Harmonic fields based surface and volume mapping

Xia, Jiazhi

2011

Xia, J. Z. (2011). Harmonic fields based surface and volume mapping. Doctoral thesis, Nanyang Technological University, Singapore.

https://hdl.handle.net/10356/45771

https://doi.org/10.32657/10356/45771

Harmonic Fields Based Surface and Volume Mapping

A Thesis Presented
by

Jiazhi Xia

Submitted to

The Nanyang Technological University

in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

in

School of Computer Engineering

Nov. 24, 2010

Acknowledgments

This thesis could not be presented in this form without the help and support from so many people who are gratefully acknowledged here.

At the very first, I would like to express my deepest gratitude to my supervisor, Dr. Ying He, for leading me into this research field, for his excellent guidance, for his encouragement during my postgraduate career, and for the days and nights we have worked together.

I would like to thank Dr. Philip Fu for his helpful instruction, discussions and advice during my study.

Special thanks should go to Dr. Lin Juncong, Dr. Chen Xiaoming, Mr. Han Shuchu, Mr. Ismael García Fernández, Dr. Zhang Long, Ms. Sun Qian and Ms. Quynh. I really enjoy the collaboration with them and sincerely thank for the consistent help in every joint project.

I am also grateful to my friends Mr. William Lai, Mr. Ying Xiang, Dr. Qiu Jie, and Dr. Liu Dongquan.

Last, but not least, deepest appreciation is given to my family for their support. Their love and sacrifice made it possible for me to achieve this work.

Contents

List of Figures

List of Tables

Abstract

Harmonic fields are widely used in computational science and engineering for their computational efficiency and promising properties. Radó theorem strongly supports the application of harmonic fields in parameterization by proving that harmonic maps from 2D Riemannian manifold to convex planar domain are diffeomorphism. But the limitations of harmonic maps are also obvious: first, there are a lot of real applications requiring the mapping constraints to be set inside the domain. However, in such cases, harmonic maps could not be guaranteed to be one-to-one. Second, Radó theorem holds true only in the 2D case. Volumetric harmonic maps could not be guaranteed to be a diffeomorphism.

In this thesis, we systematically study the harmonic fields and their applications in surface and volume parameterization by overcoming the two aforementioned drawbacks technically.

In the first part of the thesis, we present an editable polycube map framework in Chapter 4, that, given an arbitrary high-resolution polygonal mesh and a simple polycube representation plus optional sketched features indicating relevant correspondences between the two, provides a uniform, regular and artist-controllable quads-only mesh with a parameterized subdivision displacement scheme. The proposed method is based on a divide and conquer strategy. The mesh surface is divided into patches according to the feature constraints. A diffeomorphism is built between each pair of topological disk patches. After that, a global smoothing step is adopted to improve the continuity along the segmentation boundaries. In Chapter 5, we also develop another method to overcome the drawback of harmonic maps in parameterizing 3D facial expressions. With the salient features (such as the eyes, mouth and nose) in the captured expression,

we first compute a geodesic mask with the user-specified radius to segment the facial expression. Then we cut the 3D faces along a geodesic curve connecting the eyes, nose and mouth. As a result, each 3D face is topologically equivalent to an annulus. Next, we solve a harmonic function using the Dirichlet boundary condition. Finally, we compute the consistent mapping among all the captured frames by tracing the integral curves that follow the gradient field of the harmonic function. Observing that both the geodesic and harmonic functions are intrinsic and independent of the embedding, the proposed method is invariant to the expression, and also guarantees the exact correspondences of the salient features.

In the second part of the thesis, we systematically study volume parameterization. In Chapter 6, we investigate star-shaped volumes and prove that the Green's function on the star shape has a unique critical point and all level sets inside the star shape are topological spheres. Then we show that the Green's function can induce a diffeomorphism between two star-shaped volumes. Next, we develop an algorithm to parameterize general volumes by using star-shaped decomposition. The computation of the Green's function is based on harmonic fields. In Chapter 7, we present an algorithm to parameterize handlebody with a boundary surface embedded in 3D space. The intuition is to decompose the volume as the direct product of a two-dimensional surface and a one-dimensional curve. We first partition the boundary surface into ceiling, floor and walls. Then we compute the harmonic field in the volume with the Dirichlet boundary condition. By tracing the integral curve along the gradient of the harmonic function, we can parameterize the volume to the parametric domain. This method is guaranteed to produce bijection for handlebodies with complex topology, including topological balls as a degenerate case. Furthermore, the parameterization is regular everywhere. We compare this method with the star-shaped decomposition method and other existing techniques.

Based on our theoretical findings, we apply our algorithms to several real-world applications. Through the experiments, we demonstrate that the proposed surface and volume parameterization methods are effective and powerful for general surface and solid modeling and processing.

Chapter 1

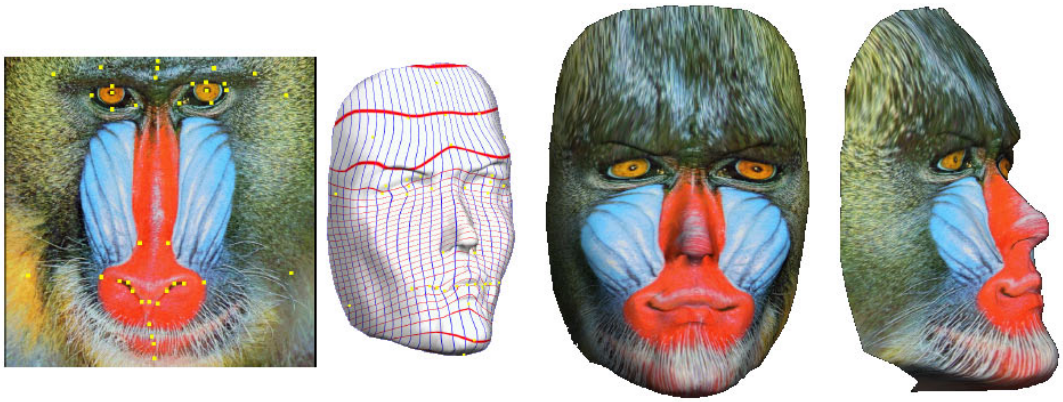
Introduction

This thesis is motivated by two different but highly related research fields: harmonic fields and shape parameterization.

Harmonic field is a function which defines a scalar- or vector- valued field. Harmonic fields can be constructed as solutions to Laplacian equations with certain boundary conditions. Due to their computational efficiency and promising properties such as smoothness and concentration of local extrema only at the boundaries, harmonic fields are widely used in a variety of applications in computational science and engineering [4, 13, 14, 15, 39, 76].

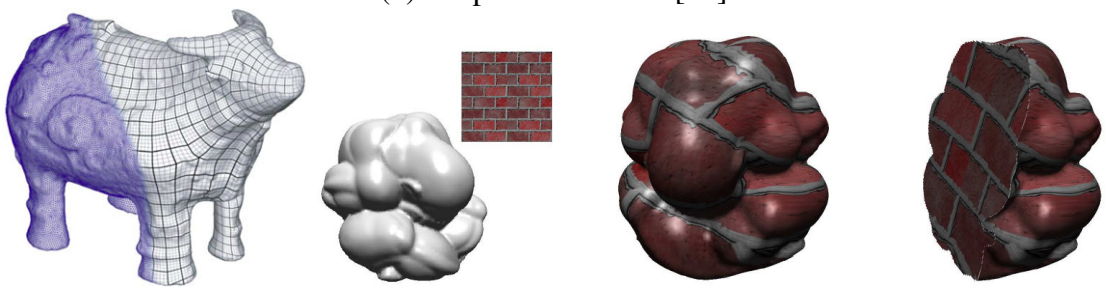
On the other hand, surface and volume mappings or parameterizations are fundamental tools in geometry processing and have been widely used in a wide range of applications in computer-aided design, scientific computation, graphics, etc. The goal of surface and volume parameterizations is to find a bijective map between the source object (surface or volume) and a parametric domain with the same topology. We briefly discuss some applications of shape parameterization below.

- Texture mapping: texture mapping(Fig. 1 (a)) is an efficient representation of a detailed model [46, 5, 51]. In traditional surface texture mapping, the color



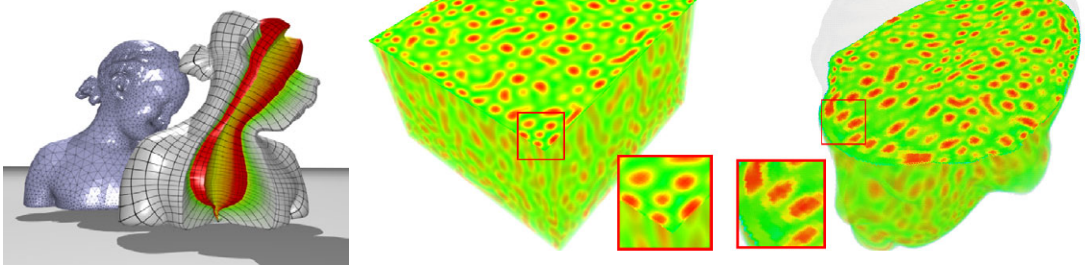
(a) Texture mapping [46]

(b) Shape deformation [83]



(c)Surface remeshing [62]

(d)Solid texture [47]



(e)Volume remeshing [52]

(f)Volume computing

Figure 1.1: Applications of surface and volume mapping.

details stored in a 2D array are mapped onto a polygonal mesh. The natural way to map details to surfaces is using planar parameterization. Later, more details such as normal, displacement and bump are presented as textures and mapped to the polygonal mesh. With the progress of graphics hardware, volume texture mapping is also achievable in nowadays applications [73].

- Shape morphing and detail transfer: A mapping between the surface or volume of two objects allows the transfer of details from one object to another [17, 83] (Fig. 1 (b)). The interpolation between the shape and appearance of several objects is also supported by the mapping. Morphing animations could be produced by varying the interpolation ratios over time. The map could be constructed either directly between the two objects or by parameterizing the two objects (surface or volume) to a common domain. In specific applications, the mapping is constructed with feature or skeleton correspondence constraints to achieve a more natural morphing or transfer.

- Remeshing: There are many possible mesh structures that represent the same surface or volume with similar levels of accuracy(Fig. 1 (c)(e)). Some mesh representations might be more desirable than others in different applications. For example, for numerical simulations on surfaces, well-shaped triangles that are not too small or too skinny are important for convergence of the algorithm and numerical accuracy. Surface parameterization is one common way to remesh surfaces [61, 62]. New mesh is easily constructed in well-understood and highly desirable domains such as the planar domain and then mapped back to the original surface. It is also highly desirable to represent the volume by hexahedral mesh in FEA(Finite Element Analysis) [47]. Similarly, the hexahedral mesh could be constructed in the parametric domain, such as polycube, and then mapped back to the original volume.

- GPU-based computing: Due to the rapid advances in graphics hardware, GPU (Graphics Processing Unit) becomes much more powerful than CPU (Central Processing Unit). It is highly desirable to use GPU for general computing (Fig. 1 (f)). To be compatible with the current GPU, data should be packed as image textures or volume textures. Surfaces and volumes are parameterized to a regular domain and rasterized to image or volume textures respectively.

1.1 Problem Statements

According to Radó theorem, harmonic fields are proven to guarantee a diffeomorphism from a 2D manifold to a convex planar domain with certain boundary conditions (more details can be found in Section 2.2). Radó theorem strongly supports the application of harmonic fields in surface parameterization. Tremendous harmonic fields-based surface and volume parameterization methods have been proposed [15, 87, 80, 47, 52]. However, there are also drawbacks of harmonic fields: first, there are a lot of real applications requiring the mapping constraints to be set inside the domain. However, in such cases, harmonic maps could not be guaranteed to be one-to-one. Second, Radó theorem holds true only in 2D cases. Volumetric harmonic maps could not be guaranteed to be a diffeomorphism.

In this thesis, we systematically study the harmonic fields-based surface and volume parameterizations aiming at high quality parameterization methods in different practical applications. The requirements of shape parameterization are application- dependent and vary from case to case. This section lists several common issues in shape parameterization.

- Distortion: The ideal shape parameterization is isometric, i.e., with no metric distortion. Unfortunately, isometric parameterization does not exist for most of

the real-world shapes. Thus, it is desirable to find a good parameterization with small angle and/or area distortion.

- Parametric domain: The bijective property of shape parameterization requires that the parametric domain must have the same topology as the input shape. However it is well known that the parameterization quality greatly depends on the geometry of the parametric domain. Taking genus-0 open surface for example, the parametric domain can be a rectangle, a free-boundary topological disc, or even a 3D polycube. The choice of parametric domain is usually application-dependent.

- Feature correspondence: Parameterizations often require some correspondences between the input shape and the parametric domain. These correspondences can be either specified by the users or determined automatically. The desired parameterization algorithm should guarantee the exact feature correspondences.

1.1.1 Surface Parameterization

Surface parameterization was initially introduced into computer graphics for mapping textures onto surfaces [5, 51]. It aims at building a bijective map between two surfaces with the same topology. Since the geometry of the two surfaces is different, almost every surface parameterization introduces distortion in either angles or areas. The existing surface parameterization approaches have contributed to minimizing these distortions to some extent.

Despite the great success achieved in surface parameterization, there are still limitations in the existing work. From the user's point of view, an ideal surface mapping algorithm should have at least the following features:

- Quality: The map is a bijection with low angle and area distortion.
- Feature correspondence: the feature correspondence between the source surface and the parametric domain should be guaranteed. Feature correspondence is important in some applications including shape morphing/animation, texture editing, and so on.
- User control: the user can easily control the mapping by specifying optional features on the source surface and their desired locations on the parametric domain. Both the parameterization and the optional features are editable.

There are numerous harmonic fields-based surface parameterization methods [15, 87]. However, none of the existing work has all the desired features. Direct harmonic maps only work well with constraints on the boundary. But a lot of applications require a more flexible constraints setting method, such as constraints inside the domain. A controllable, intuitive and effective surface parameterization method with high quality mapping results remains a challenge.

1.1.2 Volume Parameterization

Despite the great success in surface parametrization, most real-world objects are in fact volumes rather than surfaces. Rich sources of volumetric data are captured by current volume scanners. Volume parameterization, which is fundamental for volume data processing, is strongly called for by scientific and graphical applications. From the analysis and processing point of view, an ideal volume parameterization should have the following properties:

- The constructed mapping is a diffeomorphism. It means that the mapping should be bijective and smooth.

- The distortion of the mapping is low.
- The algorithm is able to parameterize volumes with complex topology such as shapes with high genus.

However, due to the intrinsic differences between surfaces and volumes, many classical results on surface parameterization cannot be directly generalized to produce volume parameterization. For example, it is well known that a harmonic map between a topological disk (a genus zero surface with a single boundary) and a planar convex domain is diffeomorphic (i.e., bijective and smooth), if the boundary map is homeomorphic (i.e., bijective and continuous). This result plays an important role in surface parameterization. However, the same result does not hold true for volumetric cases, i.e., a volumetric harmonic map cannot guarantee that the resulting parameterization will be bijective even though the target domain is convex. It remains both unclear and challenging as to how existing surface parameterization methods can be generalized from surfaces to volumes.

1.2 Contributions

In this thesis, we systematically study the surface and volume parameterization. For surface parameterization, we present an editable polycube map framework through which the user could control the mapping in an intuitive and effective manner(Chapter 3) . We also propose an expression-invariant method to parameterize 3D facial expressions. Our method is guaranteed to map the salient features, such as the eyes, nose and mouth, consistently among all frames (Chapter 4). For volume parameterization, we propose a star-shaped volume parameterization method and then extend it to general shapes (Chapter 5). We further develop an algorithm to parameterize handlebodies embedded in $\mathbb{R}^3$ (Chapter 6).

Our specific contributions include:

- **User Controllable Polycube Map** We present a method that efficiently and intuitively creates a high quality and artist controllable polycube map from a general mesh. Our method allows the user to modify the map easily and fine-tune the mapping. The user is also able to control the number of patches in the base mesh of the subdivision scheme by the construction of the base polycube. The provided polycube does not need to resemble the shape of the object accurately. In fact, coarse polycubes will suffice (Chapter 4).

- **Expressions Parameterization** We develop a geometry video framework to model 3D facial expressions. Our framework consists of a set of algorithms, including hole filling, geodesic-based face segmentation and expression-invariant parameterization. Our algorithms are efficient and robust, and can guarantee the exact correspondences of the salient features (eyes, mouth and nose) among frames (Chapter 5)

- **Star-shaped Volume Parameterization** We prove that the Green's function on the star shape has a unique critical point and all level sets inside the star shape are topological spheres. Then we prove that the Green's function can induce a diffeomorphism between two star shapes. To our knowledge, this is the first constructive proof of the existence of a diffeomorphism between two non-trivial shapes. Based on our theoretical results, we develop algorithms to parameterize star shapes to star-shaped domains, such as solid balls and star-shaped polycubes. We also extend our algorithm to parameterize general volumes by using star-shaped decomposition (Chapter 6).

- **Handlebodies Parameterization** We propose a volumetric parameterization algorithm which is able to parameterize volumes with complex topology. It can

also be downgraded and applied to volumes with trivial topology (i.e. topological balls). The parametric domain is simple and easy to construct. Broadly speaking, it is the direct product of a surface patch and a line segment. The resulted mapping between the original volume and the parametric domain is homeomorphism, and there is no singularity. To our knowledge, our method is the first one that can guarantee the bijectivity of the parameterization for volumes with non-trivial topology other than topological balls. At any point in the volume, the w iso-parametric line (which follows the gradient of the harmonic field) is orthogonal to the u and v iso-parametric lines (which span the iso-surface of the harmonic field), which is a natural consequence of enforcing a conformal parameterization on the surface and a gradient field in the volume (Chapter 7).

- **Applications** Based on our theoretical findings, we put the proposed algorithms into use in several real-world applications, including GPU-based subdivision displacement, geometry video compression, volumetric morphing, anisotropic solid texture transfer, GPU-based volumetric computation and hexahedral shell space construction (Chapter 7). Through the experiments, we demonstrate that the proposed surface and volume parameterization methods are effective and powerful for general surface and solid modeling and processing.

1.3 Thesis Organization

The remaining part of the thesis is organized in the following fashion: in Chapter 2, we introduce the mathematical preliminaries on differential geometry and harmonic maps. In Chapter 3, we briefly review the surface parameterization techniques followed by their volume counterpart. In Chapter 4, we present the editable polycube map framework, which allows the user to construct the polycube map in an intuitive and easy

manner. The user is able to sketch features on source shape and polycube domain to specify feature correspondences. In Chapter 5, we develop the expression-invariant surface parameterization for 3D faces. Exact feature correspondences among frames are guaranteed by our method. In Chapter 6, we investigate the Green's function-based volume parameterization in star shapes and extend it into general shapes. The mapping between the star shape and the parametric domain is proved to be a diffeomorphism. We also generalize this method to parameterize general volumes by star-shaped decomposition. In Chapter 7, we propose a volume parameterization method for handlebodies embedded in $\mathbb{R}^3$. The mapping between the original volume and the polycube domain is proved to be a bijection and has no singularity. In Chapter 8, we showcase several applications that benefit from our proposed surface and volume parameterization methods. The applications include GPU-based subdivision, geometry video compression, volume data processing, and shell space construction. Finally, we conclude the thesis and point out several future research directions in Chapter 9.

Chapter 2

Mathematical Background

This chapter introduces the mathematical preliminaries on harmonic maps and differential geometry to help the readers have a better understanding of the following chapters.

2.1 Differential Geometry

Differential geometry studies geometrical objects using techniques of calculus on differentiable manifold.

Manifold A n-manifold is a space that is locally like $\mathbb{R}^n$, although it lacks a preferred system of coordinates. More formally, a n-dimensional topological manifold M is a second countable, Hausdorff topological space that is homeomorphic to open subsets of $\mathbb{R}^n$. n is called the dimension of the manifold, e.g. a line and a circle are one-dimensional manifolds, while a plane and sphere are two-dimensional manifolds.

Differentiable manifold A differentiable manifold is a topological manifold with a globally defined differential structure. Thus differential and integral calculus is allowed on differentiable manifolds.

Diffeomorphism Given two manifolds, M and N, a bijective map f from M to N is called a diffeomorphism if both

$$f : M \to N$$

and its inverse

$$f^{-1} : N \to M$$

are differentiable. If these functions are r times continuously differentiable, f is called a C^r-diffeomorphism. Two manifolds M and N are diffeomorphic if there is a diffeomorphism between them.

From the geometry processing point of view, it is highly desirable to find a diffeomorphism between a manifold and the Euclidean space of corresponding dimension. Specifically, this thesis focuses on finding diffeomorphism between 2-manifolds and 3-manifolds of certain shapes, including human face mesh, polycube surface and star-shaped volume.

2.2 Harmonic maps

Harmonic function A harmonic function is a twice continuously differentiable function: $f : \Omega \to R$(where Ω is an open subset of R^n) which satisfies the laplacian equation $\Delta f = 0$ everywhere in Ω. where

$$\Delta = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \cdots + \frac{\partial^2}{\partial x_n^2}$$

Harmonic functions can be generalized on arbitrary Riemannian manifolds as Laplace-Beltrami operator. The Laplaciae-Beltrami operator is also the divergence of the gradient: $\Delta f = Div(\nabla f)$. More details could be found in Schoen and Yau [65].

Harmonic maps If M and N are two Riemannian manifolds, then a harmonic map $f : M \to N$ is defined to be a stationary point of the Dirichlet energy

$$E(f) = \frac{1}{2}\int \|\mathrm{df}\|^2 dVol \qquad \text{(Eq. 2.1)}$$

where *Vol* is the local n-dimensional volume. In the surface case, a harmonic map minimizes the Dirichlet energy and indicates a minimal surface.

Radó Theorem: Suppose $\Omega \subset R^2$ is a convex domain with a smooth boundary $\partial\Omega$ and suppose that D is the unit disc. Then given any homeomorphism $\mu : \partial D \to \partial\Omega$, there exists a unique harmonic function $f : D \to \Omega$ such that $f = \mu$ on ∂D and μ is a diffeomorphism [65].

This theorem gives the theoretic foundation for harmonic surface mapping between 2D convex domains. However, the Radó theorem does not hold for 3-manifolds. Thus, the volumetric harmonic map is not guaranteed to be bijective even though the target domain is convex. One of the concerns of this thesis is how to construct a diffeomorphism between 3-manifolds.

Chapter 3

Related Works

This chapter surveys some related work in surface parameterization and volume parameterization.

3.1 Surface Parameterization

Surface parameterization is a fundamental tool of surface geometry processing. It was introduced into the graphics community for mapping textures onto surfaces [5, 51]. In recent years, surface parameterization has been demonstrating its power in more and more applications including surface remeshing, shape matching, spline construction, mesh completion, and so on. With the increased interest in applying surface parameterization, various methods have been developed for different kinds of parametric domains and parameterization properties. This section reviews the literatures in planar, spherical, and global parameterization.

3.1.1 Planar Parameterization

Due to the difference in geometry between the source surface mesh and the parametric domain, almost every surface parameterization introduces distortion in either angles or

areas. Surface parameterization methods aim to produce a good parameterization that minimizes the distortion in some sense. Many existing works have been surveyed in Floater [21] and Sheffer *et al.* [68].

In practice, surfaces are usually represented by triangular meshes which are piecewise linear. A commonly used parameterization technique is barycentric coordinates. Each vertex could be represented by the weighted linear combination of its neighbors. Eck *et al.* [15] introduced the discrete harmonic coordinates into planar surface parameterization. The surface parameterization is computed by solving a sparse linear system. The sparse linear system could be efficiently solved with state-of-the-art methods such as TAUCS. But Eck's method cannot guarantee the bijection of the mapping due to negative weights. Floater [20] presented another kind of barycentric coordinates called mean value coordinates by discretizing the mean value theorem. Because the weights are always non-negative, the mapping is guaranteed to be bijective when the parametric domain is convex.

Many planar surface parameterization methods map the source surface to a rectangle or circle domain. Such a convex domain could guarantee the bijection of the map if positive barycentric coordinates are used. The rectangle and circle domains are sufficient for most applications. But the maps have large distortions near the boundary. To alleviate this problem, Lee *et al.* [45] built a virtual boundary around the original boundary. The original mesh is augmented by adding extra triangles around the boundary. Then the original boundary is mapped to the parametric domain in a relatively free manner.

Most methods use a similar framework. A special energy function is defined to present the overall distortion from the isometric parameterization. The parameterization is then computed by minimizing the energy function. Energy functions based on different properties minimize the distortion in different ways. Eck et al. [15] considered the

Dirichlet energy of the inverse function and constructed the harmonic map. Desbrun *et al.* [12] presented the conformal energy. In their method, only two of the boundary vertices need to be fixed in order to give a unique solution. But the result is heavily sensitive to the two vertices. Mullen et al. [54] studied how to get the best choice of the two vertices. Floater [19] presented a shape preserving parameterization based on a specific weight to improve the quality of the mapping in terms of area deformation and conformality. Hormann *et al.* [34] introduced the MIPS energy, which also favors conformality. This distortion measure is symmetric with respect to inversion. MIPS does not require the boundary constraint and the boundary points can move freely in a minimization process. However, minimizing the energy function is a non-linear problem. There is also a variety of local distortion measures [51, 64, 72] that yield a non-linear optimization problem.

Instead of using vertex coordinates to represent the planar parameterization, there is a series of angle-space methods defining the planar parameterization in terms of the angles of planar triangles. Compared with direct conformal methods, angle-space methods produce significantly less stretch in models with high Gaussian curvature. Sheffer *et al.* [69] introduced the Angle Based Flattening (ABF) method to flatten a mesh to a planar plane. Their method is based on the observation that a planar triangulation is uniquely defined by the corner angles of its triangles. To guarantee that the 2D angles define a valid triangulation, they introduced a set of constraints into the framework: 1) the summation of angles of a triangle should equal to π; 2) for each interior vertex, the summation of surrounding angles should equal to 2π; 3) all angles should be larger than 0; 4) edges shared by pairs of triangles should have the same length. They searched for angles that are as close as possible to the original 3D mesh angles and satisfied those constraints using the Lagrange multipliers method. Their method is relatively slow and suffers from stability problems in the angle-to-uv conversion stage for

large meshes. ABF++ [67] was proposed to address those problems. Sheffer *et al.* [67] introduced a stable angle-to-uv conversion using the LSCM method. The computing is greatly speeded up by introducing both direct and hierarchical solution approaches. In recent years, several modifications have been proposed for different cases. To guarantee global bijectivity, Zayer *et al.* [86] introduced additional constraints on the angles enforcing the parameter domain to have convex boundaries. Kharevych *et al.* [40] used a circle pattern based approach for mesh flattening. They introduced the local Delaunay property in addition to the angle constraints. But the optimization problem is still non-linear.

Gu *et al.* [26] introduced a method to parameterize an either open or close mesh to a completely regular domain called geometry image. In their method, the closed mesh must be cut into topological disks and then mapped to a rectangle domain. Although their method could deal with arbitrary mesh, artifacts are introduced near the cutting boundary.

3.1.2 Spherical Parameterization

3D objects are often represented by closed, genus-zero surfaces. For such objects, the sphere is the most natural parameterization domain, since it does not require cutting the surface into disk(s) and thus it allows for seamless and continuous parameterization. Due to these favorable properties, sphere parameterization has attracted increasing interest recently. The literatures could be divided into four main types[35]: 1) Gauss-Seidel iterative extension of planar barycentric methods; 2) stereographic projection; 3) spherical generalization of barycentric coordinates; and 4) multi-resolution embedding.

Inspired by the successful planar parameterization methods, many approaches attempted to extend the barycentric, convex boundary planar methods to the sphere. Alexa *et*

al. [1] and Kobbelt *et al.* [41] proposed methods along this line. They grew the original mesh to its convex hull, and then projected the mesh directly to the bounding sphere. Gauss-Seidel iterations were used to achieve the final parameterization. Isenburg *et al.* [36] introduced a divide-and-conquer method. They split the mesh into two topological disks, mapped the cutting boundary onto a circle and embedded each half-mesh into a hemisphere using a modified Tutte procedure. The modified Tutte procedure computed positions on half a sphere, rather than in the plane. But Saba *et al.* [63] proved that the projected Gauss-Seidel iterations decrease the residual for only a limited number of iterations. As a result, the system is unstable and collapses to a degenerate solution.

The stereographic projection is a conformal mapping that projects a sphere onto a plane. The continuous projection is defined on the entire sphere, except at the single projection point. The mapping is smooth and bijective everywhere except the projection point. Additionally, the projection is conformal. Inspired by the favorable properties, Haker *et al.* [32] introduced a sphere parameterization algorithm based on discrete inverse stereo mapping. Firstly, they computed a planar parameterization of the mesh, using one of the triangles as a boundary. Then the stereographic projection was employed to obtain the spherical mapping. The mapping result depends heavily on the choice of the boundary triangle. But the stereographic projection is bijective only for the continuous cases. There is no theoretical guarantee that it works for discrete cases. In practice, triangle flips might occur. A simple proof is given in Hormann *et al.* [35]. Furthermore, the stereograph projection is not distance or area preserving.

Gotsman et al. [25] presented a theoretically sound spherical parameterization method by employing an intrinsic relationship between spherical parametrization and spectral graph theory. The basic idea is to generalize the method of barycentric coordinates to

the sphere while keeping all its advantages. The challenge in generalizing the planar barycentric coordinates method to the sphere case lies in the difference between the geometry of the plane and sphere. In the planar case, each vertex position could be some convex combination of the positions of its neighbors. But this is not the case for the spherical scenario. By forcing the vertices on the sphere, they construct a large quadratic system which is non-linear. It is quite time-consuming to solve this system. Saba *et al.* [63] introduced an efficient and practical method to overcome this problem. Firstly, they partitioned the mesh into two topological disks, and then embedded each half on a hemisphere using a planar parameterization followed by a stereographic projection. Their numerical solution mechanism for the final parameterization combines Gauss-Seidel iteration with nonlinear minimization. With their method, surface mesh containing up to a hundred thousand vertices could be parameterized in only a few minutes.

Praun and Hoppe [61] observed the two challenges of sphere parameterization: 1) to obtain a robust algorithm, it must prevent parametric "foldovers" in order to guarantee a 1-to-1 spherical map; 2) while all genus-zero surfaces are in essence spherical-shaped, some can be highly deformed, hence creating a parametrization that adequately samples all surface regions is difficult. They achieved robustness by a coarse-to-fine optimization strategy. Firstly, the surface mesh is simplified to a tetrahedron and a progressive mesh is created, favoring triangles with good aspect ratios. After that, the base tetrahedron is mapped to the sphere. Then the vertices are added back progressively. To avoid undersampling and preserving high-frequency surface details, they adopted the penalized undersampling using a stretch-based parameterization metric during the mesh refining. Compared to their method, Shapiro *et al.* [66] computed the embedding using purely topological operations but did not attempt to minimize any type of distortion. Thus heavy stretch might occur.

3.1.3 Global Parameterization

The surface and sphere parameterization methods are restricted to domains with simple topology, such as disks and spheres. Parameterization of surfaces with arbitrary topology remains a challenging problem. The common solution is to introduce cuts into the original mesh to form one or more disks. But the cuts might introduce artifacts near the cutting boundary due to the discontinuity. Global parameterization methods, which target at obtaining globally smooth parameterization on the model of arbitrary genus, are proposed to deal with this problem.

The problem of computing global conformal parameterizations for closed meshes with arbitrary genus is first solved by Gu and Yau [29]. Their method approximated the De Rham cohomology by simplicial cohomology, and computed a basis of holomorphic one-forms. Global continuity could be achieved except at a number of singular points. Although their method is theoretically sound, it has some limitations in the geometric realization of the homology basis. Each homology base curve can only intersect its conjugate once. Hence, the method requires users' guidance. Moreover, their method does not work for surfaces with boundaries. Gu and Yau [30] later presented a purely algebraic method that gives the solution for most of the limitations of their earlier work [29]. In their method, the unique locations of the singular points that satisfy conformality are given. They also solved for the optimal conformal transformation that minimizes global stretch. They analyzed the structure of the space of all global conformal parameterizations of a given surface and found all possible solutions by constructing a basis of the underlying linear solution space. This method has no restrictions on the geometric realization of the homology basis and is therefore automatic. The method was also generalized to handle surfaces with boundaries by double covering.

Boier-Martin et al. [6] proposed a method for parameterizing triangular meshed models over polyhedral domains with quadrilateral faces. They used a combination of centre-based clustering techniques to partition the model into regions that are suitable for local parameterization. A coarse quadrangulated mesh is defined using the region boundaries to obtain a parameterization domain. Ray et al. [62] presented the periodic global parameterization method that aligns the parameterization with orthogonal input vector fields. The parameterization is constructed by the optimization of two periodic scalar functions so that their gradients are as tangential as possible to the input vector fields. Inspired by [62], Kälberer et al. [38] proposed the QuadCover method for global parameterization. The algorithm is automatic and the parametric lines align optimally with a user-defined frame field, e.g., the principal curvature directions. The algorithm consists of two stages. In the first stage, the frame field is slightly changed to be a locally integrable field. Then the surface is cut into a set of open meshes. The frame field is integrated by inducing a parameterization, which is discontinuous near the cutting boundaries. In the second stage, the frame field is altered once more, so that the parametric lines close globally, even at the cuttings.

Although numerous surface parameterization methods have been proposed, few of the above methods allow mapping constraints inside the domain. From the users' point of view, it would be highly desirable to set interior mapping constraints in some practical operations such as texture mapping, surface morphing, motion data alignment, etc. In this thesis, we propose two different techniques to take interior features as mapping constraints for two different applications respectively. In the polycube map, we use a divide-and-conquer strategy that segments the surface into patches according to the features. Then the features serve as the boundary constraints in the mapping between each pair of patches. A user controllable mapping framework is proposed and post-editing of the mapping result is allowed. In facial expression parameterization, interior

constraints are also important for the alignment of the whole sequence. We cut the face mesh into a topological annulus according to the salient features so that all the interior features are on the boundaries. Then a one-to-one map is induced by harmonic fields. Our methods give general solutions for harmonic maps with interior constraints.

3.2 Volume Parameterization

Volume parameterization is a fundamental part of volume data processing. In recent years, volume parameterization has received increasing attention. But due to the gap between surface and volume, most successful surface parameterization methods cannot be extended to volume cases directly. Compared with the extensive research of surface parameterization, volume parameterization remains less investigated.

