Multiresolution Solid Objects over Simplicial Complexes

A Dissertation Presented

by

Yu-Sung Chang

to

The Graduate School

in Partial fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

Stony Brook University

August 2005

Abstract of the Dissertation

Multiresolution Solid Objects over Simplicial Complexes

by

Yu-Sung Chang

Doctor of Philosophy
in
Computer Science

Stony Brook University

2005

Even though there has been significant growth in solid and volumetric representations, rapid advances in manufacturing technology, imaging technology, and material science in recent years pose new challenges which have not been fully resolved by the current representation techniques.

In this dissertation, we address the challenges by proposing a novel framework for solid model representation based on subdivision process. We call our approach Multiresolution Solid Objects, or MSO. Within the MSO framework, we represent volumetric objects by a series of new solid subdivision schemes. Subdivision solids share many benefits of subdivision surfaces, yet they have the complexity and the mathematical challenges that are unique to the high dimensional cases.

This dissertation follows our development of MSO framework. We begin with the discussion of general polyhedral mesh structures in 3D space and develop a new structured

mesh in 3D based on simplicial complexes, called Octet-truss. The dissertation continues with the derivation of new solid subdivision schemes over the regular meshes, and the generalization to arbitrary tetrahedral and hexahedral meshes. We present (i) a box-spline based approximate solid scheme; (ii) an interpolatory solid scheme over arbitrary complex meshes; (iii) an interpolatory solid scheme over arbitrary hexahedral meshes; and finally (iv) a unified subdivision schemes for multidimensional objects based on box-splines. Each scheme is derived from volumetric splines, a weighted perturbation of linear interpolation, and the Lagrange interpolating polynomials. The derivation is followed by the proving of the convergence and smoothness of the schemes using well-established mathematical techniques. In rare cases, we provide the empirical data which highly suggest the convergence and the smoothness of the schemes.

In addition to the theoretical contributions, this dissertation presents numerous practical implementations to emphasize the benefits of our framework. These include arbitrary shape design, heterogeneous material modeling, free-form design, non-manifold object representation, boundary and feature representation and free-form deformation. Finally, we describe few new applications including high quality partitioning and fitting, and feature-preserving volume filtering, which lead toward a wide-range of applications in the future.

For my parents, my daughter, Erin and my wife, Hyunjoo

Contents

List of Tables

List of Figures

List of Algorithms

Acknowledgments

I would like to express my sincere gratitude to my thesis advisor, Professor Hong Qin during my studies. I can not imagine myself completing this dissertation without his inspiration, encouragement and guidance. He showed such kindness and patience that no graduate students expect more from their advisors. I am thankful to Professors Arie Kaufman, Michael Ashikhmin, Klaus Mueller, Dimitris Samaras and Manuel Oliveira for their advice, as well as for serving on various committees. I also thank Doctor Ioana Boier-Martin for taking the time to serve as the external member of my dissertation committee.

I wish to thank Kevin T. McDonnell for the contents of Appendix C. We spent numerous hours together in discussing fascinating topics which ultimately led to this dissertation. I am especially grateful for his friendship and integrity. It has also been a great pleasure knowing many of the Visualization Lab members, especially Jing Hua, Chris Carner, Ying He, Haixia Du, Ye Duan, Hui Xie, Rob Wlodarczyk, Xiaohu Guo, Kexiang Wang and Sumantro Ray.

My research was supported in part by the following grants awarded to Professor Qin: NSF CAREER award CCR-9896123, the NSF grants IIS-0082035, IIS-0097646, IIS-0326388, and ACR-328930, and an Alfred P. Sloan Fellowship, and a Honda Initiation Award.

Last, but certainly not least, I want to thank my parents, wife and daughter for their endless love and support. Particularly, I do not know how to express my gratitude to my wife, Hyunjoo, who have sacrificed so much for my sake.

This dissertation is dedicated to them.

Publications

Yu-Sung Chang, Kevin T. McDonnell, and Hong Qin. A new solid subdivision scheme based on box splines. In *Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications*, pages 226–233, June 2002.

Yu-Sung Chang, and Hong Qin, Spline-based Solid Subdivision Schemes over Arbitrary Tetrahedral Meshes, Submitted for journal review, 2005.

Yu-Sung Chang, Kevin T. McDonnell, and Hong Qin. An interpolatory subdivision for volumetric models over simplicial complexes. In *Proceedings of Shape Modeling International 2003*, pages 143–152, May 2003.

Yu-Sung Chang and Hong Qin, Multi-dimensional Non-manifold Subdivision over Simplicial Complexes, Submitted for journal review, 2005.

Yu-Sung Chang and Hong Qin, A framework for multi-dimensional adaptive subdivision objects. In *Proceedings of Solid Modeling 2004*, pages 123–134, 2004.

Kevin T. McDonnell, Yu-Sung Chang, and Hong Qin, Interpolatory, Solid Subdivision of Unstructured Hexahedral Meshes, *The Visual Computer*, 20(6):418–436, August 2004.

Kevin T. McDonnell, Yu-Sung Chang, and Hong Qin. DigitalSculpture: A subdivision-based approach to interactive implicit surface modeling. *Graphical Models*, 67(4):347–369, July 2005.

Yu-Sung Chang and Hong Qin. High Quality Surface Partitioning and Fitting for Subdivision, Submitted for journal review, 2005.

Yu-Sung Chang and Hong Qin. Multiresolution Volumetric Filtering based on Box Splines, In preparation, 2005.

Notation

Through out this proposal, we use multi-index notation which is common in multivariate analysis. We use a bold lowercase letter to indicate an n-tuple of variables, or vectors:

$$\mathbf{z} = (z_1, z_2, ..., z_n) \in \mathbb{R}^n.$$

Likewise, bold *i, j, k,* and *l* are used in place for integer n-tuple indexes:

$$\mathbf{i} = (i_1, i_2, ..., i_n) \in \mathbb{Z}^n.$$

A bold uppercase bold letter represents a matrix. Possible exceptions are the directional matrices for box splines, which are written in capital Greek letters, such as Ξ.

A power $\mathbf{z}^\mathbf{i}$ is defined as

$$\mathbf{z}^\mathbf{i} = {z_1}^{i_1} {z_2}^{i_2} \cdots {z_n}^{i_n},$$

and a summation is also defined as

$$\sum_{\mathbf{i}} = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_n},$$

through out the proposal. By the norm $|\mathbf{i}|$ of an index $\mathbf{i}$, we mean,

$$|\mathbf{i}| = \sum_{k=1}^{n} i_k.$$

Unlike the index norm, the norm of arbitrary n-tuple $\mathbf{z}$ is defined when it is in need. A modulus operation is also defined in multi-index manner. For instance,

$$\mathbf{i} = (i_1 (\mathrm{mod}\ p), i_2 (\mathrm{mod}\ p), ..., i_n (\mathrm{mod}\ p)).$$

All other binary operators follow the conventional multi-index notation. We show some of common examples.

$$\begin{aligned} \mathbf{y} &= (y_1, y_2, ..., y_n) \in \mathbb{R}^n \\ \mathbf{z} &= (z_1, z_2, ..., z_n) \in \mathbb{R}^n \\ \mathbf{z} + \mathbf{y} &= (z_1 + y_1, z_2 + y_2, ..., z_n + y_n) \\ c\mathbf{z} &= (cz_1, cz_2, ..., cz_n) \end{aligned}$$

For simplicity, we use a bracket notation to indicate openness of an interval. For instance, the expression,

$$x \in [a, b), \quad y \in (c, d]$$

means $a \leq x < b$, $c < y \leq d$, respectively. A special symbol $\square$ denotes a half-open unit n-cube, $[0, 1)^n$.

Chapter 1

Introduction

Solid modeling in computer graphics involves the techniques regarding the computer representation, design, and manipulation of 3D objects or volume data. In the beginning of the geometric modeling, the representation of 3D objects are limited to their surfaces, or boundary representations. During the period, it was considered adequate to represent 3D objects with planes, simple primitives such as cubes, spheres, and cylinders, parametric surfaces, or patches of these 2D objects. This trend still plays a major role in the geometric modeling even today. For example, Boeing's design of the commercial 777 aircraft, which is considered to be the most advanced computer-aided design up to date, had been done by CATIA [26], a commercial system which is developed by Dassualt in France, and is marketed by IBM in the US. In reality, the most parts of CATIA program is based on aged technologies, such as surface representations for modeling and some early stage solid modeling techniques for engineering analysis.

Currently, the volume representations in solid modeling can be categorized as followings: Implicit function representations such as Constructive Solid Geometry (CSG), boundary representations (B-reps), parametric representations, polyhedral mesh representations and cell decomposition (voxels). Each representation has its own benefits and limitations.

For instance, CSG is well suitable for representing mechanical objects which usually consist of simple primitives such as spheres and cylinders, and their set operations. However, it does not represent the boundaries of the objects explicitly, and it should be extracted using other methods. Boundary representations are vastly used in manufacturing industries, based on standardized surfaces such as NURBS (non-uniform rational B-splines), yet they do not represent the inside of 3D objects in reality and as such any complex set operations are limited and are prone to approximate errors. Parametric representations of volume objects have an ability to represent continuous 3D fields, but they are limited by the topology of the parametric space. Polyhedral meshes and cell decomposition are discrete representations of solid objects. As a result, analytic information such as derivatives and gradients can be only acquired by approximation. In spite of the disadvantages, these representations have been served as underlying solid modeling techniques for the past decades.

However, the recent advances in technologies makes it necessary to develop new class of solid representation. One pivotal example is stereolithography, or 3D laying technology used in rapid prototyping process (Figure 1.2). This technology involves the thin-layering of 3D models, the injection process of special liquid photopolymer, the hardening process using the stereolithograph apparatus (SLA). The technology can produce a real object of an arbitrary solid models using the liquid plastic. Since the process requires the continuous layering of the inside of 3D models, the ideal computer models for this process should be able to represent continuous varying material properties inside of the models. Functionally graded materials (FGM) [43, 65] are another examples of the materials with continuously varying properties (Figure 1.3). These type of materials, often referred as heterogeneous, or anisotropic materials are emerging rapidly in many practical fields. On the contrary, the underlying assumption of the current solid representations, especially CSG and B-reps, is that the material is homogeneous. This fact makes most of the current solid representations very inadequate for representing the new materials. Another challenge associated with new

technologies are the growing size of the volumetric data. When the first magnetic resonance imaging (MRI) system was introduced, the resolutions were very limited. However, the cutting edge 3 Tesla MRI machines [42] can achieve the resolution as high as 1 mm^3 per voxel. As a consequence, they can generate very large volumetric data in short period of time. In addition, the data from the other sources, such as the seismic data from geo-scientific survey, various medical images, and the point clouds from 3D scanning machines increase the amounts and the complexity of the volume data that should be dealt by the solid modeling techniques. Moreover, new paradigms in shape design, such as the free-form design [62], the interactive design, the physics-based modeling, and the integration of shape modeling and engineering analysis [103] introduce new challenges to the existing solid representations.

To resolve the current challenges in solid modeling, we propose Multiresolution Solid Objects, or MSO, a new solid representation framework based on the subdivision method. We base our representation upon simplicial complex meshes in 3D, which are numerically stable and robust. We continue our dissertation by deriving a series of novel subdivision schemes over the simplicial complex and other arbitrary meshes. The new schemes are based on sound mathematical theories, such as trivariate spline solids and the generating function method. We employ existing analysis techniques to examine and prove the convergence of the schemes and the smoothness of the subdivision solid objects in the limit. We extend our schemes to represent a variety of objects, such as objects with arbitrary topology, multi-dimensional objects, non-manifold objects, and objects with sharp features. Finally, we exploit the advantage of our approach by applying the schemes to practical applications, such as heterogeneous material modeling and non-manifold object modeling.

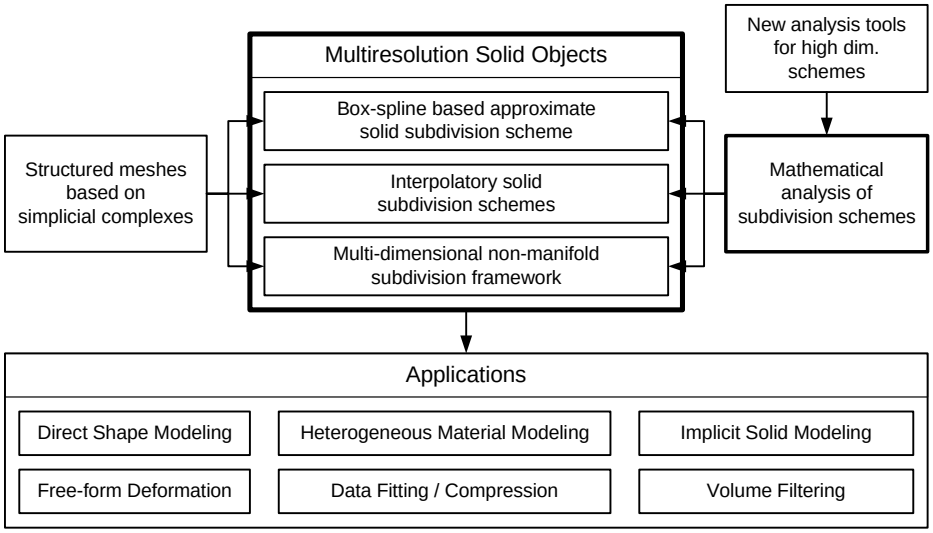


Figure 1.1: Conceptual hierarchy of the dissertation research.

1.1 Problem Statement

The recent emergence of new technologies poses new challenges to the current solid modeling techniques, especially the volume representations. The challenges such as new heterogeneous materials, the increasing complexity of volume data and solid models, the demand of interactive design, and the integration of designing and analysis processes have not been fully resolved by the state-of-the-art solid modeling techniques. We content that Multiresolution Solid Objects, or MSO, is a possible answer to the new problems. Based on the subdivision method, the MSO shares the advantages of subdivision surfaces in 2D modeling, yet it has the benefits unique to solid modeling. The first step to the approach is the realization of the following advantages and potentials of the MSO framework:

- It can take advantage of the benefits of other solid representations. It can be considered as parametric representations, yet its behavior resembles those of cell decomposition or polyhedral mesh representations.

- Its base meshes comprise simplices, which are proven to be the most simple and stable structures and are favored in many numerical simulations, such as finite element

methods (FEM).

- It provides hierarchical structures inherently. Moreover, it can provide level-of-detail control through local subdivision.
- It can represent arbitrary 3D geometry, heterogeneous material properties and boundary surfaces in a single framework.
- It can be generalized to represent multidimensional objects. Non-manifold objects can be represented as a result of the generalization.
- It can be easily extended to provide the sharp feature representation within objects.
- It employs linear stationary procedures during the computation. It only involves simple linear combinations in each step. It leads to straightforward implementation, numerical stability and easy prediction of computational costs.
- It can utilize many existing mathematical tools to analyze subdivision surfaces.

In spite of the advantages, the MSO is not a trivial extension of current subdivision surface schemes. Much work has to be done before real-world applications are able to take advantage of the full potential of the MSO. Most of the problems are related to the complexity of 3D space, and it is unique to the subdivision solid schemes. We address the following problems in this dissertation:

- The choice of simplicial complex meshes is proven to be both beneficial and detrimental. It has the advantages of simplicity and stability, yet it lacks the properties that are desired for the stationary subdivision schemes.
- Unlike the tensor-product approaches, we cannot deduce our subdivision rules from curve or surface subdivision schemes. Instead, we have to rely on naive trivariate

splines and their properties to derive subdivision rules. The subdivision masks and rules are far more complex than those of the subdivision surfaces as a result.

- The generalization of the schemes to arbitrary 3D meshes introduces very complex situations. Some extraordinary cases that never occur in 2D meshes are introduced.

- The mathematical analysis tools that are used for the analysis of surface subdivision schemes cannot be directly applied. Especially, lack of symmetry in 3D meshes makes the symbolic matrix computations and the derivation of general solutions impossible.

- Non-manifold representations cover a wider range of cases than the cases that are only involving surfaces. The same argument goes to feature representations. Most of cases cannot be solved by any subdivision scheme of lower dimension.

1.2 Contributions

In this dissertation, we propose the MSO framework as a solution of the challenges to current solid representations. Specifically, we have proposed several subdivision schemes and their applications to demonstrate the potential of the MSO framework. The details of the contributions of this dissertation are as follows:

- A new structured mesh based on simplicial complexes for solid subdivision schemes is proposed. We prove that the proposed mesh, called octet-truss, satisfies the properties that are desired for the subdivision method. This provides the foundation of the derivation of subdivision solid schemes over simplicial complexes.

- We propose a new approximate solid subdivision scheme based on box splines. Based on a particular class of trivariate box splines, it achieves high smoothness

with lower degree polynomials than those of tensor-product subdivision schemes. Underlying octet-truss structure is generalized into arbitrary tetrahedral meshes in 3D. It offers shape modeling of arbitrary topology as well as heterogeneous material objects.

- We develop a new interpolatory solid subdivision scheme over simplicial complexes. The derivation of the rules are based on a weighted perturbation of linear interpolation. The scheme provides powerful interpolatory modeling tool and non-manifold representation without much modification. The smoothness of the regular cases is proven by the generating function method.

- We propose a new interpolatory solid subdivisions scheme over unstructured hexahedral meshes. The scheme is founded upon tri-cubic Lagrange interpolating polynomials. We also illustrate the proof of the smoothness in the regular cases.

- We unify the existing subdivision schemes for curves and surfaces based on the same class of box-splines with the developed approximate solid subdivision scheme and present new framework for representation of multi-dimensional objects. Non-manifold cases are categorized and several solutions are suggested. The scheme is extended to incorporate with boundary and sharp feature representations.

- We present the current mathematical tools for the convergence and smoothness analysis of subdivision schemes and their application to the solid subdivision schemes. We discuss the new challenges associated with the solid schemes and apply the analysis techniques to certain extend. We present the empirical data from the analysis which highly suggest the desirable properties of our schemes.

- We utilize the proposed schemes to demonstrate the potential of the MSO framework. We implement a wide range of applications using the schemes, including arbitrary

shape design, heterogeneous material modeling, free-form design, non-manifold, boundary and feature representations, implicit solid modeling, and free-form deformation. In addition, we implement high quality surface partitioning and fitting for subdivision employing our MSO framework and a volume filtering algorithm based on anisotropic trivariate box splines.

1.3 Dissertation Organization

This dissertation is organized in the following fashion. In Chapter 2, we shall begin with a brief review on contemporary solid modeling techniques. It is followed by the detailed review on parametric representations and the subdivision method. In Chapter 3, we discuss the meshes for the subdivision method in detail. We first present the 2D regular meshes employed in various subdivision surface schemes. We extend the idea to the 3D structured meshes, discuss the desired properties, and suggest a new structured mesh based on simplicial complexes. In Chapter 4, we derive, generalize and analyze a novel solid subdivision scheme based on box splines. We present the complete derivation of the rules and the detailed analysis. In Chapter 5, we propose a new interpolatory solid subdivision scheme over simplicial complexes. We derive the rules using the generating functions and provide the proof of its smoothness in brief. In Chapter 6, we extend our the box-spline based scheme to handle special cases, and unify it with the lower dimensional box-spline based subdivision schemes. In Chapter 7, our schemes are implemented in various applications to demonstrate the potential of the MSO representation. Finally, the dissertation is concluded in Chapter 8 with the summary and the discussion on future research directions. A collaborative research result with Kevin T. McDonnell is presented as an appendix in Appendix C.

Figure 1.2: 3D Systems stereolithography machine (left), its photopolymer platform (right top), and the end of a print run process (right bottom). (Images courtesy of PT CAM, Inc.)

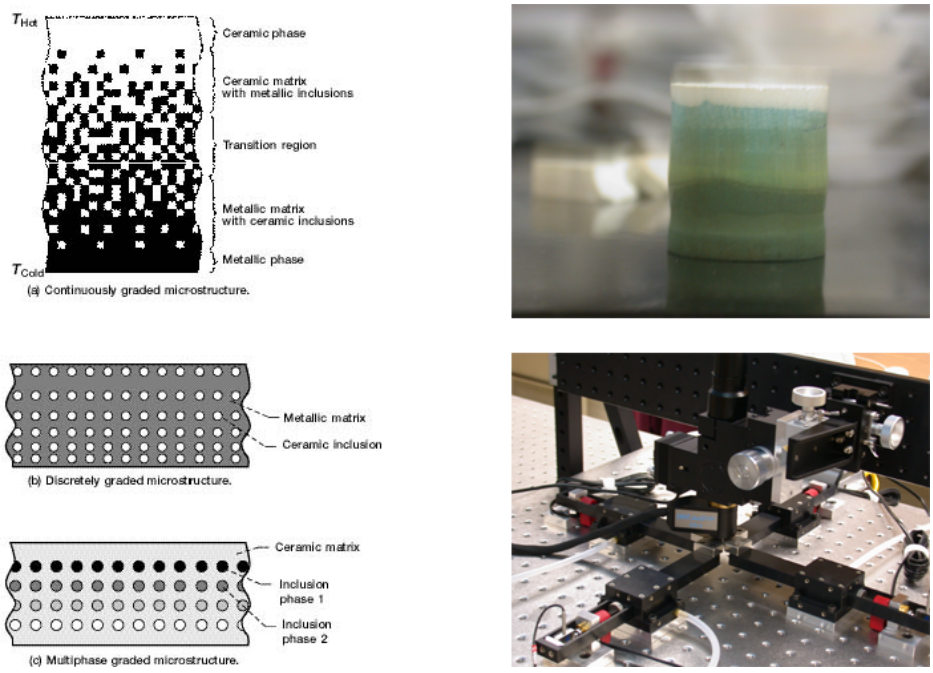


Figure 1.3: Functionally graded microstructures (left), a fabricated FGM object (right top), and a FGM fabrication machine (right bottom). (Images courtesy of NASA and FGM Development Lab. at University of Maryland.)

Chapter 2

Background Review

The most relevant topic in computer graphics to our Multiresolution Solid Objects approach is the subdivision method. In this chapter, we start with a brief overview of the current solid representation techniques and their history. It is followed by the detailed survey on the subdivision method. Finally, analysis techniques for subdivision schemes are also reviewed.

2.1 Volume Representations in Solid Modeling

Since Requicha and Voelcker [85]'s the famous survey paper in 1982, the past two decades have witnessed significant growth in solid modeling, especially in the development of volume representations. The volume representations in solid modeling can be categorized as followings: Implicit function representations such as Constructive Solid Geometry (CSG), boundary representations (B-reps), parametric representations, and cell decomposition (voxels). B-reps are not volume representations of true sense, therefore it is omitted in the review.

2.1.1 Implicit Function Representations

Constructive Solid Geometry (CSG) [2, 71] and blobby models [9, 101] are two most important examples of implicit function representations. They represent a solid as the solution set of an implicit function,

$$w = f(x, y, z).$$

In general, a level set

$$w = f(x, y, z),\ w = w_0$$

represents the boundary of the solid, and portions where $w < w_0$ comprise the interior. In implicit function representations, it is easy to differentiate the interior from the exterior and the boundary. Hence, they are well incorporated with algebraic operations on models [70]. Also, it is free from the complex topology, since there is no explicit form of the boundary. However, there are few disadvantages. For instance, there is no simple way to evaluate them in general case [2] for rendering purpose. In addition, directly manipulating the level-set geometry is very challenging because the solid boundary is implicitly defined.

2.1.2 Parametric Representations

In the beginning of the computer graphics, parametric representations have been one of the most important techniques to represent geometric objects. In 1984, Boehm *et al.*[12] surveyed parametric curves and surfaces which had been widely used especially in computer-aided design and manufacturing. Basically, we express an object as a mapping from a domain to three-dimensional Euclidean space.

$$\mathbf{y} = f(\mathbf{x}),\ \mathbf{y} \in \mathbb{R}^3 \tag{2.1}$$

where $\mathbf{x} \in \mathcal{D} \subseteq \mathbb{R}^k$, for some domain $\mathcal{D}$. Obviously, k depends on the Euclidean dimension of the object that we want to represent. For instance, $k = 1$ for a curve, $k = 2$ for a surface, and $k = 3$ for a solid object. Bernstein-Bézier solids [54], B-spline solids, and other tensor product based [44] approaches are examples of parametric solids.

If f is a function that *globally* satisfies the required properties, *i.e.*certain continuity, it becomes hard for a user to control an object. In other word, if f is a cubic polynomial curve that is written as

$$f(t) = c_3 t^3 + c_2 t^2 + c_1 t + c_0, \tag{2.2}$$

It would be impossible to modify the curve intuitively, by controlling the coefficients c_i's. Therefore, we consider a parametric object f that is an affine combination of piecewise polynomials with local support:

$$f(\mathbf{x}) = \sum_i \mathbf{p}_i \, N_i(\mathbf{x}), \; \mathbf{x} \in \mathcal{D} \subseteq \mathbb{R}^k, \tag{2.3}$$

where the N_i's are called *basis functions*. N_i's satisfy the following properties: (1) piecewise polynomial; (2) local support; (3) non-negative definite; (4) partition of unity. Tensor product based solids in particular have basis functions of the form

$$N_{\mathbf{I}}(t, s, r) = N_{i_0}(t) \otimes N_{i_1}(s) \otimes N_{i_2}(r),$$

where $\mathbf{I} = (i_0, i_1, i_2)$. Since each N_i is defined only on finite support, the modification of $\mathbf{p}_i$ only effects the local region of the object. In general, the basis functions can be evaluated efficiently and robustly.

2.1.3 Cell Decomposition

To date, cell decomposition techniques [41, 93] have been widely used in the volume visualization field. These techniques discretize the entire three-dimensional space into a rectangular grid. An object is represented as a collection of the volumetric elements, or *voxels*. Not only the spatial occupation, voxels can contain other material properties, such as density, temperature, color, etc. It could be understood an analogue to pixel-based images in two-dimensional space. Oftentimes, volume visualization techniques are used to render voxelized objects. One of the disadvantages of cell decomposition methods is the lack of high-level geometric interpretation. However, many researchers have presented techniques that address this issue by incorporating voxel spaces with implicit functions for practical applications [39].

2.2 Subdivision Method

Since the beginning of ancient Greek geometry, *recursion* process has been an essential tool to design complex geometric objects. For instance, Euclid described the recursive construction of a *logarithmic spiral* by using the Golden Rectangle and successively dividing it (See Figure 2.1). Nonetheless, the process is simple, yet it results in the elegant curve that has captured many artistic minds during the centuries.

Subdivision modeling is a modern analogue to the recursive construction in computer era. It is tightly related to the choice of a domain mesh and the evaluation of parametric representations. The subdivision method can be understood as the procedural and recursive process to evaluate geometric objects in parametric representations, especially in the form of Equation 2.3. In the next few sections, we shall review the current subdivision modeling techniques and discuss their properties.

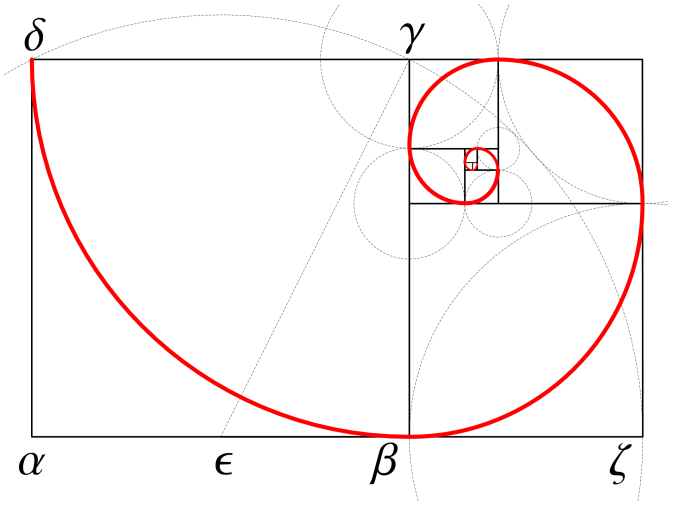
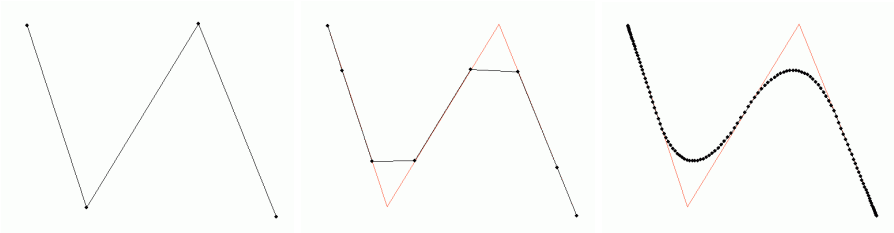


Figure 2.1: A logarithmic curve from the Golden Rectangle.

2.2.1 Stationary Subdivision Curves



(a) (b) (c)

Figure 2.2: An example of a curve generated by the Chaikin's algorithm. (a) The initial control points. (b) The first level of the subdivision. (c) In the limit.

It is considered that "Chaikin's algorithm" [15] to generate a spline curve is the first case of subdivision schemes in computer graphics. In its simplest form, the subdivision process can be expressed as

$$\begin{cases} \mathbf{p}_{2i}^{\ell+1} &= \frac{3}{4}\mathbf{p}_i^{\ell} + \frac{1}{4}\mathbf{p}_{i+1}^{\ell} \\ \\ \mathbf{p}_{2i+1}^{\ell+1} &= \frac{1}{4}\mathbf{p}_i^{\ell} + \frac{3}{4}\mathbf{p}_{i+1}^{\ell}, \end{cases}$$

where ℓ is subdivision level and i is an index for control points. Since we cut the corners

of each line segments to acquire the finer approximation, it is often referred as "Chopping corner" method. A curve generated by this formula forms a C^1 spline curve in the limit. Note that the curve does not interpolate the original control points, which makes it an *approximating* scheme.

Another example of an approximating subdivision curve is "Cubic spline" scheme. The process can be formulated as

$$\begin{cases} \mathbf{p}_{2i}^{\ell+1} &= \frac{1}{2}\mathbf{p}_i^{\ell} + \frac{1}{2}\mathbf{p}_{i+1}^{\ell} \\ \\ \mathbf{p}_{2i+1}^{\ell+1} &= \frac{1}{8}\mathbf{p}_i^{\ell} + \frac{3}{4}\mathbf{p}_{i+1}^{\ell} + \frac{1}{8}\mathbf{p}_{i+2}^{\ell}. \end{cases}$$

The limit curve of the scheme results in C^2, which is higher continuity than Chaikin's. However, it requires slightly larger neighbors to calculate new points. Nonetheless, these two schemes share similarity in derivation since they are both originated from spline curves. In the limit, the both schemes converge to curves of the form:

$$f(t) = \sum_i \mathbf{p}_i B_i(t), \tag{2.4}$$

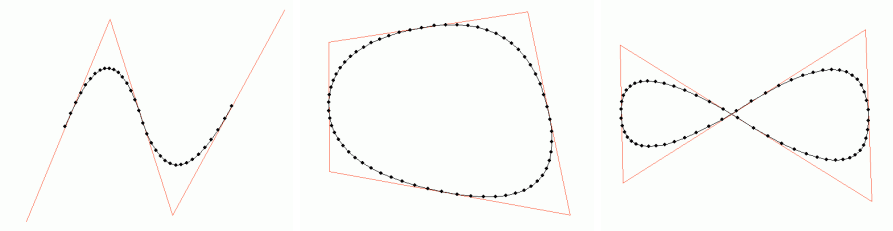
where B_i's are basis functions for splines and $\mathbf{p}_i$'s are initial control points.

Unique examples of subdivision curves are "4-point subdivision curve" scheme by Dyn *et al.*and its variations [35, 38, 32, 37, 53, 34]. Their uniqueness is due to the interpolating aspect, *i.e.*the limit curves interpolate original control points. Those schemes are called *interpolatory* schemes. The simplest form of the algorithms can be written as

$$\begin{cases} \mathbf{p}_{2i}^{\ell+1} &= \mathbf{p}_i^{\ell} \\ \\ \mathbf{p}_{2i+1}^{\ell+1} &= (\frac{1}{2} + w)(\mathbf{p}_i^{\ell} + \mathbf{p}_{i+1}^{\ell}) - w(\mathbf{p}_{i-1}^{\ell} + \mathbf{p}_{i+2}^{\ell}), \end{cases}$$

where $w \in \mathbb{R}$ controls a tension of the curve which is generated by the scheme. In fact, the choice of w determines the convergence of the process. If $w = 0$, it is clear that the process is simply linear interpolation. In the range of $w \in (0, \frac{1}{8})$, the limit curve which is produced by the scheme belongs to C^1 class. Otherwise, the whole process diverges. Moreover, the higher continuity and convexity preserving can be achieved if we choose w more restrictively [53, 34]. Because they are interpolatory, control points are geometrically invariant, which makes them useful in data fitting and constraint enforcement.

Since a curve is an image of one-dimensional Euclidean space, its domain has limited variety of topology [66, 98]. We say two spaces A and B are topologically equivalent if there exists an injective, subjective and continuous mapping f from A to B.



(a) (b) (c)

Figure 2.3: An example of the different topologies in 1-D. The curves are the Chaikin's. (a) An open interval (b) The S^1. (c) The modification to the S^1.

The trivial case is that the domain is topologically equivalent to closed unit interval $[0, 1]$. Another case is when the domain is equivalent to S^1, the fundamental group of the circle in $\mathbb{R}^2$, *i.e.*$S^1 = \{e^{2\pi i\theta} \mid \theta \in [0, 1)\}$. The first case requires special rules for both ends of the interval, which could be chosen to be interpolate the end points or to be open. The aforementioned algorithms have no problem with the latter case. Other topology cases can be covered by the two cases. However, in the case of Figure 2.3(c), we cannot find the tangent at the intersecting point.

2.2.2 Stationary Subdivision Surfaces

Subdivision surfaces have been well researched in the past two decades. It is because unlike subdivision curves that show little advantages over parametric curves, subdivision surfaces can actually handle topologically challenging cases in which parametric representation fails or becomes unpractically complex.

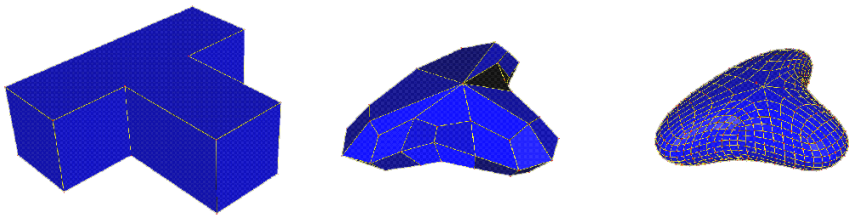
A trivial way to extend subdivision curves into surfaces is using the tensor product. In particular, if B_i's are basis functions for spline curves, we can define spline surfaces as

$$f(t,s) = \sum_i \sum_j \mathbf{p}_{i,j} B_i(t) B_j(s), \tag{2.5}$$

where $p_{i,j}$'s are initial control points. Since the tensor product of intervals in $\mathbb{R}^1$ is topologically equivalent to rectangular region in $\mathbb{R}^2$, the domain mesh for the subdivision surface based on the tensor product shall be quadrilateral. Another way to derive subdivision surfaces is using multivariate spines. In this case, the mesh could be different than quadrilateral. In many cases, a triangular mesh is chosen.

2.2.2.1 Approximating Surface Schemes

Catmull-Clark Scheme



(a) (b) (c)

Figure 2.4: An example of the Catmull-Clark subdivision surface. (Image courtesy of Kenneth Joy.)

In the late 70's, Catmull and Clark [14] published an approximating subdivision surface based on quadrilateral meshes. The scheme introduces new vertices recursively in each level from affine combinations of existing vertices. Using new vertices, we split each quadrilateral face into 4 sub-faces. This process is sometimes called as *face-split* or *primal*. In the limit, the scheme produces bi-cubic B-spline surfaces, which are the tensor product version of cubic B-spline curves. In addition to regular vertices of which valences are always 4, the scheme employs rules for *extraordinary* vertices. Because an initial mesh is finite quadrilateral lattice, we can assume that extraordinary vertices are finite. During the process, the number of extraordinary vertices never changes. Therefore, the limit surfaces belong to C^2 class except near extraordinary vertices, where they achieve C^1 continuity. The tangent-plane continuity of the scheme was proved by Ball and Storry [5]. The C^1 continuity at extraordinary vertices are analyzed by Peters and Reif [73]. The scheme can be summarized as the following rules:

- Face point: For each quadrilateral face $\square\mathbf{p}_0\mathbf{p}_1\mathbf{p}_2\mathbf{p}_3$, we introduce new point $\mathbf{f}$ which is its centroid, *i.e.*$\mathbf{f} = \frac{1}{4}(\mathbf{p}_0 + \mathbf{p}_1 + \mathbf{p}_2 + \mathbf{p}_3)$.

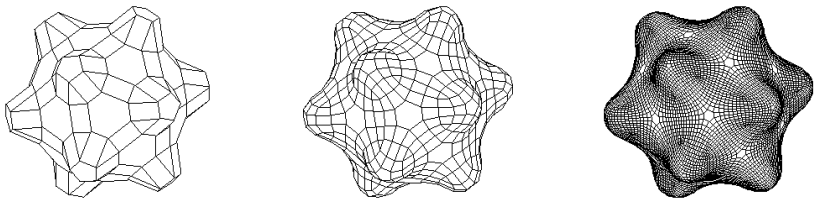
- Edge point: For each edge $\overline{\mathbf{q}_0\mathbf{q}_1}$, we introduce new edge point $\mathbf{e}$ which is a weighted average of its end points and 4 other vertices from two faces that share the edge. The formula can be written as $\mathbf{e} = \frac{3}{8}(\mathbf{q}_0 + \mathbf{q}_1) + \frac{1}{16}(\mathbf{q}'_0 + \mathbf{q}'_1 + \mathbf{q}'_2 + \mathbf{q}'_3)$ where $\mathbf{q}'_i$'s are remaining vertices from two faces that share the edge.

- Vertex point: For each vertex $\mathbf{p}$ with the valence n, we calculate new vertex point $\mathbf{v}$ by the following formula: $\mathbf{v} = \frac{1}{n}\big(\mathbf{f}_{avg} + \mathbf{m}_{avg} + (n-3)\mathbf{p}\big)$, where $\mathbf{f}_{avg}$ is the average of the centroids of the faces that share the vertex, $\mathbf{m}_{avg}$ is the average of the mid-points of the edges that contains the vertex.

Many other researchers have intensively studied the Catmull-Clark scheme and its variations due to its low degree and high continuity. Especially, DeRoes *et al.*[30] discussed

further extension of the scheme to handle special situations, like creases, sharp features and boundaries. The motivation was to extend the scheme to be used in commercial modeling, especially studio Pixar's animated films. Recently, Biermann *et al.*[6, 7] also suggested the modification of the scheme to introduce sharp features. Stam [91] proposed an exact evaluation of the scheme without explicit subdivision process by utilizing eigenvector spaces of subdivision matrices.

Doo-Sabin Scheme



(a) (b) (c)

Figure 2.5: An example of the Doo-Sabin subdivision surface. (Image courtesy of Ha Quang Le.)

In the same year, Doo and Sabin [31] also proposed new approximating subdivision surfaces on quadrilateral meshes. The limit of this process results in a bi-quadratic B-splines. Unlike the Catmull-Clark scheme, they use the *vertex-split* method to generate sub-faces. Instead of introducing 4 sub-faces for each quadrilateral, each vertex is replaced by n vertices where n is the number of faces that share the vertex. For regular topology, it only requires the following rule:

$$\mathbf{v}_{new} = \frac{9}{16}\mathbf{p}_0 + \frac{3}{16}(\mathbf{p}_1 + \mathbf{p}_2) + \frac{1}{16}\mathbf{p}_3, \tag{2.6}$$

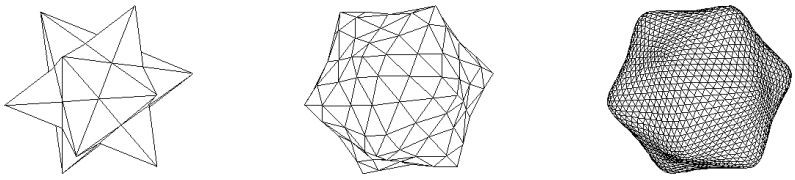
where $\mathbf{p}_i$'s are the vertices of a quadrilateral that contains the new vertex. $\mathbf{p}_0$ is the closest

to the vertex and $\mathbf{p}_3$ is the farthest. Exceptional rules are introduced for extraordinary vertices and boundaries. Overall, the process achieve C^1 continuity [73]. Narsi [67] suggested different rules for boundaries. Later, Habib and Warren [45] and Peters and Reif [74] proposed even simpler version of the Doo-Sabin scheme, that is called "Midedge" subdivision which only uses 3 out of 4 vertices that described above. The regular rule can be reduced to

$$\mathbf{v}_{new} = \frac{1}{2}\mathbf{p}_0 + \frac{1}{4}(\mathbf{p}_1 + \mathbf{p}_2). \tag{2.7}$$

In addition to the scheme, Doo and Sabin [31]'s work also has an important role in the subdivision analysis. They introduced the usage of subdivision matrix and Discrete Fourier Transform (DFT) as a main tool to analyze subdivision scheme at extraordinary vertices.

Loop Scheme



(a) (b) (c)

Figure 2.6: An example of the Loop subdivision surface. (Image courtesy of Ha Quang Le.)

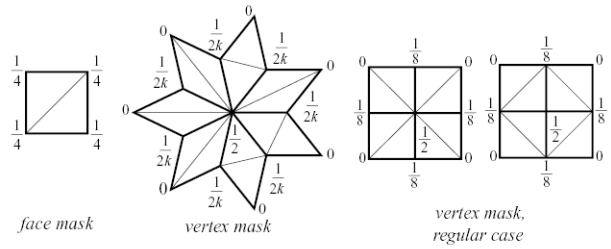
Charles Loop [58] proposed an approximating subdivision surfaces based on triangular meshes in his master thesis. The scheme is different from others in many ways. First, the underlying domain is triangular. Each triangle is split into 4 sub-faces by introducing new vertex and edge points. Triangular meshes are often more flexible than quadrilateral meshes, which makes it ideal for complex models. Also, any polyhedral meshes can be

easily converted into triangular meshes by any triangulation. Even Many popular algorithms exist to generate triangular meshes from unorganized point sets, such as Delaunay triangulation [27] and Voronoi diagrams [1] . Moreover, the scheme is based on the double 3-directional box spline [29], which attains C^2 continuity without requiring high degree of polynomial. In fact, the evaluation of the box splines through subdivision had been researched before [28, 10, 11, 25, 22], but Loop's work includes the extraordinary vertex rule and its analysis, which guarantees C^1 continuity over the mesh. The rule can be specified into two cases.

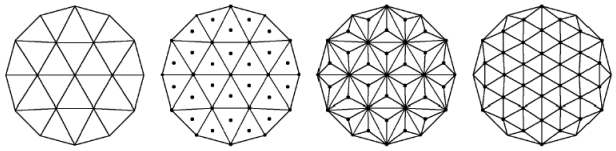
- Edge point: For each edge $\overline{\mathbf{p}_0\mathbf{p}_1}$, we introduce new edge point $\mathbf{e} = \frac{3}{8}(\mathbf{p}_0 + \mathbf{p}_1) + \frac{1}{8}(\mathbf{q}_0 + \mathbf{q}_1)$, where $\triangle\mathbf{p}_0\mathbf{p}_1\mathbf{q}_0$ and $\triangle\mathbf{p}_0\mathbf{p}_1\mathbf{q}_1$ are the neighboring triangles.

- Vertex point: For each vertex $\mathbf{p}$ with the valence n, we introduce new vertex point $\mathbf{v}$ by averaging adjacent vertices with weight. The rule is: $\mathbf{v} = (1 - n\beta)\mathbf{p} + \beta(\mathbf{q}_0 + \cdots + \mathbf{q}_{n-1})$ where $\beta = \frac{1}{n}\big(\frac{5}{8} - (\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n})^2\big)$.

The C^1 continuity of the Loop scheme up to $n < 100$ was proven by Schweitzer [88]. Later, Zorin proved the continuity for all valence in his work [104]. The choice of β is appeared in Loop's paper along with the analysis of extraordinary vertices. Later Warren [94] proposed simpler rule by choosing $\beta = \frac{3}{8n}$ for $n > 3$ and $\beta = \frac{3}{16}$ for $n = 3$. Hoppe *et al.*[46] and Biermann *et al.*[8] suggested extensions of the Loop scheme with many special rules, including edges, corners, *etc.* Many geometric properties of the scheme has been proven by researchers, including Zorin *et al.*'s [106] explicit evaluation of the scheme. Ying *et al.*[102] proposed new extension of the Loop scheme that can handle non-manifold objects.



(a)



(b)

Figure 2.7: (a) The subdivision rules for the 4-8 subdivision. (Image courtesy of Luiz Velho.) (b) The face split in the $\sqrt{3}$-subdivision. (Image curtesy of Leif Kobbelt)

Other Approximating Surface Schemes

Recently, Velho [92] proposed "4-8 subdivision". The scheme is based on the four-directional box spline, but uses a different split, called the $[4.8^2]$ Laves tiling. Unlike the face-split, which quadruples the number of faces in each subdivision, it only doubles the number of faces. Also, it has higher C^4 continuity almost everywhere, except extraordinary vertices where it is C^1.

Kobbelt [51] suggested new subdivision scheme, which is called "$\sqrt{3}$-subdivision". It only introduces one new point for each face. However, in each step, the subdivision process is applied twice, which causes tri-section of each edge, unlike the common face-split, which causes quad-section. Therefore, it is called the $\sqrt{3}$-subdivision. This results in a slower increase of the number of faces. Its masks (or stencils) for the subdivision rules have minimum size and maximum symmetry. It has C^2 continuity with C^1 at extraordinary vertices. Because of new splitting strategy, it also enables possible adaptive refinement under preservation of the mesh consistency.

