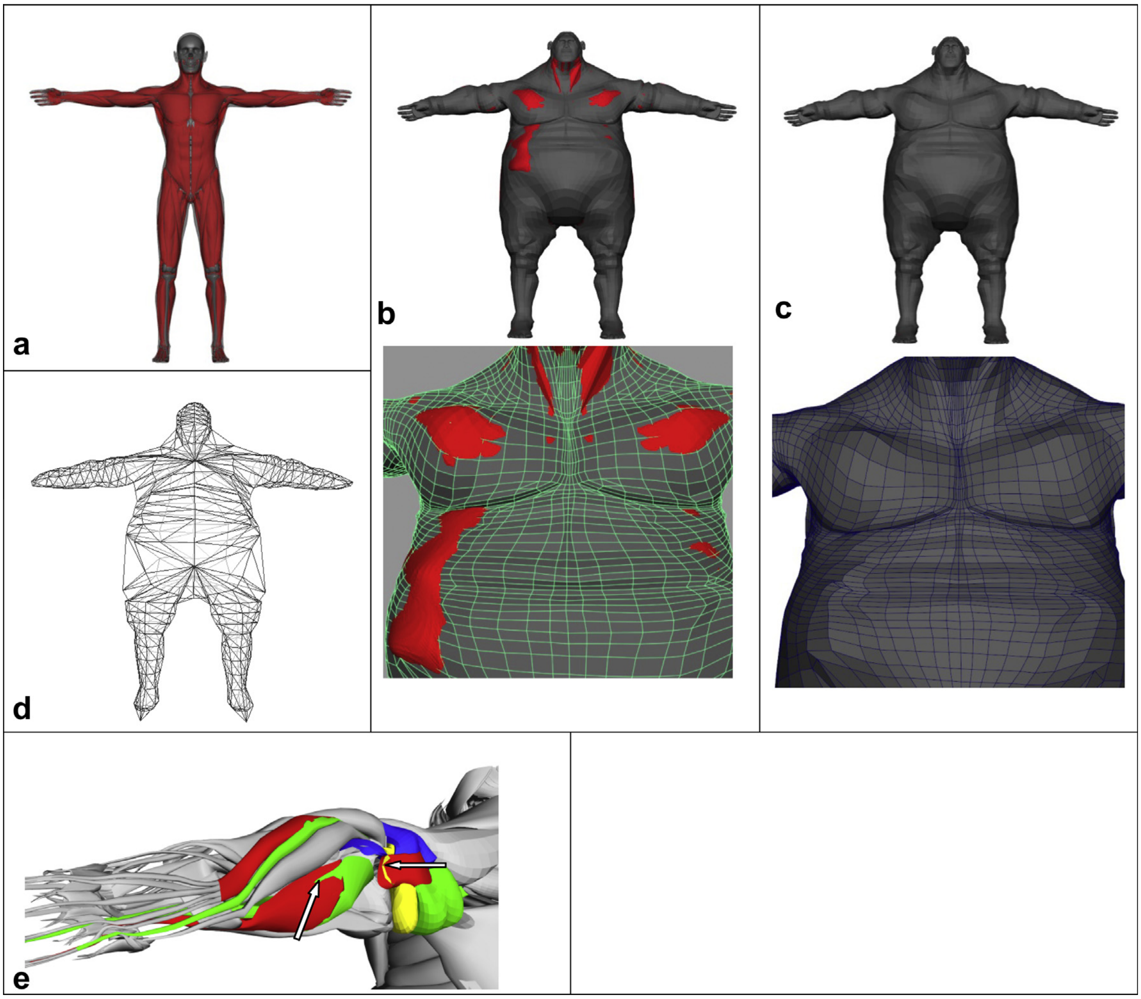

Fig. 7. Retargeting a standard 3D human body model: (a) shows the standard 3D anatomic model including skin, muscles and bones; (b) shows the results of applying non-iterative RBF scheme. The muscles (red) intersect with the skin; (c) shows the result by our iterative algorithm; (d) shows the skin cage that contains 508 vertices as constraint points, while the shin mesh contains 14,652 vertices; (e) shows the intersection of the upper limb muscles. To highlight the intersection of muscles, muscles are depicted by different colours. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

points on the precise skeletons of the objects. In our implementation, imprecise skeleton constrained points are fine, which guides to the desired shape.

