Domain Construction for Volumetric Cross-Parameterization

Tsz-Ho Kwok and Charlie C. L. Wang[†]

Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong
[†]Corresponding Author; E-mail:cwang@mae.cuhk.edu.hk; Fax: (852)26036002.

Abstract

We present an algorithm in this paper for constructing volumetric domains with consistent topology to parameterize three-manifold solid models having homeomorphic topology. The volumetric parameterizations generated by our approach share the same set of base domains and are constrained by the corresponding anchor points. Our approach allows users to control interior mappings by specifying interior anchor points, and the anchor points are interpolated exactly. With the help of a novel construction algorithm developed in this work, the volumetric cross-parameterization computed by our method demonstrates its functionality in several examples.

Keywords:
domain construction, volumetric parameterization, consistent connectivity, cross-parameterization

1. Introduction

Many geometric processing applications require a bijective mapping between models (for example, texture mapping, detail transfer, morphing, and shape analysis). Computing a bijective mapping between two-manifold surfaces has been widely studied in computer graphics. A general solution for constructing such mappings can be computed through global parameterization approaches (such as [1, 2, 3]). However, for the applications such as morphing and detail transfer, parameterization must be constrained by semantic features, which are usually specified as anchor points on the surfaces of input models. In surface parameterization approaches [4, 5, 6, 7], common base-domains are constructed for the surfaces of input models to establish mappings that satisfy the constraints prescribed by anchor points.

The information provided by the boundary surfaces of a model may be insufficient for describing interior information like material, intensity, and micro-structure, which should be defined in the entire solid model. Therefore, researchers have paid more attention to volumetric parameterization recently. Similar to the surface cases, volumetric cross-parameterization is also constrained by semantic features. Computing a constrained mapping between three-manifold models is more challenging than between two-manifold models. Some existing approaches [8, 9, 10, 11, 12] formulate the computation of volumetric mapping globally based on surface correspondences. Bijective mappings can be obtained in some specific types of domain shapes (such as star shape by the method of Xia et al. [12]). In real applications, models can have complex geometry, nontrivial topology, and even interior structures. The global domain of such models must be decomposed into sub-domains with simpler shapes to ensure bijection in mappings. This motivates our work to find a solution for constructing a complex of base-domains with consistent topology. Directly computing a volumetric mapping by using the result of surface cross-parameterization as constraints cannot guarantee the result of bijective mapping. For instance, the radial basis functions (RBFs) based mapping presented in [13] can have self-intersection. Similar problem occurs when tetrahedral mesh based deformation [14] is applied to generate the volumetric mapping. Warping a volumetric mesh (with 171.7k tetrahedra) for the cylinder in Figure 2 to the rabbit leads to 15k degenerated tetrahedra.

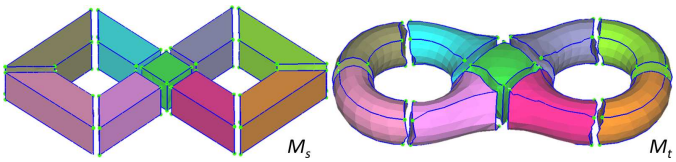

Figure 1: The polyhedral cells in the template set of base-domains (left) and their corresponding C-polys (right), where the C-polys corresponding to different base-domains are displayed in different colors.

In this paper, we propose a method to compute domains of volumetric cross-parameterization on three-manifold models having homeomorphic topology, where the parameterization is constrained by anchor points. Here, a few heuristics are applied to specify anchor points for generating successful volumetric mapping:

- First, the anchor points should be defined according to the semantic correspondences – e.g., mapping two toes of a human model to the ears and shoulders of another human body will generate highly distorted mapping.
- Second, relatively uniform distribution of anchor points is expected to result in base-domains with good shape, which is helpful to reduce the stretch in mapping.

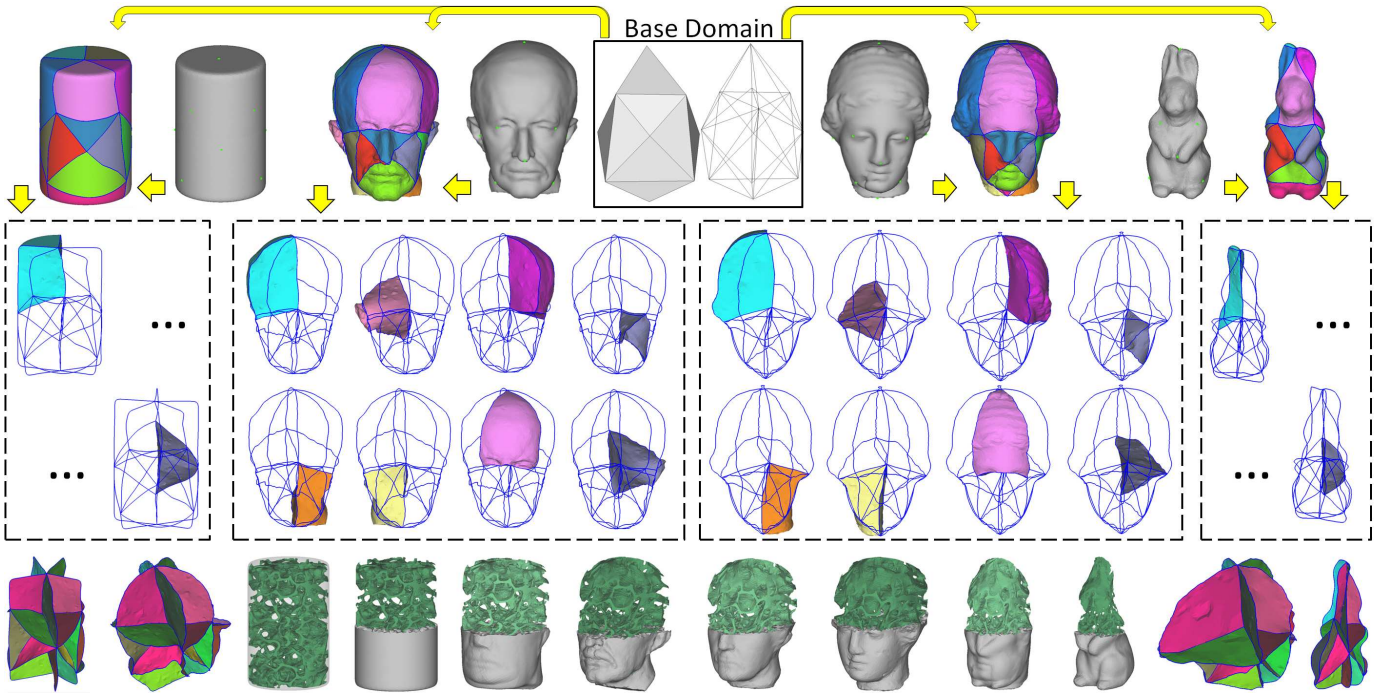

Figure 2: Volumetric cross-parameterization can be established among models with different shapes. Given the prescribed anchor points, the entire solid model can be decomposed into curved polyhedral cells having the same connectivity as that of the common base-domains. By the volumetric cross-parameterization, interior structures of the cylinder model can be transferred across all the models (bottom row).

