Eurographics Symposium on Geometry Processing 2009
Marc Alexa and Michael Kazhdan
(Guest Editors)

Volume 28 (2009), Number 5

Diamond Hierarchies of Arbitrary Dimension

Kenneth Weiss[1] and Leila De Floriani[2]

[1]University of Maryland, College Park
[2]University of Genova, Italy

Abstract

Nested simplicial meshes generated by the simplicial bisection decomposition proposed by Maubach [Mau95] have been widely used in 2D and 3D as multi-resolution models of terrains and three-dimensional scalar fields, They are an alternative to octree representation since they allow generating crack-free representations of the underlying field. On the other hand, this method generates conforming meshes only when all simplices sharing the bisection edge are subdivided concurrently. Thus, efficient representations have been proposed in 2D and 3D based on a clustering of the simplices sharing a common longest edge in what is called a diamond. These representations exploit the regularity of the vertex distribution and the diamond structure to yield an implicit encoding of the hierarchical and geometric relationships among the triangles and tetrahedra, respectively. Here, we analyze properties of d-dimensional diamonds to better understand the hierarchical and geometric relationships among the simplices generated by Maubach's bisection scheme and derive closed-form equations for the number of vertices, simplices, parents and children of each type of diamond. We exploit these properties to yield an implicit pointerless representation for d-dimensional diamonds and reduce the number of required neighbor-finding accesses from $O(d!)$ to $O(d)$.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Hierarchy and geometric transformations I.3.6 [Computer Graphics]: Methodology and Techniques—Graphics data structures and data types

1. Introduction

Mathematical and scientific computing applications are often approached using a divide-and-conquer paradigm. Examples are the analysis and visualization of two- and three-dimensional scalar fields, where the domain of the field is adaptively decomposed into nested cells of a simple geometric shape. This divide-and-conquer paradigm produces effective multi-resolution models of the scalar field, which allow an efficient manipulation of the field representation by generating suitable variable-resolution representations according to user requirements.

Subdivision strategies based on quadtrees and octrees have been popular for regularly sampled data points inside square or cubic domains. The drawback of these techniques is that they introduce an exponential number of cells (of the order d of the dimension of the domain) during each subdivision. Furthermore, both quadtrees and octrees (as well as their d-dimensional generalization, that we call a 2^d-tree) are less suitable for generating conforming (i.e. crack-free) decompositions since the bilinear or trilinear interpolant over each square or cubic cell generates discontinuities on the boundary of two adjacent cells if the cells have different sizes. Thus, additional rules must be applied to ensure compatibility between neighboring cells.

In contrast, the simplicial bisection decomposition proposed by Maubach [Mau95] enables the generation of more adaptive meshes over the same domain by breaking up each 2^d-tree subdivision into d steps. This method generates conforming meshes, and thus crack-free representations of the underlying field, only when all simplices sharing the bisection edge are subdivided concurrently. This is typically accomplished through a neighbor-finding operation which cycles through all such neighbors of a simplex σ and requires $O(|Neighbors(\sigma)|)$ accesses to the data structure. However, despite the widespread use of these hierarchies, no general analysis of the number of such neighbors in arbitrary dimensions has been carried out.

Efficient representations have been proposed for multiresolution models of scalar fields in 2D and 3D based on clustering maximal simplexes (triangles or tetrahedra, in these cases) sharing their bisection edge into a so-called *diamond*. These representations exploit the regularity of the vertex distribution, and of the subdivision rule, which produces diamonds of certain fixed shapes, to yield an implicit encoding of the hierarchical and geometric relationships among the triangles and tetrahedra, respectively.

These subdivisions have many applications in the analysis and visualization of scientific and medical data. They have been applied to interactive terrain rendering [EKT01, LP02, HDJ05] and multiresolution isosurface [GDL*02] and interval volume extraction [WD08], as well as volume segmentation [KTY*04], surface reconstruction [MVT03] and finite element analysis [Mau95, RL92]. Four-dimensional applications include multiresolution representations for time-varying volumetric datasets [LDS04, LPD*04], acceleration structures for ray tracing [AMM07] and the analysis of bivariate complex functions [WB96]. Possible higher dimensional applications include five-dimensional weather data [HAF*96], fixed point computations [Mau95] and the solution spaces of parametrized equations.

Our contribution here is a formalization of the notion of diamonds in arbitrary dimensions and their relationship to the simplices generated by Maubach's bisection scheme. We frame our discussion of diamonds in terms of a cross product of two related simplicial decompositions of hypercubes. Through a careful analysis of the properties of these structures, we prove that diamonds can be decomposed as a cross product of these two structures and derive closed-form equations for the number of vertices, simplices, parents and children of each type of diamond. Specifically, we prove that d-dimensional diamonds contain $O(d!)$ d-simplices, and thus, in general, neighbor-finding on these simplicial meshes requires $O(d!)$ iterations. However, due to the regularity of the diamond subdivision operation these simplices can be grouped into $O(d)$ clusters that are subdivided simultaneously. Consequently, while simplex-based representations require $O(d!)$ time to update and space to encode an extracted simplicial complex, diamond-based representations can index $O(d!)$ d-simplices, using only $O(d)$ time and space. Finally, we provide a compact pointerless representation for d-dimensional diamonds, enabling efficient representations for subsets of a hierarchy of diamonds.

The remainder of this paper is organized as follows. Section 2 provides some background notions on hypercubes and simplicial complexes and introduces the notions of cross simplex and cross complexes. Section 3 presents a review of related work on regular simplicial subdivisions and on extractions of conforming representations. Section 4 discusses simplicial d-complexes generated by subdividing a d-dimensional hypercube. We first review Kuhn's subdivisions and Maubach's decomposition. We then introduce the notion of fully subdivided hypercubes and prove relevant properties. Section 5 discusses the hierarchy of simplexes, which is a common representation for complexes generated through Maubach's subdivision rule and which has been investigated in arbitrary dimensions. In Section 6 we define the hierarchy based on diamonds, that we call a *hierarchy of diamonds* in a dimension-independent way, while in Section 7, we discuss and prove properties of diamonds related to their geometry and to the hierarchical organization. In Section 8, we introduce a dimension-independent representation for a hierarchy of diamonds. In Section 9 we present an efficient pointerless representation for conforming simplicial meshes based on the diamond primitive and compare this representation to those based on the simplex primitive. Finally, we draw some concluding remarks in Section 10.

2. Background Notions

In this Section, we review some background notions on hypercubes and on simplicial complexes and introduce some concepts that we will use in the rest of the paper.

2.1. Hypercubes

Hypercubes, or d-cubes, are the d-dimensional analogues of squares (2-cubes) and cubes (3-cubes). A 0-cube is a single point, and a d-cube is created by extruding a $(d-1)$-cube one unit along a direction orthogonal to the previous $(d-1)$ directions (see Figure 1). Unless otherwise indicated, we refer to axis-aligned hypercubes, where all such directions are parallel with a coordinate axis in $\mathbb{R}^d$.

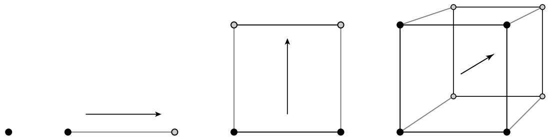


Figure 1: *Hypercubes: recursive definition.*

Given a d-cube h, an *i-face* of h is any i-cube on the boundary of h, where $0 \leq i \leq d$. The number of i-faces of a d-cube is given by $2^{d-i}\binom{d}{i}$. Of special importance are the 0-faces, called *vertices*, the 1-faces, called *edges* and the $(d-1)$-faces, called *facets* of h.

The *diameter* of a polytope p is defined as the maximum distance between any two points on the boundary of p. For a d-cube h, the diameter is referred to as a *diagonal* and is defined by *opposite vertices*, i.e. a pair of vertices whose only common face is h. Let $(\mathbf{v}_1,\mathbf{v}_2)$ be an unordered pair of opposite vertices. Then the edge between $\mathbf{v}_1$ and $\mathbf{v}_2$ forms a diagonal of h and has length $\sqrt{d}$. Similarly, a diagonal of an i-face of h has length $\sqrt{i}$.

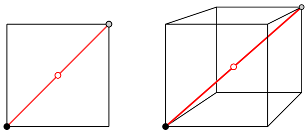


Figure 2: *Diagonal of a hypercube*

2.2. Simplicial complexes

A d-dimensional *simplex*, or d-simplex, is the convex hull of $(d+1)$ affinely independent points in the n-dimensional Euclidean space. An i-face of a d-simplex σ is the i-simplex defined by any $(i+1)$ vertices of σ. The number of i-faces of a d-simplex is thus $\binom{d+1}{i+1}$.

