Carnegie Mellon University

CARNEGIE INSTITUTE OF TECHNOLOGY

THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF **Doctor of Philosophy**

TITLE: Centroidal Voronoi Tessellation with Applications in Image and Mesh Processing

PRESENTED BY: Kangkang Hu

ACCEPTED BY THE DEPARTMENT OF : Mechanical Engineering

APPROVED BY THE COLLEGE COUNCIL :

Centroidal Voronoi Tessellation with Applications in Image and Mesh Processing

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Department of Mechanical Engineering

Kangkang Hu

B.S., Material Science and Engineering, Huazhong University of Science and Technology

M.S., Mechanical Engineering, Shanghai Jiao Tong University

Carnegie Mellon University

Pittsburgh, PA

July 2016

Abstract

Centroidal Voronoi tessellation (CVT) is a clustering technique by partitioning a set of discrete data points into non-overlapping clusters, where the generators of the tessellations are also the centroids of the corresponding Voronoi regions. It has been a powerful tool in many applications ranging from data analysis, numerical partial differential equations, and geometric modeling. In order to extend its applications in both image and mesh processing, in this thesis we present: (a) a harmonic edge-weighted CVT (HEWCVT) based algorithm for image segmentation and adaptive superpixel generation; (b) two algorithms for surface segmentation and polycube construction via the harmonic boundary-enhanced CVT (HBECVT); (c) a feature-aligned surface parameterization algorithm using Loop subdivision and secondary Laplace operator (SLO); and (d) a CVT based 3D image segmentation algorithm for tetrahedral/hexahedral meshing and quality improvement using anisotropic diffusion flow.

First, we develop the HEWCVT model for image segmentation by introducing a harmonic form of clustering energy which combines the image intensity with cluster boundary information. Compared to other CVT-based methods, the HEWCVT algorithm can not only overcome the sensitivity to the seed point initialization and noise, but also improve the accuracy and stability of clustering results, as verified in several types of images. We then present an adaptive superpixel generation algorithm based on HEWCVT. First, an innovative initial seed sampling method based on quadtree decomposition is introduced, and the image is divided into small adaptive segments according to a density function. Then, the local HEWCVT algorithm is applied to generate adaptive superpixels.

We then develop CVT-based surface segmentation algorithms for polycube construction and hexahedral mesh generation. Given a smooth surface triangle mesh, we segment triangles into six clusters in the surface normal space while satisfying the constraints of polycube construction. A bijective mapping between the input mesh and polycube surfaces is then built via a planar domain parameterization. Inspired by the HEWCVT model for image segmentation, we develop the HBECVT method by including local neighbouring information in the energy function. Improving upon the classic CVT method, the HBECVT algorithm can not only overcome the sensitivity to the initialization and noise, but also improve the segmentation results and the resulting polycubes by reducing non-monotone boundaries. Based on the constructed polycube, we then generate quality all-hexahedral (all-hex) meshes. Uniform all-hex meshes and volumetric T-meshes can be obtained through the octree subdivision and mapping. We can also generate adaptive all-hex meshes by extracting the dual mesh from a hybrid octree, which consists of polyhedral cells and each grid point is always shared by eight cells.

As a follow-up, we develop a new two-step surface segmentation scheme for polycube construction using generalized CVTs. In the first step, eigenfunctions of the SLO are coupled with the HBECVT model to classify vertices of the surface into several components based on concave creases and convex ridges of an object. Neighbouring vertex information is incorporated into the clustering energy function to avoid over-segmentation, jaggy boundaries and noise effect. For each segmented component, in the second step we apply the skeleton information to define local coordinates and include them into the HBECVT model to further segment it into several patches, which are revised using predefined geometric constraints for valid polycube construction. Our skeleton-based CVT algorithm is suitable for slim cylindrical objects and can reduce unnecessary singularities with compact polycube structures. Based on the constructed polycubes, we generate quality all-hex meshes and volumetric T-meshes via parametric mapping.

However, the computation of the above SLO eigenfunctions utilizes quadrilateral control meshes with Catmull-Clark basis functions, which limits its application on triangle meshes. To address this issue, we calculate the eigenfunctions of SLO on triangle meshes by using Loop subdivision basis functions. Multiple modes are used for CVT-based surface segmentation and boundaries of the segmented regions are extracted as the feature lines which contain concave creases and convex ridges. For each boundary edge, its two adjacent triangles are used as the guidance for the cross field construction. A constrained surface parameterization is then computed, where feature lines are preserved and aligned to the parametric lines. We also apply our algorithm to generate T-meshes, truncated T-splines and weighted T-splines for isogeometric analysis applications.

Finally, we extend the HEWCVT algorithm to 3D image segmentation for multi-material tetrahedral/hexahedral mesh generation and quality improvement. Given an input 3D image, we first segment voxels into several clusters by eliminating the noise effect. The Dual Contouring method is then applied to construct multi-material tetrahedral meshes by analyzing both material change edges and interior edges. Hexahedral meshes can also be generated by analyzing each interior grid point. For boundary vertices, we develop an anisotropic Giaquinta-Hildebrandt operator (GHO) based geometric flow method to smooth the surface and improve the mesh quality with feature preservation. For interior vertices, we improve the aspect ratio by using optimization-based smoothing and topological optimizations.

In summary, we develop CVT-based methods for image and mesh segmentation. Based on the surface segmentation results, we generate polycubes and all-hex meshes with good quality. We also combine it with geometric operators for surface parameterization and mesh quality improvement, with concave/convex features preserved.

Keywords: Centroidal Voronoi Tesselllation, Image Segmentation, Polycube, All-Hexahedral Mesh, Surface Parameterization, Geometric Operators, Loop Subdivision, Geometric Flow

Acknowledgements

First of all, my deepest gratitude goes first and foremost to my advisor Dr. Yongjie Jessica Zhang for her constant encouragement, support, guidance, and inspiration. She has walked me through all the stages in the course of doctoral study. Without her invaluable advices and illuminating instruction in all aspects, this thesis could not have reached its present form, for which I am extremely grateful.

Furthermore, I would like to express my heartfelt gratitude to my committee, Dr. Kenji Shimada, Dr. Levent Burak Kara, and Dr. John Andrew Evans for their support and insightful comments.

I am also greatly indebted to the members of our research group, Dr. Wenyan Wang, Dr. Xinghua Liang, Dr. Jin Qian, Dr. Tao Liao, Dr. Lei Liu, 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 owe my sincere gratitude to our collaborators Xinge Li and Dr. Guoliang Xu for the happy collaboration which has broaden my research perspectives.

Last my thanks would go to my beloved family for their loving considerations, infinite support and unconditional faith in me throughout my life. I hope that this work makes them be proud of.

This work was supported by Y. Zhang's PECASE award N00014-16-1-2254, NSF CAREER Award OCI-1149591 and CIT Dean's Fellowship.

To My Family.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Image segmentation has been one of the core topics in computer vision and image processing for decades. Its central task is to partition an image into subsets of pixels that share similar characteristics, such as color, brightness or texture. The success of most tasks requiring image analysis is often a direct consequence of the success of segmentation. As an over-segmentation of the image, superpixels have also become popular as they provide an efficient preprocessing tool for various computer vision applications. A superpixel is defined as a homogeneous image region that aligns well with object boundaries. It has been shown that using superpixels is advantageous because they can preserve natural image boundaries and reduce redundant information of the image data as well as enforce local consistency. Although there have been many algorithms developed for image segmentation and superpixel generation, there are still big obstacles in generating correct and stable segmentation results. Also, it is still challenging to efficiently generate inhomogeneous, i.e. adaptive in size, and compact superpixels. In this thesis, we develop the harmonic edge-weighted centroidal Voronoi tessellation (HEWCVT) model by introducing a harmonic form of clustering energy to generate stable results.

Unstructured all-hexahedral (all-hex) mesh generation for complex 3D domains plays an important role in many finite element simulations. All-hex meshes often outperform tetrahedral meshes by increased simulation accuracy, smaller element counts and improved reliability. However, generating adaptive all-hex meshes with high quality while preserving a conformal boundary is still challenging. Volumetric polycube is an appealing approach for all-hex meshing, which converts a volume to an all-hex mesh by first embedding axis-aligned boxes into the 3D volume and then mapping the volume to several jointed boxes. Given an input surface triangle mesh, the polycube construction can be viewed as a mesh segmentation task which assigns each triangle to one of axis-aligned planes in the parametric domain. With several connectivity constraints, the resulting segmentation leads to a correct polycube structure. To address these issues, we develop the harmonic boundary-enhanced centroidal Voronoi tessellation (HBECVT) method, which extends the CVT-based image segmentation to mesh segmentation for automatic polycube construction.

However, the HBECVT based method segments the surface according to the normals defined in the global coordinate system, it may result in large number of unnecessary singularities (polycube corners) for slim cylindrical surfaces, such as blood vessels. It is still challenging to generate compact polycubes with limited number of singularities and low distortion for complex geometry. Improving upon Laplace-Beltrami operator (LBO), the secondary Laplace operator (SLO) [67] was developed recently based on the second fundamental form of the surface. Curvature related surface features, such as concave creases and convex ridges, can be captured by its eigenfunctions. In this thesis, we develop a new two-step surface segmentation scheme for polycube construction using generalized centroidal Voronoi tessellation (CVT). In the first step, vertices of the surface are classified into several components by extending the HBECVT algorithm to the SLO eigenfunction space. In the second step, a novel skeleton-based CVT model is presented by using local coordinates to define the generators flexibly in the normal space.

Surface parameterization is of great importance for many applications, such as quadrangulation, texture mapping and surface fitting. An important issue for surface parameterization is how to align parametric lines with the feature directions. In this thesis, we develop a constrained surface parametrization, where feature lines are preserved and aligned to the parametric lines. The computation of the SLO eigenfunctions in [67] utilizes quadrilateral control meshes with Catmull-Clark basis functions, which limits its application on triangle meshes. To solve this problem, we calculate the eigenfunctions of SLO on triangle meshes by using Loop subdivision basis functions. Eigenfunction are used for CVT-based surface segmentation and boundaries of the segmented regions are extracted as the feature lines which contain concave/convex edges. For each edge of the feature lines, its two adjacent triangles are used as the guidance for the cross field construction.

Although many image segmentation methods have been developed, it is still challenging to generate finite element meshes directly from images bridging image segmentation and mesh generation. To address this issue, we extend the HEWCVT algorithm to 3D image segmentation for tetrahedral and hexahedral mesh generation. Mesh smoothing is a powerful tool for denoising and quality improvement. Although there already exist a variety of mesh denoising methods, research on feature preserving denoising remains active due to its challenging nature. In this thesis, we develop an anisotropic Giaquinta-Hildebrandt operator (GHO) flow for surface smoothing and quality improvement. Since GHO is defined based on the second fundamental form of the surface, it is more sensitive to the curvature-related features. The anisotropic scheme preserves surface features while removing the noise with an anisotropic diffusion weighting function.

1.2 Problem Statement

In this thesis, several CVT-based algorithms are developed for image and mesh processing. There are five main problems:

- **Image segmentation and adaptive superpixel generation based on harmonic edge-weighted centroidal Voronoi tessellation.** We extend the basic edge-weighted centroidal Voronoi tessellation (EWCVT) for image segmentation to a new advanced model, namely harmonic edge-weighted centroidal Voronoi tessellation (HEWCVT). This extended model introduces a harmonic form of clustering energy by combining the image intensity with cluster boundary information. Improving upon the classic CVT and EWCVT methods, the HEWCVT algorithm can not only overcome the sensitivity to the seed point initialization and noise, but also improve the accuracy and stability of clustering results, as verified in several types of images. We then present an adaptive superpixel generation algorithm based on HEWCVT. First, an innovative initial seed sampling method based on quadtree decomposition is introduced, and the image is divided into small adaptive segments according to a density function. Then, the local HEWCVT algorithm is applied to generate adaptive superpixels. The presented algorithm is capable of generating adaptive superpixels while preserving local image features efficiently.

- **Centroidal Voronoi tessellation based polycube construction for adaptive all-hexahedral mesh generation.** Given a smooth surface triangle mesh, we segment triangles into six clusters in the surface normal space while satisfying the constraints of polycube construction. A bijective mapping between the input mesh and polycube surfaces is then built via a planar domain parameterization. We develop a new harmonic boundary-enhanced centroidal Voronoi tessellation (HBECVT) method by including local neighbouring information in the energy function. Improving upon the classic CVT method, the HBECVT algorithm can not only overcome the sensitivity to the initialization and noise, but also improve the segmentation results and the resulting polycubes by reducing non-monotone boundaries. Based on the constructed polycube, we then generate quality all-hexahedral (all-hex) meshes. The uniform all-hex mesh and volumetric T-mesh can be obtained through the octree subdivision and mapping.

We can also generate adaptive all-hex meshes by extracting the dual mesh from a hybrid octree, which consists of polyhedral cells and each grid point is always shared by eight cells.

- **Surface segmentation for polycube construction based on generalized centroidal Voronoi tessellation.** Given an input triangle mesh, we first apply the two-step surface segmentation scheme (eigenfunction-based CVT and skeleton-based CVT) to split the surface domain, and then build valid polycube for all-hex mesh and volumetric T-mesh generation. In the first step, eigenfunctions of the secondary Laplace operator (SLO) are coupled with the harmonic boundary-enhanced CVT (HBECVT) model to classify vertices of the surface into several components based on concave creases and convex ridges of an object. Neighbouring vertex information is incorporated into the clustering energy function to avoid over-segmentation, jaggy boundaries and noise effect. For each segmented component, in the second step we apply the skeleton information to define local coordinates and include them into the HBECVT model to further segment it into several patches, which are revised using predefined geometric constraints for valid polycube construction. Our skeleton-based CVT algorithm is suitable for slim cylindrical objects and can reduce unnecessary singularities with compact polycube structures. Based on the constructed polycube, we generate quality all-hexahedral meshes and volumetric T-meshes via parametric mapping.

- **Feature preservation in surface parameterization using secondary Laplace operator and Loop subdivision.** We introduce a CVT-based surface segmentation method using LBO/SLO to extract surface features for input triangle mesh, and apply the cross field based method to generate a feature-aligned surface parameterization. The computation of the LBO/SLO eigenfunctions in [67] utilizes quadrilateral control meshes with Catmull-Clark basis functions, which limits its application on triangle meshes. To solve this problem, we introduce the Loop subdivision basis functions to calculate the eigenfunctions of LBO/SLO on triangle meshes. Eigenfunctions are used for CVT-based surface segmentation and the boundaries of the segmentation are extracted as the feature lines which contain concave/convex edges. A constrained surface parameterization is then computed, where feature lines are preserved and aligned to parametric lines. We also apply our algorithm to generate T-meshes and T-spline surfaces for IGA application.

- **CVT-based 3D image segmentation and quality improvement of tetrahedral/hexahedral meshes using anisotropic Giaquinta-Hildebrandt operator.** Given an input 3D image with multi-material parts, we first segment it into several clusters by using the harmonic Edge-Weighted CVT (HEWCVT) method. The Dual Contouring method is then applied to construct segmented tetrahedral meshes by analysing both material change edges and interior edges. Hexahedral meshes can also be generated by analyzing each interior grid point. An anisotropic Giaquinta-Hildebrandt operator (GHO) based geometric flow method is developed to smooth the surface and improve the quality of triangle/tetrahedral meshes, which preserves the volume and surface features. Optimization based smoothing and local connectivity modifications such as face swapping, edge removal and pillowing are then applied to improve the quality of the interior mesh.

1.3 Contributions

This thesis will have five main contributions:

- Develop a harmonic edge-weighted centroidal Voronoi tessellation for image segmentation and superpixels generation;
- Develop a centroidal Voronoi tessellation based polycube construction algorithm for adaptive all-hexahedral mesh generation;

- Develop a surface segmentation algorithm for polycube construction based on generalized centroidal Voronoi tessellation;
- Develop a feature-aligned surface parameterization algorithm using secondary Laplace operator based on Loop subdivision;
- Develop a 3D image segmentation algorithm for tetrahedral/hexahedral mesh generation with quality improvement via geometric flow.

1.4 Publication

Journal publications:

- K. Hu, Y. J. Zhang, G. Xu. "CVT-Based 3D Image Segmentation and Quality Improvement of Tetrahedral/Hexahedral Meshes Using Anisotropic Giaquinta-Hildebrandt Operator". *Computer Methods in Biomechanics and Biomedical Engineering*. Submitted, 2016.
- K. Hu, Y. J. Zhang, X. Li, G. Xu. "Feature Preservation in Surface Parameterization Using Secondary Laplace Operator and Loop Subdivision". *Computer Methods in Applied Mechanics and Engineering*. Submitted, 2016.
- K. Hu, Y. J. Zhang, T. Liao. "Surface Segmentation for Polycube Construction Based on Generalized Centroidal Voronoi Tessellation". *Computer Methods in Applied Mechanics and Engineering*, 2016. DOI: 10.1016/j.cma.2016.07.005.
- K. Hu, Y. J. Zhang. "Centroidal Voronoi Tessellation Based Polycube Construction for Adaptive All-Hexahedral Mesh Generation". *Computer Methods in Applied Mechanics and Engineering*. 305:405-421, 2016.
- K. Hu, Y. J. Zhang. "Image Segmentation and Adaptive Superpixel Generation Based on Harmonic Edge-Weighted Centroidal Voronoi Tessellation". *The Special Issue of*

CompIMAGE'14 in Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization. 4(2):46-60, 2016.

Conference publications:

- K. Hu, Y. J. Zhang, X. Li, G. Xu. "Feature-Aligned Surface Parameterization Using Secondary Laplace Operator and Loop Subdivision". *25th International Meshing Roundtable*. Washington, DC. Sept. 27-30, 2016.
- K. Hu, Y. J. Zhang, G. Xu. "CVT-based 3D Image Segmentation for Quality Tetrahedral Meshing". *CompIMAGE (Computer Modeling of Objects Presented in Images: Fundamentals, Methods, and Applications)*. Niagara Falls, USA. Sept. 21-23, 2016.
- K. Hu, Y. J. Zhang. "Surface Segmentation and Polycube Construction Based on Generalized Centroidal Voronoi Tessellation". *24th International Meshing Roundtable*. Austin, TX. Oct. 12-14, 2015. Research Notes.
- K. Hu, Y. J. Zhang. "Extended Edge-Weighted Centroidal Voronoi Tessellation for Image Segmentation". *CompIMAGE (Computer Modeling of Objects Presented in Images: Fundamentals, Methods, and Applications)*. 8461:164-175, 2014.
- K. Hu, J. Qian, Y. Zhang. "Adaptive All-Hexahedral Mesh Generation Based on A Hybrid Octree and Bubble Packing". *22nd International Meshing Roundtable*. Orlando, FL. Oct. 13-16, 2013. Research Notes.

1.5 Outline of Dissertation

Following the introduction, Chapter 2 reviews related previous work. Chapter 3 talks about the image segmentation and superpixels generation algorithm using the HEWCVT method. Chapter 4 discusses the HBECVT based polycube construction for adaptive all-hex mesh generation. Chapter 5 presents the two-step surface segmentation algorithm for polycube construction based on generalized CVT. Chapter 6 shows our approach to generate surface parameterization using the SLO based on Loop subdivision. Chapter 7 illustrates tetrahedral/hexahedral mesh generation from 3D images and quality improvement via geometric flow.

Chapter 2

Literature Review

2.1 Centroidal Voronoi Tessellation

CVT-based model is essentially a clustering technique by partitioning a set of discrete data points into non-overlapping clusters, where points belonging to the same cluster have maximum homogeneity by certain measure of proximity. After defining L generators, all the data points can be partitioned into L clusters by assigning each point to its nearest generator. Then the algorithm iteratively updates the generators to be the centroids of their associated clusters and starts a new partition with new generators. Such an algorithm aims at minimizing an energy function E which measures how tightly each cluster is packed until certain criterion is met. CVTs have many applications [25, 28, 54], including image and data compression, mesh generation and optimization, multi-dimensional integration, and partial differential equations. The classic CVT algorithm for image segmentation and compression applications was proposed in [27], which is sensitive to noise and may fail to provide accurate segmentation results. To address this issue, an edge-weighted centroidal Voronoi tessellation (EWCVT) model was proposed in [131], which greatly improves upon the classic CVT by adding an edge energy term in the energy functional. However, the classic CVT and EWCVT methods are still sensitive to seed point initialization, which may result in incorrect and unstable segmentation results. In this thesis, we extend the EWCVT for image segmentation to a new advanced algorithm named harmonic edge-weighted CVT (HEWCVT), which overcomes the sensitivity to the initialization and noise, and improves the accuracy and stability of segmentation results.

Cohen-Steiner *et al.* [21] extended the concept of planar proxies to drive the distortion error down, and used a flooding scheme to compute the constrained CVT as a good geometric approximation of surfaces. A metric-driven Discrete Voronoi Diagram [126] construction algorithm was proposed to directly compute the constrained CVT as clusters of triangles, which can be used for 3D surface remeshing. The CVT for line segments and graphs [75] was computed, which can be used to get meaningful segmentations of 3D models. The concept of CVT was further extended from the Euclidean space to spherical and hyperbolic spaces [106], called the universal covering spaces of surfaces, and graphics processing unit (GPU) was used to accelerate the CVT construction. Inspired by the HEWCVT model for image segmentation, in this thesis we develop the harmonic boundary-enhanced CVT (HBECVT) for surface segmentation, which can be used for polycube construction, all-hex mesh generation and surface parameterization.

2.2 Image Segmentation

Many methods have been proposed for image segmentation in the literature [16, 89]. Thresholding [4, 123] is a very common approach in which an image is represented by groups of pixels created by partitioning the image based on the intensity values and a given threshold. The edge detection method [112, 76] partitions an image by finding the pixels on the region boundary. However, these methods usually do not consider the spatial details and may cause leakage of the boundary for images with low contrast and large variation of intensity. The level set method [128, 62, 93, 18] is typically a partial differential equation based variational method, which represents evolving contours using a signed function. Level-set models are topologically flexible, but quite expensive in both computational time and memory. The graph-based methods [116, 31, 10] are highly efficient and effective approaches for image segmentation in which images are partitioned into a small number of homogeneous regions, but it is still difficult to accurately segment a complex image in some cases. The clustering techniques [40, 63, 12, 85] are also used for region segmentation by partitioning an image into sets or clusters of pixels with similarity in some feature space. K-means [40, 57] and fuzzy c-means (FCM) [17, 2, 19, 95] are the most well-known methods among existing clustering algorithms. The k-means algorithm partitions data points into K clusters, so as to minimize the sum of the squared distances between the data points and cluster centers. The FCM technique introduces fuzzy clustering, in which a pixel can belong to more than one cluster simultaneously. The conventional FCM algorithm does not utilize spatial information, making it very sensitive to noise and other imaging artifacts. CVT based methods have been extensively studied for image segmentation. To improve the segmentation result, a fuzzy EWCVT model [30] was proposed by assigning membership to each pixel corresponding to each cluster. Improving upon the classic CVT and EWCVT methods, our HEWCVT algorithm is less sensitive to the initialization and noise, and generates more accurate segmentation results by combining the image intensity with cluster boundary information.

As an over-segmentation of the image, superpixels [116, 31, 22, 127, 23, 132] have also become popular as they provide an efficient preprocessing tool for various computer vision applications. Ren and Malik [103] defined the perceptually uniform regions as superpixels using the normalized cuts (NCuts) algorithm, which is powerful in obtaining regular superpixels. However, the computational cost is very expensive when the number of superpixels increases greatly. The graph cut based method [83] was developed to construct an optimal solution, which took into account both the edges and the coherence of resulting superpixel lattices. Levinshtein *et al.* [59] presented an efficient TurboPixel superpixel algorithm using the level set based geometric flow evolution from the uniformly placed seeds in the image. This method allows a direct control on the number of superpixels and integrates a compactness constraint. However, it exhibited relatively poor boundary adherence because of its numerical stability issues especially for complicated textures. Achanta *et al.* [1] presented a simple linear iterative clustering (SLIC) superpixel algorithm, and adopted the *k*-means clustering approach to generate the superpixels with relatively lower computational cost. In order to generate superpixels, an EWCVT based algorithm (VCell) [132] was proposed to generate uniform superpixels and preserves image boundaries. In this thesis, we develop an adaptive superpixel generation algorithm by using quadtree decomposition, density function and the local HEWCVT model.

2.3 Surface Segmentation ans Surface Parameterization

Surface segmentation, whose main task is to decompose a surface mesh into meaningful components, is a critical step toward mesh processing and analysis. Traditional surface segmentation algorithms, such as the watershed algorithm [78], randomized cuts [38] and shape diameter function (SDF) [113], decompose an individual 3D surface mesh according to classical geometric features, or using specific data-driven statistical techniques. The SDF expresses a measure of the diameter of volume in the neighborhood of each point on the surface. It first uses a soft clustering of the mesh elements (facets) to *k* clusters based on their SDF values, and then finds the partitioning using *k*-way graph-cut to smooth the boundaries between segments. Some segmentation algorithms make use of the shape skeleton to deduce the different segments. First of all, an approximate skeleton of the input mesh is computed [11], then each critical node of the skeleton will correspond to a segment.

The Laplace-Beltrami operator (LBO) is a geometric operator defined based on the first fundamental form of the surface and its eigenfunctions are capable of capturing structural features of an object [61]. Various discretization schemes of the LBO have been proposed on discretized surface meshes [140, 141, 139], such as the cotangent scheme [82], Fujiwara's discretization [37], and Mayer's discretization [81]. A discretized LBO with convergent property was recently constructed in [64]. The Shape-DNA [105, 104] employs level sets of the LBO eigenfunctions for statistical shape analysis. Point clustering [73] and isocontours [130] were used together with the LBO eigenfunctions to segment surface into several components. Interactive approaches were presented in [147] to choose eigenfunctions and the Mumford-Shah model was then applied to segment the surface into several components. The concavity-aware Laplacian (CL) [130] was developed to detect concavities, and its eigenfunctions can be used to generate a single segmentation field. Improving upon LBO, the secondary Laplace operator (SLO) [67] was developed recently based on the second fundamental form of the surface. Curvature related surface features, such as concave creases and convex ridges, can be captured by its eigenfunctions. The SLO eigenmodes were used for surface segmentation by mapping surface vertices onto a high dimensional space, and then clustering vertices into a series of groups or surface components using the Prediction Analysis for Microarrays (PAM) method [122].

Surface parameterization computes a one-to-one mapping between a 3D surface mesh and an isomorphic planar patch. It has a variety of applications in different fields, including surface quadrangulation [56, 102, 47], texture mapping [133, 60] and surface fitting [55]. For parameterization-based quadrilateral (quad) meshing, vector field guided methods [36, 51, 90, 154, 13] such as mixed-integer quadrangulation [9], periodic global parametrization [102] and QuadCover [56] have been developed to generate quality quad meshes. A typical vector field guided method usually consists of three main steps [8]: a cross field is firstly constructed on the input surface which specifies the orientation and size of quad elements; the surface is then partitioned by a set of curves to a topological disk and parametrized into an integer grid map; and finally, a quad mesh can be extracted by tracing the integer parametric lines on the surface. These methods generate curvature-oriented quad meshes by optimizing the cross field which is derived from the principal curvatures. Eigenfunctions of the LBO were used in [68] as the guidance to capture major structure features during the cross field based global parameterization. Harmonic field based methods [24, 124] construct conformal parameterizations with singularities and offer a degree of control over the size and structure of the domain mesh. In this thesis, we develop a feature-aligned surface parameterization by using the SLO eigenfunctions, where the CVT-based surface segmentation in SLO eigenfunction space is firstly introduced to extract surface features and the cross field based method is then applied for surface parameterization.

2.4 Hexahedral Mesh Generation and Polycube Construction

Several distinct strategies have been proposed in the literature for unstructured all-hex mesh generation. Medial surface methods [98, 97], plastering [7, 119] and whisker weaving [33] have successfully generated all-hex meshes for some geometry, but these methods are not robust and reliable to be applied for general arbitrary geometric domains. CubeCover [84] generates uniform all-hex meshes with the guidance of a valid 3D frame field via a global parametrization. This method was later extended to singularity-restricted field guided volume parametrization [66]. Although these methods can generate all-hex meshes with high quality, they concern about uniform hex meshes only. The grid-based mesh generation method is another promising alternative to create all-hex meshes automatically by generating a fitted 3D grid of hexes in the volume and then adding hex elements at the boundaries to fill gaps. Schneiders *et al.* [110, 108, 109] proposed the first template-based method using four templates for node, edge, face and volume refinement. This method was further developed and improved in recent years [86, 146, 94, 52, 88, 100, 29, 87]. This method is robust and reliable for an arbitrary geometry, but it generates poorly-shaped elements around the boundary and mesh quality needs to be improved. The isosurface extraction methods extract the boundary surface and construct uniform and adaptive hex meshes [149, 148]. Moreover, this method has been extended to meshing domains with multiple materials [152, 101]. In

2009, Maréchal [79] developed a novel octree-based adaptive hex meshing method, which connects hanging nodes using polyhedra, and all-hex meshes can be extracted as the dual of polyhedral cells.

Volumetric polycube [39, 65, 50] is also an appealing approach for all-hex meshing, which converts a volume to an all-hex mesh by first embedding axis-aligned boxes into the 3D volume and then mapping the volume to several jointed boxes. Tarini *et al.* [121] firstly proposed the term polycube and defined a mapping from input objects to the surface of manually constructed polycubes. Several automatic or semi-automatic polycube construction algorithms have been proposed in the literature. In [69], an automated approach for polycube creation was presented using a protrusion based segmentation of the input and fitting of box-primitives, but this method produces very coarse polycubes with significant distortion and is unlikely to work on complex models. A divide-and-conquer strategy was used in [41] to approximate the input for automatic polycube map construction. Although the bijectivity of mapping is guaranteed, this method is sensitive to off-axis features and may generate over-refined polycubes with complex connectivity. Gregson *et al.* [39] presented a volumetric polycube method to convert a volume to a polycube by using Rotation-Driven deformation and Position-Driven deformation. As a follow up, Livesu *et al.* [74] generated polycubes with low distortion based on a valid and all-monotone segmentation using Graph-Cut based labelling and iterative local search via a hill climbing algorithm. An effective volumetric polycube parameterization algorithm [129, 145] was proposed by optimizing the polycube domain based on homotopic morphological operations, which balance the domain simplicity and adequate resemblance to the input model. In [134], a smooth harmonic scalar field with proper boundary conditions was used to generate polycube for arbitrary genus geometry and Boolean operations [70] were further introduced to deal with torus-like objects or holes. The skeleton-based polycube generation method [71] first constructs interior cubes directly from the skeleton branches, and then projects their corners onto the surface to generate new boundary cubes. Interior and boundary cubes are then combined together to split the whole model into different cubic regions. For these methods, hex quality is directly linked to mapping distortion. Computing the polycube structure with a low-distortion mapping still remains a challenging problem for general shapes. In this thesis, we develop two HBECVT based methods for polycube construction. Based on the constructed polycube, we generate uniform and adaptive all-hex meshes as well as volumetric T-meshes via parametric mapping.

2.5 Tetrahedral Mesh Generation and Quality Improvement

Tetrahedral meshing techniques can be classified into three main categories: octree, Delaunay and advancing front. The octree-based methods [114, 115] recursively subdivide an octree containing the geometric model until the desired resolution is reached. Tetrahedra are then constructed from both the irregular cells on the boundary and the internal regular cells. Zhang *et al.* [149] developed an octree-based Dual Contouring method to generate uniform and adaptive tetrahedral meshes. Tetrahedral meshes for complicated domains with topology ambiguity can be generated by splitting the ambiguous leaf cells into tetrahedra and analyzing the edges of these tetrahedra [153]. A parallel Image-to-Mesh conversion algorithm [34] was proposed to generate quality tetrahedral meshes via dynamic point insertions and removals. Delaunay criterion was first utilized for developing algorithms to triangulate a set of vertices via the empty sphere property, which requires that the circumcircle of every triangle is empty or the circumsphere of every tetrahedron is empty. The criterion was later used in developing tetrahedral meshing algorithms [117]. The advancing front methods generate the tetrahedra progressively starting from the boundary, usually layer by layer, until the whole domain is meshed [111]. This method tends to create good quality elements close to the boundary where new vertices can be created at optimal positions. However, the closure algorithms for tetrahedra at the interior are still unstable, especially if the two overlapping elements have a mismatch in element size.

It is crucial to improve the mesh quality in order to avoid the ill-conditioned linear systems during the finite element analysis. Smoothing methods improve mesh quality by relocating vertices without changing the connectivity [35]. However, traditional smoothing techniques are heuristic and sometimes invert or degrade the local elements. To solve this problem, optimization-based smoothing methods were proposed, where each node is relocated to the optimum location based on the local gradient of the surrounding element quality [14]. Surface feature preservation remains another challenging problem. Methods based on local curvature and volume preserving geometric flows were developed to identify and preserve the main surface features [150]. Topological optimization techniques, such as face swapping and edge swapping [58], are utilized to improve the node valence and mesh quality. In this thesis, we extend the HEWCVT algorithm to 3D image segmentation and develop an anisotropic GHO flow method for quality improvement of the tetrahedral/hexahedral meshes.

Chapter 3

Image Segmentation and Adaptive Superpixel Generation Based on Harmonic Edge-Weighted Centroidal Voronoi Tessellation

To extend the EWCVT method for both image segmentation and superpixel generation, in this chapter we first develop an innovative model named harmonic edge-weighted centroidal Voronoi tessellation (HEWCVT) for image segmentation. This method extends and improves upon the EWCVT method by using the harmonic formulation in the clustering energy function. We then present an adaptive superpixel generation algorithm based on the local HEWCVT. Compared to the classic CVT and EWCVT methods, our algorithms have three main advantages:

1. Our HEWCVT-based segmentation scheme is much more stable and less sensitive to the initializations due to an imposition of a soft membership onto the data points;

2. The segmentation accuracy is also improved since the spatial information of local image features is integrated into the harmonic form energy function to compensate for the effect of noise; and

3. We develop an innovative adaptive superpixel generation algorithm using quadtree decomposition and density function, yielding high quality adaptive superpixels with image features preserved efficiently.

3.1 Review of CVT and EWCVT for Segmentation

Since our presented algorithms are based on CVT and EWCVT, let us first review them. Given an image $I(h,w)$ of size $H \times W$, where (h,w) are integer pairs that range over the image domain, $h = 1,\ldots,H$ and $w = 1,\ldots,W$. Let the dataset $X = \left\{x_{P(i)}\right\}_{i=1}^{n}$ denote all the pixel values $x_{P(i)} = I(h,w)$ of the image, where $n = H \times W$ is the total number of pixels and $i = 1,\ldots,n$. $P(i)$ represents the i^{th} pixel in the physical space. Let $C = \{c_l\}_{l=1}^{L}$ denote a set of typical colors for a color image or intensity values for a grayscale image. The Voronoi region V_k in X corresponding to the level c_k $(k = 1,\ldots,L)$ is defined as

$$V_k = \left\{x_{P(i)} \in X : dist\left(x_{P(i)}, c_k\right) \leq dist\left(x_{P(i)}, c_l\right), \qquad for\; l = 1,\ldots,L\right\}, \tag{3.1}$$

where $dist\left(x_{P(i)}, c_k\right)$ is a predefined measure of distance between $x_{P(i)}$ and c_k. Note that here the differences between color values are compared instead of the physical distances between pixels. The set $V = \{V_l\}_{l=1}^{L}$ is called a *Voronoi tessellation* of the data set X. The set of chosen values $C = \{c_l\}_{l=1}^{L}$ are referred to as the *Voronoi generators*.

For the classic CVT [27], $dist\left(x_{P(i)}, c_k\right) = \left|x_{P(i)} - c_k\right|$ is the Euclidean distance in the color space. Given any set of values $C = \{c_l\}_{l=1}^{L}$ and any partition $U = \{U_l\}_{l=1}^{L}$ of X, the classical clustering energy of $(C;U)$ can be defined as follows:

$$E(C;U) = \sum_{l=1}^{L} \sum_{x_{P(i)} \in U_l} \left| x_{P(i)} - c_l \right|^2. \tag{3.2}$$

Note that $U = \{U_l\}_{l=1}^{L}$ are not necessarily the Voronoi regions corresponding to the set of generators $C = \{c_l\}_{l=1}^{L}$ in the definition. The construction of CVTs often can be viewed as an energy minimization process. Given a partition of X, denoted by $U = \{U_l\}_{l=1}^{L}$, the *centroid* of each cluster U_l is defined to be the color $c^* \in U_l$ which minimizes the following objective function:

$$\min_{c \in U_l} \sum_{x_{P(i)} \in U_l} \left| x_{P(i)} - c \right|^2. \tag{3.3}$$

The classical clustering energy $E(C;U)$ is minimized only if $(C;U)$ form a CVT of X, i.e., U are Voronoi regions of X associated with the generators C and simultaneously C are the corresponding centroids of Voronoi regions U.

As a follow-up, the EWCVT model [131] was developed by adding an edge-related energy term to the CVT energy function. The domain of an image $I(h,w)$ is an index set $D = \{P(i) : i = 1,\ldots,n\}$, where $P(i)$ represents the i^{th} pixel in the physical space. Suppose that we have determined the clusters $\{U_l\}_{l=1}^{L}$ for a given image represented in the color space by $x_{P(i)}$ for each pixel $P(i) \in D$, there is a natural segmentation of the image which has L segments $D = \{D_l\}_{l=1}^{L}$ in the physical space defined by

$$D_l = \left\{ P(i) : x_{P(i)} \in U_l \right\}. \tag{3.4}$$

For each pixel $P(i) \in D$, denote a local neighborhood as $N_\omega(P(i))$, which can be a $\omega \times \omega$ square centered at pixel $P(i)$ or a disk centered at pixel $P(i)$ with radius ω. For EWCVT, $dist\left(x_{P(i)}, c_k\right) = \sqrt{\left|x_{P(i)} - c_k\right|^2 + 2\lambda \tilde{n}_k(P(i))}$, where $\tilde{n}_k(P(i)) = |N_\omega(P(i))| - n_k(P(i)) - 1$ is the edge energy term which represents the number of pixels within $N_\omega(P(i)) \setminus (D_k \cup P(i))$. $|N_\omega(P(i))|$ represents the number of pixels within the set $N_\omega(P(i))$. $n_k(P(i))$ denotes the number of pixels within $D_k \cap N_\omega(P(i)) \backslash P(i)$. Note that this distance term combines the color information together with the physical information of the pixel $P(i)$. Here we can view $dist\left(x_{P(i)}, c_k\right)$ as the edge-weighted distance in the color space between $x_{P(i)}$ and c_k.

For any set of values $C = \{c_l\}_{l=1}^L$ and any partition $U = \{U_l\}_{l=1}^L$ (corresponding to partition $D = \{D_l\}_{l=1}^L$ in the physical space) of X, we define the edge-weighted CVT energy as

$$E_{EW}(C;U) = \sum_{l=1}^{L} \sum_{x_{P(i)} \in U_l} \left(\left| x_{P(i)} - c_l \right|^2 + 2\lambda \tilde{n}_l(P(i)) \right), \tag{3.5}$$

where λ is a positive weighting factor to control the balance between the clustering energy and the edge energy. For the EWCVT model, since the edge energy at each pixel is fixed, the *centroid* of each cluster U_l is also defined as the color $c^* \in U_l$ which minimizes the objective function (3.5). The edge-weighted clustering energy $E_{EW}(C;U)$ is minimized only if $(C;U)$ form an edge-weighted CVT of X, i.e., U are edge-weighted Voronoi regions of X associated with the generators C and simultaneously C are the corresponding centroids of the region U. In its simplest form, CVT and EWCVT algorithms reduce to the k-means clustering technique. It is also clear that the EWCVT algorithm will reduce to the classic CVT-based algorithm when the weight $\lambda = 0$.

Both classic CVT and EWCVT have a drawback: since they are sensitive to initializations and noise, the segmentation results are inaccurate and unstable in certain cases. In the next section, we will present an extended model by combining the harmonic algorithm with the EWCVT that overcomes these limitations.

3.2 Harmonic Edge-Weighted CVT for Image Segmentation

Improving upon EWCVT, HEWCVT uses the harmonic average form of the energy function to calculate the centroids. For HEWCVT, the definitions of Voronoi region and edge-weighted distance $dist\left(x_{P(i)}, c_k\right)$ are the same as EWCVT. The EWCVT energy function defined in Section 3.1 is slightly different from the algorithm proposed in [131]. Here, we define the edge-weighted distance first and then use the edge-weighted distance to define the EWCVT energy function. For any set of color values $C = \{c_l\}_{l=1}^{L}$ and any partition $U = \{U_l\}_{l=1}^{L}$ of X, we define the HEWCVT energy by introducing the harmonic idea into the EWCVT method:

$$E_H(C;U) = \sum_{i=1}^{n} \left(L \middle/ \sum_{l=1}^{L} dist^{-2}\left(x_{P(i)}, c_l\right) \right), \tag{3.6}$$

where $dist\left(x_{P(i)}, c_k\right) = \sqrt{\left|x_{P(i)} - c_k\right|^2 + 2\lambda \tilde{n}_k(P(i))}$. The HEWCVT energy uses edge-weighted distance from all the centroids to calculate the harmonic average for each point. To calculate the updated centroids $\left\{c_k^*\right\}_{k=1}^{L}$, we minimize the HEWCVT energy function with respect to the centroid c_k $(k = 1, \ldots, L)$. Since the local edge energy term $\tilde{n}_k(P(i))$ at each pixel $P(i)$ is fixed, we have

$$\frac{\partial E_H(C;U)}{\partial c_k} = -L \sum_{i=1}^{n} \frac{2\left(x_{P(i)} - c_k\right)}{dist^4\left(x_{P(i)}, c_k\right) \left(\sum_{l=1}^{L} dist^{-2}\left(x_{P(i)}, c_l\right) \right)^2} = 0, \tag{3.7}$$

then we can obtain an iterative formula as follows:

$$c_k^* = \frac{\sum_{i=1}^{n} u_{ik}^H x_{P(i)}}{\sum_{i=1}^{n} u_{ik}^H}, \tag{3.8}$$

where $u_{ik}^H = \left(\sum_{l=1}^{L} \frac{dist^2\left(x_{P(i)}, c_k\right)}{dist^2\left(x_{P(i)}, c_l\right)} \right)^{-2}$. $u_{ik}^H = \mu_{ik} w_{P(i)}$, where $\mu_{ik} = \frac{dist^{-4}\left(x_{P(i)}, c_k\right)}{\sum_{l=1}^{L} dist^{-4}\left(x_{P(i)}, c_l\right)}$ and $w_{P(i)} = \frac{\sum_{l=1}^{L} dist^{-4}\left(x_{P(i)}, c_l\right)}{\left(\sum_{l=1}^{L} dist^{-2}\left(x_{P(i)}, c_l\right) \right)^2}$. μ_{ik} is the membership function which represents the degree of possibility of $P(i)$ that is associated with the k^{th} cluster, and $w_{P(i)}$ is the dynamic weighting function which defines how much influence $P(i)$ has in the next iteration of computing new centroids. Hence, HEWCVT is less affected by the presence of uncertainty or noise since it has a soft membership function and a varying weight function. For an arbitrary Voronoi tessellation $\left(\{c_l\}_{l=1}^{L}; \{V_l\}_{l=1}^{L}\right)$ of X where $V = \{V_l\}_{l=1}^{L}$ are the corresponding edge-weighted Voronoi regions associated with $C = \{c_l\}_{l=1}^{L}$, we often have $c_l \neq c_l^*$, for $l = 1, \dots, L$, where $\left\{c_l^*\right\}_{l=1}^{L}$ are the corresponding centroids of $\{V_l\}_{l=1}^{L}$. The harmonic edge-weighted clustering energy $E_H(C; V)$ is minimized only if $(C; V)$ form a HEWCVT of X, i.e., V are Voronoi regions of X associated with the generators C and simultaneously C are the corresponding centroids of the region V.

The construction of HEWCVT is an energy minimization process, where the generators are updated by minimizing the corresponding objective functions. The detailed implementation is explained as follows.

Algorithm of HEWCVT (Implementation)

Given a digital image $X = \left\{x_{P(i)}\right\}_{i=1}^{n}$, positive integer L and error tolerance ε ($\varepsilon = 10^{-4}$ in this thesis), choose L random pixels in the image and take their color values $\{c_l\}_{l=1}^{L}$ as the initialization. E_i denotes the HEWCVT energy in the i^{th} iteration. Then perform the following:

1. Determine the edge-weighted Voronoi clusters $\{V_l\}_{l=1}^{L}$ of X associated with $\{c_l\}_{l=1}^{L}$ by (3.1);

2. For each cluster V_l ($l = 1, \dots, L$), determine the cluster means c_l^* by (3.8); and

3. If a termination criterion such as $\frac{E_{i+1} - E_i}{E_i} < \varepsilon$ is reached, return $(\{c_l\}_{l=1}^{L}; \{V_l\}_{l=1}^{L})$ and exit; otherwise, set $c_l = c_l^*$ for $l = 1, \dots, L$ and return to Step 1.

Remark. Due to the property of the harmonic mean, HEWCVT usually yields more accurate results than EWCVT. EWCVT imposes hard membership on the data points: each data point is assigned to exactly one center. This means that each data point only has an influence over the center to which it is assigned. In contrast, the objective function in HEWCVT uses the distances to all centroids for each data point. This means that all centroids partially influence the harmonic average for each point. It is less affected by the presence of uncertainty or noise in the data since it has a soft membership function. Thus HEWCVT is less sensitive to the initialization than EWCVT, especially when the number of clusters is large.

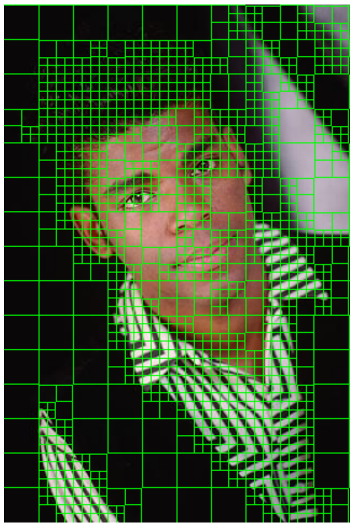
3.3 Adaptive Superpixel Generation based on HEWCVT

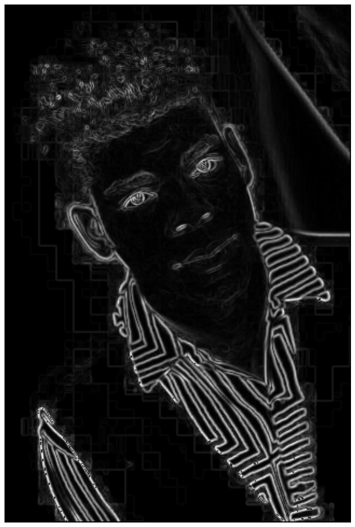
In this section, we aim to generate adaptive superpixels using HEWCVT. Given an input image, quadtree-based decomposition is used for the initial seed sampling and the density function is calculated to measure the regularity degree of the variation in the image. The classic CVT algorithm is then applied in the physical space to generate adaptive segments based on the initial seed sampling and density function. Finally, local HEWCVT is applied to these segments to generate adaptive superpixels. We describe in detail how to initialize seed points and generate adaptive superpixels below.