Most existing volume parameterization methods are based on the volumetric harmonic map, which is usually defined on the tetrahedral mesh. Thus generating a tetrahedral mesh from a given surface mesh attracts much research interest. Labelle and Shewchuk introduced the iso-surface stuffing algorithm to generate tetrahedron meshes with bounded dihedral angles in [43]. The volumetric discrete Laplace-Beltrami operator used in this work generalized the *cotan* formula in the surface case; the *cotan* value of dihedral angles is used to replace those of corner angles. The range of the dihedral angles affects the parameterization quality. A Delaunay-based variational approach for isotropic tetrahedral meshing was introduced by Alliez et al. in [2]. This method can produce well-shaped tetrahedra by energy minimization. The tandem algorithm was introduced for iso-surfaces extraction and simplification in[3]. The volumetric harmonic map depends on volumetric Laplacian. Zhou et al. [89] applied volumetric graph Laplacian to large mesh deformation.

Wang et al. [80] introduced the volumetric harmonic map to volume parameterization. They proposed a method for parameterization and tetrahedral meshing of brain image data. First, a tetrahedral finite element mesh (FEM) was employed to reconstruct the brain volume from a sequence of magnetic resonance images (MRI). After that, the boundary mesh is conformally mapped to a sphere. The surface map serves as the boundary condition in the volumetric mapping. Then the harmonicity in volumes is similarly defined via the vanishing Laplacian, which governs the smoothness of the mapping function. A heat flow method was used to solve the volumetric harmonic mapping problem. Both brain data and synthetic data showcase the power of their method. However, their method can only handle genus-0 objects.

Li et al. [47] introduced a fundamental solution method into volumetric parameterization. The mapping was represented as a set of points with different weights in the vicinity of the solid boundary. Given two solid objects and the surface map between them, they placed the source points and the collocation points by the surface map. Then they used a method of fundamental solutions yielding a coefficient matrix. After that, the coefficient matrix was decomposed by the singular value decomposition method. Finally, the harmonic volumetric mapping was obtained by solving the linear system with input boundary mapping constraints. Their method is meshless and automatic. However, the parameterization computing is time-consuming.

Other than that, harmonic volumetric parameterization for cylindral volumes is applied for constructing tri-variate spline fitting in [52]. They used a skeleton-based method for parameterizing cylinder-like objects. First, two critical points were chosen to establish a surface parameterization induced by orthogonal harmonic functions. Then a structured quadrilateral mesh was constructed using the surface parameterization. Then they constructed a volume harmonic field by setting the value of interior skeleton to 1

and the outer boundary to 0. A hexahedral mesh was generated based on the volume harmonic fields. Finally, they extracted a tri-variate spline structure from the hexahedral mesh. However, their method also has limitations. First, it can only handle genus-0 volumes. Second, it has singularities in the skeleton and the two critical points.

All the above approaches rely on the use of volumetric harmonic maps. Unfortunately, as pointed out in Chapter 1, volumetric harmonic maps cannot guarantee bijective mappings even though the target domain is convex.

Besides the volumetric harmonic maps, another stream of research studies the mean value coordinates for closed triangular meshes [37, 22]. Mean value coordinates are a powerful and flexible tool to define a map between two volumes. However, like other types of barycentric coordinates, mean value coordinates suffer the same problem as harmonic fields in that there is no guarantee that the computed map is a diffeomorphism.

3.3 Polycube Map

The polycube map is a novel global cross-surface parameterization technique, where the polycube shape can roughly approximate the geometry of modeled objects while retaining the same topology. The first call of the polycube map comes from texture mapping. Standard texture mapping of real-world meshes suffers from the presence of seams. In contrast, the cube map provides a mechanism that could be used for seamless texture mapping with low distortion, but only if the object roughly resembles a cube. Compared with other global parameterization techniques, the quality of a polycube map (in terms of angle and area distortion) can be quantitatively controlled by designing the polycube such that it resembles the geometry of the input shape and shares the same topology. Because of the highly regular structure (i.e., each face is a square or poly-

square) and the nature of the "one-piece" global parametric domain (i.e., no cutting and abutting), it is especially suitable for image space tasks and current graphics hardware.

Tarini *et al.* [74] pioneered the concept of polycube maps. They roughly approximated the input 3D model by a polycube, and then constructed its dual space. By carefully examining the different configurations that may occur in a non-empty cell, they designed six projection functions that map the points inside a cell of the dual space to the polycube surface. These projection functions are elegantly designed to guarantee a globally continuous map. However, their method does not produce a bijective mapping since two vertices on the same projection line share the same image, which is a fundamental feature for a large range of applications. Furthermore, their method has strict requirements on the shape of the polycube, for example the dual space should completely enclose a slightly modified version of the input model in an intermediate coordinate space.

Rather than projecting the 3D surface to the polycube, Wang *et al.* [78] introduced an intrinsic approach. Firstly, they mapped the 3D model and the polycube to the canonical domain (e.g., sphere, Euclidean plane or hyperbolic disc), and then sought for the mapping between the two canonical domains. The resulting polycube map was guaranteed to be a diffeomorphism. However, in this scheme it is difficult to control the polycube map, i.e., a feature on the 3D model may not be mapped to a desired location on the polycube. In their follow-up work, Wang *et al.* [79] proposed the user-controllable polycube map where the user can specify the pre-images of the polycube corners. Their method works well for shapes with simple geometry and topology, but is not feasible for complicated models since it is very tedious and error-prone to specify the polycube structure manually. Conversely, the method presented in this thesis provides much more user control, which allows us to create polycube maps with a much

smaller number of patches, and gives much more control over the quality of the induced subdivision surface, which is what makes this method practical for real-time rendering on modern hardware.

It is known that the quality of the polycube map (in terms of angle and area distortion) depends greatly on the shape of the polycube. There have been some research efforts that aim to construct the polycube automatically. Lin *et al.* [48] proposed an automatic polycube map method. Firstly, they decomposed the input mesh into a series of feature regions, which were further split into patches. Then each region was approximated by a polycube primitive and each patch of it was mapped to a rectangular sub-surface of the polycube primitive. After that, each polycube primitive was locally parameterized. Finally, the global parameterization was smoothed iteratively. He *et al.* [33] employed a similar strategy that first broke down the input model into smaller and simpler components and then used polycube primitives to approximate each one. As heuristics (Reeb graph in Lin *et al.*[48] and harmonic function in He *et al.*[33]) are usually used in the segmentation and polycube approximation, these approaches may not work for models of complicated geometry and topology. Furthermore, none of the existing algorithms allows the user to edit the maps easily.

3.4 Harmonic Maps

Eck *et al.* [15] introduced discrete harmonic maps into the computer graphics community. The basic approach first fixes the boundary mapping and then finds the piecewise linear mapping which minimizes the Dirichlet energy. The quadratic minimization problem could be reduced to solving a linear system of equations and thus it could be computed easily. There are many favorable properties of harmonic maps: 1) They are easy to compute. Solving the harmonic fields could be deduced to a linear system which

could be solved quickly using state-of-the-art libraries such as Cholmod. 2) Harmonic mappings between a 3D bounded surface and a 2D planar surface have been proven to be diffeomorphisms if the parametric domain is convex. 3) When we focus on mappings from general surfaces $S \subset \mathbb{R}^3$ to a plane, the optimum properties of conformal and harmonic mappings are essentially the same [21]. 4) The harmonic maps framework is compatible with some new advances in barycentric coordinates, such as mean value coordinates. Attracted by the promising properties, a lot of harmonic fields-based algorithms have been proposed [87, 80, 47, 52].

Chapter 4

Editable Polycube Map

4.1 Motivation

It is a well known fact that most artists prefer to modeling with quads, as quad geometry provides a better flow, tessellates cleaner and deformations under animation that are noticeably smoother, especially around joints [57].

Tarini *et al.* [74] pioneered the concept of polycube maps as a technique to parameterize 3D shapes to the polycube domain, which is a natural generalization of the cube space and can be useful for the parametric domain of shapes with complicated topology and geometry. Compared to other global surface parameterization techniques, a polycube map has two unique features that make them promising for graphics applications. First, the parametric domain has a regular structure that naturally induces to a quadrangulation. The parametric domain can be easily constructed and visualized. Second, the singularities (polycube corners) have fixed structures, i.e., of valence 3, 5 or 6. The reduced number of possible singularities results in a small number of topological combinations.

Constructing a polycube map is a challenging work. From the users' point of view, an ideal polycube mapping algorithm should have at least the following features:

- Quality: the map is a bijection with low angle and area distortion.
- User control: the user can easily control the mapping by specifying optional features on the 3D model and their desired locations on the polycube domain.
- Performance: the algorithm is efficient, robust, and automatic except the editable user-specified constraints to control the map.

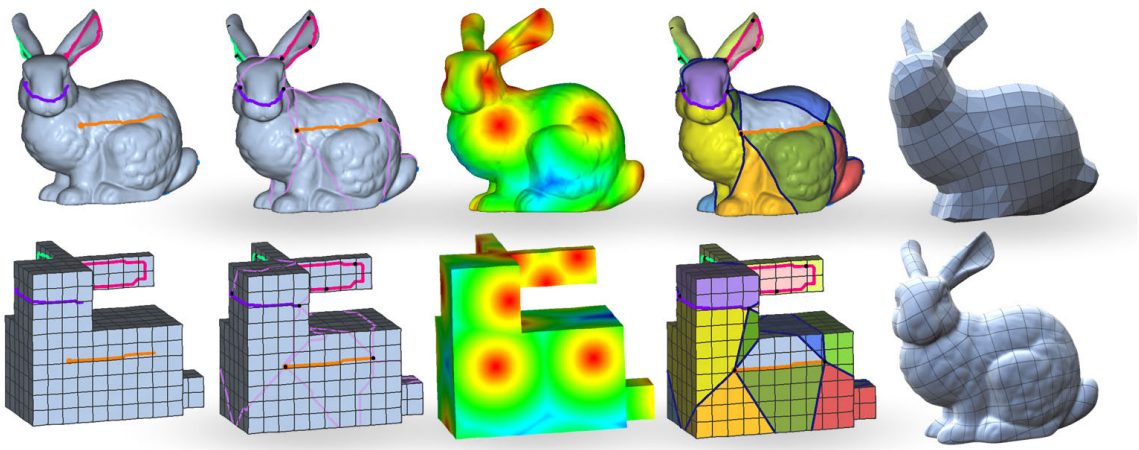
There are several approaches to construct a polycube mapping [74, 78, 79, 48, 33]. Unfortunately, none of them has all the desired features. For instance, Tarini *et al.* [74] does not guarantee a bijection, Wang *et al.* [78], Lin *et al.* [48], and He*et al.* [33] do not allow user control. Wang *et al.* [79] requires a large amount of user interaction to specify the polycube structure on the 3D model and is not suitable for large-scale models with complicated geometry and topology. More importantly, none of them allows the users to *edit* the polycube map in an easy and intuitive fashion: In general, current tools for editing in planar parametric domains can be awkward to use. On the other hand, editing on a polycube, which more closely resembles the gross structure of the model, could indeed be simpler, especially where the user is allowed to control the rough shape of the mapping.

In this work, we present the editable polycube map to overcome the limitations of the existing approaches. Our method allows the users to construct the polycube map in an intuitive and easy manner: given a 3D model M and its polycube domain P, the user is able to sketch features on M and P to specify feature correspondences. Then, our system will automatically compute the map in such a way that the features on M are mapped to the user-specified locations on P. Later on, the user is allowed to edit the features on M, P or both, providing fine-grained control over the mapping. This way, the editable features can help to *mark* and preserve sharp features on the polycube-mapped subdivision scheme.

4.2 Algorithm

4.2.1 Overview

As mentioned above, the objective of the proposed technique is, starting from a high-resolution model coming either from an artist or a 3D scanner (plus its cleaning stage), to build a new subdivision scheme that allows user control over a reduced number of singularities and, at the same time, have that model ready for seamless texturing, watertight displacement, etc. This should be done in an artist-controllable way, using only quads, and without wasted space in texture space.



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

(e)

Figure 4.1: Algorithmic pipeline. (a) The user sketches the constraints on the given 3D model M and polycube P. (b) The black dots are the sample points on the user-specified constraints. (c) We compute the distance fields using the sample points as sources. (d) The distance fields induce a triangulation on M. Then using the correspondence of the user sketched features, we compute the geodesic triangulation on P. Then the model can be cut into genus-0 patches. We compute the constrained map between each pair of patches such that the user-defined features are mapped consistently. With a carefully designed boundary condition, the two maps can be glued seamlessly. (e) The iso-parametric line on M. The top shows the base level of the subdivision surface.

The input to our technique is a high resolution model plus a simple polycube representa-

tion. By controlling the position and location of the cubes, and by providing additional controlling sketches, the user has control at all times over the number and location of the singular vertices in the resulting quads-only mesh. Then the user-specified features are sampled with points, from which we compute the shortest distance for each other sampled point using multi-source Dijkstra's shortest path algorithm. Although the computed distance field induces a triangulation on the 3D model, the path between two points is not straight and the resulting patch can have a complex non-triangular shape. To ensure the quality of the triangulation, we compute the geodesic triangulation on the polycube by using the correspondence of the user-specified features on the 3D model and the polycube. This way, the resulting triangulation is smoother and more visually pleasing than the one that could be obtained directly from the multi-source Dijkstra computation. After that, both shapes are segmented into genus-0 patches, keeping an identification between the segments in the polycube and in the high resolution model. We compute the map between each pair of patches using a series of harmonic maps. By setting the boundary conditions carefully, the computed map is guaranteed to be continuous along the cutting boundaries. Finally, a simple and effective global diffusion algorithm is applied to improve the map quality. The resultant map is bijective and satisfies the user-specified constraints.

The polycube map induces a quad-remeshed version of the high-res model. An immediate benefit of this process is that the patches of the resulting remeshed model come from tessellated squared faces, so a very low resolution version of the model can be built from the original polycube faces. This low resolution model not only has quads as its only primitive, but also has only vertices with a small and restricted valence number, only 3, 4, 5 and 6.

Then, polycube texture mapping is performed, working only with the polycube version of the model, and consists of creating a 2D texture atlas which contains all externally

visible polycube faces, which is an easy task as the previous step already kept only the visible faces that are not shared by more than one cube. To build the atlas, we just placed each face in consecutive squares in the final texture atlas.

Finally, in runtime, only the very low resolution model needs to be sent to the GPU, along with the texture maps built in the pre-processing, to generate a continuous subdivision scheme with all the benefits already mentioned in Section 4.1. As the very low resolution model can be animated, its run-time tessellated version also can. It is important to mention that the user/artist has full control over the process through the segmented input polycubes, the sketched features and the segmentation step, and he can introduce further tweaking in the processed mesh without the need to re-parameterize the model. The entire algorithmic pipeline is illustrated in Figure 4.1.

4.2.2 Constructing Polycube Map

The input model M for the method presented in this chapter comes from two kinds of possible input models: a model from a 3D scanner, after a cleaning pass, or a high resolution model created directly by an artist.

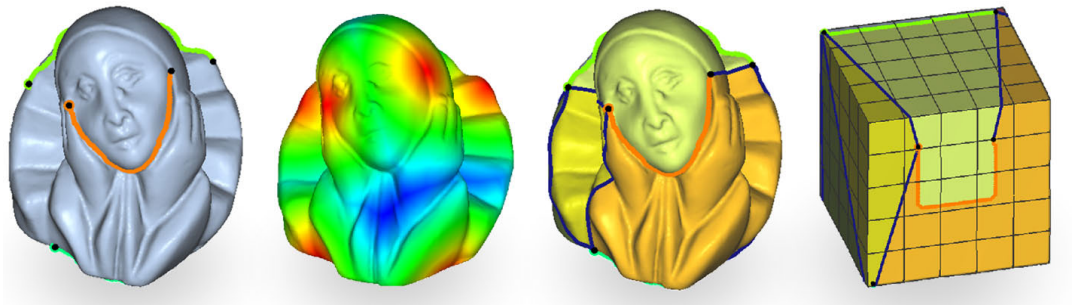
Along with the high resolution model, the artist/modeler should also provide a polycube surface P that follows the shape of the original model. This polycube model can be coarse or fine, both being able to be used to create a good dual parameterization of the original mesh, but depending on how much fine the polycube model is, it would be much easier to capture high frequency details in the final tessellated model.

4.2.2.1 Segmentation

The divide-and-conquer approach He *et al.* [33] constructs the polycube map by breaking down the models into genus-0 patches and then computing the piecewise map inde-

pendently. Although their method is able to divide the model in an automatic fashion, all cutting planes are horizontal. Thus, the segmentation highly depends on the orientation of the model and may result in too many small patches. Furthermore, the cutting boundary may not represent any features.

In our framework, the users are allowed to sketch the features freely on the models. Here we assume that the user-specified features are consistent. For example, as shown in Figure 4.1(b), the user sketches a few features on the 3D model M, and the same number of features must be specified on the polycube P. Furthermore, the spatial relation of the features should be consistent in the sense that they are aligned in a similar order, otherwise, the induced harmonic maps may not be matched.



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

Figure 4.2: Multi-source Dijkstra's based distance field and its induced triangulation. The black dots in (a) are the sample points on the user sketches (the color curves). Using these sample points as sources, we compute the Dijkstra's based distance field on the 3D model, as shown in the color map in (b), where warm colors mean small distance to the sources, and cold colors represent large distance. The curves in (c) are the shortest paths between sample points, which induces a triangulation on M. By using the correspondence of the user sketched features, we construct the geodesic triangulation on the polycube P, as shown in (d).

Our segmentation algorithm is as follows:

Step 1. Given the user-specified sketches, $\gamma_i \in M$, $\gamma_i^{'} \in P$, $i = 1, \cdots, n$, we sample the sketches with set of points. Let $S = \{p_j\}_{j=1}^{m}$ and $S' = \{p_j^{'}\}_{j=1}^{m}$ denote the sample points on M and P respectively.

Step 2. Use $p_j \in M$, $j = 1, \cdots, m$, as source points and compute the shortest distance for every point on M using multi-source Dijkstra's algorithm. So each vertex v is associated with a distance $d(v, p_j)$ where p_j is the closest sample point to v. Let $c(v) \in S$ be the closest sample point of vertex v.

Step 3. Consider each mesh edge $e_{ij} = (v_i, v_j)$, where v_i and v_j are neighboring mesh vertices. If $c(v_i) \neq c(v_j)$, let $s_1 = c(v_i)$ and $s_2 = c(v_j)$ be the two sample points. Mark the two sample points s_1 and s_2 as neighbors.

Step 4. For every pair of sample points s_i and s_j which are marked as neighbors, find the shortest path between s_1 and s_2. It can be shown that two shortest paths can not meet except at the regions of the two ending sample points. Then on the polycube P, compute the geodesic between $s_i^{'}$ and $s_j^{'}$.

Step 5. Segment M and P along the computed shortest path and geodesic path. The segmentations on M and P are consistent in most case when user draws reasonable feature lines. But there are some tricky cases. e.g. there are two feature points q and p on a cylinder, the orientation of the line l_{qp} could have two possibilities. In practice, we just compute the segmentation on M, then transfer the segmentation to P.

In the above algorithm, we compute a distance field on the 3D model M using the user sketched constraints. This distance field naturally induces a triangulation on M.

4.2.2.2 Constrained Map

Let $P_i \in P$ and $M_i \in M$ be the pair of segmented patches, each of which is a genus-0 surface with only one boundary. We want to find a bijective and smooth map $\phi : M_i \rightarrow P_i$. Rather than computing the map directly, we first parameterize M_i to the unit disc using harmonic map, i.e., $f : M_i \rightarrow \mathbb{D}$ such that $\triangle f = 0$ and f maps the boundary of M_i

to the boundary of $\mathbb{D}$ using arc length parameterization, $f(\partial M_i) = \partial\mathbb{D}$. Similarly we also parameterize P_i to the unit disc using harmonic map $g : P_i \to \mathbb{D}$.

Then we seek a smooth map between two unit discs $h : \mathbb{D} \to \mathbb{D}$. This map h is also computed using harmonic map $\triangle h = 0$ and the boundary condition is set as follows: Let s_1, s_2 and s_3 be the sample points on ∂M_i, and $f(s_j) \in \partial\mathbb{D}$, $j = 0,1,2$ be the images on the boundary of unit disc. Similarly, let $g(s_j^{'}) \in \partial\mathbb{D}$ be the images of the sample points $s_j^{'} \in P_i$. Then we require the function h maps $f(s_j)$ to $g(s_j^{'})$, i.e., $h \circ f(s_j) = g(s_j^{'})$, $j = 0,1,2$. The images for the points between $f(s_j)$ and $f(s_{(j+1)\%3})$, $j = 0,1,2$, are computed using arc length parameterization.

Finally, the polycube parameterization is given by the composite map $\phi = f \circ h \circ g^{-1}$ as shown in the following commutative diagram:

$$\begin{array}{ccc} M_i & \xrightarrow{\phi} & P_i \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle g} \\ \mathbb{D} & \xrightarrow[h]{} & \mathbb{D} \end{array}$$

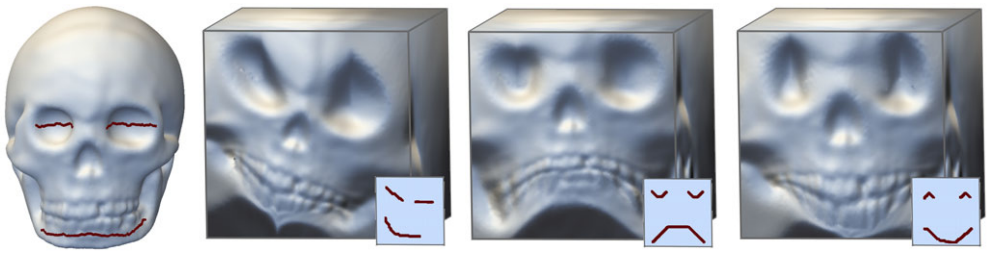


Figure 4.3: The users can take full control of the polycube map by simple sketches. The thumbnail of each figure shows the sketched constraints on the polycube.

4.2.2.3 Globally Smoothing Map

We glue the piecewise maps $\phi_i : P_i \to M_i$ together. With the above boundary conditions, the piecewise maps are consistent along the boundaries which can be glued

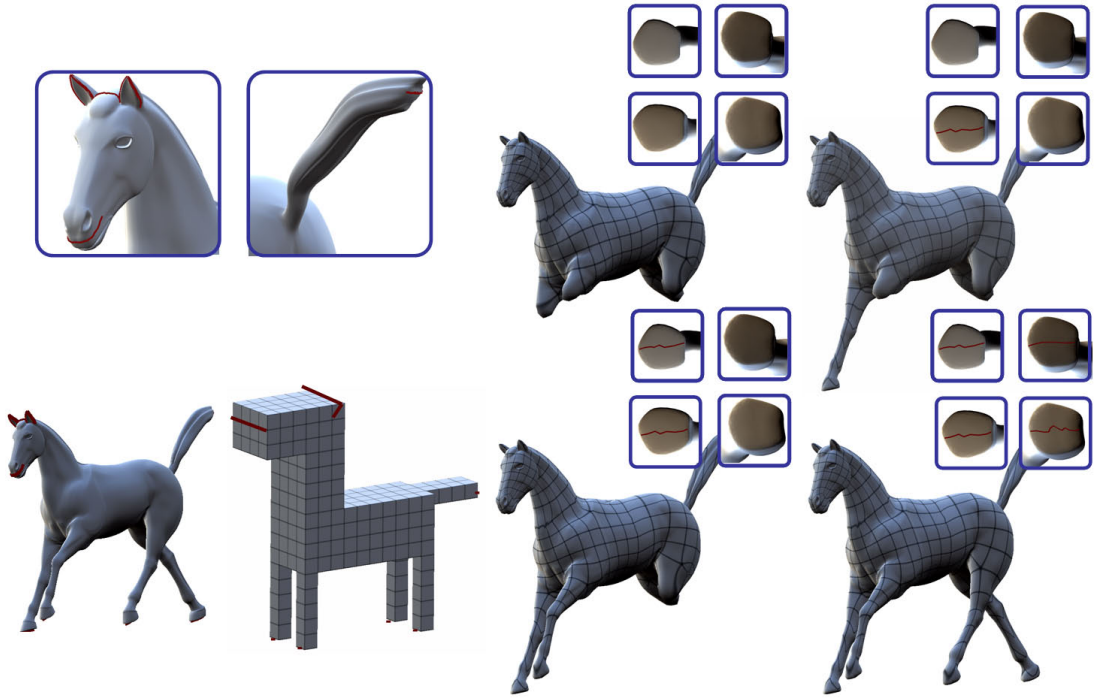


Figure 4.4: Interactive editing the polycube map. Initially, the user only draws three features on the mouth, ears and tail. As a result, the horse head, body and tail are parameterized well, but the four legs are poorly sampled. Then the user can improve the map quality by specifying additional features on the feet. Finally, the whole model is parameterized well.

seamlessly. The resulting map $\cup_i \phi_i$ can guarantee to be C^0 continuous along the segmentation boundaries.

We use Laplacian smoothing Field [18] to improve the continuity along the segmentation boundaries. Given the initial polycube map $\phi : P \to M$, let $p' = \phi(p) \in M$ denote the image of $p \in P$. Then we solve the following diffusion function:

$$\frac{\partial p'(t)}{\partial t} = -(\triangle p'(t))_{\|}, \qquad \text{(Eq. 4.1)}$$

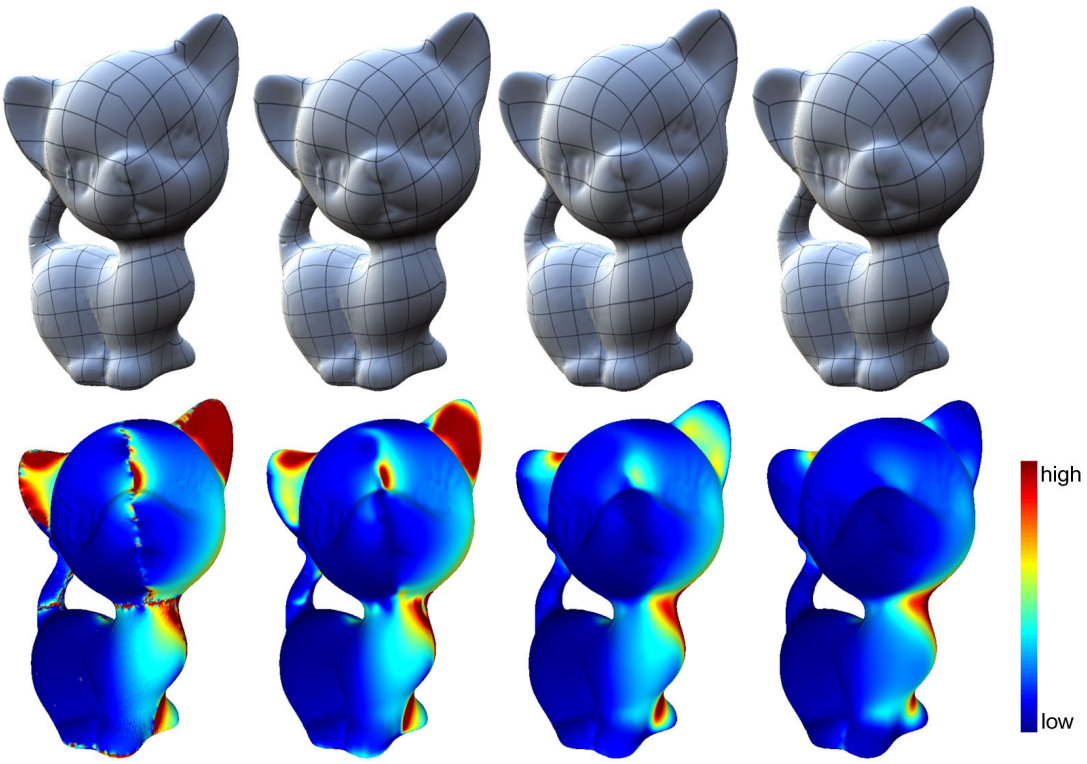
where $\mathbf{v}_{\|} = \mathbf{v} - (\mathbf{v}, \mathbf{n})\mathbf{n}$ is the tangent component of $\mathbf{v}$, $\mathbf{n}$ is the normal vector and $(,)$ is the dot product.

Since the given polycube map ϕ is represented in a quadrilateral mesh induced by the

tessellation of the polycube in which each quad in P is a square, we use the following Laplace operator:

$$\triangle p' = p' - \frac{1}{m} \sum_{pq \text{ is edge}} q', \qquad \text{(Eq. 4.2)}$$

where m is the valence of p.



(a) initial map 1.151	(b) 100 iterations 1.125	(c) 500 iterations 1.105	(d) 2000 iterations 1.087

Figure 4.5: Smoothing the polycube map. The initial map has only C^0 continuity along the segmentation curves and user-specified features. Thus, one can clearly see the large distortion and unsmoothness in (a). Using the Laplacian smoothing algorithm, the distortion smoothly spreads out over the entire model, see (b)-(d). The value below each figure is the angle distortion. The step length $\delta = 0.05$.

The above diffusion equation can be solved easily using the Euler method. We set the step length $\delta = 0.05$ in our experiments.

Following Degener *et al.* [11] and Tarini *et al.* [74], we measure the map quality in terms

of angle and area distortions which integrate and normalize the values $\sigma_1\sigma_2 + 1/\sigma_1\sigma_2$ and $\sigma_1/\sigma_2 + \sigma_2/\sigma_1$, where σ_1 and σ_2 are the singular values of the Jacobian matrix of ϕ. $\varepsilon_{angle} = \varepsilon_{area} = 1$ when the map ϕ is isometric. As shown in Figure 4.5, our method leads to visually pleasing results in only a few hundred iterations.

4.2.3 Texture Mapping and Run-Time Data Generation

For any further processing, we should provide a map from the polycube space to regular texture space, a process that only needs the polycube texture coordinates of the model. With them, a 2D texture atlas is created containing the textures corresponding all *visible* polycube faces, in a similar way to the proposal by Cohen at al. [9]. As already mentioned, this is an easy task as the previous steps already discarded internal faces shared by more than one cube. Once the visible faces are found, they are placed in consecutive squares in the final texture atlas. Each vertex in the model is assigned its respective texture coordinates in the atlas, thus smoothly integrating texturing to the polycube mapping process.

Once the mapping stage has finished, we are in possession of a bijective mapping between the tiles in texture space (one for every patch) and the high resolution 3D geometry the artist provided as starting point. Probably, the artist also provided normal maps, color maps, specular maps, etc. We have developed a completely automatic scheme that transfers (bakes) that information to texture tiles, and encodes them along with the subdivision information. Traditionally, this is done with applications like ZBrush [59] that perform an explicit projection over different resolution versions of the same model, but this can have serious problems with an arbitrary simplification scheme. In our case, as we have defined a *dual parameterization* between the original 3D model and texture space through the polycube-based coarse model, this is done in an automatic way without loosing any detail in the process. To do it, we rasterize each high resolution patch

Table 4.1: Comparison of polycube map construction methods. Symbols: ● good, ◐ fair, ○ poor.

Features	Tarini [74]	Wang [78]	Wang [79]	Lin [48]	He [33]	Our method
Map quality	●	◐	◐	●	●	●
Bijection	○	●	●	●	●	●
User control	○	○	◐	○	○	●
Editing	○	○	○	○	○	●
Polycube construction	○	○	○	●	●	○
Automatic	●	○	○	●	●	●
Large models	●	○	○	○	◐	●
Arbitrary topology	○	○	○	○	◐	◐

Table 4.2: Statistics of experimental results. #△: # of triangles in the input mesh; #□: # of squares in the polycube of base level; l: # of subdivision levels for the high resolution polycube; n_c: # of corners in P; n_f: # of user-specified features; T: time measured in seconds; ε_1: angle distortion; ε_2: area distortion.

	#△	#□	l	n_c	n_f	T	ε_1	ε_2
Armadillo	346K	1012	6	82	28	148	1.16	1.18
Arthur	252K	57	7	12	5	76	1.02	1.13
Bunny	144K	452	4	22	6	35	1.01	1.13
Horse	67K	436	5	28	8	11	1.04	1.07
Lucy	526K	980	5	38	15	210	1.12	1.23
Skull	52K	24	7	8	3	7	1.02	1.06
Tylo head	500K	920	5	32	12	194	1.10	1.25
Isidore Horse	151K	74	10	14	6	48	1.01	1.07

into its associated texture space, saving the required information (normals, occlusion maps, etc). Special care should be taken when doing this process with the displacement maps, either if they are scalar displacement along the surface normal or full 3D displacements, as we also need to have access to neighboring patches to correctly generate this information. The whole process can easily be done with tools like XNormal [58].

4.3 Results and Discussion

We tested our method with a wide-range of models with various geometry and topology configurations as shown in Fig. 4.6. Table 4.2 shows the statistics of our experiments.

Figure 4.4 shows an example of interactive editing of the polycube map of the horse model.

A good global bijective parameterization is very important for modeling and texturing, as typical projections as used in Tarini *et al.* [74] unavoidably have problems since two vertices on the same projection line share the same image, something that produces artifacts that are clearly visible in Figure 4.7.

Figure 4.6: More polycube mapping results.

