ICES REPORT 13-19

July 2013

Volumetric T-spline Construction Using Boolean Operations

by

L. Liu, Y. Zhang, T.J.R. Hughes, M.A. Scott, T.W. Sederberg

The Institute for Computational Engineering and Sciences
The University of Texas at Austin
Austin, Texas 78712

Reference: L. Liu, Y. Zhang, T.J.R. Hughes, M.A. Scott, T.W. Sederberg, Volumetric T-spline Construction Using Boolean Operations, ICES REPORT 13-19, The Institute for Computational Engineering and Sciences, The University of Texas at Austin, July 2013.

Volumetric T-spline Construction Using Boolean Operations

Lei Liu[1], Yongjie Zhang[1,*], Thomas J.R. Hughes[2], Michael A. Scott[3] and Thomas W. Sederberg[4]

[1] Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA
[2] Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX 78712, USA
[3] Department of Civil and Environmental Engineering, Brigham Young University, Provo, UT 84602, USA
[4] Department of Computer Science, Brigham Young University, Provo, UT 84602, USA

Summary. In this paper, we present a novel algorithm for constructing a volumetric T-spline from B-reps inspired by Constructive Solid Geometry (CSG) Boolean operations. By solving a harmonic field with proper boundary conditions, the input surface is automatically decomposed into regions that are classified into two groups represented, topologically, by either a cube or a torus. We perform two Boolean operations (union and difference) with the primitives and convert them into polycubes through parametric mapping. With these polycubes, octree subdivision is carried out to obtain a volumetric T-mesh, and sharp features detected from the input model are also preserved. An optimization is then performed to improve the quality of the volumetric T-spline. Finally we extract trivariate Bézier elements from the volumetric T-spline, and use them directly in isogeometric analysis.

Key words: volumetric T-spline, Boolean operations, polycubes, parametric mapping, sharp feature, isogeometric analysis

1 Introduction

Isogeometric analysis [7, 13] bridges Computer Aided Design (CAD) and Finite Element Analysis (FEA) by using the same basis functions for geometric modeling and numerical simulation. For many important application areas, it has been demonstrated that isogeometric analysis, using smooth basis functions, is more accurate and robust than traditional FEA which uses C^0 basis functions [8, 5, 32]. Additionally, the exact CAD geometry is embedded in

*Corresponding author. Tel: (412) 268-5332; Fax: (412) 268-3348; Email: *jessicaz@andrew.cmu.edu* (Yongjie Zhang).

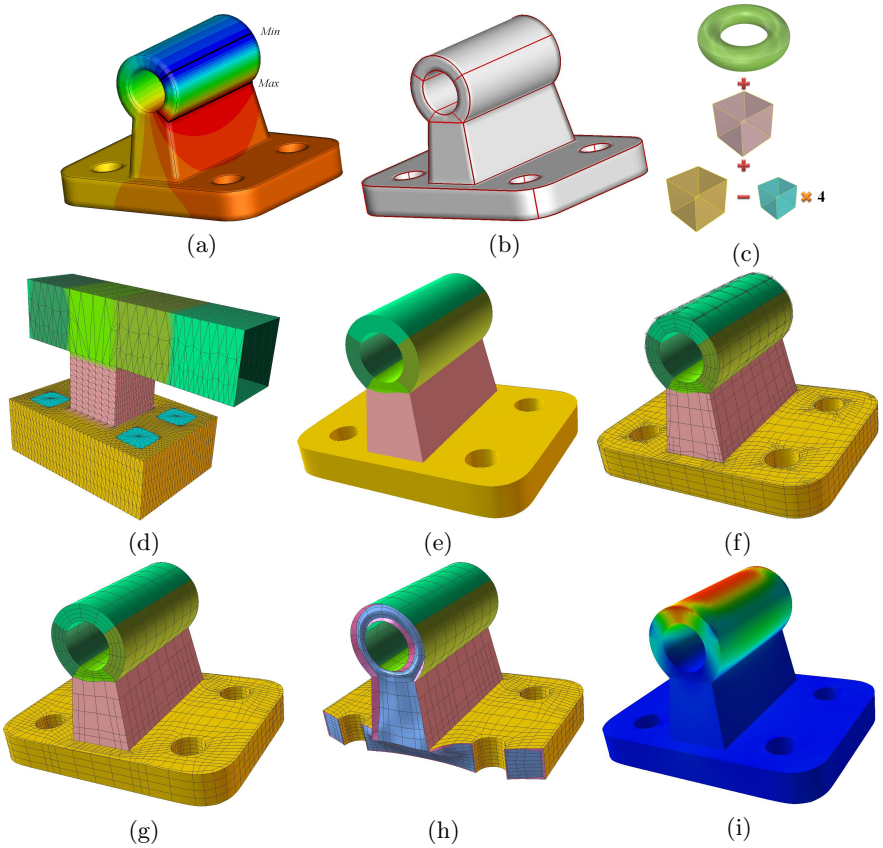


Fig. 1. CAD Assembly. (a) One temperature field to split the top torus region; (b) splitting result; (c) Boolean operations; (d) parametric mapping result (the torus primitive is used for the top component, and the difference operation is used to create four holes in the bottom base component); (e) solid T-spline; (f) solid T-spline with T-mesh; (g) solid T-spline with Bézier representation; (h) some elements are removed to show the interior of (g); and (i) isogeometric analysis result.

the analysis at the coarsest level of discretization. In many cases, a trivariate (solid) description of an object is required for analysis. Unfortunately, current CAD representations of solid objects are composed of a collection of surfaces, see Fig. 1. To employ the isogeometric paradigm for solids, a trivariate parameterization of the interior of the solid must be generated. This is an important and challenging problem in isogeometric analysis [20].

Several papers have studied isogeometric analysis using solid NURBS (Non-uniform Rational B-spline) construction [13, 32, 4, 30]. However, NURBS have some drawbacks that limit their use for isogeometric analysis. For example, NURBS [18] does not support local refinement. In addition, gaps often happen between two neighboring NURBS surface patches. To overcome

these limitations, Sederberg invented T-splines [23], which support local refinement naturally by introducing T-junctions [22]. T-splines were introduced into isogeometric analysis in [6, 4]. The initial research on T-splines-based isogeometric analysis was limited to surface models. Reference [21] introduced a data structure for isogeometric analysis using T-splines. Conversion of unstructured meshes to T-splines has also been studied [28, 29]. A generalized algorithm was also developed to extract Bézier elements from volumetric T-splines, connecting the spline modeling with analysis data structure.

