Feature-Preserving T-mesh Construction Using Skeleton-based Polycubes

Lei Liu[a], Yongjie Zhang[a,*], Yang Liu[b], Wenping Wang[c]

[a]*Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[b]*Microsoft Research, Beijing, China*
[c]*The University of Hong Kong, Hong Kong, China*

Abstract

This paper presents a novel algorithm which uses skeleton-based polycube generation to construct feature-preserving T-meshes. From the skeleton of the input model, we first construct initial cubes in the interior. By projecting corners of interior cubes onto the surface and generating a new layer of boundary cubes, we split the entire interior domain into different cubic regions. With the splitting result, we perform octree subdivision to obtain T-spline control mesh or T-mesh. Surface features are classified into three groups: open curves, closed curves and singularity features. For features without introducing new singularities like open or closed curves, we preserve them by aligning to the parametric lines during subdivision, performing volumetric parameterization from frame field, or modifying the skeleton. For features introducing new singularities, we design templates to handle them. With a valid T-mesh, we calculate rational trivariate T-splines and extract Bézier elements for isogeometric analysis.

Keywords: Skeleton, Polycube, Frame Field, Feature Preservation, T-mesh, Trivariate T-splines, Isogeometric Analysis

1. Introduction

Isogeometric analysis is a novel analytical method which has attracted a lot of attention recently [1, 2], with its advantage of accuracy and robustness studied in detail [3, 4]. NURBS models were first used in various isogeometric analysis [2, 5, 4], and then T-splines [6] were incorporated for its local refinement property [7, 2]. Based on these pioneering research, an isogeometric design-through-analysis methodology [8] was proposed, with the purpose of integrating the whole process from designing of models to the analysis results. However, automatically and robustly constructing trivariate T-spline models is still a challenging problem.

Only a few approaches have been designed for volumetric spline construction. Parametric mapping plays an important role in this research area, such as trivariate B-spline fitting using harmonic functions [9], parametric mapping of tetrahedral meshes [10], and Periodic Global Parameterization [11]. Polycubes and parametric mapping were used together to generate solid T-spline models [12, 13], and Boolean operations were further introduced to manipulate polycubes [14]. A generalized polycube method using T shape templates was introduced to handle high-genus models and extraordinary nodes for T-spline construction [15]. The generalized polycubes were further extended in [16] to generate volumetric splines from surface meshes, with no singularity and controllable number of ill-points. However, in these polycube-based methods, the generated solid T-splines follow the directions of the initial polycubes and certain detailed features cannot be preserved in the final result.

Since T-mesh can be recognized as a special type of hexahedral (hex) mesh which allows T-junctions, we may apply some promising hex meshing algorithms to solid T-spline construction. CubeCover [17] used 3D frame fields to perform volumetric parameterization and all-hex mesh generation. This research was extended in [18], which brought forward a method to generate a singularity-restricted frame field for all-hex meshing. A boundary aligned cross-field was also studied in [19], which uses spherical harmonics to represent the 3D field. The field was then improved with singularity correction for hex mesh extraction [20]. However, performing volumetric parameterization from frame field is not robust, especially for complex geometry. And the frame field may yield very dense hex meshes with singularity restrictions and corrections. Harmonic volumetric mapping was employed in hex meshing with better boundary feature capture [21]. Common base-domains [22] were designed for volumetric parameterization of models with homeomorphic topology. Polycubes were also used to construct hex meshes [23]. A constrained discrete optimization technique was developed in [24] for better mesh segmentation and volumetric parameterization using polycubes. In [25], hex remeshing was performed based on polycube construction and optimization. L_1 based polycubes for complex geometries were proposed for hex meshing with better quality [26]. Octree-based methods were developed to generate adaptive hex dual-meshes [27, 28], and were improved to preserve sharp features [29, 30]. However, they yield many singular points on the surface.

To robustly build T-meshes following the geometry topology and preserve detailed features, in this paper we develop a new algorithm to use the skeleton as a guide for polycube construction. In the generated polycubes, each cube has at most one patch on the boundary, which provides the necessary condition to generate

[*]Corresponding author: Y. Zhang. Tel: (412) 268-5332; Fax: (412) 268-3348; E-mail address: jessicaz@andrew.cmu.edu.

good-quality elements. The singularity graph of the constructed T-mesh follows the cube edges in the interior. Instead of singular edges, there are only a few singular points on the surface. With this property, we can control the singularity distribution on the constructed solid T-spline. For surface features, we classify them into three groups: open curves, closed curves and singular features, and design different schemes to preserve them.

The main contribution of this paper lies in a new skeleton-based polycube construction method, and different approaches to preserve surface features. Compared to other methods, our algorithm has the following three unique properties: (1) the constructed T-splines follow the topology of the input model, and all singular edges lie on the polycube edges in the interior; (2) open and closed curve features are preserved with different methods, and Boolean operations are introduced to simplify the T-mesh construction; and (3) three templates are developed to introduce and preserve certain singular points on the surface.

The remainder of this paper is organized as follows. The main steps of the algorithm are overviewed in Section 2. Section 3 discusses how to generate polycubes and split the domain. Different approaches of feature preservation are described in Section 4. The results are shown in Section 5. Section 6 draws conclusions and points out the future work.

2. Algorithm Overview

The overview of our algorithm is shown in Fig. 1. We use polycubes to split the domain and perform parametric mapping to construct the T-mesh. With the skeleton generated from a mean curvature flow algorithm [31], we split the skeleton into different branches. Each branch yields one interior cube and several boundary cubes. These cubes split the domain into different cubic regions. From the input model, we classify the surface features into three groups and preserve them with different approaches.

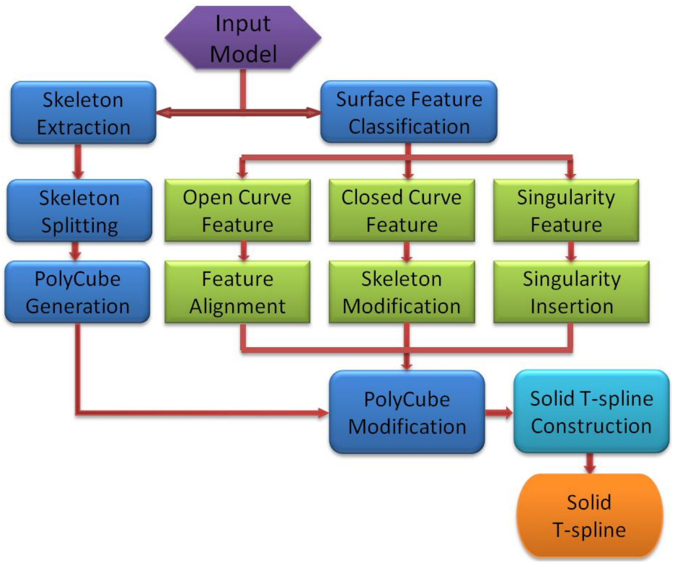


Figure 1: Overview of feature preservation in skeleton-based polycube construction and volumetric parameterization.

Polycube Construction. We use the generalized cube definition here. For a generalized cube, it is one boxed region enclosed by six surface patches. There are two kinds of cubes here, interior cube and boundary cube. We first construct interior cubes directly from the skeleton branches, and then project their corners onto the surface to generate new boundary cubes. Interior and boundary cubes are combined together to split the whole model into different cubic regions. Since for the boundary cubes, there are at most one face on the surface, all the singularity edges from the cubes stay in the interior.

Feature Preservation. The input surface features are classified into three groups: open curve, closed curve and singularity feature. Here, the open curve is required to satisfy the condition that it can be mapped onto one certain parametric line. We use parametric mapping and volumetric parameterization from frame field to preserve such features. The closed curve is required to topologically enclose a disc area and each closed curve can be mapped onto a unit square. To preserve such features, the skeleton is modified to add or remove branches. The singular feature is a singular point on the surface. It can be a sharp corner, saddle point of a function, or point with discontinuous curvatures. We develop three templates to insert new singularity points on the surface. With the modified polycube containing surface features, we construct T-meshes by octree subdivision and projection [12, 14].

Solid T-spline Construction. From the T-mesh, we build rational solid T-splines [32]. The basis function of rational solid T-splines has the property of partition of unity by definition, which makes it suitable for analysis. Different templates are developed to deal with the singular nodes in the T-mesh, to make it valid for gap-free solid T-spline calculation. From the valid T-mesh, we extract the local knot vectors [6, 33] and construct solid T-splines. The Oslo knot insertion algorithm [34, 33] is used to calculate the transformation matrix from rational T-spline basis functions to Bézer basis functions. This matrix is then used to extract Bézier elements from T-splines, which can be directly used for isogeometric analysis.

3. Skeleton-based Polycube Construction

Skeletons are simplified 1D representation of 3D objects, which can reflect the geometry and topology. They contain geometrical information for volumetric parameterization and can be used to assist our polycube construction.

