# Discrete Differential Form Subdivision and Vector Field Generation over Volumetric Domain

Thesis by

Jinghao Huang

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2012

(Defended June 5, 2012)

# Acknowledgements

First I would like to thank my mentor, Peter Schröder, for his consistent guidance, support and encouragement. In the past four years, I have been inspired so much by his instruction. I truly believe that, in my future career, I will continue to benefit from what I have learned from Peter's professionalism and enthusiasm.

I would like to thank the other committee members, Michael Aivazis, Oscar Bruno and Mathieu Desbrun, for their time and their valuable suggestions and advice on this thesis.

I would like to thank Michael Aivazis for helpful discussions about the implementation of the algorithm. I am also grateful to Alex Gittens for his proofreading of the thesis, advice on Mathematica and many other joyful discussions.

I am grateful to Yao Sha, Jie Cheng, Hong Zhong and other senior students who helped me to get used to the new life when I had just come to this country. Also, I would like to thank all my friends who made my graduate school years very enjoyable. I especially want to express my gratefulness to Xiao Liu who supported every aspect of my life.

Finally, I would like to express my most sincere gratitude to my family for their everlasting love and support.

This research is mainly supported by the King Abdullah Scholar Award. The supported visit to Jeddah in 2008 was very delightful and enriching. I am grateful to KAUST and the award committee for their generosity and their foresight in my field.

# Abstract

This thesis presents a new method to construct smooth l- and 2-form subdivision schemes over the 3D volumetric domain. Based on the subdivided 1- and 2-form coefficient field, smooth vector fields can be constructed using Whitney forms. To obtain stencils in the regular setting, classical 0-form subdivision and linear 1- and 2-form subdivision over the octet mesh are introduced. Then, convoluting with a smooth operator, smooth 1- and 2-form subdivision schemes in the regular case can be determined up to one free parameter. This parameter can be determined by a novel technique based on spectrum and momentum considerations. However, artifacts exist in boundary regions because of the incomplete regular support and the shrinking feature of the original 0-form subdivision scheme. To address these problems, the projection-scaling method and the expansion method are introduced and compared. The former method projects arbitrary discrete differential forms to a subspace spanned by low-order potential fields. The algorithm subdivides these potential fields and reconstructs the discrete form in the refined level using linear combinations. Scaling is included for elements near the boundary to offset the effect of mesh shrinkage. Alternatively, for the expansion method, a compatible nonshrinking 0-form subdivision scheme is constructed first. Based on the new 0-form subdivision method, extending 1- and 2-forms beyond the boundary becomes natural. In the experiment, no noticeable artifacts, including attenuation, enlarging or undesirable bend, are found in practice.

# Contents

# List of Figures

# Chapter 1

# Introduction

Subdivision surfaces are standard tools for modeling free-form surfaces in the computer graphics industry [15, 34, 44, 46] because of their flexibility and adaptability to smooth surfaces with complicated topology. In this thesis we will extend the subdivision to the volumetric domain, i.e., we are dealing with tetrahedral meshes instead of surfaces. Because of the complexity of the volumetric domain problem, in this section, we first review the mathematical background of surface subdivision schemes in Section 1.1. Section 1.2 reviews the basic ideas, definitions and notations in discrete differential geometry (DDG) that are used extensively throughout later chapters. Using DDG, we introduce an edge-based (1-form) subdivision scheme in Section 1.3. Section 1.4 reviews basic concepts and previous works in the area of 3D volumetric domain subdivision. Section 1.5 summarizes the structure of the thesis.

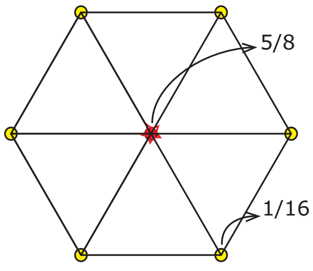
## 1.1 Classical Vertex-Based Subdivision on Surfaces

In computer graphics, a smooth surface is usually represented by a piecewise-linear polygonal mesh. A smoother representation can be obtained by recursively subdividing the facets of the coarse mesh into several finer facets. Given the initial mesh $M^0$, the subdivision process can be defined through the recursive equation

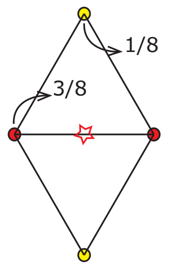
$$\mathcal{M}^{k+1} = S\mathcal{M}^k, \ \ k \geq 0. \tag{1.1}$$

A good subdivision algorithm associated with the linear operator $S$ leads to a smooth surface $\mathcal{M}^\infty$ in the limit as $k \to \infty$ in the form of a sequence of piecewise-linear meshes. Usually a surface subdivision process can be factorized into a topological splitting process followed by a geometric smoothing process that is usually in the form of a weighted averaging of all vertices in a finite support. Such a factorization is also available for the edge-/facet-based subdivision and the 3D volumetric domain subdivision we discuss in later chapters.

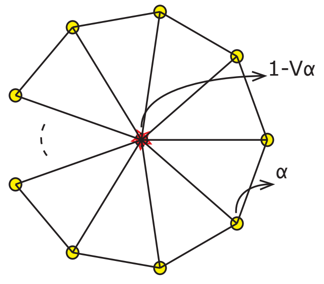
The above-mentioned subdivision algorithms were proposed for the construction of smooth free-form surfaces but have found applications well beyond geometric modeling [20, 26]. Some of these schemes are formulated for meshes with data living at either vertices, e.g., positions, or faces, e.g., colors [9, 18, 27, 28, 33, 36]. The former schemes are called *primal schemes* while the latter are called *dual schemes*. There are other ways to classify subdivision schemes. For example, schemes come in flavors distinguished by their topological splitting rules, e.g., based on triangles or quadrilaterals ([9, 18] vs. [28]) or based on dyadic or more exotic splitting ([28] vs [27]); and by their geometric smoothing rules, e.g., piecewise linear or higher order.



(1) Even vertex, regular



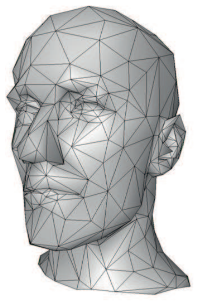
(2) Odd vertex

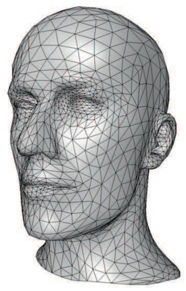


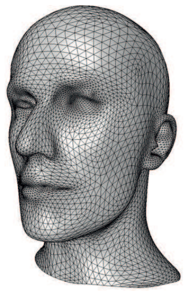
(3) Even vertex, extraordinary

Figure 1.1: Subdivision stencils for Loop's scheme. Old vertices with different colors have different contributions to the target vertex (the star).

Local geometric smoothing rules of a subdivision scheme can be represented by a subdivision stencil. As an example, Fig. 1.1 shows subdivision stencils for the Loop subdivision scheme. Note that the stencils for even vertices (vertices from the coarse mesh) and odd vertices (newly inserted vertices) are different. Further, the stencil also depends on local topological configurations such as valence ($\alpha$ in the figure is a constant number which solely depends on the valence). Fig. 1.2 shows the process to subdivide mannequin data using Loop's scheme.


(1) $\mathcal{M}^0$ (initial mesh)


(2) $\mathcal{M}^1$


(3) $\mathcal{M}^2$

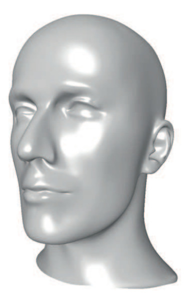

(4) $\mathcal{M}^\infty$

Figure 1.2: Subdividing mannequin data using Loop's scheme

**Subdivision Matrix** For a finite subdivision scheme (i.e., the stencil has a compact support), a useful tool for surface subdivisions is the *subdivision matrix* that relates the control points in a fixed neighborhood domain of the coarse and refined meshes. Take Loop's scheme as an example. As shown in Fig. 1.3, the scheme's support is restricted in two rings around a vertex (the *control set*) in the coarse mesh, the subdivision operation $S$ in Eq.(1.1) maps the local mesh into a finer mesh with identical local topological structure (i.e., the $k$th ring in the coarse mesh corresponds to the $k$th ring in the refined mesh (as shown by different colors) and the indices of the vertices can also be perfectly matched up) and can be represented by a $(3v+1) \times (3v+1)$-matrix where $v$ is the valence of the central vertex. The matrix element $S_{ij}$ represents the contribution of the $i$th vertex in the coarse mesh ($p_i^k$) on the $j$th vertex in the refined mesh ($p_j^{k+1}$).

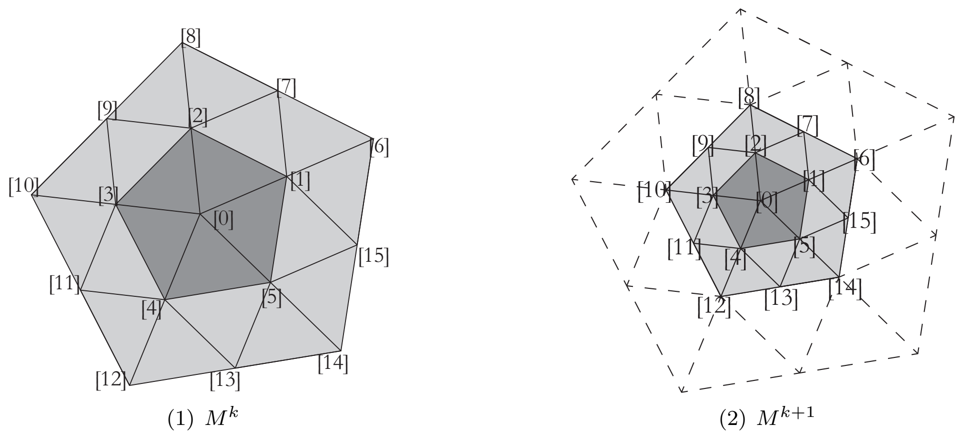


Figure 1.3: A finite subdivision operation can preserve the topological structures in a fixed neighborhood. Here we applied the subdivision on a mesh around a vertex with valence 5.

Because of symmetries in the local topological structure, the subdivision matrix has circulant symmetric features. For example, the contribution of $p_1^k$ on $p_7^{k+1}$ is the same as the contribution of $p_2^k$ on $p_9^{k+1}$ and $p_1^k \dots p_5^k$ have the same contribution on $p_0^{k+1}$. For simplicity, we can define a equivalence relation $\sim_r$ with respect to the central vertex $v_0$ such that $v_i \sim_r v_j$ if and only if we can transform $v_i$ to $v_j$ by a rotation around $v_0$ while all other topological structures are preserved. Under this equivalence relation, the vertices in the shaded neighborhood in Fig. 1.3 can be classified into four classes. Equivalently, the subdivision matrix can be divided into a $(4 \times 4)$-block matrix in which each block is circulant symmetric.

## 1.2 Discrete Differential Geometry

Differential forms are a fundamental concept in the mathematical fields of differential topology and tensor analysis. They facilitate an intrinsic approach to multivariable calculus that is independent of coordinates. We will come back to this coordinate-free feature many times throughout this thesis. The concept of differential forms and the formulation of exterior algebra using wedge products and exterior derivatives were both introduced by Cartan [8]. Using these tools, classical operations such as integration can all be reformulated in a coordinate-free way; this helps expose the intrinsic geometric features and the topological invariants of problems.

One of the most important motivations for the discretization of the above theoretical framework is the popularization of digital computers. Many works have discussed how to use purely computational techniques such as finite difference to discretize differential geometries. However, most of these approaches have at least two drawbacks. First, the results depend on coordinates, or the metric space in which the geometric objects are embedded, so intrinsic topological properties become obscured. Second, most of these methods are based on numerical approximations. Therefore, many important global and invariant features of the geometric objects are lost during the discretization process.

As shown in Chapter 3, a more intrinsic discretization approach (*discrete exterior calculus*, or DEC) can be introduced, based on combinatorial and topological analogues of many concepts and operations in continuous differential geometry. Under this approach, topological manipulations (coordinate-free) and geometric processing (embedding into a metric space) are kept separated. For example, in the aforementioned classical subdivision case, this separation becomes the factorization into the topological splitting step and the geometric smoothing step. Such separation becomes more clear when we come to the 1- and 2-form subdivision cases later. More details can be found in classical algebraic topology books [21, 30].

*Differential forms* are one of the core concepts in modern differential geometry. Informally speaking, a *k-form* can be thought of as an entity ready to be integrated on a $k$-dimensional region [1, 14, 16]. In other words, in the continuous domain, a $k$-form defines a linear mapping from a $k$-dimensional region on a manifold $\mathcal{K}$ to $\mathbb{R}$. Formally, a $k$-form on a manifold $\mathcal{K}$ locally defines an antisymmetric multilinear map from the product of tangent spaces $\wedge_{i=1}^{k} T_p\mathcal{K}$ to $\mathbb{R}$. It is a smooth section of the $k$-th power of the cotangent bundle [14, 37, 40]. Obviously, the set of all $k$-forms on a manifold forms a linear space, often denoted as $\Sigma^k(\mathcal{K})$. The restriction of such multilinear maps to the tangent space of a submanifold induces differential forms on that submanifold. As we see below, such feature simplify our calculations when we construct discrete form subdivision stencils.

The concepts above have perfect analogues in the discrete domain where the "manifold" is now a mesh $\mathcal{M}$. The role of integration in the continuous domain is now played by the evaluation of a discrete form on a *chain*, which is essentially a weighted sum of simplices. The integrands, discrete differential forms, or *cochains*, are sets of scalar values associated with each simplex in $\mathcal{M}$. Cochains are the discrete analogues of differential forms [16].

An interesting questions is: given discretized differential forms associated with $\mathcal{M}$, can we interpolate a continuous differential forms from this data? The answer is yes: this interpolation task is accomplished through *Whitney forms* [4, 5, 45]. Note that a continuous 0-form is just the value of a function in the continuous domain, so the discretized 0-form is simply the value of that function restricted to the vertices. The Whitney 0-forms are the vertex-based interpolation basis (i.e., hat functions) defined as $\phi_i = 1$ at vertex $v_i$ and 0 at all the other vertices. The linear combination of hat functions yields a linear interpolation of the discrete 0-form. For higher-order forms, reconstruction can be accomplished through a similar approach using Whitney forms of the corresponding order. A detailed discussion can be found in Section 3.2.

Another building block of modern differential geometry [12, 17, 40], the exterior derivative d, generalizes the classical derivative of a function to higher-order differential forms. Specifically, $\mathrm{d} : \Sigma^k(\mathcal{K}) \to \Sigma^{k+1}(\mathcal{K}),\ f \mapsto \mathrm{d}f$ is a linear map that satisfies the following 3 conditions: (1) If $f$ is a smooth function (i.e., a 0-form), then $\mathrm{d}f$ is the classical derivative $f'$ (i.e., a 1-form); (2) $\mathrm{d}(\mathrm{d}f) = 0$ (this is called the *Poincaré lemma*); (3) If $f$ is a $p$-form and $g$ is a $q$-form, then $\mathrm{d}(f \wedge g) = \mathrm{d}f \wedge g + (-1)^p f \wedge \mathrm{d}g$. These formula can be used to define the exterior derivative of higher-order differential forms. More specifically, if the local coordinate chart of the manifold is defined as $(x^1, \cdots, x^n)$, then for a $k$-form, we have $\mathrm{d}(f\ \mathrm{d}x^{j_1} \wedge \cdots \wedge \mathrm{d}x^{j_k}) = \sum_{i=1}^{n} (\partial f / \partial x^i)\ \mathrm{d}x^i \wedge \mathrm{d}x^{j_1} \wedge \cdots \wedge \mathrm{d}x^{j_k}$. In $\mathbb{R}^3$, using the exterior derivative, we can express the usual vector calculus operators:

$$\text{Gradient: } \nabla = \mathrm{d}^0, \quad \text{Curl: } \nabla\times = \mathrm{d}^1, \quad \text{Divergence: } \nabla\cdot = \mathrm{d}^2. \tag{1.2}$$

As discussed in Chapter 3, while the discrete exterior derivative operator has some of the same features as these classical operators, it has a much more succinct interpretation.

## 1.3 Edge-Based Subdivision on Surface

Unlike the above-mentioned primal or dual schemes, subdivision schemes for data living on the edges of a mesh were not investigated until recently [43]. Under this framework, scalar coefficients on directed edges in the coarse mesh are linearly combined to give the scalar coefficients on directed edges in the refined mesh. We can apply these edge-based schemes to discrete 1-forms which, as mentioned in Section 3.1.2, are essentially the discrete analogues of vector fields. A sequence of everywhere-defined differential 1-forms can be reversely constructed using a suitable interpolation method (e.g., the Whitney forms) from these scalar coefficients. This sequence can be identified with tangent vector fields [43]. In the finite elements context, the scalar coefficient field can also be interpreted as giving rise to *edge elements*, which can be essential in the numerical solution of certain partial differential equations [2, 5]. In this way, edge-based subdivision schemes provide new construction methods for hierarchically refinable edge elements. We will present the theoretical framework of the edge-based subdivision schemes in Chapter 3.

In the surface case, the 1-form subdivision scheme [43] was first constructed based on 0-form subdivision using Loop's scheme [28]. A 1-form scheme based on Catmull-Clark subdivision can be constructed using the similar approach [42]. In these approaches, the stencil of the 1-form subdivision is derived by specifying the *commutative relations* ($d^0S_0 = S_1d^0,\ d^1S_1 = S_2d^1$, see Section 3.4 for details). The intuition behind commutative relations is that taking the discrete exterior derivative on the subdivided mesh should yield the identical results to taking it first and then subdividing. If the 0-form scheme (e.g., Loop or Catmull-Clark subdivision) and 2-form scheme (e.g., half-box spline) are fixed, the 1-form subdivision scheme is uniquely determined. Similar to the 0-form subdivision, for 1- and 2-form subdivisions, geometric smoothing rules can also be represented using local stencils (Fig. 1.4).

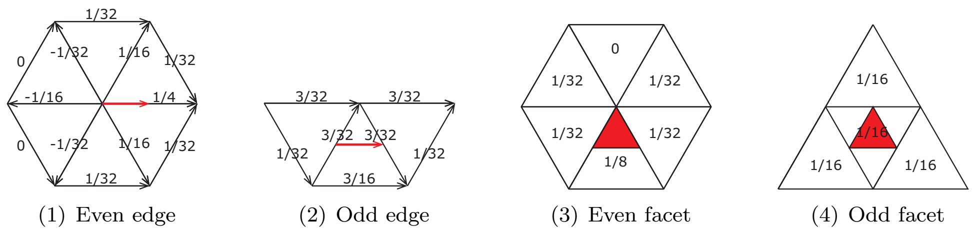


Figure 1.4: Stencils for 1- and 2-form subdivisions (only regular cases are shown here)

The construction of a 1-form scheme using $\sqrt{3}$-subdivision is more complicated [24]. In this work, no 2-form subdivision scheme is specified in advance. As the commutative relation between the 0- and 1-form subdivision does not uniquely determine the 1-form stencil, the authors developed a new approach based on spectral and momentum considerations to uniquely fix the 1-form stencil. Such analytical tools proved to be useful in volumetric subdivision cases addressed below.

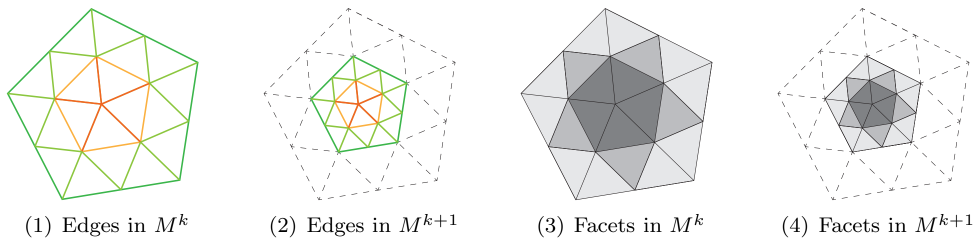


Figure 1.5: A finite subdivision operation can preserve the topological structures (including edges and facets) in a fixed neighborhood.

*Remark:* Similar to the 0-form subdivision, 1- and 2-form subdivision operations can be written in the form of Eq.(1.1) where $\mathcal{M}^k$ is the form coefficient field instead of vertices coordinates. The subdivision matrix for 1- and 2-forms defines a mapping links a fixed neighborhood of edges and facets in two sequential subdivision levels (Fig. 1.5). We can introduce similar equivalence relations among edges and facets such that the subdivision matrix can be written as a block matrix in which each block has a circulant symmetric structure.

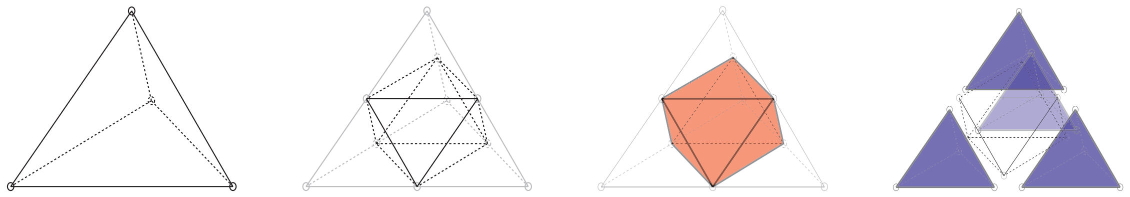
## 1.4 Subdivision in the Volumetric Domain

The definition of smoothness in the volume domain is not as straight forward as in the surface case (Section 1.1). Here we have adopted the deformation-based definition of smoothness [38, 39]. Assume the original mesh $M^0$ has $|\Gamma|$ control points, where $\Gamma$ is the index set. Consider the mesh sequence $\mathcal{M}^1, \mathcal{M}^2, \cdots, \mathcal{M}^\infty$. During the subdivision, because of the linearity of the subdivision algorithm, any vertex $v \in \mathcal{M}^k$ can be expressed as a linear combination of vertices in $M^{k-1}$, so a linear combination of the vertices in $M^{k-2}$, and so on. In the limit,

$$\forall v \in \mathcal{M}^\infty,\ \exists\ \{\alpha_1, \alpha_2, \cdots, \alpha_{|\Gamma|}\} \in \mathbb{R}^{|\Gamma|},\ \text{s.t. } v = \sum_{i\in\Gamma} \alpha_i \mathcal{M}_i^0. \tag{1.3}$$

If we perturb the original control mesh from $\mathcal{M}^0$ to $\mathcal{M}'^0$ and construct $v' = \sum_{i\in\Gamma} \alpha_i \mathcal{M}_i'^0$ using the same coefficients and the perturbed vertices, then we can define $f(v) = v'$ as the influence from the perturbation of the original control mesh. The smoothness of the subdivision scheme is then defined as the smoothness of the function $f$. Moreover, assume the volumetric domain covered by $\mathcal{M}^\infty$ is $D^\infty$, the deformation can be defined for any point $x \in D^\infty$: the vertices of $\mathcal{M}^\infty$ are dense in $D^\infty$ and $f$ is continuous, so we can find a vertex arbitrarily closed to $x$ and define the deformation of $x$ to be that of this nearby vertex. Indeed, deformation is also one of the most important applications of vertex-based 3D volumetric domain subdivision algorithms [23, 38].

Early researchers proposed subdivision based on unstructured hexahedral control meshes [3, 25, 29]. However, generating a starting hexahedral mesh subject to given boundary conditions is a difficult task in general.



(1) Step 0: Original mesh (a single tetrahedral cell)

(2) Step 1: Insert vertices in the center of each edge, then split/insert edges and facets

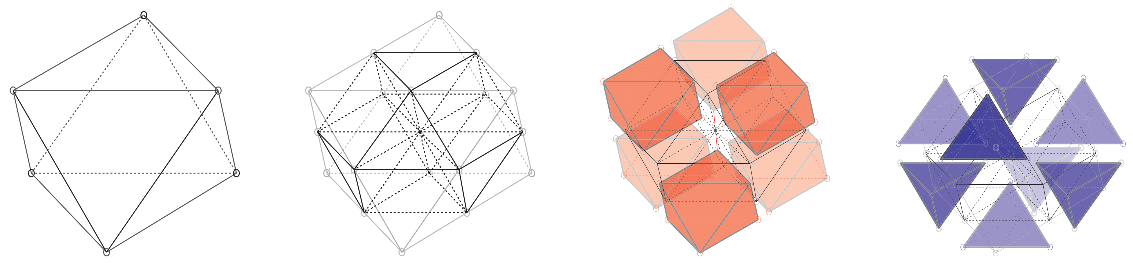
(3) Step 2: Insert 1 octahedron in the center of the parent tetrahedron

(4) Step 2': Insert 4 tetrahedra in the corner of the parent tetrahedron

Figure 1.6: Topological splitting of a single tetrahedral cell

Trivariate box-splines can be used as the basis of a subdivision scheme for unstructured tetrahedral meshes [10, 11]. The topological splitting process is conducted in a dyadic fashion. As illustrated in Fig. 1.6, a single tetrahedral cell with 4 vertices, 6 edges and 4 facets is split into 4 tetrahedra and 1 octahedron with 10 vertices, 24 edges and 20 facets. The details of this process can be found in Appendix A.2. However, in order to proceed to the geometric smoothing process defined on tetrahedral cells, an additional edge needs to be introduced between a pair of opposite vertices in the central octahedron to split the octahedron into four tetrahedra. The arbitrariness of the choice of this vertex pair introduces tremendous difficulty in the study of the smoothness properties of the subdivision scheme. For example, the regularity of vertices cannot be defined in the classical way: even if two vertices are of the same type (e.g., they have the same valence) in the coarse mesh, their valences can be different after one step of topological splitting because the arbitrarily inserted edges.

To avoid this problem, instead of splitting the octahedron, researchers proposed the **octet** mesh. The splitting rule for an octahedral cell is illustrated in Figure 1.7. A single octahedral cell with 6 vertices, 12 edges and 8 facets will be split into 8 tetrahedra and 6 octahedra with 19 vertices, 60 edges and 56 facets. The details of this process are also discussed in Appendix A.2.



(1) Step 0: Original mesh (a single octahedral cell)

(2) Step 1: Insert vertices in the center of the cell and every edge, then split/insert edges and facets

(3) Step 2: Insert 8 octahedra in the corner of the parent octahedron

(4) Step 2': Insert 6 tetrahedra inside the parent tetrahedron

Figure 1.7: Topological splitting of a single octahedral cell

We can define vertex that has the same topological structure as the newly inserted octahedron centroid as regular vertex. One property associated with this regularity is that regular vertices in the coarse mesh remain regular after the subdivision. The types of vertices and other properties of the octet mesh are discussed in Section 2.1. We review the details of the subdivision rules in Section 2.2.

## 1.5 Overview and Contribution

This chapter provides a brief review and summary of the previous literature in related fields. Surface subdivision schemes have been extensively studied for more than 30 years, but volumetric domain subdivision has not been studied until recently because of its complexity. Edge-based 1-form subdivision was studied more recently using DDG tools. This chapter focused on the motivations and applications of these concepts; the details of these issues are addressed in Chapter 2 and Chapter 3. Chapter 3 formalizes the theory behind the 3D 1- and 2-form subdivision schemes. Most of this theory is a natural generalization of the corresponding theory in the surface case.

Based on the theory introduced in Chapter 3, Chapter 4 discusses how to determine the 1- and 2-form subdivision stencils for the regular case, while Chapter 5 discusses the stencil near the boundary of the volumetric mesh. These two chapters focus on the calculation of the subdivision stencils and a few technical details. A discussion of implementation issues (details of data structures and algorithms used) are included in Appendix A.

# Chapter 2

# Classical 0-Form Subdivision in the Volumetric Domain

We briefly review the construction of the classical vertex-based subdivision over the volumetric domain; this material serves as the foundation of the edge- and facet-based subdivision. Section 1.4 discusses the motivation behind the introduction of the octet mesh. Section 2.1 expands upon the octet mesh and related topological, combinatorial and geometric properties. The underlying 0-form subdivision [38] is reviewed in Section 2.2, where we rederive the stencil because the smooth operator during the derivation will be useful for our construction of 1- and 2-form subdivision stencils later.

## 2.1 Geometric and Topological Properties of the Octet Mesh

### 2.1.1 Types of Vertices in the Octet Mesh

Recall that in the case of 2D surface, vertices in the triangular mesh can be classified as having regular type (i.e., having a valence of 6) or irregular type. For the latter, the topological configuration around the vertex can is determined by the valence, i.e., vertices are topologically equivalent if their valences are the same. However, in the 3D volumetric case, the topology situation is much more complicated. Suppose the initial unstructured mesh $\mathcal{M}^0 = \{V^0, E^0, F^0, T^0, O^0\}$ corresponds to the continuous manifold $D^0$ with boundary $\partial D^0$ and interior $(\partial D^0)^C$. After $k$ subdivisions we get the unstructured mesh $\mathcal{M}^k = \{V^k, E^k, F^k, T^k, O^k\}$ corresponding to a continuous manifold $D^k$. The vertices of $V^k$ can be classified into 7 types:

▶ Boundary-Type-III (B3): $x$ is at one boundary vertex of the original cells:

$$x \in V^0 \cap \partial D^0. \tag{2.1}$$

▶ Irregular-Type-III (X3): $x$ is at one interior vertex of the original cells:

$$x \in V^0 \cap (\partial D^0)^C. \tag{2.2}$$

▶ Boundary-Type-II (B2): $x$ is on one boundary edge of the original cells and is not a vertex of any original cells:

$$x \in (E^0 - V^0) \cap \partial D^0. \tag{2.3}$$

▶ Irregular-Type-II (X2): $x$ is on one interior edge of the original cells and is not a vertex of any original cells:

$$x \in (E^0 - V^0) \cap (\partial D^0)^C. \tag{2.4}$$

▶ Boundary-Type-I (B1): $x$ is on one boundary facet of the original cells and is not a vertex of type B2 or B3:

$$x \in (F^0 - E^0 - V^0) \cap \partial D^0. \tag{2.5}$$

▶ Irregular-Type-I (X1): $x$ is on one interior facet of the original cells and is not a vertex of type X2 or X3:

$$x \in (F^0 - E^0 - V^0) \cap (\partial D^0)^C. \tag{2.6}$$

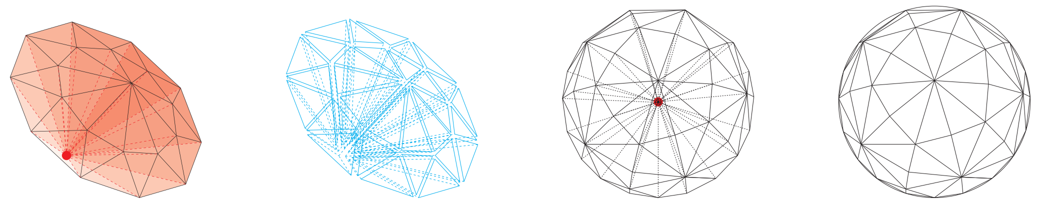
▶ Regular-Type (R): $x$ belongs to none of the above, i.e., $x$ is in the interior of one of the original cells:

$$x \in D^0 - F^0 - E^0 - V^0. \tag{2.7}$$

Notice that the code in the bracket is the notation for each types. B stands for "boundary" while X stands for "extraordinary" which is interchangeable with "irregular" in subdivision literature. For simplicity, we sometimes call the B1, B2 and B3 vertex as *boundary vertex, border vertex* and *corner vertex*, respectively.

**Properties of B3/X3 vertices** The topological configurations of vertices of B3/X3 are shown in Fig. 2.1. These types have the highest level of irregularity. Their topologies are highly variable and are almost unconstrained. Even when two vertices have the same valence, they may still be topologically different because of other combinatorial properties. Indeed, each topologically independent setting corresponds to a particular triangular tessellation of the sphere (or semisphere in the B3 case) [7, 6]. Fig. 2.1 is drawn using this tessellation interpretation. Unlike the other types of vertices, in general, there is no transformation under which the local mesh around B3/X3 vertices is guaranteed to be invariant.

Recall that in the 2D surface case, the number of irregular vertices is determined by the initial mesh and does not change during subdivision. Thus, in the limit mesh $\mathcal{M}^\infty$, any irregular vertex



(1) B3 vertex: the red vertex. Boundary facets are represented by red.

(2) Arbitrary number of tetrahedron can be attached to the B3 vertex.

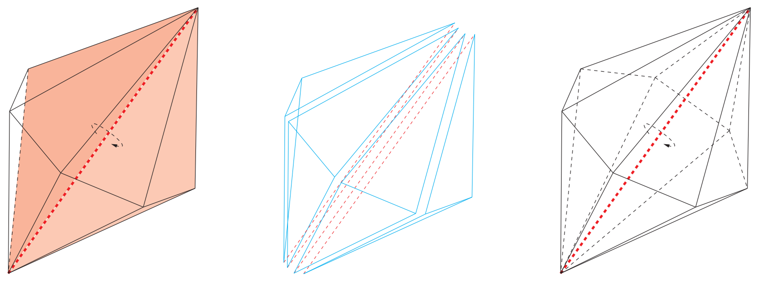
(3) X3 vertex: the red vertex.

(4) The topological configuration is equivalent to a tessellation of the sphere.

Figure 2.1: Vertex of type B3 and X3

is infinitely far away (in the topological sense) from the other irregular vertices. This fact greatly simplifies the analysis of the subdivision scheme because we can treat every irregular vertex as isolated. In the volumetric case, X3/B3 vertices have the same feature, i.e., no additional vertices of these types are introduced during subdivision.

**Properties of B2/X2 vertices** The topological situation around the B2/X2 vertices is shown in Fig. 2.2. X2 vertices are the analogues of the irregular vertices in the surface subdivision case, i.e., their topological configurations are completely determined by the number of tetrahedra they are adjacent to. Assume there are $m$ tetrahedra around the edge on which an X2 vertex lies, then the rotation operations $C_m$ around the edge form the basic symmetry operations of the local topological structure. Similarly, B2 vertex, whose neighborhoods are homeomorphic to semispheres, are the analogues of the boundary vertices in the surface subdivision case.



(1) B2 vertex: vertex lies on the initial boundary edges (red edge). Boundary facets are represented by red color.

(2) Arbitrary number of tetrahedron (wedges) can be attached to the initial edge.

(3) X2 vertex: vertex lies on the initial interior edges (red edge).