As for the construction of volumetric T-spline, different approaches have been developed. A method based on Periodic Global Parameterization was proposed to convert triangular meshes to T-splines [16]. Other research focuses on parametric mapping of an input tetrahedral meshes to construct solid T-splines [9]. A harmonic mapping method has been proposed for developing a 3D solid sphere from a 2-manifold for use in computer graphics and medical imaging [11]. In [25], a parametric mapping between a polycube and a surface geometry was presented to construct trivariate T-splines from input triangular meshes. Mapping, subdivision and pillowing techniques have been used to generate good quality T-splines for genus-zero [33] and arbitrary genus objects [27]. Li *et al.* [15] proposed a generalized polycube method using T shape templates to handle high-genus models and extraordinary nodes.

Despite these advances, it remains a challenging problem to automatically create a volumetric T-spline for models with arbitrary complicated geometry and topology. How to automatically and robustly split complex geometry into different components and transfer the input geometric information to the desired volumetric models are still open problems.

Inspired by CSG Boolean operations [1, 3, 24], in this paper we present a novel algorithm to construct trivariate solid T-spline models using two Boolean operations: union and difference. In our algorithm, we compute a harmonic field together with the boundary information to split the domain, and use primitives (cube and torus) and Boolean operations to generate polycubes[1]. Parametric mapping is then employed to transfer the input information to the volumetric T-spline. The four main contributions that this paper makes to the problem of volumetric T-splines parametrization are: (1) a harmonic field with proper boundary conditions is computed to automatically split the input geometry into different hexahedral components; (2) two Boolean operations (especially the difference operation) are developed to construct polycubes conveniently and flexibly; (3) a novel torus primitive is introduced to deal with torus-like objects or holes, yielding few number of extraordinary nodes and high quality elements; and (4) sharp features are preserved and mesh quality is improved.

[1]Conventionally, a polycube is comprised of cubes of equal sizes with two neighboring cubes sharing a complete face [2]. In this paper, the "cubes" can be arbitrary hexahedra of different sizes.

Although the proposed algorithm is automatic and robust for a large class of complex models, it also has limitations. For example, it cannot handle some special objects such as a tetrahedron or a cone, and it cannot preserve the input surface parameterization.

The remainder of this paper is organized as follows. The main steps of the algorithm (illustrated in Fig. 1) are overviewed in Section 2. Section 3 discusses extracting boundary information. Section 4 talks about different primitives and Boolean operations among them. Section 5 explains T-spline construction. Section 6 shows some results, and Section 7 draws conclusions and points out the future work.

2 Algorithm Overview

Polycube-based methods for volume parametrization [27, 26, 14] perform domain decomposition by splitting the model into hexahedral regions that map to cubes. However, sometimes the models are so complicated that it is difficult to split the domain automatically. Inspired by CSG Boolean operations [1], here we propose to use Boolean operations to build the polycubes. As shown in Fig. 2, there are three main stages to construct a trivariate solid T-spline from the given CAD model: curve extraction, domain decomposition and Boolean operations, and volumetric T-spline construction.

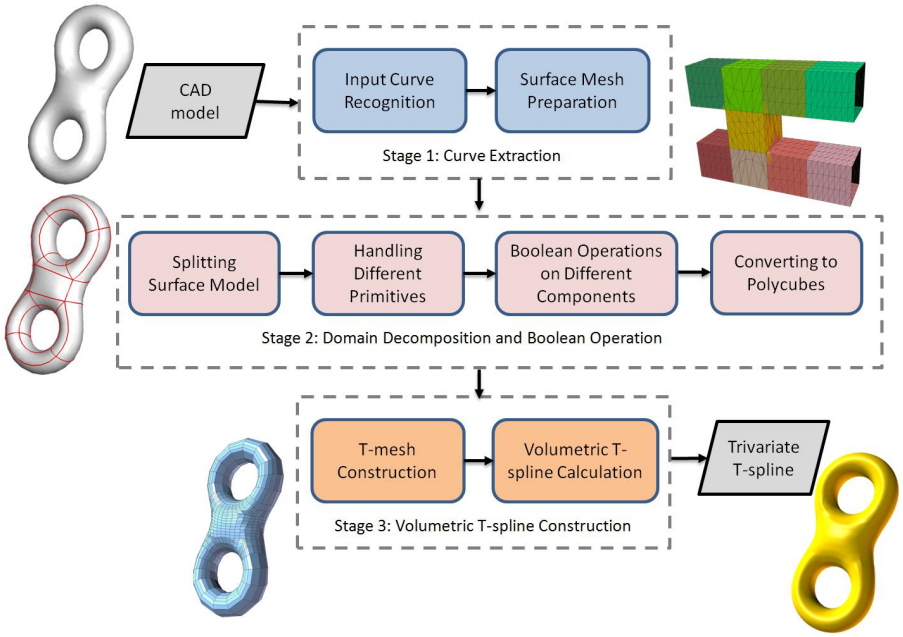


Fig. 2. Three stages of volumetric T-spline construction using Boolean operations.

The first stage initializes all the necessary boundary information for the following stages. We first classify the curve information from the CAD model into two groups, and then use the commercial software ABAQUS to generate the surface mesh.

Based on the curve information and surface mesh, we perform domain decomposition and Boolean operations to generate polycubes. A harmonic field with proper boundary conditions is computed to automatically split the surface model into different components, topologically equivalent to either a cube or a torus. As shown in Fig. 1, each torus is composed topologically of four cubes. All cubes generated by the domain decomposition are then union together and holes (represented topologically as cubes) are subtracted (see Fig. 1(d)). We will refer to the resulting configuration as a polycube, realizing that we take some liberties in using the term in this way. The CAD surface is then mapped to the polycube surface.

The volumetric T-spline is obtained by performing an octree subdivision on the polycube. Here we use a separate octree for each cube and force two neighboring cubes to have the same parameterization at the shared boundary. All the detected sharp feature information is preserved in this step. Pillowing, smoothing and optimization are then used to improve the quality of the T-mesh. To obtain a gap-free T-mesh, we apply templates [28, 29] to each irregular node in the T-mesh. Finally, volumetric T-spline is generated and Bézier elements are extracted for isogeometric analysis.

3 Curve Extraction

Most CAD models contain sharp edges or features. It is best if these features map to edges of the polycube (although we do not require that each edge of the polycube maps to a feature in the CAD model). We need to identify which of these curves are best represented as polycube edges during the Boolean difference stage of the algorithm. We call such edges *feature curves* and the remaining edges we call *difference curves*. For example in Fig. 3, the model is the subtraction of a cylinder from a cube. The blue lines are the feature curves of the model, and the red lines are the difference curves.

