Adaptive Mesh Generation I: Packing Space

Doug Moore
Joe Warren

CRPC-TR90039
May, 1990

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

Revised December, 1990.

Adaptive Mesh Generation I: Packing Space *

Doug Moore †
Rice University

Joe Warren ‡§
Rice University

December 1, 1990

Abstract

In applications such as finite element analysis and computer graphics, the ability to pack space with a conformal mesh of elements of similar shape but different size is useful. This paper describes a technique for adaptively packing space with a conformal mesh of tetrahedra of bounded aspect ratio.

The method consists of three primary steps, each of which relies on a new observation or algorithm.

- *Adaptively subdivide space using an octree of tetrahedral components.*
  We show that such an octree is possible thanks to the existence of a provably unique symmetric tetrahedron that can be recursively subdivided into eight similar tetrahedra.
- *Maintain balance in the octree.*
  A balanced octree is one in which the edge lengths of neighboring elements differ by at most a factor of two. We give two new algorithms for maintaining balance during adaptive subdivision.
- *Produce a conformal tetrahedral octree from a balanced tetrahedral octree.*
  We give a simple algorithm that constructs a conformal octree with tetrahedra of only a few different shapes and avoids the introduction of extra vertices in the interior of elements of the balanced octree.

Each of these components may be generalized to produce conformal simplicial partitions of arbitrary dimension.

Keywords - mesh generation, finite elements, multidimensional geometry

AMS MOS classification - 51-04, 51M20, 52A45, 68U05

---

*Submitted to *The International Journal of Computational Geometry and Applications*

†Supported in part by Center for Research in Parallel Computation

‡Supported in part by NSF grants IRI 88-10747 and CCR 89-03431

§Authors' address: Department of Computer Science, P.O. Box 1892, Houston, Tx 77251

1 Introduction

An essential component of many geometric algorithms is the construction of a partition of space. In such applications as finite element analysis, solid modeling, computer-aided geometric design and graphical rendering, among others, important problems can be solved efficiently by decomposing the geometric domain into polyhedral elements, solving a problem over each element of the decomposition, and unifying the separate solutions into an exact or approximate solution to the larger problem. Among the well-known space partitions are triangulations, quadtrees [Sam90], k-D trees [Ben75] and binary space partition trees [FKN80].

Several properties are desirable in a partition of space, including *adaptivity*, which allows elements to differ considerably in size, *conformality*, which requires that the intersection of two elements be a proper face of each, and *quasi-regularity*, which ensures that the elements do not degenerate into flat or distorted shapes. All the hierarchical space partitions named above are fully adaptive, but not conformal. When conformality is necessary, elements from these partitions can be triangulated in a post-processing step to achieve it. Only triangulations can be both conformal and completely adaptive. Unless precautions are taken, however, an adaptive triangulation can be highly irregular.

In this paper, we present new techniques that produce an adaptive, conformal packing of space with well-shaped elements. An instance of the problem consists of an initial, possibly trivial, partition of a region of space and an *error predicate* $e(S)$ that is true exactly when the tetrahedron S is sufficiently small. The solution is a conformal packing of space with tetrahedra each of which satisfy the error predicate. The construction of this packing consists of three major steps.

First, we construct a *tetrahedral region octree*, like the more familiar cubical region octree in many ways, but composed of similar tetrahedra instead of cubes. The existence and uniqueness of the tetrahedron that allows this hierarchical decomposition of space is proven in an appendix. The tetrahedra are subdivided sufficiently to ensure that each satisfies the error predicate.

Additional tetrahedra are decomposed, either during or after the construction of the

tetrahedral octree, to produce a *balanced* tetrahedral octree. In this balanced octree, tetrahedra that share a vertex have edges that differ in length by at most a factor of two. The competing approaches of maintaining balance throughout the decomposition or restoring balance after the subdivision lead to two new techniques, each with its own advantages.

Finally, we construct from a balanced tetrahedral octree a conformal octree. Unlike some techniques, this *vertex marking* method does not require the placement of new vertices within the interiors of elements. The only additional vertices required are at the midpoints of edges of elements present in the balanced octree. Each of the tetrahedra present in the conformal partition is similar to one of five dissimilar tetrahedra, none of which is particularly badly shaped.