Figure 2.2: Vertex of type B2 and X2

**Properties of B1/X1 vertices** The topological configurations of B1/X1 vertices are shown in Fig. 2.3. The topological configurations in the neighborhoods the B1/X1 vertices after one time of dyadic splitting are also shown in the figure. Clearly, these two types of vertices are highly regular. An X1 vertex is always adjacent to 12 edges (this is called the *coordinate number* or the *valence* of the vertex), 24 facets, 8 tetrahedra and 6 octahedra. The reflection operation is one of the key symmetry operations of the local topological structures.

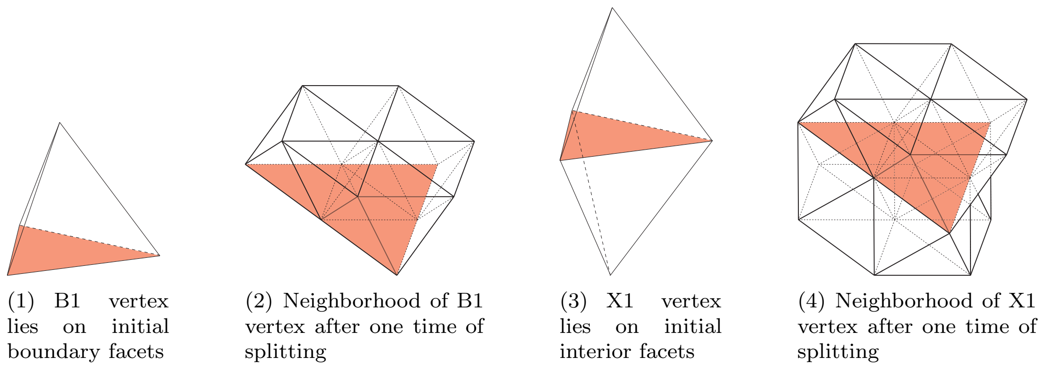

(1) B1 vertex lies on initial boundary facets
(2) Neighborhood of B1 vertex after one time of splitting
(3) X1 vertex lies on initial interior facets
(4) Neighborhood of X1 vertex after one time of splitting

Figure 2.3: The topological configuration of the regions around vertex of type X1 and B1 and the neighborhood after one step of dyadic splitting

**Properties of regular vertices** Fig. 2.4 shows the topological configuration around a regular vertex. Obviously, the vertices inserted at the centroid of each octahedral cell during the subdivision are regular (other vertices may also be regular). Regular vertices fall in the interior $(\partial D_0)^C$ for sure because the boundary discrete submanifold is 2-dimentional (formed only by facets, edges and vertices).

A regular vertex is always adjacent to the same numbers of edges (12), facets (24), tetrahedra (8) and octahedra (6) as an X1 vertex. In contrast, such adjacency relations around an X3 vertex are arbitrary (Fig. 2.1(3)). Furthermore, for an edge between two regular vertices, the edge is adjacent to 2 tetrahedra and 2 octahedra which are sorted alternatively. In contrast, there is no such constraint for the edges which are part of the initial edges (Fig. 2.2(3)). Finally, the only difference between the topological configurations of R vertices and X1 vertices is that, if a facet contains three regular vertices, then it is adjacent to a tetrahedron on one side and an octahedron on the other side. However, facets lying on initial facets are always adjacent to two tetrahedra or two octahedra simultaneously (Fig. 2.3(3)). The detailed discussion of R vertices will be discussed in Section 2.1.2.

We define a partial order $\preceq$ over all vertex types as

$$R \preceq X1 \preceq X2 \preceq X3 \preceq B1 \preceq B2 \preceq B3. \tag{2.8}$$

Using this partial order, the types of geometric objects other than vertices can be defined as the

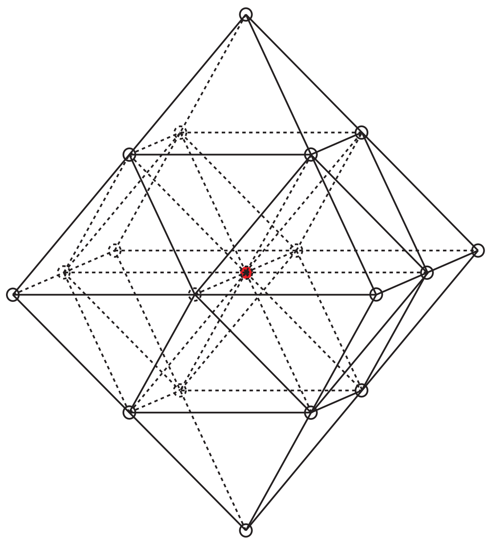


Figure 2.4: The topological configuration of the 1-ring domain around a regular vertex

highest type of their vertices, i.e.,

$$\mathrm{Type}(x) = \max_{\text{Vertex } v \in x} \mathrm{Type}(v). \tag{2.9}$$

Similar to 1-form subdivision in the 2D surface case, in the volumetric domain the discrete differential form coefficients associated with geometric objects of different types should be updated using different stencils.

A (nondegenerate) 3D manifold embedded in a 3D space (e.g., $\mathbb{R}^3$) has a simpler topological structures than a lower-dimensional manifold embedded in the same space. Exotic structure such as unorientable surfaces (e.g., the Möbius strip, the Klein bottle, etc.) do not exist in the 3D case. Intuitively, given any volumetric domain, we can fill the interior of the domain with a regular octet mesh (although the design of such a volume-padding algorithm is highly nontrivial) and leave irregular vertices only on the boundary surface.

## 2.1.2 Properties of the Regular Mesh

The 1-ring neighborhood domain of a regular vertex is shown in Fig. 2.4. As we will see in Chapter 4, this 1-ring neighborhood is the support of a smooth subdivision scheme. For simplicity, we call this neighborhood $\mathcal{M}_E$.

As shown in Fig. 2.5, the regular octet mesh has the *face-centered cubic (FCC)* structure which is the crystal structure of many chemical elements such as calcium and gold. The point group for the FCC lattice is $O_h$, which has 48 key symmetry operations: $E,\ 8C_3,\ 6C_2,\ 6C_4/3C_2,\ i,\ 6S_4,\ 8S_6,\ 2\sigma_h$ and $6\sigma_d$. Fig. 2.6 shows the operations $8C_3,\ 6C_2$ and $6C_4/3C_2$ which will simplify our calculations

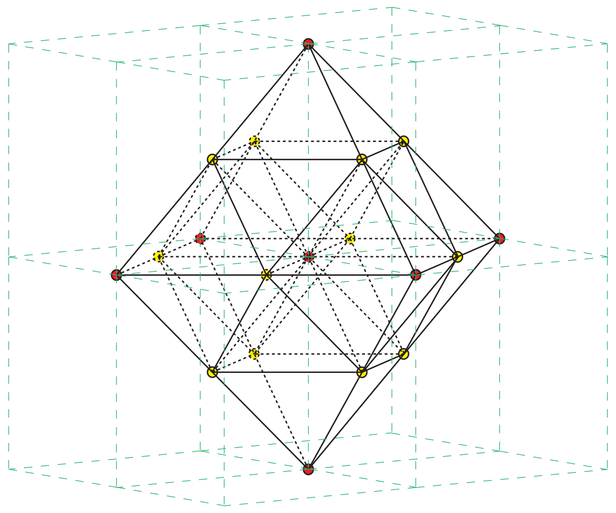
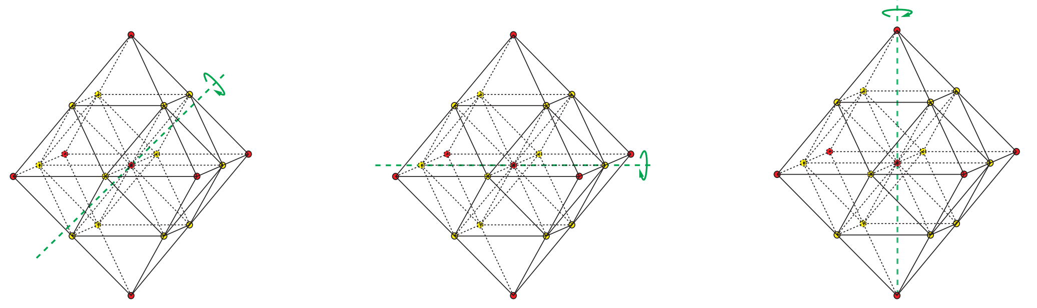


Figure 2.5: The topological configuration of the 1-ring domain around a regular vertex. This configuration has the well-known face-centered cubic structure.

when we write down the 1- and 2-form subdivision matrices later.



(1) 6 $C_3$ operations

(2) 8 $C_2$ operations

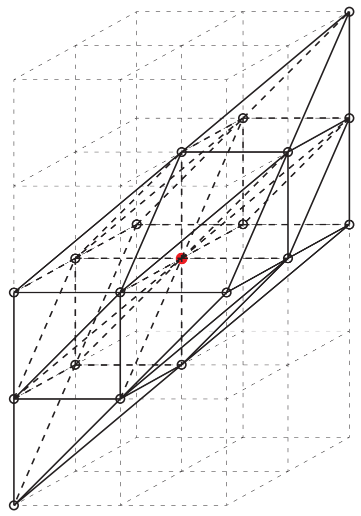
(3) 6 $C_4$ and 3 $C_2$ operations

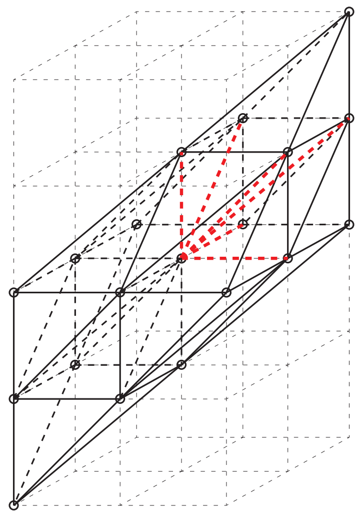
Figure 2.6: Some key symmetry operations for the $O_h$ point group

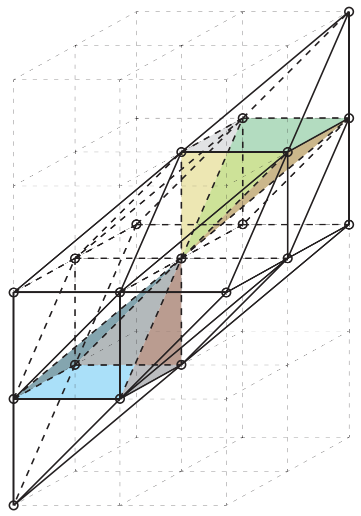
We can embed the regular mesh into a $\mathbb{Z}^3$-lattice as we did in Fig. 2.5. The drawback of this mapping is that the vertices here will not completely iterate all grid points in the $\mathbb{Z}^3$-lattice. However, as shown in Fig. 2.7, a one-to-one correspondence can be constructed between the vertices in the regular octet mesh and the $\mathbb{Z}^3$-lattice. Furthermore, similar one-to-one mappings also exist for the edges, facets and cells of the regular mesh.

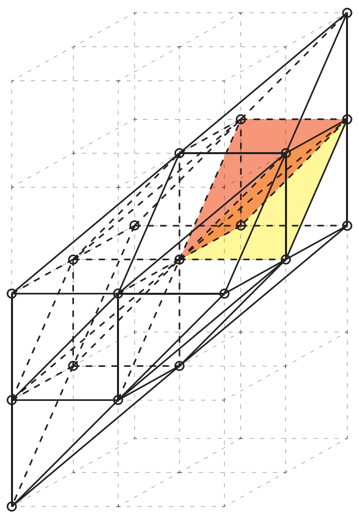
If we define a binary relation $\sim_e$ on the edges of a regular octet mesh $\mathcal{M}$ such that for $\forall e_1, e_2 \in M_r$, $e_1 \sim_e e_2$ if and only if $e_2$ can be obtained by translating $e_1$ along the $\mathbb{Z}^3$-lattice, it is straightforward to show that $\sim_e$ is an equivalence relation. Similarly, we can define $\sim_f, \sim_t$ and $\sim_o$ on the facets, tetrahedra and octahedra of $\mathcal{M}$.

As shown in Fig. 2.7, the edges of the regular octet mesh can be divided into 6 equivalence classes.


(1) Vertices


(2) Edges


(3) Facets


(4) Tetrahedra

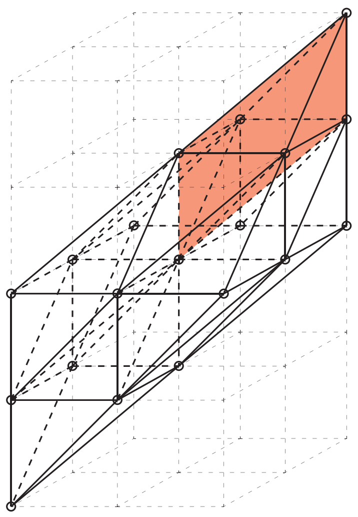

(5) Octahedra

Figure 2.7: Mappings from the $r$-simplices in the regular mesh into the $\mathbb{Z}^3$-lattice

Similarly, there are 8, 2 and 1 equivalence classes for facets, tetrahedra and octahedra, respectively. Some examples of the translation transformation are shown in Fig. 2.8. Note that in the vertices, edges and facets cases, if we remove one simplex, in order to reproduce all the simplices in the corresponding equivalence class we need to translate using an offset that iterates whole $\mathbb{Z}^3$-lattice. However, in the tetrahedra and octahedra cases, the offset only needs to iterate $2 \times \mathbb{Z}^3$, i.e., the translation always takes an even number of steps.

## 2.2 Subdividing the Octet Mesh

### 2.2.1 Linear 0-Form Subdivision

The linear subdivision proposed here is simply topological dyadic splitting without any geometric smoothing. Specifically, the scheme is summarized in the stencil in Fig. 2.9.

We use two datasets as examples for the 0-form subdivision task. One is $\mathcal{M}_E$, the 1-ring neighborhood around a regular vertex (Fig. 2.10(1)). In order to test all the subdivision schemes on a more exotic mesh, we also created a concave version of $\mathcal{M}_E$ (denoted as $\mathcal{M}_X$, Fig. 2.10(2)).

Because the linear subdivision involves only splitting, the smoothness of the initial mesh does not increase after subdivision. The volumetric domain associated with the mesh also does not change. In other words, linear subdivision is a purely mesh generating process (Fig. 2.11).

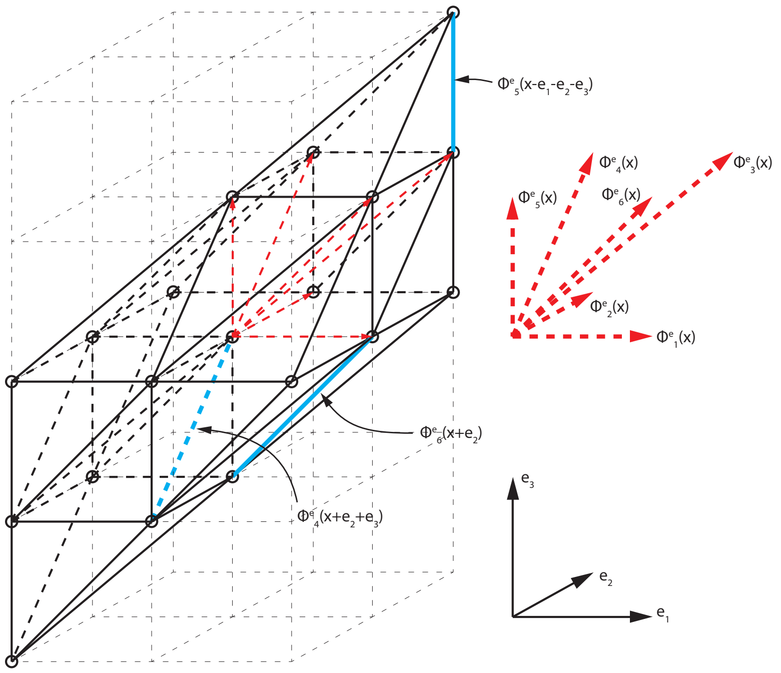
### 2.2.2 Smooth 0-Form Subdivision

In order to derive the smooth subdivision stencil, we used techniques discussed in Section 3.3.2 and Section 3.4, where the theoretical framework of generating functions and the construction of smoother subdivision schemes through convolution are introduced. Readers can skip the derivation and return after reading these sections.

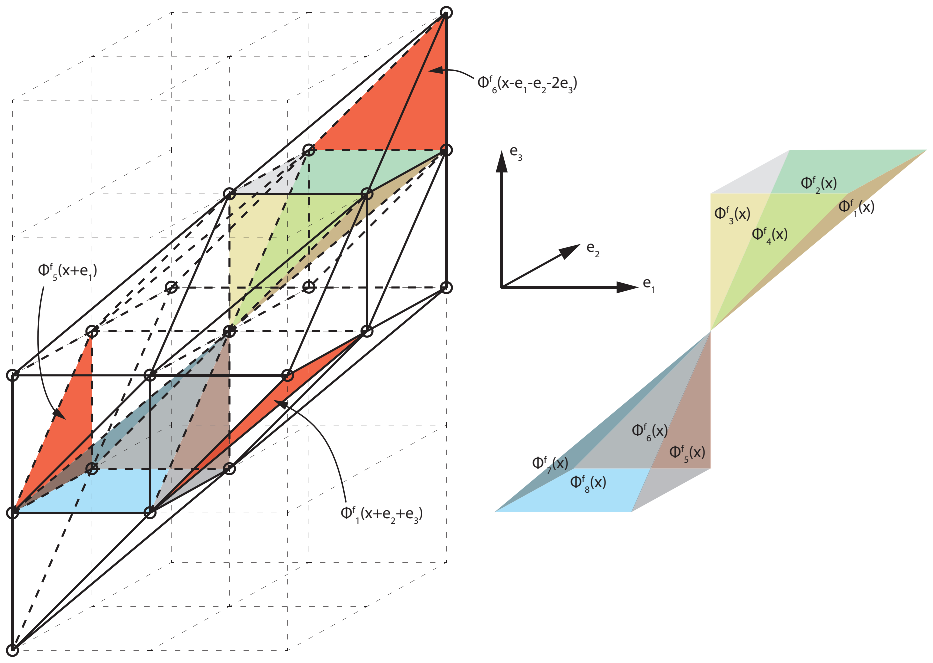
In the regular setting, subdivision schemes with higher regularity can be constructed from a given subdivision scheme using convolution [44]. Specifically, based on the one-to-one mapping from the regular mesh to the $\mathbb{Z}^3$-lattice (Fig. 2.7(1)), we can write down the generating function of the linear subdivision stencil:

$$\begin{aligned}\widehat{S_v^L} = 1 &+ \frac{1}{2}\left(xyz + \frac{1}{xyz} + xz + \frac{1}{xz} + x + \frac{1}{x} + yz + \frac{1}{yz} + y + \frac{1}{y} + z + \frac{1}{z}\right) \\ &+ \frac{1}{6}\left(xyz^2 + \frac{1}{xyz^2} + xy + \frac{y}{x} + \frac{x}{y} + \frac{1}{xy}\right). \end{aligned} \tag{2.10}$$

A smooth subdivision scheme can be obtained by convoluting the linear stencil with itself. In the Fourier domain, the generating function for this smooth scheme is the square of that of the linear



(1) Edges



(2) Facets

Figure 2.8: Examples of the correspondence between the edges/facets and the $\mathbb{Z}^3$-lattice

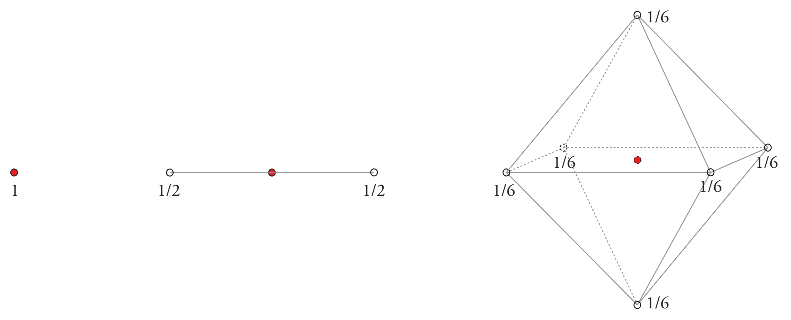
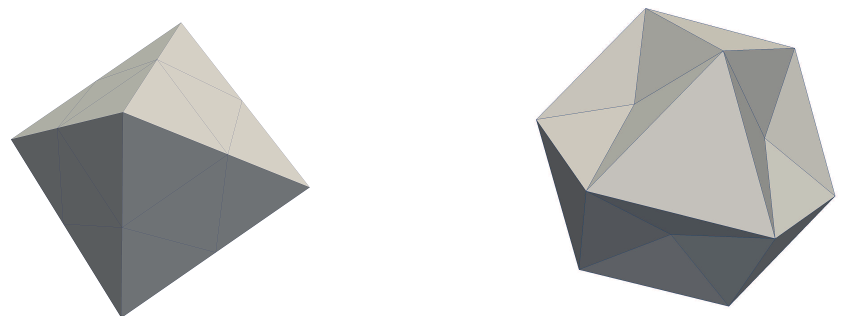


Figure 2.9: 0-form linear subdivision stencil for three types of vertices: vertices in the coarse mesh, vertices introduced in the center of each edge and vertices introduced in the centroid of each octahedral cell



(1) $\mathcal{M}_E$ dataset (2) $\mathcal{M}_X$ dataset

Figure 2.10: Data files used in this chapter. Note that for the $\mathcal{M}_X$ data, one side is concave.

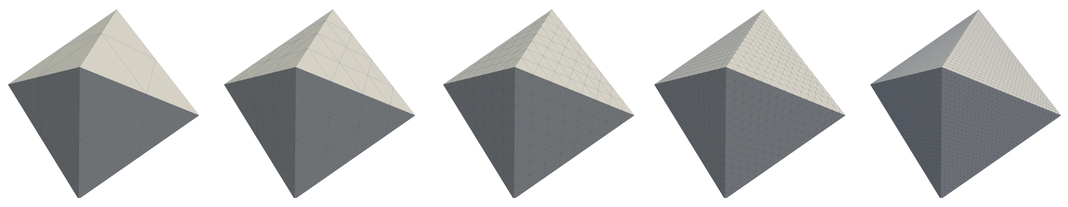


Figure 2.11: 0-form linear subdivision results for $\mathcal{M}_E$: initial mesh and meshes after $1 \sim 4$ times of subdivision. For $\mathcal{M}_X$, the results are similar.

scheme, modulo a constant factor that normalizes the subdivision stencil:

$$\widehat{S_v} = \frac{1}{8}\ \widehat{S_v^L} \times \widehat{S_v^L}. \tag{2.11}$$

We map the generating function $\widehat{S_v}$ back to a subdivision stencil in the real domain by referring to the $\mathbb{Z}^3$-lattice to match the vertices with the corresponding frequency modes. The resulting subdivision stencil is illustrated in Figure 2.12.

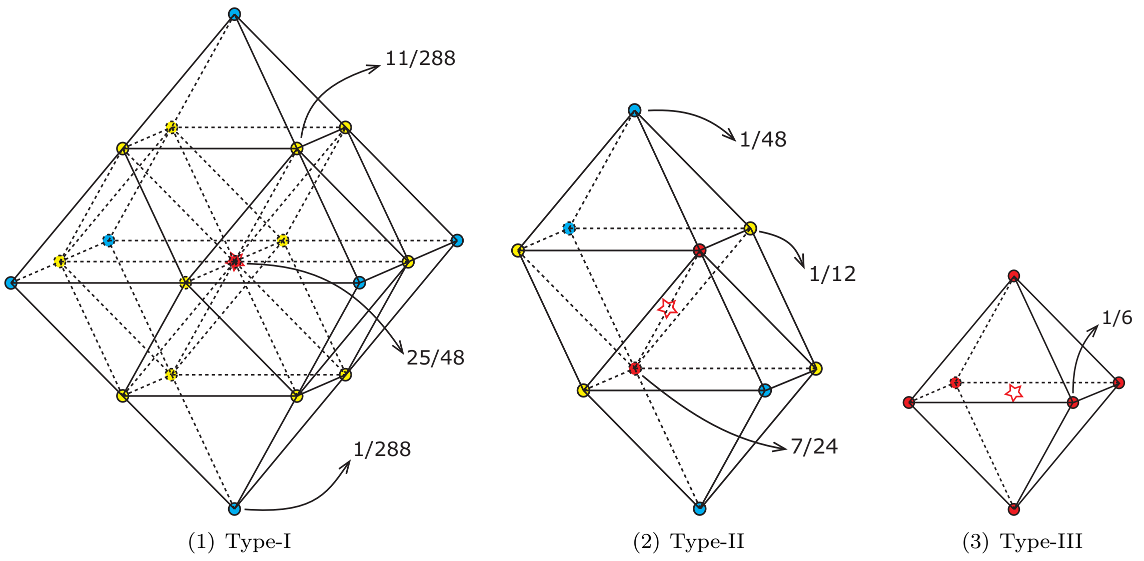
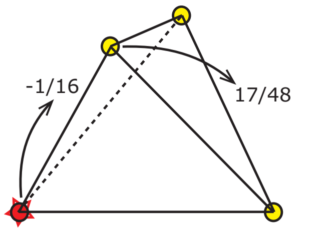


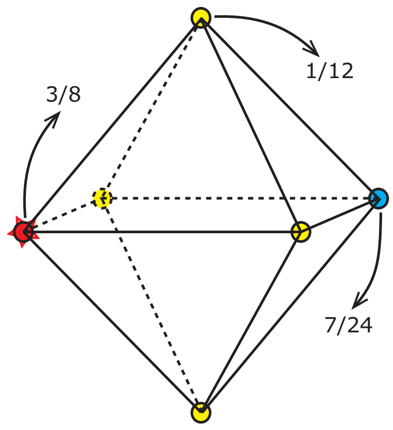
Figure 2.12: 0-form subdivision stencil in the regular setting. The star marks the position of the target vertex in the refined mesh. We use different colors to distinguish the vertices with different contributions in the coarse mesh.

The 0-form smooth subdivision scheme can be factorized into a simpler stencil defined on single cells (Fig. 2.13). In practice, to find a vertex in the refined mesh, we must iterate over all the adjacent cells (8 tetrahedra and 6 octahedra), calculate the new position of the vertex in each cell and compute the arithmetic mean of these 14 positions.

Subdivision results for both datasets based on the smooth 0-form stencil just derived are shown in Fig. 2.2.2. From the results we can clearly see that the smoothness of the mesh is improved every time we subdivide the mesh. It can be proved that the smooth scheme generates $C^2$ deformations everywhere except at the edges and vertices in the initial mesh [38].

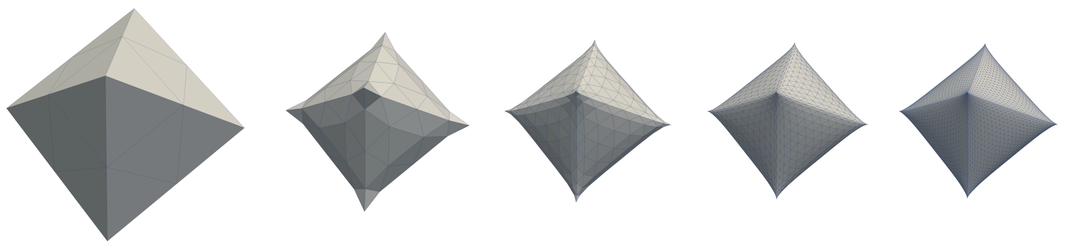


(1) Stencil in tetrahedron

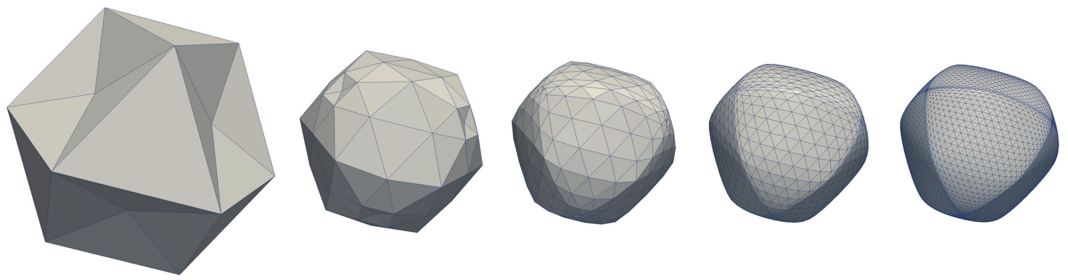


(2) Stencil in octahedron

Figure 2.13: The 0-form linear subdivision after factorization



(1) Subdivision results of $\mathcal{M}_E$



(2) Subdivision results of $\mathcal{M}_X$

Figure 2.14: 0-form smooth subdivision results for both data sets: initial mesh and meshes after 1 $\sim$ 4 times of the smooth subdivision

# Chapter 3

# Discrete Differential Form Subdivision over Volumetric Domain

This chapter provides a thorough review of discrete differential geometry concepts related to the 3D volumetric domain 1- and 2-form subdivision problems. Section 3.1.1 reviews the definition and properties of the simplicial complex; we extend the discussion to include the octahedral cells in the octet mesh. Sections 3.1.2 and 3.1.3 discuss the discrete analogues of the integration operator, differential forms and exterior derivatives. Section 3.2 introduces the definition of Whitney forms and their realizations in tetrahedral and octahedral cells. Section 3.3 introduces the related theoretical background behind the construction of the 1- and 2-form subdivision schemes for the volumetric domain. Most of these theories are directly adapted from its analogues in the surface subdivision theory. Section 3.4 briefly reviews on the commutative relations between subdivision operators that is an analogue of Stokes' rule in the continuous manifold. The theoretical background of constructing smooth subdivision rule from linear subdivision scheme is also discussed here.

## 3.1 Basic Discrete Exterior Calculus

### 3.1.1 Simplicial Complexes and Boundary Operators

Simplices are the building blocks in the DDG framework. An *$r$-simplex* $\sigma_r$ is defined as the convex hull of $r+1$ geometrically independent points $v_0, \cdots, v_r$ and can be written as $\sigma_r = \{x \in \mathbb{R}^n | \ x = \sum_{i=0}^{r} \alpha^i v_i$ where $\alpha^i \geq 0$ and $\sum_{i=0}^{r} \alpha_i = 1\}$. $\sigma_r$ can also be interpreted as a purely combinatorial quantity represented by an $(r+1)$-tuple $\{v_0, \cdots, v_r\}$. The elements in this $(r+1)$-tuple are called *vertices* while the order $r$ is called the *dimension* of the simplex.

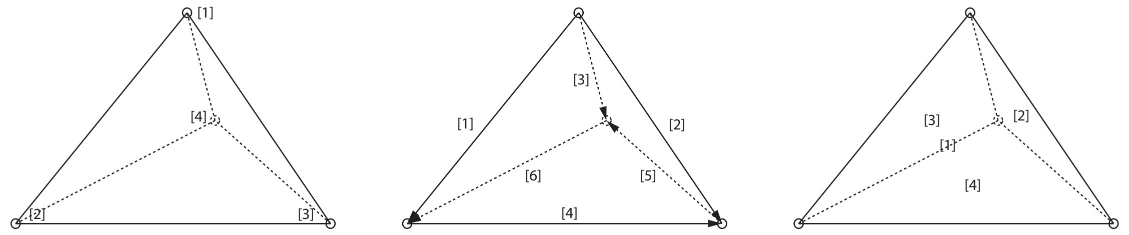
Any subset of $\{v_0, \cdots, v_r\}$ defines a *facet* of $\sigma_r$. Note that here "facet" generalizes the concept of a facet in conventional sense where a facet is a 2-simplex. A *simplicial complex* $\mathcal{K}_d$ is a set of simplices that satisfies two additional conditions: (1) Every facet of a simplex in $\mathcal{K}_d$ is itself a simplex; (2) The intersection of two simplices in $\mathcal{K}_d$ is either empty or an entire facet of each of them. We define the dimension of $\mathcal{K}_d$ to be the highest of the dimensions of its contained simplices. The collection of all $r$-simplices in $\mathcal{K}_d$ is denoted as $\mathcal{K}_d^r$.

A simplicial complex may be boundaryless; one example is a triangular tessellation of a sphere. If a simplicial complex has a boundary, the boundary facets constitute a lower-dimensional simplicial complex. Furthermore, for an interior vertex or point inside an $r$-dimensional simplicial complex, its local neighborhood is homeomorphic to an $r$-sphere. In contrast, for the boundary vertex or point inside any of the boundary simplices, its local neighborhood is homeomorphic to an $r$-semisphere.

In Chapter 2 we introduced the unstructured tetrahedral mesh (which, after one step of classical subdivision, becomes an octet mesh). The tetrahedral mesh is an example of a 3D simplicial complex: R, X1, X2 and X3 vertices are interior vertices while B1, B2 and B3 vertices are boundary vertices (Section 2.1.1). One can convince oneself that the boundary of the whole mesh forms a 2D simplicial complex without boundary. This is a consequence of Poincaré's lemma in the discrete domain.

The simplices in $\mathcal{K}_d$ are undirected. However, to simplify the discussions in this thesis and facilitate the calculations of 1- and 2-form subdivisions, we extend the above definitions to the oriented (or directed) context. If we introduce an equivalence relation among all the $(r+1)$-tuples such that $\sigma_r^{(1)} \sim \sigma_r^{(2)}$ if and only if one can be obtained by an even permutation of the other, then all the $r$-simplices formed by vertices $v_0, \cdots, v_r$ can be divided into two equivalence classes. Each of these two classes is called an *orientation*. A simplex together with its orientation is called an *oriented simplex* and is denoted as $[v_0, \cdots, v_r]$. All the simplices mentioned below, if not mentioned explicitly, are oriented. We use the same notation $\sigma_r$ for simplicity. While the orientation we assign to a simplex is arbitrary and depends on the ordering of the vertices, the intrinsic orientations are fixed and so are the associated geometric quantities. For example, suppose a simplex $\{v_i v_j\}$ is immersed in a vector field, if the contour integral of the field along $[v_i v_j]$ yields $I$ then the contour integral of the field along $[v_j v_i]$ yields $-I$. In other words, the 1-form coefficient associated with the edge is bundled with the assumed orientation.

The boundary operator $\partial$ applied to an $r$-simplex yields a sum of oriented $(r-1)$-facets of the simplex. Formally, $\partial([v_0, \cdots, v_r]) = \sum_{i=0}^{r}(-1)^i[v_0, \cdots, \widehat{v_i}, \cdots, v_r]$ where $\widehat{v_i}$ indicates that the $i$th vertex is missing from the tuple. A single tetrahedral cell is a useful example here. As shown in Fig. 3.1, we assign each simplex an index. While the orientations of 0- and 3-simplices are always positive, the 1-simplices' orientation are indicated by the arrow in Fig. 3.1(2). For simplicity, for 2-simplices, if not mentioned explicitly, the orientations are assumed to be towards the exterior of the tetrahedron. We emphasize again that, while we define the indexing and orientation rules in a specific way and the use of "exterior" here makes the orientation coordinate dependent, these rules are not intrinsic and can be arbitrarily changed. We follow this convention only to facilitate our discussions.