Curve Classification: We classify the input boundary information into three groups: corners, curves, and patches. All the surface models are formed by these three groups. *Curves* are the parametric boundary lines on the surface. In Fig 3, there are 14 curves: $C_1 \sim C_{14}$ (blue and red lines), which are the edges of the cube and the cylinder. *Corners* are the intersection points of the curves, which are also the corners of the cubes (the eight blue dots $V_1 \sim V_8$). Several curves connecting consecutively form the boundary of a surface *patch*. In Fig. 3, there are 7 patches: six cube faces and one circumferential surface of the cylinder (the gray and red surfaces, $S_1 \sim S_7$). In this model, curves $C_1 \sim C_{12}$ are feature curves. Curves $C_{13} \sim C_{14}$ are difference curves. These curves contain the input sharp feature information and will be used to split one model into different components.

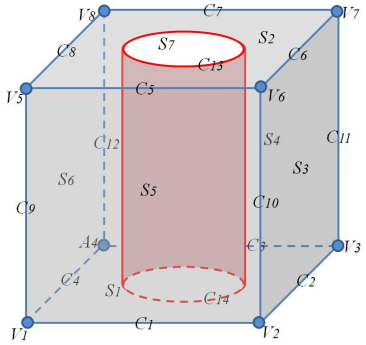


Fig. 3. Classification of curve information. Blue line: feature curves; and red lines: difference curves.

Sharp Feature Detection: There are two types of sharp features in the designed models: sharp curves and sharp corners. *Sharp curves* are those curves across which the surface continuity is C^0, and the *sharp corners* are the intersection points of the sharp curves. For example in Fig. 3, all the 12 edges of the cube ($C_1 \sim C_{12}$) and the top and bottom outlines of the cylinder ($C_{13} \sim C_{14}$) are sharp curves, and the 8 corners of the cube ($V_1 \sim V_8$) are sharp corners.

4 Domain Decomposition and Boolean Operations

To perform Boolean operations, we first split the model into different hexahedral components, and then use primitives to represent them.

4.1 Domain Decomposition

For simple CAD models, we can directly use the feature curves to generate the polycube edges, and use the difference curves to define virtual components. Here a *virtual component* is a component which does not exist in the real model, but it can be deducted from the design process and boundary information. These virtual components are the result of CSG difference operation in design. For example in Fig. 3, the feature and difference curves can split the model into one cube and one virtual cylinder.

Harmonic fields have been used successfully to split a complex geometry into coherent regions [33, 27]. Temperature distribution is an example of a harmonic field. The idea is to assign high and low temperature values to two different points on the model, and the harmonic field computed with those two boundary conditions will express the steady-state temperature distribution across the model. For example in Fig. 4, we use the following five steps to split the torus model into four hexahedral components:

1. First we find out the points with largest/smallest Z-coordinates, and assign them the max and min temperature respectively;
2. A harmonic field is calculated on the surface mesh, see Fig. 4(a);

3. We find out the critical points in the field, which are Min, Max, and two saddle points (C_1, C_2). They form two cross sections;
4. We assign min temperature to one cross section, and max temperature to the other one. The harmonic field is recalculated using the new boundary conditions and the temperature distribution is shown in Fig. 4(b); and
5. Four equally-spaced points are selected on each cross section curve (black curves) in Fig. 4(b), which will be set as the cube corners. Then we trace the gradient lines and finally split them into four parts to obtain all the red curves in Fig. 4(c).

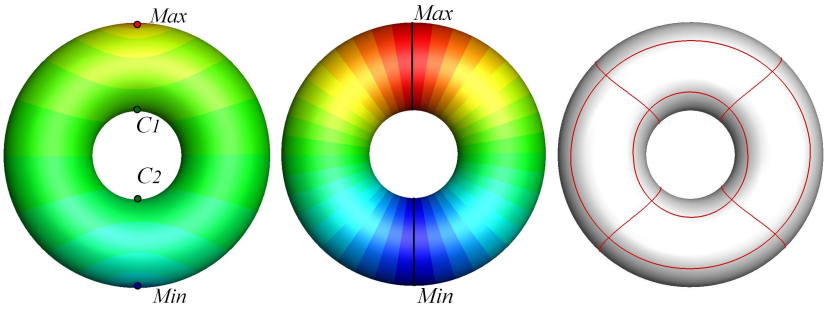
Discussion: By using a harmonic field with proper boundary conditions, we can in many cases automatically split a complex geometry into multiple hexahedral components. Finding a proper boundary condition often requires user interactions. Sometimes we may need to compute the harmonic field several times before we can obtain an optimal domain decomposition result.

4.2 Two Primitives

Primitives are basic objects in design and geometrical modeling. Typical primitives in CSG include cuboids, cylinders, prisms, pyramids, spheres and cones. In our algorithm, we only use two primitives: the *cube* and the *torus*. Furthermore, unlike conventional CSG, our primitives are used in a topological sense, so, for example, the edges of our cubes need not have equal length. Fig. 5 shows how to map these two primitives from the physical space to the parametric space. It is easy to map one of our cubes to a unit cube. For a torus, we use four consecutive unit cubes to represent it, with the left face of the first cube connecting to the right face of the last cube.

4.3 Two Boolean Operations

There are two basic Boolean operations in our polycube generation: **union** and **difference**. We develop templates to handle the Boolean operations among



(a) (b) (c)

Fig. 4. Splitting one torus model into four cubes. (a) Set the top and bottom points with max and min temperature respectively, calculate the harmonic temperature field, and find out critical points (extreme and saddle points); (b) recalculate the harmonic field by setting the whole cross section to be max/min temperature; and (c) split the model with isoparametric and gradient lines.

Fig. 5. Two primitives from the physical space to the parametric space. (a) Cube; and (b) torus.

the primitives: union of two cubes, difference of two cubes, union of a cube and a torus, difference of a cube and a torus. Since two cubes may have different sizes and relative position, we have multiple cases for the union and difference operations between them, see Fig. 6(a-b). As for the operations between a cube and a torus, we will select one representative cube out of the four cubes of the torus (the red cube in Fig. 6(c, d)), and then use it to perform all the Boolean operations with other cubes. Of the two Boolean operations, difference is a special one in our polycube generation. Based on difference curves, we build virtual components. As shown in Fig. 7, after finding out the boundary of the cylinder in the input model, we fill the holes on the surface mesh by adding new triangles. Then a virtual cylinder is reconstructed and we carry out all the following work using the new mesh. After building T-meshes, elements inside the filled holes will be deleted by using the difference operation.