5.1.1. Computational complexity

The core of our RBF-based deformation approach is to update the RBF coefficients at each iteration. The main cost is therefore to compute the inverse of a real symmetric matrix, which costs $O(nm^3)$, where m is the number of the constraint points and n denotes the dimensionality of the dataset. The time complexity can be estimated as $O(knm^3)$, where k denotes the iteration number. However, in practice, updating the whole dataset costs the most of the running time compared to updating the RBF coefficients, e.g. volume data. This is because the number of constraint points is always far smaller than that of the points within a dataset. And our algorithm usually converges with 3–6 iterations. Although it is usually costly to invert a matrix, because the dimension of the matrices to be inverted is small, in practice the cost is negligible.

5.1.2. Limitations

Our iterative deformation mechanism guarantees to produce diffeomorphic deformation fields and the experiments in Figs. 1 and 4 also validate the effectiveness of handling large deformations. However it does not always guarantee to converge to the most ideal positions for large bending deformations. Experiments in Fig. 4 suggest that our approach can work well when the bending angle is less than 60°. Therefore, bending the bar by 150° in Fig. 4, we had to do it in steps. In this example, we bent the bar by 50° for each step for three steps.

5.2. Application to brain volume data

Voxel-based volume deformation has been widely used in image segmentation and registration, e.g. CT and MRI datasets. Our iterative algorithm is suitable for the volume data, since the RBF scheme is mesh-free. Our algorithm can prevent foldovers with only a small number of constraint points. This is in contrast to the other non-iterative deformation methods, such as the B-spline FFDs, which cannot overcome this numerical difficulty even if many constraint points are added. For comparison, we performed a large deformation on the precentral gyrus of a segmented MRI brain volume dataset by a cubic B-spline FFDs [17] and our algorithm respectively, as shown in Fig. 5a and d. Since we were concerned about the large deformation rather than image registration here, the cubic B-spline FFDs did not take into account the voxel intensity similarity measure, nor the constraint of the foldover-free condition. Foldovers within the volume deformation are usually hidden inside the volume dataset rather than on its surface.