2.2.2.2 Interpolatory Surface Schemes

Interpolatory schemes are subdivision schemes that interpolate initial control points in the limit. During the subdivision process, the control points are geometrically invariant once they are introduced. Interpolatory schemes have the following merits over approximating schemes:

- it is straightforward to enforce constraints during physics-based simulation, it supports intuitive, direct manipulation of control points,
- there is no need for an auxiliary subdivision matrix for vertex points, and
- it is not necessary for subdivision matrix inversion during data fitting applications.

Butterfly Scheme

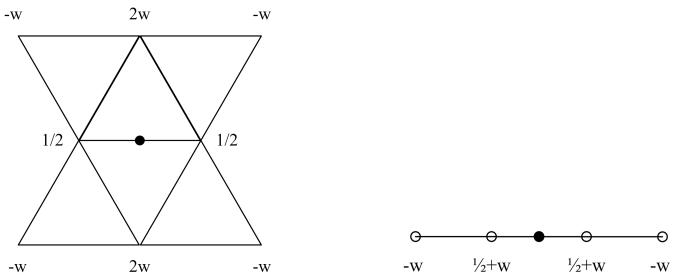


Figure 2.8: The subdivision masks for the Butterfly scheme. (a) The regular mask. (b) The boundary mask (the 4-point scheme).

Dyn *et al.*[35, 38, 32, 37, 34] developed interpolatory subdivision curves and extended them to surface cases. They introduced an interpolatory subdivision scheme [36, 38] based on their interpolatory curves. The scheme is oftentimes called “Butterfly scheme” because of the shape of its neighbor mask. It is a face-split and triangular-based scheme. At level ℓ,

we introduce new edge point $\mathbf{e}^{\ell+1}$ by the following rule:

$$\mathbf{e}^{\ell+1} = \frac{1}{2}(\mathbf{p}_0^\ell + \mathbf{p}_1^\ell) + 2w(\mathbf{p}_2^\ell + \mathbf{p}_3^\ell) - w(\mathbf{p}_4^\ell + \mathbf{p}_5^\ell + \mathbf{p}_6^\ell + \mathbf{p}_7^\ell). \tag{2.8}$$

Like the 4-point scheme, the choice of w determines the convergence and the continuity of the scheme. They proved [38] the range where the scheme is convergent and therefore guarantees C^1 continuity. Typically, $w = \frac{1}{16}$ is chosen. It is proven that the scheme is convergent when $w \in (0, 0.096)$. Dyn *et al.*[33] also suggested the variations of the scheme, including "The Averaged-Butterfly scheme" and "A truncated tensor product scheme". For the extraordinary cases, Zorin [104] proved it is C^1 for $k \leq 7$ and $k \neq 3$. Zorin *et al.*[108] proposed a modified version of the Butterfly scheme, which guarantees C^1 everywhere. A detailed extension of the Butterfly scheme is well documented in the course note of SIGGRAPH 2000 [107].

Kobbelt Scheme

Kobbelt [49] presented an interpolatory subdivision surface defined on arbitrary quadrilateral meshes. It is the tensor product of the 4-point scheme. For extraordinary vertices, it achieve C^1 continuity for any valence. One shortfall of the scheme is that it is impossible to compute eigenvectors of the subdivision matrix explicitly. Therefore, there are no precise expression for tangents.

2.2.3 Variational Subdivision Schemes

Although stationary schemes are desirable in many cases because of their simplicity, most of the stationary subdivision schemes have C^1 continuity at extraordinary vertices. It is possible to acquire a subdivision rule which is C^2 continuous everywhere [80, 81], but it requires very large masks or has zero curvature at extraordinary vertices. Prautzsch

and Umlauf [82] suggested a solution to this situation by introducing G^2 surfaces near extraordinary vertices. However, the problem of degeneracy is still visible for most of the stationary subdivision schemes. Also, for the interpolatory schemes, the irregularities become more noticeable at extraordinary vertices.

Variational methods for subdivision modeling are related to the *multigrid* method in the numerical analysis [52, 96, 13]. The methods consist of two factors.

- $\mathcal{E}$: the energy or fairness functional.
- $\mathcal{M}$: the discretized domain grid.

Basically, a variational approach try to solve the minimization problem for a given continuous functional over a domain. We discretize the domain into a grid $\mathcal{M}$, and the variational solution can be approximated by a discrete coefficient vector $\mathbf{p}$ with one value per grid point in $\mathcal{M}$. The problem can be expressed as

$$\min_{\mathcal{M}}(\mathbf{p}^T\mathbf{E}\mathbf{p}), \tag{2.9}$$

where $\mathbf{E}$ is a symmetric, positive definite matrix that depends on the functional $\mathcal{E}$. The trivial solution for the minimization is $\mathbf{p} = 0$. To ensure non-trivial solution, we choose extra conditions. For example, a simple linear system

$$\mathbf{E}\mathbf{p} = \mathbf{b} \tag{2.10}$$

can be used as a constraint, where $\mathbf{b}$ is associated boundary condition. The system can be solved directly using numerical methods, such as the Gaussian elimination, or the Jacobi or Gauss-Seidel iteration. However, the direct methods can be expensive in computation time and memory requirement. The multigrid method try to solve the system hierarchically.

Using a sequence of grid $\mathcal{M}_\ell$ which becomes finer over the level ℓ, and the corresponding matrix $\mathbf{E}_\ell$, the solution can be computed in the following three steps [13].

(a) Prediction: Compute an initial guess $\mathbf{p}_\ell^0$ at level ℓ, by computing $\mathbf{p}_\ell^0 = \mathbf{S}_{\ell-1}\mathbf{p}_\ell$. $\mathbf{S}_{\ell-1}$ is a matrix that maps vectors from $\mathcal{M}_{\ell-1}$ to $\mathcal{M}_\ell$. It is called a "prolongation operator". The simplest way is using piecewise linear interpolation.

(b) Smoothing: Improve the quality of the initial guess $\mathbf{p}_\ell^0$ by k iteration of a direct method, *i.e.*Jacobi or Gauss-Seidel iteration. We obtain $\mathbf{p}_\ell^k$ and the residual $\mathbf{r}_\ell^k$.

(c) Coarse grid correction: Eliminate all entries in $\mathbf{r}_\ell^k$ which do not correspond to a grid in $M_{\ell-1}$. Using this restricted residual $\mathbf{r}_{\ell-1}$, solve $\mathbf{E}_{\ell-1}\mathbf{e}_{\ell-1} = \mathbf{r}_{\ell-1}$. Add the correction term $\mathbf{e}_\ell = \mathbf{S}_{\ell-1}\mathbf{e}_{\ell-1}$ to the current $\mathbf{p}_\ell^k$.

If we apply this in subdivision schemes, S_ℓ can be considered as a subdivision matrix. If a variational rule always generates no residue, we can get curves or surfaces that minimizes given energy functional.

Kobbelt [50, 52] summarized the approach to variational subdivision schemes, and displayed several examples of subdivision rules for curves and surfaces that minimize certain fairness functionals, *i.e.*a subdivision rule for curvature minimizing curves. He proposed a systematical way to acquire rules for variational subdivision schemes.

Weimer and Warren [99, 95, 100] published series of research related on variational subdivision schemes that satisfy partial differential equations, for instance, fluid or thin plate equations.

Since the variational schemes are less susceptible to the complex topology and connectivity of meshes, and it is easy to satisfy desired conditions in the limit, this approach has been acquiring popularity recently.

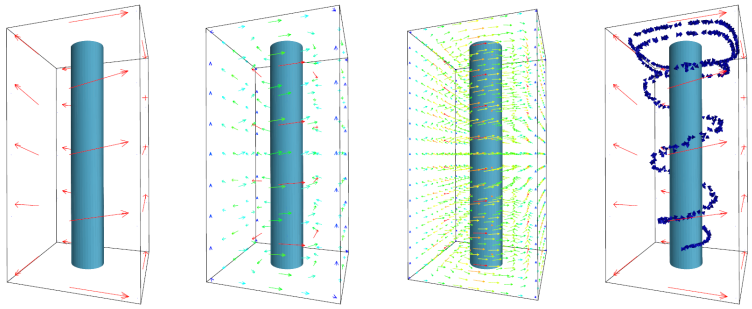


Figure 2.9: An example of a variational subdivision scheme. This example simulates fluid flow in the limit. (Image curtesy of Henrik Weimer)

2.2.4 High-Dimensional Subdivision Schemes

Despite the popularity of subdivision, there has been less research regarding high-dimensional subdivision objects, *i.e.*, subdivision solids or multi-dimensional objects. This is due to the complexity of the domain topology and the lack of proper analysis tools for high-dimensional schemes.

MacCracken and Joy [59] proposed a tensor product extension of the Catmull-Clark scheme in the volumetric setting, mainly for the purpose of free-form deformation in three-dimensional space. In the limit, the process generates tri-cubic B-spline solids, except at extraordinary vertices. Obviously, it is a hexahedral-based, approximating scheme. The rules can be summarized as the following:

- Cell point: For each cell, we introduce new cell point $\mathbf{c}$ which is its centroid.
- Face point: For each face, we introduce new face point $\mathbf{f}$ by computing the weighted average $\mathbf{f} = \frac{1}{4}(\mathbf{c}_0 + 2\overline{\mathbf{f}} + \mathbf{c}_1)$ where $\overline{\mathbf{f}}$ is the face's centroid and $\mathbf{c}_0$ and $\mathbf{c}_1$ are the centroids of the cells on either side of the face.
- Edge point: For each edge, we introduce new edge point $\mathbf{e}$ by computing the weighted average $\mathbf{e} = \frac{1}{n}\big(\mathbf{c}_{avg} + 2\overline{\mathbf{f}}_{avg} + (n-3)\mathbf{m}\big)$, where $\mathbf{c}_{avg}$ and $\overline{\mathbf{f}}_{avg}$ are the averages of the centroids of the cells and the faces that contain the edge, respectively.

$\mathbf{m}$ is the midpoint of the edge, and n is the number of faces that share the edge.

- Vertex point: For each vertex $\mathbf{p}$, we introduce new vertex point $\mathbf{v}$ by computing the weighted average $\mathbf{v} = \frac{1}{8}(\mathbf{c}_{avg} + 3\overline{\mathbf{f}}_{avg} + 3\mathbf{m}_{avg} + \mathbf{p})$, where $\mathbf{c}_{avg}$ and $\overline{\mathbf{f}}_{avg}$ are the averages of the centroids of the cells and the faces that contain the vertex, respectively. $\mathbf{m}_{avg}$ is the average of midpoints of the edges that shares the vertex.

Since the scheme is based on the Catmull-Clark scheme, the rules are similar. Even though the Catmull-Clark scheme has been intensively studied, the analysis, especially for extraordinary cases, of the solid scheme has not been fully exploited. Joy and MacCracken [48] proved the continuity on regular meshes. However, it appears that current techniques to prove the continuity along extraordinary vertices on the subdivision surfaces cannot be directly applied the solid case.

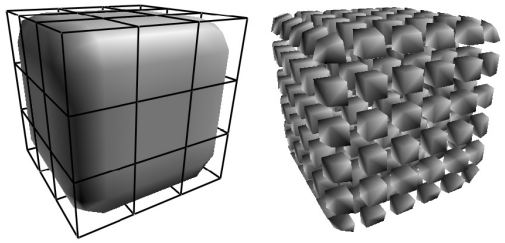
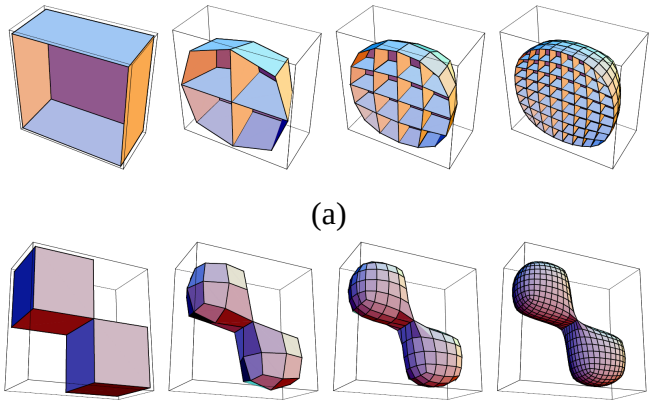


Figure 2.10: A simple example of the MacCracken and Joy's scheme (Image curtesy of Kevin T. McDonnell).

Later on, Bajaj *et al.*[4] further extended the scheme with the analysis based on numerical experiments. They approached the problem by separating the process into multi-linear subdivision followed by a cell averaging operation. Therefore, they called it "MLCA" or "Multi-linear Cell Averaging" scheme. Since the scheme makes no assumption on the local topology of the hexahedral mesh, it can be directly applied to non-manifold situations. They also devised special rules for creases and boundaries. The volumetric analysis for the

continuity is based on numerical experiments. They calculated eigenvalues for extraordinary cases and confirmed that the values are within the range of the convergence.



(a)

(b)

Figure 2.11: The MCLA scheme. (a) An simple solid example. (b) A model with non-manifold topology. (Image curtesy of Chandrajit Bajaj)

Peters *et al.*[75, 76] used the 7-directional trivariate box spline to blend CSG primitives. They used the tri-variate box spline as a filter of three-dimensional space to blend CSG objects embedded in it. To evaluate the filter, well-known subdivision process for computing the box splines [29] is employed. Since the filter is only applied to the regular hexahedral grid, or even the voxel space, it does not have to deal with the complex problems, like the extraordinary topology or the connectivity. Linsen *et al.*[57] recently proposed "$\sqrt[4]{2}$ subdivision" to represent time-varying volume data in hierarchical fashion. This four-dimensional subdivision scheme provides both spatial and temporal scalability. Also, they used quadri-linear B-spline wavelet lifting scheme based on the $\sqrt[4]{2}$ subdivision for high-quality data approximation. Pascucci [69]'s recent work on high dimensional subdivision scheme suggested interesting cell-split of arbitrary polyhedra to slow the increase of cells during the subdivision process. It introduces special *diamond* cells during the process, which makes it hard to analysis.

The variational scheme that was proposed by Weimer and Warren [100] could be considered as solid subdivision scheme, since the fluid flow equation which the scheme is approximating in the limit is based on three-dimensional space. Basically, by choosing a suitable partial differential equation and following the framework [52], it is almost systematic to acquire a variational scheme of which domain is high-dimensional. However, it should be further investigated to fit the purpose of geometric modeling and interactive design.

Finally, the recent subdivision scheme proposed by Schaefer *et al.*[87] is the most relevant subdivision scheme to our own subdivision solids schemes [16, 17] up to date. Their scheme is based on the structure that we have proposed in [16] and the paper contains analysis on special cases, especially, the face-to-face case, using the joint spectral analysis by Levin *et al.*[55].

2.2.5 Analysis of Subdivision Schemes

Many researchers has been working on the subdivision analysis near extraordinary topology, especially for the extraordinary connectivity. More recently, Micchelli [64], Prautzsch [78, 82, 80, 79, 81], Reif [84, 83], and Zorin [108, 104, 105, 106] investigated the sufficient and necessary conditions of the convergence and the continuity. In most cases, they analyze eigenvalues and eigenvectors of subdivision matrices by using spectral analysis tools. Since their approaches to the problem share many similarities, we shall discuss the analysis in unified way. Note that the above analysis is mostly for subdivision surfaces.

Subdivision Matrix

As we have seen in the previous sections, a subdivision rule can be written as linear combinations of control points, thus it can be represented as a linear system using a matrix

form. Since we are concerned in the continuity near extraordinary vertices, we denote an extraordinary vertex at level ℓ as $\mathbf{p}_0^\ell$. Let $\mathbf{p}_1^\ell, \cdots, \mathbf{p}_m^\ell$ be the neighbors of the vertex defined by the subdivision mask. Then the subdivision process can be written by the linear system

$$\mathbf{p}^{\ell+1} = S\mathbf{p}^\ell, \tag{2.11}$$

where $\mathbf{p}^\ell$ consists of $\mathbf{v}_i^\ell$ as its row vectors. We call the matrix S the *subdivision matrix*.

Eigen Analysis by Discrete Fourier Transform

The subdivision matrix has particular properties. First, it forms a cyclic structure inside, because the same subdivision rules are applied around the neighbors and the *planar symmetry* of the mask. Also its eigenvalues and eigenvectors are directly related to the convergence and continuity of the subdivision scheme.

Suppose we find eigenvalues λ_i and corresponding eigenvectors $\mathbf{v}_i$ for the matrix S. Let us assume that λ_i is non-increasing order. Initial control point sets $\mathbf{p}^0$ can be expressed in the eigenspace and eigenbasis explained by these eigenvectors:

$$\mathbf{p}^0 = \mathbf{a}_0\mathbf{v_0} + \mathbf{a}_1\mathbf{v_1} + \cdots + \mathbf{a}_n\mathbf{v_n}. \tag{2.12}$$

Therefore, the control points at level ℓ can be expressed as

$$\mathbf{p}^\ell = S^\ell\mathbf{p}^0 = \lambda_0^\ell\mathbf{a}_0\mathbf{v_0} + \lambda_1^\ell\mathbf{a}_1\mathbf{v_1} + \cdots + \lambda_n^\ell\mathbf{a}_n\mathbf{v_n}. \tag{2.13}$$

Many researchers [78, 84, 80, 105] suggested that the condition

$$\lambda_0 = 1 \gneqq \lambda_1 \geq \lambda_2 \gneqq \lambda_3, \cdots, \lambda_n \tag{2.14}$$

is a necessary condition for the subdivision scheme to be convergent and tangent-plane continuous at the extraordinary vertex $\mathbf{p}_0^0$. If we enforce the condition (Equation 2.14), it is clear to see the limit vertex of the vertex $\mathbf{p}_0^0$ is $\mathbf{a}_0$. Also, the tangent directions are $\mathbf{a}_1$ and $\mathbf{a}_2$ at the same point. The normal vector at the point can be calculated by,

$$\mathbf{n}_v = \mathbf{a}_1 \times \mathbf{a}_2. \tag{2.15}$$

To acquire the subdivision rules near the extraordinary vertex or to prove the continuity in general, we need to compute eigenvalues and eigenvectors of S, *symbolically*. There are many mathematical processes to compute them, but Doo and Sabin [31] and Ball and Storry [5] used Discrete Fourier Transform (or DFT). This was an excellent choice for subdivision surfaces, since the masks for extraordinary vertices always have the planar symmetry. This makes the matrix S forms a cyclic structure inside, and after applying DFT, it is easy to calculate eigenvalues and eigenvectors symbolically. Almost all subdivision analysis follow the same method afterward.

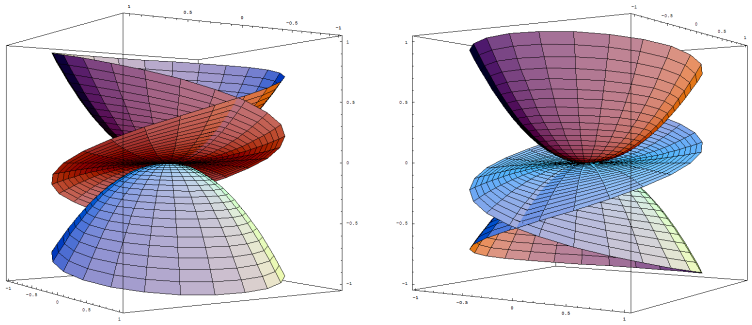


Figure 2.12: 3-D Plotting of the function $f : (r,t) \rightarrow (\cos 3t, r\sin 3t, r^2 \sin t),\ (r,t) \in [0,1] \times [0, 2\pi)$. It is tangent-plane continuous, but not C^1 continuous near the origin.

Characteristic Maps

Note that the condition (Equation 2.14) is only a necessary condition. Also, it is not a C^1 continuous condition. There is an example where a surface satisfy tangent-plane

continuity but it fails to be C^1 continuous (See Figure 2.12). Reif [84] introduced the idea of the *characteristic map* near extraordinary vertices. The map is defined on the patches of the eigenvectors by using S. It can be viewed as a two-dimensional spline function with two-dimensional control points given by the eigenvectors of S. He proved that the subdivision scheme is C^1 continuous if and only if the map is regular and injective. Zorin [105] extended the method further especially for triangular meshes.

Chapter 3

Subdivision Meshes

The key element of any subdivision scheme is its underlying domain mesh and how we split it into finer mesh. It is increasingly important in high dimensional schemes, since the mesh itself becomes non-trivial. In this chapter, we discuss the desired properties of subdivision meshes for stationary schemes and present the regular structured meshes in various dimensions. Later on, we address the definition of simplicial complexes, the motivation behind the choice of simplicial complex meshes, the problems associated a subdivision mesh based on simplicial complex, and a possible solution to the problems.

3.1 Introduction and Motivation

During any subdivision process, we introduce new vertices associated existing elements – vertices, edges, faces, and cells for solid cases– using the information about their neighbors and the derived weight values from the subdivision scheme. The information about the neighbors is acquired from the connectivity among the vertices, which is given by the form of *subdivision meshes*. In the most general case, subdivision meshes can be considered as undirected graphs in $\mathbb{R}^m$. The subdivision meshes are essential in any subdivision

schemes. Not only they provide connectivity information during the process, but also they are deeply connected with the derivation and the generalization of the scheme. In particular, it is desired for subdivision meshes of stationary subdivision schemes to satisfy the following properties.

- *Cell decomposition*: A subdivision mesh consists of polygons or polyhedra of dimension m in $\mathbb{R}^m$. If the mesh admits proper polyhedral decomposition, we call each decomposed element as a *cell*.

- *Self-congruency*: Each cell, if divided by a proper split method, yields sub-cells which are congruent to the original cell.

It is possible that the mesh does not yield a valid polyhedral decomposition in $\mathbb{R}^m$. However, it becomes complicated to generalize the subdivision rules in this case. In the ideal case, the subdivision process will be infinitely repeated to acquire the limit object. Considering the infinite splits of each cell, it is obvious that any cells that satisfy the self-congruency property can tile the entire $\mathbb{R}^m$ space. Therefore, it is clear that any finite subdivision meshes that satisfy the above two properties automatically satisfy the following property:

- *Regularity*: Possible exception of very small number of cases, the number of vertices with different valences are not increasing during the subdivision process. In other words, the extraordinary valences are finite, given by the initial mesh.

It is trivial that a part of any regular structured mesh in $\mathbb{R}^m$ satisfies the properties. Such regular structured mesh is called a *tiling*, or *honeycomb*. The understanding of the regular structured meshes in $\mathbb{R}^m$ is essential to understand the subdivision process.

3.2 Tiling of The Spaces

A regular structured mesh in $\mathbb{R}^m$ can be considered as a tiling of $\mathbb{R}^m$. More precisely, a tiling means a covering of a space or a domain with certain primitive(s) in structured manner. Each primitive is called a *tile*. Generally, it is an interval, a polygon or a polyhedron, depending on the dimension of the domain. A tiling is notably depending on the dimension of a domain [47]. If our domain is one-dimensional, there is only trivial tiling by intervals. It could be uniform tiling or a tiling by intervals of various lengths. Nonetheless, the tiling problem here is almost non-existing.

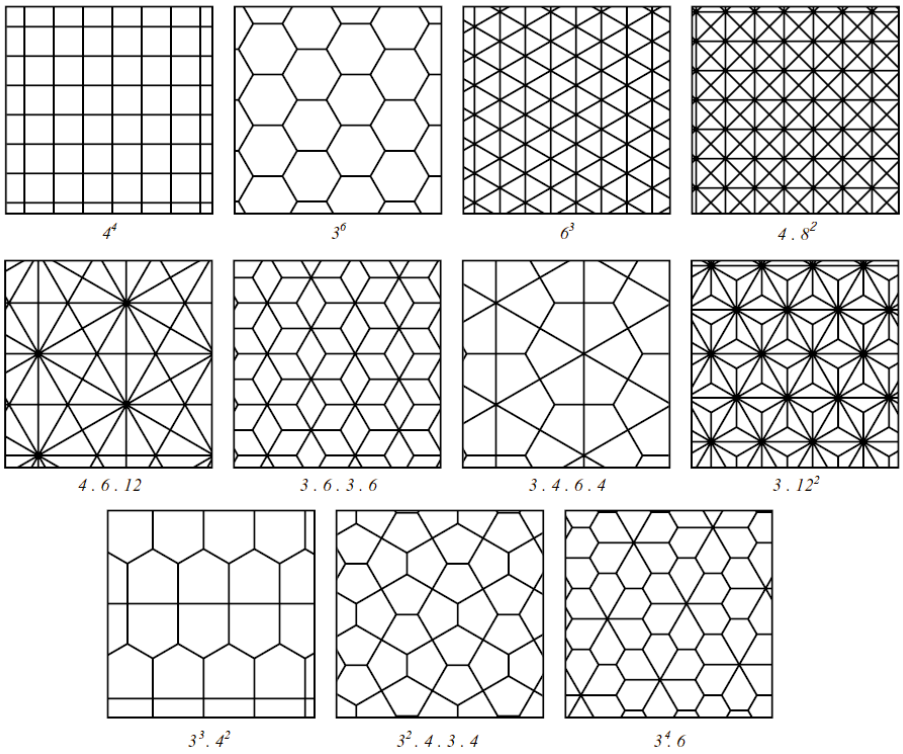


Figure 3.1: The 11 isohedral (Laves) tilings of 2-D space. (Image curtesy of Denis Zorin)

In two-dimensional space, the problem becomes slightly complex. If a tiling consists of only one type of a polygon, it is called an *isohedral*, or a *Laves* tiling. It is known [47, 107] that there are eleven such tilings for the plane. Their *dual* tilings, which can be obtained by

connecting the centroid of each tile, are called *Archimedean* tilings. In these cases, we only consider the convex polytopes as tiles. Archimedean tilings consist of regular polygons, in general. However, they are not isohedral tilings. We refer them as *anisohedral* tilings. For instance, the Catmull-Clark scheme [14] is based on the $[4^4]$ tiling and the Loop scheme [58] and the Butterfly scheme [36] are based on the $[6^3]$ tiling on regular meshes. Kobbelt's $\sqrt{3}$-subdivision scheme [51] produce the $[3.12^2]$ tiling when it is applied once, and then it regenerate the finer $[6^3]$ tiling at the second pass. Velho's 4-8 subdivision [92] is based on the regular $[4.8^2]$ tiling. See Figure 3.1 and Figure 3.2 for the complete listing of isohedral tilings and their duals.

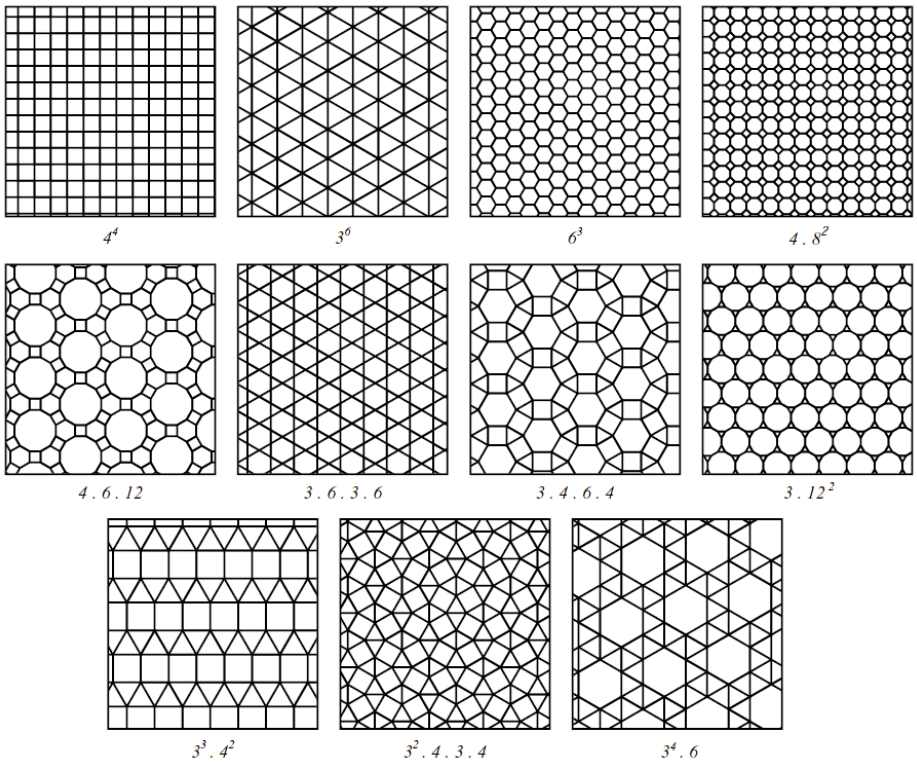


Figure 3.2: The Archimedean tilings, dual to Laves. (Image curtesy of Denis Zorin)

3.3 Regular and Quasi-Regular Structured Meshes

To continue investigating the structured meshes in high dimensional spaces, we adopt the Schläfli symbol to represent regular structures [24]. The Schläfli symbol is defined as the following:

Definition 3.1. *The Schläfli symbol $\{p\}$ defines a regular p-gon. The Schläfli symbol $\{p, q\}$ represents a regular structured mesh in $\mathbb{R}^2$, such that each cell is a regular p-gon and there are q of them at each vertex. The Schläfli symbol $\{\begin{smallmatrix} p \\ q \end{smallmatrix}\}$ represents a dual quasi-regular structured mesh of $\{p, q\}$. In 3D space , the Schläfli symbol $\{p, q, r\}$ defines a regular structured mesh in $\mathbb{R}^3$, such that each cell is $\{p, q\}$, i.e.it is a regular polytope with faces of a regular p-gon and there are q of them at each vertex, and there are r of them surrounding each edge. The same analogue holds for higher dimensional structures.*

Trivially, the regular structure means each cell is a regular polygon or regular polyhedra, and they are all equal. To understand *quasi-regular* structure, we first understand the *vertex figure*. The vertex figure of a vertex of a structured mesh is defined by a polyhedron whose vertices are the mid-points of all the edges that emanate from the given vertex. A mesh is called quasi-regular if its cells are regular while its vertex figures are quasi-regular. It is clear to represent the isohedral tilings using the Schläfli symbol. For instance, $[4^4]$, $[3^6]$, and $[6^3]$ can be rewritten as $\{4, 4\}$, $\{6, 3\}$ and $\{3, 6\}$ in the Schläfli symbol, respectively.

By examining the properties of the Schläfli symbols, one can easily prove that for every $m \geq 2$, $\{4, 3^{m-2}, 4\}$ admits a regular structured mesh in $\mathbb{R}^m$. In fact, this is nothing more than the simple lattice structure $\mathbb{Z}^m$. Unfortunately, it is the only regular structure for $m = 3$ and $m \geq 5$. For $m = 4$, there exist two more regular structured meshes, $\{3, 3, 4, 3\}$ and the reciprocal of it, $\{3, 4, 3, 3\}$. Note that the only regular structure in $\mathbb{R}^3$ is $\{4, 3, 4\}$, the cubic structured mesh.

3.4 Simplicial Complex

A k-simplex S can be defined as a set in $\mathbb{R}^n$,

$$S = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{x} = \sum_{i=1}^{k} c_i(\mathbf{x}_i - \mathbf{x}_0)\}, \tag{3.1}$$

where

$$c_i \geq 1, \ \sum_{i=1}^{k} c_i = 1, \ \mathbf{x}_i \in \mathbb{R}^n. \tag{3.2}$$

Since S can be uniquely determined by $k+1$ points $\mathbf{x}_0$, $\mathbf{x}_1$, ... ,$\mathbf{x}_k$, and is independent of their ordering, we simply use a set notation $S := \{\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_k\}$. In this paper, we limit k to be less than or equal to three. Note that any subset of S also forms a simplex. Geometrically, each subset can be considered as a face, an edge, or a vertex. We call k the *dimension* of the simplex S, or $\dim(S)$. A simplicial complex, or a complex, $\mathcal{C}$ is a collection of simplices where: (1) the subsets of each simplex in $\mathcal{C}$ is in $\mathcal{C}$; (2) the intersection of any two simplices of $\mathcal{C}$ is a subsimplex of both. The second property prevents the introduction of T-junctions or the improper incursion among simplices.

During our Multiresolution Solid Object framework, we choose to use a simplicial complex as our underlying mesh, except for the one occasion (Chapter C). The reasons behind our choice are the followings:

- A simplex is the most simple polytope in any dimensional spaces.
- A simplex is numerically stable. For instance, if you choose the lengths of edges, there exists a unique configuration of the simplex. It is not true for other polytopes in general, for instance, a hexahedron.
- Other polyhedral meshes can be easily converted into simplicial complex meshes.
- Simplicial complex meshes can be effectively acquired from point clouds, volume

data and other high dimensional data by employing robust algorithms, such as Delaunay tessellation algorithm.

- Often times, simplicial complex meshes are the choice of the underlying mesh structure for many numerical analysis and simulations, such as finite element analysis (FEA).

- Since a projection of a simplex is always a simplex, any simplicial complex structure can be displayed as triangular meshes on computer display. Therefore, we can easily take advantage of current triangle-driven, accelerated graphics hardware.

To serve simplicial complex meshes as our subdivision meshes, we first investigate the possibility of the regular structure based on simplicial complexes. In 2D and 3D spaces, there exist regular simplices, a triangle ($\{3\}$) and a tetrahedron ($\{3,3\}$), respectively. The regular triangles facilitate the regular structured mesh in 2D, $\{3,6\}$. In fact, by simple edge bisection, we can easily acquire congruent sub-triangles (Figure 3.3(a)). Therefore, the triangular mesh is well suitable for the subdivision method. The subdivision surfaces, such as the Loop scheme [58] and the Butterfly scheme [36], employ the $\{3,6\}$ meshes as their underlying structure.

However, a simple cell-split by edge bisection of a tetrahedron does not yield congruent sub-tetrahedra. Instead, it produces an octahedron surrounded by four tetrahedra (Figure 3.3(b)). This result makes tetrahedral cells inadequate for the subdivision process. Other splits of a tetrahedron, such as a cell-centroid split and a face-centroid split (Figure 3.3(c)) provide us no better results. Even worse, these splits actually increase the valence of each vertices exponentially throughout the subdivision.

This apparent fallback from the simplicial complex meshes can be overcome by employing a quasi-regular structure in 3D space. By using the Schläfli symbols, one can

deduce that the only structured mesh that involves a regular tetrahedron $\{3, 3\}$ is the quasi-regular structure $\{3, \frac{3}{4}\}$ [24]. As the Schläfli symbol suggests, the structure consists of tetrahedra $\{3, 3\}$ and octahedra $\{3, 4\}$. Figure 3.3(e) illustrate a typical mesh of $\{3, \frac{3}{4}\}$. In fact, this is the only quasi-regular structured mesh in 3D space and is a well-known structure in mathematics. In particular, we call it an *octet-truss* following the naming by an American architect R. B. Fuller [40]. As shown in Figure 3.3(b) and (d), the alternating cell-splits of tetrahedra and octahedra by edge bisection produced the octet-truss structure. The result is very regular. Each vertex has always 14 adjacent vertices and each edge has 2 octahedra and 2 tetrahedra around it.

Still there are several issues that should be addressed before utilizing this particular structure. Specifically, the problems are associated with the subdivision schemes that we are developing. We address the problems during the derivation of each solid subdivision scheme that constitutes our Multiresolution Solid Object framework.

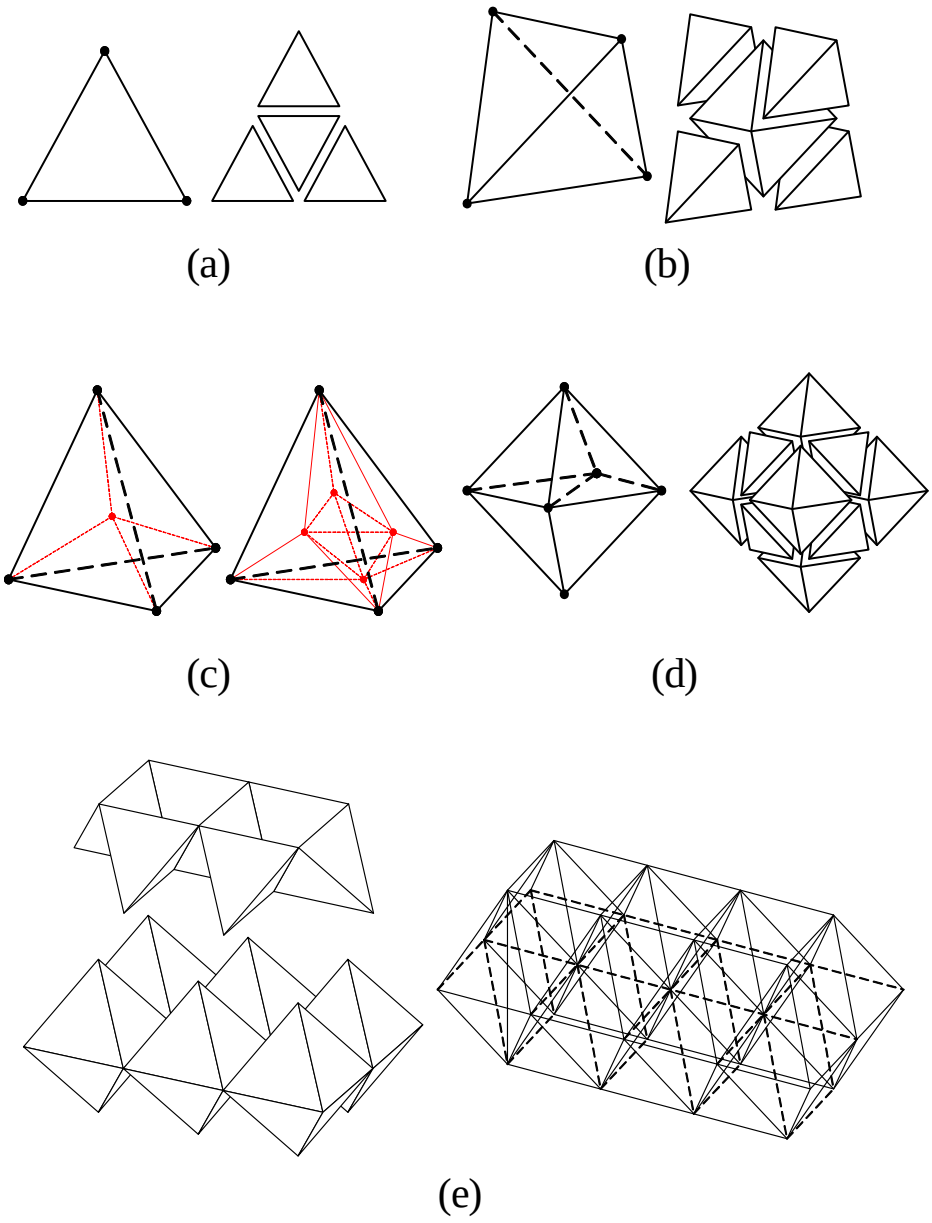


Figure 3.3: Examples of polyhedra splits. (a) A face-split of a triangle. (b) A cell-split of a tetrahedron. (c) Different types of splits of a tetrahedron. (e) A cell-split of an octahedron. (f) An octet-truss structure

Chapter 4

Box-spline based Approximate Solid Subdivision Scheme

Box splines satisfy many properties that make them good candidates for the subdivision method. In this chapter, we propose a new solid subdivision scheme based on a special class of trivariate box splines. The definitions and properties of box splines, the derivation and generalization of the rules, and brief analysis are presented.

The box-spline based approximate solid subdivision scheme was introduced in a paper presented in the 7th ACM Symposium on Solid Modeling Applications 2002 [16]. It also provided a part of the scheme presented in the 9th ACM Symposium on Solid Modeling Applications 2004 [20]. The complete derivation and detailed analysis of the scheme appear in a technical report[21] and a paper that is submitted to Computer Aided Geometric Design for review[19].

4.1 Introduction and Motivation

There have been several attempts to apply the subdivision method to solid modeling. The first attempt has been done by MacCracken and Joy [59], followed by the MLCA scheme by Bajaj *et al.*[4]. These schemes are based on hexahedral meshes in essence, which is no coincidence since they are derived from the tensor-product of other subdivision schemes. In fact, the MacCracken and Joy's scheme and the MLCA scheme are based on the tensor-product of the Catmull-Clark scheme [14]. There are few exceptions including the tetrahedral scheme by Schaefer *et al.*[87]. However, the Schaefer's tetrahedral scheme is founded on the structured meshes that we have proposed [16].

Our motivation is to combine the benefits of various existing solid modeling representations by employing subdivision as its foundation. We base our new subdivision scheme on trivariate box splines and take advantage of its strong mathematical foundation. In addition, we utilize non-tensor-product approach to ensure topological freedom. We introduce the octet-truss structured meshes to take advantage of both the regularity and the flexibility. For this purpose, we choose a special class of splines who has the recursive property and whose supports can be easily embedded in the octet-truss structure. We continue to expand the scheme by generalizing its rules to non-regular cases, *i.e.*arbitrary tetrahedral meshes. We use existing subdivision analysis techniques to prove the C^1 smoothness of our subdivision empirically.

4.2 Properties of Box Splines

4.2.1 Definitions

There are several ways to define the box spline. One constructive way is by considering a projective image of the n-dimensional unit cube, or n-*cube*, in a m-dimensional space

[11] associated an $m \times n$ projection matrix Ξ. We will follow the notations and the definition in *the* box spline book [29] by de Boor and colleagues. Analytically, we define the box spline M_Ξ associated with the matrix Ξ as the distribution given by,

$$M_\Xi : \varphi \mapsto \langle M_\Xi, \varphi \rangle := \int_{\square} \varphi(\Xi \mathbf{t}) d\mathbf{t}, \tag{4.1}$$

where $\varphi \in C(\mathbb{R}^m)$ and $\square = [0,1)^n$, the half-open unit n-cube. By decomposing $\mathbb{R}^n = (\ker \Xi) \bigoplus (\ker \Xi)^\perp$ and applying Fubini's Theorem, we can derive that the distribution can be expressed as

$$\langle M_\Xi, \varphi \rangle := \int_{\mathrm{ran}\Xi} \varphi(\mathbf{x}) \mathrm{vol}_{n-d}\big(\Xi^{-1}\{\mathbf{x}\} \cap \square\big) d\mathbf{x} / |\det \Xi|, \tag{4.2}$$

where $\mathrm{ran}\Xi = (\ker \Xi)^\perp$ and $d = \dim(\mathrm{ran}\Xi)$. We identify the M_Ξ with the function

$$M_\Xi(\mathbf{x}) = \mathrm{vol}_{n-d}\big(\Xi^{-1}\{\mathbf{x}\} \cap \square\big) / |\det \Xi|, \tag{4.3}$$

when $\mathbf{x} \in \mathrm{ran}\Xi$. At each point $\mathbf{x}$, the value $M_\Xi(\mathbf{x})$ is defined by the volume of the cross-section area of $\square$ with $\Xi^{-1}\{\mathbf{X}\}$, divided by the volume of the projected image of a unit volume in $\mathrm{ran}\Xi$. This intrinsic interpretation is important in understanding subdivision process.

Finally, the analytic definition leads us to very useful inductive definition. When Ξ is invertible (*i.e.*$n = m = d$), it is clear that M_Ξ is the normalized characteristic function of $\Xi\square$

$$M_\Xi = \frac{1}{|\det \Xi|} \chi_{\Xi\square}. \tag{4.4}$$

In addition, if $\Xi \cup \zeta$ is any matrix formed from Ξ by the addition of the column $\zeta \in \mathbb{R}^m$,

the box spline $M_{\Xi\cup\zeta}$ is given by the convolution equation

$$M_{\Xi\cup\zeta}(\mathbf{x}) = \int_0^1 M_{\Xi}(\mathbf{x} - t\zeta)dt, \tag{4.5}$$

or simply,

$$M_{\Xi\cup\zeta} = M_{\Xi} * M_{\zeta}. \tag{4.6}$$

4.2.2 Geometric Interpretation

In addition to the analytical definition using the directional matrix presented in the previous chapter, we can define the box splines using purely geometric interpretation. A box B of an n-dimensional space $\mathbb{R}^n$ is defined as

$$B_{(p,p_1,\cdots,p_n)} = \{\mathbf{v} \in \mathbb{R}^n \mid \mathbf{v} = \mathbf{p} + \sum_j c_j\mathbf{p}_j,\ c_j \in [0,1]\},$$

where $\mathbf{p}$ is a box vertex and $\mathbf{p}_j$ are linearly independent vectors in $\mathbb{R}^n$ that are representing n edges of an n-dimensional box. If all of the $\mathbf{p}_j$'s are of unit length, we call it a cube or *hypercube*.

An affine map $\pi : \mathbb{R}^n \to \mathbb{R}^m$ denotes a projection onto an m-dimensional affine space $\mathbb{R}^m$. Consider the image of $B_{(\mathbf{p},\mathbf{p}_1\cdots,\mathbf{p}_n)}$ with respect to the map π. We have

$$\pi\big(B\big) = \{\mathbf{w} \in \mathbb{R}^m \mid \mathbf{w} = \mathbf{q} + \sum_j c_j\mathbf{q}_j,\ c_j \in [0,1]\},$$

where $\mathbf{q} = \pi(\mathbf{p})$ and $\mathbf{q}_j = \pi(\mathbf{p}_j)$.

Since $m < n$, it is obvious that the $\mathbf{q}_j$'s are not linearly independent. Hence, the pre-image of each point $\mathbf{w} \in \mathbb{R}^m$ forms a non-trivial affine subspace:

$$\pi^{-1}(\mathbf{w}) = \{\mathbf{v} \in \mathbb{R}^n \mid \pi(\mathbf{v}) = \mathbf{w}\}, \tag{4.7}$$

which is called a *fibre* of the map π at $\mathbf{w}$. We also derive the fibre by solving the linear system

$$c_1\mathbf{q}_1 + c_2\mathbf{q}_2 + \cdots + c_n\mathbf{q}_n = \mathbf{w} - \mathbf{q},$$

which has dimension $d = n - m$.

We now define a box spline as

$$M_B(\mathbf{w}) = \frac{vol\big(\pi^{-1}(\mathbf{w}) \cap B\big)}{vol\ U(\mathbf{w})},$$

where U is a fixed, d-dimensional unit box parallel to the fibre. Note that M_B has local support and satisfies C^d continuity inside the support. Finally, the normalized version N_B is obtained so that it forms a partition of unity over the lattice $\mathbb{Z}^m$.

