Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

Constructing B-spline solids from tetrahedral meshes for isogeometric analysis

Hongwei Lin[a,*], Sinan Jin[a], Qianqian Hu[b], Zhenbao Liu[c]

[a] *Department of Mathematics, State Key Lab. of CAD&CG, Zhejiang University, Hangzhou, 310027, China*
[b] *Department of Mathematics, Zhejiang Gongshang University, Hangzhou, 310018, China*
[c] *School of Aeronautics, Northwestern Polytechnical University, Xi'an, 710072, China*

ARTICLE INFO

Article history:
Available online 20 March 2015

Keywords:
Isogeometric analysis
Volume parameterization
B-spline solid
Iterative fitting

ABSTRACT

With the advent of isogeometric analysis, the modeling of spline solids became an important topic. In this paper, we present a discrete volume parameterization method for tetrahedral (tet) mesh models and an iterative fitting algorithm with a B-spline solid. The discrete volume parameterization method maps the vertices of a tet mesh into a parameter domain by solving a system of linear equations. Each equation is explicitly constructed for an inner vertex in terms of the geometric information adjacent to the inner vertex. Moreover, we show the validity of the parameterization system of linear equations thus constructed. Next, because the number of tet mesh vertices is usually very large, we develop an iterative algorithm for fitting a tet mesh with a B-spline solid. The iterative algorithm exploits the geometric information of the control hexahedral (hex) mesh and the local support property of the spline function, so the total amount of computation in each iteration is unchanged when the number of control hex mesh vertices of the B-spline solid is increased. Therefore, the iterative fitting algorithm performs very well in incremental fitting of a tet mesh with a large number of vertices. Finally, four experimental examples presented in this paper show the efficiency and effectiveness of the developed algorithms.

1. Introduction

Although traditional finite element analysis (FEA) methods are usually based on linear basis functions, computer-aided design (CAD) models are represented by nonuniform rational basis spline (NURBS) with nonlinear NURBS basis functions. Therefore, when a CAD model simulation is performed using FEA methods, the NURBS-based model should be transformed into a linear mesh representation in a process called mesh transformation. The mesh transformation operation is very tedious and time-consuming, so it has become the bottleneck in FEA methods. To avoid mesh transformation and advance the seamless integration of CAD and computer-aided engineering, isogeometric analysis (IGA) was invented by Hughes et al. (2005). Unlike traditional FEA methods, IGA is based on the NURBS basis functions. Thus, the CAD models represented by NURBS can be directly analyzed by IGA without a tedious mesh transformation, greatly reducing the number of computations in the FEA procedure.

However, the advent of IGA brings a new problem for geometric design, i.e., the modeling of NURBS solids. Although solid analysis is an important issue in FEA, geometric design usually involves the creation of curves and surfaces, and lacks

[*] Corresponding author. Tel.: +86 571 87951860 8304; fax: +86 571 88206681.
E-mail address: hwlin@zju.edu.cn (H. Lin).

http://dx.doi.org/10.1016/j.cagd.2015.03.013
0167-8396/© 2015 Elsevier B.V. All rights reserved.

methods of generating NURBS solids. Therefore, the invention of efficient and convenient methods for creating NURBS solids for IGA becomes an urgent task for geometric design. It is noted that traditional methods generate spline solids usually by filling the inside of given triangular mesh models (Wang et al., 2013; Wang and Qian, 2014). However, the generation of spline solids by *fitting* tetrahedral (tet) mesh vertices is much easier than that by *filling* the inside of triangular meshes. Moreover, the tet meshes are easy to produce by some well-developed softwares, such as TetGen, NetGen, etc. In addition, since tet meshes are widely employed in FEA for solid analysis, many tet mesh models currently exist. Therefore, fitting tet mesh model with NURBS solids is a feasible and convenient method of modeling NURBS solids.

Given a tet mesh model, in this paper, we develop a discrete parameterization formula for parameterizing the tet mesh vertices. As in the widely employed parameterization methods for a triangular mesh, each parameterization formula is constructed for each inner vertex of the tet mesh, and all of these parameterization formulae constitute a system of linear equations. After the parameter values of the boundary mesh vertices are assigned, the system of linear equations can be solved to generate the parameter values of the inner vertices. The proposed tet mesh parameterization method is equivalent to solving a boundary problem consisting of a Laplace equation, which simulates an isothermal field in the tet mesh, so the generated isoparametric surfaces (i.e., the isothermal surfaces) can be guaranteed to be strictly non-overlapping.

Moreover, because the number of the tet mesh vertices is usually very large, we propose an iterative method for fitting the tet mesh with a trivariate B-spline solid and show its convergence. The iterative fitting method starts with an initial B-spline solid and contains four steps in each iteration:

(1) Calculate the difference vector for each tet mesh vertex;
(2) Distribute the above vectors to the corresponding control vertices of the B-spline solid;
(3) Average the vectors distributed to each control point to produce the difference vector for the control point;
(4) Generate the new control point by adding the average difference vector to the corresponding control vertex.

These four steps are performed iteratively until the stop criterion is reached. In this way, the B-spline solid fitting the given tet mesh is generated. The proposed iterative fitting method fully exploits the local support property of the B-spline basis functions so that the amount of computation in each iteration is unchanged when the number of control points of the B-spline solid is increased. Therefore, the performance of the iterative fitting algorithm is desirable for incremental fitting of tet meshes with a large number of vertices.

The structure of this paper is as follows. In Section 1.1, some related work is briefly reviewed. Section 2 presents the discrete volume parameterization method, and Section 3 develops the iterative fitting algorithm. After some results are presented in Section 4, Section 5 concludes this paper.

1.1. Related work

In this section, we will briefly review related work on spline solid modeling, mesh parameterization, and iterative fitting.

Spline solid modeling: As stated above, geometric design mainly involves the modeling of curves and surfaces, so it lacks effective methods for spline solid modeling. To model the three-dimensional physical domain in IGA, NURBS (B-spline) solids, T-spline solids, and subdivision solids are developed. They are usually constructed from boundary-represented models, such as closed NURBS patches, and a closed triangular mesh model.

On the basis of the given boundary conditions and guiding curves, a NURBS solid that parameterizes a swept volume was generated by a variational approach (Aigner et al., 2009). In Xu et al. (2013), Wang and Qian (2014), optimization-based approaches were employed to construct trivariate B-spline solids with positive Jacobian values from boundary-represented models. On the other hand, to analyze arterial blood flow by IGA, a skeleton-based method was proposed to construct a NURBS solid (Zhang et al., 2007). Moreover, in Martin et al. (2009), a harmonic-function-based volumetric parameterization method was first performed, and then a trivariate B-spline solid was constructed.

Because of their flexibility for geometric modeling, T-spline solids were constructed for IGA. After a mesh untangling and smoothing procedure was applied, a T-spline solid was generated in Escobar et al. (2011) for fitting a genus-zero triangular mesh model. Moreover, a mapping-based rational trivariate solid T-spline construction method was developed in Zhang et al. (2012), also for genus-zero geometry, and starting from a boundary surface triangulation. Furthermore, in Wang et al. (2013), a method of constructing T-spline solids from boundary triangulations with an arbitrary genus topology was presented by polycube mapping.

Moreover, subdivision solids (Burkhart et al., 2010) are also employed in the IGA method to model the physical domain. Other recently developed spline solid representations include simplex splines (Hua et al., 2004) and polycube splines (Li et al., 2010; Wang et al., 2012).

Mesh parameterization: Mesh parameterization has been extensively studied for a triangular mesh in three dimensions with applications such as mesh fitting and texture mapping. A triangular mesh parameterization constructs a one-to-one mapping from the mesh in three dimensions to a planar domain. According to the various requirements of different applications, mapping methods such as discrete harmonic mapping (Eck et al., 1995; Pinkall and Polthier, 1993), convex combination mapping (Floater, 2003), or discrete conformal mapping (Hormann and Greiner, 2000; Gu et al., 2004; Gu and Yau, 2003) can be chosen. For more details on triangular mesh parameterization methods, please refer to Floater and Hormann (2005).