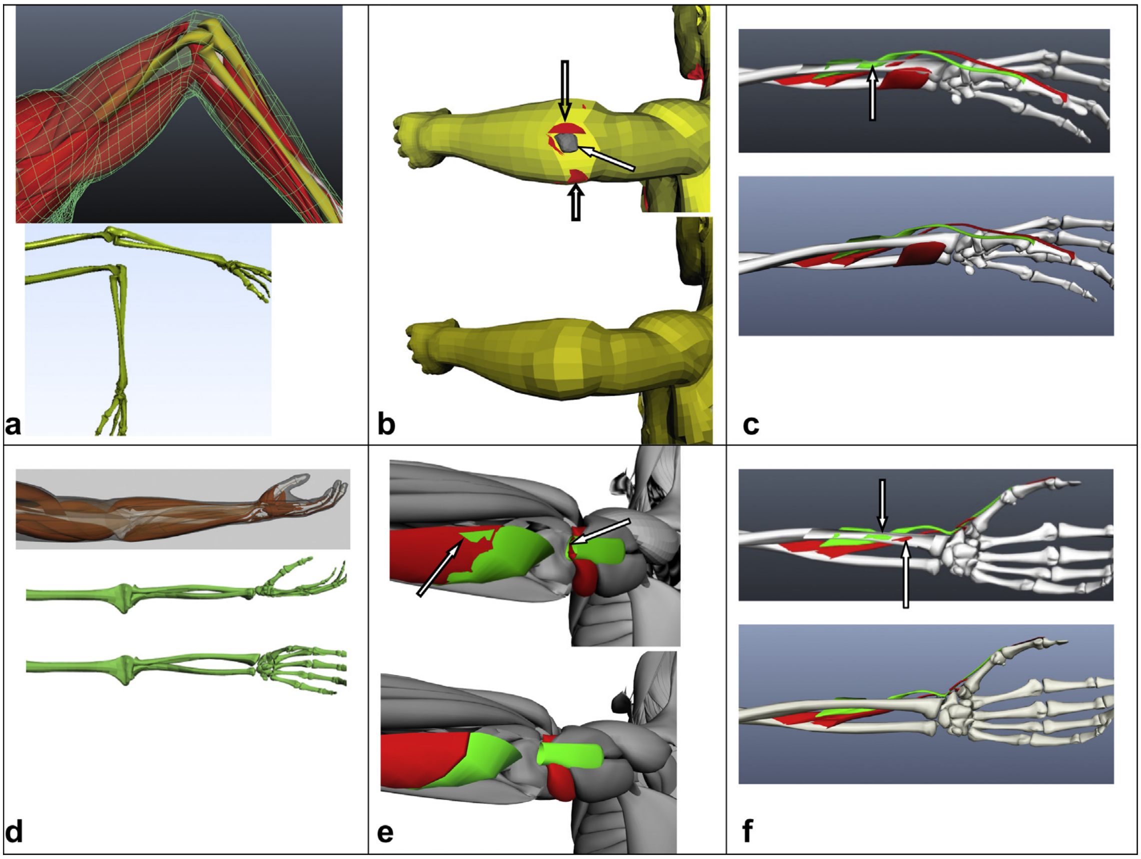

Fig. 8. Illustration of bending and turning up forearm: (a) and (d) show the results by our iterative algorithm. Herein, the skeletons illustrate the forearm movements. To compare with the non-iterative deformation (i.e. no constraint of scaling factor), (b), (c), (e) and (f) show the intersections of the tissues after the non-iterative deformation on the individual upper halves, while our iterative results being put on the individual lower halves for comparison; (b) shows the muscles (red) and bone (grey) piercing the skin; (e) shows the intersection between the muscles (highlighted by red and green); (c) and (f) further show the intersections of the two muscles and bones. Note that there are 35 constraint points to be placed on the bones. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