3.1. Skeleton Generation and Splitting

There are different algorithms developed to extract skeletons from surface meshes, such as mesh contraction [35], mean curvature flow [31], and the generalized sweeping method [36]. In this paper, we use the algorithm given in [31]. With the skeleton, we first split it into different branches. For each branch, we define a B-spline curve and calculate the tangent direction at each point. We decide if it needs further splitting by calculating the angle change of the tangent directions at each point compared to the starting point: $\theta = acos(\vec{t_0} \cdot \vec{t_i})$. A predefined threshold $\theta_0 = 30^\circ$ is used here. Some user interactions may be involved to simplify

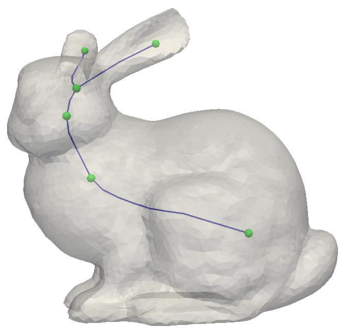
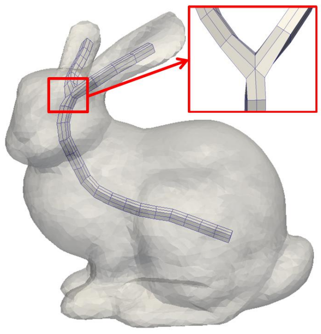
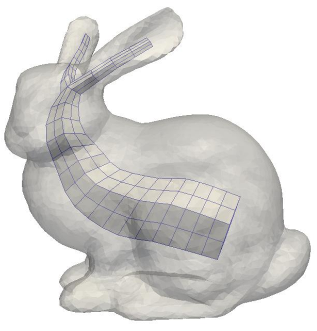

(a)

(b)

(c)

Figure 2: Polycube generation for Bunny model. (a) Skeleton splitting results; (b) generating interior cubes by shifting the skeleton branches; and (c) updated interior cubes by iteratively enlarging the cross-sections and smoothing.

the skeleton in this step, such as cleaning up small branches, combining nearby bifurcations to trifurcations, or making the local branching region coplanar. Fig. 2(a) shows the extracted skeleton and splitting result of the Bunny model.

3.2. *Interior Cube Construction*

To construct a generalized cube from one skeleton branch, we need to generate its 6 bounding patches. These patches can be either planar or curved surfaces. We first shift the branches about itself 8 times to generate 20 curves, as shown in Fig. 3(a). For each point on the skeleton, we generate one plane perpendicular to the skeleton, and then calculate 8 equally-spaced direction vectors on this plane to perform the shifting. Sometimes this method may produce interior cubes with improper orientations, which can be adjusted interactively to yield good parameterization results. The black curve is the original branch, the 8 blue curves are generated from shifting, and the 8 green curves and 4 red curves are generated by connecting the starting/ending points of the shifted curves. These curves are defined as quadratic B-spline curves. With four B-spline curves, we define one Coons patch [37]. So for a skeleton, we generate 6 patches from the 20 curves. With these 6 patches, we define a cubic domain.

Deal with Branches. To join cubes from different branches together at the bifurcation or trifurcation, we split the cube patches to half planes and combine them together. The detailed algorithm was present in [4, 38]. During this process, singularities will be introduced to the polycubes along the shared edges of the half planes. Fig. 3(b) and (c) show how to combine the cubes at the bifurcation and trifurcation situations. Instead

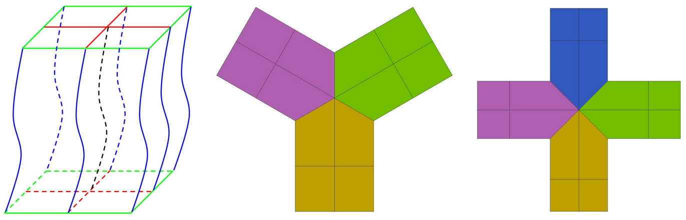

(a) (b) (c)

Figure 3: Construction of interior cubes. (a) Generate an interior cube by shifting the skeleton; (b) use half planes to deal with bifurcation; and (c) trifurcation.

of Coons patches, the half planes at the intersection region are defined as planar patches, and points on the plane are calculated from a linear interpolation of the corner points. Fig. 2(b) shows the generated interior cubes of the Bunny model.

After all the interior cubes are connected properly, we iteratively enlarge each cross-section of the cubes to adapt to the input model. For each node, we project it onto the surface along the radial direction from the cross-section center. Smoothing is performed to reduce the distortion from enlargement [27]. The enlarged and smoothed interior cubes of the Bunny model are shown in Fig. 2(c).

3.3. *Boundary Cube Construction*

With the interior cubes constructed, we can generate the boundary cubes to split the whole model. The boundary cubes are generated by projecting the patches of the interior cubes onto the surface. The detailed steps are explained as follows with one patch of a sphere model in Fig. 4 as an example.

1. **Project corners onto the surface.** For one corner c_i of the interior cube, if shared by one cube, the projection direction is defined as $\overrightarrow{d} = -(\overrightarrow{u} + \overrightarrow{v} + \overrightarrow{w})$ (Fig. 4(b)), where $\overrightarrow{u}, \overrightarrow{v}, \overrightarrow{w}$ are the unit direction vectors along the edges at c_i. If shared by two cubes, the direction is $\overrightarrow{d} = -(2\overrightarrow{u} + \overrightarrow{v_1} + \overrightarrow{v_2} + 2\overrightarrow{w})$ (Fig. 4 (c)). For bifurcation or trifurcation, the projection direction is perpendicular to the plane defined by the skeleton branches at that intersection point.
2. **Generate curves.** Suppose the corresponding point of c_i on the surface is c'_i, see Fig. 4(a). The curve connecting c_i and c'_i is named a *connecting curve* (blue curves). For one curve of the interior cube (black curves), we can find out one *projected curve* on the surface (red curves) by finding the geodesic shortest path between the projected corners. After projecting 4 corners of the interior patch, we define 4 connecting curves and 4 projected curves. Intersections are not allowed between any pair of boundary curves except at the endpoints. So when finding the path, vertices lying on the path between two projected corners will not be revisited. If the projected corners are far away from each other, or the geometry changes severely, we can project the middle or quarter points of the interior curves onto the surface, and use them to help find the path. However, if the projected corners are crowded or the valence of the projected corners is low, we may locally subdivide the input mesh to ensure there is no intersection among the paths. This projection curve searching step is crucial to our surface decomposition.

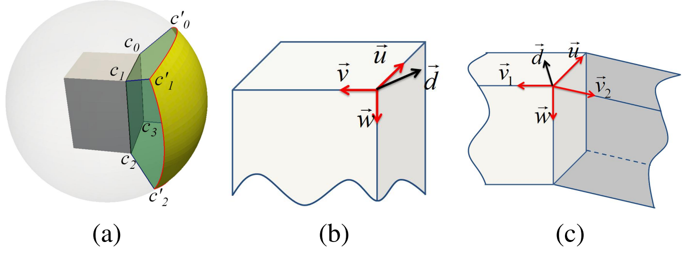


Figure 4: Construct boundary cubes from an interior cube. (a) Sphere model with one patch of the interior cube projected onto the surface; (b) projection direction of an interior cube corner; and (c) the projection direction if a corner is shared by two cubes.

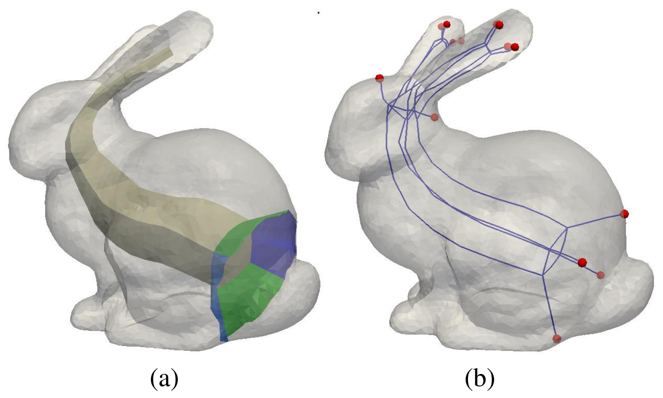


Figure 5: Four connecting patches of Bunny model after optimization (blue and green patches); and (b) its singular graph (red dots represent singular points on the surface).

Our method works well in general, but some improved surface splitting methods like the greedy strategy [39] can help generate better results for very complicated models.

3. **Build patches.** We define a connecting Coons patch with an interior curve, its corresponding boundary curve and two connecting curves (light green patches in Fig. 4(a)). Four connecting patches will be generated after the projection of an interior patch. The four boundary curves define one boundary patch (yellow patch in Fig. 4(a)). For the boundary patch, instead of using Coons patch, we directly use the surface region surrounded by these four curves.
4. **Generate boundary cubes.** With each interior patch, its corresponding boundary patch and four connecting patches, we define the enclosed domain as a boundary cube.