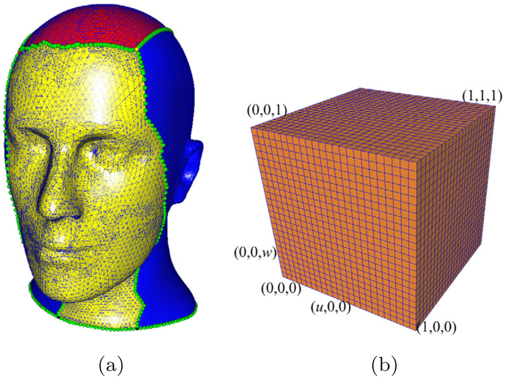

Fig. 1. An input triangular mesh model with six boundary mesh patches partitioned in advance (a) is parameterized into the parameter domain $[0, 1] \times [0, 1] \times [0, 1]$ (b).

In triangular mesh parameterization, the linear system for parameterization is usually established explicitly and geometrically by assigning a parameterization formula for each inner mesh vertex, and each formula corresponds to an equation of the linear system (Floater and Hormann, 2005). However, in spline solid construction, the discrete volume parameterization is always determined indirectly by solving a partial differential equation (Martin et al., 2009; Martin and Cohen, 2010), using the finite element method. Therefore, to employ the conventional volume parameterization methods, users must be familiar with the finite element method. In this paper, in an analogy to triangular mesh parameterization methods, we construct a parameterization formula for each inner vertex of a given tet mesh explicitly and geometrically, and all of these formulae constitute the linear parameterization system. Thus, the method developed in this paper makes the discrete volume parameterization as intuitive and easy to implement as the triangular mesh parameterization methods.

Iterative fitting: The progressive-iterative approximation (PIA), proposed in Lin et al. (2004, 2005), is an iterative fitting algorithm with explicit geometric meaning for curves and surfaces with totally positive basis functions. In Shi and Wang (2006), the PIA algorithm was proven to be convergent for NURBS, and the convergence rate was accelerated in Lu (2010). Moreover, a local format of the PIA iterative algorithm was developed in Lin (2010). Furthermore, the PIA format has also been extended to subdivision surface fitting (Cheng et al., 2009; Fan et al., 2008; Chen et al., 2008). It is worth mentioning that a PIA-like iterative algorithm was employed in the spline solid construction method developed in Martin et al. (2009).

The limit of the PIA algorithm mentioned above is that it interpolates the given data points. Some iterative fitting formats were presented recently whose limits approximate the given data points, such as extended PIA (Lin and Zhang, 2011), and least square PIA (Deng and Lin, 2014). In Lin and Zhang (2013), an efficient iterative fitting algorithm using T-spline was proposed for fitting large data set.

In this paper, an iterative method for fitting the tet mesh with a trivariate B-spline solid is developed. It should be noted that in the problem of tet mesh fitting, there are usually a large number of mesh vertices. In general, the linear system for fitting a tet mesh with a large number of vertices is usually ill-conditioned, or even singular (Pereyra and Scherer, 2003). Conventional methods cannot deal with the tet mesh fitting problem with a large number of vertices efficiently and effectively when it is ill-conditioned or singular. However, because the averaging iteration time of the iterative method developed in this paper is nearly constant, it can efficiently deal with such tet mesh fitting problem. Moreover, it is also robust enough to solve a singular linear system effectively.

2. Discrete parameterization of tetrahedral mesh

The input to our algorithm is a tet mesh model with six boundary triangular mesh patches partitioned in advance [see Fig. 1(a)]. The six boundary mesh patches intersect at twelve boundary curves and eight corners. The purpose of the discrete volume parameterization is to construct a mapping that maps the tet mesh model into the parametric space $[0, 1] \times [0, 1] \times [0, 1]$ [see Fig. 1(b)].

As illustrated in Fig. 1, before the discrete volume parameterization is performed, the parameter value at each vertex of the six boundary mesh patches should be assigned. We first choose one corner and designate its parameter values as $(0, 0, 0)$. The vertices at the three boundary curves adjacent to the chosen corner, each of which is a common boundary of two adjacent boundary mesh patches, are mapped to $[0, 1]$ using the normalized accumulated chord length method. Specifically, the parameter values of the vertices at the three boundary curves take the forms $(u_i, 0, 0)$, $(0, v_j, 0)$, and $(0, 0, w_k)$, respectively. Accordingly, the parameter values of the vertices at the other nine boundary curves can be calculated in a similar manner.

Next, the six boundary triangular meshes can be parameterized using the appropriate triangular mesh parameterization method. In our implementation, we take the method developed in Pinkall and Polthier (1993). In this way, the parameter

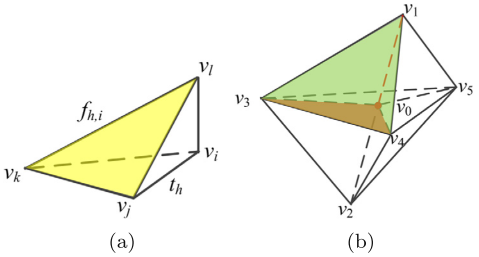

Fig. 2. (a) A tetrahedron t_h. (b) A tet mesh with an inner vertex v_0.

values of the boundary mesh vertices are determined. The next task is to calculate the parameter values at the inner vertices of the tet mesh.

The parameter values at the inner vertices of a tet mesh are determined by solving a linear system. The system consists of a set of parameterization formulae, each of which corresponds to an inner vertex, and can be constructed geometrically. In the following, we will explain the construction of the parameterization formula in detail.

In a tet t_h with vertices v_i, v_j, v_k, and v_l [refer to Fig. 2(a)], we denote the triangular face opposite to vertex v_i as $f_{h,i}$, the area of $f_{h,i}$ as $S_{h,i}$, and the volume of the tet t_h as V_h. Now, consider an inner mesh vertex v_i [e.g., the vertex v_0 in Fig. 2(b)]. Supposing the parameter value at v_i is u_i, and the parameter values at its adjacent vertex v_j are u_j, the *parameterization formula* corresponding to the inner vertex v_i can be constructed as

$$\Big(\sum_{\substack{\text{tet } t_k\\ \text{adj. to } v_i}} \frac{S_{k,i}^2}{9V_k}\Big)u_i + \sum_{\substack{\text{vertex } v_j\\ \text{adj. to } v_i}} \Big(\sum_{\substack{\text{tet } t_l\\ \text{adj. to}\\ \text{edge } v_i v_j}} \frac{S_{l,i}S_{l,j}}{9V_l}\cos\theta_{l,ij}\Big)u_j = 0, \tag{1}$$

where V_k is the volume of the tet t_k adjacent to the vertex v_i; $S_{k,i}$ is the area of the triangular face $f_{k,i}$ opposite to the vertex v_i in the tet t_k; V_l is the volume of the tet t_l adjacent to the mesh edge $v_i v_j$; $S_{l,i}$ and $S_{l,j}$ are the areas of the triangular faces $f_{l,i}$ and $f_{l,j}$ opposite to the vertices v_i and v_j in the tet t_l, respectively; and $\theta_{l,ij}$ is the dihedral angle of $f_{l,i}$ and $f_{l,j}$ in the tet t_l.

Take the inner vertex v_0 illustrated in Fig. 2(b) as an example. The vertex v_0 is adjacent to six tets, $t_i, i = 0, 1, \cdots, 5$, where t_0, t_1, and t_2 represent the tets $v_0 - v_1v_3v_4$, $v_0 - v_1v_3v_5$, and $v_0 - v_1v_4v_5$, respectively, adjacent to the mesh edge v_0v_1. Therefore, the coefficient prior to the parameter value u_0 at the vertex v_0 is $\sum_{k=0}^{5} \frac{S_{k,0}^2}{9V_k}$, the coefficient prior to u_1 is $\sum_{l=0}^{2} \frac{S_{l,0}S_{l,1}}{9V_l} \cos\theta_{l,01}$, and so on.