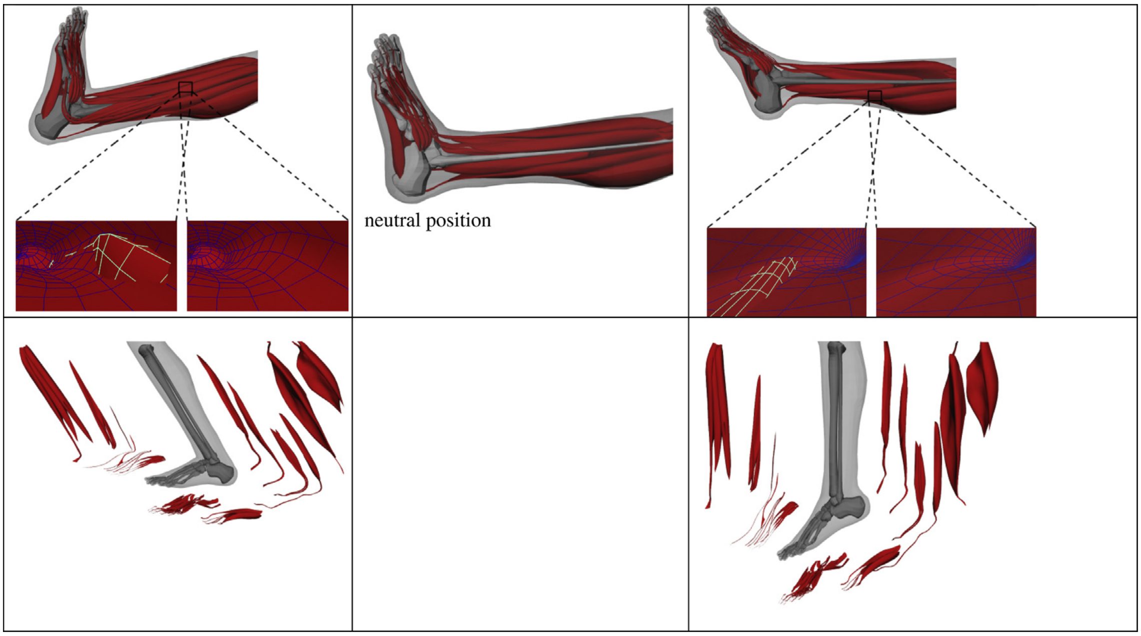

Fig. 9. Illustration of the jumping/landing of the left leg. The top row shows the results by the non-iterative and iterative algorithms. There is no intersection inbetween two muscles by our iterative algorithm. The 2nd row shows the change of the muscle shapes. There are 20 constraint points to be placed on the bones.

To see it clearly, we utilised the Marching Cubes to extract the isosurface for rendering purposes. This can highlight the foldovers occurred in the 3D volume data by comparing the meshes shown in Fig. 5c and f. One can note the self-intersections of triangles in Fig. 5c.

In the next example, we applied our deformation algorithm to the simulation of child brain development. We demonstrate that our method is very useful in the analysis of the pediatric and adult neuroimaging data in a common stereotactic space for developmental neuroscience. Growing from a child to an adult gives rise to a significant deformation of the brain, but matching both brain objects requires topology preservation. Due to the complex topological structure of the brain white matter, ensuring foldover-free during the deformation is essential. Fig. 6 shows the prediction growth of the segmented white matter of a 5-year child brain toward that of an adult brain. The segmented brain models and the relevant landmarks by experts are available on the Internet Brain Segmentation Repository (www.cma.mgh.harvard.edu/ibsr). Since the landmark correspondence depends on the expert, i.e. keep-the-user-in-the-loop paradigm, we were only able to obtain the correspondence of a small number of landmarks. Because of the limited landmark pairs, some local distortions are inevitable. In order to give a convincing prediction, we need to reduce the unwanted distortion as much as possible. In Fig. 6, we therefore deformed these two brain white matter models separately in the opposite directions.

5.3. Application to biomechanics

In biomechanics, the dynamical behaviours are often analysed using a musculoskeletal model, where the forces, stresses and strains of the relevant tissues and organs are computed, because it is impossible to measure these quantities directly that occur inside the human body during an activity. One can first manually measure a set of markers placed on the skin and then retarget a standard human body anatomic model accordingly [13]. The resulting deformed model can be employed to estimate these quantities. Topology preservation is again a critical constraint for such a deformed model. In Fig. 7, we retargeted the anatomic model of an ordinary human body to a much fatter body model based on a set of markers on the skin. The original human body model is reconstructed from the Visual Human Dataset. For comparison, Fig. 7b and c show the results of applying the non-iterative RBF scheme and our iterative approach respectively. It can be noted that our algorithm can effectively prevent the unexpected intersections of different tissues. Fig. 7b only shows the intersection of the muscles and the skin surface. Removing the skin layer, one can find the more unexpected intersections between muscles as shown in Fig. 7e.