- Third, anchors are specified to help decompose the domains into nearly convex regions and help add constraints inside solids.

The computation is conducted on solid models represented by tetrahedral meshes, which can be generated from an intersection-free triangular mesh model (e.g., by the publicly available tool in [15]). A tetrahedral mesh M can be represented by a tuple $M = (V, E, F, T)$ which is a collection of vertices (V), edges (E), faces (F), and tetrahedra (T). In addition to the tuple, a solid model M to be parameterized also has a set of pre-defined anchor points G, which could be specified on the boundary surface of M or inside M. The models are then to be parameterized onto a set of *common base-domains* (CBD). The CBD can be considered as a volumetric mesh model formed by polyhedral cells, where every vertex of the cells is corresponding to an anchor point. By this setup, the volumetric cross-parameterization on a set of models M_i ($i = 1, 2, \cdots$) sharing the same set of anchor points G is converted into a problem of how to consistently partition a model into a set of *curved polyhedral* regions (called *C-poly*s in this paper) according to the CBD. The template of CBD is also represented by a tuple $\Psi = (G_\Psi, E_\Psi, F_\Psi, D_\Psi)$ – G_Ψ is the set of vertices, E_Ψ and F_Ψ are the collections of edges and faces, and D_Ψ is the set of polyhedral cells (see Figure 1 for the base-domains of a genus-two model).

The basic idea of domain construction on a given model M is to compute curves, patches, and C-polys in M between anchor points according to edges E_Ψ, faces F_Ψ and polyhedral T_Ψ in Ψ respectively. The connectivity of constructed C-polys needs to be consistent with D_Ψ. When all models are parameterized to the same set of CBD, the cross-parameterization between them is established (e.g., our examples use the mean value coordinates [16]). Figure 2 shows an example of how to establish cross-parameterization in the volumes of models by our domain construction algorithm.

1.1. Related work

Surface parameterization has been studied for many years and comprehensive surveys can be found in [17, 18]. Here, our review only focuses on the approaches that find constrained bijective mappings between a pair (or a set) of models. Praun et al. [4] used the connectivity of a predefined template as base-domains and traced the boundary of patch layouts on each of the input meshes by linking the given anchor points in a consistent way as that in the template. Kraevoy et al. [5] and Schreiner et al. [6] further extended the idea of Praun et al. [4] by automating the generation of common base-domains. Our work presented in this paper generalizes the cross-parameterization to volumetric domains.

Volumetric parameterization plays an important role in many solid modeling applications and has attracted more and more attention recently. Ju et al. [19] and Floater et al. [16] extended the mean-value coordinates [20] from surfaces to volumes to compute the interpolation of volumetric data. Mean-value coordinate is a powerful and flexible tool to compute the mapping between two volumes. However, to use it on general solid models with complex shape, a domain construction method as what we propose in this paper is needed.

Wang et al. [10] generalized the harmonic mapping to tetrahedral meshes and reduced the discrete harmonic energy by a variational procedure. They proposed an algorithm to map a genus-zero volume to a solid sphere. Li et al. [8, 9, 21] computed the harmonic volumetric mapping between three-manifold models having the same topology. The basic idea of their approach is to simulate an electrical charging system over sample points by using harmonic functions. The *Method of Fundamental Solutions* (MFS) is used in order to manipulate the boundary map. Li et al. [21] also pointed out the importance of considering heterogeneous structures in volumetric data. However, the volumetric parameterization based on harmonic function is not always bijective and the constraints are only approximated but not enforced. The approach introduced by Martin et al. [22] can compute harmonic volumetric parameterization for cylindrical volumes, which is used for trivariate spline construction. Xia et al. [12] parameterized star-shaped volumes by using Green functions, and showed that the constructed map is bijective and smooth except at only one unique critical point. Xia et al. also proposed an algorithm [11] to decompose a volume into the direct product of a two-dimensional surface and a one-dimensional curve. By tracing the integral curve along the harmonic function, a bijective mapping is constructed between the volume and the domain. However, their approach is not as flexible as ours that allows to add anchor points inside solid models.

1.2. Main results

The technical contributions of this paper are summarized as follows.

- We develop a novel *patch construction* algorithm to find a two-manifold surface patch composed of the mesh faces to approximate an intersection-free surface interpolating a boundary loop embedded in three-manifold models. The area of patch is minimized to make the boundary of domain compact.

- An automatic construction algorithm is investigated in this paper to construct *curved polyhedra* (named as C-polys) which serve as *common base-domains* (CBD) for volumetric parameterization.

Based on the CBD constructed by our approach, constrained cross-parameterization between three-manifold models can be established.

The rest of our paper is organized as follows. Section 2 presents the algorithms for CBD construction and volumetric cross-parameterization. The operators for constructing the boundary curves and patches of C-polys are presented in section 3. After that, the operators for reducing distortion in volumetric parameterization are introduced in section 4. Examples and applications to demonstrate the functionality of our approach are given in section 5.

2. Algorithms

In this section, we present the algorithms for CBD construction and the volumetric cross-parameterization on CBD. Assumptions for the input of these algorithms are as follows.

- Input models are required to be homeomorphic to each other, i.e., having the same topology.

- Anchor points (on/in the models) should have one-to-one correspondences and are specified in a consistent manner.

2.1. Consistent boundary surface decomposition

Given the template of CBD which could be treated as a special set of connected polyhedral cells, its boundary surface $\Psi^B = (G^B_\Psi, E^B_\Psi, F^B_\Psi)$ is actually a polygonal mesh. All the vertices, G^B_Ψ ($G^B_\Psi \subset G_\Psi$), of Ψ^B should already have their corresponding anchor points defined on the boundary surface of the input model M. The boundary surface can be decomposed into patches P^B having the same connectivity as Ψ^B by a variant [7] of the consistent surface decomposition method in [4]. As a result, the boundary surfaces of input models are decomposed into a set of polygonal patches having the same topology as that of Ψ^B (see the top row of Figure 2 for an example). These patches serve as an initial front to construct C-polys in the following sub-section.

2.2. Construction of curved polyhedral domains

After constructing the boundary surface patches, P^B, of a solid model M according to Ψ^B, P^B is used to construct *curved polyhedral domains* (C-poly) inside M. In order to generate C-polys in a consistent connectivity as the polyhedral cells D_Ψ, an advancing-front strategy is adopted here. Starting from a polygonal surface patch in the front, its adjacent C-poly (corresponding to a polyhedron in Ψ) can be constructed and the front is updated accordingly. Therefore, by using the polygonal patches in P^B as the initial front, we can progressively construct C-polys adjacent to the front one by one. To ensure that the constructed C-polys have the same connectivity as D_Ψ in Ψ, the following requirements must be satisfied during the construction.