A *simplicial mesh* Σ is a collection of simplices Σ such that all faces of a simplex $\sigma \in \Sigma$ belong to Σ, and the interiors of simplices from Σ are disjoint. A simplicial mesh is *conforming* if the intersection between any two simplices σ_1 and σ_2 is either empty or a face belonging to the boundary of both of them. A *simplicial complex* is a conforming simplicial mesh.

If d is the maximum dimension of any simplex in a simplicial complex Σ, then we call Σ a *simplicial d-complex*, and refer to its d-simplices as *cells*. Further, Σ is a *pure simplicial complex* if all of its simplices are faces of cells in Σ. For the remainder of this paper, all simplicial complexes are assumed to be pure.

2.3. Cross simplexes and cross complexes

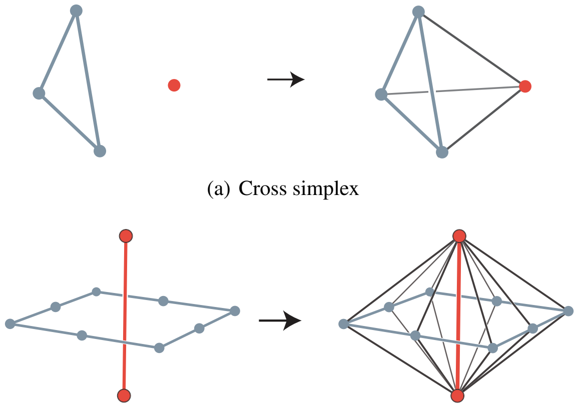
We utilize the *simplicial join* operation [RS72,Lic99] to generate higher-dimensional simplices from a pair of affinely independent simplices and refer to the result as a *cross simplex*. Given an a-simplex σ_a and a b-simplex σ_b in affinely independent subspaces, the cross simplex is the d-simplex $\sigma = \sigma_a \otimes \sigma_b$, defined by the vertices of σ_a and σ_b, where $d = a+b+1$.

For example, if σ_a is a 2-simplex (triangle) defined by 3 vertices and σ_b is a 0-simplex defined by a single vertex, then the cross simplex $\sigma = \sigma_a \otimes \sigma_b$ is the 3-simplex (tetrahedron) defined by the vertices of σ_a and σ_b (see Figure 3(a)).

Given simplicial i-complex Σ_i and simplicial j-complex Σ_j whose cells are pairwise affinely independent, we define the *cross complex* $\Sigma_d = \Sigma_i \otimes \Sigma_j$ as the simplicial d-complex whose d-simplices are cross simplices of cells from Σ_i and Σ_j, i.e., $\forall$ cells $\sigma \in \Sigma_d$, $\sigma = \sigma_i \otimes \sigma_j$, where σ_i is a cell of Σ_i and σ_j is a cell of Σ_j (see Figure 3(b)).

3. Related work

In this Section, we discuss methods related to simplicial subdivisions of a regular domain as well as the extraction of conforming simplicial complexes from these representations.



(a) Cross simplex

(b) Cross complex

Figure 3: *(a) The cross simplex of a triangle (blue) and a vertex (red) is a tetrahedron. (b) A cross complex defined by an 8 edge complex (blue) and a one edge complex (red).*

The canonical subdivision of a d-cube into $d!$ simplices was originally proposed by Freudenthal [Fre42] and popularized by Kuhn [Kuh60] in the context of fixed points computations. [Bey00] provides a comprehensive review of simplicial refinement strategies.

Simplex bisection techniques subdivide each d-simplex into two d-simplices along an edge. In 2D, Mitchell's *newest vertex bisection* [Mit92] chooses the edge opposite the most recently introduced vertex of a triangle to bisect. An alternative approach [RL92] is to bisect the longest edge of each triangle or tetrahedron. Maubach [Mau95] generalizes these techniques to grids of arbitrary dimension, and proves that simplex bisection, when applied to d-dimensional Kuhn-subdivided grids, generates at most d similarity classes of d-simplices.

Simplicial bisection admits the extraction of highly adaptive simplicial complexes when all d-simplices sharing the bisection edge are bisected concurrently. When using d-simplices as the modeling primitive, this requires an efficient *neighbor-finding* algorithm to locate all such neighbors. Symbolic neighbor-finding algorithms that run in logarithmic [Heb94, Mau96] or constant [EKT01, LDS04, AMM07] time have been developed for regular grids in 2D [EKT01], 3D [Heb94], 4D [LDS04] and arbitrary dimensions [Mau96, AMM07]. Maubach [Mau96] proves that the set of neighbors can be characterized by a connected $(d-2)$-surface and conjectures that this surface is simply-connected. Although the neighbor-finding algorithm must run $O(|Neighbors(\sigma)|)$ times to bisect a simplex σ, to the best of our knowledge, there have been no attempts to describe the number of such neighbors of a d-simplex on a regular grid. We prove that there are $O(d!)$ such neighbors, and reduce the number of required spatial accesses to $O(d)$.

An alternate approach is to cluster the set of d-simplices that share a bisection edge into *diamonds*. Efficient encodings of diamonds have been developed in 2D [LP02, HDJ05] and 3D [GDL*02, WD08], where the regularity of the updates and vertex distribution enables an implicit encoding of the geometric and hierarchical relationships among the simplices. Pascucci [Pas02] generalizes the diamond subdivision paradigm to general d-dimensional meshes but does not provide a representation for such meshes or an analysis of their properties in d dimensions. Linsen et al. [LPD*04] use this paradigm in the context of subdivision wavelets with local support for 2D, 3D and 4D scalar fields. In 3D, their adaptive structure is that of Gregorski et al. [GDL*02].

4. Simplicial Subdivisions

In this Section, we discuss simplicial d-complexes generated by subdividing a d-dimensional hypercube.

4.1. Kuhn Subdivisions

Assume, without loss of generality, that a unit d-cube h is embedded in a subspace $[0,1]^d$ of $\mathbb{R}^n$. Let 0^d and 1^d denote a pair of opposite vertices forming a diagonal $\psi = (0^d, 1^d)$ from the lower left corner of h to its upper right corner. Also, let e_0 denote 0^d and e_i the i-th unit vector in $\mathbb{R}^d$, e.g. $e_1 = (1,0,0,\ldots)$, $e_2 = (0,1,0,0,\ldots)$.

We refer to the d-simplex with vertices

$$v_i = \sum_{j \leq i} e_j$$

as the *base simplex* S_0. For example, when $d = 3$, S_0 has vertices $(0,0,0)$, $(1,0,0)$, $(1,1,0)$ and $(1,1,1)$.

Let π be a permutation of the integers $\{0,1,\ldots,d-1\}$ and let $\mathbf{v}' = \pi\mathbf{v}$ indicate the application of permutation π to the coordinates of vertex $\mathbf{v}$. For example, if $\mathbf{v} = (1, \frac{1}{2}, 0)$ and $\pi = \{2,0,1\}$, then $\mathbf{v}' = \pi\mathbf{v} = (0,1,\frac{1}{2})$. Finally, let πS denote the application of π to each vertex of simplex S.

Then, a simplicial decomposition of h (due to Freudenthal [Fre42] and popularized by Kuhn [Kuh60]) into $d!$ cells (and their lower dimensional faces), which we denote as $\mathcal{K}(h)$, is defined by the mapping of each distinct d-permutation π onto the vertices of the base simplex S_0, i.e.,

$$\mathcal{K}(h) = \{\pi\, S_0 | \pi \text{ is a permutation of } \{0,1,\ldots,d-1\}\}. \quad (1)$$

Since coordinate permutations do not modify 0^d or 1^d, every d-simplex in $\mathcal{K}(h)$ contains diagonal ψ of h. Also, the i^{th} vertex v_i of any cell $\sigma \in \mathcal{K}(h)$ contains $(d-i)$ coordinates of value zero and i coordinates of value one. Thus, the edge $(0^d, v_i)$ of σ is a diagonal of an i-face of h, and edge $(v_i, 1^d)$ is a diagonal of a $(d-i)$-face of h.

Kuhn subdivisions can be generalized to any d-cube h' with diagonal $\psi' = (v_1, v_2)$ by an affine mapping from the vertices of ψ' to $(0^d, 1^d)$.

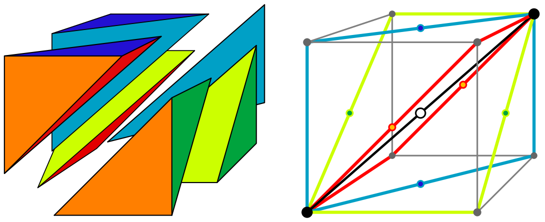


Figure 4: *Decomposition of a 3-cube into* $3! = 6$ *simplices.*

An interesting property of Kuhn subdivisions, which will be of use later and we prove now, is that it provides a Kuhn-subdivision to all faces of the initial hypercube.