Moreover, to simulate the kinematics of body limbs, muscles (or soft tissues) and their deformed shapes must be taken into account. For example, simulating the elbow bending and twisting, we need to calculate the joint angles, forces and moments. Such calculation is particularly sensitive to the deformed shapes of the muscles close to the joint rotation centres. Since many muscles (or soft tissues) wrap around more than one bone, any movement must avoid bone-muscle penetration [6,15]. Simulating such activities without taking care of the anatomical structure tends to result in unexpected intersections. Fig. 8 shows the results of bending the elbow (around 75°) and turning the forearm upward (around 70°) by our iterative algorithm. For comparison, we also performed the conventional non-iterative RBF scheme here. The intersections of tissues are indicated by the arrows. In this experiment, the movement of the forearm's skeleton is given in advance. The constrained points are placed on the bones. (For more details, refer to the video demos in the Supplementary Material.)

Fig. 9 shows an application of 3D muscle modelling for the ankle joint. [12] applied the musculoskeletal model to the biomechanical analysis of lateral ankle ligaments. They used a motion capture device to capture the left-legged jump shots of an athlete for the ankle joint parameter estimation. However, they did not further reconstruct the 3D models of muscles. [1] emphasised that the muscle parameters for musculoskeletal modelling, e.g. cross-sectional area and moment arm data, should be obtained based on 3D models of muscles. To this end, they extracted the 3D models of muscles from a set of the quantitative CT scans manually. In Fig. 9, we simulated the jumping/landing action for the left leg. 3D muscle modelling is performed by our shape deformation approach.

All the experiments were undertaken using Matlab on an Intel Pentium 4 3.2 GHz PC with 1 Gbyte of RAM. Although the code is far from optimised, because our method has a low computation complexity as discussed earlier, it is very fast for 2D meshes and 3D polyhedrons.

6. Conclusion and future work

In this paper, we have incorporated the high dimensional deformable fields into the deformation of high-dimensional datasets, so that the deformed datasets both satisfy the positional constraints and preserve the topology. This is useful for the deformation of medical volume data and tetrahedral meshes due to its ability to handle the geometric complexity of such datasets. The RBF scheme maintains many good numerical properties, such as smoothness, flexibility and low degrees of freedom. But such traditional transformations are not invertible in general. Thus, they cannot preserve the topology during deformation. Our main contribution is to design the RBF-based functions that are diffeomorphic in an iterative way. We have also demonstrated that the proposed iterative mechanism can be extended to incorporate other deformation schemes, e.g. B-spline FFDs.

In our implementation, the thin-plate spline (TSP) is used as the radial basis function. TSP offers global support but not compact support. Compact support has the advantage of reducing the computational cost and is fast for optimisation. Thus, introducing the compact support RBF into our approach will be one of our future objectives. In addition, we also note that the RBF scheme does not work well when dealing with large bending deformations. The current strategy is to add several intermediate states of the constrained points, which works fine. This motives us to seek more powerful functions to overcome this weakness in our future work.

7. Acknowledgement

This project is partially supported by the following research grants: National Science Council (contracts NSC-100-2628-E-006-031-MY3 and NSC-100-2221-E-006-188-MY3), Taiwan; Department of Business, Innovation and Skills of the British Government (Sino-UK Higher Education Research Partnership for PhD studies, 2012).

Appendix A

We rewrite Eq. (1) here, such that the displacements linearly depend on the constraint points. To this end, the RBF coefficients (λ_i, a_i) can be computed by the following linear system, where $\lambda_i = (\lambda_{i1}, \ldots, \lambda_{im})^T, a_i = (a_{i0}, \ldots, a_{in})^T$,

$$\begin{pmatrix} \lambda_i \\ a_i \end{pmatrix} = K^{-1} \begin{pmatrix} \Delta c_i \\ 0 \end{pmatrix}, \quad i = 1, \ldots, n. \tag{A1}$$

Herein, there are m given constraint points. The displacements of all the constraint points are denoted as $\Delta c_i = (\Delta x_i^1, \ldots, \Delta x_i^m)^T$. K is a symmetric matrix filled with the radial basis functions $\phi_{kj} = \phi(\|C_k - C_j\|)$, $k, j = 1, \ldots, m$ and the constraint points' coordinates. (For a detailed RBF representation, refer to [4].)