1. **Free of intersections:** The newly constructed curves and surface patches must not intersect with any other anchor points, curves or patches.
2. **Consistent cyclic order:** The cyclic order of patches around a curve must be consistent to that of the corresponding path in Ψ (see Figure 3 for an illustration).
3. **Non-blocking:** A C-poly should not enclose any anchor points that do not belong to it (see Figure 4 for an illustration).

It is known that restricted brushfire algorithm [4] could be used to trace a path between anchor points in a topologically equivalent manner on surface. To prevent intersections and wrong cyclic order when generating patches and paths, we develop a novel method in section 3 to construct patches in an intersection-free manner and in a correct cyclic order.

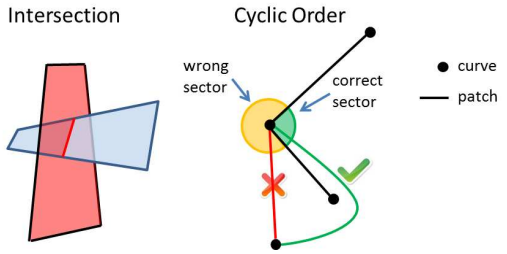

Figure 3: Intersections between patches/curves should be prevented, and cyclic order around a curve must be enforced.

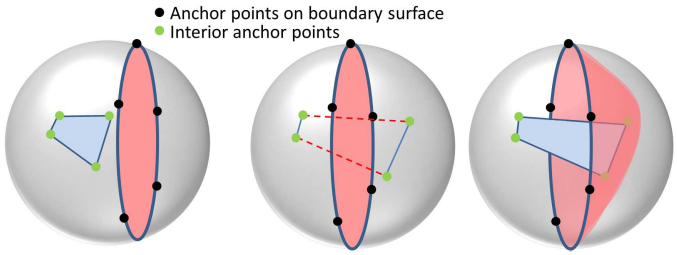

Figure 4: An illustration of blocking: (left) a blue patch and a red patch defined in template, (middle) the blue patch fails to be constructed due to the blocking generated by the red patch, and (right) blocking can be prevented by constructing the red patch in a restricted way.

Furthermore, the construction algorithm will fail to terminate if an anchor point is enclosed by other C-polys and becomes unreachable. For example as shown in Figure 4, a blue patch should be constructed as shown in the left, but it is possible that the prior constructed patch (in red) blocks the anchor points of the blue patch to be connected – in other words, the red patch divids the anchor points of the blue patch into two disconnected regions. This is also known as the *blocking* problem in the consistent surface parameterization setup (ref. [4, 5]). Non-blocking can be guaranteed by regional restriction in our algorithm (details can be found in section 3.3).

Remark: Throughout the advancing-front process, we keep the front being two-manifold until no more C-ploys can be constructed.

2.3. Volumetric mappings

Each C-poly constructed by the above algorithm is composed by a set of tetrahedra. The volumetric mapping from a C-poly to its corresponding polyhedron in the template Ψ can be established then (e.g., by Green's functions [12], MFS [9], harmonic field [11], or the mean-value coordinates [16, 19]).

For two solid models, M_s and M_t, having the consistent sets of anchor points, the cross-parameterization can be established between their volumes if both of them are parameterized to the same template of base-domains, Ψ. By the model-to-domain mappings $\Gamma_s : M_s \Rightarrow \Psi$ and $\Gamma_t : M_t \Rightarrow \Psi$, the model-to-model mapping is $\Gamma_{st} = \Gamma_t^{-1} \cdot \Gamma_s$. The volumetric cross-parameterization established in this way is continuous across the boundaries of C-polys. To have a compact representation for the result of cross-parameterization, we warp the tetrahedral mesh for M_s into a mesh for M_t that has new distribution of vertices but the same connectivity. The warping can be generated by using the mean-value coordinates (e.g., [16, 19]). The tetrahedral meshes with consistent connectivity are the final results stored for the cross-parameterization.

2.4. Generation of template complex

The template complex of CBD for a set of solid models M_i ($i = 1, 2, \cdots$) sharing the consistent sets of anchor points can be constructed in an automatic way. First, we pick one solid model M_b and apply the surface domain decomposition method of Kwok et al. [7] to construct a layout of triangular surface patches linking the anchor points. Second, by converting patch boundaries into straight edges, a triangular surface mesh is obtained, which has anchor points on the surface of M_b as its vertices. Last, using the interior anchor points of M_b and the triangular surface mesh as constraints, the algorithm of *Constrained Delaunay Tetrahedralization* (CDT) such as [15, 23] can be used to build the connectivity of a tetrahedral mesh that use only the anchor points of M_b as vertices. The complex of this tetrahedral mesh is good enough to be employed as the complex of CBD, Ψ. To get a better performance for the parameterization, we can further merge some of the tetrahedral domains to be polyhedral domains if the merging improves the shape of domains, e.g., a regular cube is better than the same region composed by a few tetrahedra.

3. Operators for Domain Construction

3.1. Tracing boundary curves

A boundary curve of C-polys is a path linking two anchor points. Basically, any part of the path should *NOT* run beyond the space enclosed by the input solid model M. To satisfy the requirement of being intersection-free, the path also should *NOT* pass through any boundary surface patch that has been constructed. Here, we trace the shortest path in M for each edge in E_Ψ by employing a standard brushfire algorithm [24]. To prevent self-intersections in domain construction, the edges should not be considered as a possible path if they are adjacent to any mesh edges or faces that have been classified as parts of the existing C-polys. In addition, to overcome topological obstacles, virtual edges are added between the center of a face and the vertex opposite to the face when tracing paths. If the resultant shortest path passes through a virtual edge, the tetrahedron holding this virtual edge will be subdivided.

3.2. Patch construction

A more challenging work is to construct a surface patch defined according to anchor points $\mathbf{a}_i$ ($\forall i \in 1, 2, \ldots, n$). The boundary curves linking these anchor points have been generated by the above method, and these boundary curves form a loop C to be filled by a surface patch. Similar to curves composed of mesh edges, we construct patches composed of mesh faces here. Again, the constructed patches bounding C-polys should not intersect any other existing curves or patches and

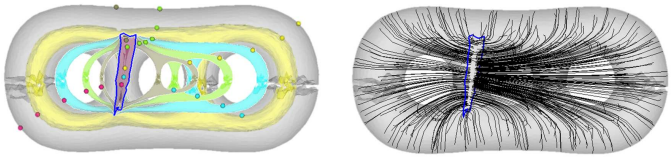

Figure 5: Applying the Laplacian operator to a solid model can shrink it into a membrane interpolating the constrained loop (in blue): (left) the models in different colors illustrate the shrinkage of the input volume in different stages, and (right) the black curves give the moving trajectories of vertices.

must be two-manifold. Moreover, the constructed patch is expected to be a discrete surface with a minimal area to reduce the distortion in volumetric mapping. Our basic idea of computing a surface patch that interpolates the given loop is to mimic a physical phenomenon of shrinking a volume of viscous fluid around a ring. Keeping on the shrinkage, the volume of fluid will become a membrane that is stretched and fills the ring.

