Carnegie Mellon University

CARNEGIE INSTITUTE OF TECHNOLOGY

THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF **Doctor of Philosophy**

TITLE: Volumetric T-spline Construction for Isogeometric Analysis

– Feature Preservation, Weighted Basis and Arbitrary Degree

PRESENTED BY: Lei Liu

ACCEPTED BY THE DEPARTMENT OF : Mechanical Engineering

APPROVED BY THE COLLEGE COUNCIL :

Volumetric T-spline Construction for Isogeometric Analysis – Feature Preservation, Weighted Basis and Arbitrary Degree

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Department of Mechanical Engineering

Lei Liu

B.S., Mechanical and Aerospace Engineering, Harbin Institute of Technology

M.S., Engineering Mechanics, University of Science and Technology of China

Carnegie Mellon University

Pittsburgh, PA

September 2015

Abstract

Constructing spline models for isogeometric analysis is important in integrating design and analysis. Converting designed CAD (Computer Aided Design) models with B-reps to analysis-suitable volumetric T-spline is fundamental for the integration. In this thesis, we work on two directions to achieve this: (a) using Boolean operations and skeletons to build polycubes for feature-preserving high-genus volumetric T-spline construction; and (b) developing weighted T-splines with arbitrary degree for T-spline surface and volume modeling which can be used for analysis.

In this thesis, we first develop novel algorithms to build feature-preserving polycubes for volumetric T-spline construction. Then a new type of T-spline named the weighted T-spline with arbitrary degree is defined. It is further used in converting CAD models to analysis-suitable volumetric T-splines. An algorithm is first developed to use Boolean operations in CSG (Constructive Solid Geometry) to generate polycubes robustly, then the polycubes are used to generate volumetric rational solid T-splines. By solving a harmonic field with proper boundary conditions, the input surface is automatically decomposed into regions that are classified into topologically either a cube or a torus. Two Boolean operations, union and difference, are performed with the primitives and polycubes are generated by parametric mapping. With polycubes, octree subdivision is carried out to obtain a volumetric T-mesh. The obtained T-spline surface is C^2-continuous everywhere except the local region surrounding irregular nodes, where the surface continuity is elevated from C^0 to G^1. Bézier elements are extracted from the constructed solid T-spline models, which are further used in isogeometric analysis. The Boolean operations preserve the topology of the models inherited from design and can generate volumetric T-spline models with better quality.

Furthermore, another algorithm is developed which uses skeleton as a guidance to the polycube construction. From the skeleton of the input model, initial cubes in the interior are first constructed. By projecting corners of interior cubes onto the surface and generating a new layer of boundary cubes, the entire interior domain is split into different cubic regions.

With the splitting result, octree subdivision is performed 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.

Weighted T-spline basis functions are designed to satisfy partition of unity and linear independence. The weighted T-spline is proved to be analysis-suitable. Compared to standard T-splines, weighted T-splines have less geometrical constraint and can decrease the number of control points significantly. Trimmed NURBS surfaces of CAD models are reparameterized with weighted T-splines by a new edge interval extension algorithm, with bounded surface error introduced. With knot interval duplication, weighted T-splines are used to deal with extraordinary nodes. With Bézier coefficient optimization, the surface continuity is elevated from C^0 to G^1 for the one-ring neighborhood elements. Parametric mapping and sweeping methods are developed to construct volumetric weighted T-splines for isogeometric analysis.

Finally, we develop an algorithm to construct arbitrary degree T-splines. The difference between odd degree and even degree T-splines are studied in detail. The methods to extract knot intervals, calculate new weights to handle extraordinary nodes, and extract Bézier elements for analysis are investigated with arbitrary degrees. Hybrid degree weighted T-spline is generated at designated region with basis functions of different degrees, for the purpose of performing local p-refinement. We also study the convergence rate for T-spline models of different degrees, showing that hybrid degree weighted T-splines have better performance after p-refinement.

In summary, we develop novel methods to construct volumetric T-splines based on polycube and sweeping methods. Arbitrary degree weighted T-spline is proposed, with

proved analysis-suitable properties. Weighted T-spline basis functions are used to reparameterize trimmed NURBS surfaces, handling extraordinary nodes, based on which surface and volumetric weighted T-spline models are constructed for isogeometric analysis.

Keywords: Volumetric T-spline Construction, Feature Preservation, Weighted T-spline, Extraordinary Node, Arbitrary Degree, Isogeometric Analysis

Acknowledgements

First of all, I am deeply grateful to my advisor Dr. Yongjie Jessica Zhang for her support, guidance, and inspiration through my doctorial study. The advices and help have been invaluable at both an academic and a personal level, for which I am extremely grateful.

Furthermore, I would like to thank my committee, Dr. Kenji Shimada, Dr. Levent Burak Kara, and Dr. John Brigham for their support and insightful comments.

I thank the members of our research group, Dr. Wenyan Wang, Dr. Xinghua Liang, Dr. Jin Qian, Dr. Luoting Fu, Dr. Tao Liao, Yuanfeng Jiao, Kangkang Hu, Xiaodong Wei, Arjun Kumar, Ling Zhan, Runtian Liu, Yicong Lai, Joshua W. Chen, Aishwarya Pawar and Yang Gao. I feel lucky to get to know and collaborate with them, and I will always treasure the time we spent together. I also thank our collaborators Xinge Li, Hugo Casquero Penelas, Dr. Hector Gomez and Onofre Marco Alacid for the happy collaboration which has broaden my research perspectives.

I would like to thank my parents, my brother and my sister for their love and infinite support throughout my life. I hope that this work makes them be proud of.

This work was supported by Y. Zhang's PECASE award N00014-14-1-0234, ONR-YIP award N00014-10-1-0698, ONR grant N00014-08-1-0653 and CIT Dean's Fellowship.

To My Family.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

To integrate engineering design and analysis, isogeometric analysis was proposed which utilizes the same basis for both geometrical representation and numerical analysis. Pioneering research has demonstrated the efficiency and accuracy of isogeometric analysis. However, boundary representation with NURBS surfaces is commonly used in design, while polygonal meshes are mainly used in analysis. The integration requires one reparameterization step to generate spline models from the boundary representations. Besides, the designed CAD models are mostly not water-tight, which need to be reparameterized with T-splines for analysis. Furthermore, not all T-spline models can be used in isogeometric analysis, and only a subset of T-splines possess analysis-suitable properties. To facilitate isogeometric analysis, a fundamental step is to automatically construct analysis-suitable spline models. However, due to the technical challenges, very few research has been done in this area.

T-spline, as one superior alternative to NURBS, has more flexibility for both geometric modeling and analysis. It allows local refinement and works for non-rectangular domain in 2D and non-cubic domain in 3D. The localization of T-spline basis functions makes it possible for complicated geometry modeling and volumetric spline construction. However,

not all T-splines are analysis-suitable. Only a subset of standard T-splines with geometrical constraints can be used directly in analysis. Developing new types of analysis-suitable T-spline basis functions with little constraints on geometric modeling is one crucial step in the design and analysis integration study.

Polycubes have been used to generate solid T-spline models for analysis, but they only work for certain types of geometry. In this thesis, we intend to make the construction of trivariate solid T-spline from the polygonal meshes more robust and adaptive for complex geometries with feature preservation. Besides, a new type of T-spline, named the weighted T-spline, is developed with analysis-suitable properties. Based on weighted T-splines, the reparameterization of trimmed NURBS surfaces and handling of extraordinary nodes are studied. Most T-spline models used in analysis are cubic. However, complex analysis problems such as vibration of shell structures requires higher degree T-spline models. The construction of arbitrary degree T-spline models is also an open problem.

T-spline modeling can be used in various analysis fields, such as linear elasticity, heat transfer and fluid-structure interaction problems. After demonstrating the advantage of using T-spline models in analysis, the next step would be using complex T-spline models in more complicated analysis applications.

1.2 Problem Statement

In this thesis, the detailed algorithms are presented for volumetric T-spline construction for isogeometric analysis. There are four main problems:

- **Using Boolean operations in volumetric T-spline construction.** Boolean operations in CAD design are used in the polycube construction for volumetric T-splines. Input models are first split into different regions, represented with two types of primitives: cube and torus. Two Boolean operations (union and difference) are used to construct polycubes from the split regions. Smoothing and optimization are used to

improve the quality of the T-meshes. With a degree elevation technique, the surface continuity near the extraordinary node is increased from C^0 to G^1.

- **Skeleton based feature-preserving T-mesh construction.** Skeletons of the input models are used as a guidance to the polycube construction. Interior and boundary cubes are constructed from the skeleton, with limited number of singularity points introduced. The generated T-mesh follows the topology of the input model. Surface features are classified into three types: open curve, closed curve, and singularity features. Parametric mapping, volumetric parameterization, skeleton modification and templates are the developed algorithms to preserve them during T-mesh construction.

- **Analysis-suitable weighted T-spline.** A new type of T-spine, named the weighted T-spline, is developed for modeling and analysis. The weighted T-spline basis functions are designed to satisfy partition of unity and linear independence. The weighted T-splines are demonstrated to be analysis-suitable. Compared to standard T-splines, for the same level of refinement, weighted T-spines use fewer control points and with less geometric constraints to the control mesh. An edge interval extension algorithm is introduced to reparameterize trimmed NURBS surfaces. Weighted T-splines can also be used to deal with extraordinary nodes with designed knot interval duplication, with G^1 surface continuity across the edges shared by two one-ring neighboring elements. Parametric mapping and sweeping methods are developed to construct volumetric weighed T-splines for analysis.

- **Hybrid degree T-spline surface construction for isogeometric analysis.** T-spline basis functions with arbitrary degree are investigated in detail. Knot interval extraction, T-junction extension, and Bézier extraction are studied and compared for odd degree and even degree T-splines. Hybrid degree weighted T-splines with multiple degree basis functions are developed for the p-refinement of designated local region. To achieve the same convergence rate in analysis, even degree T-spline models need

fewer degrees of freedom than the one degree higher odd degree models. Hybrid degree weighted T-splines can achieve better performance with local p-refinement in analysis.

1.3 Contributions

This thesis focuses on surface and volumetric T-spline modeling. Two polycube-based algorithms are developed for volumetric T-spline construction. Based on weighted T-splines, schemes are developed to reparameterize trimmed NURBS surfaces, handle extraordinary nodes and perform local p-refinement. This thesis has five main contributions:

- Developed a novel method using Boolean operations to generate polycubes for volumetric T-spline construction;
- Developed a robust algorithm to use skeletons of input models as a guidance for generating feature-preserving T-meshes;
- Propose a weighted T-spline basis concept which can generate analysis-suitable basis functions for T-spline modeling;
- Developed an algorithm for reparameterizing trimmed NURBS surfaces with T-spline surfaces with bounded surface error, and handling extraordinary nodes; and
- Developed an algorithm with efficient data structures for arbitrary degree T-splines and hybrid degree weighted T-splines construction for complex surface models.

1.4 Publication

During my Ph.D. study, I have co-authored five journal papers, five conference papers, one book chapter, with another four journal papers in preparation. I will finish these four journal papers with our collaborators and submit them out before my graduation.

Journal publications:

- **L. Liu**, Y. Zhang. "Hybrid Degree Weighted T-spline Surface Construction for Isogeometric Analysis". *In preparation*.

- **L. Liu**, Y. Zhang. "Handling Extraordinary Nodes and Volumetric T-spline Construction Using Weighted Basis Functions". *In preparation*.

- H. Casquero, **L. Liu**, C Bona-Casas, Y. Zhang, A. Reali, H. Gomez. "Isogeometric Collocation for Second and Fourth Order Problems Using Arbitrary Degree T-splines". *In preparation*.

- G. Lorenzo, M. A. Scott, K. B. Tew, T. J. R. Hughes, Y. Zhang, **L. Liu**, G. Vilanova, H. Gomez. "Tissue-scale, Patient-specific Modeling of Prostate Cancer Growth: a New Perspective". 2015. *In preparation*.

- H. Casquero, **L. Liu**, C Bona-Casas, Y. Zhang, H. Gomez. "A hybrid Galerkin-Collocation Immersed Method for Fluid-Structure Interaction Using Unstructured T-splines". *International Journal for Numerical Methods in Engineering*, accepted, 2015.

- **L. Liu**, Y. Zhang, X. Wei. "Weighted T-spline and Its Application in Reparameterizing Trimmed NURBS Surfaces". *Computer Methods in Applied Mechanics and Engineering*, 295:108-126, 2015.

- **L. Liu**, Y. Zhang, Y. Liu, W. Wang, "Feature-Preserving T-mesh Construction Using Skeleton-based Polycubes". *Computer-Aided Design*. 58(0):162-172, 2015.

- **L. Liu**, Y. Zhang, T. J. R. Hughes, M. A. Scott, T. W. Sedergerg. "Volumetric T-Spline Construction Using Boolean operations". *Engineering with Computers*, 30(4):162-172, 2014.

- W. Wang, Y. Zhang, **L. Liu**, T. J.R. Hughes. "Trivariate Solid T-spline Construction from Boundary Triangulations with Arbitrary Genus Topology". *Computer Aided Design*. 45(2):351-360, 2013.

Conference publications:

- **L. Liu**, Y. Zhang, X. Wei. "Handling Extraordinary Nodes with Weighted T-spline Basis Functions". *24th International Meshing Roundtable*. Austin TX, USA. Oct 12-14, 2015.

- **L. Liu**, Y. Zhang, X. Wei. "NURBS Surface Reparameterization Using Truncated T-splines". *23rd International Meshing Roundtable*. London, UK. Oct. 12-15, 2014.

- **L. Liu**, Y. Zhang, Y. Liu, W. Wang, "Feature-Preserving T-mesh Construction Using Skeleton-based Polycubes". *Symposium on Solid and Physical Modeling*. Hong Kong. Oct. 26-28, 2014.

- **L. Liu**, Y. Zhang, T. J. R. Hughes, M.A. Scott, T. W. Sedergerg. "Volumetric T-Spline Construction Using Boolean Operations". *22nd International Meshing Roundtable*, pp. 405-424. Orlando, FL. Oct. 13-16, 2013.

- W. Wang, Y. Zhang, **L. Liu**, T. J.R. Hughes. "Solid T-spline Construction from Boundary Triangulation with Arbitrary Genus Topology". *Symposium on Solid and Physical Modeling*. University of Burgundy, Dijon, France. Oct. 29-31, 2012.

Book Chapter:

- Y. Lai, **L. Liu**, Y. Zhang, J. Chen, E. Fang, J. Lua. "Rhino 3D to Abaqus Design-Through-Analysis: A T-spline Based IGA Software Platform". The edited volume of the Modeling and Simulation in Science, Engineering and Technology Book Series devoted to AFSI 2014 - a birthday celebration conference for Tayfun Tezduyar. Springer Publisher. Editors: Yuri Bazilevs and Kenji Takizawa, accepted, 2015.

1.5 Outline of Dissertation

Following the introduction, Chapter 2 gives a background literature review. Chapter 3 specifically discusses using Boolean operations to construct polycubes for solid T-splines. Chapter 4 presents an algorithm for skeleton based polycube construction with feature preservation. In Chapter 5, weighted T-splines are developed with analysis-suitable properties, and are used to reparameterize trimmed NURBS surfaces, handle extraordinary nodes and generate volumetric T-splines. Chapter 6 discusses arbitrary-degree T-spline modeling, hybrid degree weighted T-splines and their application in isogeometric analysis.

Chapter 2

Literature Review

2.1 T-splines

T-spline is a new type of mathematical tool for geometrical modeling [85]. It is developed based on and compatible with NURBS [69]. NURBS is the standard form for representing free-form curves and surfaces in design and manufacturing. However, there are two drawbacks: 1) they do not allow local refinement. When refining a NURBS patch, the refinement is performed globally and redundant control points are inserted; 2) all the control points must lie in a rectangular grid topologically, and for complicate geometries, multiple NURBS patches are used. Also non-water-tight problem exists in the multiple patch NURBS models. To overcome these drawbacks and provide a better tool for design, T-spline is developed [85]. Different from NURBS, T-splines allow T-junctions and extraordinary nodes. For T-spline surfaces, a T-junction terminates a row or column of control points in the control grid. With T-junctions, local refinement can be carried out within desired region. An extraordinary node has valence other than four and is not a T-junction. They ensure that the control mesh can be non-rectangular or non-cubic region. Generally, NURBS are recognized as a subset of T-splines, which have rectangular control mesh without T-junctions.

T-spline simplification and local refinement were first discussed in [83], introducing the nesting property and refinement matrix. Then the coarsening of T-spline was studied in [93] as the reverse of T-spline local refinement. The linear independence of T-spline basis functions [16, 53] presents the topological constraints to the T-mesh configuration. The T-spline-to-NURBS transformation matrix should be in full rank for linearly independent T-splines [53]. An optimized T-junction extension algorithm was developed for local refinement of analysis-suitable T-splines [80]. Some adaptive refinement schemes have a linear computational complexity [64], preserving linear independence of the basis functions. Such scheme was further extended to volumetric T-splines [63]. Hierarchical T-splines were proposed in [27], together with its analysis-suitability and refinability. Truncated hierarchical basis functions were also studied to perform local refinement while ensuring partition of unity and linear independence, such as truncated hierarchical B-splines (THB) [32], and truncated hierarchical Catmull-Clark surfaces (THCCs) [94].

T-spline models were first constructed with cubic basis functions. Then T-splines were extended to arbitrary odd and even degrees [29], as well as mixed degrees in two parametric directions. Analysis-suitable T-splines of arbitrary degrees were first proposed in [14, 27], in which T-junction extensions from different parametric directions should not meet each other. Cubic analysis-suitable T-splines were then studied in [51, 49], characterizing the refineability and linear independence. Analysis-suitable T-spline spaces were characterized as spaces of piecewise polynomials with appropriate linear constraints on the sub-domain interfaces [15].

2.2 Isogeometric Analysis

Isogeometric analysis uses the same basis functions for geometric modeling and numerical analysis. Isogeometric analysis using NURBS models was first introduced in [40], demonstrating its high accuracy and robustness. A Bézier extraction method was proposed for

NURBS based isogeometric analysis, providing an element structure similar with conventional finite element method [78]. T-splines were introduced into isogeometric analysis [9, 2, 25] with its local refinement property. The detailed data structure for isogeometric analysis using T-splines was studied in [79]. The T-spline basis functions was proved to satisfy analysis-suitable properties for analysis such as non-negativity, partition of unity and linear independence [66]. Error estimation of local h-refinement [8] provides a theoretical foundation to the evaluation of accuracy and convergence rate of isogeometric analysis.

Isogeometric analysis was first used to solve linear elasticity and heat transfer problems [9]. Isogeometric analysis using NURBS models were then used to analyze blood flow in patient-specific vascular structures [98]. Isogeometric analysis in structure vibrations were studied with structural models including rods, thin beams, membranes, and thin plates [23]. NURBS represented circular geometries was used to study the flows about rotating components with a stationary flow domain without geometric incompatibility [13]. Frictionless contact problems between deformable bodies can also be solved with isogeometric analysis [24]. Isogeometric analysis can also be solve fluid-structure interaction problems. The theory was first introduced in [10], then its application was studied in arterial blood flow [11], and wind turbines [12]. Taking the advantage of higher degree basis functions, isogeometric collocation methods were developed, showing better accuracy compared to Galerkin method [6, 3]. Even order NURBS and T-splines have the same convergence rate with the one degree higher odd degree NURBS and T-spline models [76] in solving fluid-structure interactions [17] and higher order boundary-value problems [18].

2.3 Surface and Volumetric T-Spline Modeling

The main obstacle in isogeometric analysis using T-splines is how to robustly construct surface and volumetric T-spine models. Different approaches have been developed for surface and volumetric T-spline modeling. By locally inserting T-junctions, T-spline can

be constructed while ensuring that the refined T-spline surface is still analysis-suitable [80]. Quad meshes were recognized as initial T-meshes to constructed standard T-spline surfaces with template method [91]. The template method was further extended to 3D, which converts hexahedral meshes to volumetric T-splines [92]. Periodic Global Parameterization was proposed to convert triangular meshes to T-splines [48]. Solid T-splines were constructed by parametric mapping of tetrahedral meshes generated by Meccano method from the input surface mesh [26]. In [88], a parametric mapping between a polycube and a surface geometry was presented to construct trivariate T-splines from input triangular meshes. Polycubes and parametric mapping were used together to generate solid T-spline models [101, 90]. T-spline surfaces were directly used to construct conformal volumetric T-splines of genus zero [102]. A generalized polycube method using T shape templates was introduced to handle high-genus models and extraordinary nodes for T-spline construction [88]. The generalized polycubes were further extended to generate volumetric splines from surface meshes, with no singularity and controllable number of ill-points [47].

2.4 Hexahedral Meshing and Volumetric Parameterization

Since T-mesh can be recognized as hexahedral mesh which allows T-junctions, we may apply hex meshing algorithms to solid T-spline construction. Octree-based methods were developed to generate adaptive hex dual-meshes [7, 77, 96, 100], and improved to preserve sharp features [60, 74]. These grid-based methods are robust, but they yield many singular points on the surface. And the resulting mesh is highly influenced by the orientation of the grid. Polycubes were used in hexahedral meshing with better quality [35, 95]. A constrained discrete optimization technique was developed for better mesh segmentation and volumetric parameterization using polycubes [59]. In [95], hex remeshing was performed based on polycube construction and optimization. L_1 based polycubes for complex geometries were

proposed for hex meshing [38]. Harmonic volumetric mapping was employed in hex meshing with better boundary feature capture [52]. CubeCover method uses 3D frame fields to perform volumetric parameterization and all-hex mesh generation [67] . This method was then extended [54] with developed algorithm to generate a singularity-restricted frame field for all-hex meshing. A boundary aligned cross-field was studied in [39], which uses spherical harmonics to represent the 3D field. The field was then improved with singularity correction for hex mesh extraction [41]. However, performing volumetric parameterization from frame field is not robust, especially for complex geometries.

2.5 Converting CAD models to Spline Models for Analysis

Algorithms have been developed to build water-tight T-spline surfaces [84] and volumetric T-splines [102]. But there is no algorithms that can directly convert designed CAD models from design software to T-spline models for analysis. The main reason is that most CAD models are represented with boundary NURBS patches, and they are not water-tight. In addition, Boolean operations in design result in trimmed NURBS patches which do not have a tensor product representation. For these trimmed surfaces, the original NURBS surfaces, the trimming curves and the trimming operations are stored in the output files (i.e. the IGES files). Such data storage format prohibits the direct use of CAD models in analysis. Isogeometric analysis for trimmed CAD models was studied in [43], in which high order triangular mesh was used instead of NURBS along the trimming curve. This method is further extended to arbitrary complex topology [44]. Edge graphs were used to split the CAD model into four types of base hex, which can then be used for volumetric NURBS and T-spline construction [42, 65]. The method can work with both convex and non-convex edges, but the splitting result may have large distortions.

2.6 Degree Elevation of B-splines

To construct T-splines of arbitrary degree, we also study the degree elevation of B-splines. Degree elevation of B-spline curves was first studied in [70] presenting an algorithm to elevate the degree by one. Efficient algorithms were developed to raise the degree of B-spline curves and surfaces [19, 20]. They present a solution to the problem of representing a curve of degree m with the linear combination of curves of degree $m+1$. Then a fast algorithm was developed to elevate the degree by arbitrary times [71]. A software-engineering approach was presented for degree elevation of B-spline curves [68], which provides competitive performance in speed and numerical accuracy. Bézier basis functions were also used for the degree elevation of B-splines, which first raise the degree of extracted Bézier elements and then remove unnecessary knots to obtain the desired surface continuity [69].

Chapter 3

Volumetric T-spline Construction Using Boolean Operations

In this chapter, we present a novel algorithm for constructing a volumetric T-spline from input surface triangle meshes inspired by 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. The obtained T-spline surface is C^2 everywhere except the local region surrounding irregular nodes, where the surface continuity is elevated from C^0 to G^1. The extracted trivariate Bézier elements from the volumetric T-spline can be directly in isogeometric analysis.

The Boolean operation method preserves the topology of the input models, and can decrease the number of extraordinary nodes on the surface. The generated polycubes follow the designing process, and preserve the topology feature of the input.

3.1 Algorithm Overview

Polycube-based methods for volume parameterization [90, 89, 46] 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, here we propose to use Boolean operations to build the polycubes. As shown in Fig. 3.1, 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.

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 a commercial software (in this case, 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. Each torus is composed topologically of four cubes. All cubes generated by the domain decomposition are then combined together and holes (represented topologically as cubes) are subtracted. 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 onto 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 [91, 92] to each irregular node in the T-mesh. Finally, volumetric T-spline is generated and Bézier elements are extracted for isogeometric analysis.

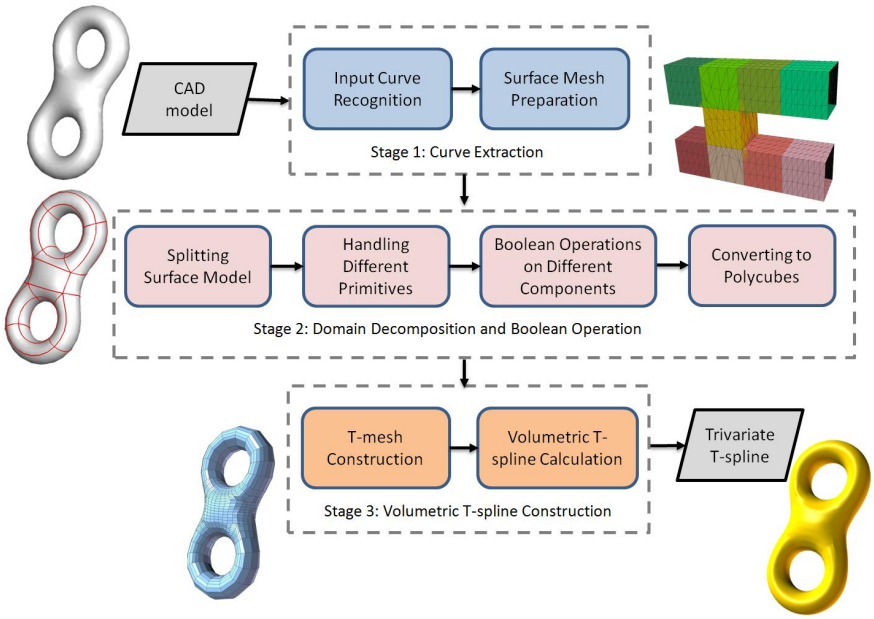


Figure 3.1: Three stages of volumetric T-spline construction using Boolean operations.

3.2 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. The edges which are used to represent the polycube edges are *feature curves* and the remaining edges we call *difference curves*. For example in Fig. 3.2, 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 3.2, 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. 3.2, 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. The criterion to classify the curves into different groups is whether the feature curves can be easily used in the following parametric mapping.

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. 3.2, 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.

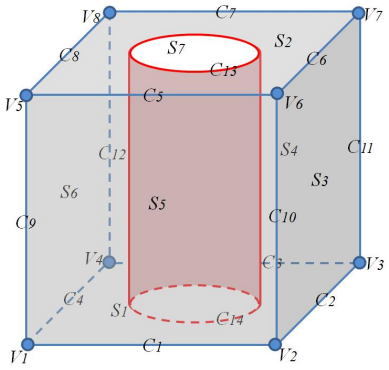


Figure 3.2: Classification of curve information. Blue line: feature curves; and red lines: difference curves.

3.3 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.

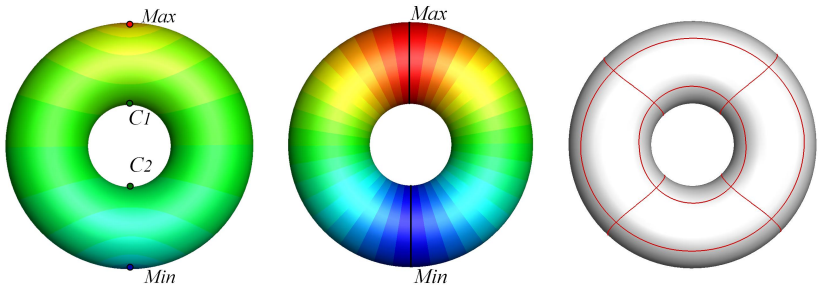


Figure 3.3: Splitting one torus model into four cubes. (a) Set the top and bottom points with max and min temperature respectively, calculate the harmonic temperature field, and find out critical points (extreme and saddle points); (b) recalculate the harmonic field by setting the whole cross section to be max/min temperature; and (c) split the model with isoparametric and gradient lines.

3.3.1 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 deduced from the design process and boundary information. These virtual components are the result of CSG difference operation in design. For example in Fig. 3.2, the feature and difference curves can split the model into one cube and one virtual cylinder.

Different from pants decomposition of surfaces [37], which relies more on the topology of the models, we use harmonic fields to split a complex geometry into coherent regions [101, 90]. 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. 3.3, we use the following five steps to split the torus model into four hexahedral components:

1. First we find out the geometrically highest/lowest points, and assign them the max and min temperature respectively;

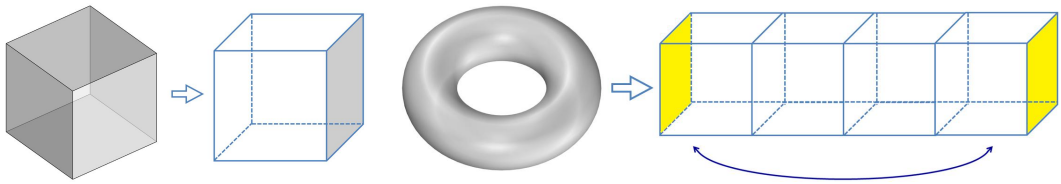
2. A harmonic field is calculated by solving a heat transfer problem using a discretized Laplace operator over the surface mesh, see Fig. 3.3(a);

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. 3.3(b); and

5. Four equally-spaced points are selected on each cross section curve (black curves) in Fig. 3.3(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. 3.3(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. The proper boundary condition can produce one harmonic field with isoparametric lines and gradient lines for better domain splitting results which follow the geometry. 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. For example we compute the field twice for the torus model. Proper boundary conditions may not be easy to find, in this paper we assume they are given by the user.



(a) (b)

Figure 3.4: Two primitives from the physical space to the parametric space. (a) Cube; and (b) torus.

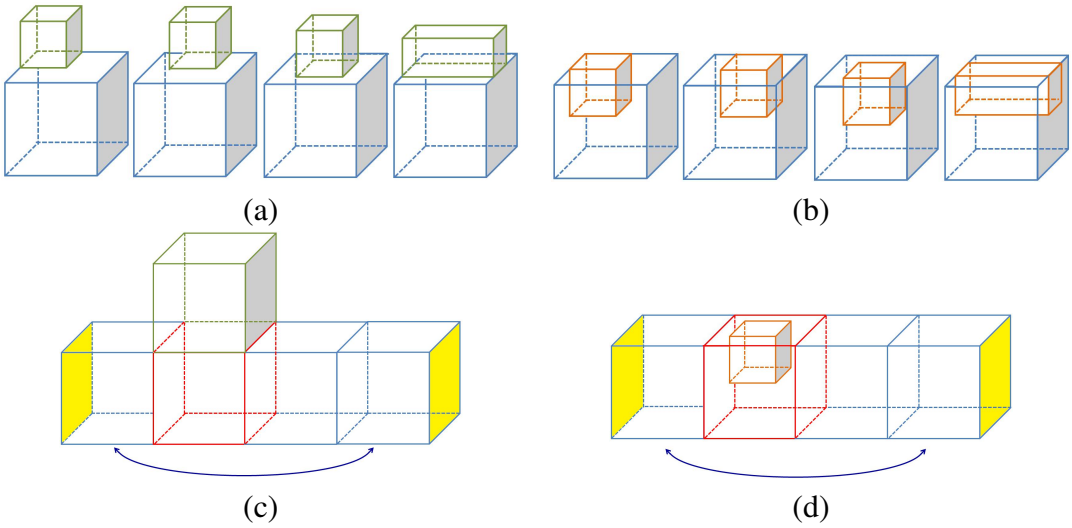


Figure 3.5: Boolean operations of cubes and torus with different sizes and relative position. (a) Four cases for the union operation of two cubes; (b) four cases for the difference operation of two cubes; (c) the union operation of a cube and a torus; and (d) the difference operation of a cube and a torus.

3.3.2 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 do not need to have the same length. Fig. 3.4 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.

3.3.3 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 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

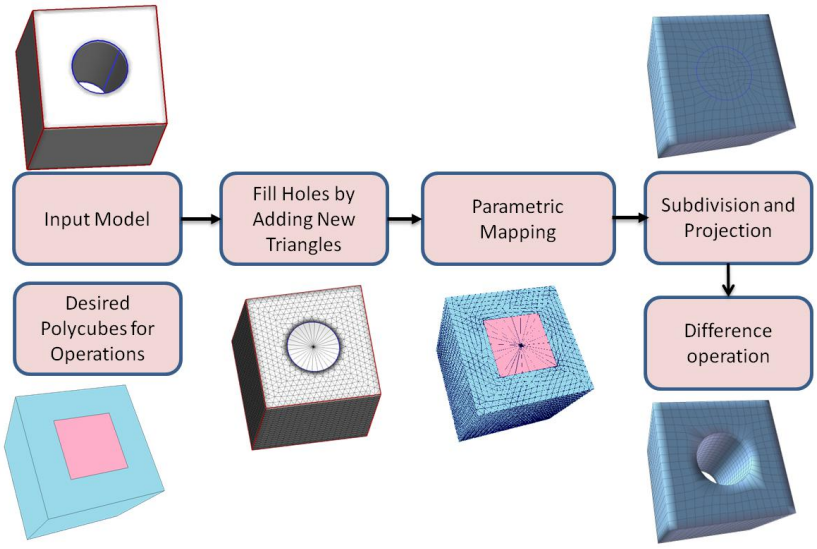


Figure 3.6: Steps to perform the difference operation. Holes are filled to create a virtual component (virtual cylinder).

cases for the union and difference operations between them, see Fig. 3.5(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. 3.5(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. 3.6, 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.

Discussion: The torus primitive and the difference operation are two new features in our polycube generation, which provide more convenience and flexibility in handling designed CAD models. Since torus shape components are common in CAD models, representing them with four consecutive cubes connected with each other can produce elements following the circumferential directions without distortion. 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.3.2). 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.

3.4 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.

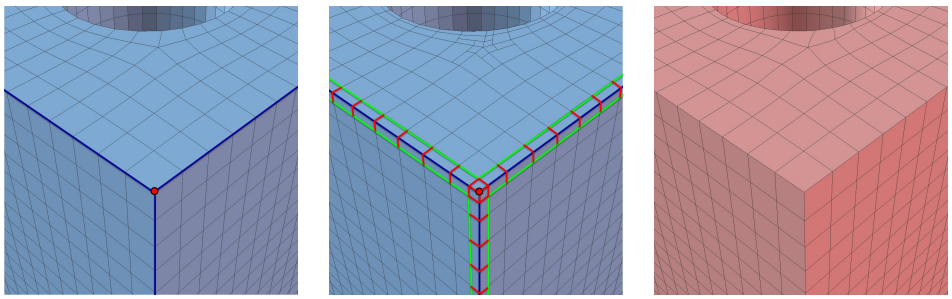
3.4.1 Adaptive Octree Subdivision and Mapping

An initial T-mesh is generated by applying an adaptive octree subdivision to the polycubes and mapping to the boundary [30, 101]. 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 Euclidean 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 input boundary. The physical coordinates of each interior node are calculated by a linear interpolation. The linear interpolation can be expressed as $p = \sum p_i w_i / \sum w_i$, where p_i are

the physical coordinates of a neighboring node and w_i is the interpolation weight. T-junctions are introduced if two neighboring elements have different subdivision levels.

3.4.2 Sharp Feature Preservation and Quality Improvement

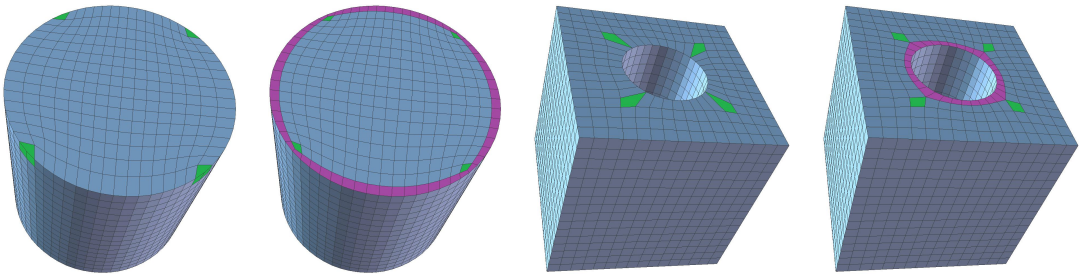
Sharp Feature Preservation: To preserve the detected sharp features, we duplicate their corresponding parametric lines in the polycubes [91]. It aims to decrease the local boundary surface continuity across the sharp curves to C^0 by repeating knots. As shown in Fig. 3.7, a sharp curve (blue curve) is 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. 3.7(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.



(a) (b) (c)

Figure 3.7: 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.

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 [62, 75, 97], which guarantees each element has at most one face lying on the boundary and also improves the surface continuity across the polycube edges from C^0 to C^2. The sharp feature information on the input surface can also be



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

Figure 3.8: 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.

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. Fig. 3.8(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. 3.8(b). This method can also be applied to the surface of virtual cylinders, see Fig. 3.8(c-d). Since after pillowing there are one layer of elements with zero volume and zero Jacobian value, smoothing and optimization [101] 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 can only 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

$$J = det(J_M) = \begin{vmatrix} \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \zeta} \\ \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \zeta} \\ \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \zeta} \end{vmatrix}, \tag{3.1}$$

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

$$J_s = \frac{J}{\| J_M(\cdot,0) \| \| J_M(\cdot,1) \| \| J_M(\cdot,2) \|}, \tag{3.2}$$

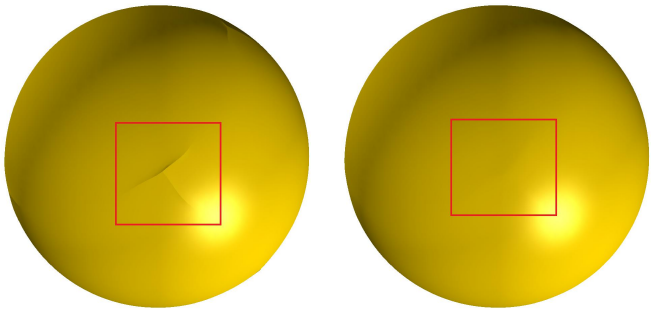
where $J_M(\cdot,0)$, $J_M(\cdot,1)$ and $J_M(\cdot,2)$ represent the first, second and last column of the Jacobian matrix, J_M, respectively. The objective function is equivalent to the maximization of the minimum scaled Jacobian. To get better optimization results, we further improve our optimization 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 [55], 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 optimized step size. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) method [36] is used to perform the optimization.