We also quantitatively compared our methods with existing methods like Tarini *et al.* [74], Wang *et al.* [78], Wang *et al.* [79] and He *et al.* [33] as shown in in Table 7.2 and Figures 4.7 and 4.8. Tarini *et al.* [74] is efficient for large scale models, however it can not guarantee the bijectivity due to the projection of the 3D model to the polycube, therefore, it may result in some undesired artifacts in texture mapping and painting, see Figure 4.7. Wang *et al.* [78] computed the polycube map in an intrinsic way by conformally parameterizing M and P to canonical domains and then seeking the map between the domains. For genus-0 shape with complex geometry (like the

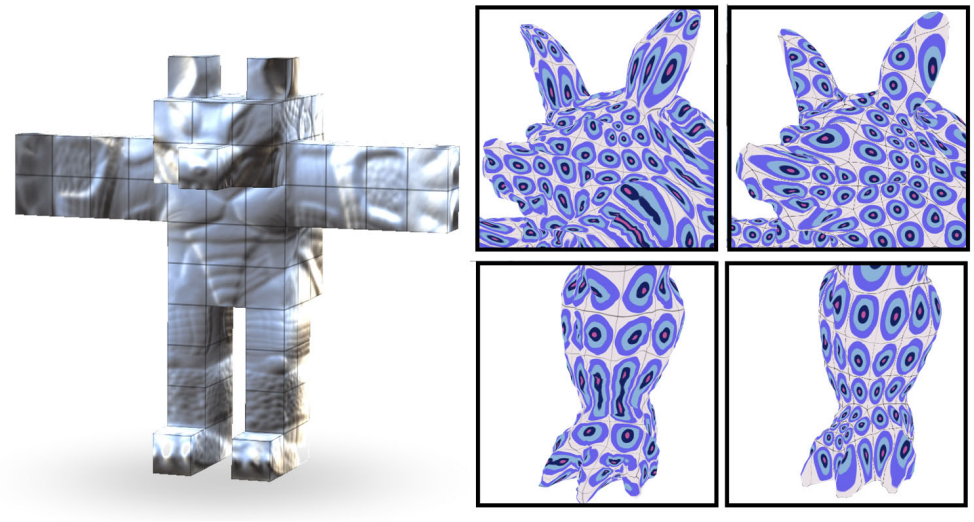
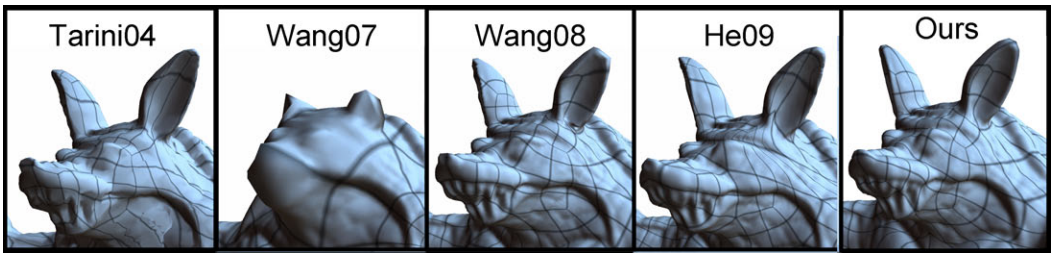


Figure 4.7: Comparison of the bijectivity between the editable polycube map and the original polycube implementation, at the left the original [Tarini et al. 2004]. Top row: insets of the Armadillo head. Bottom row: insets of its feet. As we can see, non-bijectivity in [Tarini et al. 2004] results in a dependence of multiple points on the mesh with a single point in texture space: painting this single point stains other places than the one originally intended.

Armadillo), the conformal spherical parameterization has very large area distortion on the elongated parts (e.g., the arms, legs and tail). Thus, the induced polycube map also has a large area distortion with uneven sampling. Wang *et al.* [79] required the users to manually specify the images of the polycube corners and edges on the 3D model and then computed the map for each polycube face individually. The user-defined polycube structures on M can be considered as the constraints in our method. However, it is very tedious and error-prone to specify the features manually if the polycube is complicated, which precludes its usage for large-scale models. Furthermore, the user can not specify other features or constraints in Wang *et al.* [79]. Automatic approaches as Lin *et al.* [48] and He *et al.* [33] use heuristics and may not work well for complex models. In He *et al.* [33], both M and P are segmented by horizontal cutting planes, and the cutting locus serve the constraints. Thus, the generated polycube map is orientation dependent. Due to the complex geometry of the Armadillo, e.g., the arms are not axis aligned, there

are large distortions on the upper arms and shoulder, see Figure 4.8. Compared to the existing approaches, our method is more intuitive and flexible in terms of user control and editing, and can generate polycube of better quality.



1.32	1.35	1.29	1.25	1.16
1.22	88.62	1.24	1.23	1.18

Figure 4.8: Comparisons of our method with [Tarini et al. 2004], [Wang et al. 2007], [Wang et al. 2008] and [He et al. 2009]. We use the same polycube to make the comparison fair. Our method is more intuitive and flexible in terms of user control and editing, and can generate polycube map of better quality. The values below each figure are the angle and area distortions. For this example, 3 feature lines are drawing on the ears and mouth to give an accurate mapping constraints by our method

Discussions We want to emphasize the differences between our method and He *et al.* [33]. He *et al.* [33] first segmented the 3D model and polycube by horizontal planes, then computed a map between each pair of segmented components, and finally smoothed the whole map by solving a harmonic map for the entire shape. Although using the same divide-and-conquer strategy, our method is completely different from He *et al.* [33] in all the below three steps: First, since the cutting loci are also the constraints of the map, He *et al.* [33] can only map the horizontal, planar features from the 3D model. Our segmentation allows the users to cut the model by arbitrary closed curves, resulting in more flexible and meaningful constraints. Furthermore, our method supports the user control and editing that are not provided in He *et al.* [33]. Second, He *et al.* [33] mapped the segmented components to the multiply connected rings by an uniformization metric, and then computed a harmonic map between the rings. It is known that computing this metric is a nonlinear time-consuming process. Our method

computed a harmonic map between two topological disks, thus is more efficient than He *et al.* [33]. Third, rather than solving a harmonic map for the entire shape, we smoothed the whole map by an iterative method to diffuse the angle distortion. As shown in Figure 4.5, our method is very effective and leads to a high quality map with only a few hundred iterations. Since very simple vertex operations are involved in each iteration, the diffusion method is very efficient.

It is known that the distortions of polycube parameterization highly depend on the shape of the polycube. In general, the more accurate representation of the polycube, the lower distortion of the polycube map. However, the price to pay is the larger number of extraordinary points (polycube corners). Thus, there is a tradeoff between quality and complexity. Through our experiments, we observed that for shapes with extruding regions, e.g., the Armadillo's fingers and toes, it is usually a good idea to design an accurate polycube to model these features. In our framework, we leave the choice to the users.

It would be interesting to discuss the relationship between user drawing and mapping results. First, our method do need user inputs as the boundary conditions of harmonic fields. Second, user drawings on the object surface and polycube domain should keep salient consistence, e.g. if user draws a feature line on the leg of the horse model, a feature line should be drawn on the corresponding leg of the polycube. Figure 4.4 gives a example of the feature line correspondences between object shape and polycube.

Chapter 5

Expression-invariant 3D Face Parameterization

5.1 Motivation

In Chapter 4, we have proposed a user-controllable polycube map method based on harmonic fields and divide-and-conquer method. Technically, it suggest a general manner to handle 2-manifold mapping with constraints inside the domain. However, while being user controllable and editable, the method cannot afford an automatic framework. It also needs a considerable number of features to induce a reasonable segmentation.

In this chapter, we present an automatic surface parameterization method which needs only a few salient feature constrains. Specifically, we propose an expression-invariant human facial expressions parameterization method. The salient features are labeled only in eyes, nose and mouth. Expression-invariant refers to the property that the same salient features of different expressions are mapped to the same position respectively in the parametric domain. Thus, the parameterized facial expressions are aligned in the parametric domain. We also showcase the power of our method in the specifical application, human facial expressions modeling.

Human facial expressions is widely used in movie and game industries. It is usually scanned by a 3D camera. However, the latest 3D camera (structured light base camera) has several serious drawbacks that inhibits its use in broader applications:

- First, each frame of the captured motion data is in the reference system of the scanner, and it is not registered in object space. Thus, the correspondences between points in different frames are not available. However, from the modeling point of view, it is highly desirable to have such correspondence among frames.
- Second, the captured raw data may contain noise and/or holes due to various reasons, such as camera occlusion, specular reflection, shadows, light interference, depth discontinuity, etc. Thus, much efforts are needed to clean and repair the datasets.

(a) (b) (c)

Figure 5.1: The 3D camera is capable of capturing high-resolution motion data at 30 fps. Both geometry (vertex coordinate) and texture (greyscale color) are encoded in a quadrilateral mesh with approximately 250K vertices. (a) The image captured by conventional 2D camera. (b) The 3D mesh captured by 3D camera. (c) Another view of the 3D mesh.

Geometry images [26] are a novel concept that intelligently encodes the 3D geometry into an image format, in which each pixel $\{r,g,b\}$ represents a 3D vertex $\{x,y,z\}$. To process 3D motion data, it is natural to extend geometry images to geometry videos (GV) which bridges the gap of 3D motion data and 2D video, and provides a way to

apply the well-studied video processing techniques to motion data compression and processing. Modeling human facial expressions with geometry video would be highly desirable from the transferring and processing point of view. However, the existing GV techniques (e.g. Briceño *et al.* [55]) applied only to datasets that are created by the animators, of which the correspondences among frames are available and the data are usually simple and noise-free. As a result, the user only needs to parameterize one frame, and the remaining frames can be easily induced by the given correspondence. Unfortunately, as mentioned above, such correspondence is not available for the 3D motion data acquired by 3D camera. Thus, it is usually very challenging to construct GVs for real-world datasets.

To solve the aforementioned challenges and promote GV to real-world applications, this chapter presents a novel framework that can capture high-resolution 3D facial expressions in a less restrictive manner, store the recorded data in a well organized way, and allow users to manage, manipulate, and render the data easily. Given the captured expression data, GV first analyzes the geometry and detects salient features, and then parameterizes the motion data to a rectangular domain such that the detected features in all frames can be mapped consistently. Finally, the parameterized motion data are converted into a video format such that the well-developed video processing techniques can be used to the motion data.

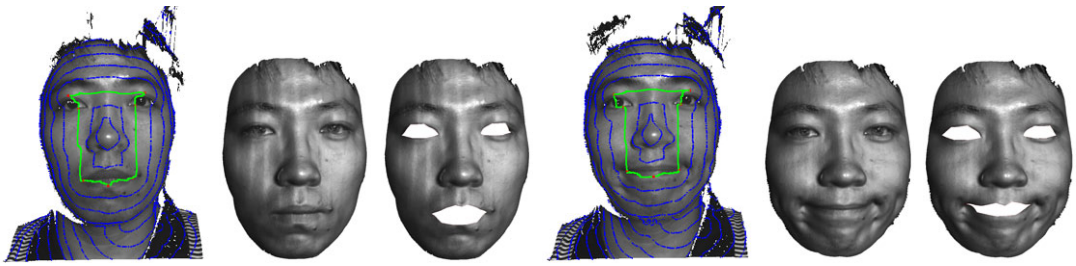
The specific contributions of this work include:

- We propose an expression-invariant parameterization algorithm to map expression data into a canonic domain. Our algorithm provides the exact correspondence of the salient features (eyes, mouth, and nose) among frames which is highly desirable in geometry video processing.

- we present a set of pre-processing algorithms, including hole filling, background segmentation and geodesic-based face salient feature segmentation. Our efficient and robust algorithms is able to provide expression data in a fast and well designed manner.

- Through a quantitative comparison to some existing methods, such as discrete Ricci flow and harmonic map, we demonstrate that the proposed approach can guarantee the exact feature correspondence among frames, which is highly desirable from the analysis point of view.

5.2 3D Motion Data Acquisition and Pre-processing



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

Figure 5.2: Face segmentation using geodesic mask. Human expressions are approximate isometry, thus, the geodesic distance is independent of the expressions. We first compute a geodesic mask from the detected features on mouth and eyes (see (a) and (d)), then segment the front face by the user-specified radius (see (b) and (e)). Finally, we remove the mouth and eyes (see (c) and (f)).

We employ the structure light-based 3D camera system [88] to capture the moving objects in real time. The system contains a video camera and a structured light projector. The projector projects digital fringe patterns composing of vertical straight stripes to the object. The stripes are deformed due to the surface profile. Then a high-speed CCD camera synchronized with the projector captures the distorted fringe image. Finally, by analyzing the fringe images, the 3D information is obtained based on the deformation

using triangulation. The system is able to capture the geometry and texture of the moving objects in real time. Despite of high speed, the 3D camera system is not robust due to various reasons, such as ambient light interference, occlusions, shadows, and depth discontinuity. Therefore, much efforts are needed to pre-process the captured raw data.

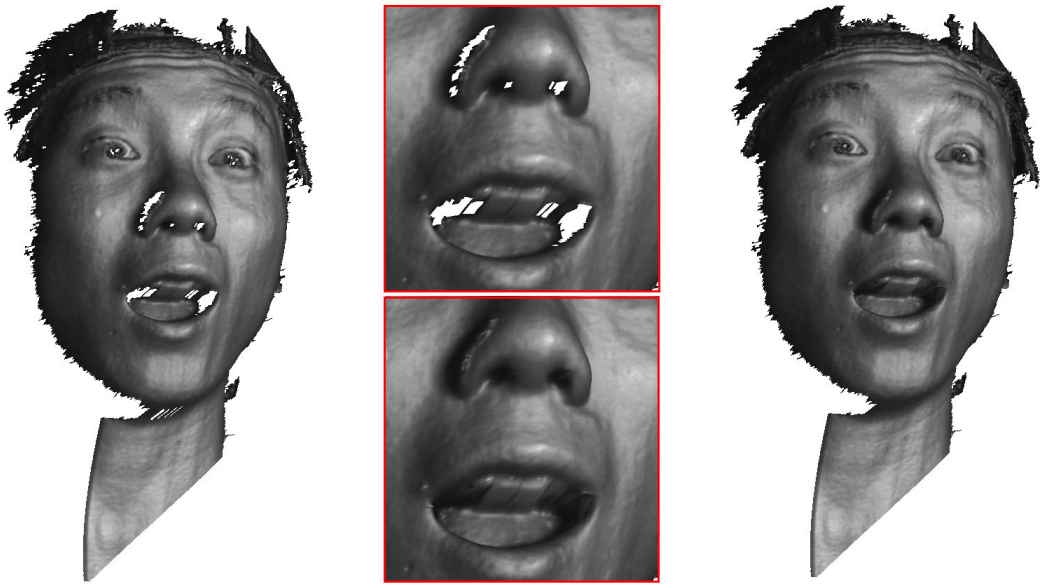


Figure 5.3: The captured raw data usually contains holes due to the occlusions. We fill both the geometry and texture of the holes by constructing a minimal surface which has C^1 continuity along the hole boundaries.

Feature tracking We first project the captured 3D expressions to 2D images, and then denote the salient features, including nose tip, eyes, mouth, eyebrows, etc, on the first frame. Next we use Active Appearance Model (AAM) [10] to track the feature points for the remaining frames automatically. Finally, the 2D feature points are mapped back to the 3D meshes.

Hole filling The captured raw data is a genus-0 open surface M. Let $\partial M = \gamma_0 \cup \gamma_1 \cup \cdots \gamma_k$ denote boundaries where γ_0 is the outer boundary and γ_i, $i \geq 1$ the interior

holes. To fill the hole γ_i, we construct a minimal surface H_i that satisfies the following Laplacian equation:

$$\triangle v = 0, \ \ \forall v \notin \partial H_i$$

with boundary conditions

$$v|_{\partial H_i} = v|_{\gamma_i}$$
$$\bigtriangledown v|_{\partial H_i} = \bigtriangledown v|_{\gamma_i}.$$

The boundary conditions guarantee that the filled surface is of C^1 continuity along γ_i, thus, leads to visually pleasing results. Note that we can also fill the colors using the same equation except that the vertex position (x,y,z) is replaced by the color (r,g,b). Figure 5.3 shows the hole filling results.

Face segmentation The captured raw data contains not only the 3D faces, but also some unnecessary information, such as cloth, hair, and background. Observe that human expressions are approximate isometry, thus, the intrinsic properties, such as Gaussian curvature, first fundamental form, geodesic, conformal factor, etc, which are invariant under isometry, can be used to segment the face. In our framework, we adopt the geodesic since it is fairly easy to compute and highly robust to the mesh resolution and triangulation. With the eyes and mouth as source points, we compute the "multiple-sources all-destinations" geodesic using the modified Xin and Wang's algorithm [85] which takes only a few seconds for each frame. Then we segment the facial expressions using the user-specified radius. Finally, we remove the eyes and mouth. As shown in Fig. 5.2, our method leads to highly consistent segmentation results.

Input:
$M \in \mathbb{R}^3$, the input 3D facial expression of genus-0 mesh with four boundaries, $\partial M = \gamma_0 \cup \cdots \cup \gamma_3$;
$D \in \mathbb{R}^2$, the parametric domain with the same topology of M;
Output:
The one-to-one map $\phi : M \to D$ such that the salient features (eyes, mouth and nose) are mapped to the corresponding features on D.

1. *Compute the geodesic c between* γ_1 *and* γ_2.
2. *Compute the geodesic d from the middle point of c to* γ_3.
3. *Cut M along c and d, the resulted mesh* $\overline{M}$ *is of genus-0 with 2 boundaries.*
4. *Process the parametric domain D in the similar way (as steps 1 to 3). Let* $\overline{D}$ *denotes the processed mesh of genus-0 with 2 boundaries.*
5. *Compute the harmonic function* $f : \overline{M} \to \mathbb{R}$ *with Dirichlet boundary condition,* $\triangle f = 0$, $f|_{\partial \overline{M}_0} = 0$, $f|_{\partial \overline{M}_1} = 1$
6. *Compute the harmonic function* $g : \overline{D} \to \mathbb{R}$ *with Dirichlet boundary condition,* $\triangle g = 0$, $g|_{\partial \overline{D}_0} = 0$, $g|_{\partial \overline{D}_1} = 1$
7. *Parameterize* $\partial \overline{M}_1$ *and* $\partial \overline{D}_1$ *by the arc length parametrization,* $h : \partial \overline{M}_1 \to \partial \overline{D}_1$.
8. *For each point* $v \in \partial \overline{M}_1$
8.1 *Trace the integral curve* $\alpha \in \overline{M}$ *of the gradient vector field* ∇f.
8.2 *Trace another integral curve* $\beta \in \overline{D}$ *starting from* $h(v) \in \partial \overline{D}_1$ *and following the vector field* ∇g.
8.3 *Construct the one-to-one map* $\overline{\phi} : \overline{M} \to \overline{D}$ *as* $\overline{\phi}(\alpha) = \beta$
9. *The parametrization* $\phi : M \to D$ *is induced from* $\overline{\phi} : \overline{M} \to \overline{D}$.

Algorithm 1: Expression-invariant 3D face parametrization

5.3 Expression-invariant Parameterization

Expression-Invariant Parameterization Algorithm

In each frame of the captured motion data, the geometry is given in the reference system of the scanner, and it is not registered in object space, and correspondences between points in different frames are not available. From the analysis and editing point of view, it is highly desirable to find the correspondence among the captured data. Motion data parametrization serves this purpose by mapping all frames to a parametric domain and then re-sample the data on the domain.

Although there are large amount of literatures in surface parametrization [21] [68], there is little work on the motion data parametrization. The key challenging in motion data parametrization is that it must take the temporal coherence into consideration, i.e., the features in all frames should be mapped consistently to the parametric domain.

This section presents a novel algorithm to parameterize the 3D facial expression data. The proposed algorithm is guaranteed to be bijective and the salient facial features (such as mouth, nose and eyes) are mapped consistently onto the parametric domain.

The input of our algorithm is a sequence of genus-0 meshes with four boundaries. Let us denote by M the mesh and $\partial M = \gamma_0 \cup \cdots \cup \gamma_3$ the boundaries, where γ_0 is the boundary of the human face, γ_1 and γ_2 are the two eyes and γ_3 is the mouth. We define the parametric domain $D \in \mathbb{R}^2$ as a rectangle with three holes, as a result, D has the same topology of M.

We first compute the geodesic c between two eyes, i.e., γ_1 and γ_2. Then we compute the geodesic d from the middle point of c to the mouth γ_3. Note that geodesic is an intrinsic property, thus, independent of the expressions which are approximate isometry. By slicing the mesh along the geodesics c and d, the number of boundaries is reduced to 2.

The resulted mesh $\overline{M}$ is a genus-0 mesh with two boundaries, i.e., γ_0 and $\gamma_1 \cup \gamma_2 \cup \gamma_3 \cup c \cup d$. In the following, we use $\partial\overline{M}_0$ and $\partial\overline{M}_1$ to denote the two boundaries of $\overline{M}$.

Then we compute the harmonic function $f : \overline{M} \to \mathbb{R}$, $\triangle f(v) = 0, \quad \forall v \notin \partial\overline{M}$, with Dirichlet boundary condition:

$$\begin{aligned} f(v) &= 0, \quad \forall v \in \partial\overline{M}_0, \\ f(v) &= 1, \quad \forall v \in \partial\overline{M}_1. \end{aligned}$$

Since the function f is harmonic, all its local extrema are on the boundaries. Furthermore, the mesh $\overline{M}$ is of genus-0 with two boundaries. According to Morse theory, f has no critical point (the point with vanishing gradient) inside $\overline{M}$. Therefore, the gradient vector field ∇f has no singularity. The integration curve of ∇f is a curve such that the tangent vector to the curve at any point v along the curve is precisely the vector $\nabla f(v)$. In the Appendix, we show that each integral curve has unique ending points, one on $\partial\overline{M}_0$, and the other on $\partial\overline{M}_1$. Furthermore, any two integral curves do not intersect.

We process the parametric domain D in the same way and let $\overline{D}$ denote the sliced mesh with two boundaries. We compute the harmonic function $g : \overline{D} \to \mathbb{R}$ with the same boundary condition as f. We also construct a bijective map between two boundary curves $h : \partial\overline{M}_1 \to \partial\overline{D}_1$ by arc-length parametrization.

Then the parametrization $\phi : \overline{M} \to \overline{D}$ is constructed as follows: For each vertex $v \in \partial\overline{M}_1$, trace the integral curve $\alpha \in \overline{M}$ following the gradient ∇f. Then, starting from $h(v) \in \partial\overline{D}_1$, trace another integral curve $\beta \in \overline{D}$. Thus, we build a one-to-one map between two integral curves α and β. By going through every point $v \in \partial\overline{M}_1$, we build the one-to-one map between $\overline{M}$ and $\overline{D}$ which in turn induces a one-to-one map $\phi : M \to D$.

Remark In the parametrization algorithm, we cut the 3D face along the geodesics connecting the three holes (i.e., eyes and mouth). So the resulted mesh is of genus-0 with 2 boundaries. The inner boundary $\gamma_1 \cup \gamma_2 \cup \gamma_3 \cup c \cup d$ is invariant to the expressions, thus, highly consistent among all frames. Furthermore, the outer boundary is determined by a geodesic mask with the user-specified radius applied to all expressions. Thus, the outer boundary is also invariant to the expression. Observe that the harmonic function is intrinsic to the geometry and independent of the expressions. As a result, the proposed parametrization is invariant to the expression. Furthermore, as proven in the Appendix, the inner boundary of $\overline{M}$ is mapped to the inner boundary of $\overline{D}$ precisely, thus, guarantees the exact correspondence of salient features, such as eyes, mouth and nose. As shown in Fig. 5.4, two expressions are parameterized consistently using our approach.

The fundamental proof of expression-invariant parameterization

We also give the concrete proof that the proposed parametrization algorithm leads to a one-to-one map and guarantees the exact boundary correspondence.

Theorem Given two surfaces A and B, both are of genus-0 with 2 boundaries (Figure 5.5). Let A_i, B_i, $i = 1,2$ denote the boundaries, i.e., $\partial A = A_0 \cup A_1$ and $\partial B = B_0 \cup B_1$. Define harmonic function $f : A \to \mathbb{R}$ on A, $\triangle f = 0$, with Dirichlet boundary condition, $f|_{A_0} = 0$ and $f|_{A_1} = 1$.

Let $C_f : A_0 \times [0,1] \to A$ be the integral curve of the gradient field ∇f such that given an arbitrary point $v_0 \in A_0$, $C_f(v_0,0) = v_0$, $C_f(v_0,1) = v_1$ and $C_f(v_0,t) = v_t$, where $v_1 \in A_1$ is the other ending point and $v_t \in M$ is the point satisfying $f(v_t) = t$. Similarly, we define the harmonic function on B, $g : B \to \mathbb{R}$ and the integral curve $C_g : B_0 \times [0,1] \to B$.

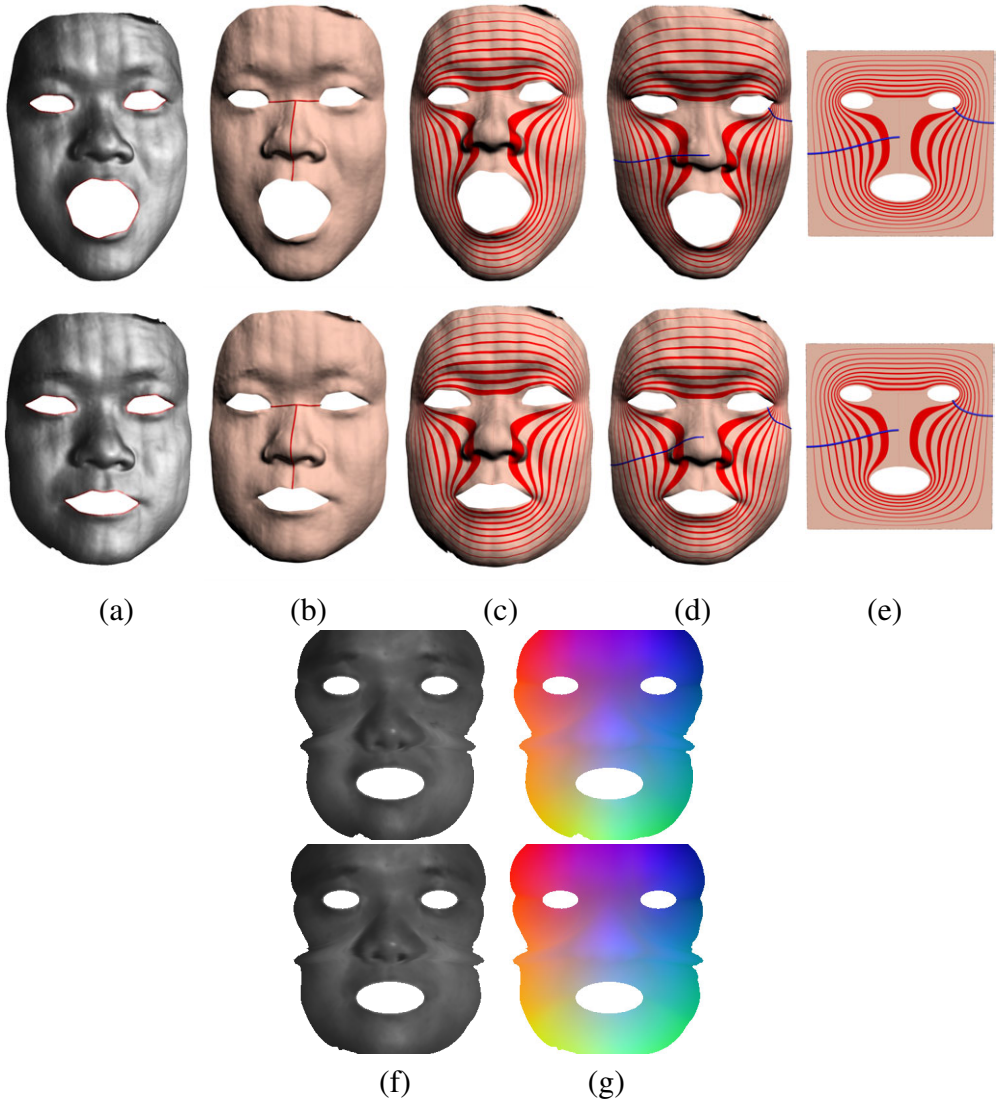


Figure 5.4: Expression-invariant parametrization. (a) Input mesh M. (b) Geodesics connecting the eyes, nose and mouth. (c) Harmonic function with Dirichlet boundary condition. As human facial expressions are approximate isometric transformation, the computed harmonic functions are insensitive to expressions. (d) Integral curves follow the gradient of the harmonic function. (e) Integral curves on the parametric domain. (f) The integral curves induce the parametrization between M and D, which guarantees the exact correspondence of the eyes, mouth and nose. As a result, different expressions have very similar parameterization. (g) Converting the parameterized mesh to geometry image, in which the pixel color (r,g,b) encodes the vertex position (x,y,z).

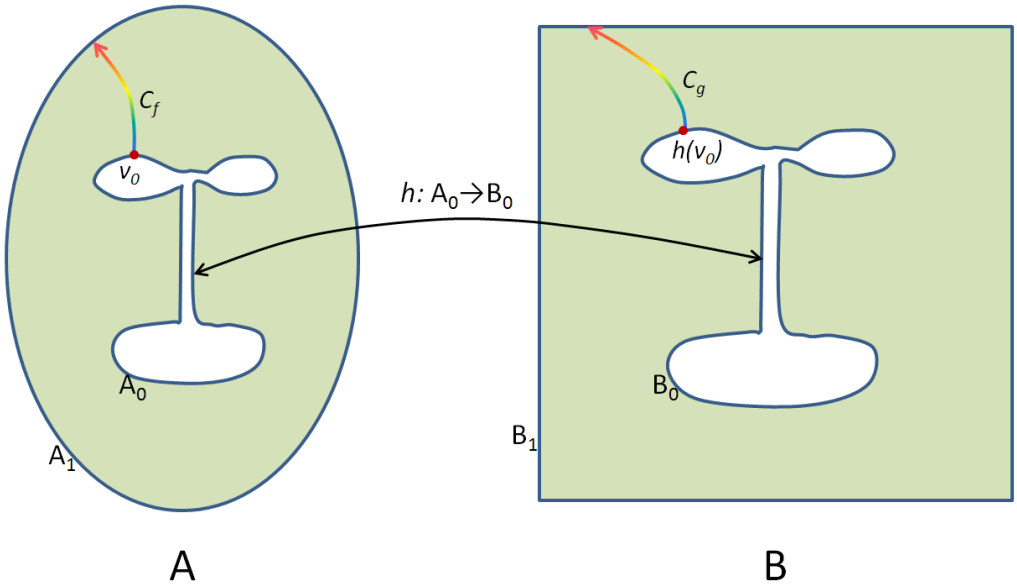


Figure 5.5: proof of expression-invariant parameterization

Define a homeomorphic map $h : A_0 \rightarrow B_0$ and construct the parametrization $\phi : A \rightarrow B$ by mapping the integral curve $C_f(v_0, \cdot)$ to $C_g(h(v_0), \cdot)$ for $\forall v_0 \in A_0$.

Then the map ϕ is one-to-one and ϕ maps the inner boundary A_0 to B_0, i.e., $\phi(A_0) = B_0$.

Proof First, we show no integral curve of ∇f or ∇g form a loop, since f or g is smooth function and its gradient vector field is curl-free.

Second, we show that no integral curve that starts and ends on the same boundary curve. The function f or g is harmonic function that does not have critical points (where the gradient vanishes) inside the surface. Thus, the function value is strictly monotonic along the integral curve. Note that all points on the same boundary curve have the same function value, so the ending points of each integral curve must be on different boundary curve.

Third, we show that two integral curves do not intersect. Without loss of generality, assume two integral curves $\gamma_1 \in A$ and $\gamma_2 \in A$ intersect at a point p. Then p is a critical point and the gradient ∇f vanishes at p. We consider two cases:

Case 1: $p \notin \partial A$ is an interior point. Since f is harmonic, the maximum and minimum must be on the boundaries. Therefore the Hessian matrix at p has negative eigenvalue values. Suppose $f(p) = s$, then according to Morse theory, the homotopy types of the level sets $f^{-1}(s-\varepsilon)$ and $f^{-1}(s+\varepsilon)$ will be different. At all the interior critical points, the Hessian matrices have negative eigenvalues, the homotopy type of the level sets will be changed. Therefore, the homotopy type of A_0 is different from that of A_1. This contradicts the given condition, as A_0 is homotopic to A_1.

Case 2: $p \in \partial A$ is on the boundary. Without loss of generality, say $p \in A_0$. Then we can glue two copies of the same surface, along A_0. And reverse the gradient field of one surface. The union of the two gradient fields gives us a harmonic function field. Then there is no interior critical point on the doubled surface. p becomes one interior critical point, that leads to a contradiction.

Therefore γ_1 and γ_2 have no intersection points anywhere.

Last, we show ϕ is one-to-one. From the above, we know that for an arbitrary interior point, there is a unique integral curve passing through and intersecting on the inner and outer boundaries. The two ending points are also unique. Furthermore, the given boundary map $h : A_0 \to B_0$ is homeomorphic, thus, it induces a homeomorphism between integral curves in A and B, $C_f(v_0, \cdot) \to C_g(h(v_0), \cdot)$. The boundary A_0 is mapped to B_0 because the map ϕ restricted on A_0 is the boundary map $h : A_0 \to B_0$, i.e., $\phi|_{A_0} = h$.

5.4 Experimental Results

In this section, we present the experimental results of our expression-invariant parameterization method.

Test sequence capturing There are 4 test video sequences containing human expressions captured by our camera from 4 subjects, who were asked to make varied face expressions, e.g. laughing, shouting, etc., during the video capturing. These 4 sequences are named as "HumanFace1" to "HumanFace4".

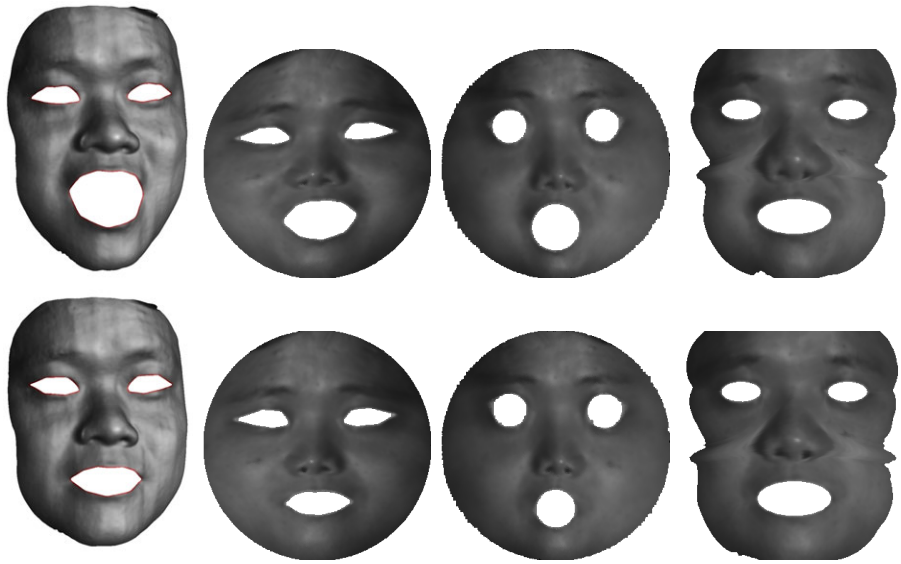
Experimental setups We test our parameterization method with a workstation equipped with quad-core Xeon 2.50GHz CPU, 8GB memory and Windows XP 64-bit system.