For an interior cube, depending on whether the bounding patches are shared by other cubes, it can generate at most 6 new boundary cubes. For one boundary cube, it shares one interior patch with the interior cube from which it is derived, and has only one bounding patch on the surface. We can calculate a series of points on the Coons patches by giving $m \times n$ pairs of parametric values, and use them for parametric mapping and octree subdivision. The connecting patches may be distorted if the surface is bumpy or has a lot of features. We can optimize the control points of interior and boundary curves. To perform optimization, we should first unify the number of control nodes on the two opposite sides of one patch, then generate one coarse hex mesh. With this hex mesh, we optimize the control points by moving them toward the direction which can produce the maximum scaled Jacobian [12]. Fig. 5(a) shows the four connecting patches generated and optimized from one interior cube of the Bunny model.

With the interior and boundary cubes, we can split the model into different sub-domains. This domain splitting result follows the skeleton of the input model and thus the generated T-mesh follows the topology of the input. If we want to change the orientation or the number of cubes, we can simply modify the skeleton at the beginning. In addition, the location of the projected cube corners on the surface can be optimized to help generate better parameterization results [40].

3.4. *Singularity of Polycubes*

An interior cube edge is a singular edge if it is not shared by 4 cubes. All the control nodes lying on these singular edges are singular nodes. The singular graph of the T-mesh is the graph which connects all the singular nodes. This graph satisfies the constraint that the singular graph of a hex mesh should not start or end in the interior of the volume [41, 17]. After polycube construction, the singular graph is fixed. We can predict the positions of singular points generated from octree subdivision. Fig. 5(b) shows the singular graph of the Bunny model.

4. Feature Preservation

Surface features, such as smooth curves, sharp curves, and singular points, play an important role in representing the surface details. In our algorithm, feature preservation is carried out during T-mesh construction. For each cube, we project it onto a unit cube in the parametric domain and perform octree subdivision to generate the T-mesh. This T-mesh contains all the information from the input. The detailed projection and subdivision algorithm was present in [12, 13]. The main difference between our T-mesh generation method and previous research on skeleton-based volumetric composition and structured grid generation [42] is that our T-mesh allows T-junctions, and there is no singular edge lying on the surface. We classify surface features into three different groups: open curves, closed curves and singularity points. They are dealt with different approaches.

4.1. *Open Curves*

In this section, two methods are developed to preserve open curves: parametric mapping and volumetric parameterization from frame field. We require that the open curves preserved here can be mapped onto parametric lines. For curves with self-intersection or spiral shape, we have no way to map them onto any parametric line in our algorithm, so we cannot preserve them.

Parametric Mapping. For an open curve feature, we align it to a certain parametric line during parametric mapping. By doing this the generated T-mesh contains a sequence of nodes to follow this curve. If one feature line crosses two different boundary cubes, we would constrain that the shared point on the boundary of these two cubes should be mapped to the same parametric value. Then there would be no discontinuity in the

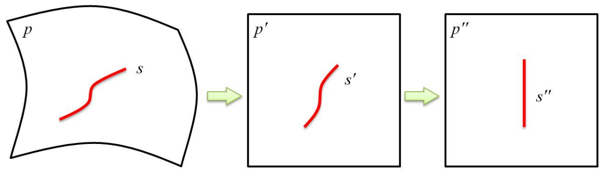


Figure 6: Feature alignment during parametric mapping.

resulting subdivision between the two cubes. The detailed steps are as follows (Fig. 6):

1. For a feature curve s, we first find out the patch p containing it in the cube C and map patch p to a unit square p' in the parametric domain. The feature curve s will be mapped to curve s' on p';
2. Calculate the average angle $\bar{\theta}$ between the tangent direction at each point and the u axis. If $\bar{\theta} < \pi/4$, we align p' to the v direction. Otherwise we align it to the u direction;
3. Set the coordinate at the aligned direction to be the same value for all the points lying on p, calculate the parametric coordinate at the other direction by a chord length parameterization, and then perform surface mapping again to get results with aligned features.

For an open curve within one surface patch, if the tangent directions at the two end points vary a lot (e.g., they form an angle greater than 60°), or the curve intersects with two adjacent boundaries of one patch, we may need to map half of the curve to the parametric u direction and the other half to the v direction. The turning point is C^0-continuous along the curve.

It is convenient to perform the alignment during parametric mapping. However, it is difficult to propagate this feature information into the interior of the T-mesh. This is because nodes on the surface are calculated from mapping and projection, but nodes in the interior are from a linear interpolation [12]. So the deeper into the interior, the less influence the feature information has. As a consequence, it may yield distorted T-mesh elements even with smoothing performed. To resolve this issue, in the following we use parameterization from 3D frame field to preserve these open curve features.

Frame Field. A volume parameterization of geometry $\mathcal{V}$ from frame field can be recognized as an atlas of maps f: $\mathcal{V} \to \mathbb{R}^3$, $p \mapsto (u, v, w)^T$. f is a piecewise linear field in each input tetrahedral mesh element. The integer grids in $\mathbb{R}^3$ would induce a hex tessellation of the geometry. The volume parametrization from the field [17] is performed by:

$$min \sum_t vol \cdot D_t, \tag{1}$$

where

$$D_t = \|c\nabla f(u) - \mathbb{U}_t\|^2 + \|c\nabla f(v) - \mathbb{V}_t\|^2 + \|c\nabla f(w) - \mathbb{W}_t\|^2, \tag{2}$$

vol is the volume of a tetrahedron, c is the length scale of parameterization, and $\{\mathbb{U}_t, \mathbb{U}_v, \mathbb{U}_w\}$ are the initialized frame field.

For a detected feature curve lying in cube C, we use the six patches of the cube to generate a high quality uniform tetrahedral mesh using TetGen [43] and apply a frame field to it. We initialize the frame field cube by cube. For each cube, we first initialize the cross field on the bounding patches with one direction following the patch normal, and then propagate it to the interior. Field optimization is also performed after the propagation. The permutation matrix [17] between any pair of neighboring tetrahedral elements is set to be the identity matrix if they are in the same cube. The permutation matrix among different cubes is set properly to ensure that the shared cube edges are singular edges. During frame field initialization, the feature line information is used to guide the field. Then we perform volumetric parameterization to get an all-hex mesh. This mesh will be used as the initial T-mesh, combined with other cubes for subdivision to generate the T-mesh for the whole model. For one cube, if subdivided by n times without T-junctions, there will be $2^n + 1$ control points at one parametric direction. To make the parameterization result compatible with the subdivision of neighboring cubes, we adjust the length scale c and modify the isoparametric line spacing to perform remeshing.

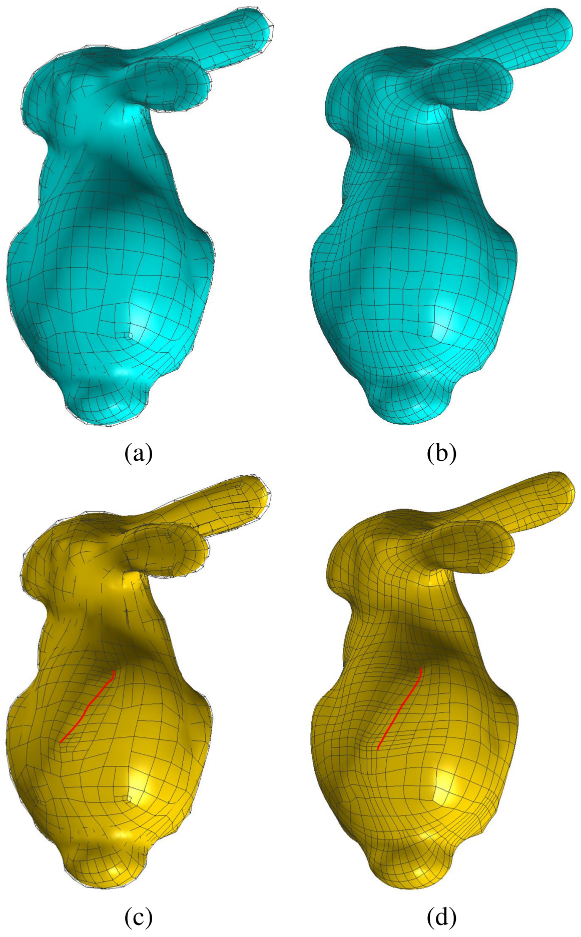


Figure 7: Feature alignment for the Bunny model. (a) T-mesh without feature alignment; (b) Bézier elements without feature alignment; (c) T-mesh with feature alignment; and (d) Bézier elements with feature alignment.