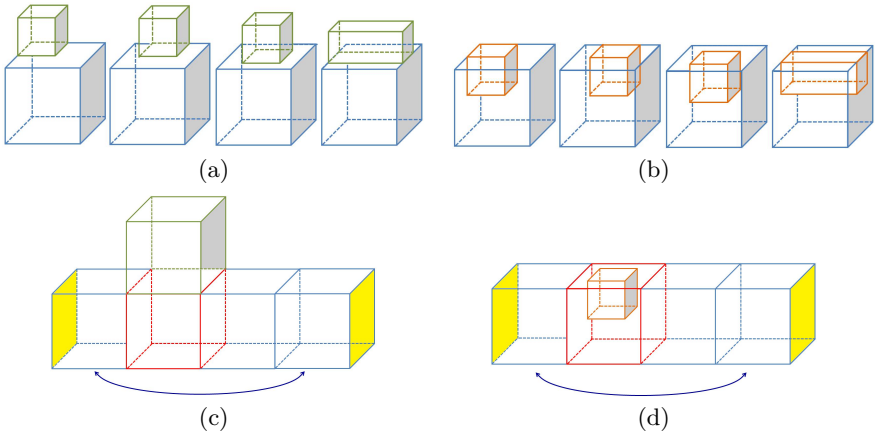


Fig. 6. Boolean operations of cubes and torus with different sizes and relative position. (a) Four cases for the union operation of two cubes; (b) four cases for the difference operation of two cubes; (c) the union operation of a cube and a torus; and (d) the difference operation of a cube and a torus.

Discussion: The torus primitive and the difference operation are two new features in our polycube generation, which provide more convenience

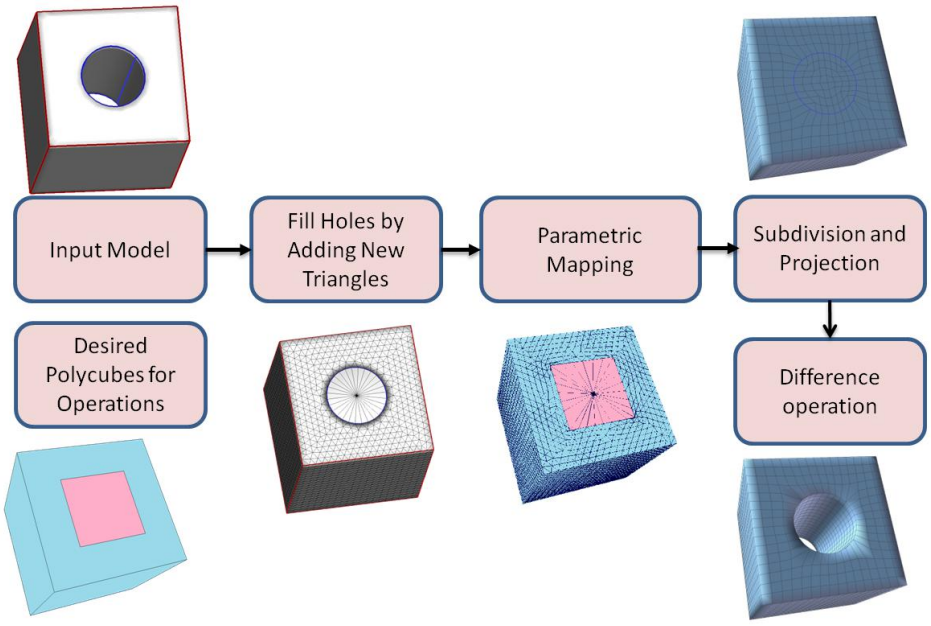


Fig. 7. Steps to perform the difference operation. Holes are filled to create a virtual component (virtual cylinder).

and flexibility in handling designed CAD models. The resulting T-splines will have better surface continuity and high quality elements.

There is a special situation we should discuss here. Let us take a cube and subtract a cylinder from it (Fig. 3). Topologically, it can be represented either as cube-minus-cylinder using the difference operation, or as a torus. Our algorithm can represent the object in either way. If the inner and outer boundaries of the object have no sharp corner, then we consider it more like a torus or hollow cylinder and choose the torus primitive. Otherwise, if sharp corner happens in the inner and/or outer boundaries, we choose to use the difference operation to handle it.

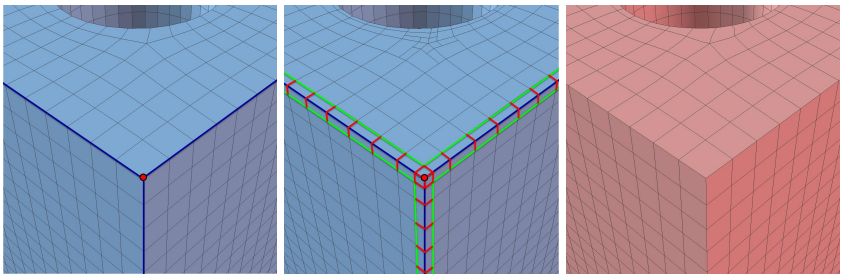
5 Volumetric T-spline Construction

To construct volumetric T-splines, we first need to generate the T-spline control mesh, or T-mesh. There are five main steps in this stage: adaptive octree subdivision and mapping, sharp feature preservation, pillowing and quality improvement, handling irregular nodes, trivariate T-spline construction and Bézier extraction.

5.1 Adaptive Octree Subdivision and Mapping

An initial T-mesh is generated by applying an adaptive octree subdivision to the polycubes and mapping to the boundary. For each cube, we create

one hexahedral root element, and then we subdivide one element into eight smaller ones recursively to obtain the T-mesh after mapping. For each boundary element, we check the local distance from the T-mesh boundary to the input boundary, and subdivide the element if the distance is greater than a given threshold ε. Each obtained T-mesh node has both parametric and physical coordinates. The parametric coordinates represent its position in the polycubes. For each boundary node, the physical coordinates are its corresponding position on the boundary. The physical coordinates of each interior node are calculated by a linear interpolation. T-junctions are introduced if two neighboring elements have different subdivision levels.



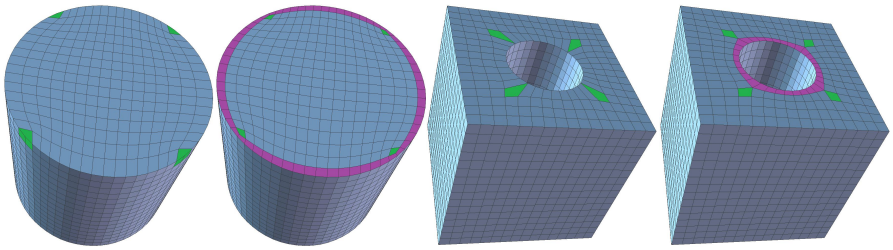
(a) (b) (c)

Fig. 8. Preserving sharp corner and sharp curve. (a) Sharp corner (red corner) and sharp curves (blue curves) before preservation; (b) preserving sharp features by duplicating sharp curves (green curves) and inserting zero-length edges (red edges); and (c) Bézier element representation of the model.