Theorem 4.1 Let $\mathcal{K}(h)$ be the simplicial decomposition of a d-cube h, and h_i an i-face of h. Then $\mathcal{K}(h_i) = h_i \cap \mathcal{K}(h)$ is an i-dimensional Kuhn subdivision of the domain of h_i.

Proof If $i = 0$ then $\mathcal{K}(h_i)$ is trivially a Kuhn subdivision. Assume, without loss of generality, that h is a unit d-cube with diagonal $\psi = (0^d, 1^d)$. We show that the $(d-1)$-faces of h are Kuhn-subdivided. Since d was arbitrary, the proof for the remaining i-faces follows by induction.
Consider the simplicial $(d-1)$-complex Σ obtained by removing vertex 1^d from every simplex $\sigma \in \mathcal{K}(h)$. All cells of Σ are defined by d vertices and are thus $(d-1)$-simplices. In fact, since ψ was the only diagonal of h, we can decompose Σ into d subcomplexes, each containing simplices within a $(d-1)$-dimensional axis aligned hyperplane of $\mathbb{R}^d$. In the n^{th} such subcomplex $\Sigma_n \subset \Sigma$ this hyperplane can be defined by the equation $x_n = 0$. Thus, the d vertices of a cell $\sigma \in \Sigma_j$ are of the form

$$v'_k = \pi(\sum_{j=0}^{n-1} e_j + \sum_{j=n+1}^{i} e_j). \quad (2)$$

By projecting Σ_n onto the $(d-1)$-dimensional subspace of $\mathbb{R}^d$ that excludes coordinate x_n, we obtain the $(d-1)!$ cells of a Kuhn subdivided $(d-1)$-cube (compare Equation 2 to Equation 1).
Similarly, the simplicial complex defined by removing vertex 0^d gives us Kuhn subdivisions for the d remaining $(d-1)$-faces of h, where each hyperplane is of the form $x_n = 1$. □

4.2. Maubach Bisection

To bisect a d-simplex σ along one of its edges $\mathbf{e}$, we insert a new vertex $\mathbf{v}_m$ at the midpoint of $\mathbf{e}$ and bisect σ by the hyperplane defined by $\mathbf{v}_m$ and the $(d-1)$ vertices of σ that are not adjacent to $\mathbf{e}$ (see Figure 5). This creates two new d-simplices, covering the same domain as σ, each containing vertex $\mathbf{v}_m$ and one (but not both) of the endpoints of $\mathbf{e}$.

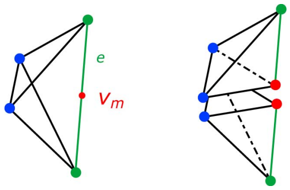


Figure 5: *Bisection of a 3-simplex along an edge* ***e****.*

Maubach's bisection scheme [Mau95] specifies the *bisection edge* for any cell σ in an initial simplicial d-complex Σ or generated by repeated application of the bisection rule to the cells of Σ. It depends only on the ordering of the vertices of σ, and on the subdivision level ℓ_σ of σ, which is initialized to zero for any cell in the original complex Σ. Given a d-simplex

$$\sigma = (v_0, v_1, \dots, v_{k-1}, \mathbf{v}_k, v_{k+1}, \dots, v_d),$$

where $k = d - (\ell_\sigma \bmod d)$, the bisection edge is defined by vertices v_0 and v_k, and its midpoint is $\mathbf{v}_m = (v_0 + v_k)/2$. The two cells generated by the bisection rule have vertices

$$\sigma_0 = (v_0, v_1, \dots, v_{k-1}, \mathbf{v}_m, v_{k+1}, \dots, v_d)$$
$$\sigma_1 = (v_1, v_2, \dots, v_k, \mathbf{v}_m, v_{k+1}, \dots, v_d),$$

and the level of these simplices is incremented, e.g.

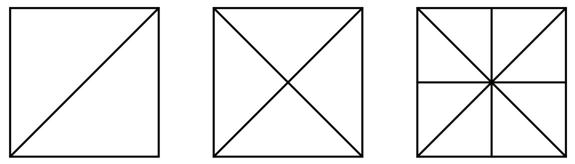
$$\ell_{\sigma_0} = \ell_{\sigma_1} = (\ell_\sigma + 1).$$

Maubach proves that when his bisection scheme is applied to a Kuhn-subdivided d-cube h whose simplex vertices are ordered as in Section 4.1, the generated d-simplices belong to at most d similarity classes [Mau95]. Recall that simplices are similar if there is an affine mapping consisting of only uniform scaling, reflection, rotation and translation between them. Since coordinate permutations are rigid mappings, the $d!$ cells in $\mathcal{K}(h)$ belong to the same similarity class. Furthermore, all cells at level $(\ell \bmod d)$ belong to the same similarity class.

We denote the cells of $\mathcal{K}(h)$ as *class*-0 simplices, and in general, a d-simplex σ as a *class-i* simplex if $i = (\ell_\sigma \bmod d)$. Observe that the bisection edge of a class-i simplex is aligned with the diagonal of a $(d-i)$-cube.

4.3. Fully Subdivided Hypercube

Consider the set of simplicial d-complexes $\mathcal{M}_i(h)$ generated through repeated application of Maubach's bisection to the cells of $\mathcal{K}(h)$, where i denotes the level of the d-simplices in $\mathcal{M}_i(h)$ and $\mathcal{M}_0(h)$ contains the $d!$ cells of $\mathcal{K}(h)$. Since each d-simplex in $\mathcal{M}_0(h)$ is replaced by two d-simplices in $\mathcal{M}_1(h)$, $\mathcal{M}_1(h)$ contains $2 * d!$ cells. $\mathcal{M}_d(h)$ thus contains the $2^d * d!$ cells resulting from d such iterations, each of which is of class-0 and is a factor of two smaller than those of $\mathcal{M}_0(h)$. We call $\mathcal{M}_d(h)$ a *Fully Subdivided Hypercube*, which we denote as $\mathcal{F}(h)$.



(a) $\mathcal{M}_0(h) = \mathcal{K}(h)$ (b) $\mathcal{M}_1(h)$ (c) $\mathcal{M}_2(h) = \mathcal{F}(h)$

Figure 6: *Three consecutive $\mathcal{M}_i(h)$ simplicial complexes in 2D. (a) $\mathcal{M}_0(h)$ is equivalent to $\mathcal{K}(h)$ and has $2! = 2$ triangles. (b) $\mathcal{M}_1(h)$ has 4 triangles. (c) $\mathcal{M}_2(h)$ is equivalent to $\mathcal{F}(h)$ and has $4!! = 8$ triangles.*

We simplify the notation by observing that $2^d * d!$ can be defined in terms of the *double factorial* function [Mes48] as

$$2^d * d! = (2d)!!$$

where the double factorial $n!!$ is equal to 1 if $n \in \{0,1\}$ and $n*(n-2)!!$ otherwise. The values of $(2i)!!$ for $i = 1 \dots 4$ are 2, 8, 48, 384.

4.3.1. Properties of $\mathcal{F}(h)$

Let h be a d-cube with midpoint $\mathbf{v}_c$. Let $\mathcal{K}(h)$ be the Kuhn subdivision of h along diagonal ψ, and $\mathcal{F}(h)$ its corresponding fully subdivided hypercube.

Lemma 4.1 For all cells $\sigma \in \mathcal{F}(h)$, $\mathbf{v}_c$ is a vertex of σ. Furthermore, the bisection edge of σ is defined by $\mathbf{v}_c$ and one of the 2^d vertices of h.

Proof This follows from the generation of $\mathcal{F}(h)$ in terms of the Maubach complexes $\mathcal{M}_i(h)$ starting with $\mathcal{M}_0(h) = \mathcal{K}(h)$. After the first application of the bisection rule to the cells of $\mathcal{M}_0(h)$, all cells σ of $\mathcal{M}_1(h)$ have the midpoint $\mathbf{v}_c$ of ψ as their d^{th} vertex. Since none of the next $d-1$ applications of the Maubach bisection scheme modify the d^{th} vertex, all cells of $\mathcal{M}_d(h) = \mathcal{F}(h)$ contain $\mathbf{v}_c$.
Since all cells in $\mathcal{F}(h)$ are class-0, the bisection edge is determined by the first and last vertices of σ. As described above, the last vertex of σ is $\mathbf{v}_c$. Since σ is a class-0 simplex, its bisection edge must be the diagonal of a d-cube. The only edges of $\mathcal{F}(h)$ that satisfy this constraint are those between $\mathbf{v}_c$ and a vertex of h. □

Recall that for a d-cube h, $\mathcal{K}(h)$ contains $d!$ class-0 simplices. An alternate interpretation of $\mathcal{F}(h)$ is as a collection of Kuhn-subdivided subcubes covering the domain of h and centered at the midpoint of h.