3.2.1. Laplacian Movement Field (LMF)

The volumetric shrinkage of a solid mesh can be achieved by minimizing an energy as $E(S) = \frac{1}{2}\int_\Omega S_u^2 + S_v^2 + S_w^2 dudvdw$ (see [25] for reference). The variational derivative of $E(S)$ is a Laplacian operator. In other words, the volumetric shrinkage can be realized by applying Laplacian operators. When the shrinkage is constrained to a given loop, the result is a minimal surface interpolating the loop. The discrete Laplacian operator [26] at vertex $\mathbf{v}_i$ is defined as $L(\mathbf{v}_i) = \sum_{j\in N(\mathbf{v}_i)} w_j(\mathbf{v}_j - \mathbf{v}_i)$ with $N(\mathbf{v}_i)$ being the one-ring neighbors of $\mathbf{v}_i$. $w_j > 0$ are the weighting coefficients satisfying $\sum_j w_j = 1$. There are different choices of w_j (ref. [27]), and here we choose edge-length-based coefficients as

$$w_j = \frac{\|\mathbf{v}_j - \mathbf{v}_i\|}{\sum_{j\in N(\mathbf{v}_i)} \|\mathbf{v}_j - \mathbf{v}_i\|}. \quad (1)$$

Notice that the vertices located on the curves to be interpolated are set as hard constraints not to move during the Laplacian evolution.

After applying Laplacian operators, all vertices are moved towards the minimal surface (as illustrated in Figure 5). In practice, the position of a vertex on the resultant membrane can be directly computed by solving a linear system: $L(\mathbf{v}_i) = 0$ subject to the position constraints. For a point in the given model, the distance between its positions before and after applying the Laplacian operator indicates its distance to the minimal surface. These values in the entire solid model M form a scalar field named as *Laplacian Movement Field* (LMF). The peeling process introduced below can form a membrane enclosing C and the order of peeling is governed by LMF.

3.2.2. Construction by topology guaranteed peeling

Suppose the loop, C, to be interpolated is enclosed by a genus-zero solid S that is represented as a tetrahedral mesh, we first compute the values of LMF at every vertex of S. The LMF value of a tetrahedron is then defined as the average of the

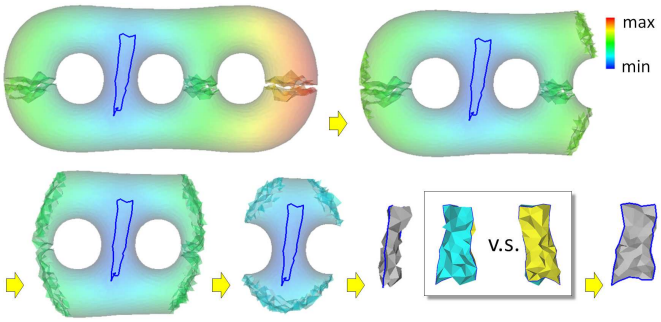

Figure 6: Progressive results of topology-preserved peeling that form a genus-zero solid $Th(S)$ with its boundary surface, $\partial Th(S)$, interpolating the loop C of boundary curves. $\partial Th(S)$ is subdivided by C into two surface regions with disk-like topology, where the one with a smaller area is selected as the surface patch of C-poly.

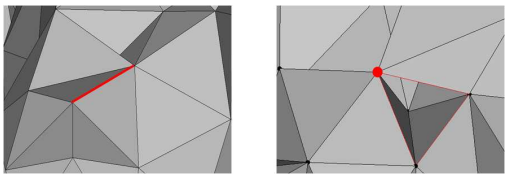

Figure 7: Peeling step that will generate non-manifold entities should be prevented: (left) non-manifold edges and (right) non-manifold vertices.

LMF values at its four vertices. We progressively change the solid S into a thin solid $Th(S)$ with its boundary interpolating C by removing tetrahedra one by one, where the tetrahedron with a larger LMF value has a higher priority to be removed (which can be implemented with the help of a maximum heap). The solid S is therefore changed into $S_0, S_1, \cdots$ until no more tetrahedron can be removed – i.e., a membrane containing only a layer of tetrahedra has been obtained. The solid membrane is denoted by $Th(S)$. See an illustration shown in Figure 6. As the boundary surface, $\partial Th(S)$, of $Th(S)$ is two-manifold and has genus-zero topology, the interpolated loop C on $\partial Th(S)$ separates the surface into two two-manifold regions. We compare their areas, and the one with a smaller area is selected as the surface patch P interpolating C. P has disk-like topology and is a collection of the mesh faces in M.

Interpolation: To ensure the boundary surface of $Th(S)$ interpolate C, when removing a tetrahedron adjacent to an edge, e, in C, we check whether it is the last tetrahedron adjacent to e. If so, the removal is prevented.

Topology Faithfulness: To ensure the boundary surface of $Th(S)$ is two-manifold and genus-zero, we need to check the topology of the resultant solid before removing a tetrahedron. Two cases are prevented.

1. The removal will lead to a non-manifold edge or a non-manifold vertex (see Figure 7).
2. The removal will generate an empty void, which changes the genus-number.

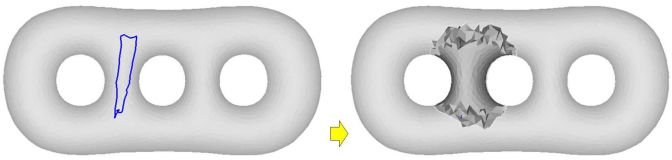

Figure 8: Topology-preserved volume growing to form the initial genus-zero solid for peeling.

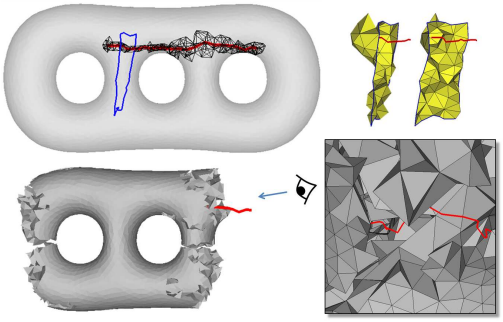

Figure 9: Intersection prevention in the surface patch construction – the tetrahedra adjacent to any other curves or patches must not be added to the grown region which will be peeled later for surface construction.