5.2 Sharp Feature Preservation and Quality Improvement

Sharp Feature Preservation: To preserve the detected sharp features, we duplicate their corresponding parametric lines in the polycubes [28]. It aims to decrease the local boundary surface continuity across the sharp curves to C^0 by repeating knots. As shown in Fig. 8, a sharp curve (blue curve) is shared by two neighboring surface patches. We duplicate the sharp curve on each patch (green curves), and connect corresponding points using edges with zero parametric length (red short edges). Then the spline surface is C^0-continuous across the sharp curves. In Fig. 8(a-b), a sharp corner is shared by three sharp curves. By duplicating each sharp curve on its neighboring surface patches, the sharp corner is also preserved.

Quality Improvement: To improve the initial T-mesh quality, we adopt pillowing, smoothing and optimization techniques. Pillowing is a sheet insertion technique that inserts one layer around the boundary [19, 31], which guarantees each element has at most one face lying on the boundary and also improve the surface continuity across the polycube edges from C^0 to C^2. The sharp feature information on the input surface can also be transferred to the new surface. When the corner of one cube lies on a smooth sharp curve, the parametric mapping method may generate poor quality elements around that



(a) (b) (c) (d)

Fig. 9. Pillowing along the circumferential direction of a cylinder. (a-b) A solid cylinder before (a) and after (b) pillowing; (c-d) a cube with a cylindrical hole before (c) and after (d) pillowing.

corner. Fig 9(a) shows the T-mesh of a solid cylinder model. On its top face the four cube corners of the polycube have bad quality (green elements). To deal with this situation, we insert one new layer around the circumferential direction, see the magenta layer in Fig 9(b). After smoothing, the mesh quality is improved significantly. This method can also be applied to the surface of virtual cylinders, see Fig. 9(c-d).

After pillowing, smoothing and optimization [33] are used to improve the T-mesh quality. There are four types of nodes in the T-mesh: sharp corners, sharp curve nodes, surface nodes and interior nodes. In smoothing, they are relocated in different ways. Sharp corners are fixed; sharp curve nodes move along the curve direction; surface nodes move on the surface; and interior nodes move towards its mass center. In optimization, each node is moved toward an optimal position that maximizes the worst Jacobian. The Jacobian is defined based on trilinear basis functions of T-mesh elements. For a T-mesh element, the Jacobian is defined as

$$J = det(J_M) = \begin{vmatrix} \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \zeta} \\ \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \zeta} \\ \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \zeta} \end{vmatrix}, \tag{1}$$

where N_i is a trilinear shape function. The scaled Jacobian is

$$J_s = \frac{J}{\parallel J_M(\cdot,0) \parallel \parallel J_M(\cdot,1) \parallel \parallel J_M(\cdot,2) \parallel}, \tag{2}$$

where $J_M(\cdot,0)$, $J_M(\cdot,1)$ and $J_M(\cdot,2)$ represent the first, second and last column of the Jacobian matrix, J_M, respectively. To get better optimization results, we further improve our optimization method in two ways: (1) optimize the Jacobian value defined based on Bézier basis functions; and (2)

optimize the step size when moving the control nodes. Due to the enhanced robustness of high order basis functions, distorted T-meshes may still be used in isogeometric analysis [17], and the scaled Jacobian value is one quantitative standard to evaluate the quality of T-splines. The Jacobian is evaluated at the Gaussian integration points and the corner points of one element. In step size optimization, the objective function is $f(\delta) = \min(1 - J'_s(\delta))$, where J'_s is the new Jacobian value with respect to updated coordinates, and δ is the step size. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) method [12] is used here to perform the optimization and get an optimal step size.

5.3 Irregular Nodes and Volumetric T-spline Construction

Extraordinary nodes or partial extraordinary nodes [33] are two types of irregular nodes in T-spline construction, which may introduce gaps to solid T-spline. These irregular nodes will reduce the continuity in its neighborhood and increase the degrees of freedom during analysis. Different templates have been developed to handle the irregular nodes. The basic idea is to insert zero parametric length edges around the irregular nodes to make sure the extracted knot interval is correct. In referring knot vectors, knot values are repeated whenever an irregular node is met. The detailed templates and knot insertion algorithm are explained in [28, 29].

The rational solid T-spline is defined in [29]. Its basis function has the property of partition of unity by definition, which makes it suitable for analysis. With the valid T-mesh, referred local knot vectors and the definition of rational basis function, we can construct desired volumetric T-splines. Since the volumetric T-spline is defined on local knot vectors, we extract Bézier representation of solid T-spline for isogeometric analysis. Transformation matrix from T-spline basis functions to Bézier basis functions is calculated by the Oslo knot insertion algorithm [10]. With the extracted Bézier elements, we can perform isogeometric analysis on the volumetric T-spline models.

6 Results and Isogeometric Analysis

We have applied the construction algorithm to several models on a 2.93GHz Intel Xeon CPU with 16GB RAM. Table 1 provides the statistics of four models: torus (Fig. 10), eight (Figs. 11-12), rod (Fig. 13), and CAD assembly (Fig. 1). We use the scaled Jacobian with Bézier basis function to evaluate the quality of the trivariate T-splines. The number of irregular nodes on the surface and in the interior are also counted. We can see that our algorithm is fast and it produces high quality volumetric T-splines for isogeometric analysis.

Fig. 10 shows the result of our torus primitive. It has no irregular nodes on the surface, and the generated elements have high quality with the minimum Jacobian of 0.42. For the eight model in Figs. 11-12, we compute the harmonic field twice to obtain the desired domain decomposition result. Similar to Fig. 4, we first set the bottom and top points with the min and max temperature respectively, compute the harmonic field and critical points to define three cross sections. As shown in Fig. 12(a), we then set two cross sections with the