Corollary 4.2 $\mathcal{F}(h)$ consists of 2^d Kuhn-subdivided d-cubes covering h and with side length half that of h. Thus, each of the 2^d subcubes contributes $d!$ cells to $\mathcal{F}(h)$ for a total of $2^d d! = (2d)!!$ cells.

Theorem 4.3 Each i-face h_i of a fully subdivided d-cube $\mathcal{F}(h)$ is a fully subdivided $(d-1)$-cube $\mathcal{F}(h_i)$.

Proof Consider the simplicial $(d-1)$-complex obtained by removing vertex $\mathbf{v}_c$ from each cell $\sigma \in \mathcal{F}(h)$. Since each j-face h_j of a Kuhn-subdivided cube is a Kuhn-subdivided j-cube, the removal of $\mathbf{v}_c$ from a Kuhn-subdivided subcube within $\mathcal{F}(h)$ adds the $(d-1)!$ cells of a Kuhn-subdivided $(d-1)$-cube to each of the d facets of h on which it is adjacent (see the proof of Theorem 4.1 for details). Since there are 2^{d-1} subcubes adjacent with each facet h_i of h, h_i contains $2^{d-1}(d-1)! = (2(d-1))!!$ cells. Since these $(d-1)$-simplices are from a Kuhn-subdivided $(d-1)$-simplicial complex, they are all class-0 cells of dimension $d-1$. Further, all cells of $\mathcal{F}(h_i)$ contain the midpoint of h_i coinciding with the midpoint of the 2^{d-1} subcubes adjacent to h_i. □

Corollary 4.4 Since all i-faces of $\mathcal{F}(h)$ contain their midpoint, $\mathcal{F}(h)$ contains $\sum 2^{d-i}\binom{d}{i} = 3^d$ vertices.

This enables us to define a fully subdivided d-cube as a cross-complex of its boundary faces and $\mathbf{v}_c$.

Corollary 4.5 Let h_i denote one of the $2*d$ facets of h. A fully subdivided d-cube $\mathcal{F}(h)$ with midpoint $\mathbf{v}_c$ can be decomposed as a cross-complex of the $(d-1)$-simplices from each $\mathcal{F}(h_i)$ and the singleton simplicial complex $\{\mathbf{v}_c\}$. E.g.

$$\mathcal{F}(h) = \{\bigcup \mathcal{F}(h_i) \otimes \{\mathbf{v}_c\} | h_i \text{ is a facet of } h\}.$$

This provides a motivation for the double factorial notation. Each of the $2d$ facets h_i contributes $(2(d-1))!!$ cells of $\mathcal{F}(h_i) \otimes \mathbf{v}_c$, so $\mathcal{F}(h)$ contains $2d*(2(d-1))!! = (2d)!!$ cells.

We are also interested in the simplicial complex defined by the simplices on the $2*d$ facets h_i on the boundary of $\mathcal{F}(h)$, which we call a *fully-subdivided i-cube boundary* and denote as $\mathcal{B}_F(h)$. Thus, $\mathcal{B}_F(h) = \cup \mathcal{F}(h_i)$ and is defined by $(2d)!!$ cells of dimension $(d-1)$. Each such cell corresponds to a cell of $\mathcal{F}(h)$ where the vertex at the center of $\mathcal{F}(h)$ has been removed. Figure 7 shows examples of fully subdivided i-cube boundaries for $i = 1,2,3$, and highlights the center vertex of each facet of $\mathcal{B}_F(h)$.

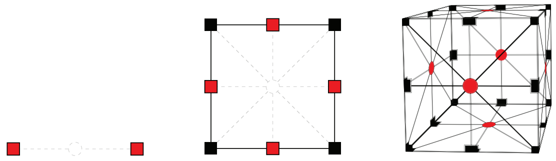

(a) 1-cube (b) 2-cube (c) 3-cube

Figure 7: *Fully subdivided i-cube boundary $\mathcal{B}_F$ for (a) 1-cube (b) 2-cube and (c) 3-cube.*

5. Simplex Hierarchies

Consider a d-dimensional hypercubic domain h subdivided according to the Kuhn subdivision rule into $d!$ cells. A hierarchical relationship exists between the simplices generated by Maubach's scheme. The two d-simplices σ_1 and σ_2 generated through a bisection operation on simplex σ are the *children* of σ, and conversely, σ is the parent of σ_1 and σ_2.

We can represent this relationship as a *simplex tree*, a binary tree whose root is a d-simplex from $\mathcal{K}(h)$. Furthermore, the entire simplicial complex can be represented as a forest of $d!$ simplex trees whose roots are the class-0 cells of $\mathcal{K}(h)$. All simplices in level ℓ of a simplex tree are congruent, and the class, i, of a d-simplex σ at level ℓ_σ is $(\ell_\sigma \bmod d)$. Further, the bisection edge of σ is the diagonal ψ of a $(d-i)$-cube whose center is the new vertex.

The fundamental operation performed in applications on such a forest of simplex trees is the extraction of adaptive simplicial complexes. This is the basis for a well-known operation in computer graphics and scientific data visualization, called *selective refinement*. Let σ be a cell (node) of a simplex tree τ and σ_1 and σ_2 its children. Since σ_1 and σ_2 cover the same domain as σ the hierarchical relationship between cells of τ defines a nested simplicial mesh. Thus, since $\mathcal{K}(h)$ is a simplicial decomposition of the domain, repeated application of the simplex bisection operation to cells in the forest always provides a non-overlapping simplicial decomposition of the domain.

However, due to the local nature of the simplex bisection rule, it does not, in general, generate valid simplicial complexes. Consider the faces of a cell σ in a simplicial complex Σ generated according to the simplex bisection rule. Since Σ is a simplicial complex, all faces adjacent to those of σ intersect only at common faces. However, after σ is bisected along edge ψ, faces that were previously adjacent to ψ are no longer conforming.

Thus, the bisection rule requires an additional constraint to ensure the generation of valid simplicial complexes. Namely, the level of all cells sharing bisection edge ψ of a cell σ must be equal to that of σ before the bisection and that all such cells are bisected concurrently with σ.

To satisfy this constraint, we must first find the set of neighbors of cell σ along bisection edge ψ. The so-called *neighbor-finding* operation, finds all simplices sharing $(d-1)$-faces that have ψ as a common face. Neighbor finding can be accomplished by storing pointers to each of the $d+1$ neighboring cells [Mau95] or symbolically by manipulating *location codes* that uniquely identify each cell in the forest [Heb94, Mau96, EKT01, LDS04, AMM07]. Symbolic neighbor-finding enables a pointerless representation for cells in the forest, thus enabling each neighbor-finding operation to be carried out in $O(1)$ time. However, since each neighbor must be found, this operation must be performed $O(|\mathit{Neighbors}(\sigma)|)$ times.

6. Hierarchy of Diamonds

We have seen that conforming updates to a simplicial complex generated using Maubach's bisection scheme are related to the set of d-simplices surrounding a common bisection edge. An alternative to simplex bisection consists of clustering d-simplices surrounding a common bisection edge into a new primitive, called a *diamond* [Pas02, GDL*02] and considering the hierarchical relationships between diamonds rather than those between simplices.

A *diamond* is the set of all d-simplices with a common bisection edge, called the *spine* of the diamond. Since all d-simplices within a diamond are congruent, there are d similarity classes of diamonds and a diamond whose d-simplices belong to class-i is referred to as an i-diamond. Its spine is the diagonal of a $(d-i)$-cube.

A diamond δ is subdivided by bisecting all of its d-simplices using Maubach's bisection scheme. Thus, subdivision doubles the number of cells within δ and we denote its corresponding *subdivided diamond* as δ_s.

Let σ_i be an i-face of a d-simplex in diamond δ. Then, σ_i is said to be on the *interior* of δ if all simplices σ' that contain σ_i as an i-face belong to δ. Otherwise, σ_i is said to be on the *boundary* of δ. An important property of the diamond subdivision is that all changes occur within the interior of the subdividing diamond δ. Consequently, the faces on the boundary of δ are unaffected by the subdivision operation. The local effect of the subdivision of a diamond δ is to (a) remove its spine (b) add a vertex $\mathbf{v}_c$ at the midpoint of its spine, which we refer to as its *central vertex* and (c) add edges from $\mathbf{v}_c$ to each vertex $\mathbf{v}$ of δ.

The hierarchical relationship between the d-simplices generated by Maubach's bisection scheme defines a direct dependency relation on the diamonds. A diamond δ_c is a *child* of a diamond δ_p, and conversely δ_p is a *parent* of δ_c, if δ_c contains at least one d-simplex generated by the bisection of a d-simplex in δ_p.