Substituting (λ_i, a_i) of Eq. (A1) into Eq. (1), we therefore obtain a new expression of Eq. (1) as follows, $i = 1, \ldots, n$,

$$\begin{cases} \Delta u_i = M(X)K^{-1}\begin{pmatrix}\Delta c_i\\ 0\end{pmatrix} \\ M(X) = (\phi(\|X - C_1\|), \ldots, \phi(\|X - C_m\|), x_1, \ldots, x_n, 1) \end{cases} \tag{A2}$$

Note that Eq. (A2) describes a linear system that solves the displacement of any point X (i.e. $\{\Delta u_i\}$) using the displacements of the constraint points, (i.e. $\{\Delta c_i\}$). The kernel is $M(X)K^{-1}$ that describes the current configuration of the constraint points. The displacements $\{\Delta c_i\}$ indicate the directions that the constraint points are moving in. The derivatives of $\partial(u_1, \ldots, u_n)/\partial(x_1, \ldots, x_n)$ are computed as follows,

$$\frac{\partial u_i}{\partial x_j} = \begin{cases} 1 + M_{x_j}K^{-1}\begin{pmatrix}\Delta c_i\\ 0\end{pmatrix} & i = j \\ M_{x_j}K^{-1}\begin{pmatrix}\Delta c_i\\ 0\end{pmatrix} & i \neq j, i, j = 1, \ldots, n \end{cases}, \tag{A3}$$

where M_{x_j} denote the partial derivatives of $M(X)$.

Moreover, substituting the derivatives of Eq. (A3) into Eq. (3) yields,

$$1 + M_{x_i}K^{-1}\begin{pmatrix}\Delta c_i\\ 0\end{pmatrix} > \sum_{j=1, j\neq i}^{n}\left|M_{x_j}K^{-1}\begin{pmatrix}\Delta c_i\\ 0\end{pmatrix}\right|, \quad i = 1, \ldots, n. \tag{A4}$$

In general, the displacements of the constraint points $\{\Delta c_i\}$ can be obtained by the differences of the current coordinates of the constraint points and their individual destinations' coordinates. To satisfy the above inequalities, we need to limit the length of each displacement vector by scaling the vectors $\{\Delta c_i\}$ as follows,

$$1 + M_{x_i}K^{-1}\begin{pmatrix}\Delta c_i\\ 0\end{pmatrix}\delta > \sum_{j=1, j\neq i}^{n}\left|M_{x_j}K^{-1}\begin{pmatrix}\Delta c_i\\ 0\end{pmatrix}\right|\delta, \tag{A5}$$

where δ denotes a scaling factor and $\delta > 0$.

The regions in R^n defined by the inequalities of Eq. (A5) can be further described as follows, $i = 1, \ldots, n$,

$$\Omega_i(\delta) = \left\{ \begin{array}{l} (r_1, \ldots, 1 + r_i, \ldots, r_n) : |r_j| \leqslant \left|M_{x_j}K^{-1}\begin{pmatrix}\Delta c_i\\ 0\end{pmatrix}\right|\delta, j = 1, \ldots, n \\ 1 + r_i > \sum_{j=1, j\neq i}^{n} |r_j| \end{array} \right\}, \tag{A6}$$

where $(r_1, \ldots, 1 + r_i, \ldots, r_n)$ denotes a point in $\Omega_i(\delta)$. These regions are essentially of the ranges of the corresponding 1st order partial derivatives.