This definition is given as a reference purpose. Throughout the chapter, we adhere to the box-spline notations defined on the previous section.

4.2.3 General Properties

Generally, the box spline has the following properties.

(a) *Positive definition*: $M_\Xi \geq 0$ and $\int_{\mathrm{ran}\Xi} M_\Xi = 1$

(b) *Partition of unity*: $f(\mathbf{x}) = \sum_{\mathbf{i}\in\mathbb{Z}^m} M_\Xi(\mathbf{x} - \mathbf{i}) = 1$.

(c) *Piecewise polynomial*: M_Ξ is a piecewise polynomial of degree $n - m$.

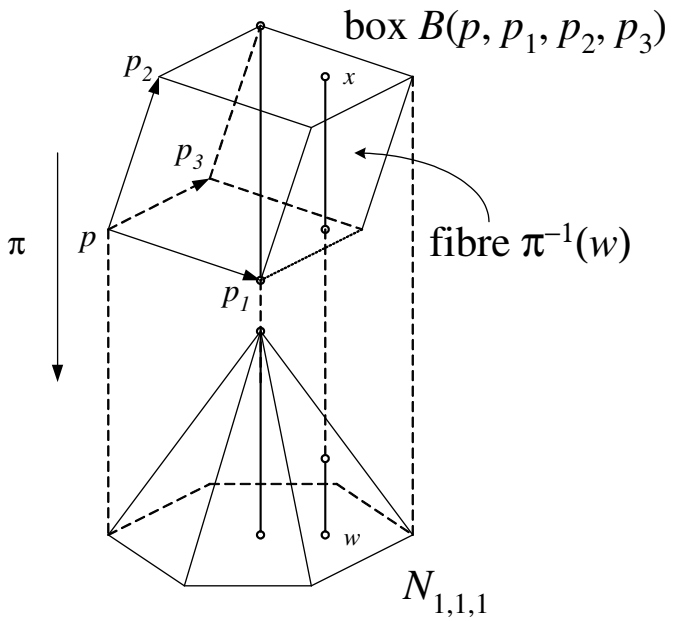


Figure 4.1: Box spline as a projection.

(d) *Continuity*: M_Ξ is a $C^{n-\widetilde{n}-2}$ function, where $\widetilde{n}$ is the maximal number of columns of Ξ that does not span $\mathbb{R}^m$.

We refer the readers who are interested in the proofs of the properties to the Box Spline book [29] by de Boor *et al.*. In addition to the properties aforementioned, a box spline can be expressed by the linear combination of the integer shifts of the box splines with the support of a half-size. It can be formulated as

$$M_\Xi(\mathbf{x}) = \sum_{\mathbf{i}\in\mathbb{Z}^m} s_{\mathbf{i}} M_\Xi(2\mathbf{x} - \mathbf{i}). \tag{4.8}$$

The formula is called a *subdivision* formula. The $s_{\mathbf{i}} \in \mathbb{R}$ is called a *subdivision* coefficient, and the process is called a *subdivision algorithm*.

The simplest case is when $\Xi = \mathbf{E}$, the $m \times m$ identity matrix. It is obvious that $M_{\mathbf{E}} = \chi_{\mathbf{E}\square}$, and it has a simple subdivision formula,

$$M_{\mathbf{E}}(\mathbf{x}) = \sum_{\mathbf{i}\in\mathbb{Z}^m} M_{\mathbf{E}}(2\mathbf{x} - \mathbf{i}). \tag{4.9}$$

We can easily derive subdivision formulae for the other box splines by using the convolution equation (4.5) and the following theorem.

Theorem 4.1. *Suppose M_Ξ has a subdivision formula $M_\Xi = \sum_{\mathbf{i}} s_{\mathbf{i}} M_\Xi(2\mathbf{x} - \mathbf{i})$ and $\Phi = \Xi \cup \zeta$, where ζ is a column vector in $\mathbb{Z}^m$. Then M_Φ satisfies the following subdivision formula,*

$$M_\Phi(\mathbf{x}) = \frac{1}{2}\sum_{\mathbf{i}} (s_{\mathbf{i}} + s_{\mathbf{i}-\zeta}) M_\Phi(2\mathbf{x} - \mathbf{i}). \tag{4.10}$$

Proof 4.1. *By the convolution formula,*

$$\begin{aligned}
M_\Phi(\mathbf{x}) &= \int_0^1 M_\Xi(\mathbf{x} - \mathbf{t}\zeta) d\mathbf{t} \\
&= \int_0^1 \sum_{\mathbf{i}} s_{\mathbf{i}} M_\Xi(2(\mathbf{x} - \mathbf{t}\zeta) - \mathbf{i}) d\mathbf{t} \\
&= \frac{1}{2}\sum_{\mathbf{i}} s_{\mathbf{i}} \int_0^2 M_\Xi(2\mathbf{x} - \mathbf{i} - \mathbf{u}\zeta) d\mathbf{u} \\
&= \frac{1}{2}\sum_{\mathbf{i}} \Big[s_{\mathbf{i}} \int_0^1 M_\Xi(2\mathbf{x} - \mathbf{i} - \mathbf{u}\zeta) d\mathbf{u} + s_{\mathbf{i}} \int_1^2 M_\Xi(2\mathbf{x} - \mathbf{i} - \mathbf{u}\zeta) d\mathbf{u} \Big] \\
&= \frac{1}{2}\sum_{\mathbf{i}} \Big[s_{\mathbf{i}} M_\Phi(2\mathbf{x} - \mathbf{i}) + s_{\mathbf{i}} \int_0^1 M_\Xi(2\mathbf{x} - \mathbf{i} - \zeta - \mathbf{u}\zeta) d\mathbf{u} \Big] \\
&= \frac{1}{2}\sum_{\mathbf{i}} \big[s_{\mathbf{i}} M_\Phi(2\mathbf{x} - \mathbf{i}) + s_{\mathbf{i}} M_\Phi(2\mathbf{x} - \mathbf{i} - \zeta) \big] \\
&= \frac{1}{2}\sum_{\mathbf{i}} (s_{\mathbf{i}} + s_{\mathbf{i}-\zeta}) M_\Phi(2\mathbf{x} - \mathbf{i}).
\end{aligned}$$

Even though it is possible to derive the subdivision rules for a particular box spline by Theorem 4.1, it is more convenient to inspect the coefficients themselves, as a discrete

convolution. (See [97]). We explain the method in the next section.

4.3 Generating Functions for Subdivision

4.3.1 Generating Function Method

A subdivision algorithm can be considered as the discrete convolution among polynomials (See [97], [38]). Once the relation between the subdivision coefficients and the polynomials (usually called *generating function*) are established, the subdivision formula can be acquired systematically.

For example, the univariate B-spline function of degree 0 $B_0(t)$ has the trivial subdivision formula which can be expressed as

$$B_0(t) = B_0(2t) + B_0(2t-1). \tag{4.11}$$

In addition, B-splines can be defined by successive convolution [29]. For univariate case, the definition of B-spline $B_n(t)$ of degree n can be described as the following:

$$B_0(t) = 1 \text{ if } t \in [0,1),\ B_0(t) = 0 \text{ otherwise.} \tag{4.12}$$

$$B_{n+1}(t) = B_n(t) * B_0(t). \tag{4.13}$$

The convolution operator $*$ between functions f and g is defined as

$$(f * g)(t) = \int f(\tau) g(t-\tau) d\tau. \tag{4.14}$$

From this definition, we can directly derive a *refinement equation* [107] which is essential

to the subdivision process for B-spline curves. The equation is given by

$$B_n(t) = \frac{1}{2^n} \sum_{k=0}^{n+1} \binom{n+1}{k} B_n(2t-k). \tag{4.15}$$

Now, we associate the formal generating function $f_{B_0}(z) = 1 + z$ with the spline. It is done by associating the translations in the subdivision formula and their coefficients with the powers of z and their coefficients. It has been proven [107, 97] that if f and g are associated generating functions for the subdivision rule $A(t) = \sum a_i A(2t-i)$ and $B(t) = \sum b_j B(2t-j)$, then the subdivision formula C for the convolution of A and B is $C(t) = \sum c_k C(2t-k)$ where c_k's can be found by comparing the coefficients of its formal generating function $h(z) = \frac{1}{2} f(z)g(z)$. By using Equation 4.13, or the recursive definition, we can easily find that the generating function of the spline of degree n is

$$f_{B_n}(z) = \frac{1}{2^n}(1+z)^{n+1}. \tag{4.16}$$

We can easily verify that the coefficients acquired from Equation 4.16 match exactly with the coefficients from Equation 4.15, by using binomial expansion.

In specific, The Chaikin's algorithm is quadratic case of the univariate B-spline curve. Therefore, by letting $n = 2$, we have the generating function:

$$f_{B_2}(z) = \frac{1}{4}(1 + 3z + 3z^3 + z^3), \tag{4.17}$$

which can be reorganized as

$$f_{B_2}(z) = (\frac{1}{4} + \frac{3z^2}{4}) + (\frac{3z}{4} + \frac{z^3}{4}). \tag{4.18}$$

The coefficients for this formula are exactly corresponding to the well-known subdivision

coefficients of the Chaikin's algorithm (Figure 4.2).

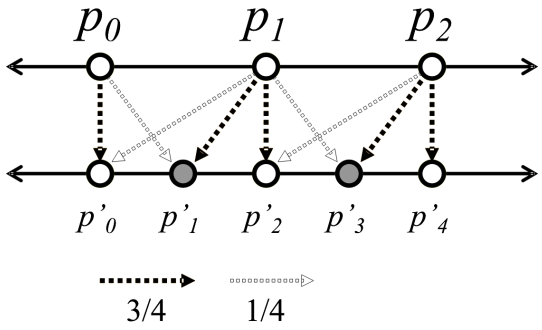


Figure 4.2: The subdivision coefficients for Chaikin's algorithm.

4.3.2 Generating Functions for Box-spline Subdivision

We will associate a subdivision formula for box splines with a multivariate polynomial which represents the generating function. Suppose the box spline M_Ξ has a subdivision formula $M_\Xi = \sum_{\mathbf{i}} s_{\mathbf{i}} M_\Xi(2 \cdot - \mathbf{i})$, then we assign,

$$f_\Xi(\mathbf{z}) = \sum_{\mathbf{i} \in \mathbb{Z}^m} s_{\mathbf{i}} \mathbf{z}^{\mathbf{i}}, \tag{4.19}$$

to its generating function, where $\mathbf{z} = (z_1, \cdots, z_m)$ and $\mathbf{z}^{\mathbf{i}} = ({z_1}^{i_1}, \ldots, {z_m}^{i_m})$. It is clear to see that one can derive the analog of Theorem 4.1 for the generating function, as following:

Corollary 4.1. *Suppose M_Ξ has a subdivision formula associated with a generating function $f_\Xi(\mathbf{z}) = \sum_{\mathbf{i} \in \mathbb{Z}^m} s_{\mathbf{i}} \mathbf{z}^{\mathbf{i}}$. If $\Phi = \Xi \cup \zeta$, then M_Φ has a subdivision formula whose generating function is*

$$f_\Phi(\mathbf{z}) = \frac{1}{2} \sum_{\mathbf{i} \in \mathbb{Z}^m} (s_{\mathbf{i}} + s_{\mathbf{i}-\zeta}) \mathbf{z}^{\mathbf{i}}, \tag{4.20}$$

The result of Corollary 4.1 can be rewritten as $f_\Phi(\mathbf{z}) = \frac{1}{2}(1 + \mathbf{z}^\zeta) f_\Xi(\mathbf{z})$. By applying Corollary 4.1 recursively, we can acquire the following general formula for the box splines.

Corollary 4.2. *For any box spline M_Ξ represented by the directional matrix Ξ, its generating function f_Ξ can be written as*

$$f_\Xi(\mathbf{z}) = \frac{1}{2^{d-k}} \prod_\zeta (1 + \mathbf{z}^\zeta), \tag{4.21}$$

where d is the number of columns in the matrix Ξ and ζ is each column vector of Ξ. For the proofs and further discussion on generation functions, we refer readers to the book by Warren *et al.*([97]).

4.4 Derivation of Solid Subdivision Scheme

4.4.1 Double Directional Box Splines

We focus on the particular box spline of our interest, namely, the *double directional* box splines for our subdivision scheme. First, we consider a piecewise linear box spline function. Suppose $\Xi_1 = \mathbf{E} \cup \delta$, where $\delta = [1, \cdots, 1] \in \mathbb{R}^m$. Then, M_{Ξ_1} forms a piecewise linear function over $\mathbb{R}^m$. The support of the spine function is the form of two unit squares, or cubes, sharing one vertex, with additional edges that join corresponding vertices in each.

The double directional box spline can be understood as a projection of the unit $2(m+1)$-cube into $\mathbb{R}^m$, as previously mentioned. The spline can be defined by the directional matrix $\Xi_2 = \Xi_1 \cup \Xi_1$. Since the matrix projects a pair of edges into the same place in $\mathbb{R}^m$, the support of the double directional box spline is the exactly same shape as the piecewise linear box spline. The supports for the cases of $m = 1$ and $m = 2$ are shown in Figure 4.3.

For example, the Loop's scheme employs the bivariate double directional box spline. In this case, the directional matrix is $\Xi = \{(1,0),(0,1),(1,1),(1,0),(0,1),(1,1)\}$, therefore

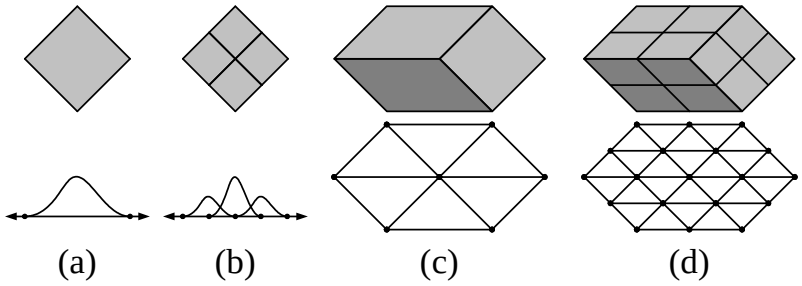


Figure 4.3: The supports of the box splines and their subdivision.

the generating function is

$$f_{\Xi}(z_1, z_2) = \frac{1}{16}(1 + z_1)^2(1 + z_2)^2(1 + z_1 z_2)^2, \tag{4.22}$$

which can be deduced from Corollary 4.2.

In our research, we restrict ourselves to the case of $\mathbb{R}^3$, which represents solid objects in $\mathbb{R}^3$. By applying Corollary 4.2 to the directional matrix Ξ_2 in 3-dimensional case, we acquire the following generating function:

$$f_{\Xi_2}(z_1, z_2, z_3) = \frac{1}{32}(1 + z_1)^2(1 + z_2)^2(1 + z_3)^2(1 + z_1 z_2 z_3)^2. \tag{4.23}$$

The coefficient of each polynomial term $z_1^{i_1} z_2^{i_2} z_3^{i_3}$ represents the weight value for the subdivision algorithm at (i_1, i_2, i_3) in $\mathbb{Z}^3$.

4.4.2 Regular Subdivision Masks

To derive the subdivision rules for the regular cases systematically, we proceed with the following steps:

(a) Compute the generating function of the subdivision algorithm.

(b) Associate the acquired subdivision coefficients with $\mathbb{Z}^m$ by the power of their polynomial terms. For instance, if the coefficient c is for the term $\mathbf{z}^{\mathbf{i}} = ({z_1}^{i_1}, \cdots, {z_m}^{i_m})$, the c is assigned to the lattice point $\mathbf{i} = (i_1, \cdots, i_m)$.

(c) Index a part of our subdivision mesh properly so that we can establish one-to-one correspondence between the non-zero coefficients and the mesh vertices.

(d) Extract the subdivision masks for the vertices, edges, and cells if necessary.

(e) Reduce the number of the masks by considering rotational symmetry.

We start with the Loop's scheme as an example. As shown in Figure 4.3, the projected image of the subdivided 6-cube by the matrix $\Xi = \{(1,0),(0,1),(1,1),(1,0),(0,1),(1,1)\}$ are the shape of a hexahedron, and its coefficients can be properly embedded in 2-ring vertex neighbor of $\{3,6\}$ structure. From Equation 4.22, we acquire the subdivision coefficients and place them on the part of $\mathbb{Z}^2$ as shown in Figure 4.4 (a). We establish one-to-one correspondence with the $\{3,6\}$ structure by simply adding diagonal edges.

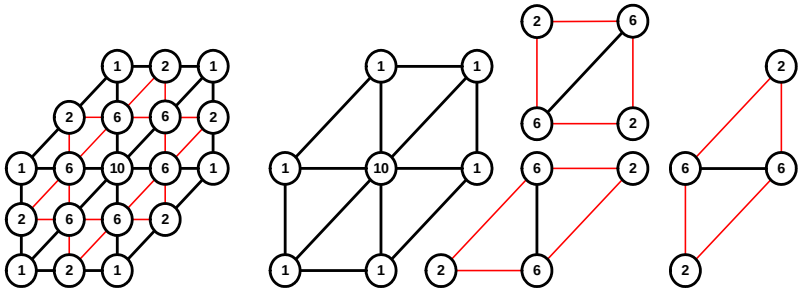


Figure 4.4: (a) The subdivision coefficients of the Loop's scheme are embedded in the part of the $\{3,6\}$ regular structured mesh. (b) and (c) The extracted vertex and edge masks. All the values are to be multiplied by $\frac{1}{16}$.

Now we can extract the subdivision masks. For vertex cases, we consider the center vertex as our vertex point. Since we are performing the subdivision using the edge bisection, the edges of our masks should have the size of two in each direction. Therefore, we can extract the Loop's subdivision mask for regular vertices, shown in Figure 4.4 (b). For edge cases, we consider newly introduced edge point is on the center. Then there are 3 choices of the length 2 edges and the associated masks, shown in Figure 4.4 (c). Since all these masks are identical upon the rotation of the degree $\frac{2\pi}{3}$, we can reduce them into a single edge subdivision mask.

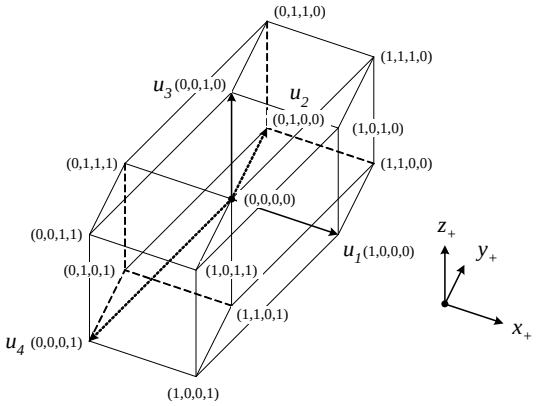


Figure 4.5: The projected image of 8-cube by the directional matrix Ξ_2.

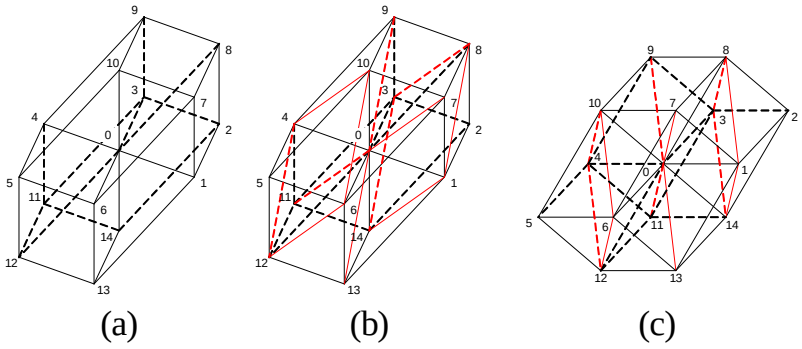


Figure 4.6: (a) The projected image of the hypercube. (b) New edges (red) are added to make proper polyhedral decomposition. (c) The one-to-one correspondence with a part of the octet-truss mesh.

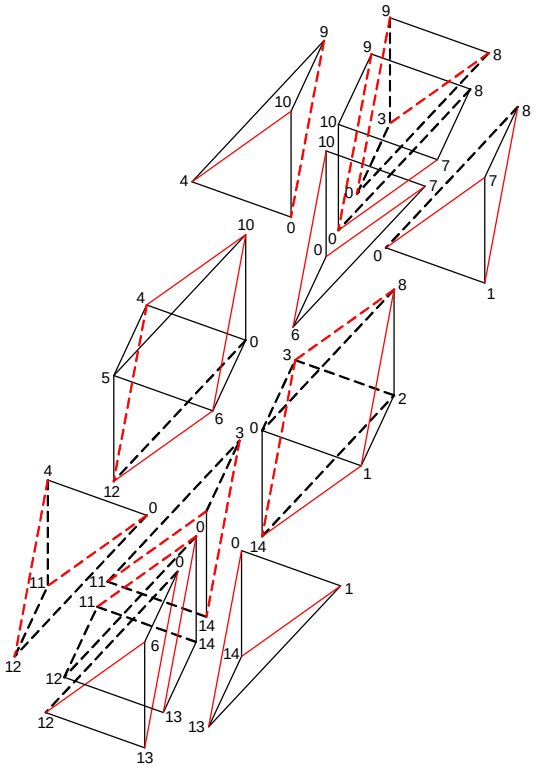


Figure 4.7: The decomposition of the hypercube image with the additional edges (red).

The extraction of the subdivision masks from our double 4-directional box spline are far more complicated than the Loop's scheme. First, consider the projected image of 8-cube by the directional matrix Ξ_2 (Figure 4.5). Actually, the boundary of the image forms a space-filling rhombic dodecahedron. However, if you consider the other edges, it does not admit any proper polyhedral decomposition. Therefore, we have to add few more edges (Figure 4.6 (b)), so that we get proper polyhedral decomposition (Figure 4.7), as well as the one-to-one correspondence with a part of the octet-truss mesh (Figure 4.6 (c)). Now, we consider the one level subdivision of the mesh and assign the subdivision coefficients from Equation 4.23 on the vertices as shown in Figure A.1 in Appendix A. During the mask extraction, one should be cautious that there are edges between each layers as well as vertices within the same layer. In fact, all diagonals in Figure A.1 represent the edges

between layers, except the blue diagonals in the third layer. Figure A.2 in Appendix A displays the vertex masks for our subdivision scheme. The edge masks are shown from Figure A.3 to Figure A.8 in Appendix A. Unlike the Loop's scheme, we have two types of the edge masks which are not identical by rotation. The edge masks (a), (b), (c), and (d) have 6 neighbors in addition to each end point of the edge, whereas the edge masks (e) and (f) only have 4 neighbors. In fact, The masks (e) and (f) are similar to the cell mask, shown in Figure A.9. This occurs because of the unified direction choice of the diagonal in each octahedron, which are shown as blue edges in the figures. Note that the original octet-truss mesh and its subdivision does not requires the choice of the diagonal inside the octahedral cells. However, during the assignment of the coefficients onto the mesh, we need to choose one particular direction (defined by the octahedral diagonal), because of the asymmetry of the projected image of the 8-cube in 3D space.

We call this choice of the octahedral diagonal as the *major diagonal*. Moreover, we have to choose the diagonal in such way that no vertex is shared by more than two major diagonals. This causes a problem in the implementation of the subdivision scheme. We will discuss more about the implementation issues in the upcoming sections.

Definition 4.1. *For each octahedral cell* $[\mathbf{x}_i, ..., \mathbf{x}_{i+5}]$ *that comprises the mesh, we choose one pair of vertices* $\{\mathbf{x}_{j_1}, \mathbf{x}_{j_2}\}$ *which are not adjacent. The pair is called the* major diagonal *and is denoted by* $\mathbf{m} = [\mathbf{x}_{j_1}, \mathbf{x}_{j_2}]$. *We choose the major diagonals in such way that for any vertex in the mesh, it is not shared by more than two major diagonals. When subdividing, sub-octahedral cells inherit the directions of the major diagonals from their parent cells.*

4.4.3 Regular Subdivision Rules

Since we have already identified the subdivision masks, it is relatively trivial to present the regular subdivision rules for our scheme. The rules reduce into one vertex rule, two

edge rules and one cell rule. $\rho(\cdot)$ denotes the one-neighbor of the adjacent vertices of a vertex or an edge. We emphasize that the adjacency is defined by the existence of an edge or a major diagonal, not a cell. More precisely:

Definition 4.2. *For each vertex* $\mathbf{x}_i$*, we say* $\mathbf{x}_j \in \rho(\mathbf{x}_i)$ *if and only if there exists an edge* $\mathbf{e} = [\mathbf{x}_i, \mathbf{x}_j]$ *or a major diagonal within an octahedral cell,* $\mathbf{m} = [\mathbf{x}_i, \mathbf{x}_j]$.

For each edge $\mathbf{e}_i = [\mathbf{x}_i, \mathbf{x}_{i+1}]$*, we say* $\mathbf{x}_j \in \rho(\mathbf{e}_i)$ *if and only if there exists an edge* $\mathbf{e} = [\mathbf{x}_k, \mathbf{x}_j]$ *or a major diagonal within an octahedral cell,* $\mathbf{m} = [\mathbf{x}_k, \mathbf{x}_j]$ *for both* $k = 1, 2$.

4.4.3.1 Vertex Rule

Each regular vertex $\mathbf{x}_i$ has the valence of 14. It is shared by 6 octahedra and 8 tetrahedra. However, because of the choice of the major diagonals that satisfy Definition 4.1 and the definition of neighbors by Definition 4.2, we have only 14 adjacent vertices to be counted. Therefore, we introduce new vertex point $\mathbf{v}_{new}$ by:

$$\mathbf{v}_{new} = \frac{1}{32}\Big\{18\mathbf{x}_i + \sum_{\mathbf{x}_j \in \rho(\mathbf{x}_i)} \mathbf{x}_j\Big\}. \tag{4.24}$$

4.4.3.2 Edge Rules

Each edge $\mathbf{e}_i = [\mathbf{x}_i, \mathbf{x}_{i+1}]$ is shared by 2 octahedra and 2 tetrahedra. According to the position of the major diagonals of the octahedra, there are two types of the regular edge rules to compute new edge point $\mathbf{e}_{new}$ (Figure 4.8). If $|\rho(\mathbf{e}_i)| = 6$ then,

$$\mathbf{e}_{new} = \frac{1}{32}\Big\{10(\mathbf{x}_i + \mathbf{x}_{i+1}) + 2\sum_{\mathbf{x}_j \in \rho(\mathbf{e}_i)} \mathbf{x}_j\Big\}. \tag{4.25}$$

If $|\rho(\mathbf{e}_i)| = 4$ then,

$$\mathbf{e}_{new} = \frac{1}{32}\Big\{8(\mathbf{x}_i + \mathbf{x}_{i+1}) + 4\sum_{\mathbf{x}_j \in \rho(\mathbf{e}_i)} \mathbf{x}_j\Big\}. \tag{4.26}$$

4.4.3.3 Cell Rule

For each octahedral cell $[\mathbf{x}_{i_0}, ..., \mathbf{x}_{i+5}]$ with the major diagonal $\mathbf{m} = [\mathbf{x}_{j_1}, \mathbf{x}_{j_2}]$, the new cell point $\mathbf{c}_{new}$ is computed by:

$$\mathbf{c}_{new} = \frac{1}{32}\Big\{4(\mathbf{x}_{i_0} + \cdots + \mathbf{x}_{i+5}) + 4(\mathbf{x}_{j_1} + \mathbf{x}_{j_2})\Big\}. \tag{4.27}$$

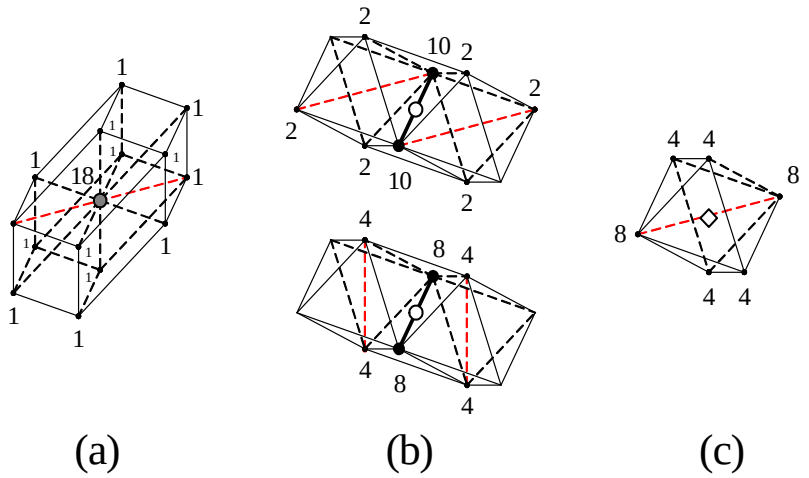


Figure 4.8: The regular subdivision rules for the box-spline based solid subdivision. The red dashed edges indicate the major diagonals.

4.5 Generalization of Solid Subdivision Scheme

From the regular subdivision rules that are defined over the structured meshes based on the octet-truss, we extend our scheme to arbitrary tetrahedral meshes. Unlike the octet-truss, where each vertex and edge have a regular number of adjacent vertices, we can have arbitrary number of vertices adjacent to vertices and edges in the tetrahedral mesh. However, once subdivided, the sub-structure inside each cell becomes regular again. In fact, the same situation happens even for subdivision surfaces. Figure 4.11(b) shows an extraordinary vertex with the valence m and its subdivision rule. In solid schemes, the extraordinary

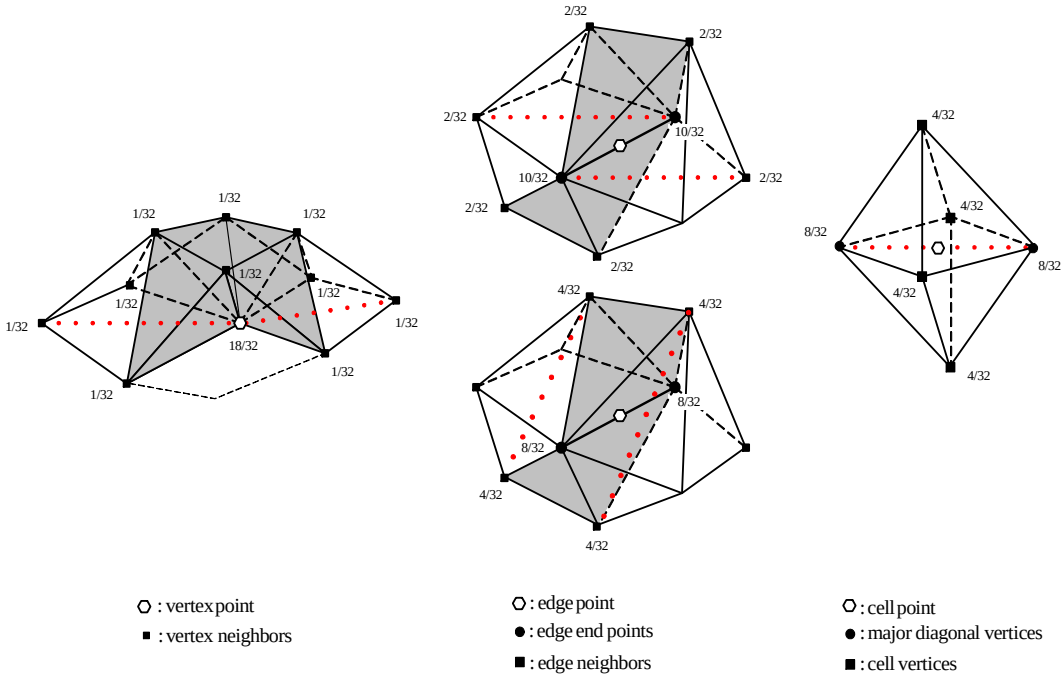


Figure 4.9: The regular subdivision rules for the box-spline based solid subdivision. The vertices are in general position. (a) shows only the upper portion of the mask. The red dotted edges indicate the major diagonals. The grey cells are tetrahedral.

cases include not one but two different types of irregularity. One is the extraordinary vertex cases, Another is the extraordinary edge case.

4.5.1 Extraordinary Subdivision Rules

The following generalization of the subdivision rules are based on spatial averaging. They require analysis to guarantee a certain level of smoothness across the extraordinary topologies.

4.5.1.1 Vertex Rule

Similar to surface schemes, an arbitrary tetrahedral mesh can contain extraordinary vertices as shown in Figure 4.10(a). Suppose the valence of the vertex $\mathbf{x}$ is m ($|\rho(\mathbf{x}_i)| = m$), then our vertex rule can be rewritten as:

$$\mathbf{v}_{new} = \frac{9}{16}\mathbf{x}_i + \frac{7}{16m}\sum_{\mathbf{x}_j \in \rho(\mathbf{x}_i)} \mathbf{x}_j. \tag{4.28}$$

4.5.1.2 Edge Rules

The extraordinary edge case is not present in subdivision surfaces. Suppose the edge $\mathbf{e} = [\mathbf{x}_i, \mathbf{x}_{i+1}]$ is surrounded by m vertices ($|\rho(\mathbf{e}_i)| = m$), then the edge rule is modified as follows:

$$\mathbf{e}_{new} = \frac{5}{16}(\mathbf{x}_i + \mathbf{x}_{i+1}) + \frac{3}{8m}\sum_{\mathbf{x}_j \in \rho(\mathbf{e}_i)} \mathbf{x}_j. \tag{4.29}$$

4.5.2 Boundary Representation

Unlike surfaces which do not have to contain the boundaries in 3D, solid objects should have boundaries in 3D space. Therefore, we need to employ special rules to represent the boundaries of our solid objects. We simply use the modified Loop's scheme (Figure 4.11) to represent boundaries. Because of the small sizes of our subdivision masks, there is no apparent trouble between the transition area of the boundary and the interior.

4.6 Analysis of Arbitrary Topology

By definition, our scheme is C^2-continuous on the octet-truss structured meshes, since it evaluates the trivariate double-directional box spline functions defined on each regular vertex. Figure 4.12 shows a single basis function evaluated by our subdivision algorithm

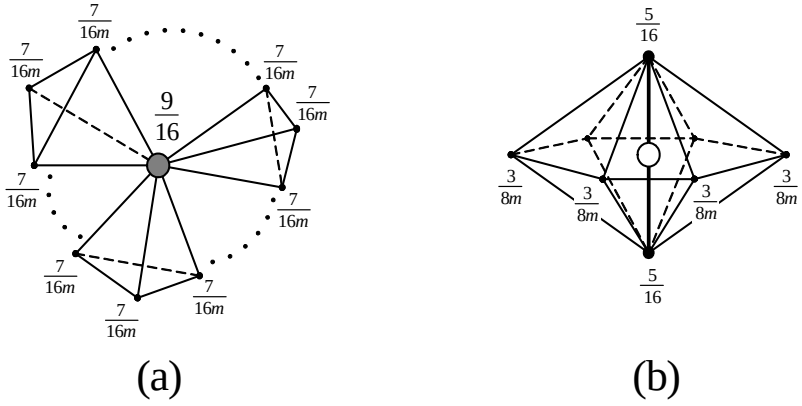


Figure 4.10: The extraordinary subdivision rules for the box-spline based solid subdivision.

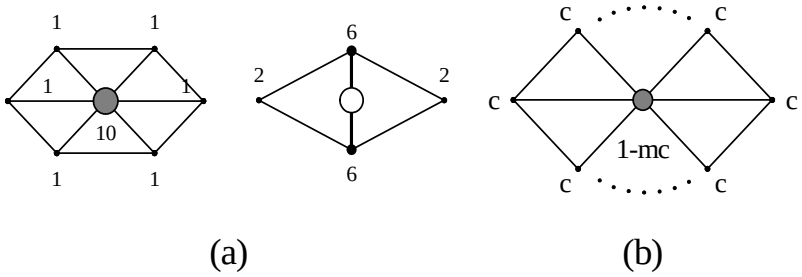
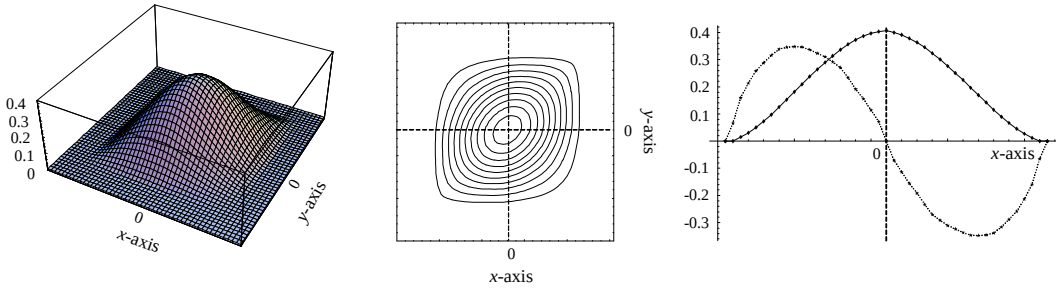


Figure 4.11: The boundary subdivision rules for the the box-spline based solid subdivision. It is basically the same rules as the Loop's scheme.

on the regular mesh. Figure 4.12(a) shows the density of the basis function cut by the x-y plane. Figure 4.12(b) is the iso-contour lines of the same function. Since we choose the direction of the line $y = x$ as our major diagonal direction, the shape is symmetry along the line. Figure 4.12(c) is the function value and the directional derivative $\frac{\partial f}{\partial x}$ along the x-axis. The plot data are from the 4th level of subdivision. The derivative is acquired by the central differences of the discrete data.

The convergence and the continuity across the extraordinary topology requires separate analysis. Many researchers have been working on the subdivision analysis near extraordinary topology, especially for the subdivision surfaces. For instance, [38, 32, 37], [64], [78, 82, 80, 79, 81], [84, 83], [108, 104, 105, 106], and most recently [55] investigated the



(a) (b) (c)

Figure 4.12: The basis function of the subdivision algorithm on the regular mesh. (a) The density values of the basis function over the x-y plane. (b) The iso-contour lines of the density. (c) The function value and its directional derivative along the x-axis. The derivative values have been re-scaled.

sufficient and necessary conditions of the convergence and the C^1-continuity for various subdivision curves and surfaces. Two major techniques of such analysis are the spectral analysis of subdivision matrices and the characteristic map method by [84]. Unfortunately, these analysis techniques for subdivision surfaces are not fully extended to the solid subdivision algorithms and there is no known method specially developed for the solid schemes. Hence, we approach this situation as follows:

(a) Categorize the extraordinary cases for the solid subdivision algorithm.

(b) For each case, compute the subdivision matrix.

(c) Perform the spectral analysis of the subdivision matrix numerically.

(d) Construct the characteristic map and confirm the satisfactory conditions for the C^1-continuity through empirical data.

Even though these steps do not guarantee the continuity of the subdivision algorithm on every possible case, they suggest a strong evidence that our subdivision algorithm is indeed C^1-continuous for many situations, especially that can be occurred in real-world.

Since our subdivision algorithm is based only on stationary linear combinations, we can describe the subdivision process of each step as a simple matrix computation:

$$\mathbf{p}^{j+1} = \mathbf{S}\mathbf{p}^{j}, \tag{4.30}$$

where $\mathbf{p}^{j} = [\mathbf{p}^{j}{}_{0}, ..., \mathbf{p}^{j}{}_{N}]^{T}$ is a matrix of control points around $\mathbf{p}^{j}{}_{0}$ at the subdivision level j. The number N of the control points is determined so that the linear system is invariant. In our case, we need 2 rings of vertex neighbors.

4.6.1 Subdivision Matrix

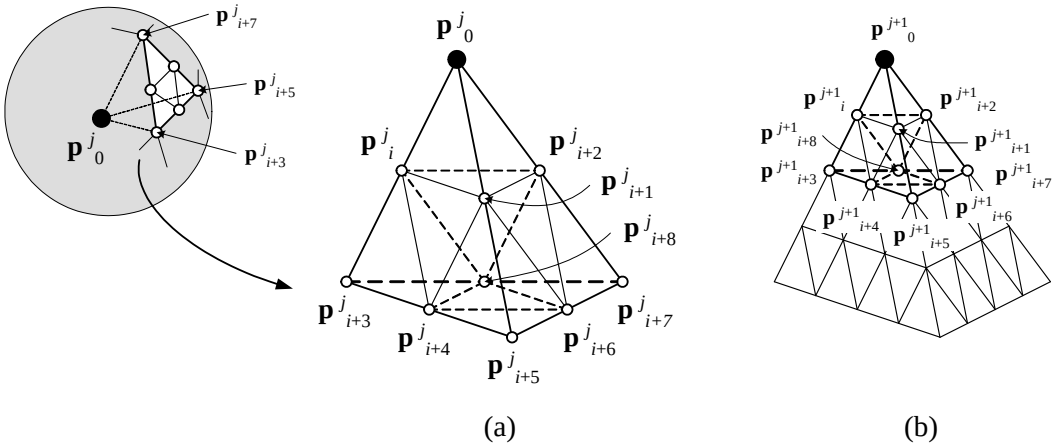


Figure 4.13: The invariant neighborhood of an extraordinary vertex and their indices.

We begin the analysis with computing the subdivision matrix for each extraordinary case. We first examine the case of extraordinary vertices. This case involves a vertex with k vertices adjacent to it. As shown in Figure 4.13, we can establish a correspondence between the k adjacent vertices and k vertices on the sphere centered by the extraordinary vertex $\mathbf{p}^{j}{}_{0}$. By considering different triangulations of the k vertices, we can understand the different configurations of the extraordinary vertex subdivision matrix. Each triangle is associated

with the tetrahedral area that is surrounding the extraordinary vertex. Because we need 2-ring vertex neighbors to acquire the invariant system, we subdivide each tetrahedron once, as illustrated in Figure 4.13. Using the Poincaré formula and the relation between triangular faces and edges:

$$\begin{aligned} v - e + f &= 2, \\ 2e &= 3f, \end{aligned}$$

we can deduce that the number of such tetrahedral area surrounding the vertex is $f = 2(k-2)$. In addition, the 1-ring vertex neighbor contains k vertices and each subdivided triangular faces on the 2-ring vertex neighbor contains 6 vertices, 3 of which are shared by each edge. Therefore, the actual number N of the vertices including the extraordinary vertex to form the invariant system is:

$$\begin{aligned} N &= 1 + k + 6f - 3e + k \\ &= 1 + k + 6(2k-4) - 3(3k-6) + k = 5k - 5. \end{aligned}$$

Hence, we can conclude that the size of the subdivision matrix for each extraordinary vertex with the valence k is $N \times N$ where $N = 5k - 5$. With a proper reordering of the indexes of the vertices, the matrix $\mathbf{S}_v$ can be written as:

$$\mathbf{S}_v = \begin{pmatrix} \mathbf{M} & \mathbf{O} \\ \mathbf{A} & \mathbf{B} \end{pmatrix},$$

where $\mathbf{M}$ is a $(k+1)\times(k+1)$ matrix and $\mathbf{O}$ is the square zero matrix with the size of $4k-6$. We use the spherical coordinates to define an order between vertices. It is important to know that the dominant and the subdominant eigenvalues of the $\mathbf{S}_v$, especially the first 5 largest

eigenvalues, are identical to those of the submatrix $\mathbf{M}$. Since the matrix $\mathbf{M}$ can be easily acquired by the k 1-ring vertex neighbors of the vertex $\mathbf{p}^j{}_0$, we can reduce the amounts of the computations during the analysis process significantly. It is worth mentioning that, unlike the surface cases, there exist several different configurations of neighboring vertices for each valence k. In fact, it is related to the planar triangulation of k points and the recent result by [86] suggests that the upper bound for the number of the configurations is $O(59^n n^{-6})$ for large n. Since each configuration yields a unique subdivision matrix, it is difficult to compute the eigensystem systematically.

The extraordinary edge with the valence k is surrounded by k tetrahedra sharing the edge $e = [\mathbf{p}^j{}_0, \mathbf{p}^j{}_2]$, as shown in Figure 4.14. Again, we subdivide each tetrahedron once to make the neighbor invariant. It is easy to deduce that the size of the subdivision matrix $\mathbf{S}_e$ is $(4k+3) \times (4k+3)$. Similar to the extraordinary vertex subdivision matrix, the matrix $\mathbf{S}_e$ can be described as:

$$\mathbf{S}_e = \begin{pmatrix} \mathbf{L} & \mathbf{O} \\ \mathbf{P} & \mathbf{Q} \end{pmatrix},$$

with the proper index reordering. In the edge case, $\mathbf{L}$ is a $(2k+3) \times (2k+3)$ matrix. It consists of the subdivision coefficients of the 1-ring neighbors of the extraordinary edge. Once more, the dominant and subdominant eigenvalues of the subdivision matrix $\mathbf{S}_e$ can be acquired from the submatrix $\mathbf{L}$. The subdivision matrix $\mathbf{S}_e$ and its eigensystem can differ by the choice of the major diagonals. It will be discussed in the next section.

4.6.2 Prerequisites

There is a question to be answered before continuing the spectral analysis. It is the question about the face-to-face case between the tetrahedral cells. Even though a tetrahedron always faces with an octahedron and vice versa in the regular octet-truss meshes, this

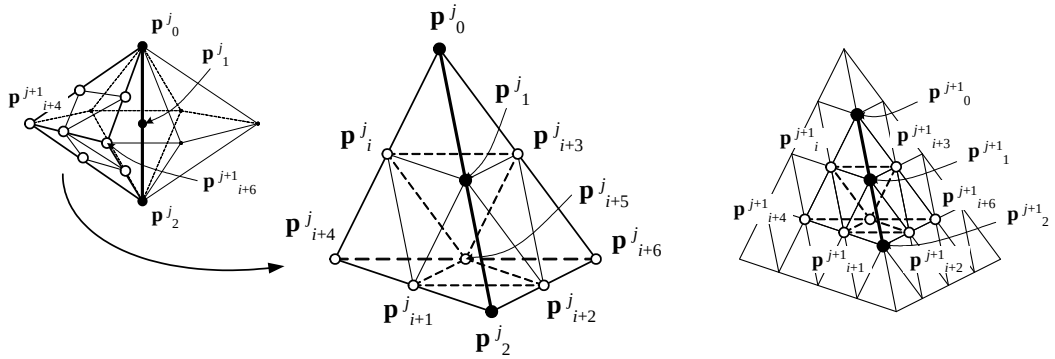
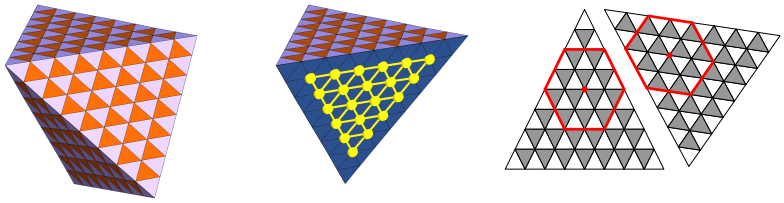


Figure 4.14: The invariant neighborhood of an extraordinary edge and their indices.

property does not hold in general. Especially, an arbitrary tetrahedral mesh does not satisfy it at all. Once subdivided, the interior of each tetrahedron becomes the regular octet-truss structure. However, the faces shared by two tetrahedra initially given by the mesh remain the same during the subdivision processes. In fact, in the paper [87], Schaefer *et al.* resolved the issue by applying the joint spectral radius test by [55]. Our situation is slightly different, since their rules are based on the cell-averaging, while ours are based on the vertex and edge-averaging. Moreover, the choice of the major diagonals plays a major role to keep the structures in the shared face regular.