3.4.3 Irregular Nodes and Volumetric T-spline Construction

Extraordinary nodes or partial extraordinary nodes [101] are two types of irregular nodes in T-spline construction. 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 [91, 92].

The rational solid T-spline was defined in [92]. 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 functions, 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. The transformation matrix M from T-spline basis functions to Bézier basis functions is calculated by the Oslo knot insertion algorithm [34]. When calculating the matrix M, we find out that it is unique and always full rank, which means that the T-spline basis functions are linearly independent. With the extracted Bézier elements, we can perform isogeometric analysis on the volumetric T-spline models.



(a) (b)

Figure 3.9: Surface continuity elevation of sphere model. (a) Before continuity elevation (the surface continuity is C^0 in the two ring neighborhood of one irregular node); and (b) after degree elevation (the surface continuity is G^1).

3.4.4 Surface Continuity Elevation

The continuity around the extraordinary and partial extraordinary nodes is C^0 after applying the templates. While C^0 smoothness is sufficient for many types of analyses, it has been shown that increased smoothness in the FE basis improves solution accuracy and robustness

[28, 23]. Additionally, on the surface of the geometry, it is important to recover the smoothness present in the original CAD surface geometry. For these reasons, we use a continuity elevation technique [81] to improve the surface continuity around irregular nodes, and extend this technique into volumetric T-spline with the following four steps.

1. First we find out all the irregular nodes on the surface which do not lie on the sharp feature;

2. Templates are applied to these nodes, yielding C^0-continuity in their two-ring neighborhoods;

3. Bézier control nodes are calculated with the basis function transformation matrix M; and

4. All the Bézier control nodes on the surface form the control net of the surface patches. For each irregular node on the surface, we apply the constrained optimization framework [81] to the Bézier control nodes in its two-ring neighborhood, then the local surface continuity around irregular nodes is elevated from C^0 to G^1.

Fig. 3.9 shows the surface continuity elevation result of a sphere model. In our volumetric T-spline construction, only the control nodes on the surface have non-zero basis functions on the T-spline surface, which ensures that we only have to optimize the Bézier control nodes on the surface to improve the surface continuity.

Table 3.1: Statistics of all the tested models.

Model	T-mesh nodes	T-mesh elements	Irregular nodes (surface, interior)	Bézier elements	Jacobian (worst, best)	Modeling Time (s)
Torus	5,920	3,072	(0, 128)	3,072	(0.42, 1.00)	11.8
Eight	8,347	3,472	(8, 196)	4,096	(0.31, 1.00)	15.6
Rod	27,282	13,136	(24, 448)	27,296	(0.34, 1.00)	99.2
Assembly	29,850	9,248	(48, 692)	11,776	(0.32, 1.00)	87.1
Cross-hole	29,058	9,864	(32, 680)	9,920	(0.43, 1.00)	84.4
Joint	22,402	8,808	(32, 455)	8,808	(0.30, 1.00)	70.1

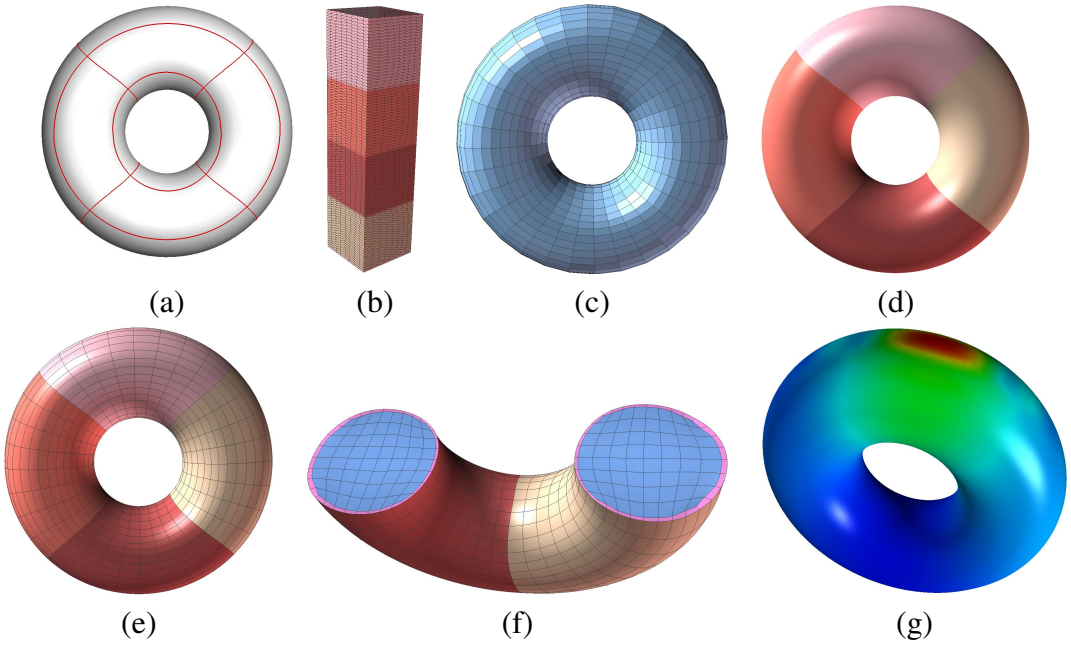


Figure 3.10: Torus model. (a) Splitting result; (b) Boolean operation and parametric mapping result; (c) T-mesh; (d) solid T-spline; (e) solid T-spline with Bézier representation; (f) some elements are removed to show the interior of (e); and (g) analysis result.

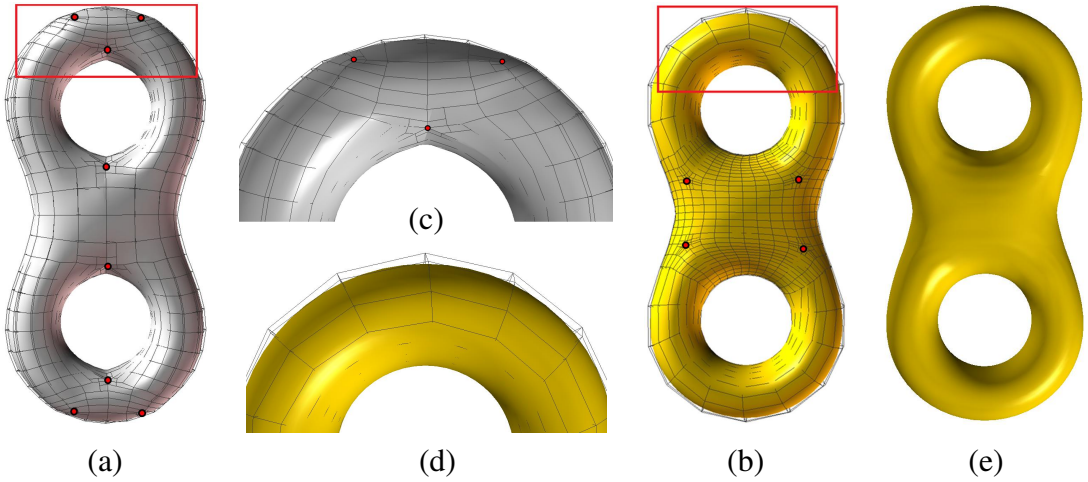


Figure 3.11: Distribution of irregular nodes on the T-spline surface of the Eight model. (a) Polycube method in [90] with details in (c); (b) Boolean operation method with details in (d); and (e) smooth surface with continuity elevation.

3.5 Results and Isogeometric Analysis

We have applied the construction algorithm to several models. Fig. 3.10 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. 3.11-3.12, we compute the harmonic field twice to obtain the desired domain decomposition result. Similar to Fig. 3.3, 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. 3.12(a), we then set two cross sections with the 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. For the middle regions, the isoparametric and gradient lines cannot provide proper decomposition result, so we use the shortest distance method to find the splitting curves. Finally we obtain

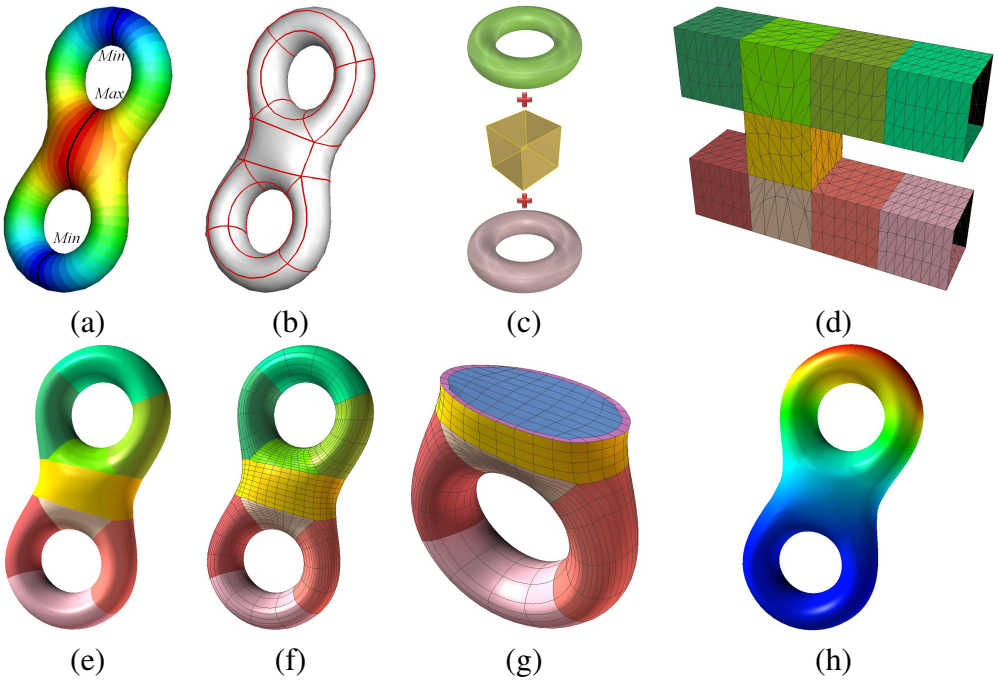


Figure 3.12: Eight model. (a) Temperature field to split the two torus regions; (b) splitting result; (c) Boolean operations; (d) mapping result; (e) solid T-spline; (f) solid T-spline with Bézier representation; (g) elements are removed to show the interior of (f); and (h) analysis result.

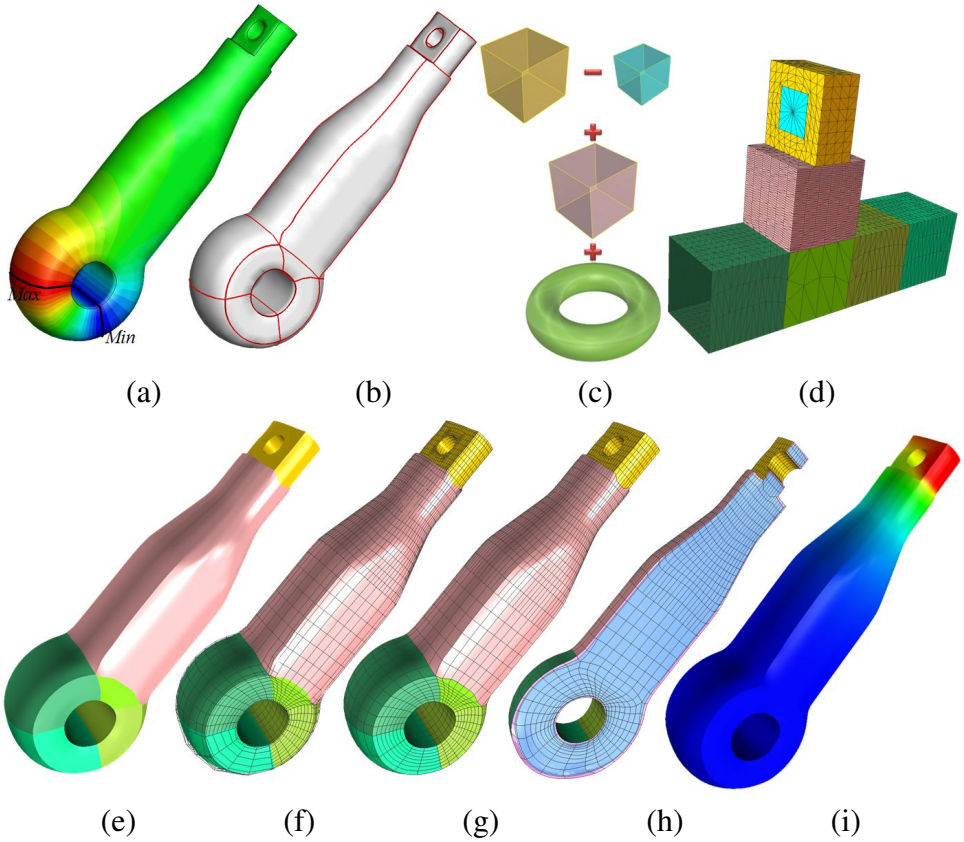


Figure 3.13: Rod model. (a) One temperature field to split the bottom torus region; (b) splitting result;(c) Boolean operations; (d) parametric mapping result (the torus primitive is used in the bottom component, and the difference operation is used to create the small hole in the top component); (e) 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) analysis result.

the splitting result as shown in Fig. 3.12(b). The parametric mapping result, the Boolean operations and the constructed volumetric T-spline model are shown in Fig. 3.12(c-g). We also compared our result with the result from another polycube method [90]. Our method yields fewer number of irregular nodes on the surface (8 vs. 16) with a better min Jacobian (0.31 vs. 0.10).

With the degree elevation technique, the eight model is smooth everywhere on the surface (Fig. 3.11(e)). In the rod model (Fig. 3.13), CAD assembly model (Fig. 3.16), cross-hole model (Fig. 3.14), and joint model (Fig. 3.15), we compute the harmonic field to split the torus or hole region. For the other regions, we trace the shortest distance among the

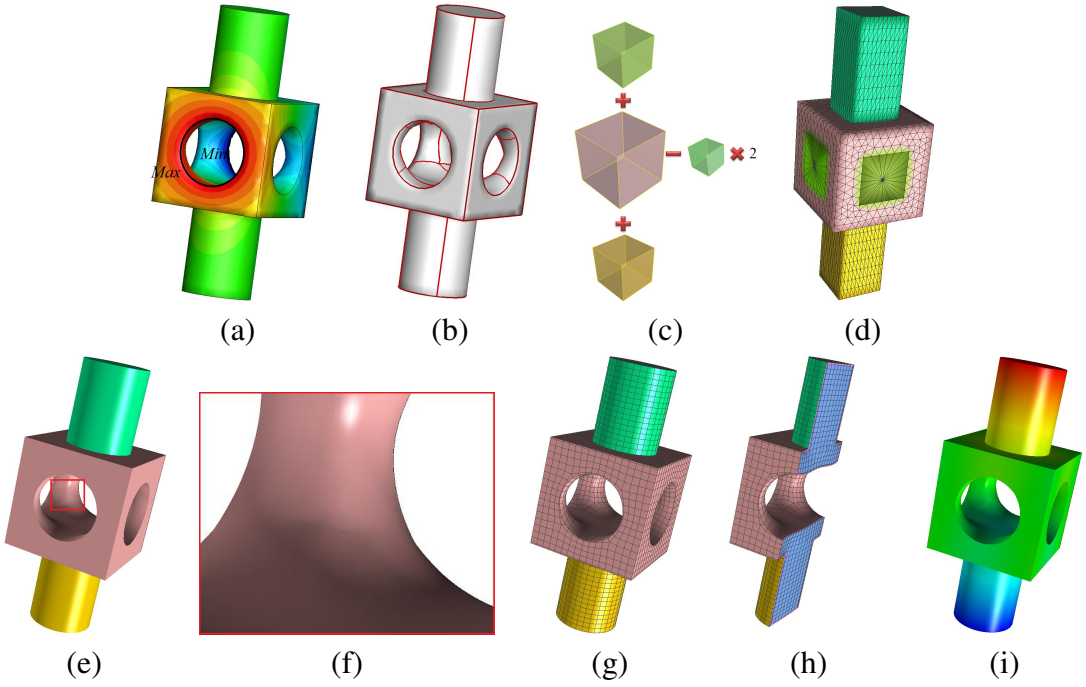


Figure 3.14: Cross-hole model. (a) One temperature field to split the two hole region; (b) splitting result; (c) Boolean operations; (d) parametric mapping result; (e) solid T-spline; (f) zoom in to show the continuity elevation of (e); (g) solid T-spline with Bézier representation; (h) some elements are removed to show the interior of (g); and (i) analysis result.

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 [90]. For the Torus, Eight, CAD assembly, Cross-hole and Joint models, We fix the bottom and apply a displacement load on the top part. Differently for the rod model, we fix the two flat faces of the torus shape region and apply a displacement load on the top of the model, then solve the linear elasticity problem. The analysis results show the displacement distribution along the Z direction. For the CAD assembly model, we also solved the same problem with Abaqus using tetrahedral mesh and finite element method (Fig. 3.16(g)). Comparison shows that our volumetric T-splines can produce results with similar displacement distribution. Our analysis results are very preliminary and comprehensive studies need to take place to verify the analysis suitability of the volumetric models produced by our procedures. Specific issues that need to be addressed are the effect of three-dimensional extraordinary points and

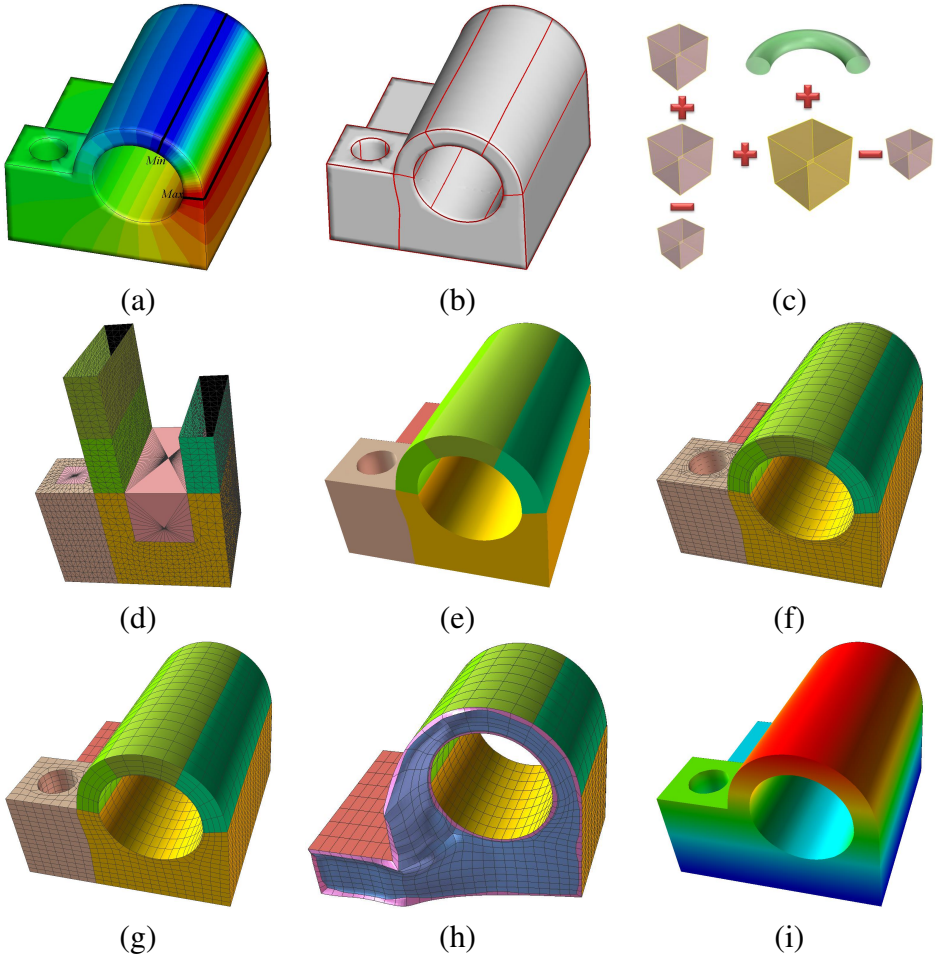


Figure 3.15: Joint model. (a) One temperature field to split the half torus region; (b) splitting result; (c) Boolean operations; (d) parametric mapping result; (e) solid T-spline; (f) solid T-spline and T-mesh; (g) solid T-spline with Bézier representation; (h) some elements are removed to show the interior of (g); and (i) analysis result.

T-junctions on convergence. We note that these issues have not been studied at all in the volumetric case and represent important computational and mathematical problems. We hope to have more to say about them in the future.

3.6 Conclusion

In this chapter we have presented a novel algorithm to use Boolean operations to generate trivariate volumetric T-splines from input models. With proper boundary conditions, a harmonic field is computed to split the input geometry into hexahedral components. In

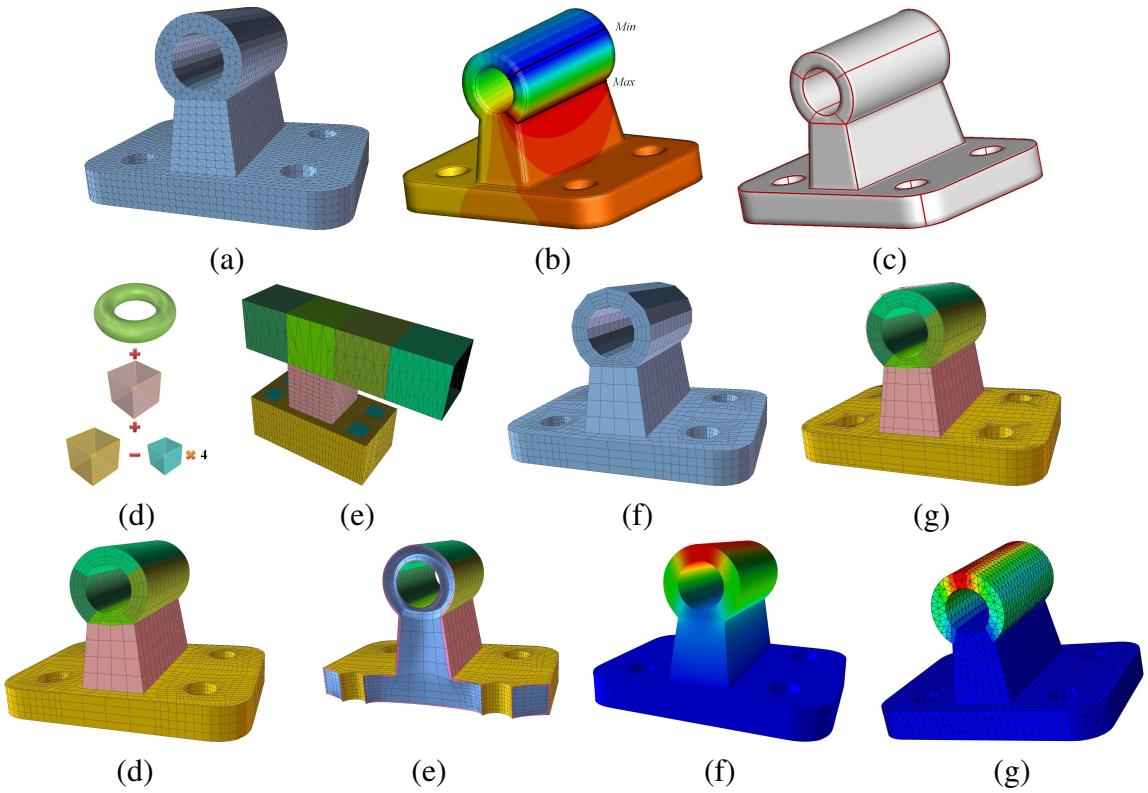


Figure 3.16: CAD Assembly. (a) Surface mesh of the model; (b) temperature field to split the top torus region; (c) splitting result; (d) Boolean operations; (e) mapping result (torus primitive is used for the top component, and difference operation is used to create four holes in the bottom base component); (f) subdivision result; (g) solid T-spline with T-mesh; (h) solid T-spline with Bézier representation; (i) some elements are removed to show the interior of (h); (j) analysis result. To solve the traditional linear elasticity problem, we fix the bottom and apply a displacement load on the top part (Young's modulus: $200GPa$, density: $7900kg/m^3$, Poisson's ratio: 0.3); and (k) Abaqus analysis result.

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 and extract their Bézier representation.

Chapter 4

Skeleton-based T-mesh Construction

This section 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.

4.1 Algorithm Overview

The overview of our algorithm is shown in Fig. 4.1. 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 [86], we split the skeleton into different branches. Each

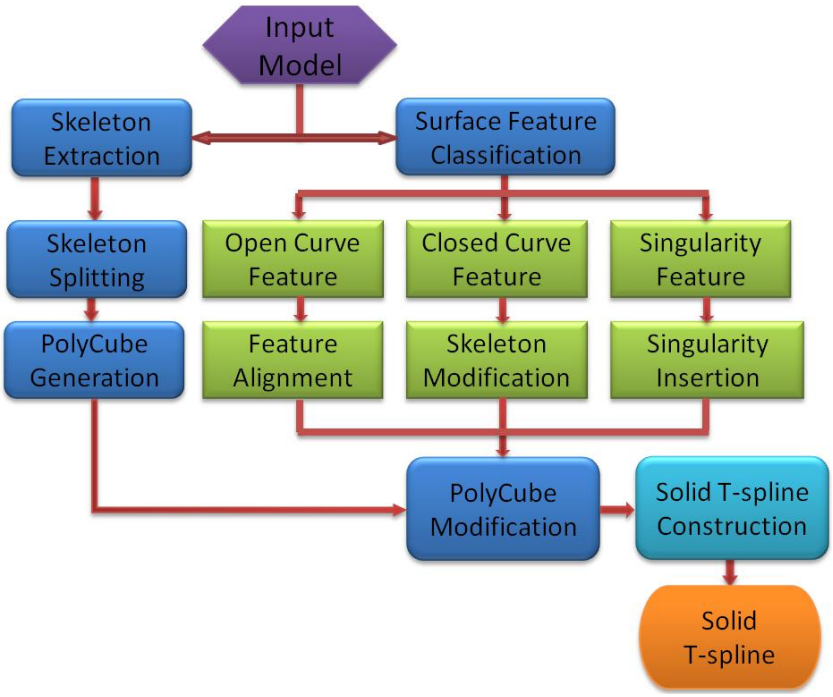


Figure 4.1: Overview of feature preservation in skeleton-based polycube construction and volumetric parameterization.

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.

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 [101, 56].

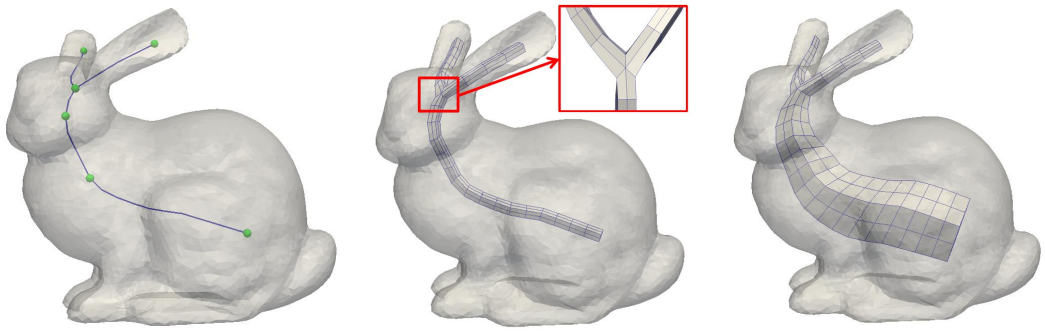
Solid T-spline Construction. From the T-mesh, we build rational solid T-splines [92]. The basis function of rational solid T-splines has the property of partition of unity by definition, which makes it suitable for analysis. Different templates are developed to deal with the singular nodes in the T-mesh, to make it valid for gap-free solid T-spline calculation. From the valid T-mesh, we extract the local knot vectors [85, 91] and construct solid T-splines. The Oslo knot insertion algorithm [34, 91] 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.

4.2 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.

4.2.1 Skeleton Generation and Splitting

There are different algorithms developed to extract skeletons from surface meshes, such as mesh contraction [5], mean curvature flow [86], and the generalized sweeping method [61]. In this paper, we use the algorithm given in [86]. With the skeleton, we first split it into different branches. For each branch, we define a B-spline curve and calculate the tangent



(a) (b) (c)

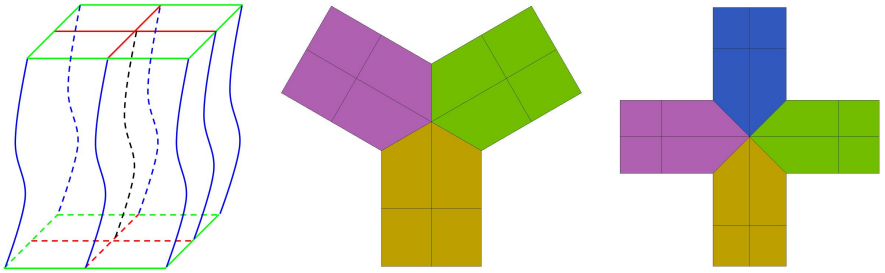
Figure 4.2: Polycube generation for Bunny model. (a) Skeleton splitting results; (b) generating interior cubes by shifting the skeleton branches; and (c) updated interior cubes by iteratively enlarging the cross-sections and smoothing.

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 $\theta_0 = 30°$ is used here. Some user interactions may be involved to simplify the skeleton in this step, such as cleaning up small branches, combining nearby bifurcations to trifurcations, or making the local branching region coplanar. Fig. 4.2(a) shows the extracted skeleton and splitting result of the Bunny model.

4.2.2 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. 4.3(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 [21]. So

for a skeleton, we generate 6 patches from the 20 curves. With these 6 patches, we define a cubic domain.



(a) (b) (c)

Figure 4.3: Construction of interior cubes. (a) Generate an interior cube by shifting the skeleton; (b) use half planes to deal with bifurcation; and (c) trifurcation.

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 [98, 99]. During this process, singularities will be introduced to the polycubes along the shared edges of the half planes. Fig. 4.3(b) and (c) show how to combine the cubes at the bifurcation and trifurcation situations. Instead 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. 4.2(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 [96]. The enlarged and smoothed interior cubes of the Bunny model are shown in Fig. 4.2(c).

4.2.3 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. 4.4 as an example.

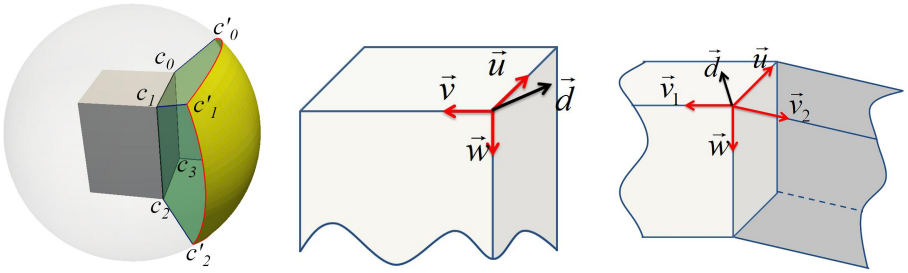
1. **Project corners onto the surface.** For one corner c_i of the interior cube, if shared by one cube, the projection direction is defined as $\overrightarrow{d} = -(\overrightarrow{u} + \overrightarrow{v} + \overrightarrow{w})$ (Fig. 4.4(b)), where $\overrightarrow{u}, \overrightarrow{v}, \overrightarrow{w}$ 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. 4.4 (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. 4.4(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.

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. 4.4(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. 4.4(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.



(a) (b) (c)

Figure 4.4: 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.

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 using BFGS method [101, 36]. Fig.4.5(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

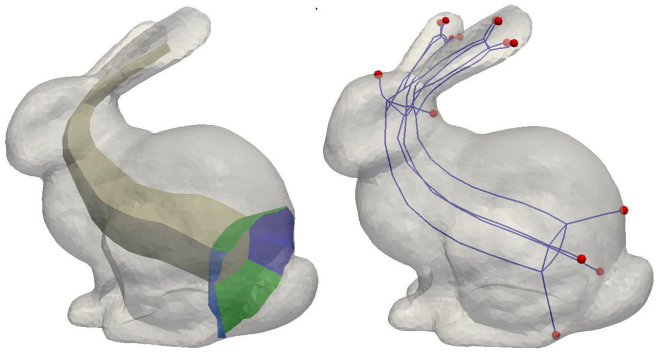
T-mesh follows the topology of the input. If we want to change the orientation or the number of cubes, we can simply modify the skeleton at the beginning. In addition, the location of the projected cube corners on the surface can be optimized to help generate better parameterization results [87].

The projection curve searching in step 2 is crucial to our surface decomposition. Our method works well in general, but some improved surface splitting methods like the greedy strategy [50] can help generate better results for very complicated models.

4.2.4 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[72, 67]. After polycube construction, the singular graph is fixed. We can predict the positions of singular points generated from octree subdivision. Fig.4.5(b) shows the singular graph of the Bunny model.

With the proposed method to generate the interior and boundary polycubes, the singularity graph follows the topology of the geometry. Most of the singularities are embedded



(a) (b)

Figure 4.5: 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).

into the interior except a few singular points on the surface. Since singularities in the solid T-spline are critical to obtain good results in isogeometric analysis, we would like to introduce as few singular points as possible. This method performs well except those CAD models with surface singularities, since no singular edge can be generated on the surface with our method.

4.3 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 [101, 90]. The main difference between our T-mesh generation method and previous research on skeleton-based volumetric composition and structured grid generation [22] 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.

4.3.1 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

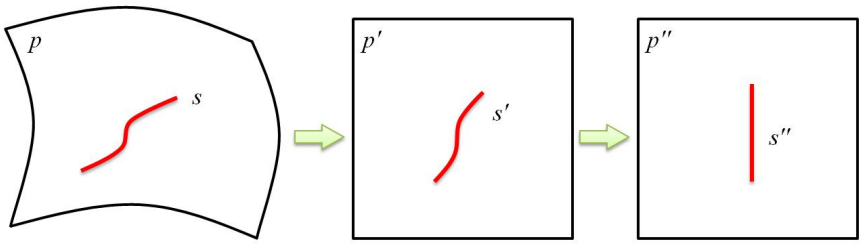


Figure 4.6: Feature alignment during parametric mapping.

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 resulting subdivision between the two cubes. The detailed steps are as follows (Fig. 4.6):

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';

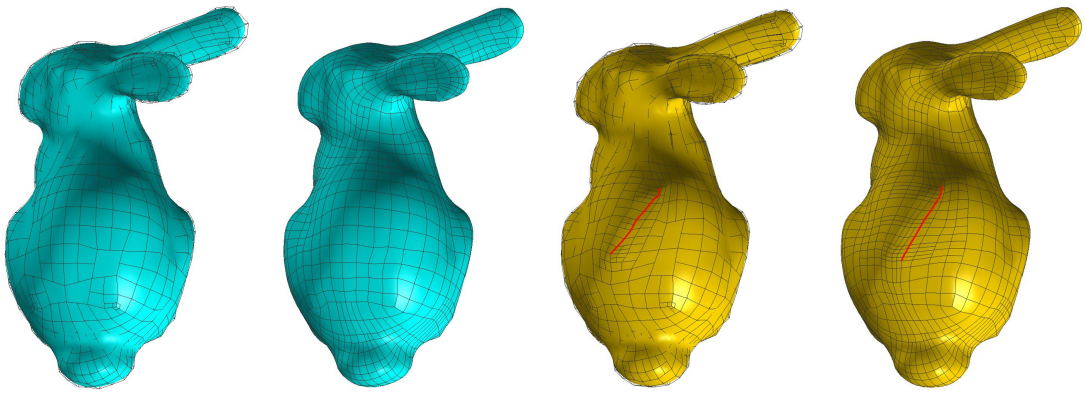
2. Calculate the average angle $\bar{\theta}$ between the tangent direction at each point and the u axis. If $\bar{\theta} < \pi/4$, we align p' to the v direction. Otherwise we align it to the u direction;

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.

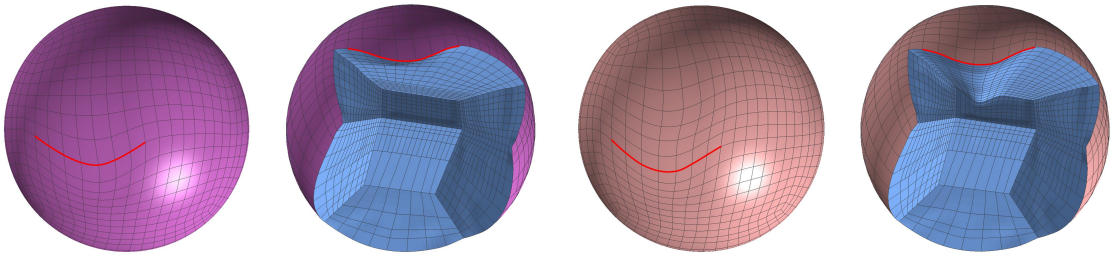
It is convenient to perform the alignment during parametric mapping. However, it is difficult to propagate this feature information into the interior of the T-mesh. This is because

nodes on the surface are calculated from mapping and projection, but nodes in the interior are from a linear interpolation [101]. 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.



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

Figure 4.7: 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.



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

Figure 4.8: 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.

Frame Field. A volume parameterization of geometry $\mathcal{V}$ from frame field can be recognized as an atlas of maps $f\colon \mathcal{V} \to \mathbb{R}^3$, $p \mapsto (u,v,w)^T$. 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 [67] is performed by:

$$min \sum_{t} vol \cdot D_t, \tag{4.1}$$

where

$$D_t = \|c\nabla f(u) - \mathbb{U}_t\|^2 + \|c\nabla f(v) - \mathbb{V}_t\|^2 + \|c\nabla f(w) - \mathbb{W}_t\|^2, \tag{4.2}$$

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 [1] 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 by minimizing one energy function given in [54]. The energy function quantitatively evaluates the smoothness of the frame field between each pair of neighboring tetrahedral elements by calculating the difference along each corresponding parametric direction. The permutation matrix [67] 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.