Appendix B

According to the 2D cubic B-spline function defined in Eq. (7), let $X = (x_1, x_2)^T$, $U = (u_1, u_2)^T$ and $C_{ij} = \left(c_1^{ij}, c_2^{ij}\right)^T$. The relevant derivatives can be given as follows,

$$\frac{\partial u_1}{\partial x_1} = 1 + B_w(\Delta c_1)\frac{1}{B_w(c_1)} + B_v(\Delta c_1)\frac{1}{B_v(c_1)},$$

$$\frac{\partial u_1}{\partial x_2} = B_w(\Delta c_1)\frac{1}{B_w(c_2)} + B_v(\Delta c_1)\frac{1}{B_v(c_2)},$$

$$\frac{\partial u_2}{\partial x_1} = B_w(\Delta c_2)\frac{1}{B_w(c_1)} + B_v(\Delta c_2)\frac{1}{B_v(c_1)},$$

$$\frac{\partial u_2}{\partial x_2} = 1 + B_w(\Delta c_2)\frac{1}{B_w(c_2)} + B_v(\Delta c_2)\frac{1}{B_v(c_2)},$$

where

$$B_w(\Delta c_1) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i' B_j \Delta c_1^{ij}$$

$$B_w(c_1) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i' B_j c_1^{ij}$$

$$B_v(\Delta c_1) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i B_j' \Delta c_1^{ij}$$

$$B_v(c_1) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i B_j' c_1^{ij}$$

$$B_w(\Delta c_2) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i' B_j \Delta c_2^{ij}$$

$$B_w(c_2) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i' B_j c_2^{ij}$$

$$B_v(\Delta c_2) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i B_j' \Delta c_2^{ij}$$

$$B_v(c_2) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i B_j' c_2^{ij}.$$

Moreover, in terms of Eqs. (5) and (6), introducing the scaling factor δ yields,

$$\delta = \min\left\{ \begin{array}{l} \frac{1}{\sqrt{2}}\left(\begin{array}{c} \left(\frac{B_w(\Delta c_1)}{B_w(c_1)} + \frac{B_v(\Delta c_1)}{B_v(c_1)}\right)^2 + \\ \left(\frac{B_w(\Delta c_1)}{B_w(c_2)} + \frac{B_v(\Delta c_1)}{B_v(c_2)}\right)^2 \end{array} \right)^{-1/2}, \\ \frac{1}{\sqrt{2}}\left(\begin{array}{c} \left(\frac{B_w(\Delta c_2)}{B_w(c_1)} + \frac{B_v(\Delta c_2)}{B_v(c_1)}\right)^2 + \\ \left(\frac{B_w(\Delta c_2)}{B_w(c_2)} + \frac{B_v(\Delta c_2)}{B_v(c_2)}\right)^2 \end{array} \right)^{-1/2} \end{array} \right\}.$$

Appendix C. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jbi.2013.12.011.

References

[1] Anderson DE, D'Agostino JM, Bruno AG, Manoharan RK, Bouxsein ML. Regressions for estimating muscle parameters in the thoracic and lumbar trunk for use in musculoskeletal modeling. J Biomech 2012;45(1):66–75.

[2] Ashburner J, Friston KJ. Voxel-based morphometry—the methods. Neuroimage 2000;11(6):805–21.

[3] Beg MF, Miller MI, Trouve A, Younes L. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. Int J Comput Vision 2005;61(2):139–57.

[4] Carr JC, Beatson RK, Cherrie JB, Mitchell TJ, Fright WR, McCallum BC, et al. Reconstruction and representation of 3D objects with radial basis functions. In: Proc of ACM SIGGRAPH'01, Los Angeles, CA; 2001. p. 67–76.

[5] Cai J, Chang Z, Wang Z, Segars WP, Yin FF. Four-dimensional magnetic resonance imaging (4D-MRI) using image-based respiratory surrogate: a feasibility study. Med Phys 2011;38(12):6384–94.

[6] Charlton IW, Johnson GR. Application of spherical and cylindrical wrapping algorithms in a musculoskeletal model of the upper limb. J Biomech 2001;34(9):1209–16.

[7] Fujimura K, Makarov M. Foldover-free image warping. Graph Models Image Process 1998;60(2):100–11.