(1) Vertices (2) Edges (3) Facets

Figure 3.1: Indexing rule for the tetrahedral cell

By specifying the indexing and orientation rules we can write down the boundary operator $\partial$ in a matrix format: it encodes the incidence relations between all $r$-simplices and all $(r-1)$-simplices in $\mathcal{K}_d$. For the example above we have

$$\partial^v = \left(\begin{array}{ccc|ccc} -1 & -1 & -1 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -1 \end{array}\right), \quad \partial^e = \left(\begin{array}{ccc|c} 1 & 0 & -1 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ \hline 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{array}\right), \quad \partial^f = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \tag{3.1}$$

which are boundary operators on 3-, 2- and 1-form, respectively. For example, the $i$th column of $\partial^v$ records two ends of the edge, while the $i$th column of $\partial^e$ records three surrounding edges.

One can easily verify that $\partial\partial = 0$ for the above example. This identity can be extended to general simplicial complexes because, when we apply two boundary operations on a $k$-simplex, the resulting $(k-2)$-simplices are paired up with opposite signs. This identity can be interpreted as stating that "the boundary of a boundary is empty".

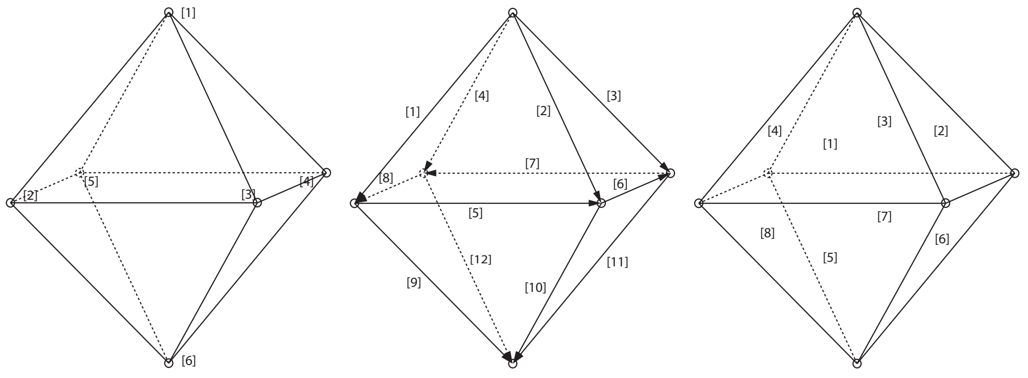
The concept of a simplicial complex can be extended to general unstructured meshes by allowing non-simplicial polyhedrons to exist in the discrete manifold. Unlike the simplicial complex case where we only have to specify the highest-dimensional simplices (e.g., from the configuration of the tetrahedron, we are able to identify all facets and corresponding incidence relations with tetrahedral cells, and then identify all edges and vertices), for a general unstructured mesh, we need to specify all surrounding discrete submanifolds. For example, in the 3D case, for a polyhedron $P$ we need to specify that it is surrounded by $n_1$ triangles and $n_2$ quadrangles, etc. and also specify the vertex-sequence of each polyhedron. For general non-simplicial polyhedra, the concept of orientation is also induced from the equivalence classes based on the even permutation equivalence relation.

The generalization of the boundary operator is also straight forward. For an $r$-dimensional polyhedron $P = [v_0, \cdots, v_{|P|}]$, assume the collection of the $(r-1)$-dimensional boundary facets is

$\mathcal{B}(P)$, then the boundary operator can be defined as

$$\partial([v_0, \cdots, v_{|P|}]) = \sum_{P_i = \{\tilde{v}_0, \cdots, \tilde{v}_{|P_i|}\} \in \mathcal{B}(P)} (-1)^{\pi([\tilde{v}_0, \cdots, \tilde{v}_{|P_i|}])} [\tilde{v}_0, \cdots, \tilde{v}_{|P_i|}]. \tag{3.2}$$

Here, for any boundary facet $P_i$, the vertex sequence is a subset of the vertex sequence of the parent cell $P$. Here, we follow the convention that the order of the vertex sequence in each $(k-1)$-facet to be the same as in the parent cell. The orientation of the boundary submanifold is specified by $\pi \in \{-1, 1\}$, which can be arbitrarily defined. For example, we can follow the convention that all the facets are pointing towards the exterior of the polyhedron or that the edges surround the facet in a clockwise manner.



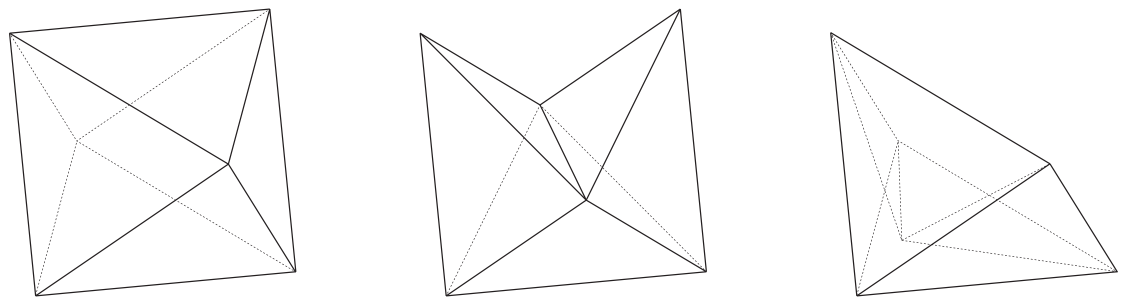
(1) 0-form stencil (2) 1-form stencil (3) 2-form stencil

Figure 3.2: Indexing rule for the octahedral cell

We want to write down the boundary operator for the octahedral cell. As shown in Fig. 3.2, the 0-, 1- and 3-simplices' orientations follow the same conventions as in the tetrahedral case. Because the octahedral cell is always adjacent to a tetrahedral case in the regular octet mesh, we define the positive orientation of 2-simplices as the direction towards the interior of the octahedron. Unlike the tetrahedral case, it is trickier to determine"interior" because a nondegenerate octahedron may not be convex. As shown in Fig. 3.3, the interior of octahedron on the left side is well-defined. For the other two octahedra, we need to pay more attention to detect the interior in practice.

If an octahedron is adjacent to a tetrahedron, because the orientation of the interface facet is defined from the tetrahedron side, the interior of the octahedron can be easily identified. When the octahedron is isolated or adjacent only to other octahedra, we use the following method to identify the interior. Again, while this two-step algorithm is coordinate dependent, the intrinsic orientation of the octahedron is fixed.

*Step 1:* Choose an arbitrary point $p$ in the space and create an arbitrary half-line from it. We



(1) Convex octahedron

(2) Nonconvex octahedron

(3) Another nonconvex octahedron

Figure 3.3: In a convex octahedral cell, for each facet, it is easy to tell which side is interior. However, we need a more complicated criterion to make such a judgement in the nonconvex octahedral cell.

count the number of intersections this half-line makes with all facets. If there is an even number of intersections, this point is in the exterior of the cell, otherwise it is in the interior.

*Step 2:* Create a half-line $l_i$ from $p$ such that $l_i$ intersects with facet $f_i$ at $p_i$. $l_i$ may have intersections with other facets before $p_i$. Assume $p_i$ is the $n_i$-th intersection then we identify the direction from which $l_i$ passes through $f_i$ from the following decision matrix. The interior of the cell and the induced orientation of the facet are then determined.

| | $p$ is an interior point | $p$ is an exterior point |
|---|---|---|
| $n_i$ is odd | interior $\rightarrow$ exterior | exterior $\rightarrow$ interior |
| $n_i$ is even | exterior $\rightarrow$ interior | interior $\rightarrow$ exterior |

(3.3)

Under this convention, the boundary operators for the octahedral cell can be written as

$$\partial^v = \left(\begin{array}{cccc|cccc|cccc} -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & -1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right),$$

$$\partial^e = \left(\begin{array}{cccc|cccc} 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{array}\right), \quad \partial^f = \left(\begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \\ \hline 1 \\ 1 \\ 1 \\ 1 \end{array}\right). \tag{3.4}$$

### 3.1.2 Chains and Cochains

Given an oriented simplicial complex $\mathcal{K}_d$, an $r$-*chain* is a linear combination of all $r$-simplices in $\mathcal{K}_d$ (i.e., $\sum_{\sigma\in\mathcal{K}_d^r} c(\sigma)\cdot\sigma$ where $c(\sigma)\in\mathbb{R}$) and may be represented by a $|\mathcal{K}_d^r|$-vector in which each component corresponds to one $r$-simplex in $\mathcal{K}_d$ according to some indexing rule. The collection of all $r$-chains is denoted by $\mathcal{C}_r(\mathcal{K}_d)$. An important property of chains is that the boundary of an $r$-chain is an $(r-1)$-chain. This is because boundary of an $r$-simplex is a linear combination of several $(r-1)$-simplices (with coefficients $\pm 1$).

Recall that $r$-forms are linear mappings from an $r$-dimensional submanifold of a continuous manifold to $\mathbb{R}$. The collection of all the $r$-forms, $\Omega^r(\mathcal{K}_d)$, is a linear space. The discrete analogue of an $r$-form is a *cochain* which is defined as the dual of a chain. Specifically, we associate each $r$-simplex with a real value. The set of these real values forms a $|\mathcal{K}_d^r|$-vector that is called an $r$-cochain. The collection of all $r$-cochains is a linear space and is denoted by $\Omega_d^r(\mathcal{K}_d)$. An "integral" is simply the inner product of a chain and a cochain.

One can discretize an $r$-form on a continuous manifold by integrating it over each $r$-simplex and associating the result (a scalar) with the simplex to form a cochain. Under this discretization process, evaluating the inner products is consistent with classical integration, i.e.,

$$\int_{\bar{\sigma}_r} \omega^r = \langle \omega_d^r, \sigma^r \rangle \tag{3.5}$$

where $\omega_d^r \in \Omega_d^r(\mathcal{K}_d)$, $\omega_d^r \in \Omega^r(\mathcal{K})$, $\sigma^r \in \mathcal{C}_r(\mathcal{K}_d)$ and $\bar{\sigma}^r$ is exactly the same domain but interpreted as a continuous submanifold immersed in the continuous domain.

It is clear that the above definition can be extended to the general unstructured polyhedron mesh. Again we emphasize that, although we show how to discretize continuous forms through integration over a submanifold, an underlying continuous manifold or differential form is not necessary: we can assign arbitrary real values to each of the simplices. Indeed, we can also reverse the process. As we can see in Section 3.2, given a cochain, a compatible continuous differential form can be reconstructed through Whitney forms.

### 3.1.3 Exterior Derivatives and Coboundary Operators

The *coboundary* operator d is defined as the adjoint of the boundary operator $\partial$. Because the boundary of a chain $c$ is also a chain, given a discrete form $\omega_d$ the discrete integral $\langle \omega_d, \partial c\rangle$ is well defined and $\langle \mathrm{d}\omega_d, c\rangle = \langle \omega_d, \partial c\rangle$. This is essentially the discrete analogue of Stokes' theorem:

$$\int_{\sum_i c_i \partial\sigma_i} \omega = \langle \omega, \partial(\sum_i c_i\sigma_i)\rangle = \langle \mathrm{d}\omega, \sum_i c_i\sigma_i\rangle = \int_{\sum_i c_i\sigma_i} \mathrm{d}\omega. \tag{3.6}$$

Because the boundary operator $\partial$ has a matrix representation that records the incidence relations between $\mathcal{K}_d^r$ and $\mathcal{K}_d^{r-1}$, d is simply $\partial^{\mathrm{T}}$, a $(|\mathcal{K}_d^r| \times |\mathcal{K}_d^{r-1}|)$-matrix. Further, recall that $\partial\partial = 0$, so we have the discrete version of the Poincaré lemma: $\mathrm{dd} = (\partial\partial)^{\mathrm{T}} = 0$. The analogues of this relation in the continuous domain are identities such as $\nabla \cdot \nabla \times = 0$ and $\nabla \times \nabla = 0$. $\mathrm{dd} = 0$ also induces the *discrete de Rham complex* (also known as the *cochain complex*), a sequence of linear spaces connected by the coboundary operator d, i.e.,

$$0 \rightarrow \Omega_d^0(\mathcal{K}_d) \xrightarrow{\mathrm{d}} \Omega_d^1(\mathcal{K}_d) \xrightarrow{\mathrm{d}} \cdots \xrightarrow{\mathrm{d}} \Omega_d^n(\mathcal{K}_d) \rightarrow 0. \tag{3.7}$$

## 3.2 Whitney Forms over 3D Meshes

### 3.2.1 Whitney Forms

Whitney forms map from cochains to forms. Consider a simplicial complex $\mathcal{K}_d$ and the space of $r$-forms $\Omega^r(\mathcal{K}_d)$. For any $\omega \in \Omega^r(\mathcal{K}_d)$ and any $r$-chain $c \in \mathcal{C}_r(\mathcal{K}_d)$, the integration $\int_c \omega$ can be defined and is a linear function on chains. We denote this integration by $R(\omega) \cdot c$ and name it the *de Rham map* $R: \ \Omega^r(\mathcal{K}_d) \rightarrow \Omega_d^r(\mathcal{K}_d)$.

Consider the inverse problem for 0-forms. The linear interpolation of discrete 0-forms to the continuous domain is straightforward. We introduce the vertex-based interpolation basis (i.e., the hat function): the basis function $\phi_i$ associated with vertex $v_i$ is a piecewise-linear function such that $\phi_i = 1$ at $v_i$ and $\phi_i = 0$ at the rest of the vertices. Note that $\sum_i \phi_i = 1$ and $\sum_i \mathrm{d}\phi_i = 0$.

Formally, the *Whitney map* is a linear map $W : \Omega_d^r(\mathcal{K}_d) \rightarrow \Omega^r(\mathcal{K}_d)$ such that for any $r$-simplex $\sigma_r = [v_{i_1} \cdots v_{i_r}] \in \mathcal{K}_d$,

$$W\sigma_r = r! \sum_{j=0}^{r} (-1)^{j-1} \phi_{i_j} \mathrm{d}\phi_{i_1} \wedge \cdots \wedge \widehat{\mathrm{d}\phi_{i_j}} \wedge \cdots \wedge \mathrm{d}\phi_{i_r}. \tag{3.8}$$

$W\sigma_r$ is called the *Whitney form* of $\sigma_r$. Let $\mathcal{W}^r(\mathcal{K}_d)$ denote the space of the Whitney $r$-forms on $\mathcal{K}_d$. For any $r$-simplex in $\mathcal{K}_d$ the integration of the corresponding Whitney form is well-defined.

Consider Whitney forms on a 3-dimensional simplicial complex $\mathcal{K}_d$. Assume $\phi_i$ is the Whitney 0-form associated with the vertex $[v_i]$ in $\mathcal{K}_d$, $\phi_{ij}$ is the Whitney 1-form associated with the edge $e_{ij} = [v_i v_j]$ in $\mathcal{K}_d$, $\phi_{ijk}$ is the Whitney 2-form associated with the facet $f_{ijk} = [v_i v_j v_k]$ in $\mathcal{K}_d$ and $\phi_{ijkl}$ is the Whitney 3-form associated with the tetrahedron $t_{ijkl} = [v_i v_j v_k v_l]$ in $\mathcal{K}_d$. Based on the above definition of Whitney forms, we have

$$\phi_{ij} = \phi_i \mathrm{d}\phi_j - \phi_j \mathrm{d}\phi_i \tag{3.9}$$

$$\phi_{ijk} = 2(\phi_i \mathrm{d}\phi_j \wedge \mathrm{d}\phi_k + \phi_j \mathrm{d}\phi_k \wedge \mathrm{d}\phi_i + \phi_k \mathrm{d}\phi_i \wedge \mathrm{d}\phi_j) \tag{3.10}$$

$$\phi_{ijkl} = 6(\phi_i \mathrm{d}\phi_j \wedge \mathrm{d}\phi_k \mathrm{d}\phi_l - \phi_j \mathrm{d}\phi_i \wedge \mathrm{d}\phi_k \mathrm{d}\phi_l + \phi_k \mathrm{d}\phi_i \wedge \mathrm{d}\phi_j \mathrm{d}\phi_l - \phi_l \mathrm{d}\phi_i \wedge \mathrm{d}\phi_j \mathrm{d}\phi_k) \tag{3.11}$$

Using the fact that $\phi_i + \phi_j + \phi_k + \phi_l = 1$ and $\mathrm{d}\phi_i + \mathrm{d}\phi_j + \mathrm{d}\phi_k + \mathrm{d}\phi_l = 0$, the expression of the Whitney 3-form can be simplified to:

$$\begin{aligned}\phi_{ijkl} &= 6(\phi_i \mathrm{d}\phi_j \wedge \mathrm{d}\phi_k \mathrm{d}\phi_l - \phi_j \mathrm{d}\phi_i \wedge \mathrm{d}\phi_k \mathrm{d}\phi_l + \phi_k \mathrm{d}\phi_i \wedge \mathrm{d}\phi_j \mathrm{d}\phi_l - \phi_l \mathrm{d}\phi_i \wedge \mathrm{d}\phi_j \mathrm{d}\phi_k) \\ &= 6[\mathrm{d}\phi_j \wedge \mathrm{d}\phi_k \wedge \mathrm{d}\phi_l - \phi_j(\mathrm{d}\phi_i + \mathrm{d}\phi_j) \wedge \mathrm{d}\phi_k \wedge \mathrm{d}\phi_l \\ &\quad - \phi_k(\mathrm{d}\phi_i + \mathrm{d}\phi_k) \wedge \mathrm{d}\phi_l \wedge \mathrm{d}\phi_j - \phi_l(\mathrm{d}\phi_i + \mathrm{d}\phi_l) \wedge \mathrm{d}\phi_j \wedge \mathrm{d}\phi_k] \\ &= 6[\mathrm{d}\phi_j \wedge \mathrm{d}\phi_k \wedge \mathrm{d}\phi_l + \phi_j(\mathrm{d}\phi_k + \mathrm{d}\phi_l) \wedge \mathrm{d}\phi_k \wedge \mathrm{d}\phi_k \\ &\quad + \phi_k(\mathrm{d}\phi_j + \mathrm{d}\phi_l) \wedge \mathrm{d}\phi_l \wedge \mathrm{d}\phi_j + \phi_l(\mathrm{d}\phi_j + \mathrm{d}\phi_k) \wedge \mathrm{d}\phi_j \wedge \mathrm{d}\phi_k] \\ &= 6\mathrm{d}\phi_j \wedge \mathrm{d}\phi_k \wedge \mathrm{d}\phi_l = 6\mathrm{d}\phi_k \wedge \mathrm{d}\phi_l \wedge \mathrm{d}\phi_i = 6\mathrm{d}\phi_l \wedge \mathrm{d}\phi_i \wedge \mathrm{d}\phi_j = 6\mathrm{d}\phi_i \wedge \mathrm{d}\phi_j \wedge \mathrm{d}\phi_k \end{aligned} \tag{3.12}$$

If we embed the simplicial complex into a metric space (here we use $\mathbb{R}^3$), we can visualize the Whitney 0-, 1-, 2- and 3-forms. Under the Euclidean metric, inside the tetrahedron, the Whitney 0-forms are piecewise-linear hat functions. More specifically, only the corresponding vertex has value 1 while other vertices are zero. Values inside the cell are linearly interpolated. Further, the 3-forms are constant functions over the tetrahedron (proportional to the reciprocal of the volume). The 1- and 2-forms cases, together with the forms inside the octahedron, are not so straightforward. The following sections addresses these difficulties.

### 3.2.2 Whitney Forms in the Tetrahedron

Now we start to calculate Whitney 1- and 2-form inside a 3-simplex $T$. We embed $T$ into a 3-dimensional affine coordinate system shown in Fig. 3.4.

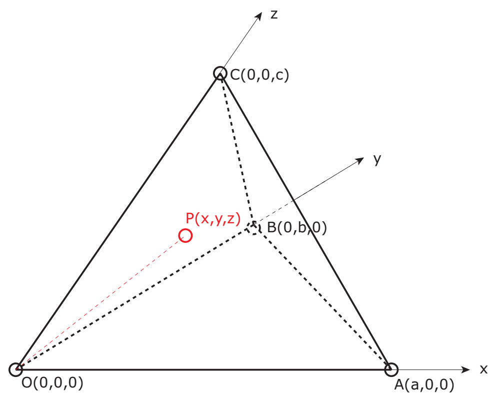


Figure 3.4: Mathematical setting for calculating Whitney forms in a 3-simplex

In this setting we have

$$\phi_A = \frac{x}{a},\ \phi_B = \frac{y}{b},\ \phi_C = \frac{z}{c}\ \Rightarrow\ \mathrm{d}\phi_A = \frac{1}{a}(1,0,0),\ \phi_B = \frac{1}{b}(0,1,0),\ \phi_C = \frac{1}{c}(0,0,1). \tag{3.13}$$

Thus, the Whitney 1-form vector with respect to the edge $\overline{AB}$ is

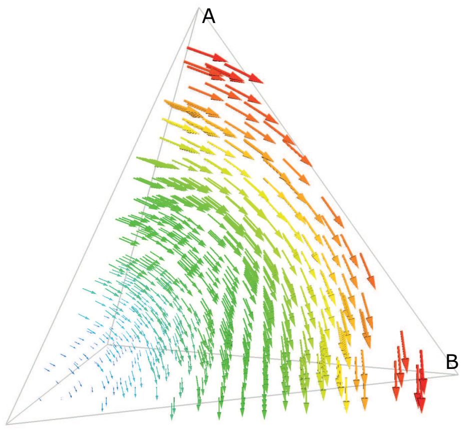
$$\phi_{AB} = \phi_A \mathrm{d}\phi_B - \phi_B \mathrm{d}\phi_A = \frac{1}{ab}(-y, x, 0). \tag{3.14}$$

The vector field associated with this result is shown in Fig. 3.5(1). All the vectors are perpendicular to the edge $\overline{OP}$ and the integral of the vector field along all edges only yields nonzero value on $\overline{AB}$.

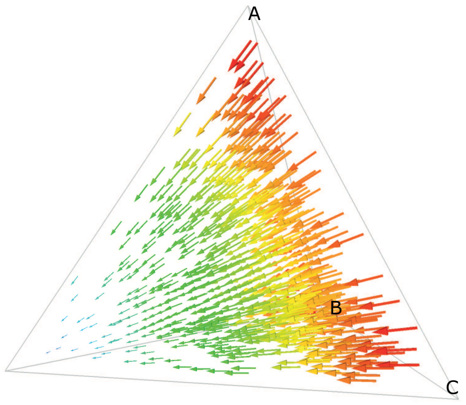
Furthermore, the Whitney 2-form vector with respect to the facet $\Delta ABC$ is

$$\phi_{ABC} = 2\phi_A \mathrm{d}\phi_B \wedge \mathrm{d}\phi_C + 2\phi_B \mathrm{d}\phi_C \wedge \mathrm{d}\phi_A + 2\phi_C \mathrm{d}\phi_A \wedge \mathrm{d}\phi_B = \frac{2}{abc}(x, y, z). \tag{3.15}$$

The vector field associated with this result is shown in Fig. 3.5(2). For each vector, $\phi_{ABC}$ is parallel to $\overline{OP}$. Similar to the 1-form case, the integral of the vector field on all facets only yields nonzero value on $\Delta ABC$.



(1) 1-form

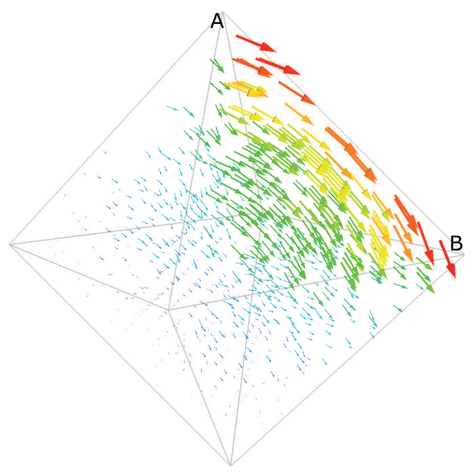


(2) 2-form

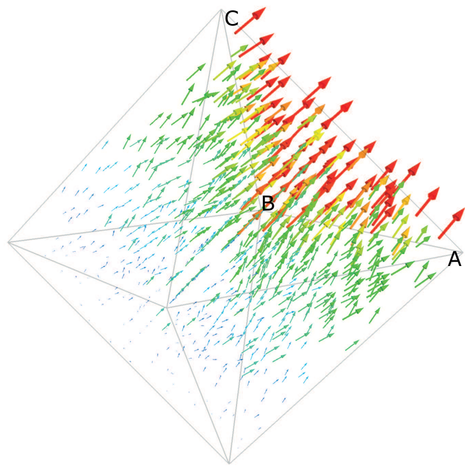
Figure 3.5: Whitney forms in the tetrahedral cell

### 3.2.3 Whitney Form in the Octahedron

Unfortunately, Whitney forms were originally only defined on simplices and there is no closed-form formula for the Whitney forms inside the octahedron. However, we can use linear 1- and 2-form subdivision (Section 4.1) to approximately model the Whitney forms in the octahedron. We only present the results here. Details of this method are discussed in Appendix A.3.2. As one can see from Fig. 3.6, integrating the vector field on all edges or facets yields nonzero value only on the edge or facet which carries nonzero 1- or 2-form coefficient.



(1) 1-form



(2) 2-form

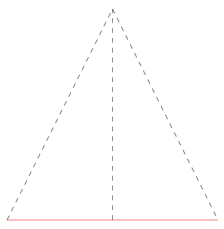
Figure 3.6: Whitney forms in the octahedral cell

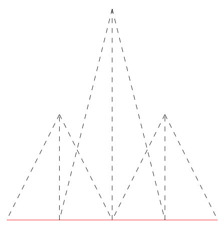
## 3.3 Cochain Subdivision

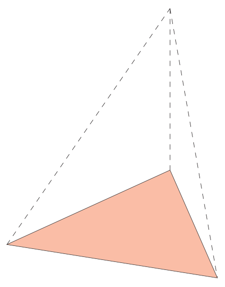
### 3.3.1 Subdividing the Vector Field

Refinable functions play a vital role as reconstruction bases in subdivision theory and scaling functions in wavelet theory. These functions have a certain level of self-similarity properties. For example, a single-variable function $f$ is said to be refinable with respect to the mask $h$ if it satisfies the refinement equation (also known as the dilation equation) $f(x) = \sum_{k\in\mathbb{Z}} h_k f(2x-k)$. The refinement equation can be interpreted as the discrete convolution with the discrete mask $h$ followed by a dilation operation.

Take Whitney 0-forms as an example. In one dimension, these are just piecewise-linear hat functions (Fig. 3.7(1)) which satisfy a simple two-scale dilation relation ($f(x) = \frac{1}{2}f(2x-1) + f(2x)+\frac{1}{2}f(2x+1)$, Fig. 3.7(2)) that yields piecewise-linear interpolation if we subdivide the function infinitely often.


(1) Coarse level, 1D


(2) Refined level, 1D


(3) Coarse level, 2D

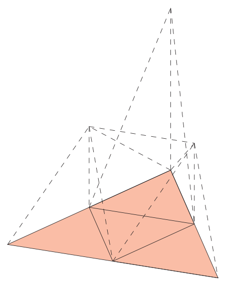

(4) Refined level, 2D

Figure 3.7: Refinement equations in 1D and 2D cases

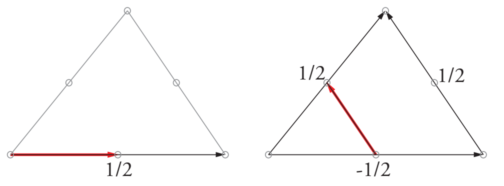
We can generalize the concept of refinability into higher dimensions. The function $f(\boldsymbol{x})$ on $\mathbb{R}^2$ is said to be refinable with respect to the discrete mask $h$ if it satisfies the refinement equation $f(x) = \sum_{\boldsymbol{k}\in\mathbb{Z}^2} h_{\boldsymbol{k}} f(2\boldsymbol{x} - \boldsymbol{k})$. The Whitney 0-form satisfies a similar refinement equation that also yields piecewise-linear interpolation (Fig. 3.7(4)). Furthermore, Whitney $r$-forms can be derived from Whitney $(r-1)$-forms (Eq.(3.8)). Thus, in the two-dimensional case, as Whitney 1-forms are induced by piecewise-linear functions, the refinement equation for Whitney 1-forms result in a piecewise-linear interpolation of vectors valued at vertices.

For $f(\boldsymbol{x})$ on $\mathbb{R}^3$, the refinement equation can be similarly defined as

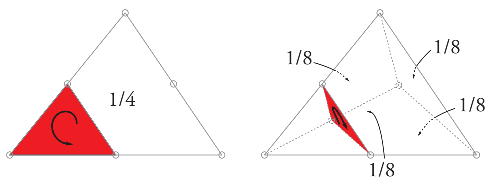
$$f(x) = \sum_{\boldsymbol{k}\in\mathbb{Z}^3} h_{\boldsymbol{k}} f(2\boldsymbol{x} - \boldsymbol{k}). \tag{3.16}$$

Although the piecewise-linear hat function in 3D domain can no longer be visualized, the concept is identical. Similarly, the refinement equations for 1- and 2-forms lead to a piecewise-linear interpolation of vectors valued at vertices. One can observe these facts by inspecting results in Fig. 3.5.

However, such subdivision rules for vector fields have at least two drawbacks. First, for the octahedral cell, the refinement relation is not obvious: we can no longer utilize a piecewise-linear interpolation of vectors valued at vertices. Second, vector fields rely not only on the topological structure of the mesh, but also on the metric space the mesh embedded in. What we really want is the refinement rule of the $r$-form coefficients which lie on $r$-simplices and is based only on the topological structure. Next, based on the refinement relation for vector fields (i.e., linear interpolation), we introduce the concept and notation of the refinement rules of the form coefficients.



(1) Whitney subdivision stencil for 1-forms in the tetrahedral cell



(2) Whitney subdivision stencil for 2-forms in the tetrahedral cell

Figure 3.8: Linear subdivision schemes for vector fields associated with Whitney 1- and 2-forms in the tetrahedral cell

The 1- and 2-form subdivision rules calculated from the linearly interpolated vector field are illustrated in Fig. 3.8. The stencil is calculated by integrating the vector field along the simplices in the refined mesh and is called the *Whitney subdivision scheme* or *linear subdivision scheme*. This will serve as the basis of a smoother subdivision scheme which we construct later. The subdivision rule for octahedral cells is still missing here. Assume the octahedral cells also have a refinement relation in a similar formulation, then for the whole octet mesh, by iterating the cells around a specific edge or facet, we are able to write down a refinement equations that in the format of matrix equations. For completeness, we write down such generic expression for the 0-, 1-, 2- and 3-forms in the octet meshes:

$$\Phi^v = \phi^v S_v^L, \;\; \Phi^e = \phi^e S_e^L, \;\; \Phi^f = \phi^f S_f^L, \;\; \Phi^c = \phi^t S_c^L. \tag{3.17}$$

Here, $\Phi$s represent the forms associated with the coarse mesh while $\phi$s correspond to the fine mesh.

Because of the difficulties for octahedral cells and the coordinate-dependent nature of the vector field subdivision approach, we start to look for a subdivision scheme which purely relies on the topological structure of the mesh. These algorithms, which are called *cochain subdivision schemes*, update the 1- and 2-form coefficient fields from the coarse mesh to the refined mesh and reconstructing the vector field (e.g., by Whitney forms) after the subdivision. In the next section, we present a formal representation of refinement equations for form coefficient fields in the octet mesh subdivision problems.

### 3.3.2 Subdividing 1- and 2-Form Coefficient Fields

Consider differential forms on a 3D regular octet mesh $\mathcal{M}$ in $\mathbb{R}^3$ space. As we discussed in Section 2.1.2, the geometric elements in a regular octet mesh can be classified into several equivalence classes based on a one-to-one mapping to the $\mathbb{Z}^3$-lattice. Assume the $r$-simplices in $\mathcal{M}$ can be divided into $N_r$ equivalence classes, then the $r$-form coefficients associated with these $r$-simplices can also be grouped into an $N_r$-vector $\Phi^r(\boldsymbol{x}) = (\Phi_1^r(\boldsymbol{x}), \cdots, \Phi_{N_r}^r(\boldsymbol{x}))$. We extend the definition of the refinement equation in Eq.(3.16) into the matrix formulation, i.e., if there exist $N_t \times N_t$ matrices $P_{\boldsymbol{k}}^r$ such that

$$\Phi^r(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^3} P_{\boldsymbol{k}}^r \Phi^r(2\boldsymbol{x} - \boldsymbol{k}), \tag{3.18}$$

the $r$-form $\Phi^r$ is said to be *refinable*.

*Remark:* Note that the above definition is well posed for $r = 0, 1, 2$ and 3. Although the translation step size of the tetrahedral or octahedral equivalence class is $2 \times \mathbb{Z}^3$, in the refinement equation we still need to sum over the $\mathbb{Z}^3$ lattice because the 3-form coefficient of a specific tetrahedron (octahedron) in the coarse level not only depends on the coefficients of the tetrahedron (octahedron) but also depends on the coefficients of the octahedron (tetrahedron) in the refined level.

Indeed, an $n \times n$ matrix coefficient $\{P_{\boldsymbol{k}} : \boldsymbol{k} \in \mathbb{Z}^3\}$ induces a *uniform stationary subdivision scheme*. If $\{\phi_{\boldsymbol{k}} \in \mathbb{R}^n : \boldsymbol{k} \in \mathbb{Z}^3\}$ in the coarse level are given, we can subdivide them and obtain $\{\Phi_{\boldsymbol{k}} \in \mathbb{R}^n : \boldsymbol{k} \in \mathbb{Z}^3\}$ in the refined level by computing

$$\Phi = S\phi, \quad \text{s.t. } \Phi_{\boldsymbol{k}} = \sum_{\boldsymbol{l}\in\mathbb{Z}^3} P_{\boldsymbol{k}-2\boldsymbol{l}}\phi_{\boldsymbol{l}}, \;\text{ for } \forall \boldsymbol{k} \in \mathbb{Z}^3. \tag{3.19}$$

Furthermore, if for any set of initial control vectors $\phi^0 \in \mathbb{R}^n \times \mathbb{Z}^3$ there exits a continuous function $f: \mathbb{R}^3 \to \mathbb{R}^n$ such that $\lim_{j\to\infty} \sup_{\boldsymbol{k}\in\mathbb{Z}^3} ||(S^j f^0)_{\boldsymbol{k}} - f(2^{-j}\boldsymbol{k})|| = 0$, the subdivision scheme is said to be uniformly *convergent*. One can refer to [13] and [22] for details.