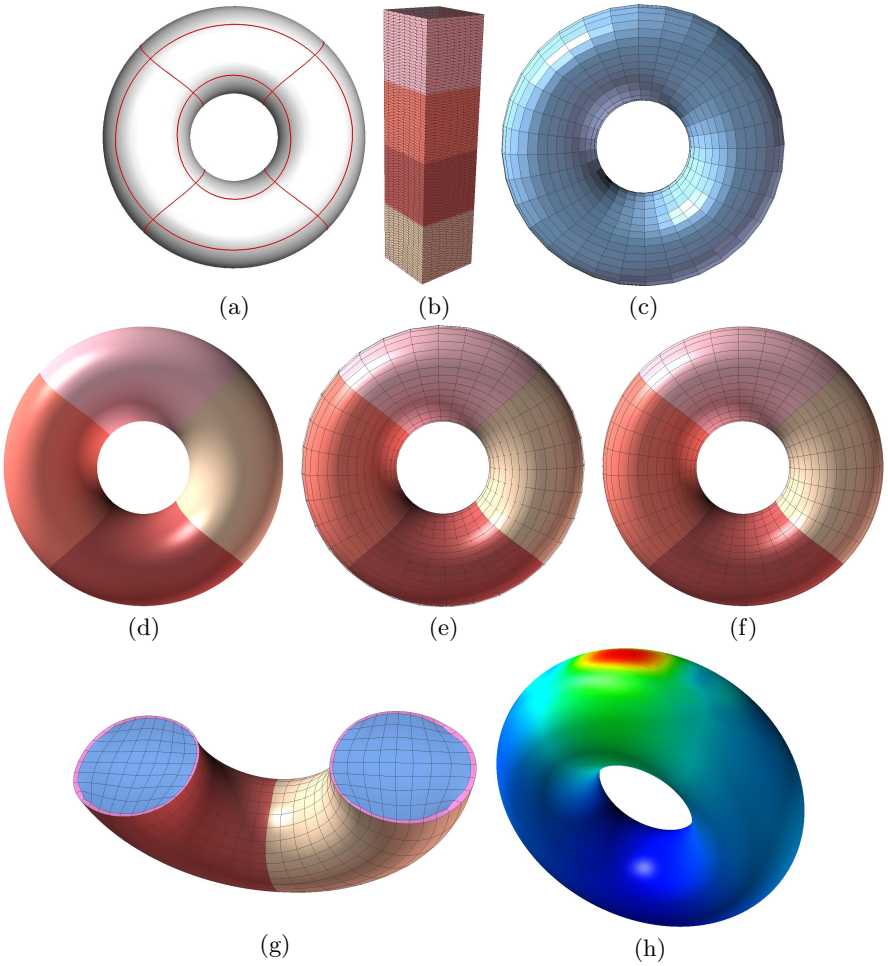


Fig. 10. Torus model. (a) Splitting result; (b) Boolean operation and parametric mapping result; (c) T-mesh; (d) solid T-spline; (e) solid T-spline with T-mesh; (f) solid T-spline with Bézier representation; (g) some elements are removed to show the interior of (f); and (h) isogeometric analysis result.

min temperature and the middle cross section with the max temperature, and obtain a new harmonic field. By tracing its isoparametric lines and gradient directions, we can split the two torus regions. Some red curves in the middle region between the two tori are obtained by finding the shortest distance on the surface. Finally we obtain the splitting result as shown in Fig. 12(b). The parametric mapping result, the Boolean operations and the constructed volumetric T-spline model are shown in Fig. 12(c-g). We also compared our result with the result from another polycube method [27]. Our method yields fewer number of irregular nodes on the surface (8 vs 16) with a better min

Jacobian (0.31 vs 0.10). In the rod model (Fig. 13) and CAD assembly model (Fig. 1), we compute the harmonic field to split the torus region. For the other regions, we trace the shortest distance among the corners to split the model. Both the torus primitive and the difference operation are used here in addition to the union of cubes, yielding good surface continuity and high quality elements. We have also developed a 3D isogeometric analysis solver for static mechanics analysis [27]. For the torus, eight and CAD assembly model, we fix the bottom and apply a displacement load on the top part. Differently for the rod model, we fix one side of the torus shape region and apply load on the other side. The analysis results are reasonable, which prove that our models are suitable for analysis.

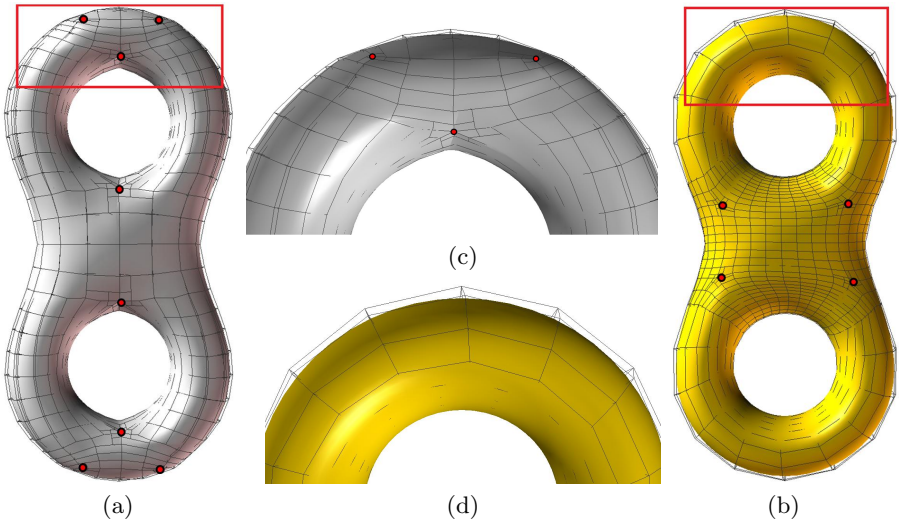


Fig. 11. Distribution of irregular nodes on the T-spline surface of eight model. (a) Polycube method in [27] with details in (c); and (b) Boolean operation method with details in (d).

Table 1. Statistics of all the tested models

Model	T-mesh nodes	Irregular nodes (surface, interior)	Bézier elements	Jacobian (worst, best)	Time (s)
Torus	5,920	(0, 128)	3,072	(0.42, 1.00)	11.8
Eight	8,323	(8, 196)	4,096	(0.31, 1.00)	15.5
Rod	27,198	(24, 448)	27,296	(0.34, 1.00)	99.1
Assembly	25,788	(48, 564)	10,408	(0.27, 1.00)	104.2

7 Conclusion and Future Work

We have presented a novel algorithm to use Boolean operations to generate trivariate volumetric T-splines from input CAD models. With proper bound-