Fig. 7 shows the Bunny model without and with feature alignment. An open curve is preserved on the back of the Bunny. Fig. 8 shows the result of a sphere model with an open curve feature aligned from direct mapping and frame field parameterization. Compared to direct mapping, the frame field parameterization method has the following advantages: (i) the change

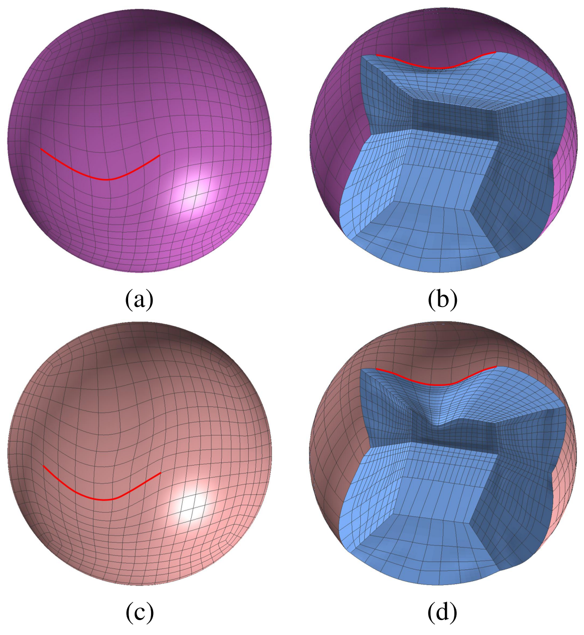


Figure 8: Feature alignment of a sphere model. (a, b) Bézier representation of solid T-spline from mapping and its interior elements; and (c, d) result from frame field parameterization and its interior.

of the element size is gradual, and the influence of the feature line to its surrounding elements is smoother; and (ii) the feature information can propagate further into the interior. As shown in Fig. 8(d), the feature curve even influences the subdivision of the interior cube.

4.2. Skeleton Modification and Boolean Operations

The domain splitting and polycube construction result depend on the skeleton. By modifying the skeleton, we can change the ways of domain splitting and the design of polycube. Since the patches of interior cubes are projected onto the surface to generate boundary cubes, we can generate one boundary cube with the enclosed region by the closed curve as its boundary patch. To build this boundary cube for the closed curve and preserve such a feature, we should add one new branch to the skeleton.

To add a new branch, we find out the center point of the enclosed surface region and connect it to the skeleton. With this branch, a new bifurcation is introduced to the skeleton. After building a new interior cube from the new branch, its four corners away from the bifurcation are projected back onto the closed curve on the surface. These four projected corners split the close curve into four consecutive ones. The region enclosed by the closed curve is defined as a boundary patch and a boundary cube is built. With the modified polycube, we can preserve the closed curves during the following subdivision.

Fig. 9 shows a torus model with skeleton modification. The original skeleton of the torus model is one circle. We modify it by adding one or two new branches. The results show that by modifying the skeleton, we can not only change the domain splitting, but also change the number of singular points on the surface. When one new branch is added, six new singular points are introduced on the surface, two from the bifurcation part and four from the corners of the new branch. Fig. 9(e-f) shows the

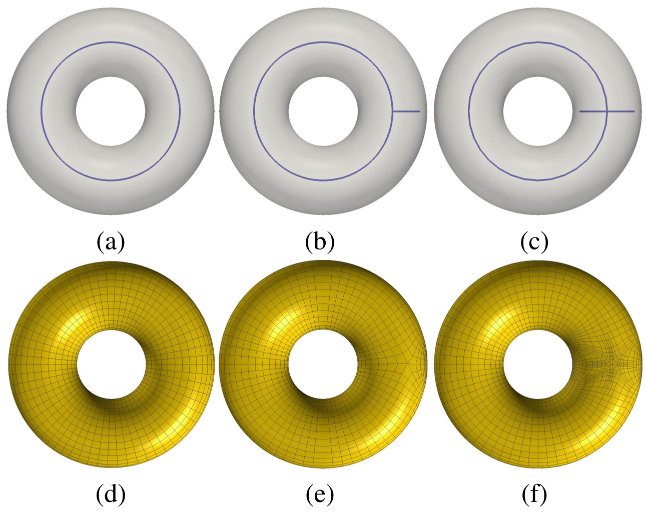


Figure 9: Skeleton modification of a torus model. The skeleton is shown in (a), (b) and (c), where (a) shows the original skeleton, (b) shows the skeleton with one new branch inserted to form bifurcation and (c) shows the skeleton with two new branches inserted to form trifurcation; the corresponding T-spline models with Bézier representation are shown in (d), (e) and (f) respectively.

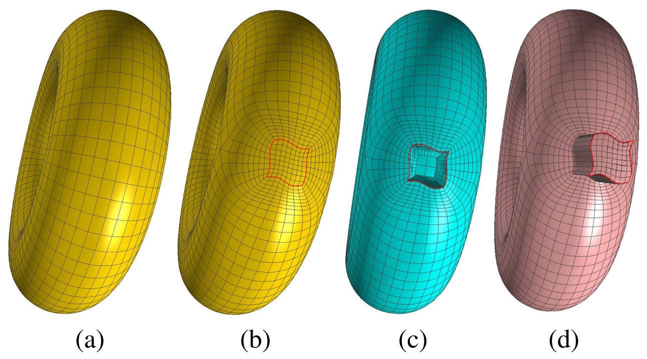


Figure 10: Skeleton modification for a closed curve on a torus model. (a) Original model; (b) preserving one feature region by adding one new branch; (c) removing all the elements generated from the new branch; and (d) extruding the feature curve region.

constructed T-spline models. Fig. 10(b) shows the torus model with one closed curve feature preserved on the surface. The closed curve feature will impact both the elements inside the enclosed region, and those from the neighboring cubes.

Boolean Operations. By combining feature alignment and skeleton modification together, we can perform different kinds of Boolean operations on the generated model, like union and subtraction. Fig. 10(c) shows the modified torus model with all the elements in the feature curve region removed. Fig. 10(d) shows the new torus model with the closed curve region extruded from the surface. With Boolean operations, we can simplify the modeling process with proper skeleton modification.

4.3. Singularity Modification

After generating the interior and boundary cubes, the topology of the singular graph is fixed. If there are other singular points on the surface to be preserved, we may have to regenerate our polycube. As indicated in Section 4.2, modifying the skeleton can change the number of the singular nodes on the surface, but this modification also changes the structure of the polycubes.

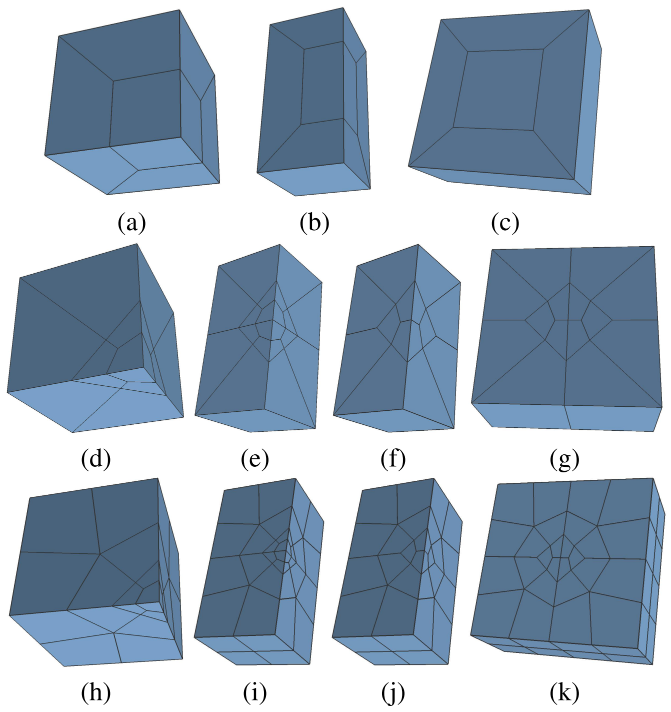


Figure 11: Designing three templates by mirroring, combining and simplifying the primitives. (a-c) Template 1; (d-g) Template 2; and (h-k) Template 3.

To preserve surface singularities without changing the polycube, we design some templates which follow the property of singularity distribution in hex meshes. As indicated in [17], the singular graph should not start or end in the interior of the hex meshes. So the designed template should provide a singular edge path connecting the desired surface singularity to the existing singular graph in the interior. We develop three templates to insert surface singularities. These templates can be applied to boundary cubes or elements containing the desired singularity. The cube or element will be split into smaller elements and new singularities are introduced on the surface and in the interior. The templates are designed with the following two constraints: (a) the introduced face singularities should only lie on the boundary patch of the polycube, or the boundary face of the element; and (b) the four edges of the face containing the face singularity should not be singular edges. These two constraints ensure that the templates will only change the interior region of the cube or element without influencing its neighbors. We design three templates, each of which changes the surface singularity of the polycube differently.