A linear system can be generated by combining the parameterization formulae (1), and the parameter values at the inner vertices can be obtained by solving the system of linear equations. The linear system is solved three times for the parameters u, v, and w, respectively. In this way, each inner mesh vertex is assigned parameter values (u_i, v_i, w_i). In the following, we will show the validity of the linear system consisting of the parameterization formulae (1).

2.1. Proof of the parameterization formulae

In this section, we will show that the linear system consisting of the parameterization formulae (1) is equivalent to that generated by solving the following boundary value problem using the FEA method (Zienkiewicz et al., 2008).

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} + \dfrac{\partial^2 u}{\partial z^2} = 0, \quad (x, y, z) \in \Omega, \\ u|_{\partial\Omega} = \varphi(x, y, z). \end{cases} \tag{2}$$

Thus, the validity of the parameterization formulae (1) is guaranteed by the theory of FEA. In fact, FEA is applied to seek the function $u(x, y, z)$ that minimizes the weak form of the boundary value problem (2), i.e.,

$$W(u(x, y, z)) = \iiint\limits_{\Omega} \left(\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial u}{\partial z}\right)^2 \right) dxdydz. \tag{3}$$

In our problem, the tet mesh model is the physical domain Ω of the problem (2). The tet mesh presents a natural triangulation for FEA, where each element is a tet. Accordingly, Eq. (3) changes to the sum of the integrals at the tet t_h,

$$W(u(x, y, z)) = \sum_h W_h(u(x, y, z)) = \sum_h \iiint\limits_{t_h} \left(\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial u}{\partial z}\right)^2 \right) dxdydz.$$

In an individual element, i.e., a tet t_h [refer to Fig. 2(a)], the shape functions λ_i, λ_j, λ_k, and λ_l for the four vertices $v_i = (x_i, y_i, z_i)$, $v_j = (x_j, y_j, z_j)$, $v_k = (x_k, y_k, z_k)$, and $v_l = (x_l, v_l, z_l)$, respectively, are taken as the barycentric coordinates (Zienkiewicz et al., 2008), that is,

$$\lambda_i(x,y,z) = \frac{\begin{vmatrix} x-x_j & y-y_j & z-z_j \\ x_k-x_j & y_k-y_j & z_k-z_j \\ x_l-x_j & y_l-y_j & z_l-z_j \end{vmatrix}}{6V_h}, \quad \lambda_j(x,y,z) = \frac{\begin{vmatrix} x-x_i & y-y_i & z-z_i \\ x_l-x_i & y_l-y_i & z_l-z_i \\ x_k-x_i & y_k-y_i & z_k-z_i \end{vmatrix}}{6V_h},$$

$$\lambda_k(x,y,z) = \frac{\begin{vmatrix} x-x_i & y-y_i & z-z_i \\ x_j-x_i & y_j-y_i & z_j-z_i \\ x_l-x_i & y_l-y_i & z_l-z_i \end{vmatrix}}{6V_h}, \quad \lambda_l(x,y,z) = \frac{\begin{vmatrix} x-x_i & y-y_i & z-z_i \\ x_k-x_i & y_k-y_i & z_k-z_i \\ x_j-x_i & y_j-y_i & z_j-z_i \end{vmatrix}}{6V_h}, \tag{4}$$

where V_h is the volume of the tet t_h in Fig. 2(a). Therefore, in each tet, the unknown function $u(x, y, z)$ can be approximated as,

$$u^{(h)}(x,y,z) = \lambda_i(x,y,z)u_i + \lambda_j(x,y,z)u_j + \lambda_k(x,y,z)u_k + \lambda_l(x,y,z)u_l, \tag{5}$$

where u_i, u_j, u_k, and u_l are the values of the unknown function $u(x, y, z)$ at v_i, v_j, v_k, and v_l, respectively.

Because the shape functions (4) are all linear, the partial derivatives of $\lambda_i, \lambda_j, \lambda_k$, and λ_l with respect to x, y, z, are all constants. Substituting $u^{(h)}$ into the integral at the tet t_h, we have

$$\begin{aligned} W_h(u(x,y,z)) \approx W_h(u^{(h)}(x,y,z)) &= \iiint\limits_{t_h} \left(\left(\frac{\partial u^{(h)}}{\partial x}\right)^2 + \left(\frac{\partial u^{(h)}}{\partial y}\right)^2 + \left(\frac{\partial u^{(h)}}{\partial z}\right)^2 \right) dxdydz \\ &= [\, u_i \quad u_j \quad u_k \quad u_l \,]\, K_h \begin{bmatrix} u_i \\ u_j \\ u_k \\ u_l \end{bmatrix}, \end{aligned}$$

where

$$K_h = V_h \sum_{r=x,y,z} \begin{bmatrix} (\frac{\partial \lambda_i}{\partial r})^2 & \frac{\partial \lambda_i}{\partial r}\frac{\partial \lambda_j}{\partial r} & \frac{\partial \lambda_i}{\partial r}\frac{\partial \lambda_k}{\partial r} & \frac{\partial \lambda_i}{\partial r}\frac{\partial \lambda_l}{\partial r} \\ \frac{\partial \lambda_j}{\partial r}\frac{\partial \lambda_i}{\partial r} & (\frac{\partial \lambda_j}{\partial r})^2 & \frac{\partial \lambda_j}{\partial r}\frac{\partial \lambda_k}{\partial r} & \frac{\partial \lambda_j}{\partial r}\frac{\partial \lambda_l}{\partial r} \\ \frac{\partial \lambda_k}{\partial r}\frac{\partial \lambda_i}{\partial r} & \frac{\partial \lambda_k}{\partial r}\frac{\partial \lambda_j}{\partial r} & (\frac{\partial \lambda_k}{\partial r})^2 & \frac{\partial \lambda_k}{\partial r}\frac{\partial \lambda_l}{\partial r} \\ \frac{\partial \lambda_l}{\partial r}\frac{\partial \lambda_i}{\partial r} & \frac{\partial \lambda_l}{\partial r}\frac{\partial \lambda_j}{\partial r} & \frac{\partial \lambda_l}{\partial r}\frac{\partial \lambda_k}{\partial r} & (\frac{\partial \lambda_l}{\partial r})^2 \end{bmatrix} \tag{6}$$

is called the *element stiffness matrix* of the tet t_h.

Denoting $\vec{i}, \vec{j}, \vec{k}$ as the unit vectors of the x, y, and z axes, respectively, $S_{h,i}$ as the area of the triangular face $f_{h,i}$, and $\vec{n}^0_{h,i}$ as the unit normal vector of $f_{h,i}$ [see Fig. 2(a)], we have

$$(\frac{\partial \lambda_i}{\partial x}, \frac{\partial \lambda_i}{\partial y}, \frac{\partial \lambda_i}{\partial z}) = \frac{1}{6V_h} \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ x_k-x_j & y_k-y_j & z_k-z_j \\ x_l-x_j & y_l-y_j & z_l-z_j \end{vmatrix} = \frac{1}{6V_h}\overrightarrow{v_k v_j} \times \overrightarrow{v_l v_j} = \frac{S_{h,i}}{3V_h}\vec{n}^0_{h,i}.$$

Therefore, the element stiffness matrix K_h (6) can be written as

$$\begin{bmatrix} \frac{S^2_{h,i}}{9V_h} & \frac{S_{h,i}S_{h,j}}{9V_h}c_{h,ij} & \frac{S_{h,i}S_{h,k}}{9V_h}c_{h,ik} & \frac{S_{h,i}S_{h,l}}{9V_h}c_{h,il} \\ \frac{S_{h,j}S_{h,i}}{9V_h}c_{h,ji} & \frac{S^2_{h,j}}{9V_h} & \frac{S_{h,j}S_{h,k}}{9V_h}c_{h,jk} & \frac{S_{h,j}S_{h,l}}{9V_h}c_{h,jl} \\ \frac{S_{k,l}S_{h,i}}{9V_h}c_{h,ki} & \frac{S_{h,k}S_{h,j}}{9V_h}c_{h,kj} & \frac{S^2_{h,k}}{9V_h} & \frac{S_{h,k}S_{h,l}}{9V_h}c_{h,kl} \\ \frac{S_{h,l}S_{h,i}}{9V_h}c_{h,li} & \frac{S_{h,l}S_{h,j}}{9V_h}c_{h,lj} & \frac{S_{h,l}S_{h,k}}{9V_h}c_{h,lk} & \frac{S^2_{h,l}}{9V_h} \end{bmatrix}, \tag{7}$$

where $c_{h,ij}$ denotes $\cos\theta_{h,ij}$, $c_{h,ik}$ denotes $\cos\theta_{h,ik}$, and so on.