Each of these steps, although described as a process in three-dimensional space, is independent of dimension. In section 3, we show that the decomposition of a tetrahedron into eight similar tetrahedra generalizes into the decomposition of a particular simplex of n dimensions into 2^n similar simplices. The decomposition of the n-cube into smaller n-cubes is the basis for a hierarchical subdivision of n-space called an n-dimensional cubical region quadtree [Sam90]. Similarly, the decomposition of the n-simplex is the basis for the *n-dimensional simplicial region quadtree*, which represents a hierarchical subdivision of n-space into simplices. Other techniques are described in their general n-dimensional form as well. The generality of these techniques is important, because time-varying problems in physics and engineering can often be expressed as problems in four dimensions, and problems in data analysis and visualization are often of high dimension.

Much of the work on mesh generation for finite element analysis has considered the space-packing problem with the constraint that the elements must include certain predetermined vertices or edges or lie on certain faces. In a sequel to this paper [MW90], we describe how a packing of space can be adjusted to satisfy these sorts of constraints.

2 Related Work

The question of which shapes fill space is a classical one, with proofs and conjectures dating back over 2000 years. Senechal [Sen81] reviews the history of attempts to fill space with

tetrahedra. Field [Fie86] describes a recent method for filling space with tetrahedra, based upon an approximate packing of space with icosahedra. The book by Coxeter [Cox73] is a source of much information on the more general problem of filling higher dimensional spaces with polytopes.

Several of the standard approaches to mesh generation are based upon the construction of a (cubical) quadtree [YS84] and recent work has shown that that general approach can lead to provably good meshes [BEG90]. In particular, balanced quadtrees have been used to construct partitions of parameter space for use in parametric surface rendering [HB87].

Other automatic mesh generators rely upon Delaunay triangulations of carefully constructed point sets to produce well-shaped elements [Ban90, Joe84]. The specific problem of triangulating polygons has also received considerable attention, with attention paid to the shape of the triangles [BGR88, Smi88] and to the efficiency of the calculation [TW88]. Other mesh generators based on iterative triangulation methods include those reported in [Riv87, FF85, CFF85].

3 Recursive decomposition of n-simplices

For generating a uniform partition that fills a region of $\Re^n$, many shapes are possible; the study of plane tilings provides many fascinating examples and open problems. For iteratively generating an adaptive partition of space, however, the elements of the partition at each step must be amenable to subdivision. A hexagonal tiling of the plane, for example, cannot be easily refined because a hexagon cannot be divided into a number of similar hexagons. The subdivision of a square into four smaller squares is the basis for the familiar quadtree data structure [Sam90], and the generalization of the square to n dimensions, called the n-cube, is the basis for hierarchical decomposition called the n-dimensional cubical region quadtree.

Less obvious, but quite as important, is the fact that in each dimension there exists a simplex that can be subdivided into 2^n similar simplices. This is well known in two dimensions, where a simplex is a triangle and three segments, each connecting a pair of edge midpoints, divide any triangle into four smaller similar triangles. In higher dimensions, only carefully chosen simplices have this property.

One particular kind of recursively decomposable simplex, which has been appreciated more for its combinatorial structure than for its geometry, is the n-dimensional *quadrirectangular simplex* [Cox73], or Q-simplex for short. In standard position, the Q-simplex Σ^n is the convex hull of the points

V_0 (0 0 0... 0)

V_1 (1 0 0... 0)

V_2 (1 1 0... 0)

V_3 (1 1 1... 0)

$\vdots$

V_n (1 1 1... 1).

In [DM82], the combinatorial structure of Σ^n is described in detail, and we refer to several of these results below.

One of the important properties of the Q-simplex is that $n!$ Q-simplices in n dimensions can be joined to form an n-cube. This may be seen by considering the number of paths of length n along edges of an n-cube that start at V_0 and end at V_n. Each such path defines a Q-simplex, there are $n!$ such paths, and the union of the Q-simplices forms the n-cube. This triangulation of the unit n-cube is called *Kuhn's triangulation* in [DM82]. Figure 1 illustrates Kuhn's triangulation of a cube.

The property of greatest interest here, however, is expressed in the following theorem.

Theorem 1 *An n-dimensional Q-simplex can be divided into 2^n identical Q-simplices, each of which has its longest edge parallel to the longest edge of the original.*

Proof: [DM82] shows that if Kuhn's triangulation is applied to an n-cube and copies of that cube are translated to form a conformal packing of space, then the resulting partition is a triangulation. Consider a packing of 2^n such translationally equivalent n-cubes around a point. The large cube containing the point has a Kuhn's triangulation, and the measure of each Q-simplex of this larger triangulation is exactly 2^n times the measure of the smaller Q-simplices. We claim that one large Q-simplex exactly contains 2^n of the smaller Q-simplices.