In contrast with the simplex hierarchy, which can be represented as a forest of binary trees, the diamond dependency relationship can be described as a *Directed Acyclic Graph (DAG)* Δ. The root of Δ is the 0-diamond subdividing the hypercubic domain h (e.g. all simplices of $\mathcal{K}(h)$). The nodes of Δ are the subdividing diamonds, e.g. the pairs (δ,δ_s) containing a diamond δ and its corresponding subdivided diamond δ_s covering the same domain. Finally, the arcs of Δ are defined by the dependency relationship among the diamonds.

7. Properties of a hierarchy of diamonds

We now focus on the structure of an arbitrary i-diamond δ in d-dimensions. This leads to the derivation of closed-form equations for the number of simplices and vertices in δ as well as the number and location of its parents and children.

Theorem 7.1 An i-diamond δ in dimension d is the cross-complex defined by $\mathcal{K}(h_k)$, a Kuhn subdivided $(d-i)$-cube, h_k, and $\mathcal{B}_F(h_i)$, a fully subdivided i-cube boundary, h_i, i.e.,

$$\delta = \{\mathcal{K}(h_k) \otimes \mathcal{B}_F(h_i)\}$$

such that h_k and the facets of h_i are in affinely independent subspaces of $\mathbb{R}^d$, and the center of h_k and of h_i coincide.

Proof Consider the vertices of an arbitrary d-simplex $\sigma \in \delta$

$$\sigma = (\underbrace{\mathbf{v}_0, v_1, \ldots, v_{k-1}, \mathbf{v}_k,}_{(d-i+1)\text{ vertices}}\ \underbrace{v_{p_1}, v_{p_2}, \ldots, v_{p_j}, \ldots, v_d}_{i\text{ vertices}}),$$

where $k = d-i$. Since δ is defined by its spine $\psi = (\mathbf{v}_0, \mathbf{v}_k)$, which is the diagonal of a $(d-i)$-cube, the vertices in position 0 in all d-simplices of δ are identical, and similarly for the vertices in position $k = d-i$. Furthermore, the midpoint $\mathbf{v}_c = \frac{1}{2}(\mathbf{v}_0 + \mathbf{v}_k)$ of ψ is the central vertex of δ and is the vertex that will be inserted in position k for all d-simplices generated during the subdivision of δ.

Due to the use of Maubach's bisection scheme, vertex v_{p_1} at position $(d-i+1)$ of σ, where $i > 0$, is the center of a $(d-i+1)$-cube h_p. Also, v_{p_1} is the central vertex of the diamond δ_p whose subdivision generated σ. Similarly, for $j \leq i$, the vertex v_{p_j} at position $(d-i+j)$ of σ is the center of a $(d-i+j)$-cube and v_{p_j} is the central vertex of the level-j ancestor diamond of δ.

The proof is split into two parts. We first show the $(d-i)$-dimensional Kuhn-subdivided component of δ, $\mathcal{K}(h_k)$, whose vertices are in the initial $(d-i+1)$ positions of any d-simplex $\sigma \in \delta$. Next, we show the fully subdivided i-cube boundary component, $\mathcal{B}_F(h_i)$, whose vertices are in the final i positions of σ. Since σ is a d-simplex, all of its vertices must be in affinely independent subspaces of $\mathbb{R}^d$, and thus σ is a cross-simplex of a $(d-i)$-simplex from $\mathcal{K}(h_k)$ and an $(i-1)$-simplex from $\mathcal{B}_F(h_i)$.

Kuhn component: Consider the set of d-simplices within δ whose final i vertices are the same, i.e. if σ_a and σ_b are two such d-simplices, then the vertex at position $(k+j)$ of σ_a is equal to the vertex at position $(k+j)$ of σ_b, for $0 < j \leq i$. Since we use Maubach's ordering for the simplices, the subspace of $\mathbb{R}^d$ spanned by these simplices is a $(d-i)$-cube h_k, whose diagonal is ψ. Furthermore, since our hierarchy began with a Kuhn subdivision of h and because of Theorem 4.1, all i-faces of a Kuhn subdivided d-cube are Kuhn subdivided, these simplices comprise a Kuhn subdivision of h_k, i.e. $\mathcal{K}(h_k)$, and there are $(d-i)!$ such simplices.

Fully subdivided component: This proof involves a grid that is *dual* to the one we have been using (i.e. the *primal grid*). A vertex of the dual grid corresponds to the center of a d-cube of the primal grid, and, in general, a j-face of the dual grid corresponds to a $(d-j)$-face of the primal grid. Observe that the vertices of this dual grid are offset from those of the primal grid by one half unit in each axis-aligned direction.

Recall that, on the primal grid, the vertex v_{p_j} at position $(d-i+j)$ of σ is the center of a $(d-i+j)$-cube whose center coincides with the central vertex of a parent of δ. Then, on the dual grid, vertex v_{p_j} of σ is the center of an $(i-j)$-cube, $1 \leq j \leq i$. Note that the central vertex of δ (which is not a vertex of δ until after it subdivides) is the center of an i-cube, h_i on the dual grid.

In the following, consider the collection of d-simplices within δ whose initial $(d-i+1)$ vertices are the same, i.e. if σ_a and σ_b are two such d-simplices, then the vertex at position j of σ_a is equal to the vertex at position j of σ_b, for $0 \leq j \leq (d-i)$. We can thus project these d-simplices into an $(i-1)$-dimensional subspace of $\mathbb{R}^d$.

Our claim, which we prove through induction on i, is that these $(i-1)$-simplices decompose the boundary of a fully subdivided i-cube h_i, e.g. $\mathcal{B}_F(h_i)$. In the base case, $i=0$, and $\mathcal{B}_F(h_i)$ is empty and is therefore trivially the boundary of a fully subdivided 0-cube.

For the inductive step, assume that in an $(i-1)$-diamond δ_p, the final $(i-1)$ vertices of each simplex correspond to the boundary of a fully-subdivided $(i-1)$-cube $\mathcal{B}_F(h_p)$, whose simplices therefore have dimension $i-2$. When δ_p is subdivided, its central vertex $\mathbf{v}'$ coinciding with the center of h_p is inserted. In addition, edges are created from $\mathbf{v}'$ to all vertices of δ_p, including the vertices of $\mathcal{B}_F(h_p)$. This increases the dimension of each of $\mathcal{B}_F(h_p)$'s simplices and generates $\mathcal{F}(h_p)$ (recall from Section 4.3 that $\mathcal{F}(h_p)$ is defined as the cross-complex of $\mathcal{B}_F(h_p)$ and the vertex at its center). All simplices generated during this subdivision contain vertex $\mathbf{v}'$ in position $d-i$. The final i vertices of each such d-simplex defines an $(i-1)$-simplex, and together these $(i-1)$-simplices form the fully subdivided $(i-1)$-cube $\mathcal{F}(h_p)$.

Now, consider the subset of these simplices that get contributed to an i-diamond δ_c that is a child of δ_p. These are characterized by having the same spine vertices (e.g. $\mathbf{v}_0$ and $\mathbf{v}_k$). Since δ_c is an i-diamond its central vertex is the midpoint of an i-cube h_i in the dual grid. Among the $2*i$ facets of h_i, one is the fully subdivided $(i-1)$-cube $\mathcal{F}(h_p)$. By symmetry, each of the other facets are subdivided similarly, and, thus, the boundary of h_i is subdivided as a fully subdivided i-cube, i.e. $\mathcal{B}_F(h_i)$. □

This decomposition of Theorem 7.1 suggests the following closed-form equations for the number of d-simplices, vertices, parents and children of any diamond δ. Let δ be an i-diamond of dimension d, $\mathcal{K}(h_k)$ be the $(d-i)$-dimensional Kuhn-subdivided component of δ and $\mathcal{B}_F(h_i)$ be the fully subdivided i-cube boundary component of δ.

Simplices. The number of d-simplices in an i-diamond is $(d-i)!(2i)!!$ This follows from the fact that δ is defined by the cross complex of $\mathcal{K}(h_k)$ which contains $(d-i)!$ cells and $\mathcal{B}_F(h_i)$ which contains $(2i)!!$ cells. The d-simplices of δ are cross simplices of those from $\mathcal{K}(h_k)$ and $\mathcal{B}_F(h_i)$. Thus the number of neighbors required in the neighbor finding operation is $O(d!)$.

Vertices. The number of vertices in an i-diamond is $(2^{d-i} + 3^i - 1)$. Since $\mathcal{K}(h_k)$ contains 2^{d-i} vertices, and $\mathcal{B}_F(h_i)$ contains $3^i - 1$ vertices and they are both in affinely-independent subspaces, the number of vertices in δ is just their sum.