Fig. 4.7 shows the Bunny model without and with feature alignment. An open curve is preserved on the back of the Bunny. Fig. 4.8 shows the result of a sphere model with an

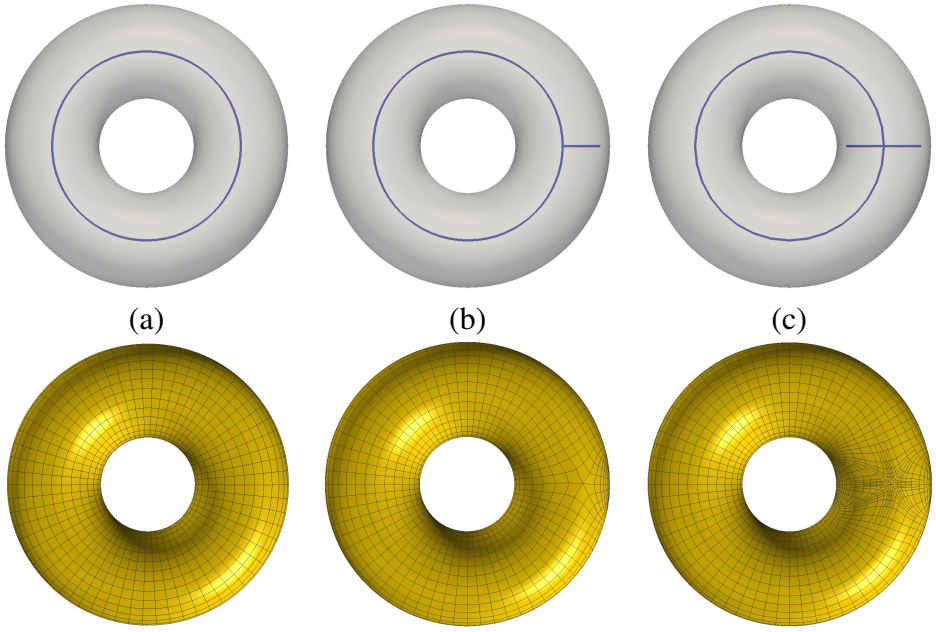


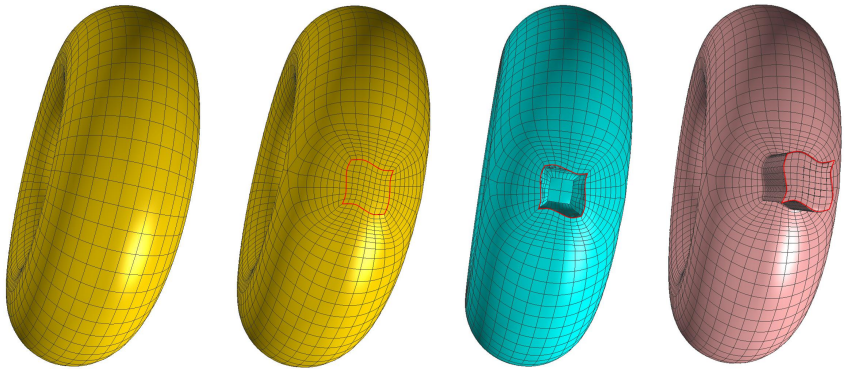
Figure 4.9: Skeleton modification of a torus model. The skeleton is shown in (a), (b) and (c), where (a) shows the original skeleton, (b) shows the skeleton with one new branch inserted to form bifurcation and (c) shows the skeleton with two new branches inserted to form trifurcation; the corresponding T-spline models with Bézier representation are shown in (d), (e) and (f) respectively.

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 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. 4.8(d), the feature curve even influences the subdivision of the interior cube.

4.3.2 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.



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

Figure 4.10: 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.

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. 4.9 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. 4.9(e-f) shows the constructed T-spline models. Fig. 4.10(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. 4.10(c) shows the modified torus model with all the elements in the feature curve region removed. Fig. 4.10(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.

4.3.3 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 4.3.2, modifying the skeleton can change the number of the singular nodes on the surface, but this modification also changes the structure of the polycubes.

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 [67], 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.

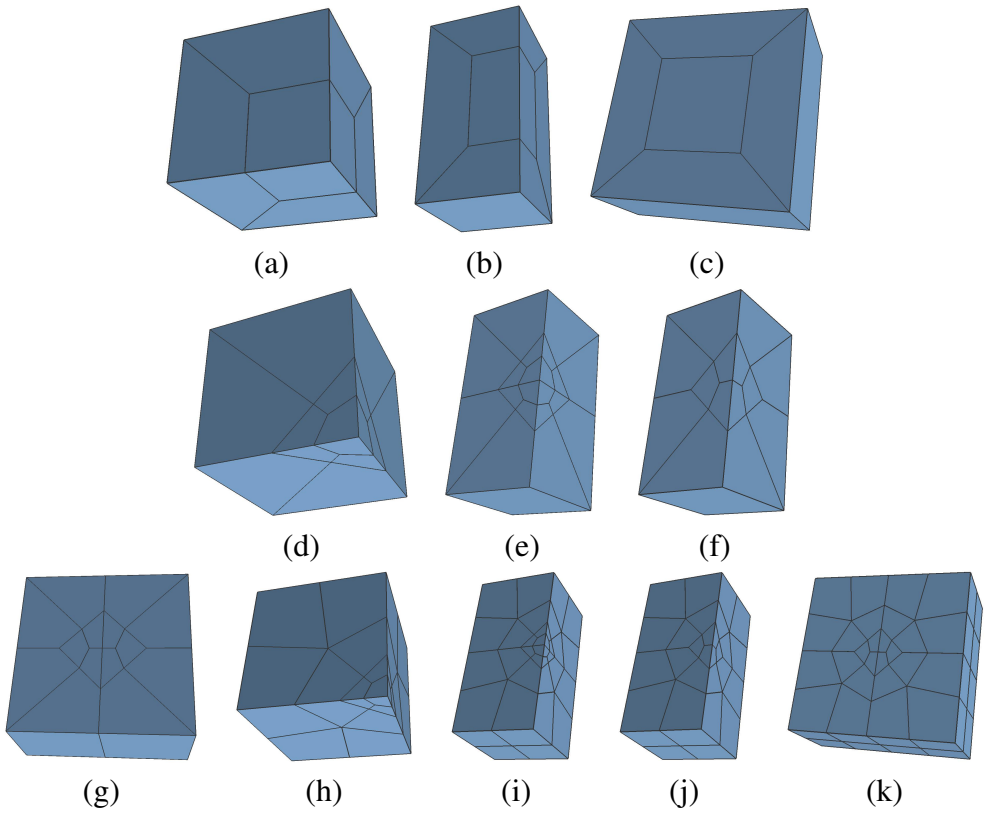


Figure 4.11: Designing three templates by mirroring, combining and simplifying the primitives. (a-c) Template 1; (d-g) Template 2; and (h-k) Template 3.

Template 1 is derived from a 2-refinement splitting primitive of unstructured hex meshes [74]. It introduces three singular nodes on three different faces, see Fig. 4.11(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. 4.11(b). The simplified mesh is combined with its mirror image again with further simplification to get the final template, as shown in Fig. 4.11(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.

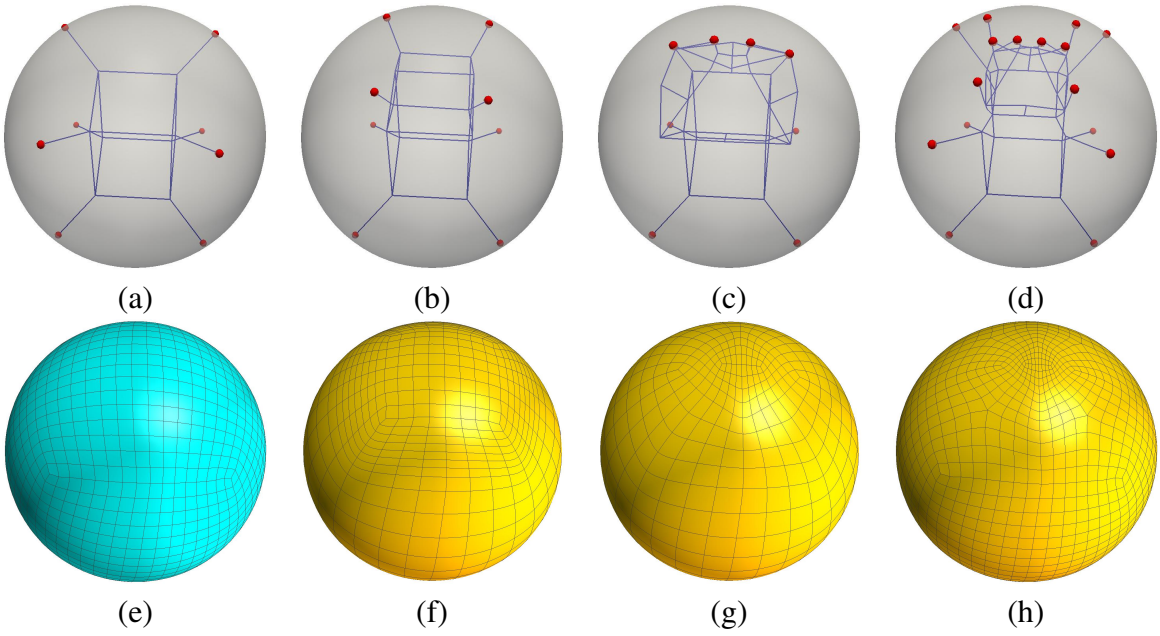


Figure 4.12: New singularity insertion of a sphere model, where (a-d) show the singular graph of the T-mesh, and (e-h) show the T-spline with Bézier element representation. The original sphere model is in (a), Template 1 is applied to (b), Template 2 is applied to (c), and Template 3 is applied to (d).

Template 2 uses two of the 13 meshable primitives introduced in [73, 72]. Fig. 4.11(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. 4.11(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. 4.11(h). As shown in Fig. 4.11(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.

Table 4.1: Statistics of all the tested models.

Model	T-mesh nodes	Irregular nodes (S, I)	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 (F)	12,758	(14, 96)	17,013	(0.20, 0.62, 1.00)	2.30e-2	74.5	OC
Amphora	24,894	(8, 319)	26,686	(0.12, 0.65, 1.00)	1.23e-2	182.3	–
Amphora (F)	34,543	(12, 419)	40,145	(0.12, 0.65, 1.00)	8.7e-3	253.7	CC
Kitten	20,281	(28, 279)	24,502	(0.15, 0.71, 1.00)	1.49e-2	115.3	OC, SG
Rod	21,536	(18, 391)	11,898	(0.35, 0.79, 1.00)	6.5e-3	106.7	OC, CC
Hanger	19,897	(16, 571)	6,564	(0.30, 0.72, 1.00)	5.1e-3	75.6	OC

Note: OC: open curve, CC: closed curve, SG: singularity, F: stands models with preserved feature, S: nodes on the surface, and I: nodes in the interior.

We applied all the three templates to a sphere model to show how they change the singularity points on the surface and the singular graph in the interior. Fig. 4.12 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, 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.

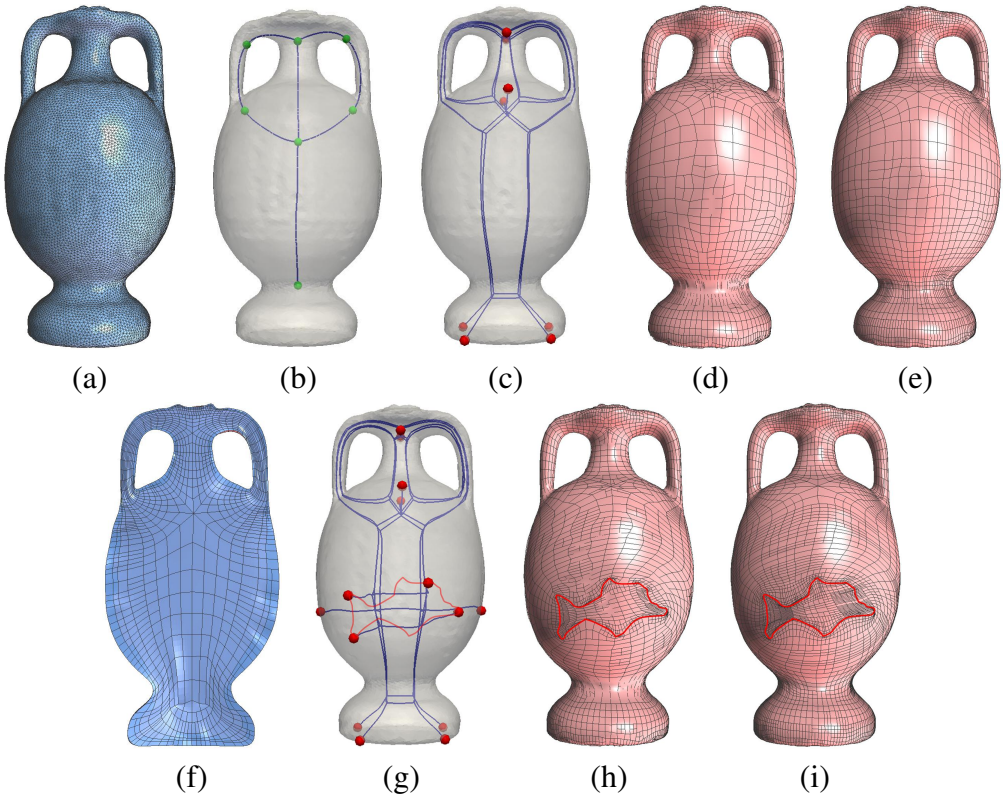


Figure 4.13: Amphora model. (a) Input boundary triangle mesh; (b) skeleton splitting result; (c) singular graph; (d) constructed solid T-spline and T-mesh; (e) solid Bézier elements; and (f) the cross-section. A designed dolphin shape closed curve is preserved and (g) singular graph after skeleton modification; (h) solid T-spline and T-mesh; and (i) Bézier elements.

4.4 Results and Discussion

We have tested our algorithm on different models. 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. 4.13 and 4.16. 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. 4.15 and 4.14.

For the Kitten model in Fig. 4.16, we have preserved detailed feature information, comparing to the result in [90], which uses a harmonic field to split the domain and build

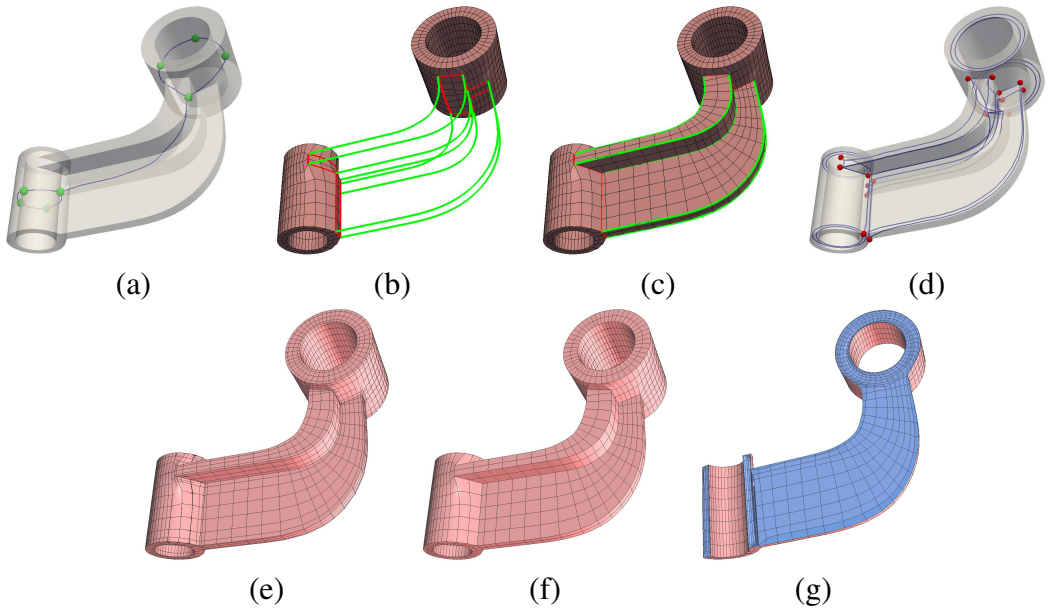


Figure 4.14: Hanger Model. (a) Skeleton splitting result, only the two ring regions are used; (b) the initial T-mesh with open curves (red) preserved, and the extrusion paths (green); (c) using extrusion and union to get the final T-mesh; (d) singular graph of the final T-mesh; (e) the final T-mesh with constructed solid T-spline; (f) the extracted solid Bézier elements; and (g) some Bézier elements are removed to show the cross-section.

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. 4.15, 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 [56]. 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.

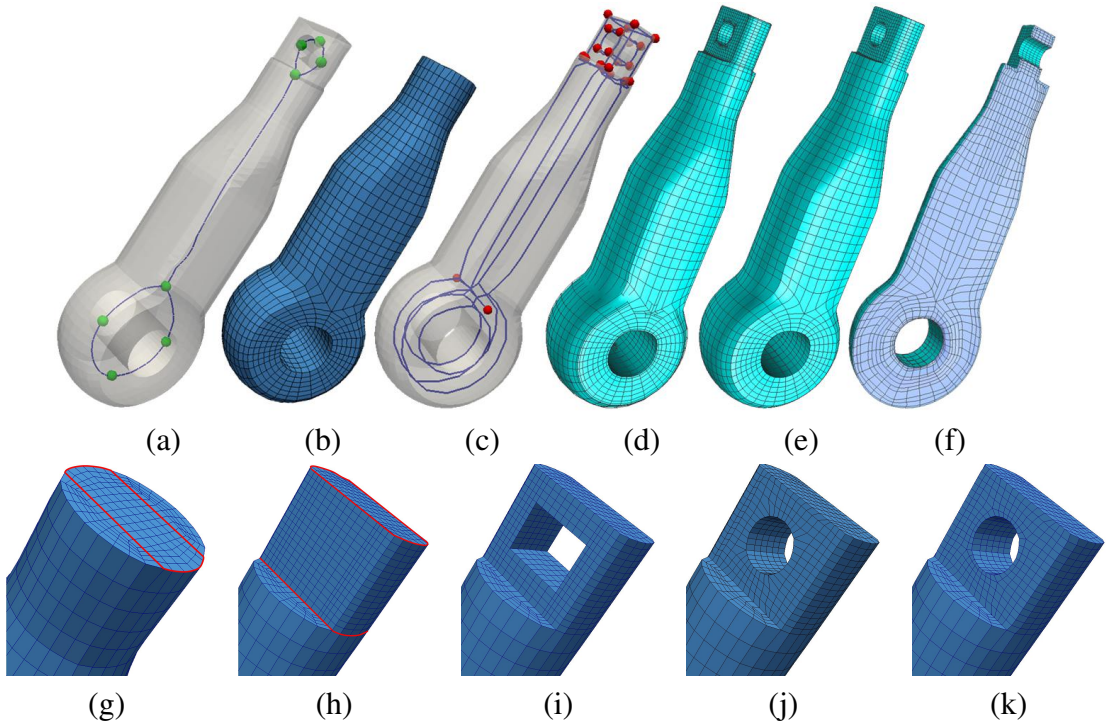


Figure 4.15: Rod model. (a) Skeleton splitting result; (b) initial T-mesh; (c) singular graph of the final T-mesh; (d) final T-mesh with solid T-spline; (e) the extracted solid Bézier elements; and (f) some Bézier elements are removed to show the cross-section. (g-k) show the Boolean operations to the 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 to improve the element quality.

For the Hanger model in Fig. 4.14, 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. 4.14(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.

Tab. 4.1 indicates that the developed algorithm is fast in generating complicated volumetric T-spline models. Only around 100s is needed to generate one model with about

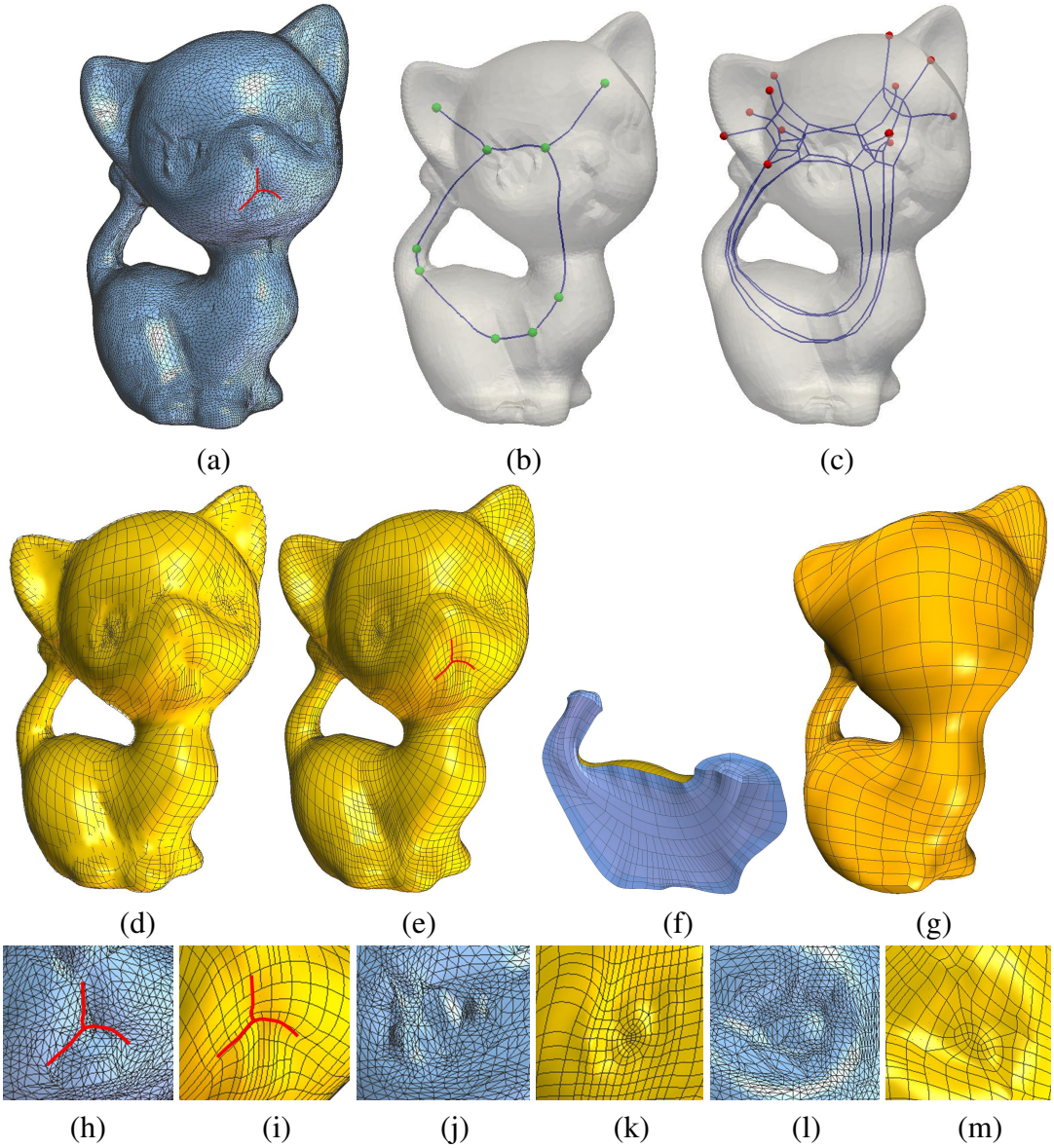


Figure 4.16: Kitten model. (a) The input boundary triangle mesh; (b) skeleton splitting result; (c) singular graph; (d) the constructed solid T-spline and T-mesh; (e) the extracted solid Bézier elements; (f) some Bézier elements are removed to show the cross-section; and (g) the extracted solid Bézier elements from [90]. 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.

20,000 nodes. The Boolean operation algorithm introduced in Chapter 3 has similar speed. But constructing volumetric T-splines within a few seconds for interactive design and modification is not possible yet. Since isogeometric analysis using volumetric T-spline models is more time consuming, volumetric T-spline construction is not the bottleneck for design-through-analysis methodology so far. To improve the modeling speed in the future, we can combine skeleton-based polycube construction with Boolean operations in-depth to construct the models incrementally without reconstructing the whole geometry after design modification.

4.5 Conclusion

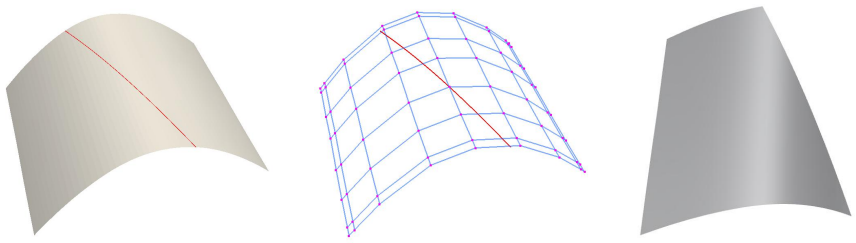
In conclusion, we have developed a new algorithm to build feature-preserving T-meshes using skeleton-based polycube generation. Surface features are classified into open curves, closed curves and singular features. Parametric mapping, volumetric parameterization from frame field, skeleton modification 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 combine medial axis and medial surfaces together to assist polycube construction. Since more geometrical and topological information is contained in medial axis and medial surfaces, the generated polycubes can better represent the geometry, and may introduce fewer singularities in the interior.

Chapter 5

Weighted T-splines

Partition of unity and linearly independence properties are the prerequisites for isogeometric analysis. Only a restricted type of T-splines can be used in analysis, such as a subset of standard T-splines named analysis-suitable T-splines [80]. One way to ensure the constructed T-spline analysis-suitable is to apply topological constraints to the control mesh, that is, the T-junction extensions from different parametric directions cannot intersect with each other [80].

To build T-spline models with less geometrical constraints for isogeometric analysis, in this chapter we present a new type of T-spline, named the weighted T-spline. Weighted T-spline basis functions satisfy partition of unity and are proved to be linearly independent. Interval duplication algorithm is developed for handling extraordinary nodes. Compared with standard T-splines, the weighted T-splines introduce fewer control points for the same level of local refinement with a bounded surface error. With weighted T-spline, we reparameterize the B-reps of CAD models and build volumetric T-spline for analysis. Interval duplication method is developed to handle extraordinary nodes with weighted T-splines. The surface continuity around the extraordinary nodes are increased to GC^1 with optimization method. Sweeping and parametric methods are also developed to generate volumetric weighted T-splines.



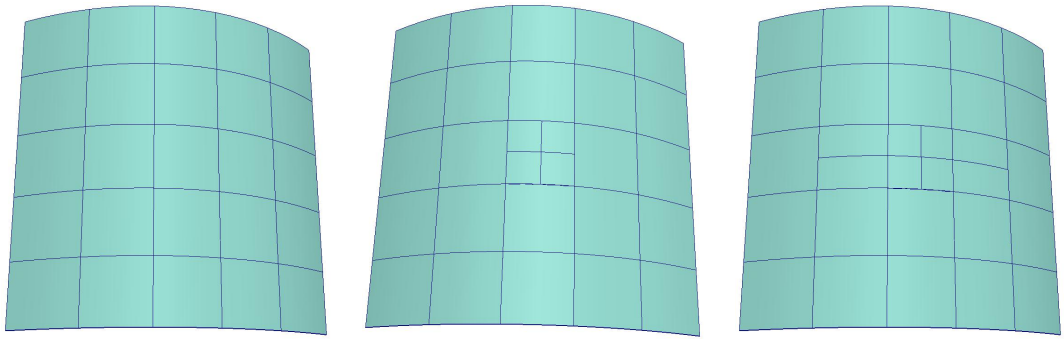
(a) (b) (c)

Figure 5.1: Trimmed NURBS surface. (a) The input NURBS and a trimming curve; (b) the control mesh of (a); and (c) the trimmed NURBS surface.

5.1 Weighted T-splines

T-spline can be used to reconstruct designed NURBS surface patches with local refinement. Quadtree subdivision with constraints is a commonly used refinement method. NURBS satisfy partition of unity and can represent constant during analysis. However, the T-spline resulting from quadtree subdivision may not be suitable for analysis because they cannot always satisfy partition of unity. Fig. 5.2(a) shows one NURBS surface. After subdividing an element in the center, we obtain a refined T-spline surface, as shown in Fig. 5.2(b). T-spline basis functions do not satisfy partition of unity anymore.

To resolve this problem, local refinement of analysis-suitable T-splines was studied in [80]. The topological constraint to the T-mesh configuration was applied to ensure that after

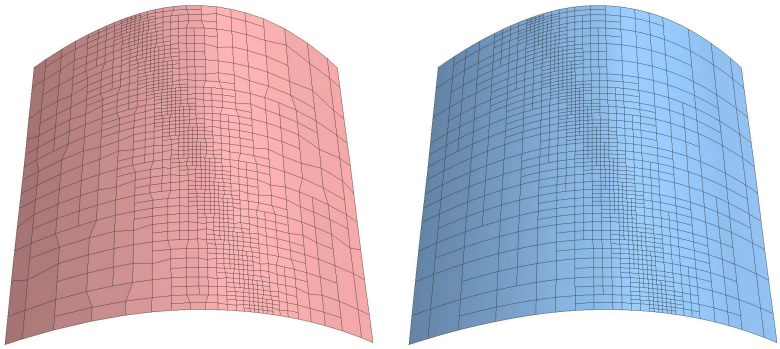


(a) (b) (c)

Figure 5.2: T-spline local refinement by subdividing one element into four smaller ones. (a) The original NURBS surface; (b) the obtained T-spline surface after subdividing one element with T-spline basis functions not satisfying partition of unity; and (c) the obtained standard T-spline surface by applying the topological constraint [80] to the T-mesh.

refinement, the resulting T-spline is a standard one and it can still be used in analysis. A T-junction extension algorithm was also introduced to the locally subdivided T-mesh. Fig. 5.2(c) shows the obtained standard T-spline surface, which preserves the geometry and is analysis-suitable.

Even though the algorithm given in [80] prevents excessive propagation of refinement, there are still a lot of new control points introduced. This becomes more significant when the to-be-refined local region is large, and the refinement level is high. Here the *refinement level* refers to the quadtree subdivision level. For instance, given the NURBS surface in Fig. 5.1(a), we use the trimming curve in Fig. 5.1(b) to decide the to-be-refined region. The T-mesh after four levels of refinement following the refinement algorithm in [80] is shown in Fig. 5.3(a), and the resulting standard T-spline surface is shown in Fig. 5.3(b). We can observe that to meet the topological constraint, some elements far away from the trimming curve also need to be refined.



(a) (b)

Figure 5.3: T-spline generated with four levels of local refinement with the algorithm given in [80]. (a) T-mesh; and (b) the obtained standard T-spline surface.

If we modify the T-spline basis functions defined on the locally subdivided T-mesh (e.g., Fig. 5.2(b)) such that they satisfy partition of unity and are linearly independent, we can release the topological constraints to the T-mesh and introduce fewer control points. To address this issue, in the following we will present a new technique named the weighted T-spline.

5.1.1 Weighted T-spline Basis Functions

The definition of *the weighted T-spline* is based on the concept of T-spline and its refinability property. We calculate and assign a new weight to each children of the identified T-spline basis functions. There are mainly three steps to construct the weighted T-spline.

- **Quadtree Subdivision in Parametric Space (Step 1).** Based on the local refinement constraints, we apply quadtree subdivision in the parametric space and obtain the corresponding T-spline basis functions;
- **Calculation of Weights (Step 2).** Identify the basis functions that do not satisfy partition of unity, and calculate a new weight for each involved children basis function; and
- **Calculation of Control Points in Physical Space (Step 3).** Calculate the control points of the refined T-mesh in physical space by using a transformation matrix and solving linear equations.

Quadtree Subdivision in Parametric Space (Step 1). First, we need to identify to-be-refined regions, which can be defined using trimming curves or the surface error. We compute a group of sampling points on the trimming curve adaptive to its local curvature, and mark elements with more than one sampling point as to-be-refined. Then we perform quadtree subdivision on all to-be-refined elements until no such element exists in the T-mesh. To ensure that the refinement is highly localized, we require that the local refinement should satisfy the following two refinement constraints:

i. The local refinement is based on strongly balanced quadtree subdivision, which means that the refinement level between any two neighboring elements cannot be greater than one; and

ii. For any basis function, the level difference between any pair of its supported elements cannot be greater than two.

After subdividing to-be-refined elements, we check the refinement level of each element and further subdivide those elements that violate the first constraint. Meanwhile, the knot vectors are obtained for each vertex, and their corresponding basis functions are defined. Then we check the second constraint and if it is not satisfied, we subdivide the supported element at the lower level. The two constraints are checked iteratively until they are satisfied everywhere.

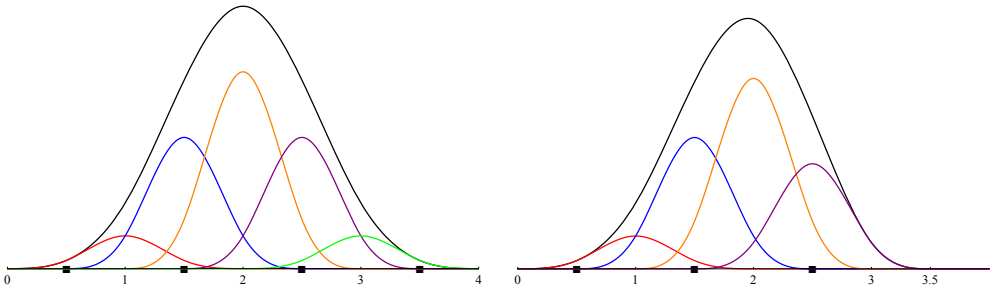
A new set of T-spline basis functions is defined on the refined T-mesh in the parametric domain, which generally does not satisfy partition of unity. However, partition of unity is a prerequisite for both geometric design and analysis. To address this issue, instead of using standard T-splines, we next develop the weighted T-spline basis functions.

Calculation of Weights (Step 2). T-spline basis functions are defined on local knot vectors inferred from the T-mesh [79]. Its refinability property is briefly reviewed here as it is fundamental to our weight calculation. T-spline basis functions inherit the refinability property directly from B-spline basis functions. Consider a local knot vector $\Xi_i = \{\xi_{i,1}, \xi_{i,2}, \xi_{i,3}, \xi_{i,4}, \xi_{i,5}\}$ on which a basis function $N_i(\xi)$ is defined. By inserting k ($k \geq 1$) knots into Ξ_i, a new knot vector $\bar{\Xi}_i$ can be obtained that $\bar{\Xi}_i = \{\bar{\xi}_{i,1}, \bar{\xi}_{i,2}, \ldots, \bar{\xi}_{i,k+5}\}$ where $\bar{\xi}_{i,1} = \xi_{i,1}$ and $\bar{\xi}_{i,k+5} = \xi_{i,5}$. Based on each local knot vector $\{\bar{\xi}_{i,p}, \bar{\xi}_{i,p+1}, \ldots, \bar{\xi}_{i,p+4}\}$ ($1 \leq p \leq k+1$) in $\bar{\Xi}_i$, one *children* basis function $N^c_{i,p}(\xi)$ is defined. Refinability indicates that $N_i(\xi)$ can be represented by a linear combination of $N^c_{i,p}(\xi)$, and we have

$$N_i(\xi) = \sum_{p=1}^{k+1} c_{i,p} N^c_{i,p}(\xi), \tag{5.1}$$

where $c_{i,p}$ ($c_{i,p} > 0$) are the refinement coefficients obtained from knot insertion [34] and $N^c_{i,p}(\xi)$ are called children of $N_i(\xi)$.

The input T-mesh supports non-uniform knot interval configurations. We always bisect non-zero intervals to perform local refinement. An interval of $1/2^n$ of the input is called a *level-n knot interval*. For the refinement of a basis function, if it is defined on uniform-



(a) (b)

Figure 5.4: Children of $N_1(\xi)$ and $N_2(\xi)$. (a) $N_1(\xi)$ (the black curve) and its five children weighted by refinement coefficients (the red, blue, orange, purple and green curves); and (b) $N_2(\xi)$ (the black curve) and its four weighted children (the red, blue, orange and purple curves). The black squares represent the inserted knots for refinement.

level knot intervals, we bisect all the intervals. Otherwise, we refine it by inserting knots to make all the intervals equal to its existing highest level knot interval. For example, the basis function $N_1(\xi)$ (the black curve in Fig. 5.4(a)) is defined on a uniform knot vector $\Xi_1 = \{0,1,2,3,4\}$. By bisecting all the knot intervals of Ξ_1, we insert four knots $\{0.5,1.5,2.5,3.5\}$. According to Eqn. (5.1), we have all the five refined knot vectors with the level-1 knot interval 0.5, where five children basis functions are defined (the red, blue, orange, purple and green curves in Fig. 5.4(a)). The corresponding refinement coefficients are $0.125,0.5,0.75,0.5,0.125$, respectively.

Consider another basis function $N_2(\xi)$ defined on a non-uniform knot vector $\Xi_2 = \{0,1,2,3,3.5\}$ as shown in Fig. 5.4(b). We insert three knots $\{0.5,1.5,2.5\}$ to make all the knot intervals equal to the level-1 knot interval 0.5. We obtain four refined knot vectors, and $N_2(\xi)$ has only four children basis functions; see the red, blue, orange, purple and green curves in Fig. 5.4(b). The refinement coefficients are $0.4,0.725,0.5,0.125$, respectively. Based on the refinability, we develop the weighted basis functions by recalculating a new weight for each children basis function to satisfy partition of unity.

Given the jth basis function $N_j^r(\xi,\eta)$ defined on the locally refined T-mesh, refinability indicates

$$N_j^r(\xi,\eta) = \sum_q c_{j,q}^r N_{j,q}^c(\xi,\eta), \tag{5.2}$$

where $N_{j,q}^c(\xi,\eta)$ is the qth children basis function of $N_j^r(\xi,\eta)$. The weighted T-spline basis function with respect to $N_j^r(\xi,\eta)$ is defined as

$$N_j^w(\xi,\eta) = \sum_q h_{j,q} N_{j,q}^c(\xi,\eta), \tag{5.3}$$

where $h_{j,q}$ is the *weighting coefficient*. Different from the truncation operations in [32, 33, 94], which directly discard the identified children basis functions, weighted T-spline basis functions calculate a new weight for each children basis function. For all $h_{j,q}$, we have $0 < h_{j,q} \leq c_{j,q}$.

Next we discuss how to calculate the weighting coefficient $h_{j,q}$. Suppose T is the T-mesh with basis functions satisfying partition of unity. T^{r} is the T-mesh obtained by locally refining T. According to partition of unity and refinability, we have

$$\sum_{i=1}^m N_i(\xi,\eta) = \sum_{i=1}^m \sum_p c_{i,p} N_{i,p}^c(\xi,\eta) = 1, \tag{5.4}$$

where $N_i(\xi,\eta)$ is a basis function defined on T, $N_{i,p}^c(\xi,\eta)$ is its pth children basis function, and m is the number of basis functions on T. For T^{r}, there are n ($n > m$) basis functions $N_j^r(\xi,\eta)$ defined on it and we have

$$\sum_{j=1}^n N_j^r(\xi,\eta) = \sum_{j=1}^n \sum_q c_{j,q}^r N_{j,q}^c(\xi,\eta), \tag{5.5}$$

where $N_{j,q}^c(\xi,\eta)$ is the qth children basis function of $N_j^r(\xi,\eta)$. Note that partition of unity is not satisfied in general.