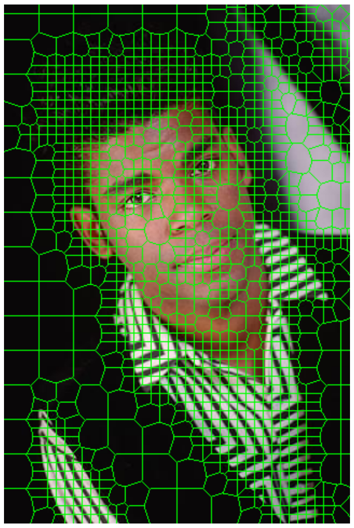
3.3.1 Initializing Seed Points Via Quadtree Decomposition

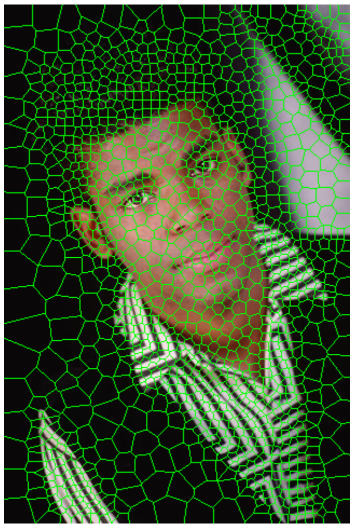
To initialize a set of seeds (or generators) for our HEWCVT based algorithm, we sample the image adaptively using quadtree decomposition. The seeds are the centers of the leaf cells in the adaptive quadtree. To detect features in a quadtree cell, we adopt an error function, $ERROR = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_{p(i)} - \bar{x}\right)^2}$, where N is the number of pixels and $\bar{x}$ is the average color/intensity in the cell. This error function estimates the standard deviation of color values in the *CIELAB* color space for all pixels in one cell. Given an error tolerance ε (*e.g.*, $\varepsilon = 5$ for the portrait model in Figure 3.1), we refine cells with a larger error ($> \varepsilon$). We can also measure the average Euclidean distance in the color space between all pixels and the average color in the cell. If the average distance is larger than ε, then the cell is subdivided into four children cells. Figure 3.1(b) shows the quadtree decomposition result for the portrait

(a) Original image


(b) Quadtree decomposition


(c) Sobel operator result


(d) Initial Voronoi diagram


(e) CVT

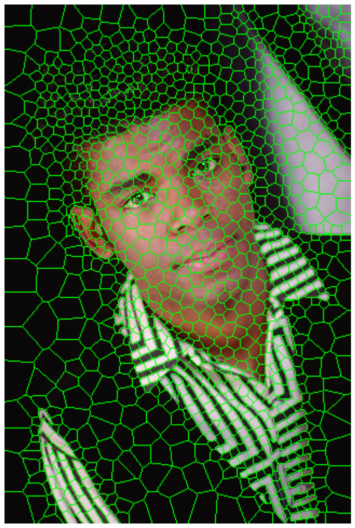

(f) Adaptive superpixels

Figure 3.1: Adaptive superpixel generation for the portrait image. (a) Original image; (b) quadtree decomposition; (c) Sobel operator result; (d) initial Voronoi diagram; (e) CVT in physical space; and (f) adaptive superpixels.

image. The seeds are simply set as the centers of the leaf cells in the adaptive quadtree. We can observe that the quadtree based sampling method provides a nice initialization of seed points for computing adaptive CVTs in the physical space.

The density function ρ is constructed to enable adaptive Voronoi regions in the image. In this study, we use the Sobel operator [77] to calculate the density function. The Sobel operator performs a $2D$ spatial gradient measurement on an image and thus emphasizes regions of high spatial frequency that correspond to edges. Typically it is used to approximate the absolute gradient magnitude at each point in an input image. Figure 3.1(c) shows the Sobel operator result of the portrait image. Brighter pixels represent higher density. The density function helps produce adaptive CVTs.

3.3.2 Adaptive Superpixel Generation

Based on the initial seed sampling and density function, we compute adaptive CVTs in the physical space by using the classic CVT algorithm [25, 53], which is capable of generating large number of Voronoi cells (or segments) very efficiently. Figure 3.1(d) shows the obtained initial Voronoi diagram based on the initial seeds, and Figure 3.1(e) shows the adaptive CVTs of the portrait image. The generation of CVTs in the physical space is independent of the image intensities and colors and the adaptation is controlled by the initial seed sampling and density function. Since the classic CVT algorithm in the physical space generates small segments with regular hexagonal shape, it provides a good initial partition for the following adaptive superpixel generation.

The resulting adaptive CVT segments are then used as the initial partition for the adaptive superpixel generation via a local HEWCVT. Here, the harmonic edge-weighted CVT energy is defined locally among the segments of the superpixels, thus the centroid of each superpixel can be updated locally. The pixel value $x_{P(i)}$ is measured in a 5D *Labxy* space with both color and spatial information. A pixel color is represented in the CIELAB color space $[L, a, b]^T$. A pixel position is represented by the position vector $[x, y]^T$. Thus, $x_{P(i)} = [L, a, b, x, y]^T$. The edge-weighted 5D distance from a pixel $x_{P(i)}$ to a centroid c_k is define as:

$$dist\left(x_{P(i)}, c_k\right) = \sqrt{\left(\frac{d_c}{N_c}\right)^2 + \rho_i\left(\frac{d_s}{N_s}\right)^2 + 2\lambda\tilde{n}_k(P(i))}, \tag{3.9}$$

where

$$d_c = \sqrt{(L_k - L_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2},$$

and

$$d_s = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}.$$

N_c and N_s are used to normalize the color and spatial proximities respectively. $N_s = \sqrt{H \times W/L}$, and ρ_i is the density function of the i^{th} pixel.

For any set of values $C = \{c_l\}_{l=1}^{L}$, which are also in the 5D *Labxy* space, and any partition $U = \{U_l\}_{l=1}^{L}$ of X, we define the local HEWCVT energy as follows:

$$E_H(C;U) = \sum_{i=1}^{n} \left(L \Big/ \sum_{l \in NC(P(i))} dist^{-2}\left(x_{P(i)}, c_l\right) \right), \tag{3.10}$$

where $NC(P(i))$ represents the neighboring clusters of the cluster containing $P(i)$. To calculate the centroids $\left\{c_k^*\right\}_{k=1}^{L}$, we also minimize the local HEWCVT energy function with respect to the centroid c_k ($k = 1, \ldots, L$). We have

$$c_k^* = \begin{cases} \dfrac{\sum\limits_{i \in NPC(U_k)} u_{ik}^H x_{P(i)}}{\sum\limits_{i \in NPC(U_k)} u_{ik}^H}, & for\ [L,\ a,\ b], \\ \dfrac{\sum\limits_{i \in NPC(U_k)} u_{ik}^H \rho_i x_{P(i)}}{\sum\limits_{i \in NPC(U_k)} u_{ik}^H \rho_i}, & for\ [x,\ y], \end{cases} \tag{3.11}$$

where $u_{ik}^H = \left(\sum\limits_{l \in NC(P(i))} \frac{dist^2\left(x_{P(i)}, c_k\right)}{dist^2\left(x_{P(i)}, c_l\right)} \right)^{-2}$ and $NPC(U_k)$ represents the neighboring clusters of the k^{th} cluster. When we update the centroid for one cluster, we only need to consider all pixels belonging to itself and its neighboring clusters. For each pixel $P(i)$ in one cluster, it can only be transferred to its neighboring clusters. The detailed implementation steps are listed as follows.

Algorithm of local HEWCVT (Implementation)

Given the adaptive CVT segments in the physical space and an error tolerance ε ($\varepsilon = 10^{-4}$ in this thesis), we initialize $\{c_l\}_{l=1}^{L}$ by using the corresponding average color values and position coordinates in each segment. Let E_i denote the local HEWCVT energy in the i^{th} iteration. Then perform the following:

1. For each boundary pixel $P(i)$, evaluate the 5D harmonic edge-weighted distance between the pixel and all centroids of $NC(P(i))$ and assign the pixel to the segment whose color centroid is nearest to it;

2. For each cluster V_l ($l = 1, \ldots, L$), determine the cluster means c_l^* by (3.11); and

3. If a termination criterion such as $\frac{E_{i+1}-E_i}{E_i} < \varepsilon$ is reached, return $(\{c_l\}_{l=1}^L; \{V_l\}_{l=1}^L)$ and exit; otherwise, set $c_l = c_l^*$ for $l = 1, \ldots, L$ and return to Step 1.

In our algorithm, we only need to evaluate the distance in which $P(i)$ is a boundary pixel. Therefore, a pixel $P(i)$ can only be assigned to a segment which is physically connected to it and thus unconnected segments can be mostly avoided. While our adaptive superpixel generation algorithm leads to very accurate clusters, it does not guarantee that the clusters always remain connected. We apply the effective strategy introduced by SLIC [1] as a post-processing step to reconnect ripped-apart or isolated superpixels.

Remark. Our adaptive superpixel generation algorithm can efficiently generate inhomogeneous superpixels that are adaptive in size and compact. The purpose is to use less number of superpixels to preserve all important features in the image. It generates smaller superpixels to achieve lower under-segmentation in structure-dense regions with high intensity or color variation, and produces larger segments to increase computational efficiency in structure-sparse regions with homogeneous appearance. Initial seed sampling using quadtree decomposition provides the high-quality adaptation. The modified local HEWCVT algorithm combines both color and spatial space information with an edge weighted penalty term and increased computational efficiency.

3.4 Results and Discussion

In this section, we apply our presented algorithms to various synthetic and real images that are either noise free or corrupted by different types of noises. All the results are computed on a PC equipped with a 2.93 GHz Intel X3470 CPU and 8GB of Memory.

3.4.1 Image Segmentation Results

Table 3.1 shows the statistics information of all the five testing examples. For image segmentation, there are a total of three parameters that can be tuned in our HEWCVT-based algorithm: L, the number of clusters; λ, the weighting parameter that balances the clustering energy and the edge energy; ω, which defines the size of the local neighborhood. [131] made a detailed discussion on the effect and setting of λ. The effect of ω is in fact very similar to that of λ. The computational cost, however, increases in a quadratic manner w.r.t. ω. Therefore, we choose a relatively small value of ω ($\omega = 4$) for all the examples.

In order to evaluate the segmentation results quantitatively, we apply two different error metrics: segmentation accuracy (SA) and boundary recall (BR). The SA is defined as [2]:

$$SA = \frac{N_{Correct}}{N_{Total}} \times 100\%, \tag{3.12}$$

where $N_{Correct}$ represents the number of correctly classified pixels and N_{Total} is the total number of pixels in the image. Boundary recall is used to evaluate the accuracy of feature preservation. It calculates the fraction of ground truth edges that fall within a small distance (two pixels in this thesis) from at least one boundary pixel. Given a ground truth image G and a resulting image R, the BR can be defined as [103]:

$$BR = \frac{TP}{TP+FN} \times 100\%, \tag{3.13}$$

Table 3.1: Image segmentation statistics of five testing models.

Image (Size)	Number of Clusters	ω	λ	Average Time (Seconds)	
Two-class (256×256)	2	4	0.50	Classic CVT	0.5
				EWCVT	2.0
				HEWCVT	3.4
Brain (181×217)	4	4	0.10	EWCVT	5.4
				HEWCVT	11.0
Head (511×484)	4	4	0.02	EWCVT	10.2
				HEWCVT	17.8
Chapel (481×321)	4	4	0.01	EWCVT	10.3
				HEWCVT	19.7
Starfish (481×321)	5	4	0.01	EWCVT	11.4
				HEWCVT	20.3

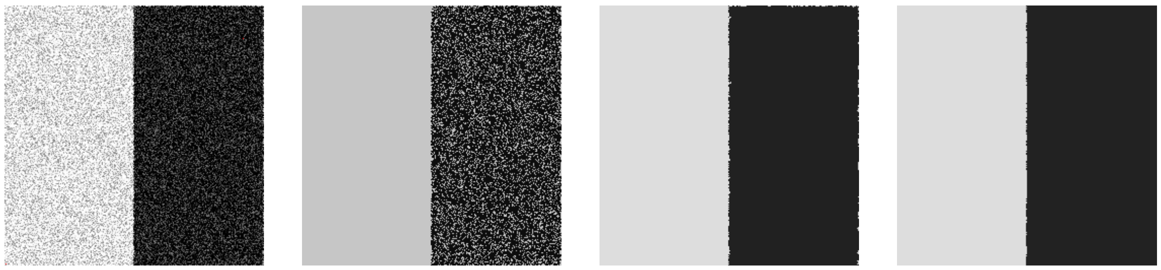
where *true positive* (TP) is the number of boundary pixels in G with at least one boundary pixel in R in range of two pixels, *false negative* (FN) is the number of boundary pixels in G with no boundary pixel in R in range of two pixels. For both SA and BR, 0 (0%) represents the worst case and 1 (100%) represents the perfect case. We also introduce a new metric named the segmentation coefficient of variation (SCV) to evaluate the stability of different methods. For each image, we test N (N = 100 in this thesis) times with different seed point initializations. We can get one minimized energy value for each test. The SCV is defined as:

$$SCV = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(MinE_i - \overline{MinE}\right)^2}}{\overline{MinE}} \times 100\%, \tag{3.14}$$

where $MinE_i$ represents the minimized energy for the i^{th} test, and $\overline{MinE}$ represents the mean of the minimized energy. Small SCV values are usually considered low-variance, otherwise high-variance.

Figure 3.2 shows the segmentation results of a noisy two-class synthetic image with two clusters. Here we choose two different intensity values (120 and 20) for initialization and apply three methods: the classic CVT, EWCVT and HEWCVT. It is obvious that

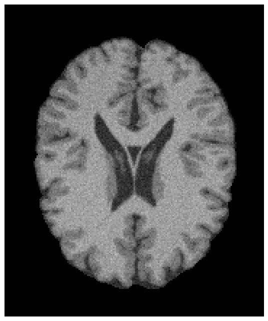
HEWCVT yields the most accurate result. We then test 100 times with different seed point initializations. The average SAs for CVT, EWCVT and HEWCVT are 84.39%, 95.48% and 99.29%, respectively. Since there are no complex boundary features in this image, we do not compare the average BRs here. The SCV results for CVT, EWCVT and HEWCVT are 7.12%, 6.27% and 0.67%, respectively. Compared to the classic CVT and EWCVT, it is evident that HEWCVT is more stable and accurate. From Table 3.1, we can also observe that HEWCVT takes reasonably longer time than EWCVT because it needs to compute distances between one pixel to all clusters when updating cluster means.

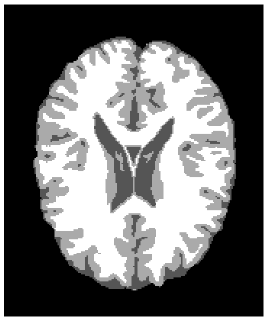


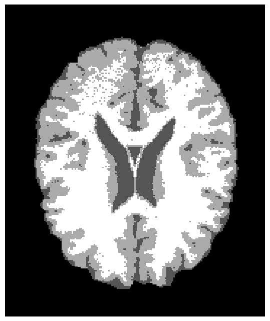
(a) Original image (b) Classic CVT (c) EWCVT (d) HEWCVT

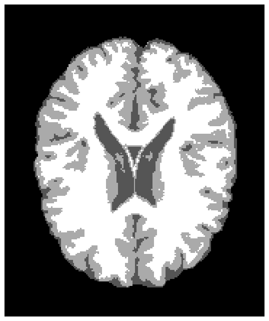
Figure 3.2: Segmentation results of a two-class synthetic image. (a) Original image; (b) classic CVT; (c) EWCVT; and (d) HEWCVT.

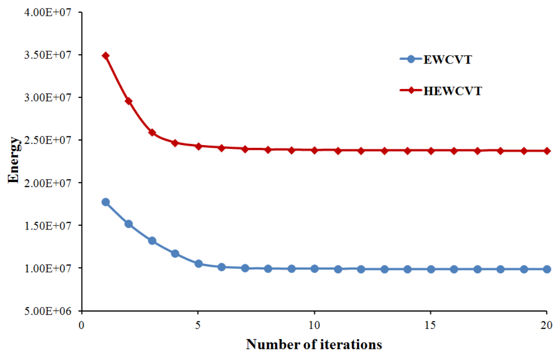
We also test our algorithm on one slice (slice 90) of a 3D MRI-T1 brain image from BrainWeb [20], with four levels of noise $(3\%, 5\%, 7\%, 9\%)$ and three levels of intensity non-uniformity (INU) $(0\%, 20\%, 40\%)$. We segment each of the twelve images 100 times with different initializations using both EWCVT and HEWCVT, and each time it is segmented into four clusters in order to distinguish the gray matter, the white matter, cerebrospinal fluid and background. The ground truth segmentation results (annotations) can be obtained from BrainWeb as well for comparison. Figure 3.3(a)-(d) shows the segmentation results of one testing image, which has 7% noise and 20% INU. Since it is difficult to plot all the convergence curves of all 100 tests, we only plot the average energy convergence curves in Figure 3.3(e). From the convergence curves, we can observe that HEWCVT usually needs fewer iterations to converge than EWCVT. Since EWCVT and HEWCVT are essentially an


(a) Original image

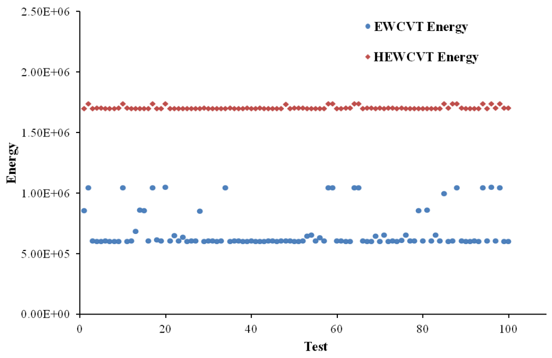

(b) Ground truth


(c) EWCVT


(d) HEWCVT



(e) Convergence curves



(f) Energy outputs

Figure 3.3: Segmentation results of a brain MRI-T1 image with 7% noise and 20% INU. (a) Original image; (b) ground truth result; (c) EWCVT; (d) HEWCVT; (e) average energy convergence curves; and (f) energy outputs.

energy minimization process, we can calculate a minimized energy for each test. Figure 3.3(f) shows the minimized energy outputs for all the 100 tests. With different initializations, the energy function converges to different values for EWCVT, while HEWCVT is much more stable and less sensitive to initializations with all $SCVs < 1\%$ in Table 3.2. Table 3.2 also shows the average SA and the average BR of the 100 tests, we can observe that HEWCVT improves the segmentation accuracy compared to EWCVT. In addition, our HEWCVT-based method is also more robust to noise since the SA, BR and SCV results do not change much for different levels of noise and INU.

We then apply the HEWCVT segmentation to a head CT image from BrainWeb [20] and two color images from the Berkeley dataset [80], see the segmentation results in Figures. 3.4-3.6. Again, we segment them with 100 different random initializations using both EWCVT and HEWCVT. Since there are no ground truth results for these real images, Table 3.3 shows

Table 3.2: SA, BR and SCV results of different methods on one brain image with different noise level and intensity non-uniformity.

Error Metric	Method	3%noise 0%INU	3%noise 20%INU	3%noise 40%INU	5%noise 0%INU	5%noise 20%INU	5%noise 40%INU	7%noise 0%INU	7%noise 20%INU	7%noise 40%INU	9%noise 0%INU	9%noise 20%INU	9%noise 40%INU
SA	EWCVT	88.94%	88.71%	88.70%	88.74%	88.11%	87.89%	87.46%	87.01%	86.67%	86.13%	85.69%	85.14%
	HEWCVT	**96.48%**	**96.23%**	**96.11%**	**96.12%**	**95.68%**	**95.13%**	**94.77%**	**94.49%**	**94.06%**	**93.46%**	**93.28%**	**92.76%**
BR	EWCVT	90.81%	90.54%	90.34%	90.26%	89.92%	89.79%	89.23%	88.78%	88.14%	87.56%	87.23%	86.78%
	HEWCVT	**97.89%**	**97.66%**	**97.49%**	**97.61%**	**97.09%**	**96.88%**	**96.34%**	**96.01%**	**95.67%**	**95.11%**	**94.73%**	**93.23%**
SCV	EWCVT	12.2%	14.1%	14.6%	15.1%	15.4%	15.9%	15.8%	16.1%	16.2%	16.0%	16.4%	16.8%
	HEWCVT	**0.74%**	**0.77%**	**0.78%**	**0.79%**	**0.79%**	**0.81%**	**0.78%**	**0.84%**	**0.85%**	**0.83%**	**0.85%**	**0.87%**

only SCVs of the testing models. From the results, we can see that EWCVT generates unstable results that sometimes mistakenly segment the structural details, shown in Figure 3.4(b), Figure 3.5(b) and Figure 3.6(b). In contrast, the HEWCVT method generates much more stable results with all $SCVs < 1\%$ (see Table 3.3). In addition, HEWCVT improves the segmentation accuracy by capturing more features, and it requires less iterations than EWCVT to converge to the optimal results.

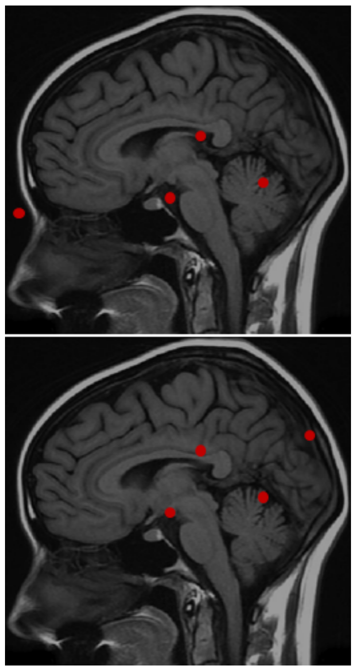
Table 3.3: Segmentation coefficient of variation (SCV) of the testing models.

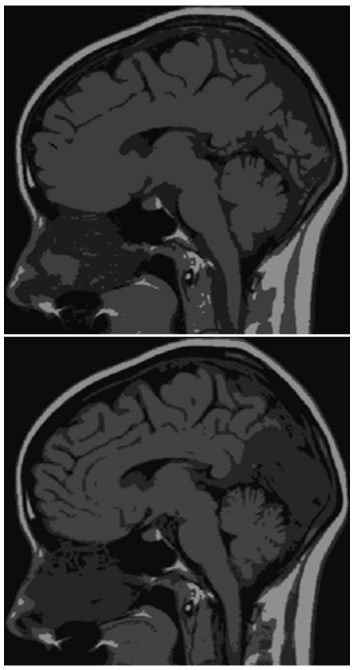
Method	Head	Chapel	Starfish
EWCVT	14.08%	47.55%	13.36%
HEWCVT	0.32%	0.73%	0.83%

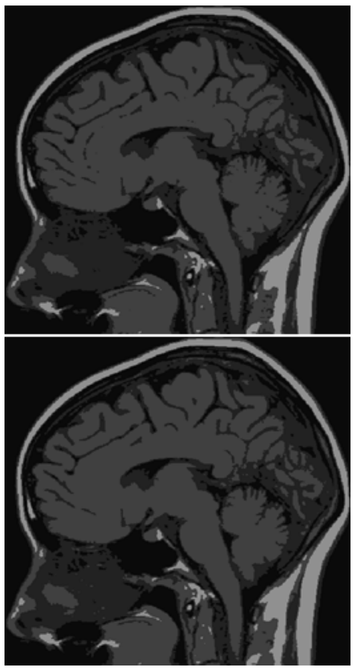
3.4.2 Adaptive Superpixel Generation Results

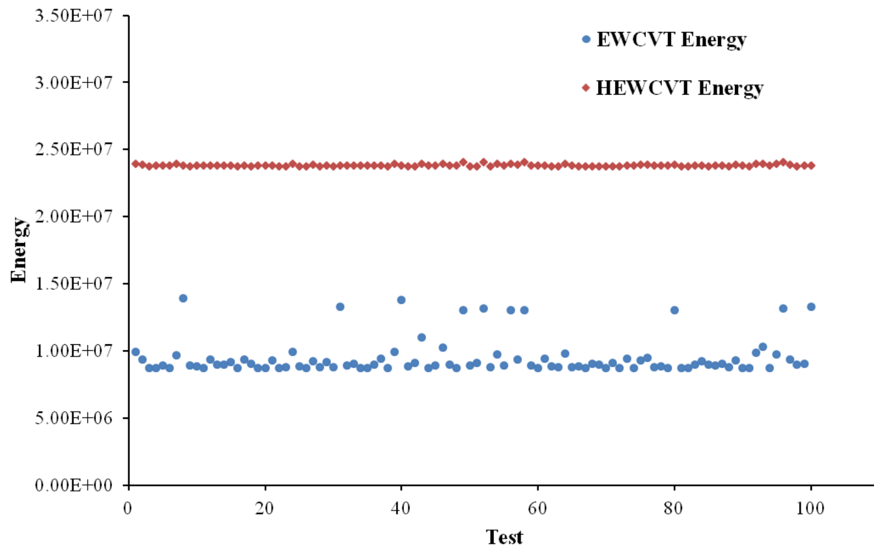
We apply our adaptive superpixel generation method to various types of color and grayscale images, see Figure 3.7. Table 3.4 shows the statistics information of all the testing examples. For our adaptive superpixel generation algorithm, in addition to λ and ω, there are a total of four more parameters that need to be defined: $MinL$ and $MaxL$, the minimal and maximal quadtree level; ε, the error tolerance for quadtree decomposition; and N_c, the color distance regularization term.

To evaluate the efficiency of our adaptive superpixel generation method, we first compare our adaptive superpixels with uniform results. Our method can generate uniform result easily


(a) Original image


(b) EWCVT results


(c) HEWCVT results

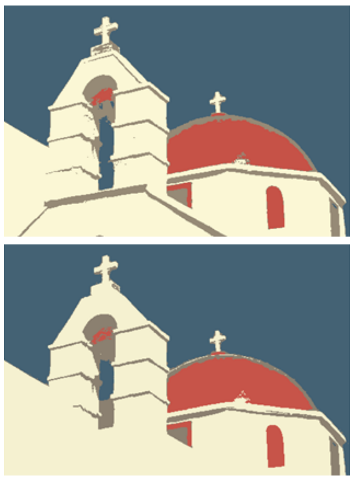


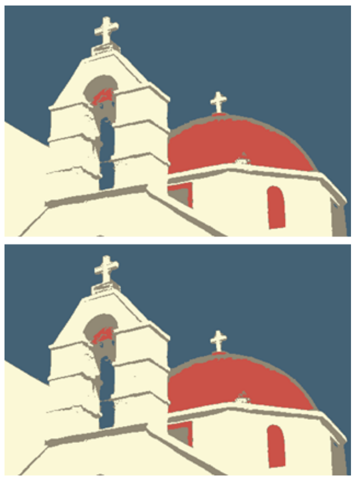
(d) Energy outputs

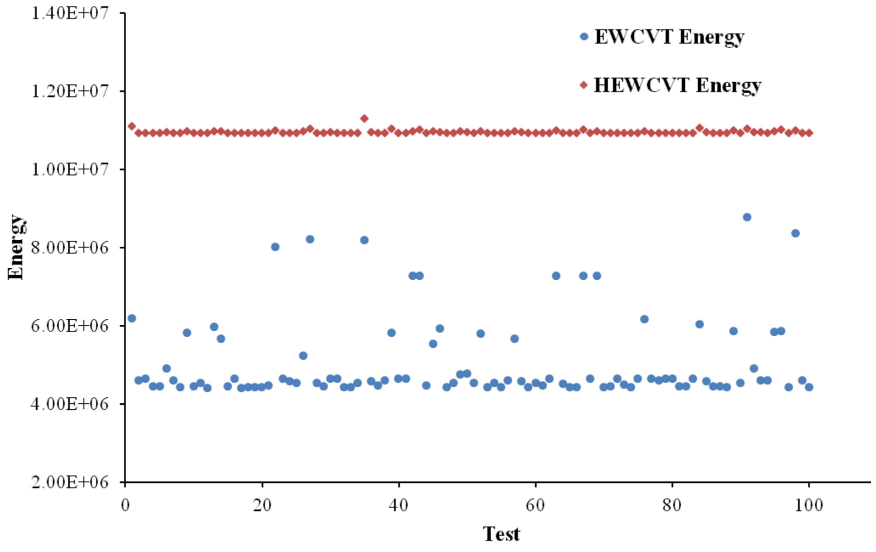
Figure 3.4: Segmentation results of a head image. (a) Original image with two different initializations (each initialization has 4 input seed points marked as red dots); (b) EWCVT; (c) HEWCVT; and (d) energy outputs.

by subdividing the quadtree uniformly. Figure 3.8 shows comparisons between uniform and adaptive superpixels for the chapel image with 800 clusters and the portrait image with 1000

(a) Original image


(b) EWCVT results


(c) HEWCVT results

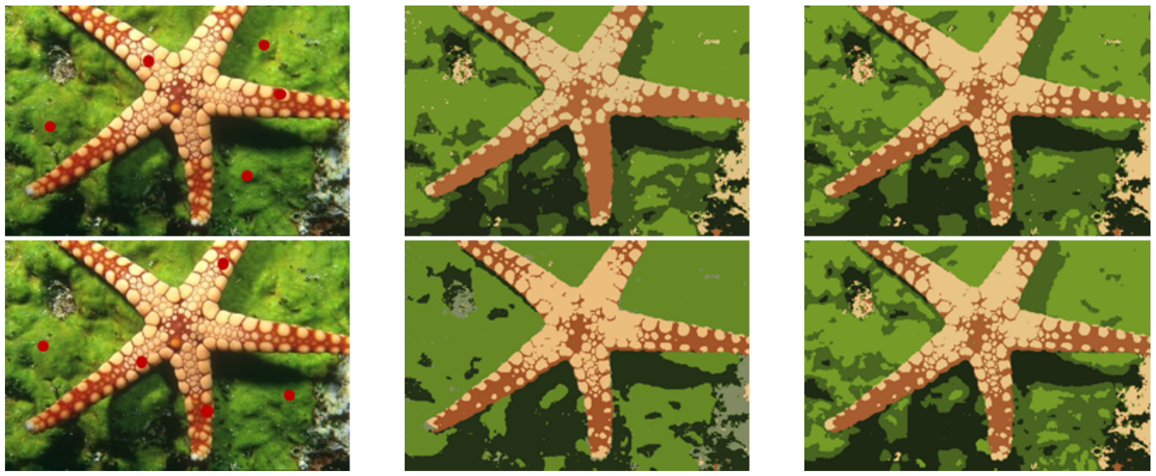


(d) Energy outputs

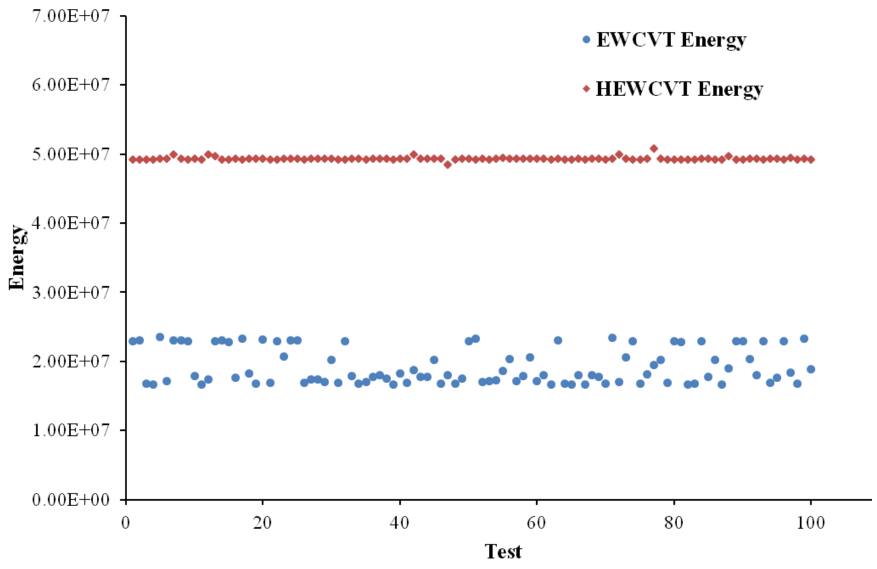
Figure 3.5: Segmentation results of a chapel image. (a) Original image with two different initializations (each initialization has 4 input seed points marked as red dots); (b) EWCVT; (c) HEWCVT; and (d) energy outputs.

clusters. It is obvious that with the same number of clusters, adaptive superpixels capture more detailed features with much fewer superpixels.

To quantitatively evaluate our method, we use the undersegmentation error (UE) and BR metrics. Given a ground truth segmentation with segments $g_1, \ldots, g_M$ and a superpixel



(a) Original image (b) EWCVT results (c) HEWCVT results



(d) Energy outputs

Figure 3.6: Segmentation results of a starfish image. (a) Original image with two different initializations (each initialization has 5 input seed points marked as red dots); (b) EWCVT; (c) HEWCVT; and (d) energy outputs.

segmentation with superpixels $s_1, \dots, s_L$, the UE is defined as [1]

$$UE = \frac{1}{n}\left[\sum_{i=1}^{M}\left(\sum_{s_j|s_j\cap g_i\neq\emptyset} Area(s_j)\right) - n\right], \tag{3.15}$$

where M is the number of ground truth segments, n is the total number of pixels and $Area(s_j)$ is the size of a superpixel segment (or the number of pixels in a superpixel segment).

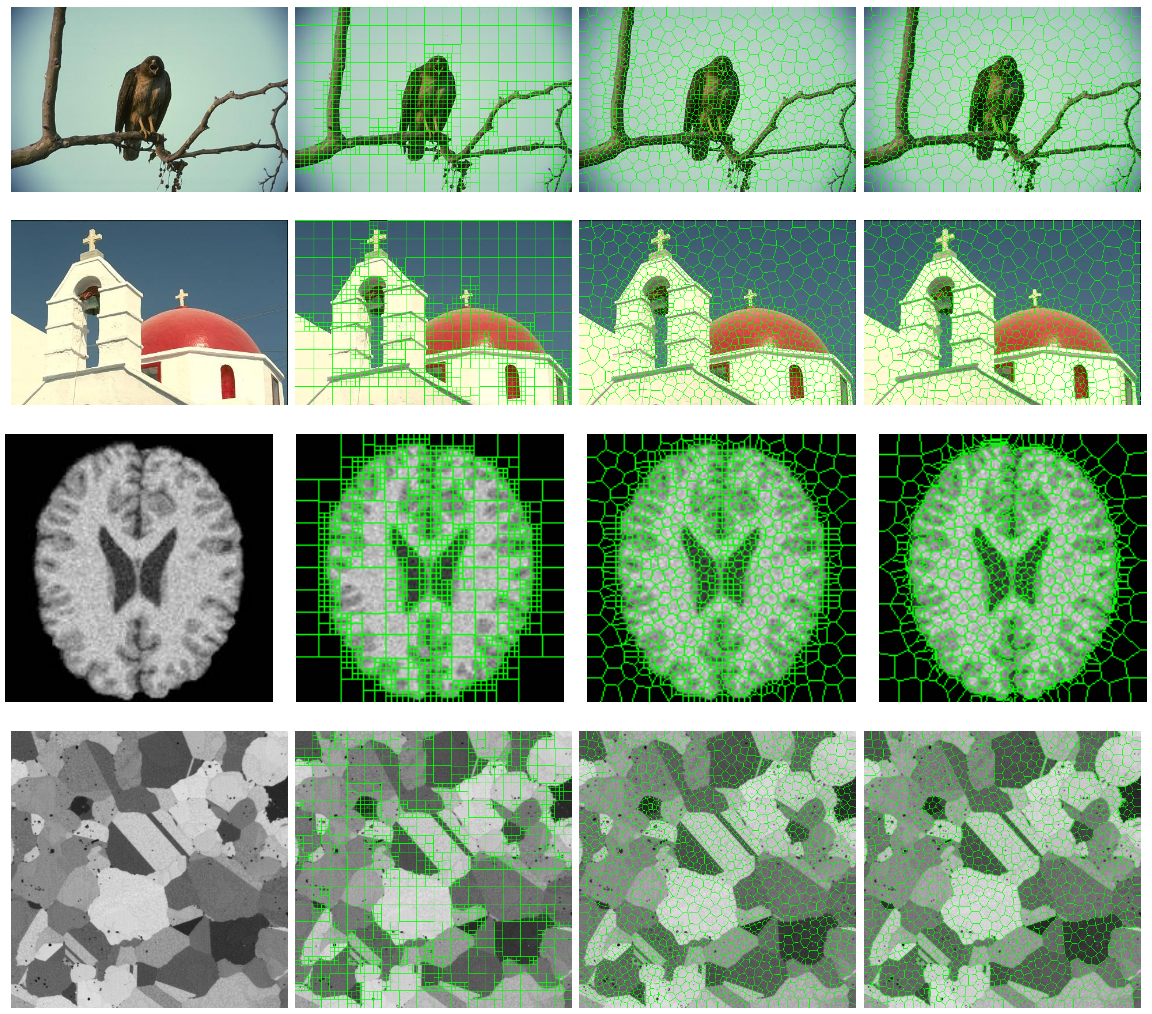


Figure 3.7: Four examples of adaptive superpixel generation. From left to right: original image, quadtree decomposition, CVT in physical space, and adaptive superpixels. From top to bottom: Bird, Chapel, Brain and Alloy.

Superpixels that tightly fit the ground truth result in a low value of UE. Figure 3.9 shows the UE and BR results of SLIC [1], VCell [132], and our proposed method, all of which are obtained by averaging 100 test images from the Berkeley dataset. The results show that the overall performance of our method is better than SLIC and VCell. The reason is that our method can generate smaller superpixels to achieve lower under-segmentation in structure-dense regions with high intensity or color variation, and produce larger segments to increase computational efficiency in structure-sparse regions with homogeneous appearance, thus it can capture more detailed features with less number of superpixels. Since our superpixel generation algorithm is based on HEWCVT, it is also less sensitive to the initialization and noise.

Table 3.4: Segmentation accuracy and boundary recall of different methods on one brain image with different noise level and intensity non-uniformity.

Image (Size)	λ	ω	$MinL$	$MaxL$	ε	N_c	Time (s)
Portrait (321×481)	0.01	4	3	6	5	10	14.8
Bird (481×321)	0.01	4	3	6	10	15	15.2
Chapel (481×321)	0.01	4	3	6	10	15	14.6
Brain (181×217)	0.1	4	2	5	5	10	5.6
Alloy (512×512)	0.02	4	3	6	10	15	22.7

After the adaptive superpixel generation, we can compute the mean color for all superpixels, which can be considered as a compact representation for the original image. This allows us to represent an image with only a couple of hundred segments instead of tens of thousands of pixels. The resulting superpixels can be used for multiple image processing applications, such as image segmentation, registration and object tracking. We can also apply image segmentation algorithms on superpixels instead of pixels, which can largely reduce the computational cost while maintaining high accuracy.

3.5 Conclusions and Future Work

In this thesis, we extend the basic EWCVT model for image segmentation to a new advanced model, namely HEWCVT. This extended model introduces a harmonic form of clustering energy by combining the image intensity with cluster boundary information, and then updates generators of the corresponding Voronoi tessellations by minimizing the objective function. Experiments indicate that the HEWCVT algorithm is more robust to the initialization and

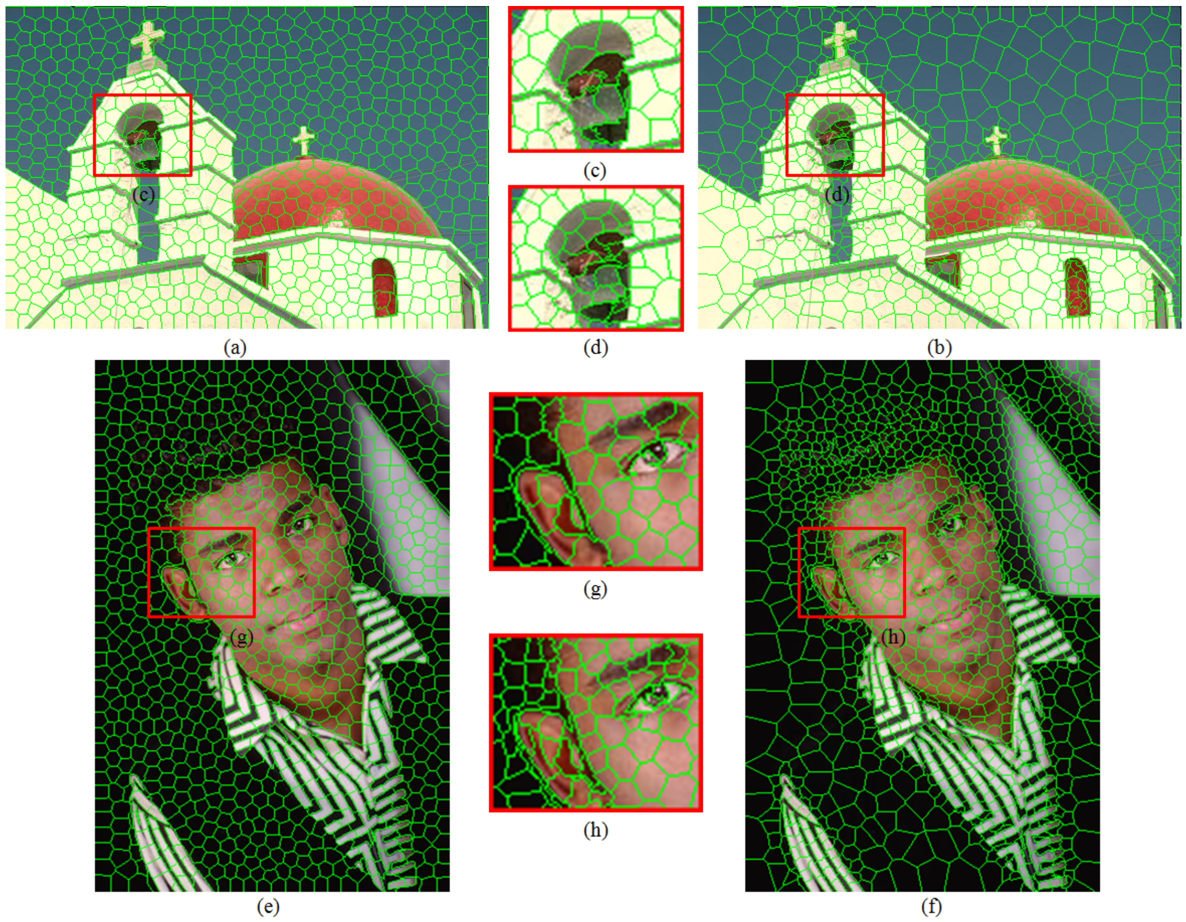
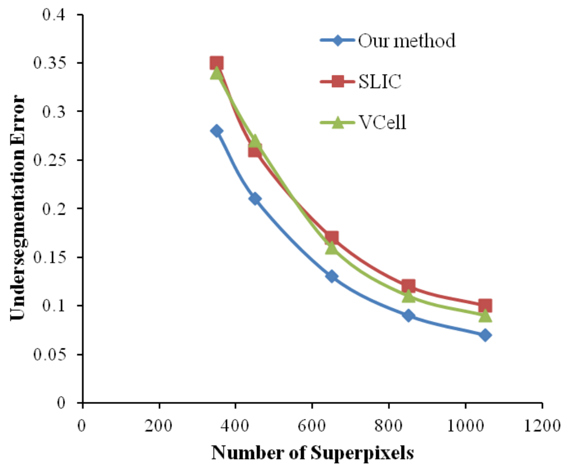
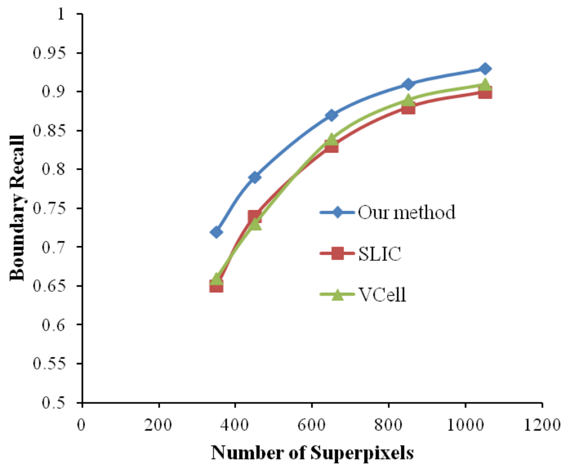


Figure 3.8: Comparisons between uniform and adaptive superpixels for the chapel image (a-d) and the portrait image (e-h). (a, e) Uniform superpixels; (b, f) adaptive superpixels; and (c, d, g and h) zoom-in details of (a), (b), (e) and (f), respectively.



(a) Undersegmentation error



(b) Boundary recall

Figure 3.9: Segmentation accuracy comparison. (a) Undersegmentation error; and (b) boundary recall.

noise and increases the accuracy of clustering results. We also develop an adaptive superpixel generation algorithm based on the local HEWCVT model. This new algorithm is capable of generating adaptive superpixels and preserving local image features efficiently. The accuracy and efficiency of the adaptive superpixel algorithm is also demonstrated by various types of images. Our future work will focus on applying adaptive superpixels to medical image registration and motion tracking.

Chapter 4

Centroidal Voronoi Tessellation Based Polycube Construction for Adaptive All-Hexahedral Mesh Generation

Given an input surface triangle mesh, the polycube construction can be viewed as a mesh segmentation task which assigns each triangle to one of axis-aligned planes in parametric domain. With several connectivity constraints, the resulting segments result in a correct polycube structure. In recent years, a lot of efforts have been put into the research of centroidal Voronoi tessellation (CVT) [25, 28, 54, 27] for various applications, where the key idea is to partition a data-set with an optimization process of updating generators with respect to a specific energy function. Inspired by these techniques, in this thesis we develop a novel automatic polycube construction algorithm by using CVT based mesh segmentation. Improving upon the classic CVT method, we develop a novel harmonic boundary-enhanced centroidal Voronoi tessellation (HBECVT) method by introducing local neighbouring information into the energy function. After the construction of polycube via parametric mapping, we can generate uniform all-hex mesh, volumetric T-mesh through octree subdivision in both parametric and physical domains. We also generate adaptive all-hex meshes by first generating a primal hybrid octree with polyhedral cells using an improved cutting procedure and then extracting the dual all-hex mesh. The key contributions of our work include:

1. A new CVT-based algorithm is developed to automatically and robustly construct polycubes for general arbitrary geometric domains. By introducing the local neighbouring information into the energy function, our HBECVT-based algorithm improves the surface segmentation and polycube construction by reducing non-monotone boundaries. It is also less sensitive to the initializations and noise; and

2. An improved hybrid octree structure is introduced for adaptive all-hex mesh generation with limited propagation. Our algorithm combines advantages of grid based method and polycube based method, which yields adaptive all-hex meshes with good quality around the boundary.

