Center for Applied Scientific Computing - Lawrence Livermore National Laboratory

SLOW-GROWING SUBDIVISIONS IN ANY DIMENSION: TOWARDS REMOVING THE CURSE OF DIMENSIONALITY

Valerio Pascucci

Abstract

The efficient representation of volumetric meshes is a central problem in scientific vislualization. The difference in perfoemnce between most visualzation algorithm for rectilinear grids and for ustructured mesh is mostly due to fundamental difference in efficiency of their representations. In Computer Graphics the gap in performance between 2D rectilinear grids and unstructuered mesh has been overcome with the development of representation schemes based on the concept of subdivision surfaces. This gap has not been bridged in the volumetric cases which is fundamental interest for Scientific Visualization.

In this paper we introduce a slow-growing volumetric subdivision scheme for meshes of any topology, any intrinsic dimension d and composed of a general type of polyhedral cells (topological balls).

The main feature of this approach is the ability to split in different stages cells of different dimensions. This allows to increase the resolution of the mesh slowly using small stencils for the smoothing rules. "Sharp features" of dimension lower than d are embedded naturally in the subdivision procedure. Automatic adaptation is provided for variable resolution.

In the uniform case the slow subdivision doubles the number of vertices in the mesh at each refinement independent of its dimension d. The bisection of all the edges in a d-dimensional simplicial mesh requires d subdivision steps. Hence the slow subdivision is a $\sqrt[d]{2}$ subdivision scheme. This algorithm generalizes a recently developed $\sqrt{2}$ subdivision scheme to 3D and higher dimensional meshes where the vertex proliferation becomes increasingly problematic as d grows (the curse of dimensionality).

We introduce a smoothing rule for both the domain mesh and for functions defined on it. Empirical evidence demonstrates the smoothness of the scheme directly on the mesh and indirectly on the isosurfaces of the functions.

Keywords: Subdivision Volumes, Multiresolution Methods, Computational Geometry, Geometric Modeling, Mesh Generation, Scientific Visualization, Solid Modeling

1 INTRODUCTION

The recent of modeling systems, 3D scanning devices and computer simulations gives rise to surfaces and volumetric meshes of increasingly high complexity. The real-time display and transmission of such high resolution data is a challenging task requiring the fast generation of approximated representations. In recent years a great deal of research has been focused on the problem of constructing hierarchical representations with multiple levels of detail. The main techniques involved in the design and use of multi-resolution representations are (among others) wavelet analysis, adaptive mesh refinement and recursive subdivision methods. Great progress has been made in the area of subdivision surfaces for Computer Graphics while volumetric methods, which are more relevant for Scientific Visulaization, remain alomost entirely limited to the case of tensor-product generalization of 1D subdivisions.

One trend in multi-resolution surface generation is the design of methods based on wavelet functions [15, 25, 2, 23]. One main advantage of the multi-resolution analysis at the basis of the wavelet approach is that it immediately gives a compact hierarchical multiresolution data-structure with guaranteed error bounds. The basic ingredient needed for wavelet analysis is the construction of nested function spaces which are best associated with the connectivity of subdivision surfaces. This restricts the class of meshes that can be processed, requiring eventual re-meshing of the input. Hybrid approaches can be designed to take advantage of the quality of wavelet analysis while keeping the generality of a simplification scheme [11].

The general framework of wavelet analysis is formalized independently of the intrinsic/embedding dimension of the geometric object. This enables multi-resolution representation and analysis for volumetric data [22, 24, 18].

Similar solutions have been designed in the meshing community for adaptively refined triangular meshes using a fixed set of templates [1]. Rivara's edge bisection approach is one of the simplest and most flexible of these [21]. A unique subdivision template is used to recursively subdivide the cells of a 2D mesh until a given adaptivity constraint is achieved. This implicitly builds a multi-resolution data-structure from a quality coarse representation. The approach generalizes immediately to 3D tetrahedralizations [21, 19] and to higher dimensions by performing the refinement process from the lower dimensional simplices of the mesh to the higher dimensional. This scheme is is combinatorially equivalent to our slow subdivision for particular tetrahedralizations of rectilinear grids.

Recursive subdivision schemes automatically produce hierarchical multi-resolution representations. This enables, for example, multi-resolution editing techniques [30]. The quality of the generated meshes depends on the subdivision mask used. For triangulated domains Loop [14] provides an approximating subdivision scheme converging to a surface that is C^2 almost everywhere. The exception is at extraordinary vertices, that do not have exactly six incident edges, where the continuity decreases to C^1. The butterfly subdivision scheme [6] converges to an interpolating surface that is C^1 everywhere except for extraordinary points with exactly three, or more than seven, incident edges. A modified version [31] has been proposed that converges to a C^1 surface everywhere. Similarly for subdivision of piecewise quadrilateral domains one can use the Catmull-Clark scheme [4] or the interpolatory scheme by Kobbelt [10] to build smooth approximations of a coarse mesh. Biermann et al. [3] have improved the normal control mechanism [9] for improving Catmull-Clark and Loop subdivisions. A compact and flexible representation of hierarchical surface models can be achieved by using subdivision surfaces to represent the tangential surface refinement information coupled with a normal field that represents the local displacement from the refined mesh [13, 8].

Recently, Kobbelt [12] addressed the problem of excessive refinement by introducing a $\sqrt{3}$-subdivision scheme that increases the number of vertices in the mesh at slower rate than previous approaches. This approach has been improved by Vehlo et al. [29, 28, 26, 27] and independently by Ducheneau et al. [5] with an edge bisection refinement that can be categorized as $\sqrt{2}$ subdivision scheme.

These approaches begin to solve a problem in 2D that gets worse as the dimension d of the mesh increases. The direct generalization of the surface subdivision techniques to volumetric meshes [16, 20] leads to excessive refinement since a uniform smoothing step requires increasing the number of vertices by a factor that grows exponentially with the dimension d of the mesh. This rapid increase of the model complexity can quickly make the use of such schemes impractical. Adaptive refinement also requires special rules to temporarily partition any cell that connects regions at different levels of resolution.