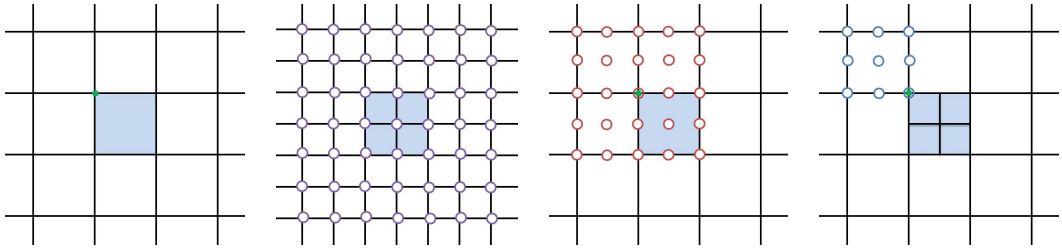


Figure 5.5: Children basis support before and after refinement. (a) T-mesh with level-ℓ ($\ell \geq 0$) knot intervals; (b) the same domain with level-$(\ell+1)$ knot intervals. Purple circles represent the basis functions defined in this local domain; (c) the green knot has 25 children basis functions (red circles); and (d) after subdividing the blue element, the green knot has only 9 children basis functions (blue circles).

$N^c_{i,p}(\xi,\eta)$ and $N^c_{j,q}(\xi,\eta)$ are defined on the uniform knot vectors with the same knot interval (without considering the boundary), therefore the set of all $N^c_{i,p}(\xi,\eta)$ equals to the set of all $N^c_{j,q}(\xi,\eta)$, denoted as $\mathcal{N}^c$. For example, Fig. 5.5(a) represents a T-mesh with the level-ℓ ($\ell \geq 0$) knot intervals and Fig. 5.5(b) shows the same domain with the level-$(\ell+1)$ knot intervals. Suppose $N_1(\xi,\eta)$ is the basis function associated with the green knot in Fig. 5.5(c). $N_1(\xi,\eta)$ has 25 children basis functions defined on the uniform level-$(\ell+1)$ knot intervals; see the 25 red circles. After subdividing the blue element, the local knot vector of the green knot is changed, so $N_1(\xi,\eta)$ changes to $N^r_1(\xi,\eta)$. $N^r_1(\xi,\eta)$ has only 9 children basis functions defined on the uniform level-$(\ell+1)$ knot intervals (blue circles in Fig. 5.5(d)). Therefore, we can express $N_i(\xi,\eta)$ as

$$N_i(\xi,\eta) = \sum_{k=1}^{n_c} R_{i,k} N^c_k(\xi,\eta), \tag{5.6}$$

where $N^c_k \in \mathcal{N}^c$, and n_c is the dimension of $\mathcal{N}^c$. We have

$$R_{i,k} = \begin{cases} c_{i,p} & \text{if } N^c_k(\xi,\eta) = N^c_{i,p}(\xi,\eta); \\ 0 & \text{otherwise.} \end{cases} \tag{5.7}$$

Similarly for $N_j^r(\xi,\eta)$, we have

$$N_j^r(\xi,\eta) = \sum_{k=1}^{n_c} R_{j,k}^r N_k^c(\xi,\eta), \tag{5.8}$$

where

$$R_{j,k}^r = \begin{cases} c_{j,q}^r & \text{if } N_k^c(\xi,\eta) = N_{j,q}^c(\xi,\eta); \\ 0 & \text{otherwise.} \end{cases} \tag{5.9}$$

Let

$$h_{j,q} = s_k R_{j,k}^r, \tag{5.10}$$

where

$$s_k = \frac{\sum_{i=1}^{m} R_{i,k}}{\sum_{j=1}^{n} R_{j,k}^r}. \tag{5.11}$$

According to Eqns. (5.3), (5.9) and (5.10), the weighted basis function $N_j^w(\xi,\eta)$ with respect to $N_j^r(\xi,\eta)$ is expressed as

$$N_j^w(\xi,\eta) = \sum_q h_{j,q} N_{j,q}^c(\xi,\eta) = \sum_{k=1}^{n_c} s_k R_{j,k}^r N_k^c(\xi,\eta). \tag{5.12}$$

From Eqns. (5.6) and (5.12), we can obtain

$$\begin{aligned} \sum_{j=1}^{n} N_j^w(\xi,\eta) - \sum_{i=1}^{m} N_i(\xi,\eta) &= \sum_{j=1}^{n}\sum_{k=1}^{n_c} s_k R_{j,k}^r N_k^c(\xi,\eta) - \sum_{i=1}^{m}\sum_{k=1}^{n_c} R_{i,k} N_k^c(\xi,\eta) \\ &= \sum_{k=1}^{n_c} N_k^c(\xi,\eta)\left(s_k \sum_{j=1}^{n} R_{j,k}^r - \sum_{i=1}^{m} R_{i,k} \right) \\ &= 0. \end{aligned} \tag{5.13}$$

Therefore $\sum_{j=1}^{n} N_j^w(\xi,\eta) = \sum_{i=1}^{m} N_i(\xi,\eta) = 1$, and thus partition of unity is satisfied for the weighted T-spline basis functions.

Example 3.1. Fig. 5.6(a) shows a local domain of T-mesh T where $N_i(\xi,\eta)$ are defined. After subdividing one element, we get the T-mesh T^r in Fig. 5.6(b), where $N^r_j(\xi,\eta)$ are defined. Note that in this local domain, T only has level-ℓ knot intervals and T^r has both level-ℓ and level-$(\ell+1)$ knot intervals.

Fig. 5.6(c) shows the same domain with the level-$(\ell+1)$ knot intervals everywhere and $N^c_k(\xi,\eta)$ are defined on it. Here we take the $N^r_6(\xi,\eta)$ as an example to show how to develop its weighted basis function $N^w_6(\xi,\eta)$. $N^r_6(\xi,\eta)$ has 9 children basis function $N^c_k(\xi,\eta)$, where $k = 1,2,\ldots,9$ (shown in Fig. 5.6(d)). In T, $N_i(\xi,\eta)$ $(i = 1,2,\ldots,16)$ has children basis functions $N^c_k(\xi,\eta)$. We have $\sum_{i=1}^{m} R_{i,k} = \sum_{i=1}^{16} R_{i,k} = 1$ since $R_{i,k} = 0$ for $i = 17,\ldots,m$. We can obtain $\sum_{j=1}^{n} R^r_{j,k}$ from Eqn. (5.9) and the corresponding s_k from Eqn. (5.11). The coefficients $R^r_{6,k}$ of $N^c_k(\xi,\eta)$ are shown in Fig. 5.6(e). We can obtain the weighted coefficients $h_{j,q}$ by Eqn. (5.10), shown in Fig. 5.6(f). $N^r_6(\xi,\eta)$ and $N^w_6(\xi,\eta)$ are plotted in Fig. 5.6(g, h). For reference, see Tab. 1 for the values of $\sum_{i=1}^{m} R_{i,k}$, $\sum_{j=1}^{n} R^r_{j,k}$, s_k, $R^r_{6,k}$ and $h_{j,q}$ $(=s_k \cdot R^r_{6,k})$, where $k = 1,2,\ldots,9$.

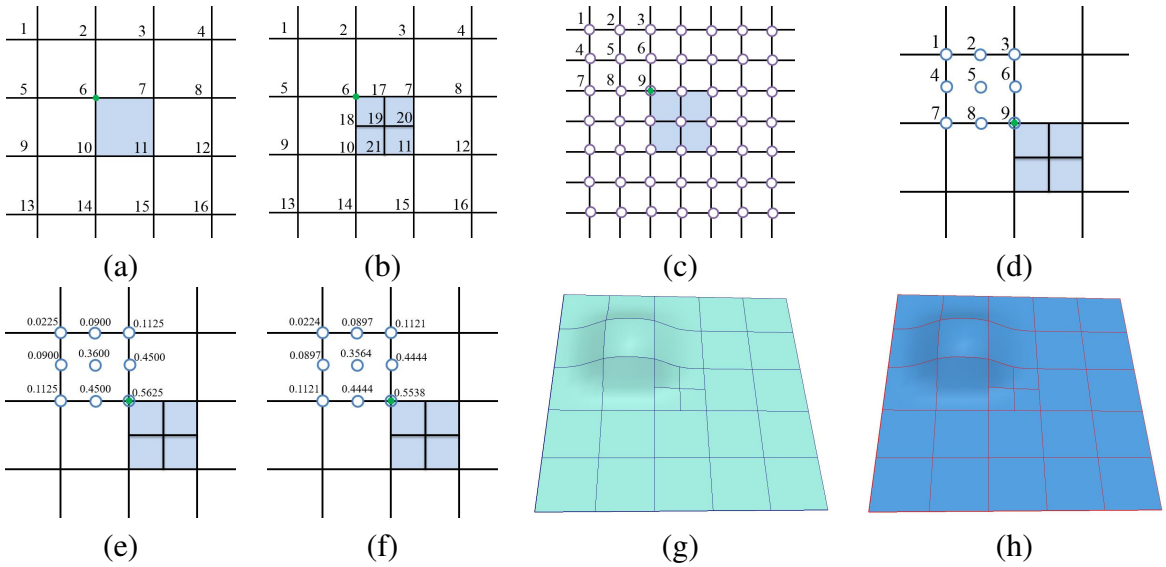


Figure 5.6: (a) A local domain with the indexing of $N_i(\xi,\eta)$; (b) the same domain after subdividing the blue element with the indexing of $N^r_j(\xi,\eta)$; (c) one level higher knot intervals and partial indexing of $N^c_k(\xi,\eta)$; (d) children basis functions with the indexing of $N^r_6(\xi,\eta)$; (e) 9 refinement coefficients of $N^r_6(\xi,\eta)$; (f) 9 weighted coefficients of $N^w_6(\xi,\eta)$; (g) shape of $N^r_6(\xi,\eta)$; and (h) shape of $N^w_6(\xi,\eta)$.

Table 5.1: The calculated $\sum_{i=1}^{m} R_{i,k}$, $\sum_{j=1}^{n} R^r_{j,k}$, s_k, $R^r_{6,k}$ and $h_{j,q}$ in Example 3.1.

k	$\sum_{i=1}^{m} R_{i,k}$	$\sum_{j=1}^{n} R^r_{j,k}$	s_k	$R^r_{6,k}$	$h_{j,q}$ $(=s_k \cdot R^r_{6,k})$
1	1.0	1.000625000	0.999375390	0.022500000	0.022485946
2	1.0	1.002500000	0.997506234	0.090000000	0.089775561
3	1.0	1.003125000	0.996884735	0.112500000	0.112149532
4	1.0	1.002500000	0.997506234	0.090000000	0.089775561
5	1.0	1.010000000	0.990099009	0.360000000	0.356435643
6	1.0	1.012500000	0.987654320	0.450000000	0.444444444
7	1.0	1.003125000	0.996884735	0.112500000	0.112149532
8	1.0	1.012500000	0.987654320	0.450000000	0.444444444
9	1.0	1.015625000	0.984615384	0.562500000	0.553846153

Remark 3.1. Since $N_i(\xi,\eta)$ satisfy partition of unity, we have $\sum_{i=1}^{m} R_{i,k} = 1$ and $s_k = (\sum_{j=1}^{n} R^r_{j,k})^{-1}$. When $s_k = 1$, we have $h_{j,q} = c^r_{j,q}$. In the second constraint of refinement, for any basis function we require the level difference between any pair of its supported elements is at most two. Therefore, the knot interval level difference of $N^r_j(\xi,\eta)$ should also be at most two. In this way the refinement is highly localized. In each refinement, we only need to compute the weighted basis functions near T-junctions.

Calculation of Control Points in Physical Space (Step 3). We first calculate the control points $\mathbf{P}^r_j$ associated with each refined basis function $N^r_j(\xi,\eta)$ using the knot insertion algorithm. Then for each T-mesh element, we identify all the T-spline basis functions with support on it and check the partition of unity property. Each refined basis function $N^r_j(\xi,\eta)$ violating this property is identified and replaced by its corresponding weighted basis function $N^w_j(\xi,\eta)$. For these weighted basis functions, we calculate the associated control points separately. Denoting I^w as the index set of the weighted T-spline basis functions, we enforce that

$$\sum_{i\in I^w} \mathbf{P}^w_i N^w_i(\xi_j,\eta_j) = \mathbf{S}^0(\xi_j,\eta_j) \tag{5.14}$$

holds for all $j \in I^w$, where $\mathbf{P}_i^w$ are the unknown control points, (ξ_j, η_j) are the parametric coordinates, and $\mathbf{S}^0$ is the input surface. We can obtain $\mathbf{P}_i^w$ by solving a linear system constructed from Eqn. (5.14).

Taking the trimming curve in Fig. 5.1(b) as a guidance, we locally refine the input NURBS surface. Fig. 5.7(a-d) shows the T-meshes of Fig. 5.1(b) with refinement levels from one to four. The corresponding weighted T-spline surfaces are shown in Fig. 5.7(e-h). For the fourth level of refinement, compared to the standard T-splines in Fig. 5.3, the number of control points is decreased from 1356 to 1016 by 25%, and the number of elements is decreased from 1142 to 813 by 29%. We can observe that the weighted T-splines have the following two obvious advantages in contrast with standard T-splines: (i) less topological constraint is applied to the local refinement of the T-mesh and T-junction extension is not necessary; and (ii) fewer control points are required for the same level of refinement.

Remark 3.2. Like hierarchical B-splines and T-splines, the weighted T-splines also support local refinement. In (truncated) hierarchical B-splines and T-splines, higher level control points have influence on a smaller local region, and in order to satisfy partition of unity, some children basis functions are discarded in reconstructing a basis function. Differently, the weighted T-splines involve a single level of T-mesh only where all control points have similar influence, and partition of unity is guaranteed by computing a new weight for each basis function instead of discarding children basis functions.

5.1.2 Linear Independence of Weighted T-spline Basis Functions

The basis functions need to be linearly independent for analysis. We use the following three propositions to prove that the weighted T-spline basis functions defined on the locally subdivided T-mesh are linearly independent.

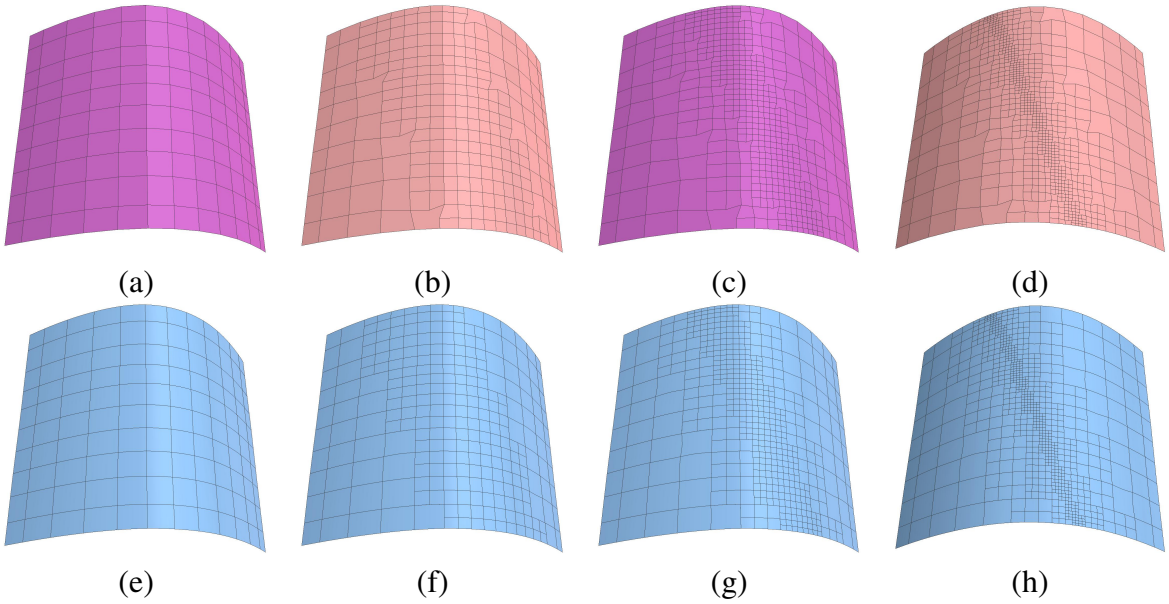


Figure 5.7: Results of four levels of refinement. (a-d) T-meshes with different refinement levels from 1 to 4; and (e-h) the corresponding weighted T-spline surfaces.

Proposition 5.1.1 *The necessary and sufficient condition for T-spline basis functions to be linearly independent is that the T-spline-to-NURBS transformation matrix* **M** *is in full rank.*

This proposition can be easily obtained from by Theorem 1 in [53]. That is, the linear independence property of T-spline basis functions can be easily determined by checking the rank of **M**.

Proposition 5.1.2 *T-spline basis functions defined on the locally-refined T-mesh via quadtree subdivision with the two constraints in Section 5.1.1 are linearly independent, provided that the initial T-mesh is linearly independent.*

proof 5.1.3 *Refining a T-mesh element with quadtree subdivision is equivalent to adding one vertical line first and then two short horizontal lines. As proved in [16], by inserting a new vertical parametric line to a linearly independent T-mesh, the T-spline basis functions are still linearly independent on the refined T-mesh. Similarly, the T-mesh remains linearly independent by inserting a new horizontal parametric line. Therefore, the T-spline basis functions are linearly independent after quadtree subdivision.*

Proposition 5.1.4 *Let* $\mathbf{N}^r$ *denote the T-spline basis functions defined on the refined T-mesh, and* $\mathbf{N}^w$ *the weighted T-spline basis functions. If* $\mathbf{N}^r$ *is linearly independent, then* $\mathbf{N}^w$ *is also linearly independent.*

proof 5.1.5 *From Eqn.* (5.8)*, we have*

$$\mathbf{N}^r = \mathbf{M}^r \mathbf{N}^c, \tag{5.15}$$

in the matrix form, where $\mathbf{M}^r$ *is the transformation matrix with the jth row, the kth column entry* $R^r_{j,k}$*. As* $\mathbf{N}^r$ *is linearly independent and* $\mathbf{N}^c$ *represents a vector of NURBS basis functions. According to Proposition 5.1.1,* $\mathbf{M}^r$ *is in full rank. Eqn.* (5.12) *can also be expressed in the matrix form*

$$\mathbf{N}^w = \mathbf{S}\mathbf{M}^r \mathbf{N}^c, \tag{5.16}$$

where $\mathbf{S} = diag(s_1, s_2, \ldots, s_{n_c})$ *(*$s_k > 0$*). Considering that* $\mathbf{S}$ *is a full rank diagonal matrix, and* $\mathbf{M}^r$ *is in full rank, therefore* $\mathbf{S}\mathbf{M}^r$ *is also in full rank. According to Proposition 5.1.1,* $\mathbf{N}^w$ *is linearly independent, and the weighted T-spline basis functions on locally refined T-mesh are linearly independent.*

With weighted T-spline basis functions, we can enforce partition of unity without modifying either the topology or the linear independence of the T-mesh. Thus, more flexibility is given to the T-mesh construction compared to standard T-splines.

5.2 Reparameterization of Trimmed NURBS Patches

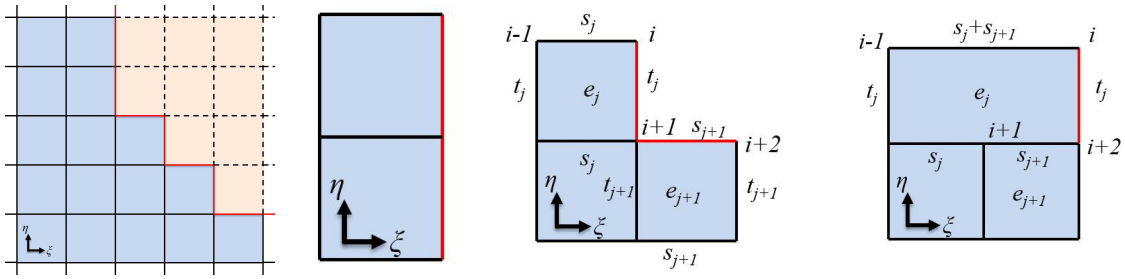
Boolean operations introduce trimming curves to the NURBS surfaces, which make the B-reps (boundary-representation) of designed models not watertight. Trimming curves are important surface features that need to be preserved during reparameterization. Watertight NURBS patches were studied in [84], where redundant control points were introduced to reconstruct the intersection curves and merge different NURBS patches together. Taking

advantage of the local refinement property, we can use T-splines to reparameterize the trimmed NURBS patches.

The input trimming curve is used to determine the region that needs local refinement. For example for the trimmed NURBS patch in Fig. 5.1(c), we use the trimming curve and the original NURBS patch (Fig. 5.1(a)) to reparameterize it. We compute n sampling points on the trimming curve. We take the input NURBS control mesh as our initial T-mesh, and refine it until in the parametric domain, there are no two sampling points falling inside the same element. After the refinement, we obtain the final T-mesh shown in Fig. 5.7(a-d).

5.2.1 Edge Interval Extension

With the refined T-mesh, the trimming curve determines the preserved and removed elements. The preserved elements are used to reconstruct the trimmed surface. By connecting all the edges shared by the preserved and removed elements, we obtain the initial control polygon for the trimming curve. In Fig. 5.8(a), the removed elements are marked in yellow, the preserved elements are marked in blue, and the red edges are the initial control polygon.

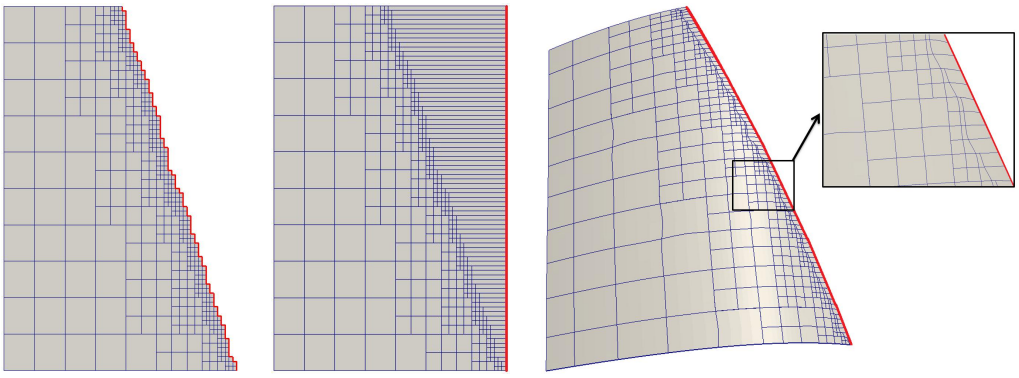


(a) (b) (c) (d)

Figure 5.8: Edge interval extension for trimming curve reconstruction. (a) Preserved elements (blue) and removed elements (yellow) determined by the input trimming curve, and the red edges represent the initial control polygon of the trimming curve; (b) the first configuration of preserved elements which does not need a configuration modification; (c) the second configuration of preserved elements that needs a configuration modification, where (e_j, e_{j+1}) are two involved elements, $(i-1, i, i+1, i+2)$ are four corner indices, and $(s_j, s_{j+1}, t_j, t_{j+1})$ are four edge intervals; and (d) the connectivity modification result of (c).

Edge Interval Extension. The input trimming curve is the boundary of the trimmed surface. So the trimming curve needs to be defined on the boundary of the parametric domain. Directly throwing away the removed elements results in zigzag lines in the parametric domain (see Fig. 5.9(a)), which follow two parametric directions. However, a boundary requires only one parametric direction, so we have to change the connectivity of the initial control polygon.

There are two configurations of preserved elements containing the boundary edges, as shown in Fig. 5.8(b-c), where the red lines represent the boundary edges. Suppose the boundary is defined along the η direction. For the configuration in Fig. 5.8(b), the edges already follow the η direction, so we do not need to modify its connectivity. For the configuration in Fig. 5.8(c), $(s_j,\, t_j)$ are the edge intervals of element e_j, and $(s_{j+1},\, t_{j+1})$ are the edge intervals of element e_{j+1}. We delete boundary edges $(i, i+1)$, $(i+1, i+2)$ and connect $(i,\, i+2)$ as a new boundary edge. Knot $i+2$ is set as the new corner of element e_j, and knot $i+1$ is added as one T-junction to e_j. A valid T-mesh requires that the sum of knot intervals on opposing edges of any element must be the same [85]. Therefore, we also change the knot interval of edge $(i-1, i)$ to $(s_j + s_{j+1})$. The knot interval of edge $(i, i+2)$ is t_j. The connectivity modification result is shown in Fig. 5.8(d).



(a) (b) (c)

Figure 5.9: Reparameterization of the trimmed surface for the T-mesh given in Fig. 5.7(d). (a) Deleting removed elements; (b) modifying the topology of preserved elements and extending edge intervals to align with the boundary of the rectangular parametric domain; and (c) reparameterized trimmed surface using the weighted T-spline.

We go through the boundary edges, modify the connectivity for the configuration in Fig. 5.8(c), and then obtain the updated control polygon for the trimming curve. The trimming curve is now defined on the boundary of the rectangular parametric domain. For example, after applying edge interval extension to Fig. 5.9(a), we obtain one rectangular parametric domain with the right boundary (red) corresponding to the trimming curve; see Fig. 5.9(b). To exactly reconstruct the trimming curve, we need to recalculate their coordinates by applying the knot insertion algorithm to the input trimming curve. The reparametrized weighted T-spline surface of the trimmed surface is shown in Fig. 5.9(c).

Degenerated T-mesh Element. When three corners are trimmed off from the original NURBS surface, there will be one degenerated T-mesh element generated, where two control points have the same coordinates. Fig. 5.10(a) shows the preserved elements in the parametric domain, and the element marked in green corresponds to the degenerated element. The rectangular parametric domain obtained from edge interval extension is shown in Fig. 5.10(b). The green element can be mapped onto the green region on the T-spline surface in Fig. 5.10(c). Although two control points of the degenerated element have the same coordinates, we can still compute the weighted T-spline basis functions and use them in analysis. The Jacobian at its Gaussian quadrature points are all positive.

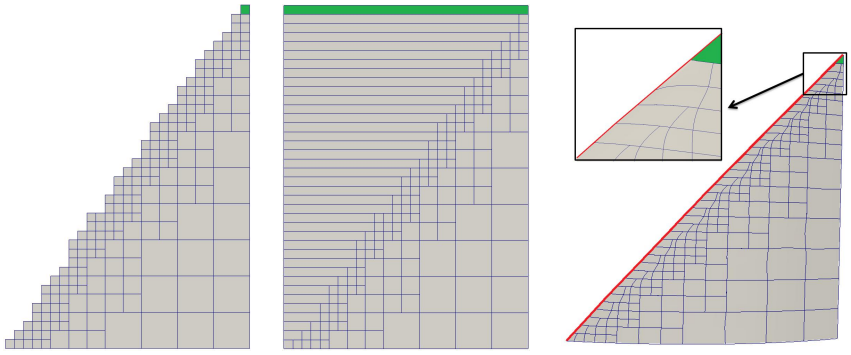
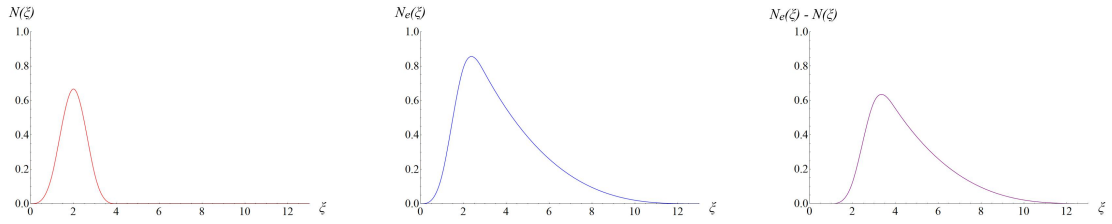


Figure 5.10: Reparameterization of trimmed surface with three corners trimmed off (the degenerated element is marked in green). (a) Deleting removed elements; (b) modifying the topology of preserved elements and extending edge intervals to align with the boundary of the rectangular parametric domain; and (c) reparameterized trimmed surface using the weighted T-spline.



(a) (b) (c)

Figure 5.11: Influence of the edge interval extension to the basis function. (a) A basis function $N(\xi)$ defined on Ξ; (b) a basis function $N_e(\xi)$ defined on Ξ_e; and (c) the difference between these two basis functions $N_e(\xi) - N(\xi)$.

Influence of Edge Interval Extension. Even though the trimming curve can be exactly reparameterized by edge interval extension, the basis functions are changed. For example, consider two knot vectors, $\Xi = \{0,1,2,3,4\}$ and $\Xi_e = \{0,1,2,3,13\}$. The basis functions defined on Ξ and Ξ_e are $N(\xi)$ and $N_e(\xi)$ respectively, as shown in Fig. 5.11(a, b). We also plot their difference $N_e(\xi) - N(\xi)$ in Fig. 5.11(c). We can observe that $N_e(\xi) - N(\xi) = 0$ for $\xi \in [0,1]$, which means that modifying the last knot interval does not influence $N_e(\xi)$ on its first knot interval, while $N_e(\xi)$ is influenced on the other three knot intervals. In other words, the edge interval extension only influences $N_e(\xi)$ on the three-ring elements around the trimming curve. In addition, we obtain the distorted curves around the trimming curve, shown in Fig. 5.9(c) and Fig. 5.10(c).

Remark 4.1. The edge intervals can be extended in either parametric direction. We choose the direction which minimizes the longest extended knot interval. In this way, we try to reduce the distortion introduced around the trimming curve.

5.2.2 Four Types of Trimmed NURBS Patches

Depending on the number of trimmed-off corners, we study four common types of trimmed NURBS patches in design, as shown in Figs. 5.9, 5.10, 5.12, and 5.13. The weighted T-spline with edge interval extension is applied for the reconstruction. The statistics of the

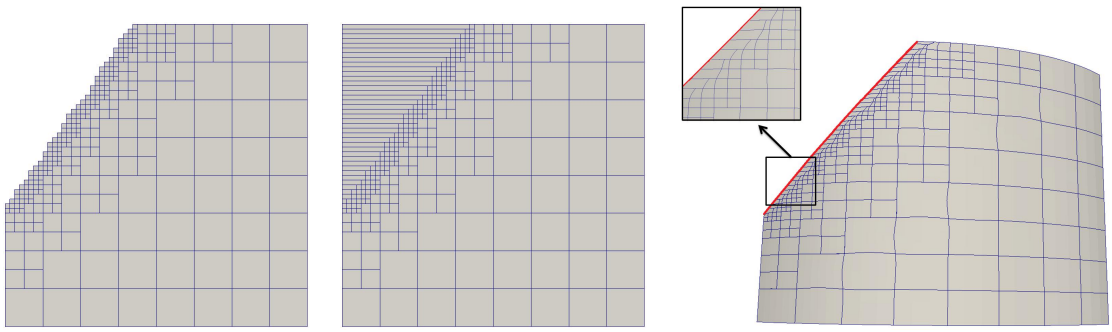
results are given in Tab. 5.2. The reparameterization results using standard T-splines are also given in Fig. 5.14 for comparison.

As shown in Tab. 2, compared with standard T-splines, the weighted T-splines decrease the number of control points and T-mesh elements. In these four trimmed patches, the number of control points is decreased by 19% ~ 31%, and the number of T-mesh elements is decreased by 14% ~ 33%. This is because in the weighted T-splines, T-junction extensions are not required to satisfy partition of unity. However the number of extracted Bézier elements is not decreased much (2% ~ 16%). The reason is that for standard T-splines, after T-junction extension, most of the T-mesh elements can only yield one Bézier element. With the weighted T-splines, many T-mesh elements yield multiple Bézier elements.

Table 5.2: Statistics of four types of trimmed NURBS surfaces and comparison with standard T-splines.

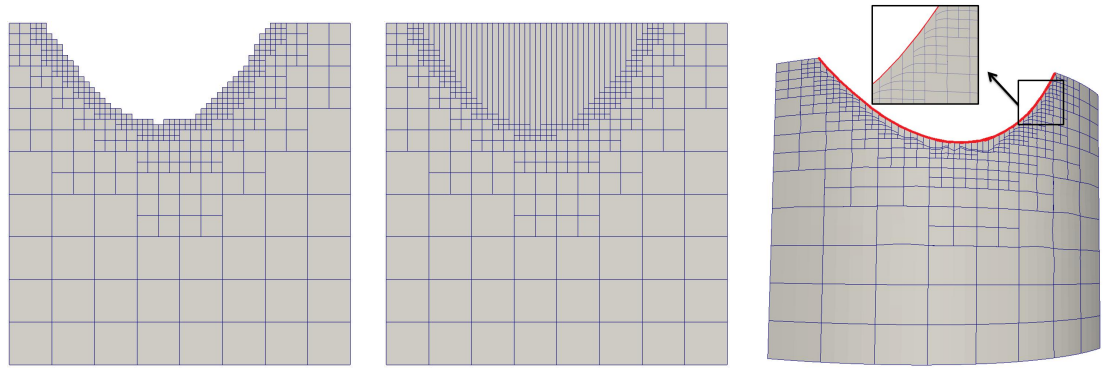
Trimmed-off Corner #	T-mesh Node # (WTSP, STSP, %)	T-mesh Element # (WTSP, STSP, %)	Bézier Element # (WTSP, STSP, %)	Surface Error (Non-Trimmed, Trimmed)
Zero	(401, 523, 19%)	(608, 747, 23%)	(684, 702, 2%)	(0.31%, 0.50%)
One	(464, 609, 31%)	(295, 431, 23%)	(524, 597, 14%)	(0.34%, 0.34%)
Two	(729, 931, 28%)	(487, 680, 33%)	(781, 911, 16%)	(0.30%, 0.43%)
Three	(342, 399, 20%)	(203, 253, 14%)	(343, 351, 2%)	(0.20%, 0.23%)

Note: WTSP stands for the weighted T-spline and STSP stands for the standard T-spline. Symbol "#" represents number and "%" represents the reduced percentage of the nodes or elements.



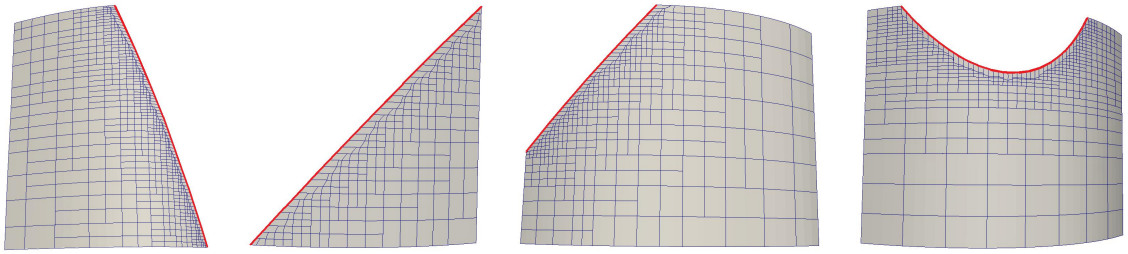
(a) (b) (c)

Figure 5.12: Reparameterization of trimmed NURBS patch with one trimmed-off corner. (a) Deleting removed elements; (b) modifying the topology of preserved elements and extending edge intervals to align with the boundary of the rectangular parametric domain; and (c) the reparameterized trimmed surface using the weighted T-spline.



(a) (b) (c)

Figure 5.13: Reparameterization of a trimmed NURBS patch with no corner trimmed-off. (a) Deleting removed elements; (b) modifying the topology of preserved elements and extending edge intervals to align with the boundary of the rectangular parametric domain; and (c) reparameterized trimmed surface using the weighted T-spline.



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

Figure 5.14: Reparameterized trimmed surfaces with the standard T-spline. (a-d) show the surfaces with two, three, one and zero trimmed-off corners, respectively.

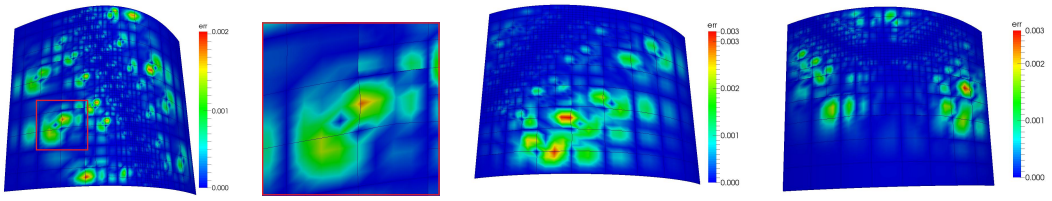
5.2.3 Surface Error

In the reparameterized trimmed surface, the surface error is introduced by the weighted T-spline basis functions and trimming curve reconstruction. The introduced relative surface error is defined as

$$err = \frac{(\mathbf{P}^w(\xi,\eta) - \mathbf{P}^0(\xi,\eta)) \cdot \mathbf{n}}{min\{E\}}, \tag{5.17}$$

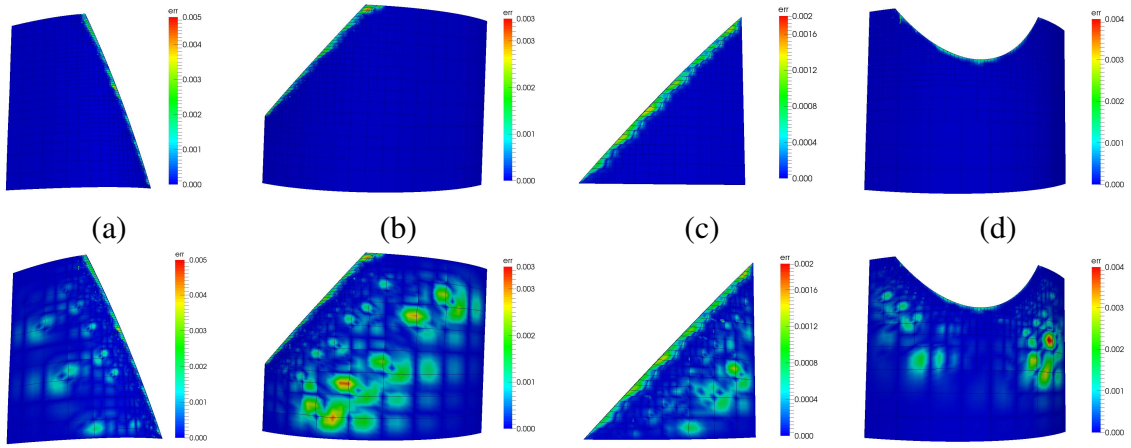
where $\mathbf{P}^w(\xi,\eta)$, $\mathbf{P}^0(\xi,\eta)$ represent a point on the weighted T-spline surface and the input NURBS surface respectively, $\mathbf{n}$ is the surface normal, and $min\{E\}$ is the minimum edge length connecting to this point. Fig. 5.15 shows the error distribution on the weighted T-spline surfaces by locally refining the input NURBS along the trimming curve, where

the maximum error is 0.3%. We can observe that the surface error is localized around the T-junctions in a coarse region (Fig. 5.15(b)). More levels of refinement reduce the error.