Template 1 is derived from a 2-refinement splitting primitive of unstructured hex meshes [30]. It introduces three singular nodes on three different faces, see Fig. 11(a). We combine four of them together and perform simplification whenever possible. The initial primitive is first extended by combining it with its mirror image corresponding to one face containing face singularity. The face singularity is therefore wrapped into the interior. Simplification is performed by merging elements together, see Fig. 11(b). The simplified mesh is combined with its mirror image again with further simplification to get the final template, as shown in Fig. 11(c).

This template introduces four new singular points on the surface of the initial cube. During mapping we align the singular points in this template with the desired singular points on the surface. This template may change the property of the original four corners on the surface. If these corners are regular points, they will become irregular after applying the template. Otherwise they will switch from singular to regular.

Template 2 uses two of the 13 meshable primitives introduced in [44, 41]. Fig. 11(d) shows two of the solid primitives combined together and their splitting pattern. The splitting pattern of these two primitives guarantees that there is no gap when matching them together. The built cube from the combination has three singular points on three different faces. We use the same merging-simplifying technique to wrap undesired face singular points into the interior. The final template is shown in Fig. 11(g). This template will insert four new collinear singularity points on the surface. The four corners of the original cube are changed in the same way as Template 1.

Template 3 is different from Template 2 in choosing meshable primitives, see Fig. 11(h). As shown in Fig. 11(k), this template will insert eight new singularity points on the surface. Four of them are collinear and the other four form one surrounding circle. The original four corners of the cube are not influenced by this template.

We have applied all the three templates to a sphere model to demonstrate how they change the singularity points on the surface and the singular graph in the interior. Fig. 12 shows the singular graph (blue curves) of the T-mesh with singular points (red point) on the surface, and solid T-spline results. Template 1 does not change the total number of singular points on the surface, but shrink the region enclosed by the four singular nodes. Template 2 erases the original four singular nodes of the boundary cube, and introduces four new collinear singular points. Template 3 does not influence the original four singular points, with four collinear singular points and the other four forming one circular region. All the singular points on the surface are connected to the singular graph in the interior.

For a singular point in T-mesh, it decreases the continuity of T-spline from C^2 to C^0 within its two ring neighborhood. Which template should be chosen to preserve certain surface singularity depends on the size of the two-ring neighborhood influenced by the singular point, and whether the singular information is allowed to propagate outward. The singularity points we insert on the surface are either valence 3 or valence 5. Our templates cannot handle higher-valence singularities.

5. Results and Discussion

We have tested our algorithm on different models based on one machine with an Intel X3470 CPU and 16GB memory. The statistics is given in Tab. 1. We can observe that the generated elements are of good quality. All the models were scaled to make the maximum edge length of the bounding box equal to 1.0. We then calculated the Hausdorff distance between our generated surface and the input boundary. From Tab. 1, we can observe that our resulting parameterization has good surface accuracy. Closed and open curve features were tested in Figs. 13 and 14.

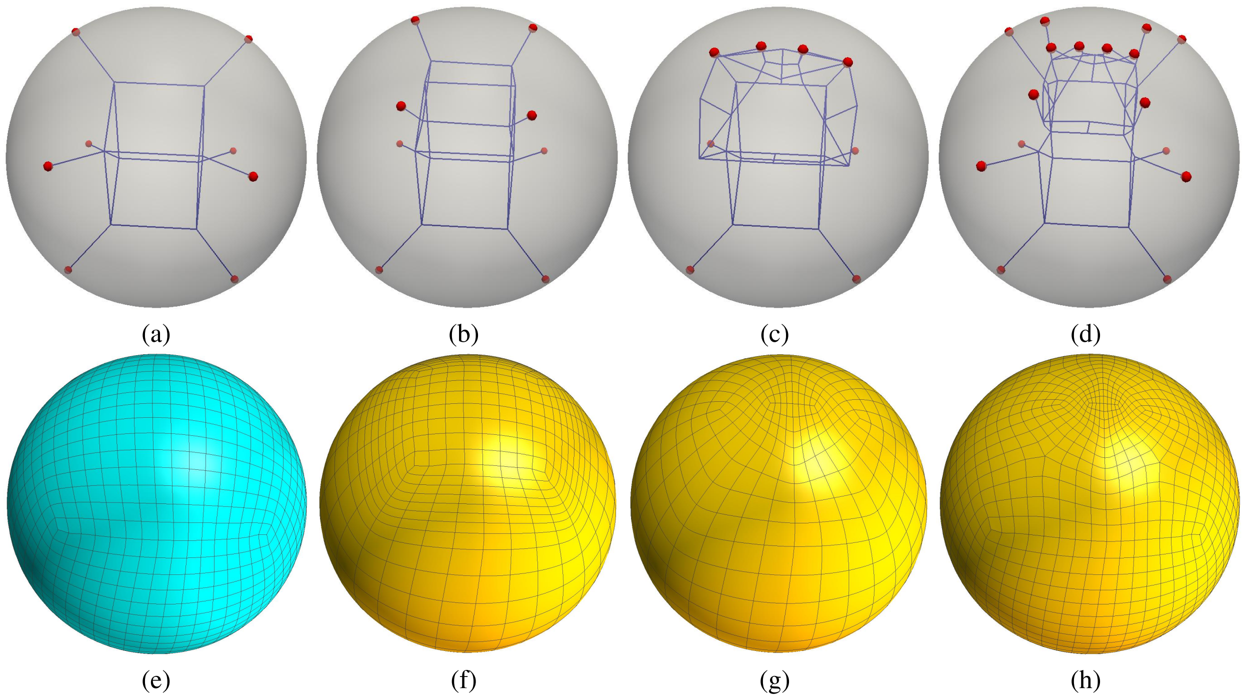


Figure 12: New singularity insertion of a sphere model, where (a-d) show the singular graph of the T-mesh, and (e-h) show the T-spline with Bézier element representation. The original sphere model is in (a), while Template 1 is applied to (b), Template 2 is applied to (c), and Template 3 is applied to (d).

Model	T-mesh nodes	Irregular nodes (surface, interior)	Bézier elements	Scaled Jacobian (min, ave, max)	Hausdorff Distance	Time (s)	Preserved feature
Bunny	12,503	(14, 95)	16,622	(0.20, 0.68, 1.00)	2.57e-2	73.8	–
Bunny with feature	12,758	(14, 96)	17,013	(0.20, 0.62, 1.00)	2.30e-2	74.5	open curve
Amphora	24,894	(8, 319)	26,686	(0.12, 0.65, 1.00)	1.23e-2	182.3	–
Amphora with feature	34,543	(12, 419)	40,145	(0.12, 0.65, 1.00)	8.7e-3	253.7	closed curve
Kitten	20,281	(28, 279)	24,502	(0.15, 0.71, 1.00)	1.49e-2	115.3	open curve, singularity
Rod	21,536	(18, 391)	11,898	(0.35, 0.79, 1.00)	6.5e-3	106.7	open curve, closed curve
Hanger	19,897	(16, 571)	6,564	(0.30, 0.72, 1.00)	5.1e-3	75.6	open curve

Table 1: Statistics of all the tested models

We also modified the singularities on the surface of the Kitten model to preserve certain singularities. Boolean operations and sharp feature preservation were tested in Figs. 15 and 16.

For the Amphora model in Fig.13, with the built polycube, we capture the feature information around the bottom neck region and the two handles. On the body of the Amphora, we have a closed feature curve preserved in the solid T-spline by adding one new branch in the interior. Compared to the result without feature preservation, the closed curve has brought a slight distortion to its neighboring elements. For the Kitten model in Fig. 14, we have preserved detailed feature information, comparing to the result in [13], which uses a harmonic field to split the domain and build polycubes. For the mouth region, we preserved the feature lines of the mouth by aligning it to parametric lines. For the left eye, we used Template 3 and for the right eye we used Template 2. The singularity information in the left eye region are constrained in the area, while in the right eye region it is propagated outward. For the Rod model in Fig. 15, we used Boolean operations during the T-mesh construction. We only used the skeleton of the torus region and the middle cylindrical region to build the polycube. With the initial subdivision result, we performed extrusion and subtracted one small cylindrical region from it. All the points on the extruded surface were projected back to the input surface to get the final T-mesh. Sharp features were also preserved. We treated these sharp features as open curves, aligned them to certain parametric lines, and duplicated them in the T-mesh [14]. For the circular closed curve at the torus region, we split it into multiple open curves because it spans multiple cubes, and then aligned each open curve to the parametric line. For the Hanger model in Fig. 16, since the components connecting the two hollow cylinders are thin, constructing interior and boundary cubes for them will lead to very thin elements. Therefore we only use the skeletons to generate the two hollow cylinders and preserve the intersection curves between the cylinders and the connecting components, as shown in Fig. 16(b). Then we found the extrusion paths for the corners of the intersection curves, and generated a group of elements to union the two cylinders together. The surfaces generated from extrusion were projected back to the input boundary. In this way, the sharp feature information of the model was preserved.