Tested methods The evaluations were based on our proposed expression-invariant parameterization (EIP for short) method, and employed the test conditions described in previous section. For compression purpose, we have tested 2 other parameterization methods including the harmonic map (HM for short) method (without salient feature alignment) and the ricci flow (RF for short) method.

Performance With the mentioned setup, the proposed expression-invariant parameterization method achieved the speed of 60 frames per hour. Each frame takes 60 seconds, in which 55 seconds for the pre-processing stage and 5 seconds for the mapping stage. Note that our current implementation is single threaded and that the main part of parameterization is highly parallelizable, our algorithm has strong potentials in speeding up with multi-core CPU or GPU.

Comparisons to other parameterization methods At last, we present the evaluation results on comparing our method to some other parameterization algorithms, e.g. the harmonic map (HM) method Wang *et al.* [81] and the discrete Ricci flow (RF) method [27]. Harmonic map method parameterizes the given surface to a convex domain, such as a unit disc. Discrete Ricci flow method computes a flat metric and then embeds the given surface to a multiply connected domain, such as a multi-hole annulus. Note that both HM and RF do not consider the salient feature correspondence of the

input model, thus, the eyes, nose and mouth are mapped to different locations for different frames, as shown in Fig. 5.6(b)-(c). In other words, HM and RF are not invariant to the expressions. Our method, in sharp contrast to HM and RF, is invariant to the expressions and can guarantee the exact feature correspondence among different frames. As shown in Fig. 5.6(d), our method can lead to results that are highly consistent and insensitive to the expressions.



(a) Input (b) Harmonic Map (c) Ricci Flow (d) Our method

Figure 5.6: Compared to other parameterization techniques, such as Ricci flow [27] and harmonic map [81], our algorithm leads to the results that are highly consistent and insensitive to the expressions.

Chapter 6

Volume Parameterization Using Green's Function

6.1 Motivation

The past decade has witnessed the great advances of surface parameterizations, evident in a wide range of applications pertaining to science and engineering. Despite these successes, most real-world objects are, in fact, volumes rather than surfaces. It remains both unclear and challenging as to how existing surface parameterization methods can be generalized from surfaces to volumes. With volume parameterization, we envision a large pool of applications that can benefit from the results, including solid texture mapping, volumetric tetrahedralization for simulation, and volumetric registration.

Due to the intrinsic differences between surfaces and volumes, many classical results on surface parameterization cannot be directly transposed to volume parameterization. For example, it is well known that a harmonic map between a topological disk (a genus zero surface with a single boundary) and a planar convex domain is diffeomorphic (i.e., bijective and smooth), if the boundary map is homeomorphic (i.e., bijective and continuous). This result plays an important role in surface parameterization. Unfortunately, such an approach is not applicable to volumes, i.e., a volumetric harmonic map is not

guaranteed to be bijective even though the target domain is convex. In this work, we aim at overcoming the challenges by proposing a theoretically sound algorithm that can produce a diffeomorphism between two star-shaped volumes.

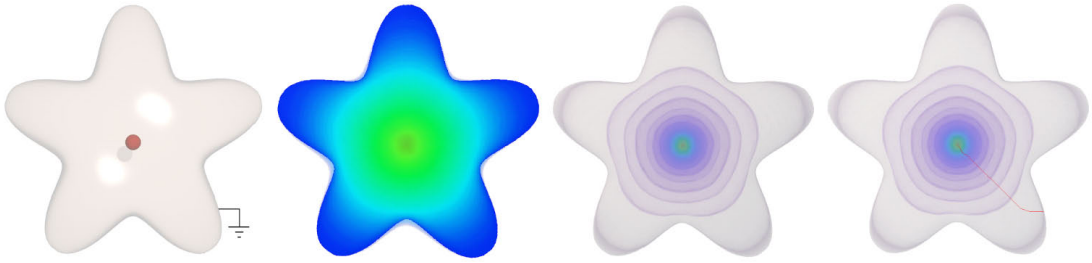
Our volume parameterization method is strongly motivated by the property of electric field. Given a closed genus-0 metal surface S, let M denote its interior volume, i.e., $\partial M = S$. We construct an electric field by putting a positive electric charge at a point c inside M, and connecting the boundary surface S to the ground. The *electric potential* inside M is a *Green's function*, $G : M \to \mathbb{R}$, such that

$$\begin{cases} \Delta G(x) &= \delta(x-c) \\ G|_{\partial M} &\equiv 0, \end{cases} \qquad \text{(Eq. 6.1)}$$

where $\delta(x-c)$ is the Dirac function. In general, the level set of G, $G^{-1}(r), r \in \mathbb{R}^+$ or an isopotential surface is a smooth surface in M. The gradient of the electric potential ∇G is the *electric field*. *Electric field lines* are the integration curves of the electric field, i.e., the tangent vectors of the electric field lines are parallel to the electric field. Different electric field lines only intersect at the points where we put the electric charge, or at the critical point of the potential. Electric field lines start from the electric charge and are orthogonal to the iso-potential surfaces everywhere, in particular to the boundary surface ∂M.

If M is a star-shaped volume, every ray cast from c intersect S only once, and there are no other critical points of the potential. Therefore, all the iso-potential surfaces can be topological spheres and all electric field lines intersect only at point c (see Figure 6.1). Since each point inside M is now uniquely determined by a corresponding electric field line and an iso-potential surface, determining the map between two star-shaped volumes is equivalent to constructing the map between the corresponding iso-potential surfaces and the electric field lines. Therefore, we map the boundary surface to the unit sphere, thereby putting each iso-potential surface to a concentric sphere, the electric field lines

to the radii, and the center **c** to the origin. In this way, the star-shaped volume can be parameterized to the unit solid ball. With the help of ball parameterization, the map between two star shapes can then be constructed by mapping each shape to the unit ball and constructing a bijective map between the two unit balls. Such a constructed volumetric map is guaranteed to be a diffeomorphism.



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

Figure 6.1: Electric field on the star shape. Given a metal surface S, we put a positive charge at the center (the red point) and then connect S to the ground (shown in (a)). The electric field is a Green's function shown in (b). If the surface S is star-shaped, then the Green's function has a unique critical point. As a result, all iso-potential surfaces are topological spheres (shown in (c)). The electric field line (red curve in (d)) is perpendicular to all iso-potential surfaces.

6.2 Theoretical Foundation

6.2.1 Harmonic Maps

Let M be a manifold with a Riemannian metric. Suppose a function $f : M \to \mathbb{R}$ is defined on the manifold. The *harmonic energy* of f is defined as

$$E(f) = \int_M |\nabla f|^2 dv.$$

The critical point of the harmonic energy is called a *harmonic function*, which satisfies the following Laplace equation,

$$\Delta f = 0,$$

where Δ is the Laplace-Beltrami operator on M. Suppose $\mathbf{f}$ is a map from M to a domain $D \subset \mathbb{R}^n$ in the Euclidean space, $\mathbf{f} : M \to D$, and each component of $\mathbf{f}$ is a harmonic function, then $\mathbf{f}$ is called a *harmonic map*. The following *heat flow* method can deform a map $\mathbf{f}$ to a harmonic map with fixed boundary conditions:

$$\frac{d\mathbf{f}}{dt} = -\Delta\mathbf{f}.$$

The following theorem lays down the theoretic foundation for harmonic surface parameterizations,

Theorem 6.1 (Radó) *Suppose M is a simply connected surface with a Riemannian metric, a harmonic map $\mathbf{f} : M \to D \subset \mathbb{R}^2$ maps M to a convex planar domain D, the restriction of $\mathbf{f}$ on the boundary $\mathbf{f}|_{\partial M} : \partial M \to \partial D$ is a homeomorphism. Then $\mathbf{f}$ is a diffeomorphism in the interior of M.*

Unfortunately, this theorem does not hold for volumetric harmonic maps.

6.2.2 Green's Functions on Star Shapes

This section introduces the theoretic foundation of star shape parameterization. The proofs are also given here.

A volume M is called a *star shape* if there exists a point $\mathbf{c} \in M$ such that any ray cast from $\mathbf{c}$ intersects the boundary of M only once. The point $\mathbf{c}$ is called the center of M. In particular, any convex volume is a star shape, where any interior point can serve as the center $\mathbf{c}$. From the implementation point of view, computing the intersection of a ray with a surface is typically computationally expensive. Thus, we use an alternative approach to define a star shape:

Lemma 6.1 *A volume M is a star shape if and only if there exists a point $\mathbf{c} \in M$ such that for any boundary point $\mathbf{p} \in \partial M$,*

$$(\mathbf{c}-\mathbf{p}, \mathbf{n}(\mathbf{p})) \leq 0, \qquad \text{(Eq. 6.2)}$$

where $\mathbf{n}(\mathbf{p})$ is the normal vector at $\mathbf{p}$ and $(,)$ is the dot product. The boundary ∂M should be smooth such that the normal vector exists on any boundary point $\mathbf{p}$.

Proof Assume the boundary surface ∂M is represented by the zero level set of an implicit function $f : \mathbb{R}^3 \to \mathbb{R}$, i.e., $\partial M = f^{-1}(0)$ and the interior points $\mathbf{r} \in M$ satisfy $f(\mathbf{r}) < 0$.

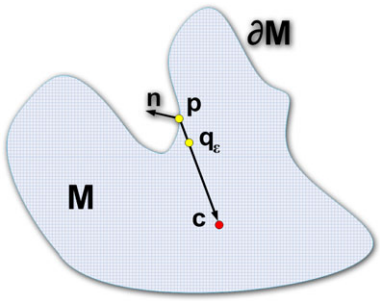


Figure 6.2: proof for lemma 6.1

($\Longrightarrow$ necessary condition) If M is a star shape, for any boundary point $\mathbf{p} \in M$, the ray $\mathbf{p}-\mathbf{c}$ intersects ∂M only once and the intersection point is $\mathbf{p}$. Thus, for any $\varepsilon \in [0,1]$, the point $\mathbf{q} = \mathbf{p} + \varepsilon(\mathbf{c}-\mathbf{p}) \in M$ is inside M. Then, for a small $\varepsilon > 0$,

$$f(\mathbf{q}) = f(\mathbf{p}) + \varepsilon \bigtriangledown f(\mathbf{p}) \cdot (\mathbf{c}-\mathbf{p}) + O(\varepsilon^2 \|\mathbf{c}-\mathbf{p}\|^2).$$

Note that $f(\mathbf{p}) = 0$ and $f(\mathbf{q}) \leq 0$, thus, $\bigtriangledown f(\mathbf{p}) \cdot (\mathbf{q}-\mathbf{p}) \leq 0$. Since $\bigtriangledown f(\mathbf{p})$ points to the normal direction $\mathbf{n}(\mathbf{p})$, and $\mathbf{c}-\mathbf{p}$ has the same direction as $\mathbf{q}-\mathbf{p}$, then $(\mathbf{c}-\mathbf{p}, \mathbf{n}(\mathbf{p})) \leq 0$.

($\Longleftarrow$ sufficient condition) Given a point $\mathbf{c} \in M$, for every boundary point $\mathbf{p}$, $(\mathbf{c}-\mathbf{p}, \mathbf{n}(\mathbf{p})) \leq 0$ holds. Assume M is not a star shape, then there exists a ray from $\mathbf{c}$ which

intersects ∂M at least twice. Without loss of generality, say $\mathbf{p}_1$ and $\mathbf{p}_2$ are the first two intersection points and $\mathbf{p}_1$ is closer to $\mathbf{c}$. Consider a point $\mathbf{q} = \varepsilon(\mathbf{p}_1 - \mathbf{p}_2) + \mathbf{p}_2$ with $\varepsilon > 0$. Clearly, $\mathbf{q}$ is on the segment $\mathbf{p}_1\mathbf{p}_2$ and out of M. Thus, $f(\mathbf{q}) > 0$. Using Taylor expansion,

$$f(\mathbf{q}) = f(\mathbf{p}_2) + \varepsilon \bigtriangledown f(\mathbf{p}_2) \cdot (\mathbf{p}_1 - \mathbf{p}_2) + O(\varepsilon^2 \|\mathbf{p}_1 - \mathbf{p}_2\|^2).$$

Note that $f(\mathbf{p}_2) = 0$ and $\bigtriangledown f(\mathbf{p}_2)$ points to the same direction as normal $\mathbf{n}(\mathbf{p}_2)$, $\mathbf{p}_1 - \mathbf{p}_2$ points to the same direction as $\mathbf{c} - \mathbf{p}_2$, thus, $\bigtriangledown f(\mathbf{p}_2) \cdot (\mathbf{p}_1 - \mathbf{p}_2) \leq 0$ and $f(\mathbf{q}) \leq 0$, contradiction! Q.E.D.

Lemma 6.2 (Green's function on a star shape) *Suppose M is a star shape with a center $\mathbf{c} \in M$, G is the Green's function (see Eqn. (*Eq. 6.1*)) with a pole at $\mathbf{c}$, then $\mathbf{c}$ is the only critical point of G.*

Proof Without loss of generality, we assume c is at the origin in $\mathbb{R}^3$. Let $B(c, \varepsilon)$ be a small ball centered at c with radius ε. Consider the following function, the inner product of the point $p = (x_1, x_2, x_3)$ and the gradient of G at p,

$$f(p) = (p, \nabla G) = x_1 \frac{\partial G}{\partial x_1} + x_2 \frac{\partial G}{\partial x_2} + x_3 \frac{\partial G}{\partial x_3}.$$

By direct computation, it is easy to verify that

$$\Delta f = (\sum_k \frac{\partial^2}{\partial x_k^2})(\sum_i x_i \frac{\partial G}{\partial x_i}) = 0.$$

In details,

$$\frac{\partial^2}{\partial x_k^2}(\sum_i x_i \frac{\partial G}{\partial x_i}) = 2\frac{\partial^2 G}{\partial x_k^2} + \sum_i x_i \frac{\partial^3 G}{\partial x_k^2 \partial x_i},$$

therefore

$$(\sum_k \frac{\partial^2}{\partial x_k^2})(\sum_i x_i \frac{\partial G}{\partial x_i}) = 2\Delta G + \sum_i x_i \Delta \frac{\partial G}{\partial x_i}.$$

Because G is harmonic, therefore, $\frac{\partial G}{\partial x_i}$ is also harmonic, and the above equation equals zero.

Therefore $f(p)$ is a harmonic function on $M/B(c,\varepsilon)$. According to the maximum principle of harmonic maps, f reaches its max and min values on the boundary surfaces ∂M and $\partial B(c,\varepsilon)$. Here by definition, and c is the pole of f, f is negative on $\partial B(c,\varepsilon)$. Because M is a star shape, on ∂M, $(\mathbf{n},p) > 0$, where $\mathbf{n}$ is the normal on p to ∂M. ∇G is orthogonal to ∂M and is on the opposite direction of $\mathbf{n}$. Therefore, f is always negative in the whole volume $M/B(c,\varepsilon)$, ∇G is non-zero in $M/B(c,\varepsilon)$. Since ε is arbitrary, ∇G is non-zero for all points in $M/\{c\}$. We conclude that G has no critical points in M except c. Q.E.D.

Lemma 6.3 *Suppose M is a star shape with a center $\mathbf{c} \in M$, G is the Green's function with a pole at $\mathbf{c}$. Then for any $r \in \mathbb{R}^+$, the level set $G^{-1}(r)$ is topologically equivalent to a sphere.*

Proof Let $r \in \mathbb{R}^+$, $G^{-1}(r)$ is the level set of G. $G^{-1}(0)$ is the boundary of M, ∂M, which is a topological sphere. By lemma 2, there is no critical points in $G^{-1}([0,r])$. According to Morse theory, $G^{-1}(r)$ and $G^{-1}(0)$ share the same topology. In fact, we can start from a point $p \in G^{-1}(r)$ and trace along the integration curve of the gradient of G and reach a unique point q on $G^{-1}(0)$, this gives us a diffeomorphism from $G^{-1}(r)$ to $G^{-1}(0)$. Q.E.D.

Theorem 6.2 *Suppose M and $\tilde{M}$ are star-shaped volumes with centers $\mathbf{c}$ and $\tilde{\mathbf{c}}$, G and $\tilde{G}$ are Green's functions with poles at $\mathbf{c}$ and $\tilde{\mathbf{c}}$, respectively. If the boundary map $\partial M \to \partial\tilde{M}$ is a diffeomorphism, then the map $f : M \to \tilde{M}$ induced by G and $\tilde{G}$ is also a diffeomorphism.*

Proof We first introduce the concepts of *foliation* and *leaf*.

A dimension *m foliation* of an n-dimensional manifold M is a covering by charts U_i together with maps $\phi_i : U_i \to \mathbb{R}^n$, such that on the overlaps $U_i \cap U_j$, the transition functions $\phi_{ij} = \phi_j \circ \phi_i^{-1}$ take the form

$$\phi_{ij}(x,y) = (\phi_{ij}^1(x), \phi_{ij}^2(x,y))$$

where x denotes the first $n-m$ coordinates, y denotes the last m coordinates. In each chart U_i the $x = const$ stripes match up with the stripes on U_j. The stripes piece together from chart to chart to form maximal connected injectively immersed submanifolds called the *leaves*.

The we show the proof. Let F_1 be the foliation of M by topological spheres induced by the level sets of G, F_2 be the foliation of M induced by the gradient lines of G. We choose an open cover of $M/\{c\}$, $\{(U_\alpha, \phi_\alpha)\}$, U_α is the union for leaves in F_2, $U_\alpha = \cup f, f \in F_2$, such that $\phi_\alpha : U_\alpha \to \mathbb{R}^3$, leaves in F_1 are mapped to the planes $z = const$, leaves in F_2 are mapped to lines $(x,y) = const$. $\{(U_\alpha, \phi_\alpha)\}$ is a differential atlas. Similarly, we can construct a differential atlas of $\tilde{M}/\{\tilde{c}\}$, $\{\tilde{U}, \tilde{\phi}_\beta\}$, the level sets and the integration lines are mapped to canonical planes orthogonal to the z-axis and lines parallel to the z-axis.

The restriction of the map $f : M \to \tilde{M}$ on the local coordinate system

$$f_{\alpha\beta} = \tilde{\phi}_\beta \circ f \circ \phi_\alpha^{-1} : \phi_\alpha(U_\alpha) \to \tilde{\phi}_\beta(\tilde{U}_\beta)$$

has the following form

$$f_{\alpha\beta}(x,y,z) = (g(x,y), z),$$

where $g(x,y)$ is determined by the restriction of f on the boundary, $f|_{\partial M} : \partial M \to \partial \tilde{M}$. The restriction is a diffeomorphism, therefore $g(x,y)$ is a diffeomorphism, and $f_{\alpha\beta}$ is a

diffeomorphism. Because U_α and $\tilde{U}_\beta$ is arbitrarily chosen, f itself is a diffeomorphism. Q.E.D.

Theorem 6.2 lays down the foundation of the proposed volume parameterization framework. We should point out that even though $\mathbf{c}$ and $\tilde{\mathbf{c}}$ are poles of the Green's functions G and $\tilde{G}$, the induced map $f : M \to \tilde{M}$ is smooth everywhere including the pole $\mathbf{c}$ since we define $f(\mathbf{c}) := \tilde{\mathbf{c}}$. This can be elucidated by the physical meaning of Green's function. Consider the phenomenon of a grounded conducting surface surrounding a charged body at the center $\mathbf{c}$. The electric potential inside the volume bounded by the surface is the Green's function. If the volume is a solid ball and the center is the origin, then parameterization induced by the Green function is equivalent to the polar coordinate. The center is the pole of the polar parameterization, but the mapping between two balls induced by the polar coordinates has no singularity [82].

Remark. Gergen showed that the gradient of a Green's function in a star-shaped three dimensional region never vanishes[24]. This implies that there is no interior singularity of the Green function, hence the level sets are topological spheres, and the integration curves of the gradient field do not intersect either. This gives alternative proof for our main theoretic result.

6.3 Parameterizing Star-Shaped Volumes

This section presents the algorithmic detail of parameterizing star-shaped volumes to simple domains, such as the unit ball and star-shaped polycubes.

6.3.1 Parameterizing a Star Shape to a Ball

1: Input: S, the boundary mesh of a star-shaped volume M
2: Output: $f : M \to \mathbb{B}^3$ is diffeomorphism
3: Find the center of M;
4: Compute the Green's function on M,$G_M : M \to \mathbb{R}$;
5: Map the center $\mathbf{c}$ to the center of $\mathbb{B}^3$, $f(\mathbf{c}) = \mathbf{0}$;
6: Parameterize the boundary points by constructing a conformal spherical mapping $\phi : \partial M \to \partial \mathbb{B}^3$;
7: **for** every interior vertex $\mathbf{p} \in M$ **do**
8: Trace the integration curve γ from $\mathbf{p}$ to the boundary point $\mathbf{q} \in \partial M$;
9: Set $f(\mathbf{p}) = \frac{\phi(\mathbf{q})}{G_M(\mathbf{p})+1}$
10: **end for**

Algorithm 2: Ball parameterization of star shapes.

Step 1. Star shape verification and center detection. The input of our algorithm is a closed genus-0 surface S which encloses a volume M, i.e., $S = \partial M$. S is represented by a triangular mesh with vertices $\{\mathbf{v}_i\}_{i=1}^n$. First, we need to verify whether M is star-shaped. If it is true, we determine the center of M. Note that for a given star shape, there could be infinite possible choices for the centers and the distribution of Green's function. A badly chosen center may introduce severe bias in the volume parameterization. Thus, we prefer a geometry-aware center, where a natural choice is a center that is close to the center of mass of M. This leads to the following linear constrained quadratic programming problem:

$$\min_{\mathbf{c}} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{c}-\mathbf{v}_i\|^2$$
$$subject\ to\ (\mathbf{c}-\mathbf{v}_i, \mathbf{n}_i) \leq 0 \quad i = 1, \cdots, n\ .$$

The objective function aims to minimize the distance between the center $\mathbf{c}$ and the center of mass, where the linear constraints precisely ensure the detected center $\mathbf{c}$ satisfies the star shape requirement (see Eqn. Eq. 6.2). If M is not star-shaped, then no valid solution will be found. In our implementation, we use the MOSEK optimization software [53] to solve this quadratic programming problem.

The critical component of our ball parameterization is to trace the intergral curves from the interior vertices to the boundary. We use half-face structure to model the tetrahedral mesh. Each interior face is shared by two tetrahedral and each boundary face is only adjacent to one tetrahedron. The detailed tracing algorithm is shown below.

Input: p, a point inside the volume
ε, step length of tracing
Output: γ, the integral curve from p and following the positive direction of the gradient vector field

Find the tetrahedron currTet such that $p \in currTet$;
while $currTet\ != NULL$ **do**
  Compute the gradient vector p_v *by linear interpolation of the vertex gradients of currTet*
  $p_{next} = p + \varepsilon * p_v$
  if $p_{next} \in currTet$ **then**
    update p_v
  else
    Find the tetrahedron nextTet which contains the p_{next}
    $currTet = nextTet$
  end
  $p = p_{next}$
end
Tracing from p to the boundary

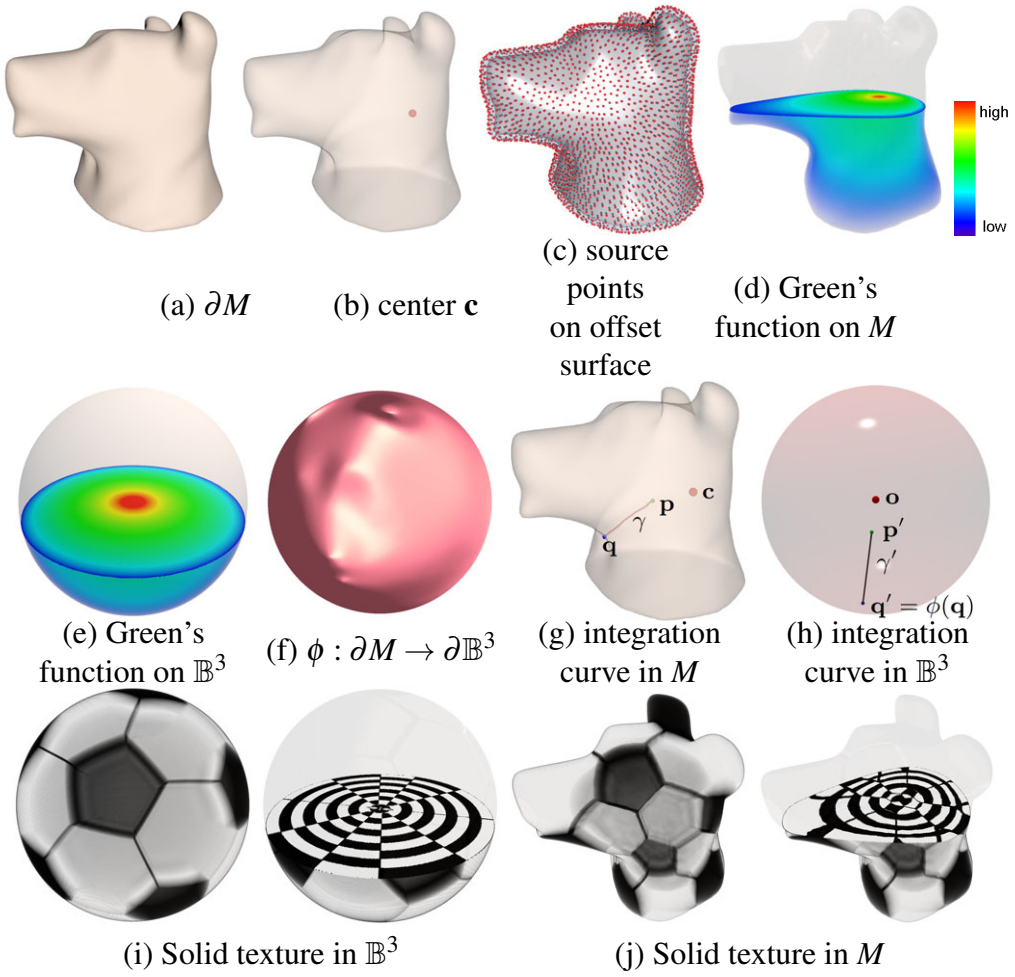
Algorithm 3: Integral Curve Tracing

Step 2. Computing Green's functions on M and $\mathbb{B}^3$. Next, we compute the Green's function on the star-shaped volume M using the method detailed in the fundamental solution [16, 47]. Suppose we have an electric charge q_i at point $\mathbf{p_i}$, the electric potential caused by q_i at point $\mathbf{r}$ is

$$K(q_i, \mathbf{p}_i; \mathbf{r}) = \frac{1}{4\pi} \frac{q_i}{|\mathbf{p}_i - \mathbf{r}|}.$$

We need to put m electric charges $\{q_i\}$ at m points $\{\mathbf{p}_i\}$ on an offset surface above the boundary surface of ∂M, such that on the boundary ∂M, the total potential equals zero,

$$G_M(\mathbf{r}) = \sum_{i=1}^{m} K(q_i, \mathbf{p}_i; \mathbf{r}) + G(1, \mathbf{c}; \mathbf{r}) = 0, \forall \mathbf{r} \in \partial M,$$

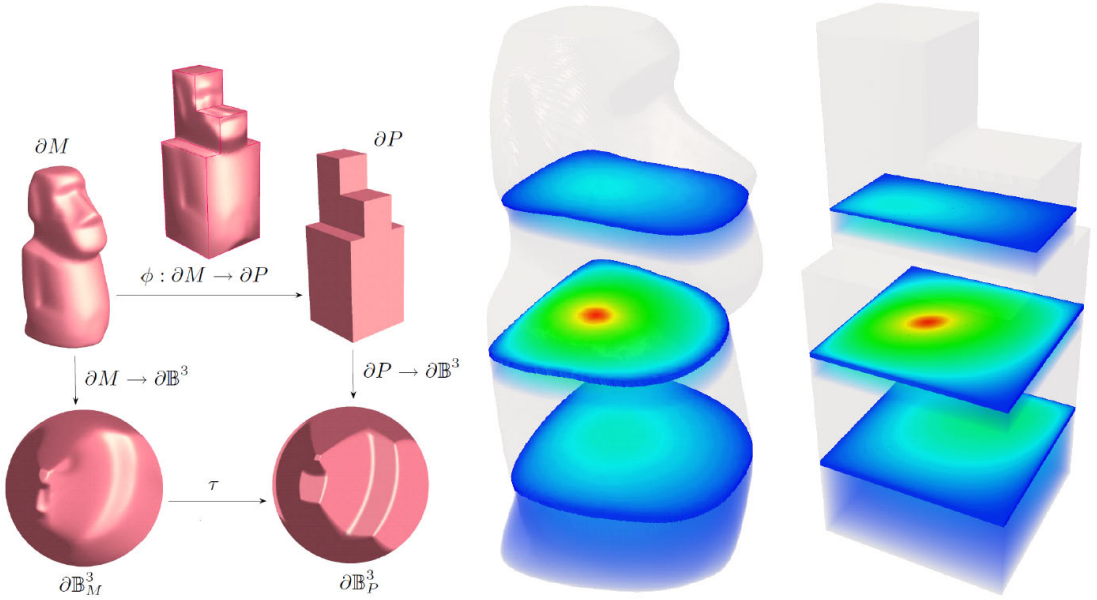


(a) ∂M (b) center $\mathbf{c}$ (c) source points on offset surface (d) Green's function on M

(e) Green's function on $\mathbb{B}^3$ (f) $\phi : \partial M \to \partial \mathbb{B}^3$ (g) integration curve in M (h) integration curve in $\mathbb{B}^3$

(i) Solid texture in $\mathbb{B}^3$ (j) Solid texture in M

Figure 6.3: Green's function induces a diffeomorphism between the star shape M and the unit ball $\mathbb{B}^3$. The input model is a triangular mesh (shown in (a)) which encloses a star-shaped volume. The red point in (b) shows the star shape center. Then we compute the Green's function on M using the fundamental solution method. (c) The source points placed on the offset surface of ∂M. (d) The Green's function on M. (e) The Green's function on $\mathbb{B}^3$ which is given by a closed form formula $\frac{1}{r} - 1$. (f) The boundary parameterization by constructing a conformal spherical map $\phi : \partial M \to \partial \mathbb{B}^3$. (g) We parameterize the interior point $\mathbf{p}$ by tracing the integration curve γ to the boundary point $\mathbf{q}$. Note that the integration curve is perpendicular to the iso-surfaces of G. (h) The image of $\mathbf{p}$ is given by $\frac{\phi(\mathbf{p})}{G_M(\mathbf{p})+1}$. (i) and (j) show the volume rendering of a soccer ball texture on M and $\mathbb{B}^3$, respectively.



(a) polycube map ϕ induces a map from $\partial \mathbb{B}^3_M$ to $\partial \mathbb{B}^3_P$ (b) Green's function on M (c) Green's function on P

Figure 6.4: Parameterizing a star shape M to a polycube P using the Green's function. We first construct the conformal polycube map $\phi : \partial M \to \partial P$. Then, we parameterize M and P to the ball using Algorithm 1. The conformal polycube map ϕ induces an identity map between $\partial \mathbb{B}^3_M$ and $\partial \mathbb{B}^3_P$. The volume parameterization is then given by $f = f_M \circ f_P^{-1}$.

where q_i's are unknowns. The equation is converted to a dense linear system, which can be solved using the singular value decomposition method provided in Matlab. As suggested in Li *et al.* [47], we place $m = 0.6n$ source points on the offset surface with offset distance equals 0.05 times the main diagonal of M.

The Green's function on $\mathbb{B}^3$ (with the origin as the center) has a closed form, $G_B(p) = \frac{1}{r} - 1$, where r is the distance from p to the origin.

Step 3. Parameterizing the boundary points. Furthermore, we compute the boundary map $\phi : \partial M \to \mathbb{B}^3$. Since the boundary of the unit ball is the sphere $\mathbb{S}^2$, the conformal spherical mapping [28] is a diffeomorphism, and thus, can serve as the boundary map.

Step 4. Parameterizing the interior points. The interior of the volume is represented by a tetrahedral mesh. We use Tetgen [70] to generate a tetrahedral mesh for a given surface mesh S to meet the boundary constraints. To improve the meshing quality, we employ the variational tetrahedral meshing technique Alliez *et al.* [2] which can significantly reduce the slivers and produce well-shaped tetrahedral meshes. The tetrahedral mesh M is represented by $M = (V, E, F, T)$ where V, E, F, and T are the vertex, edge, face, and tetrahedra sets, respectively. The Green's function G_M is represented as a piecewise linear function, $G_M : V \to \mathbb{R}$. The gradient of G can be computed as follows: suppose t_{ijkl} is a tetrahedron with vertices $\{v_i, v_j, v_k, v_l\}$, the face on the tetrahedron against vertex v_i is f_i; similarly v_j, v_k, and v_l are against f_j, f_k, and f_l, respectively. We define $\mathbf{s}_i$ to be the vector along the normal of f_i with length equal to 2 times the area of f_i, and so can $\mathbf{s}_j, \mathbf{s}_k, \mathbf{s}_l$ be defined. Then, the gradient of G_M in t_{ijkl} is a constant vector field

$$\nabla G_M = G_M(v_i)\mathbf{s}_i + G_M(v_j)\mathbf{s}_j + G_M(v_k)\mathbf{s}_k + G_M(v_l)\mathbf{s}_l.$$

We then define the vertex gradient as the average of the gradient vectors in the neighboring tetrahedra.