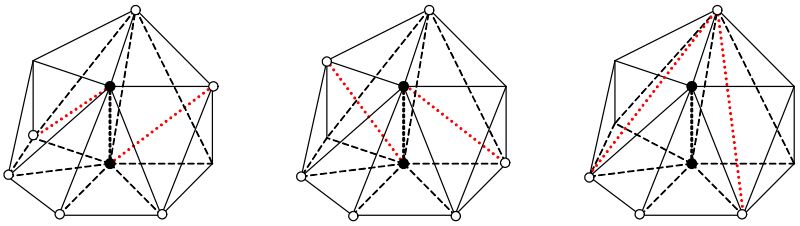
We have taken a simpler approach to guarantee the continuity of the face-to-face case. We argue that, with the proper choice of the major diagonals as defined in Definition 4.1, the vertices and edges on the shared face admit the subdivision matrices of the regular cases. We begin with two tetrahedra as shown in Figure 4.15(a). To acquire the invariant neighbors for each vertex in the shared face, we subdivide the tetrahedra 3 times. The 1-ring vertex neighbor of each yellow vertex in Figure 4.15(b) consists of 6 adjacent vertices on the face and 2 adjacent vertices in each facing tetrahedron. In addition, we can have 2 more adjacent vertices, decided by the choice of the major diagonals. Therefore, each vertex has 14 neighbors in total which form a regular vertex case. For the shared edges,

each edge on the shared face is surrounded by two adjoining octahedra and two adjoining tetrahedra. If we choose the major diagonals of the two octahedra *properly*, we can prove that each edge has the correct number of neighbors, either 6 or 4, with respect to its relative position against both major diagonals. All the possible cases are illustrated in Figure 4.16. In Figure 4.16(a) and (b), the edge has 6 neighbors, whereas it has 4 neighbors in Figure 4.16(c).



(a) (b) (c)

Figure 4.15: An example of the face-to-face case. (a) Two tetrahedra share the face. The orange-colored faces indicate the faces from octahedral cells. (b) The vertices in between. The yellow vertices has 14 neighbors with the correct choice of the major diagonals. (c) The 2-ring neighbor of the shared vertex.

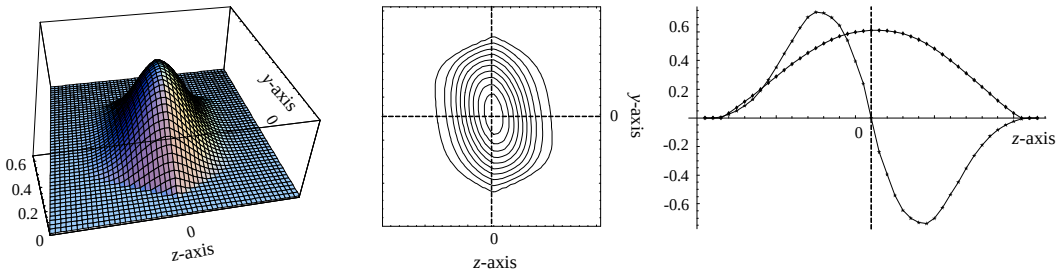


(a) (b) (c)

Figure 4.16: The different neighbors of the edge between faces by the proper choices of the major diagonals. The red dotted lines are the major diagonals.

We support our argument with some empirical results. Figure 4.17 offers a visual confirmation on the continuity between the tetrahedra. We evaluate the basis function centered on one of the shared vertices in between. The shared face of the two tetrahedra are located on the x-y plane. One tetrahedron is placed on the negative z side, while the other is placed

on the positive side. The major diagonals of the tetrahedra are chosen so that it is in the proper situation defined in Definition 4.1. The values are evaluated from the 3 levels of the subdivision algorithm. Figure 4.17 shows very smooth transition between one tetrahedron to another.



(a) (b) (c)

Figure 4.17: Evaluation of the face-to-face case. (a) The density values of the basis function centered on the shared vertex over the y-z plane. (b) The iso-contour lines of the density. (c) The function value and its directional derivative along the z-axis. The derivative values are re-scaled.

The face-to-face case entails the question of the choice of the major diagonals in arbitrary meshes. The above results suggest that the proper choice of the major diagonals are important for the continuity of the face-to-face case. It is possible that the choice cannot satisfy the properties in Definition 4.1 globally for certain meshes. Thus, we should implement the algorithm carefully so that the choice is as proper as possible at the initial stage. We discuss it in Section 4.8.

4.6.3 Spectral Analysis

Suppose the matrix $\mathbf{S}$ has the eigenvalues $\lambda_0 \geq \cdots \geq \lambda_l$ in non-increasing order with the associated eigenvectors $\mathbf{v}_0, ..., \mathbf{v}_l$. The matrix $\mathbf{S}$ is $N \times N$ matrix and each eigenvectors

are in $\mathbb{R}^N$. Now, the original control points $\mathbf{p}^0$ can be rewritten in the eigenspace:

$$\mathbf{p}^0 = \sum_i \mathbf{a}_i \mathbf{v}_i, \tag{4.31}$$

and therefore, any level j vertices $\mathbf{p}^j$ can be represented by:

$$\mathbf{p}^j = \mathbf{S}^j \mathbf{p}^0 = \sum_i (\lambda_i)^j \mathbf{a}_i \mathbf{v}_i, \tag{4.32}$$

with each coordinate coefficient matrix $\mathbf{a}_i$. The coefficient $\mathbf{a}_i$ can be computed by $\mathbf{a}_i = \mathbf{v}_i \cdot \mathbf{p}^0$. The conditions of the eigenvalues and eigenvectors for the subdivision matrix are well understood and it is independent for each coordinate operation. Therefore, we hypothesize that the similar conditions still hold for our solid scheme. The conditions are as follows:

(a) λ_0 should be equal to 1 for the subdivision to be invariant with respect to translations and rotations.

(b) λ_i should be strictly less than 1 when $j > 0$ for the convergence of the scheme. The limit positions for the original control points is then

$$\mathbf{p}^0 = \lim_{j\to\infty} \mathbf{p}^j = \sum_i \big(\lim_j (\lambda_i)^j \mathbf{a}_i \mathbf{v}_i \big) = \mathbf{a}_0 \mathbf{v}_0. \tag{4.33}$$

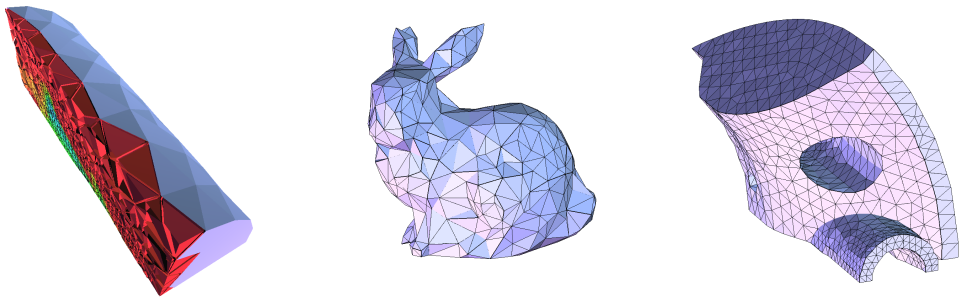
(c) λ_4 should be strictly less than λ_3. Suppose the subdominant eigenvalues $\lambda_1 = \lambda_2 = \lambda_3 = \lambda > \lambda_4$ and the vertex ${\mathbf{p}^0}_0$ is the origin. Then we get:

$$\frac{\mathbf{p}^j}{\lambda^j} = \mathbf{a}_1\mathbf{v}_1 + \mathbf{a}_2\mathbf{v}_2 + \mathbf{a}_3\mathbf{v}_3 + \mathbf{a}_4\Big(\frac{\lambda_4}{\lambda}\Big)^j + \cdots, \tag{4.34}$$

which means the control points are approaching a fixed configuration up to a scaling

factor λ^j. The remainders converge to zero in the limit since $\lambda > \lambda_j$ for $j \geq 4$. From this observation, we can derive that the three subdominant eigenvalues determine the behavior of the derivatives at ${\mathbf{p}^0}_0$ in the limit. This is a sufficient condition to define the characteristic map.

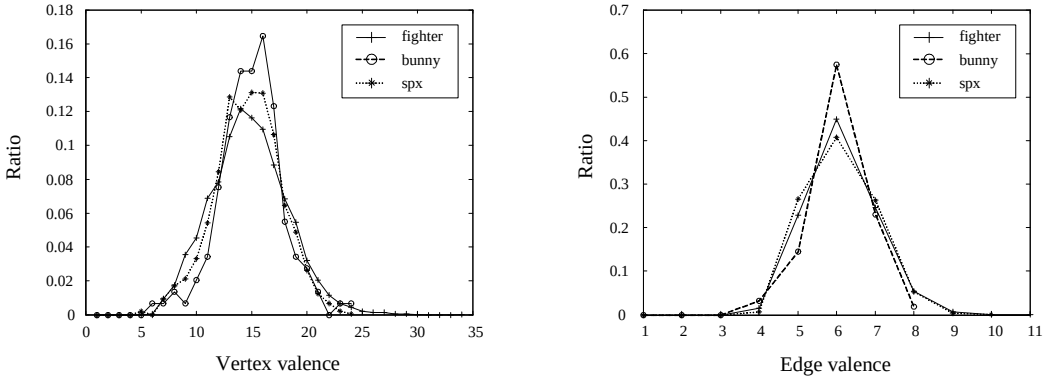
To verify these conditions, eigenvalues and eigenvectors of each subdivision matrix should be computed. As first observed in [31], the subdivision matrix for an extraordinary vertex in surfaces cases has a cyclic structure due to its planar symmetry. Therefore, we can apply discrete Fourier transform to compute the eigenstructure of every valence systematically. Unfortunately, this is no longer true for the solid cases, as discussed in the previous section. Hence, we need to rely on numerical computation of the eigenstructure for specific cases. Since we cannot compute all the possible cases, we begin with statistics on vertex and edge valences of arbitrary tetrahedral meshes to determine which valence numbers should be included in our analysis. We choose few existing tetrahedral models (Figure 4.18) to examine the valence numbers for each vertex and edge. The meshes are acquired by various tessellation methods including the 3D Delaunay triangulation and the advance front technique.



(a) (b) (c)

Figure 4.18: A selection of arbitrary tetrahedral meshes. (a) A cross-section of a fighter model with its environment (fighter). (b) Tesselated Stanford bunny (bunny). (c) A model of a mechanical part (spx)

As Figure 4.19 Table 4.18 show, the distribution of the valence numbers are concentrated on the regular valence numbers. Moreover, the averages are very close to the regular valence numbers and the deviations are relatively small. These results suggest that in real world application, we only need to analyze relatively small number of valence cases. In this paper, we choose to analyze the valence number from 5 to 22 for the extraordinary vertices and the valence number from 4 to 9 for the extraordinary edges.



(a) (b)

Figure 4.19: Histogram of the valence numbers of the selected arbitrary meshes. (a) Vertex valence. (b) Edge valence.

Model name	No. vertices	No. edges	No. cells	V−val average	V−val maximum	E−val average	E−val maximum
fighter	13832	87587	70125	13.75	33	5.12	10
bunny	575	2904	1903	13.95	23	5.06	7
spx	2896	17212	12936	13.65	23	5.09	9

Table 4.1: Statistics on the valence numbers of the selected arbitrary meshes

Table 4.2 presents the list of the first 6 eigenvalues of the selected extraordinary vertex cases. As we have mentioned, there exist several different configurations for each valence. We only show few cases in the table. Table 4.3 shows the list of the first 6 eigenvalues of the selected extraordinary edge cases. Unlike the vertex cases, each valence has unique configuration. However, it differs by the choice of the major diagonals. Figure 4.20 shows

two different choices of the major diagonals for the extraordinary edge with the valence 9. In the first row, the diagonals are chosen such that they are in skew positions mutually. In the second row, the diagonals point toward the center of the extraordinary edge. As Figure 4.20(b) and (c) suggest, the first choice yields much smoother results. The eigenvalues of the first case are shown in Table 4.3, whereas the 6 dominant eigenvalues of the second case are:

$$\{1., 0.619939, 0.619939, 0.528431, 0.528431, 0.5\}.$$

The difference between two cases becomes more apparent as the valence is getting higher. In fact, the characteristic map of the second case fails to confirm C^1 continuity for the valence larger than 8, as discussed in the next section. The values in Table 4.3 are taken from the configurations similar to the skew case for each valence. During the implementation of the subdivision algorithm, we try to optimize the choice of the major diagonals so that it produces the similar result as the first row for the most of the extraordinary edges. In Appendix B, we present the subdivision matrices and their eigenvalues for the lowest valence cases in the tables. Two different choices of the major diagonals and their subdivision matrices are shown in Section B.2.

It is clear from the computed values that the eigenvalues of the most of the extraordinary cases satisfy the eigenvalues conditions for subdivision algorithms. However, the results so far only guarantee the convergence of the subdivision algorithm at the particular vertex or edge. In the next step, we examine the characteristic map of each case to verify C^1 continuity near the extraordinary vertex or edge.

4.6.4 Characteristic Map

If the eigenvalues satisfy $\lambda_0 = 1 \gneq \lambda_1 \geq \lambda_2 \geq \lambda_3 \gneq \lambda_4$, we are able to apply the characteristic map method by [84] to prove C^1 continuity. From the real eigenvectors

Valence	λ_0	λ_1	λ_2	λ_3	λ_4	λ_5
5	1.	0.3125	0.292083	0.15	0.125	0.125
6	1.	0.312499	0.25	0.25	0.25	0.15
7	1.	0.327254	0.327254	0.3125	0.275888	0.15
8(a)	1.	0.480205	0.3125	0.3125	0.249998	0.2375
8(b)	1.	0.375	0.375	0.3125	0.270178	0.15
9	1.	0.405872	0.405872	0.3125	0.26545	0.19437
10(a)	1.	0.477404	0.418566	0.418566	0.2375	0.206434
10(b)	1.	0.426777	0.426777	0.3125	0.261451	0.25
11	1.	0.441511	0.441511	0.3125	0.293412	0.293412
12	1.	0.480205	0.480205	0.480205	0.250002	0.2375
13	1.	0.460313	0.460313	0.353854	0.353854	0.3125
14(a)	1.	0.577132	0.449431	0.449431	0.34832	0.3125
14(b)	1.	0.517404	0.517404	0.480205	0.3125	0.3125
15	1.	0.471364	0.471364	0.392016	0.392016	0.3125
16	1.	0.541169	0.541169	0.480204	0.372645	0.372645
17	1.	0.571212	0.511703	0.511703	0.371472	0.358853
18(a)	1.	0.623289	0.463128	0.463128	0.457191	0.374739
18(b)	1.	0.557148	0.557148	0.480205	0.418566	0.418566
20(a)	1.	0.571212	0.549072	0.549072	0.3875	0.3875
20(b)	1.	0.568361	0.568361	0.480206	0.453454	0.453454
22	1.	0.616629	0.525774	0.525774	0.4625	0.427853
22(b)	1.	0.576511	0.576511	0.480205	0.480205	0.480205

Table 4.2: Eigenvalues for a selection of the extraordinary vertex cases.

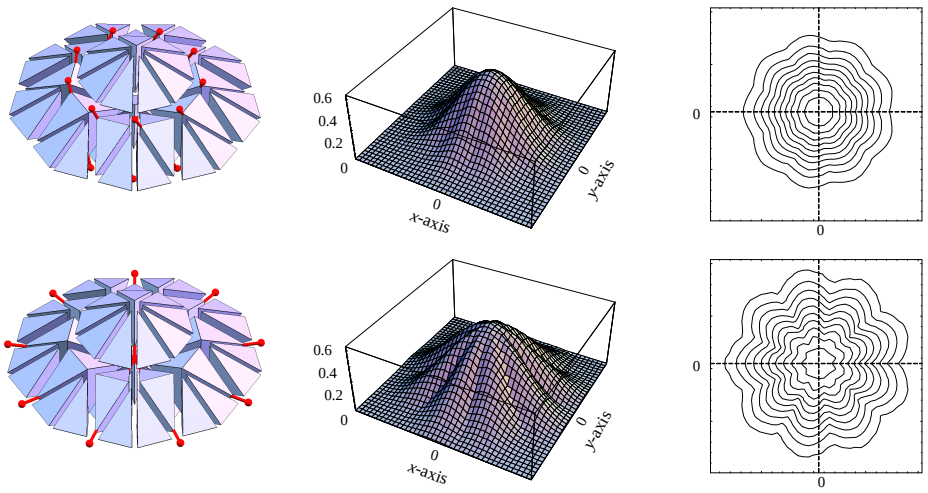
Valence	λ_0	λ_1	λ_2	λ_3	λ_4	λ_5
4	1.	0.477404	0.418566	0.418566	0.2375	0.206434
5	1.	0.480205	0.480205	0.480205	0.25	0.2375
6	1.	0.517404	0.517404	0.480205	0.3125	0.3125
7	1.	0.541169	0.541169	0.480204	0.372645	0.372645
8	1.	0.557148	0.557148	0.480205	0.418566	0.418566
9	1.	0.568361	0.568361	0.480206	0.453454	0.453454

Table 4.3: Eigenvalues for a selection of the extraordinary edge cases.

$\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ associated with the first 3 subdominant eigenvalues, we define a map:

$$\Psi = N[\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3] : U \times \mathbb{Z}_k \longrightarrow \mathbb{R}^3, \tag{4.35}$$

Where U is a unit simplex in 3D pivoted on the origin with the corner containing the origin has been removed at the half points of the edges. Figure 4.21 illustrates an example of the control net of the characteristic map and its cross-section. Note that the characteristic map by Reif *et al.*is originally defined on 2D space. We generalize the map to the solid case and



(a) (b) (c)

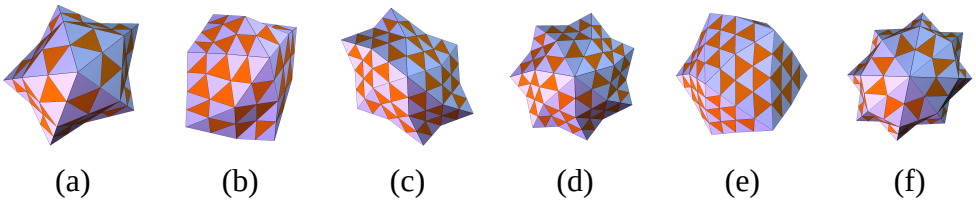
Figure 4.20: Two different choices of the major diagonals for the extraordinary edge with the valence 9. (a) The different choices of the major diagonals indicated in the red lines. (b) The density values. (c) The Iso-contour lines.

assume that it is correct without any proof. Now, we presume that if we can prove that the characteristic map Ψ is regular and injective, then our subdivision algorithm satisfies C^1 continuous. For 2D, the regularity and injectivity can be proven by considering the complex plane. However, due to lack of any planar symmetry in general, the direct analogy of the method has been proven to be difficult in 3D. Instead, we rely on the experimental results to conform its regularity and injectivity. For the selected cases, we perform the subdivision and prolongation [84] successively, up to certain levels, followed by the visual inspection of the results. In most cases, it has been relatively easy to deduce that the process will not produce any irregularity or self-intersections. In contrast, Figure 4.24 shows the characteristic maps for the valence 9 and 11 cases with specific choice of the major diagonals. In these cases, the diagonals are chosen to point toward the center, as shown in the second case of Figure 4.20. During the subdivision and prolongation process, the maps form a single saddle surface and therefore they are not injective. In these two cases, we cannot

determine the C^1 continuous of the subdivision algorithm near the extraordinary edges by the characteristic map method.

(a) (b)

Figure 4.21: The control net of the characteristic map of the extraordinary vertex with the valence 11. (a) The control net. (b) The cross-section of the control net.



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

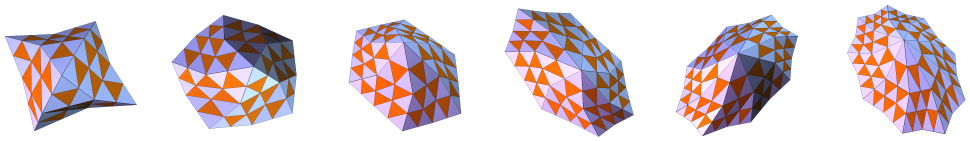
(g) (h) (i) (j) (k) (l)

(m) (n) (o) (p) (q) (r)

Figure 4.22: Control nets for a selection of the characteristic maps of the extraordinary vertex with the valences from 7 to 22. The orange-colored faces indicate the faces from octahedral cells.

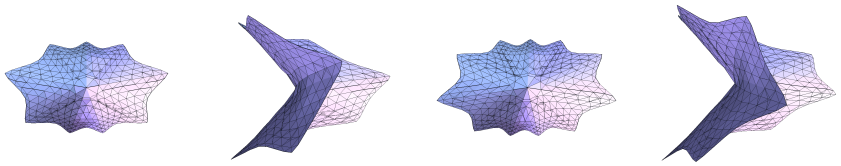
4.7 Averaged Subdivision Scheme

As proven in the previous sections, the choice of the major diagonal plays major role over both the regular structured meshes and the analysis over arbitrary tetrahedral meshes.



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

Figure 4.23: Control nets for a selection of the characteristic maps of the extraordinary edges with the valences from 4 to 9. The orange-colored faces indicate the faces from octahedral cells.



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

Figure 4.24: The characteristic maps for the extraordinary edges with the valences 9 and 11. The major diagonals are chosen to point toward the center. The maps are not injective. (a-b) The valence 9 case. (c-d) The valence 11 case.

However, there are some disadvantage involving the diagonals. First, for an arbitrary mesh, MAJOR-DIAGONAL-CHOOSE (Algorithm 4.2) does not return proper choice of the major diagonals *globally*, in general. Secondly, it requires meticulous bookkeeping on the entire meshes and across the subdivision levels. Finally, it generates some "favored" direction in the represented object, which can cause asymmetry. It is obvious because the basis function, or the box spline, produced by the subdivision algorithm on the structured mesh is *not* radially symmetric. This fact could cause some problem during heterogeneous material modeling, if the mesh and the diagonals are not carefully chosen.

In fact, there is a simple solution to avoid the major diagonals by averaging the basis function of each direction. In $\mathbb{Z}^m$ with the octet-truss structure, there are 3 choices in the directions of the major diagonals, *i.e.*$(1,1,0)$, $(-1,1,0)$, and $(0,0,1)$. Suppose we describe our object S_Ξ associated the particular major diagonal direction defined by the Ξ as:

$$S_\Xi(\mathbf{x}) = \sum_{\mathbf{i}\in\mathbb{Z}^m} \mathbf{p}_\mathbf{i} M_\Xi(\mathbf{x}-\mathbf{i}), \tag{4.36}$$

where $\mathbf{p_i}$ are initial control points. Let us denote the relevant directional matrices as Ξ_1, Ξ_2, and Ξ_3, respectively. Then, we can define the new object S without any major diagonals:

$$S(\mathbf{x}) = \frac{1}{3}\sum_{j=1}^{3} S_{\Xi_j}(\mathbf{x}). \tag{4.37}$$

It is easy to find the corresponding subdivision rules. One can simply alternate the major diagonals in the regular rules (Figure 4.8), sum the weights up at each vertex and divide it by 3. The computed values are shown in Figure 4.25. We call the scheme the *averaged solid subdivision scheme*. Note that the face-to-face case becomes irregular with these averaged masks and the argument in Section 4.6.2 becomes invalid. The extraordinary analysis of this particular scheme requires a new mathematical tool, which has not been exploited yet.

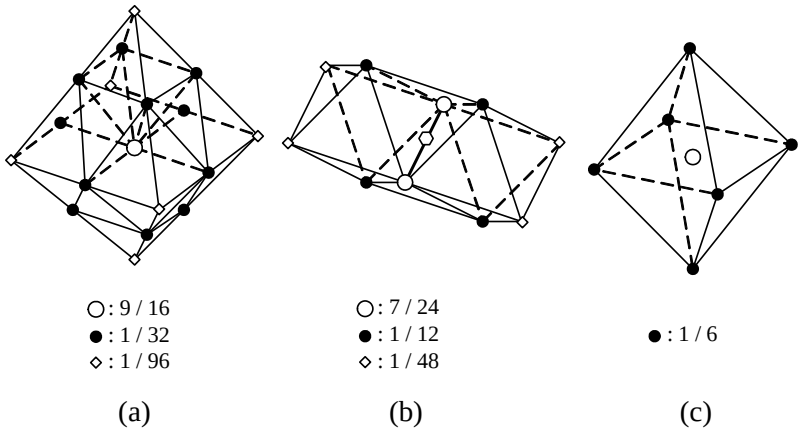


Figure 4.25: The regular rules for the averaged subdivision scheme.

4.8 Implementation

The implementation of the subdivision algorithm is straightforward, except the maintenance of the major diagonals. As outlined in Algorithm 4.1, the subdivision object is represented by the vertices, edges and cells. The edges provide the connectivity information between the vertices. The major diagonal information is stored with each cell. For tetrahedral cells, the major diagonal is represented by a pair of non-adjacent edges. For

octahedral cells, the major diagonal is represented by a pair of non-adjacent vertices. Since there are only 3 choices for both cases, only 2 bits of additional memory is required for each cell. For each vertex and edge, we compute new vertex and edge point by the subdivision rules, using their neighbors. Then, we split each cell into subcells using the new vertices. During the split, if the cell is octahedral, we need to compute the cell point. Each subcell inherits the information on the major diagonal from its parent cell.

```
1: OCTET-SUBDIVISION ({v_i}, {e_j}, {c_k})
   {Input {v_i}, {e_j}, {c_k}: A set of vertices, edges, and cells, respectively}
2: if this is the first subdivision then
3:    call MAJOR-DIAGONAL-CHOOSE ({c_k})
4: end if
5: for all vertex v_i do
6:    compute a new vertex point p_i
7: end for
8: for all edge e_j do
9:    compute a new edge point q_j
10: end for
11: for all cell c_k do
12:    if c_k is tetrahedral then
13:       split c_k into 5 subcells c^0_k, ..., c^4_k
14:    else if c_k is octahedral then
15:       compute a new cell point r_k
16:       split c_k into 14 subcells c^0_k, ..., c^13_k
17:    end if
18:    inherit the major diagonal choice from c_k
19:    construct new edges {e'_l} from {p_i} ∪ {q_j} ∪ {r_k}
20: end for
21: return {p_i} ∪ {q_j} ∪ {r_k}, {e'_l}, {c^m_k}
```

Algorithm 4.1: OCTET-SUBDIVISION.

If the subdivision is performed for the first time with an arbitrary tetrahedral mesh, there is no given major diagonal information. In this case, we use a function MAJOR-DIAGONAL-CHOOSE (Algorithm 4.2) to choose the major diagonal for each cell properly. The algorithm is based on the breath-first search of the adjacency tree of the cells. During

the search, it tries to minimize the conflict of the major diagonal choice between adjacent cells, so that the choice is as proper as possible globally.

```
1: MAJOR-DIAGONAL-CHOOSE ({c_i})
   {Input {c_i}: A set of tetrahedral cells
      c_i.visited; true if the cell is visited during the search.
      c_i.edges; A set of edges of the cell.
      c_i.major-diag; A major diagonal of the cell
      (c_i.major-diag ⊂ c_i.edges, |c_i.major-diag| = 2). }
2: initialize a queue q and a set s
3: q.push(c_0)
4: while q ≠ ∅ do
5:    c ← q.pop()
6:    s ← ∅
7:    for each adjacent cell c' of c do
8:       if c'.visited is true then
9:          s ← s ∪ c'.major-diag
10:      else
11:         q.push(c')
12:      end if
13:   end for
14:   if |c.edges−s| > 1 then
15:      choose c.major-diag from c.edges−s
16:   else
17:      choose c.major-diag arbitrary
18:   end if
19:   c.visited ← true
20: end while
21: return
```

Algorithm 4.2: MAJOR-DIAGONAL-CHOOSE.

4.9 Experimental Results

A major benefit of our solid subdivision scheme is the ability to represent complex solid models with heterogeneous materials. Figure 4.27 shows a selection of subdivision models designed by a simple modeling tool that we developed for the subdivision scheme. Models with complex topology can be easily represented with the unified scheme. Not

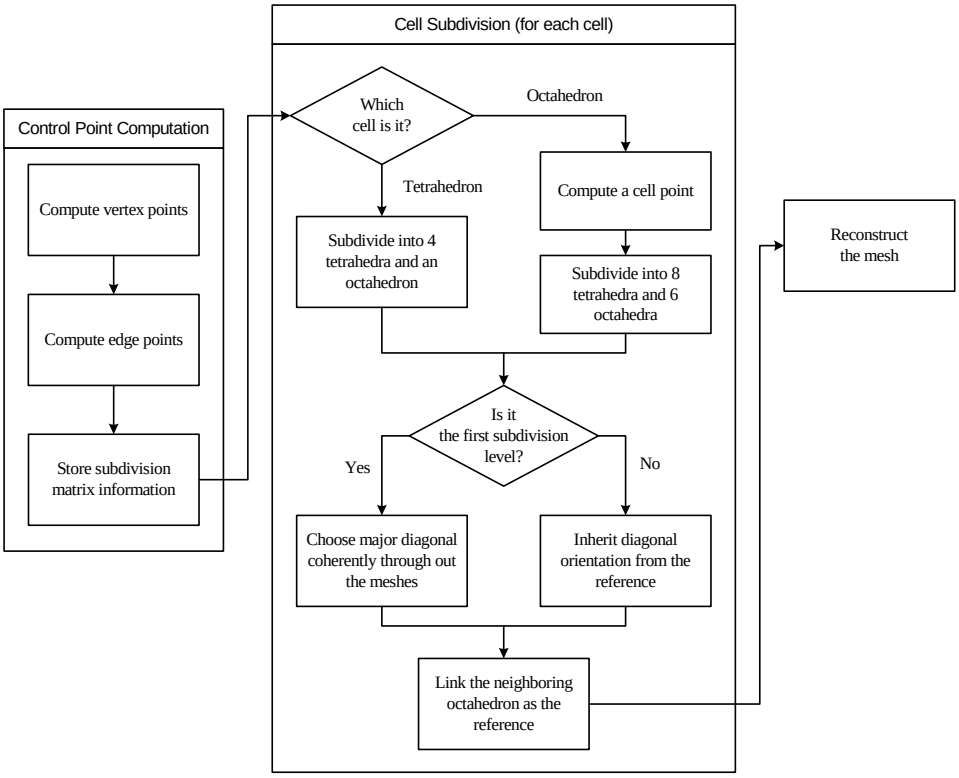


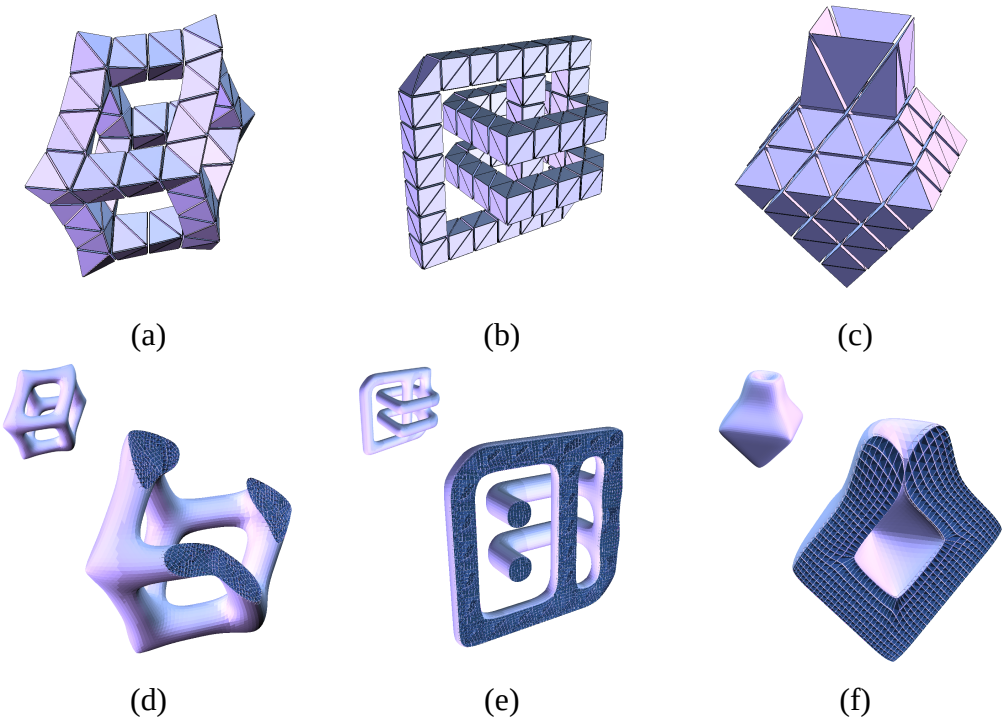
Figure 4.26: The flow chart of our subdivision scheme implementation.

only the boundaries, but also the internal structures are smooth and well-defined. Figure 4.27(f) illustrates a model with non-trivial topology, which cannot be represented by surface subdivision schemes without serious modification. Our algorithm can handle it with no exceptional rule, since we make no special assumption on the connectivity of 3D meshes.

Figure 4.28 shows a torus model with hybrid dimensionality. It consists of solid and surface parts. With some additional rules, our scheme can be used for non-manifold object representations. Extensive research on non-manifold representation using the solid subdivision scheme is discussed in [20].

Figure 4.29 and Figure 4.30 present volumetric models with heterogeneous material properties associated with geometry. In Figure 4.29, the cylinder model has different material densities at the top and the bottom of the object. By utilizing our algorithm, we can blend the different densities smoothly. As Figure 4.29(c) shows, the result is much

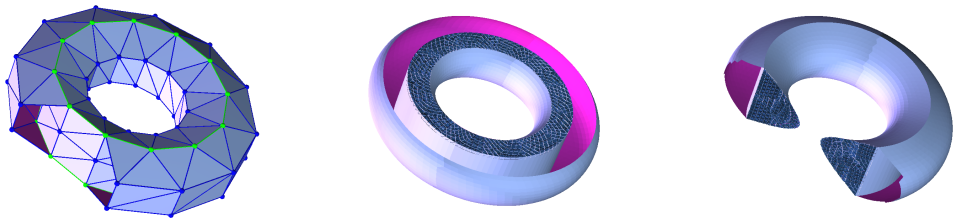
smoother than simple linear interpolation. In Figure 4.30, we assign tensions at each vertex on the coarsest level using simple Laplace's equation with initial condition. Each time step, instead of solving the equation on the fine levels, we simply apply our subdivision rules to interpolate the values using the coarsest level as initial values. The results are shown in Figure 4.30(b) and (c). Even though the potential has not been fully investigated, we believe that our subdivision scheme can serve as better blending functions or prolongation operators for finite element analysis or multigrid method based on tetrahedral meshes, instead of trivariate linear interpolation.



(a) (b) (c)

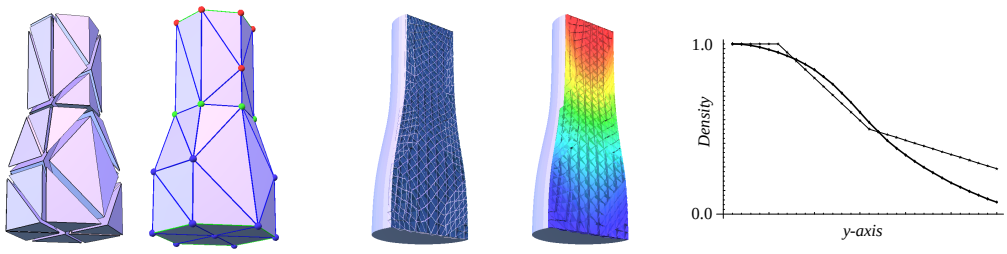
(d) (e) (f)

Figure 4.27: Solid subdivision models with non-trivial topology. (a-c) Initial control meshes. (d-f) The models at subdivision level 3 and their cross-sections.



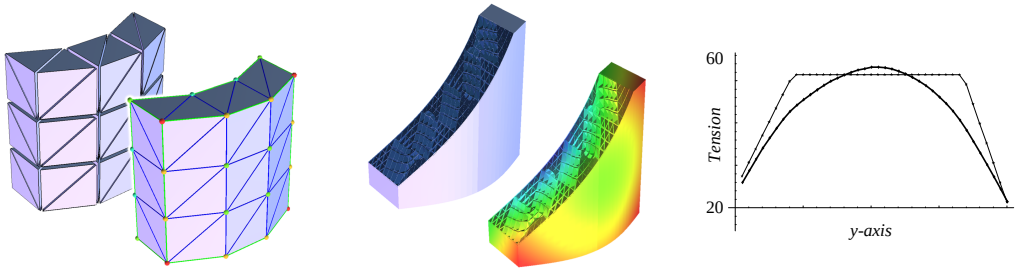
(a) (b) (c)

Figure 4.28: A torus model that consists of a solid and a surface. (a) Initial control mesh. The cut shows the internal structure. Purple areas are the backside of a surface. (b) A cross-section of the model at subdivision level 3. (c) Another cross-section of the model at subdivision level 3.



(a) (b) (c)

Figure 4.29: A cylindrical model with heterogeneous material. (a) Initial control mesh and assigned material density (color-coded). (b) A cross-section of the model at subdivision level 3 and its density distribution. (c) Comparison of density distribution by our subdivision algorithm (bold line) and tri-linear interpolation (thin line).



(a) (b) (c)

Figure 4.30: A panel model with simulated tension force. (a) Initial control mesh and assigned tension (color-coded). (b) A cross-section of the model at subdivision level 3 and its tension interpolation. (c) Comparison of tension interpolation by our subdivision algorithm (bold line) and tri-linear interpolation (thin line).

Chapter 5

Interpolatory Solid Subdivision Scheme over Simplicial Complexes

Interpolatory subdivision schemes have very desirable properties that approximate schemes cannot satisfy. In this chapter, we propose a novel interpolatory solid subdivision scheme defined over arbitrary tetrahedral meshes. We derive the rules using a weighted perturbation and prove its smoothness using the generating function method for regular cases.

The interpolatory solid subdivision scheme over simplicial complexes was introduced in a paper presented in the 2003 International Conference on Shape Modeling and Applications [17]. A paper with the detailed analysis is in preparation.

5.1 Introduction and Motivation

Interpolatory subdivision schemes interpolate initial control point sets. In each step, once new vertices are introduced, their geometric positions and associated properties are invariant. Because of this, an interpolatory scheme has unique advantages over an approximate scheme, such as:

- It is easy to enforce constraints during physics simulation.
- It supports intuitive, direct manipulation of control points.

- It has no need for an auxiliary subdivision matrix for vertex points.
- It has no need for subdivision matrix inversion during data fitting applications.

Therefore it would be more suitable for practical applications such as physics simulation, finite element method (FEM), interactive design and data fitting. Our motivation is to develop a solid subdivision scheme that can interpolate control points scattered in 3D space. However, due to the complexity and ambiguity associated with interpolatory schemes, almost no attempt has been made to develop an interpolatory solid subdivision scheme.

We propose a novel interpolatory solid subdivision scheme, specifically over arbitrary tetrahedral meshes. Based on the 3D analogue of the Butterfly scheme [36], we derive the subdivision rules using a weighted perturbation of linear interpolation. Using mathematical technique developed by Dyn *et al.*[35, 36, 38, 32, 37, 33], we prove the C^1 smoothness of our scheme in regular cases. Like the box-spline based solid scheme, we utilize the octet-truss as the regular structure, and expand our rules to cover arbitrary cases.

5.2 Derivation of Solid Subdivision Scheme

We employ the generating functions of subdivision process to derive the interpolatory scheme, which are already discussed in Chapter 4. Also, since our goal is to provide the subdivision scheme over simplicial complexes, we assume the same octet-truss structured meshes as our regular meshes.

We consider the subdivision process as a linear interpolation with weighted perturbation. The simplest form of interpolatory schemes is a linear interpolation. For any vertex $\mathbf{x}$ and edge $\mathbf{e} = [\mathbf{x}_0, \mathbf{x}_1]$, the interpolation can be simply described as:

$$\mathbf{x}_{new} = \mathbf{x}, \tag{5.1}$$

$$\mathbf{e}_{new} = \frac{1}{2}(\mathbf{x}_0 + \mathbf{x}_1). \tag{5.2}$$

Now, for the new edge point, we introduce perturbation using information from its neighbors:

$$\mathbf{e}_{new} = \frac{1}{2}(\mathbf{x}_0 + \mathbf{x}_1) + f(w, \mathbf{q}) + g(w, \mathbf{r}). \tag{5.3}$$

Here f and g are perturbation function with weight w, $\mathbf{q} = [\mathbf{q}_0, ..., \mathbf{q}_{N-1}]$ are one-neighbor vertices, *i.e.*each $\mathbf{q}_i \in \rho(\mathbf{e})$, and $\mathbf{r} = [\mathbf{r}_0, ..., \mathbf{r}_{M-1}]$ are two-neighbor vertices, *i.e.*each $\mathbf{r}_j \in \rho^2(\mathbf{e})$. We define $\rho^2(\cdot)$ by:

Definition 5.1. *For an edge* $\mathbf{e}$*, two-neighbor* $\rho^2(\mathbf{e})$ *is defined by*

$$\rho^2(\mathbf{e}) = \{\mathbf{p} \mid \mathbf{p} \in \rho(\mathbf{e}) \; or \; \mathbf{p} \in \rho(\mathbf{p}') \; for \; some \; \mathbf{p}' \in \rho(\mathbf{e})\}. \tag{5.4}$$

We choose a linear function f and g for our scheme. The convergence and the continuity of the scheme depend on the choice of the weight w, which will be decided in the next section.

5.2.0.1 Vertex Rule

Since our scheme is interpolatory, it is obvious that a vertex point is geometrically invariant in each level. Therefore, if $\mathbf{x}$ is a vertex in the current level, a vertex point $\mathbf{x}_{new}$ is simply assigned by

$$\mathbf{x}_{new} = \mathbf{x}. \tag{5.5}$$

5.2.0.2 Edge Rules

For each edge $\mathbf{e} = [\mathbf{x}_0, \mathbf{x}_1]$, a new edge point $\mathbf{e}_{new}$ can be written as three parts of linear combinations, or as a weighted average of both edge end points, one, and two-neighbors. In the regular case (Figure 5.2(a) and Figure 5.3(a)), the number of one- and two-neighbors are both 6. Therefore, we can express

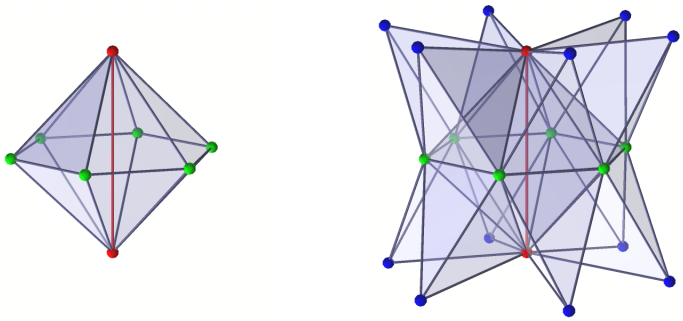
$$\mathbf{e}_{new} = \frac{1}{2}(\mathbf{x}_0 + \mathbf{x}_1) + w\sum_{i=0}^{5}\mathbf{q}_i - w\sum_{j=0}^{5}\mathbf{r}_j, \tag{5.6}$$

where $\mathbf{q}_i$ and $\mathbf{r}_i$ are one and two-neighbors, respectively.

In a more general case (Figure 5.1), we average them by the number of their neighbors, *i.e.*,

$$\mathbf{e}_{new} = \frac{1}{2}(\mathbf{x}_0 + \mathbf{x}_1) + \frac{M}{N} w \sum_{i=0}^{N-1} \mathbf{q}_i - w \sum_{j=0}^{M-1} \mathbf{r}_j, \tag{5.7}$$

where N and M are the numbers of one-neighbors and two-neighbors, respectively. It should be noted that Equation (5.7) only ensures convergence around irregular vertices. There is a slight, however noticeable, degeneracy when a vertex has valence of 4, which rarely occurs in a real-world model.



(a) (b)

Figure 5.1: Edge neighbors in general cases. (a) Green vertices denote one-neighbors. The edge is colored in red. (b) Blue vertices denote two-neighbors. They consist of the vertices from adjacent cells of one-neighbor cells.