4.1 Centroidal Voronoi Tessellation and Polycube Construction

In this section, we extend the CVT-based model to mesh segmentation for automatic polycube construction. Given an input surface triangle mesh T, we partition it into a set of clusters $U = \{U_l\}_{l=1}^{L}$ and then use surface parametric mapping to construct the polycubes. For a certain partition $U = \{U_l\}_{l=1}^{L}$ of T, there exists a valid polycube structure if it satisfies the following three conditions:

1. T is partitioned into six clusters $L = 6$ and each cluster corresponds to one principal orientation;

2. Each segment of all clusters $\{U_l\}_{l=1}^{L}$ has more than three boundaries, where each boundary is shared by two neighboring segments; and

3. Each corner vertex is shared by three segments of three perpendicular clusters in parametric domain.

We set these sufficient conditions as topological constraints in the following CVT-based surface segmentation procedure.

4.1.1 Centroidal Voronoi Tessellation

Given an input surface triangle mesh T, let the dataset $X = \left\{x_{T(i)}\right\}_{i=1}^{n}$ denote all the unit normals $x_{T(i)}$ of the triangle mesh in the normal space, where n is the total number of triangles and $T(i)$ represents the i^{th} triangle in the physical space. Let $C = \{c_l\}_{l=1}^{L}$ denote a set of typical unit normal vectors. We set $L = 6$ in this thesis since we want to segment the mesh into six clusters, where each triangle is assigned to one of them. The Voronoi region V_k $(k = 1,\ldots,L)$ in X corresponding to the normal vector c_k is defined as

$$V_k = \left\{x_{T(i)} \in X : dist\left(x_{T(i)}, c_k\right) \leq dist\left(x_{T(i)}, c_l\right),\ for\ l = 1,\ldots,L\right\}, \tag{4.1}$$

where $dist\left(x_{T(i)}, c_k\right) = \left|x_{T(i)} - c_k\right|$ is the Euclidean distance between $x_{T(i)}$ and c_k in the normal space. Note that here the differences between normal vectors in the normal space are compared instead of that in color space [27]. The set $V = \{V_l\}_{l=1}^{L}$ is called a *Voronoi tessellation* of the data set X. The set of chosen unit normals $C = \{c_l\}_{l=1}^{L}$ are referred to as the *Voronoi generators*. Obviously, we have $X = \cup_{l=1}^{L} V$ and $V_i \cap V_j = \emptyset$ if $i \neq j$. Given any set of values $C = \{c_l\}_{l=1}^{L}$ and any partition $U = \{U_l\}_{l=1}^{L}$ of X, the classical clustering energy of $(C;U)$ can be defined as

$$E(C;U) = \sum_{l=1}^{L} \sum_{x_{T(i)} \in U_l} \left|x_{T(i)} - c_l\right|^2. \tag{4.2}$$

Note that $U = \{U_l\}_{l=1}^{L}$ are not necessarily the Voronoi regions corresponding to the set of generators $C = \{c_l\}_{l=1}^{L}$ in the definition. Once we have determined the clusters $\{U_l\}_{l=1}^{L}$ for a given input mesh represented in the normal space by $x_{T(i)}$ for each triangle $T(i) \in D$, where $D = \{T(i) : i = 1, \ldots, n\}$ is an index set, there is a natural segmentation of the surface which has L segments $D = \{D_l\}_{l=1}^L$ in the physical space defined by $D_l = \left\{T(i) : x_{T(i)} \in U_l\right\}$. Given a partition of X, denoted by $U = \{U_l\}_{l=1}^L$, the *centroid* of each cluster U_l is defined to be the normal vector c_l^* which minimizes the classical clustering energy with respect to c_l, thus we have

$$c_l^* = \frac{1}{|D_l|} \sum_{x_{T(i)} \in U_l} x_{T(i)}, \tag{4.3}$$

where $|D_l|$ is the number of triangles in D_l. For an arbitrary Voronoi tessellation $\left(\{c_l\}_{l=1}^L ; \{V_l\}_{l=1}^L\right)$ of X, we often have $c_l \neq c_l^*$ for $l = 1, \ldots, L$, where $\left\{c_l^*\right\}_{l=1}^L$ are the corresponding centroids of $\{V_l\}_{l=1}^L$.

Definition. If the generators of the Voronoi regions $\{V_l\}_{l=1}^L$ of X equal to their corresponding centroids, i.e., $c_l = c_l^*$ for $l = 1, \ldots, L$, then we call the Voronoi tessellation $\{V_l\}_{l=1}^L$ a centroidal Voronoi tessellation (CVT) of X and refer to $\{c_l\}_{l=1}^L$ as the corresponding CVT generators.

The construction of CVT can be viewed as an energy minimization process, where the classical clustering energy is minimized iteratively to update centroids of the clusters. The classical clustering energy $E(C; U)$ is minimized only if $(C; U)$ form a CVT of X, i.e., U are Voronoi regions of X associated with the generators C and simultaneously C are the corresponding centroids of Voronoi regions U. The detailed implementation of CVT is explained as follows

Algorithm of CVT (Implementation)

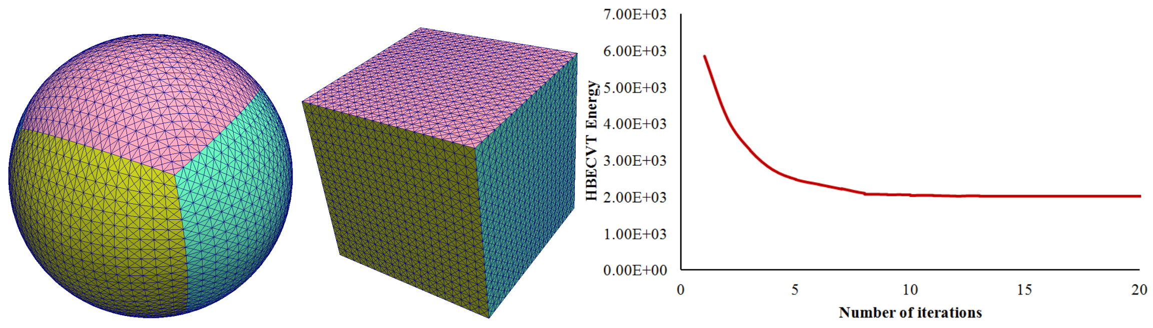
Given a surface triangle mesh $X = \left\{x_{T(i)}\right\}_{i=1}^n$, positive integer $L = 6$ and error tolerance ε ($\varepsilon = 10^{-4}$ in this thesis), we first apply the principal components analysis (PCA) to surface normals and take three principal normal vectors and their opposite normals as the initialization of generators $\{c_l\}_{l=1}^{L}$. E_i denotes the CVT energy in the i^{th} iteration. Then perform the following:

1. Determine the Voronoi clusters $\{V_l\}_{l=1}^{L}$ of X associated with $\{c_l\}_{l=1}^{L}$ by assigning each triangle to the nearest cluster by (4.1). If triangle $T(i)$ has any neighbouring triangle being assigned to the opposite direction cluster, transfer it to one of other neighbouring clusters;

2. For each cluster V_l ($l = 1, \ldots, L$), determine the cluster centroids c_l^* by (4.3); and

3. If a termination criterion such as $\frac{E_{i+1}-E_i}{E_i} < \varepsilon$ is reached, return $(\{c_l\}_{l=1}^{L}; \{V_l\}_{l=1}^{L})$ and exit; otherwise, set $c_l = c_l^*$ for $l = 1, \ldots, L$ and return to Step 1.

To improve the connectivity and satisfy the constraints of polycube construction, we merge each segment in the resulting segmentation with less than four boundaries to one of its neighbouring clusters. Fig. 4.1 shows the segmentation result of a Sphere model. We randomly initialize the generators $\{c_l\}_{l=1}^{L}$ by choosing L random triangles in the mesh and take their normal vectors as the initialization. Fig. 4.1(a) shows the final segmentation result after 20 iterations, where neighbouring patches are rendered with different colors. The curve in Fig. 4.1(c) shows that the CVT energy keeps decreasing and converges after 20 iterations.



(a) (b) (c)

Figure 4.1: Results of the Sphere model. (a) Segmentation result after 20 iterations; (b) parametric mapping result in parametric domain; and (c) the classic CVT energy curve.

4.1.2 Polycube Parametric Mapping

After the CVT-based mesh segmentation, each cluster is now aligned with one of the six principal axes ($\pm X$, $\pm Y$, $\pm Z$) which will result in a unique valid polycube structure. The next step is to create a surface parametric mapping which aims at creating a one-to-one mapping f from the given surface T to a parameter domain T^*.

We correspond the resulting six clusters to three PCA principal normals and their opposite normals in parametric domain. The polycube structure is then extracted by positioning all the corner points of each segment with hard planarity constraints in the corresponding principal axis. For boundary vertices and interior vertices in each segment, we use a planar domain parameterization [32] to create a parametric mapping between the boundary triangle mesh T and planar polygon in parametric space. Each surface boundary edge is mapped onto its corresponding edge of the polycube via a chord length parameterization, in which the parameter value increases proportionally to the chord length from the start of the edge. In this way, for each cluster U_l we have the parameterization for its boundary nodes. We then apply the Embed algorithm [142] to position the interior vertices of each segment, which uses a nonlinear method to embed a triangle mesh in a planar boundary by minimizing the unsigned area of the planar triangulation. This planar domain parameterization computes a bijective map from the input mesh to the polycube surface and enforces vertices in each segment to have the same coordinate value along the corresponding axis. Note that polycubes generated in [134] and [70] consist of either one unit cube or multiple equal-sized cubes with two neighbouring cubes sharing one face. Differently, our generated polycubes are general polyhedra composed of polygonal surfaces in parametric domain. Fig. 4.1(b) shows the mapping result for the Sphere model, we can observe that each triangle of the Sphere is mapped to its corresponding position in the parametric domain and the mapping is bijective.

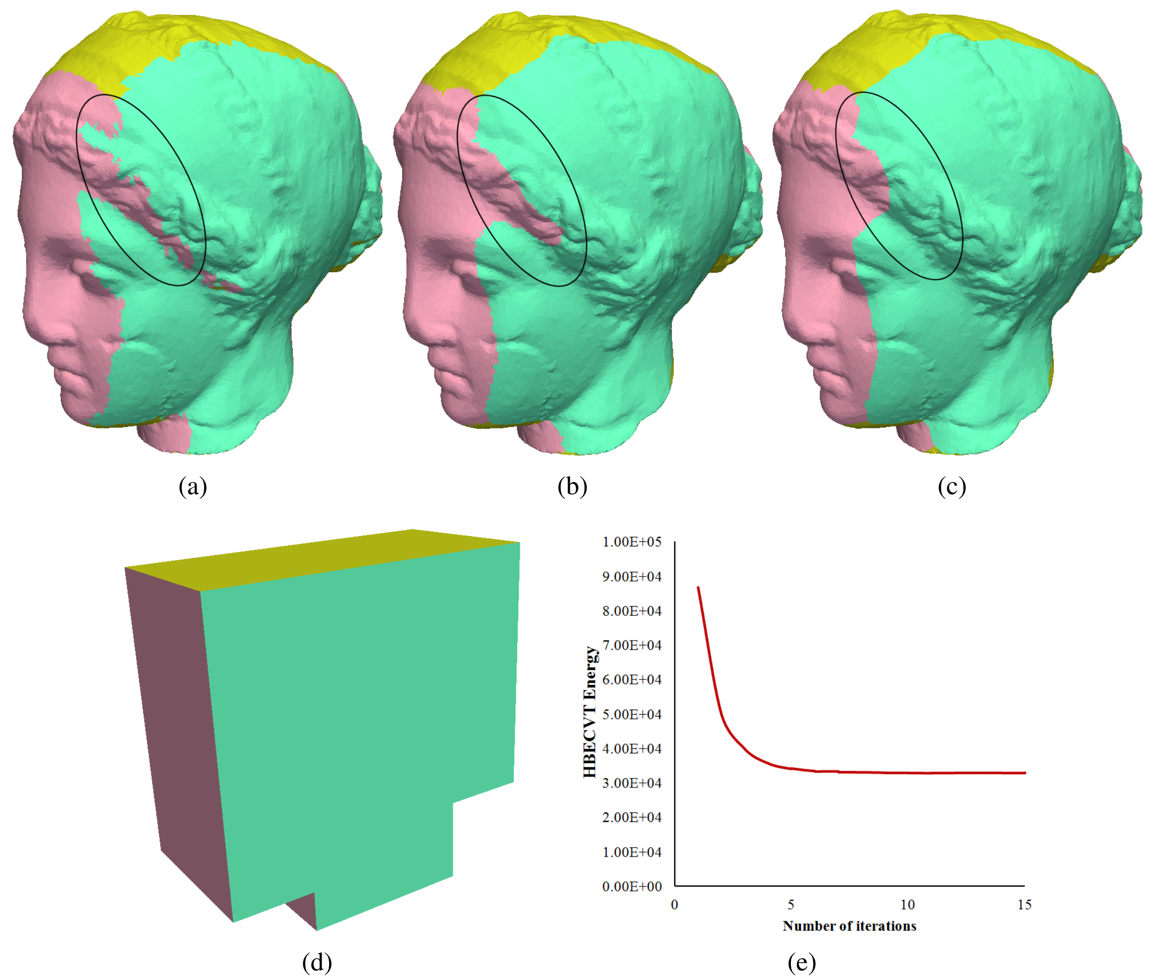


Figure 4.2: Results of the Igea model. (a) Classic CVT segmentation result; (b-c) HBECVT segmentation results after 4 and 10 iterations respectively; (d) parametric mapping result; and (e) HBECVT energy curve. Black circle in (a) highlights the region with non-monotone boundary, which is reduced gradually in (b-c).

4.2 Harmonic Boundary-Enhanced Centroidal Voronoi Tessellation

The classic CVT method can generate segmentation with valid polycube structure for simple models, such as sphere and torus. However, the classic CVT has some drawbacks for complex geometry: since it only considers normal information but does not take any physical information into account, the resulting segmentation may give us too many details and sometimes these extra details may be noises. Moreover, the classic CVT method can not eliminate non-monotone boundaries [74], where the direction changes, as shown in Fig. 4.2(a). As mentioned in [74], the non-monotone boundaries can result in extreme distortion when they are mapped to corresponding polycube edges. The classic CVT imposes hard membership on the triangle elements, that is, each triangle is assigned to exactly one cluster. Each triangle only has an influence over the centroid to which cluster it is assigned, making it sensitive to initialization.

To overcome limitations of the classic CVT method, in this section we present an extended model named harmonic boundary-enhanced centroidal Voronoi tessellation (HBECVT). Inspired by the EWCVT model [131], we similarly define a new distance metric by adding a boundary-enhanced term in order to include local neighbouring information. For each triangle $T(i) \in D$, we denote a local neighborhood as $N_\omega(T(i))$, which is a ω-rings neighboring region centered at triangle $T(i)$. The boundary-enhanced distance between $x_{T(i)}$ and c_k can be defined as

$$dist\left(x_{T(i)}, c_k\right) = \sqrt{\left|x_{T(i)} - c_k\right|^2 + \lambda \tilde{n}_k(T(i))}, \tag{4.4}$$

where $\tilde{n}_k(T(i))$ represents the number of triangles that do not belong to the k^{th} cluster within $N_\omega(T(i))$. λ is a positive weighting factor to control the balance between the Euclidean distance in the normal space and the boundary-enhanced term in the physical space. Note that the new distance term combines the normal information between the triangle $T(i)$ and clusters together with the local neighbouring information of the triangle $T(i)$. Similarly, we can define the boundary-enhanced Voronoi region V_k ($k = 1, \ldots, L$) in X corresponding to the Voronoi generator c_k as

$$V_k = \left\{x_{T(i)} \in X : dist\left(x_{T(i)}, c_k\right) \leq dist\left(x_{T(i)}, c_l\right),\ for\ l = 1, \ldots, L\right\}. \tag{4.5}$$

The boundary-enhanced term $\tilde{n}_k(T(i))$ extends the edge-weighted idea of EWCVT [131] to the geometry space, and represents the probability that triangle $T(i)$ belongs to the

k^{th} cluster in physical space by introducing local neighbouring information. $\tilde{n}_k(T(i))$ is small if the majority of neighbourhoods within $N_\omega(T(i))$ belong to the k^{th} cluster, and vice versa. For a noisy triangle or a triangle near non-monotone boundaries, since the majority of its neighbourhoods belong to the same cluster, the use of boundary-enhanced term increases the possibility that it will be assigned to the same cluster with the majority of its neighbourhoods. As a result, the robustness to the noise can be improved and the smoothness of the resulting boundaries is controlled effectively. Fig. 4.3(a) shows a two-ring neighbourhood region of the boundary triangle $T(1)$ with clusters U_1 and U_2, where $T(1)$ is initially assigned to the cluster U_2. Since $dist\left(x_{T(1)}, c_1\right) = \sqrt{\left|x_{T(1)} - c_1\right|^2 + 16\lambda}$ and $dist\left(x_{T(1)}, c_2\right) = \sqrt{\left|x_{T(1)} - c_2\right|^2 + 20\lambda}$, $T(1)$ will more likely be closer to the generator of U_1 when λ is large. Similarly, Fig. 4.3(b) shows a two-ring neighbourhood region of the boundary triangle $T(2)$ with clusters U_1, U_2 and U_3, where $T(2)$ is initially assigned to the cluster U_3. Since $dist\left(x_{T(2)}, c_1\right) = \sqrt{\left|x_{T(2)} - c_1\right|^2 + 20\lambda}$, $dist\left(x_{T(2)}, c_2\right) = \sqrt{\left|x_{T(2)} - c_2\right|^2 + 24\lambda}$ and $dist\left(x_{T(2)}, c_3\right) = \sqrt{\left|x_{T(2)} - c_3\right|^2 + 28\lambda}$, $T(2)$ will more likely be closer to the generator of U_1 when λ is large.

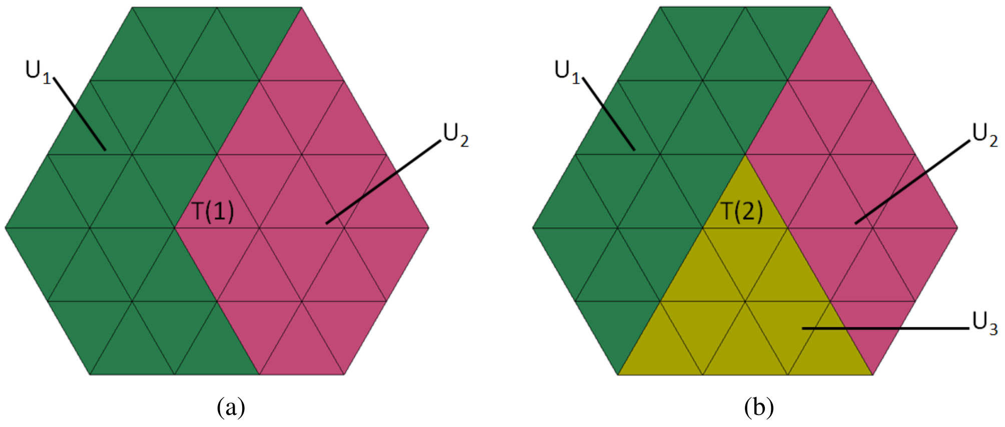


Figure 4.3: Neighbourhood regions ($\omega = 2$) of two triangle $T(1)$ and $T(2)$. (a) Boundary triangle $T(1)$ of two neighbouring clusters; and (b) boundary triangle $T(2)$ of three neighbouring clusters.

We then use the boundary-enhanced distance to define a new energy function by using the harmonic average form [46]. For any set of generators $C = \{c_l\}_{l=1}^{L}$ and any partition $U = \{U_l\}_{l=1}^{L}$ of X, we define the harmonic boundary-enhanced clustering energy as

$$E^H(C;U) = \sum_{i=1}^{n} \left(L \Big/ \sum_{l=1}^{L} \left(\left| x_{T(i)} - c_l \right|^2 + \lambda \tilde{n}_l(T(i)) \right)^{-1} \right). \tag{4.6}$$

The above energy function uses squared boundary-enhanced distance from all the generators to calculate the harmonic average for each triangle. To calculate the updated centroids $\left\{c_k^*\right\}_{k=1}^{L}$, we minimize the harmonic boundary-enhanced clustering energy function with respect to the generator c_k ($k = 1, \ldots, L$). Since the local boundary-enhanced energy term $\tilde{n}_k(T(i))$ at each triangle $T(i)$ is fixed, we have

$$\frac{\partial\left(\left|x_{T(i)} - c_k\right|^2 + 2\lambda \tilde{n}_k(T(i))\right)}{\partial c_k} = \frac{\partial\left(\left|x_{T(i)} - c_k\right|^2\right)}{\partial c_k} = 2\left(x_{T(i)} - c_k\right). \tag{4.7}$$

We then calculate the derivative of $E^H(C;U)$ with respect to the generator c_k and set it to be zero as follows

$$\frac{\partial E^H(C;U)}{\partial c_k} = -L \sum_{i=1}^{n} \frac{2\left(x_{T(i)} - c_k\right)}{dist^4\left(x_{T(i)}, c_k\right)\left(\sum\limits_{l=1}^{L} dist^{-2}\left(x_{T(i)}, c_l\right)\right)^2} = 0, \tag{4.8}$$

where $dist\left(x_{T(i)}, c_k\right) = \sqrt{\left|x_{T(i)} - c_k\right|^2 + \lambda \tilde{n}_k(T(i))}$, then we can obtain an iterative formula as follows:

$$c_k^* = \frac{\sum\limits_{i=1}^{n} u_{ik}^H x_{T(i)}}{\sum\limits_{i=1}^{n} u_{ik}^H}, \tag{4.9}$$

where $u_{ik}^H = \left(\sum\limits_{l=1}^{L} \frac{dist^2(x_{T(i)}, c_k)}{dist^2(x_{T(i)}, c_l)}\right)^{-2}$. We can observe that the calculation of c_k^* is influenced by all triangles. u_{ik}^H can be viewed as a soft membership function which reflects the degree of possibility of $T(i)$ that is associated with the kth cluster and how much influence $T(i)$ has in the calculation of new centroids. For an arbitrary Voronoi tessellation $\left(\{c_l\}_{l=1}^L; \{V_l\}_{l=1}^L\right)$ of X where $V = \{V_l\}_{l=1}^L$ are the corresponding boundary-enhanced Voronoi regions associated with $C = \{c_l\}_{l=1}^L$, we often have $c_l \neq c_l^*$, for $l = 1, \ldots, L$, where $\left\{c_l^*\right\}_{l=1}^L$ are the corresponding centroids of $\{V_l\}_{l=1}^L$.

Definition. If the generators of the harmonic boundary-enhanced Voronoi regions $\{V_l\}_{l=1}^L$ of X equal to their corresponding centroids, i.e., $c_l = c_l^*$ for $l = 1, \ldots, L$, then we call the Voronoi tessellation $\{V_l\}_{l=1}^L$ a harmonic boundary-enhanced centroidal Voronoi tessellation (HBECVT) of X and refer to $\{c_l\}_{l=1}^L$ as the corresponding HBECVT generators.

The construction of HBECVT is also an energy minimization process, where the centroids are updated by minimizing the corresponding harmonic boundary-enhanced clustering energy function. The harmonic boundary-enhanced clustering energy $E^H(C; V)$ is minimized only if $(C; V)$ form a HBECVT of X, i.e., V are Voronoi regions of X associated with the generators C and simultaneously C are the corresponding centroids of the region V. The detailed implementation of HBECVT is explained as follows.

Algorithm of HBECVT (Implementation)

Given a surface triangle mesh $X = \left\{x_{T(i)}\right\}_{i=1}^n$, positive integer $L = 6$, weighting factor λ, neighbourhood size ω and error tolerance ε (*e.g.* $\varepsilon = 10^{-4}$), we apply the CVT algorithm to find a good initial configuration and set centroids of resulting clusters as the initialization of generators $\{c_l\}_{l=1}^L$. E_i^H denotes the HBECVT energy in the i^{th} iteration. Then perform the following:

1. Assign each triangle T(i) to the cluster whose generator has the shortest distance to it by (4.5), and determine the boundary-enhanced Voronoi clusters $\{V_l\}_{l=1}^L$ of X associated with boundary-enhanced Voronoi generators $\{c_l\}_{l=1}^L$. If triangle $T(i)$ has any neighbouring triangle being assigned to the opposite direction cluster, transfer it to one of other neighbouring clusters;

2. For each boundary-enhanced Voronoi cluster V_l ($l = 1, \ldots, L$), determine the cluster centroids c_l^* by (4.9); and

3. If a termination criterion such as $\frac{E_{i+1}^H - E_i^H}{E_i^H} < \varepsilon$ is reached, return $(\{c_l\}_{l=1}^L; \{V_l\}_{l=1}^L)$ and exit; otherwise, set $c_l = c_l^*$ for $l = 1, \ldots, L$ and return to Step 1.

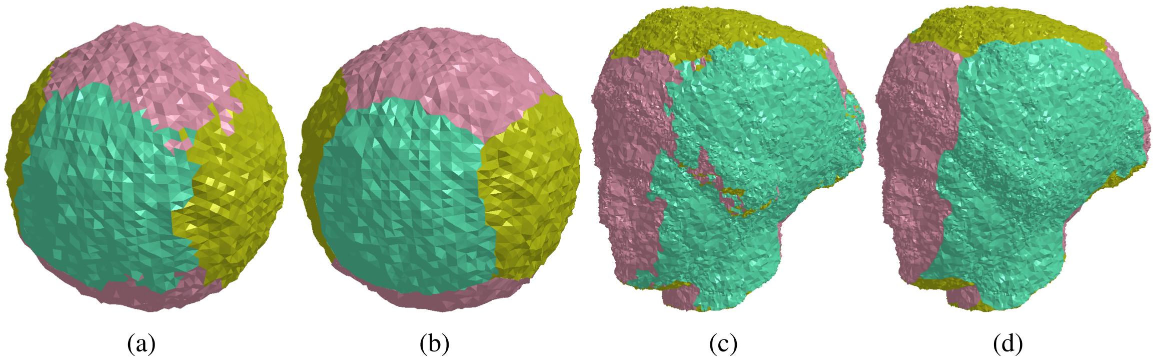


Figure 4.4: Segmentation results of noisy Sphere and noisy igea models. (a, c) Classic CVT segmentation results; and (b, d) HBECVT segmentation results.

In the resulting segmentation, we merge each segment with less than four boundaries to one of its neighbouring clusters in order to satisfy the constraints of polycube structure. Fig. 4.2 shows segmentation results of the Igea model using both classic CVT and HBECVT. The black circle in Fig. 4.2(a) highlights the region with non-monotone boundary, which can not be avoided for classic CVT method. Fig. 4.2(b) and Fig. 4.2(c) show the HBECVT segmentation results after 4 and 10 iterations respectively. Fig. 4.2(d) shows the parametric mapping result and Fig. 4.2(e) shows that the HBECVT energy keeps decreasing and converges after 10 iterations. From the results, we can observe that HBECVT eliminates non-monotone boundaries gradually over the iterations, which will lead to low distortion parametrization later. Fig. 4.4 shows segmentation results of noisy Sphere and noisy Igea models. We can observe that the classic CVT method can not generate compact polycubes, lots of unnecessary details caused by noise are segmented. Compared with the classic CVT, HBECVT generates compact and smooth segmentations by reducing non-monotone boundaries. For the HBECVT algorithm, the weighting parameter λ balances the clustering energy term in the normal space and the boundary-enhanced term in the physical space. Different λ values will result in different polycube structures, which can be used to control the compactness of the resulting polycube. Fig. 4.5 shows the segmentation and mapping results of polycube construction for the Bunny model with different λ values. From the result, we can observe that the increasing weighting parameter λ reduces the number of polycube faces with fewer number of singularities but increases the distortion.

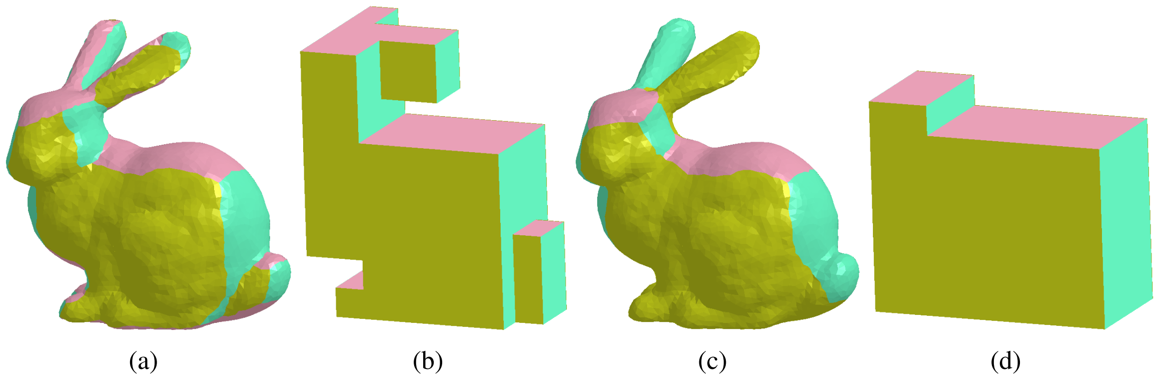


Figure 4.5: The segmentation and mapping results for the Bunny model. (a, c) The segmentation results, where $\lambda = 0.1$ for (a) and $\lambda = 1.0$ for (c); and (b, d) the corresponding parametric mapping results of (a, c) respectively.

Remark. The objective energy function in HBECVT uses the squared distances to all generators for each triangle, which means that all generators partially influence the harmonic average for each triangle. Compared to the classic CVT method, HBECVT is less affected by the presence of uncertainty or noise in the data since it has a soft membership function. EWCVT, FEWCVT and HEWCVT have been successfully used in image segmentation, our HBECVT extends the similar idea to mesh segmentation for the polycube construction. Similar to the edge-weighted idea from EWCVT, the boundary-enhanced term in HBECVT takes into account the local neighbouring information of each triangle, which tends to make the boundary of the final segmentation shorter and smoother. The harmonic average form of the energy function makes it less sensitive to initializations and noise. With some validity constraints described in Section 4.1, our HBECVT can automatically and robustly construct polycubes for general arbitrary geometric domains.

4.3 Adaptive Hexahedral Mesh Generation

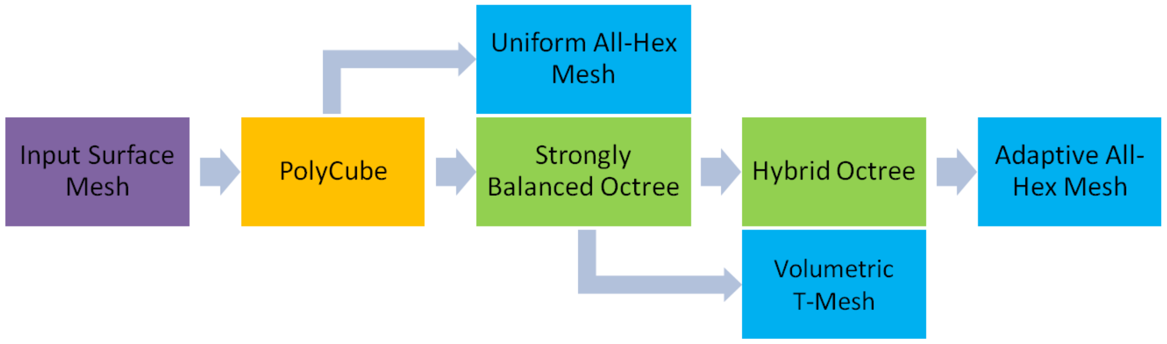


Figure 4.6: An overview of all-hex mesh generation based on polycube construction.

Based on the constructed polycubes, we can generate uniform hex meshes, T-spline control meshes (T-meshes) and adaptive all-hex dual meshes. The main steps are shown in Fig. 4.6. Uniform all-hex meshes C can be generated by first meshing the resulting

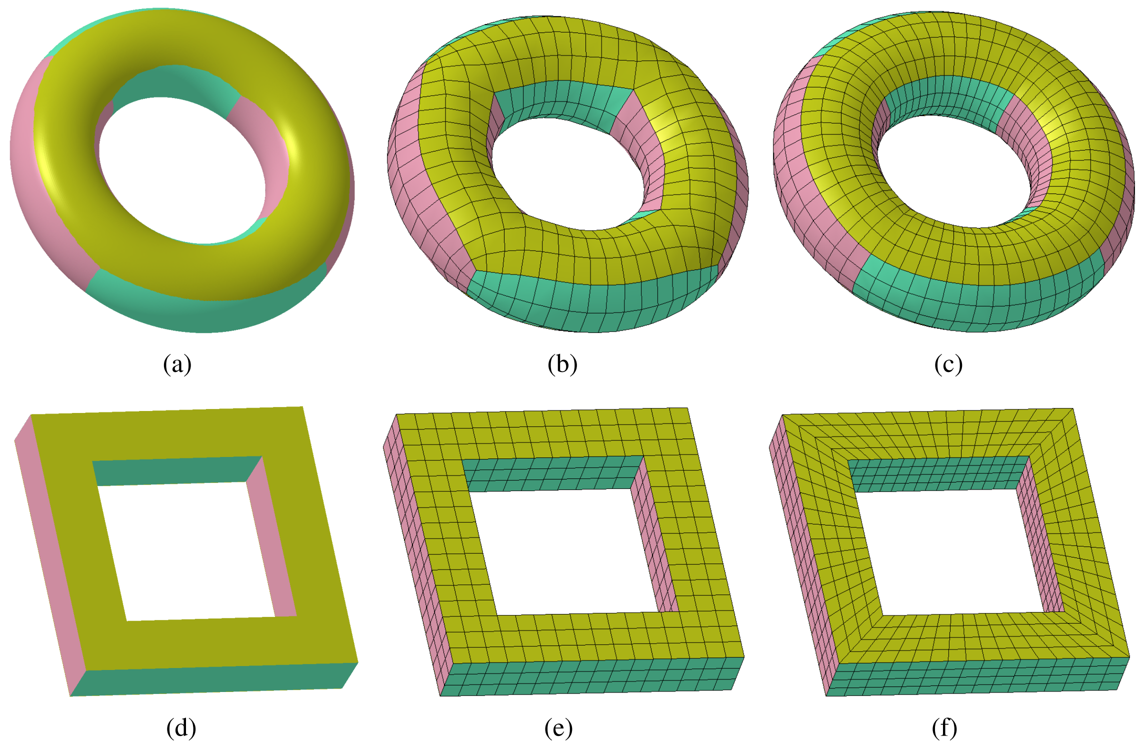


Figure 4.7: Results of the Torus model. (a, d) CVT based mesh segmentation and parametric mapping results; (b, e) uniform hex mesh in the physical and parametric domain without using the torus primitive; and (c, f) uniform hex mesh in the physical and parametric domain using the torus primitive.

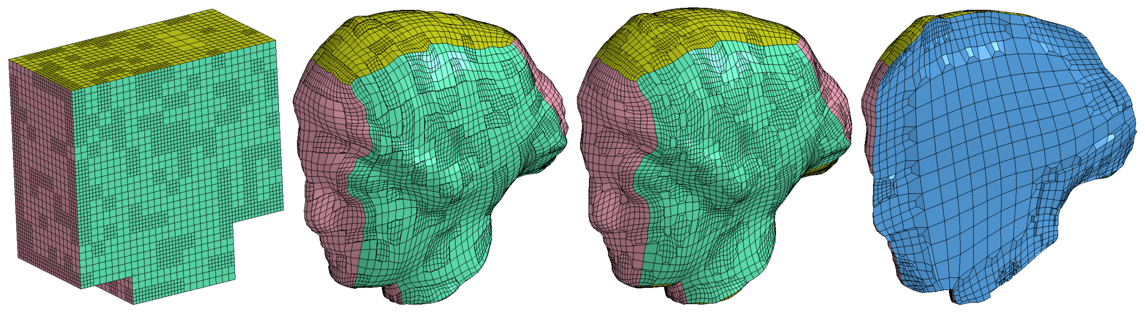

(a) (b) (c) (d)

Figure 4.8: All-hex meshing results of the Igea model. (a) Octree subdivision result in parametric domain; (b) octree subdivision result in physical domain (volumetric T-mesh); (c) adaptive all-hex dual mesh; and (d) some elements are removed to show the interior of (c).

polycube with an axis-aligned grid, and then mapping it back onto the input surface via parametric mapping for boundary vertices and interpolation for interior vertices. Each node C_i in C has its parameter coordinates and physical coordinates. For each boundary node, the physical coordinates are its mapped location on the triangular mesh. Given the parameter coordinates of one node C_i, we first find which triangle contains it, and then compute its physical coordinates using the barycentric coordinates in this triangle. In this thesis, we also combine our CVT based polycube construction method with Boolean operations [70] to deal with the torus-like objects. We split the input surface into different components, topologically equivalent to either a cube or a torus primitive. Since torus can be topologically considered as four consecutive cubes with the first face and the last face connected, instead of meshing the parametric domain in the traditional way (Fig. 4.7(b, e)), we mesh it by using four end-to-end octree domains, shown as Fig. 4.7(c, f). From the result, we can observe that the improved method generates zero irregular nodes on the surface, and the generated elements have much better quality (the minimum Jacobian 0.44 vs. 0.23).

Starting from the coarse uniform mesh C, we subdivide each element into eight smaller ones recursively to get the adaptive octree, which will be the volumetric T-mesh in the physical space. We adopt a surface approximation error to control the local refinement. The balancing and pairing rules are used here to build a strongly balanced octree [79, 42]. Fig.

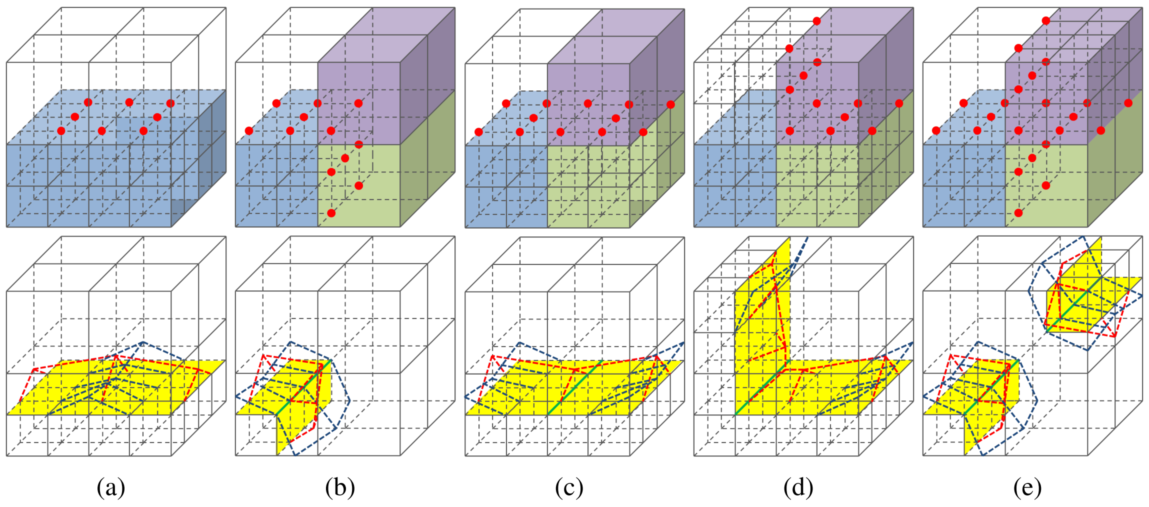


Figure 4.9: First row: five unique transition cases, where octants with the same color belong to the same block; second row: the corresponding cutting processes in our method. (a) Transition on face; and (b-e) transitions on edge.

4.8(a) shows the octree subdivision result of the Igea model in parametric domain and Fig. 4.8(b) shows the volumetric T-mesh result. Based on the strongly balanced octree, we can also generate adaptive all-hex dual meshes. In the octree, eight children octants belonging to the same parent octant form an octree block since they have the same behaviour, that is, either all or none of them are subdivided. A transition case happens when the neighbouring blocks are at different levels. A cutting procedure is used in this thesis to eliminate hanging nodes and guarantee that each grid point in the resulting hybrid octree is always shared by eight polyhedral cells. There are five transition cases (see Fig. 4.9), including one transition case on face and four transition cases on edge. Compared to Maréchal's cutting method [79], our improved cutting method reduces the propagation by cutting in the finer blocks to eliminate hanging nodes. Please refer to [79, 42] for detailed discussions on the hybrid octree construction.

An adaptive all-hex dual mesh can be extracted from the constructed hybrid octree. Since all grid points are shared by eight polyhedral cells, the obtained dual mesh is all-hex. If sharp features are detected by CVT-based segmentation, they can be aligned and preserved during the mapping and projection. The extracted hex meshes may have poorly-shaped elements around the boundary, therefore the mesh quality needs to be improved. Pillowing [99] is a sheet-insertion technique that inserts one layer around the boundary. After the pillowing, optimization is implemented to further improve the quality [150]. We use BFGS method [70] to perform the optimization, where the objective function is equivalent to the maximization of the minimum scaled Jacobian. Fig. 4.8(c) shows the adaptive all-hex mesh of the Igea model and Fig. 4.8(d) shows interior cross section of the mesh.

4.4 Results and Discussion

We have applied the presented algorithms to several datasets, and generated both valid polycubes and adaptive all-hex meshes with good quality. All results were computed on a PC equipped with a 2.93 GHz Intel X3470 CPU and 8GB of Memory. Table 4.1 shows the statistics of all testing models. For HBECVT-based mesh segmentation, there are a total of two parameters that need to be defined: λ, the weighting parameter that balances the clustering energy and the boundary-enhanced energy; and ω, which defines the size of the local neighborhood. Since the effect of ω is essentially similar to that of λ. We choose a relatively small value of ω ($\omega = 2$) for all the examples in order to reduce the computational cost.

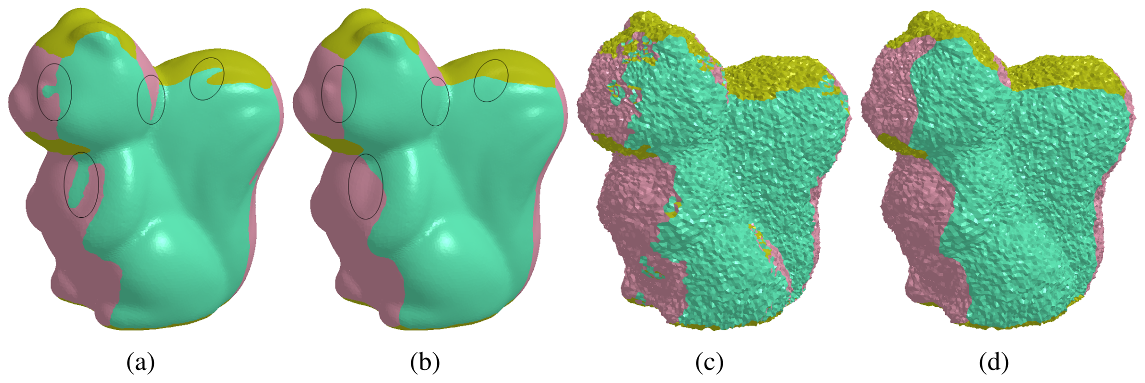


Figure 4.10: Segmentation results of the Squirrel model. (a) Classic CVT and HBECVT segmentation results of the noise free model, black circles in (a) highlight regions with non-monotone boundaries, which are removed in (b); and (c-d) classic CVT and HBECVT segmentation results of the noisy model.

Table 4.1: Statistics of all the tested models.