Let $[0,1]^n$ be the large n-cube and consider the Q-simplex Σ^n within it. The volume of Σ^n is defined by the inequalities $x_i \geq x_{i+1} (1 \leq i < n)$. Associate with each of the vertices V_j of Σ^n the subcube C_j that contains V_j. That is, $C_j = [0.5,1]^j \times [0,0.5]^{n-j}$. Thus, subcube

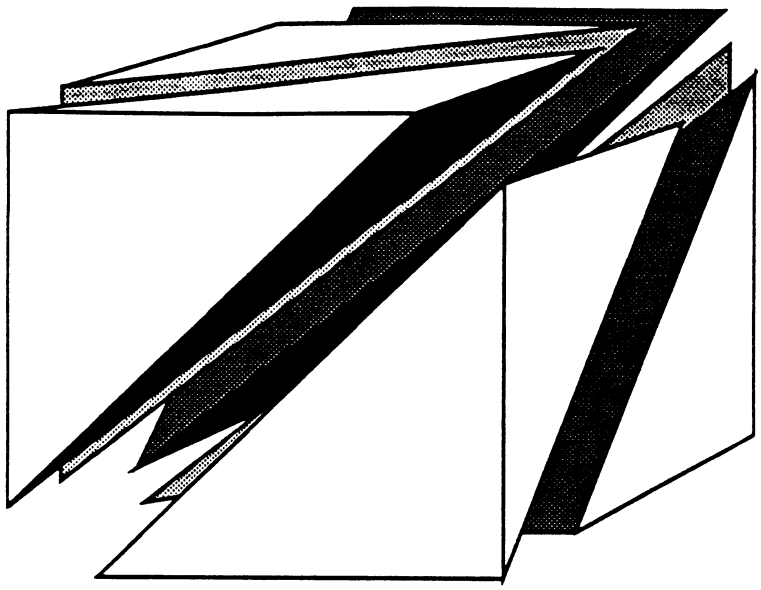

FIGURE 1: Kuhn's triangulation of a cube

C_j is bounded by the inequalities $x_i \geq 0.5$ $(0 < i \leq j)$ and $x_i \leq 0.5$ $(j < i \leq n)$. Over C_j, the set bounded by the inequalities $x_i \geq x_{i+1}$ $(0 < i < j)$ and $x_i \geq x_{i+1} (j < i < n)$ lies entirely within Σ^n, since $x_j \geq x_{j+1}$ for all points in C_j.

These two sets of inequalities define a *simploid* over C_j. A simploid is the product of two Q-simplices. After translation and scaling to make C_j coincide with the unit cube, the simploid defined by these inequalities is denoted $\Sigma^{j,n-j}$. [DM82] shows that a subset of Kuhn's triangulation of $[0,1]^n$ triangulates $\Sigma^{j,n-j}$. Therefore, a subset of Kuhn's triangulation of C_j is a triangulation of $C_j \cap \Sigma^n$. The union of the triangulations of the simploids forms a triangulation for Σ^n. Moreover, because all the Kuhn's triangulations are for identically oriented cubes, all the longest edges of all the subsimplices are aligned. □.

Figure 2 illustrates the three dimensional case.

The Q-simplex is not the only recursively decomposable simplex, nor is it the one of best shape. Several measures of the shape of a simplex have appeared; the *aspect ratio* as defined in [Fie86] seems most natural. The aspect ratio of a simplex is the ratio of the radius of an inscribed ball to the radius of a circumscribing ball, times the dimension. The aspect ratio of a regular simplex is one in all dimensions, and falls toward zero as the simplex becomes less regular.