The slow subdivision introduced here roughly doubles the number of vertices independently of the intrinsic dimension of the input mesh. In contrast, tensor-product refinements in 2D quadruple the number of vertices at each refinement [29]. The difference in 3D is even more dramatic since the number of vertices in our scheme is doubled instead of being increased by a factor of eight as in a tensor product scheme. In general the number of vertices is increased by a factor of two (on average) instead of a factor that grows exponentially with the dimension of the mesh. Three major characteristics make the slow subdivision algorithm attractive for a practical purposes.

Slow The rate of refinement (new vertices introduced in the mesh) is independent of the dimensionality of the mesh.

General. The scheme applies for any complex of polyhedral cells such that each cell is topologically a ball. This includes tetrahedral meshes, hexahedral meshes and any mesh with convex elements, for example.

Adaptive. The scheme naturally includes a mechanism for locally adapted refinements and for handling lower dimensional "sharp" features. There is no need to introduce separate classes of cells to connect regions at different levels of resolution.

Small support. The masks that we use for the basic smoothing scheme are as small or smaller than those of Catmull-Clark subdivision.

2 REVIEW OF THE SURFACE SUBDIVISION SCHEME

The $\sqrt{2}$ recursive subdivision scheme [5], also known as 4-8 subdivision surface [29], follows the edge bisection refinement rules introduced by Rivara in [21]. Figure 1(a-e) shows the subdivision scheme in the case of a rectilinear grid. The base mesh is a square divided into two triangles. At any refinement each triangle is split into two halves by bisecting its longest edge. The 4-8 subdivision rule follows these combinatorial rules and adds an averaging step that repositions the vertices on the surface. This deformation of the triangles might alter the classification in the triangle. To maintain the same combinatorial subdivision structure one has to use the "oldest bisection" rule. In each triangle one has to bisect the edge that was unaltered in the previous refinement (there is only one).

Figure 1(a'-e') shows the equivalent subdivision strategy for quadrilateral elements [5]. Each refinement is performed by inserting a point at the center of each *diamond* and splitting the diamond into four triangles. Then each pair of triangles adjacent along an old edge are merged into a new diamond.

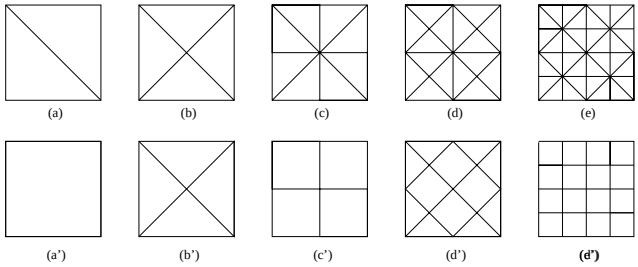


Figure 1: Slow subdivision in two dimensions. (a-e) Classical longest edge bisection of a rectilinear grid. (a'-e') Equivalent $\sqrt{2}$ subdivision where pairs of adjacent triangles are merged into one square.

The advantage of this scheme is that doubling the resolution of a rectilinear grid is performed in two steps instead of one. In the following sections we show how this procedure can be generalized to meshes of any dimension with cells of any type. The refinement of a d-grid is performed in d steps with a stationary rule to be applied at each subdivision.

3 VOLUMETRIC SUBDIVISION

This section introduces a generalization of the $\sqrt{2}$ scheme to 3D polyhedral complexes. The mesh generated by the refinement process is marked in subdivision levels and passes. Each level l has four passes, from 0 to 3, where pass 3 of level l is coincident with pass 0 of level $l+1$. The base mesh is level 0 pass 0. At each refinement the pass is increased by one. At pass 3 the level is increased by one and the pass is reset to 0. The cells, facets, edges and vertices of the base mesh are denoted C_i, f_i, e_i and v_i respectively.

Note that in the present context we consider only the combinatorial structure of the scheme. For example we qualify a vertex as the "center" of a cell/face just to give an intuitive notion of the relation between the point and the cell but we do not refer to its actual geometric location. Different choices of the vertex position may yield different smoothness properties of the limit mesh.

Figure 2: 3D cell refinement from pass 0 to pass 1. Two cells of pass 0 are shown on the left and on the right. The gray cell on the right is the new cell F created by merging the two pyramids whose base is the gray facet f. Note that the figure shows only the refinement edges incident to f.

From Pass 0 to Pass 1 For each cell C_i in the complex the center p_i is computed. The cell C_i, having n facets is decomposed into n pyramidal cells by connecting the center p_i with each facet of C_i. Let's denote by $p_i \lhd f_k$ the pyramid built by connecting p_i with a given facet f_k. For each pair of cells C_i, C_j adjacent along a facet f_k a new cell F_k is created by merging the pyramid $p_i \lhd f_k$ with the pyramid $p_j \lhd f_k$:

$$F_k = (p_i \lhd f_k) \cup (p_j \lhd f_k), \qquad \text{with} \quad f_k = C_i \cap C_j .$$

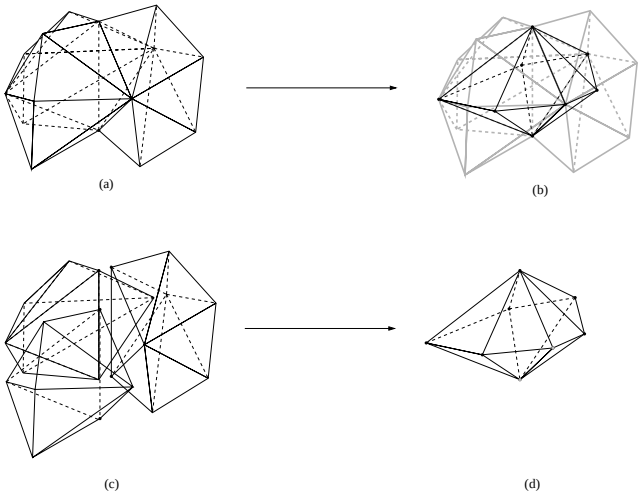


Figure 3: Cell refinement from pass 1 to pass 2. (a) Three cells of pass 1 are shown in black lines. (b) A cell of pass 2 in black is shown overlapped with the corresponding cells of pass 1 in gray. (c) The three cells of (a) are shown separated from each other. (d) The new cell of pass 2 is shown isolated from the neighborhood.