Model	Input mesh	λ	Number of	Octree	Output mesh	Jacobian	Jacobian (T-mesh)	Time
	(vertices, elements)		singularities	levels	(vertices, elements)	(worst, best)	(worst, best)	(s)
Igea	(4,500, 8,996)	0.08	16	3	(49,714, 44,586)	(0.14, 1.00)	(0.18, 1.00)	175.6
Squirrel	(30,254, 58,416)	0.08	16	3	(22,818, 19,482)	(0.14, 1.00)	(0.20, 1.00)	124.3
Bunny	(4,833, 9,662)	0.1	40	3	(36,832, 32,454)	(0.11, 1.00)	(0.13, 1.00)	143.5
Kitten	(80,000, 160,000)	0.1	28	3	(33,466, 30,104)	(0.10, 1.00)	(0.12, 1.00)	242.7
Rod	(4,406, 8,816)	0.001	20	2	(18,384, 16,066)	(0.22, 1.00)	(0.32, 1.00)	47.3
Base	(5,292, 10,600)	0.001	44	2	(12,720, 10,048)	(0.21, 1.00)	(0.33, 1.00)	53.8

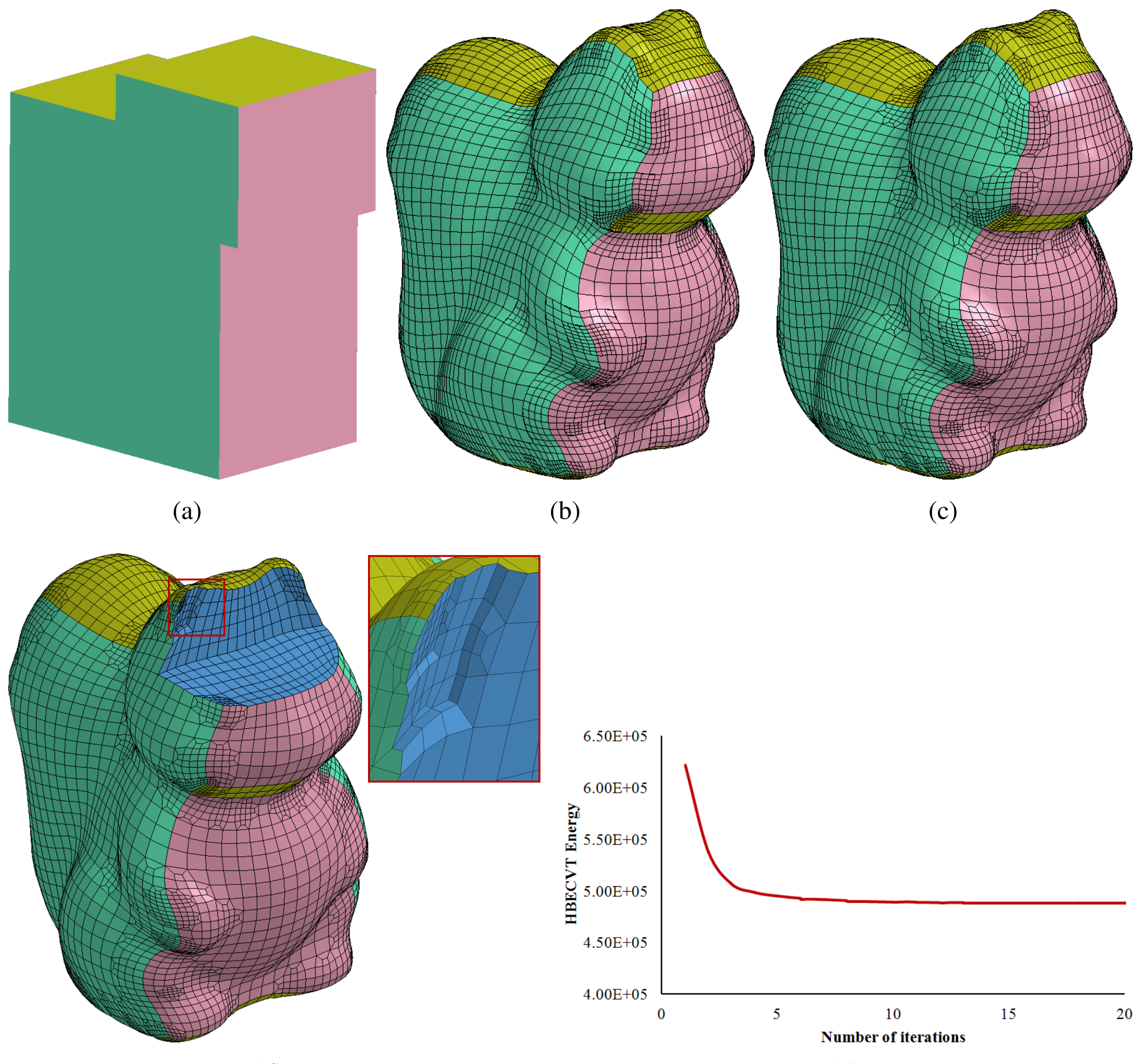


Figure 4.11: Results of the Squirrel model. (a) parametric mapping result; (b) subdivision result in physical domain (volumetric T-mesh); (c) adaptive all-hex dual mesh; (d) some elements are removed to show the interior of (c); and (e) HBECVT energy curve.

Fig. 4.10(a-b) show segmentation results of the noise-free Squirrel model using the classic CVT and HBECVT respectively. Black circles in Fig. 4.10(a) highlight regions with non-monotone boundaries, which can not be avoided in the classic CVT method. Fig.

4.10(b) shows the HBECVT segmentation results after 15 iterations. From the result, we can observe that HBECVT can eliminate non-monotone boundaries gradually and also smooth boundaries, which leads to low distortion parametrization later in polycube construction. Fig. 4.10(c-d) show segmentation results of the noisy Squirrel model using the classic CVT and HBECVT respectively. We can observe that the classic CVT method can not generate compact polycubes for noisy inputs, lots of unnecessary details caused by noise are segmented. Compared to the classic CVT, HBECVT generates compact and smooth segmentations by eliminating the effect of noises and reducing non-monotone boundaries.

After the polycube construction, we also generate volumetric T-meshes and adaptive all-hex dual meshes for several testing models: Igea (Fig. 4.8), Squirrel (Fig. 4.11), Bunny (Fig. 4.12), Kitten (Fig. 4.13), Rod (Fig. 4.14), and Base (Fig. 4.15). We use the scaled Jacobian to evaluate the quality of adaptive all-hex mesh and volumetric T-mesh. For each model, we show the HBECVT based segmentation result, HBECVT energy curve, corresponding polycube mapping result, subdivision results in both parametric domain and physical domain, and adaptive all-hex dual mesh result. The HBECVT based segmentation and polycube mapping results for the Bunny model are shown in Fig. 4.5(a-b). From Table 4.1, we can observe that the obtained adaptive all-hex meshes and volumetric T-meshes all have good quality (minimal Jacobian > 0.1). Figs. 4.14 and 4.15 show results of two CAD models with sharp features. Since the input surfaces of CAD models usually have clear features along each direction, the surface can be segmented with very few iterations. As shown in Figs 4.14(e) and 4.15(c), these two CAD models need much fewer iterations to reach convergence. For the Rod and Base model, the torus primitive is introduced to deal with torus-like objects and holes, which yields few number of extraordinary nodes and high-quality elements [70]. From the results we can observe that torus-like holes of these two models have no irregular nodes on the surface and the generated elements have very good quality.

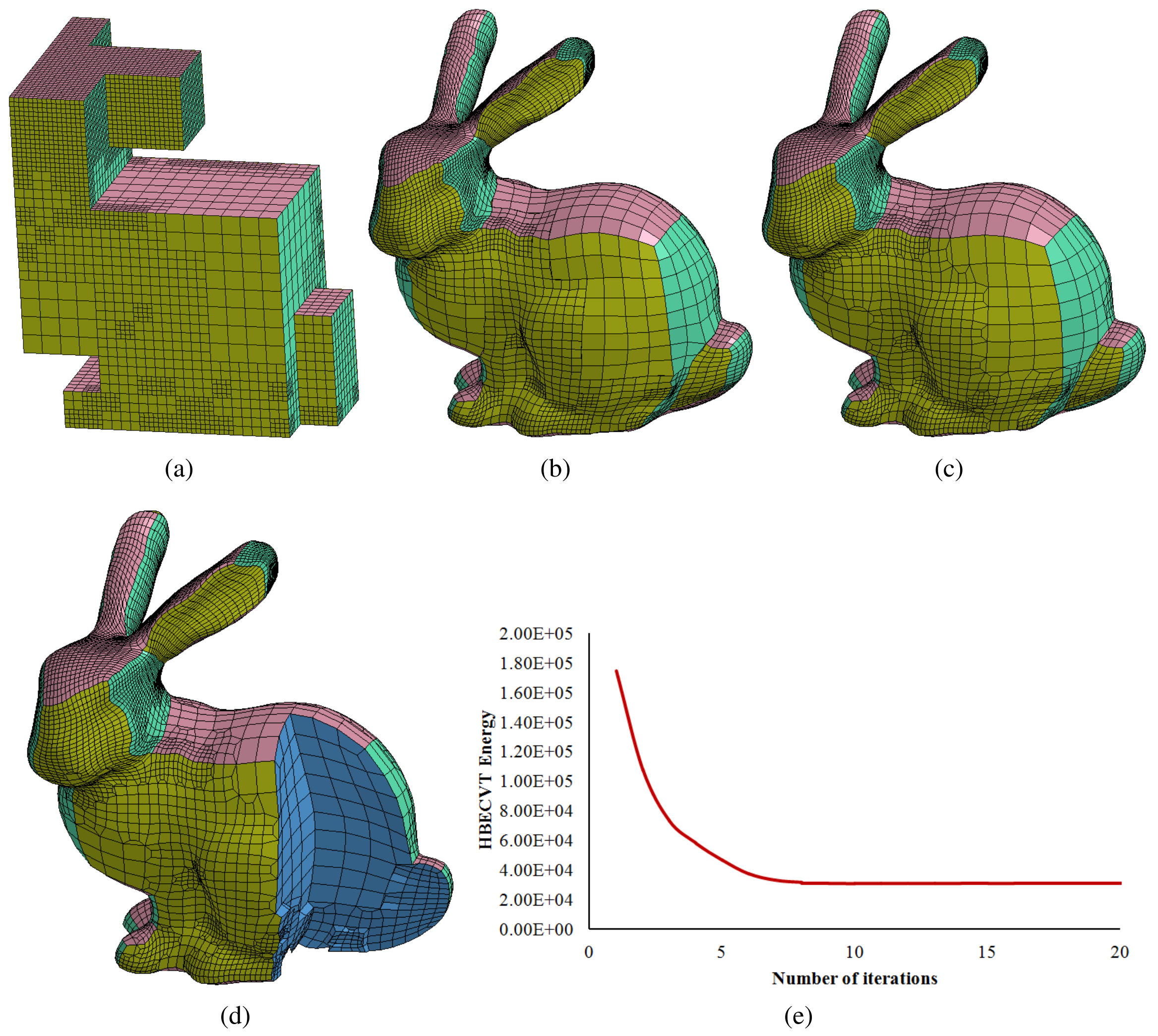


Figure 4.12: Results of the Bunny model. (a) Octree subdivision result in parametric domain; (b) subdivision result in physical domain (volumetric T-mesh); (c) adaptive all-hex dual mesh; (d) some elements are removed to show the interior of (c); and (e) HBECVT energy curve.

To compare the stability of the classic CVT and HBECVT for mesh segmentation and polycube construction, we can segment the test model N (N = 100 in this thesis) times with different initializations using both CVT and HBECVT. We count the number of singularities (cluster corners) of the resulting polycube for each test. Fig. 4.16 shows the testing results of the Igea, Squirrel, Bunny and Kitten models respectively, where CAD models are not tested using different initializations since their input surfaces have clear segmented features. For each model, we show the number of singularities for each test using both CVT and HBECVT. From the results we can observe that HBECVT is much more stable and less

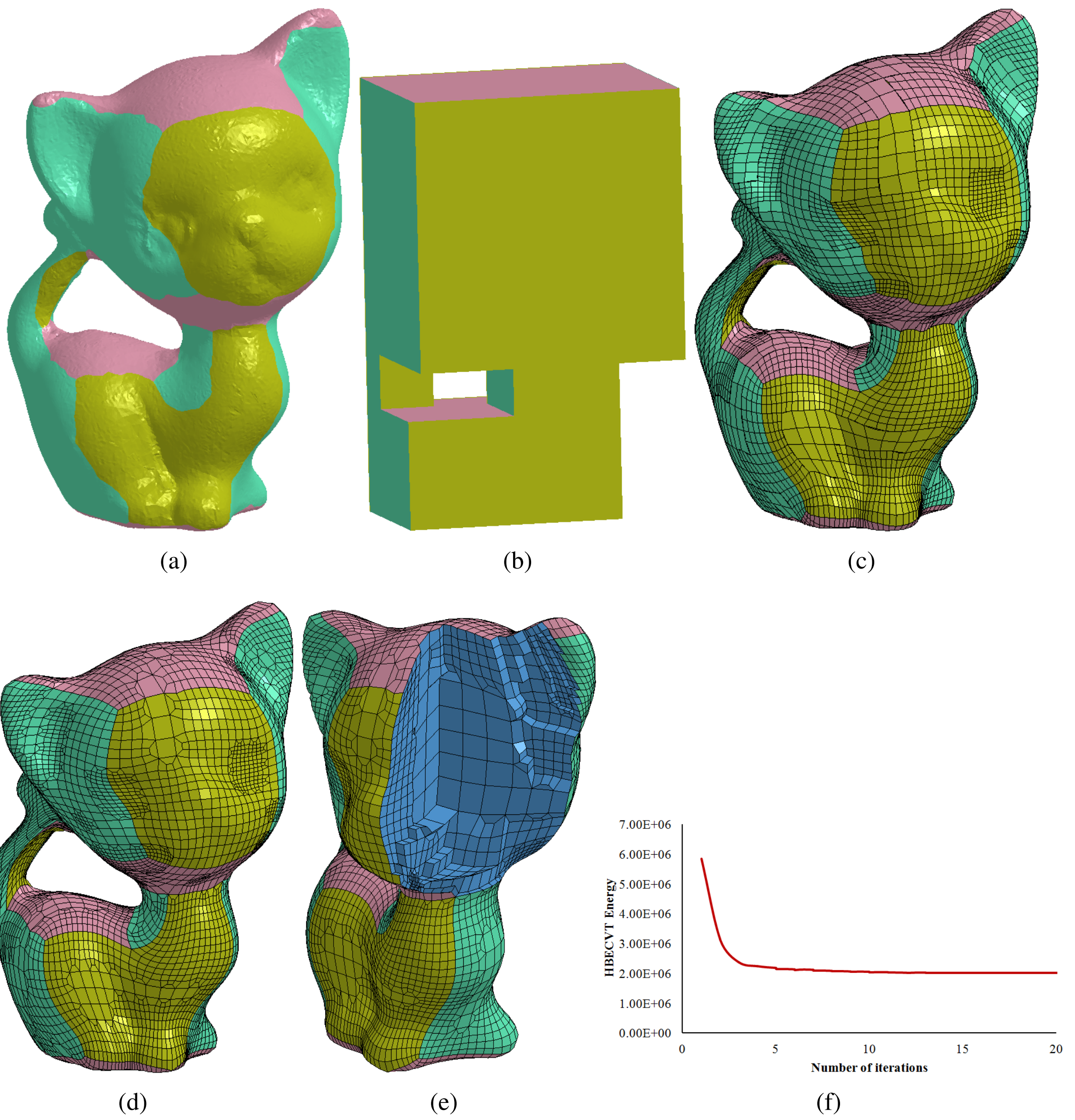


Figure 4.13: Results of the Kitten model. (a) HBECVT based mesh segmentation result; (b) parametric mapping result; (c) subdivision result in physical domain (volumetric T-mesh); (d) adaptive all-hex dual mesh; (e) some elements are removed to show the interior of (d); and (f) HBECVT energy curve.

sensitive to initializations. The classic CVT method usually generates more singularities, while HBECVT usually results in more compact polycube structure with fewer number of singularities. We also introduce a new metric named the segmentation coefficient of variation (SCV) to evaluate the stability of different methods. Based on the number of singularities value we get from each test, the SCV is defined as:

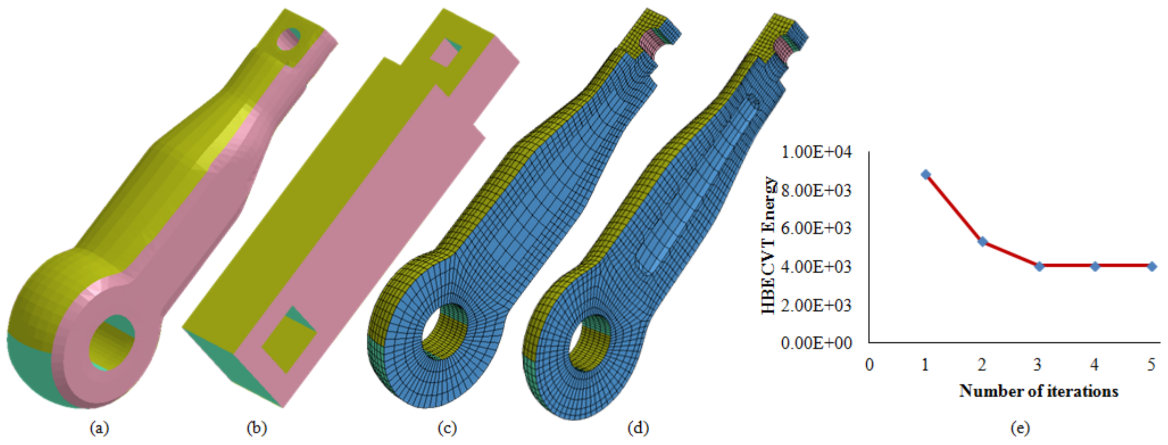


Figure 4.14: Results of the Rod model. (a) HBECVT based mesh segmentation result; (b) parametric mapping result; (c) subdivision result in physical domain (volumetric T-mesh); (d) adaptive all-hex dual mesh with some elements removed to show the interior; and (e) HBECVT energy curve.

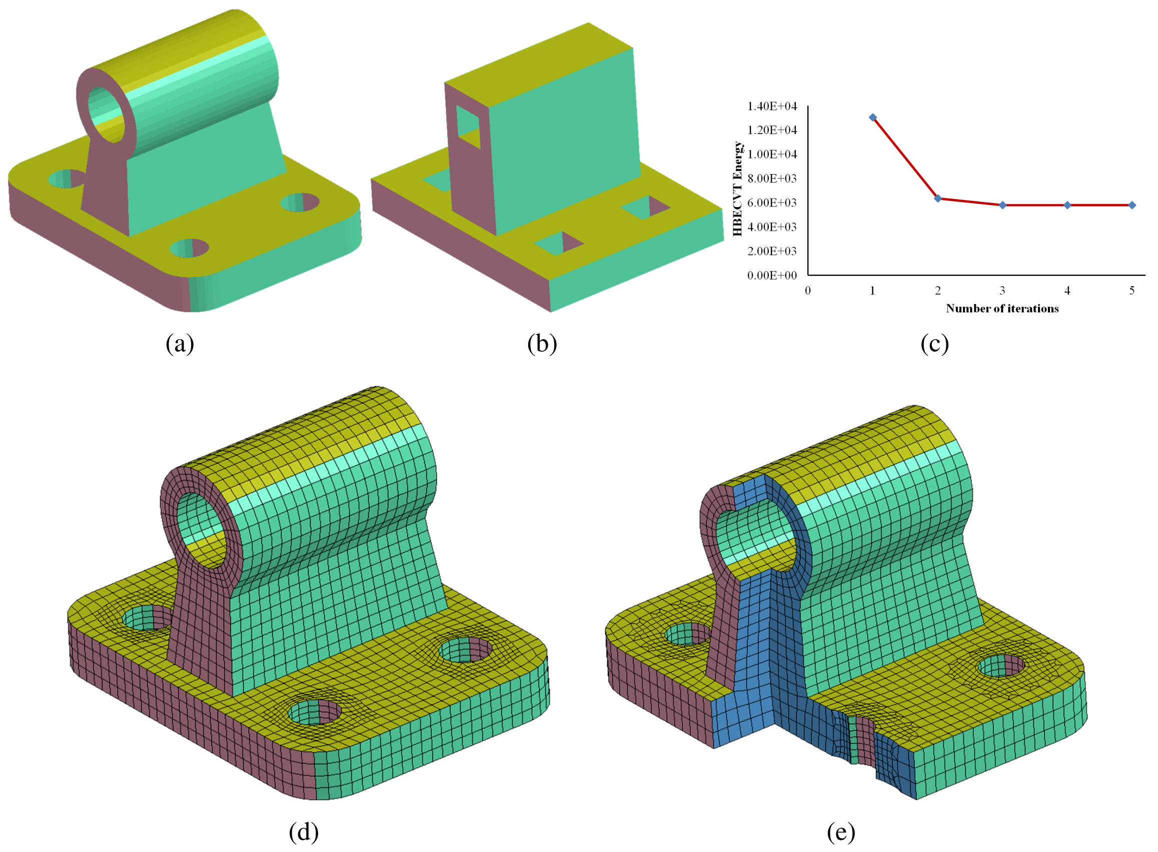


Figure 4.15: Results of the Base model. (a) HBECVT based segmentation result; (b) parametric mapping result; (c) HBECVT energy curve; (d) subdivision result in the physical domain (volumetric T-mesh); and (e) adaptive all-hex dual mesh with some elements removed to show the interior.

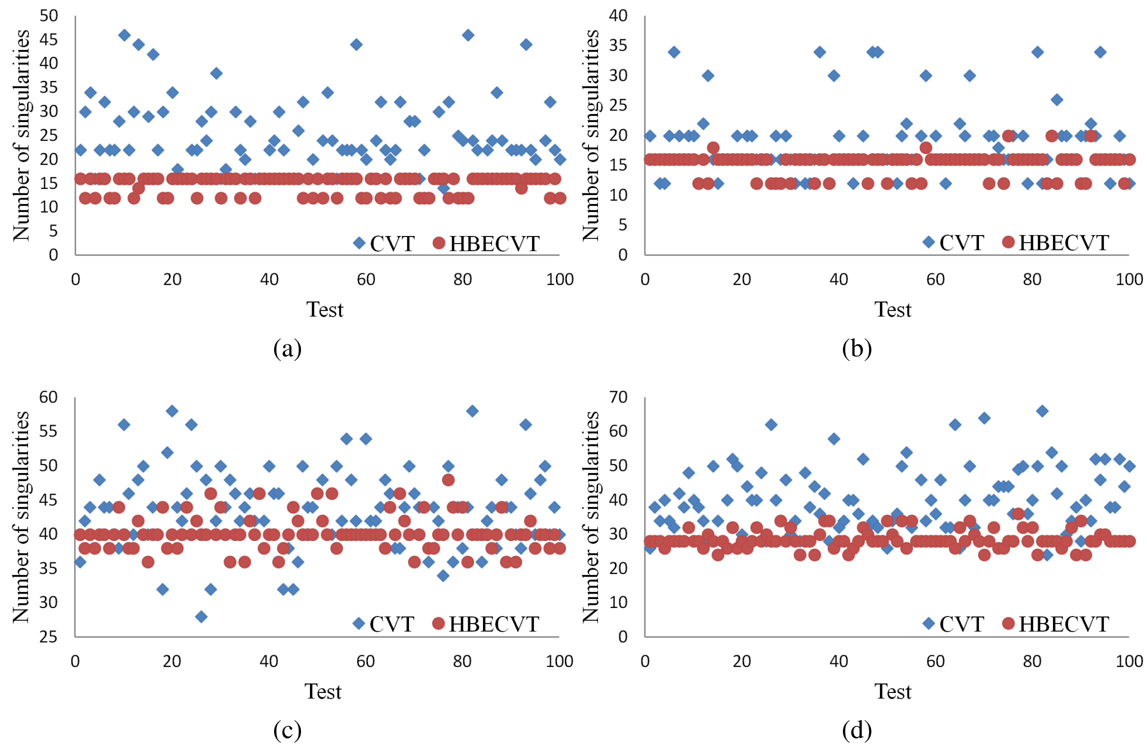


Figure 4.16: The number of resulting singularities of test models with random initializations. (a) Igea model; (b) Squirrel model; (c) Bunny model; and (d) Kitten model.

$$SCV = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(NumS_i - \overline{NumS}\right)^2}}{\overline{NumS}} \times 100\%, \tag{4.10}$$

where $NumS_i$ represents the number of singularities for the i^{th} test, and $\overline{NumS}$ represents the mean of the number of singularities. Small SCV values are usually considered low-variance, otherwise high-variance. The SCV results of CVT and HBECVT for the Igea model are 32.19% and 7.17% respectively, while 30.25% and 9.58% for Squirrel model, 20.21% and 7.53% for Bunny model and 25.27% and 8.48% for Kitten model. Compared to the classic CVT, it is obvious that HBECVT is more stable and accurate. Since different initializations may result in different polycube structures, in this thesis we apply PCA analysis first to get a better initialization which results in an unique solution. Users can also explore other ways to set the initial generators.

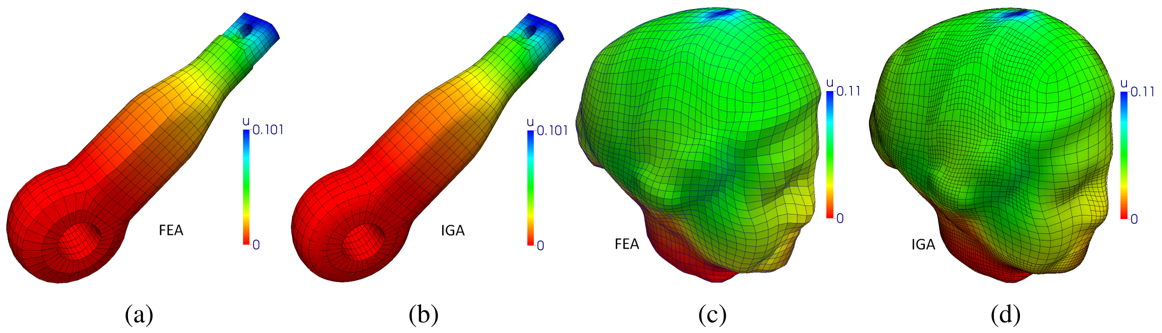


Figure 4.17: Analysis results. (a, b) FEA and IGA results for the Rod model; and (c, d) FEA and IGA results for the Igea model. All-hex meshes are shown in (a, c) and Bézier elements are shown in (b, d).

We tested Rod and Igea models for isogeometric analysis (IGA) by using the weighted T-splines [72]. Volumetric weighted T-splines can be generated from the T-meshes with local knot vectors and weighted basis functions, and Bézier elements are extracted for the IGA analysis. Here we solve the traditional linear elasticity problem. For both models, we fix the movement on one end and apply a displacement load on the other end (Young's modulus: 200 GPa; density: 7900 kg/m^3 and Poisson's ratio: 0.3). We also apply the finite element analysis (FEA) for each model with similar loading and boundary conditions. The displacement results obtained for both FEA and IGA are shown in Fig. 4.17. From the results we can observe that our algorithm can generate valid volumetric T-meshes and weighted T-splines for IGA applications.

Limitations. Both the classic CVT and HBECVT based methods can not guarantee a valid polycube construction with random initializations. With a bad initialization of generators, the surface segmentation result may lead to incorrect polycube structures. The topological constraints mentioned in Section 4.1 are sufficient but not necessary conditions for polycube construction.

4.5 Conclusion and Future Work

In this thesis, a novel automatic CVT-based polycube construction algorithm is developed. Since we mainly use the normal information of the input surface, our method can naturally generate polycubes with low distortion. As a follow up from the classic CVT method, our extended HBECVT-based algorithm improves the segmentation by reducing non-monotone boundaries and is less sensitive to the initializations and noise. HBECVT usually generates polycube with fewer singularities and can also control the compactness of the resulting polycube with the weighting parameter λ. After the construction of polycube, we can generate uniform all-hex meshes and T-meshes via octree subdivision and parametric mapping. In addition, an improved mesh generation algorithm is presented based on a hybrid octree to construct adaptive all-hex dual meshes. Both uniform and adaptive all-hex meshes can be directly used for the FEA analysis in engineering practice. With the valid T-meshes, we can also construct analysis-suitable T-splines (weighted T-splines) for IGA applications. The robustness and efficiency of our presented algorithms are demonstrated by various types of input surfaces. In the future, we plan to apply our method to constructing solid T-splines for complex geometry.

Chapter 5

Surface Segmentation for Polycube Construction Based on Generalized Centroidal Voronoi Tessellation

In this thesis, we develop a new two-step surface segmentation scheme using generalized CVT methods for polycube construction, which can be used for all-hex mesh generation and is suitable for objects with slim cylindrical components. We first classify vertices of the surface into several components by extending the HBECVT algorithm to the SLO eigenfunction space. The key idea is that we update the generators in the p-dimensional eigenfunction space and physical information is introduced in the CVT-based clustering energy function, which tends to reduce jaggy boundary and over-segmentation. Based on the pre-segmentation results, each component is further segmented into several patches associated to the polycube faces. A novel skeleton-based CVT model is presented by using local coordinates to define the generators flexibly in the normal space which naturally embed the topological skeleton information into polycube construction. Predefined geometric constraints are used to revise the segmented results to obtain a valid polycube structure. After that, we generate quality all-hex meshes and volumetric T-meshes via parametric mapping. The key contributions of our two-step surface segmentation based polycube construction include:

1. A new eigenfunction-based CVT method is developed for surface segmentation by using the SLO eigenfunctions. Surface vertices are classified into several components based on concave creases and convex ridges of an object while generators are iteratively updated in the high dimensional eigenfunction space. Compared to other methods, it is efficient and eliminates unsmoothed boundaries, over-segmentations and noise effect; and

2. The skeleton-based CVT algorithm generalizes the HBECVT method by introducing local coordinates to define and update the generators flexibly in the normal space. Compared to other polycube construction methods, it reduces unnecessary singularities on the surface and is suitable for objects with slim cylindrical components. The generated all-hex meshes align to the detected concave creases and convex ridges from segmentation.

5.1 Two-Step Surface Segmentation

- **Eigenfunction-based CVT (Step 1).** Since the SLO is defined based on the second fundamental form of the surface, here we choose a higher order finite element solver to compute its eigenfunctions, that is, an isogeometric analysis solver using cubic Catmull-Clark basis functions [136]. We first apply the cross field-based surface parameterization [68] to convert the input triangular mesh T_1 to a semi-structured quadrilateral mesh Q_1, and then build the Catmull-Clark surface and compute the SLO eigenfunctions. After that, we convert Q_1 back to a triangular mesh T_2 by splitting each quadrilateral into two triangles. We then map vertices of T_2 onto a p-dimensional space by selecting the first p modes of the SLO eigenfunctions. To apply the eigenfunction-based CVT method, we first initialize L generators by selecting L vertices and taking their p-dimensional eigenfunction values. Generators are updated iteratively in the high dimensional eigenfunction space and vertices of T_2 are then segmented into L clusters. To reduce jaggy boundary and over-segmentation, neighbouring vertex information is also introduced into the clustering energy function. The boundaries among clusters are constructed to extract the segmentation result.

- **Skeleton-based CVT (Step 2).** In this step, we further segment each component from Step 1 into several patches, where each patch corresponds to one face of the polycube. For objects with slim cylindrical components, we extract the skeleton information K and calculate local coordinates. Surface normals are transformed to the corresponding nearest local coordinate system along the skeleton. We then define and iteratively update the generators in the HBECVT scheme using local coordinates in the normal space. For other general objects, the HBECVT scheme can also be directly applied to segment each component in the global normal space.

- **Geometric Constraints.** With a set of predefined connectivity and geometric constraints [45], we revise the segmentation results to obtain a valid polycube structure.

Based on the above two-step surface segmentation with geometric constraints, we construct valid polycubes by creating a surface parametric mapping from T_2 to a parameter domain T^*. We can then generate quality all-hex meshes and volumetric T-meshes in both parametric and physical domains.

5.1.1 Eigenfunction-based CVT

Different from the widely employed LBO eigenfunctions, the SLO eigenfunctions can capture the curvature-related surface features such as concave creases and convex ridges [67], which automatically distinguish different components of an object. Let $S = \{\mathbf{x}(u,v), (u,v) \in \mathbb{R}^2\}$ be a smooth parametric surface, where (u,v) can also be written as (u^1,u^2) for convenience. The coefficients of the first fundamental form of S can be defined as $g_{\alpha\beta} = \langle \mathbf{x}_{u^\alpha}, \mathbf{x}_{u^\beta} \rangle$

$(\alpha,\beta = 1,2)$, where $\mathbf{x}_{u^\alpha} = \frac{\partial \mathbf{x}}{\partial u^\alpha}$ and $\mathbf{x}_{u^\beta} = \frac{\partial \mathbf{x}}{\partial u^\beta}$. The coefficients of the second fundamental form of S can be defined as $b_{\alpha\beta} = \left\langle \mathbf{n}, \mathbf{x}_{\alpha\beta} \right\rangle$, where $\mathbf{x}_{u^\alpha u^\beta} = \frac{\partial^2 \mathbf{x}}{\partial u^\alpha \partial u^\beta}$ and $\mathbf{n} = (\mathbf{x}_u \times \mathbf{x}_v)/\left\|\mathbf{x}_u \times \mathbf{x}_v\right\|$. Let $[b^{\alpha\beta}] = [b_{\alpha\beta}]^{-1}$, and $\forall f \in C^2(S)$, the SLO $\blacktriangle$ [67] can be defined implicitly satisfying

$$\int_S (h \blacktriangle f + \langle \blacklozenge f, \blacklozenge h \rangle) \mathrm{dA} = 0, \ \forall h \in C^1(S), \tag{5.1}$$

where $\blacklozenge$ is the generalized second tangential operator (GSTO) defined as

$$\blacklozenge f = [\mathbf{x}_u, \mathbf{x}_v] \Phi \left[b^{\alpha\beta} \right] [f_u, f_v]^{\mathrm{T}}. \tag{5.2}$$

Different choices of the parameter Φ can be used for detecting different types of surface features such as concave creases/regions and convex ridges [67]. Let λ_s be the eigenvalues of the SLO, we have $\blacktriangle f = -\lambda_s f$, which can be solved using the Catmull-Clark subdivision based isogeometric analysis [136]. Note that a remeshing process is required for the input triangle mesh in order to use the Catmull-Clark basis functions and quadrilateral control meshes [67]. Since main components of 3D objects are usually connected with concave creases and convex ridges, correct detection of these features yields reasonable surface segmentation results. Several examples were presented in [67] to show the advantages of the SLO eigenfunctions using PAM for surface segmentation. Compared with traditional surface segmentation methods based on LBO eigenfunctions, the SLO-based method can segment the surface at the convex creases and concave ridges, which better reflects main components of an object. However, the PAM-based surface segmentation is time consuming in identifying the optimal number of segmentation sets. It may also result in jaggy boundary and over-segmented results since no physical information is included during the classification.

In this thesis, we apply the efficient CVT-based clustering idea to the SLO eigenfunctions by incorporating physical information to avoid over-segmentation and jaggy boundaries, and reduce the computational cost. The CVT-based model has been introduced to various fields and applications, including both image and mesh processing [27, 131, 26]. It is essentially a clustering technique by partitioning a set of discrete data points into non-overlapping clusters. Given an input dataset and L generators, all the data points can be partitioned into L clusters by assigning each point to its nearest generator. Then an energy function E which measures how tightly each cluster is packed can be defined and minimized with respect to the generators. In this way, all the generators are iteratively updated to be the centroids of their associated clusters, where a new partition can be computed with new generators. Such an algorithm aims at minimizing the energy function E until a certain criterion is met.

We first compute eigenfunctions of the SLO and map surface vertices onto a p-dimensional space by selecting the first p modes. Let $\Psi = \{\psi_i\}_{i=1}^{nv}$ denote the p-dimensional eigenfunctions for all vertices of the surface mesh T_2 with $\psi_i = \left(\psi_{i1}, \ldots, \psi_{ip}\right)^T$, where nv is the total number of vertices and ψ_i represents the assigned p-dimensional vector of the i^{th} vertex $v(i)$. The initial Voronoi generators $C = \{c_l\}_{l=1}^{L}$ are defined as a set of typical p-dimensional eigenfunction vectors. For the generator c_k ($k = 1, \ldots, L$), its corresponding Voronoi region V_k in Ψ can be defined as

$$V_k = \{\psi_i \in \Psi : dist(\psi_i, c_k) \leq dist(\psi_i, c_l), for\ l = 1, \ldots, L\},\ k = 1, \ldots, L, \tag{5.3}$$

where $dist(\psi_i, c_k)$ is a certain metric to calculate the distance between the p-dimensional vector ψ_i and the generator c_k. Note that $V = \{V_l\}_{l=1}^{L}$ denote the Voronoi tessellation of the data set Ψ according to generators $C = \{c_l\}_{l=1}^{L}$. Since each vertex is assigned to one of L clusters, we have $\Psi = \cup_{l=1}^{L} V$ and $V_i \cap V_j = \emptyset$ if $i \neq j$. Given any set of generators $C = \{c_l\}_{l=1}^{L}$ and any partition $U = \{U_l\}_{l=1}^{L}$ of Ψ, we can define the classical clustering energy of $(C; U)$ as

$$E(C; U) = \sum_{l=1}^{L} \sum_{\psi_i \in U_l} dist^2(\psi_i, c_l), \tag{5.4}$$

where $dist(\psi_i, c_k) = ||\psi_i - c_k||$ measures the Euclidean distance between vector ψ_i and generator c_k in the p-dimensional eigenfunction space. Since the classic CVT clustering energy does not take into account any physical information, the resulting segmentation may have jaggy boundaries and over-segmentation results, especially for noisy surfaces. To overcome limitations of the classic CVT method, we extend the idea of HBECVT to the eigenfunction-based CVT for surface segmentation. By including the local neighbouring information of vertex $v(i)$, we can similarly define a boundary-enhanced distance metric as

$$dist(\psi_i, c_k) = \sqrt{\|\psi_i - c_k\|^2 + \eta \tilde{n}_k(v(i))}. \tag{5.5}$$

In this distance metric, the first term $\|\psi_i - c_k\|^2$ measures the Euclidean distance in the p-dimensional eigenfunction space. The second term $\tilde{n}_k(v(i))$ introduces the neighbouring physical information which measures the number of vertices that do not belong to the k^{th} cluster within $N_\omega(v(i))$, which is a ω-rings neighboring region centered at vertex $v(i)$. η is a positive weighting factor to balance these two terms. The boundary-enhanced term $\tilde{n}_k(v(i))$ extends the idea of HBECVT [45] from triangle element in the normal space to vertex in the eigenfunction space, and represents the probability that vertex $v(i)$ belongs to the k^{th} cluster in the physical space. Note that the boundary-enhanced term smooths the boundary and reduces over-segmentation by balancing the size of clusters in the neighbouring region of each vertex. $\tilde{n}_k(v(i))$ is small if the majority of neighbourhoods within $N_\omega(v(i))$ belong to the k^{th} cluster, and vice versa. Inspired by the HBECVT clustering energy [45] defined in the normal space, we extend it to the p-dimensional eigenfunction space to define the eigenfunction-based CVT clustering energy function as

$$E^E(C;U) = \sum_{i=1}^{nv} \frac{L}{\sum\limits_{l=1}^{L} dist^{-2}(\psi_i, c_l)}. \tag{5.6}$$

Note that $U = \{U_l\}_{l=1}^L$ are not necessarily the Voronoi regions corresponding to the set of generators $C = \{c_l\}_{l=1}^L$. The above clustering energy function measures the similarity of the data by using squared boundary-enhanced distance from all the generators to calculate the harmonic average for each vertex. The updated centroid of each cluster U_k is defined as the

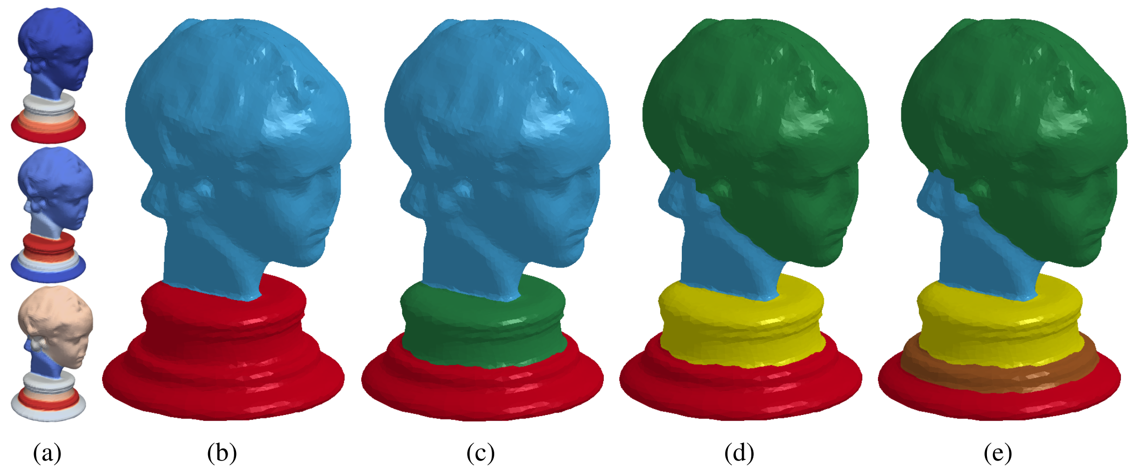


Figure 5.1: Segmentation results of the Bust model with different numbers of clusters. (a) Modes 1-3 of the SLO; (b) 2 clusters; (c) 3 clusters; (d) 4 clusters; and (e) 5 clusters.

p-dimensional vector c_k^* which minimizes the above clustering energy with respect to c_k. An iterative formula can be obtained as

$$c_k^* = \frac{\sum_{i=1}^{nv} \mu_{ik} w_i \psi_i}{\sum_{i=1}^{nv} \mu_{ik} w_i}, \text{ where } \mu_{ik} = \frac{dist^{-4}(\psi_i, c_k)}{\sum_{l=1}^{L} dist^{-4}(\psi_i, c_l)} \text{ and } w_i = \frac{\sum_{l=1}^{L} dist^{-4}(\psi_i, c_l)}{\left(\sum_{l=1}^{L} dist^{-2}(\psi_i, c_l)\right)^2}. \quad (5.7)$$

μ_{ik} can be viewed as a soft membership function which reflects the degree of possibility of $v(i)$ that is associated with the k^{th} cluster and w_i is a weighting function which defines the influence of vertex $v(i)$ on computing new centroids of clusters. After initializing L generators $\{c_l\}_{l=1}^{L}$ by randomly selecting L vertices and taking their p-dimensional eigenfunction values, for each iteration we assign each vertex to the cluster whose generator has the shortest distance to it by Eq. (5.5). The cluster centroids $\{c_l^*\}_{l=1}^{L}$ are updated iteratively by Eq. (5.7) until the energy variation is less than a predefined threshold.

For the eigenfunction-based CVT, there are a total of three parameters that need to be specified: L as the number of clusters, η as the weighting factor used to balance the Euclidean distance in the eigenfunction space and the boundary-enhanced distance in neighbouring physical space, and ω as the size of neighbourhood used to control the neighbouring physical information. The resulting segmentation results are effected by all these three parameters, i.e., L determines the number of resulting clusters, η and ω control the segmentation accuracy and robustness. Fig. 5.1 shows Modes 1-3 of the SLO eigenfunctions and the segmentation results of the Bust model with two, three, four and five clusters, respectively. From the results we can observe that more features can be detected with more clusters. The results demonstrate that our proposed eigenfunction-based CVT algorithm is flexible and robust with respect to different numbers of clusters. To illustrate the effect of η, we apply the eigenfunction-based CVT to a noisy Bust model with different values of η ($\omega = 2$). Fig. 5.2(a) shows Modes 1-3 of the SLO eigenfunctions, which are used to segment the noisy Bust model into 5 clusters. Without the boundary-enhanced term ($\eta = 0$), there are jaggy boundaries and over-segmentation caused by the noise, shown as Fig. 5.2(b). If η is small ($\eta = 0.05$), more detailed features detected by SLO eigenfunctions can be kept. However, the noise effect can not be completely eliminated, see the over-segmentation in the red window of Fig. 5.2(c). When η becomes larger ($\eta = 0.1$), the eigenfunction-based CVT will tend to shorten and smooth boundaries of the resulting segmentation, thus the noise effect, jaggy boundaries and over-segmentation can be gradually removed, shown in Fig. 5.2(d). Note that when η is very large ($\eta = 0.2$), boundaries are over-smoothed and features are gradually erased, see the result in the yellow window of Fig. 5.2(e). Thus we need to select the proper parameters L and η in order to generate reasonable segmentation results for given surfaces. The effect of ω is essentially similar to that of η. A larger ω can eliminate more noises and obtain more smoothed boundaries, while a smaller ω results in higher accuracy in capturing detailed features of the surface. Note that a large ω requires more computation, therefore we set $\omega = 2$ in this thesis for all the examples in order to reduce the computational cost.

Fig. 5.3 shows a comparison of the PAM method and our eigenfunction-based CVT scheme, with the SLO eigenfunctions used in both of them. We segment the Bust into 6 clusters, where neighbouring components are rendered with different colors. Since the SLO eigenfunctions can detect all concave creases, both the PAM and eigenfunction-based

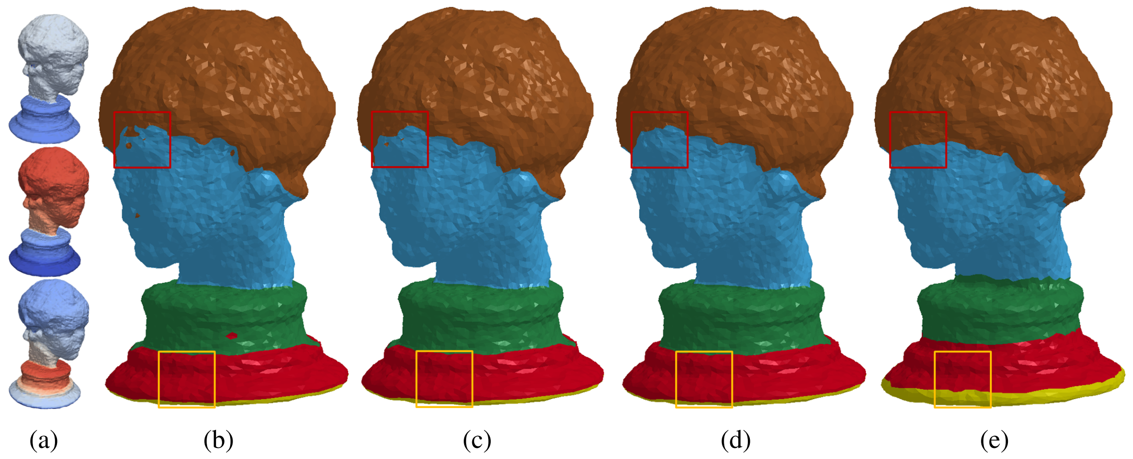


Figure 5.2: Segmentation results of the noisy Bust model with different values of η ($\omega = 2$). (a) Modes 1-3 of the SLO; (b) $\eta = 0$; (c) $\eta = 0.05$; (d) $\eta = 0.1$; and (e) $\eta = 0.2$.

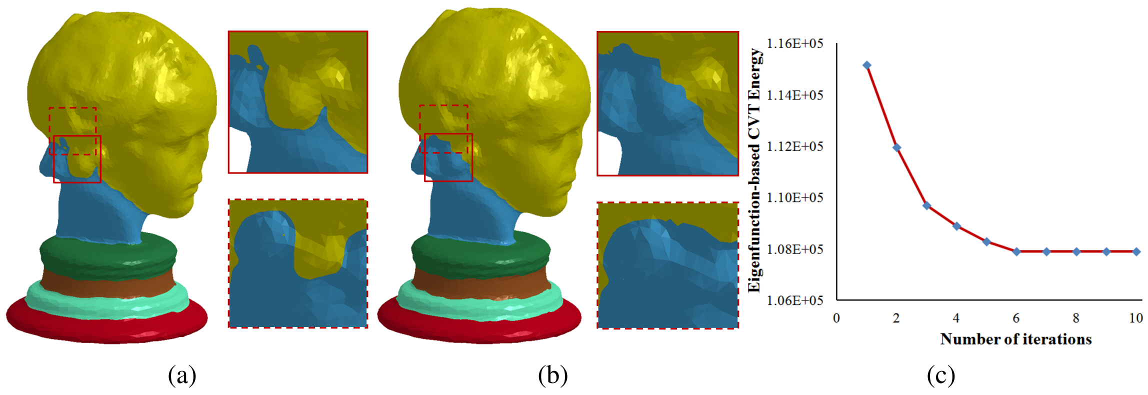


Figure 5.3: The Bust model. (a) Segmentation result using the PAM method; (b) segmentation result using our eigenfunction-based CVT method; and (c) the eigenfunction-based CVT energy curve.