**Fourier Domain** Because each element in one of the $N_r \times N_r$ blocks in $\{P^r_{\boldsymbol{k}}\}$ corresponds to a point in the $\mathbb{Z}^3$ lattice, we can perform the discrete Fourier transformation on the matrix coefficients $\{P^r_{\boldsymbol{k}}\}_{\boldsymbol{k}\in\mathbb{Z}^3}$ to get the generating function of the corresponding subdivision stencil by computing

$$\widehat{S_r}(\boldsymbol{z}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^3} P^r_{\boldsymbol{k}} \boldsymbol{z}^{\boldsymbol{k}}, \quad \boldsymbol{z} = (x, y, z). \tag{3.20}$$

Mapping $\widehat{S}^r$ back to the subdivision scheme in the real domain is straightforward. To obtain the contribution of an $r$-form in the coarse mesh to an $r$-form in the refined mesh, we just collect the coefficient of the term with the correct power in the generating polynomial.

One may wonder why we need the matrix subdivision scheme in the Fourier domain: after all, we update form coefficients on the mesh in the spatial domain. We do so because the generating function not only provides a succinct expression of the subdivision scheme, it also simplifies the process of creating smoother subdivision schemes by convolving with a smoothing operator.

## 3.4 Commutative Relations and Smooth Subdivision Schemes

Given a simplicial complex $\mathcal{K}_d$ with the corresponding continuous manifold $\mathcal{K}$, if $\omega \in \Omega^r(\mathcal{K})$, then $\mathrm{d}\omega \in \Omega^{r+1}(\mathcal{K})$. For any $(r+1)$-chain $c \in \mathcal{C}_{r+1}(\mathcal{K}_d)$, we know $\partial c \in \mathcal{C}_r(\mathcal{K}_d))$, *Stokes' Theorem* says

$$R(\mathrm{d}\omega) \cdot A = R(\omega) \cdot \partial A = \mathrm{d}R(\omega) \cdot A \;\; \Rightarrow \;\; R(\mathrm{d}\omega) = \mathrm{d}R(\omega) \tag{3.21}$$

where $R$ is the *de Rham map* mentioned in Section 3.2.1. Correspondingly, Whitney forms defined in Eq.(3.8) satisfy *commutative relations*. That is, given an $r$-simplex $\sigma_r \in \mathcal{K}_d$, for any $\omega_d \in \mathcal{C}^r(\mathcal{K}_d)$ the Whitney form $W\sigma_r$ satisfies

$$W \cdot \mathrm{d}\omega_d = \mathrm{d}W \cdot \omega_d. \tag{3.22}$$

As an example, we verify Eq.(3.22) using the explicit expression of Whitney forms (Eqs.(3.9)–(3.11)). Specifically, consider a 3-dimensional simplex $v_i v_j v_k v_l$. For $r = 1$, between 0- and 1-forms we have:

$$\begin{aligned} &\mathrm{d}(c_i\phi_i + c_j\phi_j + c_k\phi_k + c_l\phi_l) \\ &= (c_j - c_i)(\phi_i \mathrm{d}\phi_j - \phi_j \mathrm{d}\phi_i) + (c_k - c_i)(\phi_i \mathrm{d}\phi_k - \phi_k \mathrm{d}\phi_i) + (c_l - c_i)(\phi_i \mathrm{d}\phi_l - \phi_l \mathrm{d}\phi_i) \\ &\quad + (c_k - c_j)(\phi_j \mathrm{d}\phi_k - \phi_k \mathrm{d}\phi_j) + (c_l - c_j)(\phi_j \mathrm{d}\phi_l - \phi_l \mathrm{d}\phi_j) + (c_l - c_k)(\phi_k \mathrm{d}\phi_l - \phi_l \mathrm{d}\phi_k) \\ &= (c_j - c_i)\phi_{ij} + (c_k - c_i)\phi_{ik} + (c_l - c_i)\phi_{il} + (c_k - c_j)\phi_{jk} + (c_l - c_j)\phi_{jl} + (c_l - c_k)\phi_{kl}. \end{aligned} \tag{3.23}$$

Similarly, we can verify Eq.(3.22) by noticing that

$$\begin{aligned}
&\mathrm{d}(c_{ij}\phi_{ij} + c_{ik}\phi_{ik} + c_{il}\phi_{il} + c_{jk}\phi_{jk} + c_{jl}\phi_{jl} + c_{kl}\phi_{kl}) \\
&= (c_{ij} + c_{jk} - c_{ik})\phi_{ijk} + (c_{ij} + c_{jl} - c_{il})\phi_{ijl} + (c_{ik} + c_{kl} - c_{il})\phi_{ikl} + (c_{jk} + c_{kl} - c_{jl})\phi_{jkl}
\end{aligned} \tag{3.24}$$

and

$$\mathrm{d}(c_{ijk}\phi_{ijk} + c_{ijl}\phi_{ijl} + c_{ikl}\phi_{ikl} + c_{jkl}\phi_{jkl}) = (c_{ijk} - c_{ijl} + c_{ikl} - c_{jkl})\phi_{ijkl}. \tag{3.25}$$

Notice that the coefficient lying on an edge is the difference, or signed sum, of the coefficients at the two ends (the value at the end minus the value at the beginning). The signs depend on orientations of edges. Similarly, the coefficient lying on a facet is the signed sum of the coefficients on surrounding edges whose signs depend on orientations of both facets and edges: the orientation of the facet induces orientations of the surrounding 3 edges (e.g., in a clockwise manner). If the real orientation of the edge equals to the induced orientation, then the sign is positive, otherwise it is negative. Finally, the coefficient of a tetrahedron is the signed sum of the coefficients on surrounding facets whose signs depend on orientations of facets. If the facet points towards the interior of the tetrahedron, the sign is negative, otherwise it is positive.

Actually, the de Rham complex of Whitney forms on a 3D simplicial complex embedded in $\mathbb{R}^3$ can be expressed using vector calculus operators as

$$0 \to \mathcal{W}^0(M) \xrightarrow{\nabla} \mathcal{W}^1(M) \xrightarrow{\nabla\times} \mathcal{W}^2(M) \xrightarrow{\nabla\cdot} \mathcal{W}^3(M) \to 0. \tag{3.26}$$

Furthermore, in the 3D case, Eq.(3.22) together with the refinement relations in Eqs.(3.17) give us the commutative relations $\mathrm{d}S_v^L = S_e^L\mathrm{d}$, $\mathrm{d}S_e^L = S_f^L\mathrm{d}$, $\mathrm{d}S_f^L = S_c^L\mathrm{d}$ which, as we will see soon, play an crucial role during our construction of 1- and 2-forms subdivision rules.

The subdivision operators for the general finite subdivision stencil introduced in Eq.(3.19) also follow similar commutative relations,

$$\mathrm{d}^v S_v = S_e\mathrm{d}^v, \quad \mathrm{d}^e S_e = S_f\mathrm{d}^f, \quad \mathrm{d}^f S_f = S_c\mathrm{d}^f, \tag{3.27}$$

if the kernel of the coboundary operator is an invariant subspace of the subdivision operator [42].

In Section 2.2.2 we mentioned that, in the regular setting, subdivision schemes with higher regularity can be constructed using convolution. From Eq.(2.11) we know that, based on the linear subdivision scheme, the smoothing filter in the Fourier domain is

$$C = \frac{1}{8}\widehat{S_v^L}, \tag{3.28}$$

where $S_v^L$ is explicitly given in Eq.(2.10). Similar to Eq. (2.11), we can construct the generating functions for the smoother 1-, 2- and 3-form subdivision schemes by computing

$$\widehat{S_e} = C\widehat{S_e^L}, \quad \widehat{S_f} = C\widehat{S_f^L}, \quad \widehat{S_c} = C\widehat{S_c^L}. \tag{3.29}$$

Note that, unlike in the 0-form case, the generating functions for these higher-order form subdivision stencils are matrices. However, because the smoothing operator $C$ is a scalar, the above constructions are still well defined and we can write down the stencil in the real domain through inverse Fourier transformation. This corresponds to simply collecting the coefficients associated with corresponding polynomial terms.

Similarly to the subdivision operators, the coboundary operators can be represented in terms of generating polynomials. For example, from Fig. 2.7, we know

$$\widehat{\mathrm{d}^v} = \begin{pmatrix} -1+x^{-1} \\ -1+y^{-1} \\ -1+x^{-1}y^{-1}z^{-1} \\ -1+y^{-1}z^{-1} \\ -1+z^{-1} \\ -1+x^{-1}z^{-1} \end{pmatrix}, \quad \widehat{\mathrm{d}^e} = \begin{pmatrix} 0 & x^{-1}z^{-1} & -1 & 0 & 0 & 1 \\ -y^{-1}z^{-1} & 0 & 1 & -1 & 0 & 0 \\ 0 & -z^{-1} & 0 & 1 & -1 & 0 \\ z^{-1} & 0 & 0 & 0 & 1 & -1 \\ 0 & -yz & 0 & yz & -z & 0 \\ xz & 0 & 0 & 0 & z & -xz \\ 0 & xyz & -xyz & 0 & 0 & xz \\ -xyz & 0 & xyz & -yz & 0 & 0 \end{pmatrix},$$

$$\widehat{\mathrm{d}^f} = \begin{pmatrix} 0 & 1 & 0 & y^{-1} & y^{-1}z^{-1} & 0 & x^{-1}y^{-1}z^{-1} & 0 \\ 1 & 0 & x^{-1} & 0 & 0 & x^{-1}z^{-1} & 0 & x^{-1}y^{-1}z^{-1} \\ 1 & 1 & 1 & 1 & x^{-1}y^{-1}z^{-2} & x^{-1}y^{-1}z^{-2} & x^{-1}y^{-1}z^{-2} & x^{-1}y^{-1}z^{-2} \end{pmatrix}. \tag{3.30}$$

It is simple to verify that the commutative relations (Eq.(3.27)) also hold for linear subdivision schemes in the Fourier domain, i.e.,

$$\widehat{\mathrm{d}^v}\widehat{S_v^L} = \widehat{S_e^L}\widehat{\mathrm{d}^v}, \quad \widehat{\mathrm{d}^e}\widehat{S_e^L} = \widehat{S_f^L}\widehat{\mathrm{d}^e}, \quad \widehat{\mathrm{d}^f}\widehat{S_f^L} = \widehat{S_c^L}\widehat{\mathrm{d}^f}. \tag{3.31}$$

For the smooth schemes constructed in Eq.(3.29), because the smoothing operator $C$ is just a scalar, we have $\widehat{\mathrm{d}^v}\widehat{S_v} = \widehat{\mathrm{d}^v}C\widehat{S_v^L} = C\widehat{\mathrm{d}^v}\widehat{S_v^L} = C\widehat{S_e^L}\widehat{\mathrm{d}^v} = \widehat{S_e}\widehat{\mathrm{d}^v}$. Similarly we can prove $\widehat{\mathrm{d}^e}\widehat{S_e} = \widehat{S_f}\widehat{\mathrm{d}^e}$ and $\widehat{\mathrm{d}^f}\widehat{S_f} = \widehat{S_c}\widehat{\mathrm{d}^f}$. Thus, the commutative relations also hold in the Fourier domain for these smooth subdivision stencils.

# Chapter 4

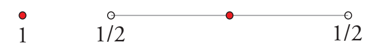
# Subdivision in the Regular Case

This chapter constructs smooth 1- and 2-form subdivision stencils step-by-step using the theory introduced in Chapter 3. Section 4.1 reviews the construction of linear subdivision schemes. The commutative relations between 0-, 1-, 2- and 3-form subdivision operators determine the stencils up to one free parameter. In section 4.2, we carry this free parameter and construct smooth subdivision schemes using convolutions. Section 4.3 develops a method to determine the free parameter based on spectral and momentum considerations.

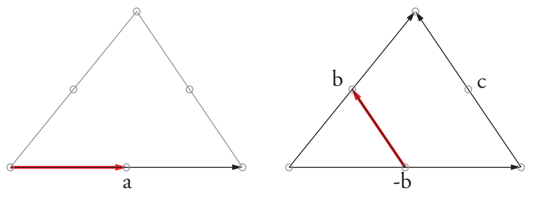
## 4.1 Linear Schemes for 1- and 2-Form Subdivision

### 4.1.1 Tetrahedral Cells

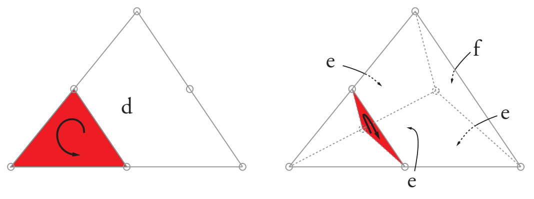
The indexing rule for a single tetrahedral cell is illustrated in Fig 3.1. Here, Fig 4.1 shows the topological setting of the linear stencil. The orientation of edges are marked with arrow heads. To orient facets, we use the convention that directions always point towards the exterior of tetrahedra.



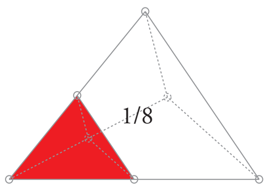
(1) 0-form stencil



(2) 1-form stencil



(3) 2-form stencil



(4) 3-form stencil

Figure 4.1: Linear subdivision stencil for a single tetrahedral cell

Linear subdivision matrices and boundary operators for a single tetrahedral cell can be written as

$$S^L_{v,t} = \left(\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{array}\right), \quad S^L_{e,t} = \left(\begin{array}{ccc|ccc} a_t & 0 & 0 & 0 & 0 & 0 \\ 0 & a_t & 0 & 0 & 0 & 0 \\ 0 & 0 & a_t & 0 & 0 & 0 \\ \hline -b_t & b_t & 0 & c_t & 0 & 0 \\ 0 & -b_t & b_t & 0 & c_t & 0 \\ b_t & 0 & -b_t & 0 & 0 & c_t \end{array}\right), \quad S^L_{f,t} = \left(\begin{array}{ccc|c} d_t & 0 & 0 & 0 \\ 0 & d_t & 0 & 0 \\ 0 & 0 & d_t & 0 \\ \hline -e_t & -e_t & -e_t & f_t \end{array}\right), \quad S^L_{c,t} = \left(\frac{1}{8}\right), \tag{4.1}$$

$$d^{v,L}_t = \left(\begin{array}{c|ccc} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ \hline 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 1 & 0 & -1 \end{array}\right), \quad d^{e,L}_t = \left(\begin{array}{ccc|ccc} 1 & -1 & 0 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 & 1 & 0 \\ -1 & 0 & 1 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & -1 & -1 & -1 \end{array}\right), \quad d^{f,L}_t = \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \end{array}\right), \tag{4.2}$$

where we use the footnote $t$ to distinguish these coefficient from the octahedral case later. Substituting these variables into the commutative relations Eqs(3.27), we have

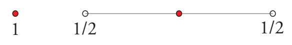
$$\begin{cases} d^{v,L}_t S^L_{v,t} = S^L_{e,t} d^{v,L}_t \\ d^{e,L}_t S^L_{e,t} = S^L_{f,t} d^{e,L}_t \\ d^{f,L}_t S^L_{f,t} = S^L_{c,t} d^{f,L}_t \end{cases} \Rightarrow \quad a_t = \frac{1}{2}, \; b_t = \frac{1}{4}, \; c_t = \frac{1}{4}, \; d_t = \frac{1}{4}, \; e_t = \frac{1}{8}, \; f_t = \frac{1}{8}. \tag{4.3}$$

*Remark:* From Fig. 4.1(2) and Fig. 4.1(3) (left) we find that, for linear subdivision schemes, if the target geometric object in the refined mesh is restricted to a facet $f$ in the coarse mesh, then the stencil is $f$. Consider subdivision schemes restricted on $f$: the subdivision results should be identical to the results from subdivision schemes of Whitney forms on triangles (i.e., the 2D surface case). Based on this fact, we can directly conclude that $a_t = \frac{1}{2}, \; b_t = \frac{1}{4}, \; c_t = \frac{1}{4}$ and $d_t = \frac{1}{4}$. After substituting these values into original subdivision matrices in Eq.(4.1), we can solve for $e_t$ and $f_t$ using commutative relations (which are now a smaller linear system) and get the stated solutions.

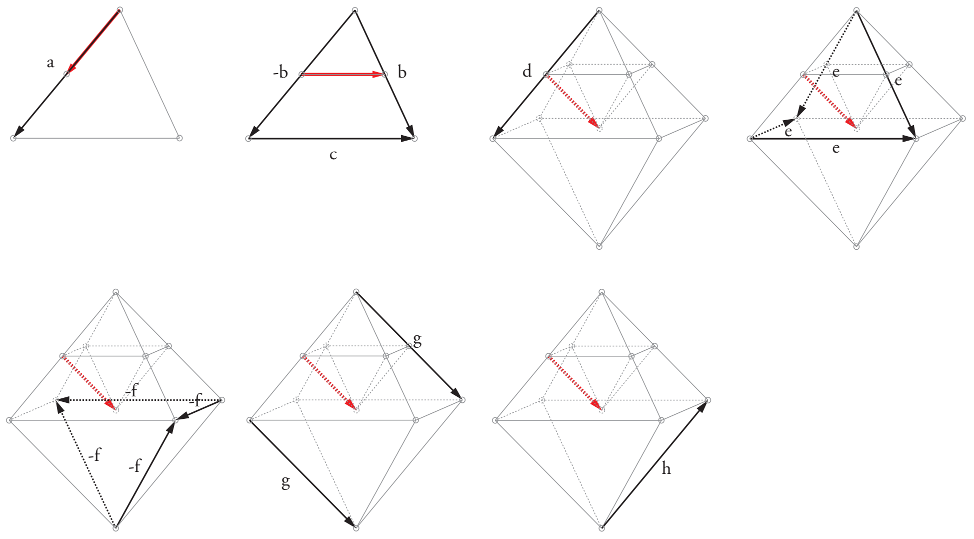
### 4.1.2 Octahedral Cells

The indexing rule and the topological setting for a single octahedral cell are illustrated in Fig. 3.2 and Fig. 4.2. We follow the convention that orientations of facets point towards the interior of their adjacent octahedral cells.

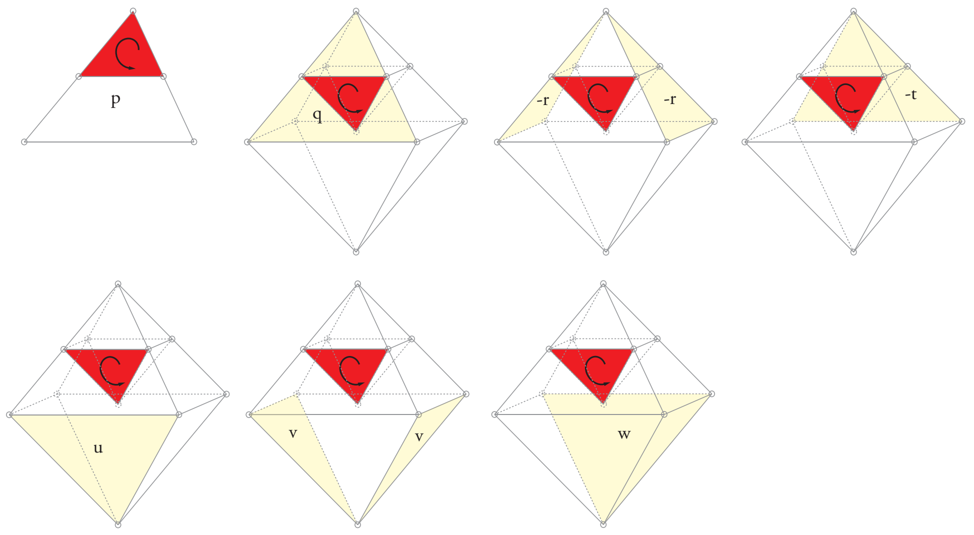
Similarly, linear subdivision matrices and boundary operators for a single octahedral cell can be



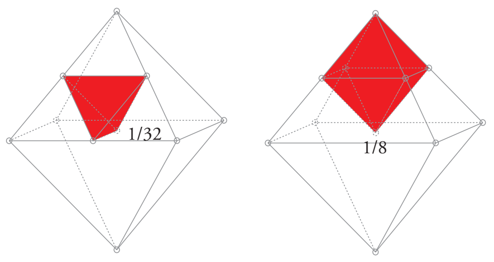
(1) 0-form stencil



(2) 1-form stencil



(3) 2-form stencil



(4) 3-form stencil

Figure 4.2: Linear subdivision stencil for a single octahedral cell

written as

$$S_{v,o}^L = \left(\begin{array}{c|cccc|c} 1 & 0 & 0 & 0 & 0 & 0 \\ \hline \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 \\ \hline \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \end{array}\right), \quad S_{e,o}^L = \left(\begin{array}{cccc|cccc|cccc} a_o & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_o & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_o & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_o & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline -b_o & b_o & 0 & 0 & c_o & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -b_o & b_o & 0 & 0 & c_o & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -b_o & b_o & 0 & 0 & c_o & 0 & 0 & 0 & 0 & 0 \\ b_o & 0 & 0 & -b_o & 0 & 0 & 0 & c_o & 0 & 0 & 0 & 0 \\ \hline d_o & e_o & g_o & e_o & e_o & f_o & -f_o & -e_o & g_o & f_o & h_o & f_o \\ e_o & d_o & e_o & g_o & -e_o & e_o & f_o & -f_o & f_o & g_o & f_o & h_o \\ g_o & e_o & d_o & e_o & -f_o & -e_o & e_o & f_o & h_o & f_o & g_o & f_o \\ e_o & g_o & e_o & d_o & f_o & -f_o & -e_o & e_o & f_o & h_o & f_o & g_o \end{array}\right),$$

$$S_{f,o}^L = \left(\begin{array}{cccc|cccc} p_o & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & p_o & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & p_o & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_o & 0 & 0 & 0 & 0 \\ \hline q_o & -r_o & -t_o & -r_o & u_o & v_o & -w_o & v_o \\ -r_o & q_o & -r_o & -t_o & v_o & u_o & v_o & -w_o \\ -t_o & -r_o & q_o & -r_o & -w_o & v_o & u_o & v_o \\ -r_o & -t_o & -r_o & q_o & v_o & -w_o & v_o & u_o \end{array}\right), \quad S_{c,o}^L = \left(\frac{1}{8}\right), \tag{4.4}$$

$$d_o^{v,L} = \left(\begin{array}{c|cccc|c} -1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ \hline 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{array}\right), \quad d_o^{e,L} = \left(\begin{array}{cccc|cccc|cccc} 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 1 \end{array}\right),$$

$$d_o^{f,L} = \left(\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right). \tag{4.5}$$

To simplify calculations, we use the argument mentioned above that takes advantage of the known linear stencil restricted to a facet, i.e.,

$$a_o = \frac{1}{2}, \;\; b_o = c_o = p_o = \frac{1}{4}. \tag{4.6}$$

Then we solve for the coefficients by substituting above matrices and operators into commutative relations:

$$\begin{cases} d_o^{v,L} S_{v,o}^L = S_{e,o}^L d_o^{v,L} \\ d_o^{e,L} S_{e,o}^L = S_{f,o}^L d_o^{e,L} \\ d_o^{f,L} S_{f,o}^L = S_{c,o}^L d_o^{f,L} \end{cases} \Rightarrow \quad \begin{array}{l} d_o = 0,\ e_o = \frac{17+\zeta}{96},\ f_o = \frac{9+\zeta}{96},\ g_o = -\frac{1+\zeta}{48},\ h_o = 0, \\ q_o = -\frac{7}{96},\ r_o = -\frac{\zeta}{96},\ t_o = \frac{19+2\zeta}{96},\ u_o = -\frac{3+2\zeta}{96},\ v_o = \frac{8+\zeta}{96},\ w_o = \frac{1}{96}. \end{array} \tag{4.7}$$

Here, the linear system is underdetermined so one free parameter $\zeta$ remains.

*Remark:* We are able to obtain $d_o = 0$ and $h_o = 0$ by using the symmetries in the topology. For example, from Fig. 4.2 we can see that, if we flip the orientation of the edge $d$ or $h$, the contribution of the flipped edge in the coarse mesh to the target edge in the refined mesh does not change. Therefore the contribution must be zero.

## 4.2 Smooth Subdivision Rules

### 4.2.1 Topological Settings

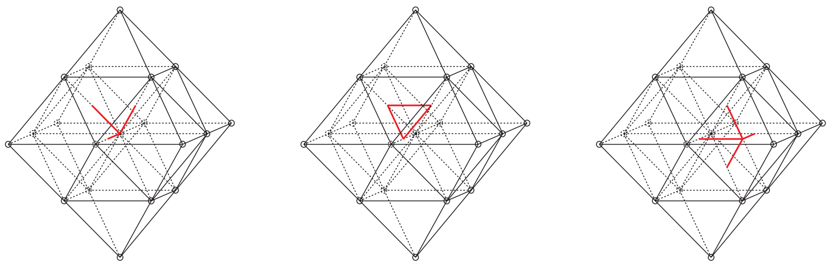
In this section, we construct smooth subdivision schemes based on linear schemess obtained in Section 4.1. Schemes with higher regularity have larger supports than linear schemes. As discussed in Section 2.2, vertices in the regular octet mesh can be grouped into three types: (V1) old vertices, (V2) edge centers and (V3) octahedron centroids (see Fig. 2.9). Here, we also group edges and facets into 3 and 4 types respectively:

*Edges:*

► *E1 edges:* Connected to one V1 vertex and one V2 vertex (Fig. 4.3(1)). These edges come from edges in the coarse mesh.

► *E2 edges:* Connected to two V2 vertices (Fig. 4.3(2)). These edges are in the interior of facets in the coarse mesh.

► *E3 edges:* Connected to one V2 vertex and one V3 vertex (Fig. 4.3(3)). These edges are in the interior of octahedra in the coarse mesh.



(1) E1 edges (2) E2 edges (3) E3 edges

Figure 4.3: Three types of edges. Note that for each type, we only highlight some of the edges for simplicity.

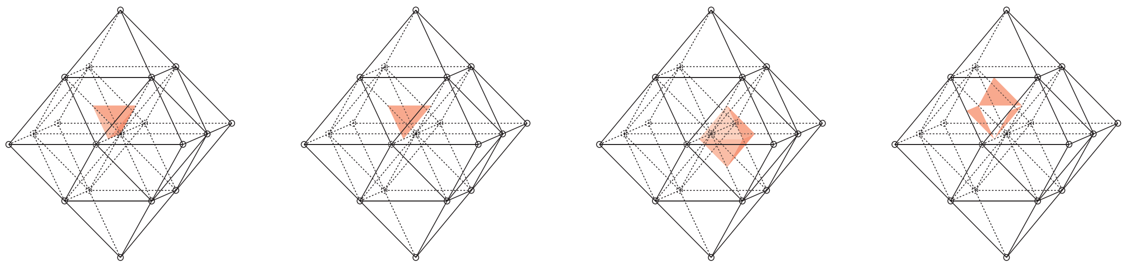

(1) F1 facets (2) F2 facets (3) F3 facets (4) F4 facets

Figure 4.4: Four types of facets. Note that for each type, we only highlight some of the facets for simplicity.

*Facets:*

▶ *F1 Facets:* Connected to one V1 vertex and two V2 vertices (Fig. 4.4(1)). These facets come from facets in the coarse mesh.

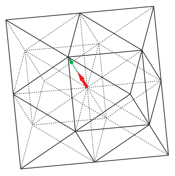
▶ *F2 Facets:* Connected to three V2 vertices (Fig. 4.4(2)). These facets are in the interior of tetrahedra in the coarse mesh.

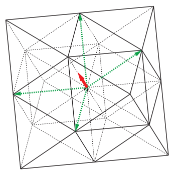
▶ *F3 Facets:* Connected to two V2 vertices and one V3 vertex (Fig. 4.4(3)). These facets are in the interior of octahedra in the coarse mesh.

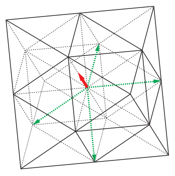
▶ *F4 Facets:* Connected to three V2 vertices (Fig. 4.4(4)). This type is the center piece of four refined facets after subdividing a facet in the coarse mesh.

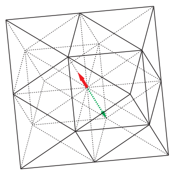
Specifically, stencil coefficient variables are shown in Fig. 4.5–4.11. For dimensions of subdivision matrices, $S_v$ is $(19 \times 19)$, $S_e$ is $(60 \times 60)$, $S_f$ is $(56 \times 56)$, $d^v$ is $(60 \times 19)$ and $d^e$ is $(56 \times 60)$. For succinctness, we include exact forms of these matrices in the supplementary electronic material rather than here.

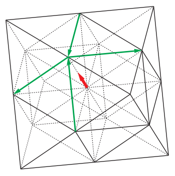
A simpler method for deriving these large subdivision matrices is to utilize symmetries in the regular octet mesh. In surface subdivisions, we can group edges into $L$ classes, e.g., in Fig. 1.5 where $L = 4$. Then the 1-form subdivision matrix can be written as an $(L \times L)$-block matrix in which every block has a circulant symmetric structure. In volumetric domain subdivisions, symmetries are more complicated than circulant symmetries. As an example, consider the contributions of E2 edges in the coarse mesh to E3 edges in the refined mesh (i.e., the (1,3) block of the subdivision matrix). As shown in Fig. 4.12, the blue edge in the coarse mesh have the same contribution to the red edge in the refined mesh for both figures. To identify this equivalence, one can flip/rotate the mesh using $8C_3$, $6C_2$ and $6C_4/3C_2$ operations shown in Fig. 2.6 until the red edge overlaps with the red edge in another mesh and then identify the location of the blue edge. Formally, these symmetries can be summarized using a permutation table (Eq.(4.8)). The column and row indices represent indices of edges in the coarse and refined mesh ($\{e_i^C\}_{i=1,\cdots,24}$ and $\{e_i^R\}_{i=1,\cdots,24}$). From the table we know that $e_i^C$ have the same contribution $A$ on $e_i^R$ for all $i = 1, \cdots, 24$. We can also express the