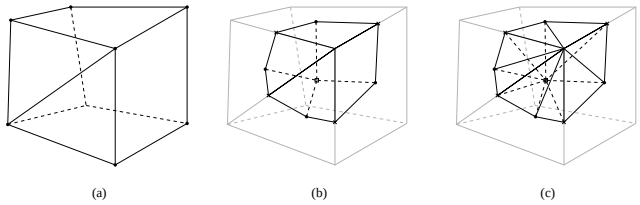


Figure 4: Cell refinement from pass 0 to pass 3. (a) Cell from the mesh of pass 0. (b) Cell of pass 3 in black is shown overlapped with the ancestor cell of pass 0 in gray. (c) The same refined cell with spurious additional edges that may be left during the refinement procedure.

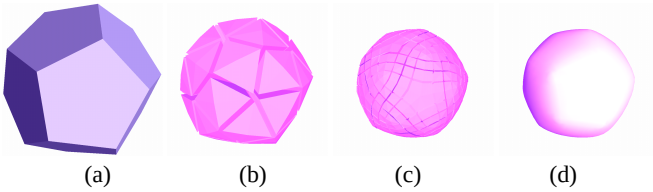


Figure 5: Successive subdivision stages of a dodecahedron. (a-c) Mesh elements, (d) boundary.

Figure 2 shows one cell created in this step of refinement from two adjacent cells of a base mesh. Note that for boundary cases we build only one half of the cell F_k. Similarly for 2-dimensional sharp features, that is surfaces that need to be preserved in the refinement process, one builds two halves F_k' and F_k'' without merging them at f_k. In such cases the boundary pyramid F_k, or the two halves F_k' and F_k'', are called "sharp" since they maintain the facet f_k of pass 0. One can also build only one half F_k' of F_k to achieve a locally adapted refinement connecting the refined pyramids of C_i with the unrefined C_j. In such case F_k is not sharp. It is instead marked as "non-refinable" until merged with its second half not-yet existent.

From Pass 1 to Pass 2. For each cell F_k in the complex of pass 1, one determines the "center" q_k. Let $\{g_i\}$ denote the set of new facets generated in the pass 1 complex. Each cell F_k is decomposed into a set of pyramids each given by $q_k \lhd g_i$. For a sharp cell F_k, the center q_k is joined only with the new facets g_i and not with the old facet f_k. In this case it would be more appropriate to consider q_k as the center of f_k. In fact, for 2-dimensional sharp features, q_k would be shared between the two cells F_k' and F_k'' adjacent along f_k.

Each pyramid built in this way contains exactly one edge e_j of the pass 0 mesh of level l. All the pyramids incident to an edge e_j are merged into a cell E_j. The collection of the cells E_j forms the mesh of pass 2. Figure 3 shows the construction of one cell of pass 2 from three cells of pass 1.

At this stage sharp features of dimension one are dealt with. In particular if an edge e_j is part of a sharp feature then the pyramids around e_j are not merged but are marked as "sharp". Moreover if some of the cells incident to e_j are not generated pass 1 then the pyramids are not merged but are marked as non-refinable. They are used for connecting cells of different resolution.

From Pass 2 to Pass 3. As in the previous two steps one determines the center r_i of each cell E_i. The cells E_i are refined by joining r_i with each facet of E_i. As usual, for sharp cells the point r_i should be considered as the center of e_i and should be shared among all the cells around e_i.

The last merging step is among cells that contain the same pair of vertex v_i and are contained in the cell C_j. Note that, during this last merging stage, particular care needs to be taken in removing any spurious edges introduced during the refinement procedure. In particular any edge connecting two vertices introduced in passes that differ by more than one need to be removed. Figure 4 shows one cell of pass 0 (a) and one of its descendants of pass 3 (b). Figure 4(c) shows the refined cell with the spurious edges (facet diagonals) connecting the vertex of pass 0 with vertices of pass 2 or the vertex of pass 1 with vertices of pass 3.

The cells generated in this refinement process can be characterized in a simple way.

Definition 1 *A d-**diamond** is a d-dimensional cell that can be partitioned into a set of d-simplices all sharing an edge.*

For any base mesh in 3D all the cells generated by one step of the subdivision procedure are either three-diamonds or pyramids. In the first step each cell is either a pyramid or formed by two merged pyramids (composing a diamond). In the second step each cell is a diamond composed of a set of the tetrahedra sharing an edge of the base mesh. In the third subdivision step each cell is a diamond composed of tetrahedra sharing one vertex of the level's base mesh and one vertex at the center of one cell of the level's base mesh.

Figure 5 shows the scheme applied to a base mesh consisting of a single dodecahedron.

4 SUBDIVISION IN ANY DIMENSION

Consider a d-dimensional complex $\mathcal{C}$ of m cells $c_0, \ldots, c_{m-1}$. The complex is subdivided in $d+1$ passes that are applied recursively. As in the 3D case, pass d of level l is coincident with pass 0 of level $l+1$. Refining from pass p to pass $p+1$, one adds a vertex at the center of a $(d-p)$-dimensional face of the base mesh M_0^l at the same level (the last mesh of pass 0).