CVT methods yield similar segmentation results, which outperform other methods (LBO, concavity-aware Laplacian and shape diameter function method) through a statistical study in [67]. The curve in Fig. 5.3(c) shows that the eigenfunction-based CVT energy keeps decreasing and converges after 6 iterations. Compared with the PAM method, our eigenfunction-based CVT clustering scheme eliminates jaggy boundaries and over-segmentation (see the comparison in the red windows), and the computational time is reduced from 287.6 to 2.2 seconds.

Remark. Since the SLO is defined based on the second fundamental form of the surface, its eigenfunctions can capture surface features sensitive to curvature changes, such as concave creases and convex ridges. Our proposed eigenfunction-based CVT segmentation algorithm maintains properties of SLO eigenfunctions by using the efficient CVT-based clustering idea in a p-dimensional eigenfunction space. By incorporating neighbouring physical information into the eigenfunction-based CVT clustering energy function, we eliminate jaggy boundaries and avoid over-segmentation with an efficient computation. For a noisy input surface, the boundary-enhanced term increases the possibility that each vertex will be assigned to the same cluster with the majority of its neighbourhoods, which improves the robustness to the noise during the surface segmentation. Furthermore, the harmonic average form of the energy function restricts that all vertices partially influence the calculation of each centroid, which makes it less sensitive to initializations of the generators.

5.1.2 Skeleton-based CVT

The HBECVT-based algorithm [45] improves the segmentation and polycube construction by introducing the local neighbouring information into the energy function, which tends to reduce non-monotone boundaries and generate compact polycubes with low distortion. Based on the pre-segmentation using the eigenfunction-based CVT as shown in Fig. 5.3(b), we apply the HBECVT to further segment each component into several patches using the global normal space, where each patch corresponds to one face of the polycube in certain principal orientation. Singularities, corner vertices that do not have exactly 4 neighbouring patches, are marked in red nodes as shown in Fig. 5.4(a). However, this method results in large number of unnecessary singularities for slim cylindrical surfaces, such as the Cylindrical-helix model shown in Fig. 5.5(a). There are 28 singularities generated with the HBECVT-based segmentation in the global normal space, leading to a complex polycube structure. To improve the polycube construction, in this section we apply the skeleton information to compute local coordinates and use them for polycube construction.

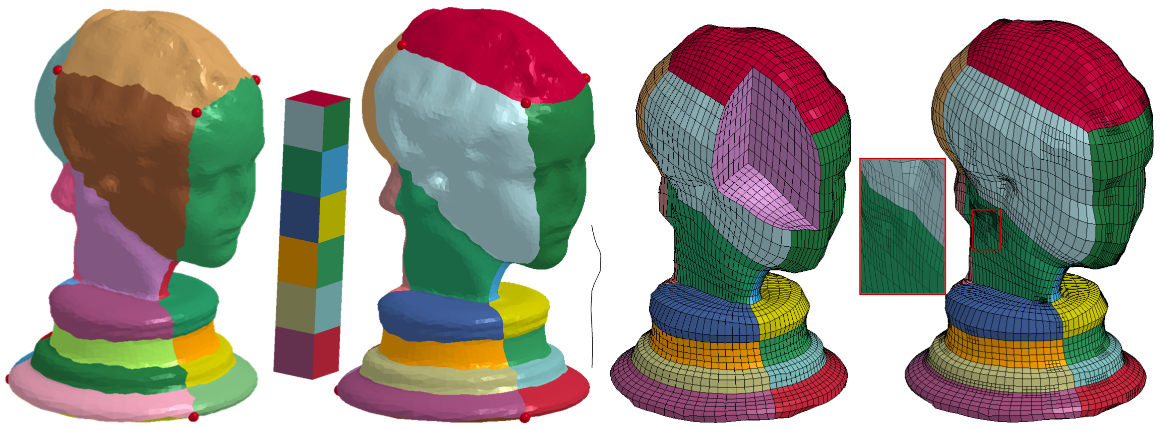


Figure 5.4: Results of the Bust model. (a) Segmentation result of the Bust model using HBECVT in the global normal space; (b) segmentation result using the skeleton-based CVT method with the skeleton and polycube; (c) all-hex mesh with some elements removed to show the interior; and (d) volumetric T-mesh.

Skeletons are simplified 1D description, which can capture the main geometric features and topology of the input 3D surface. Several algorithms have been developed in the literature for the skeleton extraction. The mean curvature flow [120] extracts the skeleton by iteratively moving each surface point along the inverse of its normal direction with a speed proportional to its curvature. The mesh contraction method computes the skeleton of an input object by progressively shrinking it along a constrained mean curvature flow [5]. These algorithms produce skeletons that are enclosed by the surface as medial axis. Given an input triangle mesh T, here we adopt the mean curvature flow algorithm [120] to generate its skeleton $K = \left\{K_j\right\}_{j=1}^{m}$, where m is the total number of skeleton points. Let $X = \left\{x_{T(i)}\right\}_{i=1}^{n}$ denote all the unit normals $x_{T(i)}$ of the triangle mesh according to the global coordinate system, where n is the total number of triangles and $T(i)$ represents the i^{th} triangle in the physical space.

We first compute the local coordinate system of each point s_j and also the transformation matrix Q from the global to the local coordinate system. The local coordinate system of each curve-skeleton point can be defined using the tangent and its perpendicular plane. Fig. 5.5(b) shows the skeleton of the Cylindrical-helix model, the local coordinate system of each

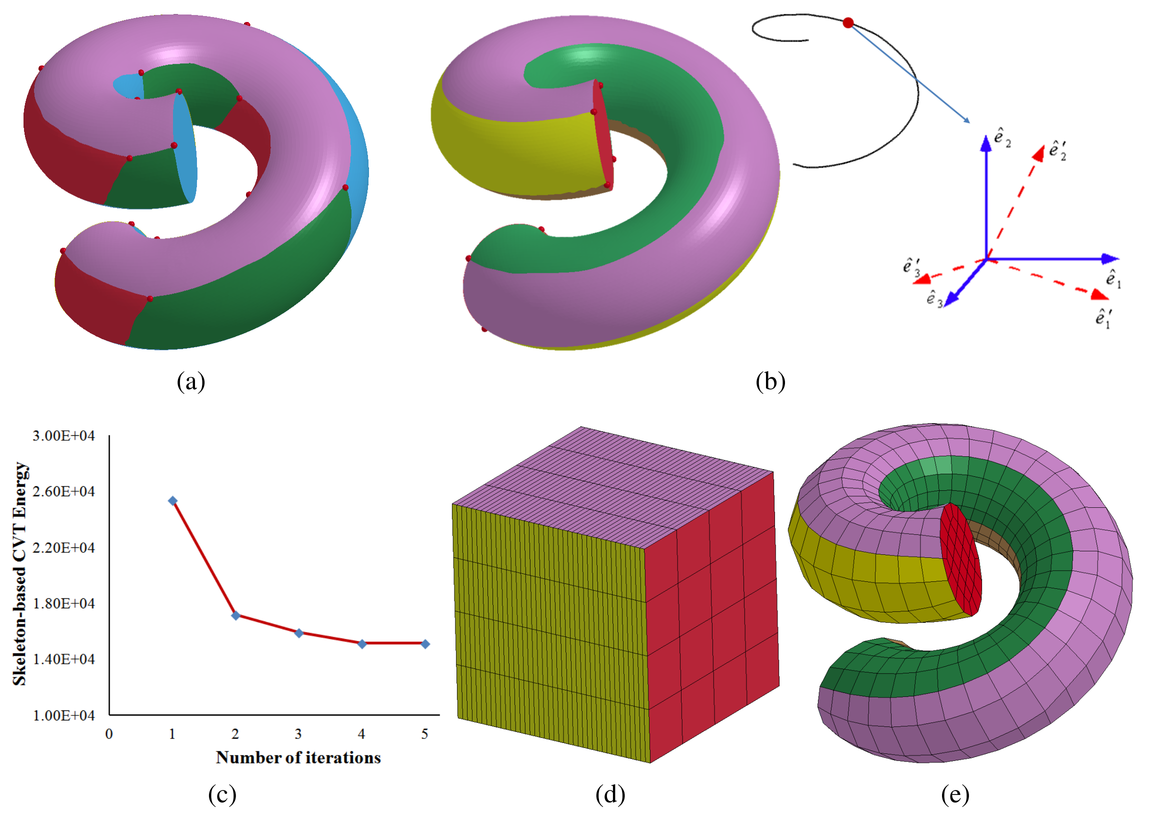


Figure 5.5: Results of the Cylindrical-helix model. (a) Segmentation result using HBECVT in the global normal space; (b) skeleton-based CVT segmentation result using the local normal space with the skeleton information; (c) skeleton-based CVT energy curve; (d) parametric domain; and (e) all-hex mesh.

skeleton point can be defined using the tangent and the plane perpendicular to the tangent. Any unit vector $\hat{u}$ in the global coordinate system $\{\hat{e}_1, \hat{e}_2, \hat{e}_3\}$ can be transferred to the local coordinate system $\left\{\hat{e}_1', \hat{e}_2', \hat{e}_3'\right\}$ by $\hat{u}' = Q\hat{u}$, where Q is a 3×3 rotation matrix and its entry $Q_{ij} = \cos\left(\hat{e}_i, \hat{e}_j'\right) = \hat{e}_i \cdot \hat{e}_j'$. Since each triangle can be assigned to its nearest skeleton point, we can transform its normal vector in the global coordinate system to the local coordinate system by $x'_{T(i)} = Q_{T(i)} x_{T(i)}$, where $Q_{T(i)}$ is the transformation matrix between the local coordinate system $T(i)$ assigned to and the global coordinate system. Thus, we can calculate a new set of unit normals $X' = \left\{x'_{T(i)}\right\}_{i=1}^{n}$ from $X = \left\{x_{T(i)}\right\}_{i=1}^{n}$. Note that the normal vector for each triangle $T(i)$ is transformed via a rotation matrix with the help of the skeleton, which makes it suitable for slim cylindrical objects.

Let $C = \{c_l\}_{l=1}^{L}$ (L = 6) denote a set of typical unit normal vectors ($\{(\pm 1, 0, 0), (0, \pm 1, 0), (0, 0, \pm 1)\}$) and $U = \{U_l\}_{l=1}^{L}$ denote any partition of X'. Inspired by the HBECVT model [45], we can define a new skeleton-based CVT clustering energy as

$$E^S(C; U) = \sum_{i=1}^{n} \left(L \Bigg/ \sum_{l=1}^{L} \left(\left\| x'_{T(i)} - c_l \right\|^2 + \lambda \tilde{n}_l(T(i)) \right)^{-1} \right), \tag{5.8}$$

where $\tilde{n}_k(T(i))$ is the boundary-enhanced term and λ is a positive weighting factor. Similar to HBECVT, the construction of skeleton-based CVT is also an energy minimization process, where the centroids are updated iteratively by minimizing E^S with respect to the generator c_k ($k = 1, \ldots, L$). Fig. 5.5(b) shows the segmentation results of the Cylindrical-helix model using the skeleton-based CVT. Compared to Fig. 5.5(a), we can observe that our skeleton-based CVT algorithm aligns to the shape better by segmenting the surface in the transformed normal space, and it yields only 8 singularities in total. The curve in Fig. 5.5(c) shows that the skeleton-based CVT energy keeps decreasing and converges after 4 iterations.

Remark. Our skeleton-based CVT segmentation algorithm is suitable for slim cylindrical objects and can reduce unnecessary singularities with compact polycube structure. Since local coordinate systems follow the skeleton direction, the number of singularities (polycube corners) can be reduced, resulting in a more compact polycube structure. Moreover, the skeleton-based CVT algorithm inherits the properties of the HBECVT model. The objective energy function uses the squared distances to all generators for each triangle in a transformed normal space, where all generators partially influence the harmonic average for each triangle. In addition, the skeleton-based CVT is also less sensitive to the initialization of generators and noise.

5.2 Geometric Constraints

Polycubes are orthogonal polyhedra bounded by axis-aligned planes. A segmentation produces an invalid polycube if any patch has less than four neighbours, or if patches associated with opposite orientations of the same axis share a boundary [74]. Thus we need to set geometric constraints [45] to the segmentation results in order to generate a valid polycube structure. Here are some definitions. Based on the eigenfunction-based CVT segmentation, a *component boundary* is the boundary between two neighbouring components. In a segmented patch of a specific component, the corner vertex lying along the component boundary is called a *boundary corner*. The corner vertex lying inside a component is called an *interior corner*. For a certain partition of a component, the sufficient conditions of a valid polycube structure are as follows: 1) two patches with opposite orientations of the same axis can not share a boundary; 2) each boundary corner vertex is shared by two patches and each interior corner vertex is shared by three patches; and 3) each patch must have more than three boundaries.

We first merge each patch with less than four boundaries to one of its neighbouring patches that has the longest boundary, and also prevent two triangles sharing an edge or a vertex from being assigned to two opposite patches. Since we segment each component separately, we need to ensure that two neighbouring components have the same number of corners placed along the component boundary. To achieve this, we place all corners close to the boundary of different components onto the component boundary. In Fig. 5.6(a), the interior red corner is near the boundary of the Bust's yellow component shown in Fig. 5.3(b). We thus merge the green and pink patches to place the corner onto the component boundary, shown in Fig. 5.6(b). For two neighbouring components, corner vertices along the component boundary may not be consistent. We thus merge nearby corners along the boundary in order to generate conformal polycube structure. In Fig. 5.6(c), we can observe that there are two nearby red corners between Bust's yellow and blue components shown in Fig. 5.3(b). These two corners need to be merged so that the neighbouring components can be well matched, see Fig. 5.6(d). Fig. 5.4(b) shows the skeleton-based CVT segmentation result of the Bust model. Since the skeleton of the Bust model is close to a straight line, the segmentation result is similar to the HBECVT-based segmentation shown in Fig. 5.4(a).

There are eight corner points for each component corresponding to eight corners of the cube in parametric space. All corner points between two neighbouring components are constrained to be on the component boundaries, which ensures that the resulting polycube is conformal.

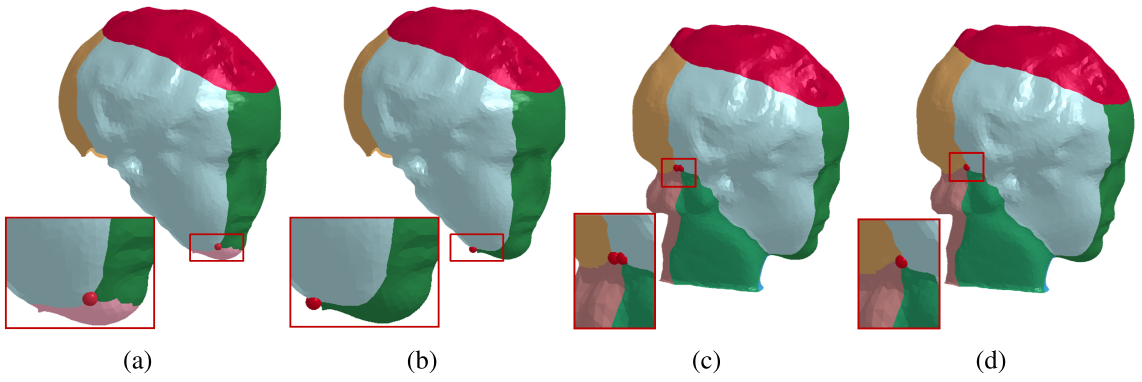


Figure 5.6: Geometric constraints. (a, b) The green patch and the pink patch are merged to move the nearby interior corner (the red dot) to the boundary of the yellow component in Fig. 5.3(b); and (c, d) two nearby boundary corners are merged.

5.3 Polycube Construction and Mesh Generation

After the two-step surface segmentation as discussed in Section 5.1, each patch is now aligned with one of the six global principal axes, leading to a valid polycube structure. The next step is to create a one-to-one mapping f from the surface T_2 to a parameter domain T^* through surface parameterization. For each patch, we use a planar domain parameterization [32] to create a parametric mapping between the triangle mesh T_2 and a planar polygon in the parametric space. Each boundary edge is mapped onto its corresponding polycube edge via a chord length parameterization. The embedding algorithm [142] is then applied to position the interior vertices of each patch. This planar domain parameterization computes a bijective map from the input mesh to the polycube surface and enforces vertices in each patch to have the same coordinate value along the corresponding axis. Fig. 5.5(d) shows the parametric mapping result of the Cylindrical-helix model, and we can observe that

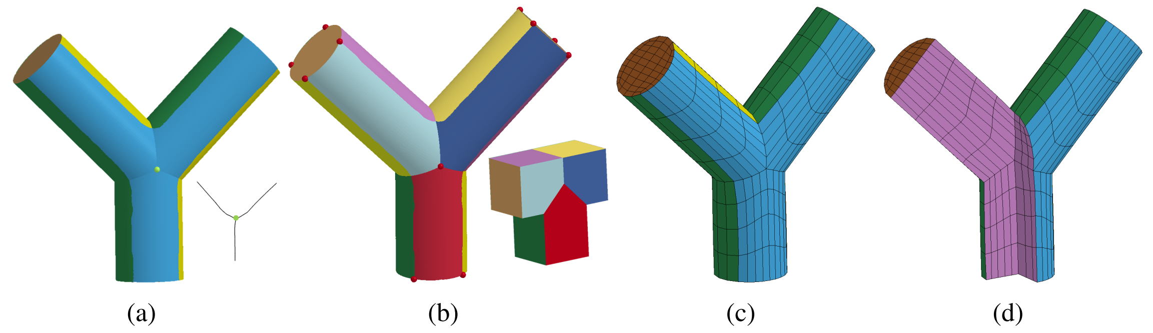


Figure 5.7: Results of the Bifurcation-tube model. (a) Skeleton-based CVT segmentation results with the skeleton; (b) splitting results based on the segmentation with the polycube; (c) all-hex mesh; and (d) some elements are removed to show the interior of (c).

each triangle of the input surface is mapped to its corresponding position in the parametric domain and the mapping is bijective.

As mentioned in [151], there is a bifurcation when three branches of the skeleton join together, which is very common in blood vessels. To handle the bifurcation case, we apply the decomposition template idea [151] during the parametric mapping process, which encloses singularities into the interior domain. Fig. 5.7 shows the segmentation result and parametric mapping process for the Bifurcation-tube model. Fig. 5.7(a) shows the skeleton-based CVT segmentation result with its skeleton information. Then two saddle points (marked as green dots, another one is on the back) can be obtained by projecting the skeleton intersection point to the surface. We then decompose the skeleton-based CVT segmentation result into 15 patches by tracing from the saddle points to the neighbouring boundaries, shown in Fig. 5.7(b), where each patch is rendered in a different color. Based on the decomposition result, we can use the skeleton information to assign each component to the corresponding planar parametric domain and calculate a bijective mapping result. In this way, singularities are introduced to the polycube.

Based on the constructed polycube, we generate uniform all-hex meshes in both parametric and physical domains. Uniform all-hex meshes $\mathcal{H}$ can be generated by firstly meshing the resulting polycube, and then mapping it back onto the input surface via parametric mapping for boundary vertices and interpolation for interior vertices. The mesh quality needs to be improved since the extracted all-hex meshes may have poorly-shaped elements around the boundary [150]. We first use the pillowing technique to generate one layer along the boundary to ensure that each hex element has at most one face on the boundary. Here we choose the scaled Jacobian to measure the mesh quality. The optimization-based smoothing, where the objective function is equivalent to the maximization of the minimum scaled Jacobian, is applied to further improve the mesh quality. Fig. 5.4(c) shows the all-hex mesh of the Bust model with some elements removed to show the interior. We can observe that the hex mesh aligns to concave/convex features detected through the surface segmentation. Fig. 5.5(e) shows the all-hex mesh of the Cylindrical-helix model after the quality improvement (minimal Jacobian ≥ 0.30). As shown in Fig. 5.7(c, d), we can also generate hex mesh for the bifurcation template in the parametric domain. Starting from an uniform mesh, we can generate volumetric T-mesh by subdividing one element into eight children elements. We check the projection distance from the hex mesh boundary to the input triangular mesh, and subdivide the element if the distance is greater than a given threshold. Fig. 5.4(d) shows the volumetric T-mesh of the Bust model.

5.4 Results and Discussion

We have applied the presented algorithms to several datasets, including the Bust (Fig. 5.4), Aneurysm (Fig. 5.8), Human (Fig. 5.9) and Ant (Fig. 5.10). For each model we generated both surface segmentation results and valid polycubes with all-hex meshes and volumetric T-meshes. All results were computed on a PC equipped with a 2.93 GHz Intel X3470 CPU and 8GB of Memory. Statistics of all tested models are given in Table 5.1. All the hex meshes are in reasonably good quality after applying quality improvement techniques.

Most branching blood vessels in the human vascular system are bifurcations, where we need to decompose the domain into mapped regions for polycube construction. Fig. 5.8 shows the result of the Aneurysm model. We first apply the eigenfunction-based CVT

Table 5.1: Statistics of all the tested models.

Model	Modes	η	λ	Number of singularities	Output mesh (vertices, elements)	Jacobian (worst, best)	Time (s)
Bust	1-3	0.1	0.01	8	(27,735, 25,344)	(0.10, 1.00)	53.4
Aneurysm	1-3	0.001	0.001	14	(2,881, 2,272)	(0.33, 1.00)	14.4
Human	1-5	0.01	0.01	30	(27,435, 23,424)	(0.11, 1.00)	49.6
Ant	1-6	0.05	0.02	64	(15,999, 12,928)	(0.12, 1.00)	42.4
Fertility ([145])				146	(38,034, 31,333)	(0.06, 1.00)	
Fertility ([74])				94	(61,147, 53,702)	(0.25, 1.00)	
Fertility (ours)	1-6	0.06	0.03	32	(27,498, 23,688)	(0.26, 1.00)	72.3
3-Torus ([145])				40	(31,339, 26,600)	(0.07, 1.00)	
3-Torus (ours)	1-3	0.001	0.001	8	(28,024, 24,600)	(0.26, 1.00)	59.6
Rockerarm ([74])				62	(63,363, 56,667)	(0.37, 1.00)	
Rockerarm (ours)	1-4	0.02	0.01	42	(48,022, 42,440)	(0.35, 1.00)	103.3

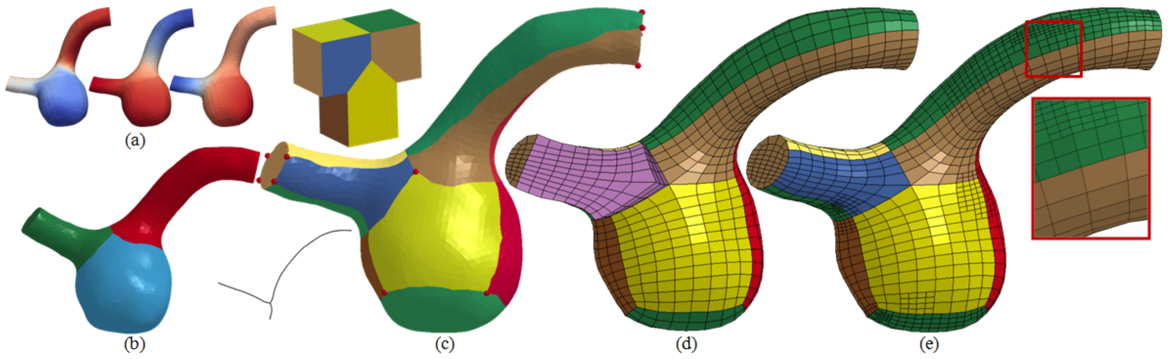


Figure 5.8: Results of the Aneurysm model. (a) Modes 1-3 of the SLO; (b) eigenfunction-based CVT segmentation result; (c) segmentation result using skeleton-based CVT method with the skeleton and polycube; (d) all-hex mesh with some elements removed to show the interior; and (e) volumetric T-mesh.

method to segment the surface into 3 components from Modes 1-3 of the SLO, see Fig. 5.8(b). Based on the pre-segmentation, we partition each component into multiple patches using the skeleton-based CVT, shown in Fig. 5.8(c). Singularities on the surface are marked in red dots. We can observe that each component is segmented into 5 patches, which automatically satisfies the bifurcation template [151]. Each patch can be mapped onto one polycube face. By enclosing some singularities into the interior domain using the bifurcation template, we can reduce the number of singularities on the surface.

Figs. 5.9 and 5.10 show results of the Human model and the Ant model. Both PAM and eigenfunction-based CVT methods are used to segment the Human model into 7 clusters and the Ant model into 11 clusters from Modes 1-5 and Modes 1-6 of the SLO respectively.

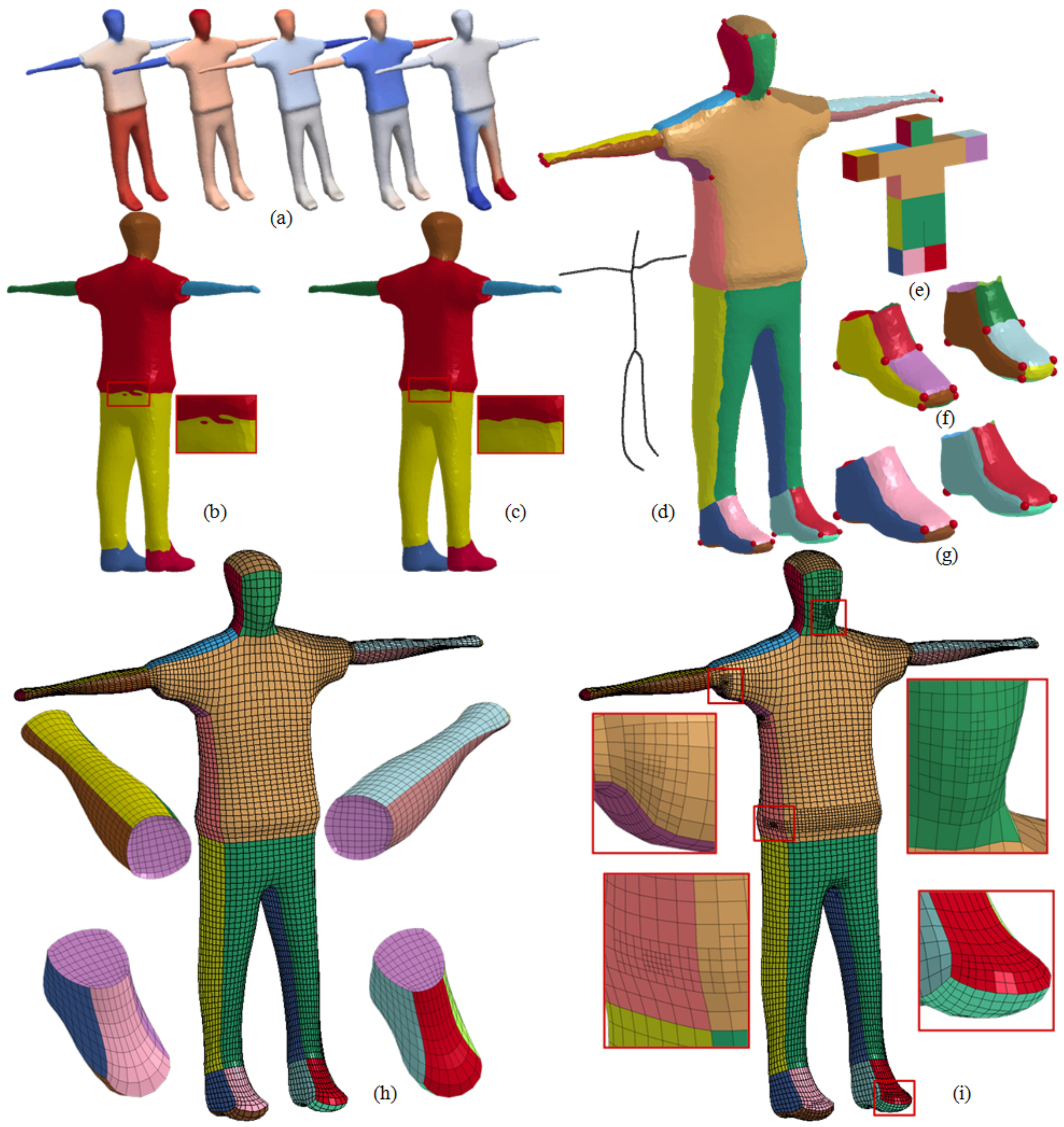


Figure 5.9: Results of the Human model. (a) Modes 1-5 of the SLO; (b) segmentation result using the PAM-based method; (c) segmentation result using eigenfunction-based CVT; (d) segmentation result using skeleton-based CVT with the skeleton; (e) polycube topology; (f) HBECVT-based segmentation of two feet using the global normal space; (g) Zoom-in of skeleton-based CVT segmentation of two feet using the local normal space; (h) all-hex mesh with some elements removed to show the interior; and (i) volumetric T-mesh.

From the results we can observe that the SLO eigenfunctions can detect concave creases and convex ridges, which well reflect main components of the 3D objects. Compared to the PAM-based segmentation, our eigenfunction-based CVT clustering method eliminates

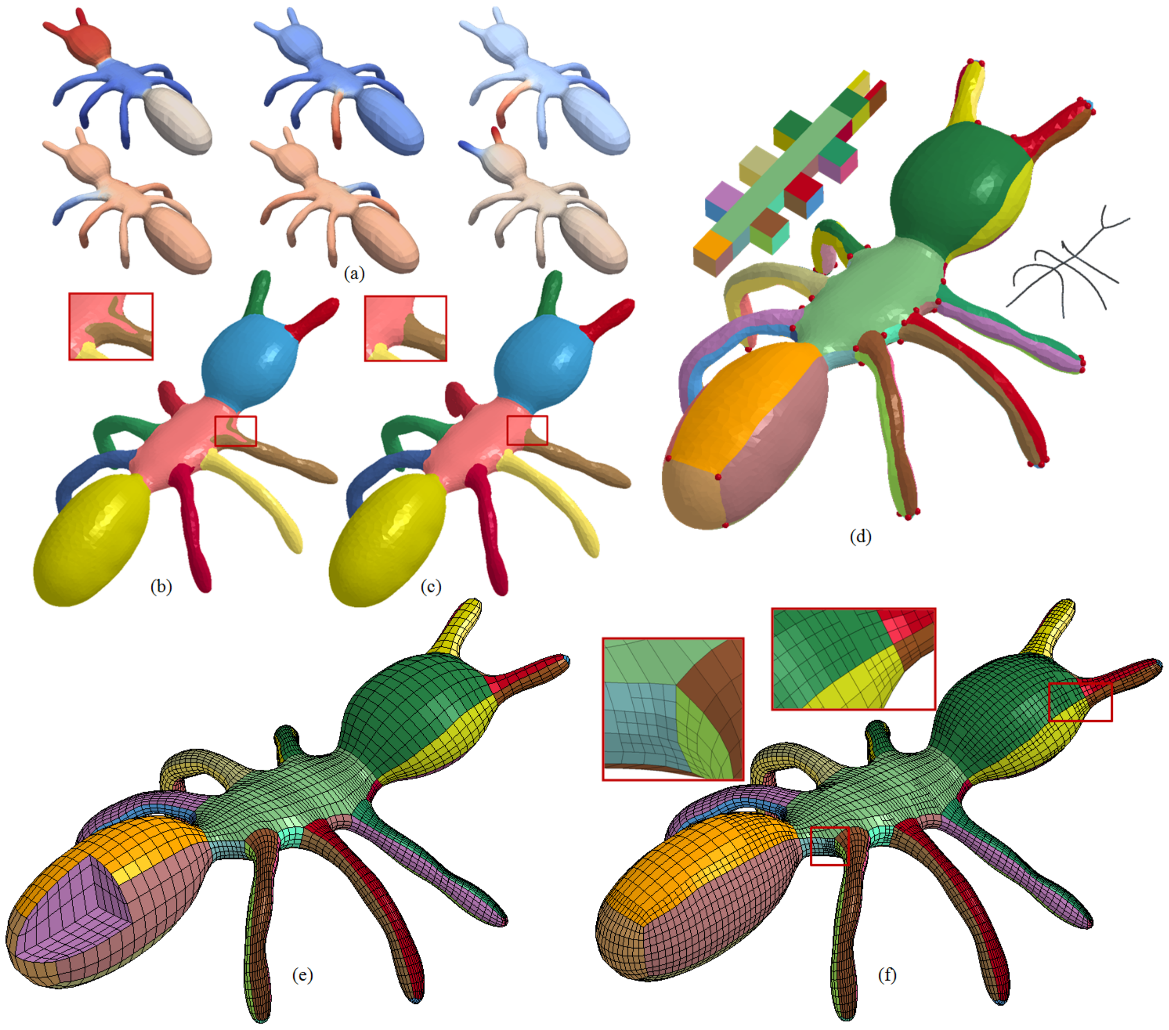


Figure 5.10: Results of the Ant model. (a) Modes 1-6 of the SLO; (b) segmentation result using the PAM-based method; (c) segmentation result using eigenfunction-based CVT; (d) segmentation result using skeleton-based CVT with the skeleton and polycube; (e) all-hex mesh with some elements removed to show the interior; and (f) volumetric T-mesh.

jaggy boundaries and over-segmentation by including neighbouring physical information into the energy function, see the red windows in Figs. 5.9(b, c) and 5.10(b, c). In addition, the computational time is reduced from 269.4 to 3.5 seconds for the Human model and from 187.6 to 2.9 seconds for the Ant model. With the correct detection of concave/convex features and smooth component boundaries, the mapping distortion of the resulting polycube can be reduced. Each tubular component can be further segmented in the local normal space using the skeleton-based CVT method, which generates compact polycubes with limited number of singularities, shown as Figs. 5.9(d) and 5.10(d). Compared to the HBECVT-based segmentation using the global normal space in Fig. 5.9(e), our skeleton-based CVT method segments the two feet with fewer number of singularities by using the local coordinates, shown as Fig. 5.9(f). Although six legs of the Ant have various shapes, we can use one cube with eight singularity points to represent each of them. Based on the constructed polycubes, we first generate all-hex meshes in the polycubes and then map them back to the physical domain through parametric mapping. All generated hex meshes have quality with the minimum scaled Jacobian ≥ 0.10 and also align to the concave/convex features. We also generate volumetric T-meshes by subdividing the uniform mesh in order to capture more surface features.

We compared the all-hex meshes generated using our method and other existing algorithms [145, 39, 74], see Fig. 5.11. For our results in (c, e, g), we show the eigenmodes used for the eigenfunction-based CVT, skeleton information and all-hex meshes. We measure the number of surface singularities and the scaled Jacobian in Table 5.1. Due to the usage of local coordinates which follow the skeleton direction, our method reduces the number of singularities significantly in models with slim cylindrical structures, such as the Fertility and 3-Torus models. For the Fertility model, the number of surface singularities is reduced from 146 [145], 96 [39] and 94 [74] to 32 by using our method, while the scaled Jacobian is comparable. Compared to [145], we reduce the number of singularities from 40 to 8 for the 3-Torus model (Fig. 5.11(e)), while improving the minimal Jacobian from 0.07 to 0.26. Compared to [39] and [74], we also reduce the number of singularities of the Rockerarm (from 70 and 62 to 42) around the cylindrical hole which follows the circular skeleton, see the red windows in (f, g). Regions with concave creases and convex ridges are well segmented using the SLO eigenfunctions and also preserved in our all-hex meshes, see the green windows in (c, g). To compare our method with [125], we extract quad layouts by editing the generated all-hex meshes. Fig. 5.12 shows quad layouts of the Fertility and

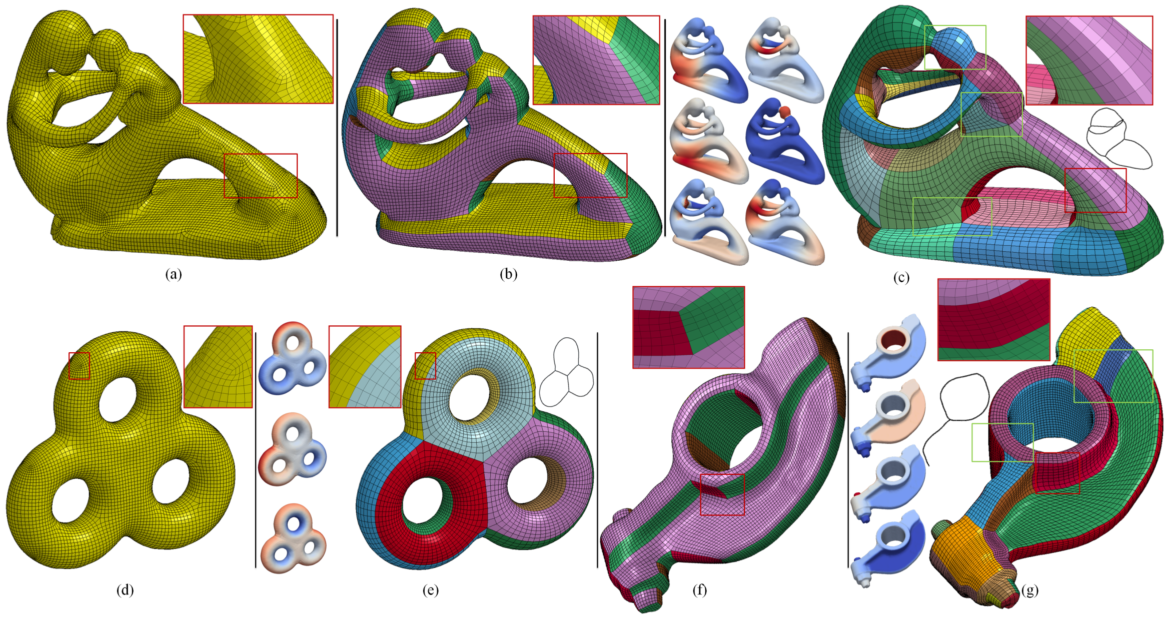


Figure 5.11: All-hex meshes of the Fertility, 3-Torus and Rockerarm. (a, d) Results of [145]; (b, f) results of [74] (each color corresponds to one axis direction); and (c, e, g) results of our method. Red windows highlight the reduction of surface singularities.

Guy models. Compared to [125], our quad layouts align to sharp creases better (see the comparison in the red windows).

The all-hex meshes and volumetric T-meshes generated from our algorithm can be directly used in finite element analysis (FEA) and isogeometric analysis (IGA). Fig. 5.13 shows the analysis results of the Fertility and Rockerarm models. We apply both FEA and

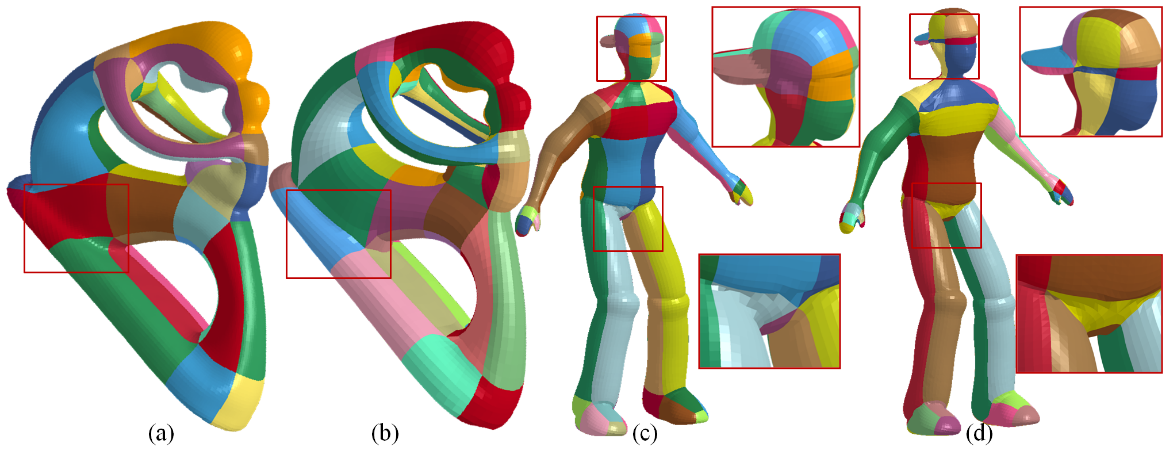


Figure 5.12: Comparison of the quad layouts. (a, c) Results of [125]; and (b, d) our results.

IGA for each model with similar loading and boundary conditions to solve the traditional linear elasticity problem. For both models, we fix the movement on one end and apply a displacement load on the other end (Young's modulus: 200 GPa; density: 7900 kg/m^3 and Poisson's ratio: 0.3). The weighted T-splines [72] are generated for both models from the T-meshes with local knot vectors and weighted basis functions. Bézier representations of solid T-splines are then extracted for the isogeometric analysis. The displacement results obtained for both FEA and IGA are shown in Fig. 5.13. From the results we can observe that our algorithm generates all-hex meshes for the FEA analysis in engineering practice. With the valid volumetric T-meshes, we also create analysis-suitable T-splines (weighted T-splines) for IGA applications.

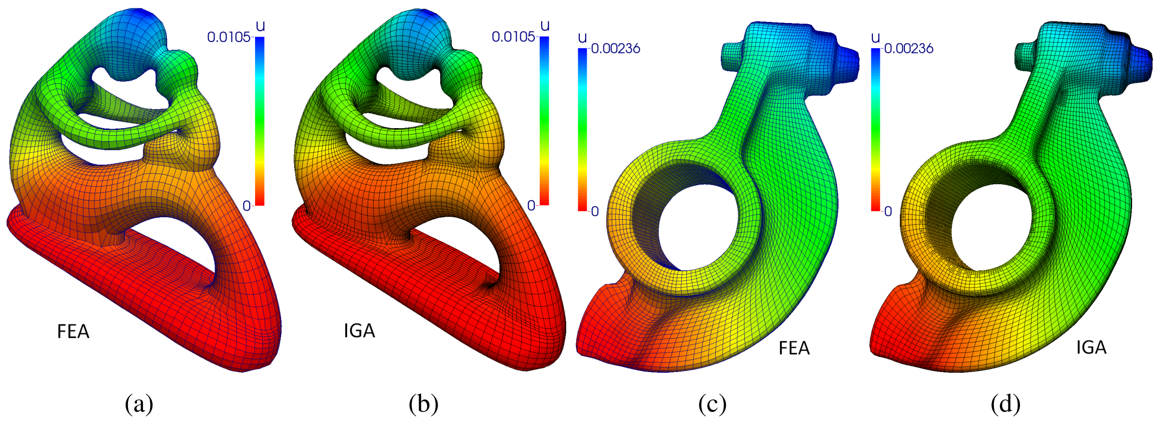


Figure 5.13: Analysis results. (a, b) FEA and IGA results for the Fertility model; and (c, d) FEA and IGA results for the Rockerarm model. All-hex meshes are shown in (a, c) and Bézier elements are shown in (b, d).

Limitations. For the computation of SLO eigenfunctions, the Catmull-Clark basis functions and quadrilateral control meshes are used in order to obtain a high-order representation of the surface, thus a remeshing process between triangle and quadrilateral meshes is needed. For the eigenfunction-based CVT, the surface segmentation results vary with different initializations of generators. The suitable shapes for the skeleton-based CVT are those that can be described well with curve skeletons. For other shapes, we can still generate polycubes by using the HBECVT scheme directly in the global normal space. To generate a valid polycube, geometric constraints are applied to edit the segmentation results. Such editing is not fully automatic and sometimes user interactions are needed. In addition, we do not have a strict theoretical proof that the imposed geometric constraints guarantee a valid polycube generation for an arbitrary geometry.

5.5 Conclusion and Future Work

In conclusion, we have developed a new two-step surface segmentation process for polycube construction, which can be used to generate all-hex meshes and volumetric T-meshes. A novel eigenfunction-based CVT algorithm is presented and vertices are clustered into several components in the p-dimensional SLO eigenfunction space, which avoids over-segmentation and jaggy boundaries, eliminates effect of noise and reduces the computational cost. We also present an automatic polycube construction algorithm based on the skeleton-based CVT. Inherited from the HBECVT model, our proposed skeleton-based CVT algorithm can automatically and robustly construct polycubes for general arbitrary geometric domains. It is suitable for slim cylindrical objects and can remove unnecessary singularities with compact polycube structure. By applying pre-defined geometric constraints, we obtain a valid polycube and then generate quality all-hex meshes and volumetric T-meshes. In the future, we plan to improve the robustness of the skeleton-based CVT by including more advanced methods for singularity placement. We also plan to construct solid T-splines and use them in isogeometric analysis.

Chapter 6

Feature Preservation in Surface Parameterization Using Secondary Laplace Operator and Loop Subdivision

Surface parameterization is of great importance for many applications such as quadrangulation, texture mapping and surface fitting. An important issue for surface parameterization is how to align parametric lines with feature directions. To address this issue, in this chapter we first utilize Loop subdivision basis functions and isogeometric analysis (IGA) to calculate eigenfunctions of the secondary Laplace operator (SLO) on triangle meshes. Eigenfunctions are then used for centroidal Voronoi tessellation (CVT) based surface segmentation, and boundaries of the segmented regions are extracted as feature lines which contain concave creases and convex ridges. Along each feature line, adjacent triangles are defined as guidance triangles to parameterize the surface using a constrained cross field method, where feature lines are preserved and aligned to parametric lines. We also apply our algorithm to generate T-meshes, truncated T-splines and weighted T-splines to complex geometries for IGA analysis. The key contributions of this chapter include:

1. Loop subdivision basis functions are coupled with isogeometric analysis (IGA) to solve the eigenproblem of the SLO over triangular meshes, which is defined based on the second fundamental form of the surface;

2. A CVT-based surface segmentation approach is developed in the eigenfunction space, where the L_∞ norm is used as the distance measurement. Compared to the L_2 norm distance measurement, the L_∞ norm distance metric is more robust to identify surface features by taking the dominant feature of the difference vector between a vertex and its associated generator. By considering both the eigenfunction similarity and segmented boundary smoothness, regions surrounded by curvature related features can be segmented while generators are iteratively updated in the eigenfunction space; and

3. Boundaries of the segmented regions are used to define guidance triangles and their guidance directions. The constrained cross field method parameterizes the surface with all feature lines aligned to the parametric lines.

6.1 SLO Eigenfunction Computation Using Loop Subdivision Basis Functions

Given an input triangle mesh, we use Loop subdivision basis functions to define a smooth representation of the surface. The SLO eigenfunctions are calculated using the subdivision-based IGA method.