Moreover, the global stiffness matrix K can be constructed by adding all of the element stiffness matrices according to the subscripts of their elements. For example, the element $\frac{S_{h,i}S_{h,j}}{9V_h}\cos\theta_{h,ij}$ in K_h [Eq. (7)] is added to the (i, j) element of the global matrix K. Therefore, K can be represented as $K = [k_{ij}]$ with

$$k_{ij} = \begin{cases} \displaystyle\sum_{\substack{\text{for each tet } t_k \\ \text{adjacent to } v_i}} \frac{S_{k,i}^2}{9V_k}, & \text{if } i = j, \\ \displaystyle\sum_{\substack{\text{for each tet } t_l \\ \text{adjacent to} \\ \text{the edge } v_i v_j}} \frac{S_{l,i}S_{l,j}}{9V_l}\cos\theta_{l,ij}, & \text{if } v_i \text{ is adjacent to } v_j, \\ 0, & \text{if } v_i \text{ is not adjacent to } v_j. \end{cases} \tag{8}$$

Further, the weak form (3) can be represented approximately as

$$W(u(x, y, z)) = \sum_h W_h(u(x, y, z)) \approx \sum_h W_h(u^{(h)}(x, y, z)) = U^T KU. \tag{9}$$

Here $U = [u_0, u_1, \cdots, u_n]^T$ are the parameter values at the vertices of the tet mesh model. The solution U that minimizes the weak form (9) can be found by solving the linear system

$$KU = 0. \tag{10}$$

Note that the parameter values at the boundary mesh vertices have been assigned. Thus, the equations corresponding to the boundary mesh vertices should be deleted from the linear system (10), which yields

$$\bar{K}U = 0. \tag{11}$$

Clearly, each equation in the linear system (11) is the parameterization formula (1) corresponding to an inner mesh vertex.

In conclusion, the linear system consisting of the parameterization formulae (1) is just the system (11) generated by FEA, so the validity of the discrete parameterization method developed in this paper is guaranteed by the theory of FEA.

3. Iterative fitting with B-spline solid

In this section, we will present the iterative algorithm for fitting the tet mesh model with a trivariate B-spline solid and show its convergence. Suppose the tet mesh model has $n + 1$ vertices $\boldsymbol{Q}_l,\ l = 0, 1, \cdots, n$

3.1. Construction of the initial B-spline solid

The iterative fitting algorithm starts with an initial B-spline solid, which is constructed as follows.

In Section 2, the tet mesh model was parameterized into the parameter domain $[u_s, u_e] \times [v_s, v_e] \times [w_s, w_e]$. We uniformly sample $n_u + 1$ values, $u_i,\ i = 0, 1, \cdots, n_u$ in $[u_s, u_e]$; $n_v + 1$ values, $v_j,\ j = 0, 1, \cdots, n_v$ in $[v_s, v_e]$; and $n_w + 1$ values, $w_k,\ k = 0, 1, \cdots, n_w$ in $[w_s, w_e]$, all including the end values of each interval. The sampling points $(u_i, v_j, w_k), i = 0, 1, \cdots, n_u, j = 0, 1, \cdots, n_v, k = 0, 1, \cdots, n_w$ constitute a grid in the parameter domain. By inverse parameterization mapping, these sampling points in the parameter domain can be mapped into the tet mesh model, generating a set of points $\{\boldsymbol{P}_{ijk}^{(0)}\}$. The points in the set $\{\boldsymbol{P}_{ijk}^{(0)}\}$ in the tet mesh are connected according to the adjacent relationship of the grid in the parameter domain, forming the control grid of the initial B-spline solid. In our implementation, the knot vectors of the initial B-spline solid are chosen as uniform vectors with Bézier end conditions.

To make the initial B-spline solid as desirable as possible, the sampling numbers $n_u,\ n_v,\ n_w$ are made proportional to the length of the boundary curves of the tet mesh model. Suppose the average length of the four u-directional boundary curves of the tet mesh model is L_u, where the parameter values of the points on the u-directional boundary curves have the form $(0, u, 0),\ (0, u, 1),\ (1, u, 0),\ (1, u, 1)$, respectively; the average length of the four v-directional boundary curves is L_v; the average length of the four w-directional boundary curves is L_w. In addition, suppose the number of vertices in the tet mesh is $n + 1$. We first solve the constrained optimization problem,

$$\begin{aligned} &\min_{M_u, M_v, M_w} \left((\frac{M_u}{L_u} - \frac{M_v}{L_v})^2 + (\frac{M_v}{L_v} - \frac{M_w}{L_w})^2 \right) \\ &s.t. \quad M_u M_v M_w = \frac{n+1}{C}, \end{aligned} \tag{12}$$

where C is a constant distinct in different examples. Then, the sampling numbers n_u, n_v, n_w can be determined by rounding M_u, M_v, M_w, respectively.

3.2. Iterative fitting algorithm

Starting with the initial B-spline solid constructed in Section 3.1, suppose the iteration has been performed for m steps, and the mth B-spline solid $\boldsymbol{P}^{(m)}(u,v,w)$ is generated:

$$\boldsymbol{P}^{(m)}(u,v,w)=\sum_{i=0}^{n_u}\sum_{j=0}^{n_v}\sum_{k=0}^{n_w}\boldsymbol{P}_{ijk}^{(m)}B_i(u)B_j(v)B_k(w).$$

One iteration step for producing $\boldsymbol{P}^{(m+1)}(u,v,w)$ from $\boldsymbol{P}^{(m)}(u,v,w)$ includes the following operations.

First, the *difference vector for each data point* is calculated:

$$\boldsymbol{\delta}_l^{(m)}=\boldsymbol{Q}_l-\boldsymbol{P}^{(m)}(u_l,v_l,w_l). \tag{13}$$

Each vector $\boldsymbol{\delta}_l^{(m)}$ is distributed to the control points $\boldsymbol{P}_{ijk}^{(m)}$, whose corresponding basis function is $B_i(u_l)B_j(v_l)B_k(w_l)\neq 0$, in the form $B_i(u_l)B_j(v_l)B_k(w_l)\boldsymbol{\delta}_l^{(m)}$. In this way, a control point $\boldsymbol{P}_{ijk}^{(m)}$ may be distributed to several weighted vectors. All of these vectors are averaged, forming the *difference vector for the control point* $\boldsymbol{P}_{ijk}^{(m)}$,

$$\boldsymbol{\Delta}_{ijk}^{(m)}=\frac{\sum_{l\in I_{ijk}}B_i(u_l)B_j(v_l)B_k(w_l)\boldsymbol{\delta}_l^{(m)}}{\sum_{l\in I_{ijk}}B_i(u_l)B_j(v_l)B_k(w_l)}, \tag{14}$$

where I_{ijk} is the set of indices l such that $B_i(u_l)B_j(v_l)B_k(w_l)\neq 0$.