From pass p to pass $p+1$ (with $p < d-1$). Each cell c in the mesh of pass p is divided by inserting a vertex q. The cell c is partitioned into pyramids by connecting q to all its facets. All the pyramids containing a common $(d-p-1)$-dimensional face of

M_0^l are merged together. If one $(d-p)$-dimensional face f of M_0^l belongs to the boundary of c (there can be at most one), then q is placed at the center of f and the pyramids partitioning c are built by connecting q to all the facets of c that do not contain f.

From pass $d-1$ to pass d. Each cell c is split by inserting a vertex q at the edge of M_0^l. For non-boundary and non-sharp features, q is also the center of c itself. The cell c is partitioned into pyramids by connecting q to the facets that do not contain q. The new cells are built by merging all the pyramids containing the same vertex of M_0^l and center of a cell of M_0^l.

In both refinement rules adaptivity and sharp features are dealt with simply by not performing the merge step. Similarly to the 3D case we can characterize in a simple way the structure of the cells produced by the subdivision scheme.

Theorem 1 *All the non-boundary/non-sharp cells generated by the subdivision scheme after the first $d-1$ passes are d-diamonds.*

Proof: The proof of the theorem follows from the equivalence of our scheme to the d-dimensional edge bisection scheme of Maubach, for which the theorem also holds as shown in [17]. First partition each cell into tetrahedra by joining each edge of the cell with the center of each incident face of higher dimension (in decreasing dimension order). That is, if (v_0, v_1) are the vertices in an edge of a cell then one builds simplices $(v_0, v_1, v_2, \ldots, v_d)$ where v_2 is the center of the cell and v_d is the center of an edge incident to (v_0, v_1). Call the first edge (v_0, v_1) of the simplex the *oldest edge*. By merging all the simplices sharing the same oldest edge one obtains the d-diamond that is created by the first $d-1$ refinements of our slow subdivision.

From this stage on, the slow subdivision is equivalent to the oldest edge bisection of the simplicial complex just built. In particular each refinement is obtained by replacing the simplex $(v_0, v_1, v_2, \ldots, v_d)$ with $(v_0, v_2, \ldots, v_d, v^*)$ and $(v_1, v_2, \ldots, v_d, v^*)$ where v^* is the midpoint of (v_0, v_1). Merging all the cells with the same oldest edge one obtains the same cells as the slow subdivision algorithm. ◇

5 DIRECT AND INDIRECT SMOOTHING

The combinatorial rules defined in the previous section need to be coupled with some averaging rules to determine the actual position of the vertices of the mesh. We propose the following stationary rule.

- A new vertex v inserted at the center of a cell/face c is the average of the vertices of c.
- Each old vertex v is repositioned by linearly combining its old position v_{old} with the average w of the vertices that are edge-connected to it.

$$v = v_{old} * \alpha + (1-\alpha)w$$

In the 2D case this rule is equivalent to those reported in fig. 4 of [29] since in the vertex masks the edge-connected vertices with weight 0 are exactly those that are removed by the merge stage of the recursive subdivision defined here. For boundary cases and sharp features the neighborhoods are computed restricting the search to the features themselves.

Figure 6 shows the mesh elements generated by repeatedly applying the slow subdivision scheme starting from a base mesh composed of eight cubes. This subdivision is performed with an averaging coefficient $\alpha = 0.5$. The parameter α can be used to alter the behavior of the smoothing mask. Figure 7 shows the successive refinements of the same mesh using two different values of $\alpha = 0.89$ (left columns) and $\alpha = 0.5$ (right column).

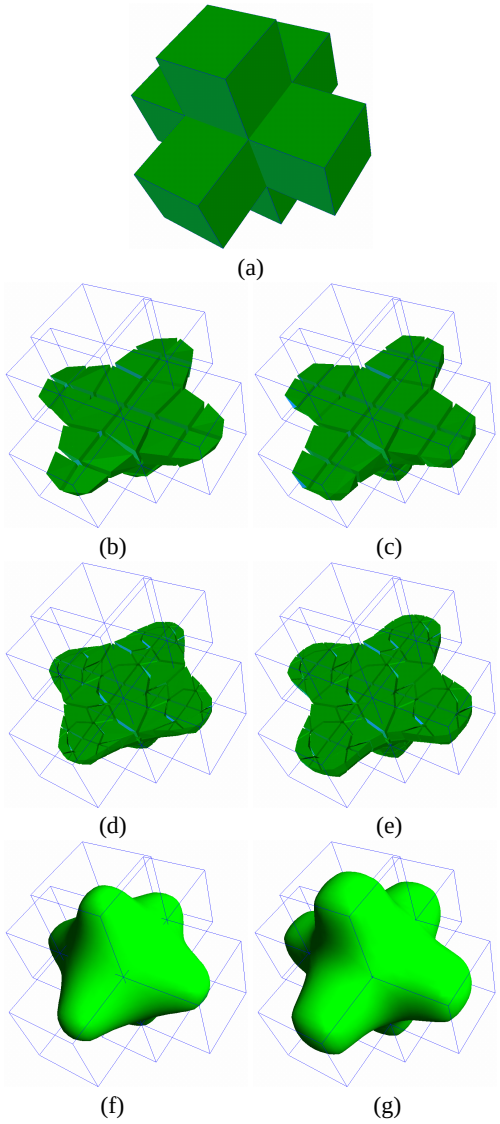
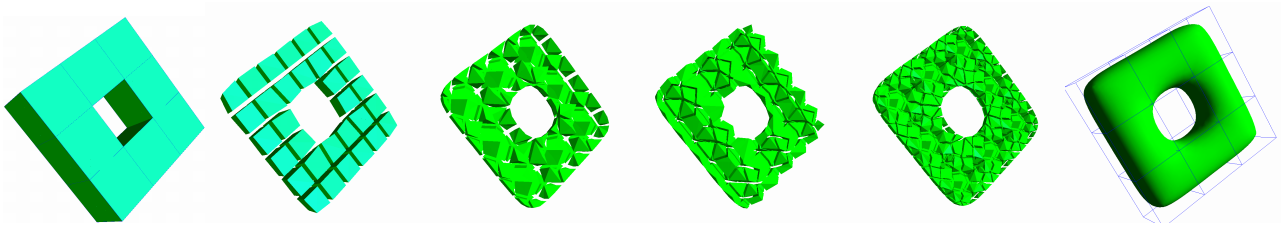