5.2.0.3 Cell Rules

A cell point occurs during a split of an octahedron. Since we maintain the major diagonal $\mathbf{m} = [\mathbf{x}_0, \mathbf{x}_1]$, the rule can be considered as an edge rule applied on the major diagonal shown in Figure 5.2(b). In this situation, we can express a new cell point $\mathbf{c}_{new}$ as:

$$\mathbf{c}_{new} = \frac{1}{2}(\mathbf{x}_0 + \mathbf{x}_1) + w \sum_{i=0}^{5} \mathbf{q}_i - w \sum_{j=0}^{5} \mathbf{r}_j. \tag{5.8}$$

Because of the size of the edge mask, it is relatively difficult to apply it directly on the mesh, especially when an extraordinary case occurs. We devise a modified mask (Figure 5.3(b)) which is easier to apply in general. Then the new cell point rule for the regular case

becomes:

$$\mathbf{c}_{new} = \frac{1}{2}(\mathbf{x}_0 + \mathbf{x}_1) + 2w \sum_{i=0}^{3} \mathbf{q}_i - w \sum_{j=0}^{7} \mathbf{r}_j, \tag{5.9}$$

where $\mathbf{q}_i$ is a vertex from the octahedron except the major diagonal vertices and $\mathbf{r}_i$ is a vertex from the tetrahedra around the tetrahedron.

The choice of the weight w is subject to the convergence and smoothness analysis of the scheme on the regular meshes. We have proved that the scheme is convergent when $w < \frac{1}{8}$ and it is C^1 smooth when $w < \frac{1}{16}$. It is discussed in the next section.

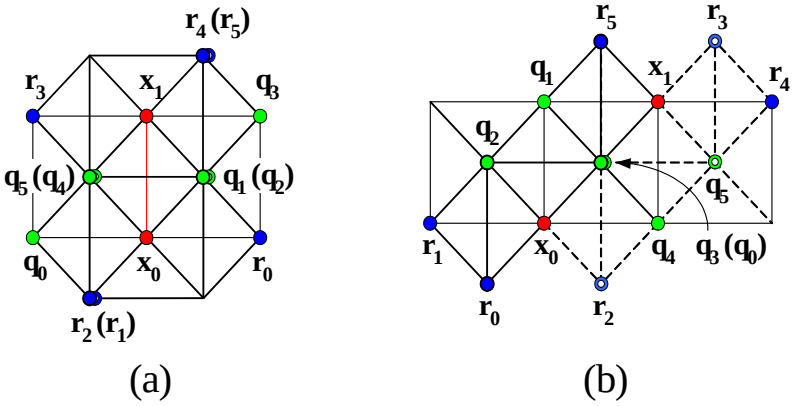


Figure 5.2: (a) The top-view of the one-neighbor (green) and two-neighbor (blue) of the regular edge case (red) (b) The top-view of the one-neighbor (green) and two-neighbor (blue) of the cell case (the major diagonal in red).

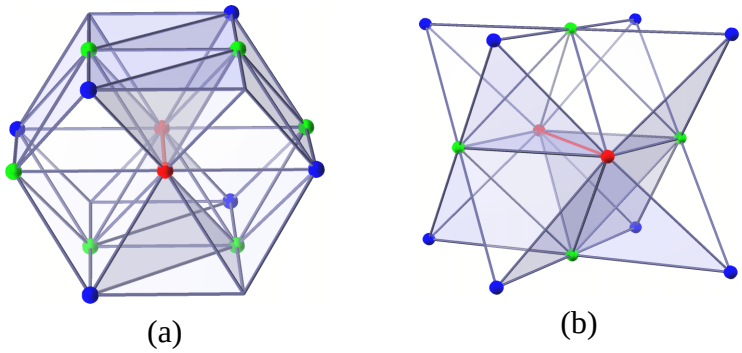


Figure 5.3: (a) The one-neighbor (green) and two-neighbor (blue) of the regular edge case (red). (b) The one-neighbor (green) and two-neighbor (blue) of the cell case (the major diagonal in red).

5.3 Analysis of Subdivision Scheme

5.3.1 Regular Cases

The subdivision scheme can be expressed in matrix form as $\mathbf{p}^{k+1} = \mathbf{S}\mathbf{p}^k$, where $\mathbf{p}^k$ is the vector of points at subdivision level k, $\mathbf{S}$ is the local invariant subdivision mask, and $\mathbf{p}^{k+1}$ is the resulting vector of new points. It is relatively straightforward to confirm that the scheme is convergent by spectral analysis of the subdivision matrix $\mathbf{S}$. In particular, the subdominant eigenvalue of $\mathbf{S}$ is strictly less than 1 for $w < \frac{1}{8}$ which is sufficient to show the convergence of our interpolatory scheme.

We will use techniques explained by Dyn *et al.* [38] to prove that our subdivision scheme is C^1 continuous over the octet-truss structured meshes. Since our subdivision scheme has no closed-form expression for its basis functions, we cannot simply extract the basis functions and examine them analytically. Therefore, we rely on analysis of subdivision matrices and characteristic functions to study the scheme's convergence and continuity properties. By showing that the characteristic polynomials of the subdivision process have certain properties, we will demonstrate that the algorithm generates volumes that are C^1 in the limit.

Generally, any binary stationary subdivision scheme for solids can be written as

$$P_{k+1}(z) = a(z)P_k(z^2), \qquad z \in \mathbb{R}^3, \tag{5.10}$$

where $P_k(z) = \sum_{\mu\in\mathbb{Z}^3} p_\mu^k z^\mu$ is a formal generating function associated with the control points $\mathbf{p}^k = \{p_\mu^k\}_{\mu\in\mathbb{Z}^3}$ at the level k, and $a(z)$ is the characteristic polynomial derived from the local subdivision matrix $\mathbf{S}$:

$$a(z) = \sum_{\mu\in\mathbb{Z}^3} a_\mu z^\mu. \tag{5.11}$$

By comparing these coefficients after n iterations of the subdivision process, one can

show (see [33]) that

$$||\mathbf{S}^n||_\infty = \max_\gamma \sum_{\nu \in \mathbb{Z}^3} a^{[n]}_{2^n\nu+\gamma}, \tag{5.12}$$

where $\gamma \in \{0, 1, \ldots, 2^n - 1\}^3$ and $a^{[n]}(z) = \prod_{j=0}^{n-1} a(z^{2^j}) = \sum_\mu a^{[n]}_\mu z^\mu$. This relation will be used to calculate the norm of the subdivision matrix. We utilize the coefficients of the characteristic *polynomial* (or *Laurent polynomial*, to be correct) instead of the matrix itself to compute the norm. The rest of the proof will follow the steps below:

(a) Find the characteristic polynomial of the scheme.

(b) Derive the *difference processes* of the scheme along the directions that are associated with the characteristic polynomial of the scheme.

(c) Prove that the difference processes are continuous by using their characteristic polynomials, and thereby show that the scheme is C^1.

Steps 2 and 3 are special cases of the following two theorems. Readers who are interested in the proofs of the theorems are referred to the work by Dyn *et al.* [38].

Theorem 5.1. *Let the characteristic polynomial of* $\mathbf{S}$ *have the form*

$$a(z) = q(z) \prod_{i=1}^{s} (z^{\theta^{(i)}} + 1), \tag{5.13}$$

where q is a Laurent polynomial and $\theta^{(i)} \in \mathbb{Z}^s$ *satisfies*

$$|det(\theta^{(1)}, \ldots, \theta^{(s)})| = 1. \tag{5.14}$$

Let D_i *be the subdivision matrix corresponding to the polynomial* $a(z)(z^{\theta^{(i)}} + 1)^{-1}$. *Then the subdivision scheme associated with* $\mathbf{S}$ *is uniformly convergent if and only if for some* $L \in \mathbb{Z}_+$,

$$||D^L||_\infty = \max_{1 \le i \le s} ||D_i^L||_\infty < 1. \tag{5.15}$$

Theorem 5.2. *Let* $\mathbf{S}$ *be convergent with a characteristic polynomial*

$$a(z) = (z^\theta + 1)^\nu 2^{-\nu} q(z), \tag{5.16}$$

where $\theta \in \mathbb{Z}^s$, $\nu \in \mathbb{Z}_+$, and q is a Laurent polynomial. If the subdivision scheme associated with q converges uniformly, then for all initial control points $\mathbf{p}^0$,

$$\partial_\theta^\nu \mathbf{S}^\infty \mathbf{p}^0 \in C(\mathbb{R}^s), \tag{5.17}$$

where ∂_θ means the directional derivative in the direction θ, i.e.,

$$\partial_\theta f(x) = \lim_{t \to 0} \left(f(x + t\theta) - f(x) \right). \tag{5.18}$$

Theorems 5.1 and 5.2 provide us the sufficient conditions to guarantee the C^1 continuity of the subdivision scheme. The conditions for the norm of the matrix will be confirmed by means of the relation explained in Equation 5.12. During most of the process, we will rely on numerical experiments to verify the satisfaction of the conditions.

The characteristic polynomial of our new scheme has the form of

$$a(z) = \sum_{\mu \in \mathbb{Z}^3} a_\mu z^\mu. \tag{5.19}$$

To obtain the coefficients, we successively apply our subdivision masks over the octet-truss structured regular grid on $\mathbb{Z}^3$. We perform the same process over each edge $\overline{p_{2k}p_{2k+1}}$, $k = 0, \cdots, 6$,where $p_0(-1,0,0)$, $p_1(1,0,0)$, $p_2(0,-1,0)$, $p_3(0,1,0)$, $p_4(0,0,-1)$, $p_5(0,0,1)$, $p_6(-1,-1,-1)$, $p_7(1,1,1)$, $p_8(-1,0,-1)$, $p_9(1,0,1)$, $p_{10}(0,-1,-1)$, $p_{11}(0,1,1)$, $p_{12}(-1,-1,0)$, and $p_{13}(1,1,0)$. The coefficient is explicitly formulated as

$$a_\mu = 1, \quad \mu = (0,0,0)$$

$$\begin{aligned} a_\mu = \tfrac{1}{2}, \quad \mu &= (\pm 1, 0, 0) \\ &= (1,1,1), (-1,-1,-1) \\ &= (1,1,0), (-1,-1,0) \end{aligned}$$

$$\begin{aligned} a_\mu = w, \mu &= (2,1,2),(-2,-1,-2) \\ &= (2,1,1),(-2,-1,-1) \\ &= (2,1,0),(-2,-1,0) \\ &= (2,1,-1),(-2,-1,1) \\ &= (1,1,-1),(-1,-1,1) \\ &= (1,-1,0) \end{aligned}$$

$$\begin{aligned} a_\mu = -w, \mu &= (3,2,1),(-3,-2,-1) \\ &= (3,1,1),(-3,-1,-1) \\ &= (3,1,0),(-3,-1,0) \\ &= (2,3,2),(-2,-3,-2) \\ &= (2,-1,0),(-2,1,0) \\ &= (2,-1,-1),(-2,1,1) \end{aligned}$$

Finally, we have

$$a_{\mu'} = a_\mu, \text{if} \quad \mu' = \sigma(\mu), \quad \sigma \in S_3,$$

where S_3 denotes the set of all permutations over $\{1,2,3\}$, which is followed by the symmetry of the subdivision mask in the regular case. The function can be factored using $(1+z_1)(1+z_2)(1+z_3)(1+z_1z_2z_3)$. Therefore, it can be written as

$$a(z) = \frac{1}{2}(z_1z_2z_3)^{-1}p(z)\prod_{i=1}^{4}(1+z^{\theta^{(i)}}) \tag{5.20}$$

$$\{\theta^{(i)}\} = \{(1,0,0),(0,1,0),(0,0,1),(1,1,1)\}, \tag{5.21}$$

where $p(z)$ is a Laurent polynomial with respect to z_1, z_2, and z_3 which is of the form $p(z) = 1 - wq(z_1,z_2,z_3)$. If $w = 0$, Equation (5.20) becomes the generating function of a linear interpolation.

To prove that the scheme is C^1, it is sufficient to show that $||D^L_{(i_1,i_2)}|| < 1$ and $||D'^L_{i_1}|| <$

1, $i_1 \neq i_2$ for some L where

$$\begin{aligned} D_{(i_1,i_2)}(z) &= 2(1+z_{i_1})^{-1}(1+z_{i_2})^{-1}\, a(z) \\ D'_{i_1}(z) &= 2(1+z_{i_1})^{-1}(1+z_1z_2z_3)^{-1}\, a(z). \end{aligned}$$

Because $a(z)$ is invariant for a permutation on indices, it is equivalent to show $||D^L_{(1,2)}|| < 1$ and $||D'^L_1|| < 1$ where

$$\begin{aligned} D_{(1,2)}(z) &= 2(1+z_1)^{-1}(1+z_2)^{-1}\, a(z) \\ D'_1(z) &= 2(1+z_1)^{-1}(1+z_1z_2z_3)^{-1}\, a(z), \end{aligned}$$

respectively.

It has been proven numerically that $||D^L_{(1,2)}|| < 1$ and $||D'^L_1|| < 1$ for $w = \frac{1}{16}$, especially when $L > 5$. In more general cases, the conditions are satisfied for some L, if $w > 0$ is small enough.

5.3.2 Tension Control

By controlling the weight w, we have different effects on the geometry of our models. In the case of $w = 0$ (see Figure 5.4(a)), the subdivision simply performs a linear interpolation. By increasing w, we expect to receive many ripples in a model. Eventually, the subdivision diverges if w exceeds a certain threshold (see Figure 5.4(f))

5.3.3 Extraordinary Cases

Unlike surface subdivision schemes whose irregular analysis involves only extraordinary vertices, we must take care of both extraordinary vertices and edges in solid schemes [4]. Unfortunately, existing spectral analysis using Discrete Fourier Transform (DFT) [31, 105] cannot be directly adopted for solid schemes, as the technique is based on spectral behavior over a 2-dimensional domain. However, we can still employ eigenvalue and

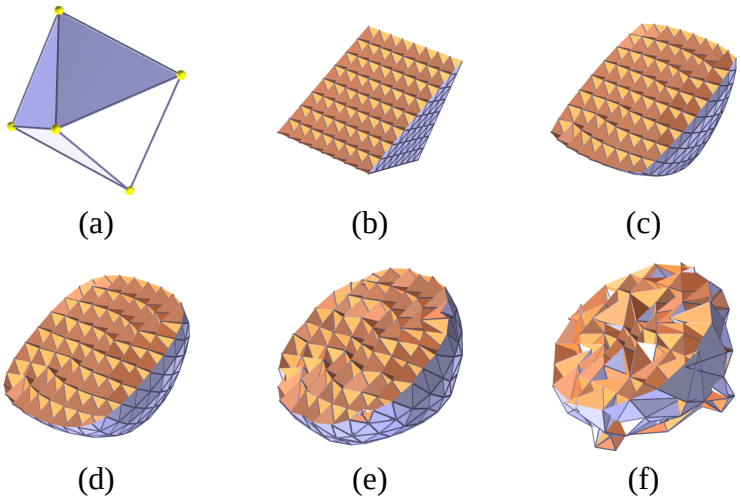


Figure 5.4: Tension control. (a) The original control points. (b-f) show the model at level 4 with $w = 0$, $w = \frac{1}{32}$, $w = \frac{1}{16}$, $w = \frac{1}{8}$, and $w = \frac{1}{4}$, respectively.

characteristic map analysis [84] numerically, at least for restricted cases, which are well-understood techniques for surface subdivision analysis.

For instance, the local invariant subdivision matrices $\mathbf{S}$ of the scheme around extraordinary vertices and edges satisfy an eigenvalue property of

$$\lambda_0 = 1 \gneqq \lambda_1 \geq \lambda_2 \geq \lambda_3 \gneqq \lambda_4, \ldots, \lambda_n, \tag{5.22}$$

where λ_i's are eigenvalues of $\mathbf{S}$ in decreasing order. It is worth mentioning that we have triple subdominant eigenvalues which are strictly less than 1. Also, we have numerically generated characteristic maps from those eigenvalues and associated eigenvectors as their control nets, and have confirmed that the maps are one-to-one and regular over a large number of iterations. Some of examples are shown in Figure 5.5.

Even though we can confirm the regularity of the characteristic map for each specific case, it is impossible to prove it symbolically due to the aforementioned reason (*i.e.*, DFT is not applicable in volumetric settings). In general, extraordinary edge cases are limited and simple to generate. However, the number of extraordinary vertex cases is exponentially bounded by the vertex valence n and is associated with the number of triangulation of

n points (in general position) over a spherical domain [47]. Further research should be conducted to exploit a more systematic way in order to prove extraordinary cases in solid subdivision schemes. Such an investigation might result in a new spectral analysis tool for even higher domains.

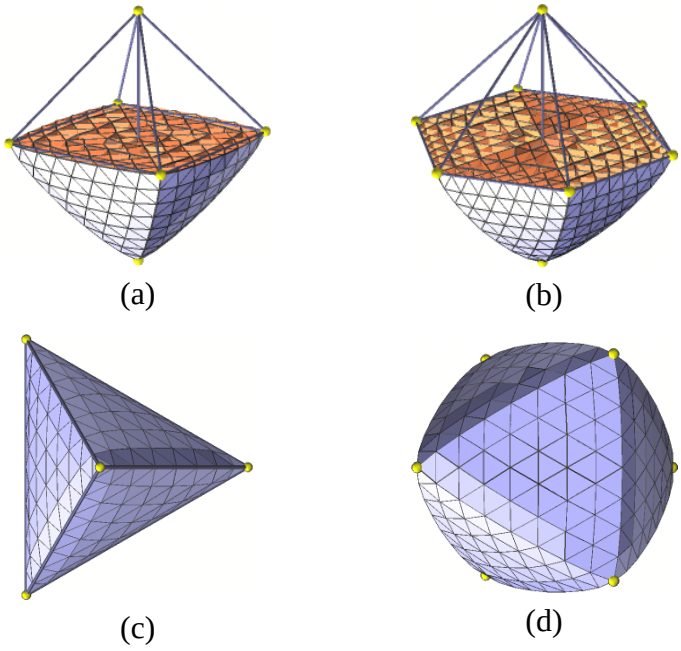


Figure 5.5: Control nets for rings of characteristic maps for our scheme. (a-b) Control nets for extraordinary edges with 4 and 6 incident vertices, respectively (cross-sections). (c-d) Control nets for extraordinary vertices of valence 4 and 6, respectively.

Chapter 6

Multi-dimensional Non-manifold Subdivision Framework

In real-world modeling, curves, surfaces, and solid objects are often mixed and represented in a single workplace. In this chapter, we propose a novel method that can represent multi-dimensional non-manifold subdivision objects in a single framework. The framework also supports the boundary and the sharp feature representations.

The Multi-dimensional Non-manifold Subdivision Framework was introduced in a paper presented in the 9th ACM Symposium on Solid Modeling and Applications 2004 [20]. The extended paper is submitted to Graphical Model for review [18].

6.1 Introduction and Motivation

Many industrial design projects include a wide range of shape representations in a single place. For instance, in car design, a hood of a car can be represented as thin plate, while volume representation is more appropriate for the engine parts. Such situation leads to complicated non-manifold objects, where curves and surfaces meet together, or multiple surfaces are coincide in one edge. In addition, the boundary and the feature representations are essential in mechanical design. Without modification, subdivision schemes tend to smooth objects, since the subdivision process is weighted averaging in essence. In this

chapter, we establish a framework that is based on flexible parametric domains and powerful subdivision rules which can be applied to objects with complicated dimensionality. The goals of our new approach are as follows:

- Define a parametric domain that provides high flexibility in modeling and simplicity in topological inquiry.
- Represent objects with multiple dimensions in a single framework.
- Develop unified subdivision rules for arbitrary manifolds and multiple dimensions.
- General or user-specified treatments for non-manifold regions.
- Support the boundary and the sharp feature representation
- Level-of-detail (LODs) control.

6.2 Simplicial Complex Domain

In the next few sections, we introduce several definitions related to the simplicial complex that are to be utilized for various topological inquiries during the subdivision process.

6.2.1 Set Definitions

Our domain of choice is a simplicial complex in $\mathbb{R}^n$. A k-simplex S can be defined as a set in $\mathbb{R}^n$,

$$S = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{x} = \sum_{i=1}^{k} c_i(\mathbf{x}_i - \mathbf{x}_0)\}, \tag{6.1}$$

where

$$c_i \geq 1, \ \sum_{i=1}^{k} c_i = 1, \ \mathbf{x}_i \in \mathbb{R}^n. \tag{6.2}$$

Since S can be uniquely determined by $k+1$ points $\mathbf{x}_0$, $\mathbf{x}_1$, ... ,$\mathbf{x}_k$, and is independent of their ordering, we simply use a set notation $S := \{\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_k\}$. In this paper, we

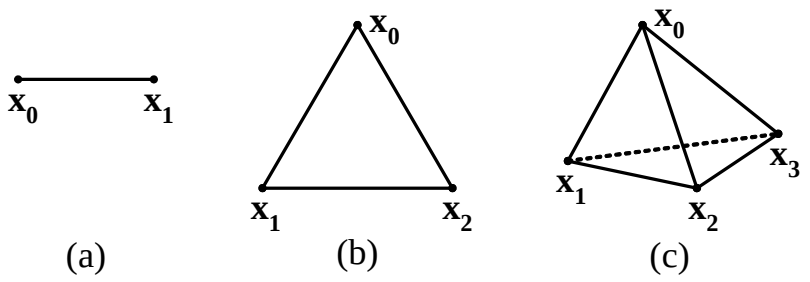


Figure 6.1: Examples of simplices. (a) A 1-simplex, (b) a 2-simplex, and (c) a 3-simplex.

limit k to be less than or equal to three. Note that any subset of S also forms a simplex. Geometrically, each subset can be considered as a face, an edge, or a vertex. We call k the *dimension* of the simplex S, or $\dim(S)$.

A simplicial complex, or a complex, $\mathcal{C}$ is a collection of simplices where: (1) the subsets of each simplex in $\mathcal{C}$ is in $\mathcal{C}$; (2) the intersection of any two simplices of $\mathcal{C}$ is a subsimplex of both. The second property prevents the introduction of T-junctions or the improper incursion among simplices. Also, a nonempty subset $\mathcal{D}$ of a simplicial complex $\mathcal{C}$ is called a *simplicial subcomplex* if it also satisfies the properties. We simply call it a *subcomplex*. The dimension of a complex is defined by the highest dimension of simplices in it.

In the complex $\mathcal{C}$, we call a simplex a *subsimplex* if it is a subset of other members of the complex. Likewise, it is called a *proper subsimplex* if it is a proper subset of a simplex. In addition, a simplex is called a *maximal* simplex if it is not a subsimplex of any other simplices in $\mathcal{C}$.

In summary, the domain space of our framework can be expressed as the pair of the following sets:

- Set of vertices

$$\mathcal{V} = \{\mathbf{x}_i \mid \mathbf{x}_i \in \mathbb{R}^3\}, \tag{6.3}$$

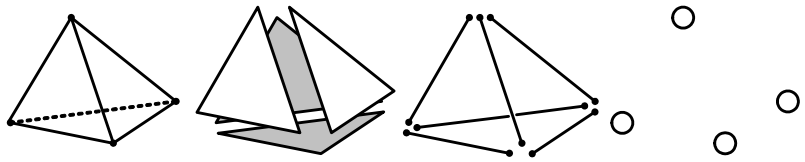


Figure 6.2: The subsimplices of a 3-simplex. (a) The 2-subsimplices, (b) the 1-subsimplices, and (c) the 0-subsimplices.

- A simplicial complex:

$$\mathcal{C} = \{S \subset \mathcal{V} \mid S \neq \emptyset,\ |S| \leq n+1\}, \tag{6.4}$$

with the following property:

$$\text{If } S \in \mathcal{C},\ \text{then } T \in \mathcal{C} \text{ for all } T \subset S,\ T \neq \emptyset. \tag{6.5}$$

6.2.2 Complex Decomposition

A complex $\mathcal{C}$ can contain simplices of different dimensions. Since each k-simplex is to be used as a part of the initial control points of a k-manifold, we need to decompose $\mathcal{C}$ with respect to the dimensions of the simplices. We define $\mathcal{C}_k$ as the largest subcomplex of $\mathcal{C}$, whose maximal simplices always have the dimension k. In other words, $\mathcal{C}_k$ comprises of all maximal k-simplices and their subsimplices in $\mathcal{C}$. We call it a *k-subcomplex*. Therefore, we can express $\mathcal{C}$ as:

- k-subcomplex decomposition

$$\mathcal{C} = \bigcup_{k=1,\cdots,m} \mathcal{C}_k, \tag{6.6}$$

where each $\mathcal{C}_k$ satisfies the following property:

$$\text{If } S \in \mathcal{C}_k \text{ and is maximal in } \mathcal{C}_k,\ \text{then } \dim(S) = k. \tag{6.7}$$

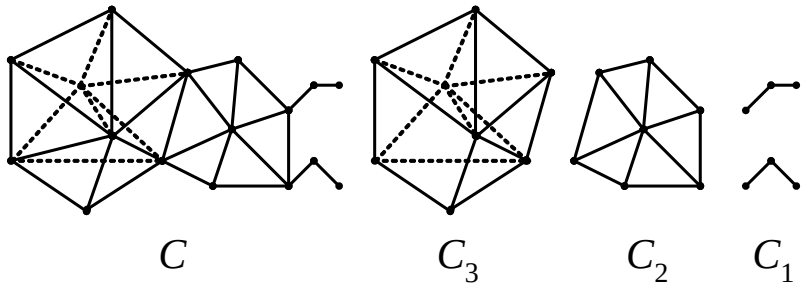


Figure 6.3: Complex decomposition. A complex $\mathcal{C}$ can be decomposed into $\mathcal{C}_k$'s with $k = 1, 2, 3$.

Later on, we define k-manifolds (with boundary) over the k-subcomplex using appropriate subdivision rules. However, $\mathcal{C}_k$'s are not mutually exclusive. This fact leads us to the need for special rules across the intersections of the k-subcomplexes. In fact, the intersections represent non-manifold regions in the result. Moreover, some non-manifold regions could appear within $\mathcal{C}_1$ and $\mathcal{C}_2$, since the complex is defined over $\mathbb{R}^3$.

6.2.3 Boundary Simplex

A face of a k-simplex S is simply defined as a $(k-1)$-subsimplex of S. A boundary of a complex can be defined as follows:

- Boundary simplex: If $(k-1)$-simplex $S \in \mathcal{C}$ is a face of a maximal k-simplex, and is not a subsimplex of any other simplices, than S defines a boundary. We call it a k-boundary simplex.

It is clear that boundary simplices and their subsimplices form a subcomplex of $\mathcal{C}$. It is denoted by $\partial\mathcal{C}$.

6.2.4 Non-manifold Simplex

If our domain consists of a single k-simplex, it is trivial to establish a manifold map from the simplex to a k-manifold. However, it is not always possible to define a manifold

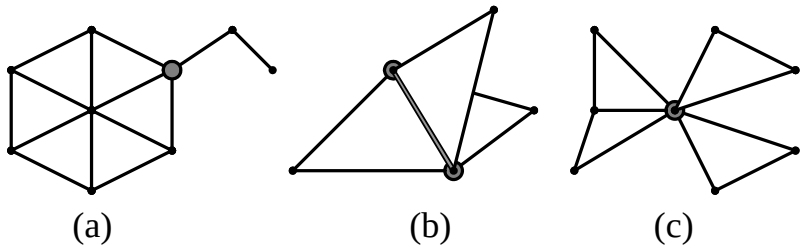


Figure 6.4: Examples of complexes containing non-manifold simplices. (a) Type 1, (b) type 2, and (c) type 3. The vertices or edges in gray are the non-manifold simplices.

map over a complex. For instance, if the domain consists of an 2-simplex and a 3-simplex joined by an edge, it is not possible to define either 1- or 2-dimensional Euclidean map across the edge. Also, if three or more 2-splices share a single edge in general position, we cannot find any single Euclidean map that can be well-defined across the edge. These cases occur only on the intersections of the simplices that comprise the domain. We call a simplex *non-manifold* simplex, if we cannot define an Euclidean map on the simplex. The following definition covers all the possibilities of non-manifold simplices:

- Non-manifold simplex: A k-simplex $S \in \mathcal{C}$ is a non-manifold simplex, if it satisfies any of the following properties.

  1. $S \in \mathcal{C}_k \cap \mathcal{C}_l$ where $k \neq l$.
  2. $S \in \mathcal{C}_k$ exclusively, $dim(S) = k - 1$ and $S = S_1 \cap S_2 \cap S_3$ for some distinct k-simplices $S_1, S_2, S_3 \in \mathcal{C}_k$.
  3. $S \in \mathcal{C}_k$ exclusively, $dim(S) < k-1$, $S = S_1 \cap S_2$ for some maximal k-simplices $S_1, S_2 \in \mathcal{C}_k$, $S_1 \neq S_2$ and S is not a proper subsimplex of any non-manifold simplex.
  4. $S \in \mathcal{C}_k$ exclusively, $dim(S) < k - 1$ and S is a subsimplex of a non-manifold simplex

These four properties are mutually distinct. We call the first three cases a type 1, a type 2 and a type 3 non-manifold simplex, respectively. We employ various strategies to tackle the non-manifold cases. Generally, non-manifold simplices create ill-posed problems. There could be several different solutions to meet a particular requirement in certain applications. We rely on a user-specific preference to resolve the problems. If no rule is specified by the user, we use the subdivision rules for 3-manifolds to spatially blend the manifolds of different dimensions. The forth case can be dealt with the same solutions for the other three cases.

6.3 Unified Subdivision Scheme

In the previous section, we defined the domain of the framework as a simplicial complex. Our object can be represented by the sum of smooth basis functions that are defined locally over the simplices in the complex:

$$f(\mathbf{x}) = \sum \mathbf{p} N(\mathbf{x}), \tag{6.8}$$

where $\mathbf{p} \in S \in \mathcal{C}$ with $\dim(S) = 1$. Therefore, the 1-simplices (or vertices) in the complex act as the control points of the shape. $N(\mathbf{x})$ is a basis function with local support defined over the complex. Basis functions form a partition of unity on $\mathcal{C}$. We choose the box spline as the function $N(\mathbf{x})$ whose support lies in the 1-ring of simplices. For multivariate cases, we do *not* use the tensor-product generalization of splines in strong contrast to many other subdivision schemes, since our domain is based on a complex. Instead, we introduce multivariate box splines with simplex support. One example is Loop's scheme [58] for surfaces. For 3-D, we use the box spline solid that has been employed in our previous work [16]. Non tensor-product box splines are particulary useful in the subdivision process, since: (1) Their subdivision rules are obtained intuitively from their definitions; (2) They

can achieve comparable smoothness with relatively low polynomial degree; (3) The choice of domain is more flexible.

6.3.1 Regular Subdivision Rules

Even though it is possible to figure out the subdivision rules using the definitions of the box splines, it is more convenient to use the generating functions of the box splines and their recursive relations. It is known that the coefficients of the generating functions can provide us the coefficients for the subdivision rules, as proven in [97]. In general, the generating function $S_D(\mathbf{z})$ for the box spline $N_D(\mathbf{x})$ can be expressed as:

$$S_D(\mathbf{z}) = \frac{1}{2^{d-k}} \prod_{i=1}^{d} (1 + \mathbf{z}^{\delta_i}), \tag{6.9}$$

where $d = |D|$. Note that the power of $\mathbf{z}$ follows the multi-index notation. For each k, the generating functions of the double $(k+1)$-directional box splines are:

- $k = 1$:

$$S_D(z) = \frac{1}{8}(1 + z)^4. \tag{6.10}$$

- $k = 2$:

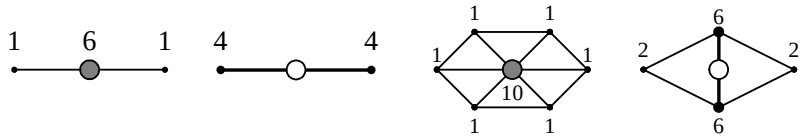
$$S_D(z_1, z_2) = \frac{1}{16}(1 + z_1)^2(1 + z_2)^2(1 + z_1 z_2)^2. \tag{6.11}$$

- $k = 3$:

$$S_D(z_1, z_2, z_3) = \frac{1}{32}(1 + z_1)^2(1 + z_2)^2(1 + z_3)^2(1 + z_1 z_2 z_3)^2. \tag{6.12}$$

We can find the subdivision rules for the regular simplicial meshes by assigning the coefficients of the $\mathbf{z}^{\delta_i}$'s to the vertex with the coordinates δ_i. We can summarize the rules as follows:

- Regular k-simplex subdivision rules:



(a) (b)

Figure 6.5: Regular subdivision rules. (a) The 1-simplex rules. (b) The 2-simplex rules.

Vertex points (for each vertex $\mathbf{x}_i$):

$$\mathbf{v}_{new} = \frac{1}{2^{k+2}}\Big\{(2^{k+1}+2)\mathbf{x}_i + \sum_{\mathbf{x}_j \in \rho(\mathbf{x}_i)} \mathbf{x}_j\Big\}. \tag{6.13}$$

Edge points (for each edge $\mathbf{e}_i = [\mathbf{x}_i, \mathbf{x}_{i+1}]$):

$$\mathbf{e}_{new} = \frac{1}{2^{k+1}}\Big\{(2^{k-1}+1)(\mathbf{x}_i + \mathbf{x}_{i+1}) + \sum_{\mathbf{x}_j \in \rho(\mathbf{e}_i)} \mathbf{x}_j\Big\}. \tag{6.14}$$

Cell points (for each octahedral cell $[\mathbf{x}_i, ..., \mathbf{x}_{i+3}, \mathbf{x}_j, \mathbf{x}_{j+1}]$, with the diagonal $[\mathbf{x}_j, \mathbf{x}_{j+1}]$):

$$\mathbf{c}_{new} = \frac{1}{8}\Big\{(\mathbf{x}_i + \cdots + \mathbf{x}_{i+3}) + 2(\mathbf{x}_j + \mathbf{x}_{j+1})\Big\}. \tag{6.15}$$

Here, we use more conventional names for 0-, 1-, 2-, and 3-simplices, namely, vertices, edges, and cells, respectively. $\rho(\cdot)$ denotes the 1-ring of neighboring vertices of a vertex or an edge. In the regular k-simplicial meshes, $|\rho(\mathbf{x})| = 2^{k+1} - 2$, and $|\rho(\mathbf{e})| = 2^k - 2$ for each vertex $\mathbf{x}$ or edge $\mathbf{e}$. Note that each k-manifold generated by the subdivision rules on the regular mesh satisfies C^k smoothness as mentioned above.

6.3.2 Extraordinary Subdivision Rules

In practice, a complex $\mathcal{C}$ could contain a vertex or an edge, that does not have a regular number of neighbors $|\rho(\cdot)|$ (or *valences* for vertices). We call them the *extraordinary* cases. They require modified rules to accommodate the lack (or the excessiveness) of neighbors.

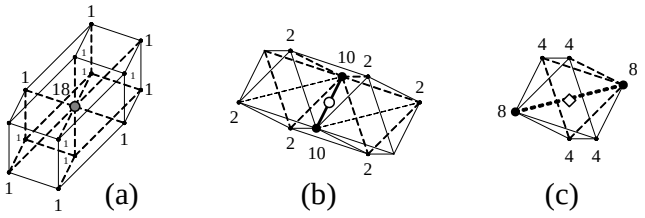


Figure 6.6: Regular 3-simplex subdivision rules. (a) Vertex point, (b) edge point, and (c) cell point rules.

Fortunately, the extraordinary cases are isolated over the subdivision processes. Also, some of the regular rules do not require any extraordinary rule. For instance, the 1-simplex rules do not have any extraordinary case. For the 2-simplex rules, there could be only extraordinary vertices. Likewise, no extraordinary cell point rule is required for the 3-simplex rules.

The extraordinary vertex rule for a 2-simplex has been well studied and there is a considerable amount of literature suggesting the coefficients for the rule that guarantee at least C^1 smoothness in the limit. For instance, the original Loop scheme [58] suggests the coefficients for a vertex with valence m that are derived from the discrete Fourier analysis and the eigenvalue analysis of the subdivision matrix. We adopt the values proposed by Warren *et al.*[97]:

- Modified 2-simplex subdivision rules:

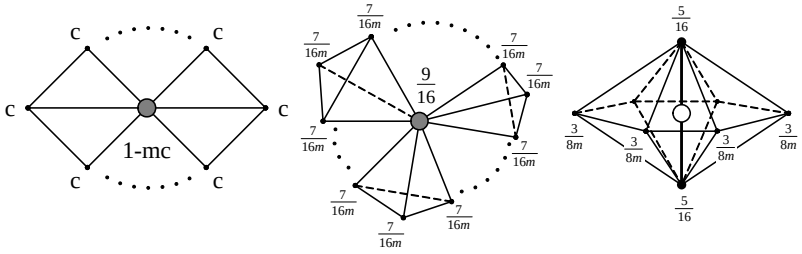
  Vertex points ($|\rho(\mathbf{x}_i)| = m$):

$$\mathbf{v}_{new} = (1 - mc)\mathbf{x}_i + c \sum_{\mathbf{x}_j \in \rho(\mathbf{x}_i)} \mathbf{x}_j, \tag{6.16}$$

  where $c = \frac{3}{16}$ for $m = 3$, $c = \frac{3}{8m}$, otherwise.

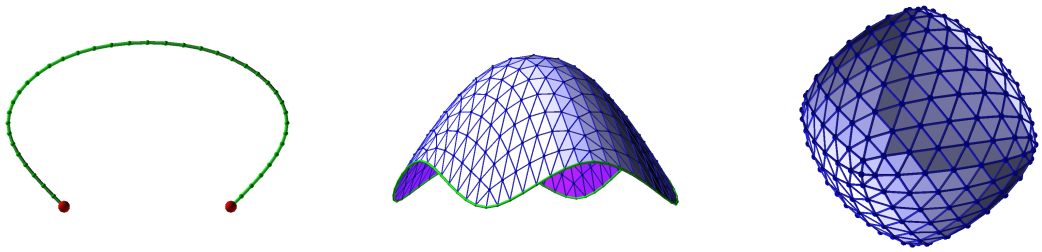
Similar modifications are required for the 3-simplex subdivision rules:

- Modified 3-simplex subdivision rules:



(a) (b) (c)

Figure 6.7: Modified k-simplex subdivision rules.



(a) (b) (c)

Figure 6.8: Examples of manifolds with boundary. (a) A 1-manifold with boundary. (b) A 2-manifold with boundary. (c) A 3-manifold with boundary.

Vertex points ($|\rho(\mathbf{x}_i)| = m$):

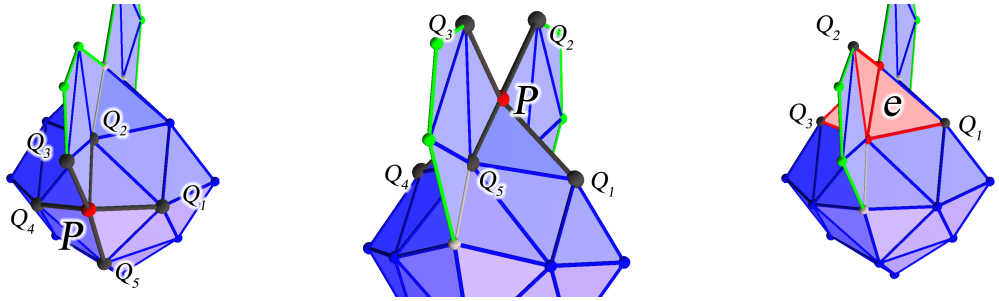
$$\mathbf{v}_{new} = \frac{9}{16}\mathbf{x}_i + \frac{7}{16m}\sum_{\mathbf{x}_j \in \rho(\mathbf{x}_i)} \mathbf{x}_j. \tag{6.17}$$

Edge points ($|\rho(\mathbf{e}_i)| = m$):

$$\mathbf{e}_{new} = \frac{5}{16}(\mathbf{x}_i + \mathbf{x}_{i+1}) + \frac{3}{8m}\sum_{\mathbf{x}_j \in \rho(\mathbf{e}_i)} \mathbf{x}_j. \tag{6.18}$$

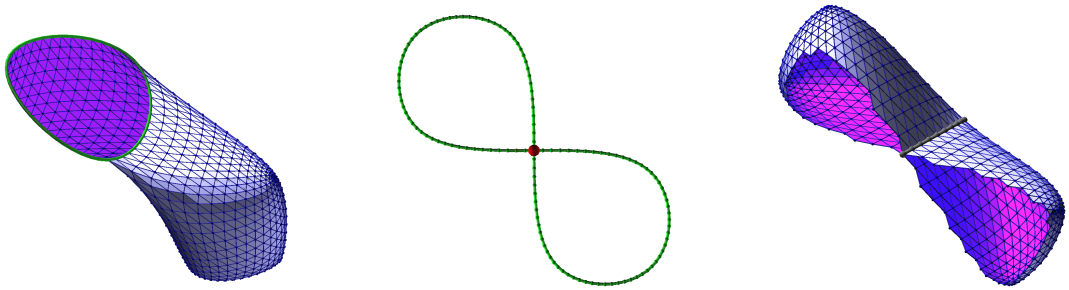
6.3.3 Boundary and Non-manifold Rules

The boundaries of k-manifolds cannot be represented by the k-simplex subdivision rules, because they are defined by the faces of k-simplices. Instead, we use the $(k-1)$-simplex subdivision rules to represent the boundaries. Since all of the subdivision rules rely



(a) (b) (c)

Figure 6.9: The 1-ring neighbors with the relieved topology condition. (a) and (b) show examples of the vertex 1-ring for type 3 non-manifold vertices. (c) shows an example of the edge 1-ring neighbors.



(a) (b) (c)

Figure 6.10: Examples of non-manifold cases. (a) A type 1 case by the 2- and 3-manifold intersection. The rule N-1 with 1-simplex rules is applied. (b) A type 2 case by a single 1-manifold. (c) The cross-section of the type 2 case.

only on the 1-rings of neighbors, this approach causes no additional trouble between the boundary and the interior simplices. It is, in fact, a standard approach for most subdivision surface schemes. Figure 6.8 demonstrates examples of such boundary cases.

Non-manifold regions require a special rules. We categorize the cases into three types, as explained in Section 6.2.3. In each case, we rely on user input to determine which rules to apply. If the user has not provided a choice, we try to find the best possible way to deal with it. Ying *et al.*[102] proposed detailed approaches to overcome non-manifold topology with subdivision surfaces. They involve a the specially modified Loop's scheme and a geometric fitting process. Since our domain is in $\mathbb{R}^3$ and we have the 3-simplex

subdivision rules that can accept an arbitrary manifold with lower dimension, our solution is much simpler, as described below. For each case, we can apply either specific rules (N-1 and N-2) or general rules (G-1 and G-2):

- The following three rules are specific for type 1 and type 2 cases.

  Rule N-1 Type 1 is a region where the manifolds with different dimensions meet. In this case, we can follow the subdivision rules for single k-simplex of our choice.

  Rule N-2 Type 2 is a region where a multiple manifold of a single dimension intersects by their faces. This region can be considered as a self-intersection. Our suggested solution is to choose one pair of the simplices on which we apply the subdivision rule.

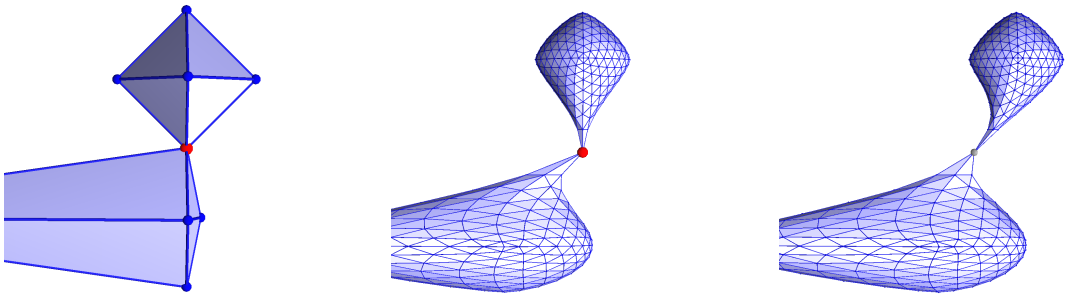
- Type 3 is a region where multiple manifolds of a single dimension intersect, but they do not share the faces. In this case, we found that the general rules described below yield the best results.

- Regardless of the type, we can apply one of the general rules as follows:

  Rule G-1 Treat the intersection as a 0-, or 1-singularity.

  Rule G-2 Use the 3-simplex subdivision rules with the relieved topology condition.

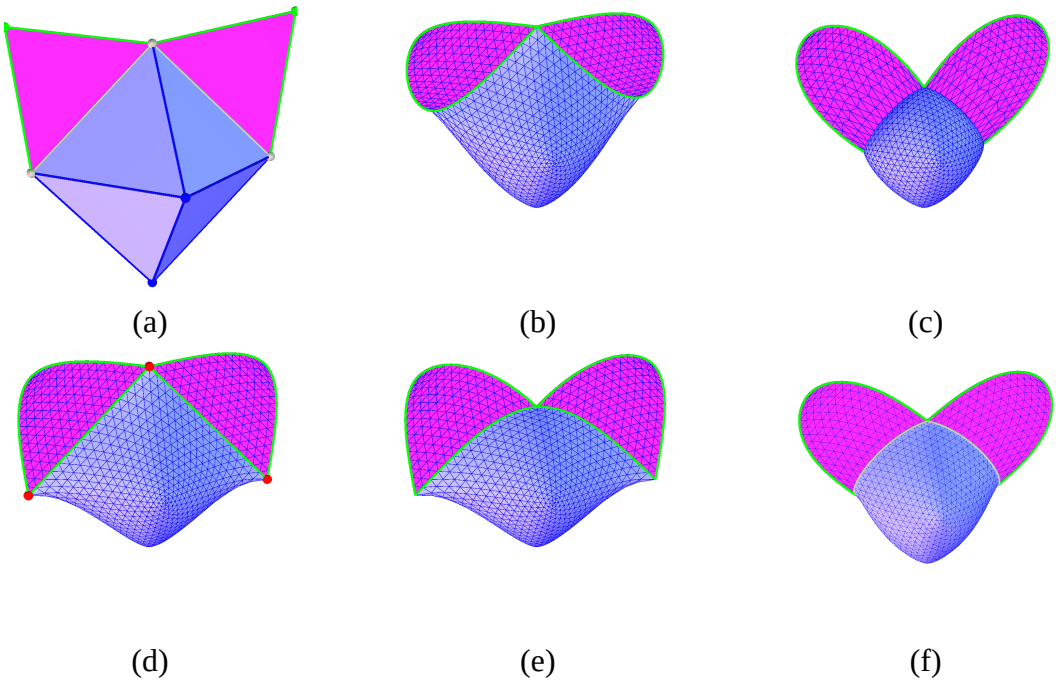
By a subdivision rule with the *relieved topology condition*, we mean that the rule only considers the connectivity between each vertex when acquiring the 1-ring of neighbors, regardless of the existence of a simplex in between. Figure 6.9 shows examples of neighbor choices by the relived topology condition. Since the intersection between simplices with different dimensions always occurs on the boundary complex $\partial\mathcal{C}$, we only choose the neighbors in $\partial\mathcal{C}$. Figure 6.10 and 6.11 illustrate examples of non-manifold cases. In



(a) (b) (c)

Figure 6.11: Type 3 non-manifold rules. (a) The initial complex. (b) The subdivision by the rule G-1. (c) The subdivision by the rule G-2. The red vertex preserves its position in (b), while it is blended in (c).