Children. The number of children of an i-diamond is $2*(d-i)$ if $i < (d-1)$ and 2^d if $i = (d-1)$. Due to the Maubach subdivision, the spines of children of an i-diamond, $i < (d-1)$, coincide with diagonals of the $2*(d-i)$ facets of h_k. When $i = (d-1)$, h_k is a 1-cube (an edge of a cube), and h_i is a $(d-1)$-cube. The spine vertices of δ's children are located at positions 0 and d of each d-simplex, corresponding to one of the two vertices of h_k and one of the 2^{d-1} vertices of h_i. There are thus, 2^d such children.

Parents. The number of parents of an i-diamond is $2*i$ if $i > 0$ and d if $i = 0$. The central vertex of each parent of an i-diamond, $i > 0$, coincides with the midpoint of one of the $2*i$ facets of h_i. When $i = 0$, h_i is a 0-cube coinciding with the central vertex of δ. Let σ denote one d-simplex of δ and let (v_0, v_x) denote the spine vertices of its parents, where v_0 is the vertex at position 0 of σ and v_x the spine vertex of the parent δ_p that generated σ. Then, since δ_p was a $(d-1)$-diamond, its spine was aligned with a coordinate axis of $\mathbb{R}^d$, thus, δ has d parents. Furthermore, if $\psi = (v_0, v_d)$ is the spine of δ then let $\mathbf{v} = v_d - v_0$ be the difference between these vertices. The spine of the j^{th} parent of δ is defined by v_0 and $v_x = v_0 + \mathbf{v} \cdot e_j$ (where $\cdot$ indicates the dot product).

8. Encoding diamonds

In this section, we generalize the diamond representation of [WD08] to yield an efficient encoding for a d-dimensional hierarchy of diamonds Δ over a hypercubic regular domain of size $(2^N+1)^d$, where N is the maximum level of resolution LEVEL_{Max}. The vertices of diamonds in Δ have integer coordinates in the range $[0 \ldots 2^N]$.

A diamond δ is defined by its spine ψ, or alternatively, by the unique midpoint, $\mathbf{v}_c$, of its spine. Due to the regularity of the vertex distribution as well as the updates, all geometric and hierarchical relationships can be derived directly from the binary representation of the coordinates $(x_1, x_2, \ldots, x_d)$ of $\mathbf{v}_c$ using efficient hardware bit shifting. Let

$$\mathbf{v}_c = \begin{bmatrix} \mathbf{x}_1 & = & x_1^1\, x_1^2 \ldots x_1^n \;\; \tau_1^1\, \tau_1^2 \;\; 00\ldots0 \\ \mathbf{x}_2 & = & x_2^1\, x_2^2 \ldots x_2^n \;\; \tau_2^1\, \tau_2^2 \;\; 00\ldots0 \\ & \vdots & \\ \mathbf{x}_d & = & x_d^1\, x_d^2 \ldots x_d^n \;\; \underbrace{\tau_d^1\, \tau_d^2}_{\tau} \;\; \underbrace{00\ldots0}_{\gamma} \end{bmatrix}^T \tag{3}$$

be the binary representation of the central vertex. Our encoding depends on two quantities which can be extracted from this representation: the *scale* γ and the *type* τ of δ.

Let $\text{TRAILING}(x_i)$ denote the number of trailing zeros in the binary representation of a coordinate x_i. Then, the minimum of the number of trailing zeros among each of the d coordinates of $\mathbf{v}_c$ encodes the *scale* γ of δ, e.g.

$$\gamma = \min_{i \leq d}(\text{TRAILING}(x_i)).$$

Thus, for a diamond at scale γ, the rightmost γ bits in any coordinate of $\mathbf{v}_c$ are zero, but at least one of the bits in position τ^2 is nonzero. We define the *depth* of an i-diamond δ in terms of the scale as $\text{DEPTH}_\delta = \text{LEVEL}_{Max} - \gamma$. A diamond's depth

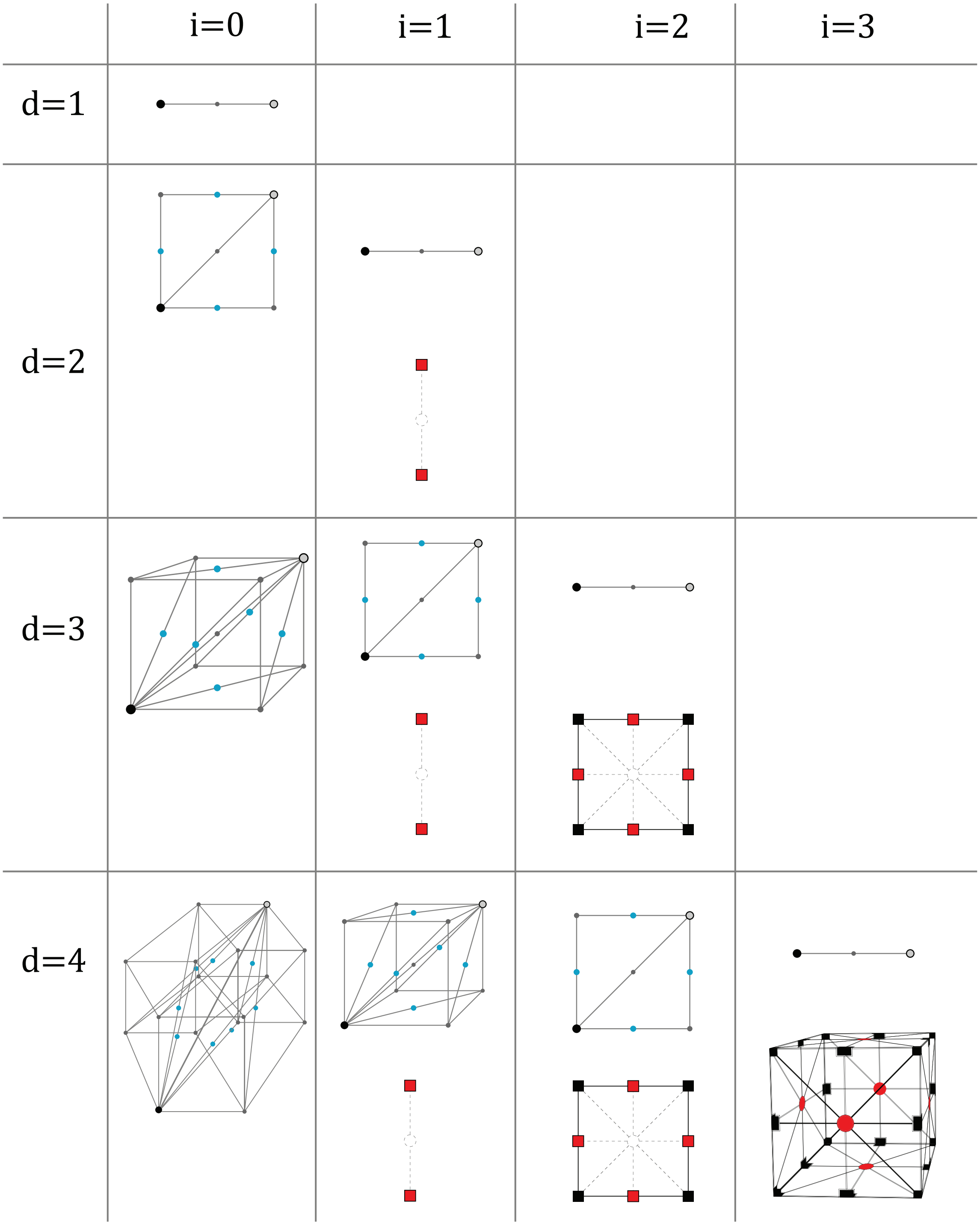


Figure 8: *An i-diamond δ in dimension d is the cross-complex of $\mathcal{K}(h_k)$, a Kuhn subdivided $(d-i)$-cube, h_k (top cube in each cell) and $\mathcal{B}_F(h_i)$, the boundary of a fully subdivided i-cube, h_i (bottom cube in each cell). For $i < d-1$, the central vertices of children of δ are located at the midpoints of each $(d-i-1)$-face of h_k (blue vertices). For $i > 0$, the central vertices of parents of δ are located at the midpoints of each $(i-1)$-face of h_i (red vertices). An example of $i = 2$ and $d = 3$ is illustrated in Figure 3(b).*

corresponds to the number of i-diamond ancestors it has in the DAG on a path from the root to δ. The level ℓ_σ of a cell σ in an i-diamond δ is thus $\mathrm{DEPTH}_\delta * d + i$.