6.1.1 Secondary Laplace Operator

Let $S = \{\mathbf{x}(u,v),(u,v) \in \mathbb{R}^2\}$ be a smooth and closed parametric surface, where (u,v) can also be written as (u^1,u^2) for convenience. The coefficients of the first fundamental form of S are defined as $g_{\alpha\beta} = \langle \mathbf{x}_{u^\alpha}, \mathbf{x}_{u^\beta} \rangle$ ($\alpha,\beta = 1,2$), where $\mathbf{x}_{u^\alpha} = \frac{\partial \mathbf{x}}{\partial u^\alpha}$ and $\mathbf{x}_{u^\beta} = \frac{\partial \mathbf{x}}{\partial u^\beta}$. The coefficients of the second fundamental form of S are defined as $b_{\alpha\beta} = \langle \mathbf{n}, \mathbf{x}_{u^\alpha u^\beta} \rangle$, where $\mathbf{x}_{u^\alpha u^\beta} = \frac{\partial^2 \mathbf{x}}{\partial u^\alpha \partial u^\beta}$ and $\mathbf{n} = (\mathbf{x}_u \times \mathbf{x}_v)/\|\mathbf{x}_u \times \mathbf{x}_v\|$. Let $g = \det[g_{\alpha\beta}]$, $[g^{\alpha\beta}] = [g_{\alpha\beta}]^{-1}$, and $[b^{\alpha\beta}] = [b_{\alpha\beta}]^{-1}$. Given $f \in C^2(S)$, the LBO [61, 130] acting on f is defined as

$$\triangle f = \operatorname{div}(\nabla f) = \frac{1}{\sqrt{g}} \left[\frac{\partial}{\partial u}, \frac{\partial}{\partial v} \right] \left[\sqrt{g} \left[g^{\alpha\beta} \right] [f_u, f_v]^T \right], \tag{6.1}$$

where ∇ is the tangential gradient operator given by

$$\nabla f = [\mathbf{x}_u, \mathbf{x}_v] \left[g^{\alpha\beta} \right] [f_u, f_v]^T, \tag{6.2}$$

and div is the tangential divergence operator defined by

$$\operatorname{div}(\mathbf{v}) = \frac{1}{\sqrt{g}} \left[\frac{\partial}{\partial u}, \frac{\partial}{\partial v} \right] \left[\sqrt{g} \left[g^{\alpha\beta} \right] [\mathbf{x}_u, \mathbf{x}_v]^T \mathbf{v} \right]. \tag{6.3}$$

Improving up the LBO which is defined based on the first fundamental form of the surface [138], the SLO $\blacktriangle$ is defined based on the second fundamental form of the surface [67]. It is given implicitly as

$$\int_S (h \blacktriangle f + \langle \blacklozenge f, \blacklozenge h \rangle) \, \mathrm{dA} = 0, \ \forall h \in C^1(S), \tag{6.4}$$

where the generalized second tangential operator (GSTO) $\blacklozenge$ can be defined as

$$\blacklozenge f = [\mathbf{x}_u, \mathbf{x}_v] \, \Phi \left[b^{\alpha\beta} \right] [f_u, f_v]^{\mathrm{T}} = g_u^{\blacklozenge} f_u + g_v^{\blacklozenge} f_v, \tag{6.5}$$

with $g_u^{\blacklozenge} = \frac{\Phi}{b}(b_{22}\mathbf{x}_u - b_{12}\mathbf{x}_v)$ and $g_v^{\blacklozenge} = \frac{\Phi}{b}(b_{11}\mathbf{x}_v - b_{12}\mathbf{x}_u)$. Different choices of the parameter Φ can be used for various applications [67]. Let λ_L and λ_S be the eigenvalues of LBO and SLO respectively, the corresponding eigenfunctions f_L and f_S should satisfy

$$\triangle f_L = -\lambda_L f_L \ \text{ and } \ \blacktriangle f_S = -\lambda_S f_S. \tag{6.6}$$

Different from the widely employed LBO eigenfunctions, the SLO eigenfunctions can capture the curvature-related surface features [67], which automatically distinguish different components of an object.

6.1.2 Eigenfunction Computation of SLO

Letting $\{\varphi_i\}_{i=1}^N$ be a set of basis functions defined on the surface, where N is the number of vertices and $\varphi_i \in C^2(S)$, f can be approximately represented as $f = \sum_{i=1}^{N} w_i\varphi_i$. Plugging $h = \varphi_j$ ($j = 1, 2, \cdots, N$) into Eq. (6.4), we obtain

$$\sum_{i=1}^{N} w_i \int_S \left\langle \nabla\varphi_i, \nabla\varphi_j \right\rangle \mathrm{dA} = \lambda_l \sum_{i=1}^{N} w_i \int_S \varphi_i\varphi_j \mathrm{dA}$$

and

$$\sum_{i=1}^{N} w_i \int_S \left\langle \blacklozenge\varphi_i, \blacklozenge\varphi_j \right\rangle \mathrm{dA} = \lambda_s \sum_{i=1}^{N} w_i \int_S \varphi_i\varphi_j \mathrm{dA}$$

for the LBO and SLO, respectively. Letting $m^L_{ij} = \int_S \left\langle \nabla\varphi_i, \nabla\varphi_j \right\rangle \mathrm{dA}$, $m^S_{ij} = \int_S \left\langle \blacklozenge\varphi_i, \blacklozenge\varphi_j \right\rangle \mathrm{dA}$, and $c_{ij} = \int_S \varphi_i\varphi_j \mathrm{dA}$, the eigenfunctions of LBO and SLO can be obtained by solving the eigenproblems

$$M^L W = \lambda_L C W \text{ and } M^S W = \lambda_S C W, \tag{6.7}$$

where $M^L = \left[m^L_{ij}\right]$, $M^S = \left[m^S_{ij}\right]$, $C = \left[c_{ij}\right]$, and $W = [w_1, w_2, ..., w_N]^T$.

Since SLO is defined based on the second fundamental form of the surface, a high-order representation of the surface is required to compute its eigenfunctions. Catmull-Clark basis functions and quad control meshes were used in [67] to represent the surface, where surface quadrangulation is required as the preprocessing for input triangle meshes. In this chapter, we directly use triangle meshes with Loop subdivision basis functions to represent the surface, and the eigenproblems of LBO and SLO can be solved using the Loop subdivision based IGA method [67, 91]. Given a triangle mesh K, the Loop subdivision scheme subdivides each triangle into four subtriangles in each subdivision step, where vertices of the refined mesh are calculated as the weighted average of vertices of the unrefined mesh. Let us consider the subdivision process from the k-th level to the $(k+1)$-th level, where $k = 0, 1, \cdots$. The initial mesh K is taken as the subdivision surface of the 0-th level. Let $\mathbf{x}_0^k$ be a vertex at level k with one-ring neighbors $\mathbf{x}_i^k$ $(i = 1, \ldots, n)$, where n is the valence of $\mathbf{x}_0^k$. Then the vertex position is updated by

$$\mathbf{x}_0^{k+1} = (1 - n\alpha)\mathbf{x}_0^k + \alpha(\mathbf{x}_1^k + \mathbf{x}_2^k + \cdots + \mathbf{x}_n^k), \tag{6.8}$$

where $\alpha = \frac{1}{n}\left[\frac{5}{8} - (\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n})^2\right]$. Letting $\mathbf{x}_l^k$ and $\mathbf{x}_r^k$ be the two wing neighbor vertices of the edge $\left[\mathbf{x}_0^k \mathbf{x}_i^k\right]$, the new vertex created on this edge is defined as

$$\mathbf{x}_{0i}^{k+1} = \frac{3}{8}\mathbf{x}_0^k + \frac{3}{8}\mathbf{x}_i^k + \frac{1}{8}\mathbf{x}_l^k + \frac{1}{8}\mathbf{x}_r^k. \tag{6.9}$$

Note that all the newly generated vertices have a valence of 6, while vertices inherited from the original mesh may have a valence other than 6. The vertex of valence 6 is referred to as *a regular vertex*, and the vertex of valence other than 6 is referred to as *an extraordinary vertex*. The limit surface of the Loop subdivision is C^2-continuous at regular vertices and C^1-continuous at extraordinary vertices [118].

To obtain a local parameterization of the limit surface for each triangle in the initial control mesh K, we choose (u, v) as two barycentric coordinates in $(1 - u - v, u, v)$ and define $\bar{T}$ as

$$\bar{T} = \left\{(u, v) \in \mathbb{R}^2 : u \geq 0, v \geq 0, u + v \leq 1\right\}.$$

For a regular surface patch whose corresponding initial triangle has three regular vertices, it can be exactly described by a quartic box-spline via twelve local control vertices and their basis functions. We have

$$\mathbf{x}(u, v) = \sum_{i=1}^{12} N_i(u, v)\mathbf{x}_i, \tag{6.10}$$

where the basis functions N_i are given in [118].

For an irregular surface patch whose corresponding initial triangle has one or more extraordinary vertices, the mesh needs to be subdivided recursively until the parameter values of interest are interior to a regular patch according to a fast evaluation strategy [118] under the assumption that any irregular patch has only one extraordinary vertex. Each subdivision of an irregular patch produces three regular subpatches and one irregular subpatch. The piecewise parametric subdomains $\bar{T}_j^k$ at the subdivision level k are given as follows:

$$\begin{aligned}
\bar{T}_1^k &= \left\{(u,v) : u \in \left[2^{-k}, 2^{-k+1}\right], v \in \left[0, 2^{-k+1} - u\right]\right\}, \\
\bar{T}_2^k &= \left\{(u,v) : u \in \left[0, 2^{-k}\right], v \in \left[2^{-k} - u, 2^{-k}\right]\right\}, \\
\bar{T}_3^k &= \left\{(u,v) : u \in \left[0, 2^{-k}\right], v \in \left[2^{-k}, 2^{-k+1} - u\right]\right\}.
\end{aligned}$$

These subdomains can be mapped onto $\bar{T}$ via the following transformations:

$$\begin{aligned}
t_{k,1}(u,v) &= (2^k u - 1, 2^k v), \quad (u,v) \in \bar{T}_1^k, \\
t_{k,2}(u,v) &= (1 - 2^k u, 1 - 2^k v), \ (u,v) \in \bar{T}_2^k, \\
t_{k,3}(u,v) &= (2^k u, 2^k v - 1), \quad (u,v) \in \bar{T}_3^k.
\end{aligned}$$

The entire irregular patch is then defined by its restriction to each regular subpatch

$$\mathbf{x}(u,v)\Big|_{\bar{T}_j^k} = \sum_{i=1}^{12} N_i(t_{k,j}(u,v))\mathbf{x}_i^{k,j}, \ j = 1,2,3; k = 1,2,\cdots, \tag{6.11}$$

where $\mathbf{x}_i^{k,j}$ are properly chosen from the control vertices around the irregular patch at the subdivision level k, and they define a regular subpatch. Hence, the main task becomes how to compute these control vertices. A Jordan canonical decomposition of the subdivision matrix [118] is used here to speed up the computation in the subdivision process.

For each vertex $\mathbf{x}_i$ of a control mesh K, we can associate it with a basis function φ_i, where φ_i is defined by the limit of Loop subdivision for zero values everywhere except at $\mathbf{x}_i$, where it is one. Note that the basis φ_i is different from the basis N_i in Eq. (6.10). φ_i

is a piecewise function whose support covers 2-ring neighboring triangles, whereas N_i is defined on one triangle only [6]. Using the basis functions $\{\varphi_i\}_{i=1}^N$, the limit surface of Loop subdivision can be expressed as $S = \sum_{i=1}^{N} \varphi_i(\mathbf{x}_i)\mathbf{x}_i$. To solve the eigenproblems of Eq. (6.7), we follow three main steps:

1. Precompute Loop subdivision basis functions and their first derivatives for each patch;
2. Evaluate matrice elements m_{ij} and c_{ij} over S using the 6-point Gauss-Legendre integral formula; and
3. Assemble matrice elements m_{ij} and c_{ij} into the eigenproblem system $MW = \lambda CW$.

As mentioned above, $(1-u-v, u, v)$ are the barycentric coordinates of the generic triangle in the mesh K. Using this parameterization, our discretized representation of K is $K = \bigcup_\alpha T_\alpha$, $\mathring{T}_\alpha \cap \mathring{T}_\beta = \emptyset$ for $\alpha \neq \beta$, where $\mathring{T}_\alpha$ is the interior of the triangular patch T_α. Each triangular patch is assumed to be parameterized locally as

$$\mathbf{x}_\alpha : \bar{T} \to T_\alpha;\ \ (u,v) \mapsto \mathbf{x}_\alpha(u,v), \tag{6.12}$$

where $\mathbf{x}_\alpha(u,v)$ is defined by Eqs. (6.10) and (6.11). Note that our parameterization has no overlap. Each point $\mathbf{x} \in K$ has its unique parameter coordinates except at the boundary of each patch. With this parameterization, the matrix element c_{ij} can be computed as

$$c_{ij} = \int\limits_S \varphi_i\varphi_j \mathrm{dA} = \sum_\alpha \int_{T_\alpha} \varphi_i\varphi_j \mathrm{dA} = \sum_\alpha \iint_{\bar{T}} \varphi_i(\mathbf{x}_\alpha(u,v))\varphi_j(\mathbf{x}_\alpha(u,v))\,\sqrt{g}\mathrm{dudv},$$

and the matrix element m_{ij} is replaced by

$$m_{ij}^L = \int\limits_S \left\langle \nabla\varphi_i, \nabla\varphi_j \right\rangle \mathrm{dA} = \sum_\alpha \iint_{\bar{T}} [\varphi_{iu}, \varphi_{iv}] \left[g^{\alpha\beta}\right] \left[\varphi_{ju}, \varphi_{jv}\right]^T\ \sqrt{g}\mathrm{dudv}$$

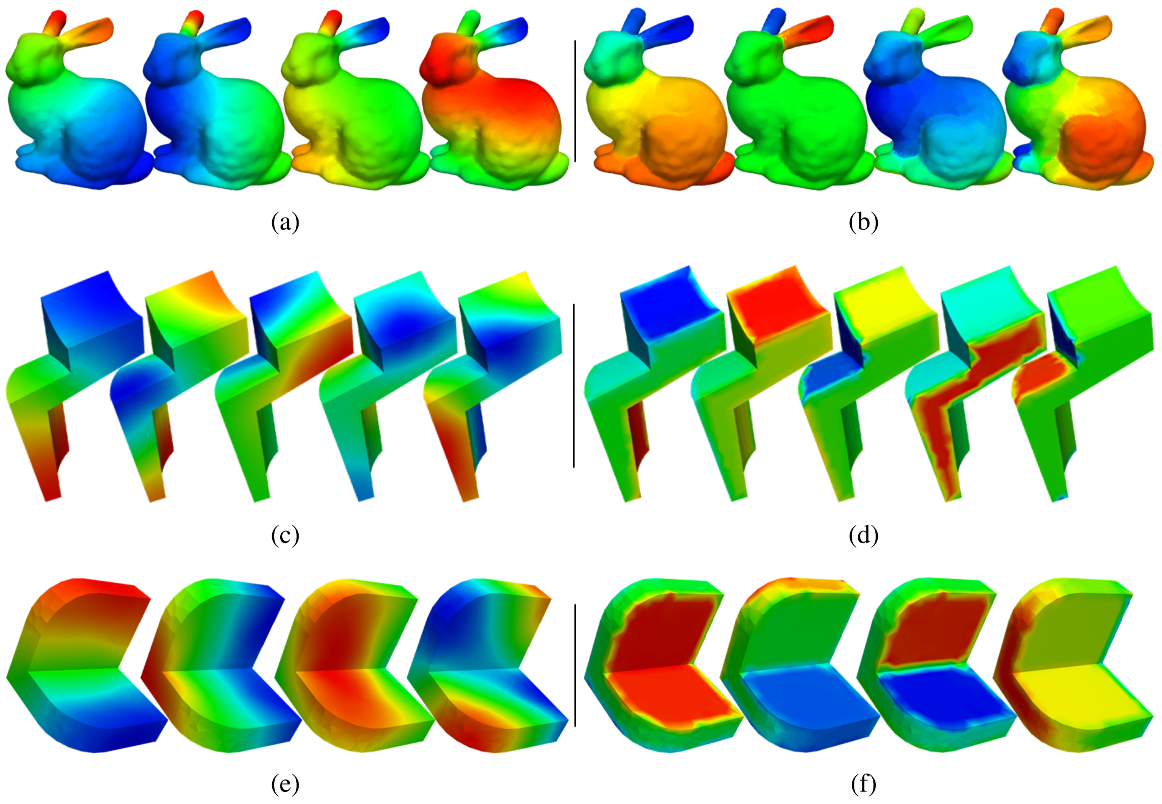


Figure 6.1: The first four or five eigenmodes of the LBO (a, c, e) and SLO (b, d, f) for the Bunny (a, b), Hook (c, d) and *L*-shape (e, f) models.

and

$$m^S_{ij} = \int_S \left\langle \blacklozenge\varphi_i, \blacklozenge\varphi_j \right\rangle \mathrm{dA} = \sum_\alpha \iint_{\bar{T}} \Phi^2 \left[\varphi_{iu}, \varphi_{iv}\right] \left[b^{\alpha\beta}\right] \left[g_{\alpha\beta}\right] \left[b^{\alpha\beta}\right] \left[\varphi_{ju}, \varphi_{jv}\right]^T \sqrt{g}\mathrm{dudv},$$

where $\varphi_{iu} = \frac{\partial\varphi_i(\mathbf{x}_\alpha(u,v))}{\partial u}$ and $\varphi_{iv} = \frac{\partial\varphi_i(\mathbf{x}_\alpha(u,v))}{\partial v}$. The integration on the triangle $\bar{T}$ is computed by subdividing the triangle adaptively and then using the Gauss-Legendre quadrature.

We can then assemble c_{ij} and m_{ij} into the global matrices C and M, and solve the eigenproblems accordingly. Various eigenfunctions of LBO/SLO reflect surface features at different scales. Fig. 6.1 shows the first four eigenfunctions of the Bunny and *L*-shape models, as well as the first five eigenfunctions of the Hook model. To detect both concave creases and convex ridges of an object, we calculate SLO eigenfunctions with $\Phi = 1$ for all the models in this chapter. We can observe that the LBO eigenfunctions follow the main structure of the surface smoothly, but it is insensitive to the variation of surface curvatures.

Compared to LBO, the SLO eigenfunctions can detect curvature related features (e.g., the concave creases of the Bunny, and sharp features of the Hook and L-shape) because it is defined based on the second fundamental form of the surface [67].

6.2 Surface Parameterization with Feature Preservation

After eigenfunction computation, in this section we first apply CVT based surface segmentation to extract feature lines, and then utilize the cross field method to generate a feature-aligned surface parameterization.

6.2.1 Feature Extraction via Surface Segmentation

We begin this section by reviewing CVT-based clustering techniques. The CVT scheme has been introduced to various fields and applications, including both image and mesh processing [27, 131, 26], which generates an optimal domain partition corresponding to an optimal distribution of generators. Given an input point cloud and L generators, each point is firstly assigned to its nearest generator with certain distance metric to construct L non-overlapping Voronoi regions. The centroid of each Voronoi region can be computed by minimizing an energy function E which measures the clustering similarity. Each generator is iteratively updated to be the centroid of its associated Voronoi region, where a new partition can be computed. Such an algorithm aims at minimizing the energy function E until each centroid coincides with the corresponding generator. The edge-weighted CVT (EWCVT) model [131, 15] can segment images with noise by combining the image intensity information together with the length of cluster boundaries in the edge-weighted clustering energy function. As a follow up, the harmonic EWCVT (HEWCVT) [43, 46, 49] outperforms the classic CVT methods [27, 131] by introducing a harmonic form of clustering energy functional to generate stable image segmentation results. Several methods have been proposed to compute the CVT on curved surfaces [21, 3, 126, 135]. The CVT energy function from the Euclidean space was further extended to the spherical and hyperbolic spaces in [106], called the universal covering spaces of surfaces. A GPU-based method was proposed in [107] for computing the CVT on the planar domain, leading to a significant speedup over CPU-based methods. CVT-based surface remeshing algorithms [143, 144] compute restricted Voronoi diagrams defined as the intersection between the input mesh and a Voronoi diagram. The CVT method [75] can also be used for line segments and graphs. The harmonic boundary-enhanced CVT (HBECVT) [45, 44, 48] extends the HEWCVT-based image segmentation to mesh segmentation in the normal space for automatic polycube construction. In this section, we further extend the HBECVT-based clustering idea to the SLO eigenfunction space for surface segmentation of triangle meshes.

We can map vertices onto a p-dimensional space by selecting p eigenmodes of the SLO. Let $\Psi = \{\psi_i\}_{i=1}^{n_v}$ denote p-dimensional eigenfunctions for all vertices of the surface mesh with $\psi_i = \left(\phi_{i1}, \ldots, \phi_{ip}\right)$, where n_v is the total number of vertices, ψ_i represents the assigned p-dimensional vector of the i^{th} vertex $\mathbf{x}_i$, and $\phi_{ij} = f_j(\mathbf{x}_i), j = 1, 2, \cdots, p$, $f_j = \sum_k w_k \varphi_k$ is the j^{th} eigenfunction of the SLO. Let $C = \{c_l\}_{l=1}^{L}$ denote a set of predefined p-dimensional vectors, which can be initialized by choosing L random vertices, and we take their p-dimensional eigenfunction vectors. The Voronoi regions $V = \{V_l\}_{l=1}^{L}$ in Ψ can be obtained by assigning each vertex to the cluster with the closest generator according to the distance metric:

$$V_k = \{\psi_i \in \Psi : \mathrm{dist}(\psi_i, c_k) \le \mathrm{dist}(\psi_i, c_l), \quad \text{for } l = 1, \ldots, L\}, \tag{6.13}$$

where $\mathrm{dist}(\psi_i, c_k)$ measures the distance between ψ_i and c_k. To measure the distance between vector ψ_i and generator c_k in the p-dimensional eigenspace, we define the L_∞ norm based distance metric as

$$\mathrm{dist}(\psi_i, c_k) = \sqrt{||\psi_i - c_k||_\infty^2 + \lambda \tilde{n}_k(\mathbf{x}_i)}, \tag{6.14}$$

where the term $||\psi_i - c_k||_\infty = \max\Big\{|\phi_{i1} - c_{k1}|, |\phi_{i2} - c_{k2}|, \ldots, \big|\phi_{ip} - c_{kp}\big|\Big\}$ measures the distance in the p-dimensional eigenspace, the boundary-enhanced term $\tilde{n}_k(\mathbf{x}_i)$ represents the number of vertices that do not belong to the k^{th} cluster within the ω-ring ($\omega = 3$ in this chapter) neighborhood of $\mathbf{x}_i$ and λ is a positive weighting factor to balance these two terms. Note that $\tilde{n}_k(\mathbf{x}_i)$ includes the local neighbouring information of vertex $\mathbf{x}_i$. Given any set of generators $C = \{c_l\}_{l=1}^L$ and any partition $U = \{U_l\}_{l=1}^L$ of Ψ, the L_∞ norm clustering energy function of $(C; U)$ can be defined as

$$E(C;U) = \sum_{i=1}^{n} \left(L \middle/ \sum_{l=1}^{L} \mathrm{dist}^{-2}(\psi_i, c_l) \right). \tag{6.15}$$

The CVT construction can be viewed as an energy minimization process, where the centroid c_k^* of each cluster V_k is calculated by minimizing the clustering energy in Eq. (6.15) with respect to c_k. Since there is no derivative available by using the L_∞ norm, we use the Powell method [96] to numerically calculate c_k^* for each cluster. If the generators of the Voronoi regions $\{V_l\}_{l=1}^L$ coincide to their corresponding centroids, i.e., $c_l = c_l^*$ for $l = 1, \ldots, L$, then we call such Voronoi tessellation $\{V_l\}_{l=1}^L$ a *CVT*. Otherwise, we set $c_l = c_l^*$ for $l = 1, \ldots, L$ to start a new iteration by Eq. (6.13). In our implementation, $\{c_l{}^*\}_{l=1}^L$ and $\{c_l\}_{l=1}^L$ are updated iteratively until the energy variation is less than a predefined threshold. Based on the L non-overlapping clusters of surface vertices, we then construct the boundaries of each cluster to obtain the final surface segmentation. Fig. 6.2(b) shows the CVT-based surface segmentation result of the Hollow-cylinder model with four clusters by using the first three SLO eigenmodes (shown in Fig. 6.2(a)), where neighbouring clusters are rendered with different colors. From the segmentation result, we can observe that regions surrounded by concave creases and convex ridges are well segmented since SLO eigenfunctions are sensitive to the variation of surface curvatures. Based on the segmentation results, boundaries of each cluster are extracted as the feature lines, which will be preserved during the following surface parameterization process.

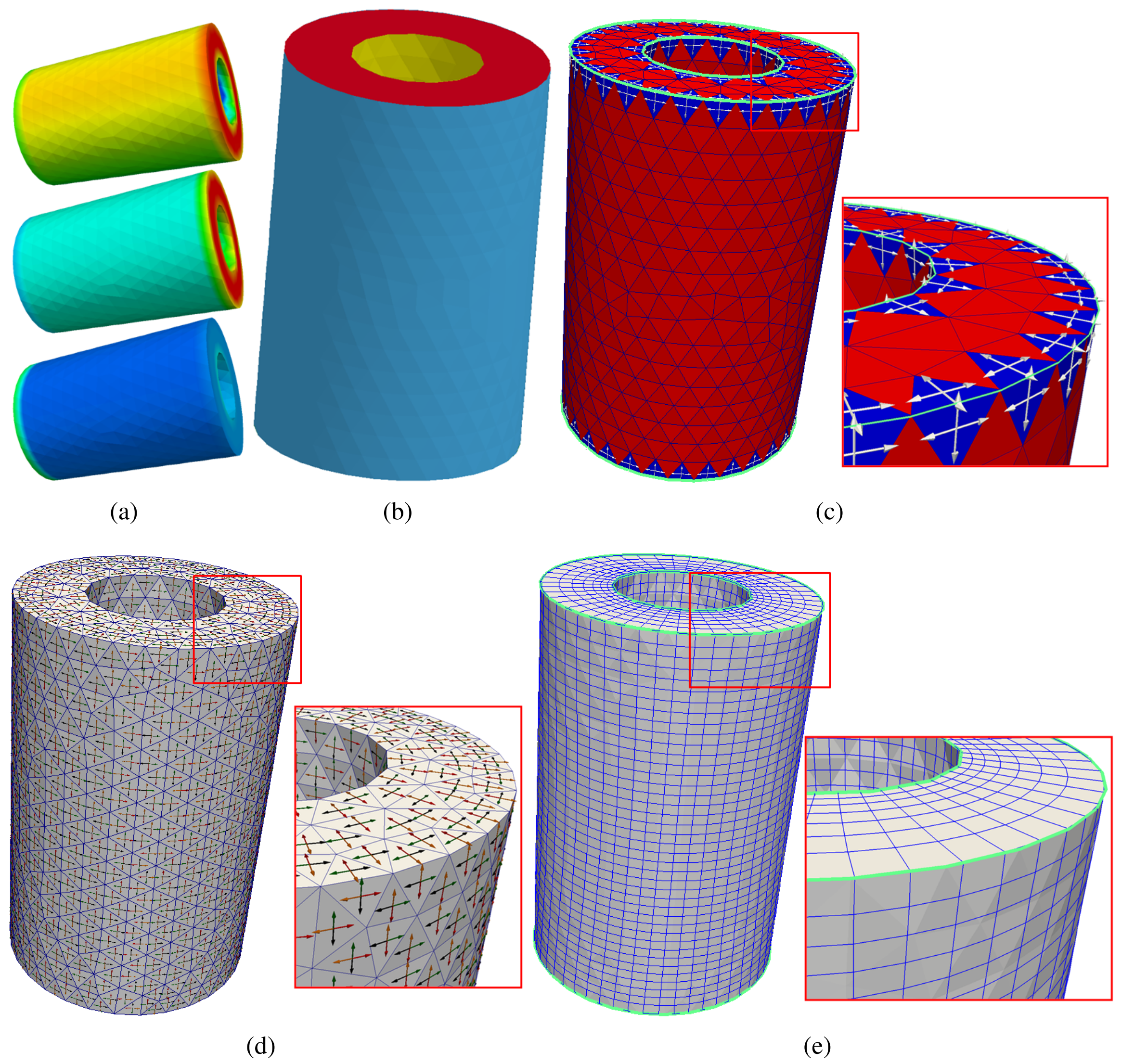


Figure 6.2: Hollow-cylinder model. (a) Modes 1-3 of the SLO; (b) CVT-based surface segmentation; (c) feature lines (green lines), guidance triangles (blue triangles) and their guidance directions (white arrows); (d) the built smooth cross field (four arrows in each triangle); and (e) surface parmaterization result with parametric lines aligned to feature lines.

6.2.2 Surface Parameterization

Using feature lines extracted from surface segmentation, we now build a constrained surface parameterization via the cross field method, where we aim to align feature lines to parametric directions. Along each feature line, we find its adjacent triangles and define them as the guidance triangles. For each guidance triangle, we can calculate its four perpendicular guidance directions by using the edge direction and normal vector. Note that the guidance triangles are used as the constraints to provide guidance for the cross field construction. For the Hollow-cylinder model in Fig. 6.2, the feature lines are extracted via boundary extraction and marked as green lines in Fig. 6.2(c). For each blue guidance triangle, we calculate its guidance directions through the direction of the feature edge and the normal vector, see the result in the red window of Fig. 6.2(c).

For the given triangle mesh K, the cross field of each triangle i is represented by an angle θ_i. Using the guidance triangles with guidance directions defined by feature lines, a cross field can be obtained by minimizing a smoothness energy function [9]

$$\Gamma_0 = \sum_{e_{ij}\in\varepsilon} (\theta_i + \kappa_{ij} + \frac{\pi}{2} p_{ij} - \theta_j)^2, \tag{6.16}$$

where ε contains all edges, e_{ij} is the edge shared by triangles i and j, κ_{ij} is the angle between reference edges e_i and e_j, and p_{ij} represents the integer period jump cross the edge e_{ij}. The mixed-integer solver [9] is used to solve the minimization problem. Fig. 6.2(d) shows the calculated cross field of the Hollow-cylinder. Four arrows for each triangle represent the four directions of the cross field that are perpendicular or parallel with each other. The surface is then partitioned into a disk-like planar region, where all the singularities are positioned on the boundary of the planar region. The surface parameterization is computed as a solution to the constrained minimization problem [9]

$$\sum_{T_i\in M} A_i(||\nabla u - \beta \mathbf{u}_i||^2 + ||\nabla v - \beta \mathbf{v}_i||^2) \rightarrow \min, \tag{6.17}$$

where β is used to control the spacing of the parametric lines (in this paper we choose $\beta = 0.2$), A_i is the area of triangle i, $\mathbf{u}_i$ and $\mathbf{v}_i$ are two directions chosen from the four directions of the cross field in triangle i, and (u, v) are the parametric coordinates. The constraints imposed on (u, v) values correspond to transitions across seams: we want the match across seams to be the same as the guiding cross field. Apart from transition constraints, constraints are imposed on vertices of each feature line. For two vertices of each sharp edge, we compare the edge direction with $\mathbf{u}_i$ and $\mathbf{v}_i$ and set constraints to ensure that these two vertices have the same u or v coordinates. For vertices along the feature lines, we also set integer constraints according to the (u,v) coordinates of the corresponding feature edges so that the generated parametric lines will align with the feature edges exactly. In addition, all singularities are constrained to be at integer locations in the parametric domain, which ensures all singularities are at quad corners and all quad edges are matched across disk-like planar regions. The quadrangulation can be generated by tracing the integer parametric lines on the surface. Fig. 6.2(e) shows the surface parameterization result of the Hollow-cylinder with parametric lines (marked as blue lines), which align to the feature lines detected by surface segmentation.

Remark. Various eigenfunctions of the SLO can capture different curvature related surface features. When we use multiple eigenmodes of the SLO for the surface segmentation, the L_∞ norm distance metric computes the distance from a vertex to a generator by taking the dominant feature of the difference vector in the p-dimensional eigenfunction space. By considering both the eigenfunction similarity and segmentation boundary smoothness in the boundary-enhanced distance metric, our CVT-based surface segmentation can extract curvature related surface features with smooth boundaries. Our surface parameterization generates feature-aligned quadrangulation results by applying two constraints. Firstly, the pre-defined directions in guidance triangles follow feature lines and they are used as constraints for the cross field construction in Eq. (6.16). Secondly, integer constraints are applied to vertices along the feature lines during the minimization process in Eq. (6.17), yielding feature-aligned surface parameterization.

6.3 Results and Discussion

We have applied the presented algorithms to several datasets, including Bunny (Fig. 6.3), Teddy (Fig. 6.4), Octopus (Fig. 6.5), Horse (Fig. 6.6), Hook (Fig. 6.8) and *L*-shape (Fig. 6.9). For each model, we calculated SLO eigenfunctions, CVT-based surface segmentation and surface parameterization with feature preservation. All results were computed on a PC equipped with a 2.93 GHz Intel X3470 CPU and 8GB of Memory. Statistics of all tested models are given in Table 6.1, where we show a summary of the number of singularities and the computational time for each model. For CVT-based surface segmentation, we need to select SLO eigenfunctions and define two parameters: L, the number of clusters; and λ, the weighting parameter that balances the clustering energy and the boundary-enhanced energy.

Table 6.1: Statistics of all the tested models.

Model (vertices, elements)	Modes	λ	Number of clusters	Number of singularities	T_E (s)	T_S (s)	T_P (s)	T_T (s)
Hollow-cylinder (927, 1,854)	1-3	0.02	4	0	6.7	1.4	5.6	13.7
Bunny (14,076, 28,148)	1	0.01	5	42	93.6	18.8	76.7	189.1
Teddy (6,104, 12,204)	1-4	0.01	6	34	41.1	8.3	33.6	83.0
Octopus (14,870, 29,736)	1-6	0.01	9	64	101.7	22.4	89.5	213.6
Horse (15,698, 31,392)	1-4	0.01	6	36	123.2	36.9	134.4	294.5
Hook (2,256, 5,128)	1-5	0.02	6	16	15.5	3.2	12.9	31.6
L-shape (538, 1,072)	1-4	0.02	6	12	4.5	0.9	3.7	9.1

Note: (vertices, elements) - the number of vertices and triangles in the input mesh; T_E - time for eigenfunction computation; T_S - computational time for CVT-based surface segmentation; T_P - computational time for surface parameterization; and T_T - the total computational time. (Time unit: second).

Fig. 6.3 shows the surface segmentation and parameterization results of the Bunny model. We segment the surface into 5 clusters by using the first eigenfunction of the SLO. As shown in Fig. 6.3(a), it is obvious that our CVT-based method segments the concave and convex regions well since the SLO eigenfunctions are sensitive to curvature-related surface features. Based on the segmentation result, we extract boundaries of each cluster as the feature lines and define the corresponding guidance triangles with their guidance directions for the cross field construction. Compared to the unconstrained surface parameterization, our constrained parameterization aligns the resulting parametric lines to the feature lines, see the comparison in the red windows of Fig. 6.3(b, c). Fig. 6.4 shows the results of the Teddy model, where Modes 1-4 of the SLO are used to segment it into 6 clusters. Since multiple eigenmodes are used, here we generate CVT-based segmentation results using the L_2 and L_∞ norm distance metrics respectively under the same initialization; see Fig. 6.4(b, c). We can observe that the segmentation using the L_∞ norm distance measurement segments surface features better, where the head and legs are well separated from the body. Fig. 6.4(d) shows the final feature-aligned surface parameterization result. Figs. 6.5 and 6.6 show results of the Octopus and Horse models, where Modes 1-4 and Modes 1-5 of the SLO are used respectively.

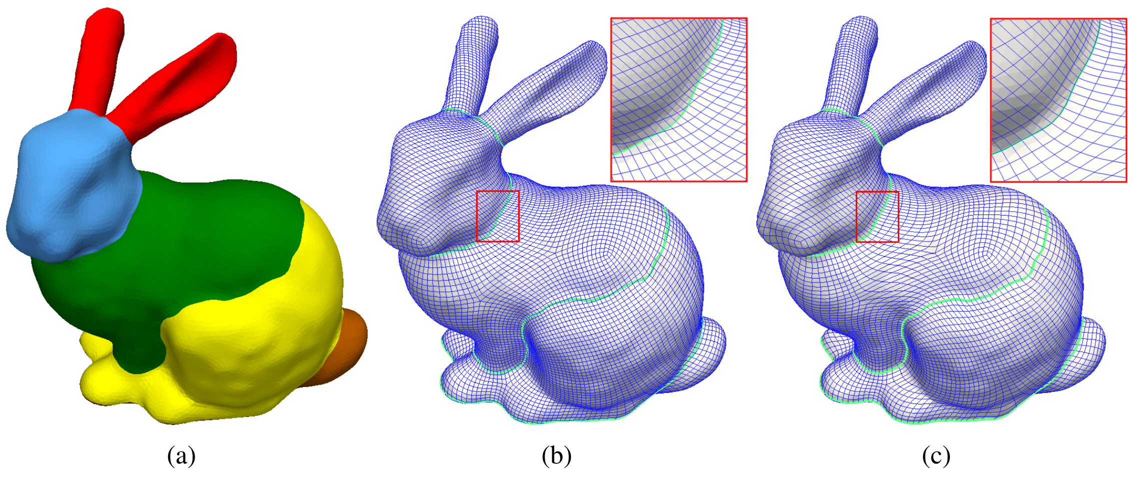


Figure 6.3: Bunny model. (a) Surface segmentation result from the first eigenfunction of the SLO; (b) unconstrained surface parameterization; and (c) constrained surface parameterization. In (b, c), green lines are the extracted feature lines.

We also tested our built Bunny, Teddy, Octopus and Horse models in IGA simulations by using the truncated T-splines [137]. For each model, we first generate the T-mesh by locally refining the uniform quad mesh obtained from surface parameterization. Quads with more than one extraordinary point are subdivided to ensure that no other extraordinary points

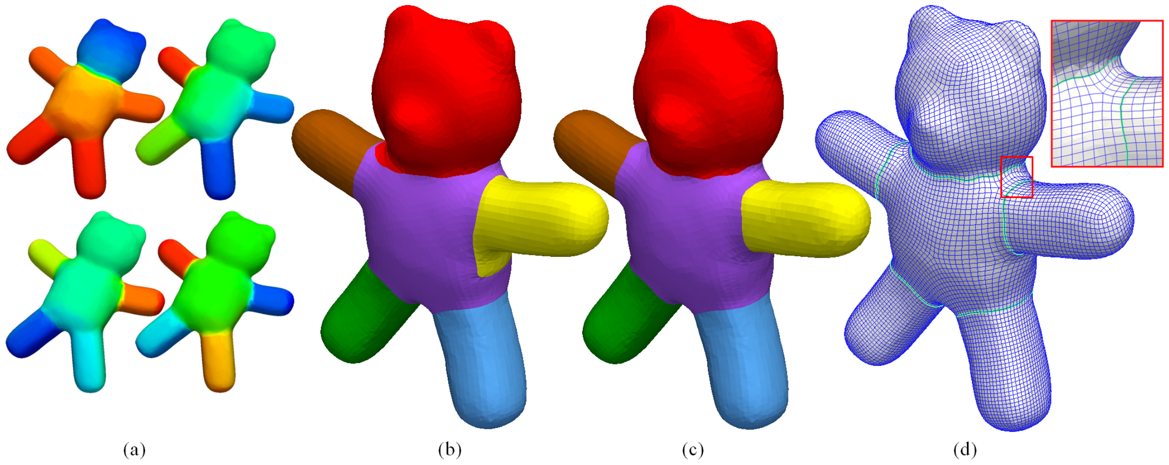


Figure 6.4: Teddy model. (a) Modes 1-4 of the SLO; (b) CVT-based surface segmentation using the L_2 norm distance metric; (c) CVT-based surface segmentation using the L_∞ norm distance metric; and (d) surface parameterization result with feature lines (green lines) aligned.

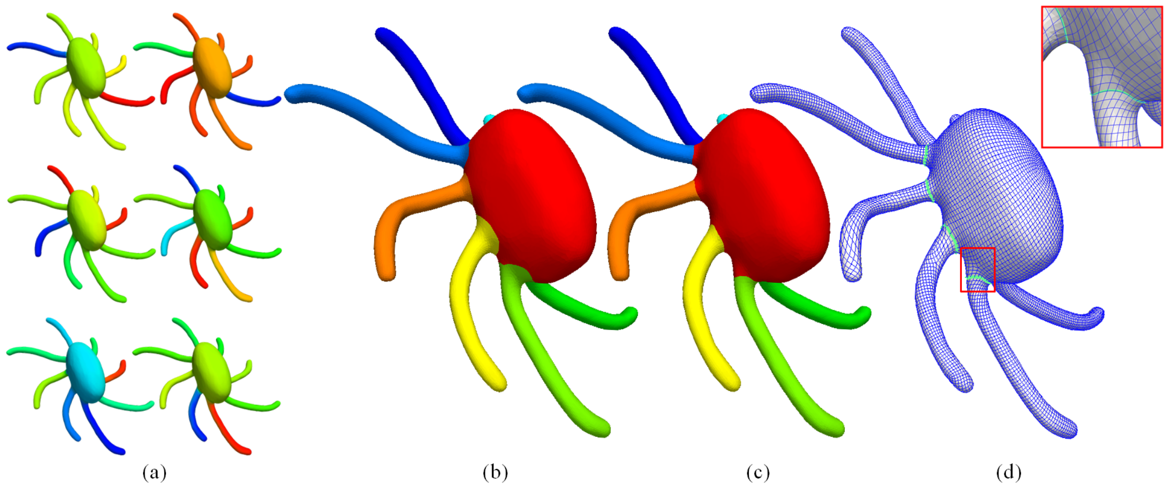


Figure 6.5: Octopus model. (a) Modes 1-6 of the SLO; (b) CVT-based surface segmentation using the L_2 norm distance metric; (c) CVT-based surface segmentation using the L_∞ norm distance metric; and (d) surface parameterization result with feature lines (green lines) aligned.

exist in the 1^{st}-ring neighborhood of any given extraordinary point. Table 6.2 shows the mesh statistics of all testing models, where we use the scaled Jacobian [150] to evaluate the quality of quad meshes and T-meshes. For each node $\mathbf{x}$ in a quad, two edge vectors are defined as $\mathbf{e}_i = \mathbf{x}_i - \mathbf{x}$ ($i = 1, 2$). We have $Jacobian(\mathbf{x}) = det(J)$, where the Jacobian matrix is defined as $J = [\mathbf{e}_1, \mathbf{e}_2]$. If $\mathbf{e}_1$ and $\mathbf{e}_2$ are normalized, $det(J)$ is also called the *scaled Jacobian*.

Truncated T-spline surfaces can be generated from the T-meshes with local knot vectors and truncated basis functions, and Bézier elements are then extracted for the IGA analysis. Here we solve the Laplace equation on all models, where the Dirichlet boundary conditions are predefined [137]. The obtained solution fields are visualized on Bézier elements in Fig. 6.7. From the results we can observe that our algorithm generates valid surface T-meshes and analysis-suitable truncated T-splines for IGA applications.

Table 6.2: Resulting mesh statistics of all the tested models.

Model	Output mesh (vertices, elements)	Output T-mesh (vertices, elements)	Jacobian (worst, best)	Jacobian (T-mesh) (worst, best)
Hollow-cylinder	(4,464, 4,464)	(5,712, 5,616)	(0.84, 1.00)	(0.84, 1.00)
Bunny	(10,551, 10,549)	(11,504, 11,431)	(0.34, 1.00)	(0.33, 1.00)
Teddy	(11,611, 11,609)	(11,944, 11,909)	(0.33, 1.00)	(0.33, 1.00)
Octopus	(12,937, 12,935)	(15,980, 15,676)	(0.29, 1.00)	(0.28, 1.00)
Horse	(14,018, 14,016)	(14,208, 14,185)	(0.30, 1.00)	(0.30, 1.00)
Hook	(13,116, 13,114)	(13,396, 13,354)	(0.48, 1.00)	(0.47, 1.00)
L-shape	(3,242, 3,240)	(3,522, 3,480)	(0.72, 1.00)	(0.72, 1.00)

In addition to the Hollow-cylinder model as shown in Fig. 6.2, we also applied our algorithm to two other CAD models with sharp features. Figs. 6.8 and 6.9 show results of the Hook and *L*-shape models. For the Hook model, we segment the surface into 6 clusters by

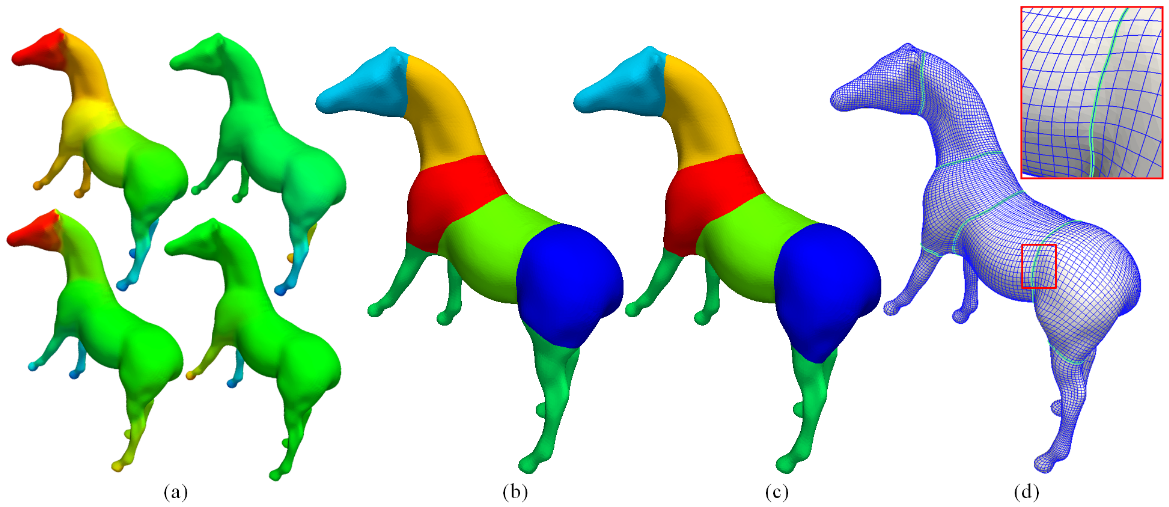
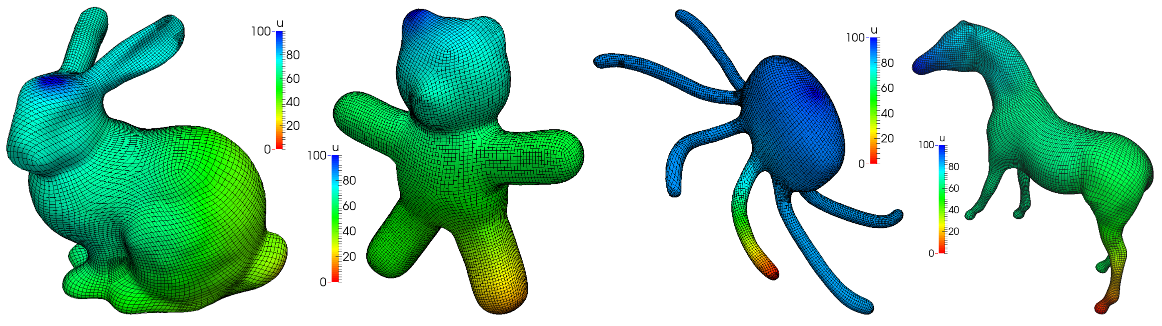


Figure 6.6: Horse model. (a) Modes 1-4 of the SLO; (b) CVT-based surface segmentation using the L_2 norm distance metric; (c) CVT-based surface segmentation using the L_∞ norm distance metric; and (d) surface parameterization result with feature lines (green lines) aligned.