Figure 6.10(a), a 2-manifold (the purple area) intersects with a 3-manifold (the blue area). Therefore, it forms a type 1 case. For this particular case, a user has decided to follow the rule N-1 with 2-simplex subdivision rule. Thus, the intersected area follows the 2-simplex boundary rule, and the 3-manifold is attached onto it. Figure 6.10(b) and (c) show typical type 2 cases. In Figure 6.10(b), the 1-manifold intersects itself at a point (the red vertex). A multiple number of surfaces intersects at an edge in Figure 6.10(c). For both cases, we use the rule G-2 to blend non-manifold parts into the bodies. Figure 6.11 shows the effects of the different rules. In Figure 6.11(b), the user selects the red vertex to be a singularity (the rule G-1). Hence, we only apply the 0-mask (*i.e.*, the 1×1 identity matrix) on the vertex during the subdivision process. Thus, it preserves the position during the subdivision. However, in Figure 6.11(c), we follow the rule G-2. As a result, the vertex has been moved according to the positions of the 1-ring neighbors because we use the subdivision rules for 3-simplices. In the end, the final shape is much smoother and all the boundaries are well blended. Figure 6.12 lists all the solutions provided by our subdivision scheme for a single configuration. Overall, the rule G-2 provides the most visually pleasing results. We should mention that the suggested rules do not represent all the possible solutions. Nonetheless, we can introduce a new rule depending on the requirement of a particular application.



(a) (b) (c)

(d) (e) (f)

Figure 6.12: Comparison between the non-manifold rules. (a) The initial control points. The complex consists of one 3-simplex (octahedron) and two 2-simplices (triangles). The intersection between the 3-simplex and 2-simplices form type 1 cases. (b) The rule N-1 is applied. In this case, we consider the intersection as a part of the boundaries of the 2-simplices (triangles). (c) The rule N-1 is applied. But, instead of using 2-simplex boundary rules, we utilize the intersection as a part of the boundary of the 3-simplex. As a result, the boundary of the 3-simplex region does not change at all. (d) We apply the rule G-1 with the vertices as 1-singularities. (e) We apply the rule G-1 with the edges as 2-singularities. The intersection create an 1-singular curve on the surface of the 3-simplex boundary. (f) Finally, the rule G-2 is applied to the intersection. No information is specified by the user. Only connectivity and 3-simplex subdivision rule is used. In the end, the intersection is smoothly blended.

6.4 Singularity and Adaptivity

Even though the subdivision rules that we have presented so far are ideal for representing smooth objects, it is desirable to have a model with sharp features, such as cusps, creases, or corners, especially in real-world applications. Also, we may want to have more details in some part of the model without subdividing the whole complex. In the following sections, we discuss the extensions of the framework that can increase its benefit in practical solid modeling.

6.4.1 Singularity Representation

Hoppe *et al.*[46] suggested a modification of Loop's scheme to represent sharp features within smooth surfaces. Our basic idea is similar to theirs. However, we generalize the approach to apply to multi-dimensional models.

A manifold defined by the subdivision rules is C^1 smooth over the complex $\mathcal{C}$ except in non-manifold regions. To represent features within the manifold: (1) We need to specify the area of the domain where the features occur; (2) We need to specify the subdivision rules to represent the features in the manifold. Among many types of features, we only consider "sharp" features, where the manifold is continuous, but is *not* differentiable. We call this type of features a *singularity* for convenience. We define a k-*singular* simplex by:

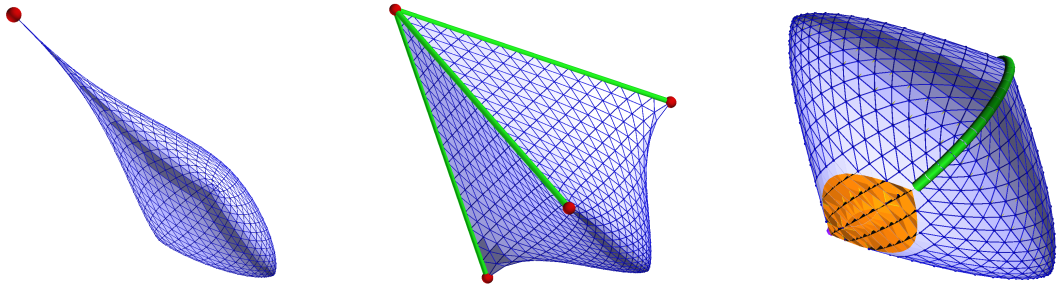
- k-singular simplex: A k-simplex $S \in \mathcal{C}$ is a k-singular simplex, if and only if: (1) There exists no C^1 map to l-manifolds defined over any simplex $T \in \mathcal{C}$, where $S \subset T$ and $k < l$. (2) It is possible to define a differentiable map on the singular simplex to k-manifolds.

We consider a subcomplex $\mathcal{S} \subset \mathcal{C}$, which is a collection of all singular simplices and their subsimplices in $\mathcal{C}$. Since they are a complex by themselves, all definitions and subdivision rules that are applied to the complex $\mathcal{C}$ are also applicable to $\mathcal{S}$. Basically, $\mathcal{S}$ generates

embedded manifolds within the original manifolds on $\mathcal{C}$. When applying the subdivision rules, if a vertex $\mathbf{x}$ or an edge $\mathbf{e}$ belongs to a maximal simplex in $\mathcal{S}$, we only follow the subdivision rules that match the dimension of the simplex, and ignore any other simplices that may contain the singular simplex. Figure 6.13 illustrates examples of singularities which our framework can represent. As shown in Figure 6.13(a), if a vertex (a 1-simplex) is assigned to be singular, then the scheme only applies the 0-mask on the vertex during the subdivision. Therefore, the vertex does not change its position at each subdivision level. However, other vertices around it follow the normal rules. As a result, we can obtain an object which is smooth except at one singular vertex and in its local area. This singularity is particularly useful to generate a cusp on the part of a manifold. In Figure 6.13(b), a user has assigned one vertex and all edges that go through it as singular. The 0-mask is applied to the vertex, and each edge follows the 1-simplex edge rule. It effectively produces a corner and three creases starting from it. The case shown in Figure 6.13(c) is more subtle. The user has introduced a 2-manifold singular region in the middle of the 3-manifold. As a result, the 3-manifold is split into two parts along with the singular surface. Both parts are smooth inside and outside, but the intersection is only smooth along with the tangent direction of the singularity. These types of singularities are especially useful if we want to design or fit objects with heterogeneous material. For instance, we can model a geological image containing streams and mineral veins (1- and 2-singularities) with ease.

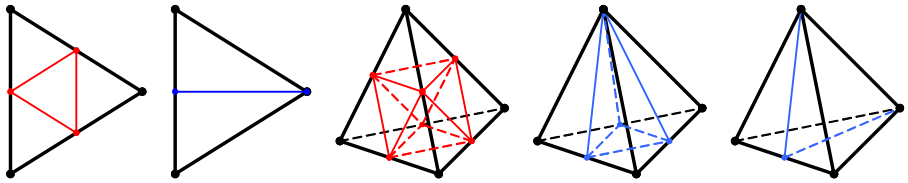
6.4.2 Local Adaptive Refinement

During the process of modeling an object represented by our framework, a situation can occur, that requires finer simplices than originally given. For instance, we may want to generate very fine details on a certain region of the manifold that is defined over one simplex originally. Since the subdivision rules generate a C^1 smooth box spline on a single simplex, it is not possible to achieve high-level of detail without splitting the simplex itself.



(a) (b) (c)

Figure 6.13: Examples of singularities in manifolds. (a) A singular vertex. (b) A corner and creases. (c) A 2-manifold embedded in the 3-manifold.



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

Figure 6.14: Local refinement rules. (a) Red rule and (b) Green rule for local triangulation. (c) Red rule, (d) Green-III rule, and (e) Green-I rule for local tetrahedralization.

One obvious solution is a global refinement of the entire complex. This surely would work, but at the expense of the size of the complex and the memory consumption. If we simply split a single simplex, the integrity of the complex will be broken, since the neighboring simplices become non-simplicial by the introduction of cracks, or T-junctions. We follow typical Red-Green split rules to avoid the situation (See Figure 6.14). For the 1-simplex case, no special rule is needed. For the 2-simplex case, only the 1-ring of the adjacent simplices are affected by Green rule (Figure 6.14(b)). For 3-simplices, the 1-ring of the adjacent simplices are split by Green-III rule (Figure 6.14(d)), while the 2-ring of the neighboring simplices and the edge-sharing simplices are modified by Green-I rule (Figure 6.14(e)). For an octahedral cell, we simply split it into four tetrahedra, without effecting the neighbors. Then we can apply Red-Green rules as usual.

6.5 Implementation

In this section, we discuss detailed issues related to the implementation of the framework and some of results that are from our experimental design system.

6.5.1 Input Data

As an input, the framework takes a combination of the vertex set $\mathcal{V}$, the complex $\mathcal{C}$, and the singular subcomplex $\mathcal{S}$. However, since subsimplices can be induced from maximal simplices, we do not need all the simplices in $\mathcal{C}$. So, in the implementation, we only take the data in Algorithm 6.1 as an input.

1: MULTI-DIMENSIONAL-SUBDIVISION ($\mathcal{V}$, $\mathcal{C}_{max}$, $\mathcal{S}_{max}$)
   $\{\mathcal{V} = \{\mathbf{x}_i \mid \mathbf{x}_i \in \mathbb{R}^3\}$
   $\mathcal{C}_{max} = \max(\mathcal{C}) = \{S \in \mathcal{C} \mid S : \ maximal\}$
   $\mathcal{S}_{max} = \max(\mathcal{S}) = \{T \in \mathcal{S} \mid T : \ maximal\}\}$
2: set $\mathcal{M} = \mathcal{C}_{max} \cup \mathcal{S}_{max}$

Algorithm 6.1: MULTI-DIMENSIONAL-SUBDIVISION.

These are the minimum data that are required to reconstruct the complex and the other information. Additional input can include user-specific preferences for each non-manifold cases. Since we heavily rely on set operations on the complex, an efficient data structure is necessary.

6.5.2 Complex Construction

In Algorithm 6.2, we reconstruct the complex $\mathcal{C}$, the decomposition $\mathcal{C}_k$, and mark the type 1 non-manifold simplices according to the following process. Remember that $\rho(S)$ is 1-ring neighbors of S. After the process, newly generated subsimplices are checked to verify whether they are boundary or type 2 non-manifold simplices. The procedure is explained in Algorithm 6.3.

```
1: COMPLEX-CONSTRUCT (𝒱, ℳ)
   {ρ(S): 1-ring neighbor of S}
2: initialize each 𝒞_k as empty
3: for all k = 0, 1, 2, 3 do
4:    for all k-simplex S ∈ ℳ do
5:       put S in 𝒞_k.
6:       for all l-subsimplex T ⊂ S with l < k do
7:          put T in 𝒞_k
8:          if T ∈ 𝒞_k', k ≠ k' then
9:             tag T as non-manifold type 1
10:         end if
11:         construct ρ(S) if l = 0, or 1
12:      end for
13:   end for
14: end for
15: return all 𝒞_k
```

Algorithm 6.2: COMPLEX-CONSTRUCT.

```
1: FIND-BOUNDARY-AND-NON-MANIFOLD (𝒞_k)
2: for all k = 1, 2, 3 do
3:    for all new (k − 1)-subsimplex (face) T ∈ 𝒞_k do
4:       if T belongs to only one k-simplex then
5:          tag T as boundary
6:       else if T belongs to more than two k-simplex then
7:          tag T as non-manifold type 1
8:       end if
9:    end for
10: end for
```

Algorithm 6.3: FIND-BOUNDARY-AND-NON-MANIFOLD.

Still, we need to figure out non-manifold type 3 non-manifold simplices and subsimplices of type 1 and type 2 non-manifold simplices. It has to be done at the end, because the process requires type 1 and type 2 information. Here, we denote $\mu(S)$ a number of maximal simplices that contains S. Algorithm 6.4 shows the steps to this process.

```
1: FIND-TYPE-THREE-NON-MANIFOLD (C_k)
   {μ(S): A number of maximal simplices that contains S}
2: for all k = 0, 1 do
3:    for all l-simplex T ∈ C_k with l < k − 1 do
4:       for all l'-simplex S ∈ C_k with l < l' ≤ k do
5:          if T is a subsimplex of S and dim(S) = k then
6:             increase μ(T)
7:             if μ(T) ≥ 2 then
8:                tag T as non-manifold type 3
9:             end if
10:         else if T is a subsimplex of S and dim(S) < k then
11:            if S is non-manifold then
12:               tag T as the same non-manifold type as S
13:            end if
14:         end if
15:      end for
16:   end for
17: end for
```

Algorithm 6.4: FIND-TYPE-THREE-NON-MANIFOLD.

Once the complex construction is complete, we are ready to choose the appropriate subdivision rules for each vertex and edge. Note that the subsimplices induced from maximal simplices are required only for the neighborhood, the boundary, and the manifold test. It can be safely removed from the memory once every step is done.

6.5.3 Subdivision Process

In Algorithm 6.5, we construct the subdivision matrix and the 1-ring neighbors for each vertex and edge using the information gathered in the previous steps. Additional user input is considered to treat the non-manifold region. Then, we output $\mathcal{V}_{new}$ as the next level of

the vertices.

```
NEW-VERTEX-POINTS (V, C)
  {C = ⋃ C_k}
for all vertex x in V do
    filter ρ(x) so that it contains only the same type of vertices as x.
    choose the subdivision matrix S_x
    compute the vertex point v_new by S_x and the filtered ρ(x)
    associate v_new with x
    put v_new in V_new
end for
return V_new
```

Algorithm 6.5: NEW-VERTEX-POINTS.

We follow the exactly same steps for each edge to obtain a set of new edge points, $\mathcal{E}_{new}$. Once the new vertex and edge points have been computed, we split each simplex. The process is detailed in Algorithm 6.6.

```
SPLIT-SIMPLEX (V_new, E_new, C)
initialize V' and C' as empty
for all k = 0, 1, 2, 3 do
    for all k-simplex S ∈ C do
        if k == 0 or 1 then
            put v_new or e_new associated with S in V'.
        else
            if S is an octahedron cell then
                compute the cell point c_new
                put c_new in V'
            end if
            split S by v_new, e_new and c_new if required
            put the split simplices in C'
        end if
    end for
end for
return V', C'
```

Algorithm 6.6: SPLIT-SIMPLEX.

As a result, we obtain the finer complex $\mathcal{C}'$ with the new vertices $\mathcal{V}'$. We may continue the steps from Section 6.5.2 to achieve more subdivision level.

Chapter 7

Applications of Multiresolution Solid Objects

We demonstrate the potential of our Multiresolution Solid Objects by presenting a series of practical applications. Each subdivision scheme that has been developed is fully employed to display the advantages of it.

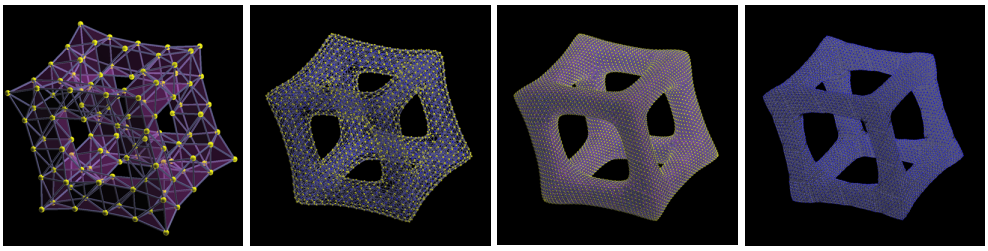
Shape modeling and heterogeneous material modeling applications where appeared in our subdivision papers [16, 17, 20] as examples. Implicit solid modeling, or DigitalSculptor *was introduced in a paper [61]. Free-form deformation was appeared in a paper presented in 7th ACM Symposium on Solid Modeling Applications 2002 [16]. The remaining applications are in preparation for submission.*

7.1 Direct Shape Modeling

Through Multiresolution Solid Objects, we can design various shapes and objects with ease. Since the meshes are based on the simplicial complexes, it offers great flexibility in topology and model shapes. Also, by using the multi-dimensional framework, we can represent complicated non-manifold shapes and feature-rich objects such as mechanical parts. In the following sections, we demonstrate the modeling ability of our MSO framework by employing the developed subdivision schemes individually.

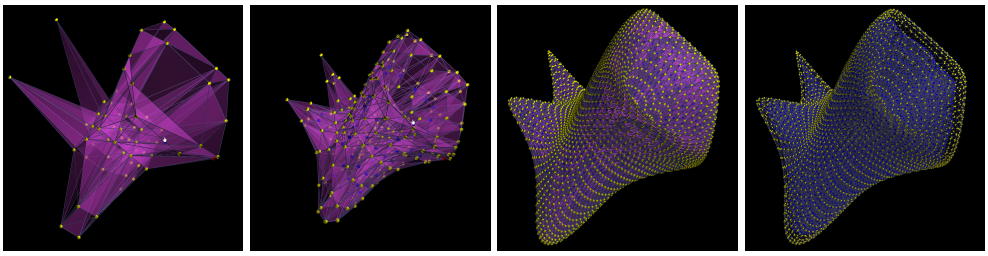
7.1.1 Box-spline based Approximate Solid Subdivision Scheme

Figure 7.1, Figure 7.2, Figure 7.3, Figure 7.4, and Figure 7.5 demonstrate examples of various models that can be obtained by the box-spline based solid subdivision. The tetrahedral structure offers the greatest freedom to generate objects of arbitrary topology. Also, by simple user interaction, we can perform real-time modifications on solids by manipulating control points at arbitrary levels. We can also introduce discontinuities on vertices or along edges by assigning exceptional rules (*i.e.*, simple bisection without weights) to desired parts of objects. We could also apply our subdivision scheme to some coarse FEM tetrahedral meshes to obtain both finer and better quality meshes (Figure 7.6).



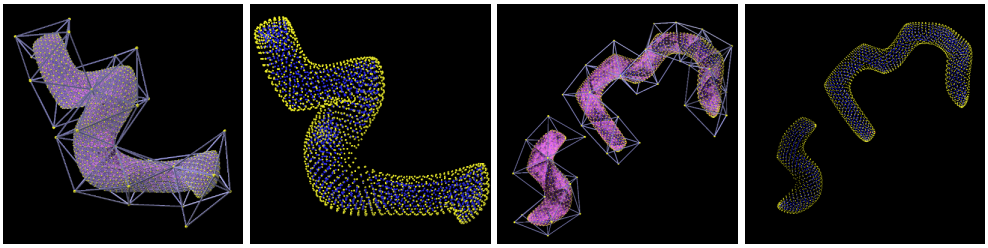
(a) (b) (c) (d)

Figure 7.1: Our subdivision algorithm can handle not only simple models, but also topologically complex models. (a) Original control points with high genus. (b) Subdivision level 2. The boundary and the interior are drawn separately. (c) Subdivision level 3. (d) The interior of the level 3 solid.



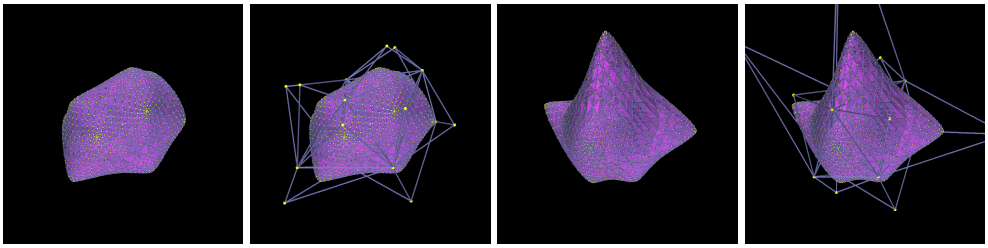
(a) (b) (c) (d)

Figure 7.2: More examples of topologically complex models. (a) Original control points with high genus. (b) Subdivision level 1. (c) Subdivision level 3. (d) The interior of the level 3 solid.



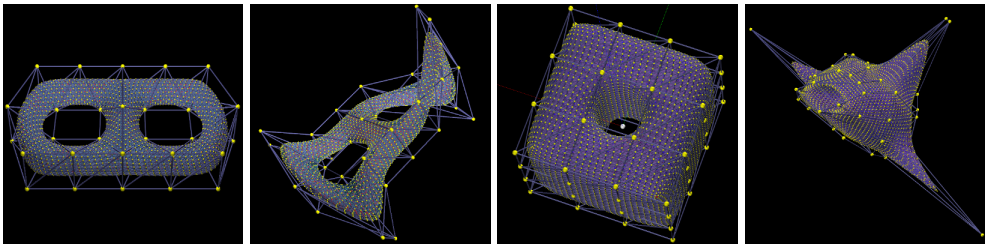
(a) (b) (c) (d)

Figure 7.3: Free-form objects. (a) The spiral solid in subdivision level 2. (b) The interior of the model. (c) SM logo in subdivision level 3. (d) The interior of the model.



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

Figure 7.4: Direct model manipulation. (a) The original model. (b) The original control points. (c) The manipulated model. (d) The manipulated control points.

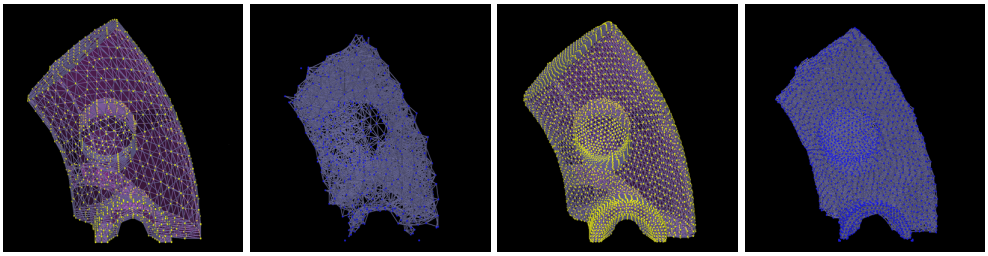


(a) (b) (c) (d)

Figure 7.5: Examples of models with arbitrary topology and their manipulations. (a) The original model with genus two. (b) The deformed model. (c) The original model with genus one. (d) The plane made from the model (c).

7.1.2 Interpolatory Solid Subdivision Scheme over Simplicial Complexes

Figures 7.8, 7.9, and 7.10 show several examples which have non-trivial topologies and are oftentimes impossible to generate as a single object by using surface subdivision schemes (also refer to Figure 7.7 and Color Plate). In Figure 7.11, we use a sweeping curve



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

Figure 7.6: Existing tetrahedral models can be subdivided using our scheme to acquire the finer mesh. (a) The original model with 12936 tetrahedra from FEM simulation. (b) The interior of the original. (c) Subdivision level 1 with 64680 tetrahedra. (d) The interior of the subdivided model.

equation with a specially designed mesh to generate a spiral model. Figure 7.12 and 7.13 (see also Color Plate) demonstrates the ability of our new subdivision scheme to handle non-manifold topology models without having to introduce a special set of rules. Not only does the scheme successfully display the solid part (which is colored in orange), it also has no difficulty in processing the surface-only region, which is colored in purple (Figure 7.12). The rule also can be applied to the degenerating case (see Figure 7.13 and Color Plate) without a significant modification. In addition, the scheme is also able to define the transition between two different regions. The weight for most of the results reported in this paper is $w = \frac{1}{16}$, unless otherwise documented. Besides the shape modeling functionality, our novel interpolatory subdivision scheme can also be readily suitable for many other solid modeling applications, such as Direct Manipulation. In contrast to an approximate scheme, changes on control points directly affect the modeled shape (see Figure 7.14). As a result, this can offer a naïve user more intuitive interaction between control points and a desired model.

7.1.3 Multi-dimensional Non-manifold Subdivision Framework

We have implemented a basic design system based on the framework. We present a few examples from the results of our system. Figure 7.15 shows a simple screw model

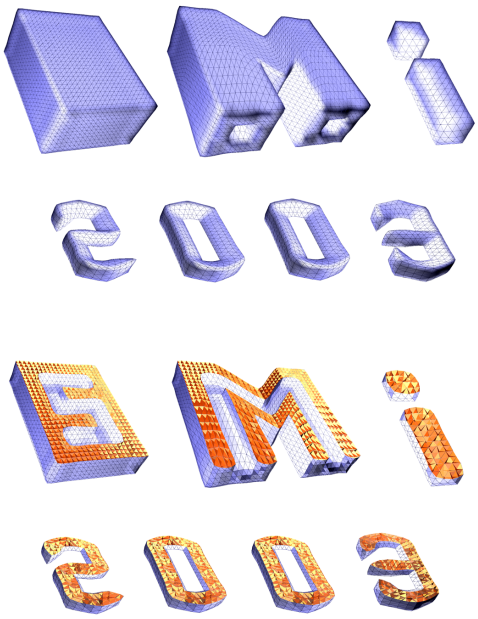
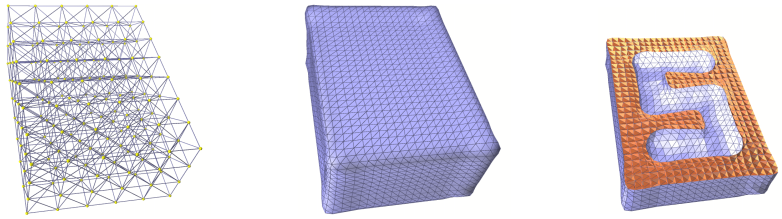
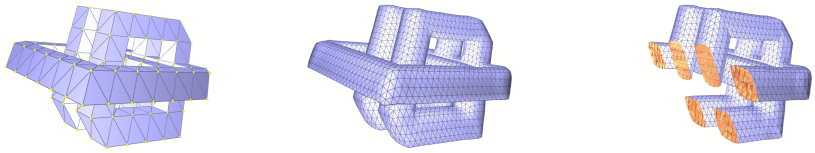


Figure 7.7: SMI logo and its cross-section created using our interpolatory subdivision scheme for solids. The new scheme successfully models complex structures inside the solid. It can also handle non-trivial topology with ease. Note that, throughout this paper the boundary of solid models is colored blue, and the cross-section of solid models is colored yellow.



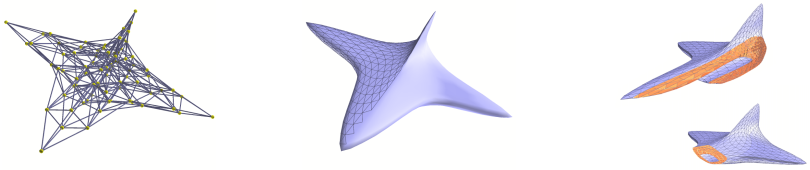
(a) (b) (c)

Figure 7.8: An embedded character. (a) The original control points. (b) The boundary of the model at level 3. (c) The cross-section of the model which reveals the "S" due to the vacancy inside the box.



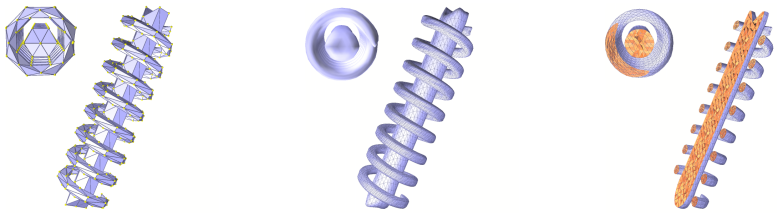
(a) (b) (c)

Figure 7.9: A knot-shaped model that has complex topology. (a) The original control points. (b) The boundary of the model at level 3. (c) A cross-section of the model.



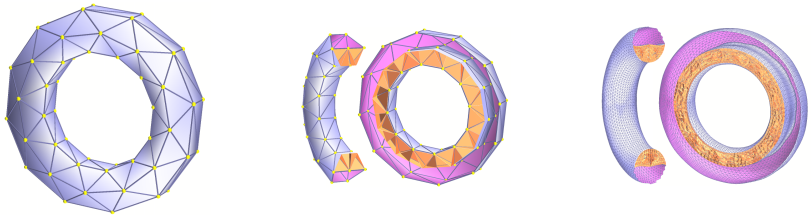
(a) (b) (c)

Figure 7.10: A simple design for space shuttle using our subdivision tools. (a) The original control points. (b) The boundary of the model at level 3. (c) Cross-sections of the model.



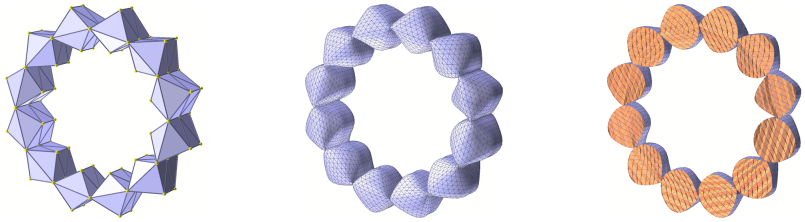
(a) (b) (c)

Figure 7.11: The scheme can be used to design a practical model such as a spiral. (a) The original control points. (b) The boundary of the model at level 3. (c) Cross-sections of the model.



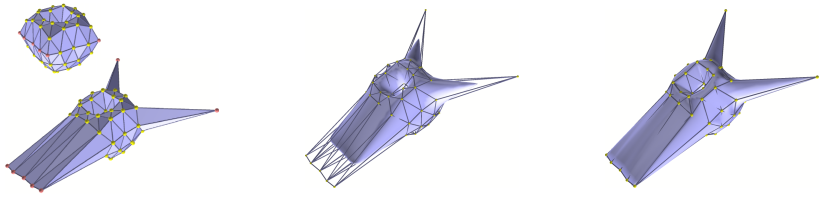
(a) (b) (c)

Figure 7.12: A torus model with non-manifold topology. (a) The original control points. (b) Cross-sections of the control points. (c) Cross-sections of the model at level 4. The purple region indicates the part of the model where only surface information is offered.



(a) (b) (c)

Figure 7.13: A segmented ring model. (a) The original control points. Each segment meets at a line which forms a non-manifold. (b) The boundary of the model at level 4. (c) A cross-section of the model at level 4.



(a) (b) (c)

Figure 7.14: A direct manipulation on control points. (a) The original control points and their modification by a user. (b) The shape generated using an approximate scheme. (c) The interpolatory scheme case. The changes on the model are more interactive and intuitive.

by less than 20 control points. Four blades consist of surfaces, where the core is a solid object. The cross-section shows the inner structure of the core. In Figure 7.16, we use a simple spiral equation to generate the solid spring part. The valve part comprises a solid cap and a cylinder which is a surface model. All parts are represented within a single complex mesh and the non-manifold parts are smoothly blended. In Figure 7.17, we have

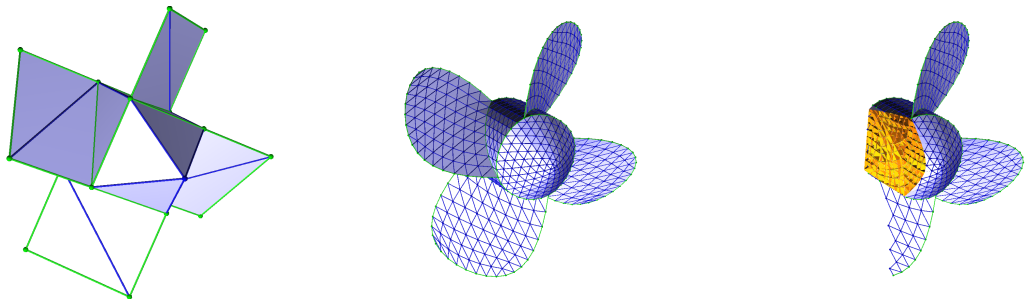
designed a part of a ship that consists of the solid bow, a few decks and a part of the hull. Figure 7.18 illustrates a mechanical part with non-trivial topology. The handle is a 2-manifold surface model, whereas the other parts are all solid. We use the singularity rules to make the rounded corners, the sharp corners, the flat surfaces and the circular holes. The framework has the potential to be a great animation character modeling tool, as shown in Figure 7.19. Oftentimes, animation characters contain manifolds of different dimensions, and our framework can handle them easily. In all figures, we use the same color scheme to represent different manifolds. Blue color represents 2-manifolds or the boundaries of 3-manifolds, while orange color is for the insides of 3-manifolds. Each vertex and edge are colored according to the subdivision rules that are applied to them. Red vertices are singular, and green and blue vertices are the vertices where we apply the 1- and 2-simplex rules, respectively. The same color scheme is applied for the edges. Table 7.1 shows the number of maximal simplices and the processing time on an Intel Pentium 4 2.4GHz machine with 1 gigabytes of memory. Note that actual number of simplices that represent each model is far greater than the number of maximal simplices listed below.

Model	Maximal simplices (initial)	Maximal simplices (level 3)	Time (sec.)
Screw (Figure 7.15)	32	4736	0.461
Ship hull (Figure 7.17)	46	4988	0.463
Mechanical part (Figure 7.18)	179	58608	16.343
Valve (Figure 7.16)	388	142376	96.098

Table 7.1: Examples of running times of the multi-dimensional subdivision framework.

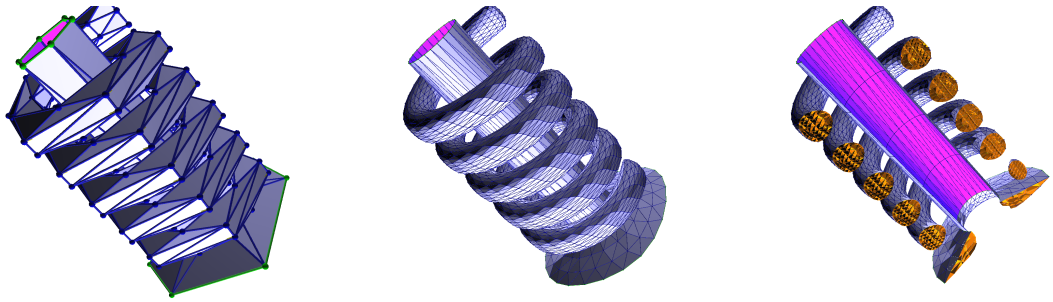
7.2 Heterogeneous Material Modeling

One of the great advantage of our Multiresolution Solid Objects over other solid representations is the ability to represent heterogeneous materials, especially continuously varying properties. Moreover, combined with the multi-dimensional scheme, we can acquire



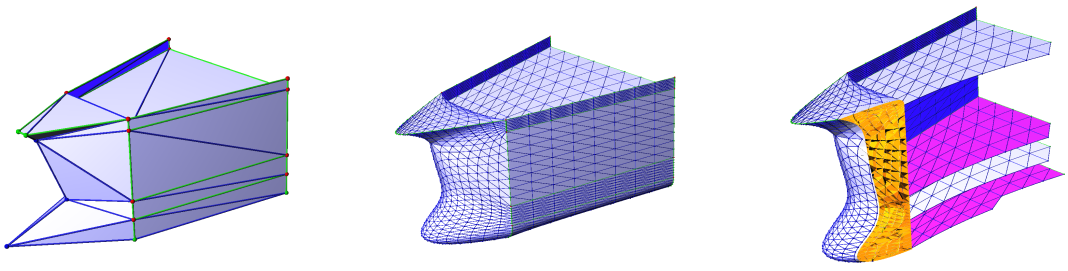
(a) (b) (c)

Figure 7.15: A screw model by the combination of 2- and 3-manifolds. It also includes creases around the 3-manifold boundary. (a) Initial control mesh. (b) After 3 levels of the subdivision. (c) The cross-section of 3-manifold region.



(a) (b) (c)

Figure 7.16: A valve model with a spring. The valve and the cylinder consists of 2- and 3-manifolds. The bottom part has creases. (a) Initial control mesh. (b) After 3 levels of the subdivision. (c) The cross-section of the model.

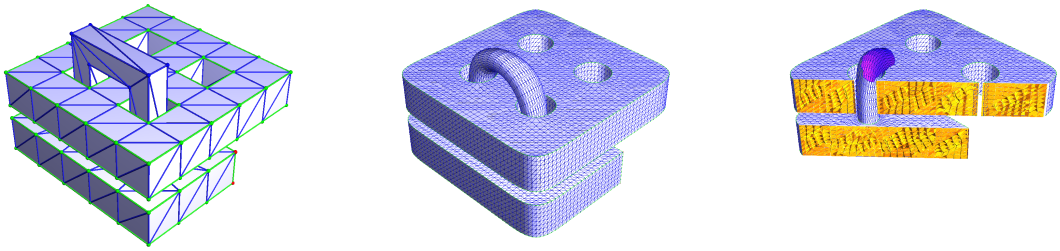


(a) (b) (c)

Figure 7.17: The bow and the part of the hull from a ship model. (a) Initial control mesh. (b) After 3 levels of the subdivision. (c) The cross-section of 3-manifold region.

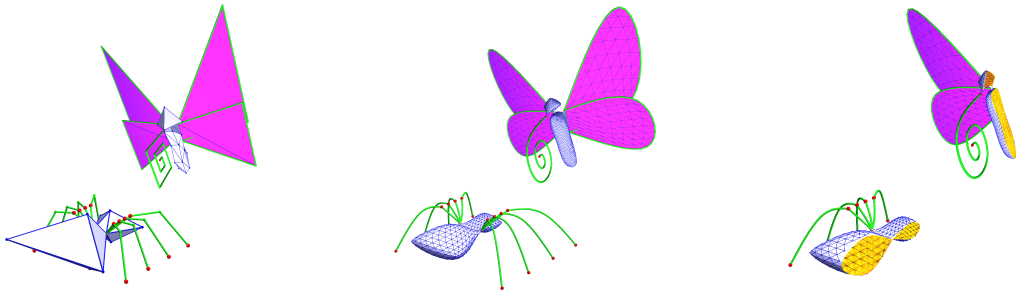
smooth material transition between surfaces and solids.

First, we present heterogeneous models represented by the box-spline based scheme. In



(a) (b) (c)

Figure 7.18: A model of a mechanical part with the complex topology. (a) Initial control mesh. (b) After 3 levels of the subdivision. (c) The cross-section of 3-manifold region.



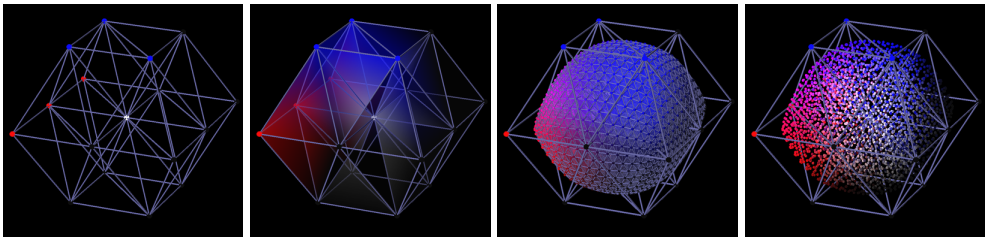
(a) (b) (c)

Figure 7.19: Insect characters for computer animation. (a) Initial control mesh. (b) After 3 levels of the subdivision. (c) The cross-section of 3-manifold region.

Figure 7.20 and Figure 7.21, it is clearly shown how material information is incorporated with geometry by our solid subdivision scheme. Because a solid generated by our algorithm exhibits high-order continuity, we can note the smooth transition of material data inside the objects. Instead of dealing with 3D coordinates, each model contains material information as a 4th or even higher order coordinates. The same subdivision process is applied to the extra coordinates. We use a 128^3 voxel grid to render each scene. A value for each voxel is obtained by 7.1. One potential cause for concern is that our scheme is an *approximating*, not an interpolating one, which in fact causes a discrepancy between the original and the values introduced by the algorithm. However, if the original model actually contains many tetrahedra, it converges rapidly even in the first couple of levels. Hence the error is relatively small. In many practical cases, it is negligible.

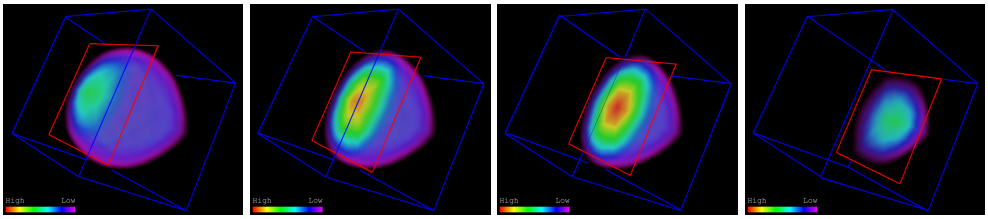
The interpolatory scheme has a unique advantage. Since it interpolates the original value, we can perform physical simulation with assigned values without approximating. We can assign material properties to control points and apply the exact same subdivision rule on the control mesh to acquire smooth interpolated properties. Figure 7.22 shows an example of the model with color information associated with control points. The colors between control points are continuously interpolated by the subdivision rules. Figure 7.23 presents various solid models with material properties. Simple diffusion process is used to initialize material data at level 1. Control points at higher levels are smoothly interpolated by the subdivision scheme. This process is completely transparent in the interpolatory scheme.

Finally, Figure 7.24 shows a non-manifold object with material properties employing our multi-dimensional scheme. We assign tension values at the initial level, and the subdivision rules smoothly blend them into the structure.



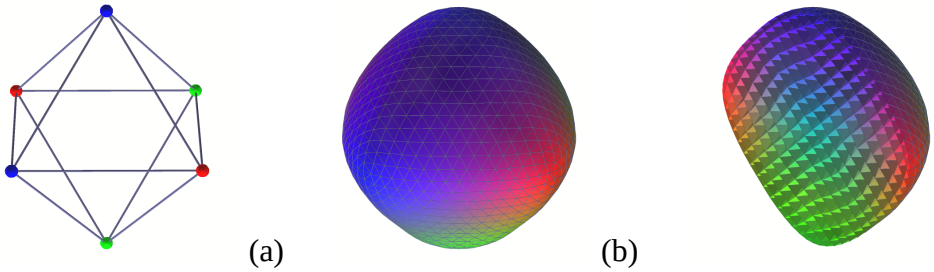
(a) (b) (c) (d)

Figure 7.20: Not only can subdivision solids represent geometry, but they can also be used to interpolate other data over 3D space. (a) Original control points and assigned colors. (b) Faces are transparently rendered by bilinear interpolating. (c) Level 3 subdivision solid. (d) The distribution of vertices inside. Colors are smoothly interpolated.



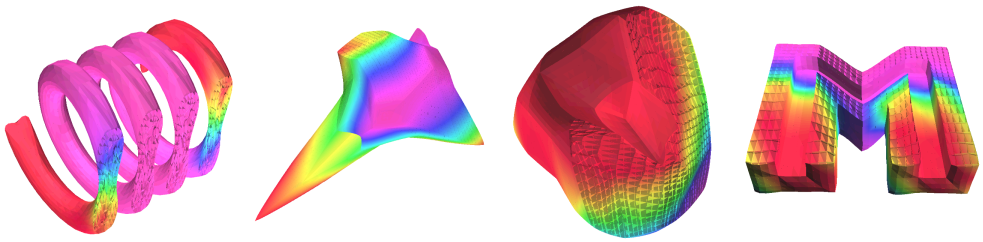
(a) (b) (c) (d)

Figure 7.21: Smooth distribution of densities inside a subdivision solid. The highest density is given to the central point. The volume is visualized using the latest low-end 3D graphic acceleration hardware.



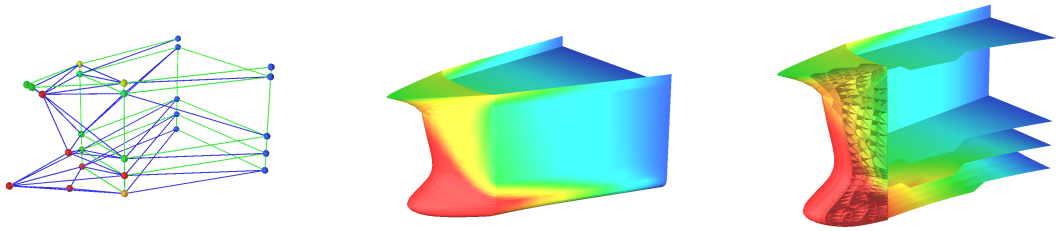
(a) (b) (c)

Figure 7.22: A model with a material property represented by colors. (a) The original control points with assigned colors. (b) The boundary of the model at level 5. (c) A cross-section of the model. Colors are smoothly interpolated.



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

Figure 7.23: Various solid models with material properties. Simple diffusion process is used to initialize material data at level 1. Control points at higher levels are smoothly interpolated by the subdivision scheme.



(g) (h) (i)

Figure 7.24: A material property representation. (a) Initially assigned values. (b) After 3 levels of the subdivision. (c) The cross-section of 3-manifold region.

7.3 Implicit Solid Modeling

Even though we have emphasized the advantage of the MSO representation as a modeling tool, it cannot compete with implicit function representations in editing, trimming, and sculpting. Implicit function representations inherently provide better solutions for set operations including union, intersection, and subtraction, collision detection, and presentation of simple primitives that can be easily defined by distance functions. Motivated by this and a number of disparate areas of research, including volumetric sculpting [3, 72, 93], we have implemented *DigitalSculpture*, a subdivision-based interactive implicit modeling system [61]. Based on the interpolatory solid subdivision scheme over unstructured hexahedral meshes [60], which are presented in Appendix C, the system utilize the computation power of the subdivision to evaluate the scalar field associated with hexahedral meshes. However, instead of manipulating the geometry, we only control the scalar fields over the meshes. An object is represented as a level-set surface of the scalar field extracted by the marching tetrahedra algorithm [90]. Naturally, the system can support set operations such as CSG operations and copy-and-paste function, local deformation, and direct modeling on surfaces. Some examples created by the system are shown in Figure 7.25 and Figure 7.26.