Finally, the parameterization from M to $\mathbb{B}^3$, $f : M \to \mathbb{B}^3$ is constructed as follows. We map the center $\mathbf{c}$ to the origin, i.e., the center of $\mathbb{B}^3$. Given an interior point $\mathbf{p} \in M$ (other than the center $\mathbf{c}$), we trace the integration curve γ of the gradient field from $\mathbf{p}$, γ intersects the boundary surface ∂M at $\mathbf{q}$, then γ corresponds to the radius of $\mathbb{B}^3$ through the point $\phi(\mathbf{q})$. Suppose the Green's function value at $\mathbf{p}$ is $G_M(\mathbf{p})$, then the image of $\mathbf{p}$ is defined by

$$f(\mathbf{p}) = \frac{\phi(\mathbf{q})}{G_M(\mathbf{p}) + 1}.$$

Figure 6.3 illustrates the pipeline of parameterizing the dog head to a solid ball. To visualize the parameterization, we design a soccer ball texture on $\mathbb{B}^3$ and then map it to the dog head. Note that the iso-parameter surfaces in M are curved, but the cut view is obtained by a cutting plane. Thus, the texture on the intersection plane in Figure 6.3(j) may look irregular.

1: Input: boundary meshes of a star shape M and a star-shaped polycube P
2: Output: $f : M \to P$ is diffeomorphism
3: Parameterize P to the unit ball $f_P : P \to \mathbb{B}^3_P$;
4: Parameterize M to the unit ball $f_M : M \to \mathbb{B}^3_M$;
5: Construct the polycube map $\phi : \partial M \to \partial P$;
6: Construct the map between two balls $\psi : \mathbb{B}^3_P \to \mathbb{B}^3_M$ induced by the polycube map ϕ;
7: Compute the composite map $f : M \to P$, $f = f_M \circ \psi \circ f_P^{-1}$.

Algorithm 4: Polycube parameterization of star shapes

6.3.2 Parameterizing a Star Shape to a Polycube

Ball parameterization is useful for the star shapes which resemble the geometry of the sphere. However, a general star shape may be significantly different from a ball. Thus, ball parameterization may result in large distortions. For such cases, we propose to use the star-shaped polycube as the parametric domain since it resembles the input object better than the ball.

Given a star shape M and a polycube P, we want to find a bijective and smooth map $f : M \to P$. Rather than computing the map directly, we first individually parameterize M and P to the unit balls using Algorithm 1 (see Sec 6.3.1). Then we seek a smooth map between two balls $\psi : \mathbb{B}^3_M \to \mathbb{B}^3_P$. Finally, the polycube parameterization is given by the composite map $f = f_M \circ \psi \circ f_P^{-1}$.

The polycube parameterization can be illustrated clearly by the following commutative diagram:

$$\begin{array}{ccc} M & \xrightarrow{f} & P \\ {\scriptstyle f_M}\downarrow & & \downarrow{\scriptstyle f_P} \\ \mathbb{B}^3_M & \xrightarrow[\psi]{} & \mathbb{B}^3_P \end{array}$$

Note that there exists infinitely many smooth maps between two unit balls, but different ψ could result in different volumetric parameterization. To find a low-distortion volumetric parameterization, e.g., mapping the head of Moai (see Figure 6.4(a)) to the top of the polycube, and so on, the polycube map can serve as a feasible boundary constraint. In our implementation, we choose the approach of conformal polycube map (for genus-0 surfaces) [33]. Figure 6.4 illustrates the pipeline of parameterizing the star shapes to the polycube.

6.4 Parameterizing General Volumes

This section extends the above parameterization algorithm to general volumes. The key idea is to partition a non-star-shaped volume into several sub-volumes, each being a star shape. Then we can parameterize each individual star shape using the algorithm developed in Section 6.3. Finally, we can glue the sub-volumes together and smooth the cutting regions by harmonic fields, which lead to a global parameterization.

The following shows the details of our algorithm:

Step 1. Map the boundary surface of M, ∂M to a polycube surface, ∂P, using the algorithm introduced in He *et al.*[33] based on the divide-and-conquer method. The polycube map is denoted as $\phi : \partial M \to \partial P$.

Step 2. Decompose P into multiple disjoint components by several cutting planes $\{p_1, p_2, \cdots, p_n\}$. Let the intersection curve of $p_k \cap \partial P$ be τ_k. Let the pre-image of τ_k, $\phi^{-1}(\tau_k)$ be $\gamma_k \subset \partial M$. Compute a minimal surface $\pi_k \subset M$ inside M with the boundary condition $\partial \pi_k = \gamma_k$. The cutting planes are specified by the user to cut the mesh into semantic and star-shaped parts. e.g. 5 cutting planes are needed to cut the model into star-shaped parts(Figure 6.5).

Step 3. $\{\pi_1, \pi_2, \cdots, \pi_n\}$ segment M to sub-volumes, denoted as $\{M_1, M_2, \cdots, M_n\}$. Verify whether each M_k is star-shaped. If there exists a non-star shape, we repeat steps 2 and 3, until all M_k's become star-shaped.

Step 4. Construct the mapping from M_k to P_k, $f_k : M_k \to P_k$, based on the Green's function using $\phi_0 : \partial M \to \partial P$ and $\phi_k : \pi_k \to p_k$ as the boundary conditions. The union of $\{f_k\}$ gives us a global map $f : M \to P$.

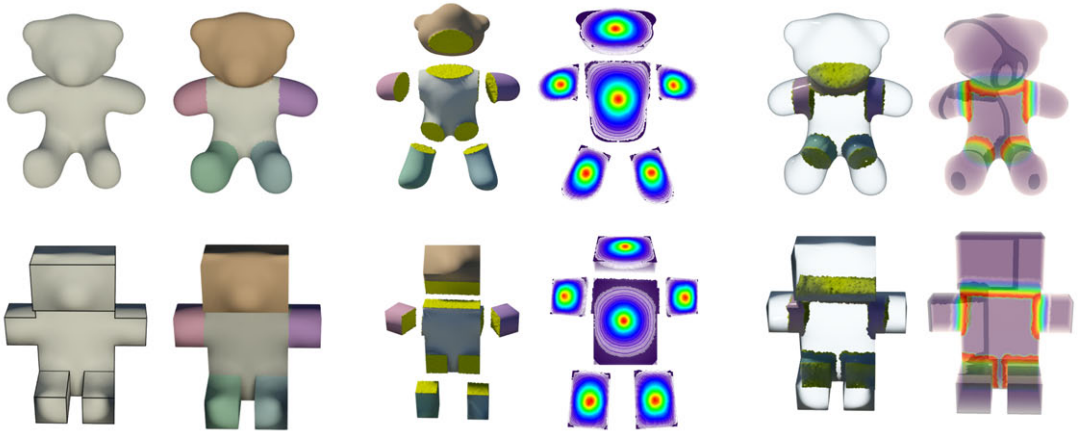
Step 5. Improve the map quality of the cutting boundaries. Find a neighborhood N_k of each cut surface $\pi_k \subset N_k \subset M$. Note that N_k is a cylinder-link shape since each π_k is a topological disk. Let T_k and B_k be the top and bottom surfaces of ∂N_k. Compute a harmonic function $g : N_k \to \mathbb{R}$, such that,

$$\begin{cases} \Delta g(p) = 0, & \forall p \notin B_k \bigcup T_k \\ g(p) = 0, & \forall p \in B_k \\ g(p) = 1, & \forall p \in T_k. \end{cases}$$

Similarly, the corresponding neighborhood $\bar{N}_k := f(N_k) \subset P$ is also a cylinder-like shape. Let $\bar{T}_k$ and $\bar{B}_k$ be the top and bottom surfaces of $\partial \bar{N}_k$, respectively. Compute a harmonic function $h : \bar{N}_k \to \mathbb{R}$, such that

$$\begin{cases} \Delta h(q) = 0, & \forall q \notin \bar{B}_k \bigcup \bar{T}_k \\ h(q) = 0, & \forall q \in \bar{B}_k \\ h(q) = 1, & \forall q \in \bar{T}_k. \end{cases}$$

From every point $p \in B_k$, trace an integration curve $\gamma_p \in N_k$ following the gradient vector field ∇g. Let $q = f(p) \in \bar{B}_k$ be the image of p. Trace another integration curve $\gamma_q \in \bar{N}_k$ following the gradient vector field ∇h. It can be proven that each integral curve γ_p (resp. γ_q) terminates on the top surface C_k (resp. $\bar{C}_k$). Furthermore, no two integral curves inside N_k (or $\bar{N}_k$) intersect as the gradient vector field never vanishes inside the volume. As a result, $\gamma_p \in N_k$ can be uniquely mapped to $\gamma_q \in \bar{N}_l$. This further induces a bijection between N_k and $\bar{N}_k$. Due to the smoothness property of the harmonic functions g and h, the integration curves of ∇g and ∇h are also smooth curves. Thus, the induced bijection N_k and $\bar{N}_k$ is smooth.



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

Figure 6.5: Parameterizing general volumes using star shape decomposition. Given the Teddy Bear M and its polycube domain P, we compute a polycube map between the boundary surfaces, see (a). Then we partition the boundary surface ∂P by cutting planes. This partition of ∂P also induces a partition on ∂M, see (b). We verify each partitioned sub-volumes to be star-shaped. If not, repeat the partition step, see (c). Then we can parameterize each individual star shape using the algorithm developed in Section 6.3, see (d). Finally, we can glue the sub-volumes together and smooth the cutting regions by harmonic fields, see (e) and (f), which lead to a global parameterization.

6.5 Experimental Results

Figure 6.6 shows the parameterization of the pig-shaped coin box to a polycube. The volume rendering and the cut views reveal the quality of the parameterization. Figure 6.8 shows the parameterization of non-star-shaped volumes.

We compared our method with the volumetric harmonic map method Wang *et al.* [80]. As mentioned above, the volumetric harmonic map is not guaranteed to be homeomorphic even though the domain is convex. In Figure 6.7, we parameterized the star model to the unit ball. As shown in the cut view and iso-parametric curves, our method is robust and leads to hexahedral meshes with better quality.

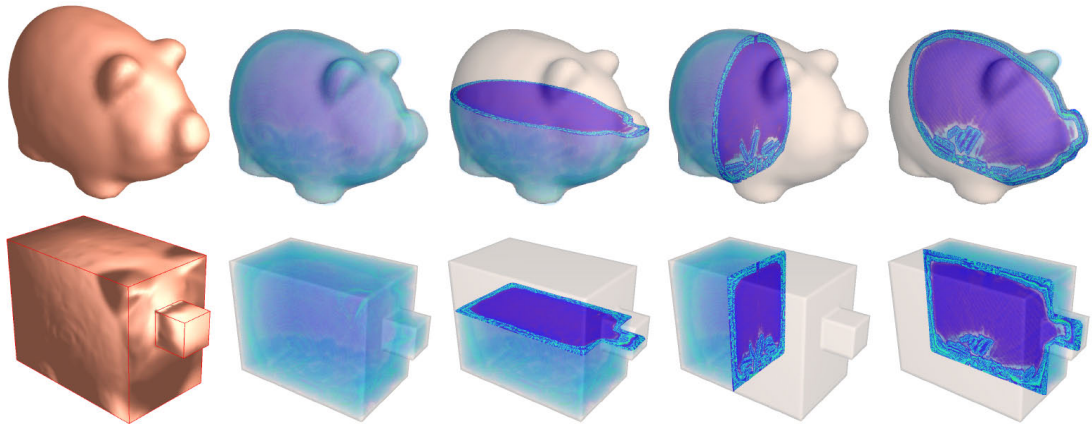
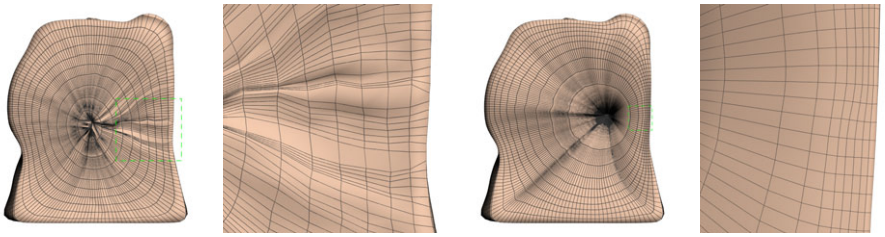
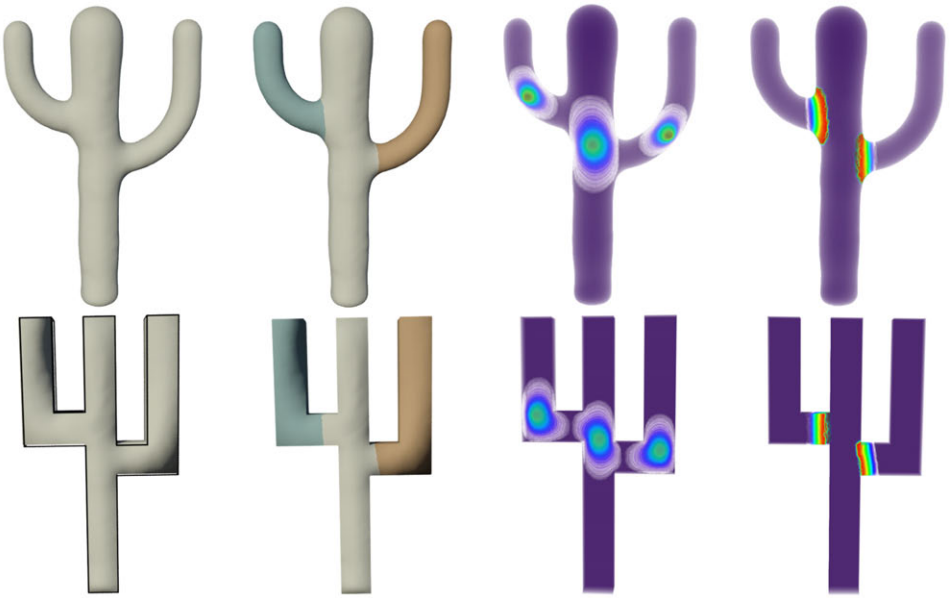


Figure 6.6: Parameterizing the pig model to a polycube. Row 1 and 2 show the volume rendering of the volumetric data and polycube parameterization respectively.

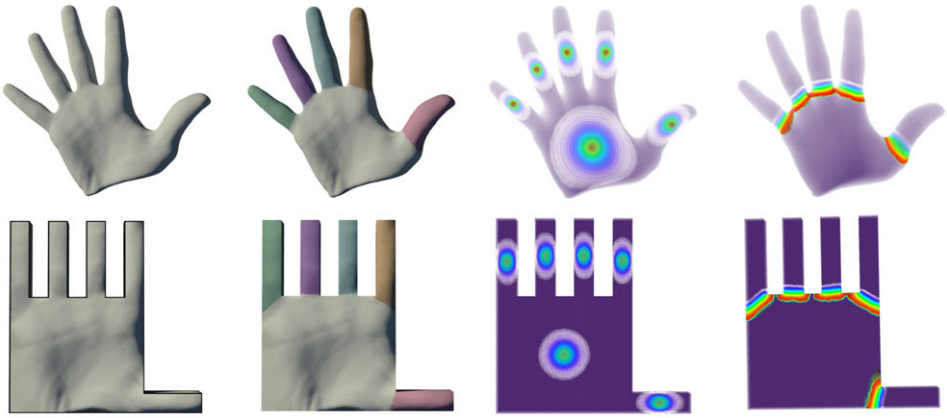


(a) Harmonic map Wang *et al.* [80] (b) Our method

Figure 6.7: Comparison with harmonic map Wang *et al.* [80]. We map the dog head to unit ball using volumetric harmonic map Wang *et al.* [80] and our approach. As shown in the cut view of iso-parametric curves, our method is more robust and leads to hexahedral meshes with better quality. Our approach also guarantees that the iso-parametric curve which follow the direction of the gradient is orthogonal to the other two iso-parametric curves which span the iso-surface of the Green's function.



(a) the parameterization of cactus model



(b) the parameterization of hand model

Figure 6.8: More parameterization results.

Chapter 7

Direct-Product Volumetric Parameterization of Handlebodies

7.1 Motivation

In this work we focus on the parameterization of 3-dimensional handlebodies that can be decomposed into the direct product of 2-dimensional surface and a 1-dimensional curve. These models are very common in engineering fields, thus, the parameterization of such models is highly desirable. Note that many canonical domains have extremely simple structures. Some of them are just direct product of shapes from lower dimensions. Figure 7.1 shows examples including a solid cube, which is the direct product of three 1-dimensional line segments, and a solid torus, which is the direct product of a 2-dimensional disk and a 1-dimensional circle.

In general, an n-hole ($n \geq 1$) handlebody is a 3-manifold whose boundary is a surface that can be continuously deformed to some unknotted n-hole torus without tearing or self intersection. For such a volume H, its boundary surface ∂H can be covered by a set of charts,

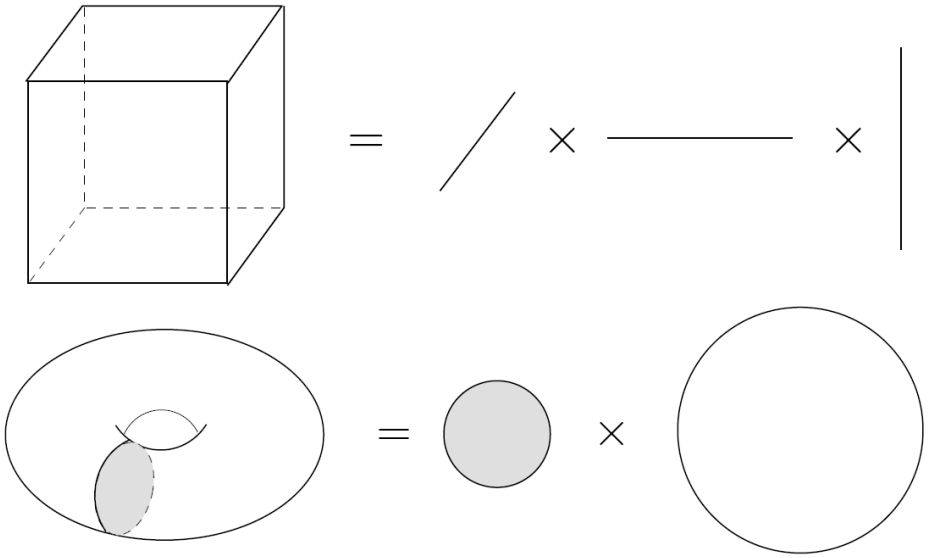
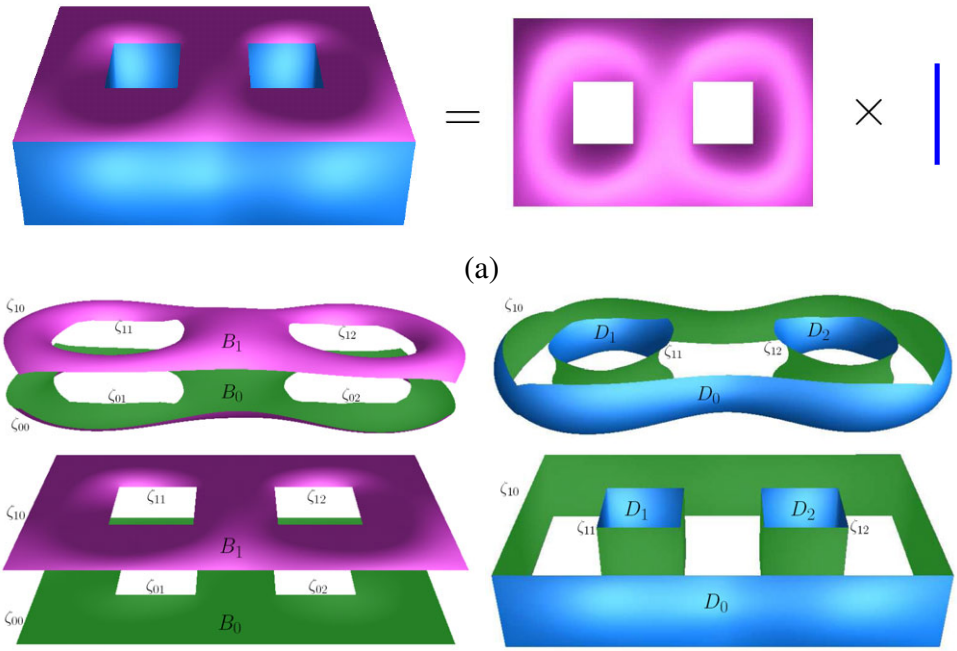


Figure 7.1: Handlebodies can be viewed as direct product of shapes in lower dimension. (a) Solid cube is the direct product of three line segments. (b) Solid torus is the direct product of a disk and a circle.

$$\partial H = B_0 \cup B_1 \cup \bigcup_{i=0}^{n} D_i$$

where B_0 and B_1 are called *bases*, or to be specific, B_0 is the *floor* and B_1 the *ceiling* respectively. They are two disjoint n-hole annuli, $B_0 \cap B_1 = \varnothing$. Each $D_i (i \in [0..n])$ is called a *wall*, which is a topological cylinder with both ends open. All the walls are pairwisely disjoint, $D_i \cap D_j = \varnothing (i \neq j)$. A base $B_k (k \in \{0,1\})$ and a wall $D_l (l \in [0..n])$ intersect at a 1-dimensional simple loop, $\zeta_{kl} = B_k \cap D_l$. Figure 7.2 shows the Figure Eight model which is a 2-hole handlebody.

As a special case, volumes without any handle (i.e. topological solid balls) can be considered as degenerate handle bodies with $n = 0$. The boundary surface for such bodies can also be partitioned into bases and walls, where each base is a topological disk and there is only one single wall.



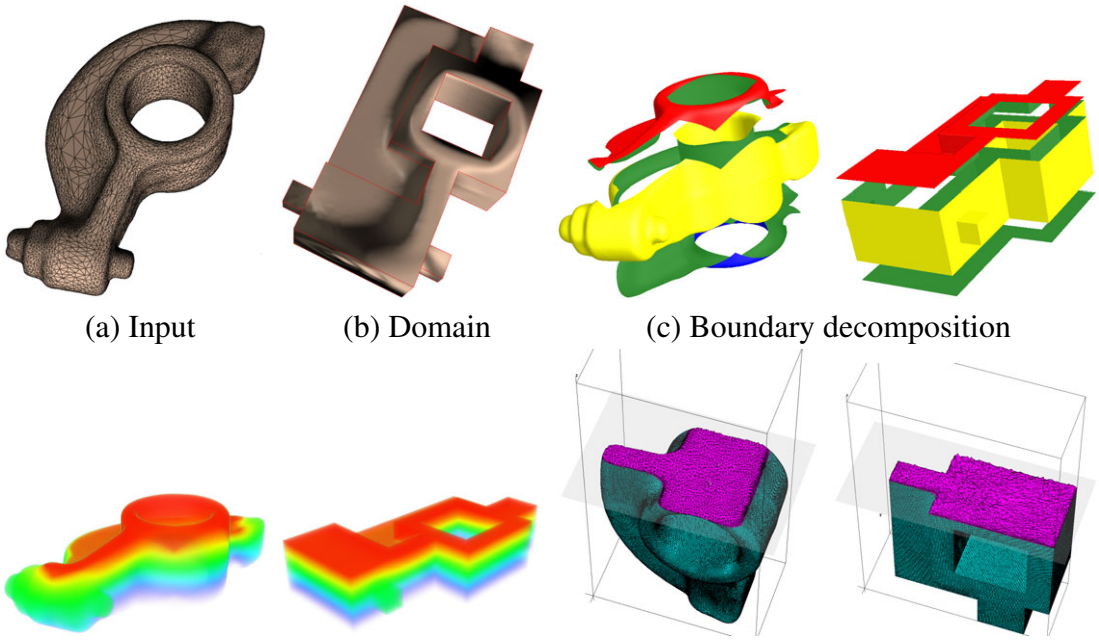
(a)

(b) Ceiling & floor

(c) Walls

Figure 7.2: Direct product of Figure Eight model. The parametric domain (a genus-2 polycube) is the direct product of 2-hole disk and line segment, see (a). Both the volume and the polycube domain can be decomposed into ceiling, floor and walls, see (b)-(c).

The reason we prefer direct product domains is multi-folded. Firstly, in a direct product domain there is no singularity, thus the parameterization would not degenerate at any point of the volume. Secondly, such a domain naturally allows a structure of orthogonality. In fact, at any point of the volume the three parameter lines are orthogonal to one another. Thirdly and consequently, such structures will make many tasks easier, such as volumetric remeshing, volumetric registration, physical simulation and so on.



(a) Input (b) Domain (c) Boundary decomposition

(d) Harmonic field (f) Volumetric parameterization

Figure 7.3: Direct product volumetric parameterization. The input is a tetrahedral mesh representing a 3D handlebody. We parameterize the boundary mesh to a polycube which also serves the parametric domain of the volume parameterization. Then we decompose the boundary mesh into ceiling (red), floor (blue) and walls (yellow). The ceiling and floors are topological annulus, the walls contain two topological cylinders. Next, we compute volumetric harmonic functions with the Dirichlet boundary conditions. Finally, the harmonic fields induce a homeomorphism between the handlebody and the polycube domain. The cut view illustrate the volumetric parameterization.

Given a 3D handlebody M, we first construct a parametric domain P. To facilitate the hexahedral remeshing, we choose polycube due to its regular structure. Then, we compute a bijective map between the boundary surfaces ∂M and ∂P. Next, we partition ∂P

into ceiling, floor and walls. The bijective map induces the partition of ∂M. We compute the volumetric harmonic map with a Dirichlet boundary condition, i.e., the ceiling points are with boundary value 1, and floor points with boundary value 0. Finally, by tracing the gradient field of harmonic functions, we construct a map between M and P. Figure 7.3 shows the parameterization of the genus-1 Rockerarm model to a polycube domain.

7.2 Algorithm

7.2.1 Overview

We choose the polycube as the parametric domain due to the following reasons: first, polycube has a regular structure and can be used to construct all-hexahedral meshes; second, there are well-developed techniques to construct the polycube map that serves the map between the boundary surfaces; third, due to the geometric simplicity, it is usually easier to partition the boundary surface of polycube than that of the input model.

Since the goal is to parameterize the given volume M to the polycube P, we need to compute the parameter functions $(u,v,w)_M$ and $(u,v,w)_P$ for M and P respectively. Then we construct a function that maps $(u,v)_M$ to $(u,v)_P$. Finally, the parameterization $\phi : M \to P$ is induced by the map $(u,v,w)_M \to (u,v,w)_P$. This idea can be illustrated using the following commutative diagram:

$$\begin{array}{ccc} M & \xrightarrow{\phi:M\to P} & P \\ \downarrow & & \downarrow \\ (u,v,w)_M & \xrightarrow[h:(u,v)_M\to(u,v)_P]{} & (u,v,w)_P \end{array}$$

The proposed direct product volume parameterization contains the following steps:

- Input: A tetrahedral mesh M for a handle body and a polycube P.
- Output: A bijection $\phi : M \to P$.
  - (i) Partition the boundary surfaces ∂M and ∂P into bases B_i and walls D_j ;
  - (ii) Compute harmonic functions f_M and f_P using the partition as Dirichlet boundary conditions;
  - (iii) Trace the integral curves for every interior vertex.
  - (iv) Trace the integral curves on the walls.

7.2.2 Computing the Harmonic Fields

Given a tetrahedral mesh M and the user-constructed polycube domain P, we first construct the polycube map between the boundary surfaces ∂M and ∂P using the divide-and-conquer approach by He *et al.* [33]. We design the polycube manually to reduce its complexity which facilitates the boundary decomposition. Then, we partition the boundary surfaces into bases (ceiling and floor) and walls. Note that the polycube domain has simpler structure than the original surface. Thus, it is much easier to partition the polycube boundary surface than the original model. To do this, the user just simply choose the desired polycube faces by several mouse clicks. By taking advantage of the bijectivity of the polycube map, the partition of ∂P naturally induces a partition of ∂M. Let B_i^M and B_i^P denote the bases for ∂M and ∂P respectively. Similar, D_j^M and D_j^P denote the walls. Figure 7.4 and 7.8 show the partition of the genus-0 Bimba and genus-2 Cup via the polycube maps.

With the ceiling/floor/wall partition, we are ready to compute the harmonic fields for both M and P. Specifically, we solve two harmonic functions $f_M : M \to \mathbb{R}$ and $f_P : P \to$

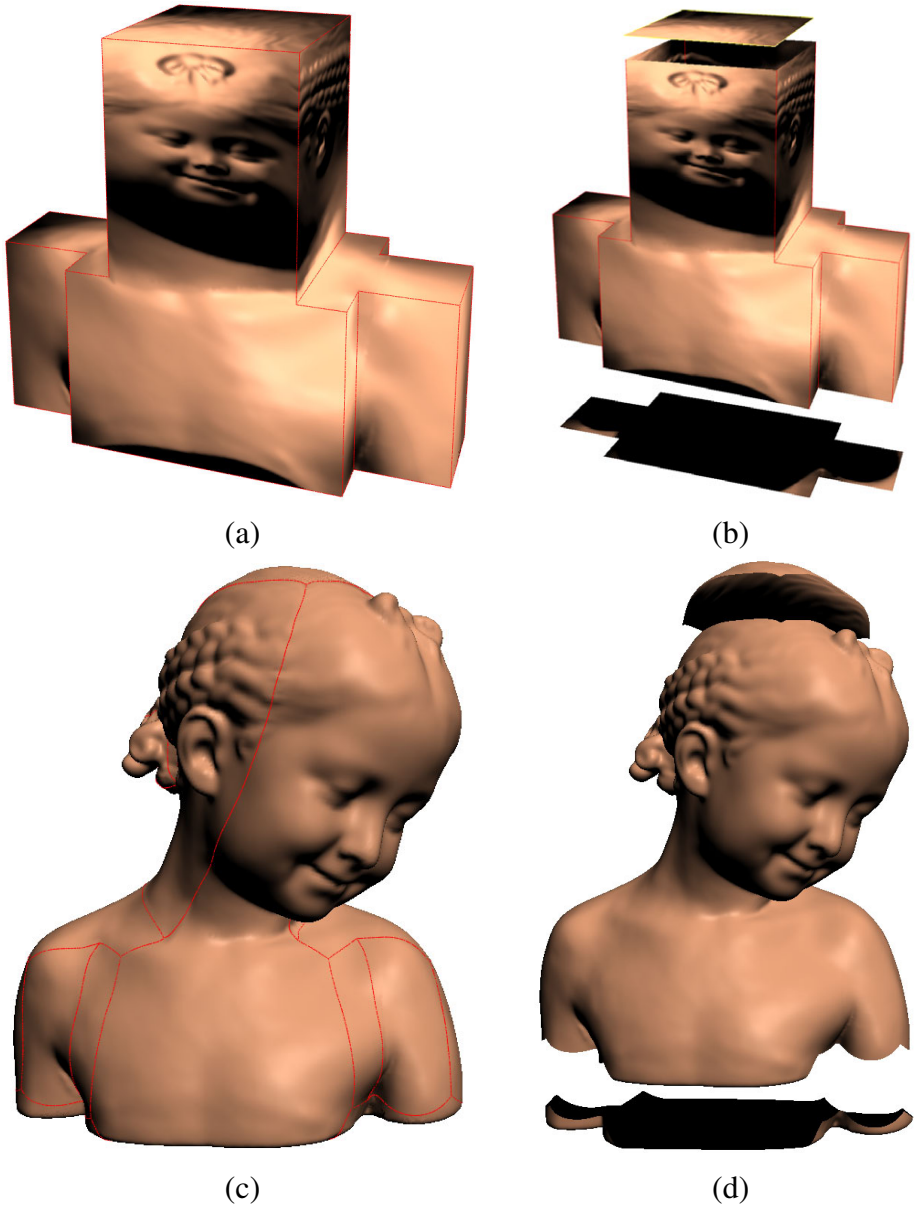
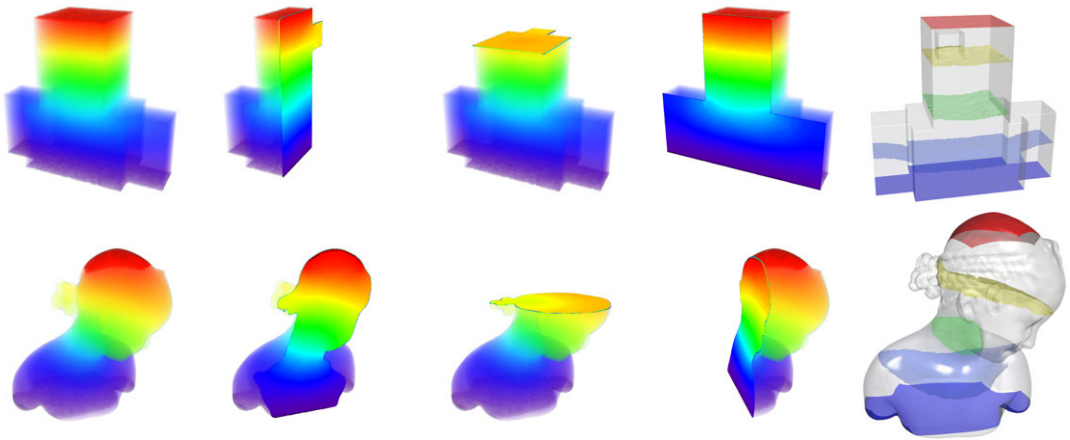


Figure 7.4: Partition the boundary surface into ceiling, floor and walls. We first construct the polycube map for the boundary surface The top and bottom faces of the polycube serve the ceiling and floor and the remaining part is the wall. The polycube map induces the surface partition on the 3D model.



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

Figure 7.5: Computing the harmonic fields of the genus-0 Bimba model. We solve a volumetric harmonic map $\triangle f = 0$ with boundary conditions $f(B_0) = 0$ and $f(B_1) = 1$. The volume renderings (a)-(d) show the computed harmonic fields on the polycube and 3D model. (e) shows several iso-surfaces of harmonic fields.

$\mathbb{R}$, such that

$$\triangle f(p) = 0,\ \forall p \notin B_0 \cup B_1$$

$$f(p) = 0,\ \forall p \in B_0$$

$$f(p) = 1,\ \forall p \in B_1.$$

Figure 7.5 show the computed harmonic fields in M and P of the genus-0 Bimba model.