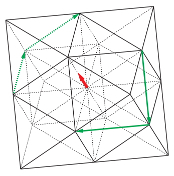

(1) $a_1$

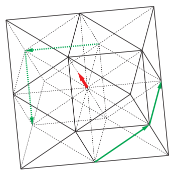

(2) $a_2$

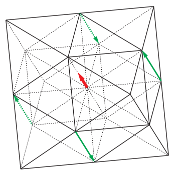

(3) $a_3$

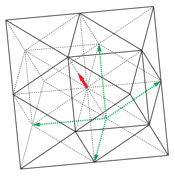

(4) $a_4$

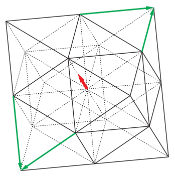

(5) $b_1$

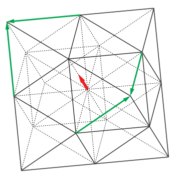

(6) $b_2$

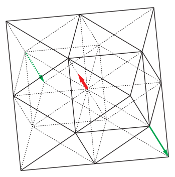

(7) $b_3$

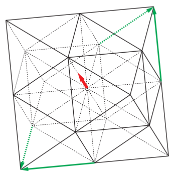

(8) $b_4$

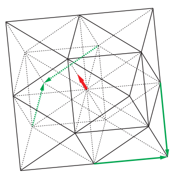

(9) $b_5$

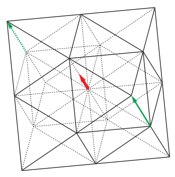

(10) $c_1$


(11) $c_2$


(12) $c_3$


(13) $c_4$


(14) $c_5$


(15) $c_6$

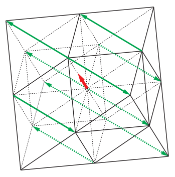

(16) 0

Figure 4.5: Smooth subdivision stencil for E1 edges

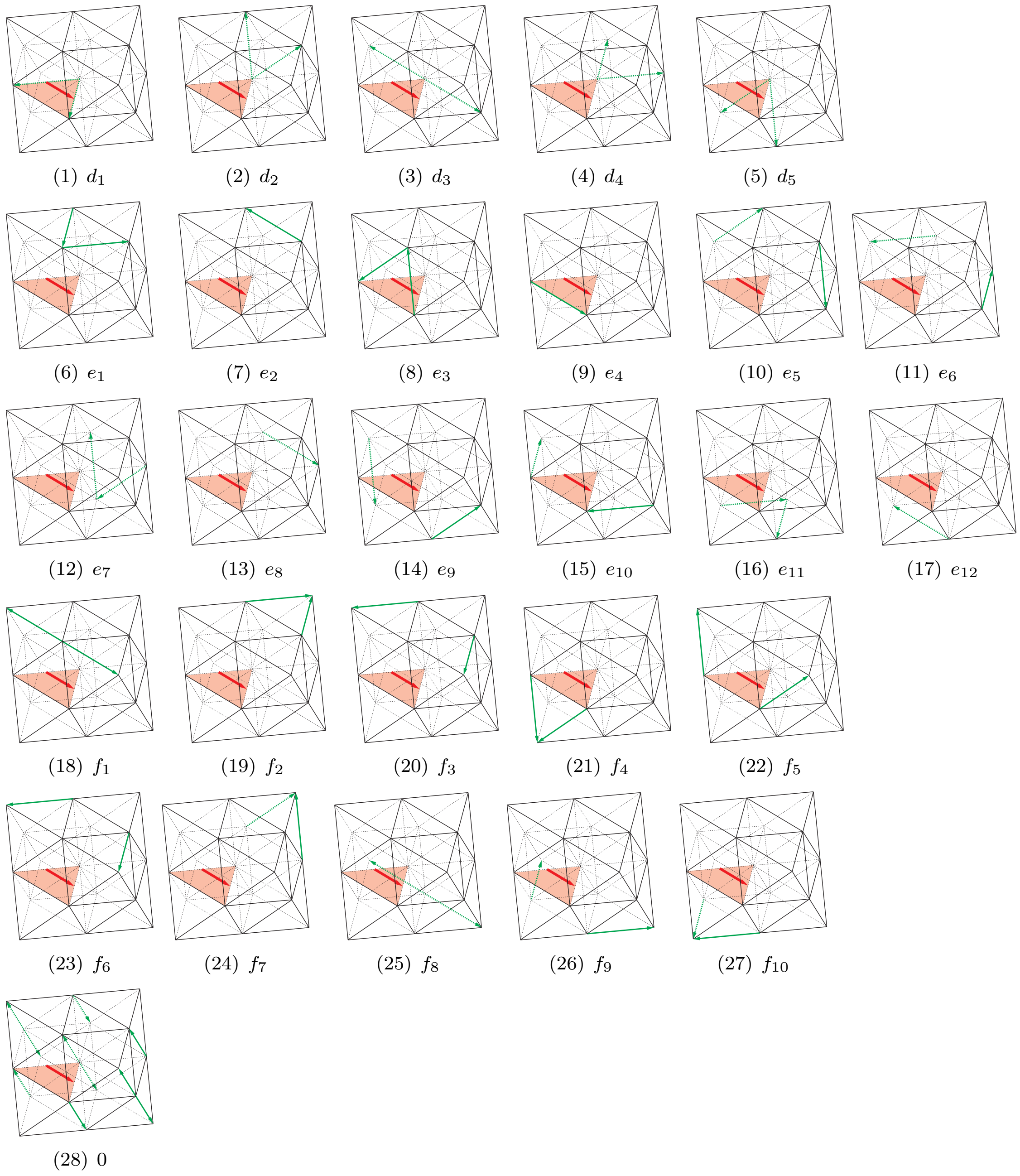


Figure 4.6: Smooth subdivision stencil for E2 edges

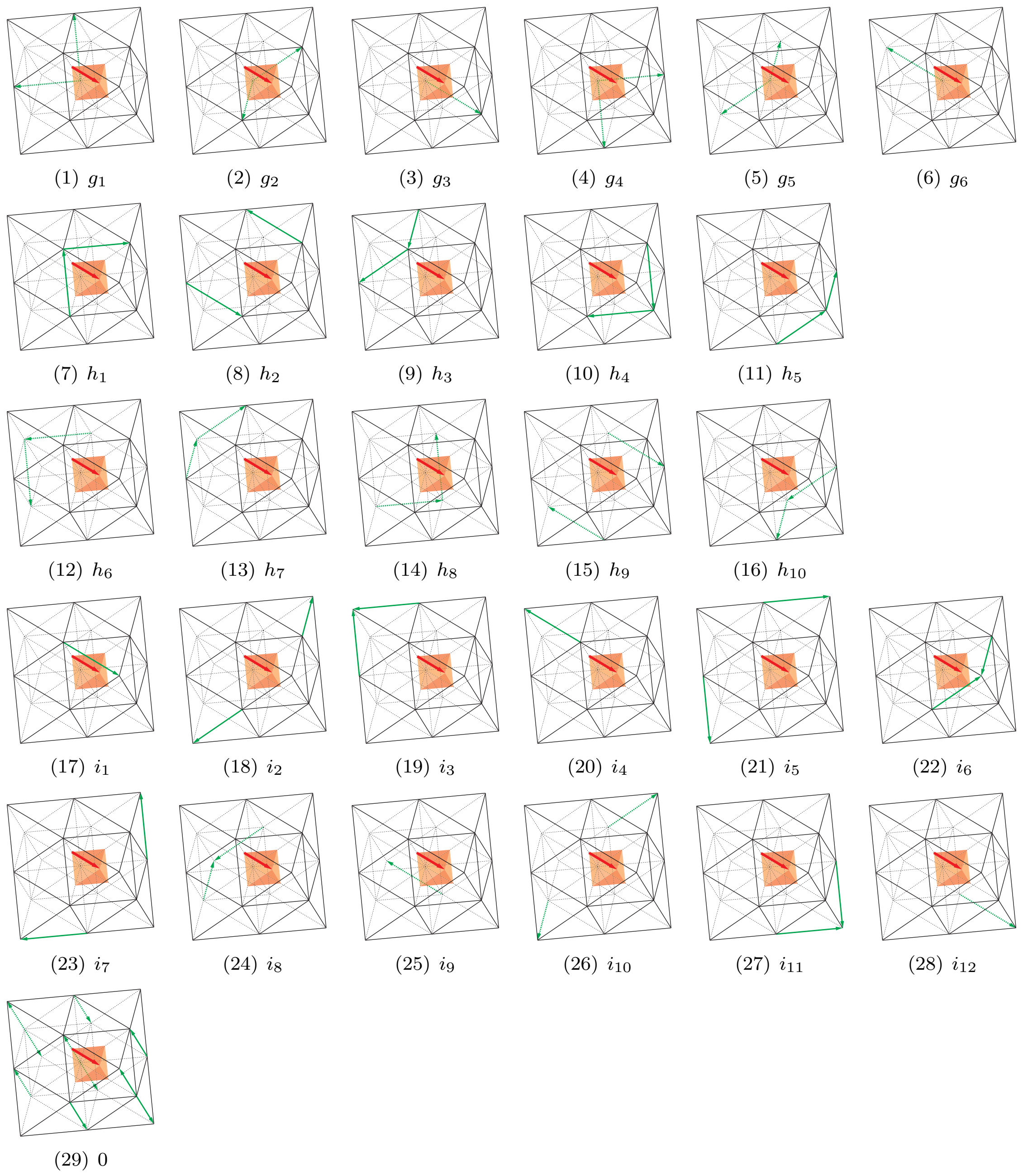


Figure 4.7: Smooth subdivision stencil for E3 edges

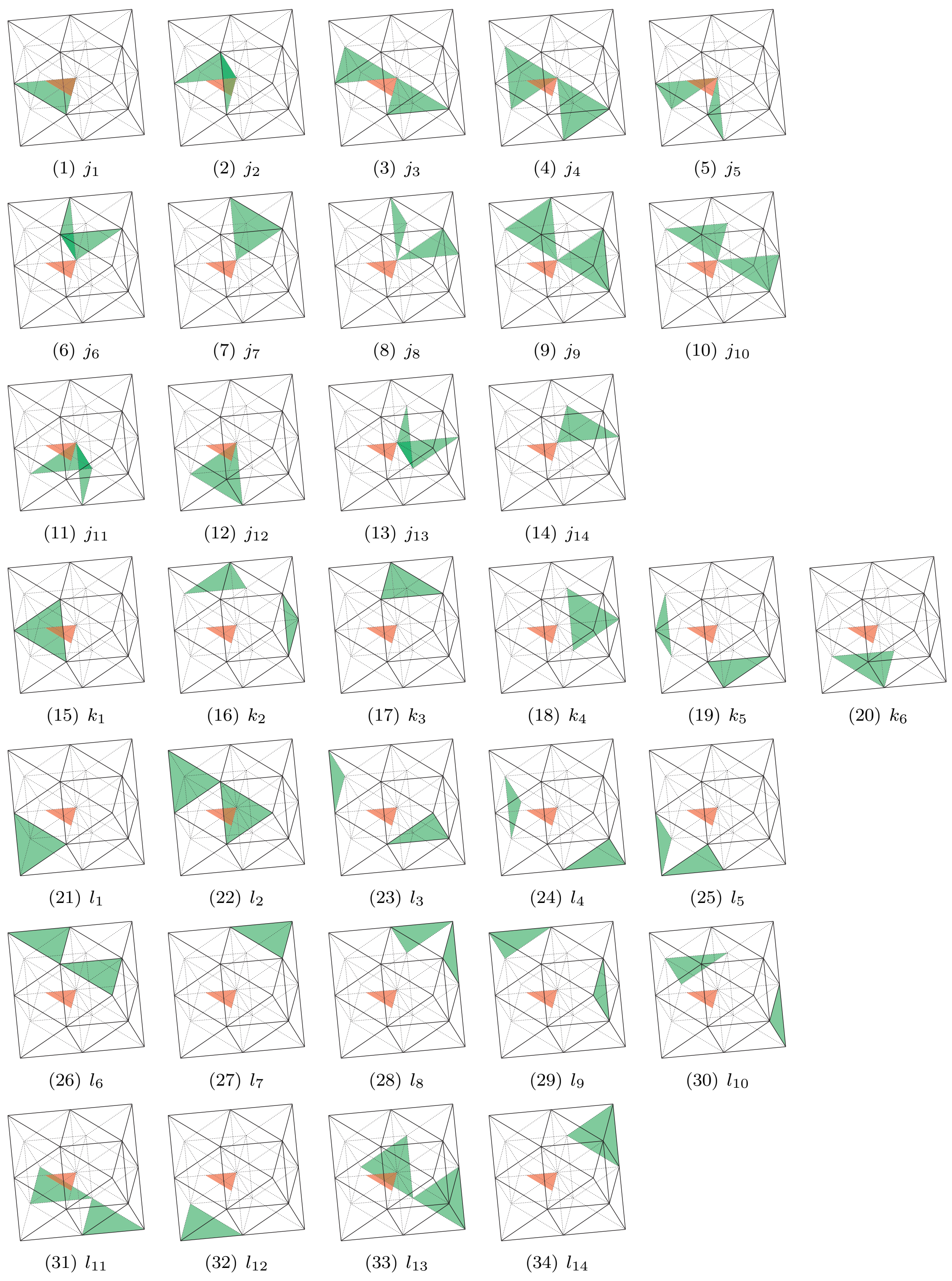


Figure 4.8: Smooth subdivision stencil for F1 facets

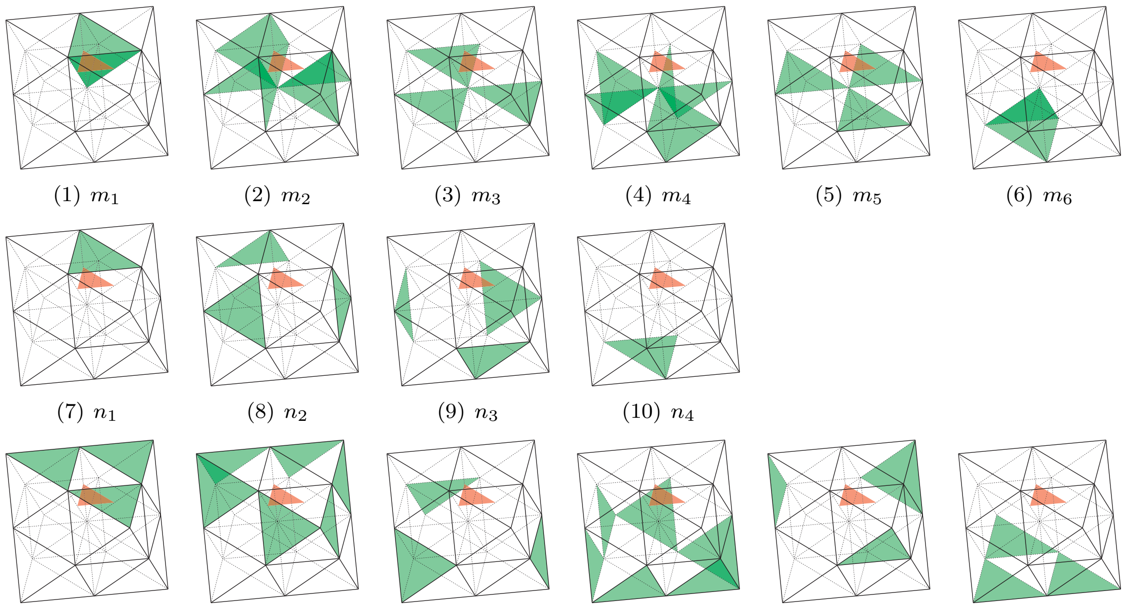
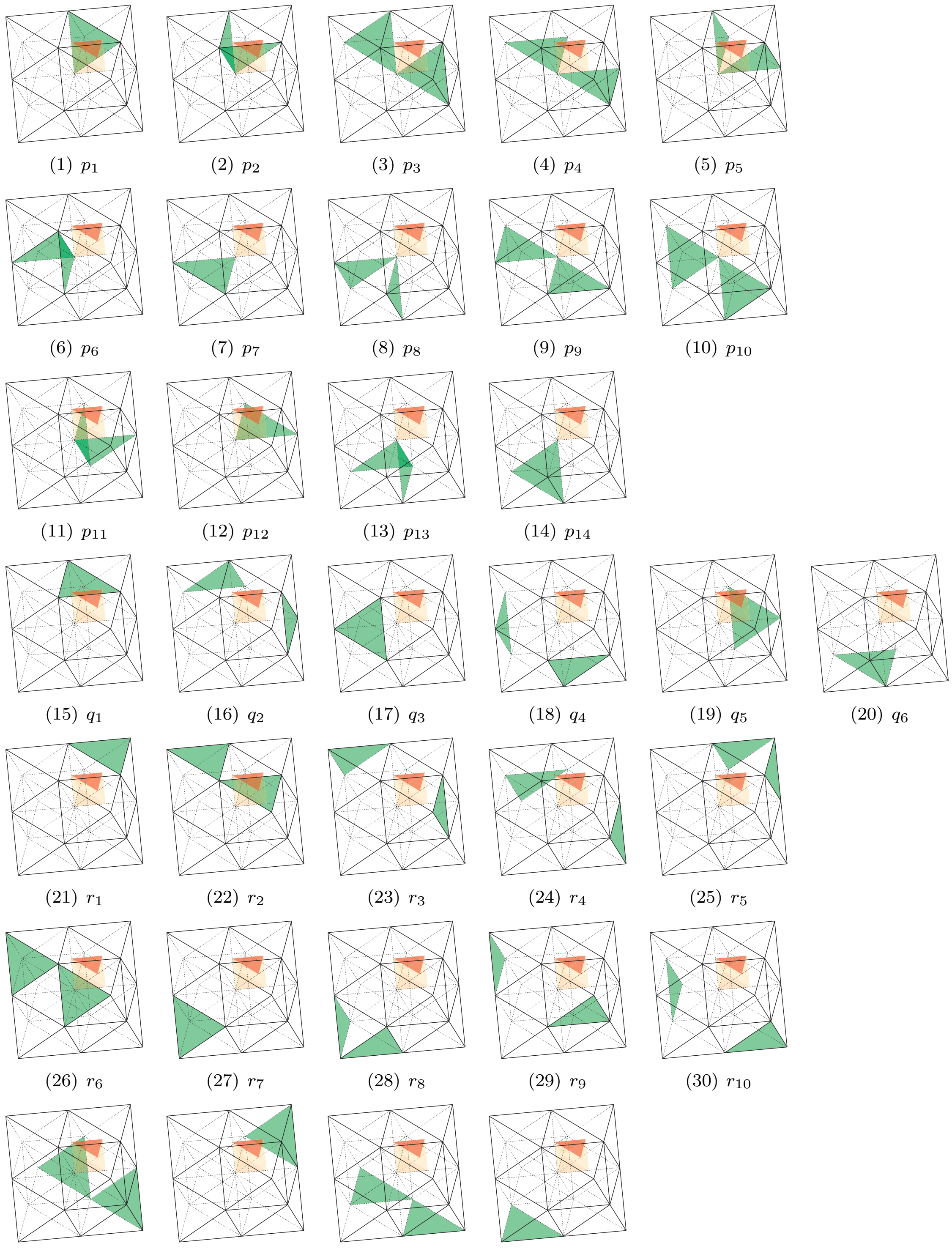


Figure 4.9: Smooth subdivision stencil for F2 facets

aforementioned example by saying "$e^C_{10}$ and $e^C_4$ have the contribution $J$ on $e^R_1$ and $e^R_2$, respectively". These permutation relations form a non-Abelian group. For each block in a subdivision matrix, we only have to write down the first row; the remainder of the block can be generated using the permutations table.

$$\left(\begin{array}{ccc|ccc|ccc|ccc|ccc|ccc|ccc|ccc}
A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X \\
C & A & B & J & K & L & D & E & F & G & H & I & P & Q & R & S & T & U & M & N & O & W & X & V \\
B & C & A & G & H & I & J & K & L & D & E & F & S & T & U & M & N & O & P & Q & R & X & V & W \\
\hline
D & E & F & A & B & C & T & U & S & R & P & Q & V & W & X & K & L & J & I & G & H & M & N & O \\
J & K & L & C & A & B & N & O & M & U & S & T & W & X & V & H & I & G & F & D & E & P & Q & R \\
G & H & I & B & C & A & Q & R & P & O & M & N & X & V & W & E & F & D & L & J & K & S & T & U \\
\hline
F & D & E & R & P & Q & A & B & C & T & U & S & K & L & J & I & G & H & V & W & X & N & O & M \\
L & J & K & U & S & T & C & A & B & N & O & M & H & I & G & F & D & E & W & X & V & Q & R & P \\
I & G & H & O & M & N & B & C & A & Q & R & P & E & F & D & L & J & K & X & V & W & T & U & S \\
\hline
E & F & D & T & U & S & R & P & Q & A & B & C & I & G & H & V & W & X & K & L & J & O & M & N \\
K & L & J & N & O & M & U & S & T & C & A & B & F & D & E & W & X & V & H & I & G & R & P & Q \\
H & I & G & Q & R & P & O & M & N & B & C & A & L & J & K & X & V & W & E & F & D & U & S & T \\
\hline
M & N & O & V & W & X & K & L & J & I & G & H & A & B & C & T & U & S & R & P & Q & D & E & F \\
P & Q & R & W & X & V & H & I & G & F & D & E & C & A & B & N & O & M & U & S & T & J & K & L \\
S & T & U & X & V & W & E & F & D & L & J & K & B & C & A & Q & R & P & O & M & N & G & H & I \\
\hline
N & O & M & K & L & J & I & G & H & V & W & X & R & P & Q & A & B & C & T & U & S & F & D & E \\
Q & R & P & H & I & G & F & D & E & W & X & V & U & S & T & C & A & B & N & O & M & L & J & K \\
T & U & S & E & F & D & L & J & K & X & V & W & O & M & N & B & C & A & Q & R & P & I & G & H \\
\hline
O & M & N & I & G & H & V & W & X & K & L & J & T & U & S & R & P & Q & A & B & C & E & F & D \\
R & P & Q & F & D & E & W & X & V & H & I & G & N & O & M & U & S & T & C & A & B & K & L & J \\
U & S & T & L & J & K & X & V & W & E & F & D & Q & R & P & O & M & N & B & C & A & H & I & G \\
\hline
V & W & X & M & N & O & P & Q & R & S & T & U & D & E & F & G & H & I & J & K & L & A & B & C \\
W & X & V & P & Q & R & S & T & U & M & N & O & J & K & L & D & E & F & G & H & I & C & A & B \\
X & V & W & S & T & U & M & N & O & P & Q & R & G & H & I & J & K & L & D & E & F & B & C & A
\end{array}\right)$$



(1) $p_1$ (2) $p_2$ (3) $p_3$ (4) $p_4$ (5) $p_5$

(6) $p_6$ (7) $p_7$ (8) $p_8$ (9) $p_9$ (10) $p_{10}$

(11) $p_{11}$ (12) $p_{12}$ (13) $p_{13}$ (14) $p_{14}$

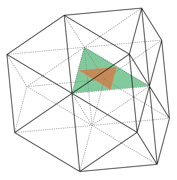
(15) $q_1$ (16) $q_2$ (17) $q_3$ (18) $q_4$ (19) $q_5$ (20) $q_6$

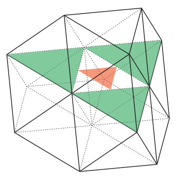
(21) $r_1$ (22) $r_2$ (23) $r_3$ (24) $r_4$ (25) $r_5$

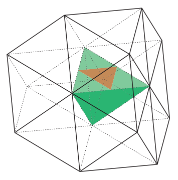
(26) $r_6$ (27) $r_7$ (28) $r_8$ (29) $r_9$ (30) $r_{10}$

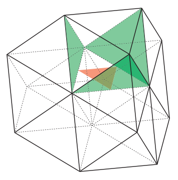
(31) $r_{11}$ (32) $r_{12}$ (33) $r_{13}$ (34) $r_{14}$

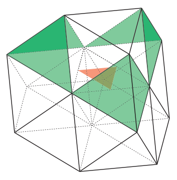
Figure 4.10: Smooth subdivision stencil for F3 facets

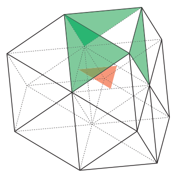

(1) $s_1$

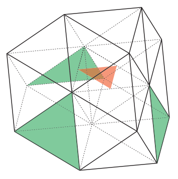

(2) $s_2$

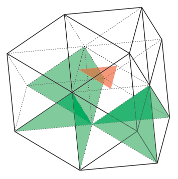

(3) $s_3$

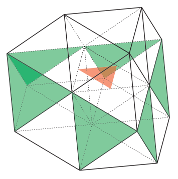

(4) $s_4$

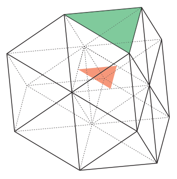

(5) $s_5$

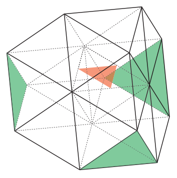

(6) $s_6$

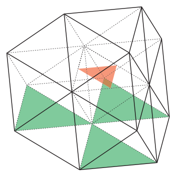

(7) $s_7$


(8) $s_8$


(9) $s_9$


(10) $s_{10}$


(11) $s_{11}$


(12) $s_{12}$

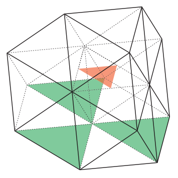
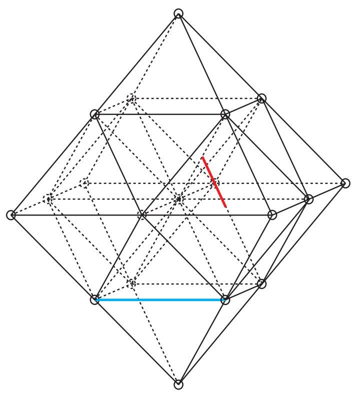

(13) $s_{13}$

Figure 4.11: Smooth subdivision stencil for F4 facets


(1) Case 1

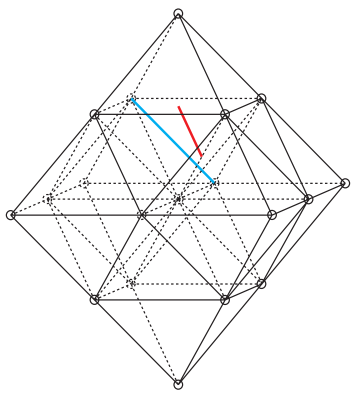

(2) Case 2

Figure 4.12: Example of equal contributions

After we write down all subdivision matrices and coboundary operators, we can substitute them into commutative relations and solve the resulting linear system. However, unlike the linear subdivision case where the solution is determined up to only one free parameter ($\zeta$), the linear system here is severely underdetermined. Nearly half of the unknown variables cannot be determined using this approach. However, these subdivision matrices are still useful when we start to calculate eigenforms of the subdivision scheme (see Section 4.3).

### 4.2.2 Generating Function and Discrete Convolution

In Section 3.3.2 we discussed that, in the regular setting, the Fourier transformation of a subdivision scheme yields the generating polynomial of the scheme. Section 3.4 introduced an approach to construct smooth subdivision schemes by multiplying the generating polynomials of linear subdivision schemes with a smooth operator. We implement this approach for the 3D 1- and 2-form subdivision problems in this chapter.

**1-Form Subdivision Generating Function** In Section 2.1.2, we grouped edges in the regular octet mesh into 6 equivalence classes, each of which corresponds to the $\mathbb{Z}^3$-lattice (Fig. 2.7(2)). Using this classification, the 1-form subdivision matrix can be written as a block matrix that has $(6 \times 6)$ blocks (Eq.(3.19)). Furthermore, as shown in Eq.(3.20), the generating function for the 1-form linear subdivision scheme can be expressed as a $(6 \times 6)$-matrix $\widehat{S_e^L}$. $\widehat{S_e^L}$ is redundant: many blocks are identical because of the symmetry of the regular octet mesh. For example, as shown in Fig. 2.7(2), for example, contributions of class-$i$ edges in the coarse mesh to class-$j$ edges in the refined mesh are identical to contributions of class-$j$ edges in the coarse mesh to class-$i$ edges in the refined mesh. Indeed, we only have to specify the first row (or first column) of blocks in $\widehat{S_e^L}$ to calculate all unknown variables in the subdivision stencil:

$$\widehat{S_e^L}_{1,1} = a_o + a_o x + c_o xyz + c_o xz + c_o z^{-1} + c_o y^{-1}z^{-1} + g_o y + g_o xy + g_o y^{-1} + g_o xy^{-1} \tag{4.8}$$

$$\widehat{S_e^L}_{1,2} = d_o x - d_o xy^{-1} + h_o xy - h_o xy^{-2} \tag{4.9}$$

$$\widehat{S_e^L}_{1,3} = b_o x + b_o y^{-1}z^{-1} + e_o z^{-1} + f_o xy + f_o y^{-2}z^{-1} + e_o xy^{-1} \tag{4.10}$$

$$\widehat{S_e^L}_{1,4} = -b_o x - b_o xy^{-1}z^{-1} - f_o xy - e_o xz^{-1} - e_o xy^{-1} - f_o xy^{-2}z^{-1} \tag{4.11}$$

$$\widehat{S_e^L}_{1,5} = -b_o x - b_o xz^{-1} - e_o xy - f_o xyz^{-1} - f_o xy^{-1} - e_o xy^{-1}z^{-1} \tag{4.12}$$

$$\widehat{S_e^L}_{1,6} = b_o x + b_o z^{-1} + e_o xy + f_o yz^{-1} + f_o xy^{-1} + e_o y^{-1}z^{-1} \tag{4.13}$$

Furthermore, from Eq. (2.11) we know that the smoothing operator can be written as

$$\begin{aligned} C &= \frac{1}{8}\widehat{S_v^L} \\ &= \frac{1}{8} + \frac{1}{16}\left(xyz + x^{-1}y^{-1}z^{-1} + xz + x^{-1}z^{-1} + x + x^{-1} + yz + y^{-1}z^{-1} + y + y^{-1} + z + z^{-1}\right) \\ &\quad + \frac{1}{48}\left(xyz^2 + x^{-1}y^{-1}z^{-2} + xy + x^{-1}y + xy^{-1} + x^{-1}y^{-1}\right). \end{aligned} \tag{4.14}$$

As discussed in Section 3.4, we can construct the generating polynomial $\widehat{S_e}$ for the smooth 1-form subdivision scheme by multiplying the generating function of the linear 1-form subdivision scheme $\widehat{S_e^L}$ with the smoothing operator $C$. We then obtain the subdivision stencil in the spatial domain by extracting the coefficients of the terms of this polynomial with the corresponding power. This leads to:

$$a_1 = \frac{11}{72} - \frac{\zeta}{3},\ a_2 = \frac{221}{4608} + \frac{5\zeta}{48},\ a_3 = \frac{49}{2304} + \frac{\zeta}{24},\ a_4 = \frac{35}{1152} - \frac{\zeta}{12} \tag{4.15}$$

$$b_1 = \frac{29}{1152} + \frac{\zeta}{12},\ b_2 = \frac{5}{384} + \frac{\zeta}{12},\ b_3 = \frac{17}{4608} + \frac{\zeta}{48},\ b_4 = \frac{35}{2304} - \frac{\zeta}{24},\ b_5 = \frac{1}{512} + \frac{\zeta}{48} \tag{4.16}$$

$$\begin{aligned} &c_1 = \frac{17}{4608} + \frac{\zeta}{48},\ c_2 = \frac{11}{1152} + \frac{\zeta}{12},\ c_3 = -\frac{1}{2304} - \frac{\zeta}{24},\ c_4 = \frac{1}{512} + \frac{\zeta}{48}, \\ &c_5 = \frac{1}{512} + \frac{\zeta}{48},\ c_6 = -\frac{1}{576} - \frac{\zeta}{6} \end{aligned} \tag{4.17}$$

$$d_1 = \frac{115}{1536} + \frac{3\zeta}{16},\ d_2 = \frac{13}{256} - \frac{\zeta}{8},\ d_3 = \frac{29}{768} + \frac{\zeta}{8},\ d_4 = \frac{25}{1536} + \frac{\zeta}{16},\ d_5 = 0 \tag{4.18}$$

$$\begin{aligned} &e_1 = \frac{43}{384} - \frac{\zeta}{4},\ e_2 = \frac{49}{1536} + \frac{\zeta}{16},\ e_3 = \frac{11}{512} + \frac{\zeta}{16},\ e_4 = \frac{1}{192},\ e_5 = \frac{17}{1536} + \frac{\zeta}{16}, \\ &e_6 = \frac{1}{192},\ e_7 = \frac{3}{512} + \frac{\zeta}{16},\ e_8 = 0,\ e_9 = 0,\ e_{10} = \frac{5}{384} - \frac{\zeta}{4},\ e_{11} = 0,\ e_{12} = 0 \end{aligned} \tag{4.19}$$

$$\begin{aligned} &f_1 = \frac{17}{768} + \frac{\zeta}{8},\ f_2 = \frac{17}{1536} + \frac{\zeta}{16},\ f_3 = \frac{1}{256} - \frac{\zeta}{8},\ f_4 = \frac{13}{768} + \frac{\zeta}{8}, \\ &f_5 = 0,\ f_6 = 0,\ f_7 = \frac{3}{512} + \frac{\zeta}{16},\ f_8 = 0,\ f_9 = 0,\ f_{10} = 0 \end{aligned} \tag{4.20}$$

$$g_1 = \frac{41}{1536} + \frac{\zeta}{16},\ g_2 = \frac{91}{1536} + \frac{3\zeta}{16},\ g_3 = \frac{53}{768} - \frac{3\zeta}{8},\ g_4 = 0,\ g_5 = 0,\ g_6 = \frac{7}{768} - \frac{\zeta}{8} \tag{4.21}$$

$$\begin{aligned} &h_1 = \frac{91}{1536} + \frac{3\zeta}{16},\ h_2 = \frac{1}{24},\ h_3 = \frac{41}{1536} + \frac{\zeta}{16},\ h_4 = \frac{59}{1536} + \frac{3\zeta}{16}, \\ &h_5 = 0,\ h_6 = 0,\ h_7 = \frac{3}{512} + \frac{\zeta}{16},\ h_8 = 0,\ h_9 = 0,\ h_{10} = 0 \end{aligned} \tag{4.22}$$

$$\begin{aligned} &i_1 = \frac{53}{768} - \frac{3\zeta}{8},\ i_2 = 0,\ i_3 = \frac{3}{512} + \frac{\zeta}{16},\ i_4 = \frac{7}{768} - \frac{\zeta}{8},\ i_5 = 0,\ i_6 = \frac{59}{1536} + \frac{3\zeta}{16}, \\ &i_7 = 0,\ i_8 = 0,\ i_9 = 0,\ i_{10} = 0,\ i_{11} = 0,\ i_{12} = 0. \end{aligned} \tag{4.23}$$

**2-Form Subdivision Generating Function** We perform the same smoothing procedure for the 2-form linear subdivision scheme. Similarly, we only have to specify the first row of blocks in the generating function $\widehat{S^L_f}$:

$$\widehat{S^L_f}_{1,1} = w_o + w_oxz + w_oxyz + t_oz + t_oyz + t_oxyz^2 + e_tx \tag{4.24}$$
$$\widehat{S^L_f}_{1,2} = u_oz + p_oxz + u_oxyz^2 \tag{4.25}$$
$$\widehat{S^L_f}_{1,3} = -f_tx - q_oxz - q_oxyz - w_oxyz^2 \tag{4.26}$$
$$\widehat{S^L_f}_{1,4} = u_oz + p_oxz + u_oxyz^2 \tag{4.27}$$
$$\widehat{S^L_f}_{1,5} = u_oxyz^3 + u_oxy^2z^3 + p_oxyz^2 \tag{4.28}$$
$$\widehat{S^L_f}_{1,6} = -f_tx^2yz^2 - w_oxyz^3 - q_ox^2yz^3 - q_oxyz^2 \tag{4.29}$$
$$\widehat{S^L_f}_{1,7} = -w_ox^2yz^2 - r_ox^2yz^3 - r_ox^2y^2z^3 - r_oxyz^2 \tag{4.30}$$
$$\widehat{S^L_f}_{1,8} = -f_tx^2yz^2 - w_oxyz^3 - q_ox^2yz^3 - q_oxyz^2. \tag{4.31}$$

The stencil of the smooth 2-form subdivision scheme is obtained using the same method as in the 1-form case:

$$j_1 = \frac{101}{1536} - \frac{7\zeta}{24},\ j_2 = \frac{1}{128} - \frac{\zeta}{12},\ j_3 = \frac{37}{1152} + \frac{\zeta}{6},\ j_4 = \frac{7}{4608},\ j_5 = \frac{53}{4608} + \frac{\zeta}{12},$$
$$j_6 = \frac{29}{4608} + \frac{\zeta}{48},\ j_7 = \frac{7}{4608},\ j_8 = -\frac{\zeta}{48},\ j_9 = \frac{1}{4608} - \frac{\zeta}{16},\ j_{10} = \frac{23}{1536} - \frac{\zeta}{24},$$
$$j_{11} = \frac{1}{128} - \frac{\zeta}{48},\ j_{12} = \frac{37}{4608} - \frac{\zeta}{8},\ j_{13} = \frac{1}{576} + \frac{\zeta}{48},\ j_{14} = \frac{19}{4608} + \frac{\zeta}{24} \tag{4.32}$$
$$k_1 = \frac{1}{128},\ k_2 = 0,\ k_3 = 0,\ k_4 = 0,\ k_5 = \frac{1}{128},\ k_6 = 0 \tag{4.33}$$
$$l_1 = \frac{55}{4608} + \frac{\zeta}{8},\ l_2 = \frac{31}{4608} + \frac{\zeta}{16},\ l_3 = \frac{1}{1536} - \frac{\zeta}{48},\ l_4 = -\frac{\zeta}{48},\ l_5 = \frac{1}{1536} - \frac{\zeta}{12},$$
$$l_6 = -\frac{1}{384} - \frac{\zeta}{6},\ l_7 = -\frac{1}{1536} - \frac{\zeta}{24},\ l_8 = \frac{1}{576} + \frac{\zeta}{48},\ l_9 = \frac{1}{144} + \frac{\zeta}{12},\ l_{10} = \frac{1}{576} + \frac{\zeta}{48},$$
$$l_{11} = \frac{19}{4608} + \frac{\zeta}{24},\ l_{12} = \frac{133}{4608} + \frac{7\zeta}{24},\ l_{13} = \frac{1}{4608},\ l_{14} = \frac{1}{4608} \tag{4.34}$$
$$m_1 = \frac{1}{64} - \frac{\zeta}{8},\ m_2 = \frac{5}{512} + \frac{\zeta}{16.0},\ m_3 = \frac{1}{192},\ m_4 = 0,\ m_5 = \frac{3}{256} - \frac{\zeta}{4},\ m_6 = 0 \tag{4.35}$$
$$n_1 = \frac{1}{16},\ n_2 = 0,\ n_3 = 0,\ n_4 = 0 \tag{4.36}$$
$$o_1 = \frac{23}{768} + \frac{\zeta}{4},\ o_2 = \frac{1}{1536} - \frac{\zeta}{16},\ o_3 = 0,\ o_4 = 0,\ o_5 = \frac{1}{96} + \frac{\zeta}{8},\ o_6 = 0 \tag{4.37}$$
$$p_1 = \frac{5}{256} + \frac{\zeta}{8},\ p_2 = \frac{7}{1536},\ p_3 = \frac{1}{128} - \frac{\zeta}{16},\ p_4 = \frac{11}{384},\ p_5 = \frac{13}{1536} - \frac{3\zeta}{16},\ p_6 = -\frac{\zeta}{16},$$
$$p_7 = \frac{1}{192},\ p_8 = 0,\ p_9 = \frac{19}{1536} + \frac{\zeta}{8},\ p_{10} = 0,\ p_{11} = 0,\ p_{12} = \frac{25}{384} + \frac{\zeta}{2},\ p_{13} = 0,\ p_{14} = 0 \tag{4.38}$$
$$q_1 = \frac{1}{64},\ q_2 = \frac{1}{128},\ q_3 = 0,\ q_4 = 0,\ q_5 = 0,\ q_6 = 0 \tag{4.39}$$

$$r_1 = \frac{11}{384} - \frac{\zeta}{2},\ r_2 = \frac{5}{1536} - \frac{\zeta}{8},\ r_3 = \frac{1}{192} + \frac{\zeta}{16},\ r_4 = 0,\ r_5 = \frac{31}{1536} + \frac{3\zeta}{16},\ r_6 = \frac{1}{192} + \frac{\zeta}{16},$$
$$r_7 = 0,\ r_8 = 0,\ r_9 = \frac{1}{1536},\ r_{10} = 0,\ r_{11} = 0,\ r_{12} = \frac{1}{768} - \frac{\zeta}{8},\ r_{13} = 0,\ r_{14} = 0 \tag{4.40}$$
$$s_1 = \frac{35}{512} + \frac{3\zeta}{8},\ s_2 = \frac{49}{1536} - \frac{\zeta}{8},\ s_3 = \frac{7}{1536},\ s_4 = \frac{5}{1536} - \frac{\zeta}{8},\ s_5 = \frac{1}{384},$$
$$s_6 = \frac{23}{1536} + \frac{\zeta}{8},\ s_7 = 0,\ s_8 = \frac{1}{384} - \frac{\zeta}{16},\ s_9 = \frac{1}{192} + \frac{\zeta}{16},\ s_{10} = -\frac{3}{512} - \frac{3\zeta}{8},$$
$$s_{11} = \frac{1}{1536},\ s_{12} = \frac{19}{1536} + \frac{\zeta}{8},\ s_{13} = \frac{1}{384}. \tag{4.41}$$

*Remark:* We can substitute these stencil variables into the subdivision matrices and verify that the commutative relations are still valid. Although the 1- and 2-form subdivision matrices are $\zeta$-dependent, they satisfy all commutative relations in Eq.(3.27). We use this calculation to check the correctness of our calculations on stencil variables.