[8] Gain J, Bechmann D. A survey of spatial deformation from a user-centered perspective. ACM Trans Graph 2008;27(4):107-1–107-21.

[9] Guo Y, Wang J, Sun H, Cui X, Peng Q. A novel constrained texture mapping method based on harmonic map. Comput Graph 2005;29(6):972–9.

[10] Holden M. A review of geometric transformations for nonrigid body registration. IEEE Trans Med Imaging 2008;27(1):111–28.

[11] Joshi SC, Miller MI. Landmark matching via large deformation diffeomorphisms. IEEE Trans Image Process 2000;9(8):1357–70.

[12] Lindner M, Kotschwar A, Zsoldos RR, Groesel M, Peham C. The jump shot – a biomechanical analysis focused on lateral ankle ligaments. J Biomech 2012;45(1):202–6.

[13] Nedel LP, Thalmann D. Anatomic modeling of deformable human bodies. Vis Comput 2000;16(6):306–21.

[14] Nealen A, Sorkine O, Alexa M, Cohen-Or D. A sketch-based interface for detail-preserving mesh editing. ACM Trans Graph 2005;24(3):1142–7.

[15] Pennestrì E, Stefanelli R, Valentini PP, Vita L. Virtual musculo-skeletal model for the biomechanical analysis of the upper limb. J Biomech 2007;40(6):1350–61.

[16] Rau G, Disselhorst-Klug C, Schmidt R. Movement biomechanics goes upwards: from the leg to the arm. J Biomech 2000;33(10):1207–16.

[17] Rueckert D, Sonoda LI, Hayes C, Hill DLG, Leach MO, Hawkes DJ. Nonrigid registration using free-form deformations: application to breast MR images. IEEE Trans Med Imaging 1999;18(8):712–21.

[18] Song W, Liu L. Stretch-based tetrahedral mesh manipulation. In: Proc graphics interface, Montreal, Canada; 2007. p. 319–25.

[19] Thevenaz P, Blu T, Unser M. Interpolation revisited. IEEE Trans Med Imaging 2000;19(7):739–58.

[20] Tang Y, Wang J, Bao HJ, Peng QS. RBF-based constrained texture mapping. Comput Graph 2003;27(3):415–22.

[21] Wu YT, Kanade T, Li CC, Cohn J. Image registration using wavelet-based motion mode. Int J Comput Vision 2000;38(2):129–52.

[22] Xue Z, Shen D, Davatzikos C. Statistical representation of high-dimensional deformation fields with application to statistically constrained 3D warping. Med Image Anal 2006;10(5):740–51.

[23] Zagorchev L, Goshtasby A. A comparative study of transformation functions for nonrigid image registration. IEEE Trans Image Process 2006;15(3):529–38.

[24] Borrel P, Rappoport A. Simple constrained deformations for geometric modelling and interactive design. ACM Trans Graph 1994;13(2):137–55.

[25] Angelidis A, Cani MP, Wyvill G, King S. Swirling-sweepers: constant-volume modeling. In: Proc of 12th Pacific conf on computer graphics and applications; 2004. p. 10–15.

[26] Gain J, Marais P. Warp Sculpting. IEEE Trans Visual Comput Graph 2005;11(2):217–27.

[27] Mason D, Wyvill G. Blendeforming: ray traceable localized foldover-free space deformation. In: Proc of computer graphic international'01; 2001. p. 183–90.

[28] von Funck W, Theisel H, Seidel HP. Vector field based shape deformations. ACM Trans Graph 2006;25(3):1118–25.

[29] Zhou K, Huang J, Snyder J, Liu X, Bao H, Guo B, et al. Large mesh deformation using the volumetric graph Laplacian. ACM Trans Graph 2005;24(3):496–503.

[30] Sdika M. A fast nonrigid image registration with constraints on the Jacobian using large scale constrained optimization. IEEE Trans Med Imaging 2008;27(2):271–81.