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

Figure 5.15: Surface error of the weighted T-spline surfaces compared with the input NURBS surface. (a) The surface refined by a trimming curve with two trimmed-off corners as in Fig. 5.9; (b) zoom-in picture of the region with maximum error in (a); (c) the surface refined by a trimming curve with one or three trimmed-off corners as in Figs. 5.10 and 5.12; and (d) the surface refined by a trimming curve with no trimmed-off corner as in Fig. 5.13.



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

(e) (f) (g) (h)

Figure 5.16: Surface error of trimmed surfaces compared to the input NURBS surface. (a-d) Reparameterized standard T-spline surfaces with two, one, three and zero trimmed-off corners, respectively; and (e-h) the corresponding weighted T-spline surfaces.

After we reparameterize the trimming curve by edge interval extension, the three-ring elements around the trimming curve introduce surface error. To evaluate the error, for each point on the reparameterized T-spline surface we measure its shortest distance to the input NURBS surface. Then the relative surface error is obtained by dividing the distance with the minimum edge length of the element, where the preimage of the surface point locates. Four types of NURBS patches are reparameterized by the weighted T-splines and the standard

T-splines, and the surface error is shown in Fig. 5.16. The standard T-spline surface only introduces surface error around the trimming curve, whereas the weighted T-spline surface introduces surface error around the trimming curve and T-junctions. The surface error introduced by trimming curve and the weighted T-spline basis functions are of the same scale, $\leq 0.5\%$ as shown in Tab. 5.2 (the last column).

5.3 Handling Extraordinary Nodes with Weighted T-spline

To obtain gap-free T-spline surfaces of arbitrary topologies, handling extraordinary nodes of the T-mesh is a prerequisite. In this section, we first introduce a new knot interval duplication method to assign knot interval vectors to vertices. Based on the assigned knot intervals, bicubic T-spline basis functions are defined, and a gap-free weighted T-spline surface is obtained. Then surface continuity elevation is performed to ensure that the extracted first-ring Bézier elements are G^1-continuous.

Instead of using children basis function, here we use Bézier basis functions to define weighted T-spline basis functions. Based on the Bézier extraction algorithm [79], $N^r(\xi,\eta)$ can also be represented as a linear combination of Bézier basis functions. We have

$$N^r(\xi,\eta) = \sum_i c_i B_i(\xi,\eta), \tag{5.18}$$

where c_i are the Bézier extraction coefficients, or weights, and $B_i(\xi,\eta)$ are Bézier basis functions. c_i are obtained from knot insertion algorithm [34]. To explain Bézier extraction, a T-spline basis function $N(\xi)$ together with seven extracted Bézier basis functions are shown in Fig. 5.17. $N(\xi)$ is defined on knot vector $\{0,1,2,3,4\}$. Seven Bézier basis functions can be extracted from $N(\xi)$, defined on knot vectors $\{0,1,1,1,2\}$, $\{1,1,1,2,2\}$, $\{1,1,2,2,2\}$, $\{1,2,2,2,3\}$, $\{2,2,2,3,3\}$, $\{2,2,3,3,3\}$, and $\{2,3,3,3,4\}$ respectively.

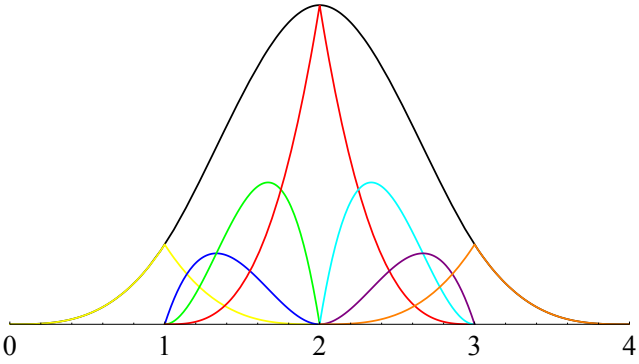


Figure 5.17: T-spline basis function $N(\xi)$ (the black curve) defined on the knot vector $\{0,1,2,3,4\}$ and the extracted seven weighted Bézier basis functions (curves rendered with different colors). The seven weighted Bézier basis functions are defined on $\{0,1,1,1,2\}$, $\{1,1,1,2,2\}$, $\{1,1,2,2,2\}$, $\{1,2,2,2,3\}$, $\{2,2,2,3,3\}$, $\{2,2,3,3,3\}$ and $\{2,3,3,3,4\}$, and the weights are $1/6, 1/3, 2/3, 2/3, 2/3, 1/3$ and $1/6$, respectively.

Analogous to the weight recalculating method to enforce partition of unity [57], a different weighted T-spline basis function can be defined by recalculating the weights of extracted Bézier basis functions. So the corresponding weighted T-spline of $N^r(\xi,\eta)$ can be represented as

$$\hat{N}^w(\xi,\eta) = \sum_i \hat{c}_i B_i(\xi,\eta), \tag{5.19}$$

where $\hat{c}_i$ are the modified weightes. Eqn. (5.19) will be used to define weighted T-spline basis functions to deal with extraordinary nodes, and we will discuss how to compute $\hat{c}_i$ in Section 5.4.

5.4 Weighted T-spline Surface Calculation

To obtain gap-free T-spline surfaces of arbitrary topologies, handling extraordinary nodes of the T-mesh is a prerequisite. In this section, we first introduce a new knot interval duplication method to assign knot interval vectors to vertices. Based on the assigned knot interval vectors, bicubic T-spline basis functions are defined, and a gap-free weighted T-

spline surface is obtained. Then surface continuity elevation is performed to ensure that the extracted first-ring Bézier elements are G^1-continuous.

5.4.1 Topological Constraints and Knot Interval Duplication

A local knot interval vector in the ξ direction is a sequence of knot intervals $\Delta\Xi = \{\Delta\xi_1, \Delta\xi_2, \cdots, \Delta\xi_{p+1}\}$, and its corresponding knot vector is a non-decreasing knot sequence $\Xi = \{\xi_1, \xi_2, \cdots, \xi_{p+2}\}$ such that $\Delta\xi_i = \xi_{i+1} - \xi_i$. Each vertex in the T-mesh is assigned with a knot interval vector along each parametric direction, based on which knot vectors and T-spline basis functions are defined. For vertices near extraordinary nodes, knot intervals cannot be directly obtained in the canonical way. Here we develop a new method to assign knot intervals to such vertices. Necessary terminologies are defined first to assist our explanation.

A *spoke edge* is an edge touching an extraordinary node. A *spoke node* is the vertex other than the extraordinary node on a spoke edge. All the other nodes besides extraordinary nodes and spoke nodes in the T-mesh are *regular nodes*. For a first-ring neighboring T-mesh element of an extraordinary node, the only regular node in this element is a *corner node*. For example in Fig. 5.18(a), the red circle is an extraordinary node, the green circle is a spoke node, and the black circle is a corner node. Three topological constraints are applied to the local region around the extraordinary nodes:

(1) No other extraordinary nodes are allowed within the four-ring neighborhood of an extraordinary node;

(2) No T-junctions are allowed within the four-ring neighborhood of an extraordinary node; and

(3) The knot intervals of all the spoke edges of an extraordinary node are non-zero.

These topological constraints are the foundation of our method to obtain a gap-free T-spline surface. They ensure that the resulting T-spline surface around an extraordinary

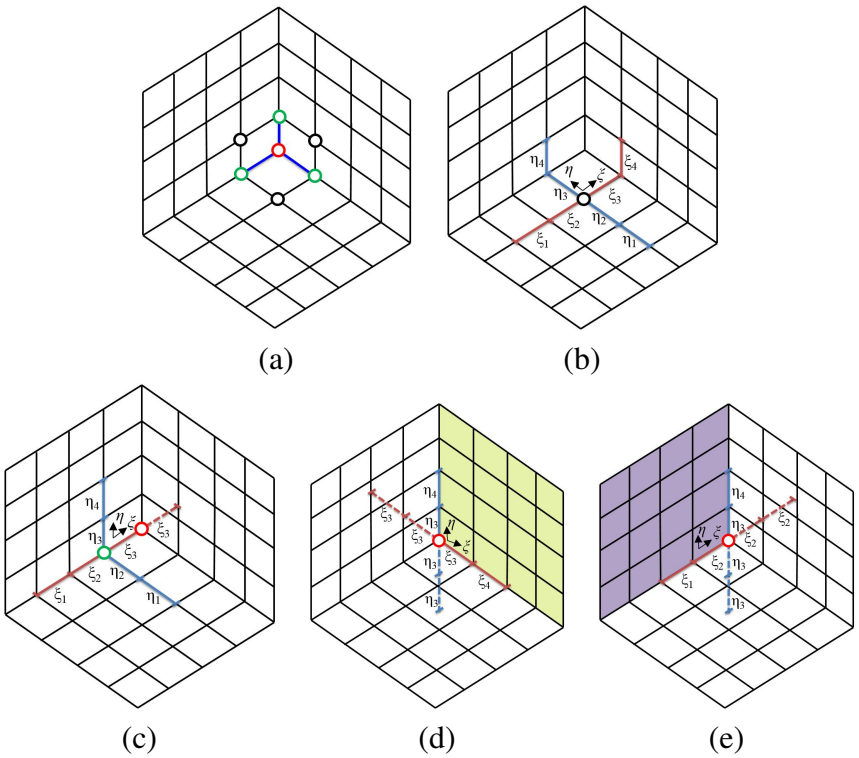


Figure 5.18: Knot interval extraction near the extraordinary node. (a) Corner nodes (the black circles), spoke nodes (the green circles), extraordinary node (the red circle) and spoke edges (the blue edges) in the T-mesh configuration; (b) corner node with ordinary knot intervals; (c) spoke node with extended knot intervals; (d) extraordinary node with knot intervals duplicated with respect to elements in the green region; and (e) the same extraordinary node with knot intervals duplicated with respect to the elements in the purple region.

node is not influenced by other extraordinary nodes or T-junctions. For elements beyond the two-ring neighborhood of any extraordinary node, we assume analysis-suitable requirements are satisfied, and weighted T-spline basis functions [57] are employed to calculate analysis-suitable T-splines.

To define T-spline basis functions of degree p, each vertex in the T-mesh is assigned with a pair of local knot interval vectors to define their local knot vectors. How to extract knot intervals from the T-mesh was explained in [79]. For each vertex, we shoot rays in each parametric direction until $p-1$ vertices or perpendicular edges are intersected. A knot interval is the parametric distance between two consecutive intersections. Thus we obtain a knot interval vector in each direction. Zero knot intervals are appended when a

boundary is crossed before $p-1$ intersections are found. However, this method fails when the ray encounters an extraordinary node before $p-1$ intersections. The reason is that the parametric direction cannot be determined for the ray. In [91], zero knot intervals are appended for this situation, resulting in repeated knots in the knot vectors.

Here we explain our interval duplication method to assign knot intervals to the vertices, which is based on the ray-shooting method. The basic idea is to set the current knot interval equal to the previous one whenever the ray encounters an extraordinary node. There are three different cases.

Regular Node (Case 1). Knot intervals are extracted by shooting rays in each parametric direction. They are not influenced by the extraordinary nodes. For example, the extracted knot interval vectors for the regular node (the black circle) in Fig. 5.18(b) are $\{\xi_1,\xi_2,\xi_3,\xi_4\}$ and $\{\eta_1,\eta_2,\eta_3,\eta_4\}$.

Spoke Node (Case 2). Interval duplication is used in this case. When the ray encounters an extraordinary node, it stops. For non-determined interval, we set it equal to the previous interval. For example for the spoke node (the green circle) in Fig. 5.18(c), the first three intervals are found by the ray-shooting method, $\{\xi_1,\xi_2,\xi_3\}$. The ray stops at the extraordinary node (the red circle). We set the last interval equal to ξ_3. The full knot interval for this spoke node in ξ direction is $\{\xi_1,\xi_2,\xi_3,\xi_3\}$.

Extraordinary Node (Case 3). Similar to spoke nodes, the previous interval value is duplicated when an extraordinary node is encountered. The interval duplication depends on the local parametric directions. For example, the red circle is an extraordinary node in Fig. 5.18(d) and (e). For the elements in the green region with the given local coordinate system, the obtained knot intervals by shooting rays are $\{\xi_3,\xi_4\}$ and $\{\eta_3,\eta_4\}$. Via interval duplication, the final knot intervals for this extraordinary node are $\{\xi_3,\xi_3,\xi_3,\xi_4\}$ and $\{\eta_3,\eta_3,\eta_3,\eta_4\}$. For

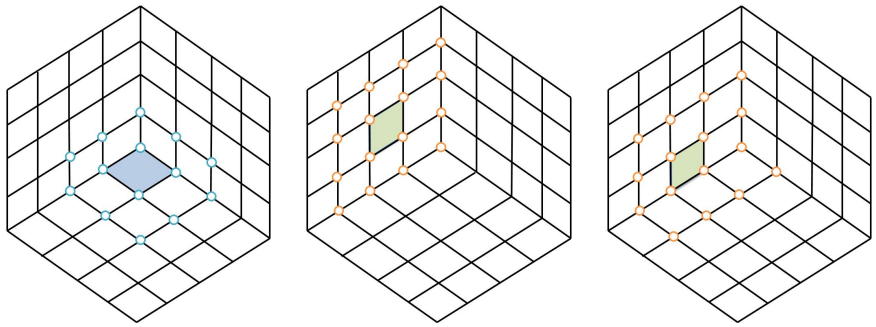


Figure 5.19: Basis functions with support over a Bézier element. (a) For a Bézier element (light blue) in the first-ring neighborhood of an extraordinary node with valance-3, 14 basis functions (cyan circles) have support over it; and (b-c) for a Bézier element (light green) in the second-ring neighborhood, 16 basis functions (orange circles) have support over it.

the elements in the purple region, the knot intervals of the same extraordinary node are $\{\xi_1,\xi_2,\xi_2,\xi_2\}$ and $\{\eta_3,\eta_3,\eta_3,\eta_4\}$.

Remark 5.4.1. The knot intervals are duplicated in such a way that there are no knots with the same value in the knot vectors, and the basis functions defined upon them are all cubic polynomials.

With the introduced interval duplication method, we define T-spline basis functions for all the vertices. For a vertex A, its associated basis function is denoted as N_A. If N_A has non-zero basis function value over the region covered by a T-mesh element, then N_A has support over the corresponding Bézier element. The support of an extraordinary node or a spoke node is its two-ring neighborhood. The number of T-spline basis functions that have support over the two-ring neighboring Bézier elements of an extraordinary node varies. For example for the element with a valance-3 extraordinary node in Fig. 5.19, 14 T-spline basis functions (marked with cyan circles) have support over the first-ring neighboring Bézier elements (light blue), shown in Fig. 5.19(a). For the second-ring neighboring elements like the light green elements in Fig. 5.19(b-c), 16 basis functions have support over it.

Gap-free Requirement. For bicubic T-spline surfaces with extraordinary nodes, two-ring neighboring elements are influenced by the extraordinary nodes. For each influenced

T-mesh element, only one Bézier element is extracted under the topological constraints given in Section 5.4.1. In the following when checking the T-spline surface continuity, we check the continuity across the boundary shared by the extracted Bézier elements. To obtain a gap-free surface, Bézier elements extracted from two adjacent first-ring neighborhood T-mesh elements should be at least C^0-continuous across the shared boundary. For example in Fig. 5.20, there is a valence-n extraordinary node P_E, n spoke nodes ($P_S^1 \sim P_S^n$) and n first-ring neighborhood elements. T-mesh element e^i and e^{i-1} share one spoke edge $P_E P_S^i$ (the red edge) in the T-mesh. The Bézier elements extracted from them need to meet along the shared boundary.

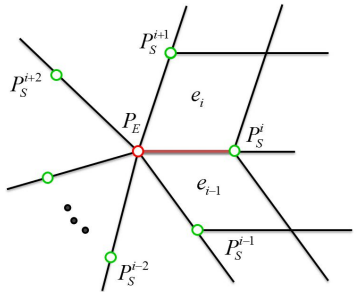


Figure 5.20: Gap-free requirement for a T-mesh with an extraordinary node P_E and n spoke nodes P_S^i. Two first-ring neighborhood T-mesh elements e^{i-1} and e^i share one red edge, and their extracted Bézier elements should be gap-free along the shared boundary.

5.4.2 Gap-free Surface Calculation

For the region beyond the two-ring neighborhood of an extraordinary node, the knot interval extraction, T-spline basis definition and Bézier element extraction follow the canonical T-spline manner [79]. For the first-ring neighboring T-mesh elements, there are two steps to calculate the weighted T-spline surface and extract Bézier elements. For each Bézier element, we find T-spline basis functions with support on it based on the defined local coordinate system, and calculate the corresponding Bézier coefficients. Then the gap-free requirement is applied by modifying the Bézier coefficients.

Bézier Coefficient Calculation. Taking the valance-n extraordinary node P_E in Fig. 5.21(a) as an example, e^i is a first-ring neighborhood T-mesh element, and e_b^i is the Bézier element extracted from it; $P_E P_S^i$ is a spoke edge with edge interval a_i; P_C^i is the corner node of e_i. As shown in Fig. 5.21(b), we define the local parametric coordinate system of e_i by setting P_E as the origin, $P_E P_S^i$ following the ξ direction, $P_E P_S^{i+1}$ following the η direction, $P_E P_S^{i+2}$ following the $-\xi$ direction and $P_E P_S^{i-1}$ following the $-\eta$ direction. Then the spoke nodes $P_S^{i-1} \sim P_S^{i+2}$, the corner nodes $P_C^{i-2} \sim P_C^{i+1}$ are selected and assigned with parametric coordinates. All other spoke nodes and corner nodes are not assigned with parametric coordinates, even through their basis functions have support on e_b^i. The reason is that in the defined local parametric coordinate system, we cannot reach these nodes from the origin by moving along mesh edges following the ξ or η directions. Regular nodes with support over e_b^i are also assigned with local parametric coordinates.

There are 16 vertices assigned with parametric coordinates, shown in Fig. 5.21(a) with circles rendered in different colors. The red circle represents the extraordinary node; the green circles represent the selected spoke nodes; the purple circles represent the selected corner nodes and the orange circles represent the selected regular nodes. Based on the assigned knot intervals, we define local knot vectors and T-spline basis functions for the 16 selected vertices. The T-spline surface can be represented as

$$S^i = \sum_j P_j N_j(\xi,\eta) = \sum_j P_j \sum_{k=1}^{16} M_{j,k}^i B_k(\xi,\eta) = \mathbf{P}_i^T \mathbf{M}^i \mathbf{B} = \sum_{k=1}^{16} Q_k^i B_k(\xi,\eta), \tag{5.20}$$

where P_j are the selected vertices (or control points), $N_j(\xi,\eta)$ are the corresponding T-spline basis functions which have support over e^i, $B_k(\xi,\eta)$ are Bézier basis functions, $M_{j,k}^i$ is the Bézier extraction matrix obtained from Eqn. (5.1), and Q_k^i are the Bézier control points.

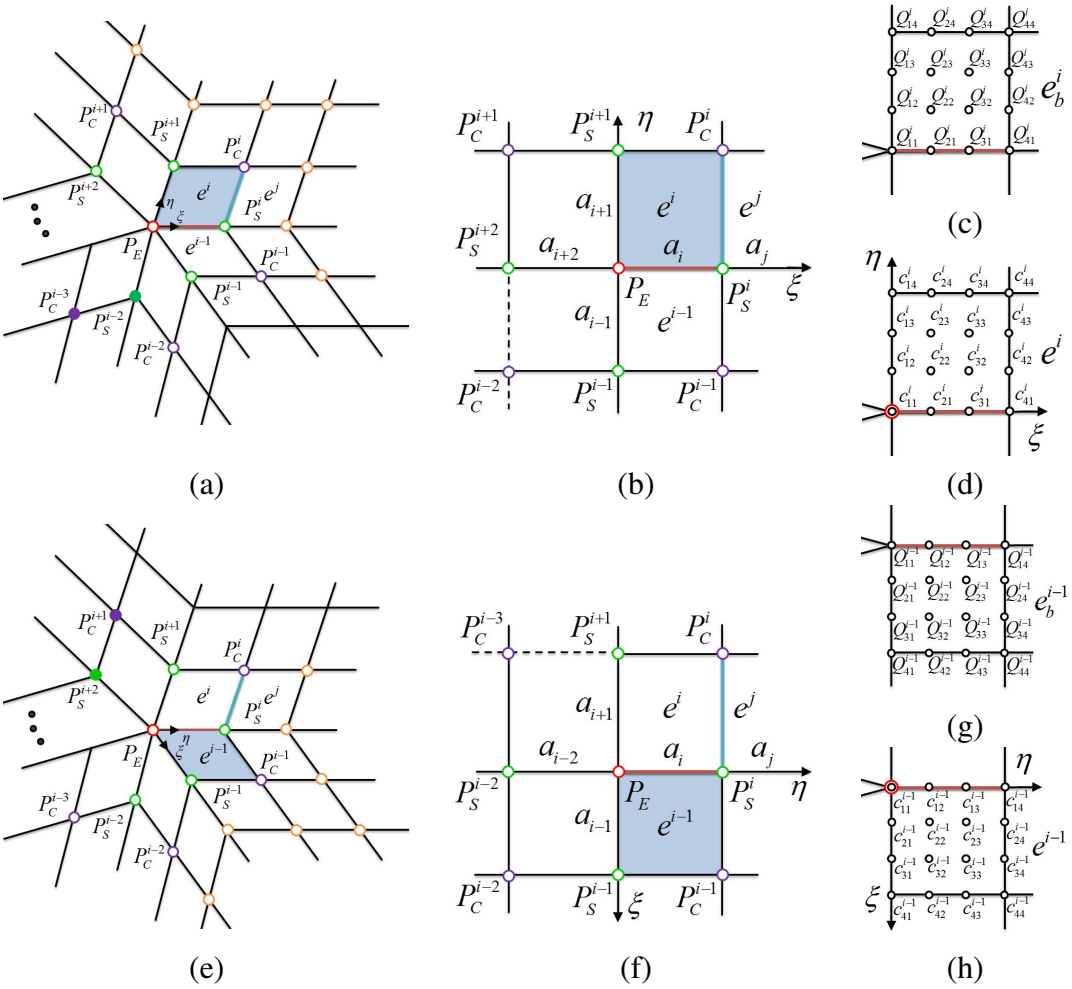


Figure 5.21: Local parametric coordinate system, selected supporting T-spline basis functions, the order of calculated Bézier control points and coefficients for T-mesh element e^i and e^{i-1}. (a) T-mesh element e^i with its selected basis functions marked with circles. Red, green, purple and orange circles represent selected basis functions defined on the extraordinary node, spoke nodes, corner nodes and regular nodes, respectively; (b) the local parametric coordinate system of e^i, where a_i represent the assigned intervals to the edges; (c) the order of calculated control points of Bézier element e^i_b extracted from e^i; (d) the overall coefficient order of e^i_b; (e) element e^{i-1} with its selected basis functions; (f) the local coordinate system of e^{i-1}; (g) the order of control points of Bézier element e^{i-1}_b extracted from e^{i-1}; and (h) the overall coefficient order of e^{i-1}_b.

Here S^i denotes the T-spline surface calculated from e^i. We have

$$Q^i_k = \sum_j P_j M^i_{j,k}. \tag{5.21}$$

Let $k = \alpha \times 4 + \beta$, Eqn. (5.20) is rewritten as

$$S^i = \sum_{k=1}^{16} Q_k^i B_k(\xi, \eta) = \sum_{\alpha=1}^{4} \sum_{\beta=1}^{4} Q_{\alpha\beta}^i B_{\alpha\beta}(\xi, \eta). \tag{5.22}$$

Each Bézier control point $Q_{\alpha\beta}^i$ has a corresponding overall coefficient

$$c_{\alpha\beta}^i = \sum_j M_{j,\alpha\times4+\beta}^i. \tag{5.23}$$

$Q_{\alpha\beta}^i$ have the same order with $c_{\alpha\beta}^i$, as shown in Fig. 5.21(c) and (d), respectively.

Similarly for element e^{i-1}, which shares the spoke edge $P_E P_S^i$ with e^i, its local parametric coordinate system, the selected T-spline basis functions, the order of Bézier control points and the overall Bézier coefficients are shown in Fig. 5.21(e-h). The spoke node P_S^{i-2} and the corner node P_C^{i-3} (marked with solid dots in Fig. 5.21(a)) are not selected for e^i to assign with local parametric coordinates. But they are selected for e^{i-1}. Similarly, P_S^{i+2} and P_C^{i+1} are selected for e^i, but not for e^{i-1}.

Note that if P_E is valance-3, P_S^{i+2} and P_S^{i-1} coincide. This means that in e^i, P_S^{i-1} is assigned with two local parametric coordinates, $(0, -a_{i-1})$ and $(-a_{i+2}, 0)$, to obtain the knot vectors. There are two basis functions defined on P_S^{i-1}. With this duplication, we can always define 16 T-spline basis functions with support over one first-ring element. In addition, we have the following proposition.

Proposition 5.4.1 *The Bézier elements extracted from the first-ring T-mesh elements do not meet with its adjacent first-ring neighbors.*

proof 5.4.2 *Bézier elements e_b^i and e_b^{i-1} are extracted from e^i and e^{i-1} respectively. Assume they meet along the shared boundary, then we have*

$$Q_{\alpha 1}^i = Q_{1\alpha}^{i-1}, \quad 1 \le \alpha \le 4. \tag{5.24}$$

Based on the local parametric coordinate system of e^i and e^{i-1}, vertices P_S^{i+2} and P_C^{i+1} (marked with empty circles in Fig. 5.21(a) and solid dots in Fig. 5.21(e)) have contribution to Q_{11}^i, but not to Q_{11}^{i-1}. P_S^{i-2} and P_C^{i-3} (marked with solid dots in Fig. 5.21(a) and empty circles in Fig. 5.21(e)) have contribution to Q_{11}^{i-1}, but not to Q_{11}^i. Then $Q_{11}^i \neq Q_{11}^{i-1}$, which contradicts the assumption of meeting along the shared boundary. Thus, we prove this proposition.

Bézier Coefficient Modification. We modify the Bézier coefficients to make the resulting T-spline surface gap-free. Based on the local coordinate systems and assigned knot intervals, it is easy to obtain that P_S^{i+2} and P_C^{i+1} only contribute to Q_{11}^i, while P_S^{i-2} and P_C^{i-3} only contribute to Q_{11}^{i-1}. So we have

$$Q_{\alpha 1}^i = Q_{1\alpha}^{i-1}, \quad 2 \leq \alpha \leq 4. \tag{5.25}$$

We only need to modify Q_{11}^i and Q_{11}^{i-1} to ensure e_b^i and e_b^{i-1} meet along their shared boundary. From Eqn. (5.20), suppose the contribution of N_S^{i+2} and N_C^{i+1} to c_{11}^i are $M^i_{I(N_S^{i+2}),1}$ and $M^i_{I(N_C^{i+1}),1}$ respectively. The contribution of N_S^{i-2} and N_C^{i-3} to c_{11}^{i-1} are $M^{i-1}_{I(N_S^{i-2}),1}$ and $M^{i-1}_{I(N_C^{i-3}),1}$ respectively, where $I(N_j)$ is the mapping of the basis function N_j to its local index in Eqn. (5.20). Q_{11}^i and Q_{11}^{i-1} should be modified as

$$Q_{11}^i = Q_{11}^i + P_S^{i-2} M^{i-1}_{I(N_S^{i-2}),1} + P_C^{i-3} M^{i-1}_{I(N_C^{i-3}),1}, \tag{5.26}$$

and

$$Q_{11}^{i-1} = Q_{11}^{i-1} + P_S^{i+2} M^i_{I(N_S^{i+2}),1} + P_C^{i+1} M^i_{I(N_C^{i+1}),1}. \tag{5.27}$$

This modification can be recognized as adding P_S^{i-2} and P_C^{i-3}, which were selected for e^{i-1} only, to the extraction of e_b^i. Similarly, P_S^{i+2} and P_C^{i+1} are added to the extraction of e_b^{i-1}.

Analogously, to constrain that all the first-ring Bézier elements meet at their shared corner, all the spoke nodes and corner nodes that are not selected for the extraction of e_b^i

should be added back. Q^i_{11} is modified as

$$\bar{Q}^i_{11} = Q^i_{11} + \sum_{j=1, j\neq i}^{n} \bar{P}_{SC} M^j_{I(\bar{N}_{SC}),1}, \tag{5.28}$$

where $\bar{P}_{SC}$ are the basis functions defined on the corner nodes and the spoke nodes not selected in the Bézier extraction of e^i, and $\bar{N}_{SC}$ are the associated T-spline basis functions. Note that $\bar{Q}^i_{11}$ is constant for all the first-ring Bézier elements. In Eqn. (5.23), since only c^i_{11} and c^{i-1}_{11} are modified, $\mathbf{M}^i$ is modified by adding new rows with non-zero entry only at the first position.

Based on the assigned knot intervals and local coordinate systems, there are always 16 T-spline basis functions selected for the extraction of one Bézier element. All the 16 basis functions are defined on local knot vectors without repeating knots. The T-spline basis functions are linearly independent and satisfy partition of unity before coefficient modification. Then from Eqn. (5.20) and Theorem 1 in [53], $\mathbf{M}^i$ is in full-rank and $c^i_{11} = 1$. Only the extraordinary node, spoke nodes and corner nodes have contribution to c^i_{11}. After Bézier coefficient modification, c^i_{11} is changed to

$$\begin{aligned} \bar{c}^i_{11} &= c^i_{11} + \sum_{j=1, j\neq i}^{n} M^j_{I(\bar{N}_{SC}),1} = 1 + \sum_{j=1, j\neq i}^{n} M^j_{I(\tilde{N}_{SC}),1} \\ &= M^i_{I(N_E),1} + M^i_{I(\bar{N}_{SC}),1} + \sum_{j=1, j\neq i}^{n} M^j_{I(\bar{N}_{SC}),1} = M^i_{I(N_E),1} + \sum_{j=1}^{n} M^j_{I(N_{SC}),1} > 1, \end{aligned} \tag{5.29}$$

where N_E is the basis function at the extraordinary node P_E, $\tilde{N}_{SC}$ are the basis functions defined on the corner nodes and spoke nodes selected for e^i, and N_{SC} are the basis functions defined on all the spoke nodes and corner nodes. To enforce $\bar{c}^i_{11} = 1$, we let

$$\bar{c}^i_{11} = M^i_{I(N_E),1} + \gamma \sum_{j=1}^{n} M^j_{I(N_{SC}),1} = 1, \tag{5.30}$$

where

$$\gamma = \frac{1 - M^i_{I(N_E),1}}{\sum_{j=1}^{n} M^j_{I(N_{SC}),1}}. \tag{5.31}$$

Eqns. (5.30)-(5.31) are used to modify the first column of $\mathbf{M}^i$. In the following we check the continuity between the first-ring and second-ring neighboring Bézier elements.

Proposition 5.4.3 *The Bézier elements extracted from the first-ring and second-ring T-mesh elements are C^1-continuous across their shared boundary.*

proof 5.4.4 *In Fig. 5.21(a), T-mesh elements e^i and e^j share one cyan edge. Bézier elements e^i_b and e^j_b are extracted from them. For e^i_b, its first derivative at the boundary shared with e^j_b is $\frac{\partial b^i(\xi,\eta)}{\partial \xi}|_{\eta=1}$, or $b^i_\xi(\xi)$. We adopt the notation*

$$\langle \delta_1, \delta_2, \cdots, \delta_{p+1} \rangle^p(\xi) = \sum_{k=1}^{p+1} \delta_k B^p_k(\xi), \tag{5.32}$$

where $B^p_k(\xi)$ is a Bernstein polynomial of degree p. Then we have

$$b^i_\xi(\xi) = 3\langle Q^i_{31} - Q^i_{41}, Q^i_{32} - Q^i_{42}, Q^i_{33} - Q^i_{43}, Q^i_{34} - Q^i_{44} \rangle^3(\xi). \tag{5.33}$$

Here we check the contribution from the extraordinary node P_E to $b^i_\xi(\xi)$. N_E is the basis function at the extraordinary node P_E. Since $\eta = 1$, we only check N_E in the ξ direction. Based on the local parametric coordinate system of e^i, the knot vector to define N_E in the ξ direction is $\{-2a_i, -a_i, 0, a_i, a_i + a_j\}$. From Eqns. (5.20)*,* (5.21) *and* (5.33)*, the contribution of P_E to $b^i_\xi(\xi)$ is*

$$3P_E\langle \tilde{c}^i_{31} - \tilde{c}^i_{41}, \tilde{c}^i_{32} - \tilde{c}^i_{42}, \tilde{c}^i_{33} - \tilde{c}^i_{43}, \tilde{c}^i_{34} - \tilde{c}^i_{44} \rangle^3(\xi), \tag{5.34}$$

where $\tilde{c}^i_{3\alpha}$ and $\tilde{c}^i_{4\alpha}$ are the contribution of N_E to $c^i_{3\alpha}$ and $c^i_{4\alpha}$ ($1 \le \alpha \le 4$). In the ξ direction, Bézier basis functions $B_{1\alpha}$, $B_{2\alpha}$, $B_{3\alpha}$ and $B_{4\alpha}$ are defined on knot vectors $\{-a_i, 0, 0, 0, a_i\}$,

$\{0,0,0,a_i,a_i\}$, $\{0,0,a_i,a_i,a_i\}$, $\{0,a_i,a_i,a_i,a_i+a_j\}$, *respectively. Based on knot insertion algorithm, we have* $\tilde{c}^i_{3\alpha} = 2\tilde{c}^i_{4\alpha}$. *Eqn.* (5.34) *changes to*

$$-3\langle P_E\tilde{c}^i_{41}, P_E\tilde{c}^i_{42}, P_E\tilde{c}^i_{43}, P_E\tilde{c}^i_{44}\rangle^3(\xi). \tag{5.35}$$

The contribution of all other T-spline basis functions to $b^i_\xi(\xi)$ *has the same expression. Based on Eqn.* (5.21)*, Eqn.* (5.33) *changes to*

$$b^i_\xi(\xi) = -3\langle Q^i_{41}, Q^i_{42}, Q^i_{43}, Q^i_{44}\rangle^3(\xi). \tag{5.36}$$

This method can also be used to obtain the the first derivative at the shared boundary from Bézier element e^j_b. *For the local parametric coordinate system of* e^j, P^i_S *is set as the origin. Its two parametric directions follow the* ξ *and* η *directions of* e^i. *The first derivative at the boundary shared with* e^i_b *is* $\frac{\partial b^j(\xi,\eta)}{\partial\xi}|_{\eta=0}$, *or* $b^j_\xi(\xi)$. *We have*

$$b^j_\xi(\xi) = -3\langle Q^j_{11}, Q^j_{12}, Q^j_{13}, Q^j_{14}\rangle^3(\xi). \tag{5.37}$$

Since $Q^i_{4\alpha} = Q^j_{\alpha 1}$ $(1 \le \alpha \le 4)$, *we have* $b^i_\xi(\xi) = b^j_\xi(\xi)$. *Therefore* e^i_b *and* e^j_b *are* C^1*-continuous across the shared boundary.*

Note that Bézier coefficient modification only changes Q^i_{11}. The C^1 continuity between e^i and e^j remains the same after the modification. With the modified Bézier coefficients, the T-spline surface is defined as

$$S^i = \mathbf{P}^i\hat{\mathbf{M}}^i\mathbf{B}, \tag{5.38}$$

where P^i are the control points including all the spoke nodes and corner nodes, $\hat{\mathbf{M}}^i$ is the modified extraction matrix.

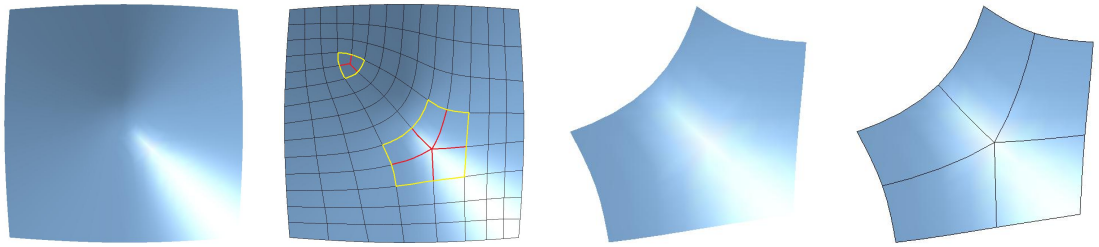
Remark 5.4.2. T-spline basis functions defined by the linear combination of Bézier basis functions with modified coefficients are still analysis-suitable. The new Bézier transformation matrix $\hat{\mathbf{M}}^i$ is obtained by first adding new rows with non-zero entry only at the first position with Eqn. (5.28). Then the first column of the resulting matrix is further modified based on Eqns. (5.30) and (5.31). These two matrix operations do not change the matrix rank. So $\hat{\mathbf{M}}^i$ is in full rank and the modified T-spline basis functions remain analysis-suitable.

For a second-ring neighboring Bézier element, there are always 16 T-spline basis functions with support over it, as shown in Fig. 5.19(b-c). We can define the local coordinate system, and each selected vertex is assigned with local parametric coordinates. The T-spline surface is calculated with Eqn. (5.20), and no Bézier coefficient modification is necessary. So the corresponding Bézier extraction over these elements is the same with the canonical manner. The resulting surface continuity of second-ring neighboring Bézier elements and beyond is C^2.

5.4.3 Surface Continuity Elevation

To obtain higher surface smoothness for the first-ring Bézier elements, we adopt the optimization method introduced in [81] to perform continuity elevation. The necessary and sufficient condition for two adjacent Bézier elements to be G^1-continuous is that they share the same tangent plane across the boundary [103]. Degree elevation is first performed to obtain biquartic Bézier coefficients. These coefficients are then optimized to satisfy the G^1 continuity requirement.