The two bits at position $\gamma+1$ and $\gamma+2$ of each coordinate x_i, which we denote as τ_i^2 and τ_i^1, respectively encode the *type* τ of δ. The *class* of δ is encoded within the bits τ^2 of τ as the number of zeros, e.g. $d - \sum \tau_i^2$. The final $n = \mathrm{LEVEL}_{Max} - (\gamma+2)$ bits in each dimension are only used in this representation to distinguish between diamonds.

The oriented *direction* of δ's spine can be extracted from τ using the following encoding. First, initialize the sign variable s to $+1$. Component u_i of the direction vector $\vec{u}$ is then:

$$u_i = \begin{cases} 1, & \text{if } \tau_i^1 = 0 \text{ and } \tau_i^2 = 1 \\ -1, & \text{if } \tau_i^1 = 1 \text{ and } \tau_i^2 = 1 \\ 0, & \text{if } \tau_i^1 = 0 \text{ and } \tau_i^2 = 0 \\ 0, & \text{if } \tau_i^1 = 1 \text{ and } \tau_i^2 = 0. \end{cases}$$

In the fourth case, where $\tau_i^1 = 1$ and $\tau_i^2 = 0$, we must also multiply the sign s by -1. The orientation of spine ψ is then $s * \vec{u}$.

The geometric and hierarchical *components* of δ can be computed as scaled offsets from $\mathbf{v}_c$. The unscaled offsets are d-vectors $\vec{f}$ such that $f_i \in \{-1, 0, 1\}$. Thus, a component of diamond δ, at scale γ, whose center is p and whose offset from $\mathbf{v}_c$ is $\vec{f}$ can be computed as:

$$p = \mathbf{v}_c + 2^\gamma * \vec{f}. \tag{4}$$

We now discuss the offset vectors to the elements of an i-diamond δ at scale γ.

Kuhn-subdivided component. Since the class i of δ is encoded in the number of zeros in the rightmost bits of τ, the subspace of $\mathbb{R}^d$ spanned by the Kuhn subdivided $(d-i)$-cube $\mathcal{K}(h_k)$ are those defined by the $(d-i)$ coordinates of τ_i^2 with value 1.

Specifically, δ's spine $\psi = (v_0, v_k)$ can be calculated using the oriented spine direction $\vec{u}$, where v_0 has offset vector $\vec{f} = -\vec{u}$ and v_k has offset vector $\vec{f} = \vec{u}$. Their coordinates can be obtained by plugging $\vec{f}$ into Equation 4. The remaining vertices and cells can be found through an affine mapping to the canonical subdivision of Section 4.1, or, if d is reasonably small, through precomputed lookup tables [GDL*02, WD08].

Since the $2(d-i)$ children of δ are located at the facets of h_k (for $i < (d-1)$), offsets to their central vertices can be computed as $\vec{f} = \pm e_j$, in all coordinates that $\tau_j^2 = 1$. For $i = (d-1)$, the 2^d children are at offsets $\vec{f} = (\pm1, \pm1, \ldots, \pm1)$, and at scale $\gamma+1$.

Fully-subdivided component. The i-dimensional subspace spanned by $\mathcal{B}_F(h_i)$ is along the coordinates in which $\tau_j^2 = 0$. Similarly, for $i > 0$, offsets to the parents of δ are $\vec{f} = \pm e_j$, in all coordinates that $\tau_j^2 = 0$. For $i = 0$, we can use the oriented spine direction to find vertices v_0 and v_d of the spine. Then, as in Section 7, $\mathbf{v} = v_d - v_0$ and the d parents are located at offset $\vec{f} = -u + \mathbf{v} \cdot e_j$.

Since the vertices and d-simplices of $\mathcal{B}_F(h_i)$ are defined along all directions spanned by h_i, they can be found by incrementally traversing in a direction within h_i orthogonal to the directions that have already been traversed. That is, since the ancestor of a diamond at the center of an i-cube of $\mathcal{B}_F(h_i)$ is the center of one of its facets, the traversal is only along a single dimension.

Figure 8 illustrates the components of all diamond classes up to dimension $d = 4$. Each cell corresponds to a diamond δ of class i (columns) in dimension d (rows). The top hypercube in each cell is a Kuhn subdivided $(d-i)$-cube containing the oriented spine, whose vertices are colored black and gray, respectively, and the children, whose central vertices (blue) are located at the center of its facets, at an offset $\vec{f} = \pm e_j$, in all coordinates that $\tau_j^2 = 1$. The bottom hypercube in each cell is the boundary of a fully subdivided i-cube. The central vertices of the parents (red) of δ are located at the center of its facets at offset $\vec{f} = \pm e_j$, in all coordinates that $\tau_j^2 = 0$. The two hypercubes intersect at their midpoints.

9. Encoding simplicial complexes

The encoding presented in Section 8 leads to an efficient pointerless representation for a simplicial complex Σ extracted from a hierarchy of diamonds. Σ can be encoded as a collection of diamonds, each of which contains a set of d-simplices, such that, the collection of simplices from all diamonds in Σ forms a simplicial complex covering the domain h. Since we can reconstruct the location of all vertices, simplices, parents and children of a diamond δ from the coordinates of its central vertex, each diamond can be entirely indexed by the d coordinates of its central vertex.

In general, not all d-simplices of a diamond will belong to the complex Σ, and thus each diamond δ requires some bookkeeping to track the set of its d-simplices belonging to the complex Σ. We observe that each d-simplex in a diamond δ was created during the subdivision of a single parent of δ. Since an i-diamond, $i > 0$, has $(d-i)!(2i)!!$ simplices and only $2i$ parents, we observe that these d-simplices are contributed to δ in clusters of size $(d-i)!(2(i-1))!!$. Similarly, the $d!$ d-simplices of a 0-diamond arrive in clusters of size $(d-1)!$ from each of its d parents. Thus, a single bit is sufficient to track each cluster of d-simplices within δ, and, consequently, $2*d$ bits are sufficient to track all $(d-i)!(2i)!!$ d-simplices of diamond $\delta \in \Sigma$.

Recall that the bisection rule for d-simplices requires that all bisection-edge neighbors are present in the mesh. The corresponding condition for conforming updates to a diamond δ is for δ to contain all of its d-simplices. This occurs when all parents of δ have subdivided. Thus, the $O(d)$

bookkeeping bits of a diamond can also be used to cache the subdivision status of each parent of δ.

As a consequence, the cost of encoding diamonds in a diamond-based simplicial complex scales linearly with respect to the dimension, even though the number of simplices scales factorially with respect to the dimension. A simple encoding for diamond-based simplicial complexes of $\text{LEVEL}_{Max} = N$ thus requires $d*N$ bits to encode the central vertex and $d*2$ bookkeeping bits to encode the contained d-simplices of each diamond containing at least one d-simplex in Σ.

In contrast, pointerless encodings for simplex-based simplicial complexes extracted from a forest of simplices typically use locations code to index the encoded simplices. The location code for a simplex σ requires (a) an encoding of the σ's root tree, requiring $O(log(d!))$ bits, (b) the level ℓ_σ of σ, requiring $\lceil log(d*N) \rceil$ bits, and (c) a sequence of bits corresponding to the tree traversal, requiring $\ell_\sigma = O(d*N)$ bits. However, in existing schemes, additional information is required for efficient (i.e. constant time) neighbor finding. In [LDS04], an additional *neighbor mask* is required to efficiently ascertain the appropriate bisection-edge neighbors, and in [AMM07] a reflection vector (consisting of an additional d bits) is required. Finally, since there are no $O(1)$ conversions between location codes and the coordinates of a simplex σ's vertices, the $d+1$ vertices of each simplex can be encoded along with the location code, or can be determined by traversing the tree from the root (requiring $O(\ell_\sigma)$ time).

Besides the storage savings achieved by diamond-based encodings, the simplex-based encodings cannot efficiently cache the status of its subdivided neighbors, and thus each simplex bisection necessarily requires $O(d!)$ iterations of the neighbor-finding operation followed by $O(d!)$ simplex bisections.

We now compare the storage requirements of the two representations in the concrete case of $d = 3$, where i-diamonds can have 6,4 and 8 tetrahedra, for i = 0, 1 and 2, respectively. Table 1 compares the number of diamonds $|\delta|$ to the number tetrahedra $|\sigma|$ in meshes extracted from several volumetric datasets. The `spheres` dataset is an artificially generated distance field for 15 randomly placed spheres. All other datasets were downloaded from the VolVis database. In all cases, diamonds containing a specific isovalue were forced to subdivide while those not containing the isovalue were not. Across all tested datasets, the average number of tetrahedra per diamond was 3.75.