Initial genus-zero solid: The initial genus-zero solid, S, of peeling can be obtained by a growing process as described below. For a loop C that passes through a set of vertices $\mathbf{v}_i$ ($\forall i \in 1, 2, \ldots, m$), we search for a tetrahedron that contains the point $\mathbf{v}_c = \frac{1}{m}\sum \mathbf{v}_i$ (or the closest tetrahedron to $\mathbf{v}_c$ if $\mathbf{v}_c$ is not inside the input model M). This tetrahedron is named as the *centric tetrahedron*. Starting from the centric tetrahedron, we can grow a region by the advancing-front method to add tetrahedra adjacent to the front one by one. The growing procedure stops when all edges of C have been enclosed by the grown region, which is then used as the initial solid for the peeling process. Again, when growing the region, the cases that will generate non-manifold entities should be prevented to guarantee the grown volume has genus-zero topology. Figure 8 shows a genus-zero solid obtained by growing from a loop. As the peeling will be conducted in this solid S, the values of LMF are computed only on the vertices of S instead of the entire input model M. Although rarely, the centric tetrahedron could be in a region separated from the region containing the loop C; the growing algorithm will report fail to enclose C in this case. We then restart the growing procedure from a tetrahedron containing an edge of C.

As mentioned in section 2.2, topologically equivalence must be ensured by preventing intersections and wrong cyclic order when constructing a surface patch. It can be realized with the help of the above growing process. During the process, the tetrahedra adjacent to any other paths or curves must not be added into the grown region (see Figure 9). Similarly, for those tetrahedra in the wrong sectors (as illustrated in Figure 3), they are not added. As a result, the initial solid S for peeling does not include any such tetrahedra. Thus, the surface patch constructed by peeling does not intersect any existing curves or patches, and will not fall in the wrong sectors. In order to bypass topological obstacles, we sometime need to refine the tetrahedra between the loop C and the existing curves or patches.

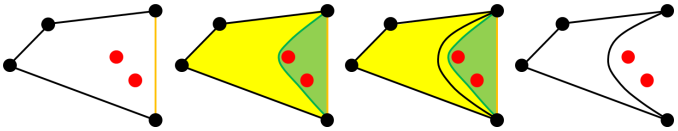

Figure 10: (From left to right) Constructing a trial patch (orange) to complete a C-poly which encloses two anchor points (red). Growing a genus-zero solid around the orange patch that subdivides the current C-poly into two regions: green and yellow, where green region contains all the red anchor points. Compute a new patch within the yellow region. The new patch forms a C-poly does not induce any blocking.

3.3. Restriction for non-blocking

The only left problem is how to prevent blocking when constructing the last patch of a C-poly. Blocking means the C-poly encloses some other anchor points that do not belong to the C-poly. Whether the aforementioned topological blocking happens after inserting the last patch for the C-poly can be easily checked by exploring all the tetrahedra inside the C-poly to see if there are any anchor points embedded. In the cases that lead to blocking, this last patch should be generated in a different way. As illustrated in Figure 10, we can apply the previous growing based method to construct a genus-zero solid inside the C-poly around the last patch without including any faces in other patches. The growing stops when all anchor points that should be excluded have been enclosed in the temporarily grown solid (see the green region in Figure 10). Then, a new patch is constructed in the rest region of the previously constructed C-poly (e.g., the yellow region in Figure 10). By this method, a new C-poly will be generated by excluding all anchors that should not be enclosed – i.e., the topological blocking problem can be solved.

4. Operators for Reducing Distortion

The boundary curves or patches of a template of CBD are represented by straight edges and facets in Ψ. However, the boundary curves and patches constructed by our method presented above are in zigzag and terrain-like shapes, which will introduce unwanted distortion in the volumetric cross-parameterization. Motivated by the smoothing operator used in [5], two operators are incorporated into the domain construction framework to reduce the distortion caused by the shape of base-domains. Note that, the operators will not be applied to the vertices on the surfaces of input models.

4.1. Curve stretching

The purpose of curve stretching is to make a boundary curve $\widetilde{\mathbf{a}_i\mathbf{a}_j}$ linking two anchor points $\mathbf{a}_i$ and $\mathbf{a}_j$ as straight as possible while staying inside the input solid M. The curve stretching operator achieves this goal through the iterative position update

of vertices. For each vertex $\mathbf{v}_k$ located on $\widetilde{\mathbf{a}_i\mathbf{a}_j}$, suppose $\mathbf{v}_{k-1}$ and $\mathbf{v}_{k+1}$ are its two adjacent vertices on the curve, we apply

$$\mathbf{v}_k \leftarrow \mathbf{v}_k + \lambda((\mathbf{v}_{k-1} + \mathbf{v}_{k+1})/2 - \mathbf{v}_k) \quad (2)$$

where λ is a relaxation factor with a value of 0.25. The update is forbidden if the new position of $\mathbf{v}_k$ makes any of its adjacent tetrahedra degenerate. For the vertices that are within three-ring neighbors of $\widetilde{\mathbf{a}_i\mathbf{a}_j}$ and are not on any other curves or patches of C-polys, their positions are updated by the relaxed uniform Laplacian. Again, any update that will lead to degeneration of tetrahedra is prevented. The above steps are repeated until the updates of vertices are trivial.

4.2. Patch stretching

The patch stretching operator makes the boundary patch of C-polys smooth by position updating, which is similar to curve stretching. For a vertex $\mathbf{v}$ on the surface patch P of a C-poly,

$$\mathbf{v} \leftarrow \mathbf{v} + \lambda(\mathbf{v}_{avg} - \mathbf{v}) \quad (3)$$

with $\lambda = 0.25$ being the relaxation factor and $\mathbf{v}_{avg}$ being the average position of $\mathbf{v}$'s one-ring neighbors on P. For vertices that are three-ring neighbors of the surface patch and are not on any other curves or patches of C-polys, their positions are updated by the relaxed uniform Laplacian. We repeat the above steps until the updates of vertices are trivial. Similarly, the update of a vertex is forbidden if it makes any of its adjacent tetrahedra degenerate.

Notice that, preventing the degeneration in tetrahedra during position update can also avoid intersections between curves and patches of C-polys. Incorporating curve stretching and patch stretching in the domain construction algorithm can greatly reduce the distortion of volumetric cross-parameterization (see Figure 11 for an example).