For an extraordinary node of valance-n, there are $20n+1$ unique Bézier coefficients and $20n$ constraint equations derived to satisfy the G^1 continuity requirement. These constraint



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

Figure 5.22: Result of degree elevation for a T-spline model with two extraordinary nodes. (a) Calculated T-spline surface; (b) extracted Bézier elements, where red and yellow edges represent Bézier element boundaries with G^1-continuity and C^1-continuity across them respectively; (c) zoom-in of the first-ring neighborhood of the valance-5 extraordinary node; and (d) extracted Bézier elements of the first-ring neighborhood.

equations are used to assemble the constraint matrix $\mathbf{G}^i$ and the corresponding right hand side vector $\mathbf{g}^i$. Besides, there are $40n$ fairing equations to obtain the fairing matrix $\mathbf{F}^i$ and the right hand side vector $\mathbf{f}^\mathbf{i}$. The detailed expressions of the constraint and fairing equations are given in [81]. We obtain the optimized Bézier coefficients by solving

$$\min \| \mathbf{F}^i \hat{\mathbf{c}}^i - \mathbf{f}^i \|_2, \tag{5.39}$$

where

$$\hat{\mathbf{c}}^i = \arg\min_j \| \mathbf{G}^j \hat{\mathbf{c}}^j - \mathbf{g}^j \|_2 \, . \tag{5.40}$$

The optimization procedure handles extraordinary nodes with different valance numbers correctly. In Fig. 5.22, a T-spline model with two extraordinary nodes of valance 3 and 5 is shown. The calculated T-spline surface is G^1-continuous across the red edges, C^1-continuous across the yellow edges, and C^2-continuous anywhere else.

5.4.4 Comparison with Other Methods

We compare our interval duplication algorithm with three other methods: the template method, the capping method, and the optimization method. The surface continuity, Bézier

Table 5.3: Comparison of four methods dealing with extraordinary nodes regarding surface continuity of edges in Fig. 5.22(b), Bézier extraction Matrix, and T-mesh modification.

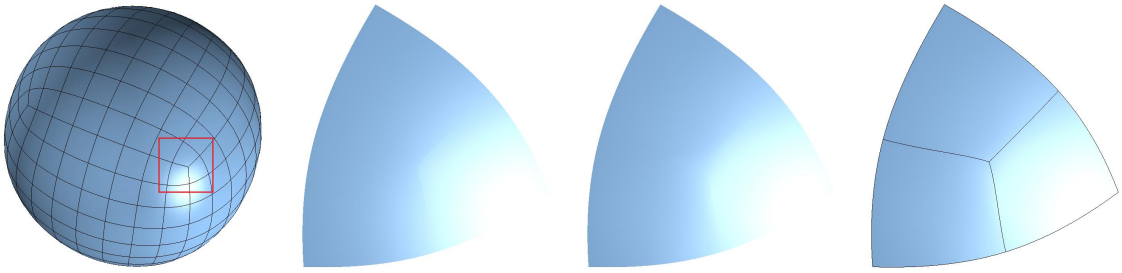
Method	Surface Continuity across Edges			Bézier Extraction	T-mesh
	Red	Yellow	2*nd*-ring	Matrix Calculation	Modification
Template	C^0	C^0	C^0	Knot Insertion	YES
Capping	G^1	C^1	C^1	NONE	NO
Optimization	G^1	C^1	C^1	Linear Interpolation	NO
Interval Duplication	G^1	C^1	C^2	Knot Insertion	NO

extraction matrix, and T-mesh modification properties of these four methods are listed in Tab. 5.3.

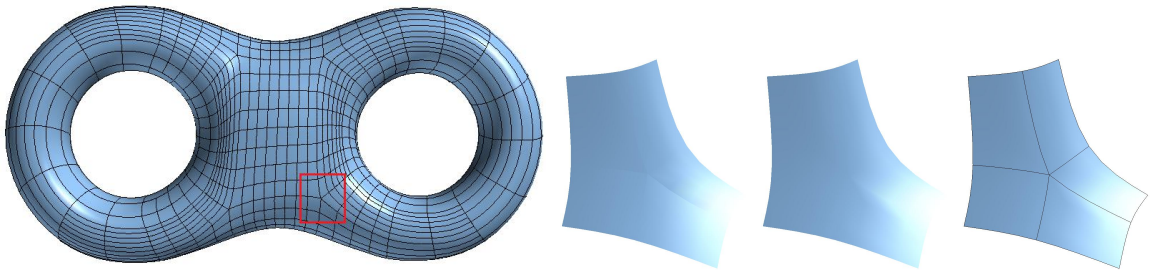
Template Method. Zero-interval edges are inserted around the extraordinary nodes [91], ensuring that the calculated T-spline surface is always gap-free. With this method, an extraordinary node can be within three-ring neighborhood of another one. But the drawback is that repeated knots are introduced to the knot vectors, and the surface continuity is C^0 between first-ring Bézier element pairs and second-ring Bézier element pairs. Furthermore, new control points are introduced for the insertion of zero-length intervals, which increases the total degrees of freedom for analysis.

Capping Method. The Bézier control points are directly calculated from T-spline control points [82]. They satisfy consistency conditions, resulting in G^1-continuous Bézier elements within the first-ring neighborhood. However, the transformation matrix from T-spline basis functions to Bézier basis functions cannot be obtained, which limits its direct usage in isogeometric analysis.

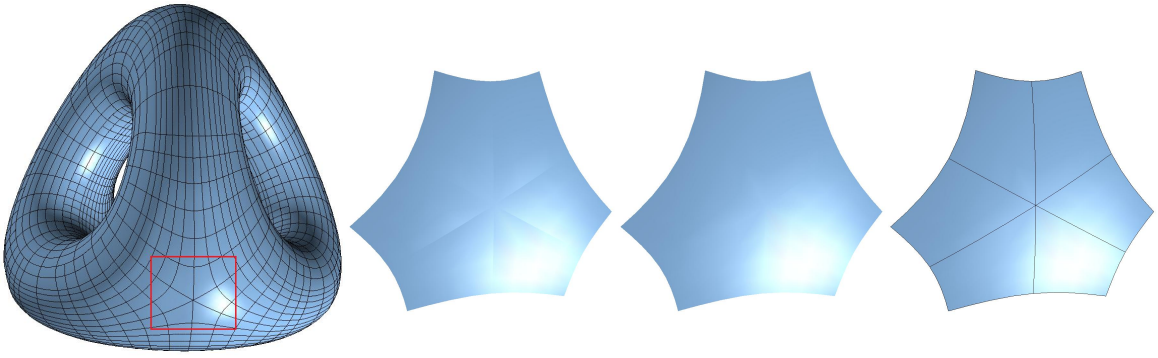
Optimization. The main difference between our method and the optimization method in [81] lies in the way to generate the gap-free T-spline surface before coefficient optimization. In the optimization method, a linear interpolation scheme is introduced to calculate Bézier control points from T-spline control points. Furthermore, the surface between the second-ring Bézier elements is C^2-continuous in our results, better than C^1-continuous from the optimization method.



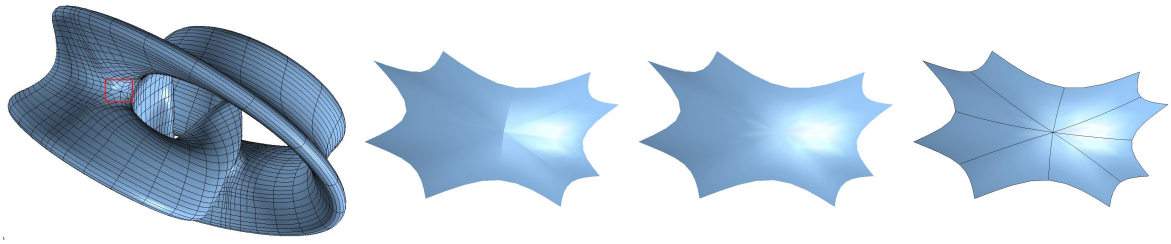
(a) Sphere model.



(b) Eight model.



(c) Tetra model.



(d) Genus-three model.

Figure 5.23: Calculated T-spline of four models. For each model, the final T-spline surface with extracted Bézier elements are shown first, followed by the zoom-in first-ring neighborhood of a selected extraordinary node before and after continuity elevation. The first-ring neighborhood Bézier elements are given in the end.

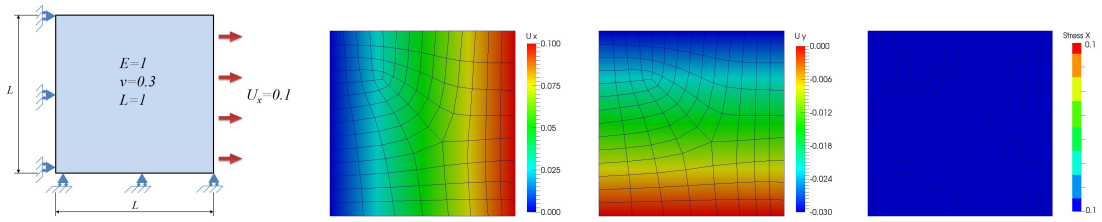
Remark 5.4.3. From comparison we can conclude that our knot interval duplication method coupled with Bézier coefficients modification results in the best surface continuity within two-ring neighborhood. No T-mesh modification is needed for the initial Bézier extraction matrix calculation. The resulting T-spline can be directly used in isogeometric analysis.

We tested our method on four models, including the Sphere, Eight, Tetra and Genus-three models, see Fig. 5.23. There are eight valance-3 extraordinary nodes on the Sphere model, eight valance-5 extraordinary nodes on the Eight model, eight valance-6 extraordinary nodes on the Tetra model, and four valance-8 extraordinary nodes on the Genus-three model. For each model, the final T-spline surface with Bézier element representation is shown first, followed by the zoom-in first-ring neighborhood of a selected extraordinary node. The surface rendering difference shows the surface change before and after degree elevation. The first-ring neighboring Bézier elements are given in the end. Our method can handle extraordinary nodes with different valance numbers correctly to generate gap-free T-spline surfaces. With Bézier coefficient optimization, the surface continuity is increased to G^1 within the one-ring neighborhood.

5.5 Isogeometric Analysis Results

5.5.1 Analysis with Reparameterized NURBS Surfaces

The weighted T-spline basis functions can be used in isogeometric analysis, as they satisfy partition of unity and are linearly independent. This can be first verified with the following patch test. We solve a 2D linear elasticity problem on a unit square. The problem setting is shown in Fig. 5.24(a). On the right boundary, the Dirichlet boundary condition is strongly imposed along the x direction by 0.01. The left and bottom boundaries are fixed in the x and y directions, respectively. We obtain linearly distributed displacement along the x direction and uniform σ_x within the model, achieving the machine precision.



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

Figure 5.24: Patch test with the weighted T-spline in a 2D linear elasticity problem. (a) Problem setting; and (b-d) analysis results with contours showing the distributions of U_x, U_y and σ_x, respectively.

We also solve one benchmark problem here. We analyze the infinite plate with circular hole under a constant far-field in-plane tension (T_x) in the x-direction, as shown in Fig. 5.25(a). The analytical solution [4] is

$$\begin{cases} \sigma_{r,r} = \dfrac{T_x}{2}(1-\dfrac{R^2}{r^2}) + \dfrac{T_x}{2}(1-4\dfrac{R^2}{r^2}+3\dfrac{R^4}{r^4})cos2\theta; \\ \sigma_{\theta,\theta} = \dfrac{T_x}{2}(1+\dfrac{R^2}{r^2}) - \dfrac{T_x}{2}(1+3\dfrac{R^4}{r^4})cos2\theta; \\ \sigma_{r,\theta} = -\dfrac{T_x}{2}(1+2\dfrac{R^2}{r^2}-3\dfrac{R^4}{r^4})sin2\theta. \end{cases} \tag{5.41}$$

Due to the symmetry of the plate, only a quarter is used for analysis (Fig. 5.25(b)). For the modeling of the quarter plate, we first generate one NURBS patch, and then trim off its left bottom corner with an arc. Then we use the weighted T-spline with edge interval extension to reparameterize this trimmed patch. The designed NURBS patch, the trimming curve and three T-meshes at different refinement levels are shown in Fig. 5.26(a-d). The extracted Bézier elements are shown in Fig. 5.26(e-g).

The σ_{xx} contours from different T-meshes are given in Fig. 5.27(a-c). The calculated stress concentration $\sigma_{xx} = 30$ locates at $r = R$, $\theta = \pi/2$, which exactly matches the analytical solution. The error of the analysis result is assessed with the L_2 norm of stress σ_{xx}. For comparison, we also use NURBS and the standard T-splines to solve this problem; see Fig. 5.28. The error with respect to different DOFs is given in Fig. 5.27(d). We can observe

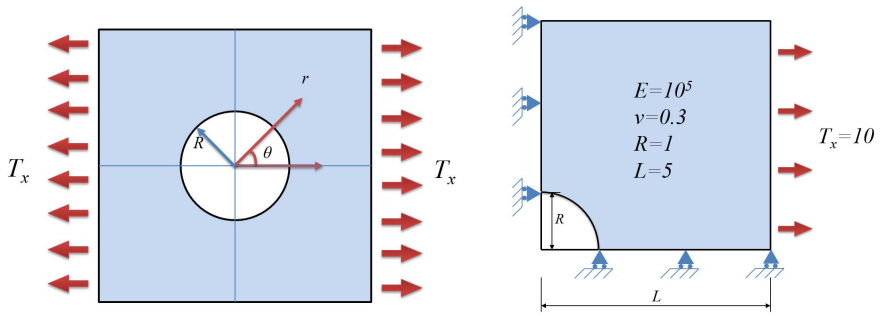


Figure 5.25: An infinite plate with a circular hole under the Newman boundary condition. (a) Problem definition; and (b) a quarter of the plate for modeling and analysis with problem set.

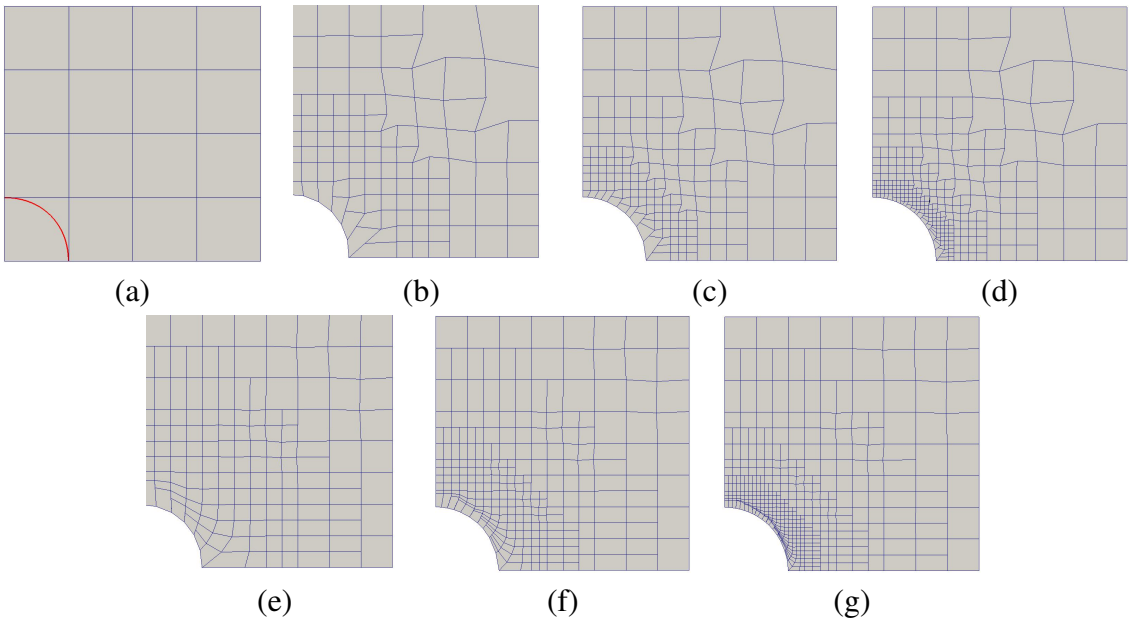
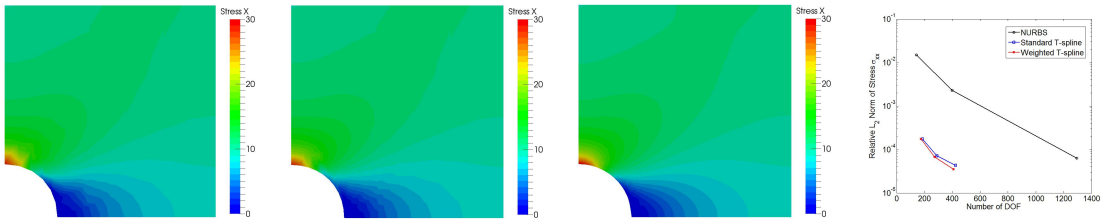
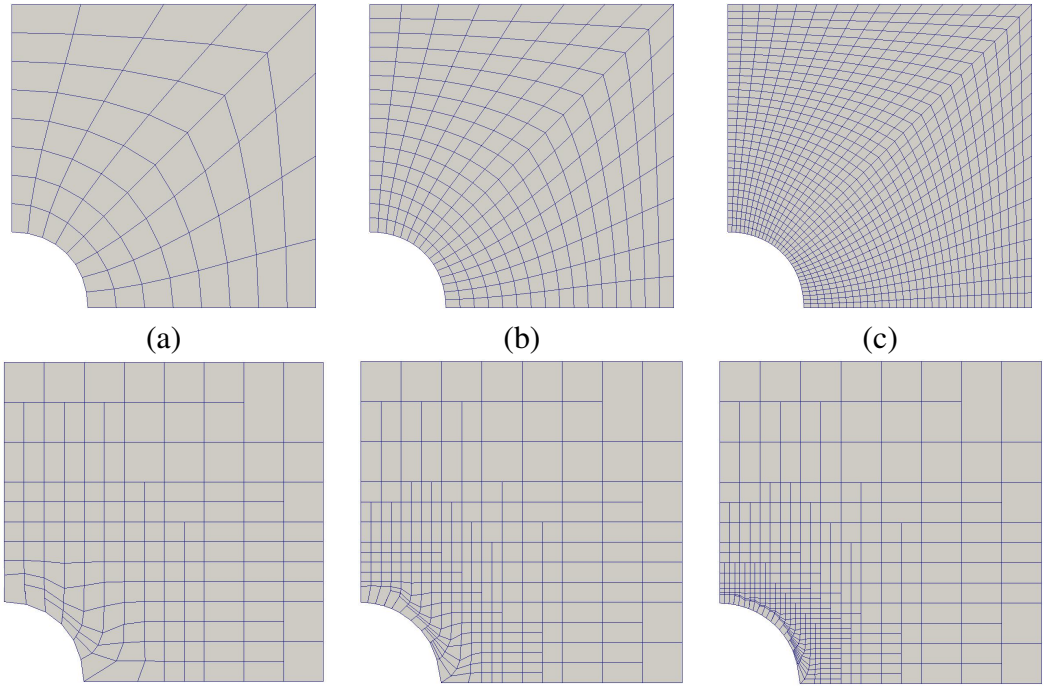


Figure 5.26: A quarter of the plate with the weighted T-spline representation. (a) Designed NURBS patch with the trimming curve (red); (b-d) T-meshes with 2, 3 and 4 levels of refinement, respectively; and (e-g) the corresponding Bézier elements with 2, 3 and 4 levels of refinement, respectively.



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

Figure 5.27: Analysis results of an infinite plate with a circular hole under a constant far-field in-plane tension in the x-direction. Contours in (a-c) show the distribution of σ_{xx} from meshes with 2, 3 and 4 refinement levels, respectively; and (d) shows the L_2 norm error of stress σ_{xx} with respect to the degrees of freedom (DOF).



(a) (b) (c)

(d) (e) (f)

Figure 5.28: The NURBS and standard T-spline models used to solve the benchmark problem. (a-c) The NURBS meshes with 2, 3 and 4 levels of refinement, and (d-f) Bézier elements of the standard T-splines with 2, 3 and 4 levels of refinement.

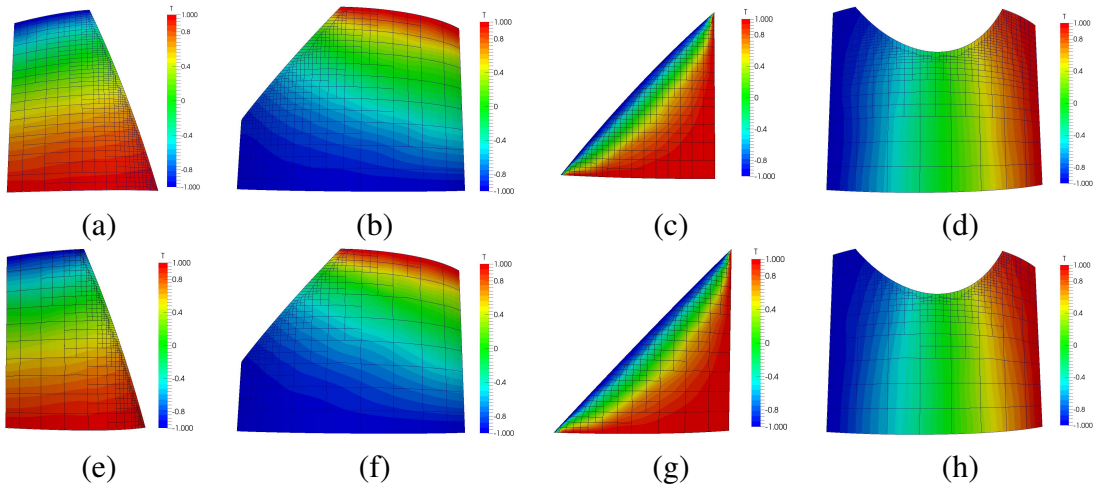


Figure 5.29: Solving the Poisson's equation on trimmed surfaces. (a-d) Results by using standard T-splines; and (e-h) the corresponding results by using weighted T-splines under the same boundary condition.

that the weighted T-splines yield simulation results of the same accuracy with fewer control points.

To test our four types of trimmed curve surfaces in Figs. 5.9, 5.10, 5.12 and 5.13, we solve the Poisson's equation $\Delta u = 0$ over them and the simulation results are shown in Fig. 5.29. Dirichlet boundary conditions with maximum (1.0) and minimum (-1.0) values are assigned to selected geometric boundary edges. The surfaces are reparameterized with both the standard T-spline and the weighted T-spline. For the same trimmed surface with the same boundary conditions, these two T-spline surfaces can produce solutions at the same level of accuracy, while the weighted T-splines need fewer control points and fewer T-spline elements, see Tab. 5.2.

5.5.2 Analysis with Surfaces with Extraordinary Nodes

A benchmark problem is to solve the Laplace equation $\Delta u = 0$ over an L-shaped domain $[-1,1]^2 \backslash [0,1]^2$ with Dirichlet boundary conditions. The analytical solution with polar

coordinates (r,θ) is

$$u(r,\theta) = r^{2/3} sin(2\theta/3 - \pi/3),\ r > 0\ and\ \pi/2 \leq \theta \leq 2\pi. \tag{5.42}$$

The geometry and problem setting are shown in Fig. 5.30(a). The initial mesh has four extraordinary nodes, two of which are valance-3 and the other two are valance-5. Fig. 5.30(b) shows the solution to the problem. The L^2-norm error is evaluated at each element compared to the analytical solution. The initial mesh with elemental error is shown in Fig. 5.30(c). Large error locates at the singular corner and along the boundaries. We performed both uniform and adaptive refinements. The final error distribution of the two refinement methods is given in Fig. 5.30(d, e). The convergence curves are shown in Fig. 5.30(f). Comparison shows that adaptive refinement can achieve the same accuracy with fewer degrees of freedom.

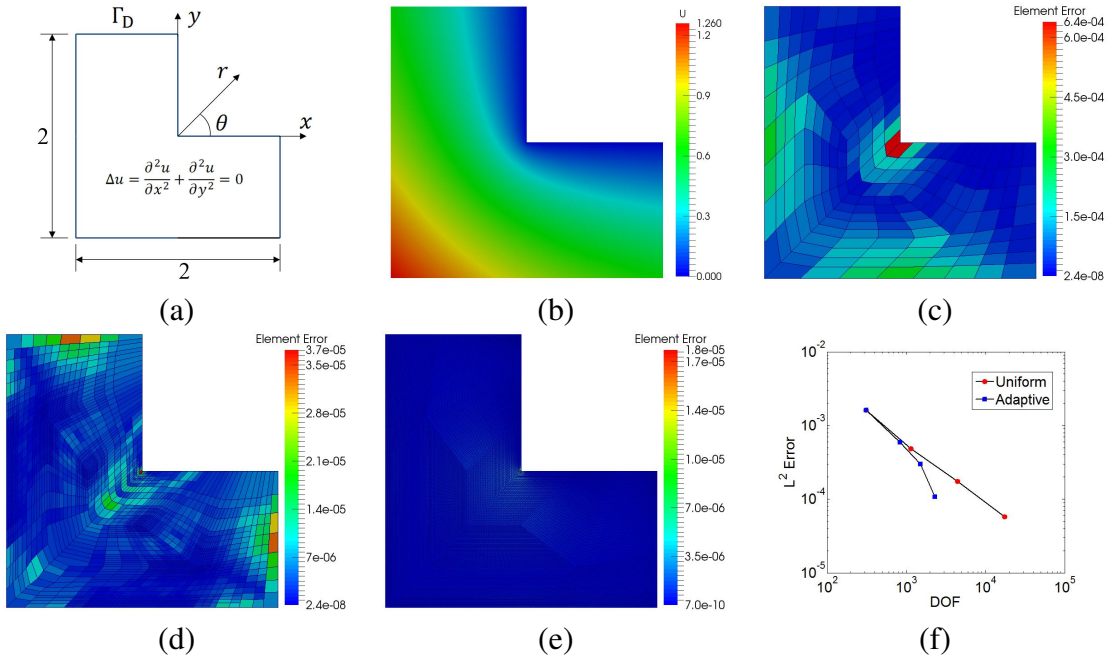


Figure 5.30: Laplace equation on the L-shaped domain solved with weighted T-splines. (a) Geometry and problem settings of the domain; (b) solved result over the domain; (c-f) L^2-error distribution over the initial mesh, adaptively refined mesh, and uniformly refined mesh; (f) convergence curves of the two refinement methods.

5.6 Volumetric Weighted T-splines

The definition of weighted T-splines can be easily extended to 3D. By adding one more dimension, volumetric weighted T-spline is defined as

$$V(\xi,\eta,\zeta) = \sum P_i N_i^w(\xi,\eta,\zeta), \tag{5.43}$$

where P_i are the control points, and $N_i^w(\xi,\eta,\zeta)$ are the associated weighted T-spline basis functions. From the T-mesh, we define basis functions based on the extracted knot vectors. The identification and modification of weighted T-spline basis functions are analogous to the 2D situations, which was explained in [57]. We have developed two methods to construct volumetric weighted T-splines: parametric mapping and sweeping.

Parametric Mapping Method. This method works for genus-zero models. The input is a water-tight T-spline surface containing eight extraordinary nodes with an isoparametric line connecting each pair. The parametric mapping method was introduced in [102], which adopts one cube as the parametric domain and uses octree subdivision to obtain the T-mesh. Two boundary layers are inserted to make sure the calculated volumetric T-spline is conformal to the input. However, extraordinary nodes are introduced in the interior. We use the same subdivision procedure to build the initial T-mesh. After subdivision, we pillow the six patches one by one. One boundary layer with zero-interval value is generated. The eight extraordinary nodes eventually become the eight corners of the cube and there is no extraordinary node in the final T-mesh. Then basis functions are defined and volumetric weighted T-splines are calculated.

Sweeping Method. This method builds solid models by sweeping a surface along a designed path. A T-spline surface is first designed with Bézier extraction matrix calculated. Then one sweeping path is designed, represented as a NURBS curve. The control points of the volumetric T-spline are directly obtained by sweeping the control points of the T-spline surface following the path. The Bézier extraction matrices of the volume are then computed

using the extracted Bézier transformation matrix of the surface and the path. Finally the volumetric T-splines are obtained with Bézier element representation, which can be further used in analysis.

We tested the parametric mapping method with a Boat model, shown in Fig. 5.31. The generated volumetric T-spline is conformal to the input surface. Compared with the method developed in [102], less nodes (1,639 vs 2,582) are required. No extraordinary nodes are introduced in the interior. The cross-sections of the boat model generated using these two methods are compared in Fig. 5.31(e, f) to show the difference.

Torus and Wrench models were generated to test the sweeping method, shown in Figs. 5.32 and 5.33. For the Torus model, one circular plane is first generated with four valance-3 extraordinary nodes introduced in the interior. The sweeping path is an NURBS circular curve, shown in Fig. 5.32(a). The generated volumetric T-spline is shown in Fig. 5.32(b), with cross-section shown in Fig. 5.32(c).

The Wrench model in Fig. 5.33 was generated by sweeping the T-spline surface designed in Rhino. The T-spline surface contains two valance-3 extraordinary nodes and two valance-6 extraordinary nodes. Its sweeping path is shown in Fig. 5.33(a) and the obtained volumetric

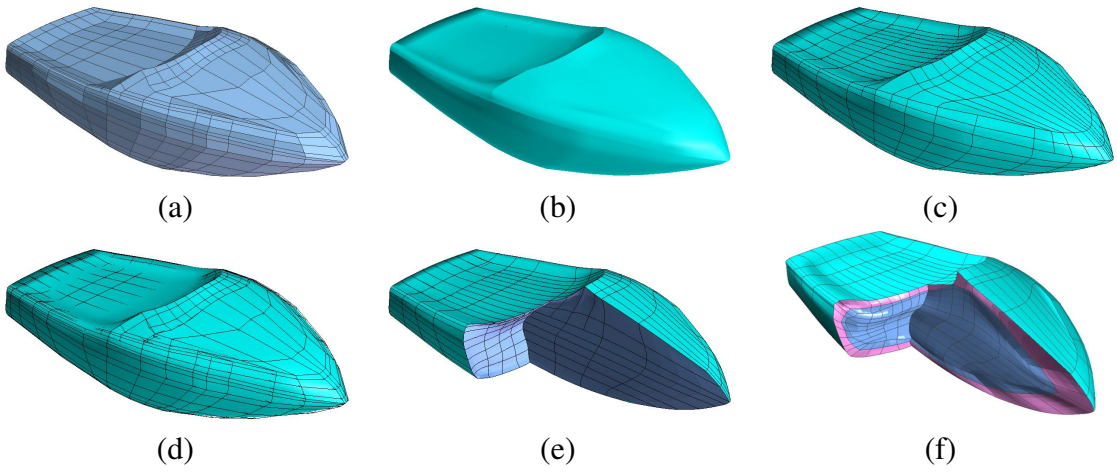
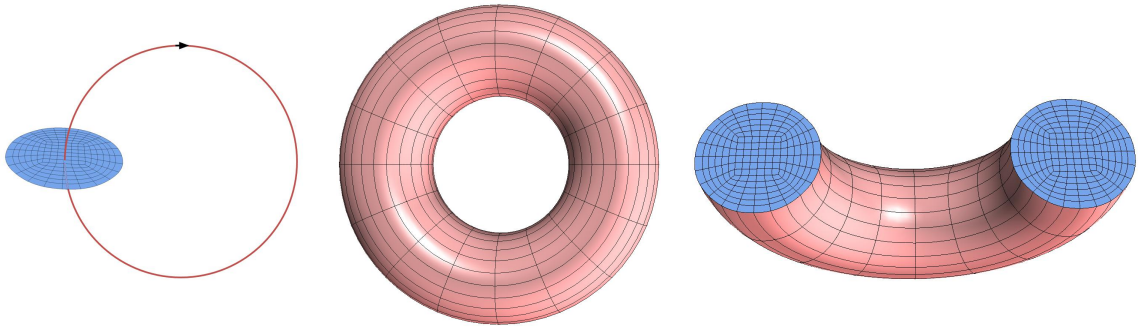
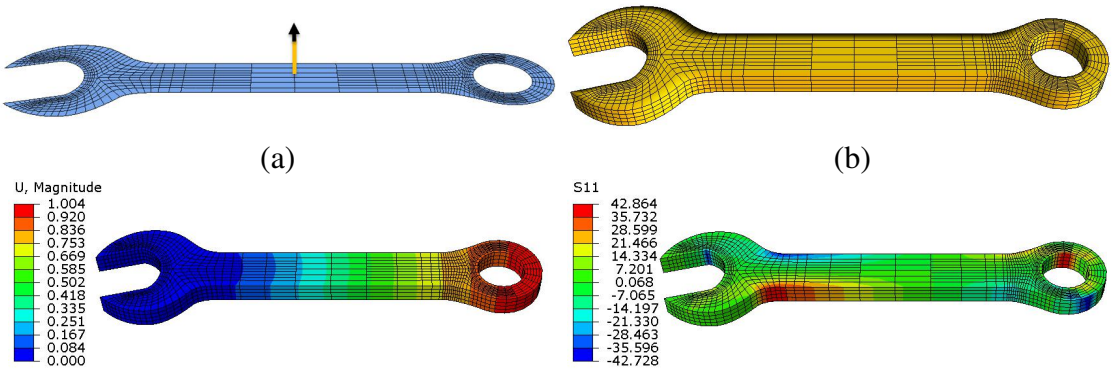


Figure 5.31: Boat model generated from parametric mapping method. (a) T-mesh of input T-spline surface; (b) weighted T-spline surface; (c) volumetric weighted T-spline with Bézier representation; (d) volumetric weighted T-spline with T-mesh; (e) some Bézier elements are removed to show the cross-section; and (e) the same boat model generated by the algorithm given in [102], which pillowed two layers and introduced extraordinary nodes in the interior.



(a) (b) (c)

Figure 5.32: Torus model generated from Sweeping method. (a) Circular cross-section with the Sweeping path; (b) the volumetric T-spline of Torus model; and (c) cutting the torus model to show the interior.



(a) (b)

(c) (d)

Figure 5.33: Wrench model generated from Sweeping method. (a) 2D Wrench model with Sweeping path; (b) the volumetric T-spline of Wrench model; isogeometric analysis result with contour showing U_x and σ_{xx} is given in (c) and (d).

T-spline is given in Fig. 5.33(b). With the developed T-spline IGA software framework [45], we performed linear elasticity analysis in Abaqus. The contours of displacement U_x and stress σ_{xx} are given in Fig. 5.33(c, d), respectively. The sweeping method can also be used to construct complicated high genus models. Besides, only partial extraordinary nodes are introduced to the volumetric weighted T-splines.

5.7 Conclusion

In conclusion, in this chapter, we have proposed one new type of T-spline, named weighted T-spline. It is used to reparameterize trimmed NURBS surfaces, handle extraordinary nodes, and generate volumetric T-spline from CAD models. The weighted T-splines satisfy partition of unity and support local refinement. Weighted T-spline basis functions are proved to be linearly independent. Comparisons with standard T-splines show that weighted T-spline can decrease the required number of control points and T-mesh elements for the same level of refinement. Specific weight adjusting algorithm was developed to deal with the extraordinary nodes for weighted T-spline. The surface is gap-free and the continuity is decreased to C^0. After continuity elevation, the surface continuity is at least G^1. Parametric mapping and sweeping methods are developed to generate volumetric weighted T-splines from CAD models, which provides conformal volumetric weighted T-splines for isogeometric analysis.

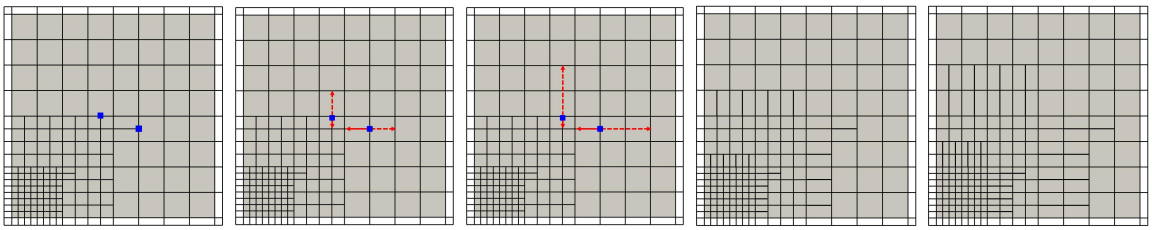
Chapter 6

Hybrid Degree Weighted T-splines

Arbitrary degree T-splines were first studied in [29], which compares the difference between even degree and odd degree T-splines. Analysis-suitable T-splines of arbitrary degree were studied in detail [14, 51, 49, 27], where admissible meshes are given. In this chapter, we study weighted T-spline basis functions of both odd and even degree in detail. With a designed T-mesh splitting scheme, hybrid-degree weighted T-splines are proposed, supporting both local p-refinement and local h-refinement. The local p-refinement introduces a limited number of new control points to the T-mesh. Basis functions of different degrees are defined over the domain. The degree of basis functions of the refined region and transition region is elevated by one after local p-refinement. An L-shaped domain is parameterized with odd-, even- and hybrid-degree weighted T-splines. The Laplace equation is solved over this domain, showing the advantage of hybrid-degree T-splines. High-genus surfaces with extraordinary nodes are also parameterized with hybrid-degree weighted T-splines.

6.1 Arbitrary Degree T-splines

T-splines of arbitrary degree are briefly reviewed here. For further details, we suggest the readers refer to [85, 83, 9, 18]. T-splines discussed here have the same degree in the two parametric directions. We classify them into even- and odd-degree T-splines. T-splines with



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

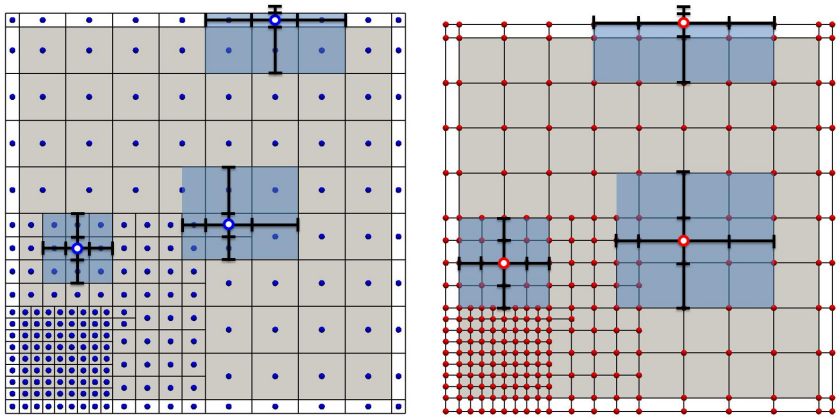
Figure 6.1: (a) T-mesh with two selected T-junctions marked with blue squares. The T-meshes with extensions of these two T-junctions are given in (b) when $p = 2$ and (c) when $p = 3$. The solid red lines represent edge extensions, and the dashed red lines represent face extensions. (d) and (e) show the elemental T-mesh when $p = 2$ and $p = 3$, respectively.

$p = 2$ and $p = 3$ are used as examples to explain the definitions and show the comparison. T-splines of higher degree can be deduced analogously.