7.2.3 Tracing Inside the Volume

To trace the integral curve, we need to compute the gradient of f_M and f_P. Given an arbitrary scalar function g, the gradient ∇g can be computed as follows [31]: suppose t_{ijkl} is a tetrahedron with vertices $\{p_i, p_j, p_k, p_l\}$, the face on the tetrahedron against vertex p_i is f_i; similarly p_j, p_k, and p_l are against f_j, f_k, and f_l, respectively. We define $\mathbf{s}_i$ to be the vector along the normal of f_i with length equal to 2 times the area of f_i, and

so can $\mathbf{s}_j, \mathbf{s}_k, \mathbf{s}_l$ be defined. Then, the gradient of g in t_{ijkl} is a constant vector field

$$\nabla g = g(p_i)\mathbf{s}_i + g(p_j)\mathbf{s}_j + g(p_k)\mathbf{s}_k + g(p_l)\mathbf{s}_l.$$

We then define the vertex gradient as the average of the gradient vectors in the neighboring tetrahedra.

The parameterization from the M to P is constructed as follows: given an interior point $s \in M$, we trace the integration curve γ of the gradient field ∇f_M. The integration curve γ intersects the ceiling and floor at $p \in B_1^M$ and $q \in B_0^M$.

Using the polycube map, we can define a floor map $h : B_0^M \to B_0^P$ that maps each floor point $q \in B_0^M$ to $q' \in B_0^P$.

Then we compute an integral curve γ' starting from q' and following ∇f_P. γ' intersects the ceiling at $p' \in B_1^P$. Then we can find a unique point $s' \in \gamma'$ such that $f_P(s') = f_M(s)$. The point $s' \in P$ is the image of s, i.e., $\phi(s) = s'$. By computing the images for every interior vertex in M, we build the parameterization between the interior of M and P.

Note that each integral curve starts from a point on the floor and then terminates at a point on the ceiling. Since the floor is a g-hole disc if ∂M is of genus-g, we can map the g-hole disc to planar domain and then assign a parameter (u, v) for each integral curve, i.e., all points on the same integral curve share the same u, and v parameters, they only differ by w-parameter, which is determined by the arc-length parameterization. Figure 7.6 and 7.9 show tracing the integral curves inside the volumes.

7.2.4 Tracing On the Walls

In the previous step, we construct the map for the interior vertices. Now, we want to build the map between walls. Tracing the integral curve on the walls can be viewed as

the degenerate case of tracing inside the volumes. Restricting the volumetric harmonic function on the walls, we have the surface harmonic function denoted by g_D.

Given a triangle $t_{ijk} = \{p_i, p_j, p_k\}$, let e_i, e_j and e_k denote the oppositive edges. We define a rotation operator *rot* that rotates the edge 90 degree outwards the triangle. Then the gradient on the triangle t_{ijk} is given by

$$\nabla g_D = g_D(p_i)rot(e_i) + g_D(p_j)rot(e_j) + g_D(p_k)rot(e_k).$$

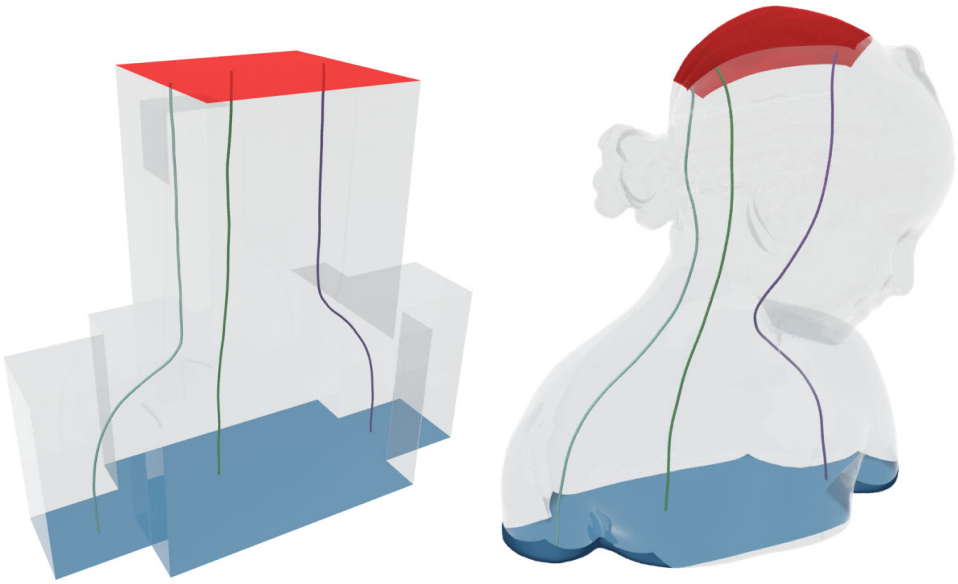
Starting from the vertices on the boundary of floor ∂B_0, we trace the integral curve following the gradient of g_D. The integral curves are perpendicular to the iso-curves of g_D and terminates on the boundary of ceiling ∂B_1. Figure 7.7 shows tracing the integral curves on the walls.

Note that the floor map $h : B_0^M \to B_0^P$ is a bijection, as h is constructed by restricting the polycube map to the floor. An integral curve is given by a unique (u, v) parameter and the points on the same integral curve are differed by the w parameter. Thus, the floor map h maps an integral curve in M to a unique integral curve in P, which in turn maps a point in M to a unique point in P. So, by tracing the integral curves for every floor point, we build the piecewise parameterization between M and P.

7.3 Experimental Results

The proposed direct product volumetric parameterization is guaranteed to be bijective. Our method can also guarantee that at any point in the volume, the w parametric line is orthogonal to u and v parametric lines, which is a natural consequence of enforcing a conformal parameterization on the surface and a gradient field in the volume. The reason is as follows: we map each integral curve in M to a unique integral curve in

P. Both integral curves follow the gradient vector field of the harmonic function, thus, are orthogonal to the iso-surfaces of the harmonic function. Note that he iso-surface induces the u and v parameters and the integral curve induces the w parameter, thus, the w iso-parametric line is perpendicular to $u-$ and $v-$ lines.

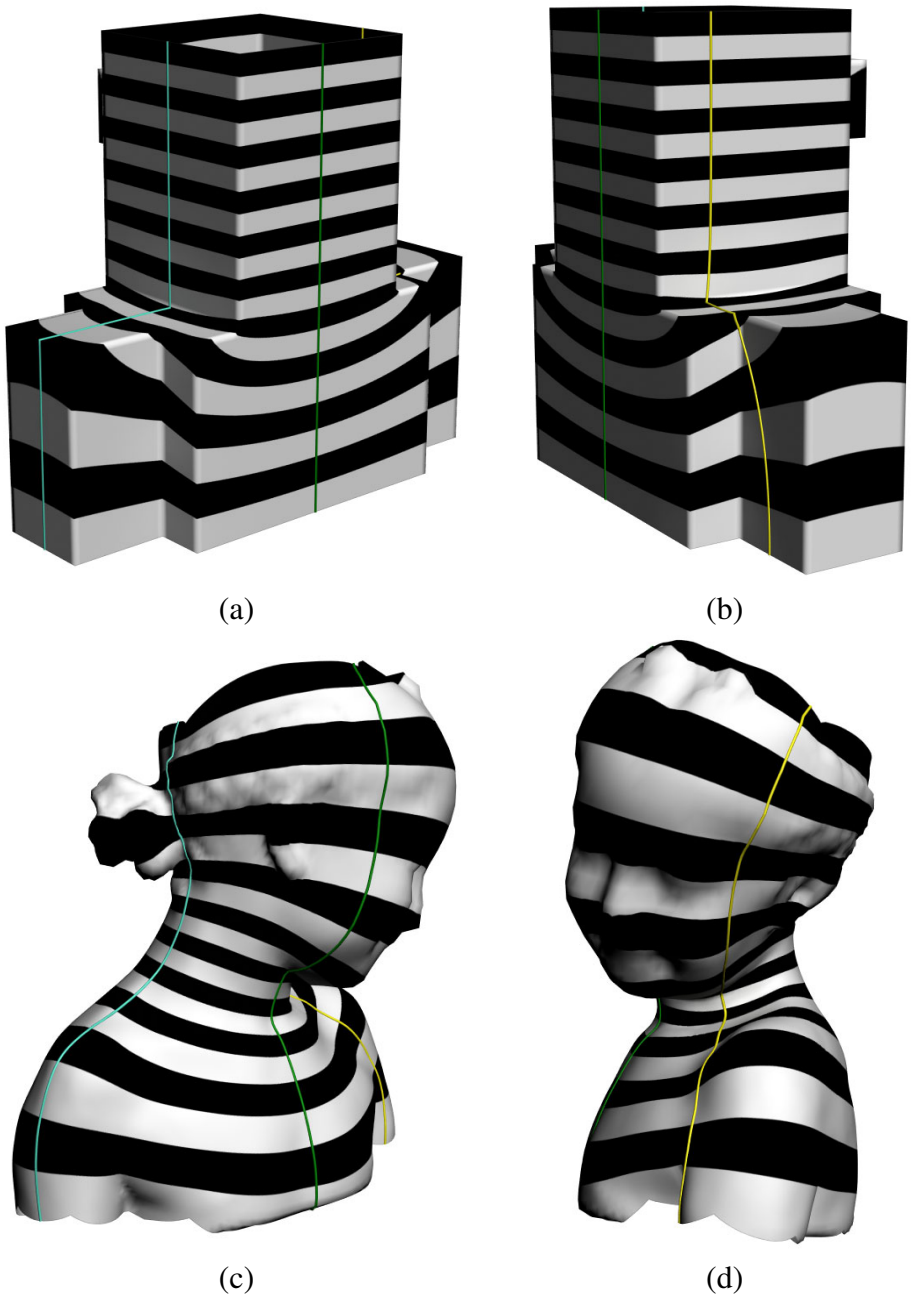


(a) (b)

Figure 7.6: Tracing the integral curves in the volumes. The integral curves follow the gradient field of the harmonic function and is perpendicular to the floor and ceiling. Three pairs of integral curves are shown in M and P. The corresponding curves are drawn in the same color. Intuitively speaking, the position of the starting point on the floor induces the u and v parameters, and the integral curve induces the w parameter.

We tested our algorithm on models with various topology, including, Bimba (genus-0), Rockerarm (genus-1), Cup (genus-2) and Eight (genus-2).

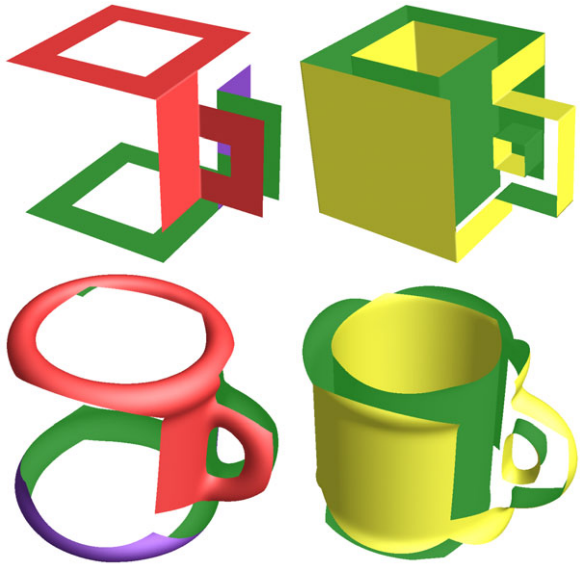
We computed the polycube map using the variant of divide-and-conquer method He *et al.* [33]. In [33], He *et al*'s method can construct the polycube automatically by segmenting the shapes using horizontal planes and then approximate each part using voxelization-like approach. The resulted polycube is often complicated and has large number of corners that are extraordinary points of the polycube map. To reduce the



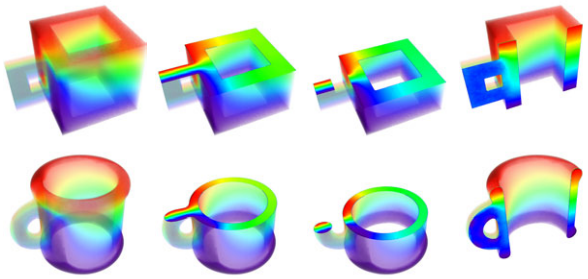
(a) (b)

(c) (d)

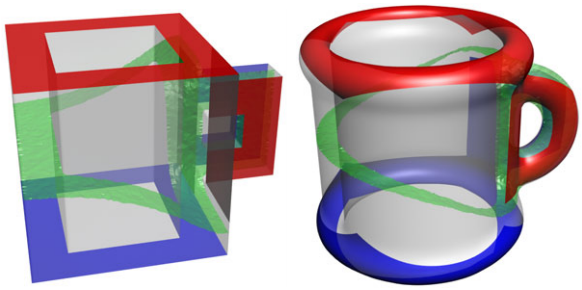
Figure 7.7: Tracing the integral curves on the walls. The texture mapping shows the iso-curves of the harmonic function restricted to the wall. The integral curve follows the harmonic function and is perpendicular to the boundary of the walls. Three pairs of integral curves are shown in M and P. The corresponding curves are drawn in the same color.



(a) Ceiling & floor (b) Walls



(c) Harmonic fields



(d) Iso-surface of harmonic fields

Figure 7.8: Direct product parameterization of the genus-2 Cup model. (a) and (b) show the ceiling, floor and wall decomposition. Each ceiling/floor is 2-hole disk and the wall contains three topological cylinders. (c) shows the volume rendering of the harmonic fields. (d) show the iso-surface of the harmonic fields.

(a) (b)

Figure 7.9: Tracing the integral curves in M and P. (a) and (b) show two different views of three integral curves. The ceiling and floor are drawn in red and blue respectively. The walls are rendered transparently. The corresponding curves are drawn in the same color.

Table 7.1: Distortion.

Model	Aspect Ratio best	Aspect Ratio average	Aspect Ratio worst
bima	1.00622	1.44828)	6.01021
cup	1.00953	1.32306	2.77817
eight	1.00571	1.61857	4.17987

number of singularities, we prefer to construct the polycube manually. The divide-and-conquer framework [33] can still apply to the manually constructed polycube.

Taking advantage of the polycube map, we developed a user-friendly interface that allows the users to easily select the polycube faces by simple mouse click. With the user-specified Dirichlet boundary condition, we solved the discrete harmonic function using finite element method. We used volume rendering to visualize the harmonic field inside the volumes, as shown in Fig. 7.3(d), 7.5 and 7.8.

We demonstrated the proposed volumetric parameterization algorithm on hexahedral mesh generation. Note that the polycube domain has very natural hexahedral tessellation. The volumetric parameterization induces a map between the polycube and the given volume. Thus, it also induces a hexahedral tessellation in the volume dataset as shown in Fig. 8.7. To measure the quality of the hexahedral mesh, we choose the aspect ratio which measures the maximum edge length ratios at hex center [75].

7.4 Discussions

We compared our method with the existing approaches Wang *et al.* [80], Li *et al.*[47], Martin *et al.* [52] and the proposed method in Chapter 5. In Wang *et al.* [80], the volume boundary is mapped to the unit sphere first, the boundary mapping is a diffeomorphism. Then the boundary diffeomorphism is extended to the interior by solving

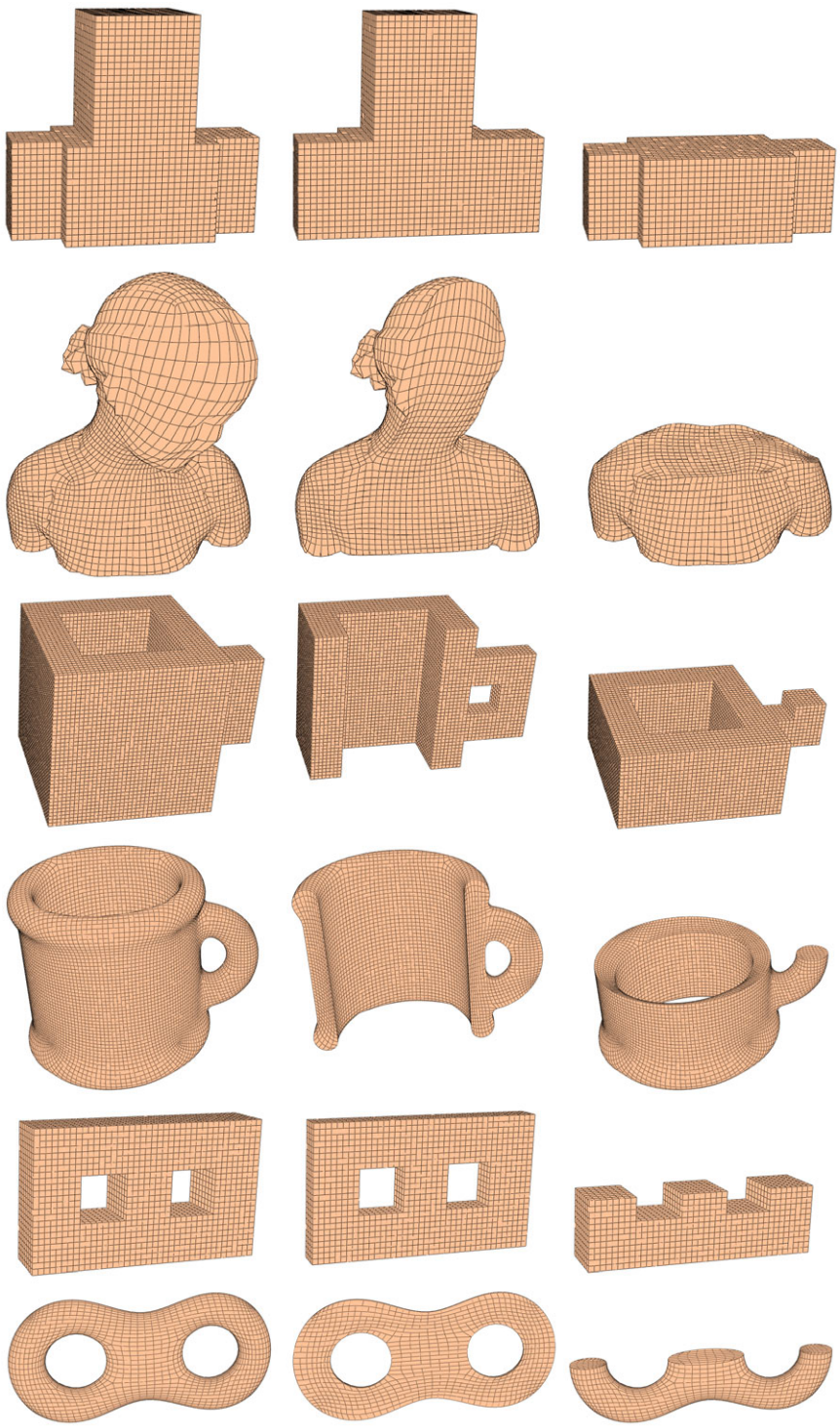


Figure 7.10: Construction of the hexahedral mesh using direct-product volumetric parameterization.

a volumetric Dirichlet problem, using the boundary mapping as the boundary conditions. Wang et al used finite element method to solve the volumetric Dirichlet problem. In [47], Li et al used fundamental solution method, which doesn't require the tessellation of the volume to solve the exact same Dirichlet problem. In two dimensional case, Rado's theorem claims that if the target domain is convex, then harmonic maps are diffeomorphisms. Unfortunately, in volumetric case, even the target is convex and boundary mapping is diffeomorphic, the volumetric harmonic map may not be diffeomorphic. Therefore, neither Wang *et al.* method [80] nor Li *et al.* method [47] can not guarantee the final mapping to be diffeomorphic. In [52], Martin *et al.* first specified a 1-dimensional skeleton inside the volume and solve the volumetric harmonic map with Dirichlet boundary conditions of the skeleton and the boundary surface. The parameterization is constructed by tracing the gradient field of the harmonic function. Martin *et al.*'s method works only for topological balls where the homotopy type of the offset surface of the skeleton is the same as the boundary surface. Furthermore, the skeleton are the critical points of the harmonic map, thus, the resulted parameterization is singular on the skeleton. In Chapter 5, the proposed method compute the Green's function in a star-shaped volume and then map it to a ball or a polycube. By a divide-and-conquer approach, the method is extended to general volumes. The advantage of the method in Chapter 5 is that the mapping between two non-trivial shapes is theoretically guaranteed to be a diffeomorphism. However, the promising properties of the method degenerate when it is extended to general shape. The partition of volume needs a lot of efforts. Additionally, although the cutting regions are locally smoothed, the smoothness of mapping on the boundary surfaces on the cutting region can not be guaranteed.

Our method is different from the above approaches in three aspects: first, our method is guaranteed to be a bijection if the floor map is bijective; second, our method works

Table 7.2: Comparison to the existing approaches.

	Wang [80]	Li [47]	Martin [52]	Green's Function Chapter 5	Ours
Topology	Topological ball	Arbitrary bounded volume	Topological ball	General volume	Handlebody
Singularity	No	No	Yes	Yes	No
Bijectivity	No	No	Yes	Yes	Yes

for volumes with complex topology and geometry; third, our parameterization does not have any singularity, thus, is regular everywhere. Table 7.2 summarizes the properties of the existing volumetric parameterization algorithms and our method.

Chapter 8

Applications

8.1 GPU Subdivision Displacement

In recent years, hardware-supported tessellation is already feasible and provides much faster model visualization than traditional methods [50], and graphics card hardware designers have added specialized programable units for the task (Shader Model 5 hardware).

Our proposed editable polycube map system (chapter 4) has several positive properties for GPU-friendly interpolative subdivison surfaces.

- The map is a bijection with low angle and area distortion. The result mapping has high quality.
- The user can easily control the mapping by specifying optional features on the 3D model and their desired locations on the polycube domain.
- The reduced number of combinations, together with a watertight sampling scheme, allow for a continuous subdivision method that smoothly integrates with current production pipelines.

- We are able to provide coarse regular base meshes with a reduced memory footprint.

The whole process should be compared with the traditional work artists do to create a model ready to be used in a real-time environment like a computer game, which usually requires a subdivision scheme in order to be used with modern tessellation hardware. In general, if the model to create is based on an existing model, for instance obtained from a laser scanning process, the final model generation process implies manually performing re-topology operations on a subdivision-based representation to then transfer corresponding details by normal projection, which usually do not ensure bijectivity. Usually, artists start from a coarse base model, quite often sculpted using a polycube as base mesh. Then, the model is imported into an application like ZBrush [59] and further subdivided with a couple of steps in a Catmull-Clark subdivision. From this moment on, the artist has to model/transfer the fine-grained details onto the model to be used. Obviously, this is a time-consuming and quite redundant procedure to get the final model. Our proposal is to avoid this by quickly establishing a bijective correspondence between the coarse polycube and the input model, and then creating the subdivision scheme on the polycube-based model by transferring the details of the high resolution input model. The resulting model will have all the enumerated properties and would be ready for usage in a production environment with a minimal user interaction.

Here we explain the details of our GPU-based subdivision displacement algorithm.

After the bijective mapping between the model and the polycube-map constructed (chapter 4), we can build the induced subdivision surface. In general, we can generate a parameterized subdivision scheme which can be effectively used in any graphics application like computer-generated movies, as well as real-time applications. In our implementation, as the bijection has already been established, we only need to recover the

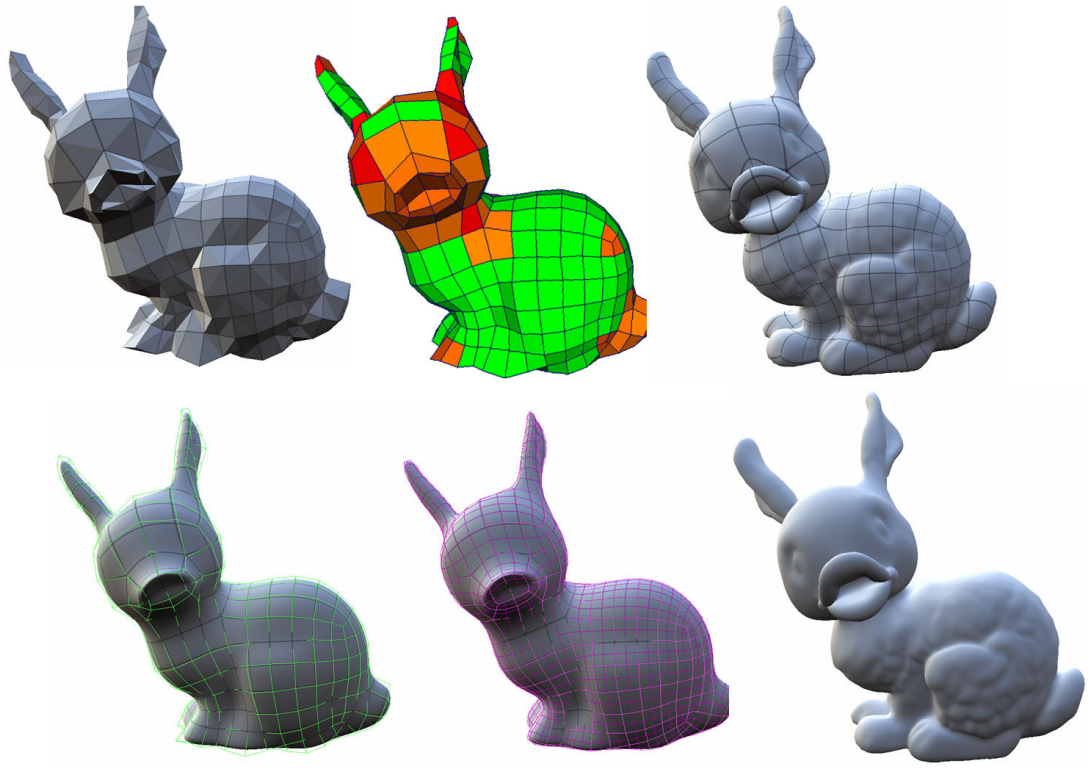


Figure 8.1: GPU-based subdivision displacement. Top left: our base mesh (level 0 of the subdivision scheme). Top middle: a mesh showing the patch structure we associate with a Catmull-Clark subdivision surface (green patches contain only valence 4 vertices, and the darkness orange levels show the number of extraordinary vertices, being darker the patches with greater number). Bottom left: the base mesh superimposed to the approximate Catmull-Clark subdivision surface in grey. Bottom middle: In lilac, the subdivision surface. Right Column: the subdivision surface with GPU-based subdivision displacement. The iso-parametric curves of the final can be seen in the top right.

coarse quads from the polycube-mapped model. In order to do that we (re-)constructed the Catmull-Clark Subdivision scheme with the algorithm described in Lanquetin *et al.* [44], and then prepare the approximate Catmull-Clark [50] data structures (textures & base mesh) for the real-time subdivision.

Later on, in real time applications, we use the subdivision algorithm presented by Loop and Schaefer [49]. The method presented here is particularly well suited for this imple-

mentation as we only generate vertices with valences 3,4,5 or 6. In this implementation, domain shaders (or vertex shaders when using instanced tessellation) are responsible of providing the final position of each new vertex from the tessellated mesh.

In order to have watertight sampling of the displacement map, we define, for each patch, the one who "owns" every single edge and corner [8, 7]. This way, all patches are coordinated with respect to what texture coordinate to use when sampling the displacement map at those locations. In practice, this amounts storing, for every edge and for every corner, the texture coordinates of the owner of those features (4 texture coordinates per vertex). At runtime, only a single texture sample is needed; and the corresponding texture coordinate can be selected with a simple calculation. Figure 8.1 shows the subdivision displacement result of our method.

8.2 Modeling 3D Facial Expressions Using Geometry Videos

We proposed an expression modeling system in chapter 5 to capture the expression animations and convert it to geometry video. But the constructed geometry video is usually quite large. For example, the latest high-resolution 3D camera [88] is able to capture 30 fps with a resolution of 512×512 of each frame, resulting in approximately 4.88MB raw data per frame, 878MB per second and 51.44GB per minute as shown in Fig. 5.1. This imposes a significant challenge for compressing the captured video efficiently while maintaining the video reconstruction quality.

To solve the aforementioned challenges and promote geometry video to real-world applications, this work presents a novel algorithm to store the recorded data in a compact way, and allow users to manage, manipulate, and render the data easily. Specifically, we propose a geometry video compression algorithm based on the state-of-the-art video

coding standard - H.264/AVC, together with our proposed progressive directional prediction scheme.

Using the proposed expression-invariant parameterization algorithm described in Chapter 4, each frame of the motion data is mapped to a rectangular domain that can be easily converted into a 2D geometry image representation, i.e. the vertex coordinates of the 3D face data x, y, and z, are represented by R, G and B colors respectively in a 2D image. We define such an image as a GV picture (GVP). Then a GV is constructed by combining a sequence of successive GVPs. Before presenting our compression scheme, we first analyze the property of geometry videos.

The temporal feature of geometry video is similar to that of natural video, i.e. a block in a geometry video picture usually closely matches another block locating at the same or close position in the neighbor picture. However, the degree of temporal correlation of geometry video is even stronger than that of natural video. We tested this property with motion estimation of H.264/AVC, the performance of which is highly relied on the temporal coherence. Fig. 8.2 shows that, for a natural video (greyscale) and its corresponding geometry video (RGB), the mean square errors (MSEs) produced after a 32x32 Full Search motion estimation of H.264/AVC (QP=8). It has been observed that, after the motion estimation, the MSE of geometry video in each color channel is significantly smaller than that of the corresponding natural video.

The spatial feature of geometry videos, on the other hand, is somewhat different from that of natural videos. As we have explained earlier, the pixel values of a geometry video picture are actually the vertex coordinates of the 3D model in the 3D space. Assuming that the 3D model surface is relatively smooth in a small local region, e.g. a 4x4 block region, the corresponding pixels within the geometry video picture will normally

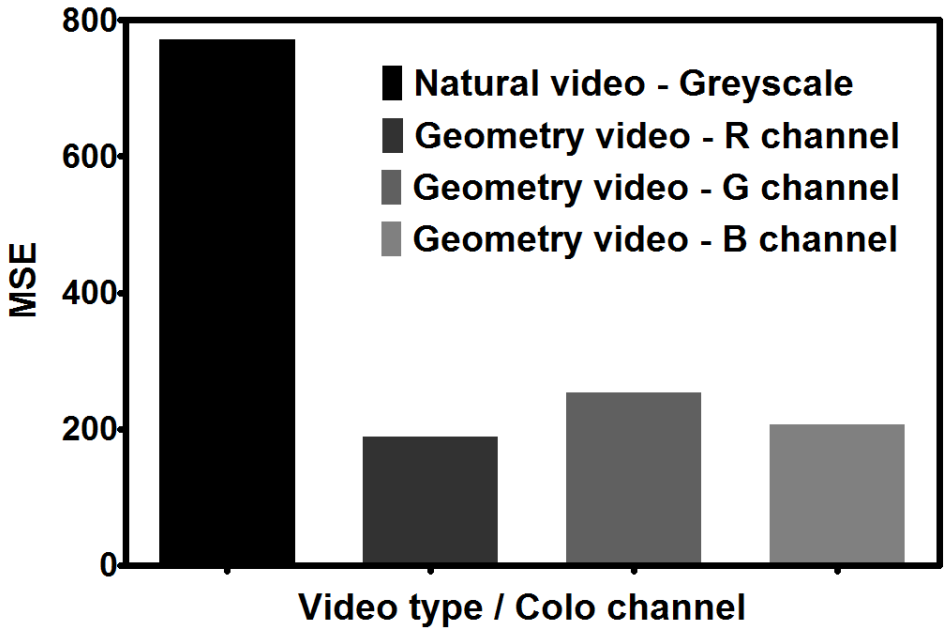


Figure 8.2: Mean square error (MSE) comparison for natural video and geometry video after a 32x32 Full Search motion estimation of H.264/AVC.

follow a special local variant pattern in that region. This special feature of geometry videos, however, has not been realized and utilized by any intra-frame prediction schemes.

To achieve even lower bitrates for compression, we have tailored the intra-frame prediction scheme in H.264/AVC to adhere the local variant pattern.

As we mentioned before, the original H.264/AVC has provided a series of prediction modes showing an insufficient respect to the local pixel variance. In this work, we have added 4 extra intra-frame prediction modes to the original H.264/AVC in our tailored scheme. These extra modes will form predicted pixels by considering the variant pattern of neighbor pixels. For a 4x4 block in a video picture, our simple yet very effective prediction modes can be written as:

$$P(x,y)=\begin{cases} P(x,y-1)+[N(x-1,y-1)-N(x-1,y)], & \text{mode 1} \\ P(x,y-1)+\frac{1}{4}\sum_{i=1}^{4}[N(x-1,y+i)-N(x-1,y+i-1)], & \text{mode 2} \\ P(x-1,y)+[N(x,y-1)-N(x-1,y-1)], & \text{mode 3} \\ P(x-1,y)+\frac{1}{4}\sum_{i=1}^{4}[N(x-1,y+i)-N(x+i-1,y-1)], & \text{mode 4} \end{cases}$$

where $N(x,y)$ denotes the neighbor pixel locating at position (x,y) and $P(x,y)$ being the predicted pixels for the block.

The tailored H.264/AVC encoder will then select the best prediction mode from the original H.264/AVC modes and our extra modes based on minimum prediction errors.

These extra prediction modes were only used for 4x4 blocks in a video picture (as our investigation shows that they will not provide significant gains for 16x16 and 8x8 blocks). The extra prediction modes may slightly impose the computational overhead on the encoder. However, the overhead is neglectable compared with those higher-complexity components in H.264/AVC, e.g. motion estimation.

In the next section, we will show that the tailored prediction scheme yields better prediction results for intra-frames and therefore provides better references to the remaining predictive frames in the GOP, consequently resulting in bitrate reductions for the entire video sequence.

Results There are 4 test video sequences containing human expressions captured by our camera from 4 subjects, who were asked to make varied face expressions, e.g. laughing, shouting, etc., during the video capturing. These 4 sequences are named as "HumanFace1" to "HumanFace4". Each of the 4 sequence consists of 200 frames with the resolution of 320x320. For each video sequence, we compress it with H.264/AVC at different bitrates and then decompress and reconstruct the video. The peak signal to noise ratio (PSNR) is used to measure the reconstructing quality of the video.

In addition to the real human expressions, we also tested 2 sequences containing synthetic 3D data including a synthetic face model and a synthetic horse model, which were named as "SynthFace" and "SynthHorse". The motion in these 2 sequences is composed of a few rigidly moving parts, such as the horse gallop shown in Fig. 8.4. For these 2 sequences, we parameterized the motion to a polycube domain which mimics the geometry of the model, e.g. Fig. 8.4(b) for the horse model. Then we constructed the GV by flattening the polycube to a rectangular domain.