### 4.2.3 Results and Discussions

We choose $\zeta = 5$ (the reason will be revealed in Section 4.3) to generate some results. In order to avoid irregularities due to boundaries in this section, we choose a larger initial mesh $\mathcal{M}_I$ (Fig. 4.13) where only a single edge and a single facet are assigned nonzero form coefficients.

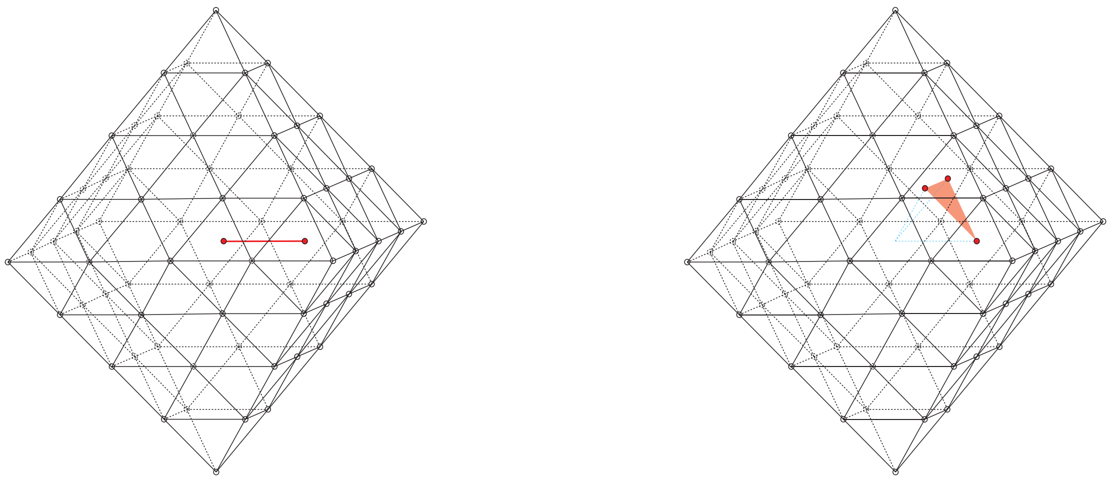
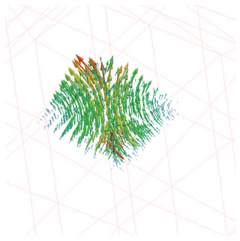
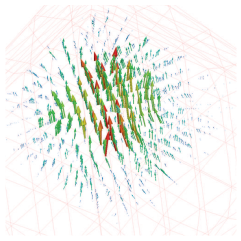

(1) Initial mesh $\mathcal{M}_I$: 1-form (2) Initial mesh $\mathcal{M}_I$: 2-form

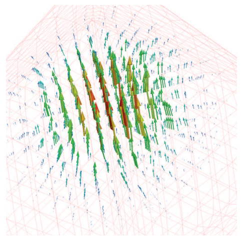
Figure 4.13: $\mathcal{M}_I$: an initial mesh with nonzero form coefficients on only one edge and one facet

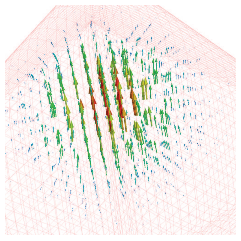
For most vector field visualization tasks in this thesis, two types of illustrations are included: vector-type visualizations and streamline-type visualizations. Advantages and disadvantages of both methods will be discussed in Section 4.3.2. Fig. 4.14 shows the process of 1- and 2-form subdivisions up to three times, starting from $\mathcal{M}_I$. These results illustrate that vector fields associated with both 1- and 2-forms in $\mathcal{M}_I$ become smoother as we progressively subdivide the mesh.

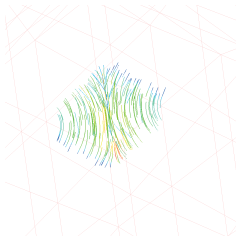
Even if the form coefficient field and the mesh are both symmetric, the generated vector field may not be because two sources of asymmetries are introduced during the visualization. First, when we calculate local vectors using Whitney forms, the seed-generation is not symmetric for technical reasons discussed in Appendix A.3.1. Second, when streamlines are generated, the sampling is random and only statistically uniform.

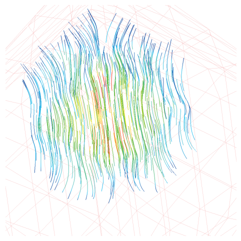

(1) Step 0, 1-form, vector-type

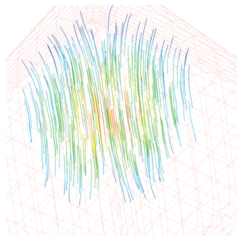

(2) Step 1, 1-form, vector-type

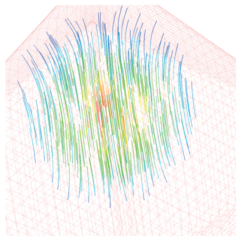

(3) Step 2, 1-form, vector-type

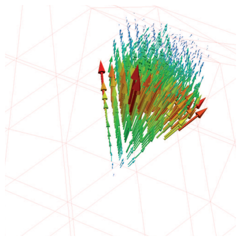

(4) Step 3, 1-form, vector-type

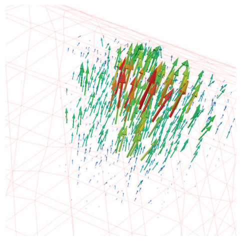

(5) Step 0, 1-form, streamline-type

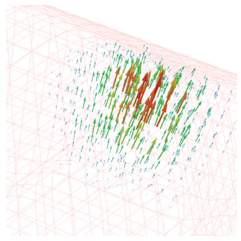

(6) Step 1, 1-form, streamline-type

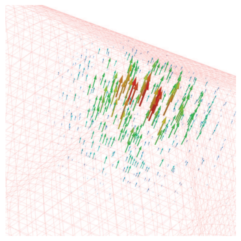

(7) Step 2, 1-form, streamline-type

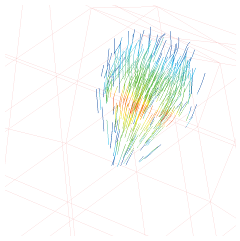

(8) Step 3, 1-form, streamline-type

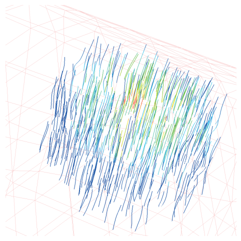

(9) Step 0, 2-form, vector-type

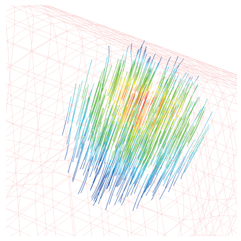

(10) Step 1, 2-form, vector-type


(11) Step 2, 2-form, vector-type


(12) Step 3, 2-form, vector-type


(13) Step 0, 2-form, streamline-type


(14) Step 1, 2-form, streamline-type


(15) Step 2, 2-form, streamline-type

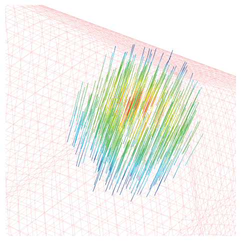
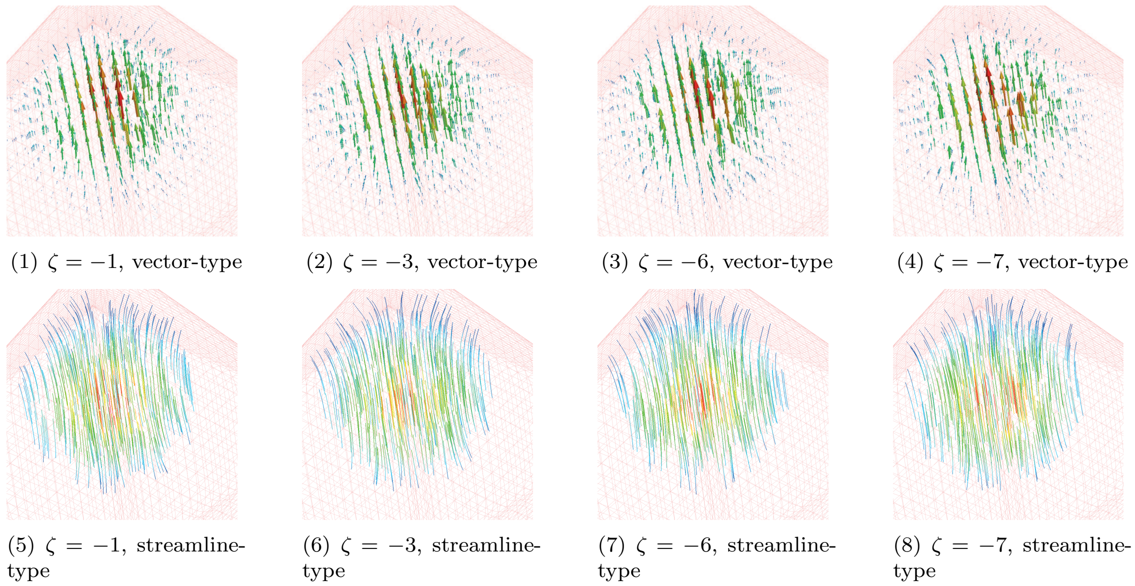

(16) Step 3, 2-form, streamline-type

Figure 4.14: Subdivision results for $\mathcal{M}_I$, $\zeta = -5$. Boundary issues are ignored here.

We try to determine the optimal $\zeta$ by substituting different values and visually inspecting the smoothness of the subdivision results. Unfortunately, as shown in Fig. 4.15, for $\zeta = -1, -3, -6$ and

$-7$, there is no significant difference between the results and the $\zeta = 5$ case (Fig. 4.14(4)). However, if $\zeta$ deviates more, as shown in Fig. 4.16 where $\zeta = 11$ and $\zeta = -17$, the subdivision results are no longer smooth. If $\zeta$ deviates further, e.g., if $\zeta = 17$ or $\zeta = -26$ as shown in Fig. 4.17, the vector field blows up during the subdivision.


(1) $\zeta = -1$, vector-type (2) $\zeta = -3$, vector-type (3) $\zeta = -6$, vector-type (4) $\zeta = -7$, vector-type

(5) $\zeta = -1$, streamline-type (6) $\zeta = -3$, streamline-type (7) $\zeta = -6$, streamline-type (8) $\zeta = -7$, streamline-type

Figure 4.15: 1-form subdivision results for $\mathcal{M}_I$ when $\zeta \neq -5$. It is difficult to distinguish the difference in the quality of the results compared to the $\zeta = 5$ case.

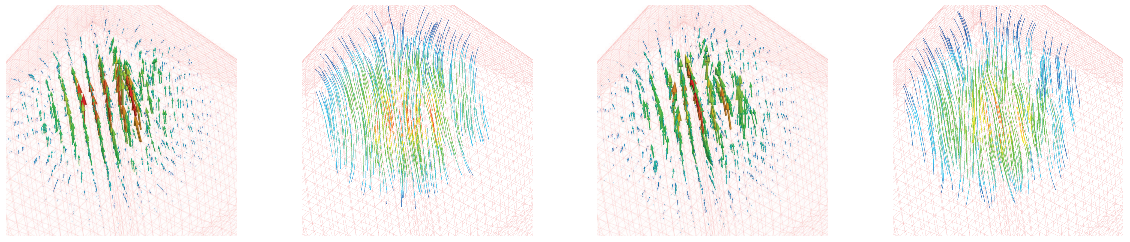

(1) $\zeta = 11$, vector-type (2) $\zeta = 11$, streamline-type (3) $\zeta = -17$, vector-type (4) $\zeta = -17$, streamline-type

Figure 4.16: When $\zeta$ deviates from the range $[-5, -7)$ more, the 1-form subdivision results start to lose smoothness.

Our conclusions are similar for 2-forms. No significant difference among the results for $\zeta = -7$ (Fig. 4.20(4)), $\zeta = -1$ (Fig. 4.18(1)) and $\zeta = -5$ (Fig. 4.14(12)) is detected. However, in our experiments, we found that 2-form results are more sensitive to the deviation of $\zeta$ than the 1-form: as shown in Fig. 4.18(3), when $\zeta = -17$, the 2-form vector field blows up more severely than the 1-form case (Fig. 4.17(1)) above. The reason will be explored in the next section.

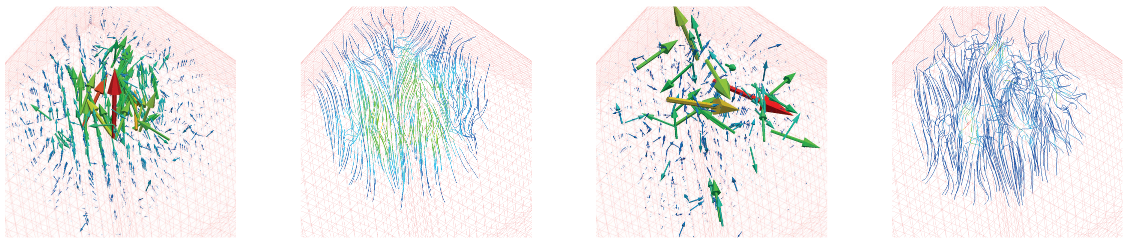
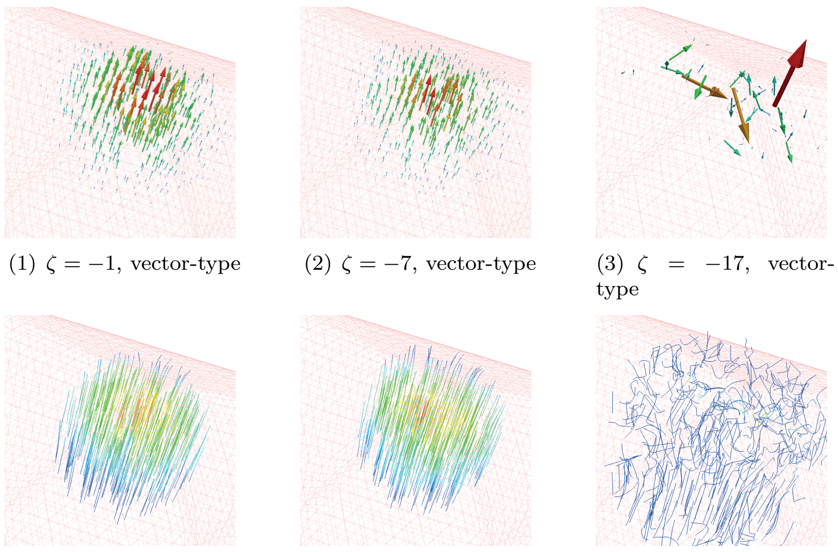

(1) $\zeta = 17$, vector-type (2) $\zeta = 17$, streamline-type (3) $\zeta = -26$, vector-type (4) $\zeta = -26$, streamline-type

Figure 4.17: If $\zeta$ deviates significantly from the range $[-5, -7)$, the 1-form subdivision results blow up.


(1) $\zeta = -1$, vector-type (2) $\zeta = -7$, vector-type (3) $\zeta = -17$, vector-type

(4) $\zeta = -1$, streamline-type (5) $\zeta = -7$, streamline-type (6) $\zeta = -17$, streamline-type

Figure 4.18: 2-form subdivision results for different values of $\zeta$

## 4.3 Using Eigenforms to Determine the Free Parameter

### 4.3.1 Eigenanalysis of the Subdivision Stencil

We can perform eigendecomposition on the 1- or 2-form subdivision matrix. Each eigenvector corresponds to an 1- or 2-form coefficient field that is invariant under the subdivision up to a scaling factor, i.e., the associated eigenvalue. Geometrically, "invariant" means that, to perform the subdivision, form coefficients from the coarse mesh, after being modulated by the associated eigenvalues, can be directly assigned to corresponding simplices in the refined mesh. These invariant forms are called *eigenforms*.

We calculate the eigenvalues of subdivision matrices. For 1-forms, we have the eigenvalues

$$\begin{gathered}\underbrace{\frac{1}{2},\frac{1}{2},\frac{1}{2}}_{3},\underbrace{\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4}}_{9},\underbrace{\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8}}_{14},\underbrace{\frac{5}{48}}_{1},\underbrace{\frac{1}{12},\frac{1}{12}}_{2},\\
\underbrace{\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16}}_{7},\underbrace{\frac{1}{32},\frac{1}{32},\frac{1}{32},\frac{1}{32}}_{4},\underbrace{\frac{1}{48},\frac{1}{48}}_{2},\underbrace{-\frac{\zeta+1}{48},-\frac{\zeta+1}{48},-\frac{\zeta+1}{48}}_{3},\\
\underbrace{-\frac{\zeta+1}{96},-\frac{\zeta+1}{96},-\frac{\zeta+1}{96},-\frac{\zeta+1}{96},-\frac{\zeta+1}{96},-\frac{\zeta+1}{96},-\frac{\zeta+1}{96},-\frac{\zeta+1}{96},-\frac{\zeta+1}{96}}_{9},\\
\underbrace{-\frac{\zeta+1}{128},-\frac{\zeta+1}{128},-\frac{\zeta+1}{128}}_{3},\underbrace{-\frac{7(\zeta+1)}{1152},-\frac{7(\zeta+1)}{1152},-\frac{7(\zeta+1)}{1152}}_{3}.\end{gathered} \tag{4.42}$$

Here, numbers under brackets indicate multiplicities of corresponding eigenvalues. Similarly, for 2-forms, we have the eigenvalues

$$\begin{gathered}\underbrace{\frac{1}{4},\frac{1}{4},\frac{1}{4}}_{3},\underbrace{\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8}}_{9},\underbrace{\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16},\frac{1}{16}}_{11},\\
\underbrace{\frac{1}{32},\frac{1}{32},\frac{1}{32},\frac{1}{32},\frac{1}{32},\frac{1}{32},\frac{1}{32},\frac{1}{32},\frac{1}{32},\frac{1}{32}}_{10},\underbrace{\frac{1}{48},\frac{1}{48}}_{2},\underbrace{\frac{1}{64},\frac{1}{64},\frac{1}{64}}_{3},\underbrace{-\frac{1+\zeta}{48},-\frac{1+\zeta}{48},-\frac{1+\zeta}{48}}_{3},\\
\underbrace{-\frac{1+\zeta}{96},-\frac{1+\zeta}{96},-\frac{1+\zeta}{96},-\frac{1+\zeta}{96},-\frac{1+\zeta}{96},-\frac{1+\zeta}{96},-\frac{1+\zeta}{96},-\frac{1+\zeta}{96},-\frac{1+\zeta}{96}}_{9},\\
\underbrace{-\frac{1+\zeta}{128},-\frac{1+\zeta}{128},-\frac{1+\zeta}{128}}_{3},\underbrace{-\frac{7(1+\zeta)}{1152},-\frac{7(1+\zeta)}{1152},-\frac{7(1+\zeta)}{1152}}_{3}.\end{gathered} \tag{4.43}$$

An arbitrary form coefficient field can be decomposed as a linear combination of eigenforms. Components associated with large eigenvalues (*principal forms*) shrink slower than the ones associated with small eigenvalues. In other words, after a few steps of subdivision, the principal forms will dominate the results. As shown in Eq.(4.42) and Eq.(4.43), if $\zeta$ stays in a small range around

-5, leading eigenvalues and associated eigenforms are independent of $\zeta$. This is why the subdivision results look similar when we vary $\zeta$ between -1 and -7 (Fig. 4.15).

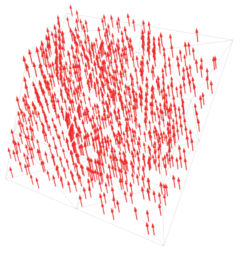
On the other hand, if $\zeta$ deviates more from -5, $\zeta$-dependent eigenvalues become the leading eigenvalues and the corresponding eigenforms become important during the subdivision. When $\zeta = -26$ or 17, the eigenvalue $-\frac{\zeta+1}{48}$ becomes the largest and exceeds $\frac{1}{2}$, so the subdivision scheme is divergent. When $\zeta = -17$ or 11, in the 1-form case, the eigenvalue $-\frac{\zeta+1}{48}$ becomes the second largest eigenvalue so the subdivision results are still rough. However, $-\frac{\zeta+1}{48}$ is still the largest eigenvalue for the 2-form subdivision matrix, hence the 2-form scheme is divergent and the vector field blows up during the subdivision.

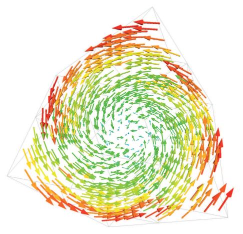
### 4.3.2 Visualization of Eigenforms

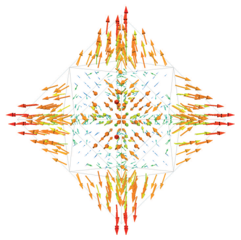
We first visualize eigenforms associated with $\zeta$-free eigenvalues using both the vector-type and the streamline-type visualization. The color in both types encodes the magnitude of the local vector in the field. Vector-type visualization works well when the field is simple (e.g., Fig. 4.19(1) and Fig. 4.19(2)) but becomes difficult to interpret when the field has many delicate local structures (e.g., Fig. 4.19(3)): we are unable to see the structure of the region with small vectors. In these situations, streamline-type visualization becomes more informative (e.g., Fig. 4.19(6)): vector field directions in the small-vector region becomes clearer. However, streamline-type visualizations have two drawbacks. First, we can only interpret the magnitude of local vectors using colors. For example, in Fig. 4.19(9), from the vector-type visualization, one can easily tell that the vector field in the corner region of the octahedral cell is dominant. But this feature is difficult to interpret using the streamline-type visualization in Fig. 4.19(12). Second, unlike the vector-type visualization which is calculated locally, streamlines are calculated using numerical integrations which introduce another source of numerical error. This error may have large effect for regions near the boundary or in the cases where delicate structures exist.

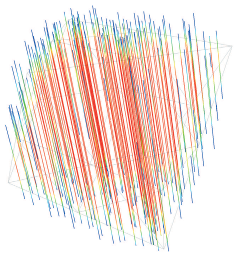
The key information conveyed by Fig. 4.19 is that, as the eigenvalue becomes smaller, the associated eigenform corresponds to a rougher vector field. Eigenforms associated with the first few largest eigenvalues are very smooth. In the extreme, the leading principal eigenform yields a constant vector field in both 1- and 2-form cases. These smooth eigenforms are likely to be dominant in results when we subdivide an 1- or 2-form for only a few steps. This is why the subdivision results for $\zeta = -1, -3, -5, -6$ and $-7$ are smooth.

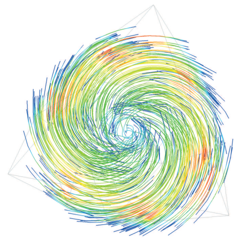
However, as we can see from Fig. 4.20, $\zeta$-dependent eigenforms are less smooth than eigenforms associated with leading $\zeta$-independent eigenvalues. Furthermore, these $\zeta$-dependent eigenforms change very little when we tune $\zeta$. If we cannot make these $\zeta$-dependent eigenforms smooth, we should make the associated $\zeta$-dependent eigenvalues as small as possible to obtain a fast shrinkage on these rough eigenforms and reduce their overall effect in the final results. Previous researchers

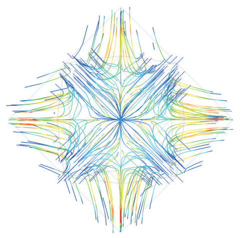

(1) Eigen-1-form associated with the largest eigenvalue $\frac{1}{2}$ (vector-type)

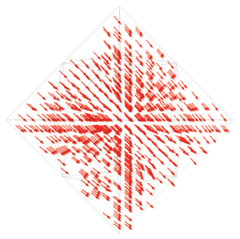

(2) Eigen-1-form associated with the second largest eigenvalue $\frac{1}{4}$ (vector-type)

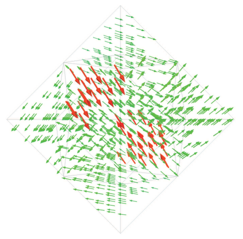

(3) Eigen-1-form associated with the fourth largest eigenvalue $\frac{5}{48}$ (vector-type)

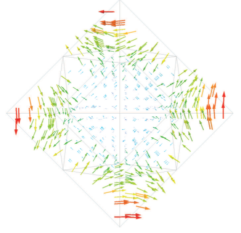

(4) Eigen-1-form associated with the largest eigenvalue $\frac{1}{2}$ (streamline-type)

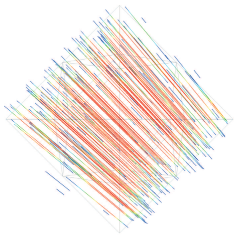

(5) Eigen-1-form associated with the second largest eigenvalue $\frac{1}{4}$ (streamline-type)

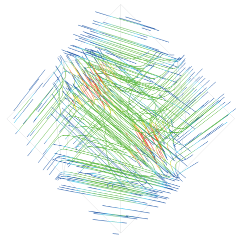

(6) Eigen-1-form associated with the fourth largest eigenvalue $\frac{5}{48}$ (streamline-type)


(7) Eigen-2-form associated with the largest eigenvalue $\frac{1}{4}$ (vector-type)


(8) Eigen-2-form associated with the second largest eigenvalue $\frac{1}{8}$ (vector-type)


(9) Eigen-2-form associated with the fourth largest eigenvalue $\frac{1}{32}$ (vector-type)


(10) Eigen-2-form associated with the largest eigenvalue $\frac{1}{4}$ (streamline-type)


(11) Eigen-2-form associated with the second largest eigenvalue $\frac{1}{8}$ (streamline-type)

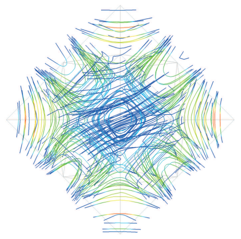
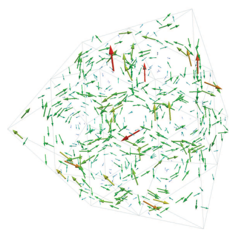
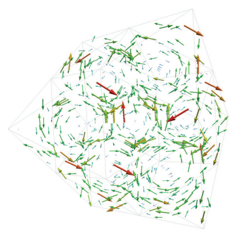
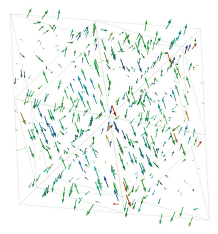
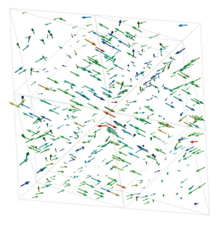

(12) Eigen-2-form associated with the fourth largest eigenvalue $\frac{1}{32}$ (streamline-type)

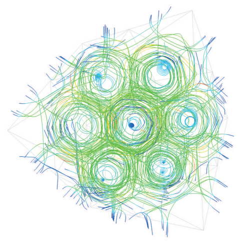
Figure 4.19: Eigenforms associated with some of the $\zeta$-independent eigenvalues

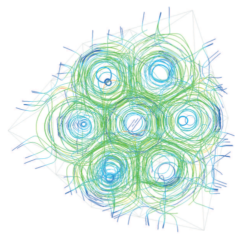

(1) Eigen-1-form associated with the largest $\zeta$-dependent eigenvalue for $\zeta = -5$ (vector-type)

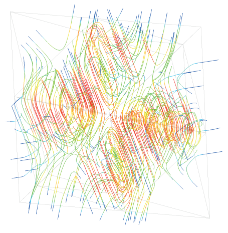

(2) Eigen-1-form associated with the largest $\zeta$-dependent eigenvalue for $\zeta = -7$ (vector-type)


(3) Eigen-2-form associated with the largest $\zeta$-dependent eigenvalue for $\zeta = -5$ (vector-type)


(4) Eigen-2-form associated with the largest $\zeta$-dependent eigenvalue for $\zeta = -7$ (vector-type)


(5) Eigen-1-form associated with the largest $\zeta$-dependent eigenvalue for $\zeta = -5$ (streamline-type)


(6) Eigen-1-form associated with the largest $\zeta$-dependent eigenvalue for $\zeta = -7$ (streamline-type)


(7) Eigen-2-form associated with the largest $\zeta$-dependent eigenvalue for $\zeta = -5$ (streamline-type)

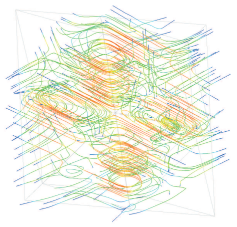

(8) Eigen-2-form associated with the largest $\zeta$-dependent eigenvalue for $\zeta = -7$ (streamline-type)

Figure 4.20: Eigenforms associated with the largest $\zeta$-dependent eigenvalue for different values of $\zeta$

have found that $\zeta$ should be within the range of $(-7, -5]$ due to stability concerns [42]. Discussions of these techniques are beyond the scope of our thesis. Here we set $\zeta = -5$ to minimize the magnitude of all the $\zeta$-dependent eigenvalues.

# Chapter 5

# Subdivision in the Boundary Case

Initial mesh $\mathcal{M}_I$ only has nonzero discrete differential form coefficients on interior geometric elements so the ignoring of boundary issues does not affect subdivision results. Because of the complexity of the boundary problems, Section 5.1 introduces similar issues in 2D surface subdivision problems. In 3D cases where boundary edges or facets carry nonzero 1- or 2-form coefficients, Section 5.2 discusses consequences if boundary issues continue to be ignored. Undesirable attenuation phenomenon is observed in the boundary region. To address this issue, two methods are discussed and compared. Section 5.3 introduces the projection-scaling method which first projects the real differential form coefficient field into a subspace spanned by low-order differential form coefficient fields and then perform the subdivision. However, because of the shrinking of the octet mesh using the original 0-form subdivision scheme, the vector field near the boundary is enlarged. Scaling is introduced for elements near the boundary to offset the effect of the mesh shrinkage, but the vector field near the boundary still exhibits some undesirable curvatures. Alternatively we can start from scratch to construct a new nonshrinking 0-form subdivision scheme (Section 5.4) and build up compatible 1- and 2-form subdivision schemes based on it (Section 5.5).

## 5.1 Introduction: Boundary Problem in 2D Cases

Fig. 5.1(1) shows a surface with boundary. The black vertex in the center is regular. Similar to the 3D volumetric domain cases we discussed in Section 2.1, we use red and blue to denote two types of boundary vertices. If we apply Loop's subdivision scheme on the mesh, the algorithm will try to identify vertices in the stencil (Fig. 1.1) for each vertex in the refined mesh and perform geometric smoothing to update the position.