Figure 7: Subdivision of the same shape with different smoothing coefficient. (a) Base mesh. (b,d,f) Intermediate steps and smooth boundary for a subdivision with smoothing coefficient $\alpha = 0.89$. (c,e,g) Same subdivision steps with $\alpha = 0.5$

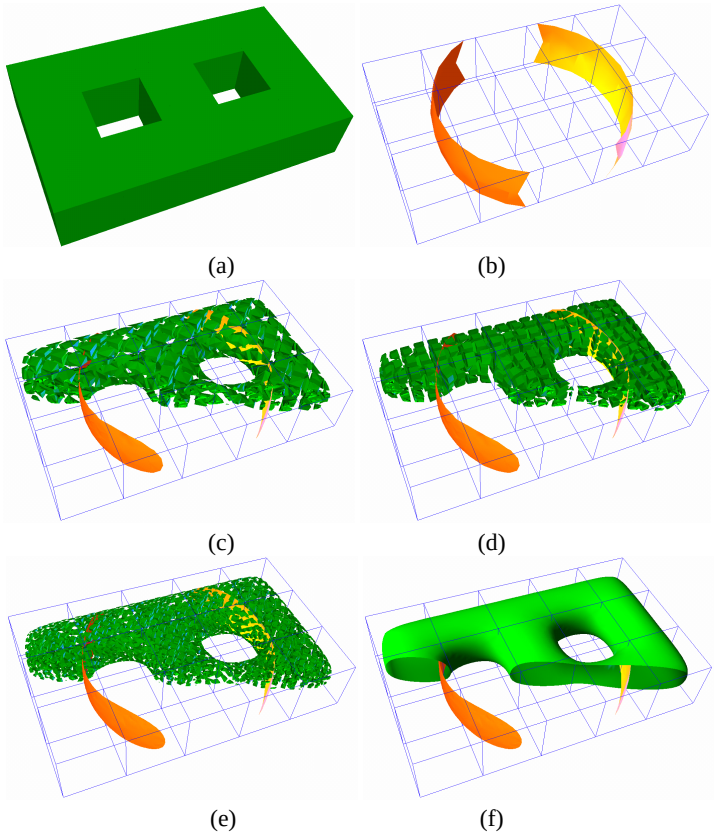
The 2d scheme has been analyzed independently in [5] and [29]. Velho and Zorin proved that the slow subdivision surfaces with $\alpha = 0.5$ have C^1 continuity everywhere and C^4 continuity at the regular points. The proof is based on the Zwart-Powell element box spline basis. Here we generalize the scheme to dimensions higher than two. This allows us to smooth the functions defined on the mesh. A more sophisticated analysis is necessary to determine the smoothness of the scheme, especially at the extraordinary points. Note that in 2D the extraordinary points can be characterized simply by their degree. In 3D, this is not sufficient since one has to analyze the local connectivity structure at the vertex (vertices with same degree can have different behavior.) Figure 8 shows the combined refinement of a mesh (in green) and isosurface (in orange) of a scalar field defined by linearly interpolating function values



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

Figure 6: Interior elements generated by the slow subdivision scheme. (a) Base mesh formed by eight cubes. (b-e) Internal structure of the mesh in successive stages of the slow subdivision scheme. The pictures show only the mesh elements whose centroid has negative z coordinate. (f) Subdivided boundary mesh.

at the vertices of the mesh. The linear interpolation is based on a decomposition of the diamonds into tetrahedra.



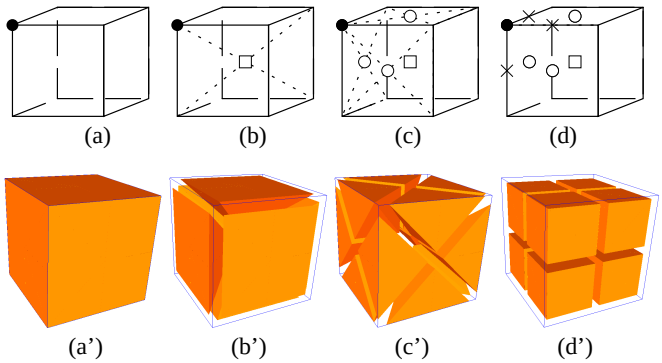
(a) (b) (c) (d) (e) (f)

Figure 8: Combined refinement of mesh domain and embedded isosurface. (a) Base mesh. (b) Base isosurface. (c-e) Intermediate subdivision steps. (f) Refined boundary surface and isosurface.

6 RATE OF REFINEMENT

This section gives a precise upper bound of the rate of refinement of the subdivision scheme.

First, consider the case of d-dimensional rectilinear grids, where the scheme can be reduced to simplicial longest edge bisection [17]. In particular, consider a rectilinear grid whose cubes are decomposed into $d!$ simplices by splitting each face along one longest diagonal. In this case, d successive steps of refinement insert a vertex at the center of each face. This is equivalent to splitting the longest



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

Figure 9: (a-d) Rate of vertex increase for each refinement in a 3D rectilinear grid. For each initial vertex (marked with a black disc) one new vertex is added in the first refinement (square), three new vertices in the second refinement (circles) and three are added in the third refinement (crosses). (a'-d') Corresponding refinement of the cells in the cube.

edge of the simplicial decomposition d consecutive times. In particular all and only the edges of the initial simplicial decomposition are bisected. Using the terminology from [12], the scheme is a $\sqrt[d]{2}$ subdivision scheme.