The video compression tool used in this work is the JM14.2 H.264/AVC reference software. The GVs were compressed by the original H.264/AVC as well as our tailored H.264/AVC as described in Section 6. The High 4:4:4 profile of H.264/AVC FREXt [23] was adopted in this work to keep high-fidelity of the motion data, i.e. each color channel of a GV picture is equally compressed and there is no subsampling used. The test videos were encoded at 30 frames per second, with Group of Picture (GOP) structure of "IPBPB", Fast Full Search motion estimation (32x32 search range, 5 reference frames and variable block sizes) and CABAC entropy coding. Rate Distortion Optimization (RDO) was turned on. The intra-frame period (interval) was set to 30 (i.e. 1 intra-frame per second). The GVs were compressed at different bitrates by adjusting the quantization parameters.

Fig. 8.3 compares the rate-distortion performance of the original H.264/AVC and our tailored H.264/AVC. The QPs range from -12 to +12 (we have adopted some negative QPs in order to encode high-quality GVs. It has been observed that both the original H.264/AVC and our tailored H.264/AVC can compress GVs into a compact size. Compared to the original H.264/AVC, our tailored H.264/AVC is able to further reduce the bitrate by 1.02% to 5.56% while increasing the PSNR by 0.04dB to 0.22dB for HumanFace GV due to the better utilization of the spatial feature of GVs. For the synthetic GVs (SynthFace and SynthHorse), the bitrate reduction obtained by our tailored

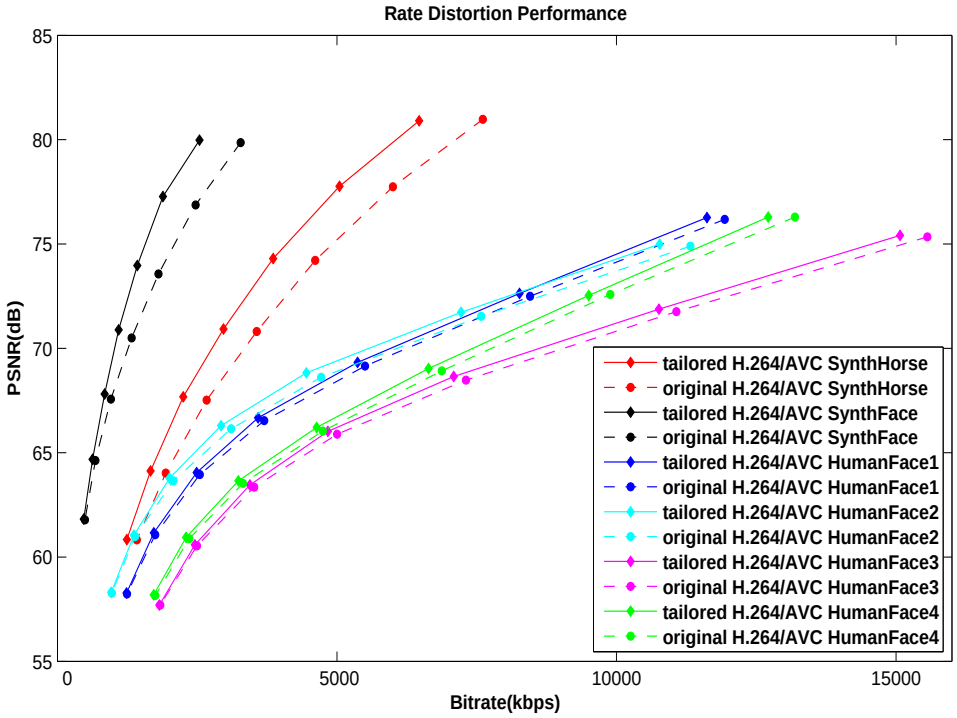
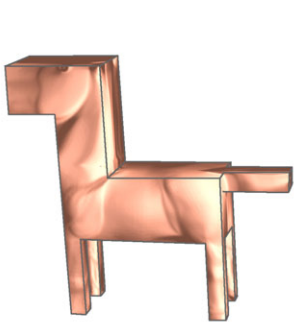


Figure 8.3: Rate-distortion performance of original and tailored H.264/AVC

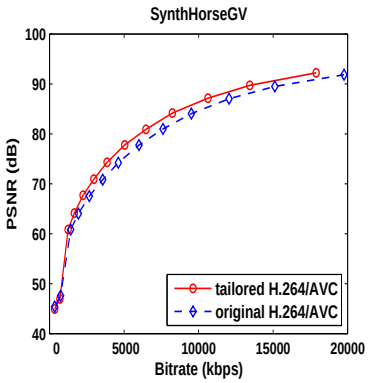
H.264/AVC over original H.264/AVC can be up to 20%, which is much more significant than those for HumanFace GVs. This is because that these synthetic GVs are articulated animation containing less noise. Also, the surfaces of the synthetic models are smoother compared to HumanFace GVs thus enabling better prediction results of our tailored H.264/AVC.

Fig. 8.5 (a)-(d) display sample frames of original HumanFace1 and the reconstructed video by the tailored H.264/AVC with different compression ratios. We observed that the tailored H.264/AVC will result in no visual distortion at low compression ratio (21:1). At medium compression ratio (212:1), the decoded frames showed a acceptable quality but there small deformation can be found in the boundary of the human face. At very high compression ratio (575:1), the decoded frames are visually distorted.

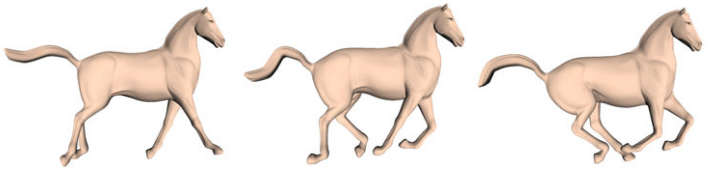


(a)Polycube mapped horse

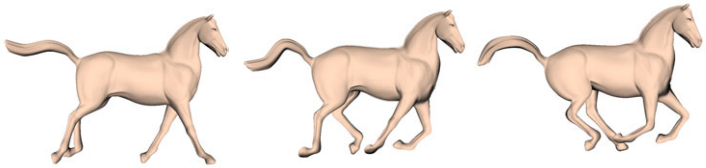
(b)Parametric domain



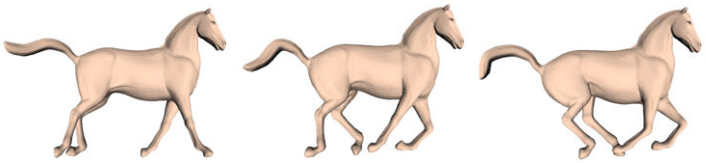
(c)Rate distortion



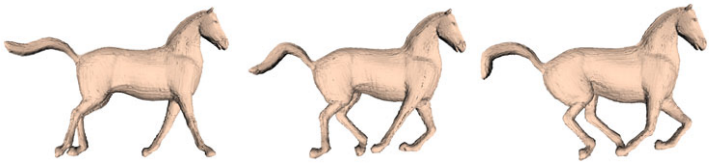
(d)Original data



(e)Compression ratio 48:1
PSNR 81.05dB

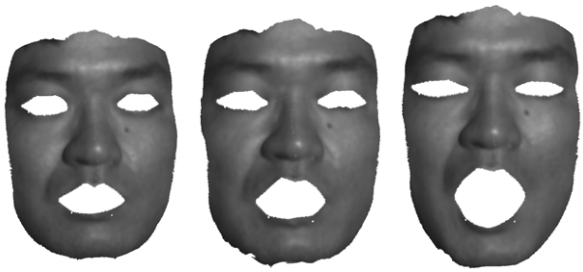


(f)Compression ratio 140:1
PSNR 67.68dB



(g)Compression ratio 466:1
PSNR 47.98dB

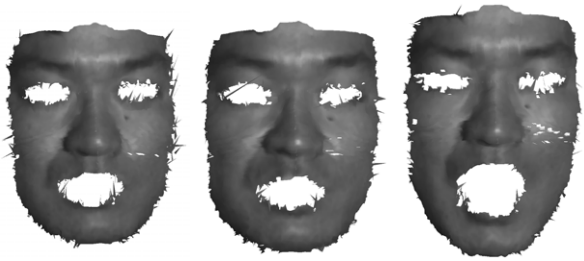
Figure 8.4: Geometry video of articulated motion (HorseGv). (a) maps the surface of horse onto polycube; (b) parameterizes the mapped polycube into rectangular domain; (c) shows the rate-distortion (PSNR vs. Bitrates) performance of the original H.264/AVC and the tailored H.264; (d) shows the original video sequence; (e), (f) and (g) show the reconstructed videos at low, medium and high compression ratios by using the tailored H.264/AVC.



(a)Original data

(b)Compression ratio 21:1, PSNR 71.73dB

(c)Compression ratio 212:1, PSNR 58.31dB



(d)Compression ratio 575:1, PSNR 46.00dB

Figure 8.5: Geometry video of HumanGV1. Row 1 shows the original video sequence; Row 2, 3 and 4 show the reconstructed videos at low, medium and high compression ratios by using the tailored H.264/AVC.

Fig. 8.4 shows sample frames of original SynthHorse and the reconstructed video sequences. We observed from the figure the similar results to HumanFace GVs, i.e. the reconstructed video frames are of good quality at low and medium compression ratios.

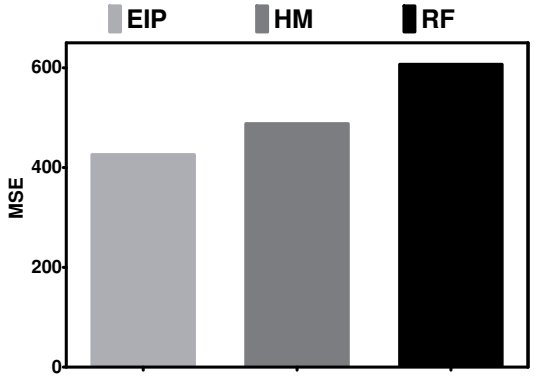


Figure 8.6: Compared to other parameterization techniques, such as Ricci flow(RF) and harmonic map(HM). The constructed geometry videos have better spatial and temporal coherence than that of other methods. As shown in (e), EIP leads to stronger temporal coherence evidenced by smaller MSE produced after a 32x32 Full Search motion estimation (QP=12).

From the temporal coherence perspective, Fig. 8.6 compares the average MSE of HM and RF after a 32x32 Full Search motion estimation of H.264/AVC (for GV2, 3 color channels and QP=12 as an example). The results show that our proposed expression-invariant parameterization algorithm can achieve smaller MSE than other methods.

To sum up, our experimental results show that the GVs can be significantly compressed by using the original H.264/AVC while maintaining the video reconstruction quality. By using our tailored H.264/AVC, a further improvement in bitrate reduction can be achieved with even increased the PSNR .

8.3 Volume Data Processing

In this section, we demonstrate the efficiency and quality of our volume parameterization algorithm to several applications, including hexahedral mesh generation, volumetric morphing, anisotropic solid texture transfer, and GPU-based volumetric computing.

Hexahedral Mesh Generation Note that the polycube domain has very natural hexahedral tessellation. The volumetric parameterization induces a bijective map between the polycube and the given volume. Thus, it also induces a hexahedral tessellation in the volume dataset as shown in Fig 8.7.

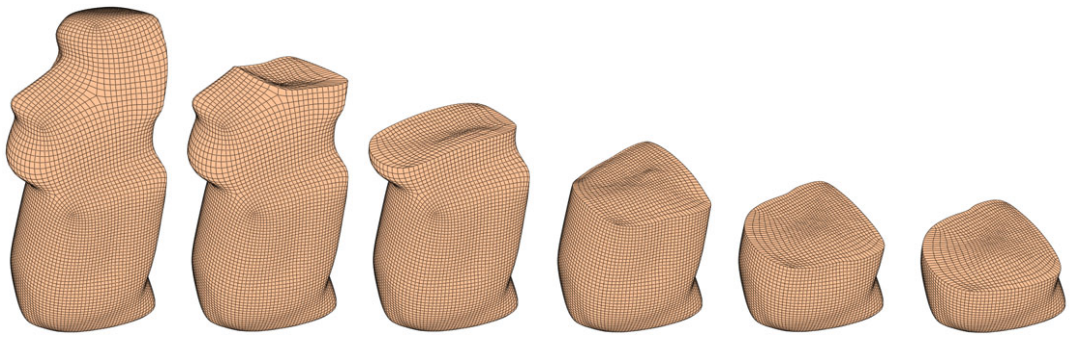


Figure 8.7: Hexahedral mesh generation using volumetric polycube parameterization.

Volumetric Morphing. To morph from one star-shape to another, we parameterize them to a common parametric domain, such as a ball and then determine a smooth map (e.g., identity map in our current implementation) between the balls. Figure 8.8 shows a running example from a star to Venus head. Figure 8.10, Figure 8.11, and Figure 8.12 shows more result of shape morphing and induced interior structure transfer.

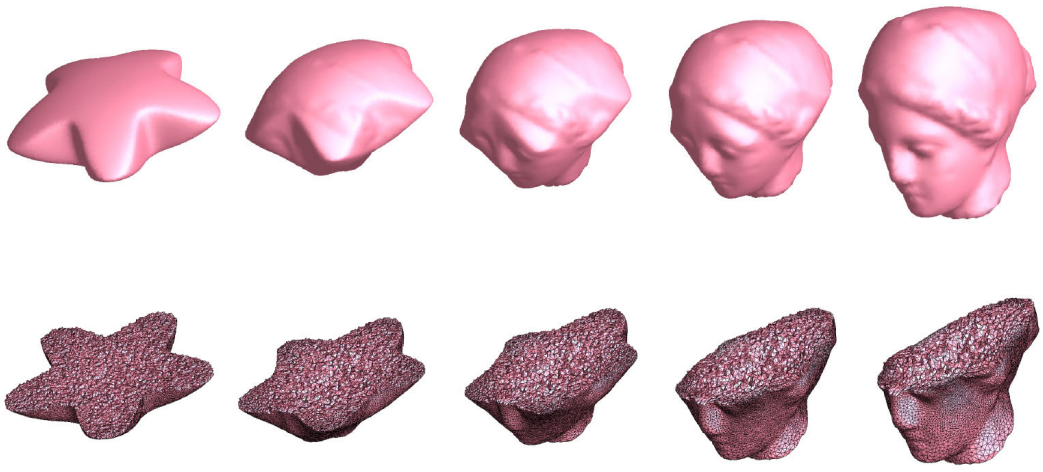


Figure 8.8: Volumetric morphing between the star and the Venus head.

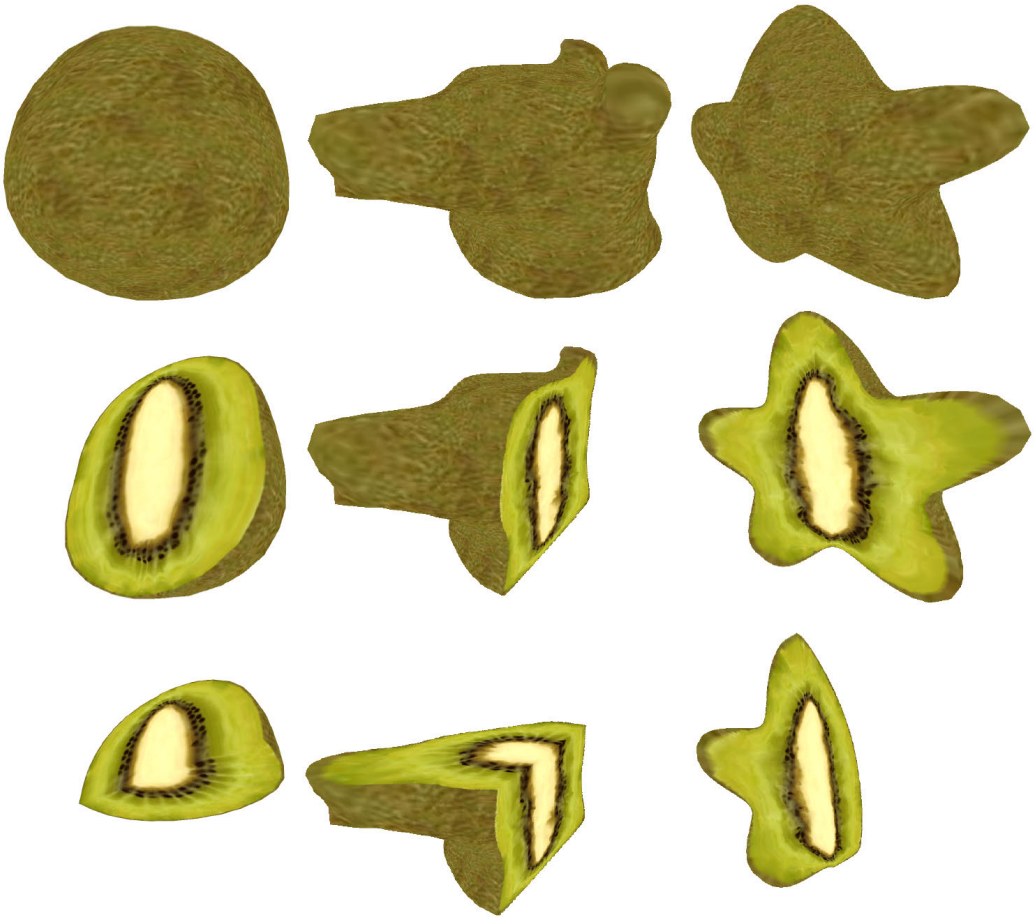


Figure 8.9: Transferring the anisotropic solid texture from kiwi (column 1) to star shapes (column 2: dog head; column 3: star).

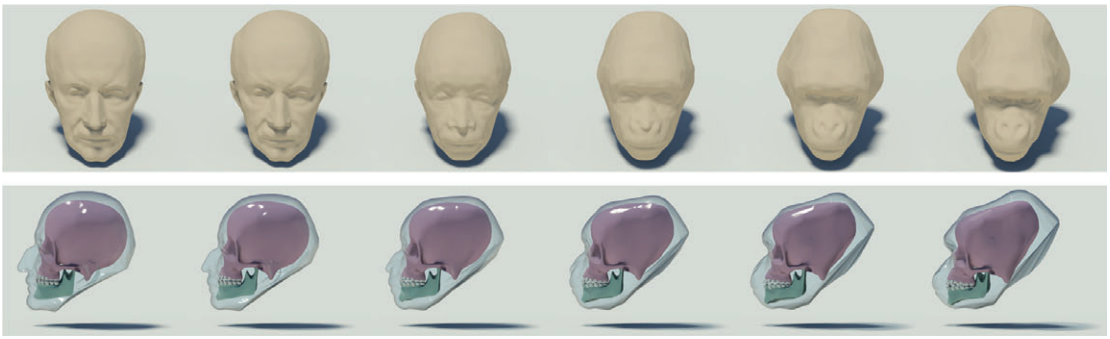


Figure 8.10: shape and interior structure morphing of Skull

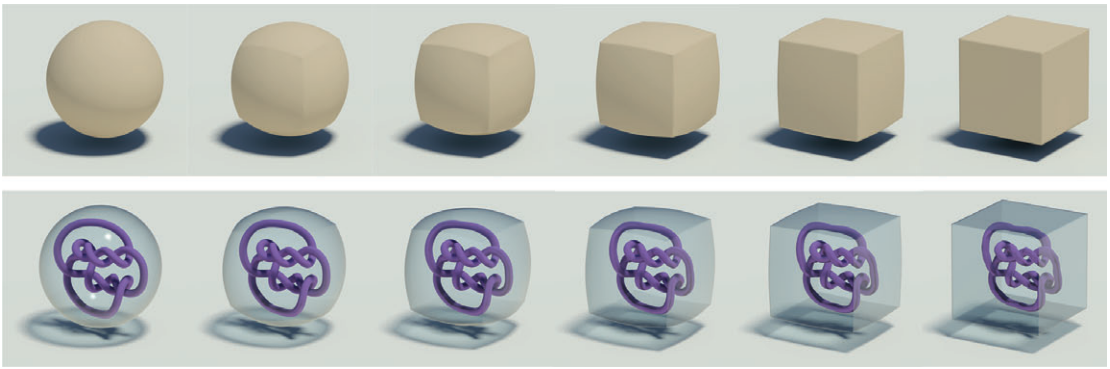


Figure 8.11: Shape and interior structure morphing of Knot

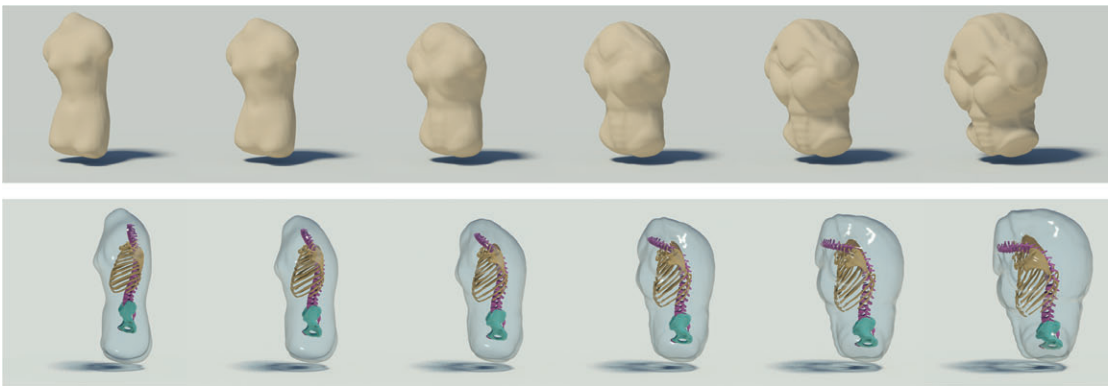


Figure 8.12: Shape and interior structure morphing of Torso

Anisotropic Solid Texture. Solid textures [42], or anisotropic solid textures [73], allow us to fill the interior of 3D models with spatially-varying and anisotropic texture patterns. Takayama *et al.* [73] proposed a lapped texture approach [60] to synthesize anisotropic solid textures by pasting solid texture exemplars [42] repeatedly over the

tetrahedron structure of 3D geometries. This approach can result in high-quality and large-scale solid textures with low computation cost; to create such a texture, the user, however, has to mark up volumetric tensor field and edit the texture in a geometry-dependent fashion.

Our star-shape volume parameterization method can further broaden the applicability of the lapped solid texture results to a larger pool of geometric models. As illustrated in Figure 8.9, we can first parameterize a given star shape that has been pre-synthesized with lapped solid texture to a solid ball using the Green's function; hence, we can transfer the synthesized texture information from the input geometric model to our star-shape model through the common parametric ground. Our approach allows the reuse of synthesized anisotropic solid textures without incurring additional texture synthesis. Furthermore, since our volume parameterization method is a diffeomorphism, we can guarantee the bijectivity and smoothness in the texture transfer process, as demonstrated in Figure 8.9.

GPU-based Volumetric Computation. Another advantage of having a smooth volume parameterization is the luxury of being able to perform computations throughout the volume by taking the computation process to the highly-structured parametric domain. Here we can parameterize the given star-shape model by a cube model (or polycube) so that the data inside the parametric domain can naturally be modeled by a 3D texture; as a result, we can carry out the computation on the GPU and further accelerate the computation performance.

In detail, we first parameterize a given star-shape model by a cube shape so that the reaction-diffusion data (concentration values, etc.) can be naturally modeled as 3D textures stored in our GPU implementation. Here, we employ and extend Turing's

reaction-diffusion model [77] to three-dimensional, and expectedly, 3D sphere-like spot patterns will be developed when the chemical concentrations reach a dynamic equilibrium state. Furthermore, we employ Witkin and Kass's method [84] to account for the distortion caused by the parameterization (since we compute the reaction-diffusion on the parameterization grid): Given the parameterization from star-shape to cube, we compute the local Jacobian per voxel element over the 3D parameterization grid; then, we can compute the metric tensor as a three-by-three matrix $M = J^T J$. Hence, we can adaptively and locally modify the rate of diffusion by the diagonal values in the metric matrix; this allows us to temper the reaction-diffusion pattern, thereby compensating the volumetric distortion in the parameterization. Figure 8.13 shows the reaction-diffusion results inside the Venus head model.

8.4 Shell Space Construction

Shell objects are three-dimensional structures wherein one dimension, the thickness, is much smaller than the other two dimensions. Shell structures are widely used in manufacturing, such as for automobile bodies, sheet metal parts, etc. On the other hand, hexahedral meshes have promising properties for finite element analysis. But construction of hexahedral mesh is more challenge than construction of tetrahedral mesh. This section focus on hexahedral meshing of shell volume.

Our hexahedral meshing algorithm consists of three steps, constructing the shell space, parameterizing the shell space and hexahedral meshing.

- In Step 1, we construct the offset surface using the user-specified thickness. Then we tessellate the shell space using a tetrahedral mesh. We also parameterize the given boundary surface to a polycube and construct the shelled polycube in a

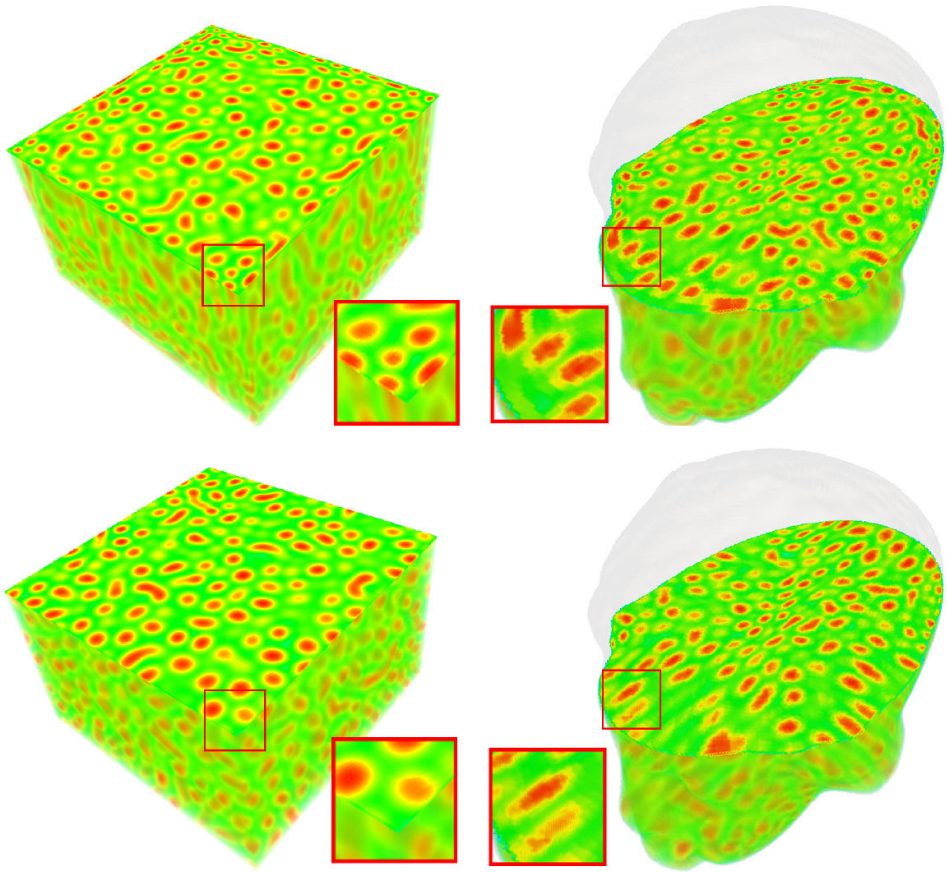


Figure 8.13: Volumetric reaction-diffusion computation on the GPU. The first column shows the reaction-diffusion results in equilibrium state over the cube-based parametric domain, whereas the second column shows the reaction-diffusion results after mapped to the star-shaped model. Without distortion compensation by the metric matrix, we can see distortion in the lower right pattern, but such a distortion can be corrected (see upper row) if we take the metric matrix into account.

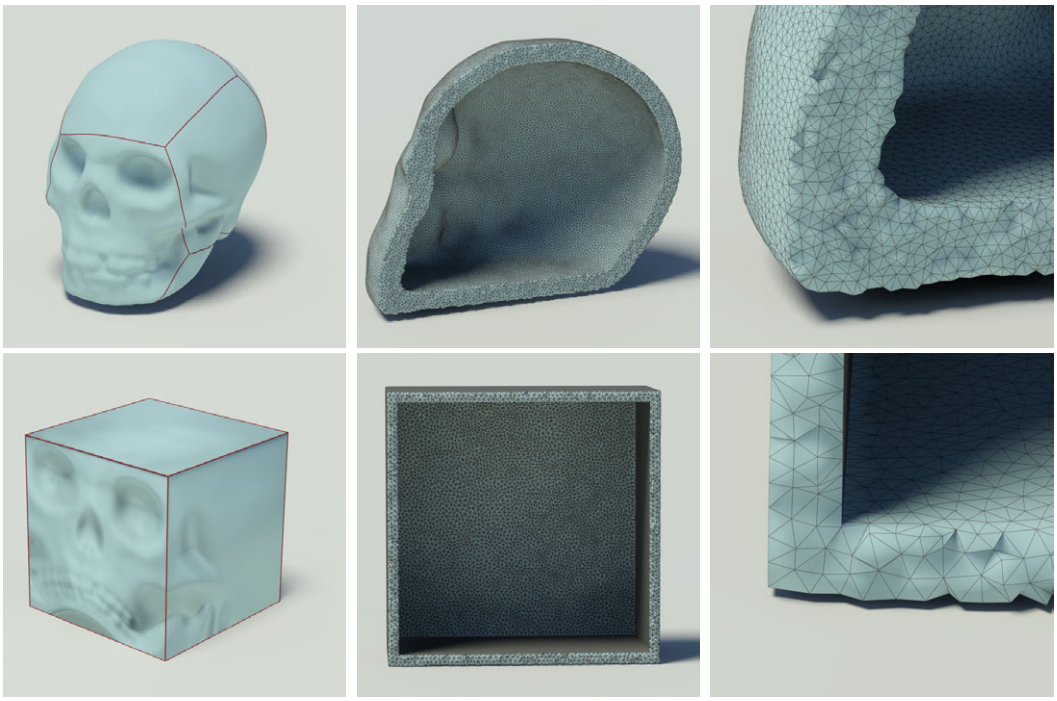
similar fashion. The method of polycube mapping has been proposed in Chapter 4

- In Step 2, we compute the harmonic field in the shell spaces by solving the Laplace equation with a Dirichlet boundary condition. By tracing the integral curves, we build a bijection between the shell spaces. The integral curve tracing algorithm has been presented in Chapter 6.
- In Step 3, we tessellate the polycube space with a regular hexahedral mesh, then construct the layered hexahedral mesh in the object space via volumetric parameterization.

In the following, we look into the details of shell space construction.

The shell space is enclosed by the give 3d model and an offset surface which are two disconnected closed surfaces. For the offset surface, The offset distance is specified by the user. We use the MPU method [56] to construct the distance field of the input model, then extract the iso-surface whose iso-value is the user-specified thickness. We then construct a tetrahedral mesh to fill the shell space using Tetgen [70]. We also apply the variational meshing algorithm [2] to improve the quality of the tetrahedral mesh.

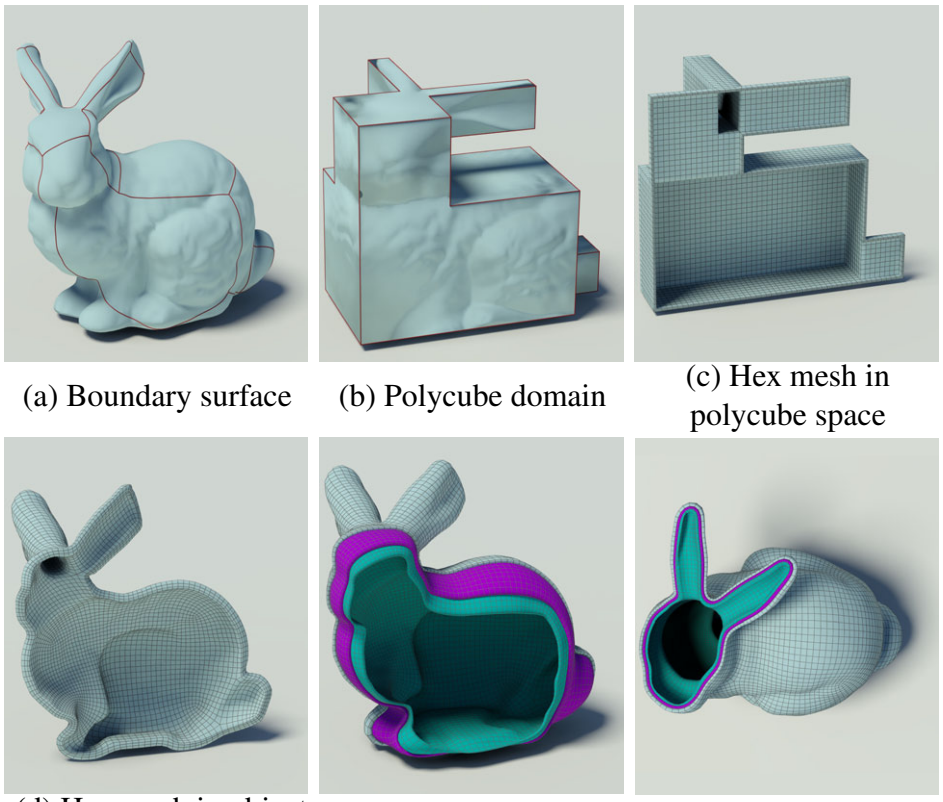
Using the polycube map method proposed in chapter 4, we map the boundary surface M_0 to the user-constructed polycube domain. Our resulting polycube map has low angle distortion and is guaranteed to be a bijection. When constructing the tetrahedral mesh for the polycube space, we must pay special attention to the sharp features (such as edges and corners) of the polycube. To preserve the features in the isotropic tetrahedral mesh, we follow the variant variational meshing technique that is tailored for mechanical models [71]. Figure 8.14 shows the construction of shell space for the Skull model and its polycube domain.