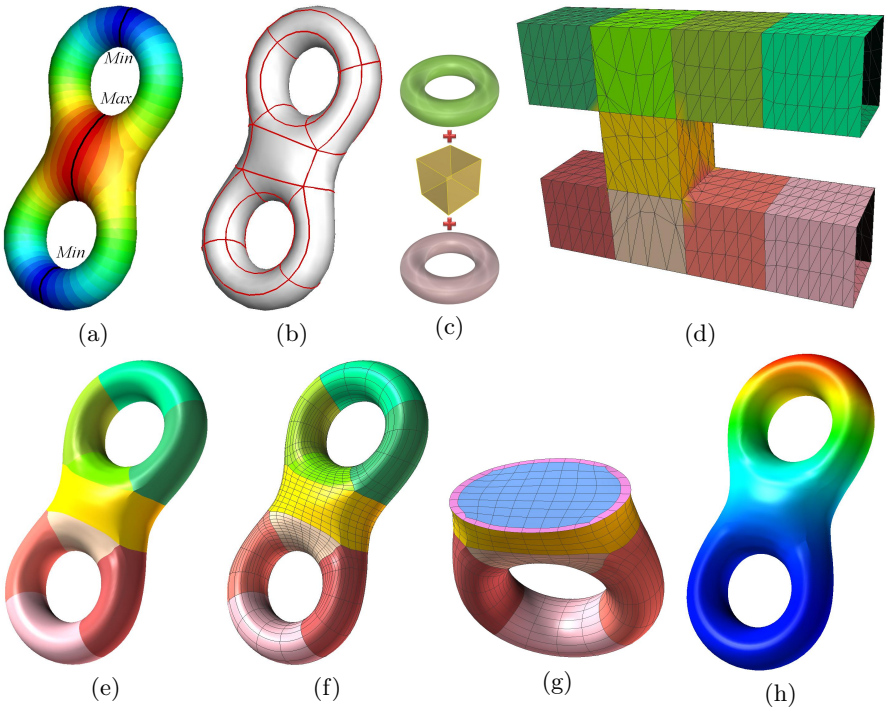


Fig. 12. Eight model. (a) One temperature field to split the two torus regions; (b) splitting result; (c) Boolean operations; (d) parametric mapping result; (e) solid T-spline; (f) solid T-spline with Bézier representation; (g) some elements are removed to show the interior of (f); (h) isogeometric analysis result;

ary conditions, a harmonic field is computed to automatically split the input geometry into hexahedral components. In addition to the cube, a new primitive (torus) is introduced in the polycube construction. After that, we perform the union and difference Boolean operations to convert the components into primitives and then map them onto the polycube. Through octree subdivision and mapping, we obtain the initial T-mesh. After making the T-mesh valid, we construct solid T-spline. The constructed solid T-spline and their Bezier representation show the efficiency and robustness of the algorithm. The presented algorithm is automatic and robust for a large class of complex models with fewer extraordinary nodes produced than other methods, but there are other models that it cannot handle, such as a tetrahedron or a cone. In addition, the input surface parameterization cannot be preserved. As part of our future work, we would like to investigate these limitations.

Acknowledgements

The work of L. Liu and Y. Zhang was supported by ONR-YIP award N00014-10-1-0698 and an ONR Grant N00014-08-1-0653. T. J.R. Hughes was sup-

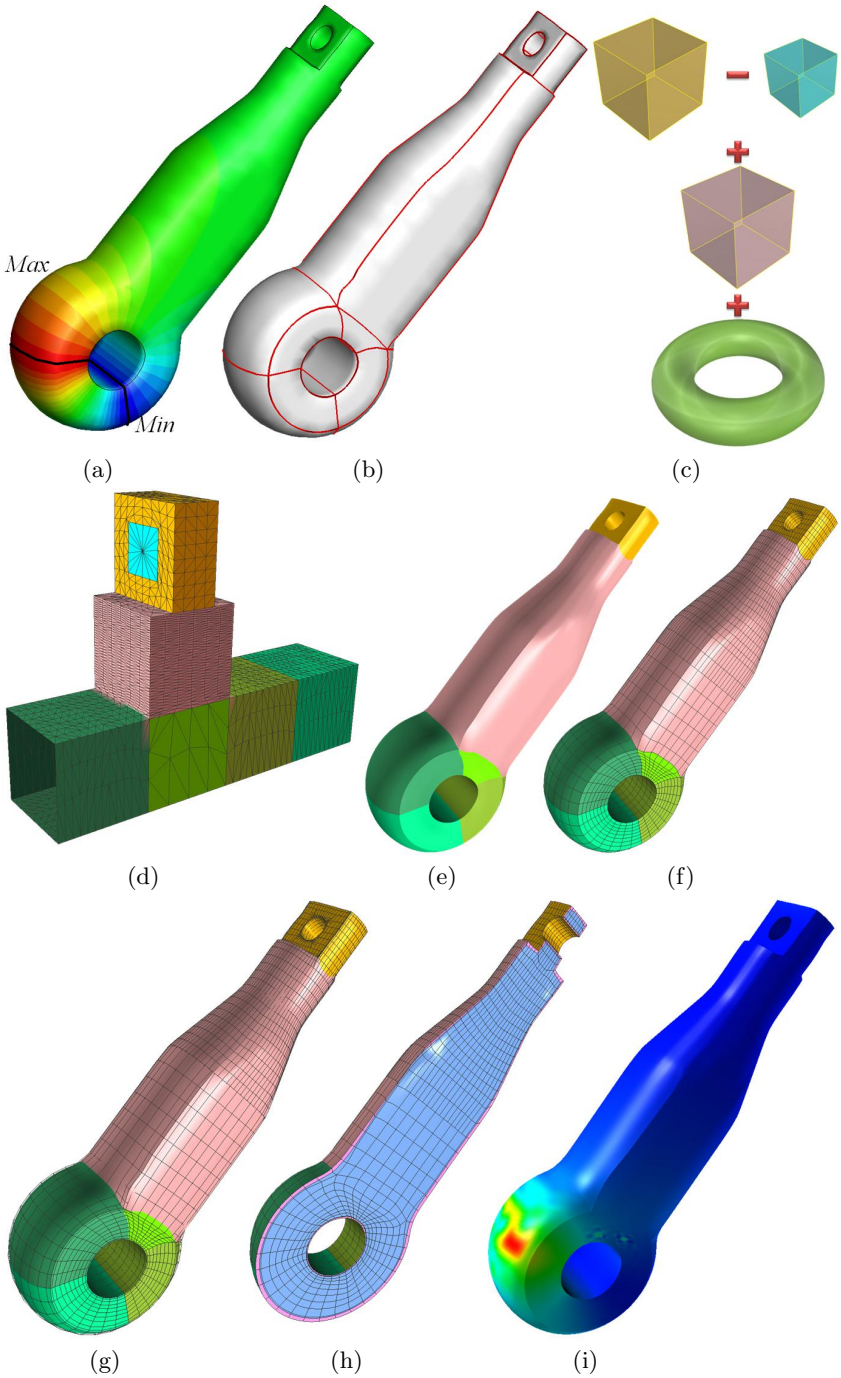


Fig. 13. Rod model. (a) One temperature field to split the bottom torus region; (b) splitting result;(c) Boolean operations; (d) parametric mapping result (the torus primitive is used in the bottom component, and the difference operation is used to create the small hole in the top component); (e) solid T-spline; (f) solid T-spline with T-mesh; (g) solid T-spline with Bézier representation; (h) some elements are removed to show the interior of (g); and (i) isogeometric analysis result.

ported by ONR Grant N00014-08-1-0992, NSF GOALI CMI-0700807/0700204, NSF CMMI-1101007 and a SINTEF grant UTA10-000374.