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

Figure 6.7: IGA simulation results. (a) Bunny model; (b) Teddy model; (c) Octopus model; and (d) Horse model.

using Modes 1-5 of the SLO. Fig. 6.8(a) and (b) show the CVT-based segmentation results using the L_2 and L_∞ norm distance metrics respectively in the 5-dimensional eigenspace under the same initialization. For the L-shape model, we segment the surface into 6 clusters by using Modes 1-4 of the SLO. We also generate CVT-based segmentation results using the L_2 and L_∞ norm distance metrics respectively; see Fig. 6.9(a, b). We can observe that regions with sharp features are not well segmented when the L_2 norm is used although each SLO eigenfunction can detect the curvature related features at various scales, while the L_∞ norm distance metric performs better to detect regions surrounded by curvature-related feature lines. Fig. 6.8(c) shows the surface parameterization result of the Hook with all sharp features aligned to the parametric lines. As shown in Fig. 6.9(c), feature lines of the L-shape are extracted as the cluster boundaries and blue triangles along the extracted feature lines (green lines) are selected as the guidance triangles to build the cross field. The resulting parametric lines align well to the feature lines after the constrained surface parameterization process, see the result in Fig. 6.9(d). We also solve the Laplace equation on these three CAD models by using the weighted T-splines [72], where the Dirichlet boundary conditions are strongly imposed. The weighted T-splines are generated for all models from the valid T-meshes with local knot vectors and weighted basis functions [72]. The mesh statistics of these CAD models are also listed in Table 6.2. Fig. 6.7 shows the corresponding solution fields over Bézier elements.

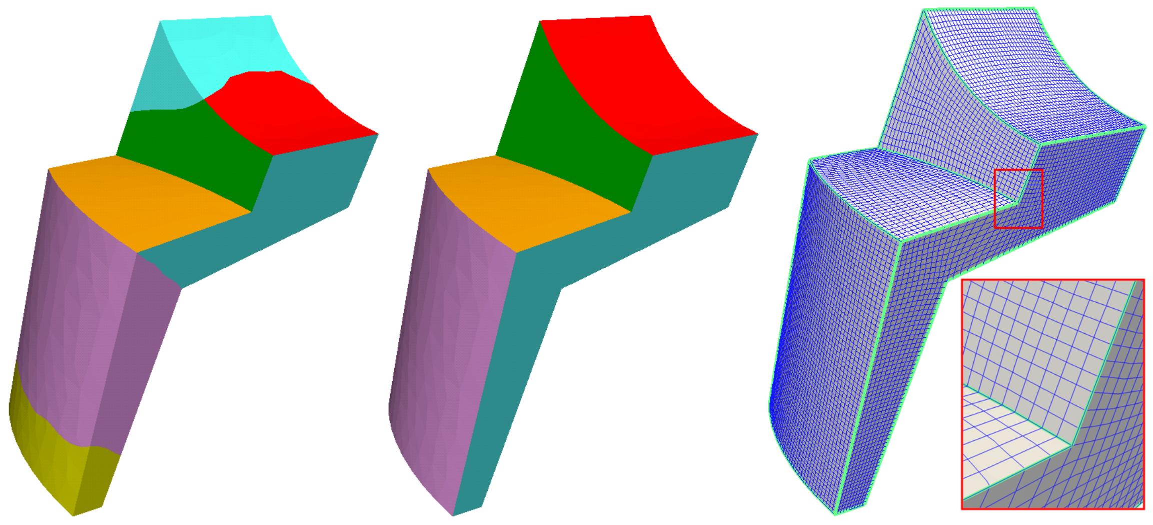

(a) (b) (c)

Figure 6.8: Hook model. (a) CVT-based surface segmentation using the L_2 norm distance metric; (b) CVT-based surface segmentation using the L_∞ norm distance metric; and (c) surface parameterization result with feature lines (green lines) aligned.

Limitations. Although eigenfunctions of the SLO can well detect curvature related surface features, selecting proper eigenmodes for our CVT-based surface segmentation is heuristic and it needs some user interactions. Our constrained surface parameterization algorithm requires aligning parametric lines with integer constraints along the feature lines, therefore large distortion may be introduced sometimes. For this situation, we need to improve the quality of the resulting quadrangulation or T-meshes using smoothing and/or optimization techniques.

6.4 Conclusions and Future Work

In this chapter, we have developed a feature-aligned surface parameterization algorithm with the help of the SLO eigenfunctions and Loop subdivision basis functions. To capture curvature related surface features, we first utilize the Loop subdivision based IGA method to calculate eigenfunctions of the SLO over triangle surfaces. We then employ the CVT-based surface segmentation in the eigenfunction space to extract feature lines which contain con-

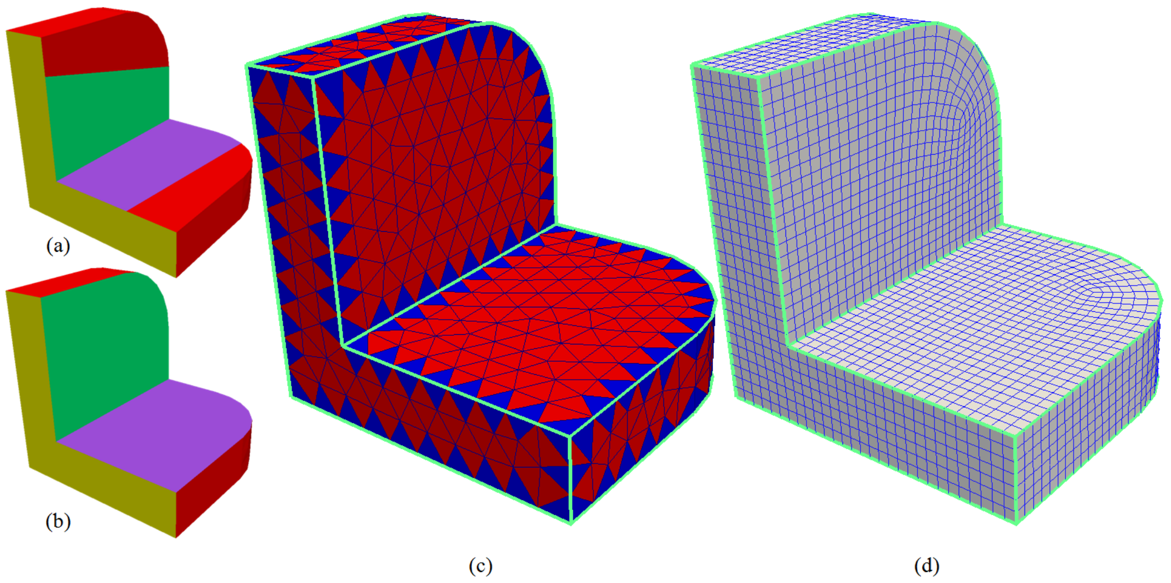


Figure 6.9: L-shape model. (a) CVT-based surface segmentation using the L_2 norm distance metric; (b) CVT-based surface segmentation using the L_∞ norm distance metric; (c) feature lines (green lines) and the corresponding guidance triangles (blue triangles); and (d) surface parameterization result.

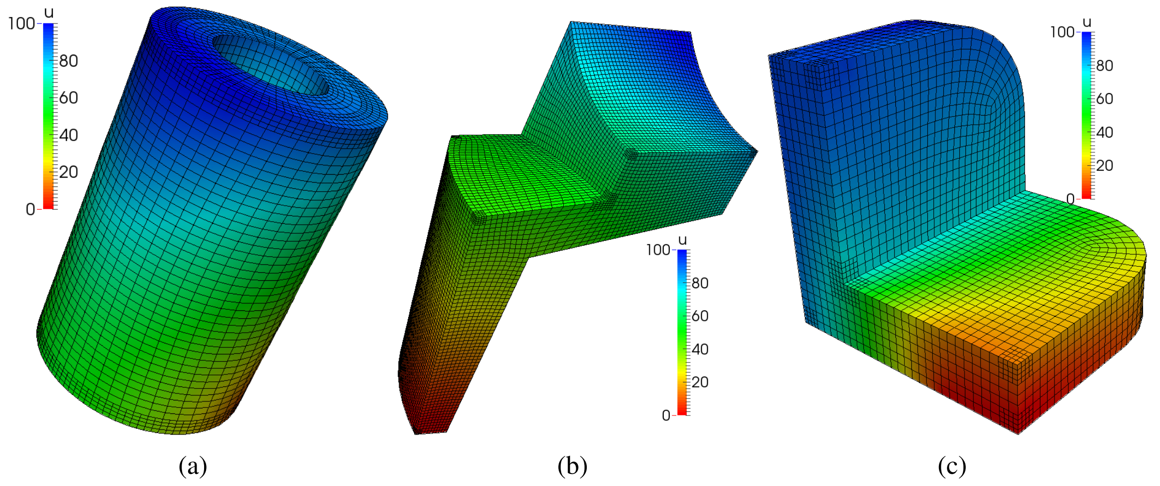


Figure 6.10: IGA simulation results. (a) Hollow-cylinder model; (b) Hook model; and (c) L-shape model.

cave creases and convex ridges. The L_∞ norm is adopted here as the distance measurement. A constrained cross field method is developed for surface parameterization, where feature lines are preserved and aligned to the parametric lines. After the construction of surface parameterization, we can generate uniform quad meshes and T-meshes. With the valid T-meshes, we can also construct analysis-suitable T-splines (truncated and weighted T-splines) for IGA applications. In the future, we plan to study the discretization scheme of the SLO over triangle meshes, and extend the presented algorithm to construct analysis-suitable T-splines for more real IGA engineering applications.

Chapter 7

CVT-Based 3D Image Segmentation and Quality Improvement of Tetrahedral/Hexahedral Meshes Using Anisotropic Giaquinta-Hildebrandt Operator

Given an input 3D image, in this chapter we first segment it into several clusters by extending the 2D harmonic edge-weighted centroidal Voronoi tessellation (HEWCVT) method to the 3D image domain. The Dual Contouring method is then applied to construct tetrahedral meshes by analyzing both material change edges and interior edges, and hexahedral meshes by analyzing interior grid points. An anisotropic Giaquinta-Hildebrandt operator (GHO) based geometric flow method is developed to smooth the surface with both volume and surface features preserved. Optimization based smoothing and topological optimizations are also applied to improve the quality of tetrahedral meshes. The key contributions of our developed algorithms include:

1. The 2D HEWCVT [46] is extended to 3D image segmentation, where 3D spatial information is included in order to eliminate the noise effect. By improving the connectivity of each segment, it generates compact and connected segments without leaving isolated voxels and keeping the connectivity of the structure; and

2. The anisotropic GHO diffusion flow is developed for surface smoothing which preserves surface features while removing the noise with an anisotropic weighting function. Since GHO is defined based on the second fundamental form of the surface, our proposed algorithm is more sensitive to curvature-related features.

7.1 CVT-Based 3D Image Segmentation

CVT-based clustering methods [25, 27] partition discrete data points into non-overlapping clusters with an initialization of generators. It first constructs Voronoi regions by assigning each point to its nearest generator with certain distance metric. For each Voronoi region, we can iteratively calculate its centroid by minimizing a pre-defined energy function until it coincides with the corresponding generator. Inspired by the HEWCVT method [46] for 2D image segmentation, here we extend it to 3D image segmentation.

The input image I is given in the form of function values, $I = \{I(x,y,z)\}$, where x, y, z are indices of X, Y, Z coordinates. Let the dataset $F = \left\{f_{P(i)}\right\}_{i=1}^{n}$ denote all the intensity values $f_{P(i)}$ of the grayscale image, where n is the total number of voxels and $P(i)$ represents the i^{th} voxel in the physical space. Let $C = \{c_l\}_{l=1}^{L}$ denote a set of Voronoi generators with intensity values, where L is the number of clusters. The Voronoi regions $V = \{V_l\}_{l=1}^{L}$ in F corresponding to the generators can be obtained by assigning each voxel to the cluster whose generator is the nearest to it according to the distance metric:

$$V_k = \left\{f_{P(i)} \in F : \text{dist}\left(f_{P(i)}, c_k\right) \leq \text{dist}\left(f_{P(i)}, c_l\right), \quad \text{for } l = 1, \ldots, L\right\}, \tag{7.1}$$

where $\mathrm{dist}\left(f_{P(i)}, c_k\right) = \sqrt{\left|f_{P(i)} - c_k\right|^2 + 2\lambda \hat{n}_k(P(i))}$ measures the edge-weighted distance between $f_{P(i)}$ and c_k in the grayscale space. The edge-weighted term $\hat{n}_k(P(i))$ represents the number of voxels that do not belong to the k^{th} cluster within ω-ring spherical neighbours of $P(i)$, which includes the local 3D spatial information in the physical space. Here we choose a relatively small value of ω ($\omega = 3$) for all the examples in order to reduce the computational cost. Given any set of generators $C = \{c_l\}_{l=1}^{L}$ and any partition $U = \{U_l\}_{l=1}^{L}$ of F, we can define the corresponding HEWCVT energy function of $(C; U)$ as

$$E(C;U) = \sum_{i=1}^{n} \left(L \Big/ \sum_{l=1}^{L} \mathrm{dist}^{-2}\left(f_{P(i)}, c_l\right) \right). \tag{7.2}$$

Note that the HEWCVT energy function uses the edge-weighted distance to all generators for each voxel. It means that all the generators partially influence the harmonic average for each voxel. By taking into account the physical information and using the harmonic form of energy function, HEWCVT is robust to the initialization and can eliminate the noise in the 3D image during the segmentation. To calculate the updated centroids $\left\{c_k^*\right\}_{k=1}^{L}$, we minimize the HEWCVT energy function with respect to the generators c_k ($k = 1, \ldots, L$). If the generators of the Voronoi regions $\{V_l\}_{l=1}^{L}$ of F equal to their corresponding centroids, i.e., $c_l = c_l^*$ for $l = 1, \ldots, L$, then we call the Voronoi tessellation $\{V_l\}_{l=1}^{L}$ a *centroidal Voronoi tessellation* of F. The detailed implementation of HEWCVT-3D is explained as follows:

Algorithm of HEWCVT-3D

Given a 3D image $F = \left\{f_{P(i)}\right\}_{i=1}^{n}$, positive integer L and error tolerance ε ($\varepsilon = 10^{-4}$ in this chapter). E_i denotes the HEWCVT energy in the i^{th} iteration. Then perform the following:

1. For each voxel, find its ω-ring neighbouring voxels in advance. Choose L random voxels in the image and take their intensity values as the initialization of the generators $\{c_l\}_{l=1}^{L}$;

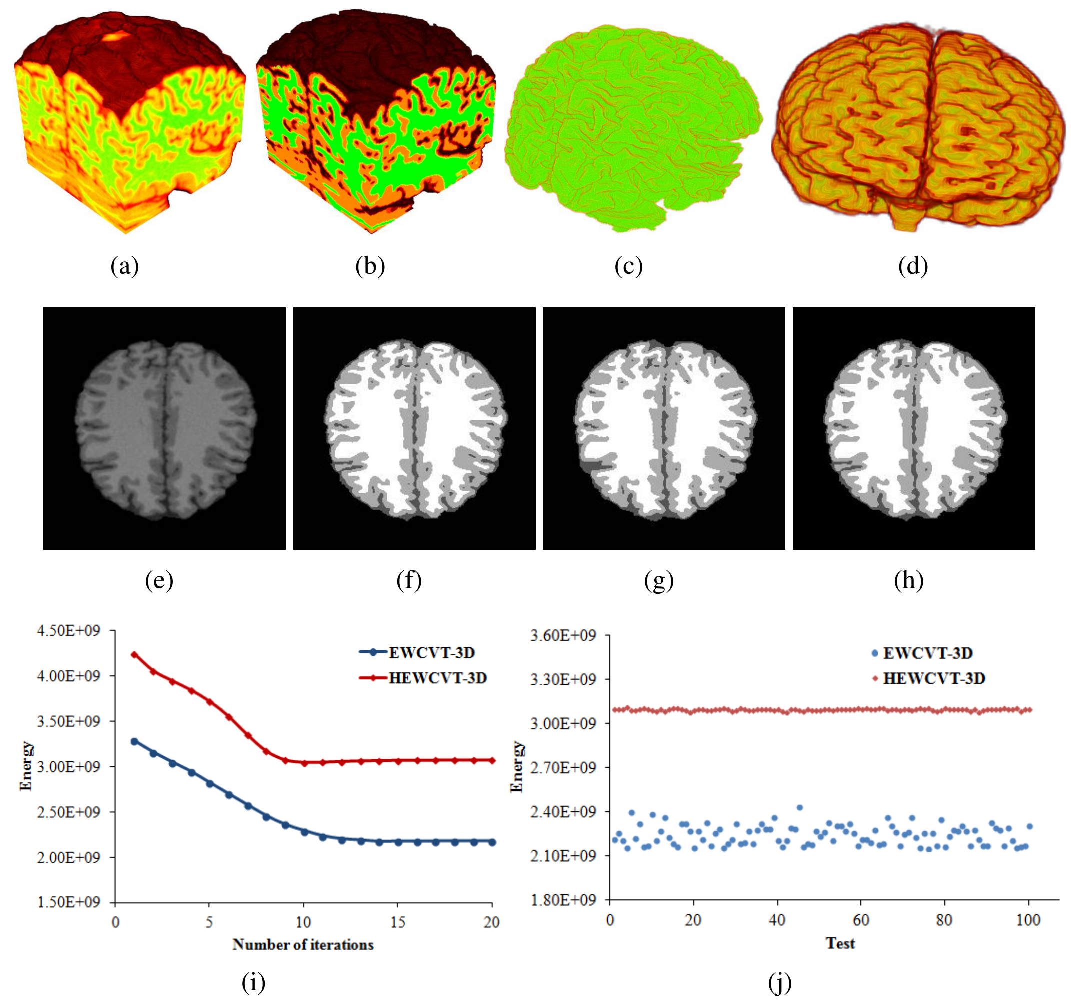


Figure 7.1: Segmentation result of the Brain-1 image with 3% noise and 20% INU. (a) Input 3D image; (b) HEWCVT-3D segmentation result; (c) segmented white matter; (d) segmented gray matter; (e-h) the original image, ground truth, EWCVT-3D and HEWCVT-3D results of the slice 103, respectively; (i) energy outputs; and (j) minimized energy outputs of 100 initializations.

2. Determine the edge-weighted Voronoi clusters $\{V_l\}_{l=1}^{L}$ of F associated with $\{c_l\}_{l=1}^{L}$ by (7.1). For each cluster V_l ($l = 1,\ldots,L$), update the cluster centroid c_l^* by minimizing the HEWCVT clustering energy function;

3. If $\frac{E_{i+1}-E_i}{E_i} < \varepsilon$ is reached, return $(\{c_l\}_{l=1}^{L};\{V_l\}_{l=1}^{L})$; otherwise, set $c_l = c_l^*$ for $l = 1,\ldots,L$ and return to Step 2.

4. Merge small isolated segments to its neighbouring cluster with the longest boundary.

We have tested our HEWCVT-based 3D image segmentation algorithm on a 3D MRI Brain-1 image ($181 \times 217 \times 181$) from BrainWeb dataset [20], with 3% noise and 20% intensity non-uniformity (INU), shown in Fig. 7.1. We segmented the 3D image into four clusters in order to extract the gray matter, white matter, cerebrospinal fluid and background. Fig. 7.1(b) shows the HEWCVT-3D segmentation result after 20 iterations, where neighbouring clusters are rendered with different colors. Fig. 7.1(c) and (d) show the segmented white matter and gray matter, respectively. We also compared our result with EWCVT-3D by extending EWCVT [131] to 3D image domain. From the slice 103 and corresponding segmented images in Fig. 7.1(e-h), we can observe that HEWCVT-3D yields more accurate results in many regions and both of them can eliminate the noise effect. Fig. 7.1(i) shows the energy convergence curves for both EWCVT-3D and HEWCVT-3D under the same initialization. Compared to EWCVT-3D, the HEWCVT-3D energy converges faster to the minimum. We also segmented the image with 100 different random initializations using both EWCVT-3D and HEWCVT-3D, and the minimized energy outputs are shown in Fig. 7.1(j). It is obvious that HEWCVT-3D is much more stable and less sensitive to the initializations than EWCVT-3D.

Remark. Compared to EWCVT-3D, HEWCVT-3D yields more accurate results by imposing a soft membership function with a harmonic average form of the energy function. By taking into account the local 3D spatial information of each voxel, HEWCVT-3D is robust to eliminate the noise effect during the segmentation process. By improving the connectivity of each segment, our HEWCVT-3D automatically and robustly generates compact segments without leaving isolated voxels. The segmented image can be used to generate tetrahedral and hexahedral meshes directly via the Dual Contouring method [149]; see Section 7.2.

7.2 Tetrahedral and Hexahedral Mesh Generation

After the segmentation, we set the image I as a scalar function, $I(x,y,z) \rightarrow J$, where $J = \{0,1,\ldots,L-1\}$ is a set of labels where 0 represents the background and $1,\ldots,L-1$ represent the other materials. Based on the labelled image, we analyze both material changes edges and interior edges to generate tetrahedral meshes by using the Dual Contouring method [149, 152, 153]. A *material change edge* is defined as an edge whose two end points have different label indices. An *interior edge* is an edge whose two end points have the same label. Each material change edge belongs to a boundary cell, while interior cells only contain interior edges. For each octree cell, a dual vertex is generated and the tetrahedral mesh is constructed by connecting the dual vertices with octree grids. For each boundary cell, we calculate the mass center as the dual vertex. The *mass center* is defined as the average of all the middle points of the material change edges in the cell. The cell center is simply selected as the dual vertex for each interior cell. For each material change edge, we first find out all its four surrounding leaf cells and corresponding dual vertices. These four dual vertices and the interior grid point of this edge construct a pyramid. For each interior edge, we also obtain four dual vertices. These four dual vertices and two endpoints of this edge form a diamond. Finally, the pyramids and diamonds can be split into two or four tetrahedra. To handle topology ambiguities, a trilinear function can be introduced to detect the ambiguous cells [153]. The ambiguous cells are split into tetrahedral cells, and tetrahedral dual meshes are then generated by analyzing the edges of these tetrahedral cells. Mesh adaptation can be achieved via an adaptive octree data structure [153].

Instead of analyzing edges, we analyze interior grid points to construct hexahedral meshes from segmented volumetric data. Since each grid point is shared by eight octree cells in a uniform case, we can obtain eight mass centres to construct a hexahedron. To generate haxahedral mesh for each material region with conforming boundaries, material change edges are used to identify the interface between two or several materials [152]. We can also generate adaptive hexahedral meshes by extracting the dual mesh from a hybrid octree [45], which consists of polyhedral cells and each grid point is always shared by eight cells.

Since the function values $I(x,y,z)$ of the segmented images are discontinuous, the surfaces of the generated tetrahedral meshes and hexahedral meshes are bumpy. Fig. 7.2(a) and (f) show the initial tetrahedral meshes of the white matter and gray matter of the Brain-1, where red/green windows in Fig. 7.2(c) and (h) highlight the bumpy surfaces. Fig. 7.3(a) and (f) show the initial hexahedral meshes of the white matter and gray matter of the Brain-1, where red/green windows in Fig. 7.3(c) and (h) highlight the bumpy surfaces. A surface smoothing technique is needed during the following mesh quality improvement.

7.3 GHO-Based Geometric Flow and Quality Improvement

In the meshes generated from the above algorithm, some elements around the boundaries may have poor aspect ratio, therefore the mesh quality needs to be improved. There are two kinds of vertices in 3D meshes, boundary vertices and interior vertices. For each boundary vertex, we use geometric flow to denoise the surface and improve the quality. The quality of interior tetrahedra and hexahedra is simultaneously improved by using the optimization-based smoothing and topological optimizations.

Laplacian smoothing is the most commonly used mesh smoothing method which iteratively relocates a vertex to the geometric center of its neighboring vertices. However, it also produces a shrinking effect and an oversmoothing result. Here we develop a new GHO based geometric flow to smooth the surface, which can preserve the concave/convex features and avoid volume shrinkage. Let $S = \{\mathbf{x}(u,v),(u,v) \in \mathbb{R}^2\}$ be a smooth parametric surface in $\mathbb{R}^3$. Note that (u,v) can also be written as (u^1,u^2) for convenience. The coefficients of the first fundamental form of S are defined as $g_{\alpha\beta} = \langle \mathbf{x}_{u^\alpha}, \mathbf{x}_{u^\beta} \rangle$ $(\alpha,\beta = 1,2)$, where $\mathbf{x}_{u^\alpha} = \frac{\partial \mathbf{x}}{\partial u^\alpha}$ and $\mathbf{x}_{u^\beta} = \frac{\partial \mathbf{x}}{\partial u^\beta}$. The coefficients of the second fundamental form of S are de-

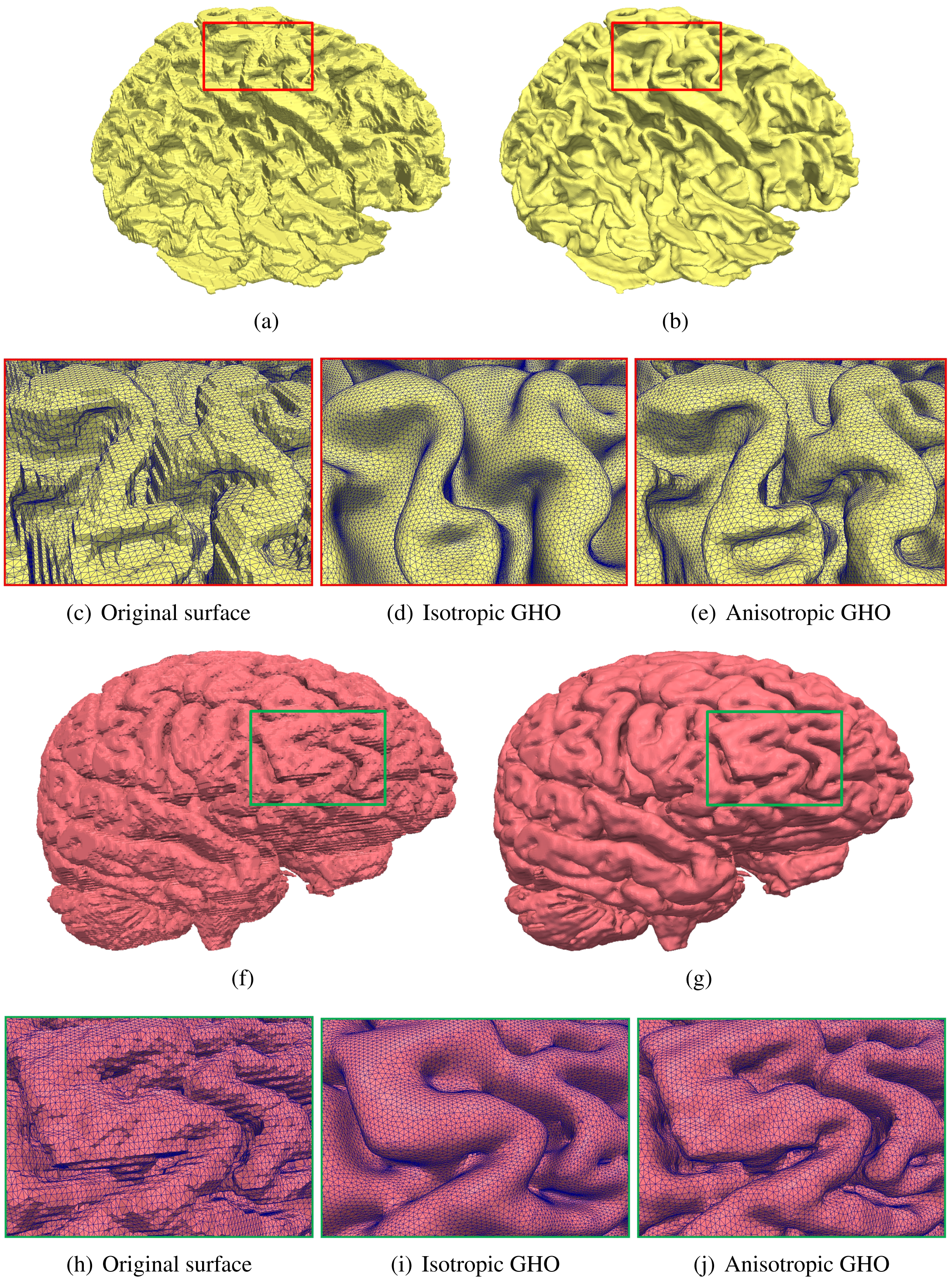
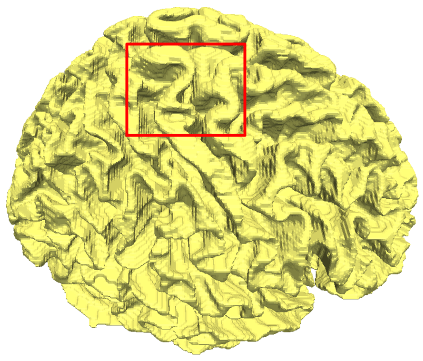
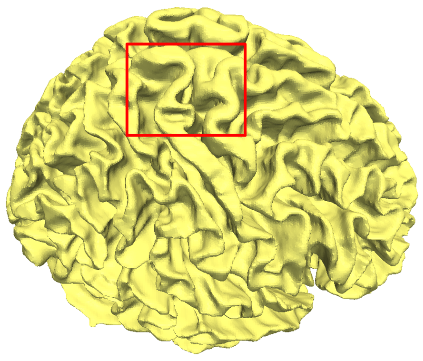
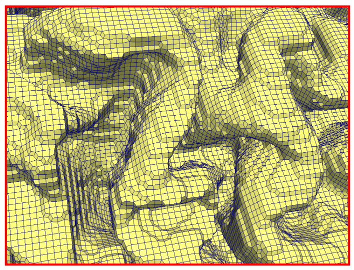
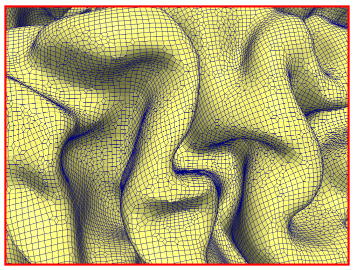


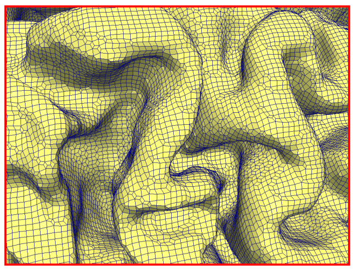
Figure 7.2: Tetrahedral meshes. (a, b) The original and smoothed meshes (using anisotropic GHO) of the white matter, respectively; (c) enlargement of the red window in (a); (d, e) zoom-in pictures showing smoothed results of isotropic and anisotropic GHO flow, respectively; (f, g) the original and smoothed meshes (using anisotropic GHO) of the gray matter, respectively; (h) enlargement of the green window in (f); and (i, j) zoom-in pictures showing smoothed results of isotropic and anisotropic GHO flow, respectively.

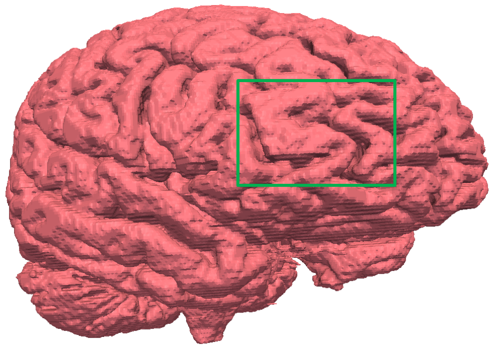

(a)

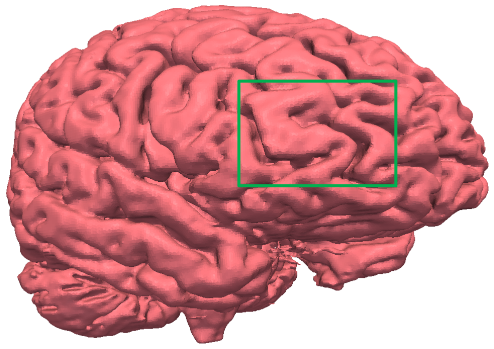

(b)

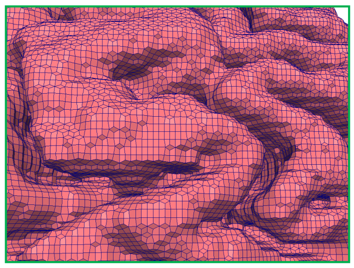

(c) Original surface

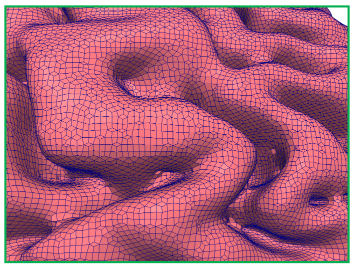

(d) Isotropic GHO


(e) Anisotropic GHO


(f)


(g)


(h) Original surface


(i) Isotropic GHO

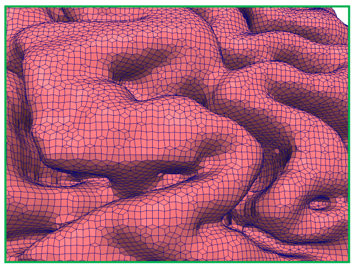

(j) Anisotropic GHO

Figure 7.3: Hexahedral meshes. (a, b) The original and smoothed meshes (using anisotropic GHO) of the white matter, respectively; (c) enlargement of the red window in (a); (d, e) zoom-in pictures showing smoothed results of isotropic and anisotropic GHO flow, respectively; (f, g) the original and smoothed meshes (using anisotropic GHO) of the gray matter, respectively; (h) enlargement of the green window in (f); and (i, j) zoom-in pictures showing smoothed results of isotropic and anisotropic GHO flow, respectively.

fined as $b_{\alpha\beta} = \langle \mathbf{n}, \mathbf{x}_{u^\alpha u^\beta} \rangle$, where $\mathbf{x}_{u^\alpha u^\beta} = \frac{\partial^2 \mathbf{x}}{\partial u^\alpha \partial u^\beta}$ and $\mathbf{n} = (\mathbf{x}_u \times \mathbf{x}_v)/\|\mathbf{x}_u \times \mathbf{x}_v\|$. Let $g = \det[g_{\alpha\beta}]$, $[g^{\alpha\beta}] = [g_{\alpha\beta}]^{-1}$, and $[b^{\alpha\beta}] = [b_{\alpha\beta}]^{-1}$. The mean curvature H and Gaussian curvature K can be given by $H = \frac{b_{11}g_{22} - 2b_{12}g_{12} + b_{22}g_{11}}{2g}$ and $K = \frac{b_{11}b_{22} - b_{12}^2}{g}$. Let $f \in C^2(S)$, the GHO acting on f is defined as

$$\Box f = \operatorname{div}(\Diamond f) = \frac{1}{\sqrt{g}} \left[\frac{\partial}{\partial u}, \frac{\partial}{\partial v} \right] \left[\sqrt{g} K \left[b^{\alpha\beta} \right] [f_u, f_v]^T \right], \tag{7.3}$$

where $\operatorname{div}(\mathbf{v}) = \frac{1}{\sqrt{g}} \left[\frac{\partial}{\partial u}, \frac{\partial}{\partial v} \right] \left[\sqrt{g} \left[g^{\alpha\beta} \right] [\mathbf{x}_u, \mathbf{x}_v]^T \mathbf{v} \right]$ is the tangential divergence operator acting on a C^1 smooth vector field $\mathbf{v}$ and $\Diamond$ is the second tangential operator (STO) given by

$$\Diamond f = [\mathbf{x}_u, \mathbf{x}_v] K \left[b^{\alpha\beta} \right] [f_u, f_v]^T. \tag{7.4}$$

To preserve the volume, here we define a surface diffusion flow using the GHO as

$$\frac{\partial \mathbf{x}}{\partial t} = \operatorname{sign}(K(\mathbf{x})) \Box H(\mathbf{x}) \mathbf{n}(\mathbf{x}). \tag{7.5}$$

Let $S(t)$ denote the smoothed surface at $t \geq 0$, $A(t)$ denote the area of $S(t)$, and $V(t)$ denote the volume of the region enclosed by $S(t)$. Then we have

$$\frac{dA(t)}{dt} = \int_{S(t)} \Box H H d\sigma, \quad \frac{dV(t)}{dt} = \int_{S(t)} \Box H d\sigma. \tag{7.6}$$

Green's formula. Let $\mathbf{v} = (v_1, v_2, v_3)^T$ be a vector field on S and $f \in C^1(S)$ with compact support. Then

$$\int_S < \mathbf{v}, \nabla f > dA = -\int_S f \operatorname{div}(\mathbf{v}) dA, \tag{7.7}$$

where $\nabla f = [\mathbf{x}_u, \mathbf{x}_v] \left[g^{\alpha\beta} \right] [f_u, f_v]^T$ is the tangential gradient operator acting on f. According to the Green's formula [141], we have

$$\frac{dA(t)}{dt} = \int_{S(t)} \Box H H d\sigma = -\int_{S(t)} (\nabla H)^T \diamond H d\sigma, \tag{7.8}$$

and

$$\frac{dV(t)}{dt} = \int_{S(t)} \Box H d\sigma = -\int_{S(t)} (\diamond H)^T \nabla(1) d\sigma = 0. \tag{7.9}$$

Hence, the proposed geometric flow is volume preserving. Since GHO is defined based on the second fundamental form of the surface, it is more sensitive to the curvature-related features. From the definition of $\diamond$ and div, we can derive that

$$\Box f = g_u^{\Box} f_u + g_v^{\Box} f_v + g_{uu}^{\Box} f_{uu} + g_{uv}^{\Box} f_{uv} + g_v^{\Box} f_v, \tag{7.10}$$

where

$$\begin{aligned}
g_u^{\Box} &= -\left[b_{11}(g_{22}g_{122} - g_{12}g_{222}) + 2b_{12}(g_{12}g_{212} - g_{22}g_{112}) + b_{22}(g_{22}g_{111} - g_{12}g_{211})\right] \big/ g^2, \\
g_v^{\Box} &= -\left[b_{11}(g_{11}g_{222} - g_{12}g_{122}) + 2b_{12}(g_{12}g_{112} - g_{11}g_{212}) + b_{22}(g_{11}g_{211} - g_{12}g_{111})\right] \big/ g^2, \\
g_{uu}^{\Box} &= b_{22}/g,\ g_{uv}^{\Box} = -2b_{12}/g,\ g_{vv}^{\Box} = b_{11}/g,
\end{aligned}$$

and $g_{\alpha\beta\gamma} = \langle \mathbf{x}_{u^\alpha}, \mathbf{x}_{u^\beta u^\gamma} \rangle$. Since b_{ij} involves the second order derivatives of the surface, a C^2-continuous surface representation is required. In this section, the Loop subdivision basis functions are adopted to evolve the triangle surface and the Catmull-Clark basis functions are used to evolve the quadrilateral surface.

The above geometric flow smooths the surface by moving each vertex along its normal direction. The isotropic smoothing in Eq. (7.5) can eliminate noise but also smooth out important features. To preserve surface features while removing the noise, we introduce an anisotropic weight $\chi(\mathbf{x})$ for each vertex by using a function of its two principal curvatures, k_1 and k_2 [82]. In order to improve the aspect ratio of the surface mesh, we also add a tangent movement in Eq. (7.5),

$$\frac{\partial \mathbf{x}}{\partial t} = \chi(\mathbf{x})\operatorname{sign}(K(\mathbf{x}))\square H(\mathbf{x})\mathbf{n}(\mathbf{x}) + v(\mathbf{x})\mathbf{T}(\mathbf{x}), \tag{7.11}$$

where

$$\chi(\mathbf{x}) = \begin{cases} 1 & \text{if } |k_1| \le T \text{ and } |k_2| \le T & \text{Case 1,} \\ 0 & \text{else if } |k_1| > T \text{ and } |k_2| > T \text{ and } K > 0 & \text{Case 2,} \\ k_1/(H|K|) & \text{else if } |k_1| = \min(|k_1|, |k_2|, |H|) & \text{Case 3,} \\ k_2/(H|K|) & \text{else if } |k_2| = \min(|k_1|, |k_2|, |H|) & \text{Case 4,} \\ 1/|K| & \text{else if } |H| = \min(|k_1|, |k_2|, |H|) & \text{Case 5,} \end{cases}$$

and T is a user-defined constant (here we select $T = 0.01$). Case 1 is used to detect uniformly noisy regions, which will be smoothed isotropically. Concave/convex features (case 2) will not be smoothed. We also smooth features detected by cases 3-5 with a speed proportional to the minimum curvature and $|K|$ is used to scale the speed of the movement. $v(\mathbf{x})$ is the velocity in the tangent direction $\mathbf{T}(\mathbf{x})$, which controls the strength of the regularization. We first calculate the mass center $m(\mathbf{x})$ for each vertex on the surface, and then project the vector $m(\mathbf{x}) - \mathbf{x}$ onto the tangent plane to obtain $\mathbf{T}(\mathbf{x})$. If the surface has no noise, we can only apply the tangent movement $v(\mathbf{x})\mathbf{T}(\mathbf{x})$ to improve the aspect ratio of the surface mesh while ignoring the vertex normal movement. Eq. (7.11) is solved over triangular surfaces using Loop subdivision based isogeometric analysis [91] and over quadrilateral surfaces using Catmull-Clark based isogeometric analysis [136].

The surface smoothing via GHO-based geometric flow improves the quality of the surface, but the quality of interior mesh also needs to be improved. To measure tetrahedral mesh quality, we choose three metrics [58]: $Q_1 = \theta_{\min}$, the minimal dihedral angle of each element; $Q_2 = \theta_{\max}$, the maximal dihedral angle of each element; and $Q_3 = 8 \cdot 3^{\frac{5}{2}} V \left(\sum_{j=1}^{6} e_j^2 \right)^{-\frac{3}{2}}$, the Joe-Liu parameter, where $\left\{ e_j \right\}_{j=1}^{6}$ are six edge lengths, and V is the volume of each tetrahedron. Three techniques are applied to improve the mesh quality: optimization-based mesh smoothing, face swapping and edge removal [58]. The optimization-based smoothing improves all tetrahedra in the mesh by minimizing the objective function $\epsilon = \sum_{\eta \in \tau} \max(\frac{1}{Q_\eta} - q, 0)^p$, where τ is the set of tetrahedra in the mesh, Q_η represents Joe-Liu value of a tetrahedron $\eta \in \tau$, and q and p are parameters. This approach can improve the overall mesh quality efficiently, but some elements still have poor quality because of the bad valence. Face swapping removes edges with valence 3 or 4 by reconnecting vertices of some elements. Edge removal removes poor quality elements by replacing one ring neighboring tetrahedra of the edge with new tetrahedra of higher quality. To measure the hexahedral mesh quality, we choose two metrics [152]: the scaled Jacobian and the condition number. For each node $\mathbf{x}$ in a hexahedron, three edge vectors are defined as $\mathbf{e}_i = \mathbf{x}_i - \mathbf{x}$ $(i = 1, 2, 3)$. Then the Jacobian matrix is defined as $J = [\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$, and we have $Jacobian(\mathbf{x}) = det(J)$. If $\mathbf{e}_1, \mathbf{e}_2$ and $\mathbf{e}_3$ are normalized, $det(J)$ is also called the *scaled Jacobian*. The *condition number* of the Jacobian matrix is defined as $\kappa(J) = |J| \left|J^{-1}\right|$. Pillowing is firstly applied to improve the mesh quality by inserting one layer around the boundary. Optimization-based smoothing is then implemented to further improve the quality, where the objective function is to maximize the minimum scaled Jacobian.

Fig. 7.2(b, g) show the improved tetrahedral meshes of the white matter and gray matter of the Brain-1 model. Both isotropic and anisotropic GHO diffusion flows are applied to denoise the bumpy surface with the same temporal step size ($t = 0.02$) and iteration number (100 iterations). As shown in Fig. 7.2(d, i), the isotropic GHO diffusion flow smooths out the noise but also blurs the surface features. Compared to the isotropic GHO diffusion flow, it is obvious that our anisotropic GHO diffusion flow preserves surface features better while removing the noise, see Fig. 7.2(e, j). Similarly, Fig. 7.3(b, g) show the improved hexahedral meshes of the white matter and gray matter of the Brain-1 model. Both isotropic and anisotropic GHO diffusion flows are applied to denoise the bumpy surface with the same temporal step size ($t = 0.02$) and iteration number (150 iterations). Compared to the isotropic GHO diffusion flow results in Fig. 7.3(d, i), it is obvious that our anisotropic GHO diffusion flow preserves surface features better while removing the noise, see Fig. 7.3(e, j).

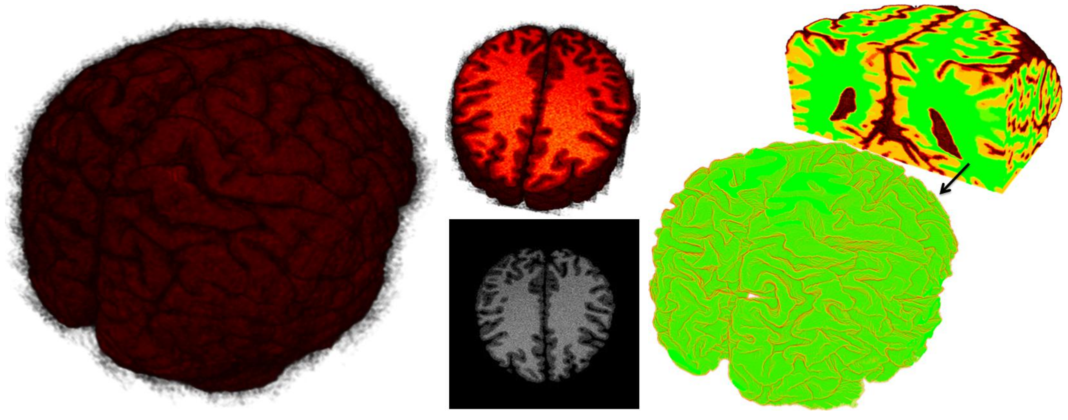
Remark. Since GHO is defined based on the second fundamental form of the surface, it is more sensitive to curvature-related surface features, such as concave creases and convex ridges. However, isotropic geometric flow smooths out important features when reducing the noise. By introducing an anisotropic weighting function which penalizes surface vertices with a large ratio between their two principal curvatures, the anisotropic GHO diffusion flow preserves concave and convex features such as brain wrinkles when removing the noise.