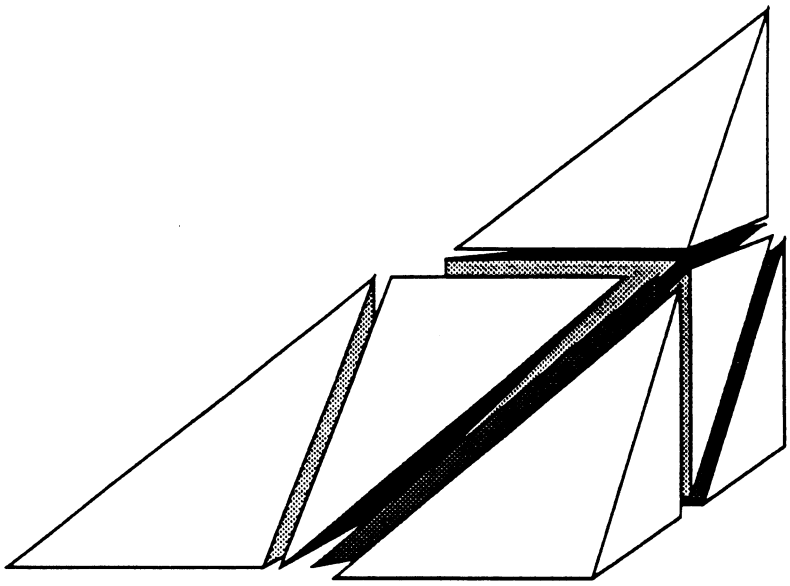

FIGURE 2: Simple symmetric subdivision of a Q-tetrahedron

The aspect ratio of a Q-simplex in n dimensions is given by

$$\frac{\sqrt{2n}}{n-1+\sqrt{2}},$$

and, in particular, the aspect ratio of a Q-tetrahedron is near 0.7174. However, because the longest edges of the subsimplices in a symmetric subdivision are parallel, a simple linear transformation can improve the aspect ratios of all the subsimplices, while maintaining their similarity to the original simplex. A scaling in the direction parallel to the longest edges preserves similarity, and a scaling by the optimal factor of $1/\sqrt{n+1}$ produces simplices with aspect ratios

$$\sqrt{\frac{6n}{(n+1)(n+2)}}.$$

We call a simplex produced by this scaling *rhombic*, because of the close geometric relationship between the optimally scaled tetrahedron and the rhombic dodecahedron. The aspect ratio of a rhombic tetrahedron is near 0.949.

The one-dimensional family of simplices that arise from stretching Q-simplices in the appropriate direction includes all the recursively decomposable tetrahedra that arise from *corner-chopping*; that is, each vertex of the parent tetrahedron is a vertex of exactly one subtetrahedron. In this family, each subsimplex is similar to its parent simplex, or to a

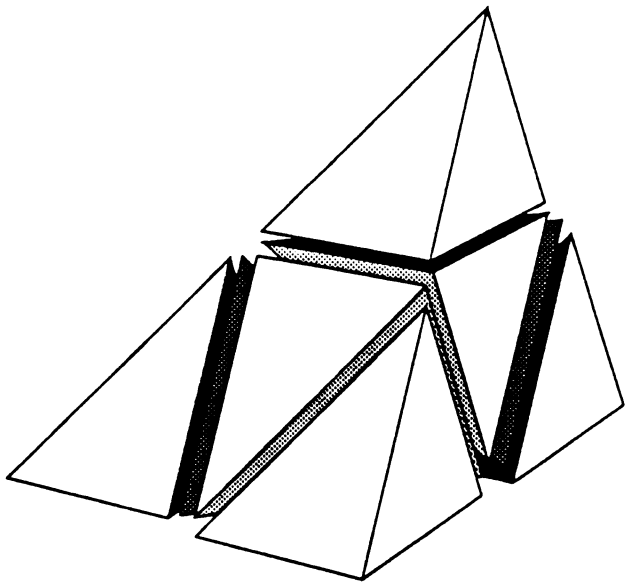

FIGURE 3: An optimally distorted Q-tetrahedron

reflection of the parent. The rhombic tetrahedron is the only recursively decomposable tetrahedron with the corner-chopping property for which the subtetrahedra are similar to the parent, without the need for reflection. It is also the tetrahedron of best aspect ratio in this family. These results are proved, and the formulas above are derived, in the Appendix.

The possibility of filling space with rhombic tetrahedra has been noted [Cox73], but has not appeared in the literature as a computational technique. Some methods for packing space [Fie86] create some elements with better aspect ratios than rhombic tetrahedra have and some elements with worse ratios. We know of no uniformly dense packing that has better aspect ratios for all its elements. Moreover, since the elements of the packing are recursively decomposable, they can form the basis for a tetrahedral octree that has many of the advantages of the familiar cubical octree, including adaptivity, and all of the advantages inherent in a tetrahedral partition, such as the existence of unique linear interpolants over elements. This construction generalizes easily to permit the creation of simplicial region quadtrees in n dimensions. We conjecture that the methods described above lead to the simplicial quadtrees with best element aspect ratios in all dimensions.