Next, the difference vector $\boldsymbol{\Delta}_{ijk}^{(m)}$ is added to the control point $\boldsymbol{P}_{ijk}^{(m)}$, forming the control point $\boldsymbol{P}_{ijk}^{(m+1)}$ for the $(m+1)$th solid, i.e.,

$$\boldsymbol{P}_{ijk}^{(m+1)}=\boldsymbol{P}_{ijk}^{(m)}+\boldsymbol{\Delta}_{ijk}^{(m)}. \tag{15}$$

Thus, the $(m+1)$th B-spline solid $\boldsymbol{P}^{(m+1)}(u,v,w)$ is generated:

$$\boldsymbol{P}^{(m+1)}(u,v,w)=\sum_{i=0}^{n_u}\sum_{j=0}^{n_v}\sum_{k=0}^{n_w}\boldsymbol{P}_{ijk}^{(m+1)}B_i(u)B_j(v)B_k(w). \tag{16}$$

These operations are performed iteratively until the termination condition is reached, i.e.,

$$\left|\frac{\sum_l\left\|\delta_l^{(m+1)}\right\|^2}{\sum_l\left\|\delta_l^{(m)}\right\|^2}-1\right|<\varepsilon. \tag{17}$$

In our implementation, ε is taken as 10^{-3}.

3.3. Convergence analysis

In this section, we will show the convergence of the iterative algorithm developed in Section 3.2.

Arrange the difference vectors $\boldsymbol{\Delta}_{ijk}^{(m)}$ into a sequence $\Delta^{(m)}$ according to the lexicographic order, i.e.,

$$\Delta^{(m)}=[\boldsymbol{\Delta}_{000}^{(m)},\boldsymbol{\Delta}_{001}^{(m)},\cdots,\boldsymbol{\Delta}_{n_u,n_v,n_w}^{(m)}]^T.$$

Denote

$$B_{ijk}(u,v,w)=B_i(u)B_j(v)B_k(w),\ \text{and,}\ \boldsymbol{\tau}_l=(u_l,v_l,w_l).$$

Because,

$$\begin{aligned}\boldsymbol{\Delta}_{ijk}^{(m+1)}&=\frac{\sum_{l\in I_{ijk}}\boldsymbol{\delta}_l^{(m+1)}B_{ijk}(\boldsymbol{\tau}_l)}{\sum_{l\in I_{ijk}}B_{ijk}(\boldsymbol{\tau}_l)}=\frac{\sum_{l\in I_{ijk}}B_{ijk}(\boldsymbol{\tau}_l)(\boldsymbol{Q}_l-\sum_{i,j,k}(\boldsymbol{P}_{ijk}^{(m)}+\boldsymbol{\Delta}_{ijk}^{(m)})B_{ijk}(\boldsymbol{\tau}_l))}{\sum_{l\in I_{ijk}}B_{ijk}(\boldsymbol{\tau}_l)}\\&=\frac{\sum_{l\in I_{ijk}}B_{ijk}(\boldsymbol{\tau}_l)(\boldsymbol{\delta}_l^{(m)}-\sum_{i,j,k}\boldsymbol{\Delta}_{ijk}^{(m)}B_{ijk}(\boldsymbol{\tau}_l))}{\sum_{l\in I_{ijk}}B_{ijk}(\boldsymbol{\tau}_l)}\\&=\frac{\sum_{l\in I_{ijk}}\boldsymbol{\delta}_l^{(m)}B_{ijk}(\boldsymbol{\tau}_l)}{\sum_{l\in I_{ijk}}B_{ijk}(\boldsymbol{\tau}_l)}-\frac{\sum_{l\in I_{ijk}}\sum_{i,j,k}\boldsymbol{\Delta}_{ijk}^{(m)}B_{ijk}(\boldsymbol{\tau}_l)}{\sum_{l\in I_{ijk}}B_{ijk}(\boldsymbol{\tau}_l)}\end{aligned}$$

$$= \mathbf{\Delta}_{ijk}^{(m)} - \frac{\sum_{l\in I_{ijk}} \sum_{i,j,k} \mathbf{\Delta}_{ijk}^{(m)} B_{ijk}(\boldsymbol{\tau}_l)}{\sum_{l\in I_{ijk}} B_{ijk}(\boldsymbol{\tau}_l)},$$

we have the iterative format in matrix form,

$$\Delta^{(m+1)} = (I - \Lambda A^T A)\Delta^{(m)},$$

where, I is an identity matrix, Λ is the diagonal matrix,

$$\Lambda = diag\left(\frac{1}{\sum_{l\in I_{000}} B_{000}(\boldsymbol{\tau}_l)}, \frac{1}{\sum_{l\in I_{001}} B_{001}(\boldsymbol{\tau}_l)}, \cdots, \frac{1}{\sum_{l\in I_{n_u,n_v,n_w}} B_{n_u,n_v,n_w}(\boldsymbol{\tau}_l)}\right),$$

and A is the collocation matrix,

$$A = \begin{bmatrix} B_{000}(\boldsymbol{\tau}_0) & B_{001}(\boldsymbol{\tau}_0) & \cdots & B_{n_u,n_v,n_w}(\boldsymbol{\tau}_0) \\ B_{000}(\boldsymbol{\tau}_1) & B_{001}(\boldsymbol{\tau}_1) & \cdots & B_{n_u,n_v,n_w}(\boldsymbol{\tau}_1) \\ \cdots & \cdots & \cdots & \cdots \\ B_{000}(\boldsymbol{\tau}_n) & B_{001}(\boldsymbol{\tau}_n) & \cdots & B_{n_u,n_v,n_w}(\boldsymbol{\tau}_n)r \end{bmatrix},$$

where $\boldsymbol{\tau}_l = (u_l, v_l, w_l), l = 0, 1, \cdots, n$.

If the matrix A is nonsingular, $\Lambda A^T A$ is positive definite, so all of its eigenvalues λ are positive real numbers. Together with $\|\Lambda A^T A\|_{L^\infty} = 1$, we have $\lambda \in (0, 1]$. Therefore, the eigenvalues of $I - \Lambda A^T A$, i.e., $1 - \lambda$, are all in $[0, 1)$, and its spectral radius is $\rho(I - \Lambda A^T A) < 1$. This means that the iterative format developed in Section 3.2 is convergent.

In fact, the limit of the iterative format developed in Section 3.2 is just the least-squares fitting result to the tet mesh vertices $\boldsymbol{Q}_l$, $l = 0, 1, \cdots, n$. Suppose,

$$Q = [\boldsymbol{Q}_0, \boldsymbol{Q}_1, \cdots, \boldsymbol{Q}_n], \text{ and, } P^{(m)} = [\boldsymbol{P}_{000}^{(m)}, \boldsymbol{P}_{001}^{(m)}, \cdots, \boldsymbol{P}_{n_u,n_v,n_w}^{(m)}].$$

Due to Eq. (14), we have,

$$\Delta^{(m)} = \Lambda A^T (Q - AP^{(m)}).$$

Therefore,

$$P^{(m+1)} = P^{(m)} + \Delta^{(m)} = P^{(m)} + \Lambda A^T(Q - AP^{(m)}) = (I - \Lambda A^T A)P^{(m)} + \Lambda A^T Q$$
$$= (I - \Lambda A^T A)^{m+1} P^{(0)} + \sum_{l=0}^{m} (I - \Lambda A^T A)^l \Lambda A^T Q.$$

Because $0 \le \lambda(I - \Lambda A^T A) < 1$, we have,

$$\lim_{m\to\infty} (I - \Lambda A^T A)^m = 0, \text{ and, } \sum_{l=0}^{\infty} (I - \Lambda A^T A)^l = (\Lambda A^T A)^{-1}.$$

Consequently, when $m \to \infty$,

$$P^{(\infty)} = (\Lambda A^T A)^{-1} \Lambda A^T Q.$$

It is equivalent to,

$$A^T A P^{(\infty)} = A^T Q, \quad (18)$$

i.e., the normal equation of the least-squares fitting system to the tet mesh vertices. In conclusion, the limit of the iterative format developed in Section 3.2 is the least-squares fitting result to the tet mesh vertices $\boldsymbol{Q}_l$, $l = 0, 1, \cdots, n$.