For the storage requirements, we assume that $\text{LEVEL}_{Max} \leq 10$, and thus, coordinates can be encoded using 2 bytes, and location codes using 30 bits (4 bytes). For the comparison of storage costs, we assume that diamonds require 7 bytes each: 6 bytes for the coordinates of the central vertex and 1 byte for the bookkeeping; and that simplices require 6 bytes each: 4 bytes for the location code, one byte for the combined level and tree root encodings, and an additional byte for efficient neighbor-finding (as discussed above). This encoding does not include storage space for the 4 vertices of each simplex (containing 3 coordinates each), and thus their coordinates must be reconstructed e.g. using a top-down tree traversal. Although these meshes are only in 3D, (where $3! = 6$), the diamond-based encoding is around 3 times more compact than a simplified simplex-based encoding. Since the storage requirements scale with the dimension d, these advantage increase as the dimension d increases.

10. Concluding Remarks

We have generalized the notion of diamond to arbitrary dimensions as cross-complexes of two related simplicial decompositions of lower-dimensional hypercubes.

This has enabled us to analyze the properties of diamonds and to derive closed-form equations for the number of d-simplices, vertices, parents and children of all types of diamonds in arbitrary dimensions.

In particular, we proved that an i-diamond in d-dimensions contains $(d-i)!(2i)!!$ d-simplices. Thus, representations in which the primitives are d-simplices become very expensive to store as the dimension d of the problem domain increases. Specifically, since neighbor-finding operations are required for extracting conforming meshes before any bisection operation, extracting conforming modifications to a simplicial complex is a problem with $O(d!)$ complexity.

In contrast, from the perspective of diamonds, we see that the d-simplices within an i-diamond are generated in clusters of size $(d-i)!(2(i-1))!!$ during the subdivision of each of the parents of the diamond. Thus, in applications that require the extraction of simplicial complexes, a diamond-based encoding requires only $O(d)$ spatial accesses to ensure conforming updates to the complex.

We have proposed an implicit pointerless encoding from which all geometric and hierarchical relationships within the hierarchy can be derived using only the d coordinates of a diamond's central vertex. Such representation requires $O(d)$ bytes per diamond. Compared to simplex trees that require $O(d)$ bytes per d-simplex, but must represent $O(d!)$ such d-simplices, our diamond-based representation is significantly more compact.

We note, however, that while the diamond representation can reduce the number of spatial accesses from $O(d!)$ to $O(d)$ during the generation of adaptive domain decompositions, diamonds still contain a number of vertices which varies exponentially with the dimension of the diamond. Thus, we envision the diamond approach to yield the greatest benefit when applied to low-dimensional problem domains of dimension greater than three.

Dataset	LEVEL$_{Max}$	Number of elements			Storage costs (MB)		
		$\|\delta\|$	$\|\sigma\|$	$\|\sigma\|/\|\delta\|$	Σ_δ	Σ_σ	$\Sigma_\sigma/\Sigma_\delta$
Spheres	7	316 K	1.18 M	3.73	2.11	6.76	3.20 x
Spheres	8	1.07 M	3.99 M	3.72	7.16	22.8	3.19 x
Fuel	6	23.1 K	87.5 K	3.78	.15	.50	3.24 x
Hydrogen	7	93.0 K	357 K	3.83	.62	2.04	3.29 x
Tooth	8	281 K	1.05 M	3.74	1.88	6.01	3.20 x
Engine	8	1.40 M	5.30 M	3.78	9.36	30.3	3.24 x
Bonsai	8	1.94 M	7.57 M	3.89	12.98	43.3	3.34 x

Table 1: *Comparison between the number of tetrahedra (σ) and diamonds (δ) in simplicial complexes extracted from the hierarchy of diamonds and forest of simplices representations. For the storage comparison, diamonds in Σ_δ are encoded using 7 bytes and simplices in Σ_σ are encoded using 6 bytes. Storage costs are listed in megabytes, where $1MB = 1024^2$ Bytes.*

Acknowledgments

We would like to thank the anonymous reviewers for their many helpful suggestions. This work has been partially supported by the National Science Foundation under grant CCF-0541032. The volume datasets are courtesy of Stefan Röttger and VolVis.

References

[AMM07] ATALAY F., MOUNT D., MITCHELL J.: Pointerless implementation of hierarchical simplicial meshes and efficient neighbor finding in arbitrary dimensions. *International Journal of Computational Geometry and Applications 17*, 6 (2007), 595–631.

[Bey00] BEY J.: Simplicial grid refinement: on Freudenthal's algorithm and the optimal number of congruence classes. *Numerische Mathematik 85*, 1 (2000), 1–29.

[EKT01] EVANS W., KIRKPATRICK D., TOWNSEND G.: Right-triangulated irregular networks. *Algorithmica 30*, 2 (2001), 264–286.

[Fre42] FREUDENTHAL H.: Simplizialzerlegungen von beschrankter flachheit. *Annals of Mathematics* (1942), 580–582.

[GDL*02] GREGORSKI B., DUCHAINEAU M., LINDSTROM P., PASCUCCI V., JOY K.: Interactive view-dependent rendering of large isosurfaces. In *Proceedings IEEE Visualization 2002* (San Diego, CA, October 2002), IEEE Computer Society, pp. 475–484.

[HAF*96] HIBBARD W., ANDERSON J., FOSTER I., PAUL B., JACOB R., SCHAFER C., TYREE M.: Exploring coupled atmosphere-ocean models using Vis5D. *International Journal of High Performance Computing Applications 10*, 2-3 (1996), 211.

[HDJ05] HWA L., DUCHAINEAU M., JOY K.: Real-Time Optimal Adaptation for Planetary Geometry and Texture: 4-8 Tile Hierarchies. *IEEE Transactions on Visualization and Computer Graphics* (2005), 355–368.

[Heb94] HEBERT D. J.: Symbolic local refinement of tetrahedral grids. *J. of Symbolic Computation 17*, 5 (May 1994), 457–472.

[KTY*04] KIMURA A., TAKAMA Y., YAMAZOE Y., TANAKA S., TANAKA H., CREST J., AGENCY T., SAITAMA J.: Parallel volume segmentation with tetrahedral adaptive grid. *Proc. Int. Conf. on Pattern Recognition 2* (2004).

[Kuh60] KUHN H.: Some combinatorial lemmas in topology. *IBM J. Res. Develop 4* (1960), 518–524.

[LDS04] LEE M., DE FLORIANI L., SAMET H.: Constant-time navigation in four-dimensional nested simplicial meshes. In *Proceedings Shape Modeling International 2004* (June 2004), IEEE Computer Society, pp. 221–230.

[Lic99] LICKORISH W.: Simplicial moves on complexes and manifolds. *Geometry and Topology Monographs 2*, 299-320 (1999), 314.

[LP02] LINDSTROM P., PASCUCCI V.: Terrain simplification simplified: a general framework for view-dependent out-of-core visualization. *IEEE Transactions on Visualization and Computer Graphics 8*, 3 (2002), 239–254.

[LPD*04] LINSEN L., PASCUCCI V., DUCHAINEAU M., HAMANN B., JOY K.: Wavelet-based multiresolution with $\sqrt[n]{2}$ Subdivision. *Dagstuhl Seminar on Geometric Modeling* (2004).

[Mau95] MAUBACH J. M.: Local bisection refinement for *n*-simplicial grids generated by reflection. *SIAM Journal on Scientific Computing 16*, 1 (January 1995), 210–227.

[Mau96] MAUBACH J. M.: The efficient location of neighbors for locally refined n-simplicial grids. *5th Int. Meshing Roundable* (1996).

[Mes48] MESERVE B.: Double factorials. *American Mathematical Monthly* (1948), 425–426.

[Mit92] MITCHELL W.: Optimal multilevel iterative methods for adaptive grids. *SIAM Journal on Scientific and Statistical Computing 13* (1992), 146–167.

[MVT03] MELLO V., VELHO L., TAUBIN G.: Estimating the in/out function of a surface represented by points. *Symposium on Solid Modeling and Applications 2003* (2003), 108–114.

[Pas02] PASCUCCI V.: Slow-Growing Subdivisions (SGSs) in any dimension: towards removing the curse of dimensionality. *Computer Graphics Forum 21*, 3 (2002).

[RL92] RIVARA M., LEVIN C.: A 3D refinement algorithm for adaptive and multigrid techniques. *Communications in Applied Numerical Methods 8* (1992), 281–290.

[RS72] ROURKE C., SANDERSON B.: Introduction to piecewise-linear topology. *Ergebnisse der Mathematik und ihrer Grenzgebiete 69* (1972).

[WB96] WEIGLE C., BANKS D.: Complex-valued contour meshing. In *Proceedings IEEE Visualization 1996* (October 1996), IEEE Computer Society, pp. 173–180.

[WD08] WEISS K., DE FLORIANI L.: Multiresolution Interval Volume Meshes. In *IEEE/ EG Symposium on Volume and Point-Based Graphics* (2008), Eurographics Association, pp. 65–72.