(a) Boundary surface (b) Shell space (c) Close-up view

Figure 8.14: Shell space construction. Row 1: we construct a distance field for the given model and extract an iso-surface with the user-specified offset distance. Then, we construct an isotropic tetrahedral mesh using variational meshing technique [2]. Row 2: we map the outer boundary surface to a polycube and construct the tetrahedral mesh of the shell space in a similar way. Note the sharp features (polycube edges and corners) are well preserved in the tetrahedral mesh.

Results We tested our algorithm using models of various topology. In our experiments, the user-specified offset distance (i.e., thickness) can be either positive or negative. The user also specifies the number of layers in the constructed hexahedral meshes. As mentioned before, each layer has exactly the same tessellation. Figure 8.16 and 8.17 show the layered hexahedral meshes. Cutaway views are used to show the high quality of the hexahedral inside the models.



(a) Boundary surface (b) Polycube domain (c) Hex mesh in polycube space

(d) Hex mesh in object space (e) Cut view

Figure 8.15: Constructing layered hexahedral mesh for shelled Bunny model. Given a closed 2-manifold and a user-specified offset distance, we first construct the shell space using the distance field and then parameterize the shell space to a shelled polycube domain. As there is a natural hexahedral tessellation in the polycube domain, the volume parameterization induces the hexahedral mesh in the object space. The constructed mesh is an all-hexahedral mesh that most of the vertices are regular, i.e., the valence is 6 for interior vertices and 5 for boundary vertices. The mesh also has a layered structure that all layers have exactly the same tessellation.

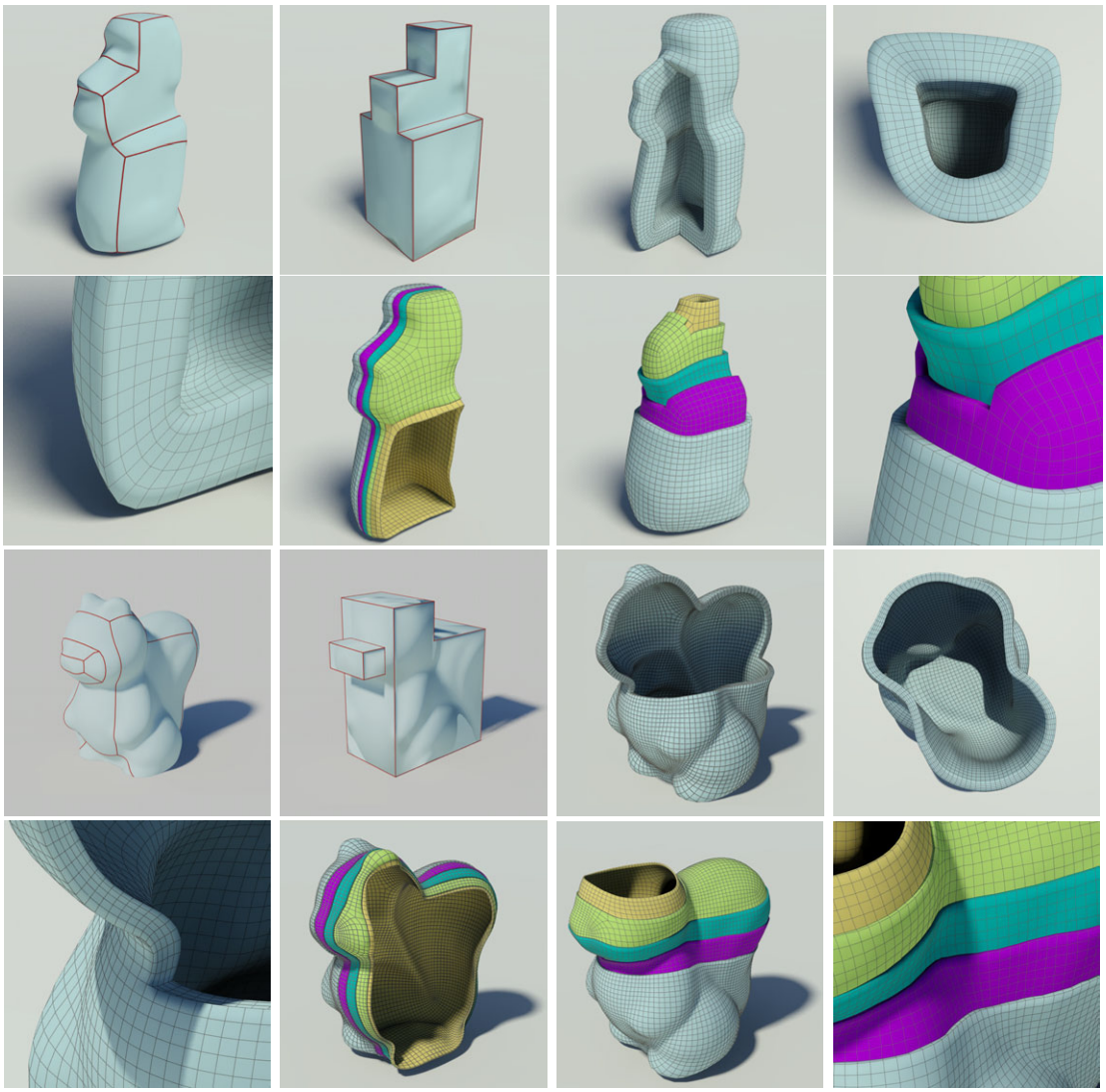


Figure 8.16: Experimental results on genus-0 models

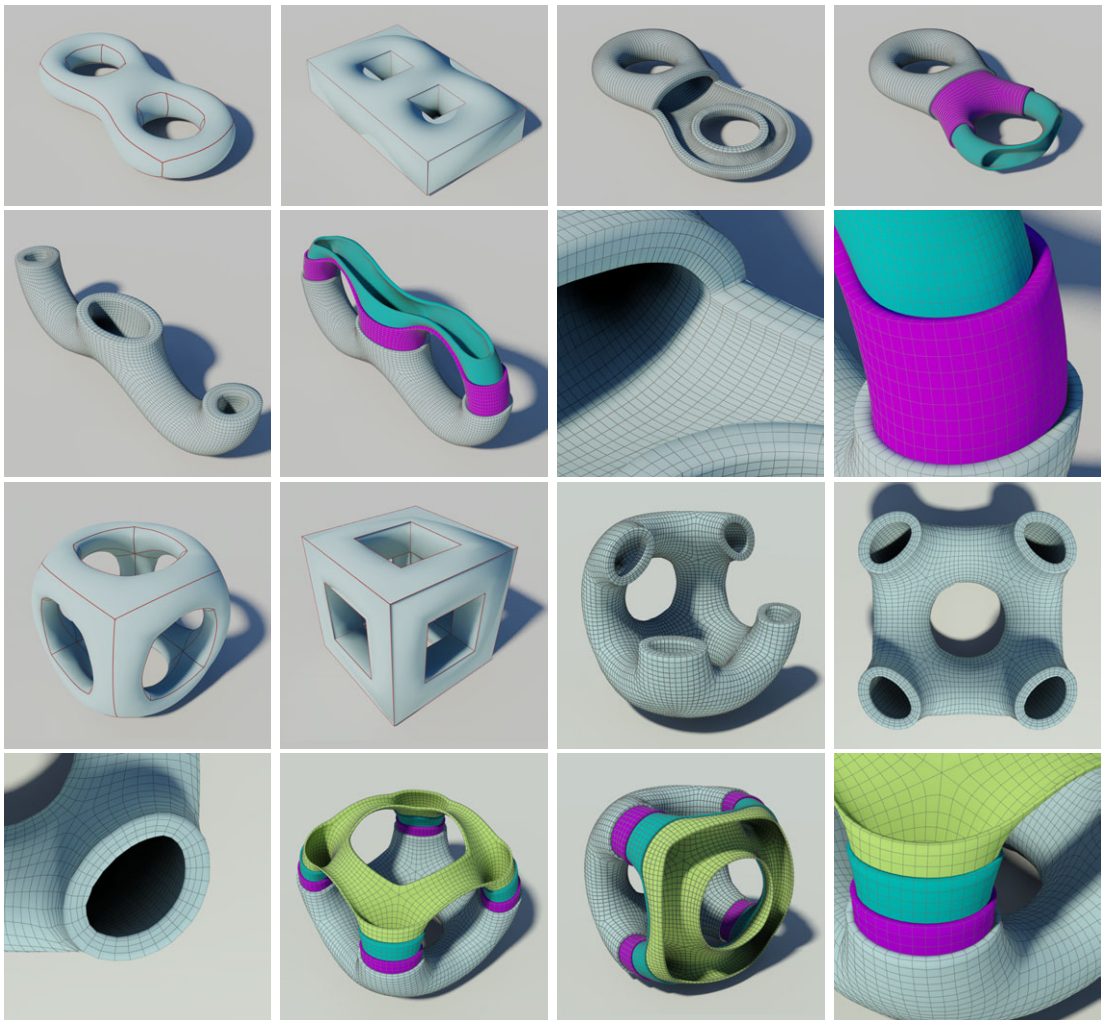


Figure 8.17: Experimental results on high genus models

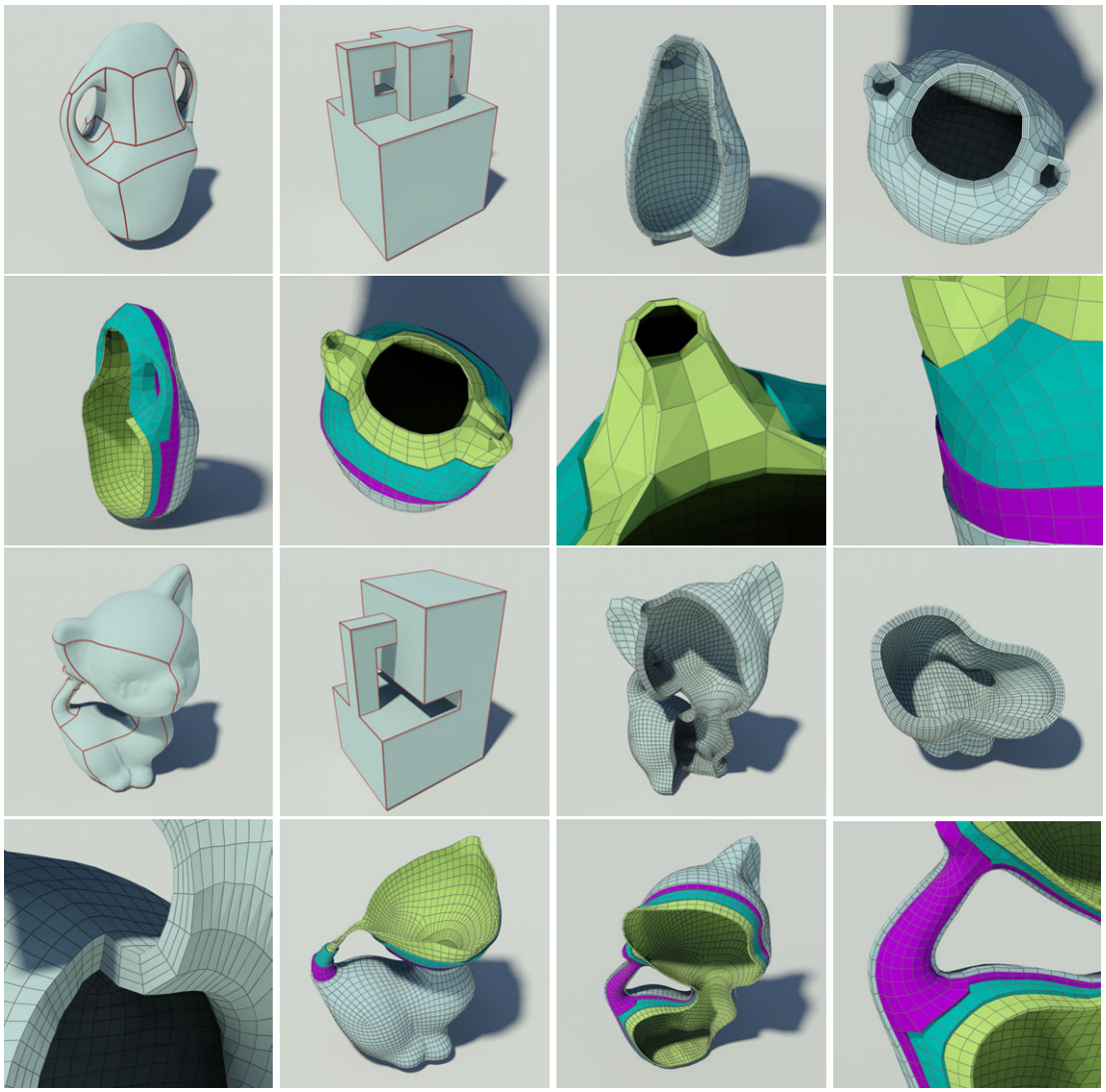


Figure 8.18: Experimental results

Chapter 9

Conclusion and Future Work

9.1 Conclusion

In this thesis, we have systematically studied the harmonic field-based surface and volume mapping, and then developed a set of algorithms to compute such mappings for real-world models.

We have managed to overcome the drawbacks of harmonic maps mentioned in Section 2.2. We proposed two elegant techniques of harmonic-based surface parameterization which allow for the mapping of constraints inside the domain rather than only on boundaries. There are many promising properties in allowing for interior mapping constraints. First, salient features could be aligned accurately. Second, interior constraints could lead to smaller angle or area distortions. Third, it allows more flexible user control operations and an interactive mapping framework. In volume parameterization, we proved the existence of a diffeomorphism between two star shapes. Our findings in volumetric parameterization have significantly contributed to the extension of harmonic maps into 3D cases.

More specifically, we presented an editable polycube map framework that, given an arbitrary high-resolution polygonal mesh and a simple polycube representation plus

optional sketched features indicating relevant correspondences between the two, provides a uniform, regular and artist-controllable quads-only mesh with a parameterized subdivision displacement scheme. Flexible mapping constraints are allowed in our framework, which is important for an interactive parameterization method. The proposed method guarantees that the computed map is bijective and conformal except at a finite number of extraordinary points (the polycube corners). Subjective and statistical comparisons between our method and previous methods demonstrated that our method outperforms its predecessors significantly. Furthermore, our method is based on a divide and conquer strategy, which enables the processing of large-scale models with complex geometry and topology. In our experiment, a large-scale model which contains 500k vertices is processed well. Based on our polycube mapping, we provided a subdivision displacement scheme that is specially built for quad patch-based tessellation at the GPU for object and character rendering. The promising experimental results demonstrated the efficacy of our method.

We also proposed an algorithm that parameterizes 3D facial expressions with guaranteed feature correspondence and stores them into a video format, hence allowing the 3D data to be significantly compressed by well-studied video compression techniques. Our experimental results demonstrate the expression-invariant property of our method. Compared to other parameterization methods, such as harmonic maps and discrete Ricci flow, our method can lead to results that are highly consistent and insensitive to the expressions, and have a greater degree of coherence of constructed geometry videos, which is highly desirable for video compression.

In volume parameterization, we proved that the Green's function on a star shape has a unique critical point and all level sets inside the star shape are topological spheres. We also showed that the Green's function can induce a diffeomorphism between two

star shapes. To our knowledge, this is the first constructive proof of the existence of a diffeomorphism between two non-trivial shapes. Based on our theoretical results, we developed algorithms to parameterize star shapes to star-shaped domains, such as solid balls and star-shaped polycubes. We further extended our algorithm to parameterize general volumes by using star-shaped decomposition. An interesting comparison is given between our harmonic field-based algorithm and the volumetric harmonic map method. The results show that our method is more robust and leads to hexahedral meshes with better quality. The advance of diffeomorphism is demonstrated by the smooth parametric curves.

We also proposed an algorithm for parameterizing volumes of handlebodies. The proposed algorithm is able to parameterize volumes with complex topology. It can also be downgraded and applied to volumes with trivial topology (i.e. topological balls). The parametric domain is the direct product of a surface patch and a line segment. The resulting map between the original volume and the parameter domain is a bijection without any singularity. Also, at any point in the volume, the w iso-parametric line (which follows the gradient of the harmonic field) is orthogonal to the u and v iso-parametric lines (which span the iso-surface of the harmonic field), which is a natural consequence of enforcing a conformal parameterization on the surface and a gradient field in the volume. Our experimental results demonstrate our highly regular parametric curves. We have also showcased a variety of applications that benefit from our volume parameterization method, including volumetric morphing, anisotropic solid texture transfer, GPU-based volumetric computation and shell space construction.

9.2 Future Work

This work paves the way for several future research directions. First, our current surface and volume parameterization methods highly rely on tracing the integral curves of the gradient vector field of the harmonic fields. Thus, the parameterization quality depends on the quality of the input triangle and tetrahedral meshes. We will develop robust algorithms to trace integral curves in meshes with poor triangulation. Second, our current volume parameterization methods work for volumes without any voids. To our knowledge, it is an open problem in the parameterization of volumes with arbitrary voids. In the future, we will investigate along this direction. Third, besides the applications presented in this thesis, parameterization can also serve as a fundamental and powerful tool of geometric modeling. It has strong potential in the scientific computation and engineering fields. In the future, we would like to promote the wide spectrum of applications of our parameterization methods.

References

[1] M. Alexa. Merging polyhedral shapes with scattered features. *The Visual Computer*, 16(1):26–37, 2000.

[2] P. Alliez, D. Cohen-Steiner, M. Yvinec, and M. Desbrun. Variational tetrahedral meshing. *ACM Transactions on Graphics*, 24(3):617–625, 2005.

[3] D. Attali, D. Cohen-Steiner, and H. Edelsbrunner. Extraction and simplification of iso-surfaces in tandem. In *Symposium on Geometry Processing*, pages 139–148, Vienna, Austria, 2005.

[4] O. K.-C. Au, H. Fu, C.-L. Tai, and D. Cohen-Or. Handle-aware isolines for scalable shape editing. *ACM Transactions on Graphics*, 26, July 2007.

[5] C. Bennis, J. Vézien, and G. Iglésias. Piecewise surface flattening for non-distorted texture mapping. In *ACM SIGGRAPH*, pages 237–246, New York, NY, USA, 1991.

[6] I. Boier-Martin, H. Rushmeier, and J. Jin. Parameterization of triangle meshes over quadrilateral domains. In *Symposium on Geometry Processing*, pages 193–203, New York, NY, USA, 2004.

[7] I. Castano. Next-generation rendering of subdivision surfaces. Los Angeles, California, USA, 2008.

[8] I. Castano. Tessellation of subdivision surfaces in direct3d 11. Gamefest speech, 2008.

[9] J. Cohen, M. Olano, and D. Manocha. Appearance-preserving simplification. In *ACM SIGGRAPH*, pages 115–122, New York, NY, USA, 1998.

[10] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(23):681–685, 2001.

[11] P. Degener, J. Meseth, and R. Klein. An adaptable surface parameterization method. In *9th International Meshing Roundtable*, pages 201–213, New Orleans, Louisiana, USA, 2003.

[12] M. Desbrun, M. M., and P. Alliez. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum*, 21(3):209–218, 2002.

[13] S. Dong, S. Kircher, and M. Garland. Harmonic functions for quadrilateral remeshing of arbitrary manifolds. *Comput. Aided Geom. Des.*, 22(5):392–423, 2005.

[14] E. Drumwright. A fast and stable penalty method for rigid body simulation. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):231–240, 2008.

[15] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In *SIGGRAPH '95*, pages 173–182, New York, NY, USA, 1995.

[16] G. Fairweather and A. Karageorghis. The method of fundamental solution for elliptic boundary value problems. *Advances in Computational Mathematics*, 9(1-2):69–95, 1998.

[17] Z. Fan, X. Jin, J. Feng, and H. Sun. Mesh morphing using polycube-based cross-parameterization. *Journal of Visualization and Computer Animation*, 16(3-4):499–508, 2005.

[18] D. A. Field. Laplacian smoothing and delaunay triangulations. *Communications in Applied Numerical Methods*, 4(6):709–712, 1988.

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

[20] M. S. Floater. Mean value coordinates. *Computer Aided Geometric Design*, 20(1):19–27, 2003.

[21] M. S. Floater and K. Hormann. Surface parameterization: a tutorial and survey. In Neil Anthony Dodgson, Michael S. Floater, and Malcolm Arthur Sabin, editors, *Advances in Multiresolution for Geometric Modelling*, pages 157–186. Berlin, Heidelberg, 2005.

[22] M. S. Floater, G. Kós, and M. Reimers. Mean value coordinates in 3d. *Computer Aided Geometric Design*, 22(7):623–631, 2005.

[23] J. S. Gary, P. Topiwala, and A. Luthra. The h.264/avc advanced video coding standard: Overview and introduction to the fidelity range extensions. In *SPIE conference on Applications of Digital Image Processing XXVII*, pages 454–474, Denver, CO, USA, 2004.

[24] J. J. Gergen. Note on the green function of a star-shaped three dimensional region. *American Journal of Mathematics*, 53(4):746–752, 1931.

[25] C. Gotsman, X. Gu, and A. Sheffer. Fundamentals of spherical parameterization for 3d meshes. *ACM Transactions on Graphics*, 22(3):358–363, 2003.

[26] X. Gu, S. J. Gortler, and H. Hoppe. Geometry images. *ACM Transactions on Graphics*, 21(3):355–361, 2002.

[27] X. Gu, S. Wang, J. Kim, Y. Zeng, Y. Wang, H. Qin, and D. Samaras. Ricci flow for 3d shape analysis. In *International Conference on Computer Vision*, pages 1–8, Rio de Janeiro, Brazil, 2007.

[28] X. Gu, Y. Wang, T. F. Chan, P. M. Thompson, and S.-T. Yau. Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE Transactions on Medical Imaging*, 23(8):949–958, 2004.

[29] X. Gu and S.-T. Yau. Computing conformal structures of surfaces. *Communications in Information and Systems*, 2:121–146, 2002.

[30] X. Gu and S.-T. Yau. Global conformal surface parameterization. In *Symposium on Geometry Processing*, pages 127–137, Aire-la-Ville, Switzerland, Switzerland, 2003.

[31] X. Gu and S.-T. Yau. *Computational conformal geometry*. International Press, 2006.

[32] S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro, and M. Halle. Conformal surface parameterization for texture mapping. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):181–189, 2000.

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

[34] K. Hormann and G. Greiner. Mips: An efficient global parametrization method. In *Curve and Surface Design: Saint-Malo*, pages 153–162, Nashville, TN, 2000.

[35] K. Hormann, B. Lévy, and A. Sheffer. Mesh parameterization: theory and practice. In *ACM SIGGRAPH 2007 courses*, New York, NY, USA, 2007.

[36] M. Isenburg, S. Gumhold, and C. Gotsman. Connectivity shapes. In *Proceedings of the conference on Visualization*, pages 135–142, Washington, DC, USA, 2001.

[37] T. Ju, S. Schaefer, and J. D. Warren. Mean value coordinates for closed triangular meshes. *ACM Transactions on Graphics*, 24(3):561–566, 2005.

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

[39] T. Kanai, H. Suzuki, and F. Kimura. 3d geometric metamorphosis based on harmonic map. In *Proceedings of the 5th Pacific Conference on Computer Graphics and Applications*, pages 97–104, Washington, DC, USA, 1997.

[40] L. Kharevych, B. Springborn, and P. Schröder. Discrete conformal mappings via circle patterns. In *ACM SIGGRAPH 2005 Courses*, New York, NY, USA, 2005.

[41] L. P. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel. A shrink-wrapping approach to remeshing polygonal surfaces. *Computer Graphics Forum*, 18(3):119–130, September 1999.

[42] J. Kopf, C.-W. Fu, D. Cohen-Or, O. Deussen, D. Lischinski, and T.-T. Wong. Solid texture synthesis from 2d exemplars. *ACM Transactions on Graphics*, 26(3):2:1–2:9, 2007.

[43] F. Labelle and J. R. Shewchuk. Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. *ACM Transactions on Graphics*, 26(3):57, 2007.

[44] S. Lanquetin and M. Neveu. Reverse catmull-clark subdivision. In *WSCG*, pages B89–B96, Plzen, Czech Republic, 2006.

[45] Y. Lee, H. S. Kim, and S. Lee. Mesh parameterization with a virtual boundary. *Computers & Graphics*, 26(5):677–686, 2002.

[46] B. Lévy. Constrained texture mapping for polygonal meshes. In *ACM SIGGRAPH*, pages 417–424, New York, NY, USA, 2001. ACM.

[47] X. Li, X. Guo, H. Wang, Y. He, X. Gu, and H. Qin. Harmonic volumetric mapping for solid modeling applications. In *Symposium on Solid and Physical Modeling*, pages 109–120, Beijing, China, 2007.

[48] J. Lin, X. Jin, Z. Fan, and C. C. L. Wang. Automatic polycube-maps. In *Geometric modeling and processing*, pages 3–16, Hangzhou, China, 2008.

[49] C. Loop and s. Schaefer. Approximating catmull-clark subdivision surfaces with bicubic patches. *ACM Transactions on Graphics*, 27(1):1–11, 2008.

[50] C. Loop, s. Schaefer, T. Ni, and I. Castano. Approximating subdivision surfaces with gregory patches for hardware tessellation. *ACM Transactions on Graphics*, 28(5):1–9, 2009.

[51] J. Maillot, H. Yahia, and A. Verroust. Interactive texture mapping. In *ACM SIGGRAPH*, pages 27–34, New York, NY, USA, 1993.

[52] T. Martin, E. Cohen, and M. Kirby. Volumetric parameterization and trivariate b-spline fitting using harmonic functions. In *Proceeding of Symposium on Solid and Physical Modeling*, Stony Brook, New York, USA, 2008.

[53] MOSEK. http://www.mosek.com/.

[54] P. Mullen, Y. Tong, P. Alliez, and M. Desbrun. Spectral conformal parameterization. In *Proceedings of the Symposium on Geometry Processing*, pages 1487–1494, Aire-la-Ville, Switzerland, Switzerland, 2008.

[55] H. M. Brice no, P. V. Sander, L. McMillan, S. Gortler, and H. Hoppe. Geometry videos: a new representation for 3d animations. In *Symposium on Computer Animation*, pages 136–146, Aire-la-Ville, Switzerland, Switzerland, 2003.

[56] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level partition of unity implicits. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 463–470, New York, NY, USA, 2003.

[57] G. Oliverio. *Maya 8: Character Modeling*. 2006.

[58] S. Orgaz. Xnormal. http://www.xnormal.net/, 2010.

[59] Pixologic. Zbrush. http://www.pixologic.com/, 2010.

[60] E. Praun, A. Finkelstein, and H. Hoppe. Lapped textures. In *ACM SIGGRAPH*, pages 465–470, New Orleans, Louisiana, USA, 2000.

[61] E. Praun and H. Hoppe. Spherical parametrization and remeshing. *ACM Transactions on Graphics*, 22(3):340–349, 2003.

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

[63] S. Saba, I. Yavneh, C. Gotsman, and A. Sheffer. Practical spherical embedding of manifold triangle meshes. In *Proceedings of the International Conference on Shape Modeling and Applications*, pages 258–267, Washington, DC, USA, 2005.

[64] P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe. Texture mapping progressive meshes. In *ACM SIGGRAPH*, pages 409–416, New York, NY, USA, 2001.

[65] R. Schoen and S.T. Yau. *Lectures on Harmonic Maps*. 1997.

[66] A. Shapiro and A. Tal. Polyhedron realization for shape transformation. *The Visual Computer*, 14(8-9):429–444, 1998.

[67] A. Sheffer, B. Lévy, M. Mogilnitsky, and A. Bogomyakov. ABF++: fast and robust angle based flattening. *ACM Transactions on Graphics*, 24(2):311–330, 2005.

[68] A. Sheffer, E. Praun, and K. Rose. Mesh parameterization methods and their applications. *Foundations and Trends in Computer Graphics and Vision*, 2(2), 2006.

[69] A. Sheffer and E. Sturler. Surface parameterization for meshing by triangulation flattening. In *Proceedings of 9th International Meshing Roundtable*, pages 161–172, New Orleans, Louisiana, USA, 2000.

[70] H. Si. A quality tetrahedral mesh generator and a 3d delaunay triangulator. http://tetgen.berlios.de/, 2009.

[71] M. S. Smit and W. F. Bronsvoort. Variational tetrahedral meshing of mechanical models for finite element analysis. In *Computer-Aided and Applications,5(1-4)*, volume 5.

[72] O. Sorkine, D. Cohen-Or, R. Goldenthal, and D. Lischinski. Bounded-distortion piecewise mesh parameterization. In *Proceedings of the conference on Visualization*, pages 355–362, Washington, DC, USA, 2002.

[73] K. Takayama, M. Okabe, T. Ijiri, and T. Igarashi. Lapped solid textures: filling a model with anisotropic textures. *ACM Transactions on Graphics*, 27(3):1–9, 2008.

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

[75] L. M. Taylor and D. P. Flanagan. Pronto 3d a three-dimensional transient solid dynamics program. Sandia National Laboratories, Albuquerque, NM, 1989.

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

[77] G. Turk. Generating textures on arbitrary surfaces using reaction-diffusion. In *ACM SIGGRAPH*, pages 289–298, New York, NY, USA, 1991.

[78] H. Wang, Y. He, X. Li, X. Gu, and H. Qin. Polycube splines. In *SPM '07: Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 241–251, New York, NY, USA, 2007.

[79] H. Wang, M. Jin, Y. He, X. Gu, and H. Qin. User-controllable polycube map for manifold spline construction. In *Symposium on Solid and Physical Modeling*, pages 397–404, Stony Brook, New York, USA, 2008.

[80] Y. Wang, X. Gu, P. M. Thompson, and S.-T. Yau. 3d harmonic mapping and tetrahedral meshing of brain imaging data. In *Proceeding of Medical Imaging Computing and Computer Assisted Intervention*, St. Malo, France, 2004.

[81] Y. Wang, M. Gupta, S. Zhang, S. Wang, X. Gu, D. Samaras, and P. Huang. High resolution tracking of non-rigid motion of densely sampled 3d data using harmonic maps. *International Journal of Computer Vision*, 76(3):283–300, 2008.

[82] H. J. Weber and G. B. Arfken. *Mathematical Methods For Physicists*. Academic Press, 2005.

[83] O. Weber and C. Gotsman. Controllable conformal maps for shape deformation and interpolation. *ACM Transactons on Graphics*, 29:78:1–78:11, 2010.

[84] A. P. Witkin and M. Kass. Reaction-diffusion textures. In *ACM SIGGRAPH*, pages 299–308, New York, NY, USA, 1991.

[85] S.-Q. Xin and G.-J. Wang. Improving chen and han's algorithm on the discrete geodesic problem. *ACM Transactions on Graphics*, 28(4):104:1–104:8, 2009.

[86] R. Zayer, C. Rössl, and H. Seidel. Convex boundary angle based flattening. In *Proceedings of Vision, Modeling, and Visualization 2003,*, pages 281–288, München, Germany, 2003.

[87] D. Zhang and M. Hebert. Harmonic maps and their applications in surface matching. In *CVPR*, pages 2524–2530, 1999.

[88] S. Zhang and P. Huang. High-resolution, real-time 3d shape acquisition. In *Computer Vision and Pattern Recognition Workshop*, pages 28–28, Washington, DC, USA, 2004.

[89] K. Zhou, J. Huang, J. Snyder, X. Liu, H. Bao, B. Guo, and H. Y. Shum. Large mesh deformation using the volumetric graph laplacian. *ACM Transactions on Graphics*, 24(3):496–503, 2005.

Publications

(1) **Jiazhi Xia**, Ying He, Dao Quynh, Xiaoming Chen, and Steven Chu-Hong Hoi. Modeling and compressing 3D facial expressions using geometry videos. *IEEE Transactions on Circuits and Systems for Video Technology*, accepted. 2011.

(2) **Jiazhi Xia**, Ismael Garcia, Ying He, Xin Shi-Qing and Gustavo Patow. Editable Polycube Map for GPU-based subdivision surfaces. *I3D'11 : Proceedings of the 2011 symposium on Interactive 3D graphics and games*, pp. 151-158, 2011.

(3) **Jiazhi Xia**, Ying He, Dao Quynh, Xiaoming Chen, and Steven Chu-Hong Hoi. Modeling 3D Motion Data Using Geometry Video. In *ACM Multimedia*, pp. 591-600, 2010.

(4) **Jiazhi Xia**,Ying He,Shuchu Han, Chi-Wing Fu, Feng Luo, and Xianfeng Gu. Parameterization of Star Shaped Volumes Using Green's Functions In *Proceedings of Geometric Modeling and Processing*, pp. 219-235, 2010.

(5) **Jiazhi Xia**, Ying He, Xiaotian Yin, Shuchu Han, and Xianfeng Gu. Direct-Product Volumetric Parameterization of Handlebodies via Harmonic Fields. In *Proceedings of IEEE International Conference on Shape Modeling*, pp. 3-12, 2010.

(6) Shuchu Han, **Jiazhi Xia**, and Ying He. Hexahedral shell mesh construction via volumetric polycube map. In *Proceedings of ACM Symposium on Solid and Physical Modeling*, pp. 127-136, 2010.

(7) Chi-Wing Fu, **Jiazhi Xia**, and Ying He. LayerPaint: a multi-layer interactive 3D painting interface. In *Proceedings of the 28th international Conference on Human Factors in Computing Systems*, pp. 811-820, 2010.

(8) Long Zhang, Ying He, **Jiazhi Xia**, Xuexiang Xie, and Wei Chen. Real-Time Shape Illustration Using Laplacian Lines. *IEEE Transactions on Visualization and Computer Graphics*, Vol 17, No. 7, pp. 993-1006, 2011.

(9) Wei Zeng, Ying He, **Jiazhi Xia**, Xianfeng Gu, and Hong Qin. C^{∞} smooth freeform surfaces over hyperbolic domains In *Proceedings of SIAM/ACM Joint Conference on Geometric and Physical Modeling*, pp. 367-372, 2009.

(10) Shuchu Han, **Jiazhi Xia**, and Ying He. Hexahedral shell mesh construction via volumetric polycube map. Invited to *Computer-Aided Design*, Under review.

(11) Hock Soon Tan, **Jiazhi Xia**, Ying He, and Yun-Qing Guan. A System for Capturing, Rendering and Multiplexing Images on Multi-view Autostereoscopic Display. In *International Conference on CYBERWORLDS*, pp. 325-330, 2010