We can also determine the average number of vertices added to the mesh by each pass. Consider an axis-aligned infinite rectilinear grid (for simplicity we do not consider boundary effects). For $1 \leq h \leq d$, one vertex can be associated with all the h-dimensional faces in which it has the minimum coordinates (the vertex closest to the origin). At the refinement from pass p to pass $p+1$, one vertex is added at the center of each of the $\binom{d}{p}$ faces of dimension $d-p$. The number of vertices added in each step for each vertex in the grid is $\sum_{h=0}^{d-1} \binom{d}{p} = 2^d - 1$ which means that on average the number of vertices in the grid doubles at each of the d refinement steps. For example, Figure 9 shows the 3D case. For each vertex in the original grid a new vertex is inserted at the center of a cube, doubling the total number of vertices. At the second pass three vertices are inserted at the face centers, increasing the number of vertices by a factor of $\frac{5}{2} = 2.5$. At the third pass three vertices are used to bisect three edges, increasing the number of vertices by a factor of $\frac{8}{5} = 1.6$.

In this case the total number number of cells is linear in the number of vertices. This is not true in general. In particular it is known that the total number N of cells in a simplicial complex (worst case scenario) with n vertices can be as large as $N = O(n^{\lfloor d/2 \rfloor})$. The total number of vertices inserted in d passes is N. If after one refinement the number of vertices increases from n to βn then we obtain $\beta^d n = n^{\lfloor d/2 \rfloor}$ which implies that $\beta = O(\sqrt[d]{n^{\lfloor d/2 \rfloor - 1}})$. For

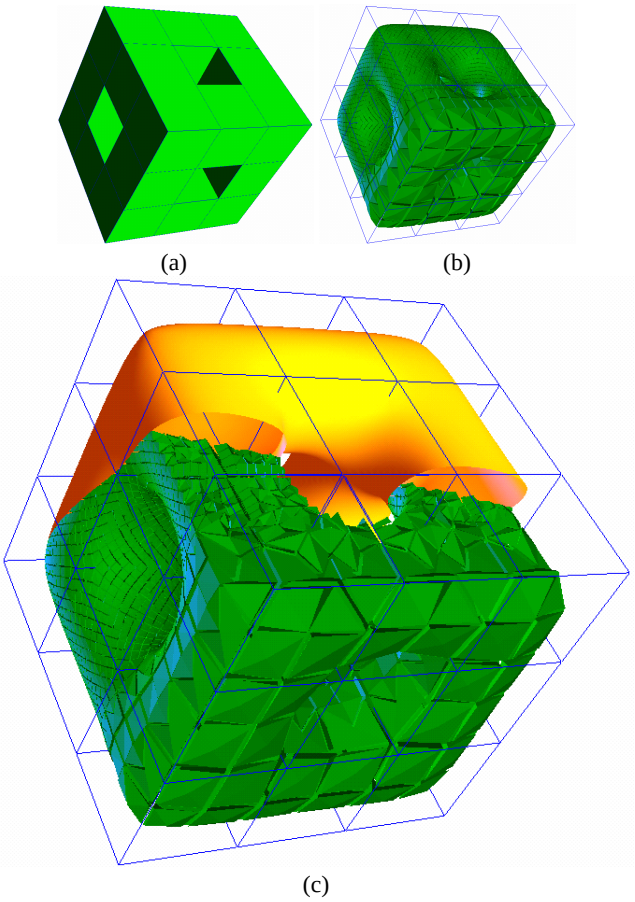


Figure 10: Adaptive refinement of a 3D mesh. (a) Base mesh. (b-c) Exterior and interior refined mesh elements.

$d = 2$ we have $\beta = O(1)$ confirming that in the worst case there is a constant rate of vertex increase. Even in 3D we have $\beta = O(1)$ guaranteeing a constant rate of refinement in the worst case. Interestingly for regular grids β is a small constant 2 independently of d. Note that for $d > 3$ β may not be a constant in the worst case. For example for $d = 4$ we have $\beta = O(\sqrt[4]{n})$.

7 ADAPTIVE REFINEMENT AND LOWER DIMENSIONAL FEATURES

The flexibility of the slow subdivision scheme derives from its independent treatment of cells having different dimension. Each of the d refinement passes subdivides the cells of one specific dimension. This allows us to easily solve two problems: (i) building mesh adaptation between regions at different resolution and (ii) esplicitly representing lower-dimensional features embedded in the mesh.

The mesh adaptation between regions at different levels of resolution does not require the introduction of special temporary partitions such those required for adqaptive Catmull-Clark subdivision. During the refinement procedure one simply does not perform some of the merging steps. This automatically cretes the necessary adaptation. Figure 10 shows an example of mesh adaptation. The element density in the mesh increases along the positive direction of the z-axis.

A similar argument holds for lower dimensional features embedded in the mesh that the scheme automatically handles in separated stages. Figure 11 shows the application of the slow subdivision to