4. Results and discussions

The discrete volume parameterization method and the iterative fitting algorithm have been implemented with Visual C++, and run on a PC with Intel Core2 Quad CPU Q9400 2.66 GHz and 4G memory. In this section, we will demonstrate some results generated by the parameterization method and iterative fitting algorithm.

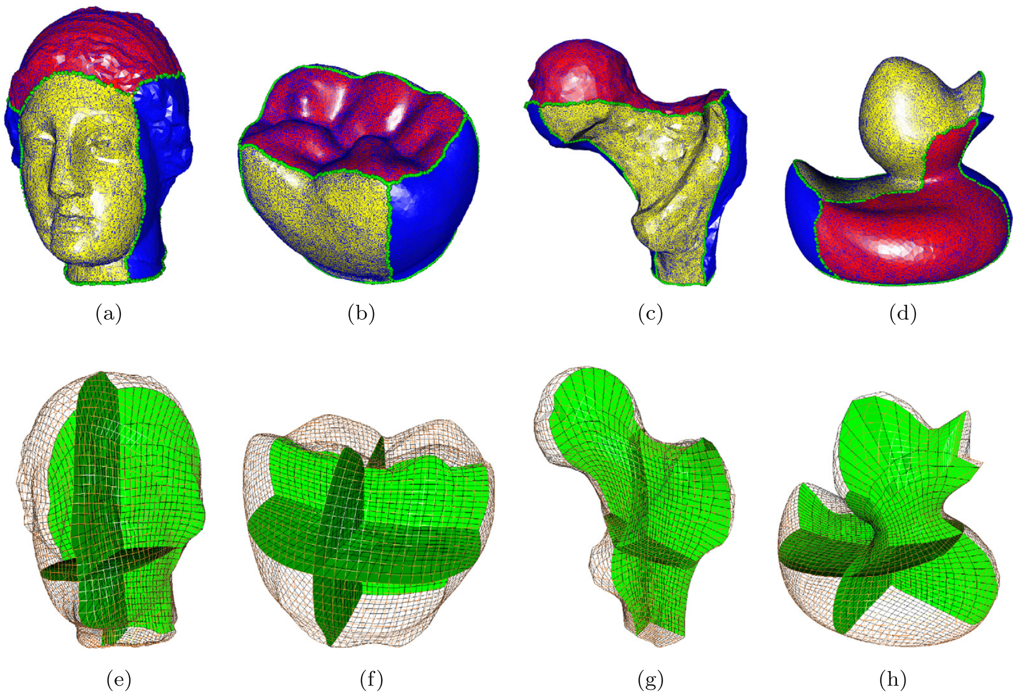

Fig. 3. The input tet mesh models, Venus (a), Tooth (b), Ball Joint (c), and Duck (d), and their parameterization results (e), (f), (g), (h) with three pieces of u, v, and w directional isoparametric patches.

Table 1
Statistics for the parameterization and iterative fitting algorithm.

Model	#vert.[1]	#tet.[2]	Para.[3]	C[4]	#control[5]	Precision	Fit.[6]	Avg. Jac.[7]	V^-/V[8]
Venus	35 858	133 256	3.71	15	$30 \times 30 \times 30$	9.13×10^{-4}	3.4	0.8821	0.14%
Tooth	61 311	227 832	7.37	10	$30 \times 26 \times 26$	2.57×10^{-4}	6.8	0.9320	0.02%
Ball Joint	43 994	163 496	4.38	8	$30 \times 22 \times 34$	7.28×10^{-4}	6.1	0.8189	0.12%
Duck	41 998	156 612	4.39	7	$30 \times 30 \times 24$	1.55×10^{-3}	3.6	0.8377	0.22%

[1] Number of the tet mesh vertices.
[2] Number of the tetrahedra.
[3] Time for parameterization is in seconds.
[4] The value of parameter C in Eq. (12).
[5] Number of control points.
[6] Time for iterative fitting is in seconds.
[7] Average scaled Jacobian value.
[8] Ratio between the volume of the regions with negative scaled Jacobian and total volume of the B-spline solid.

4.1. Parameterization results

As stated above, the input to our parameterization algorithm is a tet mesh model with six boundary triangular mesh patches segmented in advance [Fig. 1(a)]. Each of the boundary mesh patches is parameterized by the triangular mesh parameterization method developed in Pinkall and Polthier (1993). Then, the parameterization of the tet mesh is solved by the discrete volume parameterization method developed in this paper. Fig. 3 illustrates the inputting tet mesh models and the corresponding parameterization results, where the quadrilateral parametric mesh on the boundary and three isoparametric patches along u, v, and w direction, respectively, are demonstrated.

As noted in Section 2, the discrete volume parameterization method developed in this paper is equivalent to solving the harmonic equation using the finite element method. It is well known that the solution domain of the harmonic equation can be considered as a steady temperature field. Therefore, the isoparametric patches in the parameterization of the tet mesh are just the isothermal patches in the steady temperature field. Because the isothermal patches in a steady temperature field are non-self-overlapping, the isoparametric patches in the generated parameterization are also guaranteed to be non-self-overlapping.

In Table 1, we list the experimental data on the parameterization method for tet mesh models developed in this paper [refer to Fig. 3]. It can be seen that the proposed discrete volume parameterization method usually costs several seconds for parameterizing tet meshes with tens of thousands of vertices and hundreds of thousands of tetrahedra.

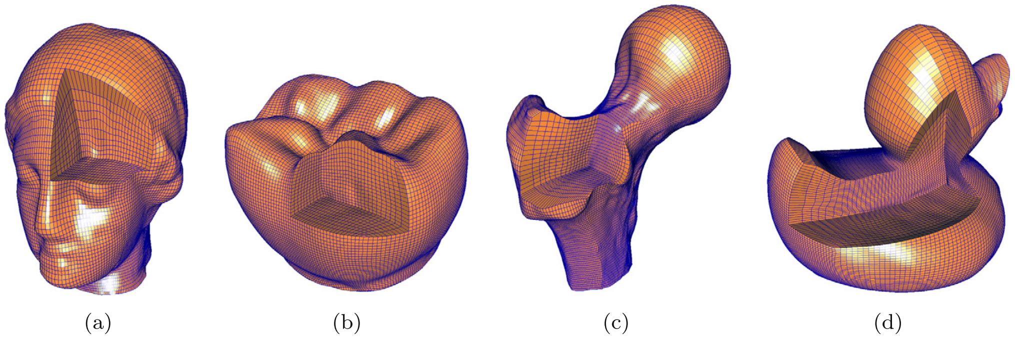

Fig. 4. The trivariate B-spline solids generated by the iterative fitting algorithm developed in this paper. (a) Venus, (b) Tooth, (c) Ball joint, (d) Duck.

4.2. Fitting results

By parameterization, each vertex v_i of a tet mesh model is assigned parameter values (u_i, v_i, w_i), and then the tet mesh can be fitted with a B-spline solid by the iterative fitting algorithm developed in Section 3.2.

In our implementation, the fitting precision is defined as the RMS error divided by the diagonal length of the bounding box of the tet mesh model, i.e.,

$$\frac{\sqrt{\frac{\sum_l \left\|\boldsymbol{\delta}_l^{(m)}\right\|^2}{n+1}}}{L}, \tag{19}$$

where, $\boldsymbol{\delta}_l^{(m)}$ (13) is the difference vector for the data point at the last iteration, $n+1$ is the number of tet mesh vertices, and L is the diagonal length of the bounding box of the tet mesh model.