Limitation. The main drawback of our work is that we have to perform pre-processing with the skeleton to make it

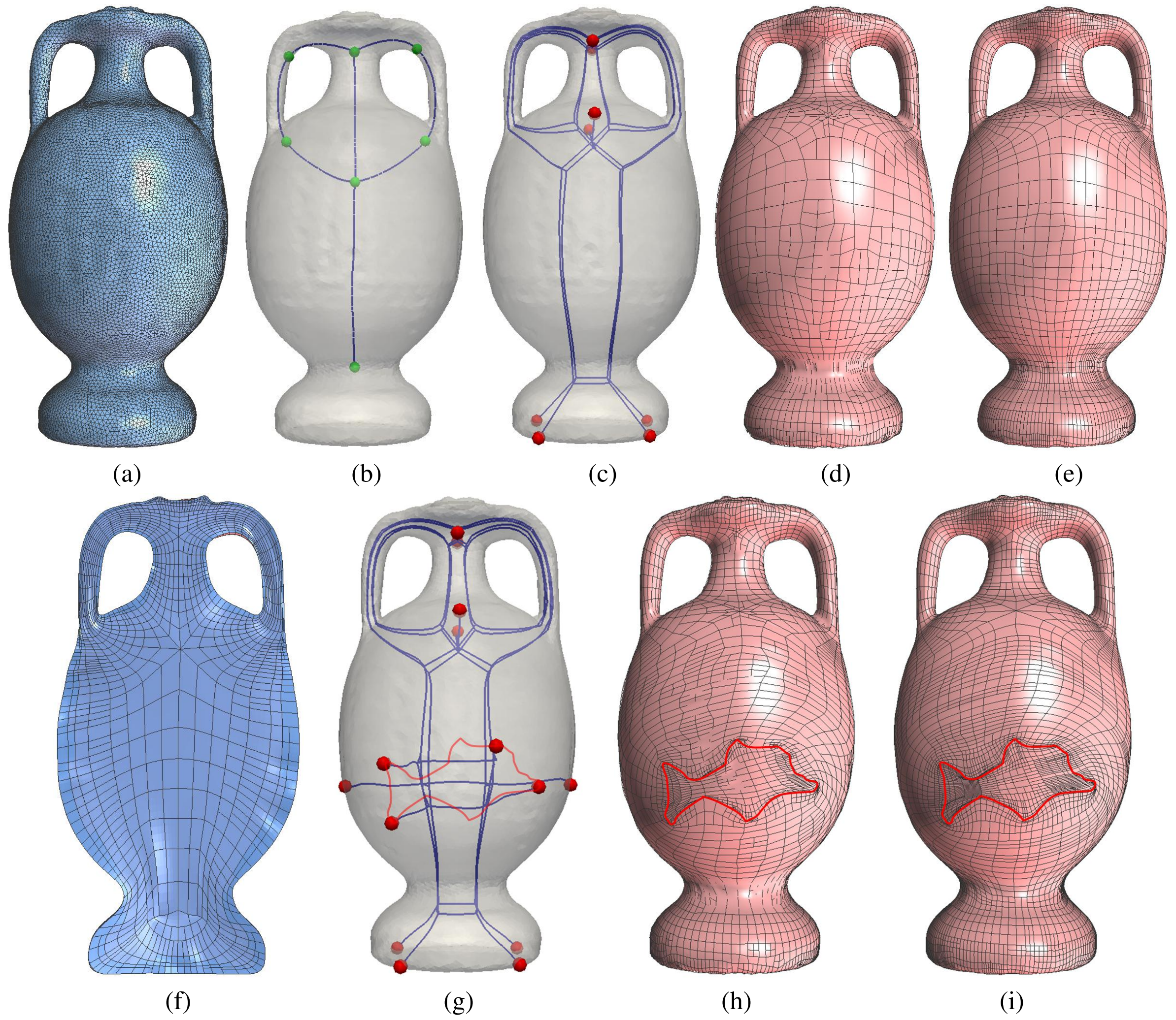


Figure 13: Amphora model. (a) The input boundary triangle mesh; (b) skeleton splitting result; (c) singular graph; (d) the constructed solid T-spline and T-mesh; (e) the extracted solid Bézier elements; and (f) some Bézier elements are removed to show the cross-section. A designed dolphin shape closed curve is preserved with (g) showing the singular graph after skeleton modification; (h) shows solid T-spline and T-mesh; and (i) shows Bézier elements.

valid for our polycube construction. In addition, we cannot perform volumetric parameterization from frame field for the whole domain due to limitations of singularity restricted field and flip-over issues as discussed in [18]. For complex CAD models, since we are enclosing all the singularities into the interior, the resulting T-mesh quality may not be good in some situations compared to volumetric polycube parameterization methods which allow boundary singular edges.

6. Conclusion and Future Work

In conclusion, we have developed a new algorithm to build feature-preserving T-meshes using skeleton-based polycube generation. Surface features are classified into open curves, closed curves and singular features. Parametric mapping, volumetric parameterization from frame field, skeleton modification and singularity modification are the corresponding methods to preserve them. The constructed solid T-spline follows the topology of the input model and detailed features are preserved. In the future, we are planning to extend the parameterization from frame field method to the whole model with designed singularities. Surface conformal solid T-spline construction from CAD models is another direction we are targeting at.

Acknowledgements

The work of Lei Liu and Yongjie Zhang was supported by the PECASE Award N00014-14-1–0234. The work of Wenping Wang was partially supported by the National Basic Research Program of China (2011CB302400), National Natural Science Foundation of China (61272019), and the Research Grant Council of Hong Kong (718209, 718010 and 718311).

Reference

[1] J. A. Cottrell, T. J. Hughes, Y. Bazilevs, Isogeometric analysis: toward integration of CAD and FEA, Wiley, 2009.

[2] T. J. Hughes, J. A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement, Computer Methods in Applied Mechanics and Engineering 194 (2005) 4135–4195.

[3] Y. Bazilevs, I. Akkerman, D. J. Benson, G. Scovazzi, M. J. Shashkov, Isogeometric analysis of Lagrangian hydrodynamics, Journal of Computational Physics 243 (2013) 224–243.