4 Maintaining and restoring balance in quadtrees

For a solid that is uniformly complex throughout its volume, a uniform quadtree provides an adequate approximation to the solid. However, many objects designed in practice have large regions of low complexity, where large elements provide an adequate approximation to the solid, and scattered areas of high complexity where small elements are needed. A general quadtree divides $\Re^n$ into similarly shaped elements of possibly different sizes. This is essential for the solution of finite element problems on large solids, where a sufficiently fine uniform quadtree could overwhelm the available computational resources and a too coarse uniform quadtree would provide insufficient accuracy.

The construction of a quadtree that adaptively approximates a solid requires an initial quadtree, possibly consisting of one element that contains the entire region of interest, and an error predicate that is true for an element exactly when the element is sufficiently small to accurately approximate the solid over its volume. The basic algorithm to construct the quadtree recursively subdivides any element that fails to satisfy the error predicate until all elements satisfy it. The quadtree that results can have very highly refined elements adjacent to coarsely refined elements.

A *balanced* quadtree is a quadtree in which corresponding edges of adjacent elements differ in length by at most a factor of two. This restriction prevents abrupt changes in element size. Balanced quadtrees have also been called *restricted* quadtrees [HB87]. Balanced simplicial quadtrees are particularly interesting because, as shown in section 5, they can easily be transformed into conformal simplicial partitions that have elements with good aspect ratios.

Either of two distinct modifications of the basic quadtree refinement algorithm produce balanced quadtrees. In the lazy-splitting algorithm, each refinement phase begins with a balanced quadtree and subdivides only those elements that fail the error predicate, without regard to balance. Then, additional elements are split in a rebalancing step to restore balance to the quadtree, and the cycle is repeated until the quadtree is sufficiently refined. This *Rebalancing method* generates a minimum balanced quadtree. It produces the same quadtree that results when the entire quadtree is constructed without regard for balance and then the minimum number of elements are split to restore balance.

The biggest apparent drawback to the Rebalancing method is that, in one rebalancing phase, splitting an element to restore balance may cause a chain reaction in which a sequence of other elements must also split. For example, in figure 4, the unit interval has been hierarchically subdivided at $1/2, 1/4, \ldots, 2^{-k}$, so that the corresponding one-dimensional quadtree is balanced. If the next level of refinement splits the interval $[2^{-k}, 2^{1-k}]$, a sequence of $k-1$ additional subdivisions is required to rebalance the quadtree. Fortunately, the set of all such chain reactions can be determined at once, efficiently, using a variant of breadth first search.

Let T denote a balanced quadtree whose elements have a maximum depth k, where an element of depth k is one that results from k refinements of an element in the original quadtree. Let C consist of those elements of T that fail to satisfy the error predicate; these elements are necessarily at depth k because elements at lesser depth that fail the error test would have been split at a previous iteration. The algorithm of figure 5 refines the balanced quadtree T, preserving balance and subdividing all elements of C. Figure 6 depicts the process of applying this method to a few elements in a balanced square quadtree.

The following theorem demonstrates the local optimality of the method.

Theorem 2 *Let T denote a balanced quadtree whose elements have a maximum depth k. Let C consist of those elements of T that fail the error predicate. The procedure* RebalancingSplit *replaces T by a quadtree T' which is the smallest balanced quadtree that is a refinement of T and has all elements in C subdivided.*

Proof: From the construction, it is clear that T' is contained in the original quadtree

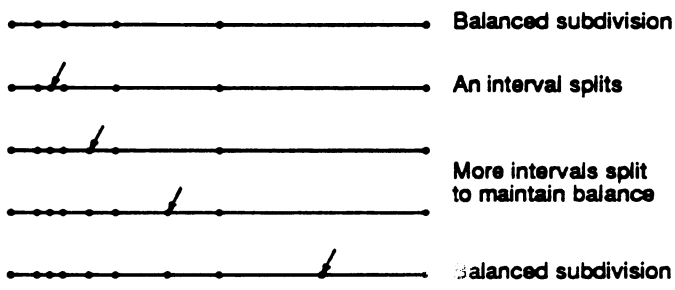

FIGURE 4: Rebalancing can require unbounded splitting of elements

```
RebalancingSplit(T: Quadtree, SplitSet: ElementSet, k: integer:)
/* T is a balanced quadtree of depth k */
/* SplitSet is the set of elements at depth k that fail the error predicate */
/* SplitNext is a set of elements that must split for balance */

    for j = k downto 1
        SplitNext = ∅
        for each element E ∈ SplitSet
            T = T − E ∪ Subdivide(E)
            for each element F ∈ T that shares a vertex with E
                if depth of F = j − 1
                    /* F must be split to maintain balance */
                    SplitNext = SplitNext ∪ F
        SplitSet = SplitNext
```