The fitting is performed incrementally. Suppose the number of control points of the B-spline in the current round of iterations is $(n_u+1)\times(n_v+1)\times(n_w+1)$. If the fitting precision (19) fails to reach the prescribed precision when the current round of iterations stops, a new round of iterations will be invoked in which the number of control points of the B-spline solid is increased to $(n_u+\lceil\frac{n_u}{10}\rceil+1)\times(n_v+\lceil\frac{n_v}{10}\rceil+1)\times(n_w+\lceil\frac{n_w}{10}\rceil+1)$, where $\lceil\frac{n_u}{10}\rceil$ denotes an integer not smaller than $\frac{n_u}{10}$. This procedure stops when the fitting precision meets the prescribed value.

Fig. 4 demonstrates the fitting results, i.e., the trivariate B-spline solids, and Table 1 lists the statistics of these B-spline solids. In Table 1, the sixth column is the number of control hex mesh vertices of the B-spline solid; the seventh and eighth columns are the fitting precision and fitting time, respectively; the ninth column is the average scaled Jacobian value (Knupp, 2003) of the B-spline solid, defined as

$$ave\_Jac = \frac{\iiint_\Omega J(x,y,z)dxdydz}{\iiint_\Omega dxdydz},$$

where, $J(x,y,z)$ is the scaled Jacobian value at (x,y,z) (Knupp, 2003). Moreover, the last column lists the ratio between the volume of the regions with negative scaled Jacobian and the total volume of the B-spline solid.

From Table 1, we can see that the iterative fitting algorithm usually costs several seconds in reaching the fitting precision 10^{-4} (10^{-3} for the model *Duck*) for fitting tet mesh model with tens of thousands of vertices and hundreds of thousands of tetrahedra. In addition, although there are some regions with negative Jacobian in the generated B-spline solids, these regions are very small. The ratio between the volume of the regions with negative Jacobian and the total volume of the B-spline solid is usually below 0.25%. By further checking, it was found that the regions with negative Jacobian are strip-shaped and around the boundary curves. Therefore, the emergence of the regions with negative Jacobian is *closely related to* the segmentation of the boundary triangular mesh of the tet mesh model. As a future work, we will study the method which can generate the trivariate B-spline solid with strictly positive Jacobian.

It should be pointed out that, though the two methods developed in Martin et al. (2009) and this paper both construct the volume parameterization with the harmonic function, and fit the tet mesh by a trivariate B-spline solid, the differences between the two methods are evident. First, the structures of the volume parameterizations generated by the two methods are different. While the parameterization method in Martin et al. (2009) maps the tet mesh into a region in the cylindrical coordinate system, the method developed in this paper maps the tet mesh into a region in the Cartesian coordinate system. Second, the parameterization methods presented in Martin et al. (2009) and this paper are different. In Martin et al. (2009), the volume parameterization is generated by solving the Laplace equation using the finite element method. In this paper, a parameterization formula with intuitive geometric significance for each inner mesh vertex is explicitly constructed, so users who are unfamiliar with the finite element method can implement the volume parameterization easily. Third, the limits of the iterative algorithms in Martin et al. (2009) and this paper are different. The limit of the iterative algorithm in Martin et al. (2009) is a trivariate B-spline solid which interpolates the tet mesh vertices. So the number of control points of the

Table 2
Statistics for the singular coefficient matrices $A^T A$ (18).

Model	Rank[1]	#nonzero elements[2]	Condition number[3]
Venus	27 000 × 27 000	4 230 202	∞
Tooth	21 870 × 21 870	3 808 126	∞
Ball Joint	20 790 × 20 790	3 849 609	∞
Duck	21 600 × 21 600	3 853 514	∞

[1] Rank of the matrix; $N \times N$ means the number of the control points is N.
[2] Number of nonzero elements of the matrix.
[3] Condition number of the matrix.

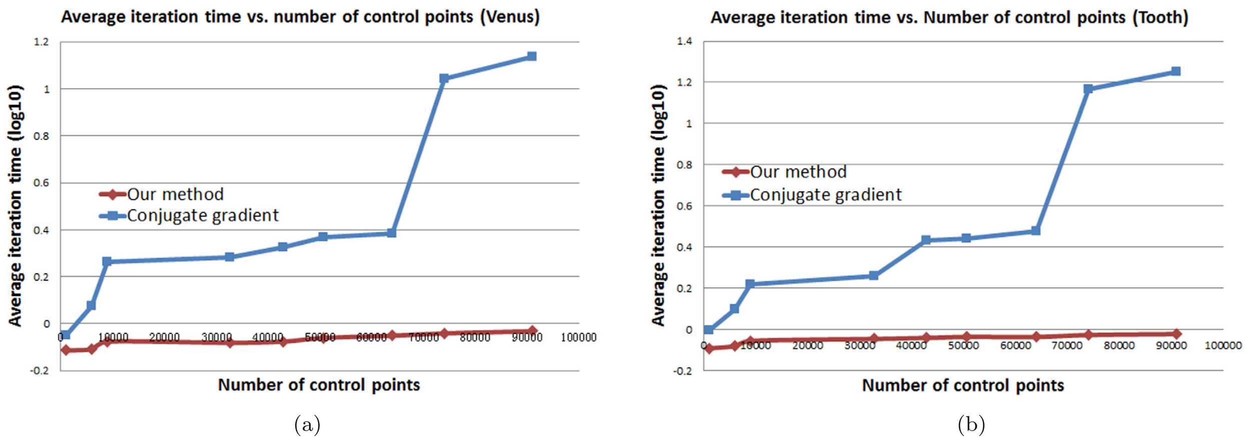

Fig. 5. Plots of the average iteration time in each round iterations vs. the number of control points of a B-spline solid. (a) Venus. (b) Tooth.

B-spline solid is equal to the number of the tet mesh vertices. However, the limit of the iterative algorithm in this paper is a trivariate B-spline solid which approximates the tet mesh vertices. Then, the number of control points of the generated B-spline solid is less than the number of the tet mesh vertices.

Capability of solving singular linear system: We compared our iterative fitting algorithm with some conventional methods. Just as pointed out in Pereyra and Scherer (2003), when the number of control points rises somewhat large, the coefficient matrices $A^T A$ (18) for fitting the four models all become singular. Table 2 lists the statistics of such coefficient matrices, including rank, number of nonzero elements, and condition number. The infinite condition numbers show that the four matrices are all singular. While conventional methods, such as the Cholesky method and the conjugate gradient method, fail to solve these singular linear system of Eqs. (18), our iterative fitting algorithm is robust enough to successfully generate the correct fitting results [refer to Fig. 4].

Nearly constant averaging iteration time: In Fig. 5, we illustrate the plots of the averaging iteration time (taking logarithm with base 10) vs. the number of control points in fitting the two models *Venus* [Fig. 4(a)] and *Tooth* [Fig. 4(b)], by the conjugate gradient method and our method, respectively. In order for fair comparison, when the coefficient matrix $A^T A$ (18) becomes singular and the conjugate gradient method fails, we run each of the two methods six times with fixed number of control points, and take the average time in each iteration.

As stated above, when the number of control points of the B-spline solid becomes very large, direct methods, such as Cholesky method, LU decomposition, etc., are invalid, and only iterative methods are feasible to solve such linear system with a great many unknowns. However, with conventional iteration methods, the time cost in each iteration will rapidly rise with the increase of the number of unknowns, just as illustrated in Fig. 5. Therefore, in the incremental fitting scheme employed in our method, the iteration will become slower and slower, when more and more control points are inserted.

On the contrary, as demonstrated in Fig. 5, with the iterative fitting method developed in this paper, the averaging iteration time keeps nearly constant with the increase of the number of control points. The reason for this phenomenon is as follows. As delineated in Section 3.2, when the number of control points of a B-spline solid increases, the non-zero area of a B-spline basis function decreases, and so does the number of vectors distributed to a control point. Therefore, in theory, the total amount of computation in an iteration is unchanged when the number of control points increases. Then, our iterative fitting method is more suitable in incremental fitting tet mesh models with a large number of vertices and tetrahedra.