However, for boundary vertices, we cannot extract a complete local stencil. A naive strategy is to simply ignore missing vertices (i.e., drop the missing terms when we use the linear combination of nearby vertices to update the position of the target vertex). Another remedy is to expand the mesh. As shown in Fig. 5.1(2) and Fig. 5.1(3), after this 2-step expansion treatment, vertices in

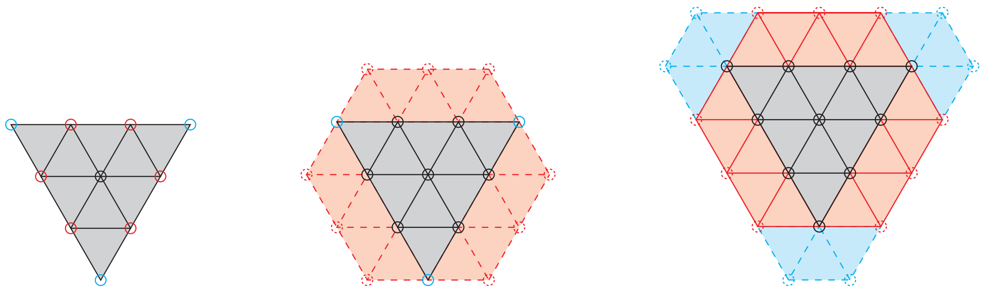

(1) Original mesh (2) Expansion: step 1 (3) Expansion: step 2

Figure 5.1: Example: expand a mesh with boundary

the original mesh all become interior vertices while all boundary vertices are in the virtual mesh grown around the real mesh. We can simply apply the original Loop's scheme on the extended mesh and trim the virtual mesh after the subdivision. Furthermore, similar issues exist for 1- and 2-form subdivision tasks. For boundary edges or facets, stencils shown in Fig. 1.4 are incomplete. After we extend the original mesh, we can also extend the differential form coefficient field onto the virtual mesh.

The rest of this chapter discusses the details of this idea in 3D volumetric domain subdivision problems.

## 5.2 Simple Method

Subdivision schemes in this chapter are tested on the initial mesh $\mathcal{M}_B$ (Fig. 5.5) where nonzero 1- and 2-form coefficients are assigned to three edges and two edges, respectively. A boundary edge or facet doesn't need to have all vertices lying on the boundary. Indeed, if we cannot extract a regular stencil (i.e., an $\mathcal{M}_E$-like structure) around the edge/facet, then it is a boundary edge/facet. Note that, for the subdivision scheme discussed in Chapter 4, this definition of boundary elements is equivalent to the definition in Eq(2.9), i.e., an edge or facet is defined to be on the boundary if it has at least one vertex lying on the boundary.

We first study consequences if we continue to ignore differential form coefficients on the boundary. Specifically, for an edge or facet in the refined mesh is on the boundary, we simply assign 0 to it. Fig. 5.3 shows results using this method.

From results we can see that, the interior part of the vector field is smooth. However, the vector field starts to attenuate as it gets closer to the boundary and completely vanishes on the boundary (Fig. 5.4). Therefore, the simple method is incorrect when initial boundary edges or facets are associated with nonzero differential form coefficients: 1- and 2-form coefficients on boundary elements cannot be replicated by integrating the vector field over these elements. We discuss several

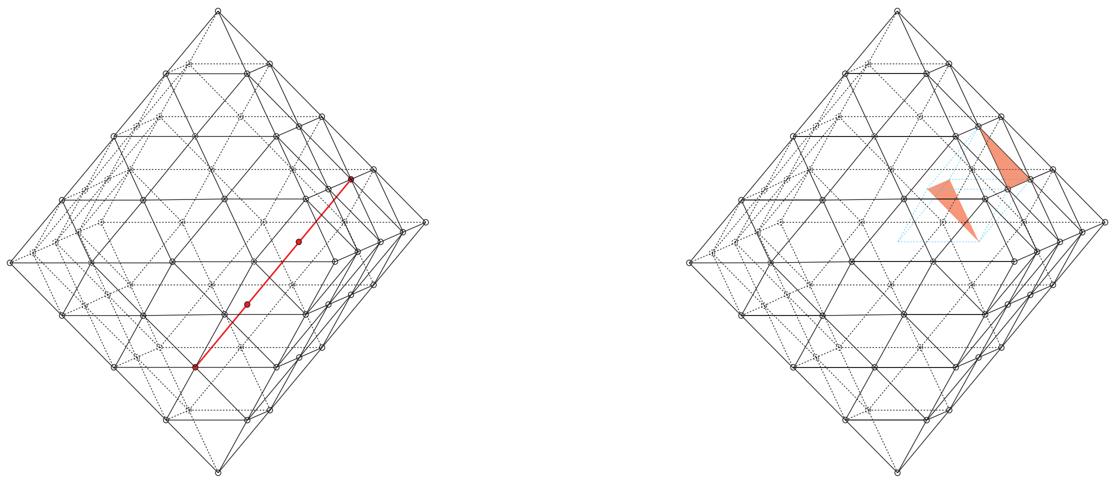

(1) Initial mesh $\mathcal{M}_B$: 1-form (2) Initial mesh $\mathcal{M}_B$: 2-form

Figure 5.2: $\mathcal{M}_B$: initial mesh with nonzero differential form coefficients in both the interior and the boundary. For simplicity, interior edges or facets with zero differential form coefficients are not shown here.

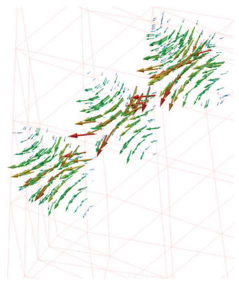
alternative methods to fix this problem in following sections.

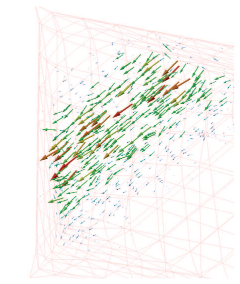
## 5.3 Projection Method

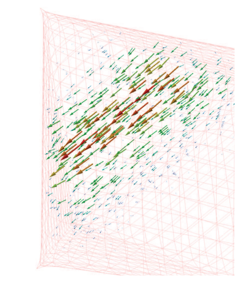
We take the 1-form field as an example in this section. Given a vector field without source or sink in the volumetric domain, the vector field corresponds to a *potential field*, i.e., a scalar coefficient field associated with vertices. Reversely, for every edge, the associated 1-form coefficient can be obtained by calculating the difference between potential values at the end and the beginning of the oriented edge, or in other words, the path integral of the vector field along this edge. Furthermore, as shown in Fig. 5.5(1), we can construct a local coordinate system in the boundary region. A *low-order potential field* is a potential field whose values on vertices follows low-order polynomials of coordinates of these vertices, e.g., $1, x, y, z, x^2, y^2, z^2, xy, xz, yz, \cdots$. A *low-order 1-form* is induced by the low-order potential field and is denoted as $\mathrm{d}1(\text{trivial}), \mathrm{d}x, \mathrm{d}y, \mathrm{d}z, \mathrm{d}x^2, \mathrm{d}y^2, \mathrm{d}z^2, \mathrm{d}xy, \mathrm{d}xz, \mathrm{d}yz, \cdots$.

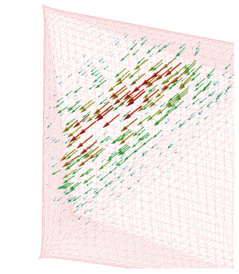
If a potential field follows a low-order polynomial whose form is known a priori, it can be naturally extend beyond the boundary by simply following the same polynomial. The motivation underlying the projection method is to preserve low-order components of the potential field and the induced vector field. Specifically, for an arbitrary 1-form field, we first project the 1-form coefficients vector into the subspace spanned by low-order 1-form coefficients vectors. Three rules are followed:

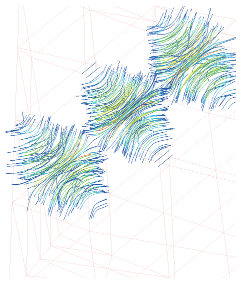
▶ During the construction of the subspace, we only use potential values on vertices belong to the boundary layer of cells, i.e., the size of the "extrapolation stencil" is similar to the size of the original 0-form subdivision stencil. In other words, we only use potential values restricted in the boundary

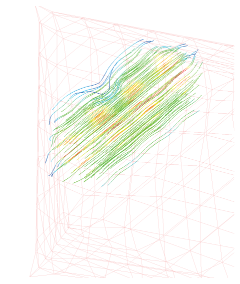

(1) Step 0, 1-form, vector-type

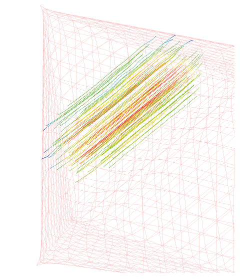

(2) Step 1, 1-form, vector-type

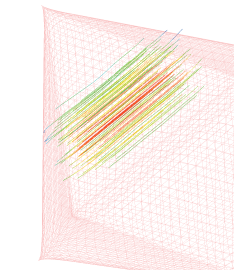

(3) Step 2, 1-form, vector-type

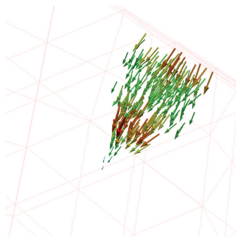

(4) Step 3, 1-form, vector-type

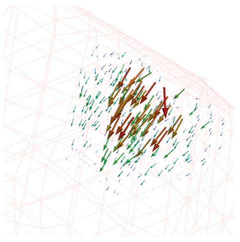

(5) Step 0, 1-form, streamline-type

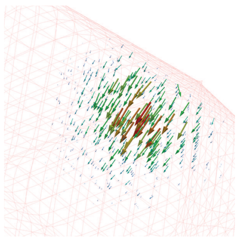

(6) Step 1, 1-form, streamline-type

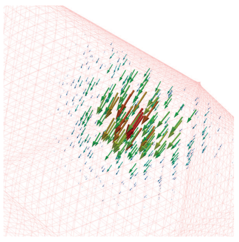

(7) Step 2, 1-form, streamline-type

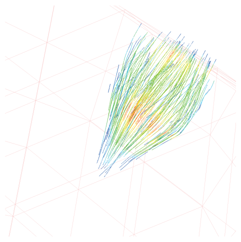

(8) Step 3, 1-form, streamline-type

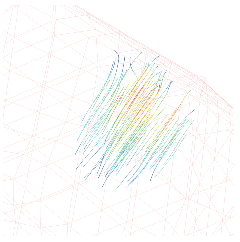

(9) Step 0, 2-form, vector-type

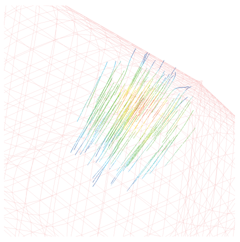

(10) Step 1, 2-form, vector-type


(11) Step 2, 2-form, vector-type


(12) Step 3, 2-form, vector-type


(13) Step 0, 2-form, streamline-type


(14) Step 1, 2-form, streamline-type


(15) Step 2, 2-form, streamline-type

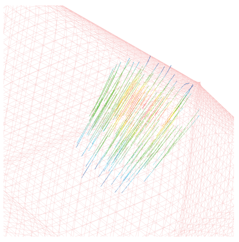
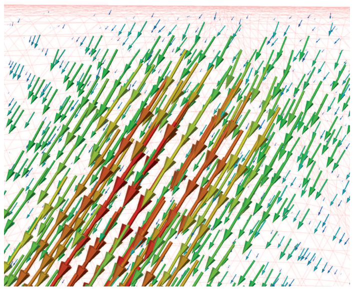

(16) Step 3, 2-form, streamline-type

Figure 5.3: Subdivision results for $\mathcal{M}_B$ using the simple method. We simply assign 0 to all boundary geometric elements in the refined level.


(1) Details near the boundary after three steps of subdivision using the simple method (1-form)

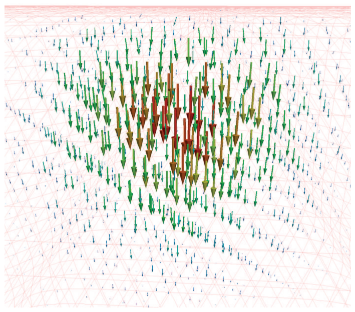
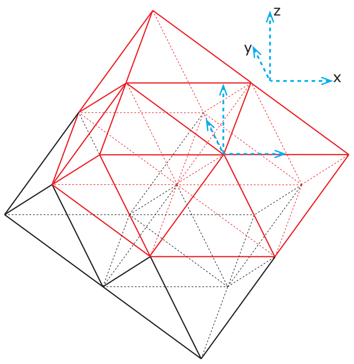

(2) Details near the boundary after three steps of subdivision using the simple method (2-form)

Figure 5.4: Details near the boundary after subdivide $\mathcal{M}_B$ for three times using the simple method. Vector fields attenuate near the boundary and vanish completely on the boundary.



(1) Boundary layer before subdivision

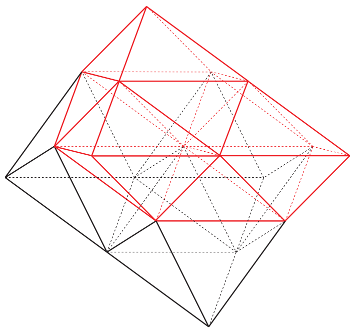

(2) Boundary layer after subdivision

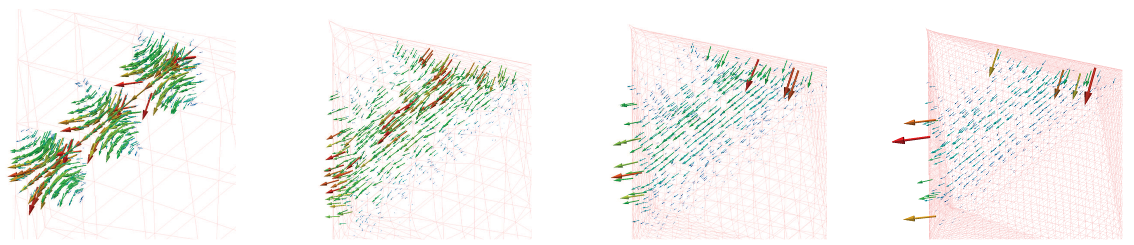
Figure 5.5: Boundary layer before and after the original 0-form subdivision. Interior edges are denoted by black while boundary edges are denoted by red. The boundary layer will shrink in the normal direction after the subdivision (for simplicity, only even edges are shown in the refined mesh). We can also introduce local coordinate system to construct local low-order potential fields for the projection method.

layer to infer what we should assign beyond the boundary.

▶ Under the "restricted-stencil" constraint mentioned above, some low-order potential fields are perfectly correlated. Geometrically, this means, for example, the potential fields of $z$ and $z^2$ are completely equivalent on the boundary layer cells so they are undistinguishable (see Fig. 5.5(1)). Thus, when we construct the basis of the subspace, we exclude any low-order field which is not linearly independent with previously included lower-order fields, i.e., the differential form coefficients vectors in the basis are mutually linearly independent and the subspace is nondegenerate.

▶ We include as many low-order fields as above constraints allow up to a certain order to minimize the error induced by this linear-projection approximation.

Based on these three rules, if we set the cut-off order to be 3 (i.e., the potential field subspace is cubic while the induced 1-form field subspace is quadratic), the subspace basis contains 12 low-order fields: $x, y, z, x^2, y^2, xy, xz, yz, x^3, y^3, x^2y, xy^2$. Subdivision results are shown in Fig. 5.6.



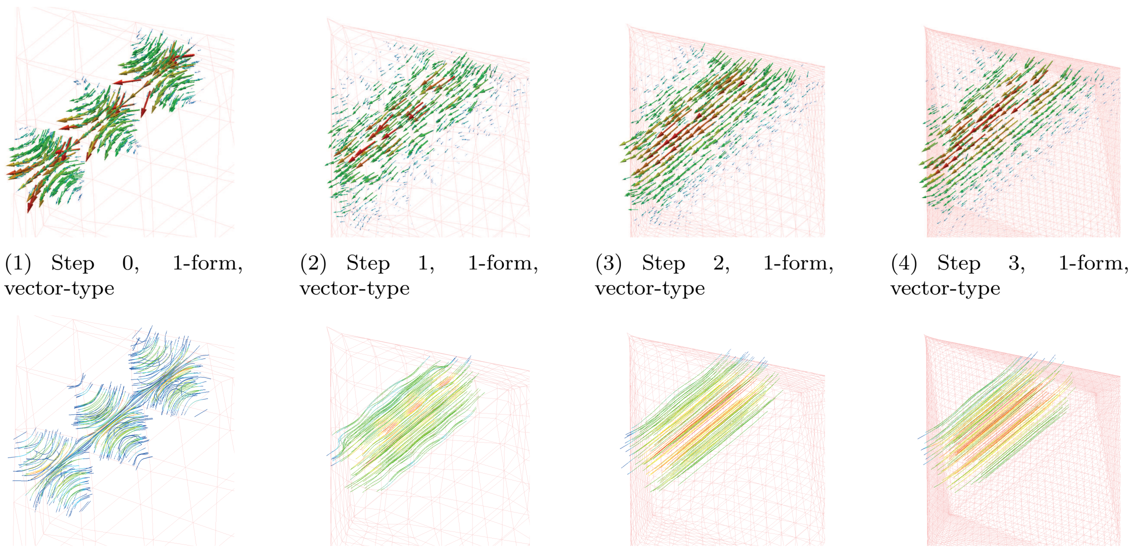
(1) Step 0, 1-form, vector-type (2) Step 1, 1-form, vector-type (3) Step 2, 1-form, vector-type (4) Step 3, 1-form, vector-type

Figure 5.6: 1-form subdivision results for $\mathcal{M}_I$ using the projection method

**Boundary Layer Shrinkage Problem** Based on the boundary rule defined in the original 0-form subdivision scheme, the size of the volumetric domain keeps shrinking during the subdivision. From the stencil in Fig. 2.13 we know that, if an even vertex lies on the boundary, through averaging with positions of adjacent vertices, the boundary vertex will move towards the side which, for a convex coarse mesh, is the interior of the boundary cell. Therefore, the boundary cells shrink in the normal direction after the subdivision (Fig. 5.5(2)). Because 1- and 2-form subdivision schemes under the projection method have not included such shrinkages into the construction, corresponding vector fields in boundary cells are enlarged (i.e., with the same differential form coefficients but shrunk edges, reconstructed vectors are larger in order to obtain same path integral results). Shrinkage features and induced enlarging artifacts near initial edges and corner vertices are more complicated.

There are two ways to solve the incompatibility between the 0-form and the 1-/2-form subdivisions: (1) multiply form coefficients associated with boundary elements with a scaling factor that equals to the shrinkage rate of those elements; (2) construct a nonshrinking 0-form subdivision scheme and compatible 1-/2-form subdivision methods. We will discuss the first method here and the second method in the next two sections.

**Scaling** For $\mathcal{M}_E$, we can calculate that the boundary layer shrinks 50% in the normal direction after one step of subdivision. Thus, a simple remedy is to shrink all 1-form coefficients associated with shrunk edges by the same ratio. A more uniform vector field is achieved using this method (Fig. 5.7). From results, we notice that the enlarging phenomenon of vector fields in boundary cells are removed. However, the projection-scaling method has two drawbacks. First, in the example above we can see that streamlines in boundary cells exhibit some curved features because subdivision results are affected by initial boundary edges and corner vertices. Second, shrinking the form coefficient for a constant ratio is ad hoc and not robust enough for general meshes with more exotic boundary shapes.



(1) Step 0, 1-form, vector-type

(2) Step 1, 1-form, vector-type

(3) Step 2, 1-form, vector-type

(4) Step 3, 1-form, vector-type

(5) Step 0, 1-form, streamline-type

(6) Step 1, 1-form, streamline-type

(7) Step 2, 1-form, streamline-type

(8) Step 3, 1-form, streamline-type

Figure 5.7: 1-form subdivision results for $\mathcal{M}_B$ using the projection method with scaling in the normal direction to the boundary facets

## 5.4 Nonshrinking 0-Form Subdivision

Conceptually, a nonshrinking subdivision scheme can be constructed through an "expansion-trim" process which is summarized in Algorithm 1.

*Remark:* The shrinkage phenomenon only exists in the boundary layer of cells. After one step of subdivision, the extended virtual layer splits into two virtual layers. Therefore, we trim one virtual layer and keep the other one as the boundary layer to protect the real mesh. We need to trim both virtual layers after the last time of subdivision.

During the expansion, we first extend initial facet by adding in additional virtual B1 boundary vertices, then we extend initial edges by adding in additional virtual B2 border vertices and finally we

**Algorithm 1** Nonshrinking 0-form subdivision

```
Require: The mesh is well defined, complete and free of conflict
1: (Preprocessing) Expand the mesh by growing an additional layer above
   the boundary.
2: for $i$ from 1 to the maximum step of subdivisions do
3:     Subdivide the expanded mesh using the original 0-form scheme
4:     Trim the boundary layer
5: end for
6: (Postprocessing) Trim the boundary layer
```

extend initial B3 corner vertices by adding in virtual corner vertices. As an example, the expansion process on $\mathcal{M}_E$ mesh is shown in Fig. 5.8. Specifically, we discuss the expansion method for all types below.

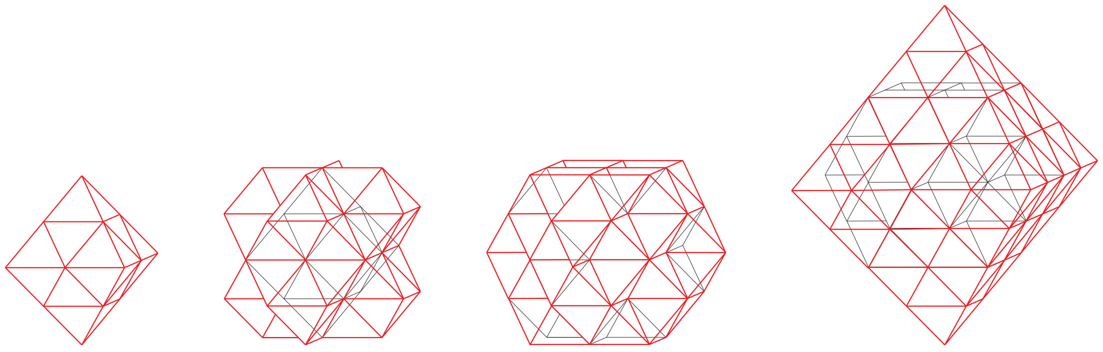
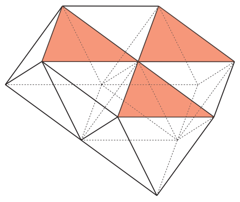


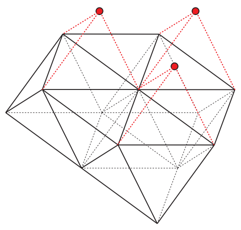
Figure 5.8: Three steps in the expansion process of the original mesh. Black lines sketch the expanded mesh in the previous step while red lines sketch additional structures added in the current step.

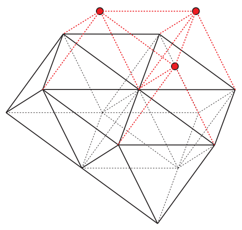
► *(Extend initial facets)* We insert a new virtual boundary vertex associated with each boundary facet that adjacent to an octahedron (red facets in Fig. 5.9). To calculate the position of the newly inserted vertex, we extract each of the three facets adjacent to the "red" facet, i.e., blue facets shown in Fig. 5.9(5), and obtain the position of the virtual vertex by forming a parallelogram. The final position of the virtual vertex is calculated through averaging these three trivial extrapolations. Equivalently, this extrapolation process can be summarized in the stencil shown in Fig. 5.9(4). Finally, after virtual vertices being inserted, we need to link them with proper edges, facets and cells.

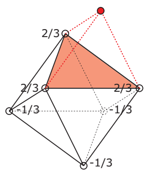
► *(Extend initial edges)* The extrapolation stencil depends on whether the edge is adjacent to a tetrahedron or an octahedron. The processes and stencils are shown in Fig. 5.10 and Fig. 5.11.

► *(Extend initial vertices)* The extrapolation stencil for the corner vertex also depends on the type of cell it adjacent to. For a corner vertex adjacent to a tetrahedron, the expansion process contains


(1) Step 0, red facets are the boundary


(2) Step 1, add one virtual vertex above each facet adjacent to octahedron using stencil in Fig. 5.9(4)


(3) Step 2, add proper edges, facets and cells



(4) Extrapolation stencil in step 1

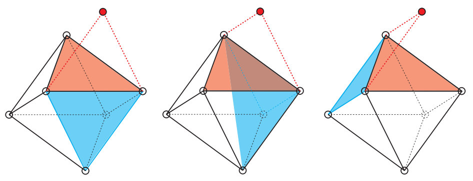
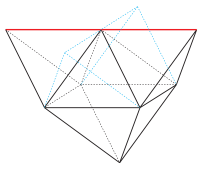
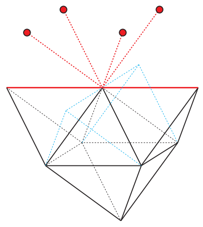
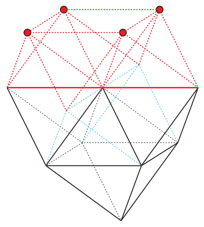
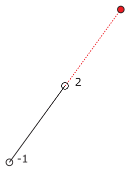

(5) The stencil in Fig. 5.9(4) is obtained through averaging three trivial extensions.

Figure 5.9: Process to extend the boundary facets


(1) Step 0, red edges are the border, blue edges are finished in the facet-extension step

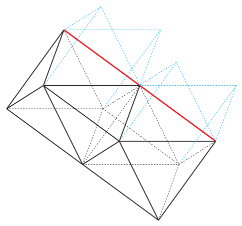

(2) Step 1, add one virtual vertex above each border vertex using stencil in Fig. 5.10(4)

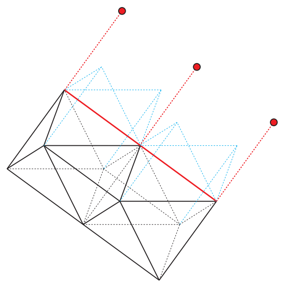

(3) Step 2, add proper edges, facets and cells

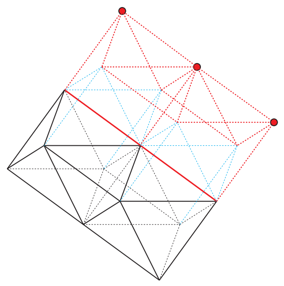


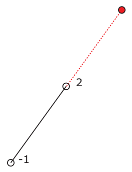
(4) Extrapolation stencil in step 1

Figure 5.10: Process to extend the boundary edges adjacent to tetrahedra


(1) Step 0, red edges are the border, blue edges are finished in the facet-extension step


(2) Step 1, add one virtual vertex above each border vertex using stencil in Fig. 5.11(4)

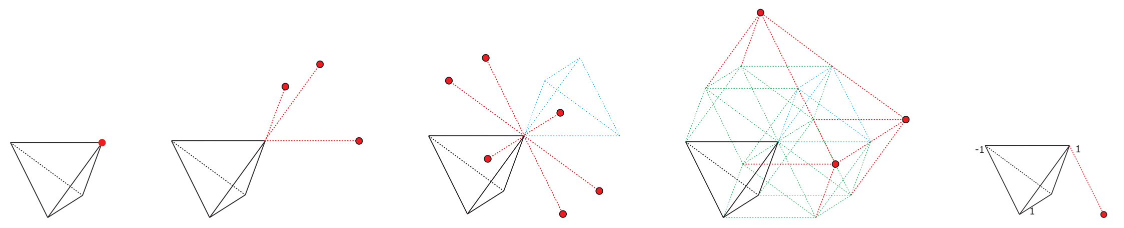

(3) Step 2, add proper edges, facets and cells



(4) Extrapolation stencil in step 1

Figure 5.11: Process to extend the boundary edges adjacent to octahedra

three steps as shown in Fig. 5.12. Furthermore, if a corner vertex is adjacent to an octahedron, then most works are already finished in the facet- and edge-extension steps. Additionally, We introduce a virtual corner vertex following the stencil in Fig. 5.13 and then introduce proper edges, facets and cells around it.



(1) Step 0, red vertex is the corner

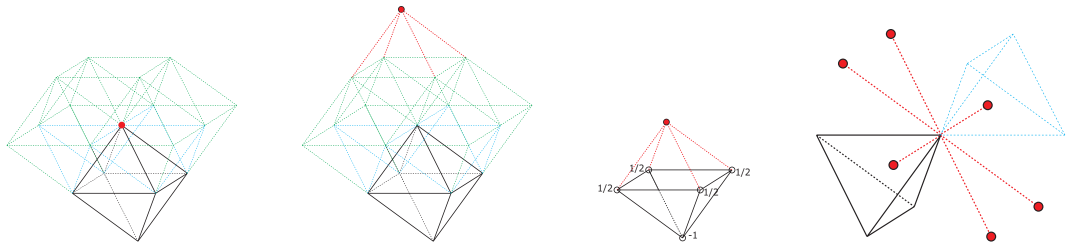
(2) Step 1, add three vertices using stencil in Fig. 5.10(4)

(3) Step 2, add six vertices using stencil in Fig. 5.12(5)

(4) Step 3, add three vertices using stencil in Fig. 5.13(3)

(5) Extrapolation stencil in step 1

Figure 5.12: Process to extend a corner vertex adjacent to a tetrahedron



(1) Step 0, red vertex is the corner, blue and green edges are finished in the facet-/edge-expansion steps

(2) Step 1, add one vertex using stencil in Fig. 5.13(3)

(3) Extrapolation stencil in step 1

(4) The stencil in Fig. 5.13(3) is obtained through averaging two trivial extensions.

Figure 5.13: Process to extend a corner vertex adjacent to an octahedron

Mathematically, for boundary vertices in the mesh, we can write down new geometric smoothing stencils which incorporate the expansion-trim process and the regular mesh subdivision stencil. New stencils can be generalized to deal with more complicated situations: if an original corner vertex is adjacent to multiple cells, we can calculate the position of the vertex in the refined mesh using each single cell and average these results to get the final position; similarly, for initial edges adjacent to multiple cells, we can also calculate the position for each wedge and average all results to update the position of the vertex.

We test the nonshrinking subdivision scheme on $\mathcal{M}_E$ and $\mathcal{M}_X$ (Fig. 2.10). For the former dataset, compared to original results in Fig. 2.2.2, the shrinkage phenomenon are completely removed in the results using the expansion method (Fig. 5.14). In the special case of $\mathcal{M}_E$, the subdivision process is identical to a topological splitting process. For initial meshes with more complicated geometric features, e.g., $\mathcal{M}_X$, compared to original results (Fig. 2.2.2), concavities are better preserved using the expansion method. The difference between two subdivision methods around concave regions can be seen more clearly from Fig. 5.16. Recall that, for the original 0-for subdivision scheme, the updating of original boundary vertices (i.e., even vertices lying on the boundary) is done through an affine combination with adjacent vertices which only lie in one side. The boundary vertex will move towards the side where other vertices are located, i.e., in concave regions, the exterior of the cell. As a result, the concavity is reduced. This difference between the two schemes becomes clearer if we increase the concavity in the original dataset $\mathcal{M}_X$ (see Fig. 5.17).

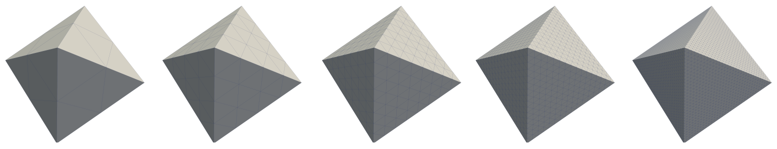


Figure 5.14: 0-form subdivision result using the expansion method for $\mathcal{M}_E$

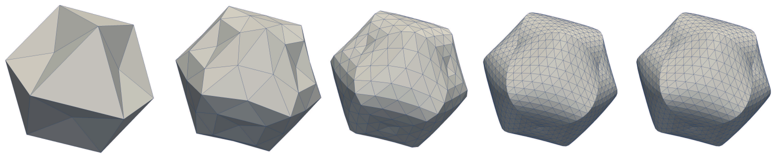


Figure 5.15: 0-form subdivision result using the expansion method for $\mathcal{M}_X$

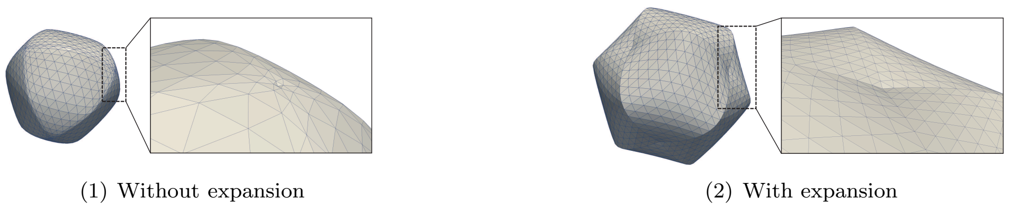


Figure 5.16: Zoomed-in results near the boundary of $\mathcal{M}_X$ using both of the simple method and the expansion method. Concavity is better preserved using the latter method.

## 5.5 Extension Methods for 1-Form and 2-Form

Fig. 5.18 shows the method to extrapolate 1- and 2-form coefficient fields to the virtual mesh obtained through the algorithm in Section 5.4.

Fig. 5.19 shows results using the extension method which has two advantages: comparing the zoomed-in illustrations in Fig. 5.4 with previous results (Fig. 5.20), one can see that the attenuation

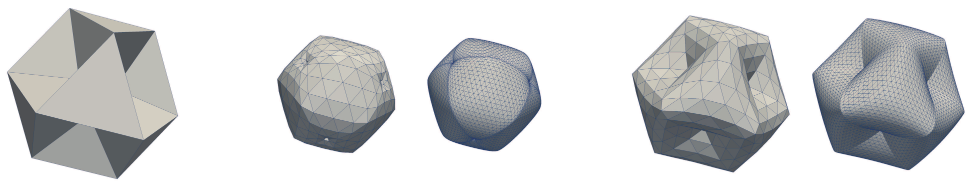
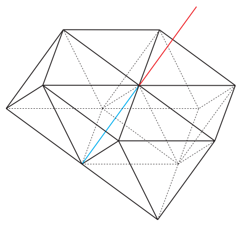
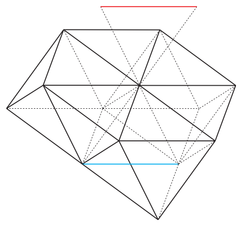
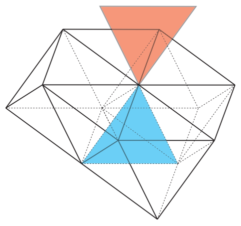
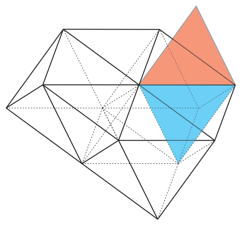


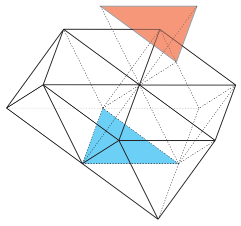
Figure 5.17: The figure in the left is the initial mesh $\mathcal{M}'_X$. The two figures in the middle are $\mathcal{M}'_X$ after 2 and 4 steps of subdivision without using the expansion method. The two figures on the right used the expansion method so the concave features are better preserved.