6.1.1 T-spline Basics

A *T-mesh* contains all the topological information of a T-spline and is composed of vertices, edges and faces. *Knot intervals* are non-negative real numbers assigned to T-mesh edges. A valid T-mesh configuration requires that the sum of knot intervals assigned to opposite edges of a face stays the same. To maintain the open knot vector property, T-meshes have $\lfloor p/2 \rfloor$ rings of edges with zero-length knot intervals, where the $\lfloor \cdot \rfloor$ represents the integer part of a real number. For example, Fig. 6.1(a) shows a T-mesh with one ring of edges with zero-length knot intervals. The unshaded faces have edges with zero-length intervals. This T-mesh can be used to define basis functions of either $p = 2$ or $p = 3$.

T-junctions are the interior vertices of valance-3 and analogous to hanging nodes in finite element meshes. The blue squares in Fig. 6.1(a) are two selected T-junctions. T-junction *extension* was first discussed in [53], including face extension and edge extension. A *face extension* is a line obtained by moving from the T-junction along the missing edge direction until $\lfloor (p+1)/2 \rfloor$ orthogonal edges are encountered. An *edge extension* is a line obtained by moving opposite to the face extension direction until $\lfloor p/2 \rfloor$ orthogonal edges are encountered. For example in Fig. 6.1(b, c), the face extensions are marked with dashed red



(a) (b)

Figure 6.2: Three basis functions of different degrees are given with their local knot vectors and supported regions shaded with light blue. (a) $p = 2$ with blue dots representing the anchors; and (b) $p = 3$ with red dots representing the anchors.

lines, and edge extensions are marked with solid red lines when $p = 2$ and $p = 3$. For general analysis-suitable T-splines of arbitrary degree, it is required that T-junction extensions cannot meet with each other from different parametric directions [14]. Edge and face extensions are used to check if analysis-suitable requirements are satisfied by the T-mesh. By only drawing all the face extensions and excluding faces with zero-length interval edges in the T-mesh, we obtain the elemental T-meshes as shown in Fig. 6.1(d, e).

Anchors are used to determine the local knot vectors and each anchor is associated with one T-spline basis function. For even degree T-splines, anchors are placed at the centers of the faces. As shown in Fig. 6.2(a), each blue dot represents an anchor. For odd degree T-splines, anchors are placed at each vertex, represented with red dots in Fig. 6.2(b). Local *knot interval vectors* are inferred from the T-mesh. A local knot interval vector in the ξ direction is a sequence of knot intervals $\Delta\widehat{\Xi} = \{\Delta\widehat{\xi}_1, \Delta\widehat{\xi}_2, \cdots, \Delta\widehat{\xi}_{p+1}\}$. There are $p+1$ interval values in each local knot interval vector. The corresponding local knot vector is a non-decreasing knot sequence $\widehat{\Xi} = \{\widehat{\xi}_1, \widehat{\xi}_2, \cdots, \widehat{\xi}_{p+2}\}$ such that $\Delta\widehat{\xi}_i = \widehat{\xi}_{i+1} - \widehat{\xi}_i$. Each anchor is assigned with two local knot vectors along two parametric directions.

To obtain local knot interval vectors, we shoot rays in each parametric direction (both positive and negative) to place a segment centered in the anchor crossing exactly $p+2$

orthogonal edges. Note that "centered in the anchor" means that the segment crosses the same number of orthogonal edges on the left- and right-hand sides of the anchor, and spans a particular set of $p+1$ edges. The knot interval values of the spanned edges are placed into the local knot interval vector consecutively. Zero-length edges are appended when a boundary is crossed before enough orthogonal edges are found.

Based on the local knot interval vectors, local knot vectors are obtained and T-spline basis functions are defined. If a T-spline basis function has non-zero value over one region covered in the T-mesh, then it has support over the geometry in the physical domain extracted from that region. The light blue regions in Fig. 6.2 show the support of different T-spline basis functions associated with the selected anchors.

T-splines of arbitrary degree are defined element-wise. For element e in the elemental T-mesh, suppose $\mathbf{N}^e = \{N_i^e(\xi,\eta)\}_{i=1}^{n^e}$ is the vector of T-spline basis functions having support over e. $N_i^e(\xi,\eta)$ is the basis function mapped to the parent element $\square = [-1,1]^2$ with the affine map introduced in [79]. Then the T-spline geometry is defined by the element geometric map, $\mathbf{x}^e : \square \to \Omega^e$, from the parent element domain to the physical domain as

$$\mathbf{x}^e = \frac{\sum_{i=1}^{n^e} w_i \mathbf{P}_i^e N_i^e(\xi,\eta)}{\sum_{i=1}^{n^e} w_i N_i^e(\xi,\eta)}, \tag{6.1}$$

where $\mathbf{P}_i^e$ is the corresponding control point of the basis function $N_i^e(\xi,\eta)$, and w_i^e is the corresponding weight. Note that $\mathbf{P}_i^e$, w_i^e are all mapped from global to local numbering by the IEN array [79] such that $\mathbf{P}_i^e = \mathbf{P}_{\mathrm{IEN}(i,e)}$ and $w_i^e = w_{\mathrm{IEN}(i,e)}$. The element control points form a matrix $\mathbf{P}^e$ of dimension $n^e \times d_s$, where d_s is the spatial dimension. Analysis-suitable T-splines of arbitrary degree satisfy polynomial partition of unity, which means $\sum_{i=1}^{n^e} N_i^e(\xi,\eta) = 1$. When $w_i^e = 1.0$ for any $\mathbf{P}_i^e$, Eqn. (6.1) is simplified as

$$\mathbf{x}^e = \sum_{i=1}^{n^e} \mathbf{P}_i^e N_i^e(\xi,\eta). \tag{6.2}$$

In the following, we set the weights of all the control points to be 1.0 for the sake of brevity.

6.1.2 Bézier Element Extraction

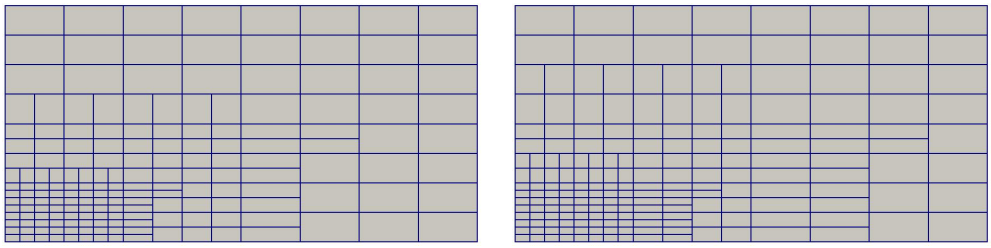
Based on the Bézier extraction algorithm [79, 34], T-spline basis functions can be represented as linear combinations of Bézier basis functions. Each element in the elemental T-mesh corresponds to one Bézier element. The e^{th} Bézier element in the physical domain can be represented as

$$\mathbf{x}^e = \sum_{i=1}^{n^e} \mathbf{P}_i^e N_i^e(\xi,\eta) = \sum_{i=1}^{n^e} \mathbf{P}_i^e \sum_{j=1}^{(p+1)^2} M_{ij}^e B_j(\xi,\eta), \tag{6.3}$$

where $B_j(\xi,\eta)$ is the j^{th} Bézier basis function defined on the parent element, and M_{ij}^e is the Bézier extraction coefficient. By converting Eqn. (6.3) to matrix format, we have

$$\mathbf{x}^e = (\mathbf{P}^e)^T \mathbf{N}^e = (\mathbf{P}^e)^T \mathbf{M}^e \mathbf{B}, \tag{6.4}$$

where $\mathbf{P}^e$ is the matrix of control points, $\mathbf{M}^e$ is the Bézier extraction matrix, and $\mathbf{B}$ is the vector of Bézier basis functions. Eqn. (6.4) is the Bézier element representation of T-splines. Fig. 6.3 shows a rectangular domain parameterized with locally h-refined T-splines and Bézier element representations when $p = 2$ and $p = 3$.



(a) (b)

Figure 6.3: Locally h-refined T-splines with Bézier representation. (a) $p = 2$; and (b) $p = 3$.

6.1.3 Arbitrary-Degree Weighted T-splines

The subset of ASTS is defined based on a simple topological constraint, namely, horizontal T-junction extensions cannot meet with vertical T-junction extensions. If the quadtree subdivision algorithm is applied to a T-mesh, the resulting T-mesh may violate the topological constraint mentioned above and polynomial partition of unity may not be satisfied. However, polynomial partition of unity can be recovered through the use of weighted T-splines [57].

Weighted T-splines enable to use more simple and localized h-refinement strategies over the T-mesh such as, e.g., quadtree subdivision. For a T-spline basis function $N_j(\xi,\eta)$ with corresponding control point $\mathbf{P}_j$ obtained from the knot insertion algorithm, if partition of unity is satisfied in all the support of $N_j(\xi,\eta)$, then the weighted T-spline basis function $\widehat{N}_j(\xi,\eta)$ equals to $N_j(\xi,\eta)$ and $\widehat{\mathbf{P}}_j$ equals to $\mathbf{P}_j$. Otherwise for $N_j(\xi,\eta)$, the weighting coefficients of children basis functions [57] or extracted Bézier basis functions [58] are modified to obtain $\widehat{N}_j(\xi,\eta)$. The control points associated with modified basis functions are computed by solving a linear system [57]. The detailed weighting coefficient modification algorithm is explained as follows.

Given a T-mesh, the weighted T-spline of degree p on the e^{th} Bézier element is defined as

$$\mathbf{x}^e = \sum_{j=1}^{n^e} \widehat{\mathbf{P}}_j^e \widehat{N}_j^e(\xi,\eta) = \sum_{j=1}^{n^e} \widehat{\mathbf{P}}_j^e \sum_{k=1}^{(p+1)^2} \widehat{M}_{jk}^e B_k(\xi,\eta) = (\widehat{\mathbf{P}}^e)^T \widehat{\mathbf{M}}^e \mathbf{B}, \tag{6.5}$$

where $\widehat{N}^e(\xi,\eta)$ are the weighted T-spline basis functions with support over this Bézier element, $\widehat{\mathbf{P}}_j^e$ are the corresponding control points. $\widehat{\mathbf{M}}^e$ is the matrix which transfers weighted T-spline basis functions to Bézier basis functions. $B_k(\xi,\eta)$ is the Bézier basis function. The IEN array is also used here to handle the mapping from global to local numbering.

To obtain $\widehat{\mathbf{M}}^e$, we first calculate the Bézier transformation matrix $\mathbf{M}^e$ for regular T-spline basis functions $N_j^e(\xi,\eta)$. Note that with $\mathbf{M}^e$, partition of unity may not be satisfied everywhere. For a Bézier basis function $B_k(\xi,\eta)$, the summation of calculated weighting

coefficients ($\sum_{j=1}^{n^e} M^e_{jk}$) from $N^e_j(\xi,\eta)$ may not be 1.0. We modify $\mathbf{M}^e$ to $\widehat{\mathbf{M}}^e$ such that

$$\widehat{M}^e_{jk} = \frac{M^e_{jk}}{\sum_{i=1}^{n^e} M^e_{ik}}. \tag{6.6}$$

Consequently, the regular T-spline basis functions are replaced with weighted T-spline basis functions and

$$\sum_{j=1}^{n^e} \widehat{M}^e_{jk} = 1,\ k = 1,2,\cdots,(p+1)^2 \tag{6.7}$$

is always satisfied. Since Bézier basis functions satisfy partition of unity, we have

$$\sum_{j=1}^{n^e} \widehat{N}^e_j(\xi,\eta) = \sum_{j=1}^{n^e} \sum_{k=1}^{(p+1)^2} \widehat{M}^e_{jk} B_k(\xi,\eta) = 1, \tag{6.8}$$

and weighted T-splines always satisfy polynomial partition of unity.

Note that we are normalizing each column of $\mathbf{M}^e$ to obtain $\widehat{\mathbf{M}}^e$ according to Eqn. (6.6). This modification only involves elementary matrix operation and does not change the rank of the matrix. Since $\mathbf{M}^e$ is in full-rank, $\widehat{\mathbf{M}}^e$ is also in full-rank. So the weighted T-splines of arbitrary degree are linearly independent [53, 57, 58].

6.2 Hybrid-Degree Weighted T-splines

In this section, we introduce our algorithm to construct hybrid-degree weighted T-splines by means of local p-refinement. We first introduce hybrid-degree B-spline curves, and then generalize the ideas to hybrid-degree weighted T-spline surfaces.

6.2.1 Hybrid-Degree B-spline Curves

Generally for B-splines, global degree elevation increases the degree of all the basis functions. Suppose a cubic B-spline is defined on the open knot vector $U =$

$\{u_0, u_0, u_0, u_0, u_1, u_2, ..., u_n, u_n, u_n, u_n\}$. There are $n+3$ basis functions. To elevate them to quartic, each unique knot value in U is duplicated once to form a new open knot vector $\overline{U} = \{u_0, u_0, u_0, u_0, u_0, u_1, u_1, u_2, u_2, ..., u_n, u_n, u_n, u_n, u_n\}$, based on which $2n+3$ new quartic B-spline basis functions are defined. The detailed algorithm can be found in [69]. B-spline global degree elevation can increase the degree of basis functions without modifying the geometry [69]. The disadvantages are that a lot of new control points are calculated and the interelement continuity is not increased.

To construct hybrid-degree B-spline curves by means of local p-refinement, we define the spline basis functions on local knot vectors instead of using a global knot vector, which is analogous to what is done in T-splines. Fig. 6.4(a) shows part of a cubic B-spline curve in the index space, where the orange edge has knot interval value 0 and the black edges have knot interval value 1. The B-spline in the parametric space is given in Fig. 6.4(b) with parametric values. The red squares in Fig. 6.4(c) represent the anchors to define cubic basis functions before p-refinement. To perform local p-refinement, we define basis functions of degree $p+1$ only on the refined region.

Knot Insertion. A *Hybrid boundary* is the edge where we add knots to its ends so as to stitch the p-refined region with the rest of the curve. In Fig. 6.4(b), suppose we want to perform local p-refinement to the region $u \leq 4$, then the edge marked in green is the hybrid boundary.

There are four cubic basis functions ($N_i^3(u), 4 \leq i \leq 7$) that have support over the green edge before p-refinement. They are defined on local knot vectors $\{0,1,2,3,4\}$, $\{1,2,3,4,5\}$, $\{2,3,4,5,6\}$, $\{3,4,5,6,7\}$, respectively. The corresponding anchors are represented with crossed-out red squares in Fig. 6.4(d). To perform local p-refinement, two new knots (3 and 4) are inserted. There are five quartic basis functions ($N_i^4(u), 5 \leq i \leq 9$) with support over the green edge (see Fig. 6.5(b)). They are defined on local knot vectors $\{0,1,2,3,3,4\}$, $\{1,2,3,3,4,4\}$, $\{2,3,3,4,4,5\}$, $\{3,3,4,4,5,6\}$, $\{3,4,4,5,6,7\}$, respectively, and the correspond-

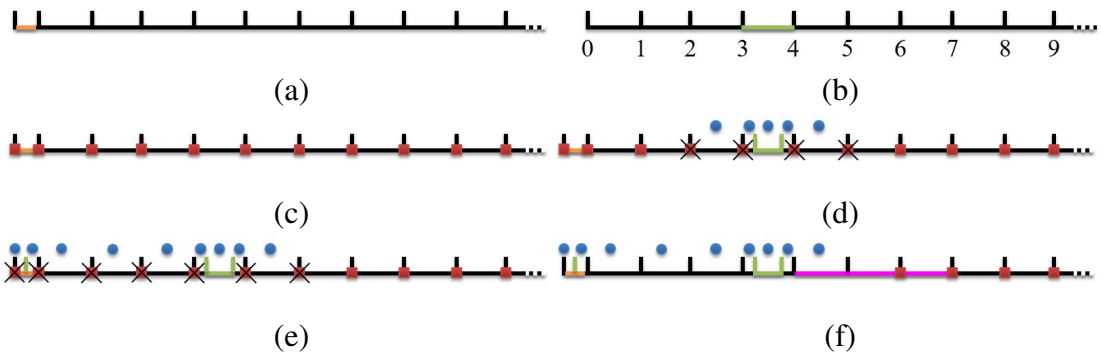


Figure 6.4: Local p-refinement of B-splines of $p = 3$. (a) The mesh of B-spline in the index space, where the orange edge has knot interval value 0, and the black edges have knot interval value 1; (b) the mesh in the parametric space, where the green edge is set as the hybrid boundary; (c) the red squares represent the anchors to define cubic basis functions; (d) for the green edge, the corresponding anchors to define supporting cubic and quartic basis functions of are represented with the crossed-out red squares and blue circles, respectively; (e) quartic basis functions are defined on the p-refined region; and (f) the resulting anchors to define cubic and quartic basis functions . Both cubic and quartic basis functions have support over the magenta transition region.

ing anchors are represented with blue circles in Fig. 6.4(d). To preserve the open knot vector property, zero knot is also inserted, shown in Fig. 6.4(e, f).

Definition of Basis Functions. We disactivate the basis functions of degree p having support over the p-refined region and activate basis functions of degree $p+1$ that have support over this region. In the *p-refined region* ($u \leq 4$), quartic basis functions are defined. The final anchors to define basis functions over the whole domain are shown in Fig. 6.4(f). The defined basis functions before and after local p-refinement as depicted in Fig. 6.4 are shown in Fig. 6.5(a, b). In Fig. 6.5(a), the dashed red lines are the removed cubic basis functions from the p-refined region. The solid blue lines in Fig. 6.5(b) show the newly defined quartic basis functions in the p-refined region.

Transition Region. A *Transition region* connects the refined region and the *unchanged region*. Only basis functions of degree $p+1$ have support over the refined region ($0 \leq u \leq 4$ in Fig. 6.5(b)). Only basis functions of degree p have support over the unchanged region ($u \geq 7$ in Fig. 6.5(b)). Basis functions of both degree p and $p+1$ have support over the transition region ($4 \leq u \leq 7$ in Fig. 6.5(b)). We use hybrid-degree B-splines to represent

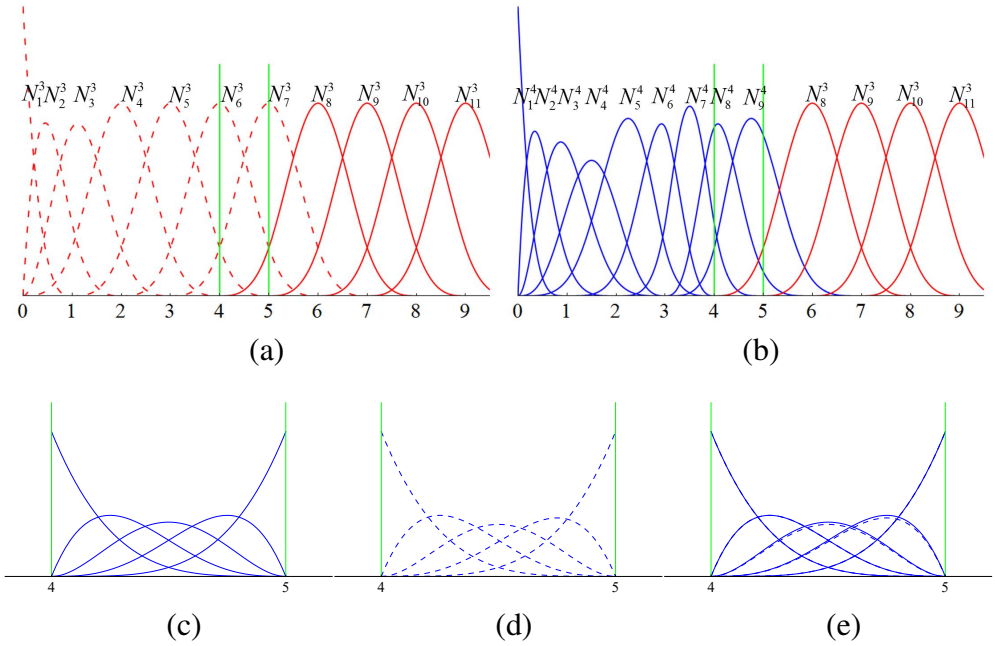


Figure 6.5: (a) Cubic basis functions (red), where the dashed red basis functions are removed; (b) quartic basis functions (blue) are defined on the p-refined region; (c) quartic Bézier basis functions over region $4 \leq u \leq 5$; (d) weighted Bézier basis functions obtained from Eqn. (6.13); and (e) the difference between (c) and (d).

the geometry over this region. Let us take the edge at $4 \leq u \leq 5$ as an example. There is one cubic basis function ($N_8^3(u)$) and three quartic basis functions ($N_i^4(u), 7 \leq i \leq 9$) having support over it, shown in Fig. 6.5(b). To define the geometry, we have

$$\mathbf{x} = \mathbf{P}_8^3 N_8^3(u) + \sum_{i=7}^{9} \mathbf{P}_i^4 N_i^4(u), \tag{6.9}$$

where $\mathbf{P}_8^3$, $\mathbf{P}_i^4$ are the corresponding control points of the basis functions $N_8^3(u)$ and $N_i^4(u)$ respectively. $\mathbf{P}_8^3$ is the 8^{th} vertex in the control polygon of the original cubic B-spline. $\mathbf{P}_i^4$ are the center points of each control polygon edge, which are obtained by linear interpolation of the two vertices of the polygon edge. By representing the B-spline basis functions with

Bézier basis functions, we have

$$\begin{aligned}\mathbf{x} &= \mathbf{P}_8^{e,p}\sum_{j=1}^{4} M_{8j}^{e,p} B_j^p(u) + \sum_{i=7}^{9}\mathbf{P}_i^{e,p+1}\sum_{k=1}^{5} M_{ik}^{e,p+1} B_k^{p+1}(u) \\ &= \sum_{j=1}^{4}\mathbf{Q}_j^{e,p} B_j^p(u) + \sum_{k=1}^{5}\mathbf{Q}_k^{e,p+1} B_k^{p+1}(u),\end{aligned} \tag{6.10}$$

where $p = 3$, $M_{8j}^{e,p}$ and $M_{ik}^{e,p+1}$ are the Bézier extraction coefficients. $\mathbf{Q}_j^{e,p}$ and $\mathbf{Q}_k^{e,p+1}$ are the Bézier control points. Based on the degree elevation algorithm for Bézier elements [69], we have

$$\mathbf{Q}_i^{e,p+1} = (1 - \frac{i-1}{p+1})\mathbf{Q}_i^{e,p} + \frac{i-1}{p+1}\mathbf{Q}_{i-1}^{e,p}, \quad i = 1, 2, \cdots, p+2. \tag{6.11}$$

When $i = 1$, $\frac{i-1}{p+1} = 0$ and when $i = p+2$, $1 - \frac{i-1}{p+1} = 0$. The undefined $\mathbf{Q}_0^{e,p}$ and $\mathbf{Q}_{p+2}^{e,p}$ do not jeopardize the integrity of Eqn. (6.11) and we set them as 0. We move the superscript p of control points and Bézier extraction matrix to subscript for convenience, and convert Eqn. (6.10) to matrix format

$$\mathbf{x}^e = (\mathbf{P}_p^e)^T\mathbf{M}_p^e\mathbf{B}^p + (\mathbf{P}_{p+1}^e)^T\mathbf{M}_{p+1}^e\mathbf{B}^{p+1} = (\mathbf{Q}_p^e)^T\mathbf{B}^p + (\mathbf{Q}_{p+1}^e)^T\mathbf{B}^{p+1}. \tag{6.12}$$

With Eqn. (6.11), Eqn. (6.12) is converted to

$$\begin{aligned}\mathbf{x}^e &= (\mathbf{Q}_p^e)^T\mathbf{T}_p^{p+1}\mathbf{B}^{p+1} + (\mathbf{Q}_{p+1}^e)^T\mathbf{B}^{p+1} = (\mathbf{P}_p^e)^T\mathbf{M}_p^e\mathbf{T}_p^{p+1}\mathbf{B}^{p+1} + (\mathbf{P}_{p+1}^e)^T\mathbf{M}_{p+1}^e\mathbf{B}^{p+1} \\ &= ((\mathbf{P}_p^e)^T\mathbf{M}_p^e\mathbf{T}_p^{p+1} + (\mathbf{P}_{p+1}^e)^T\mathbf{M}_{p+1}^e)\mathbf{B}^{p+1} = \mathbf{R}^T\overline{\mathbf{M}}_{p+1}^e\mathbf{B}^{p+1},\end{aligned} \tag{6.13}$$

where $\mathbf{T}_p^{p+1}$ is obtained from Eqn. (6.11), $\mathbf{R}$ are the control points. $\overline{\mathbf{M}}_{p+1}^e$ is the transformation matrix, and $\mathbf{B}^{p+1}$ represent Bézier basis functions.

Partition of unity is not satisfied here. We recover partition of unity through the use of weighted T-splines as explained in Section 6.1.3. Fig. 6.5(c) shows the five quartic Bézier basis functions. The calculated weights of the five Bézier basis functions obtained from $\overline{\mathbf{M}}_{p+1}^e$ are $1.0, 1.0, 0.95833, 0.95833$ and 1.0, respectively. The weighted Bézier basis

functions are shown in Fig. 6.5(d). The differences are shown in Fig. 6.5(e). Finally Eqn. (6.13) is converted to

$$\mathbf{x}^e = \mathbf{R}^T \widehat{\mathbf{M}}^e_{p+1} \mathbf{B}^{p+1}, \tag{6.14}$$

which is used to define the hybrid-degree weighted B-splines over the transition region.

Remark 6.2.1. The hybrid B-spline defined over the transition region is of degree $p+1$, since we performed degree elevation to the Bézier basis functions of degree p. But the surface continuity is C^{p-1}, which is the same as the unchanged region. The reason is that basis functions of degree p have support over this region.

6.2.2 Local *p*-refinement of T-splines

To explain local p-refinement of T-splines, we define the graph connecting all the faces to introduce new zero-length edges as the hybrid boundary. There are two types of hybrid boundaries, namely the *interior boundary* and the *extended boundary*. Interior boundaries are the loops within the domain. Extended boundaries contain faces lying on the boundary of the geometry. We constrain that the hybrid boundaries cannot be reached by face extension of any T-junction. In this way, T-junctions do not influence the faces lying on the hybrid boundaries.

There are mainly three steps to perform local p-refinement. We first determine the local parametric directions to split the faces on the hybrid boundaries. Then we split the faces along the detected directions to introduce zero-length interval edges to the T-mesh. After the splitting we define higher order basis functions over the p-refined region. The last step is to decide the active original basis functions and calculate the new control points.

Here we take the T-mesh in Fig. 6.6(a) as an example. Since there is only one ring of edges with zero-length intervals on the boundary of the T-mesh, both basis functions of $p = 3$ and $p = 2$ can be defined on it. The three steps are explained mainly based on odd

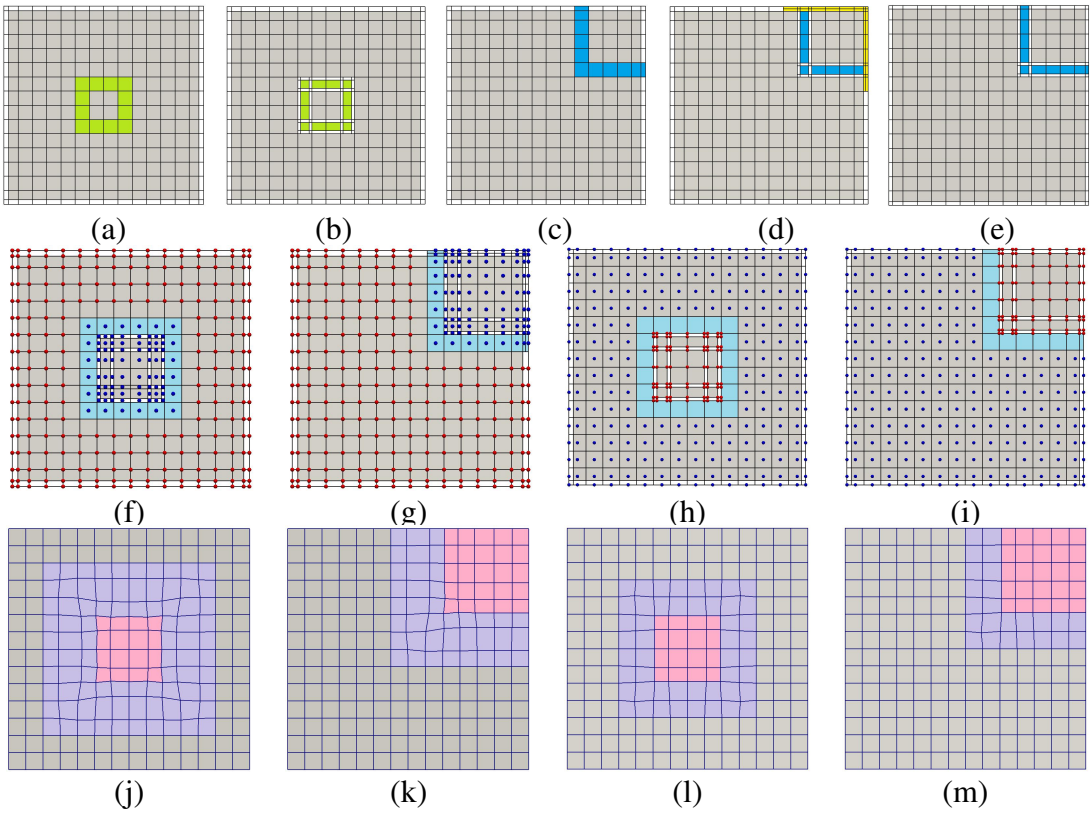


Figure 6.6: Local p-refinement of T-splines with $p = 3$ and $p = 2$. (a) Interior boundary marked in light green; (b) T-mesh splitting result of (a); (c) extended boundary marked in light blue; (d) T-mesh splitting result of (c) when $p = 3$; The yellow faces on the domain boundary are split to preserve the open knot vector property; (e) T-mesh splitting result of (c) when $p = 2$; (f-i) selected anchors to define active basis functions, where the adjacent region is marked in cyan, the red and blue dots represent anchors located at corners and centers of T-mesh faces, respectively, with $p = 3$ in (f, g) and $p = 2$ in (h, i); (j-m) the hybrid T-splines with Bézier representation, where the transition region is marked in purple, and the p-refined region is marked in pink, with $p = 3$ in (j, k) and $p = 2$ in (l, m).

degree T-splines. Even degree T-splines are used to explain the difference and show the comparison.

Local Splitting Direction Detection (Step 1). Fig. 6.6(a) and (c) show the interior and extended hybrid boundaries of the T-mesh, marked in light green and light blue, respectively. We split the faces on the hybrid boundaries to introduce zero-length interval edges across them. For a pair of neighboring faces sharing an edge on the hybrid boundary, the splitting direction is perpendicular to that edge. If a face has edges of two different directions shared by other faces on the hybrid boundary, it should be split in both directions. For example the

faces on the corners of the hybrid boundaries in Fig. 6.6(a) and (c) should be split in both directions.

T-mesh Splitting and Basis Function Definition (Step 2). With the detected splitting directions, we split the faces on the hybrid boundaries as shown in Fig. 6.6(b). If one face should be split in one direction, we split it into three new faces. The new face in the middle has the same edge interval values as the original faces, see the light green faces. The other two have zero-length interval across the splitting direction, see the unshaded faces. If a face should be split in both directions, nine new faces are generated. The one in the middle has the same interval values as the original one. The other ones have either two or four edges with zero-length interval.

In the p-refined region, if there are faces on the boundary of the domain, then the open knot vector property should be satisfied after T-mesh splitting. When performing local p-refinement to T-splines of odd degree, one new boundary layer with zero-length interval edges is introduced. For T-mesh in Fig. 6.6(c), if $p = 3$, there is one layer of faces with zero-length intervals. After refining it to $p = 4$, there should be two layers. We equally split the faces on the geometric boundary to two smaller ones. The splitting follows the direction of the geometry boundary. If the face is at the corner of the geometry, it is equally split into four smaller ones. In Fig. 6.6(d), those faces marked in yellow are the newly generated faces with zero-length intervals. If p is even, no boundary face splitting is required to satisfy the open knot vector property. In Fig. 6.6(e), faces on the geometric boundary remain unchanged, except those on the extended hybrid boundary.

After T-mesh splitting, we place anchors on both the corners and the centers of T-mesh faces. Local knot vectors are inferred for each anchor. For an anchor at the corner, a T-spline basis function of odd degree is defined. Whereas for an anchor at the center, a T-spline basis function of even degree is defined.

We define an *adjacent region* as the first-ring neighborhood of the hybrid boundary beyond the p-refined region. In Fig. 6.6(f - i), the adjacent regions are marked in cyan.

All the basis functions of degree $p+1$ in the p-refined region are set as active. All basis functions of degree p beyond the adjacent region are also set as active. When p is odd, the basis functions of degree $p+1$ at the face center in the adjacent region are set as active. In Fig. 6.6(f, g), the anchors to define active cubic basis functions are represented with red dots. The anchors to define active p-refined quartic basis functions are represented with blue dots. When p is even, the basis functions of degree $p+1$ at the corners shared by the adjacent region and the hybrid boundary are set as active. In Fig. 6.6(h, i), the anchors to define active quadratic basis functions are represented with blue dots, and the anchors to define active cubic basis functions are represented with red dots.

Control Point Calculation (Step 3). Local p-refinement may slightly change the geometry. The refinement algorithms introduced in [69] to calculate control points cannot be used here because zero-length interval edges are only introduced along the hybrid boundary. Here we introduce a direct way to calculate the new control points from the original ones by linear interpolation.

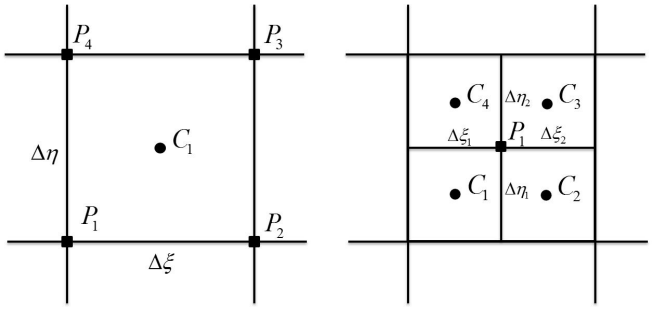
We place five control points on each T-mesh element. Four are at the corners and the last one is in the center. If p is odd, we use corner points to interpolate the center control points for the new basis functions of degree $p+1$. For example in Fig. 6.7(a), a center control point $\mathbf{C}_1$ is calculated by

$$\mathbf{C}_1 = \frac{\mathbf{P}_1 + \mathbf{P}_2 + \mathbf{P}_3 + \mathbf{P}_4}{4}, \tag{6.15}$$

where $\mathbf{P}_1 \sim \mathbf{P}_4$ are four control points at the corners. If p is even, we use center points to perform the interpolation. In Fig. 6.7(b), a corner control point $\mathbf{P}_1$ is calculated by

$$\mathbf{P}_1 = \frac{\Delta\xi_2\Delta\eta_2\mathbf{C}_1 + \Delta\xi_1\Delta\eta_2\mathbf{C}_2 + \Delta\xi_1\Delta\eta_1\mathbf{C}_3 + \Delta\xi_2\Delta\eta_1\mathbf{C}_4}{(\Delta\xi_1 + \Delta\xi_2)(\Delta\eta_1 + \Delta\eta_2)}, \tag{6.16}$$

where $\mathbf{C}_1 \sim \mathbf{C}_4$ are four control points at the centers of the elements which share $\mathbf{P}_1$. After the T-mesh splitting and control point calculation, we can define the hybrid-degree weighted T-splines.



(a) (b)

Figure 6.7: Interpolation scheme to obtain new control points. (a) Using corner control points to calculate the center control point; and (b) using center control points to calculate the corner control point.

6.2.3 Hybrid-Degree Weighted T-spline Construction

We define the hybrid-degree weighted T-splines with Bézier element representation. In the p-refined region, only basis functions of degree $p+1$ have support over extracted Bézier elements. In the unchanged region, only basis functions of degree p have support over extracted Bézier elements. Both basis functions of degree p and $p+1$ have support over the Bézier elements extracted from the transition region. There are $\lfloor (p+1)/2 \rfloor + 1$ rings of elements in the transition region (see Fig. 6.6(j-m)).

Bézier elements in the p-refined or unchanged region are calculated using Eqn. (6.4). Over the transition region, the hybrid-degree T-spline for the e^{th} element is defined as

$$\mathbf{x}^e = \sum_m \mathbf{P}_m^e N_m^{e,p}(\xi,\eta) + \sum_n \mathbf{C}_n^e N_n^{e,p+1}(\xi,\eta) \tag{6.17}$$

when p is odd, and

$$\mathbf{x}^e = \sum_m \mathbf{C}_m^e N_m^{e,p}(\xi,\eta) + \sum_n \mathbf{P}_n^e N_n^{e,p+1}(\xi,\eta) \tag{6.18}$$

when p is even. In Eqns. (6.17)-(6.18), $\mathbf{P}_m^e$, $\mathbf{P}_n^e$ are the corner control points, and $\mathbf{C}_n^e$, $\mathbf{C}_m^e$ are the center control points. $N_m^{e,p}(\xi,\eta)$ and $N_n^{e,p+1}(\xi,\eta)$ are the active T-spline basis functions defined on the T-mesh after refinement. We first derive the expression for T-splines of hybrid degrees when p is odd.

With the Bézier extraction algorithm, Eqn. (6.17) is converted to

$$\begin{aligned}\mathbf{x}^e &= \sum_m \mathbf{P}_m^e \sum_{j=1}^{(p+1)^2} M_{mj}^{e,p} B_j^p(\xi,\eta) + \sum_n \mathbf{C}_n^e \sum_{k=1}^{(p+2)^2} M_{nk}^{e,p+1} B_k^{p+1}(\xi,\eta) \\ &= \sum_{\alpha=1}^{p+1}\sum_{\beta=1}^{p+1} \mathbf{Q}_{\alpha,\beta}^{e,p} B_{\alpha,\beta}^p(\xi,\eta) + \sum_{\alpha=1}^{p+2}\sum_{\beta=1}^{p+2} \mathbf{Q}_{\alpha,\beta}^{e,p+1} B_{\alpha,\beta}^{p+1}(\xi,\eta),\end{aligned} \tag{6.19}$$

where $M_{mj}^{e,p}$ and $M_{nk}^{e,p+1}$ are the Bézier extraction coefficients. $B_j^p(\xi,\eta)$ and $B_k^{p+1}(\xi,\eta)$ are Bézier basis functions of degree p and $p+1$. We have $j = (\alpha-1)\times(p+1)+\beta$, and $k = (\alpha-1)\times(p+2)+\beta$. $\mathbf{Q}_{\alpha,\beta}^{e,p}$ and $\mathbf{Q}_{\alpha,\beta}^{e,p+1}$ are the Bézier control points.