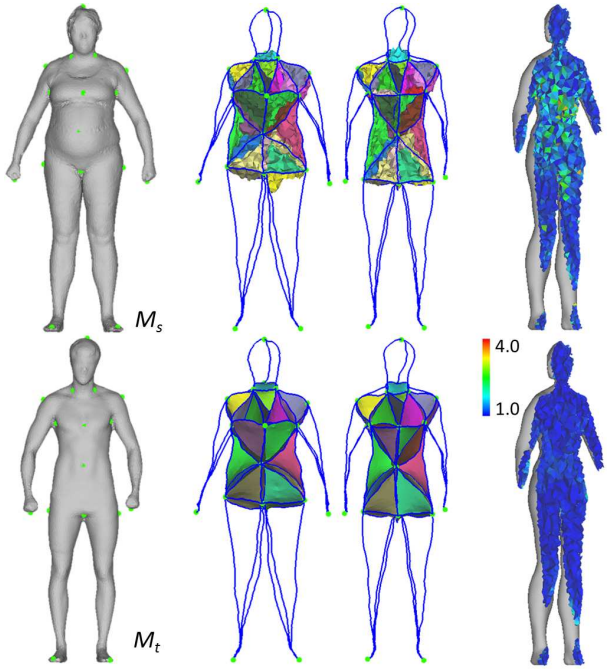

Figure 11: An example to demonstrate the functionality of stretching operators to reduce distortion in volumetric cross-parameterization between two human models M_s and M_t. The top row shows the result of domain construction without stretching. The bottom row gives the result with stretching. The color maps display the L^2-stretch [28] of tetrahedra transformed by the corresponding cross-parameterization, where the value 1.0 stands for no distortion.

5. Results and Applications

This section studies the results generated by our approach and demonstrates its performance in a variety of applications.

Given a set of models, the method proposed in this paper can construct a set of base-domains according to the connectivity of a template which is constrained by anchor points. This is useful to volumetric blending and remeshing. Figure 2 gives an example of blending the interior structure between models with a variety of shapes. Our method can decompose the shape of a complex model into sub-domains to generate bijective mappings (e.g., the hand models shown in Figure 12).

Once volumetric mapping is established between models, point-to-point correspondences are well defined. Therefore, after establishing volumetric cross-parameterization between two models, we can easily transfer all the interior structures and details from one model to another. Two scanned human models are shown in Figure 13 as an example. The bones and organs can be warped from one model to another. Compared with other shape transformation techniques (e.g., t-FFD [29]), the bijective mapping constructed by our method can effectively prevent self-intersection.

Not only interior structures but also the clothes worn on human bodies can be transformed by bijective volumetric cross-parameterization. This leads to an important application in the apparel industry – design reuse. To conduct volumetric cross-parameterization outside human models, we first generate volume meshes between a model and its offset surface, which is similar to the concept of shell map [28]. Constrained volumetric cross-parameterization can be constructed between the solid shells. As shown in Figure 14, the mapping constructed by our method is intersection-free while the result from t-FFD [29] (or spatial warping [13]) has self-intersections in the regions with high curvature (such as under the crotch). Again, the shape transformation supports non-manifold entities – see the region above the arm.

Hexahedral mesh is always demanded in *Finite Element Analysis* (FEA) or *Computational Fluid Dynamics* (CFD) analysis. With the help of our framework, after establishing the volumetric cross-parameterization on tetrahedral meshes, the hexahedral mesh can be transformed from multiple cubic regular shapes into other freeform models. Examples can be found in Figs.15 and 16.

The input constraints given to our method can be specified inside the volumes, which provides a better control for the quality of cross-parameterization. An example is shown in Figure 17. Parameterizing a cubic domain to a spherical region without adding interior anchor points may warp the L-shape model into an obstacle (in yellow) – see the top row of Figure 17. To

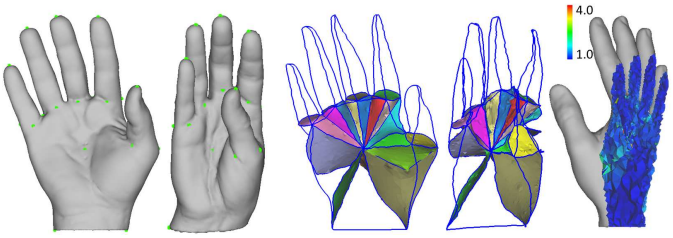

Figure 12: An example of constrained volumetric cross-parameterization in highly concave models, where common base-domains constructed by our method are also shown. The color map gives the L^2-stretch [28] of tetrahedra in the blended solid model.

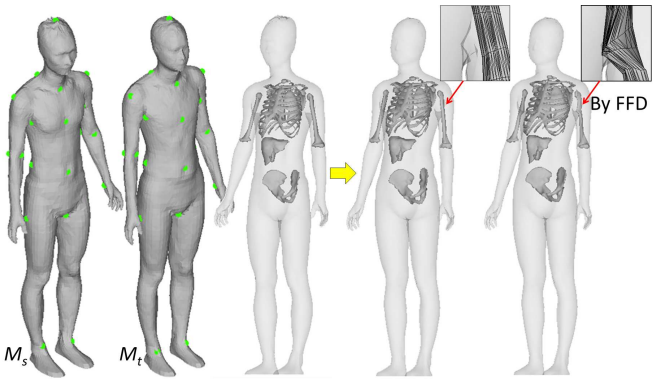

Figure 13: An example of structure transformation between two human models. The volumetric cross-parameterization constructed by our approach is a bijective mapping, thus prevents self-intersections which are common problems on the spatial-warping-based method (such as t-FFD [29]).

avoid such cases, we can add anchor points to the surface of the obstacle. With these constraints, the L-shape model will be outside the obstacle. The feature alignment method proposed by Li et al. [21] can achieve a similar goal. However, if the structures are highly complicated – which is common in real applications – it is hard to find out the surface parameterization required by their method. Furthermore, the constraints are exactly enforced in our approach while fitting errors are generated in the approximate fitting based methods.

We have also studied how the volumetric cross-parameterization is affected by the quality of tetrahedral meshes used in our algorithm. In Figure 18, we construct the cross-parameterization between a cube model to other two cubes with different volumetric meshes. The target model 1 has a mesh with similar quality as that of the source model, and the target model 2 has a much coarser mesh. Without applying the stretching operators introduced in Section 4, the denser volumetric mesh leads to a result of cross-parameterization with less distortion. This is because that a smoother patch will be generated on the denser mesh by our patch construction algorithm. After applying the stretching operators, similar level of distortion is generated on the mesh models with different density.

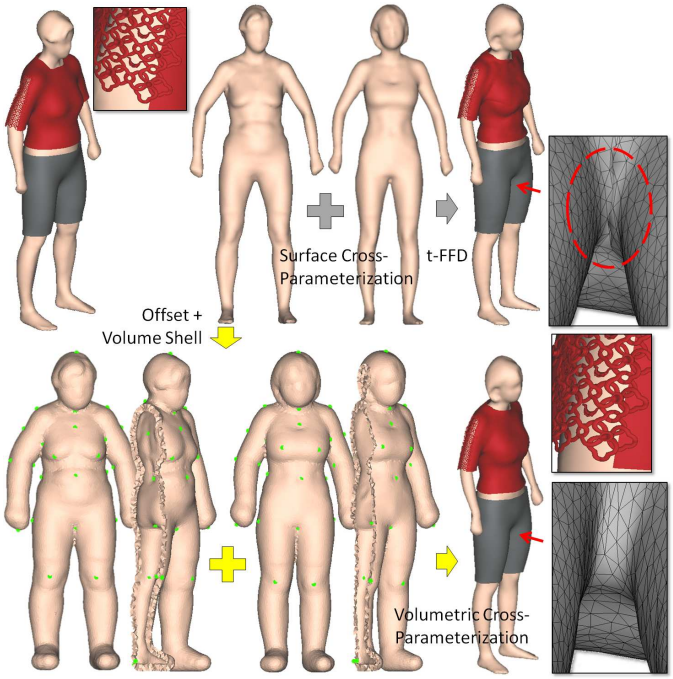

Figure 14: Having volumetric shell constructed between a model and its offset surface, the mapping of cross-parameterization can be computed for the shells. This mapping is bijective. As a result, the transformed clothes are intersection-free.