7.4 Results and Discussion

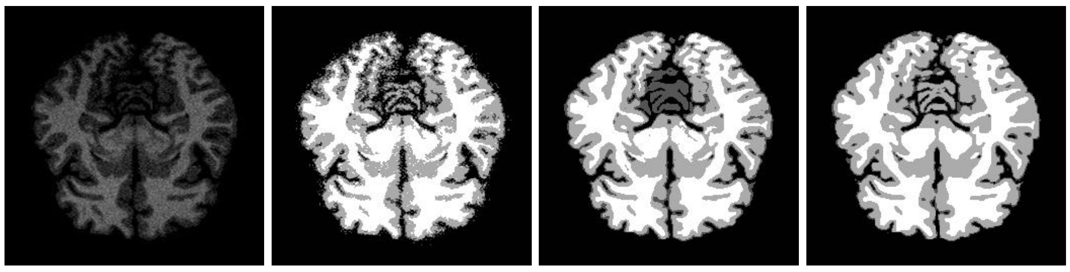
In this section, we apply our presented algorithms to eight 3D medical images that are either noise free or corrupted by different types of noises. All the results were computed on a PC equipped with a 2.93 GHz Intel X3470 CPU and 8GB of Memory. Statistics of all tested models are given in Table 7.1. For HEWCVT-3D based image segmentation, we need to define two parameters: L, the number of clusters; and λ, the weighting parameter that balances the clustering energy and the edge-weighted energy.

Table 7.1: Image segmentation statistics of all tested models.

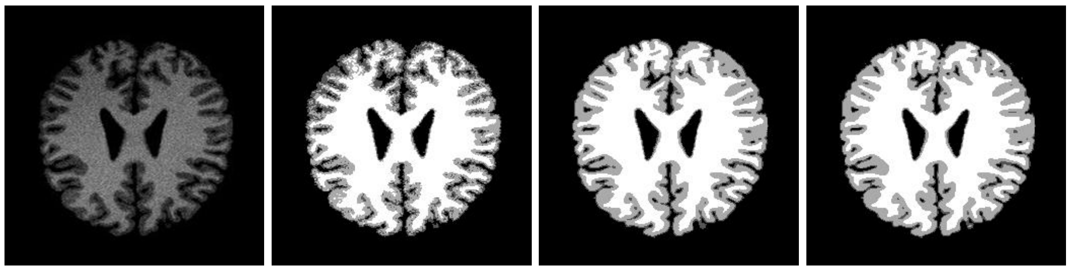
Image		Brain-1	Brain-2	Brain-3	Brain-4	Brain-5	Brain-6	Brain-7	Brain-8
Noise level		3%	3%	5%	5%	7%	7%	9%	9%
INU level		20%	40%	20%	40%	20%	40%	20%	40%
Number of clusters		4	4	4	4	4	4	4	4
λ		0.05	0.05	0.10	0.10	0.15	0.20	0.30	0.30
Average SA	*k*-means	74.38%	73.26%	69.78%	69.24%	68.84%	67.97%	67.46%	66.37%
	EWCVT-3D	89.45%	88.68%	84.49%	86.65%	85.13%	83.28%	81.69%	78.43%
	HEWCVT-3D	**93.23%**	**93.12%**	**92.88%**	**92.26%**	**92.24%**	**91.56%**	**90.96%**	**89.26%**
Average BR	*k*-means	79.88%	76.96%	74.18%	73.24%	72.89%	72.47%	71.86%	71.64%
	EWCVT-3D	91.25%	90.98%	89.69%	89.25%	88.62%	86.29%	83.57%	81.45%
	HEWCVT-3D	**95.28%**	**95.11%**	**94.89%**	**94.66%**	**93.84%**	**92.99%**	**92.16%**	**91.83%**
SCV	*k*-means	13.79%	13.94%	14.22%	14.79%	15.03%	15.87%	16.24%	16.63%
	EWCVT-3D	11.96%	12.63%	12.99%	13.78%	13.98%	14.77%	15.21%	15.68%
	HEWCVT-3D	**0.78%**	**0.78%**	**0.79%**	**0.82%**	**0.85%**	**0.88%**	**0.90%**	**0.93%**
Average time (seconds)	*k*-means	35.2	35.4	35.6	34.8	35.2	35.3	34.2	35.4
	EWCVT-3D	45.2	46.4	45.7	44.9	45.6	45.4	44.9	45.5
	HEWCVT-3D	65.8	65.9	66.7	67.8	66.2	68.3	67.9	68.1



(a) (b) (c)



(d) Original image (e) *K*-means (f) EWCVT-3D (g) HEWCVT-3D



(h) Original image (i) *K*-means (j) EWCVT-3D (k) HEWCVT-3D

Figure 7.4: Brain-6 model. (a) Input image; (b) slice 106; (c) HEWCVT-3D based segmentation; (d-g) and (h-k) from left to right: one slice of the original data, segmented slices after applying *k*-means, EWCVT-3D, and HEWCVT-3D, respectively.

These eight 3D MRI brain images ($181 \times 217 \times 181$) are from the BrainWeb [20], with four levels of noise ($3\%, 5\%, 7\%, 9\%$) and two levels of intensity non-uniformity (INU) ($20\%, 40\%$). We segmented each 3D image into four clusters in order to extract the gray matter, white matter, cerebrospinal fluid and background. Fig. 7.4(a) shows the initial Brain-6 3D image with 7% noise and 40% INU, where the slice 106 in both 3D and 2D domains are highlighted in Fig. 7.4(b). Fig. 7.4(c) shows the HEWCVT-3D based image segmentation, where the green part represent the white matter. We can observe that the noise effect can be well removed during segmentation. We also compared all results with two other methods: *k*-means [92] and EWCVT-3D [131]; see Fig. 7.4(d-k). HEWCVT-3D generates better segmentation results without leaving isolated voxels and keeping the connectivity of the structure, while *k*-means is not robust to handle the noise effect and EWCVT-3D may generate inaccurate results. We first use the segmentation accuracy (SA) [2] to quantitatively evaluate the segmentation results. Given the segmented image R and the ground truth image G obtained from BrainWeb dataset, the SA can be defined as:

$$SA = \frac{N_{Correct}}{N_{Total}} \times 100\%, \tag{7.12}$$

where $N_{Correct}$ represents the number of correctly classified voxels and N_{Total} is the total number of voxels in the image. In order to evaluate the accuracy of feature preservation, we also use the boundary recall (BR) [103] to measure the portion of boundary voxels in the ground truth that are also identified as boundary by the segmentation being evaluated. The BR can be defined as:

$$BR = \frac{TP}{TP+FN} \times 100\%, \tag{7.13}$$

where TP is the number of boundary voxels in G with at least one boundary voxel in R in range of two voxels, FN is the number of boundary voxels in G with no boundary voxel in R in range of two voxels. Large SA and BR values are usually considered high accuracy. We also introduce another metric named the segmentation coefficient of variation (SCV) [46] to evaluate the stability of different methods. For each brain image, we test N ($N = 100$ in this chapter) random initializations of the generators by using *k*-means, EWCVT-3D and HEWCVT-3D. We can get one minimized energy value for each test and the SCV can be defined as:

$$SCV = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(MinE_i - \overline{MinE}\right)^2}}{\overline{MinE}} \times 100\%, \tag{7.14}$$

where $MinE_i$ represents the minimized energy for the i^{th} test, and $\overline{MinE}$ represents the mean of the minimized energy. Large SCV values are usually considered high-variance, otherwise low-variance. The average SA, the average BR and SCV values of the each image with 100 tests are listed in Table 7.1. We can observe that HEWCVT-3D improves the segmentation accuracy compared to k-means and EWCVT-3D. From the comparison of BR values, it is evident that HEWCVT-3D can also better preserve the connectivity of structure compared to the other two methods. With different initializations, the energy function converges to different values for k-means and EWCVT-3D, while HEWCVT-3D is much more stable and less sensitive to initializations with all $SCVs < 1\%$. In addition, our HEWCVT-3D method is also robust to noise since the SA, BR and SCV values do not change much for different levels of noise and INU. Since HEWCVT-3D updates cluster centroids by calculating distances to all centroids for each voxel, the computational cost is higher than the other two methods.

Tetrahedral and hexahedral meshes consisting of the white matter, gray matter and cerebrospinal fluid are generated and the mesh quality is improved via geometric flow based smoothing and optimization. Tables 7.2 and 7.3 shows tetrahedral and hexahedral meshing results for each model. Fig. 7.5(b) shows the improved tetrahedral mesh of Brain-6 with the white matter (yellow), gray matter (red) and cerebrospinal fluid (blue). The improved mesh is in good quality with an dihedral angle range of $(15.11°, 167.17°)$. From the zoom-in pictures we can observe that smoothness and regularity of boundary surfaces between different materials are significantly improved. Fig. 7.5(d, e) and (f, g) show the improved meshes of the white matter and gray matter, respectively, with mesh adaptation highlighted in zoom-in pictures. We can observe that surface features are well preserved during the surface denoising via the anisotropic GHO diffusion flow. Fig. 7.6(b) shows

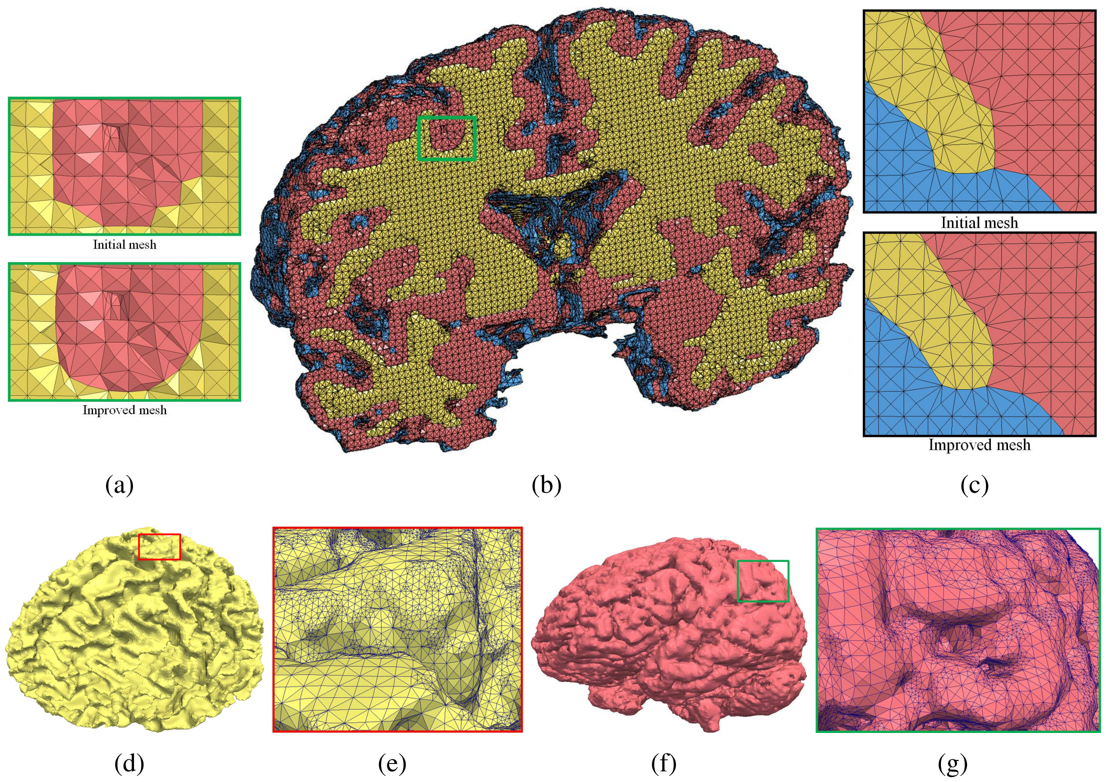


Figure 7.5: Tetrahedral mesh of the Brain-6 model. (a, b) Cross section of the final tetrahedral mesh, with zoom-in pictures of the initial and improved meshes of the green box; (c) zoom-in pictures of the initial and improved meshes with three neighboring materials; (d) improved tetrahedral mesh of the white matter; (e) enlargement of the red window in (d); (f) improved tetrahedral mesh of the gray matter; and (g) enlargement of the green window in (f).

the improved hexahedral mesh of Brain-6 with the white matter (yellow), gray matter (red) and cerebrospinal fluid (blue). We can observe that the mesh quality is also improved with surface features preserved after pillowing and anisotropic GHO based smoothing.

7.5 Conclusions and Future Work

In this chapter, we have developed an algorithm to segment 3D images and generate tetrahedral and hexahedral meshes with good quality. Given the input 3D image, we first segment the image by using the HEWCVT-3D algorithm. The Dual Contouring method is then used to extract the initial tetrahedral and hexahedral meshes. To smooth out

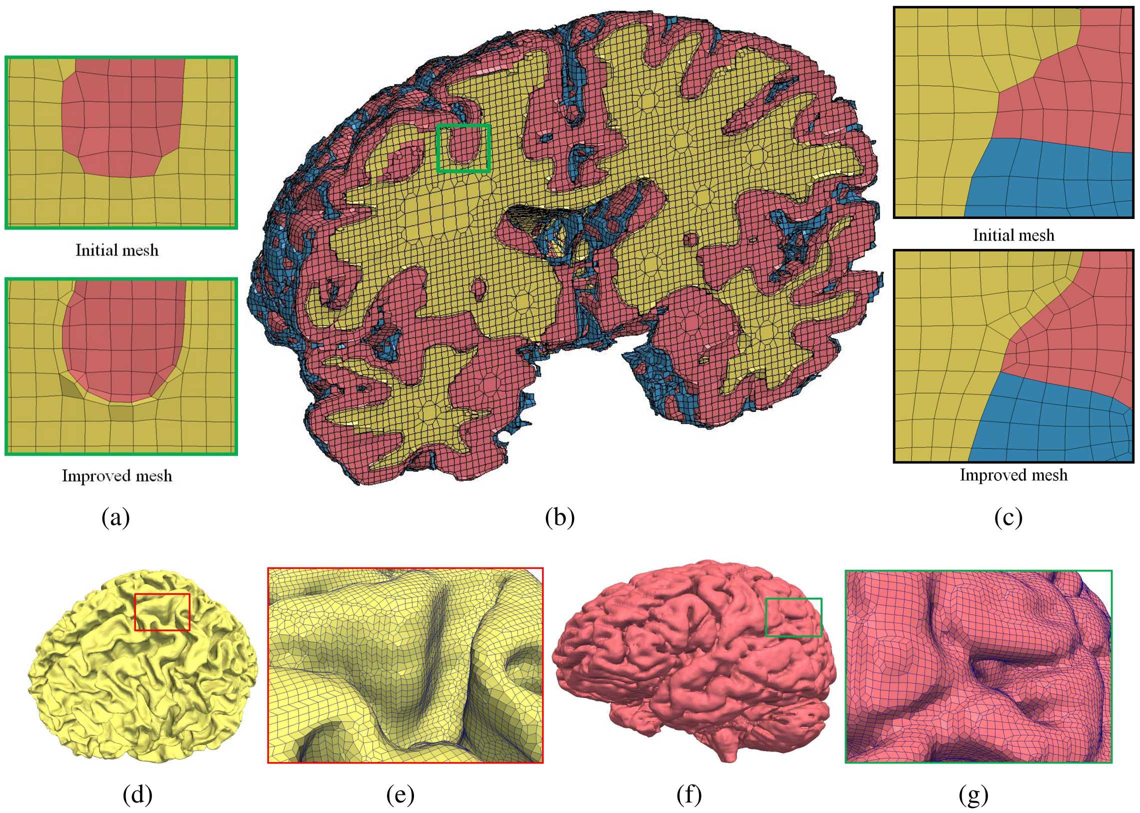


Figure 7.6: Hexahedral mesh of the Brain-6 model. (a, b) Cross section of the final hexahedral mesh, with zoom-in pictures of the initial and improved meshes of the green box; (c) zoom-in pictures of the initial and improved meshes with three neighboring materials; (d) improved hexahedral mesh of the white matter; (e) enlargement of the red window in (d); (f) improved hexahedral mesh of the gray matter; and (g) enlargement of the green window in (f).

Table 7.2: Tetrahedral mesh statistics of all tested models.

Image	Mesh size (vertices, elements)	Joe-Liu (min, max)	Dihedral angle (min, max)	Time (seconds)
Brain-1	(368,584, 1,796,748)	(0.12, 1.0)	$(15.14°, 166.56°)$	189.7
Brain-2	(326,576, 1,616,551)	(0.13, 1.0)	$(15.10°, 166.94°)$	187.8
Brain-3	(332,681, 1,630,137)	(0.12, 1.0)	$(15.11°, 167.39°)$	184.7
Brain-4	(343,268, 1,682,014)	(0.12, 1.0)	$(15.06°, 167.39°)$	186.9
Brain-5	(301,298, 1,491,425)	(0.13, 1.0)	$(15.08°, 167.76°)$	179.9
Brain-6	(282,352, 1,395,796)	(0.11, 1.0)	$(15.11°, 167.17°)$	179.6
Brain-7	(312,453, 1,534,144)	(0.12, 1.0)	$(15.02°, 167.83°)$	188.8
Brain-8	(342,683, 1,686,006)	(0.13, 1.0)	$(15.01°, 167.89°)$	187.7

Table 7.3: Hexahedral mesh statistics of all tested models.

Image	Mesh size (vertices, elements)	Scaled Jacobian (min, max)	Condition number (min, max)	Time (seconds)
Brain-1	(317,986, 256,956)	(0.10, 1.0)	(1.0, 386.8)	263.9
Brain-2	(350,768, 283,446)	(0.09, 1.0)	(1.0, 422.3)	288.2
Brain-3	(317,988, 256,956)	(0.08, 1.0)	(1.0, 408.6)	266.4
Brain-4	(370,438, 299,342)	(0.08, 1.0)	(1.0, 446.5)	311.6
Brain-5	(357,324, 288,746)	(0.09, 1.0)	(1.0, 431.9)	296.8
Brain-6	(295,826, 239,049)	(0.10, 1.0)	(1.0, 358.8)	244.1
Brain-7	(363,882, 294,046)	(0.09, 1.0)	(1.0, 440.8)	302.6
Brain-8	(340,934, 275,499)	(0.08, 1.0)	(1.0, 412.6)	287.9

surface noise and improve the quality of the tetrahedral/hexahedral mesh, we developed an anisotropic GHO diffusion flow. The quality of the interior tetrahedron/hexahedron is also improved via various optimization techniques. We have successfully tested our method using several volumetric imaging datasets. In the future, we will investigate more anisotropic schemes for the geometric flow method. We will also parallelize our algorithms and apply to more real applications.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In this thesis, we develop several CVT-based algorithms for image and mesh segmentation. Based on the image segmentation results, we can generate adaptive superpixels in 2D and quality tetrahedral/hexahedral meshes in 3D using the geometric flow method. We also generate polycubes and all-hex meshes with good quality. Furthermore, we combine the CVT with geometric operators for surface parameterization and mesh quality improvement, with concave/convex features preserved.

To overcome the sensitivity to the seed point initialization and noise, and improve the accuracy and stability of clustering results, we develop the HEWCVT based model for image segmentation which introduces a harmonic form of the clustering energy by combining the image intensity with cluster boundary information. As an application, we apply the HEWCVT algorithm locally to generate adaptive superpixels with the help of quadtree decomposition and density function. We also extend the HEWCVT algorithm to 3D image segmentation, where 3D spatial information is included to eliminate the noise effect. To smooth out surface noise and improve the quality of the tetrahedral/hexahedral mesh extracted from the segmented image, we develop an anisotropic GHO diffusion flow method which preserves surface features while removing surface noise with an anisotropic weighting function.

We then develop two CVT based surface segmentation algorithms for polycube construction and all-hex mesh generation. The HBECVT method is robust to the initialization and reduces non-monotone boundaries by including local neighbouring information in the energy function. As a follow up, our two-step surface segmentation scheme couples the HBECVT model with SLO eigenfunctions and curve skeleton information, which is suitable for slim cylindrical objects and can reduce unnecessary singularities with compact polycube structures. Surface segmentation results are revised using predefined geometric constraints for valid polycube construction. After that, we generate uniform and adaptive all-hex meshes as well as volumetric T-meshes via parametric mapping, which can be used for both FEA and IGA applications.

Furthermore, we develop a feature-aligned surface parameterization algorithm to generate quality surface quadrangulations and T-meshes. The SLO eigenfunctions over triangle meshes are utilized for the CVT based surface segmentation, where the L_∞ norm is used as the distance measurement. Boundaries of the segmented regions are used to define guidance triangles and their guidance directions. The constrained cross field method parameterizes the surface with all feature lines aligned to the parametric lines. Based on the surface parameterization results, truncated and weighted T-splines are generated for IGA applications.

8.2 Future Work

While this thesis has demonstrated the potential of using CVT-based methods for image and mesh segmentation with various applications, many opportunities for extending the scope of this thesis remain. This section presents some of these directions.

Unsupervised Image Segmentation. One goal for image segmentation is to make this process as easy as possible for the user by reducing the amount of supervision required. However, the CVT-based image segmentation methods need to predefine the expected number of clusters, which creates a problem for large data processing. One possible solution is to develop robust unsupervised clustering algorithms which decide the clustering criteria by themselves. Advanced machine learning techniques are also worthy of additional exploration for unsupervised image segmentation tasks, which utilize large training datasets and a much more contemporary machine learning approach.

Valid Polycube Construction. In this thesis, we investigate CVT-based surface segmentation for polycube construction under specific geometric constraints. The segmentation results can be affected by the initialization of generators, which may lead to incorrect polycube structures. Similarly as *k*-means methods, CVT-based clustering methods may settle at local minima. There is no guarantee that the global optimum is found using these algorithms. It would be interesting to study other surface segmentation techniques, such as graph-based methods and hierarchical clustering techniques, for valid polycube construction. Moreover, the geometric and topological constraints used in this thesis are sufficient but not necessary conditions for polycube construction, further study is still required.

T-splines Construction for Isogeometric Analysis. Isogeometric analysis using T-splines has been studied in different research fields. However, it remains a challenging problem to automatically construct T-spline models with good quality for complicated geometry and topology. In this thesis, we develop several algorithms to generate hexahedral and quadrilateral meshes, and use them to generate T-meshes and analysis-suitable T-splines for IGA applications. One interesting direction in the future is to develop an integrated modelling and analysis platform for the T-spline based isogeometric analysis.

Bibliography

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

[2] M. N. Ahmed, S. M. Yamany, N. Mohamed, A. A. Farag, and T. Moriarty. A modified fuzzy *c*-means algorithm for bias field estimation and segmentation of MRI data. *IEEE Transactions on Medical Imaging*, 21(3):193–199, 2002.

[3] P. Alliez, É. C. de Verdière, O. Devillers, and M. Isenburg. Centroidal Voronoi diagrams for isotropic surface remeshing. *Graphical Models*, 67(3):204–231, 2005.

[4] A. Z. Arifin and A. Asano. Image segmentation by histogram thresholding using hierarchical cluster analysis. *Pattern Recognition Letters*, 27(13):1515–1521, 2006.

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

[6] C. L. Bajaj and G. Xu. Anisotropic diffusion of surfaces and functions on surfaces. *ACM Transactions on Graphics*, 22(1):4–32, 2003.

[7] T. D. Blacker and M. B. Stephenson. Paving: a new approach to automated quadrilateral mesh generation. *International Journal for Numerical Methods in Engineering*, 32(4):811–847, 1991.

[8] D. Bommes, B. Lévy, N. Pietroni, E. Puppo, C. Silva, M. Tarini, and D. Zorin. Quad-mesh generation and processing: A survey. *Computer Graphics Forum*, 32(6):51–76, 2013.

[9] D. Bommes, H. Zimmer, and L. Kobbelt. Mixed-integer quadrangulation. *ACM Transactions On Graphics*, 28(3):1–10, 2009.

[10] Y. Boykov and G. Funka-Lea. Graph cuts and efficient ND image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.

[11] D. Brunner and G. Brunnett. Mesh segmentation using the object skeleton graph. *International Conferenece on Computer Graphics and Imaging*, pages 48–55, 2004.

[12] W. Cai, S. Chen, and D. Zhang. Fast and robust fuzzy *c*-means clustering algorithms incorporating local information for image segmentation. *Pattern Recognition*, 40(3):825–838, 2007.

[13] M. Campen, C. T. Silva, and D. Zorin. Bijective maps from simplicial foliations. *ACM Transactions on Graphics*, 35(4):7:1–7:15, 2016.

[14] S. A. Canann, J. R. Tristano, and M. L. Staten. An approach to combined Laplacian and optimization-based smoothing for triangular, quadrilateral, and quad-dominant meshes. *7th International Meshing Roundtable*, pages 479–494, 1998.

[15] Y. Cao, L. Ju, Q. Zou, C. Qu, and S. Wang. A multichannel edge-weighted centroidal Voronoi tessellation algorithm for 3D super-alloy image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 17–24, 2011.

[16] T. F. Chan and L. A. Vese. Active contour and segmentation models using geometric PDEs for medical imaging. In *Geometric Methods in Bio-medical Image Processing*, pages 63–75. Springer, 2002.

[17] S. Chen and D. Zhang. Robust image segmentation using FCM with spatial constraints based on new kernel-induced distance measure. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(4):1907–1916, 2004.

[18] Y. T. Chen. A level set method based on the Bayesian risk for medical image segmentation. *Pattern Recognition*, 43(11):3699–3711, 2010.

[19] K. S. Chuang, H. L. Tzeng, S. Chen, J. Wu, and T. J. Chen. Fuzzy *c*-means clustering with spatial information for image segmentation. *Computerized Medical Imaging and Graphics*, 30(1):9–15, 2006.

[20] C. A. Cocosco, V. Kollokian, R. K. S. Kwan, G. B. Pike, and A. C. Evans. BrainWeb: online interface to a 3D MRI simulated brain database. *NeuroImage*, 5(4):S425, 1997.

[21] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM Transactions on Graphics*, 23(2):905–914, 2004.

[22] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[23] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool. SEEDS: superpixels extracted via energy-driven sampling. In *European Conference on Computer Vision*, pages 13–26, 2012.

[24] S. Dong, P. Bremer, M. Garland, V. Pascucci, and J. C. Hart. Spectral surface quadrangulation. *ACM Transactions on Graphics*, 25(3):1057–1066, 2006.

[25] Q. Du, V. Faber, and M. Gunzburger. Centroidal Voronoi tessellations: applications and algorithms. *SIAM Review*, 41(4):637–676, 1999.

[26] Q. Du, M. Gunzburger, and L. Ju. Advances in studies and applications of centroidal Voronoi tessellations. *Numerical Mathematics: Theory, Methods and Applications*, 3(2):119–142, 2010.

[27] Q. Du, M. Gunzburger, L. Ju, and X. Wang. Centroidal Voronoi tessellation algorithms for image compression, segmentation, and multichannel restoration. *Journal of Mathematical Imaging and Vision*, 24(2):177–194, 2006.

[28] Q. Du, M. D. Gunzburger, and L. Ju. Constrained centroidal Voronoi tessellations for surfaces. *SIAM Journal on Scientific Computing*, 24(5):1488–1506, 2003.

[29] J. Edgel. An adaptive grid-based all hexahedral meshing algorithm based on 2-refinement. *Master Thesis, Brigham Young University*, 2010.

[30] X. Fan, X. Wang, and S. Wang. A fuzzy edge-weighted centroidal Voronoi tessellation model for image segmentation. *Computers & Mathematics with Applications*, 2015. DOI: 10.1016/j.camwa.2015.11.003.

[31] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

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

[33] N. Folwell and S. Mitchell. Reliable whisker weaving via curve contraction. *Engineering With Computers*, 15(3):292–302, 1999.

[34] P. A. Foteinos and N. P. Chrisochoides. High quality real-time image-to-mesh conversion for finite element simulations. *Journal of Parallel and Distributed Computing*, 74(2):2123–2140, 2014.

[35] L. A. Freitag. On combining Laplacian and optimization-based mesh smoothing techniques. *AMD-Vol.220 Trends in Unstructured Mesh Generation*, pages 37–44, 1997.

[36] L. Fu, L. B. Kara, and K. Shimada. Modeling flow features with user-guided streamline parameterization. *Computer-Aided Design*, 46:263–268, 2014.

[37] K. Fujiwara. Eigenvalues of Laplacians on a closed Riemannian manifold and its nets. *Proceedings of the American Mathematical Society*, 123(8):2585–2594, 1995.

[38] A. Golovinskiy and T. Funkhouser. Randomized cuts for 3D mesh analysis. *ACM Tansactions on Graphics*, 27(5):145, 2008.

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

[40] J. A. Hartigan and M. A. Wong. Algorithm AS 136: a *k*-means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979.

[41] Y. He, H. Wang, C. W. Fu, and H. Qin. A divide-and-conquer approach for automatic polycube map construction. *Computers & Graphics*, 33(3):369–380, 2009.

[42] K. Hu, J. Qian, and Y. Zhang. Adaptive all-hexexahedral mesh generation based on a hybrid octree and bubble packing. *Research Notes of the 22th International Meshing Roundtable*, 2013.

[43] K. Hu and Y. Zhang. Extended edge-weighted centroidal Voronoi eessellation for image segmentation. *CompIMAGE (Computer Modeling of Objects Presented in Images: Fundamentals, Methods, and Applications)*, 8461:164–175, 2014.

[44] K. Hu and Y. Zhang. Surface segmentation and polycube construction based on generalized centroidal Voronoi tessellation. *Research Notes of the 24th International Meshing Roundtable*, 2015.

[45] K. Hu and Y. Zhang. Centroidal Voronoi tessellation based polycube construction for adaptive all-hexahedral mesh generation. *Computer Methods in Applied Mechanics and Engineering*, 305:405–421, 2016.

[46] K. Hu and Y. Zhang. Image segmentation and adaptive superpixel generation based on harmonic edge-weighted centroidal Voronoi tessellation. *The Special Issue of CompIMAGE'14 in Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, 4(2):46–60, 2016.

[47] K. Hu, Y. Zhang, X. Li, and G. Xu. Feature-aligned surface parameterization using secondary Laplace operator and Loop subdivision. *25th International Meshing Roundtable*, 2016.

[48] K. Hu, Y. Zhang, and T. Liao. Surface segmentation for polycube construction based on generalized centroidal Voronoi tessellation. *Computer Methods in Applied Mechanics and Engineering*, 2016. DOI: 10.1016/j.cma.2016.07.005.

[49] K. Hu, Y. Zhang, and G. Xu. CVT-based 3D image segmentation for quality tetrahedral meshing. *CompIMAGE (Computer Modeling of Objects Presented in Images: Fundamentals, Methods, and Applications)*, 2016.

[50] J. Huang, T. Jiang, Z. Shi, Y. Tong, H. Bao, and M. Desbrun. L1-based construction of polycube maps from complex shapes. *ACM Transactions on Graphics*, 33(3):25:1–25:11, 2014.

[51] E. Iarussi, D. Bommes, and A. Bousseau. Bendfields: Regularized curvature fields from rough concept sketches. *ACM Transactions on Graphics*, 34(3):24:1–24:17, 2015.

[52] Y. Ito, A. M. Shih, and B. K. Soni. Efficient hexahedral mesh generation for complex geometries using an improved set of refinement templates. *Proceedings of the 18th International Meshing Roundtable*, pages 103–115, 2009.

[53] L. Ju. Conforming centroidal Voronoi Delaunay triangulation for quality mesh generation. *International Journal of Numerical Analysis and Modeling*, 4:531–547, 2007.

[54] L. Ju, Q. Du, and M. Gunzburger. Probabilistic methods for centroidal Voronoi tessellations and their parallel implementations. *Parallel Computing*, 28(10):1477–1500, 2002.

[55] B. Jüttler and A. Felis. Least-squares fitting of algebraic spline surfaces. *Advances in Computational Mathematics*, 17(1-2):135–152, 2002.

[56] F. Kälberer, M. Nieser, and K. Polthier. QuadCover-surface parameterization using branched coverings. *Computer Graphics Forum*, 26(3):375–384, 2007.

[57] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient *k*-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.

[58] J. Leng, Y. Zhang, and G. Xu. A novel geometric flow approach for quality improvement of multi-component tetrahedral meshes. *Computer-Aided Design*, 45(10):1182–1197, 2013.

[59] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. Turbopixels: fast superpixels using geometric flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2290–2297, 2009.

[60] B. Lévy. Constrained texture mapping for polygonal meshes. *ACM SIGGRAPH*, pages 417–424, 2001.

[61] B. Lévy. Laplace-Beltrami eigenfunctions towards an algorithm that understands geometry. *IEEE International Conference on Shape Modeling and Applications*, pages 13–21, 2006.

[62] C. Li, C. Xu, C. Gui, and M. D. Fox. Distance regularized level set evolution and its application to image segmentation. *IEEE Transactions on Image Processing*, 19(12):3243–3254, 2010.

[63] Q. Li, N. Mitianoudis, and T. Stathaki. Spatial kernel *k*-harmonic means clustering for multi-spectral image segmentation. *IET Image Processing*, 1(2):156–167, 2007.

[64] X. Li, G. Xu, and Y. Zhang. Localized discrete Laplace-Beltrami operator over triangular mesh. *Computer Aided Geometric Design*, 39:67–82, 2015.

[65] 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.

[66] 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, 2012.

[67] T. Liao, X. Li, G. Xu, and Y. Zhang. Secondary Laplace operator and generalized Giaquinta-Hildebrandt operator with applications on surface segmentation and smoothing. *A Special Issue of SIAM Conference on Geometric & Physical Modeling 2015 in Computer Aided Design*, 70:56–66, 2016.

[68] T. Liao, G. Xu, and Y. Zhang. Structure-aligned guidance estimation in surface parameterization using eigenfunction-based cross field. *Graphical Models*, 76(6):691–705, 2014.

[69] J. Lin, X. Jin, Z. Fan, and C. C. Wang. Automatic polycube-maps. In *Advances in Geometric Modeling and Processing*, pages 3–16. Springer Berlin Heidelberg, 2008.

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

[71] L. Liu, Y. Zhang, Y. Liu, and W. Wang. Feature-preserving T-mesh construction using skeleton-based polycubes. *Computer-Aided Design*, 58:162–172, 2015.

[72] L. Liu, Y. J. 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.

[73] R. Liu and H. Zhang. Segmentation of 3D meshes through spectral clustering. In *Pacific Conference on Computer Graphics and Applications*, pages 298–305, 2004.

[74] M. Livesu, N. Vining, A. Sheffer, J. Gregson, and R. Scateni. Polycut: monotone graph-cuts for polycube base-complex construction. *ACM Transactions on Graphics*, 32(6):1–12, 2013.

[75] L. Lu, B. Levy, and W. Wang. Centroidal Voronoi tessellations for line segments and graphs. Technical report, INRIA-ALICE, 2009.

[76] W. Y. Ma and B. S. Manjunath. Edgeflow: a technique for boundary detection and image segmentation. *IEEE Transactions on Image Processing*, 9(8):1375–1388, 2000.

[77] R. Maini and H. Aggarwal. Study and comparison of various image edge detection techniques. *International Journal of Image Processing*, 3(1):1–11, 2009.

[78] A. P. Mangan and R. T. Whitaker. Partitioning 3D surface meshes using watershed segmentation. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):308–321, 1999.

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

[80] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Eighth IEEE International Conference on Computer Vision*, volume 2, pages 416–423, 2001.

[81] U. F. Mayer. Numerical solutions for the surface diffusion flow in three space dimensions. *Computational and Applied Mathematics*, 20(3):361–379, 2001.

[82] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. *Visualization and mathematics III*, pages 35–57, 2003.

[83] A. P. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones. Superpixel lattices. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[84] M. Nieser, U. Reitebuch, and K. Polthier. CUBECOVER–Parameterization of 3D Volumes. *Computer Graphics Forum*, 30(5):1397–1406, 2011.

[85] R. Nock and F. Nielsen. On weighting clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1223–1235, 2006.

[86] S. Owen. A survey of unstructured mesh generation technology. *Proceedings of 7th International Meshing Roundtable*, pages 239–267, 1998.

[87] S. Owen and T. Shelton. Evaluation of grid-based hex meshes for solid mechanics. *Engineering with Computers*, pages 1–15, 2014.

[88] S. Owen and J. Shepherd. Embedding features in a cartesian grid. *Proceedings of the 18th International Meshing Roundtable*, pages 117–138, 2009.

[89] N. R. Pal and S. K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294, 1993.

[90] H. Pan, Y. Liu, A. Sheffer, N. Vining, C. Li, and W. Wang. Flow aligned surfacing of curve networks. *ACM Transactions on Graphics*, 34(3):127:1–127:10, 2015.

[91] Q. Pan, G. Xu, G. Xu, and Y. Zhang. Isogeometric analysis based on extended Loop's subdivision. *Journal of Computational Physics*, 299:731–746, 2015.

[92] T. N. Pappas. An adaptive clustering algorithm for image segmentation. *IEEE Transactions on Signal Processing*, 40(4):901–914, 1992.

[93] N. Paragios and R. Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *International Journal of Computer Vision*, 46(3):223–247, 2002.

[94] M. Parrish, M. Borden, M. Staten, and S. Benzley. A selective approach to conformal refinement of unstructured hexahedral finite element meshes. *Proceedings of the 16th International Meshing Roundtable*, pages 251–268, 2008.

[95] W. Pedrycz and J. Waletzky. Fuzzy clustering with partial supervision. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 27(5):787–795, 1997.

[96] M. J. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, 1964.

[97] M. Price and C. 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.

[98] M. Price, C. Armstrong, and M. 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.

[99] J. Qian and Y. Zhang. Sharp feature preservation in octree-based hexahedral mesh generation for CAD assembly models. *Proceedings of 19th International Meshing Roundtable*, pages 243–262, 2010.

[100] 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.

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

[102] N. Ray, W. C. Li, B. Lévy, A. Sheffer, and P. Alliez. Periodic global parameterization. *ACM Transactions on Graphics*, 25(4):1460–1485, 2006.

[103] X. Ren and J. Malik. Learning a classification model for segmentation. In *Ninth IEEE International Conference on Computer Vision*, pages 10–17, 2003.

[104] M. Reuter. Hierarchical shape segmentation and registration via topological features of Laplace-Beltrami eigenfunctions. *International Journal of Computer Vision*, 89(2-3):287–308, 2010.

[105] M. Reuter, S. Biasotti, D. Giorgi, G. Patané, and M. Spagnuolo. Discrete Laplace-Beltrami operators for shape analysis and segmentation. *Computers & Graphics*, 33(3):381–390, 2009.

[106] G. Rong, M. Jin, L. Shuai, and X. Guo. Centroidal Voronoi tessellation in universal covering space of manifold surfaces. *Computer Aided Geometric Design*, 28(8):475–496, 2011.

[107] G. Rong, Y. Liu, W. Wang, X. Yin, X. Gu, and X. Guo. GPU-assisted computation of centroidal Voronoi tessellation. *IEEE Transactions on Visualization and Computer Graphics*, 17(3):345–356, 2011.

[108] R. Schneiders. A grid-based algorithm for the generation of hexahedral element meshes. *Engineering with Computers*, 12(3-4):168–177, 1996.

[109] R. Schneiders. An algorithm for the generation of hexahedral element meshes based on an octree technique. *6th International Meshing Roundtable*, pages 195–196, 1997.

[110] R. Schneiders and R. Bünten. Automatic generation of hexahedral finite element meshes. *Computer Aided Geometric Design*, 12(7):693–707, 1995.

[111] J. Schóberl. NETGEN An advancing front 2D/3D-mesh generator based on abstract rules. *Computing and Visualization in Science*, 1(1):41–52, 1997.

[112] N. Senthilkumaran and R. Rajesh. Edge detection techniques for image segmentationa survey of soft computing approaches. *International Journal of Recent Trends in Engineering*, 1(2):250–254, 2009.

[113] L. Shapira, A. Shamir, and D. Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*, 24(4):249–259, 2008.

[114] M. S. Shephard and M. K. Georges. Automatic three-dimensional mesh generation by the finite octree technique. *International Journal for Numerical methods in engineering*, 32(4):709–749, 1991.

[115] J. F. Shepherd. Conforming hexahedral mesh generation via geometric capture methods. *Proceedings of the 18th International Meshing Roundtable*, pages 85–102, 2009.

[116] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[117] H. Si. TetGen, a Delaunay-based quality tetrahedral mesh generator. *ACM Transactions on Mathematical Software*, 41(2):11:1–11:36, 2015.

[118] J. Stam. Fast evaluation of Loop triangular subdivision surfaces at arbitrary parameter values. *SIGGRAPH 98 Proceedings, CD-ROM supplement, Orlando*, 1998.

[119] M. Staten, R. Kerr, S. Owen, and T. Blacker. Unconstrained paving and plastering: progress update. *Proceedings of 15th International Meshing Roundtable*, pages 469–486, 2006.

[120] A. Tagliasacchi, I. Alhashim, M. Olson, and H. Zhang. Mean curvature skeletons. *Computer Graphics Forum*, 31(5):1735–1744, 2012.

[121] M. Tarini, K. Hormann, P. Cignoni, and C. Montani. Polycube-maps. *ACM Transactions on Graphics*, 23(3):853–860, 2004.

[122] R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Class prediction by nearest shrunken centroids, with applications to DNA microarrays. *Statistical Science*, pages 104–117, 2003.

[123] O. J. Tobias and R. Seara. Image segmentation by histogram thresholding using fuzzy sets. *IEEE Transactions on Image Processing*, 11(12):1457–1465, 2002.

[124] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun. Designing quadrangulations with discrete harmonic forms. *Symposium on Geometry Processing*, pages 201–210, 2006.

[125] F. Usai, M. Livesu, E. Puppo, M. Tarini, and R. Scateni. Extraction of the quad layout of a triangle mesh guided by its curve skeleton. *ACM Transactions on Graphics*, 35(1):6:1–6:13, 2015.

[126] S. Valette, J. M. Chassery, and R. Prost. Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):369–381, 2008.

[127] O. Veksler, Y. Boykov, and P. Mehrani. Superpixels and supervoxels in an energy optimization framework. In *Computer VisionECCV 2010*, pages 211–224, 2010.

[128] L. A. Vese and T. F. Chan. A multiphase level set framework for image segmentation using the Mumford and Shah model. *International Journal of Computer Vision*, 50(3):271–293, 2002.

[129] 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.

[130] H. Wang, T. Lu, O. Au, and C. Tai. Spectral 3D mesh segmentation with a novel single segmentation field. *Graphical Models*, 76(5):440–456, 2014.

[131] J. Wang, L. Ju, and X. Wang. An edge-weighted centroidal Voronoi tessellation model for image segmentation. *IEEE Transactions on Image Processing*, 18(8):1844–1858, 2009.

[132] J. Wang and X. Wang. VCells: simple and efficient superpixels using edge-weighted centroidal Voronoi tessellations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1241–1247, 2012.

[133] L. Wang, X. Gu, K. Mueller, and S. Yau. Uniform texture synthesis and texture mapping using global parameterization. *The Visual Computer*, 21(8-10):801–810, 2005.

[134] W. Wang, Y. Zhang, L. Liu, and T. J. Hughes. Trivariate solid T-spline construction from boundary triangulations with arbitrary genus topology. *Computer-Aided Design*, 45(2):351–360, 2013.

[135] X. Wang, X. Ying, Y. Liu, S. Xin, W. Wang, X. Gu, W. Mueller-Wittig, and Y. He. Intrinsic computation of centroidal Voronoi tessellation (CVT) on meshes. *Computer-Aided Design*, 58:51–61, 2015.

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

[137] X. Wei, Y. Zhang, L. Liu, and T. Hughes. Truncated T-splines: fundamentals and methods. *Computer Methods in Applied Mechanics and Engineering, accepted*, 2016.

[138] G. Xu. Discrete Laplace-Beltrami operators and their convergence. *Computer Aided Geometric Design*, 21(8):767–784, 2004.

[139] G. Xu. Consistent approximations of several geometric differential operators and their convergence. *Applied Numerical Mathematics*, 69:1–12, 2013.

[140] G. Xu, Q. Pan, and C. Bajaj. Discrete surface modelling using partial differential equations. *Computer Aided Geometric Design*, 23(2):125–145, 2006.

[141] G. Xu and Q. Zhang. A general framework for surface modeling using geometric partial differential equations. *Computer Aided Geometric Design*, 25(3):181–202, 2008.

[142] Y. Xu, R. Chen, C. Gotsman, and L. Liu. Embedding a triangular graph within a given boundary. *Computer Aided Geometric Design*, 28(6):349–356, 2011.

[143] D. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. *Computer graphics forum*, 28(5):1445–1454, 2009.

[144] D. Yan, W. Wang, B. Lévy, and Y. Liu. Efficient computation of clipped Voronoi diagram for mesh generation. *Computer-Aided Design*, 45(4):843–852, 2013.

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

[146] H. Zhang and G. Zhao. Adaptive hexahedral mesh generation based on local domain curvature and thickness using a modified grid-based method. *Finite Elements in Analysis and Design*, 43(9):691–704, 2007.

[147] J. Zhang, J. Zheng, C. Wu, and J. Cai. Variational mesh decomposition. *ACM Transactions on Graphics*, 31(3):21:1–21:14, 2012.

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

[149] Y. Zhang, C. Bajaj, and B.-S. Sohn. 3D finite element meshing from imaging data. *Computer Methods in Applied Mechanics and Engineering*, 194:5083–5106, 2005.

[150] Y. Zhang, C. Bajaj, and G. Xu. Surface smoothing and quality improvement of quadrilateral/hexahedral meshes with geometric flow. *Communications in Numerical Methods in Engineering*, 25(1):1–18, 2009.

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

[152] Y. Zhang, T. Hughes, and C. Bajaj. An automatic 3D mesh generation method for domains with multiple materials. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):405–415, 2010.

[153] Y. Zhang and J. Qian. Resolving topology ambiguity for multiple-material domains. *Computer Methods in Applied Mechanics and Engineering*, 247:166–178, 2012.

[154] Y. Zhuang, M. Zou, N. Carr, and T. Ju. Anisotropic geodesics for live-wire mesh segmentation. *Computer Graphics Forum*, 33(7):111–120, 2014.