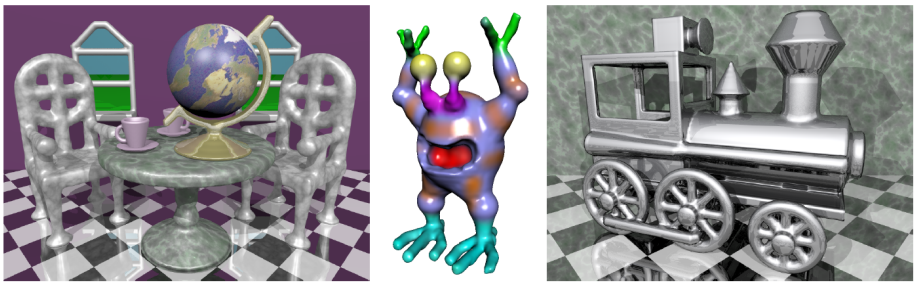


Figure 7.25: Virtual sculptures created in our DigitalSculpture modeling environment.

Figure 7.26: (a-c) Surfaces created with CSG operations in our implicit subdivision solid approach. (d-e) Objects created by curve sweeping - addition and subtraction of materials, respectively.

7.4 Free-form Deformation

Free-form deformation (FFD) is one of the important applications to which the solid subdivision is directly applied. It plays a crucial role in graphics, design and manufacturing. Usually, FFD involves generating parametric solids and translating model coordinates back to parametric space [89, 77, 23], so that changes to the solids can be reflected in the models. Our approach is similar to that of MacCracken *et al.*[59]. However, unlike the tensor-product nature of Catmull-Clark solids and volumetric splines, our solid is much more flexible due to the tetrahedral structure of the mesh. The following is an overview of our implementation of FFD.

(a) Generate an appropriate mesh that contains the model to be deformed.

(b) Subdivide the mesh up to the user-specified level using our new solid subdivision scheme.

(c) Calculate tetrahedra coordinates for each vertex in the model on the final level.

(d) The user then interactively moves control points in any coarser level.

(e) Recalculate the coordinates by following the subdivision matrices.

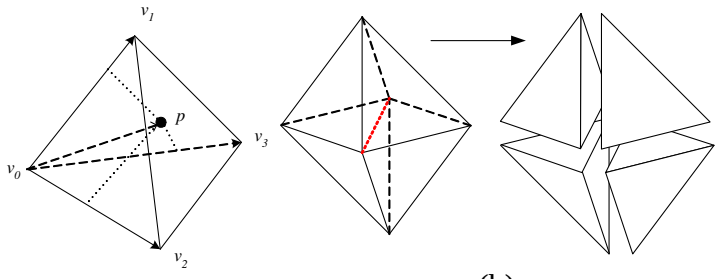
The barycentric coordinates (Figure 7.27(a)) are easy to compute. Suppose a model vertex $\mathbf{p}$ lies within a tetrahedron $[\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3]$. The coordinate (c_1, c_2, c_3) is given by

$$p = \mathbf{v}_0 + c_1\ \mathbf{u}_1 + c_2\ \mathbf{u}_2 + c_3\ \mathbf{u}_3, \tag{7.1}$$

where $\mathbf{u}_i = \mathbf{v}_i - \mathbf{v}_0$ for $i = 1, 2, 3$. Conditions $0 \leq c_1, c_2, c_3 \leq 1$ and $0 \leq c_1 + c_2 + c_3 \leq 1$ are required to be in the tetrahedron. We can solve the linear system

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} \vdots & \vdots & \vdots \\ u_1 & u_2 & u_3 \\ \vdots & \vdots & \vdots \end{pmatrix}^{-1} \begin{pmatrix} \vdots \\ \mathbf{p} - \mathbf{v}_0 \\ \vdots \end{pmatrix},$$

to obtain the coordinate. If a vertex is in an octahedral cell, we split the octahedron into 4 parts using the major diagonal and compute the barycentric coordinate within the corresponding tetrahedron (Figure 7.27(b)).

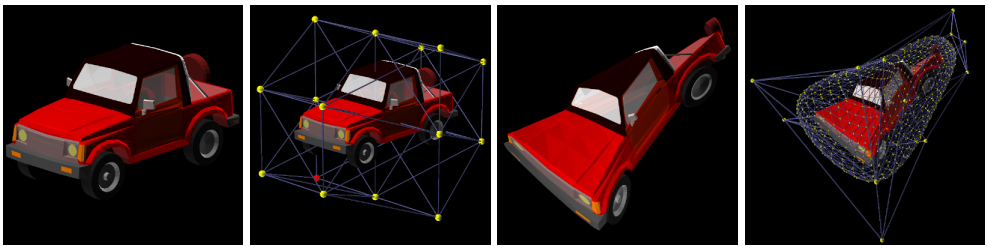


(a) (b)

Figure 7.27: (a) Barycentric coordinates are uniquely determined by an affine combination with respect to a tetrahedron that contains the point. (b) An octahedron is split into 4 parts, using a major diagonal.

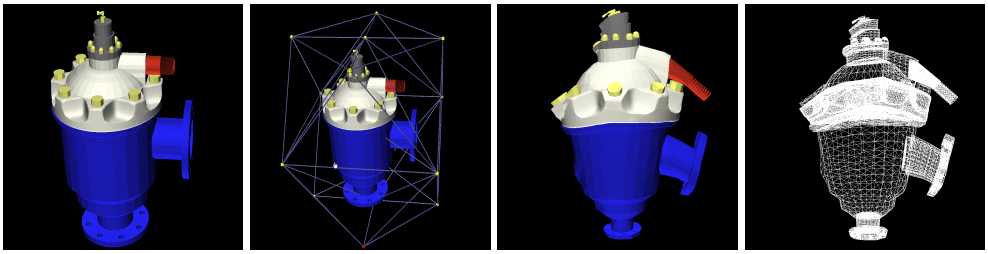
Figure 7.28 and Figure 7.29 show two applications of FFD. The mesh does not necessarily have a simple topology (*i.e.*, genus zero), as shown in Figure 7.30. It is also possible that we manipulate objects locally (Figure 7.31), by simply assigning control meshes to certain region of an object that we want to deform. Note that all processes are done in real-time except barycentric coordinate computation. Even the coordinate computation can

be done in a matter of seconds. Most of time is consumed by intersection checks between tetrahedra/octahedra and model vertices. We are working on several methods to accelerate these checks.



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

Figure 7.28: Free-form deformation of a car model. The model contains 2244 vertices with 21 different components. (a) The original model. (b) The subdivision mesh. (c) The deformed model. (d) Underlying subdivision solid. All deformation has been done in real-time.

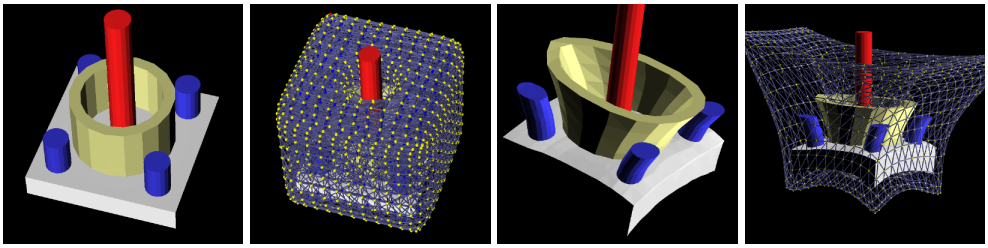


(a) (b) (c) (d)

Figure 7.29: Another example of free-form deformation of an industrial filter block model. The model contains 24877 vertices with more than 49000 faces, and is converted from the B-spline surface model. (a) The original model. (b) The subdivision mesh. (c) The deformed model. (d) The deformed model in wireframe.

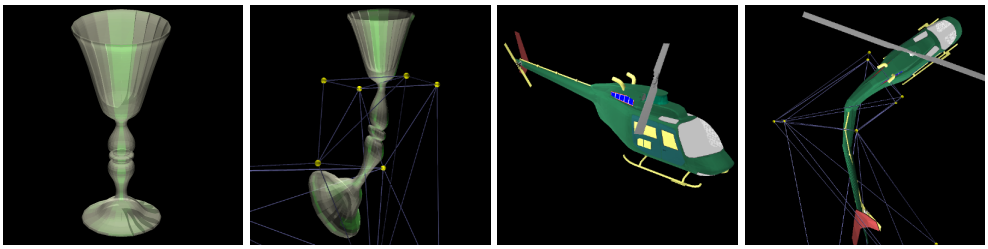
7.5 High Quality Surface Partitioning and Fitting for Subdivision

Currently, subdivision surfaces are extensively used in entertainment and game industry, where the visual quality is of most significance. However, subdivision surfaces have not yet gained their industrial acceptance in engineering design, analysis, and manufacturing applications, mainly due to a lack of higher smoothness near extraordinary vertices.



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

Figure 7.30: A mesh with non-trivial topology. In this case, our deformation mesh contains a hole. (a) The original model. (b) The subdivision mesh in level 2. (c) The part of the model (the central cylinder) that is inside the hole has not been changed. (d) The deformed subdivision mesh.



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

Figure 7.31: Localized free-form deformation. We can choose any region of the model and perform FFD. (a) The original model. (b) Locally deformed model. (c) The original model. (d) Locally deformed model. Each model contains 3760 and 7854 vertices, respectively.

The situation is aggravated for the subdivision solids, since there is no proper tool to prove even C^1 smoothness. One approach to avoid these difficulties is not to deal with the extraordinary cases, but rather to make the mesh regular *almost everywhere*, except the region where the smoothness is less important, such as sharp features or planar areas. Toward this goal, we begin with general surface subdivision cases and try to extend it to solid cases. We propose a new remeshing and data fitting technique to acquire high quality meshes for the subdivision method. The method consists of the following four steps:

(a) Feature Searching

(b) Domain Partitioning

(c) Conformal Parameterization

(d) Boundary and Surface Fitting

Given an arbitrary triangular mesh, we first find features represented by a set of vertices and edges during the Feature Searching stage. This set serves as a guide to the partitioning. At the Domain Partitioning step, we seed the surface with triangles which are furthest from the features. Then, we grow each region from the seeds to form a set of surface patches over the mesh. In the next step, we find the conformal parameterization from the canonical domain to each patch using Least Square method and a specific conformal map. Once the maps are found, we initialize control points as a regular lattice in the canonical domain and map them back to each patch using said conformal maps. Finally, Least Square fitting is utilized to fit the initial control points to the original mesh with a minimal error.

By following this procedure, we can have some unique benefits over the existing subdivision surface fitting:

(a) It provides a conformal map for each patch which are topologically isomorphic to the unit square.

(b) In each patch, all the initial control points are in regular configuration, *i.e.* they have the same regular valence numbers.

(c) All the extraordinary vertices are on, or close to, the feature sets of a given mesh, where the smoothness has less or no importance.

7.5.1 Problem Statement

Suppose that our input consists of a triangular mesh, expressed by a set of vertices $\mathcal{V} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n \mid \mathbf{x} \in \mathbb{R}^3\}$ and a set of triangles, or 1-simplices, $\mathcal{F} = \big\{\{\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \mathbf{x}_{i_3}\} \mid 1 \leq i_j \leq n, j = 1, 2, 3\big\}$. We denote this pair by $\mathcal{M} = (\mathcal{V}, \mathcal{F})$. In particular, we assume that our mesh as an input satisfies the following properties:

1. $\mathcal{F}$ and all proper subsets of its members forms a simplicial complex. For example, there is no intersection between the triangles except their edges. Formally, we define the complex as $\mathcal{C} = \{S \mid S \subset f \text{ for some } f \in \mathcal{F}\}$.

2. $\mathcal{F}$ forms a 2-manifold surface. For example, there is no edge shared by 3 or more triangles or a region only consisting of line-segments.

3. $\mathcal{F}$ represents a smooth manifold except the features.

Our goal is to generate a set of control points $\mathcal{P}$ for a subdivision scheme which represents the surface $\mathcal{F}$ in the limit. Any stationary subdivision scheme which is based on the evaluation of the splines can be guaranteed a high order of smoothness on regular valence. For instance, the Loop's scheme [58] can achieve C^2 smoothness on the regular triangular configuration where each vertex has a valence 6, whereas it is only C^1 over the extraordinary vertices. In general, most of the know stationary subdivision surface scheme can only guarantee C^1 smoothness over the extraordinary points. Therefore, it is important to have regular valences for as many initial control points as possible. However, it is not possible to have vertices with the regular valence everywhere unless the surface is topologically equivalent to the torus, or has the genus 1. Therefore, for general cases, our approach tries to reduce the number of the extraordinary points by segmenting the surface into several patches.

7.5.2 Feature Searching and Partitioning

We define features as the region where the manifold defined by the mesh $\mathcal{M}$ is not smooth, or non-differentiable. The features are embedded in the mesh as points, or curves. For the feature detection, we proceed as follows:

1. Filter the surface to suppress the extremes and reduce the noise.

2. Compute maximal and minimum curvatures κ_1 and κ_2.

3. Find the vertices such that $|\kappa_1| < \tau_1$ and $|\kappa_2| > \tau_2$.

For the filtering, we choose an anisotropic smoothing filter suggested by Meyer *et al.*[63]. This reduces the number of edges falsely detected as feature, while preserves the characteristic features. The curvatures κ_1 and κ_2 are computed by using the method explained in [63] which offers relatively accurate values. The thresholds τ_1 and τ_2 are determined empirically, but it is relatively easy to predict the proper values, depending on what type of objects the original meshes coming from. Even though the feature vertices are found, we need to connect them into the feature curves, so that they do not cluster in thin areas. Let us denote the feature vertices $\mathcal{V}_{feature}$. The following procedure is used to ensure thin features.

1. Compute local maxima in $\mathcal{V}_{feature}$ using the curvature values.

2. Beginning from each local maxima, try a neighbor search within a threshold to find another local maxima. If found, we join them.

To ensure the process, the assumption is made that the sampling rate of the given mesh is relatively uniform. Otherwise, the length of each edge should be taken into consideration during the search process. Once the feature set is constructed, we partition the entire domain using the information. The procedure is as follows:

1. For each vertex, compute the distance function from the feature set. We employ a simplified version of multi-source Dijkstra algorithm.

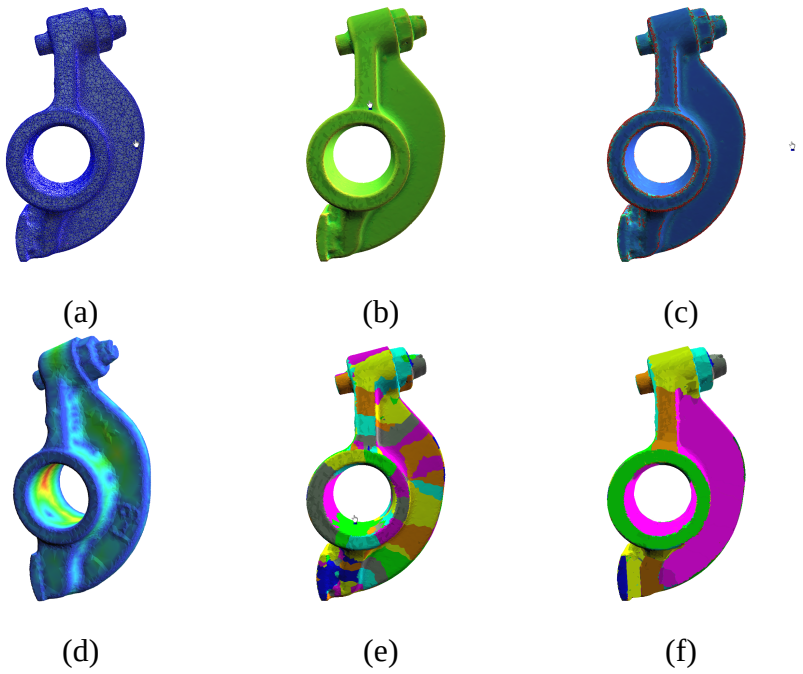
2. For each face, compute the distance function by averaging the values from its vertices.

3. Identify the local maxima among the faces.

4. From the maxima, grow the patches by adding 1-ring neighbors a time. The growing speed is governed by the curvature.

5. If two patches meet together and their intersections are far from the feature, we merge them.

We denote the partition as $\mathcal{P} = \{P_1, ..., P_n\}$ where P_i is each patch.



(a) (b) (c)

(d) (e) (f)

Figure 7.32: Feature Searching and Partitioning Procedure. (a) A rocker arm model. (b) Mean curvatures. (c) Found features. (d) Feature distance field. (e) The partitioning process in progress. (f) The final result.

7.5.3 Conformal Parameterization

As our surface has been segmented to patches which have the same topology as a disc or a unit square, the next target is to find the map from the patches to the unit square. The map will serve as a parameterization of each patch, where we perform the approximation later. Even though it is possible to choose arbitrary parameterization, we choose conformal

maps as our parameterization. A map is called *conformal* if it preserves local angles. For instance, if f is a conformal map from the unit square to 2-manifold in the space, then the images of iso-u and iso-v curves on the manifold are always mutually orthogonal. For each partition, we find a conformal map from the unit square as follows:

1. Find 4 fixed points for each patch from the intersection between the feature set and the boundary of the partition.

2. Compute a conformal map φ_i (or the inverse of it) from the patch P_i to the unit square by using the fixed points and the Least Square conformal map procedure [56]

We use an available Least Square Conformal Map solver based on [56]. The solver tries to minimize the harmonic energy and the results are reasonable if the patch does not contains much concave region on its boundary.

7.5.4 Boundary and Surface Fitting

The fitting procedures are as follows:

1. Insert initial control points on the boundary of the canonical domain of each patch. The number of control points are governed by overall curvature of the patch.

2. If two adjacent patches has different number of control points, we take the maximum.

3. Generate iso-curves and regular lattice on each canonical domain.

4. Map each lattice points back to the surface by the conformal maps.

5. Approximate the feature sets using Singular Value Decomposition.

6. Approximate each partition with the fixed control points that approximate the feature sets.

7.6 Volume Data Filtering

Most of existing volume data filtering employ tensor product of low dimensional filters. Due to their symmetry, tensor-product filters cannot preserve features. We have already exploited the power of non-tensor-product trivariate box spline in one of our subdivision schemes. Taking advantage of their asymmetry, we develop a family of anisotropic trivariate filters that can be used for feature preserving filtering, volume data refinement, and volume data fairing. We utilize the recursive power of the splines and compute the filters using the subdivision method for fast evaluation. Furthermore, by employing the box-spline based solid subdivision scheme, we develop a filtering technology for data over arbitrary tetrahedral meshes.

7.6.1 Trivariate Box Splines

We first define a trivariate box spline which serves as a convolution kernel for our filtering method. We use the trivariate double-directional box spline which is described in Chapter 4. The particular spline is spatially asymmetric as shown in Figure 4.5. Therefore, we can choose 4 different basis functions $M_{\Xi_1}, ..., M_{\Xi_4}$ such that the diagonal consist of the vertex 8 and 12 (Figure 4.6) matches with the direction of each of four major diagonals of the point surrounded by 8 voxels. The four major diagonal direction can be represented by the set of vectors $\{\mathbf{d}_1 = (1,1,1), \mathbf{d}_2 = (1,-1,1), \mathbf{d}_3 = (-1,1,1), \mathbf{d}_4 = (-1,-1,1)\}$. Our general filter has a form of:

$$F(\mathbf{x}) = \sum_{i=1}^{4} w_i M_{\Xi_i}, \tag{7.2}$$

where we need to decide the weight w_i by the gradient of each voxel.

7.6.2 Gradient Computation using Linear Regression

We require to compute a gradient for each voxel, to understand features and their directions. [68] suggested a general method to compute gradient field for volume data using simple linear regression. The size of the voxel neighbor which are used in the computation of the gradient field should coincide with the size of the filter that a user desires. Suppose $\mathbf{x}$ is a computed gradient vector, then we define the weight by computing the directional weight:

$$\varpi_i = \sqrt{1 - \frac{\mathbf{x} \cdot \mathbf{d}_i}{|\mathbf{x}||\mathbf{d}_i|}}, \tag{7.3}$$

and normalizing it:

$$w_i = \frac{\varpi_i}{\sum_j \varpi_j}. \tag{7.4}$$

7.6.3 Kernel Evaluation

Each basis function M_{Xi_i} of the filter can be easily evaluated by the subdivision process described in Chapter 4. Depending on the size of the kernel and the direction of each basis function, we apply the subdivision coefficients and the weight w_i to the appropriate number and direction of 3D convolution kernel. This process ensures the fast and rigorous evaluation of the filter function. Since the Linear Regression process can be computed by series of convolutions, we can actually achieve the filtering by convoluting appropriate kernels multiple times.

Chapter 8

Conclusion

In this dissertation, we present our recent research results, ongoing research and future research directions within the generalized framework of the solid subdivision method. Our proposed framework, Multiresolution Solid Objects, combines the advantages of current solid modeling representations, yet it provide a strong solution to the challenges that modern technology poses. Through the MSO, we can model, design, and manipulate arbitrary solid objects and multi-dimensional non-manifold objects. Underlying structured meshes based on simplicial complexes present great flexibility and simplicity during the geometric design. We have developed a series of novel solid subdivision schemes, which award the MSO such benefits that other existing representation technique cannot provide. Founded by sound mathematical theories, the subdivision schemes satisfy many desirable properties under ideal situations. By employing state-of-the-art mathematical tools, we have proved the analytical behavior of our solid subdivision schemes in practical cases.

In addition, we have implemented a variety of applications to demonstrate the potential of the MSO framework. Direct shape design, arbitrary manifold object modeling, non-manifold object modeling, boundary and feature representation, and heterogeneous material modeling are a few examples that we have presented to prove the advantages of the MSO in geometric modeling and shape design. We have also implemented free-form deformation and other applications that can draw interest of computer graphics community in

general.

For future research, we can pursue two separate, but closely related topics: theories and applications. Theoretically, the solid subdivision schemes have not been fully proved due to lack of adequate mathematical tools. These theoretical supports are essential not only to improve the existing schemes but also to derive new solid schemes. Therefore, intensive survey on current mathematical techniques are needed to make improvement on the MSO framework. For applications, it is necessary to develop a set of software systems geared up to meet practical needs, such as combination of modeling and analysis, layered and functionally graded material modeling, direct editing and trimming, and interactive design. In fact, the theoretical advances will be motivate by the practical needs, and the advances in theory will widen the application of our framework.

Bibliography

[1] N. Amenta, M. Bern, and M. Kamvysselis. A new Voronoi-based surface reconstruction algorithm. In *Computer Graphics (SIGGRAPH '98 Conference Proceedings)*, pages 415–421, 1998.

[2] P. R. Atherton. A scanline hidden surface removal procedure for constructive solid geometry. In *Computer Graphics (SIGGRAPH '83 Conference Proceedings)*, pages 73–82, 1983.

[3] J. A. Baerentzen and N. J. Christensen. Volume sculpting using the level-set method. In *Proceedings of Shape Modeling Interational 2002*, pages 175–182, 2002.

[4] C. Bajaj, J. Warren, and G. Xu. A subdivision scheme for hexahedral meshes. *The Visual Computer*, 18:343–356, 2002.

[5] A. A. Ball and D. J. T. Storry. A matrix approach to the analysis of recursively generated B-spline surfaces. *Computer-Aided Design*, 18(8):437–442, Oct. 1986.

[6] H. Biermann, I. M. Boier-Martin, D. Zorin, and F. Bernardini. Sharp features on multiresolution subdivision surfaces. In *Proceedings of Pacific Graphics 2001*, pages 140–149, Oct. 2001.

[7] H. Biermann, I. M. Boier-Martin, D. Zorin, and F. Bernardini. Sharp features on multiresolution subdivision surfaces. *Graphics Models*, 64(2):61–77, 2002.

[8] H. Biermann, A. Levin, and D. Zorin. Piecewise smooth subdivision surfaces with normal control. In *Computer Graphics (SIGGRAPH 2000 Conference Proceedings)*, pages 113–120, July 2000.

[9] J. F. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256, July 1982.

[10] W. Boehm. Subdividing multivariate splines. *Computer-Aided Design*, 15:345–352, Nov. 1983.

[11] W. Boehm. Calculating with box splines. *Computer Aided Geometric Design*, 1(2):149–162, 1984.

[12] W. Boehm, G. Farin, and J. Kahmann. A survey of curve and surface methods in CAGD. *Computer Aided Geometric Design*, 1:1–60, 1984.

[13] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. Society for Industrial and Applied Mathematics, July 2000.

[14] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10:350–355, Sept. 1978.

[15] G. Chaikin. An algorithm for high speed curve generation. *Computer Graphics and Image Processing*, 3:346–349, 1974.

[16] Y.-S. Chang, K. T. McDonnell, and H. Qin. A new solid subdivision scheme based on box splines. In *Proceedings of Solid Modeling 2002*, pages 226–233, 2002.

[17] Y.-S. Chang, K. T. McDonnell, and H. Qin. An interpolatory subdivision for volumetric models over simplicial complexes. In *Proceedings of Shape Modeling International 2003*, pages 143–152, May 2003.

[18] Y.-S. Chang and H. Qin. Multi-dimensional non-manifold subdivision over simplicial complexes. Submitted for review to Graphical Models.

[19] Y.-S. Chang and H. Qin. Spline-based solid subdivision schemes over arbitrary tetrahedral meshes. Submitted for review to Computer Aided Geometric Design.

[20] Y.-S. Chang and H. Qin. A framework for multi-dimensional adaptive subdivision objects. In *Proceedings of Solid Modeling 2004*, pages 123–134, 2004.

[21] Y.-S. Chang and H. Qin. Spline-based solid subdivision schemes over arbitrary tetrahedral meshes. Technical report, Stony Brook University, Department of Computer Science, 2004.

[22] C. K. Chui. *Multivariate Splines*. Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1988.

[23] P. Comninos. Fast bends or fast free-form deformation of polyhedral data. In *Computer Graphics '89*, pages 225–242, Nov. 1989.

[24] H. S. M. Coxeter. *Regular Polytopes*. The Macmillan Company, New York, 2nd edition, 1963.

[25] W. Dahmen and C. Micchelli. Subdivision algorithms for the generation of box spline surfaces. *Computer Aided Geometric Design*, 1(2):115–129, 1984.

[26] Dassault Systems. CATIA: PLM solution. *http://www.catia.com*.

[27] M. de Bert, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer, 1997.

[28] C. de Boor and K. Höllig. B-splines from parallelepipeds. *J. Analyse Math.*, 42:99–115, 1982.

[29] C. de Boor, K. Höllig, and S. Riemenschneider. *Box Splines*. Springer-Verlag, New York, 1993.

[30] T. DeRose, M. Kass, and T. Truong. Subdivision surfaces in character animation. In *Computer Graphics (SIGGRAPH '98 Conference Proceedings)*, pages 85–94, July 1998.

[31] D. Doo and M. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design*, 10(6):356–360, Sept. 1978.

[32] N. Dyn, J. Gregory, and D. Levin. Analysis of uniform binary subdivision schemes for curve design. *Constructive Approximation*, 7(2):127–148, 1991.

[33] N. Dyn, S. Hed, and D. Levin. Subdivision schemes for surface interpolation. In *Proceedings of the 1993 Workshop on Computational Geometry*, pages 97–118, 1993.

[34] N. Dyn, F. Kuijt, D. Levin, and R. V. Dame. Convexity preservation of the four point interpolatory subdivision scheme. *Computer Aided Geometric Design*, 16(8):789–792, 1999.

[35] N. Dyn, D. Levin, and J. Gregory. A four-point interpolatory subdivision scheme for curve design. *Computer Aided Geometric Design*, 4(4):257–268, 1987.

[36] N. Dyn, D. Levin, and J. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transactions on Graphics*, 9(2):160–169, Apr. 1990.

[37] N. Dyn, D. Levin, and D. Liu. Interpolatory convexity-preserving subdivision schemes for curves and surfaces. *Computer-Aided Design*, 24(4):211–216, Apr. 1992.

[38] N. Dyn and C. A. Micchelli. Using parameters to increase smoothness of curves and surfaces generated by subdivision. *Computer Aided Geometric Design*, 7:129–140, 1990.

[39] N. Foster and R. Fedkiw. Practical animation of liquids. In *Computer Graphics (SIGGRAPH 2001 Conference Proceedings)*, pages 23–30, 2001.

[40] R. B. Fuller. *Inventions, The Patented Works Of R. Buckminster Fuller*. St. Martin's Press, New York, 1983.

[41] T. A. Galyean and J. F. Hughes. Sculpting: An interactive volumetric modeling technique. In *Computer Graphics (SIGGRAPH '91 Conference Proceedings)*, pages 267–274, July 1991.

[42] General Electric Company. 3.0T Signa EXCITE HD. *http://www.gehealthcare.com/usen/mr*.

[43] A. Ghosh, Y. Miyamoto, and I. Reimanis. Functionally graded materials: Manufacture, properties and applications. *Ceramic Transactions*, 76, 1997.

[44] J. Greissmair and W. Purgathofer. Deformation of solids with trivariate B-splines. In *Computer Graphics Forum (Proceedings of Eurographics '89)*, pages 137–148, 1989.

[45] A. Habib and J. Warren. Edge and vertex insertion for a class of c^1 subdivision surfaces. *4th SIAM Conference on Geometric Design*, Nov. 1995.

[46] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. In *Computer Graphics (SIGGRAPH '94 Conference Proceedings)*, pages 295–302, 1994.

[47] J. O. Jocob E. Goodman, editor. *Handbook of Discrete and Computational Geometry*. CRC, New York, 1997.

[48] K. I. Joy and R. MacCracken. The refinement rules for Catmull-Clark solids. Technical Report CSE-96-1, Department of Computer Science, University of California, Davis, Feb. 1999.

[49] L. Kobbelt. Interpolatory subdivision on open quadrilateral nets with arbitrary topology. In *Computer Graphics Forum (Proceedings of Eurographics '96)*, pages 409–420, 1996.

[50] L. Kobbelt. A variational approach to subdivision. *Computer Aided Geometric Design*, 13(8):743–761, Nov. 1996.

[51] L. Kobbelt. $\sqrt{3}$-subdivision. In *Computer Graphics (SIGGRAPH 2000 Conference Proceedings)*, pages 103–112, 2000.

[52] L. Kobbelt and P. Schröder. A multiresolution framework for variational subdivision. *ACM Transactions on Graphics*, 17(4):209–237, 1998.

[53] F. Kuijt and R. van Damme. Convexity preserving interpolatory subdivision schemes. *Constructive Approximation*, 14:609–630, 1998.

[54] D. Lasser. Bernstein-bezier representation of volumes. *Computer Aided Geometric Design*, 2(1-3):145–150, 1985.

[55] A. Levin and D. Levin. Analysis of quasi uniform subdivision. *Applied and Computational Harmonic Analysis*, 15(1):18–32, 2003.

[56] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2002)*, 21(3):362–371, July 2002.

[57] L. Linsen, V. Pascucci, M. A. Duchaineau, B. Hamann, and K. I. Joy. Hierarchical representation of time-varying volume data with $sqrt[4]2$ subdivision and quadrilinear b-spline wavelets. In *Proceedings of Pacific Graphics 2002*, pages 346–355, 2002.

[58] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, Department of Mathematics, 1987.

[59] R. MacCracken and K. I. Joy. Free-Form deformations with lattices of arbitrary topology. In *Computer Graphics (SIGGRAPH '96 Conference Proceedings)*, pages 181–188, 1996.

[60] K. T. McDonnell, Y.-S. Chang, and H. Qin. Interpolatory, solid subdivision of unstructured hexahedral meshes. *The Visual Computer*, 20(6):418–436, Mar. 2004.

[61] K. T. McDonnell, Y.-S. Chang, and H. Qin. DigitalSculpture: A subdivision-based approach to interactive implicit surface modeling. *Graphical Models*, 67(4):347–369, July 2005.

[62] K. T. McDonnell and H. Qin. Dynamic sculpting and animation of free-form subdivision solids. In *Proceedings of IEEE Computer Animation 2000*, pages 126–133, May 2000.

[63] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *International Workshop on Visualization and Mathematics*, 2002.

[64] C. Micchelli and H. Prautzsch. Computing surfaces invariant under subdivision. *Computer Aided Geometric Design*, 4(4):321–328, 1987.

[65] Y. Miyamoto, W. A. Kaysser, and B. H. Rabin. *Functionally Graded Materials: Design, Processing and Applications*. Kluwer Academic, Boston, 1999.

[66] J. R. Munkres, editor. *Topology: a first course*. Prentice Hall, New Jersey, 1975.

[67] A. Nasri. Polyhedral subdivision methods for free-form surfaces. *ACM Transactions on Graphics*, 6:29–73, 1987.

[68] L. Neumann, B. Csébfalvi, A. König, and E. Gröller. Gradient estimation in volume data using 4d linear regression. In *Computer Graphics Forum (Proceedings of Eurographics 2000)*, pages 351–357, 2000.

[69] V. Pascucci. Slow growing subdivision (sgs) in any dimension: Towards removing the curse of dimensionality. *Computer Graphics Forum (Proceeding of Eurographics 2002)*, 21(3):451–460, Sept. 2002.

[70] A. Pasko and V.Savchenko. Algebraic sums for deformation of constructive solids. In *Proceedings of Solid Modeling '95*, pages 403–408, May 1995.

[71] A. A. Pasko and V. V. Savchenko. Blending operations for the functionally based constructive geometry. In *CSG 94 Set-theoretic Solid Modeling: Techniques and Applications*, pages 151–161, 1994.

[72] R. N. Perry and S. F. Frisken. Kizamu: A system for sculpting digital characters. In *Computer Graphics (Proceedings of SIGGRAPH 2001)*, pages 47–56, 2001.

[73] J. Peters and U. Reif. Analysis of generalized b-spline subdivision algorithms. *SIAM Journal of Numerical Analysis*, 1997.

[74] J. Peters and U. Reif. The simplest subdivision scheme for smoothing polyhedra. *ACM Transactions on Graphics*, 16(4):420–431, 1997.

[75] J. Peters and M. Wittman. Blending basic implicit shapes using trivariate box splines. In *The Mathematics of surfaces VII*, pages 409–426, 1997.

[76] J. Peters and M. Wittman. Box-spline based CSG blends. In *Proceedings of Solid Modeling '97*, pages 195–206, 1997.

[77] F. H. Post and F. Klok. Deformations of sweep objects in solid modelling. In *Computer Graphics Forum (Proceedings of Eurographics '86)*, pages 103–114, 1986.

[78] H. Prautzsch. Generalized subdivision and convergence. *Computer Aided Geometric Design*, 2(1-3):69–76, 1985.

[79] H. Prautzsch. Smoothness of subdivision surfaces at extraordinary points. *Advances in Computational Mathematics*, 9:377–390, 1998.

[80] H. Prautzsch and U. Reif. Necessary conditions for subdivision surfaces. Report 97/04, Sonderforschungsbereich 404, Universitat Stuttgart, 1997.

[81] H. Prautzsch and U. Reif. Degree estimates for C^k-piecewise polynomial subdivision surfaces. *Advances in Computational Mathematics*, 10(2):209–217, 1999.

[82] H. Prautzsch and G. Umlauf. A G^2-subdivision algorithm. In *Proceedings of the Dagstuhl conference on geometric modeling 1996, Computing suppl.*, pages 217–224, 1998.

[83] U. Reif. Some new results on subdivision algorithms for meshes of arbitrary topology. *Approximation Theory VIII*, 2:367–374, 1995.

[84] U. Reif. A unified approach to subdivision algorithms near extraordinary vertices. *Computer Aided Geometric Design*, 12:153–174, 1995.

[85] A. A. G. Requicha and H. B. Voelcker. Solid modeling: a historical summary and contemporary assessment. *IEEE Computer Graphics and Applications*, 2:9–23, Mar. 1982.

[86] F. Santos and R. Seidel. A better upper bound on the number of triangulations of a planar point set. *Journal of Combinatorial Theory, Series A*, 102(1):186–193, Apr. 2003.

[87] S. Schaefer, J. Hakenberg, and J. Warren. Smooth subdivision of tetrahedral meshes. In *Eurographics Symposium on Graphics Processing 2004*, pages 151–158, 2004.

[88] J. E. Schweitzer. *Analysis and Application of Subdivision Surfaces*. PhD thesis, University of Washington, Seattle, 1996.

[89] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. In *Computer Graphics (SIGGRAPH '86 Conference Proceedings)*, pages 151–160, 1986.

[90] P. Shirley and A. Tuchman. A polygonal approximation to direct scalar volume rendering. In *Computer Graphics (San Diego Workshop on Volume Visualization)*, volume 24, pages 63–70, Nov. 1990.

[91] J. Stam. Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values. In *Computer Graphics (SIGGRAPH '98 Conference Proceedings)*, pages 395–404, 1998.

[92] L. Velho. 4-8 factorization of quadrilateral subdivision. In *Proceedings of Solid Modeling 2001*, page 303, 2001.

[93] S. W. Wang and A. E. Kaufman. Volume sculpting. In *Proceedings of the 1995 Symposium on Interactive 3D Graphics*, pages 151–156, Apr. 1995.

[94] J. Warren. Subdivision methods for geometric design. Unpublished manuscript, Nov. 1995.

[95] J. Warren and H. Weimer. Variational subdivision for natural cubic splines. *Approximation Theory IX*, 2:345–352, 1998.

[96] J. Warren and H. Weimer. A cookbook for variational subdivision. *SIGGRAPH '99 Course Notes on "Subdivision for Modeling and Animation"*, 1999.

[97] J. Warren and H. Weimer. *Subdivision Methods for Geometric Design: A constructive Approach*. Morgan Kaufmann Publisher, 2001.

[98] J. R. Weeks, editor. *The Shape of Space*. Warcel Dekker, 1985.

[99] H. Weimer and J. Warren. Subdivision schemes for thin plate splines. In *Computer Graphics Forum (Proceedings of Eurographics '98)*, pages 303–313, 1998.

[100] H. Weimer and J. Warren. Subdivision schemes for fluid flow. In *Computer Graphics (SIGGRAPH '99 Conference Proceedings)*, pages 111–120, Aug. 1999.

[101] B. Wyvill, C. McPheeters, and G. Wyvill. Animating soft objects. *The Visual Computer*, 2(4):235–242, 1986.

[102] L. Ying and D. Zorin. Nonmanifold subdivision. In *Proceedings of IEEE Visualization 2001*, pages 325–332, 2001.

[103] J. Zagajac. A fast method for estimating discrete field values in early engineering design. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):35–43, Mar. 1996.

[104] D. Zorin. *Subdivision and Multiresolution Surface Representations*. PhD thesis, Caltech, Pasadena, 1997.

[105] D. Zorin. Smoothness of stationary subdivision on irregular meshes. *Constructive Approximation*, 16:359–398, 2000.

[106] D. Zorin and D. Kristjansson. Evaluation of piecewise smooth subdivision surfaces. *The Visual Computer*, 18:299–315, 2002.

[107] D. Zorin and P. Schröder. Subdivision for modeling and animation. In *SIGGRAPH 2000 Course Notes*, 2000.

[108] D. Zorin, P. Schröder, and W. Sweldens. Interpolating subdivision for meshes with arbitrary topology. In *Computer Graphics (SIGGRAPH '96 Conference Proceedings)*, pages 189–192, 1996.

Appendix A

Subdivision Coefficients for Box-Spline Based Subdivision Scheme

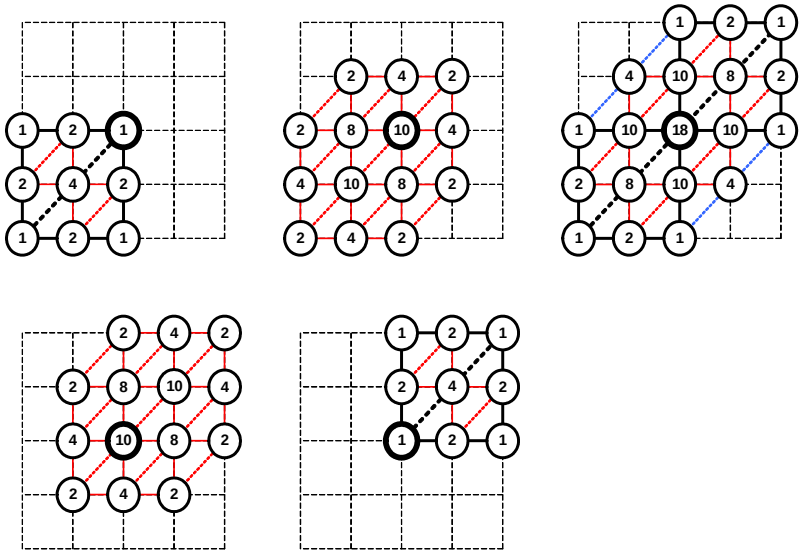


Figure A.1: The subdivision coefficients of the box-spline based solid subdivisionare embedded in the part of the octet-truss regular structured mesh. All the values are to be multiplied by $\frac{1}{32}$. Since it is 3D mesh, each layer of the mesh is displayed separately. The bold circled vertex indicates the center of the mesh.

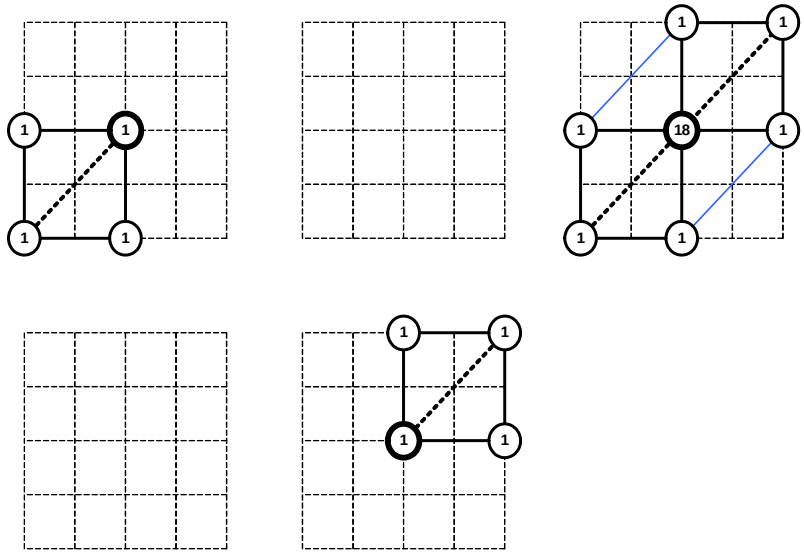


Figure A.2: The vertex mask for the box-spline based solid subdivision.

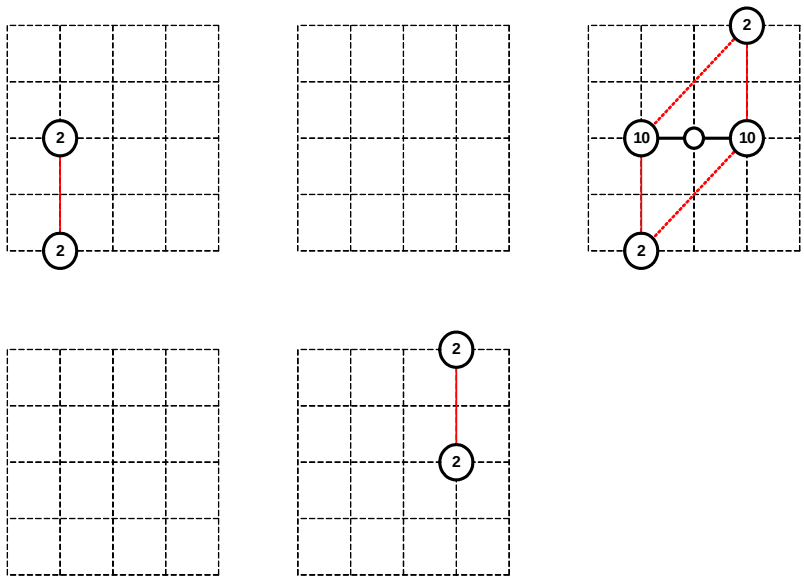


Figure A.3: An edge mask for the box-spline based solid subdivision (a).

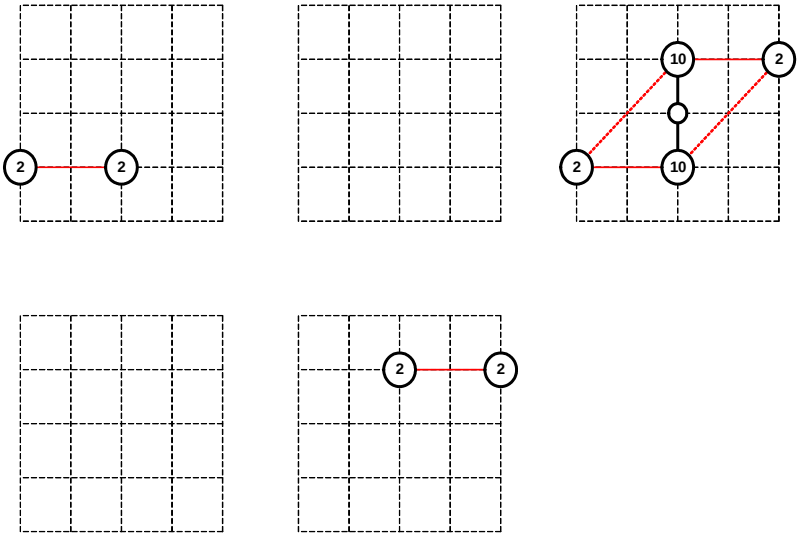


Figure A.4: An edge mask for the box-spline based solid subdivision (b).