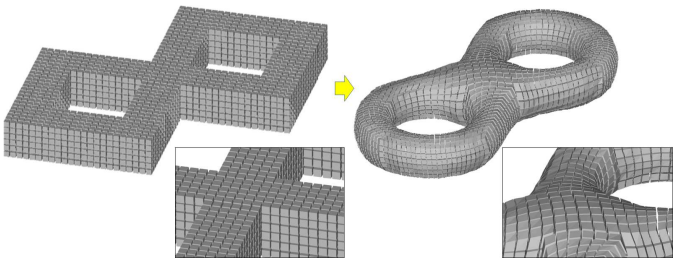

Figure 15: Volumetric cross-parameterization constructed by our approach between genus-two models in Figure 1 can be used in the hexahedral remeshing.

6. Conclusion and Discussion

In this paper, we propose a new method for the construction of CBD used in the volumetric cross-parameterization. The volumetric cross-parameterization can be established through the base-domains constructed by our approach. The mapping can be controlled in a very flexible manner by adding and adjusting anchor points, which are strictly interpolated in the mapping. The anchor points are allowed to be placed both on the boundary surface and inside the solid models. Examples and applications shown in this paper have demonstrated the functionality of our approach.

Limitation of this algorithm comes from three aspects. First, the number of anchor points required to generate a successful domain decomposition depends on the topology of models to be parameterized. Basically, more anchor points are required for high genus models. For example, for the genus zero mod-

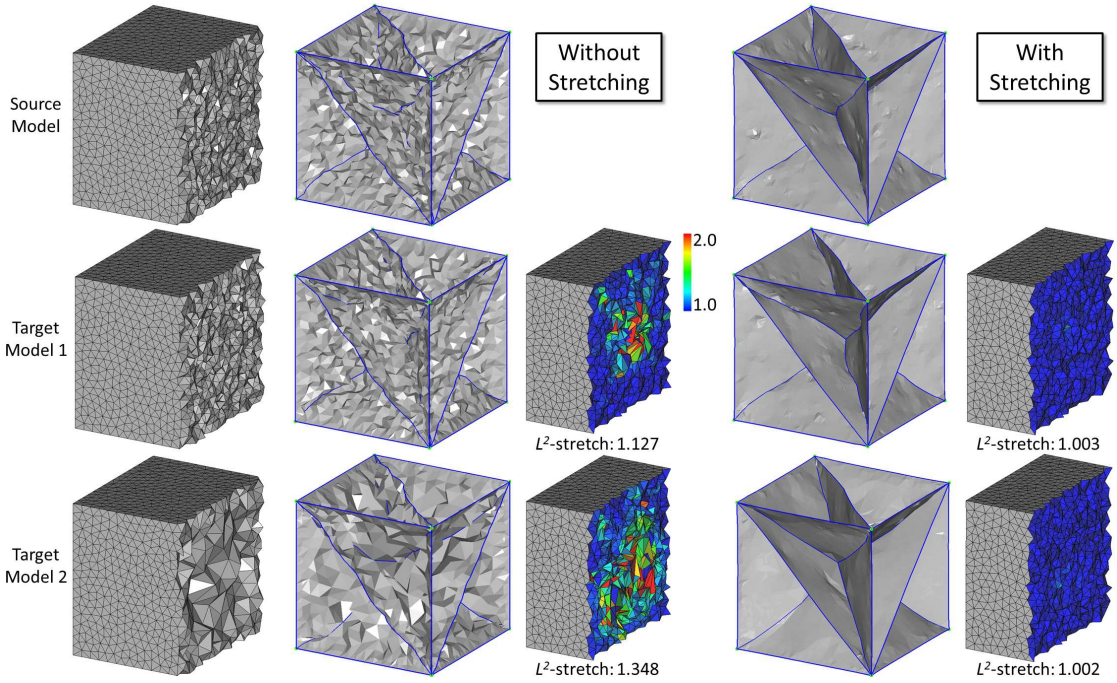

Figure 18: A study on the performance of the volumetric cross-parameterization on tetrahedral meshes with different quality. Target model 1 has a mesh with similar density as the source model, and target model 2 has a much coarser mesh. It is easy to find that cross-parameterization with similar distortion level can be generated after applying the stretching operators in Section 4.

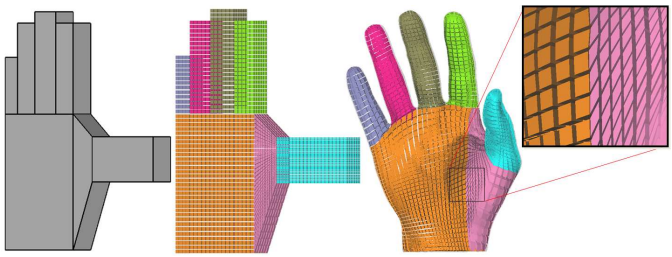

Figure 16: Structured hexahedral mesh generation on a hand model with the help of volumetric cross-parameterization.

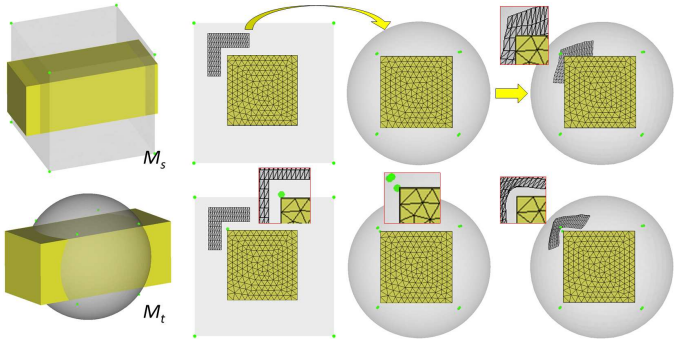

Figure 17: When mapping a cubic domain to a spherical region, adding interior anchor points can prevent the L-shape model from being warped into the obstacle shown in yellow.

els shown in Figure 2, only four anchor points are necessary to generate successful decomposition (although may have large distortion). However, for the genus one model in Figure 1, four anchor points are definitely not enough. Second, similar to surface cross-parameterization approaches, we assume that the corresponding anchor points are located consistently at the meaningful places. Third, the computation of domain boundary is still expensive in terms of time and memory consumption. Take the hand model shown in Figure 12 with 125k tetrahedra as an example, computing the base-domains constrained by 21 anchor points needs around 2 minutes on a PC with 2.66GHz CPU, and 250MB memory is used at the peak time. Our future work will focus on how to overcome these limitations. Some other possible future works include how to use this technique in isogeometric analysis (e.g., [30]), and how to use mean-curvature flow (e.g., [31, 32]) to further enhance the stretching operations.