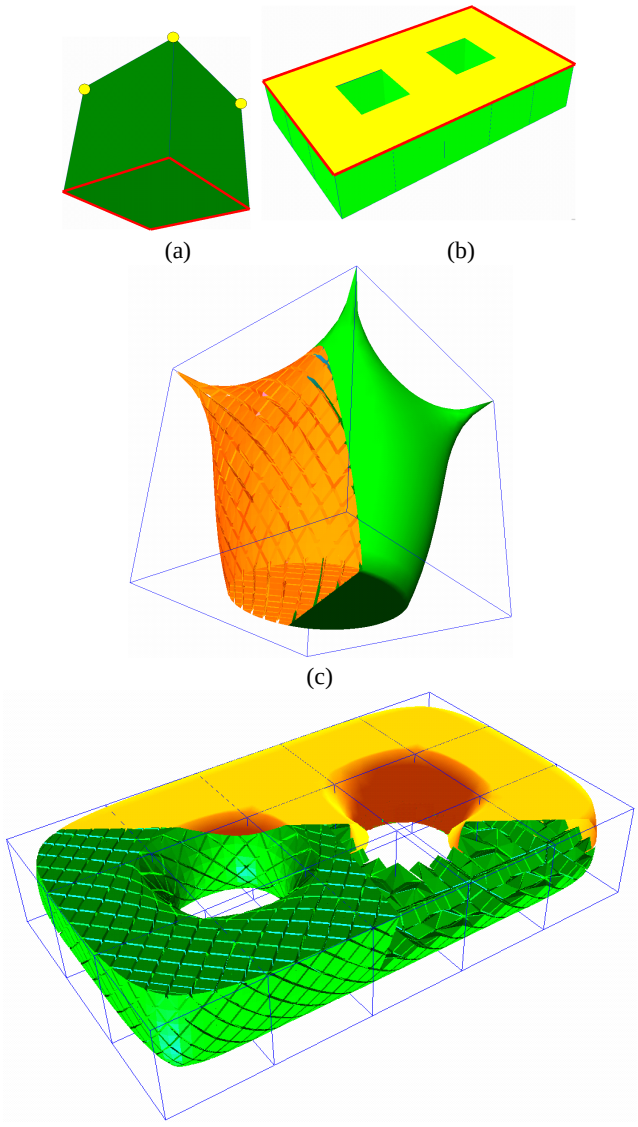


Figure 11: Recursive subdivision with user defined sharp features. (a) Cube base mesh with the four top vertices (in yellow) marked as 0-dimensional features. The square at the bottom (in red) is marked as 1-diensional feature. (b) Eight-shaped base mesh with a one-dimensional sharp feature (in red) and a two-dimensional feature (in yellow). (c) Refinement of the cube. (d) Refinement of the second mesh. Note the difference to the subdivision volume in Figure 8 using the same base mesh but without sharp features.

base meshes with user-defined sharp features.

For the same reason, the scheme presents no problem in handling non-manifold features as shown in Figures 12 and 13.

8 CONCLUSIONS

In this paper we have introduced a subdivision scheme that allows gradual refinement of a very general class of meshes. This slow subdivision scheme addresses the issue of lowering the vertex proliferation rate. This addresses the fundamental problem in Scientific Visualization of designing an efficient representation scheme

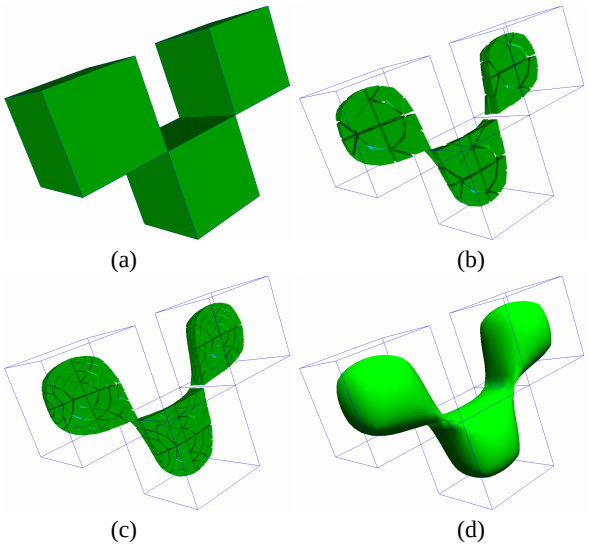


Figure 12: Refinement of a mesh with two non-manifold edges. (a) Base mesh. (b-c) Interior mesh of two intermediate refinement teps. (d) Smooth boundary.

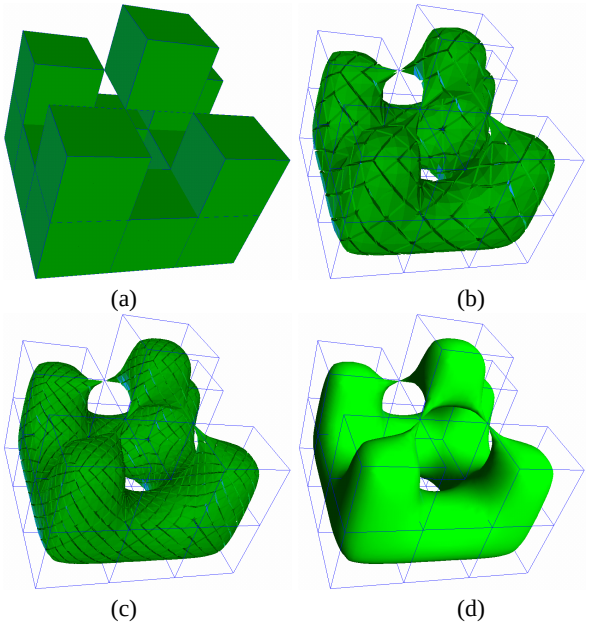


Figure 13: Refinement of a mesh with four non-manifold points. (a) Base mesh. (b-c) Mesh elements of two intermediate refinement steps. (d) Smooth boundary.

for volumetric meshes. In standard generalizations of recursive tensor-product subdivision algorithms, the rate of vertex increase is constant in any fixed dimension. Unfortunately, the constant grows exponentially with the intrinsic dimension d of the mesh. This type of problem is common to many geometric algorithms and is commonly referred to as "the curse of dimensionality". The combinatorial rules proposed here allow us to keep the rate of vertex increase down to a constant factor of two, independent of d. One fundamental feature of the slow subdivision scheme is that it refines in d independent stages the cells of different dimensions. This allows us to adaptively refine in a natural way without needing special subdivision rules for cells connecting regions at different resolutions. Similarly, the scheme naturally embodies rules for handling sharp vertices, edges, and triangles. We have experimented with a very simple averaging rule that has been proven visually smooth for the 2D case and extends naturally to the volumetric and higher dimensional cases. A complete analysis for the volumetric case remains to be done, but empirical results show a good smoothing behavior. Particularly interesting is the combined use of the scheme for smoothing of 3D meshes and scalar fields sampled at the vertices of the meshes.