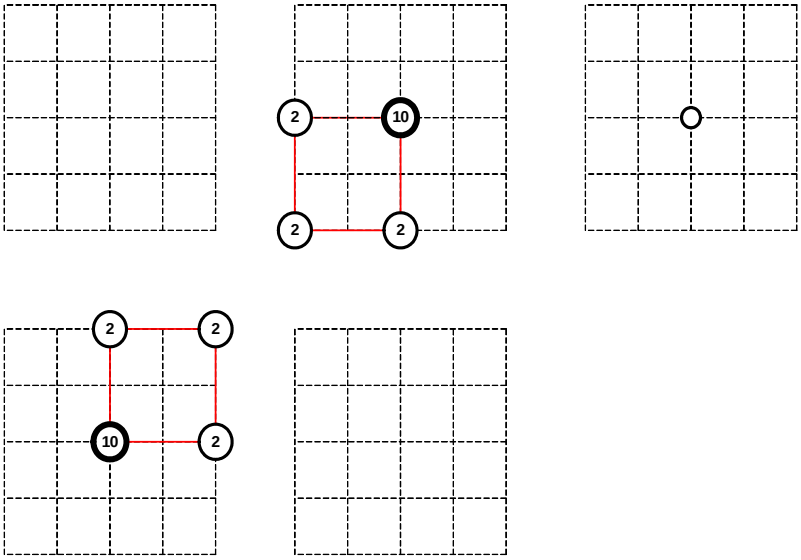


Figure A.5: An edge mask for the box-spline based solid subdivision (c).

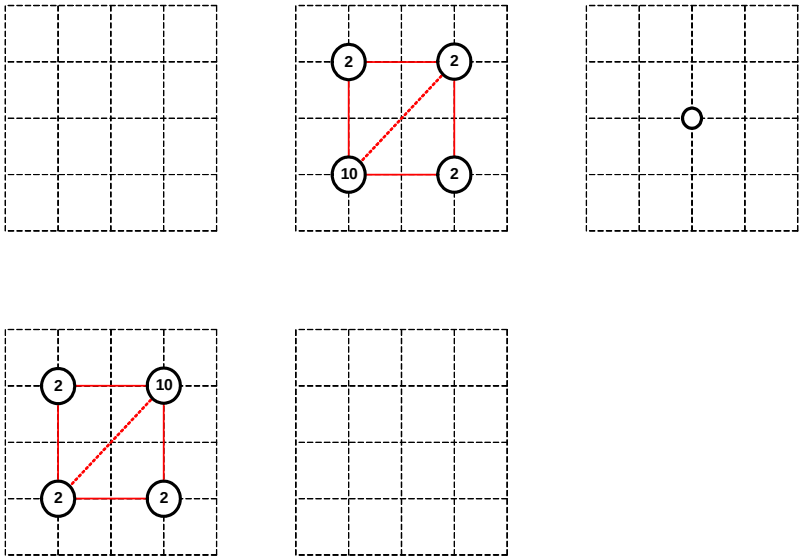


Figure A.6: An edge masks for the box-spline based solid subdivision (d).

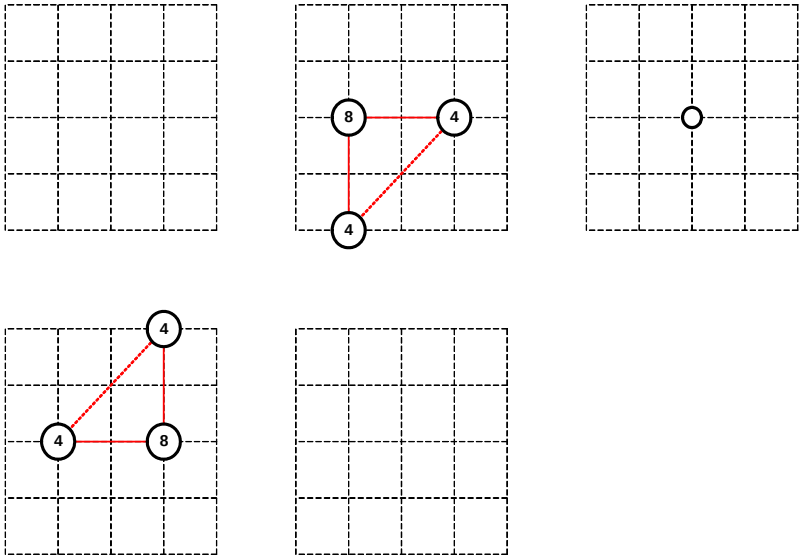


Figure A.7: An edge masks for the box-spline based solid subdivision (e).

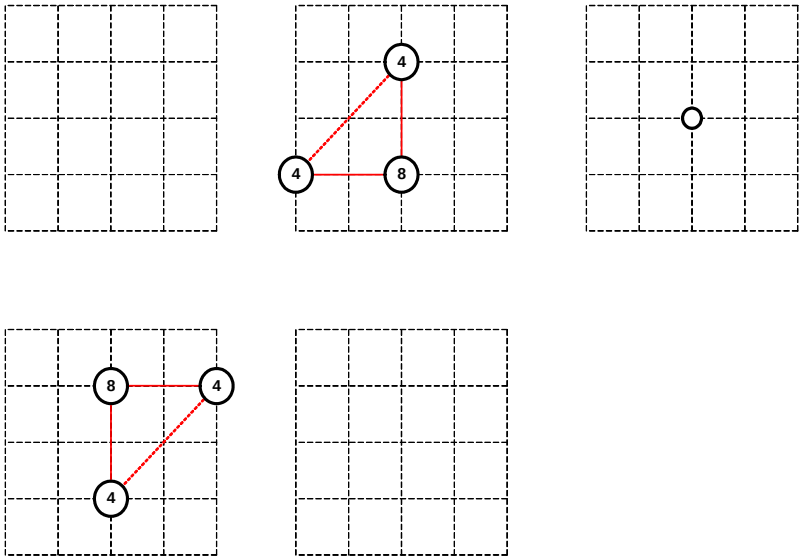


Figure A.8: An edge masks for the box-spline based solid subdivision (f).

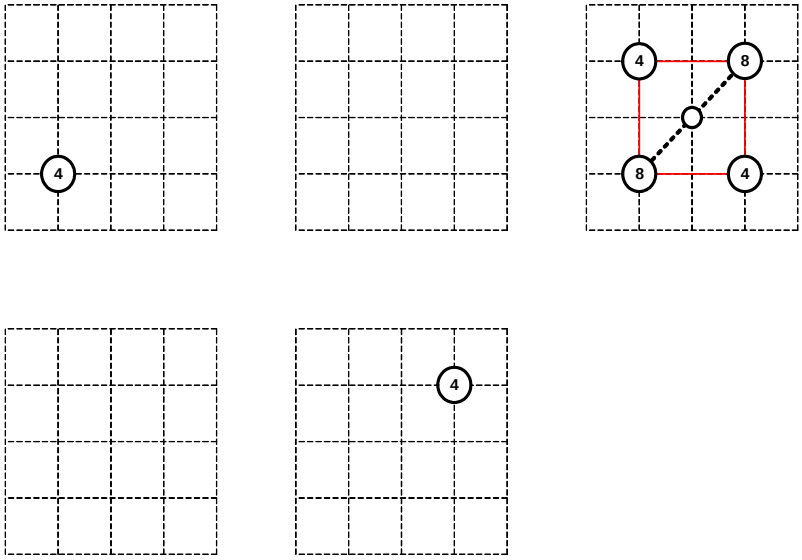


Figure A.9: The cell masks for the box-spline based solid subdivision.

Appendix B

Samples of Solid Subdivision Matrices

In this chapter, we demonstrate explicit forms of the subdivision matrices of the box-spline based solid subdivision scheme in three particular cases. Note that different configurations of neighbors yield different eigenstructure of the system.

B.1 Extraordinary Vertex Subdivision Matrix

The matrix $\mathbf{S}_v$ shown in Figure B.1 is the subdivision matrix for an extraordinary vertex with the valence 5. The size of the matrix is 20×20. The subdivision matrices for extraordinary vertices with different valences have the similar structure. The first 6 eigenvalues of the matrix $\mathbf{S}_v$ can be computed by the eigenvalues of the top-left 6×6 submatrix. The values are:

$$\{1, \frac{5}{16}, \frac{15+\sqrt{70}}{80}, \frac{3}{20}, \frac{1}{8}, \frac{1}{8}\}.$$

B.2 Extraordinary Edge Subdivision Matrix

The matrix $\mathbf{S}_e$ in Figure B.2 is the subdivision matrix for an extraordinary edge with the valence 4. The size of the matrix is 19×19. The major diagonals are chosen in the similar way to the first case of Figure 4.20. The subdivision matrices for extraordinary vertices with different valences have the similar structure. The first 11 eigenvalues computed by the

top-left 11×11 submatrix is:

$$\{1, \frac{45+\sqrt{985}}{160}, \frac{25+6\sqrt{2}}{80}, \frac{25+6\sqrt{2}}{80}, \frac{19}{80}, \frac{25-6\sqrt{2}}{80}, \frac{25-6\sqrt{2}}{80}, \frac{3}{16}, \frac{13}{80}, \frac{13}{80}, \frac{1}{8}\}.$$

The different choices of the major diagonals gives a different subdivision matrix and the eigenstructure. If we choose all the diagonals toward the extraordinary edge, we acquire a different subdivision matrix as shown in Figure B.3. In this case, there is no simple re-ordering of the vertices in $\mathbf{p}^0$ that gives the block structure. However, we can still compute all the first 11 eigenvalues from $\mathbf{S}'_e$:

$$\{1, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{25+\sqrt{385}}{160}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{8}\}.$$

As previously mentioned, the difference between the eigenvalues of the subdivision matrices due to the choice of the major diagonals are subtle if the edge valence is low. However, if the edge valence becomes higher, in particular over 8, the difference in the eigenstructure of the subdivision matrices become larger.

$$\mathbf{S}_v = \begin{pmatrix}
\frac{9}{16} & \frac{7}{80} & \frac{7}{80} & \frac{7}{80} & \frac{7}{80} & \frac{7}{80} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & \frac{5}{16} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & \frac{1}{8} & \frac{1}{4} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & \frac{1}{8} & \frac{1}{8} & \frac{1}{4} & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{4} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & 0 & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{5}{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{7}{128} & \frac{9}{16} & \frac{7}{128} & \frac{7}{128} & \frac{7}{128} & 0 & \frac{7}{128} & \frac{7}{128} & \frac{7}{128} & \frac{7}{128} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{16} & \frac{5}{16} & \frac{5}{16} & \frac{1}{16} & \frac{1}{16} & 0 & 0 & \frac{1}{16} & 0 & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{16} & \frac{5}{16} & \frac{1}{16} & \frac{5}{16} & \frac{1}{16} & 0 & 0 & 0 & \frac{1}{16} & 0 & 0 & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{16} & \frac{5}{16} & \frac{1}{16} & \frac{1}{16} & \frac{5}{16} & 0 & 0 & 0 & 0 & \frac{1}{16} & \frac{1}{16} & 0 & 0 & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 & 0 \\
\frac{3}{40} & \frac{3}{40} & \frac{5}{16} & 0 & \frac{5}{16} & \frac{3}{40} & 0 & 0 & 0 & 0 & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & \frac{3}{40} & 0 & 0 & 0 \\
\frac{7}{160} & \frac{7}{160} & \frac{9}{16} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & 0 & \frac{7}{160} & 0 & 0 & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & 0 & 0 & 0 & \frac{7}{160} & 0 & 0 & 0 \\
\frac{3}{40} & \frac{3}{40} & \frac{5}{16} & \frac{5}{16} & 0 & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{40} & 0 & 0 & 0 & 0 & \frac{3}{40} & 0 & 0 \\
\frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{9}{16} & \frac{7}{160} & \frac{7}{160} & 0 & 0 & \frac{7}{160} & 0 & 0 & 0 & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & 0 & 0 & \frac{7}{160} & 0 & 0 \\
\frac{3}{40} & \frac{3}{40} & 0 & \frac{5}{16} & \frac{5}{16} & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{40} & 0 & 0 & 0 & \frac{3}{40} & 0 \\
\frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{9}{16} & \frac{7}{160} & 0 & 0 & 0 & \frac{7}{160} & \frac{7}{160} & 0 & 0 & 0 & \frac{7}{160} & \frac{7}{160} & 0 & 0 & \frac{7}{160} & 0 \\
\frac{3}{40} & 0 & \frac{5}{16} & \frac{3}{40} & \frac{3}{40} & \frac{5}{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{40} & \frac{3}{40} & 0 & 0 \\
\frac{3}{40} & 0 & \frac{3}{40} & \frac{5}{16} & \frac{3}{40} & \frac{5}{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{40} & \frac{3}{40} & 0 \\
\frac{3}{40} & 0 & \frac{3}{40} & \frac{3}{40} & \frac{5}{16} & \frac{5}{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{40} & 0 & \frac{3}{40} & 0 \\
\frac{7}{128} & 0 & \frac{7}{128} & \frac{7}{128} & \frac{7}{128} & \frac{9}{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{7}{128} & \frac{7}{128} & \frac{7}{128} & \frac{7}{128}
\end{pmatrix}$$

Figure B.1: The subdivision matrix for an extraordinary vertex with the valence 5.

$$\mathbf{S}_e = \begin{pmatrix}
\frac{9}{16} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & \frac{3}{40} & 0 & \frac{5}{16} & \frac{3}{40} & 0 & \frac{3}{40} & \frac{3}{40} & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & \frac{3}{40} & 0 & \frac{3}{40} & \frac{5}{16} & \frac{3}{40} & 0 & 0 & \frac{3}{40} & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & \frac{3}{40} & 0 & 0 & \frac{3}{40} & \frac{5}{16} & \frac{3}{40} & 0 & 0 & \frac{3}{40} & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & \frac{3}{40} & 0 & \frac{3}{40} & 0 & \frac{3}{40} & \frac{5}{16} & \frac{3}{40} & 0 & 0 & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & 0 & \frac{3}{40} & \frac{3}{40} & 0 & 0 & \frac{3}{40} & \frac{5}{16} & \frac{3}{40} & 0 & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & 0 & \frac{3}{40} & \frac{3}{40} & \frac{3}{40} & 0 & 0 & \frac{3}{40} & \frac{5}{16} & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & 0 & \frac{3}{40} & 0 & \frac{3}{40} & \frac{3}{40} & 0 & 0 & \frac{3}{40} & \frac{5}{16} & \frac{3}{40} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & 0 & \frac{3}{40} & 0 & 0 & \frac{3}{40} & \frac{3}{40} & \frac{3}{40} & 0 & \frac{3}{40} & \frac{5}{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{16} & 0 & 0 & \frac{5}{16} & 0 & 0 & \frac{1}{16} & \frac{5}{16} & \frac{1}{16} & 0 & 0 & \frac{1}{16} & \frac{1}{16} & 0 & 0 & 0 & 0 & 0 & \frac{1}{16} \\
\frac{1}{8} & 0 & 0 & \frac{1}{4} & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{1}{4} & 0 & 0 & 0 & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{16} & 0 & 0 & \frac{1}{16} & \frac{5}{16} & 0 & 0 & 0 & \frac{5}{16} & \frac{1}{16} & 0 & 0 & \frac{1}{16} & \frac{1}{16} & \frac{1}{16} & 0 & 0 & 0 & 0 \\
\frac{1}{8} & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{1}{4} & 0 & 0 & 0 & 0 & \frac{1}{8} & 0 & 0 & 0 & 0 \\
\frac{1}{16} & 0 & 0 & 0 & \frac{1}{16} & \frac{5}{16} & 0 & 0 & 0 & \frac{5}{16} & \frac{1}{16} & 0 & 0 & 0 & \frac{1}{16} & \frac{1}{16} & \frac{1}{16} & 0 & 0 \\
\frac{1}{8} & 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & \frac{1}{8} & 0 & 0 \\
\frac{1}{16} & 0 & 0 & 0 & 0 & \frac{1}{16} & \frac{5}{16} & \frac{1}{16} & 0 & 0 & \frac{5}{16} & 0 & 0 & 0 & 0 & 0 & \frac{1}{16} & \frac{1}{16} & \frac{1}{16} \\
\frac{1}{8} & 0 & 0 & \frac{1}{8} & 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{8}
\end{pmatrix}$$

Figure B.2: The subdivision matrix for an extraordinary edge with the valence 4 (a).

$$\mathbf{S}'_e = \begin{pmatrix}
\frac{9}{16} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & \frac{7}{160} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{5}{16} & \frac{1}{16} & 0 & \frac{5}{16} & \frac{1}{16} & 0 & \frac{1}{16} & \frac{1}{16} & 0 & 0 & 0 & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 & 0 & \frac{1}{16} \\
\frac{5}{16} & \frac{1}{16} & 0 & \frac{1}{16} & \frac{5}{16} & \frac{1}{16} & 0 & 0 & \frac{1}{16} & 0 & 0 & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 \\
\frac{5}{16} & \frac{1}{16} & 0 & 0 & \frac{1}{16} & \frac{5}{16} & \frac{1}{16} & 0 & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & 0 \\
\frac{5}{16} & \frac{1}{16} & 0 & \frac{1}{16} & 0 & \frac{1}{16} & \frac{5}{16} & 0 & 0 & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 & 0 & \frac{1}{16} & 0 & \frac{1}{16} \\
\frac{5}{16} & 0 & \frac{1}{16} & \frac{1}{16} & 0 & 0 & 0 & \frac{5}{16} & \frac{1}{16} & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 & 0 & \frac{1}{16} \\
\frac{5}{16} & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & 0 & \frac{1}{16} & \frac{5}{16} & \frac{1}{16} & 0 & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & 0 & 0 & 0 \\
\frac{5}{16} & 0 & \frac{1}{16} & 0 & 0 & \frac{1}{16} & 0 & 0 & \frac{1}{16} & \frac{5}{16} & \frac{1}{16} & 0 & 0 & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & 0 \\
\frac{5}{16} & 0 & \frac{1}{16} & 0 & 0 & 0 & \frac{1}{16} & \frac{1}{16} & 0 & \frac{1}{16} & \frac{5}{16} & 0 & 0 & 0 & 0 & 0 & \frac{1}{16} & 0 & \frac{1}{16} \\
\frac{1}{8} & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & \frac{1}{8} \\
\frac{1}{4} & 0 & 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{8} & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 \\
\frac{1}{4} & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\
\frac{1}{8} & 0 & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & 0 & 0 \\
\frac{1}{4} & 0 & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 \\
\frac{1}{8} & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} \\
\frac{1}{4} & 0 & 0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{4}
\end{pmatrix}$$

Figure B.3: The subdivision matrix for an extraordinary edge with the valence 4 (b).

Appendix C

Interpolatory Solid Subdivision Scheme over Unstructured Hexahedral Meshes

We propose and derive a novel, interpolatory solid subdivision scheme over unstructured hexahedral meshes. The scheme is derived from a generalization of tri-cubic Lagrange interpolating polynomials to arbitrary topologies.

The interpolatory solid subdivision scheme over unstructured hexahedral meshes was introduced in a recent paper published in the Visual Computer [60]. The scheme was employed to implement DigitalSculpture which is accepted for publication in Graphical Models [61]. The contents of the Appendix C.1 were graciously provided by Kevin T. McDonnell and were taken directly from [60].

C.1 Derivation of Subdivision Scheme

C.1.1 Rules for Meshes of Regular Topology

First we present subdivision rules for regular hexahedral meshes, in which each vertex in the mesh has valence six. In Section C.1.2 we generalize the rules to handle topologically non-regular hexahedral meshes.

C.1.1.1 Cell-Point Rule

The cell-point subdivision mask for our scheme is obtained by computing the different naive scheme's cell-masks for the four possible orientations of the main diagonal in a cell. Adding these masks and normalizing the weights produces the averaged mask. The cell-point rule is:

$$c_p = \frac{6w+1}{8}\sum_{i=1}^{8} p_i - \frac{w}{4}\sum_{i=1}^{24} q_i. \tag{C.1}$$

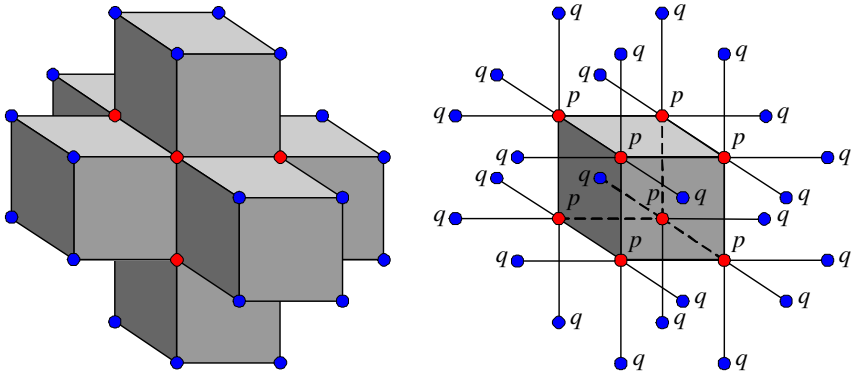


Figure C.1: Cell-point mask for our new subdivision scheme. The weights for the p-vertices and q-vertices are, respectively: $w_p = \frac{6w+1}{8}$, $w_q = -\frac{w}{4}$. The opaque rendering on the left shows the cellular structure of the mask.

C.1.1.2 Face-Point Rule

Averaging the six naive scheme's face-point masks produces the averaged mask seen in Figure C.2. The face-point rule is:

$$f_p = \frac{2w+1}{4}\sum_{i=1}^{4} p_i + \frac{w}{4}\sum_{i=1}^{8} q_i - \frac{w}{4}\sum_{i=1}^{8} r_i - \frac{w}{8}\sum_{i=1}^{16} s_i. \tag{C.2}$$

C.1.1.3 Edge-Point Rule

Averaging the 12 naive scheme's edge-point masks produces the averaged mask seen in Figure C.3. Note that many of the terms cancel each other, which results in a mask with

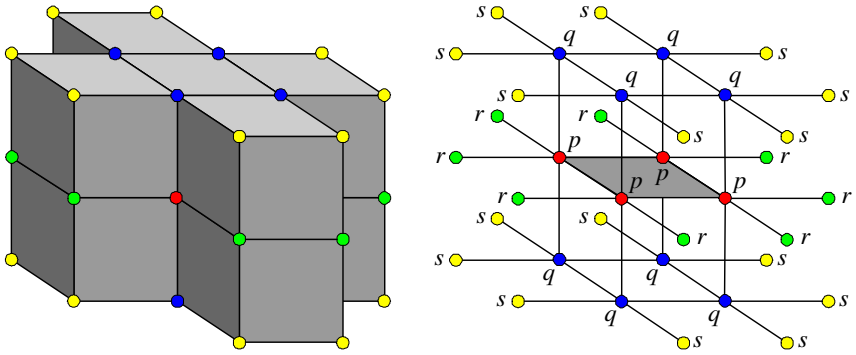


Figure C.2: Face-point mask for our new subdivision scheme. The weights for the p-vertices, q-vertices, r-vertices and s-vertices are, respectively: $w_p = \frac{2w+1}{4}$, $w_q = \frac{w}{4}$, $w_r = -\frac{w}{4}$, $w_s = -\frac{w}{8}$.

fewer vertices than one might expect. The edge-point rule is:

$$e_p = \frac{1}{2}\sum_{i=1}^{2} p_i + \frac{w}{4}\sum_{i=1}^{8} q_i - \frac{w}{4}\sum_{i=1}^{8} r_i. \tag{C.3}$$

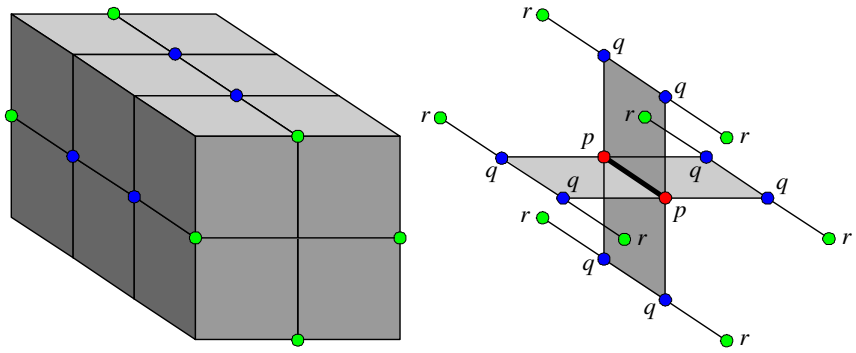


Figure C.3: Edge-point mask for our new subdivision scheme. The weights for the p-vertices, q-vertices and r-vertices are, respectively: $w_p = \frac{1}{2}$, $w_q = \frac{w}{4}$, $w_r = -\frac{w}{4}$. The cancelling of terms in the derivation leads to a mask of smaller size than expected.

C.1.2 Rules for Meshes of Non-Regular Topology

As we show shortly, our new subdivision scheme is very amenable to application over arbitrary hexahedral meshes since it does not depend on a particular ordering of the vertices or choice of cell diagonal. The rules must be generalized to handle hexahedral meshes

that feature *extraordinary edges* (*i.e.*, edges with greater than four or less than four adjacent faces) and *extraordinary vertices* (*i.e.*, vertices not of valence six). Fortunately, the rules can be generalized easily to handle these non-regular topological conditions. We now investigate how the rules must be changed to accommodate such situations.

C.1.2.1 Cell-Point Rule

In order to define subdivision rules and masks that can be applied over non-regular hexahedral meshes, we must examine the subdivision masks and determine which vertices, if any, are *shared* by cells that define the mask. We must also determine if the duplication or absence of shared vertices in a non-regular mesh causes the subdivision masks to be ill-defined. For instance, inspection of the cell-point mask in Figure C.1 reveals that the vertices adjacent to the cell itself can be identified *solely* by locating those vertices 1-adjacent to the cell's vertices. That is, the subdivision mask is obtained by taking vertices from the cell itself, as well as the cells *face-adjacent* to the cell. This means that Equation C.1 can be used to compute a cell-point even when the neighboring connectivity is very complicated. Hence, the cell-point subdivision mask does not require modification in order to handle arbitrary hexahedral meshes.

C.1.2.2 Face-Point Rule

Unlike the cell-point rule, the averaged scheme's face-point rule *does* require modification in order to handle extraordinary edges. As illustrated in Figure C.2, there are several vertices labelled r in the middle of the mask that are shared by adjacent cells that define the mask. Under a regular topological setting, there are eight such r-vertices, two per edge. If one or more of the edges connecting these vertices is extraordinary, which is the case in Figure C.4, then a special rule is required to handle the extra vertices introduced into (or subtracted from) the mask. Note that vertices labelled q, although shared by adjoining cells,

can be uniquely identified by obtaining them from the two cells that meet at the face itself. Since we assume that each face has at most two adjacent cells, there is never a problem in locating these eight q-vertices.

As we mentioned, there are eight r-vertices in the regular case of the face-point mask. Irregularities arise when one or more of the vertices p is extraordinary. Specifically, for each edge $\overline{p_i p_j}$ that is adjacent to more than four cells, the weight of each such r-vertex (now indicated by t in Figure C.4) becomes $-\frac{w}{8}$. Note that there will be at most two such vertices per end-point per edge since each face is shared by at most two cells (Figure C.4). The subdivision formula for the face-point is therefore modified as follows:

$$\widehat{f}_p = \frac{2w+1}{4}\sum_{i=1}^{4} p_i + \frac{w}{4}\sum_{i=1}^{8} q_i - \frac{w}{4}\sum_{i=1}^{8-2N} r_i - \frac{w}{8}\sum_{i=1}^{16} s_i - \frac{w}{8}\sum_{i=1}^{4N} t_i$$

where N indicates the number of extraordinary edges in the face. It is easy to confirm that this rule reduces to Equation C.2 when the mesh is regular.

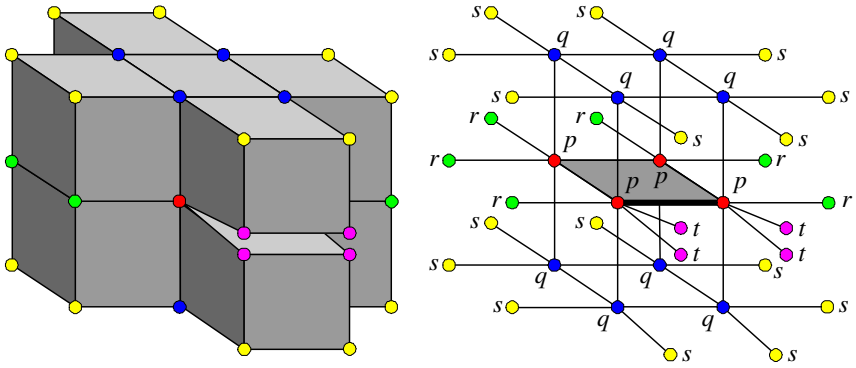


Figure C.4: Face-point mask for extraordinary vertices and edges. A mask containing two extraordinary vertices is shown here. The edge drawn in bold is an extraordinary edge. The weights for the p-vertices, q-vertices, r-vertices, s-vertices and t-vertices are, respectively: $w_p = \frac{2w+1}{4}$, $w_q = \frac{w}{4}$, $w_r = -\frac{w}{4}$, $w_s = -\frac{w}{8}$, $w_t = -\frac{w}{8}$.

C.1.2.3 Edge-Point Rule

The averaged scheme's edge-point mask must also be modified in order to handle two possible irregularities: 1. more than (or less than) four faces are incident on the edge (Figure C.5a); and/or 2. one or more adjacent edges is extraordinary (Figure C.5b). The first situation arises when one or both of the edge's end-points are extraordinary vertices. The second circumstance appears when a vertex normally shared by adjacent cells (*i.e.*, an r-vertex in Figure C.3) is replaced by two vertices because of a local topological irregularity. We call this kind of vertex a "split" vertex. (Recall that a similar situation can occur with the face-point mask when an extraordinary edge causes a normally shared vertex to be replaced by two distinct vertices; see Figure C.4.) In the regular case of four incident faces, each p-vertex receives a weight of $\frac{w}{4}$ and each r-vertex receives $-\frac{w}{4}$ (Figure C.3). For the general case of N incident faces, each p_i is given weight $\frac{w}{N}$, while each r_i receives $-\frac{w}{N}$. Figure C.5a illustrates an edge with five incident faces and the resulting vertex weights. In Figure C.5b we see that the edge drawn in bold in the middle of the mask has a non-regular number of adjacent edges, which causes the weights of split vertices (labelled s) to change from $-\frac{w}{4}$ to $-\frac{w}{8}$. In the general case of N incident faces, this weight for a split vertex is $-\frac{w}{2N}$. These two non-regular topological conditions are subsumed by the following modified edge-point formula:

$$\widehat{e}_p = \frac{1}{2}\sum_{i=1}^{2} p_i + \frac{w}{N}\sum_{i=1}^{2N} q_i - \frac{w}{N}\sum_{i=1}^{2N-M} r_i - \frac{w}{2N}\sum_{i=1}^{2M} s_i \tag{C.4}$$

where N is the number of faces incident on the edge, and M is the number of extraordinary edges incident on the p_i's that appear in the mask (*i.e.*, the number of "split" vertices). Note that Equation C.4 reduces to Equation C.3 for regular meshes (*i.e.*, $N = 4$, $M = 0$).

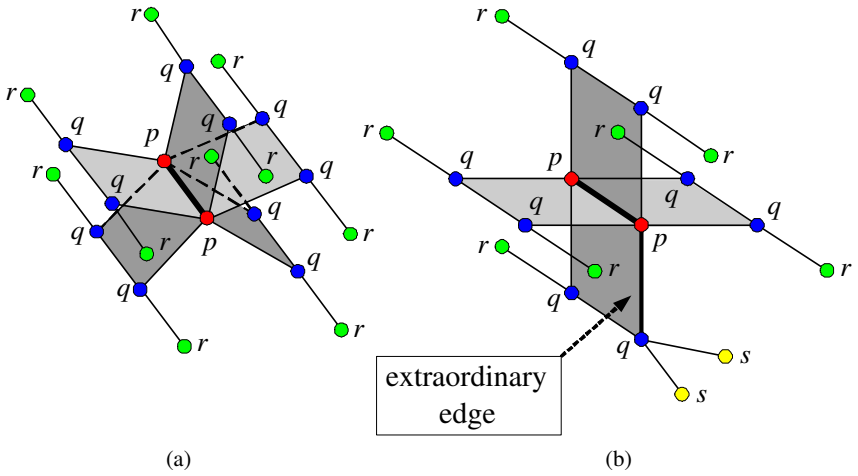


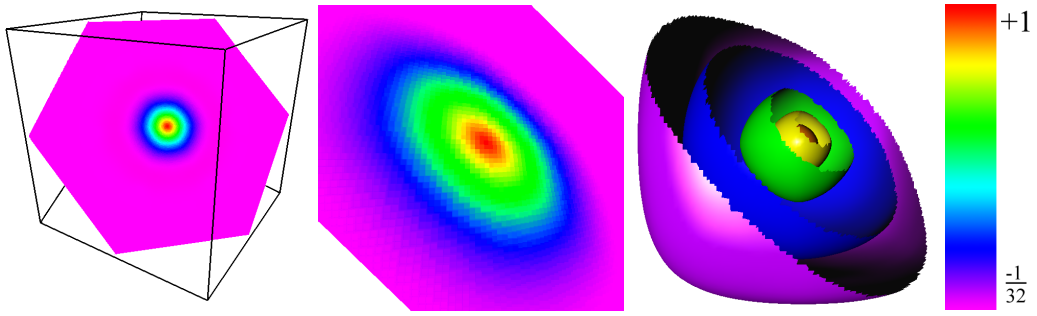
Figure C.5: Edge-point masks for non-regular topological settings. (a) Extraordinary edge. The weights for the p-vertices, q-vertices and r-vertices are, respectively: $w_p = \frac{1}{2}$, $w_q = \frac{w}{5}$, $w_r = -\frac{w}{5}$. (b) Regular edge containing one extraordinary vertex. The weights for the p-vertices, q-vertices, r-vertices and s-vertices are, respectively: $w_p = \frac{1}{2}$, $w_q = \frac{w}{4}$, $w_r = -\frac{w}{4}$, $w_s = -\frac{w}{8}$.

C.2 Analysis of Subdivision Scheme

C.2.1 Convergence and Continuity for Meshes of Regular Topology

We will use techniques explained by Dyn *et al.* [38] to prove that our subdivision scheme is C^1 continuous over hexahedral meshes of regular topology. Since our subdivision scheme has no closed-form expression for its basis functions (Figure C.6), we cannot simply extract the basis functions and examine them analytically. Therefore, we rely on analysis of subdivision matrices and characteristic functions to study the scheme's convergence and continuity properties. By showing that the characteristic polynomials of the subdivision process have certain properties, we will demonstrate that the algorithm generates volumes that are C^1 in the limit.

Note that a subdivision algorithm can be expressed in matrix form as $\mathbf{p}^{k+1} = \mathbf{S}\mathbf{p}^k$, where $\mathbf{p}^k$ is the vector of points at subdivision level k, $\mathbf{S}$ is the local subdivision mask, and $\mathbf{p}^{k+1}$ is the resulting vector of new points. Generally, any binary stationary subdivision



(a) (b) (c)

Figure C.6: Renderings of the subdivision algorithm's basis function. The scale indicates the contribution of a vertex to its neighbors. The basis function was computed explicitly via subdivision. In (b) and (c) we have changed the viewpoint to highlight better the structure of the basis function.

scheme for solids can be written as

$$P_{k+1}(z) = a(z)P_k(z^2), \qquad z \in \mathbb{R}^3, \tag{C.5}$$

where $P_k(z) = \sum_{\mu\in\mathbb{Z}^3} p_\mu^k z^\mu$ is a formal generating function associated with the control points $\mathbf{p}^k = \{p_\mu^k\}_{\mu\in\mathbb{Z}^3}$ at the level k, and $a(z)$ is the characteristic polynomial derived from the local subdivision matrix $\mathbf{S}$:

$$a(z) = \sum_{\mu\in\mathbb{Z}^3} a_\mu z^\mu. \tag{C.6}$$

Theorems 5.1 and 5.2 provide us the sufficient conditions to guarantee the C^1 continuity of the subdivision scheme. The conditions for the norm of the matrix will be confirmed by means of the relation explained in Equation 5.12. During most of the process, we will rely on numerical experiments to verify the satisfaction of the conditions.

The characteristic polynomial of the subdivision scheme can be computed by successive applications of the schemes over a regular mesh in 3D. It has the form of

$$a(z) = \sum_{\mu\in\mathbb{Z}^3} a_\mu z^\mu, \tag{C.7}$$

where the coefficients are given by

$$
\begin{aligned}
a_\mu &= \frac{6w+1}{8}, && \mu = (\pm 1, \pm 1, \pm 1) \\
a_\mu &= -\frac{w}{4}, && \mu = (\pm 1, \pm 1, \pm 3) \\
a_\mu &= \frac{2w+1}{4}, && \mu = (0, \pm 1, \pm 1) \\
a_\mu &= \frac{1}{4}, && \mu = (\pm 1, \pm 1, \pm 2) \\
a_\mu &= -\frac{w}{4}, && \mu = (0, \pm 1, \pm 3) \\
a_\mu &= -\frac{w}{8}, && \mu = (\pm 1, \pm 2, \pm 3) \\
a_\mu &= \frac{1}{2}, && \mu = (0, 0, \pm 1) \\
a_\mu &= \frac{w}{4}, && \mu = (0, \pm 1, \pm 2) \\
a_\mu &= -\frac{w}{4}, && \mu = (0, \pm 2, \pm 3) \\
a_{\mu'} &= a_\mu, && \text{if} \quad \mu' = \sigma(\mu), \quad \sigma \in S_3.
\end{aligned}
$$

Here, S_3 denotes the set of all permutation over $\{1, 2, 3\}$, which is followed by the symmetry of the subdivision mask.

It is relatively easy to confirm that the scheme is convergent by means of eigenvalue analysis of the subdivision matrix. In particular, the subdominant eigenvalue of our subdivision scheme is strictly less than 1 for $w < 0.5$, which is sufficient to show its convergence. The characteristic polynomial of the subdivision scheme can be factored by $(1 + z_1)^2(1 + z_2)^2(1 + z_3)^2$. Therefore, it can be written as

$$
a(z) = \frac{1}{2}(1 + z_1)^2(1 + z_2)^2(1 + z_3)^2 q(z, w), \tag{C.8}
$$

where $q(z, w)$ is a Laurent polynomial with respect to z for a given weight w.

Now, we employ Theorems 5.1 and 5.2 for the proof of C^1 continuity. A close inspection of the theorems reveals that, for the scheme to be C^1 continuous, it is sufficient to show

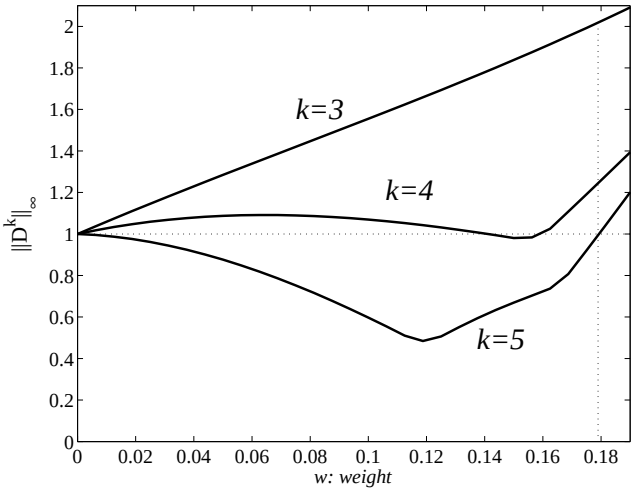


Figure C.7: A graph of $||D^k||_\infty$ with respect to the weight value w.

that $||D^L_{(i_1,i_2)}||_\infty < 1$ for some L where

$$D_{(i_1,i_2)}(z) = 2(1+z_{i_1})^{-1}(1+z_{i_2})^{-1}\, a(z).$$

Because $a(z)$ is invariant of a permutation on indices, it is equivalent to show that $||D^L_{(1,1)}|| < 1$ and $||D^L_{(1,2)}|| < 1$, where

$$D_{(1,1)}(z) = 2(1+z_1)^{-2}\, a(z),$$
$$D_{(1,2)}(z) = 2(1+z_1)^{-1}(1+z_2)^{-1}\, a(z),$$

respectively.

If we let $||D^k||_\infty = \max_{(i_1,i_2)} ||D^k_{(i_1,i_2)}||_\infty$, then when $w = \frac{1}{16}$, the norms are

$$||D^1||_\infty = 1.75,$$
$$||D^2||_\infty \simeq 1.5313,$$
$$||D^3||_\infty \simeq 1.3523,$$
$$||D^4||_\infty \simeq 1.0914,$$
$$||D^5||_\infty \simeq 0.8188.$$

Figure C.7 shows $||D^k||_\infty$ as a function of the weight w. Generally, $||D^L||_\infty < 1$ when $L \geq 5$, at least for $w \in (0, 0.1787]$. By the theorems, we can guarantee that, when w is within the range, the subdivision scheme is C^1 continuous over regular hexahedral meshes.

C.2.2 Continuity for Meshes of Non-Regular Topology

Non-regular topologies include cases in which a vertex or an edge has non-standard connectivity. In regular hexahedral meshes, each vertex has a valence of six and each edge is shared by four adjacent faces. When the mesh does not have these properties, we say that mesh has an *extraordinary vertex* or *extraordinary edge*. Because non-regular topologies become isolated from each other during the subdivision process, we can assume that we have only a finite number of extraordinary cases in any given mesh.

For subdivision surface schemes, eigenanalysis is the standard technique to prove the continuity of the scheme around non-regular topologies. In a surface scheme analysis, extraordinary vertices are the only kinds of special cases that we have to consider. However, solid scheme analysis involves not only the analysis of extraordinary vertices, but also that of extraordinary edges. Unlike the relatively simple surface cases, both of the extraordinary cases in solid schemes lack planar symmetry (in general). This situation prohibits a direct application of spectral analysis techniques such as the Discrete Fourier Transform, which is often employed in eigenanalysis to compute eigenvalues and eigenvectors of the subdivision matrix symbolically (see [31]).

To overcome these difficulties, we have computed eigenvalues and eigenvectors of the subdivision matrix around the extraordinary cases numerically. Obviously, it is not possible to acquire the proof of the general cases in this way. We have selected over 20 extraordinary cases and have analyzed their eigenvalues and eigenvectors to verify the necessary conditions for the convergence of the scheme around the extraordinary cases. For the selected cases we have also have numerically performed characteristic map analysis [84], which

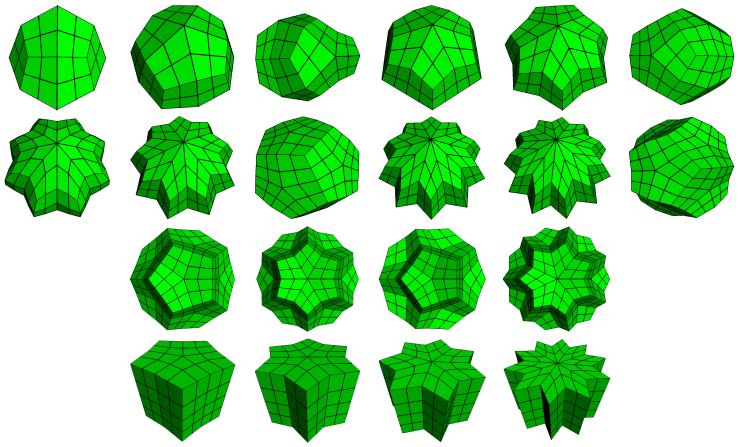


Figure C.8: A selection of non-regular topology meshes we analyzed in order to prove numerically that our scheme is C^1 continuous. The last four meshes show cases of extraordinary edges.

is a well-understood technique for surface subdivision analysis. Reif [84] proved that the regularity (one-to-one and non-singular aspect) of the characteristic maps guarantee the C^1 continuity of a subdivision scheme around extraordinary cases in the limit.

For all the cases of extraordinary topology we examined, a local subdivision matrix $\mathbf{S}$ of the scheme satisfied an eigenvalue property of

$$\lambda_0 = 1 \gneqq \lambda_1 \geq \lambda_2 \geq \lambda_3 \gneqq \lambda_4, \ldots, \lambda_n \tag{C.9}$$

where λ_i's are the eigenvalues of $\mathbf{S}$ in decreasing order. The eigenvalues of the selected cases are listed in Tables C.1 and C.2. The weight for the scheme is given as $w = \frac{1}{16}$ in all cases. It is important to mention that we have 3 subdominant eigenvalues which are strictly less than 1. In addition, Figure C.8 shows the control nets of the characteristic map from each of the subdominant eigenvectors. It is important to note that, although the extraordinary vertex cases we have chosen for each valence do not represent all the possible configurations, they constitute a broad selection of the topologies one might encounter in practice.

Valence	λ_0	λ_1	λ_2	λ_3	λ_4
5	1	0.499420	0.454695	0.454695	0.352565
7	1	0.500000	0.484286	0.484286	0.404687
8	1	0.465107	0.458056	0.455922	0.447308
8	1	0.500000	0.470715	0.470715	0.461119
9	1	0.500000	0.495818	0.495818	0.439426
10	1	0.475022	0.475022	0.459298	0.445021
10	1	0.500000	0.500000	0.500000	0.432938
11	1	0.500000	0.494283	0.494283	0.470715
12	1	0.459298	0.459298	0.459298	0.445021
12	1	0.500000	0.490543	0.490543	0.484286
13	1	0.500000	0.498844	0.498844	0.472785
14	1	0.470722	0.470722	0.459298	0.445021
17	1	0.474916	0.462847	0.462847	0.448390
20	1	0.474916	0.463449	0.463449	0.452999
22	1	0.470597	0.459411	0.459411	0.457600
23	1	0.474916	0.465931	0.465931	0.465700

Table C.1: Eigenvalues for a selection of the extraordinary vertex cases.

Face Number	λ_0	λ_1	λ_2	λ_3	λ_4
5	1	0.5	0.484286	0.484286	0.404687
6	1	0.5	0.470715	0.470715	0.461119
7	1	0.5	0.495818	0.495818	0.439426
10	1	0.5	0.490543	0.490543	0.484286

Table C.2: Eigenvalues for a selection of the extraordinary edge cases.