References

1. http://en.wikipedia.org/wiki/constructive_solid_geometry.
2. http://en.wikipedia.org/wiki/polycube.
3. B. Adams and P. Dutré. Interactive boolean operations on surfel-bounded solids. *ACM Trans. Graph.*, 22(3):651–656, July 2003.
4. M. Aigner, C. Heinrich, B. Jüttler, E. Pilgerstorfer, B. Simeon, and A.-V. Vuong. Swept volume parameterization for isogeometric analysis. In *IMA International Conference on Mathematics of Surfaces XIII*, pages 19–44, 2009.
5. Y. Bazilevs, I. Akkerman, D.J. Benson, G. Scovazzi, and M.J. Shashkov. Isogeometric analysis of lagrangian hydrodynamics. *Journal of Computational Physics*, 243:224–243, 2013.
6. Y. Bazilevs, V. M. Calo, J. A. Cottrell, J. A. Evans, T. J.R. Hughes, S. Lipton, M. A. Scott, and T. W. Sederberg. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):229–263, 2010.
7. J.A. Cottrell, T. J.R. Hughes, and Y. Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA*. Wiley, 2009.
8. J.A. Cottrell, A. Reali, Y. Bazilevs, and T.J.R. Hughes. Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics and Engineering*, 195(41-43):5257–5296, 2006.
9. J. M. Escobar, J. M. Cascón, E. Rodríguez, and 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):3210–3222, 2011.
10. R. Goldman and T. Lyche. *Knot insertion and deletion algorithms for B-spline curves and surfaces*. Society for Industrial and Applied Mathematics–Philadelphia, 1993.
11. X. Gu, Y. Wang, and S. Yau. Volumetric harmonic map. *Communications in Information and Systems*, 3(3):191–202, 2003.
12. C.A.R. Guerra. Simultaneous untangling and smoothing of hexahedral meshes. Master's thesis, University PolyTècnica De Catalunya, Barcelona Spain, 2010.
13. T. J.R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.
14. B. Li, X. Li, K. Wang, and H. Qin. Generalized polycube trivariate splines. In *Shape Modeling International Conference*, pages 261–265, 2010.
15. B. Li, X. Li, K. Wang, and H. Qin. Surface mesh to volumetric spline conversion with Generalized Poly-cubes. *IEEE Transactions on Visualization and Computer Graphics*, 99:1–14, 2012.
16. W. Li, N. Ray, and B. Lévy. Automatic and interactive mesh to T-spline conversion. In *Eurographics Symposium on Geometry Processing*, pages 191–200, 2006.
17. S. Lipton, J.A. Evans, Y. Bazilevs, T. Elguedj, and T.J.R. Hughes. Robustness of isogeometric structural discretizations under severe mesh distortion. *Computer Methods in Applied Mechanics and Engineering*, 199(58):357 – 373, 2010.

18. L. A. Piegl and W. Tiller. *The NURBS Book (Monographs in Visual Communication), 2nd ed.* Springer-Verlag, New York, 1997.
19. J. Qian, Y. Zhang, W. Wang, A.C. Lewis, MA Qidwai, and A.B. Geltmacher. Quality improvement of non-manifold hexahedral meshes for critical feature determination of microstructure materials. *International Journal for Numerical Methods in Engineering*, 82(11):1406–1423, 2010.
20. D. Schillinger, L. Dedè, M. A. Scott, J. A. Evans, M. J. Borden, E. Rank, and T. J.R. 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*, 249-252:116–150, 2012.
21. M. A. Scott, M. J. Borden, C. V. Verhoosel, T. W. Sederberg, and T. J.R. Hughes. Isogeometric finite element data structures based on Bézier extraction of T-splines. *International Journal for Numerical Methods in Engineering*, 88(2):126–156, 2011.
22. T. W. Sederberg, D. L. Cardon, G. T. Finnigan, N. S. North, J. Zheng, and T. Lyche. T-spline simplification and local refinement. In *ACM SIGGRAPH*, pages 276–283, 2004.
23. T. W. Sederberg, J. Zheng, A. Bakenov, and A. Nasri. T-splines and T-NURCCs. *ACM Transactions on Graphics*, 22(3):477–484, 2003.
24. J.M. Smith and N.A. Dodgson. A topologically robust algorithm for boolean operations on polyhedral shapes using approximate arithmetic. *Computer-Aided Design*, 39(2):149–163, 2007.
25. H. Wang, Y. He, X. Li, X. Gu, and H. Qin. Polycube splines. In *Symposium on Solid and Physical Modeling*, pages 241–251, 2007.
26. K. Wang, X. Li, B. Li, H. Xu, and H. Qin. Restricted trivariate polycube splines for volumetric data modeling. *IEEE Transactions on Visualization and Computer Graphics*, 18:703–716, 2012.
27. W. Wang, Y. Zhang, L. Liu, and T. J.R. Hughes. Solid T-spline construction from boundary triangulations with arbitrary genus topolog. *Computer Aided Design*, 45:351–360, 2013.
28. W. Wang, Y. Zhang, M. A. Scott, and T. J.R. Hughes. Converting an unstructured quadrilateral mesh to a standard T-spline surface. *Computational Mechanics*, 48:477–498, 2011.
29. W. Wang, Y. Zhang, G. Xu, and T. J.R. Hughes. Converting an unstructured quadrilateral/hexahedral mesh to a rational T-spline. *Computational Mechanics*, 50(1):65–84, 2012.
30. G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. Analysis-suitable volume parameterization of multi-block computational domain in isogeometric applications. *Computer-Aided Design*, 45(2):395–404, 2013.
31. Y. Zhang, C. L. Bajaj, and G. Xu. Surface smoothing and quality improvement of quadrilateral/hexahedral meshes with geometric flow. *Communications in Numerical Methods in Engineering*, 25:1–18, 2009.
32. Y. Zhang, Y. Bazilevs, S. Goswami, C. L. Bajaj, and T. J.R. Hughes. Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. *Computer Methods in Applied Mechanics and Engineering*, 196(29-30):2943–2959, 2007.
33. Y. Zhang, W. Wang, and T. J.R. Hughes. Solid T-spline construction from boundary representations for genus-zero geometry. *Computer Methods in Applied Mechanics and Engineering*, (249-252):185–197, 2012.