Limitations: The developed method has two main limitations. First, in the generated B-spline solid, there is a narrow strip-shaped region around the boundaries with negative Jacobian values. Second, due to the heavily non-uniform distribution of the tet mesh vertices, the fitting error can at best reduce to the order of magnitude of 10^{-4}. As the future work, the iterative method should be improved to increase the fitting precision. Moreover, a desirable triangular mesh partition method and suitable optimization techniques are required to ensure the positivity of the Jacobian value of the generated B-spline solid.

5. Conclusion

In this paper, we developed a discrete volume parameterization method and an iterative algorithm for fitting a tet mesh model with a B-spline solid. The linear system of equations for the volume parameterization is constructed geometrically, and the method is equivalent to solving a harmonic equation by the finite element method. Therefore, the generated parameterization is guaranteed to have no self-overlapping. Moreover, we presented an iterative algorithm for fitting a tet mesh model. The iteration speed of the iterative algorithm is determined by the number of tet mesh vertices rather than the number of control hex mesh vertices of the B-spline solid. Thus, the proposed iterative algorithm is desirable in fitting a tet mesh with a large number of vertices incrementally.

Acknowledgements

This paper is supported by Natural Science Foundation of China (Nos. 61379072, 61202201), and the Open Project Program of the State Key Lab of CAD&CG (Grant No. A1305), Zhejiang University.

References

Aigner, M., Heinrich, C., Jüttler, B., Pilgerstorfer, E., Simeon, B., Vuong, A.-V., 2009. Swept volume parameterization for isogeometric analysis. In: Hancock, E., Martin, R., Sabin, M. (Eds.), Mathematics of Surfaces XIII. In: Lecture Notes in Computer Science, vol. 5654. Springer, Berlin, Heidelberg, pp. 19–44.
Burkhart, D., Hamann, B., Umlauf, G., 2010. Iso-geometric finite element analysis based on Catmull–Clark subdivision solids. Comput. Graph. Forum 29, 1575–1584.
Chen, Z., Luo, X., Tan, L., Ye, B., Chen, J., 2008. Progressive interpolation based on Catmull–Clark subdivision surfaces. In: Pacific Graphic 2008. Comput. Graph. Forum 27 (7), 1823–1827.
Cheng, F., Fan, F., Lai, S., Huang, C., Wang, J., Yong, J., 2009. Loop subdivision surface based progressive interpolation. J. Comput. Sci. Technol. 24 (1), 39–46.
Deng, C., Lin, H., 2014. Progressive and iterative approximation for least squares B-spline curve and surface fitting. Comput. Aided Des. 47, 32–44.
Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M., Stuetzle, W., 1995. Multiresolution analysis of arbitrary meshes. In: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques. ACM, pp. 173–182.
Escobar, J., Cascón, J., Rodríguez, E., Montenegro, R., 2011. A new approach to solid modeling with trivariate T-splines based on mesh optimization. Comput. Methods Appl. Mech. Eng. 200 (45), 3210–3222.
Fan, F., Cheng, F., Lai, S., 2008. Subdivision based interpolation with shape control. Comput-Aided Des. Appl. 5 (1–4), 539–547.
Floater, M.S., 2003. Mean value coordinates. Comput. Aided Geom. Des. 20 (1), 19–27.
Floater, M.S., Hormann, K., 2005. Surface parameterization: a tutorial and survey. In: Advances in Multiresolution for Geometric Modelling. Springer, pp. 157–186.
Gu, X., Wang, Y., Chan, T.F., Thompson, P.M., Yau, S.-T., 2004. Genus zero surface conformal mapping and its application to brain surface mapping. IEEE Trans. Med. Imag. 23 (8), 949–958.
Gu, X., Yau, S.-T., 2003. Global conformal surface parameterization. In: Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing. Eurographics Association, pp. 127–137.
Hormann, K., Greiner, G., 2000. MIPS: an efficient global parametrization method. In: Laurent, P.-J., Sablonnière, P., Schumaker, L.L. (Eds.), Curve and Surface Design: Saint-Malo 1999. In: Innovations in Applied Mathematics. Vanderbilt University Press, Nashville, pp. 153–162.
Hua, J., He, Y., Qin, H., 2004. Multiresolution heterogeneous solid modeling and visualization using trivariate simplex splines. In: Proceedings of the Ninth ACM Symposium on Solid Modeling and Applications. Eurographics Association, pp. 47–58.
Hughes, T., Cottrell, J., Bazilevs, Y., 2005. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput. Methods Appl. Mech. Eng. 194 (39), 4135–4195.
Knupp, P., 2003. A method for hexahedral mesh shape optimization. Int. J. Numer. Methods Eng. 58 (2), 319–332.
Li, B., Li, X., Wang, K., Qin, H., 2010. Generalized polycube trivariate splines. In: Shape Modeling International Conference. SMI, 2010. IEEE, pp. 261–265.
Lin, H., 2010. Local progressive-iterative approximation format for blending curves. Comput. Aided Geom. Des. 27 (4), 322–339.
Lin, H., Bao, H., Wang, G., 2005. Totally positive bases and progressive iteration approximation. Comput. Math. Appl. 50 (3–4), 575–586.
Lin, H., Wang, G., Dong, C., 2004. Constructing iterative non-uniform B-spline curve and surface to fit data points. Sci. China, Ser. F 47 (3), 315–331.
Lin, H., Zhang, Z., 2011. An extended iterative format for the progressive-iteration approximation. Comput. Graph. 35 (5), 967–975.
Lin, H., Zhang, Z., 2013. An efficient method for fitting large data sets using T-splines. SIAM J. Sci. Comput. 35 (6), A3052–A3068.
Lu, L., 2010. Weighted progressive iteration approximation and convergence analysis. Comput. Aided Geom. Des. 27 (2), 129–137.
Martin, T., Cohen, E., 2010. Volumetric parameterization of complex objects by respecting multiple materials. Comput. Graph. 34 (3), 187–197.
Martin, T., Cohen, E., Kirby, R.M., 2009. Volumetric parameterization and trivariate B-spline fitting using harmonic functions. Comput. Aided Geom. Des. 26 (6), 648–664.
Pereyra, V., Scherer, G., 2003. Large scale least squares scattered data fitting. Appl. Numer. Math. 44 (1), 225–239.
Pinkall, U., Polthier, K., 1993. Computing discrete minimal surfaces and their conjugates. Exp. Math. 2 (1), 15–36.
Shi, L., Wang, R., 2006. An iterative algorithm of NURBS interpolation and approximation. J. Math. Res. Expo. 26 (4), 735–743.
Wang, K., Li, X., Li, B., Xu, H., Qin, H., 2012. Restricted trivariate polycube splines for volumetric data modeling. IEEE Trans. Vis. Comput. Graph. 18 (5), 703–716.
Wang, W., Zhang, Y., Liu, L., Hughes, T.J., 2013. Trivariate solid T-spline construction from boundary triangulations with arbitrary genus topology. Comput. Aided Des. 45 (2), 351–360.
Wang, X., Qian, X., 2014. An optimization approach for constructing trivariate B-spline solids. Comput. Aided Des. 46, 179–191.
Xu, G., Mourrain, B., Duvigneau, R., Galligo, A., 2013. Analysis-suitable volume parameterization of multi-block computational domain in isogeometric applications. Comput. Aided Des. 45 (2), 395–404.
Zhang, Y., Bazilevs, Y., Goswami, S., Bajaj, C.L., Hughes, T.J., 2007. Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. Comput. Methods Appl. Mech. Eng. 196 (29), 2943–2959.
Zhang, Y., Wang, W., Hughes, T.J., 2012. Solid T-spline construction from boundary representations for genus-zero geometry. Comput. Methods Appl. Mech. Eng. 249, 185–197.
Zienkiewicz, O., Taylor, R., Zhu, J., 2008. The Finite Element Method: Its Basis and Fundamentals, sixth edition. Elsevier, Singapore.