The slow subdivision scheme can be used in several application areas such as sold modeling, meshing and scientific visualization. For example, it could be used in conjunction with [7] to obtain indirect smoothing of objects modeled with distance fields.

9 ACKNOWLEDGMENTS

Special thanks go to Mark Ducheneau and Ken Joy for the discussions that inspired this work. Thanks also to Issac Trotts whose comments helped in improving the presentation.

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract no. W-7405-Eng-48.

References

[1] R. E. Bank, A. H. Sherman, and A. Weiser. *Scientific Computing (Applications of Mathematics and Computing to the Physical Science)*, chapter Refinement Algorithms and data structures for regular local mesh refinement, pages 3–17. North Holland, 1983.

[2] Martin Bertram, Mark A. Duchaineau, Bernd Hamann, and Kenneth I. Joy. Bicubic subdivision-surface wavelets for large-scale isosurface representation and visualization. In T. Ertl, B. Hamann, and A. Varshney, editors, *Proceedings Visualization 2000*, pages 389–396. IEEE Computer Society Technical Committe on Computer Graphics, 2000.

[3] Henning Biermann, Adi Levin, and Denis Zorin. Piecewise smooth division surfaces with normal control. In *Proceeding of SIGGRAPH 2000*, pages 113–120. ACM, August 2000.

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

[5] Mark A. Duchaineau, Benjamin Gregorski, and Kenneth I. Joy. Smooth centroid bintree subdivision surfaces with local wavelets. Technical Report Technical Report, CSE-2001-1, Computer Science Department, University of California, Davis, January 2001.

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

[7] Sarah F. Frisken, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones. Adaptively sampled distance fields: a general representation of shape for computer graphics. In *Proceeding of SIGGRAPH 2000*, pages 249–254. ACM, August 2000.

[8] Igor Guskov, Kiril Vidimaccarone, Wim Sweldens, and Peter Schroder. Aaron lee. In *Proceeding of SIGGRAPH 2000*, pages 95–102. ACM, August 2000.

[9] Mark Halstead, Michael Kass, and Tony DeRose. Efficient, fair interpolation using Catmull-Clark surfaces. *Computer Graphics*, 27(Annual Conference Series):35–44, 1993.

[10] L. Kobbelt. Interpolatory subdivision on open quadrilateral nets with arbitrary topology. *Computer Graphics Forum*, 15(3):C409–C420, C485, September 1996.

[11] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *SIGGRAPH 98 Proc.*, pages 105–114, 1998.

[12] Leif Kobbelt. $\sqrt{3}$-subdivision. In *Proceeding of SIGGRAPH 2000*, pages 103–112. ACM, August 2000.

[13] Aaron Lee, Henry Moreton, and Hugues Hoppe. Displaced subdivision surfaces. In *Proceeding of SIGGRAPH 2000*, pages 84–94. ACM, August 2000.

[14] C. Loop. Smooth spline surfaces over irregular meshes. In *SIGGRAPH '94 Proc.*, pages 303–310, 1994.

[15] M. Lounsbery, T. D. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics*, 16(1):34–73, 1997.

[16] Ron MacCracken and Kenneth I. Joy. Free-form deformations with lattices of arbitrary topology. In *Computer Graphics (SIGGRAPH'96)*, pages 181–188, 1996.

[17] Joseph M. Maubach. The amount of similarity classes created by local n-simplicial bisection refinement, 1996.

[18] Shigeru Muraki. Multiscale volume representation by a doG wavelet. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):109–116, June 1995.

[19] A. Plaza and G. F. Carey. About local refinement of tetrahedral grids based on local bisection. In *5th International Meshing Roundtable*, pages 123–136, 1996.

[20] Alon Raviv and Gershon Elber. Three dimensional freeform sculpting via zero sets of scalar trivariate functions. In *Proceedings of the fifth symposium on Solid modeling and applications*, pages 246 – 257, 1999.

[21] M.-C. Rivara and C. Levin. A 3-d refinement algorithm suitable for adaptive and multi-grid techniques. *Comm. in Appl. Numer. Meth.*, 8:281–290, 1992.

[22] R. Sánchez and M. Carvajal. Wavelet based adaptive interpolation for volume rendering. In *IEEE Symposium on Volume Visualization*, pages 127–134, 1998.

[23] Peter Schröder and Wim Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 161–172. ACM SIGGRAPH, Addison Wesley, August 1995. held in Los Angeles, California, 06-11 August 1995.

[24] O. G. Staadt, M. Gross, and R. Weber. Multiresolution compression and reconstruction. In *Proceedings of IEEE Visualization 1997*. IEEE, 1997.

[25] Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin. *Wavelets for Computer Graphics: Theory and Applications*. Morgann Kaufmann, San Francisco, CA, 1996.

[26] Luiz Velho. Using semi-regular 4–8 meshes for subdivision surfaces. *Journal of Graphics Tools*, 2001.

[27] Luiz Velho and Jonas Gomes. Quasi 4-8 subdivision surfaces. In *Proceedings of SIBGRAPI'99 - XII Brazilian Symposium on Computer Graphics and Image Processing*, Campinas, SP, Brazil, October 1999. SBC - Sociedade Brasileira de Computacao, IEEE Press.

[28] Luiz Velho and Jonas Gomes. Variable resolution 4-k meshes: Concepts and applications. *Computer Graphics Forum*, 2000.

[29] Luiz Velho and Dennis Zorin. 4-8 subdivision. *Computer Aided Geometric Design*, 2001. To appear (currently available at http://www.mrl.nyu.edu/publications/four-eight/.

[30] D. Zorin, P. Schroder, and W. Sweldens. Interactive multiresolution mesh editing. *Computer Graphics*, 31(3A):259–268, August 1997.

[31] D. Zorin, P. Schroeder, and W. Sweldens. Interpolating subdivision for meshes with arbitrary topology. In *SIGGRAPH 96 Proc.*, pages 189–192, 1996.