(1) (1-form) Case 1


(2) (1-form) Case 2


(3) (2-form) Case 1


(4) (2-form) Case 2


(5) (2-form) Case 3

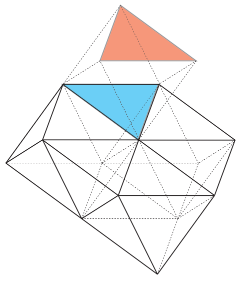
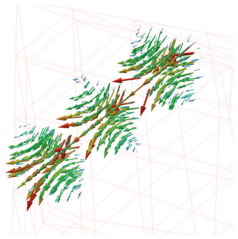
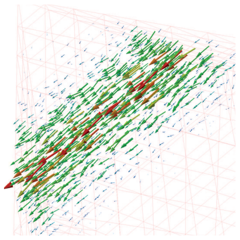
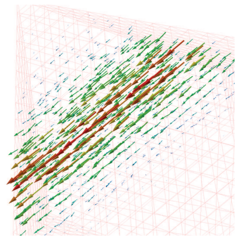

(6) (2-form) Case 4

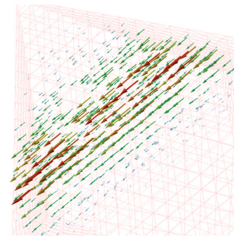
Figure 5.18: Extend the 1- and 2-form coefficients field to the virtual mesh. Blue edge is the real edge which carries nonzero 1-form coefficient while red edge is the corresponding edge in the virtual mesh to which we copy the coefficient.

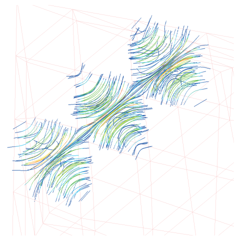
phenomenon near the boundary is removed; furthermore, comparing Fig. 5.19 with results in Fig. 5.6, initial edges or vertices, under the new extension method, introduces no noticeable curvatures in streamline-type visualizations.

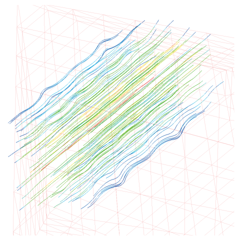

(1) Step 0, 1-form, vector-type

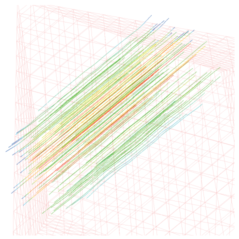

(2) Step 1, 1-form, vector-type

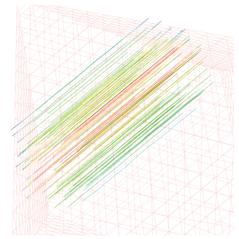

(3) Step 2, 1-form, vector-type

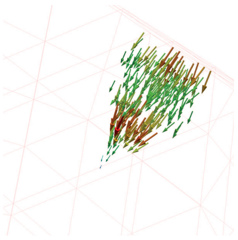

(4) Step 3, 1-form, vector-type

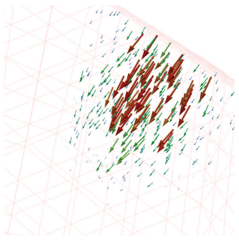

(5) Step 0, 1-form, streamline-type

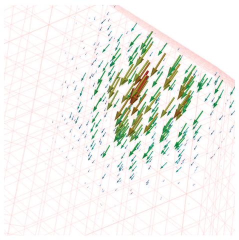

(6) Step 1, 1-form, streamline-type

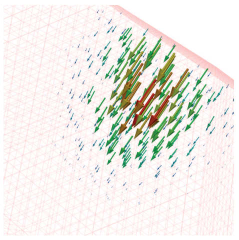

(7) Step 2, 1-form, streamline-type

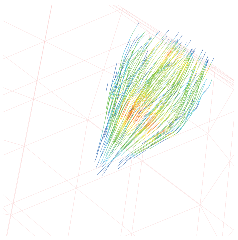

(8) Step 3, 1-form, streamline-type

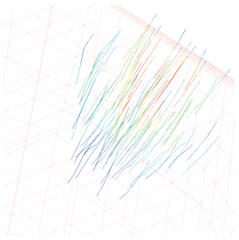

(9) Step 0, 2-form, vector-type

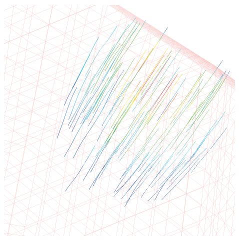

(10) Step 1, 2-form, vector-type


(11) Step 2, 2-form, vector-type


(12) Step 3, 2-form, vector-type


(13) Step 0, 2-form, streamline-type


(14) Step 1, 2-form, streamline-type


(15) Step 2, 2-form, streamline-type

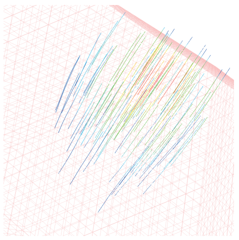
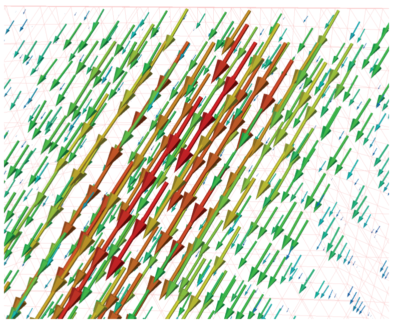

(16) Step 3, 2-form, streamline-type

Figure 5.19: 1- and 2-form subdivision results for $\mathcal{M}_B$ using the extension method. Previous artifacts (attenuation, enlarging or bend) are removed when the extension method is applied.


(1) Details near the boundary after three times subdivision using the expansion method (1-form)

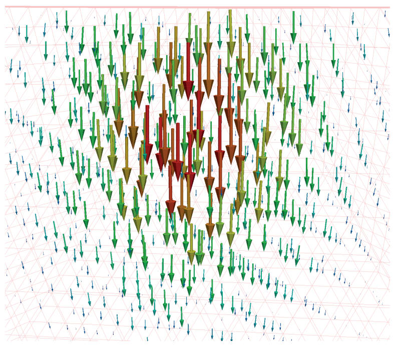

(2) Details near the boundary after three times subdivision using the expansion method (2-form)

Figure 5.20: Details near the boundary after subdividing $\mathcal{M}_B$ for three times using the extension method. The vector field is not attenuating near the boundary.

# Chapter 6

# Conclusions

Although the field of surface subdivision has existed for a long time, edge-based subdivision or 3D volumetric domain subdivision has not received much attention in the graphics community until recent years. In this thesis, we presented a novel subdivision method lying on the intersection of these two problems: a smooth differential forms subdivision method in the 3D volumetric domain.

The construction of 1- and 2-form subdivision stencils in regular cases contains three steps. First, linear subdivision schemes are constructed by solving commutative relations among 0-, 1-, 2- and 3-form subdivision matrices. Stencils for 1- and 2-forms can be determined up to a free parameter $\zeta$. We tried to solve commutative relations to obtain stencils of subdivision schemes with higher regularity but found it is an intimidating task because of the large size of the support (and hence the subdivision matrices). Alternatively, we kept the free parameter $\zeta$ and obtained stencils for smooth schemes by convoluting the generating function of linear schemes with a smoothing operator. We finally determine the optimal value for $\zeta$ using a novel technique based on spectrum and momentum considerations. We observe that, when we assign different values to $\zeta$, the $\zeta$-related eigenforms are similar and, compared to the $\zeta$-independent eigenforms, are unsmooth. In short, if we cannot make these free-parameter-related eigenforms smooth, we can choose the free parameter such that corresponding eigenvalues become small.

The convolution method does not apply to boundary edges or facets. If we simply ignore any nonzero form coefficient associated with boundary edge/facet, vector fields attenuate near the boundary and vanish completely on the boundary. To address this problem, we presented the projection method and the extension method.

The former method projects arbitrary differential forms into the subspace spanned by low-order forms (i.e., differential forms associated with potential fields following low-order polynomials). Subdivision stencil can be obtained then because we know how to extend low-order forms beyond the boundary. But under this method, the 1- or 2-form scheme is not compatible with the 0-form scheme. Under the original 0-form subdivision scheme, the volumetric domain, or more specifically, the normal direction of the boundary layer of cells, shrinks during the subdivision. This introduces enlarged vectors near the boundary of the mesh. A quick remedy is to introduce a scaling factor which offsets the shrinkage in the normal direction. However, in our experiment, some undesirable curvatures are introduced in the boundary region.

Alternatively, we amend boundary rules of the original 0-form subdivision and construct a non-shrinking version of the scheme. The new 0-form subdivision scheme can be interpreted as a product of a "expansion-trim" process. Essentially, the idea is to grow an additional virtual boundary layer above the real boundary layer to protect it. 1- and 2-forms can be straightforwardly extended into virtual elements. In our final experiment, we find no noticeable artifact such as attenuation, enlarging or undesirable curvature.

## 6.1 Future Works

The current project can be extended from at least three aspects: theory, implementation method and potential applications.

From the theoretical point of view, it is worthwhile to develop a systematic way to analyze the smoothness of the subdivision algorithm over octet meshes. For our algorithm, the smoothness in the regular setting is simple to study [44]. Further, previous researcher discussed the smoothness of the 0-form subdivision on initial facets using the joint spectral radius test [38]. However, as we proceed to the irregularities associated with initial edges (X2) or initial vertices (X3), smoothness analyses become extremely difficult because of the complexity of the topological/combinatorial configurations. For the boundary cases, although we proposed a method to expand the real mesh and perform the subdivision on the enlarged mesh, there is no guarantee that it is the optimal way to perform the cochain subdivision. Although a complete solution for either the irregular setting or the boundary setting may seems infeasible, we anticipate that future researchers can propose a systematic technique to analyze the smoothness for each specific topological configuration.

The implementation details of the current project can be found in Appendix A. This data structure is expeditious for the regular mesh. Based on this data structure, we can visit geometric elements efficiently, e.g., given a vertex and one of the adjacent octahedra, we can implement a useful helper function to visit the octahedron in the opposite direction. Iterators of the subdivision stencil can then be implemented based on the current design of data structure and these helper functions. However, to achieve the same level of efficiency in arbitrary irregular cases, e.g., visit tetrahedra or octahedra attached to an X3 vertex based on a certain sequence, one need to design a more generic and robust data structure.

Last but not least, there are a lot of potential applications of our discrete differential form subdivision. While the 1-form subdivision algorithm in the surface case can be used to construct fur shaders, the volumetric version can also be used in rendering and texture synthesis because functions such as non-uniform internal texture synthesis, volume rendering and anisotropic reflection are based on a vector/tensor field in the volumetric domain [41].

Applications also exist in fields outside graphics. As the ending of the thesis, we discuss one potential use of the method in the mixed finite element method, or specifically, in the area of computational electromagnetism. In the classical field theory, physical quantities such as $H$ and $E$ in Maxwell's equations can be recognized as 1-forms while $B$ and $J$ can be recognized as 2-forms [4, 5]. Circulations of $H$ and $E$ along certain paths or flux of $B$ and $J$ through certain facets all have physical meanings. On the other hand, Whitney forms on simplicial complex are related to finite elements, e.g., Nédélec elements and Raviart and Thomas elements are Whitney 1- and 2-forms, respectively [31, 32, 35]. Under our setting, higher-order Whitney forms are constructed through cochain subdivisions: commutative relations are preserved and smoothness across elements boundaries is guaranteed. These properties imply the sequence of de Rham complex which plays an important role in applications such as electromagnetism. Furthermore, subdivision operation and its transpose can also be recognized as prolongation and restriction in multigrid methods [19]. Finite elements based on subdivisions provide a hierarchy of granularities that can be utilized for coarse grid corrections. Therefore, we expect more explorations in the connection between discrete differential form subdivision and mixed finite elements in future research projects.

# Appendix A

# A Brief Review of the Implementation

This chapter briefly reviews implementation details of the 3D octet mesh form subdivision and vector field generation algorithms. As shown in Fig. A.1, from the top level, our process is identical with most subdivision algorithms. However, there are two main difficulties:

1. Unlike the classical 0-form subdivision scheme which is factorizable (see Section 2.2), for 1- and 2-forms, we need to create an efficient iterator over a much larger support, i.e., $\mathcal{M}_E$ (Section 2.1.2), which contains 60 edges and 56 facets. This requires us to create a robust and generic data structure based on that we can create a rich syntax to efficiently visit and iterate all geometric elements, e.g., "efficiently find out the vertex opposite to another vertex `v` in the octahedra `o`" or "efficiently find out all edges from the vertex `v` in the tetrahedron `t`". Details are discussed in Section. A.1.

2. Create a method to visualize 3D vector fields in the volumetric domain. The illustration need to catch as much information as possible and convey a clear interpretation based on that we can judge the smoothness of the subdivision scheme. Details are discussed in Section A.3.

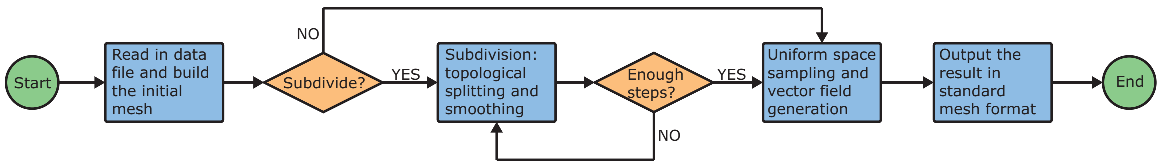

Start
Read in data file and build the initial mesh
Subdivide?
NO
YES
Subdivision: topological splitting and smoothing
Enough steps?
YES
NO
Uniform space sampling and vector field generation
Output the result in standard mesh format
End

Figure A.1: Process of the subdivision

## A.1 Data Structure

Fig. A.2 shows three layers in our data structure. This data structure provides generic and purely combinatorial containers. Based on these containers, concrete geometric elements can be defined. For example, we can use `Simplices<Facet>` to define a `Simplices` container for facets and use

Connector<Edge, Tet> to define a Connector container to store all topological connection relations between edges and tetrahedral cells.

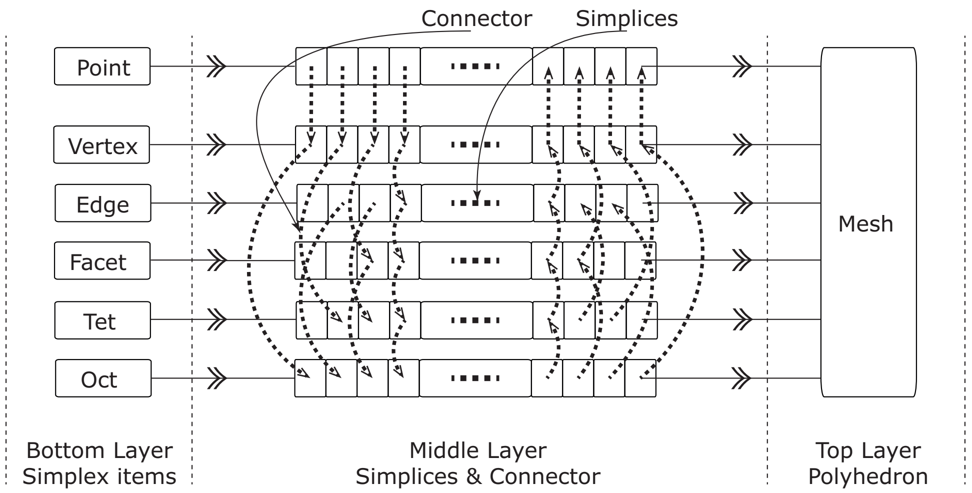


Figure A.2: The three-layer data structure in the design

## A.1.1 Bottom Layer

The bottom layer is the simplex layer where all types of the simplices are defined. In the octet mesh case, we have `Vertex`, `Edge`, `Facet`, `Tet` and `Oct`. All simplex items are `Vertex` based. For convenience, we also included the `Point` type in this layer.

Specifically, `Edge`, `Facet`, `Tet` and `Oct` contain indices of all included vertices. Indices are stored in a strictly ascending order. The sign of the form coefficient associated with a simplex may be flipped during the construction of the object. For example, if we want to define an edge from vertex 63 to vertex 57 with the 1-form coefficient 1.5, then in the mesh, what finally stored is an edge from vertex 57 to vertex 63 with the 1-form coefficient -1.5. This strictly ascending order helps us to implement the associative-container-based data structure in the middle layer and enables us to use functions similar to the hash-table `find` (Section A.1.2).

Furthermore, a one-to-one correspondence between all `Point` objects and `Vertex` objects is enforced. The former carries geometric information such as the coordinate of the control point while the latter only carries topological and combinatorial information. The geometric information defined in `Point` will only be used in the geometric smoothing step in the 0-form subdivision and in the calculation of Whitney forms. The topological splitting of the mesh can be accomplished by purely using those "topological simplex" items (i.e., `Vertex`, `Edge`, `Facet`, `Tet` and `Oct`).

### A.1.2 Middle Layer

The simplices data structure (abbreviate to `Simplices`) is a sequential-and-associative-container-based data structure which is capable to store information of all vertices, edges, facets, tetrahedra and octahedra. The `Simplices` provides two main functions which are extensively used throughout the implementation:

1. Given the index of a specific simplex, return the reference referred to the simplex;
2. Given a concrete simplex object, return the index of this object in the mesh.

*Remark:* The second function is implemented through a hash table `find`. For `Vertex`, `Edge`, `Facet`, `Tet` and `Oct`, we use indices of all included vertices as the key and use the index of the simplex as the value. Because vertices are stored in an increasing order of indices, we can compare two simplices directly. However, for `Point`, we used the coordinate of the object as the key. When we compare two points, numerical precision issues need to be considered.

The connector data structure (abbreviated to `Connector`) is an associative-container-based data structure which is capable to accomplish following tasks.

1. Maintain adjacency-inclusion information among all simplices in the mesh. For example, we can find out facets adjacent to a specific edge or vice versa, edges included in a specific facet using `Connector`.

2. Maintain parent-child relationship information between the same type of simplices (here we define the `Tet` and the `Oct` as the same type) in two consecutive levels of meshes during the subdivision. For example, the map between one facet in the coarse mesh and the corresponding four child-facets in the refined mesh can be found using `Connector`.

### A.1.3 Top Layer

The top layer (polyhedron layer in our case) is a collection of all elements belong to the middle layer. Other applications such as subdivision surfaces can also be build up based on the bottom and middle layers by excluding `Tet`- and `Oct`-related elements, containers and connectors.

One may extend the top layer to include additional data members that are useful for specific tasks. For example, in volumetric domain form subdivision tasks, we add in Boolean variables to indicate whether the geometric smoothing step is needed for a specific geometric element, i.e., for a geometric element, if all corresponding elements in the subdivision support have zero form coefficients, we only need to perform the topological splitting it. We can also include additional Boolean variable associated with every cell to indicate if the cell contains any nonzero 1- or 2-form coefficient. If all edges and facets carry zero coefficient, then we don't need to generate vector fields inside the cell. These indicator are initialized in the preprocessing step before the subdivision.

## A.2 Topological Splitting and Geometric Smoothing

The subdivision process is summarized in Algorithm 2. Given a coarse mesh with specific size, we can calculate the size of the refined mesh (Table A.1). Correct amount of containers can be dynamically acquired before the splitting.

| | Before splitting | After splitting |
|---|---|---|
| Vertices | $N_v$ | $N_v + N_e + N_o$ |
| Edges | $N_e$ | $2N_e + 3N_f + 12N_o$ |
| Facets | $N_f$ | $4N_f + 4N_t + 24N_o$ |
| Tetrahedra | $N_t$ | $4N_t + 8N_o$ |
| Octahedra | $N_o$ | $N_t + 6N_o$ |

(A.1)

## A.3 Visualization

### A.3.1 Uniform Sampling

In order to visualize the vector field associated with a form coefficient field on the mesh, we create a sample in the volumetric domain and then calculate the vector from each sample point (*seed*). In many applications, we want seeds to be uniformly distributed. For surface subdivisions, the uniform sampling, together with related problems such as isotropic surface remeshing and surface parameterization, are highly nontrivial. However, in the volumetric domain, we simply create points equally distant in all three axes. The most efficient method is to create a uniform sample for each single cell separately. However, as required by the streamline calculations, we need to embed a uniform structured rhombohedral mesh (a *sampling mesh*) into the volumetric domain and use the vertices in the sampling mesh as seeds.

After creating the sampling mesh, we need to classify all points into the correct cell. This step is time consuming although it is simple to be parallelized. For $N$ seeds and $K$ cells, the complexity is $O(NK)$. For examples we provided in previous chapters, $N \gtrsim 10^4$ and $K \gtrsim 10^5$. Because of this efficiency issue, for the sampling mesh, besides the requirement of covering the whole domain with nonzero form coefficients, we want the rhombohedral sampling mesh to be as small as possible. In order to achieve this, we collect all cells with nonzero form coefficient (i.e., with nonzero vector field inside) and perform the principal axes computation over vertices of these nonzero cells to obtain the orientation and size of the rhombohedral sampling mesh.

Similar to Section 3.1.1, in order to determine if a seed is inside a specific cell, we create a half-line from the seed and count its number of intersections with all facets in the mesh. If the there is odd number of intersections, the seed is inside, otherwise it is outside. Numerical precision issues need

**Algorithm 2** Single-Step 3D Mesh Form Subdivision

```
Require: The mesh is well defined, complete and free of conflict
1: for Each cell (tetrahedron or octahedron) in the mesh do
2:   for Each facet in the cell do
3:     for Each edge in the facet do
4:       if The edge has not been split then
5:         Insert 1 new vertex in the center of the edge and create 2 E1
           child-edges
6:         Perform the geometric smoothing on the 2 E1 child-edges
           (Fig. 4.5)
7:         Mark the parent-edge as being split
8:       end if
9:     end for
10:    if The facert has not been split then
11:      Insert 3 E2 child-edges in the center, 3 F1 child-facets in the
         corner and 1 F4 child-facet in the center of the parent facet
12:      Perform the geometric smoothing on the 3 E2 edges (Fig. 4.6),
         3 F1 facets (Fig. 4.8) and 1 F4 facet (Fig. 4.11)
13:      Mark the parent-facet as being split
14:    end if
15:  end for
16:  if The cell is a tetrahedron then
17:    Insert 6 F2 child-facets in the center, 4 child-tetrahedra in the cor-
       ner and 1 child-octahedron in the center of the parent-tetrahedra
18:    Perform the geometric smoothing on the 6 F2 facets (Fig. 4.9), 4
       tetrahedra and 1 octahedron
19:    Mark the parent-tetrahedron as being split
20:  else {The cell is an octahedron}
21:    Insert a new vertex in the center of the octahedron
22:    Insert 12 E3 child-edges in the center, 24 F3 child-facets in the
       center, 8 child-tetrahedra in the center and 6 child-octahedra in
       the corner of the parent-octahedra
23:    Perform the geometric smoothing on the 12 E3 edges (Fig. 4.7), 24
       F3 facets (Fig. 4.10), 8 tetrahedra and 6 octahedra
24:    Mark the parent-octahedron as being split
25:  end if
26: end for
```

to be treated carefully in all steps. However, if a seed lies on a facet, it induces ambiguity in the classification. This issue will almost surely be avoided if we create a slightly larger rhombohedral sampling mesh and introduce small random numbers as offsets of boundaries.

### A.3.2 Vector Field Generation

As we mentioned in Section 3.2.2, for seeds inside a tetrahedron, we can collect the form coefficients from all edges and facets to construct the vector associated with the 1- and 2-form, respectively, by using the closed form formula in Eq(3.13) and Eq(3.13).

However, there is no closed form expression for Whitney form inside the octahedral cell. Alternatively, we use the linear subdivision introduced in Section 4.1 to subdivide the single octahedral cell up to a certain number of steps. During these subdivisions, if the seed ever falls into a tetrahedron, we can calculate the vector associated with it using the closed form formula. If it always lies in octahedral cells (for simplicity, we call these parts the *octahedron-region*), we use one of the approximation rules discussed below to estimate the vector.

*Remark:* As mentioned in Section 2.2, the linear 0-form subdivision is the same as the topological splitting, i.e., the volumetric domain is preserved during the subdivision.

For a single regular octahedral cell, we can estimate the proportion of the octahedron-region after $L$ times of linear subdivision. One step of subdivision generates 8 tetrahedron and 6 octahedron in the refined level. Because the volume of one child-octahedra is 4 times of the volume of one child-tetrahedra, after one step of subdivision, $\frac{6\times 4}{6\times 4+8\times 1} = \frac{3}{4}$ of the volumetric domain remains to be in one of the child-octahedra. Similarly, after $L$ steps, $\left(\frac{3}{4}\right)^L$ of the initial volume will be the octahedron-region (e.g., after 8 times of linear subdivision, the ratio will be 10% approximately).

We propose three strategies to approximate vectors associated with seeds in the octahedron-region:

*Strategy 1:* Assign zero vectors to these seeds. This is the fastest way. If we finally illustrate the vector field using arrows, we can get nice visualizations because after enough times of linear subdivision (based on our experience, 5 to 8 times are usually enough), each octahedral cell in the octahedron-region will be small and these cells are uniformly spread out in the original single octahedral cell. However, if we the streamline-type visualization, artifacts will be introduced because of these “fake-zero-seeds”.

*Strategy 2:* For a specific octahedron in the octahedron-region, iterate all adjacent tetrahedra and calculate vectors associated with barycenters of these tetrahedra. Then we simply use the mean of these vectors to approximate vectors in the octahedron. Artifacts introduced by Strategy 1 are removed using this method.

*Strategy 3:* In Strategy 2, we essentially implemented a spatial moving average filter near the octahedron-region. Because octahedra in the octahedron-region are usually small, the filtered result is usually a good approximation. However, when ill-conditioned tetrahedron exists, the approximation error may become large.

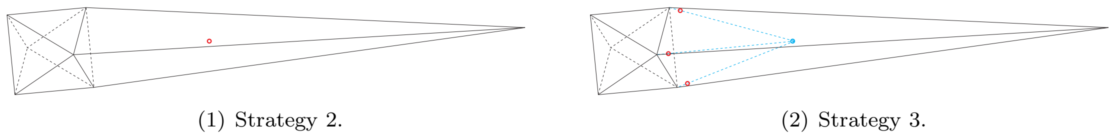

(1) Strategy 2. (2) Strategy 3.

Figure A.3: Comparison between *Strategy 2* and *3* in the ill-conditioned tetrahedron case. Red dot(s) represents the points we will collect to perform the moving average approximation.

As shown in Fig. A.3(2), to reduce the approximation error, instead of collecting barycenters, we can collect points near interface vertices to perform the vector averaging. These points (red dots) are obtained using an affine combination between interface vertices and the barycenter. All results in this thesis, such as the illustrations of Whitney forms inside octahedron (Fig. 3.6), are produced using *Strategy 3*.

# Bibliography

[1] R. Abraham, J. E. Marsden, and T. Ratiu. *Manifolds, Tensor Analysis, and Applications.* Addison-Wesley Publishing Company, London, 1983.

[2] D. N. Arnold, R. S. Falk, and R. Winther. Finite element exterior calculus, homological techniques, and applications. *Acta Numer.*, 15:1–155, 2006.

[3] C. L. Bajaj, J. Warren, and G. Xu. A smooth subdivision scheme for hexahedral meshes. Technical report, University of Texas at Austin, Austin, TX, USA, 2001.

[4] A. Bossavit. Whitney forms: a class of finite elements for three-dimensional computations in electromagnetism. *Physical Science, Measurement and Instrumentation, Management and Education - Reviews, IEE Proceedings A*, 135(8):493–500, November 1988.

[5] A. Bossavit. *Computational electromagnetism.* Academic Press, Inc., San Diego, CA, USA, 1998.

[6] G. Brinkmann, S. Greenberg, C. S. Greenhill, B. D. McKay, R. Thomas, and P. Wollan. Generation of simple quadrangulations of the sphere. *Discrete Mathematics*, 305(1–3):33–54, 2005.

[7] G. Brinkmann and B. D. McKay. Construction of planar triangulations with minimum degree 5. *Discrete Mathematics*, 301(23):147–163, 2005.

[8] É. Cartan. *Les Systèmes Differentiels Exterieurs et leurs Applications Géometriques.* Hermann, Paris, France, 1945.

[9] E. Catmull and J. Clark. Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, 1978.

[10] Y. S. Chang, K. T. McDonnell, and H. Qin. A new solid subdivision scheme based on box splines. In *Proceedings of the seventh ACM symposium on Solid modeling and applications*, SMA '02, pages 226–233, 2002.

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

[12] S. S. Chern. *Differential Manifolds.* University of Chicago, Chicago, IL, USA, 1953.

[13] A. Cohen, N. Dyn, and D. Levin. Matrix subdivision schemes. In *Tutorials on Multiresolution in Geometric Modelling, Mathematics and Visualization*, pages 25–50. Springer, 1995.

[14] R. W. R. Darling. *Differential Forms and Connections.* Cambridge University Press, Cambridge, UK, 1994.

[15] T. DeRose, M. Kass, and T. Truong. Subdivision surfaces in character animation. In *Proceedings of the 25th annual conference on computer graphics and interactive techniques*, SIGGRAPH '98, pages 85–94, 1998.

[16] M. Desbrun, E. Kanso, and Y. Tong. Discrete differential forms for computational modeling. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, pages 39–54, 2006.

[17] M. P. Do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, Englewood Cliffs, NJ, USA, 1976.

[18] D. Doo and M. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Comput. Aided Des.*, 10(6):356–360, November 1978.

[19] S. Green, G. Turkiyyah, and D. Storti. Subdivision-based multilevel methods for large scale engineering simulation of thin shells. In *Proceedings of the 7th ACM symposium on solid modeling and applications*, pages 265–272, 2002.

[20] E. Grinspun, P. Krysl, and P. Schröder. Charms: a simple framework for adaptive simulation. In *Proceedings of the 29th annual conference on computer graphics and interactive techniques*, SIGGRAPH '02, pages 281–290, 2002.

[21] A. Hatcher. *Algebraic Topology.* Cambridge University Press, 2004.

[22] C. Heil and D. Colella. Matrix refinement equations: Existence and uniqueness. *J. Fourier Anal. Appl*, 2:363–377, 1996.

[23] J. Huang, L. Chen, X. Liu, and H. Bao. Efficient mesh deformation using tetrahedron control mesh. *Comput. Aided Geom. Des.*, 26:617–626, August 2009.

[24] J. Huang and P. Schröder. $\sqrt{3}$-based 1-form subdivision. In *Curves and Surfaces*, Lecture Notes in Computer Science Volume 6920, pages 351–368. Springer Berlin/Heidelberg, 2012.

[25] K. Joy and R. MacCracken. The refinement rules for Catmull-Clark solids. Technical Report CSE-91-1, Computer Science Department, University of California, Davis, USA, January 1996.

[26] A. Khodakovsky, P. Schröder, and W. Sweldens. Progressive geometry compression. In *Proceedings of the 27th annual conference on computer graphics and interactive techniques*, SIGGRAPH '00, pages 271–278, 2000.

[27] L. Kobbelt. $\sqrt{3}$-subdivision. In *Proceedings of the 27th annual conference on computer graphics and interactive techniques*, SIGGRAPH '00, pages 103–112, 2000.

[28] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, Department of Mathematics, University of Utah, Utah, USA, August 1987.

[29] R. MacCracken and K. I. Joy. Free-form deformations with lattices of arbitrary topology. In *Proceedings of the 23rd annual conference on computer graphics and interactive techniques*, SIGGRAPH '96, pages 181–188, 1996.

[30] J. R. Munkres. *Elements of Algebraic Topology.* Addison-Wesley, Menlo Park, CA, USA, 1984.

[31] J. C. Nédélec. Mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, 35:315–341, 1980.

[32] J. C. Nédélec. A new family of mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, 50:57–81, 1986.

[33] P. Oswald and P. Schröder. Composite primal/dual $\sqrt{3}$-subdivision schemes. *Comput. Aided Geom. Des.*, 20(3):135–164, June 2003.

[34] J. Peters and U. Reif. *Subdivision Surfaces.* Springer-Verlag Berlin/Heidelberg, 2008.

[35] P. Raviart and J. Thomas. A mixed finite element method for 2nd order elliptic problems. In I. Galligani and E. Magenes, editors, *Mathematical Aspects of Finite Element Methods*, Lecture Notes in Mathematics Volume 606, pages 292–315. Springer Berlin/Heidelberg, 1977.

[36] U. Reif. A unified approach to subdivision algorithms near extraordinary vertices. *Comput. Aided Geom. Des.*, 12(2):153–174, March 1995.

[37] W. Rudin. *Principles of mathematical analysis.* International Series in Pure and Applied Mathematics. McGraw-Hill Book Co., New York, third edition, 1976.

[38] S. Schaefer, J. Hakenberg, and J. Warren. Smooth subdivision of tetrahedral meshes. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on geometry processing*, SGP '04, pages 147–154, 2004.

[39] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. *SIGGRAPH Comput. Graph.*, 20:151–160, August 1986.

[40] M. Spivak. *Calculus on manifolds: A modern approach to classical theorems of advanced calculus.* W. A. Benjamin, Inc., New York-Amsterdam, 1965.

[41] K. Takayama, M. Okabe, T. Ijiri, and T. Igarashi. Lapped solid textures: filling a model with anisotropic textures. In *ACM SIGGRAPH 2008*, pages 53:1–9, 2008.

[42] K. Wang. *A subdivision approach to the construction of smooth differential forms.* PhD dissertation, California Institute of Technology, Pasadena, CA, USA, 2008.

[43] K. Wang, Weiwei, Y. Tong, M. Desbrun, and P. Schröder. Edge subdivision schemes and the construction of smooth vector fields. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, pages 1041–1048, 2006.

[44] J. Warren and H. Weimer. *Subdivision Methods for Geometric Design: A Constructive Approach.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.

[45] H. Whitney. *Geometric integration theory.* Princeton University Press, Princeton, NJ, USA, 1957.

[46] D. Zorin and P. Schröder. Subdivision for modeling and animation. In *ACM SIGGRAPH 2000 Courses*, 2000.