We elevate the Bézier basis functions of degree p to $p+1$ by extending Eqn. (6.11) to 2D such that

$$\begin{aligned}\mathbf{Q}_{\alpha,\beta}^{e,p+1} &= (1-\frac{\alpha-1}{p+1})((1-\frac{\beta-1}{p+1})\mathbf{Q}_{\alpha,\beta}^{e,p} + \frac{\beta-1}{p+1}\mathbf{Q}_{\alpha,\beta-1}^{e,p}) \\ &+ \frac{\alpha-1}{p+1}((1-\frac{\beta-1}{p+1})\mathbf{Q}_{\alpha-1,\beta}^{e,p} + \frac{\beta-1}{p+1}\mathbf{Q}_{\alpha-1,\beta-1}^{e,p}),\end{aligned} \tag{6.20}$$

where $\alpha = 1,2,\cdots,p+2$, and $\beta = 1,2,\cdots,p+2$. Similarly, the undefined $\mathbf{Q}_{0,\beta}^{e,p}$, $\mathbf{Q}_{\alpha,0}^{e,p}$, $\mathbf{Q}_{p+2,\beta}^{e,p}$ and $\mathbf{Q}_{\alpha,p+2}^{e,p}$ do not jeopardize the integrity of Eqn. (6.20) since they all have 0 coefficients, and they are set as 0. To obtain the matrix format of Eqn. (6.19), we move superscript p of Bézier control points and Bézier extraction matrix to subscript for convenience. We have

$$\mathbf{x}^e = (\mathbf{P}^e)^T\mathbf{M}_p^e\mathbf{B}^p + (\mathbf{C}^e)^T\mathbf{M}_{p+1}^e\mathbf{B}^{p+1} = (\mathbf{Q}_p^e)^T\mathbf{B}^p + (\mathbf{Q}_{p+1}^e)^T\mathbf{B}^{p+1}. \tag{6.21}$$

With Eqn. (6.20), Eqn. (6.21) is converted to

$$\begin{aligned}\mathbf{x}^e &= (\mathbf{Q}_p^e)^T\mathbf{T}_p^{p+1}\mathbf{B}^{p+1} + (\mathbf{Q}_{p+1}^e)^T\mathbf{B}^{p+1} = (\mathbf{P}^e)^T\mathbf{M}_p^e\mathbf{T}_p^{p+1}\mathbf{B}^{p+1} + (\mathbf{C}^e)^T\mathbf{M}_{p+1}^e\mathbf{B}^{p+1} \\ &= ((\mathbf{P}^e)^T\mathbf{M}_p^e\mathbf{T}_p^{p+1} + (\mathbf{C}^e)^T\mathbf{M}_{p+1}^e)\mathbf{B}^{p+1} = (\mathbf{R}^e)^T\overline{\mathbf{M}}_{p+1}^e\mathbf{B}^{p+1},\end{aligned} \tag{6.22}$$

where

$$\mathbf{R}^e = \begin{bmatrix}\mathbf{P}^e \\ \mathbf{C}^e\end{bmatrix},\ \overline{\mathbf{M}}_{p+1}^e = \begin{bmatrix}\mathbf{M}_p^e\mathbf{T}_p^{p+1} \\ \mathbf{M}_{p+1}^e\end{bmatrix}. \tag{6.23}$$

$\mathbf{R}^e$ are the control points, $\overline{\mathbf{M}}^e_{p+1}$ is the Bézier extraction matrix, and $\mathbf{T}_p^{p+1}$ is obtained from Eqn. (6.20). Analogously we can get the same expression for Eqn. (6.18) when p is even, except that

$$\mathbf{R}^e = \begin{bmatrix} \mathbf{C}^e \\ \\ \mathbf{P}^e \end{bmatrix}. \tag{6.24}$$

As in Section 6.2.1, partition of unity is not satisfied necessarily. We use weighted T-splines in order to restore partition of unity, namely, $\overline{\mathbf{M}}^e_{p+1}$ is changed to $\widehat{\mathbf{M}}^e_{p+1}$ by normalizing each column of $\widehat{\mathbf{M}}^e_{p+1}$ with Eqn. (6.6). Thus the hybrid-degree weighted T-spline is defined as

$$\mathbf{x}^e = (\mathbf{R}^e)^T \widehat{\mathbf{M}}^e_{p+1} \mathbf{B}^{p+1}. \tag{6.25}$$

The calculated hybrid T-splines with different hybrid boundaries of $p = 3$ and $p = 2$ are given in Fig. 6.6(j - m).

With the developed way to introduce zero-interval length edges and define basis functions of degree $p+1$, the calculated T-spline surface is C^p-continuous at the p-refined region except the hybrid boundaries. It is C^{p-1}-continuous at the unchanged region, the transition region, and the hybrid boundaries.

Remark 6.2.2. For now, the degree difference of basis functions over the transition region is one. It is also possible to make the hybrid-degree weighted T-splines more general by introducing further p-refined basis functions to support the transition region, such as basis functions of degree $p+2$, or even higher. For the transition regions, the surface continuity is determined by the basis function with the lowest degree. The degree of extracted Bézier element is determined by the basis function with the highest degree.

Handling Extraordinary Nodes. Hybrid-degree weighted T-splines can also be used to perform local p-refinement to arbitrary topology surfaces with extraordinary nodes. Till now, the developed algorithms to deal with extraordinary nodes always use cubic basis functions [92, 81, 58]. Since extraordinary nodes influence their two-ring neighborhoods,

we can apply interior boundaries to enclose the two-ring neighborhood of each extraordinary node. Then beyond the hybrid boundaries, we define T-splines of $p = 2$ or $p = 4$. In this way, we can define hybrid T-spline surfaces of odd and even degrees on arbitrary shape topology. However, directly using even degree basis functions to deal with extraordinary nodes is still an open problem.

Remark 6.2.3. The main limitation of hybrid-degree weighted T-splines is that the geometry may be slightly changed after the local p-refinement. The reason is that we are defining basis functions of higher degree without sacrificing the surface continuity, and basis functions of different degree have support over the transition region. Zero-length interval edges are only introduced to the hybrid boundaries, not to all the faces in the p-refined region. If we introduce the zero-length interval edges to all the p-refined region, the surface change will only exist in the transition region. However this limitation will not bring trouble to analysis of 2D flat surfaces or 3D solids if the local p-refinement is performed such that the geometry boundary is not modified.

6.3 Results and Discussion

Here we first test hybrid-degree weighted T-splines with a common patch test of linear elasticity. Then, we use odd-, even- and hybrid-degree weighted T-splines to solve a classical benchmark problem that can be physically interpreted as a steady heat conduction. Finally, four other hybrid-degree weighted T-spline surfaces are given.

6.3.1 Isogeometric Analysis using Hybrid-Degree Weighted T-splines

We perform a patch test on a unit square. The Young's modulus is $E = 1.0$, and the Poisson's ratio is $\nu = 0.3$. Regarding the boundary conditions, the displacement in x direction is set to be 0 and 0.1 on the left and right boundaries, respectively. The displacement in y direction is set to be 0 on the bottom boundary and homogeneous Neumann boundary conditions are

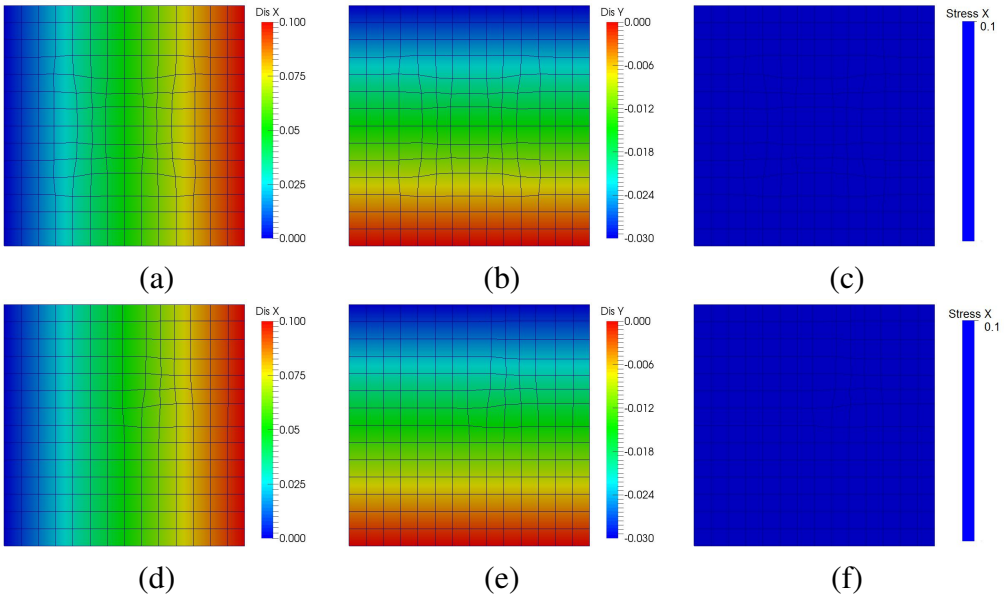


Figure 6.8: Patch test with weighted hybrid-degree T-spline with interior hybrid boundary (a-c) and extended hybrid boundary (d-f). (a, d) Displacement along x direction; (b, e) displacement in y direction; and (c, f) stress in x direction.

imposed elsewhere. Hybrid-degree weighted T-splines with interior and extended hybrid boundaries as shown in Fig. 6.6(a, c) are tested here. The results are shown in Fig. 6.8. We obtain linearly distributed displacement along x and y directions and uniform stress in x direction. The analysis results give the exact solution to the problem up to the machine precision.

The second problem solved here is a heat transfer problem defined over an L-shaped domain governed by the Laplace equation $\Delta u = 0$ with homogeneous Dirichlet boundary conditions on the re-entrant edges. Proper Neumann boundary conditions are imposed on the remaining boundaries so that the exact solution in polar coordinates (r,θ) is

$$u(r,\theta) = r^{2/3}\sin(2\theta/3), \;\; r > 0 \text{ and } 0 \le \theta \le 3\pi/2. \tag{6.26}$$

The problem setting is shown in Fig. 6.9(a). The domain is parameterized with locally h-refined T-splines of $p = 3$ and $p = 4$ first. Then the hybrid-degree weighted T-splines are

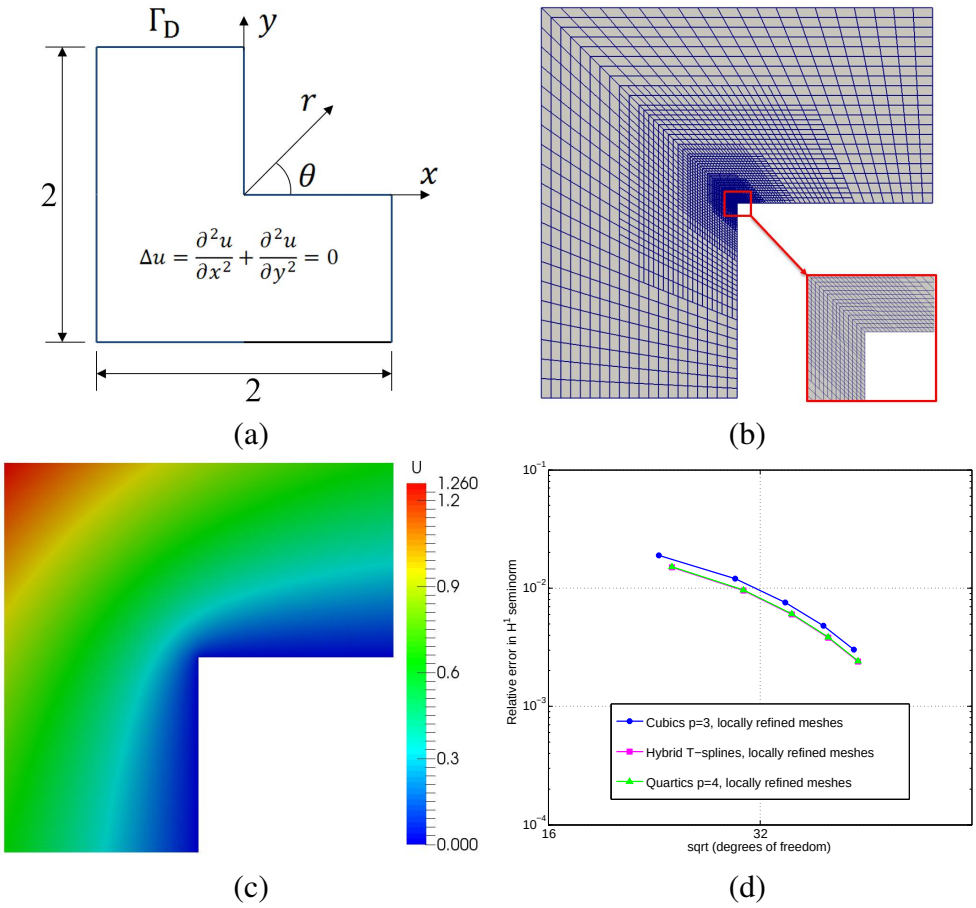
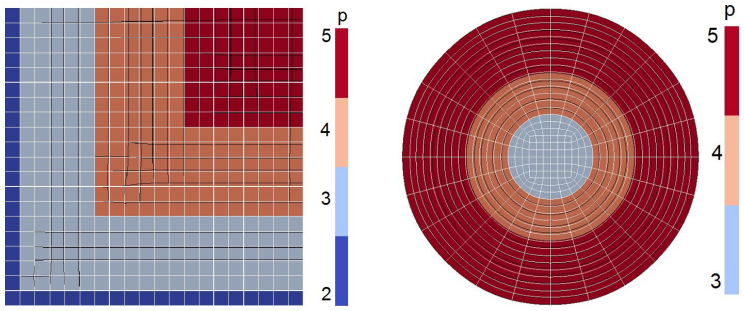


Figure 6.9: Isogeometric analysis of the Laplace equation over an L-shaped domain with re-entrant corner. (a) Geometry and problem settings; (b) Bézier elements with 4 levels of local h-refinement and local p-refinement with hybrid degrees $p = 3,4$ near the re-entrant corner; (c) solution field over the domain; (d) convergence curves of the three meshes.

obtained by performing local p-refinement to the T-spline with $p = 3$ near the re-entrant corner. Fig. 6.9(b) shows the refined meshes with hybrid degrees $p = 3,4$, respectively.

Fig. 6.9(c) shows the solution over the domain. Fig 6.9(d) shows the relative error of H^1 semi-norm with respect to the square root of degrees of freedom. With local p-refinement, hybrid-degree weighted T-splines perform better than T-splines of $p = 3$. Hybrid-degree weighted T-splines generate results as good as T-splines of $p = 4$. The reason is that the maximal error happens at the re-entrant corner where the first partial derivatives of the solution field have a singularity. Therefore, we extract Bézier elements of $p = 4$ from hybrid-degree weighted T-splines for the region close to the re-entrant corner in order to



(a) (b)

Figure 6.10: Open surface hybrid-degree weighted T-spline models. (a) A square model with extended hybrid boundaries and the difference before (white lines) and after refinement (black lines); and (b) a circle model with interior hybrid boundaries and the difference before (white lines) and after refinement (black lines).

improve the performance. In conclusion, local p-refinement gives additional flexibility to T-splines in order to enhance its efficiency in analysis.

Remark 6.3.1. Note that both h- and p-refinement are performed close to the singularity in the second problem. This is done in order to decrease the relative error of the H^1 seminorm which is concentrated near the singularity. However, in more complex settings, such as compressible gas dynamics, other hp-refinement strategies are known to perform better [31]. It is quite widespread to use local h-refinement where the solution is rough and to use local p-refinement in the rest of the domain where the solution is smooth.

6.3.2 Open and Closed Hybrid-Degree Weighted T-spline Surfaces

Here we show four hybrid-degree weighted T-spline surfaces, including two open surfaces as shown in Fig. 6.10 and two high genus closed T-spline surfaces as shown in Fig. 6.11.

A square model is shown in Fig. 6.10(a), with a combination of degrees 2, 3, 4 and 5 basis functions, marked with different colors. Extended hybrid boundaries exist in this model. Hybrid-degree weighted T-splines can combine basis functions of different degrees using different layers of transition regions. In Fig. 6.10(b), a circle model is provided with a combination of degrees 3, 4 and 5 basis functions. Interior hybrid boundaries exist in this model. There are 4 valance-3 extraordinary nodes. T-splines of $p = 3$ are used to handle

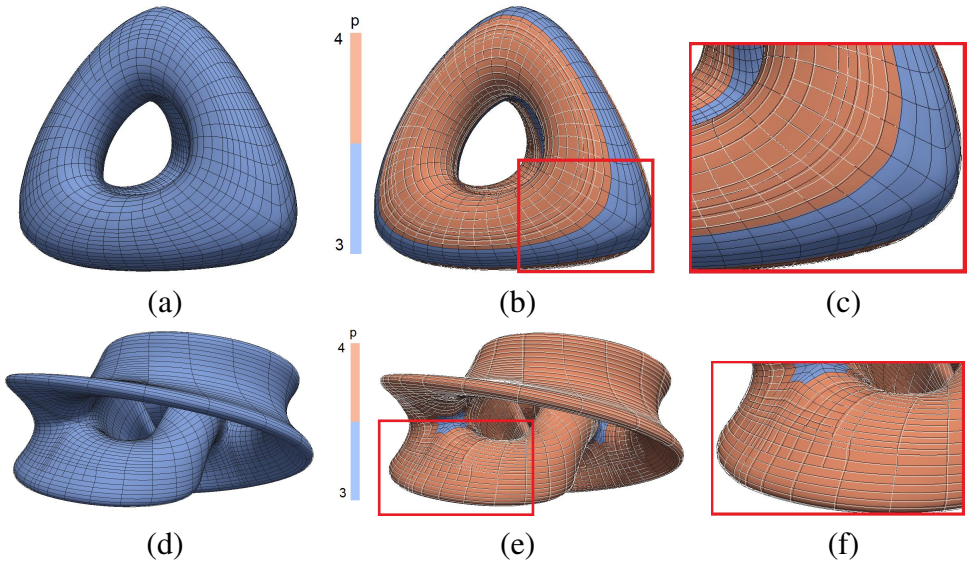


Figure 6.11: Tetra model (a-c) and Genus-three model (d-f) of hybrid-degree weighted T-splines. (a, d) Original model of degree 3; (b, e) local p-refinement results, where the white and black lines show the difference of Bézier elements before and after refinement; and (c, f) show the zoom-in of (b, e).

the extraordinary nodes. After local p-refinement with hybrid-degree weighted T-splines, the surface is changed along the radial direction near the hybrid boundaries. In Fig. 6.10, the white and black lines show the difference before and after local p-refinement. For open surface models, with the scheme to calculate the control points, the local region around the hybrid boundaries in the interior of the geometry is changed. The boundary of the geometry remains unchanged after refinement.

Two closed surface models with extraordinary nodes are given in Fig. 6.11. The Tetra model in Fig. 6.11(a) has eight valance-6 extraordinary nodes. The original T-spline surface is of degree 3. After local p-refinement, the hybrid-degree weighted T-spline is shown in Fig. 6.11(b), with the comparison before and after refinement. Fig. 6.11(d-f) shows the Genus-three model before and after local p-refinement. In this model, we set the two-ring neighborhood of the four valance-8 extraordinary nodes as degree 3. For all the other regions, the surface is of degree 4. With hybrid-degree weighted T-splines, we can generate T-spline surfaces with basis functions of different degrees in one model.

6.4 Conclusions and Future Work

In conclusion, in this chapter we develop a new algorithm to generate hybrid-degree weighted T-splines which can be used in isogeometric analysis. Weighted T-splines of arbitrary degree are proposed which satisfy all the analysis-suitable requirements. Hybrid-degree weighted T-splines are developed with two types of hybrid boundaries for local p-refinement. Both odd- and even-degree T-spline basis functions are defined. The p-refined regions and unchanged regions are connected via transition regions. Both p-refined basis functions and original basis functions have support over the transition regions, over which the hybrid-degree weighted T-splines are defined. The transition region has basis functions of the same degree as the p-refined region, and the same surface continuity as the unchanged region. Bézier elements of different degrees are extracted for isogeometric analysis. The Laplace equation is solved on an L-shaped domain, which is parameterized with odd-, even-, and hybrid-degree T-splines. Hybrid T-splines provide better performance after local p-refinement. In the future, we are planning to develop a better control point calculation algorithm to decrease the surface change after local p-refinement.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we develop algorithms for volumetric T-spline construction and arbitrary degree weighted T-spline modeling. Boolean operations and skeletons are used to generate polycubes for complicated volumetric T-spline models. Weighted T-splines are developed with less topological constraints to the T-mesh while meeting all the analysis-suitable requirements.

In the Boolean operation method, a harmonic field is computed to split the input geometry into hexahedral components. Two primitives, cube and torus, are used for the representation of components with cubes. With union and difference operations, the final polycube can be properly constructed. Then the T-mesh construction for volumetric T-splines is straight forward with the final polycubes. By introducing Boolean operations, we can construct T-splines in the same way with CAD design. Fewer extraordinary nodes are introduced. But there are certain types of models that it cannot handle, such as a tetrahedron or a cone. Another limitation of this algorithm is that we only use scaled Jacobian values to optimize the T-mesh, which would not improve the shape of the T-mesh.

Skeleton contains the extracted topological and geometrical information of 3D models, based on which the generated polycubes can better preserve geometric features from the input. Detailed algorithms have been developed to preserve different types of surface features. The constructed T-mesh contains all the surface information from the input, and transfers them to the final volumetric T-splines. Surface features are classified into three types: open curve, closed curve, and singularity features. Parametric mapping, volumetric parameterization, skeleton modification and templates are the developed algorithms to preserve them during T-mesh construction. The main drawback of this algorithm is that we have to perform pre-processing with the skeleton to make it valid for polycube construction. In addition, we cannot perform volumetric parameterization from frame field for the whole domain due to the limitations of singularity restricted field and flip-over issues discussed in [54]. 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.

Weighted T-splines are developed for T-spline surface and volume modeling with less geometrical constraints. It is proved to be analysis suitable. Weighted T-spline models can be easily constructed by simply performing quadtree/octree subdivision to the T-mesh. Both children basis functions and Bézier basis functions can be used to calculate the new weights of T-spline basis functions to enforce partition of unity. An edge interval extension algorithm is developed to reparameterize trimmed NURBS surfaces with T-splines. The reparameterization results can exactly reconstruct the trimming curves from the input, providing the foundation to generate water-tight T-spline surfaces from designed CAD models. An interval duplication method is developed to handle extraordinary nodes with better surface continuity. With parametric mapping and sweeping methods, a group of complicated volumetric T-splines can be generated.

Isogeometric collocation methods require at least C^2-continuous basis functions, and these methods show better performance with higher degree basis functions in isogeometric

analysis. Arbitrary degree T-spline modeling is studied to serve isogeometric analysis. The differences between even degree T-splines and odd degree T-splines are investigated in detail. Efficient data structure and knot interval extraction are developed to make arbitrary degree T-spline modeling possible. Localized hybrid T-splines are introduced, with weighted T-spline basis functions used to enforce partition of unity. Hybrid T-splines provide an elegant way to improve the analysis accuracy by performing local p-refinement. Surfaces with extraordinary nodes can also be handled with hybrid T-splines, which makes T-spline modeling more flexible for arbitrary degree basis functions.

Weighted T-splines have one limitation that the geometry is modified after recalculating the weighting coefficients. The surface change is introduced in reparameterizing trimmed NURBS surfaces, handling extraordinary nodes and calculating the transition regions. One promising solution to this problem is using truncated T-splines, which have released constraint on the T-mesh and do not change the geometry.

7.2 Future Work

Handling Extraordinary Nodes. For now the best surface continuity around extraordinary nodes is G^1. To use T-spline models in isogeometric collocation analysis, at least C^2-continuity is required. Developing new algorithms to handle extraordinary nodes with better surface continuity requires further study. Furthermore, in volumetric T-splines, the template method introduces redundant control points which not only decrease the continuity, but also result in ill-conditioned stiffness matrix for analysis. One possible solution is to extend the linear interpolation method developed in [81] to volumetric T-splines. However, due to the high complexity of the T-mesh topology around extraordinary nodes, further study is still required.

Volumetric T-spline Construction: Our volumetric T-spline construction algorithms depend on the input. If the input is partially changed, we have to reconstruct the T-splines

for the whole geometry. One interesting direction in volumetric T-spline construction is incremental modeling, which involves Boolean operations in-depth and only reconstructs required region when necessary. Besides, even though we can generate a group of volumetric T-spline models using the polycube and sweeping methods, it is still not possible to generate volumetric T-splines directly from complex CAD models. One reason is that the CAD models are not water-tight. Cleaning up the data from CAD files is always tedious and time consuming. One possible solution is to improve the design software and propose one new file format defined specifically for T-splines. *AutoCAD* and *Rhino* are two CAD software solutions that support T-splines. Popularizing T-spline in design industry and analysis needs more efforts from both research institutions and industrial companies.

Isogeometric Analysis Using T-splines: Isogeometric analysis using T-splines has been performed in different research fields. New algorithms have been developed for both geometrical modeling and numerical integration. One promising direction for isogeometric analysis is to apply it to dynamic problems. It can also be applied in adaptive manufacturing and 3D printing, which calls for integration of design and analysis to obtain optimal designed shape. This will also give rise to the challenges for new spline modeling techniques.

Bibliography

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

[2] 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.

[3] C. Anitescu, Y. Jia, Y. Zhang, and T. Rabczuk. An isogeometric collocation method using superconvergent points. *Computer Methods in Applied Mechanics and Engineering*, 284:1073–1097, 2015.

[4] A. E. Armenàkas. *Advanced mechanics of materials and applied elasticity*. CRC Press, 2005.

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

[6] F. Auricchio, L. Beirão Da Veiga, T. J. R. Hughes, A. Reali, and G. Sanglli. Isogeometric collocation methods. *Mathematical Models and Methods in Applied Sciences*, 20(11):2075–2107, 2010.

[7] P. L. Baehmann, S. L. Wittchen, M. S. Shephard, K. R. Grice, and M. A. Yerry. Robust, geometrically based, automatic two-dimensional mesh generation. *International Journal for Numerical Methods in Engineering*, 24(6):1043–1078, 1987.

[8] Y. Bazilevs, L. Beirão da Veiga, J. A. Cottrell, T. J. R. Hughes, and G. Sangalli. Isogeometric analysis: approximation, stability and error estimates for h-refined meshes. *Mathematical Models and Methods in Applied Sciences*, 16(07):1031–1090, 2006.

[9] 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.

[10] Y. Bazilevs, V. M. Calo, T. J. R. Hughes, and Y. Zhang. Isogeometric fluid-structure interaction: theory, algorithms, and computations. *Computational Mechanics*, 43(1):3–37, 2008.

[11] Y. Bazilevs, V. M. Calo, Y. Zhang, and T. J. R. Hughes. Isogeometric fluid-structure interaction analysis with applications to arterial blood flow. *Computational Mechanics*, 38(4-5):310–322, 2006.

[12] Y. Bazilevs, M.-C. Hsu, and M. A. Scott. Isogeometric fluid-structure interaction analysis with emphasis on non-matching discretizations, and with application to wind turbines. *Computer Methods in Applied Mechanics and Engineering*, 249-252(0):28–41, 2012.

[13] Y. Bazilevs and T. J. R. Hughes. NURBS-based isogeometric analysis for the computation of flows about rotating components. *Computational Mechanics*, 43(1):143–150, 2008.

[14] L. Beirão da Veiga, A. Buffa, G. Sangalli, and R. Vázquez. Analysis-suitable T-splines of arbitrary degree: definition, linear independence and approximation properties. *Mathematical Models and Methods in Applied Sciences*, 23(11):1979–2003, 2013.

[15] A. Bressan, A. Buffa, and G. Sangalli. Characterization of analysis-suitable T-splines. *Computer Aided Geometric Design (2015) doi:10.1016/j.cagd.2015.06.007.*, 2015.

[16] A. Buffa, D. Cho, and G. Sangalli. Linear independence of the T-spline blending functions associated with some particular T-meshes. *Computer Methods in Applied Mechanics and Engineering*, 199(23-24):1437–1445, 2010.

[17] H. Casquero, L. Liu, C.s Bona-Casas, Y. Zhang, and H. Gomez. A hybrid variational-collocation immersed method for fluid-structure interaction using unstructured T-splines. *International Journal for Numerical Methods in Engineering (2015) doi:10.1002/nme.5004.*, 2015.

[18] H. Casquero, L. Liu, Y. Zhang, A. Reali, and H. Gomez. Isogeometric collocation using analysis-suitable T-splines of arbitrary degree. *submitted, 2015.*, 2015.

[19] E. Cohen, T. Lyche, and L. L. Schumaker. Algorithms for degree-raising of splines. *ACM Transactions on Graphics*, 4(3):171–181, 1985.

[20] E. Cohen, T. Lyche, and L. L. Schumaker. Degree raising for splines. *Journal of Approximation Theory*, 46(2):170–181, 1986.

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

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

[23] 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.

[24] R. Dimitri, L. De Lorenzis, M. A. Scott, P. Wriggers, R. L. Taylor, and G. Zavarise. Isogeometric large deformation frictionless contact using T-splines. *Computer Methods in Applied Mechanics and Engineering*, 269(0):394–414, 2014.

[25] M. R. Dörfel, B. Jüttler, and B. Simeon. Adaptive isogeometric analysis by local h-refinement with T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199(58):264–275, 2010.

[26] 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.

[27] E. J. Evans, M. A. Scott, X. Li, and D. C. Thomas. Hierarchical T-splines: Analysis-suitability, Bézier extraction, and application as an adaptive basis for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 284:1–20, 2015.

[28] J. A. Evans, Y. Bazilevs, I. Babuška, and T. J. R. Hughes. *n*-Widths, sup-infs, and optimality ratios for the *k*-version of the isogeometric finite element method. *Computer Methods in Applied Mechanics and Engineering*, 198(2126):1726–1741, 2009.

[29] G. T. Finnigan. Arbitrary degree T-splines. Master's thesis, Brigham Yong University, 2008.

[30] M. S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231–250, 1997.

[31] S. Giani and P. Houston. Anisotropic *hp*-adaptive discontinuous galerkin finite element methods for compressible fluid flows. *International Journal of Numerical Analysis and Modeling*, 9(4):928–949, 2012.

[32] C. Giannelli, B. Jüttler, and H. Speleers. THB-splines: the truncated basis for hierarchical splines. *Computer Aided Geometric Design*, 29:485–498, 2012.

[33] C. Giannelli, B. Jüttler, and H. Speleers. Strongly stable bases for adaptively refined multilevel spline spaces. *Advances in Computational Mathematics*, 40(2):459–490, 2014.

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

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

[36] C. A.R. Guerra. Simultaneous untangling and smoothing of hexahedral meshes. Master's thesis, University PolyTècnica De Catalunya, Barcelona Spain, 2010.

[37] A. Hatcher. Pants decomposition of surfaces. *arXiv:math/9906084*, 1999.

[38] J. Huang, T. Jiang, Z. Shi, Y. Tong, H. Bao, and M. Desbrun. L_1-based construction of polycube maps from complex shapes. *ACM Transactions on Graphics*, to appear, 2014.

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

[40] 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.

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

[42] B. Jüttler, M. K., D.-M. Nguyen, Q. Pan, and M. Pauley. Isogeometric segmentation: The case of contractible solids without non-convex edges. *Computer-Aided Design*, 57(0):74 – 90, 2014.

[43] H.-J. Kim, Y.-D. Seo, and S.-K. Youn. Isogeometric analysis for trimmed CAD surfaces. *Computer Methods in Applied Mechanics and Engineering*, 198(3740):2982 – 2995, 2009.

[44] H.-J. Kim, Y.-D. Seo, and S.-K. Youn. Isogeometric analysis with trimming technique for problems of arbitrary complex topology. *Computer Methods in Applied Mechanics and Engineering*, 199(4548):2796 – 2812, 2010.

[45] Y. Lai, L. Liu, Y. Zhang, J. Chen, E. Fang, and J. Lua. Rhino 3D to Abaqus design-through-analysis: A T-spline based IGA software platform. In Y. Bazilevs and K. Takizawa, editors, *The edited volume of the Modeling and Simulation in Science, Engineering and Technology Book Series devoted to AFSI 2014 - a birthday celebration conference for Tayfun Tezduyar*. Springer, 2015.

[46] B. Li, X. Li, K. Wang, and H. Qin. Generalized polycube trivariate splines. In *Shape Modeling International Conference*, pages 261–265, 2010.

[47] 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*, 99:1539–1551, 2013.

[48] 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.

[49] X. Li. Some properties for analysis-suitable T-splines. *Journal of Computational Mathematics*, 33(4):428–442, 2015.

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

[51] X. Li and M. A. Scott. Analysis-suitable T-splines: characterization, refineability, and approximation. *Mathematical Models and Methods in Applied Sciences*, 24(06):1141–1164, 2014.

[52] X. Li, H. Xu, S. Wan, Z. Yin, and W. Yu. Feature-aligned harmonic volumetric mapping using MFS. *Computers & Graphics*, 34(3):242 – 251, 2010.

[53] X. Li, J. Zheng, T. W. Sederberg, T. J. R. Hughes, and M. A. Scott. On linear independence of T-spline blending functions. *Computer Aided Geometric Design*, 29(1):63–76, 2012.

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

[55] 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.

[56] L. Liu, Y. Zhang, T. J. R. Hughes, M. A. Scott, and T.W. Sederberg. Volumetric T-spline construction using Boolean Operations. *Engineering with Computers*, 30(4):425–439, 2014.

[57] L. Liu, Y. Zhang, and X. Wei. Weighted T-splines with application in reparameterizing trimmed NURBS surfaces. *Computer Methods in Applied Mechanics and Engineering*, 295:108–126, 2015.

[58] L. Liu, Y. Zhang, and X. Wei. Handling extraordinary nodes with weighted T-spline basis functions. In *24th International Meshing Roundtable*, accepted, 2015.

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

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

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

[62] S. A. Mitchell and T. J. Tautges. Pillowing doublets: refining a mesh to ensure that faces share at most one edge. In *4th International Meshing Roundtable*, pages 231–240, 1995.

[63] P. Morgenstern. 3D Analysis-suitable T-splines: definition, linear independence and m-graded local refinement (2015). *ArXiv e-prints*, 2015.

[64] P. Morgenstern and D. Peterseim. Analysis-suitable adaptive T-mesh refinement with linear complexity. *Computer Aided Geometric Design*, 34(0):50–66, 2015.

[65] D.-M. Nguyen, M. Pauley, and B. Jüttler. Isogeometric segmentation. part II: On the segmentability of contractible solids with non-convex edges. *Graphical Models*, 76(5):426 – 439, 2014.

[66] N. Nguyen-Thanh, H. Nguyen-Xuan, S.P.A. Bordas, and T. Rabczuk. Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids. *Computer Methods in Applied Mechanics and Engineering*, 200(2122):1892 – 1908, 2011.

[67] M. Nieser, U. Reitebuch, and K. Polthier. CubeCover-parameterization of 3D volumes. *Computer Graphics Forum*, 30(5):1397–1406, 2011.

[68] L. Piegl and W. Tiller. Software-engineering approach to degree elevation of B-spline curves. *Computer-Aided Design*, 26(1):17–28, 1994.

[69] L. A. Piegl and W. Tiller. *The NURBS Book, 2nd ed.* Springer-Verlag, New York, 1997.

[70] H. Prautzsch. Degree elevation of B-spline curves. *Computer Aided Geometric Design*, 1(2):193–198, 1984.

[71] H. Prautzsch and B. Piper. A fast algorithm to raise the degree of spline curves. *Computer Aided Geometric Design*, 8(4):253–265, 1991.

[72] M. A. Price and 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):111–136, 1997.

[73] M. A. Price, C. G. Armstrong, and 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):3335–3359, 1995.

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

[75] J. Qian, Y. Zhang, W. Wang, A. C. Lewis, M. A. 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.

[76] D. Schillinger, J. A. Evans, A. Reali, M. A. Scott, and T. J. R. Hughes. Isogeometric collocation: cost comparison with Galerkin methods and extension to adaptive hierarchical NURBS discretizations. *Computer Methods in Applied Mechanics and Engineering*, 267(0):170–232, 2013.

[77] R. Schneiders. A grid-based algorithm for the generation of hexahedral element meshes. *Engineering with Computers*, 12:168–177, 1996.

[78] 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.

[79] 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.

[80] M. A. Scott, X. Li, T. W. Sederberg, and T. J. R. Hughes. Local refinement of analysis-suitable T-splines. *Computer Methods in Applied Mechanics and Engineering*, 213-216(0):206–222, 2012.

[81] M. A. Scott, R. N. Simpson, J. A. Evans, S. Lipton, S. P. A. Bordas, T. J. R. Hughes, and T. W. Sederberg. Isogeometric boundary element analysis using unstructured T-splines. *Computer Methods in Applied Mechanics and Engineering*, 254(0):197–221, 2013.

[82] T. Sederberg. Bicubic capping with arbitrary knot intervals. *Research Notes*, 2008.

[83] 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.

[84] T. W. Sederberg, G. T. Finnigan, X. Li, H. Lin, and H. Ipson. Watertight trimmed NURBS. *ACM Transactions on Graphics*, 27(3):79:1–79:8, 2008.

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

[86] A. Tagliasacchi, I. Alhashim, M. Olson, and H. Zhang. Mean curvature skeletons. In *Computer Graphics Forum*, volume 31, pages 1735–1744. Wiley Online Library, 2012.

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

[88] H. Wang, Y. He, X. Li, X. Gu, and H. Qin. Polycube splines. In *Symposium on Solid and Physical Modeling*, pages 241–251, 2007.

[89] 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.

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

[91] 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.

[92] 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.

[93] Y. Wang and J. Zheng. Control point removal algorithms for T-spline surfaces. *Lecture Notes in Computer Science*, 4077:385–396, 2006.

[94] X. Wei, Y. Zhang, T. J. R. Hughes, and M. A. Scott. Truncated hierarchical Catmull-Clark surface with local refinement. *Computer Methods in Applied Mechanics and Engineering*, 291:1–20, 2015.

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

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

[97] 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.

[98] 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.

[99] 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):1130–1141, 2012.

[100] Y. Zhang, X. Liang, and 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*, 256:562–576, 2013.

[101] 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.

[102] Y. Zhang, W. Wang, and T. J. R. Hughes. Conformal solid T-spline construction from boundary T-spline representations. *Computational Mechanics*, 51(6):1051–1059, 2013.

[103] J. Zheng, G. Wang, and Y. Liang. Curvature continuity between adjacent rational Bézier patches. *Computer Aided Geometric Design*, 9(5):321–335, 1992.