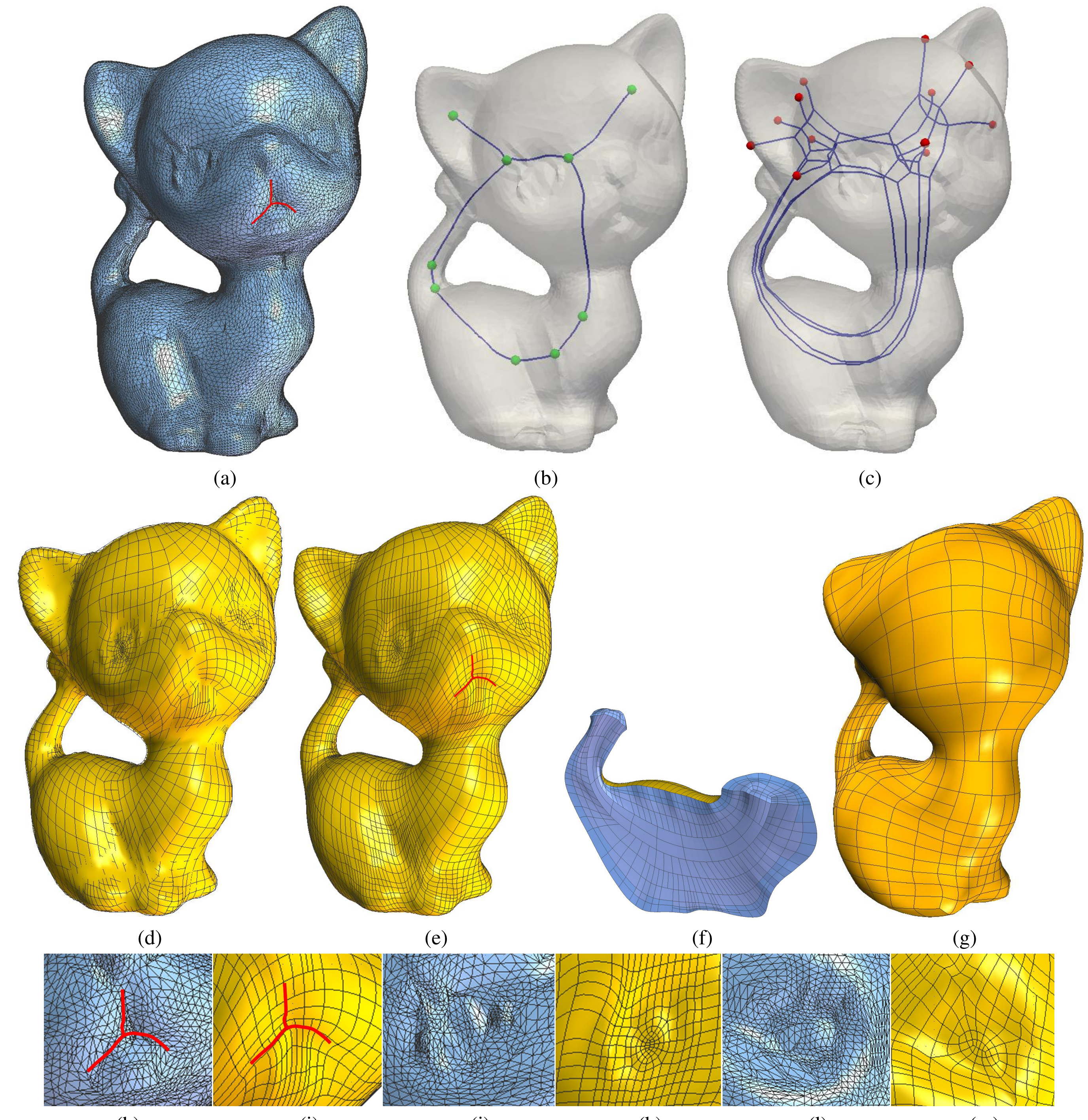


Figure 14: Kitten model. (a) The input boundary triangle mesh; (b) skeleton splitting result; (c) singular graph; (d) the constructed solid T-spline and T-mesh; (e) the extracted solid Bézier elements; (f) some Bézier elements are removed to show the cross-section; and (g) the extracted solid Bézier elements from [13]. The zoom-in pictures of the mouth, left eye and right eye are given in (h), (j) and (l); their corresponding Bézier element representations are given in (i), (k) and (m). Template 3 is applied to the left eye region, while Template 2 is applied to the right eye region.

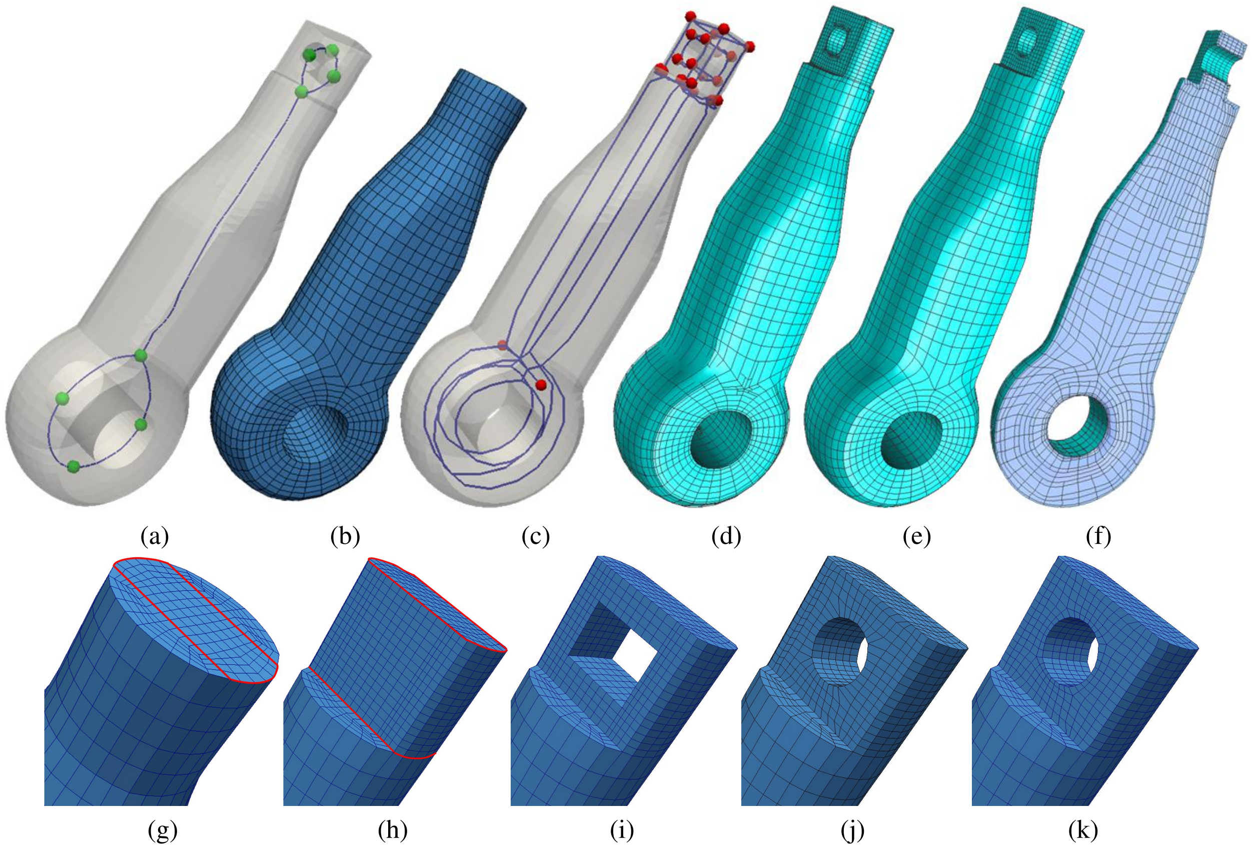


Figure 15: Rod model. (a) Skeleton splitting result; (b) initial T-mesh from subdivision; (c) singular graph of the final T-mesh; (d) final T-mesh with constructed solid T-spline; (e) the extracted solid Bézier elements; and (f) some Bézier elements are removed to show the cross-section. (g-k) show the Boolean operations to the T-mesh; (g) T-mesh from subdivision with selected extrusion region; (h) the extrusion result; (i) Boolean subtraction result by removing elements in the cylindrical hole region; (j) projecting the nodes from extrusion back to the input surface; and (k) pillowing the cylindrical hole to improve the element quality.

[4] Y. Zhang, Y. Bazilevs, S. Goswami, C. L. Bajaj, T. J. Hughes, Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow, Computer Methods in Applied Mechanics and Engineering 196 (29-30) (2007) 2943–2959.

[5] 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.

[6] T. W. Sederberg, J. Zheng, A. Bakenov, A. Nasri, T-splines and T-NURCCs, ACM Transactions on Graphics 22 (3) (2003) 477–484.

[7] Y. Bazilevs, V. M. Calo, J. A. Cottrell, J. A. Evans, T. J. Hughes, S. Lipton, M. A. Scott, T. W. Sederberg, Isogeometric analysis using T-splines, Computer Methods in Applied Mechanics and Engineering 199 (5-8) (2010) 229–263.

[8] D. Schillinger, L. Dedè, M. A. Scott, J. A. Evans, M. J. Borden, E. Rank, T. J. Hughes, An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces, Computer Methods in Applied Mechanics and Engineering 249-252 (2012) 116–150.

[9] 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.

[10] J. M. Escobar, J. M. Cascón, E. Rodríguez, R. Montenegro, A new approach to solid modeling with trivariate T-splines based on mesh optimization, Computer Methods in Applied Mechanics and Engineering 200 (45-46) (2011) 3210–3222.

[11] W. Li, N. Ray, B. Lévy, Automatic and interactive mesh to T-spline conversion, in: Eurographics Symposium on Geometry Processing, 2006, pp. 191–200.

[12] Y. Zhang, W. Wang, T. J. Hughes, Solid T-spline construction from boundary representations for genus-zero geometry, Computer Methods in Applied Mechanics and Engineering (249-252) (2012) 185–197.

[13] W. Wang, Y. Zhang, L. Liu, T. J. Hughes, Trivariate solid T-spline construction from boundary triangulations with arbitrary genus topology, Computer Aided Design 45 (2013) 351–360.

[14] L. Liu, Y. Zhang, T. J. Hughes, M. A. Scott, T. W. Sederberg, Volumetric T-spline construction using Boolean operations, Engineering with Computers, (2014), DOI = 10.1007/s00366-013-0346-6.

[15] H. Wang, Y. He, X. Li, X. Gu, H. Qin, Polycube splines, in: Symposium on Solid and Physical Modeling, 2007, pp. 241–251.

[16] B. Li, X. Li, K. Wang, H. Qin, Surface mesh to volumetric spline conversion with generalized polycubes, IEEE Transactions on Visualization and Computer Graphics 99 (2013) 1539–1551.

[17] M. Nieser, U. Reitebuch, K. Polthier, Cubecover parameterization of 3D volumes, Computer Graphics Forum 30 (5) (2011) 1397–1406.

[18] Y. Li, Y. Liu, W. Xu, W. Wang, B. Guo, All-hex meshing using singularity-restricted field, ACM Trans. Graph. 31 (6) (2012) 177:1–177:11.

[19] J. Huang, Y. Tong, H. Wei, H. Bao, Boundary aligned smooth 3D cross-frame field, ACM Trans. Graph. 30 (6) (2011) 143:1–143:8.

[20] T. Jiang, J. Huang, Y. Wang, Y. Tong, H. Bao, Frame field singularity correction for automatic hexahedralization, IEEE Transactions on Visualization and Computer Graphics, (2014), DOI = 10.1109/TVCG.2013.250.