References

[1] X. Li, Y. Bao, X. Guo, M. Jin, X. Gu, H. Qin, Globally optimal surface mapping for surfaces with arbitrary topology, IEEE Trans. Visual. Comput. Graph. 14 (4) (2008) 805–819.

[2] N. Ray, W. Li, B. Lévy, A. Sheffer, P. Alliez, Periodic global parameterization, ACM Trans. Graph. 25 (4) (2006) 1460–1485.

[3] A. Khodakovsky, N. Litke, P. Schröder, Globally smooth parameterizations with low distortion, ACM Trans. Graph. 22 (3) (2003) 350–357.

[4] E. Praun, W. Sweldens, P. Schröder, Consistent mesh parameterizations, in: ACM SIGGRAPH Papers, 2001, pp. 179–184.

[5] V. Kraevoy, A. Sheffer, Cross-parameterization and compatible remeshing of 3d models, ACM Trans. Graph. 23 (3) (2004) 861–869.

[6] J. Schreiner, A. Asirvatham, E. Praun, H. Hoppe, Inter-surface mapping, ACM Trans. Graph. 23 (3) (2004) 870–877.

[7] T.-H. Kwok, Y. Zhang, C. C. L. Wang, Constructing common base domain by cues from Voronoi diagram, Graphical Models 74 (4) (2012) 152–163.

[8] X. Li, X. Guo, H. Wang, Y. He, X. Gu, H. Qin, Harmonic volumetric mapping for solid modeling applications, in: Proceedings of the ACM symposium on Solid and Physical Modeling, SPM '07, 2007, pp. 109–120.
[9] X. Li, X. Guo, H. Wang, Y. He, X. Gu, H. Qin, Meshless harmonic volumetric mapping using fundamental solution methods, IEEE Transactions on Automation Science and Engineering 6 (2009) 409 – 422.
[10] Y. Wang, X. Gu, T. F. Chan, P. M. Thompson, S.-T. Yau, Volumetric harmonic brain mapping, in: Proceedings of the 2004 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, IEEE, 2004, pp. 1275–1278.
[11] J. Xia, Y. He, X. Yin, S. Han, X. Gu, Direct-product volumetric parameterization of handlebodies via harmonic fields, in: Shape Modeling International Conference (SMI), 2010, pp. 3–12.
[12] J. Xia, Y. He, S. Han, C.-W. Fu, F. Luo, X. Gu, Parameterization of star-shaped volumes using Green's functions, in: Advances in Geometric Modeling and Processing, Vol. 6130, Springer Berlin / Heidelberg, 2010, pp. 219–235.
[13] C. C. L. Wang, K.-C. Hui, K.-M. Tong, Volume parameterization for design automation of customized free-form products, IEEE Transactions on Automation Science and Engineering 4 (1) (2007) 11–21.
[14] W. Song, L. Liu, Stretch-based tetrahedral mesh manipulation, in: Proceedings of Graphics Interface 2007, GI '07, ACM, New York, NY, USA, 2007, pp. 319–325.
[15] H. Si, TetGen: A quality tetrahedral mesh generator and a 3D delaunay triangulator, http://tetgen.org (2011).
[16] M. S. Floater, G. Kós, M. Reimers, Mean value coordinates in 3d, Computer Aided Geometric Design 22 (2005) 623–631.
[17] A. Sheffer, E. Praun, K. Rose, Mesh parameterization methods and their applications, Foundation and Trends in Computer Graphics and Vision 2 (2) (2006) 105–171.
[18] K. Hormann, B. Lévy, A. Sheffer, Mesh parameterization: theory and practice, in: ACM SIGGRAPH courses, 2007.
[19] T. Ju, S. Schaefer, J. Warren, Mean value coordinates for closed triangular meshes, ACM Trans. Graph. 24 (3).
[20] M. Floater, Mean value coordinates, Computer Aided Geometric Design 20 (1) (2003) 19–27.
[21] X. Li, H. Xu, S. Wan, Z. Yin, W. Yu, Feature-aligned harmonic volumetric mapping using MFS, Computers & Graphics 34 (3) (2010) 242–251.
[22] T. Martin, E. Cohen, R. M. Kirby, Volumetric parameterization and trivariate B-spline fitting using harmonic functions, Computer Aided Geometric Design 26 (6) (2009) 648–664.
[23] J. R. Shewchuk, Constrained delaunay tetrahedralizations and provably good boundary recovery, in: In Eleventh International Meshing Roundtable, 2002, pp. 193–204.
[24] R. Kimmel, J. A. Sethian, Computing geodesic paths on manifolds, in: Proceedings of National Academy of Science, 1998, pp. 8431–8435.
[25] X. Wang, F. F. Cheng, B. A. Barsky, Energy and B-spline interproximation, Computer-Aided Design 29 (7) (1997) 485–496.
[26] G. Taubin, A signal processing approach to fair surface design, in: Proceedings of the annual conference on Computer graphics and interactive techniques, 1995, pp. 351–358.
[27] M. Wardetzky, S. Mathur, F. Kälberer, E. Grinspun, Discrete laplace operators: no free lunch, in: Proceedings of Eurographics Symposium on Geometry processing, SGP, 2007, pp. 33–37.
[28] K. Zhou, X. Huang, X. Wang, Y. Tong, M. Desbrun, B. Guo, H.-Y. Shum, Mesh quilting for geometric texture synthesis, ACM Trans. Graph. 25 (3) (2006) 690–697.
[29] K. G. Kobayashi, K. Ootsubo, t-FFD: free-form deformation by using triangular mesh, in: Proceedings of the ACM symposium on Solid Modeling and Applications, 2003, pp. 226–234.
[30] G. Xu, B. Mourrain, R. Duvigneau, A. Galligo, Analysis-suitable volume parameterization of multi-block computational domain in isogeometric applications, Computer-Aided Design 45 (2) (2013) 395–404.
[31] K. Hildebrandt, K. Polthier, Anisotropic filtering of non-linear surface features, Computer Graphics Forum 23 (3) (2004) 391–400.
[32] H. Pan, Y.-K. Choi, Y. Liu, W. Hu, Q. Du, K. Polthier, C. Zhang, W. Wang, Robust modeling of constant mean curvature surfaces, ACM Trans. Graph. 31 (4) (2012) 85:1–85:11.