[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.-H. Kwok, C. C. Wang, Domain construction for volumetric cross-parameterization, Computers & Graphics 38 (0) (2014) 86 – 96.

[23] J. Gregson, A. Sheffer, E. Zhang, All-hex mesh generation via volumetric polycube deformation, Computer Graphics Forum 30 (5) (2011) 1407–1416.

[24] M. Livesu, N. Vining, A. Sheffer, J. Gregson, R. Scateni, Polycut: Monotone graph-cuts for polycube base-complex construction, ACM Trans. Graph. 32 (6) (2013) 171:1–171:12.

[25] W. Yu, K. Zhang, S. Wan, X. Li, Optimizing polycube domain construction for hexahedral remeshing, Computer-Aided Design 46 (0) (2014) 58 – 68.

[26] J. Huang, T. Jiang, Z. Shi, Y. Tong, H. Bao, M. Desbrun, L_1-based construction of polycube maps from complex shapes, ACM Transactions on Graphics, (2014), DOI = 10.1145/2601097.2601352.

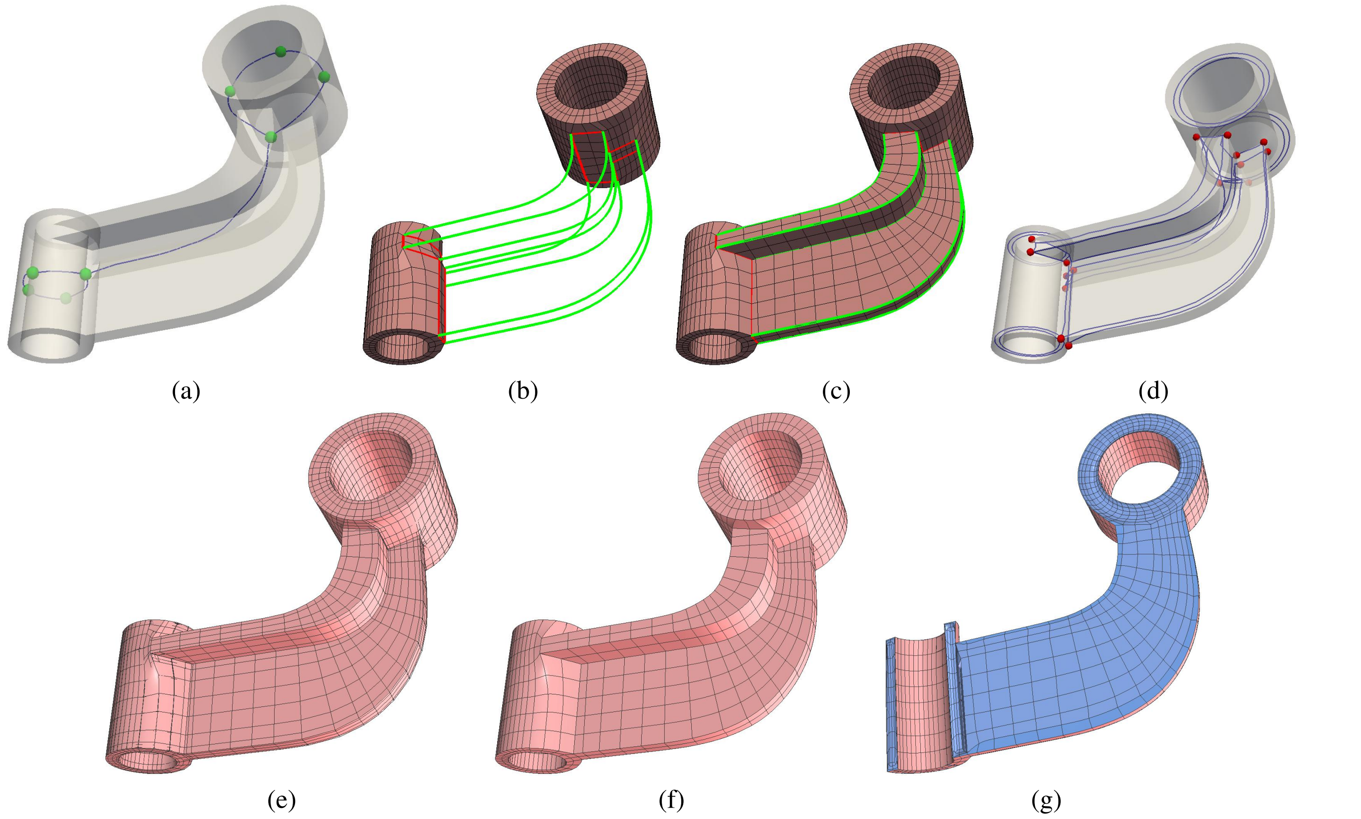


Figure 16: Hanger Model. (a) Skeleton splitting result, only the two ring regions are used; (b) the initial T-mesh with open curves (red) preserved, and the extrusion paths (green); (c) using extrusion and union to get the final T-mesh; (d) singular graph of the final T-mesh; (e) the final T-mesh with constructed solid T-spline; (f) the extracted solid Bézier elements; and (g) some Bézier elements are removed to show the cross-section.

[27] Y. Zhang, C. L. Bajaj, Adaptive and quality quadrilateral/hexahedral meshing from volumetric data, Computer Methods in Applied Mechanics and Engineering 195 (9-12) (2006) 942–960.

[28] Y. Zhang, X. Liang, G. Xu, A robust 2-refinement algorithm in octree and rhombic dodecahedral tree based all-hexahedral mesh generation, Computer Methods in Applied Mechanics and Engineering 256 (2013) 562–576.

[29] L. Maréchal, Advances in Octree-based all-hexahedral mesh generation: Handling sharp features, in: Proceedings of the 18th International Meshing Roundtable, 2009, pp. 65–84.

[30] J. Qian, Y. Zhang, Automatic unstructured all-hexahedral mesh generation from B-Reps for non-manifold CAD assemblies, Engineering with Computers 28 (4) (2012) 345–359.

[31] A. Tagliasacchi, I. Alhashim, M. Olson, H. Zhang, Mean curvature skeletons, in: Computer Graphics Forum, Vol. 31, 2012, pp. 1735–1744.

[32] W. Wang, Y. Zhang, G. Xu, T. J. Hughes, Converting an unstructured quadrilateral/hexahedral mesh to a rational T-spline, Computational Mechanics 50 (1) (2012) 65–84.

[33] W. Wang, Y. Zhang, M. A. Scott, T. J. Hughes, Converting an unstructured quadrilateral mesh to a standard T-spline surface, Computational Mechanics 48 (2011) 477–498.

[34] R. Goldman, T. Lyche, Knot insertion and deletion algorithms for B-spline curves and surfaces, Society for Industrial and Applied Mathematics–Philadelphia, 1993.

[35] O. K.-C. Au, C.-L. Tai, H.-K. Chu, D. Cohen-Or, T.-Y. Lee, Skeleton extraction by mesh contraction, ACM Trans. Graph. 27 (3) (2008) 44:1–44:10.

[36] T. Martin, G. Chen, S. Musuvathy, E. Cohen, C. Hansen, Generalized swept mid-structure for polygonal models, Computer Graphics Forum 31 (805-814) (2012) 805–814.

[37] E. Cohen, R. Riesenfeld, G. Elber, Geometric Modeling with Splines: An Introduction, A. K. Peters, Natick, MA, 2001.

[38] Y. Zhang, X. Liang, J. Ma, Y. Jing, M. J. Gonzales, C. Villongco, A. Krishnamurthy, L. R. Frank, V. Nigam, P. Stark, et al., An atlas-based geometry pipeline for cardiac hermite model construction and diffusion tensor reorientation, Medical image analysis 16 (6) (2012) 1130–1141.

[39] X. Li, Y. Bao, X. Guo, M. Jin, X. Gu, H. Qin, Globally optimal surface mapping for surfaces with arbitrary topology, Visualization and Computer Graphics, IEEE Transactions on 14 (4) (2008) 805–819.

[40] S. Wan, Z. Yin, K. Zhang, H. Zhang, X. Li, A topology-preserving optimization algorithm for polycube mapping, Computers & Graphics 35 (3) (2011) 639 – 649.

[41] M. A. Price, C. G. Armstrong, Hexahedral mesh generation by medial surface subdivision: Part II. solids with flat and concave edges, International Journal for Numerical Methods in Engineering 40 (1) (1997) 111–136.

[42] N. D. Cornea, D. Silver, P. Min, Curve-skeleton properties, applications, and algorithms, IEEE Transactions on Visualization and Computer Graphics 13 (3) (2007) 530–548.

[43] http://www.tetgen.org.

[44] M. A. Price, C. G. Armstrong, M. A. Sabin, Hexahedral mesh generation by medial surface subdivision: Part I. solids with convex edges, International Journal for Numerical Methods in Engineering 38 (19) (1995) 3335–3359.