FIGURE 5: The Rebalancing method for balanced quadtrees

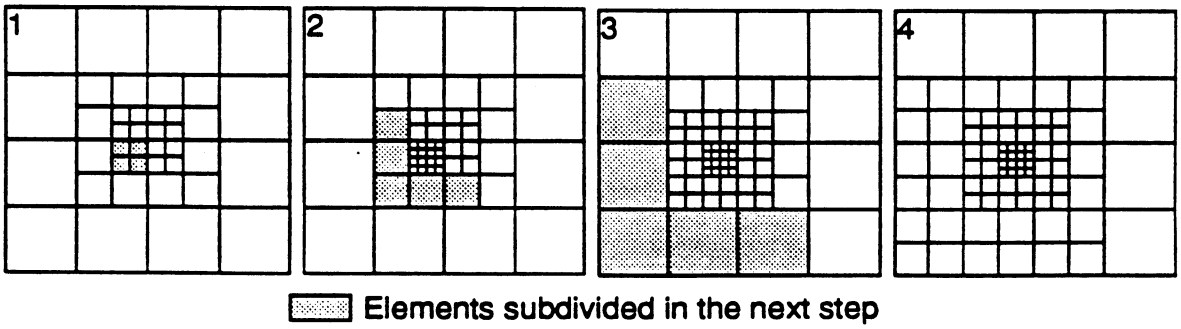

FIGURE 6: Rebalancing a quadtree after one level of adaptive subdivision

T and contains all elements that result from the subdivision of elements in C. To show that T' is balanced, assume oppositely that for two vertex-adjacent elements E and F in T', $depth(F) > depth(E) + 1$. Since T is balanced, F and E cannot both be in T; evidently, the parent of F was subdivided, and was vertex-adjacent to E. No other explanation is possible since no element is subdivided more than once in a single rebalancing phase. This behavior is inconsistent with the algorithm, however, because the subdivision of the parent of F causes E to subdivide when $depth(F) = depth(E) + 1$. By contradiction, then, T must be balanced.

Moreover, T' is the smallest quadtree with this property. Any element of depth j that appears in T', but not T, must be vertex-connected to the child of an element of C by a chain of $k - j + 1$ elements, whose depths increase sequentially. Thus, the parent of this element had to split to maintain balance in the quadtree. □.

The alternative, eager-splitting algorithm avoids the cascading sequences of element splits that may occur in the Rebalancing method. This *Buffer method* maintains two sets of elements at the most refined level of the quadtree, a core of elements that subdivide to achieve greater accuracy, and, surrounding the core, a buffer of elements that subdivide only to maintain continuity. The buffer elements prevent sudden transitions in the size of elements in the quadtree. The Buffer method builds the balanced quadtree one level at a time. The method takes as input two sets of elements at level k, the level k core elements, all of which fail the error predicate, and the level k buffer elements, which satisfy the error predicate. The algorithm of Figure 7 splits all of the core elements, and some of the buffer elements when necessary to preserve balance.

Figure 8 shows an example of the method in action. In each frame, the shaded elements fail the error predicate. Those elements and their neighbors split, while the remaining elements remain intact and may be ignored at deeper levels. Note that unshaded elements are only subdivided to maintain continuity when neighboring shaded elements subdivide. The unshaded elements are already known to provide sufficiently accurate approximations.

Theorem 3 *Call the error predicate* e hereditary *if* $e(S) \Rightarrow e(E)$ *for any child* E *of* S. *If the error predicate is hereditary, then the Buffer method produces a balanced quadtree.*

BufferedSplit(T: Quadtree, C, B: ElementSet)
/* T is a balanced quadtree of depth k */
/* C is the set of elements at depth k that fail the error predicate */
/* B is the set of elements at depth k that satisfy the error predicate */

for each element $E \in C$
    $T = T - E \cup Subdivide(E)$

for each element $E \in B$ that shares a vertex with an element of C
    $T = T - E \cup Subdivide(E)$

FIGURE 7: The Buffer method for balanced quadtrees

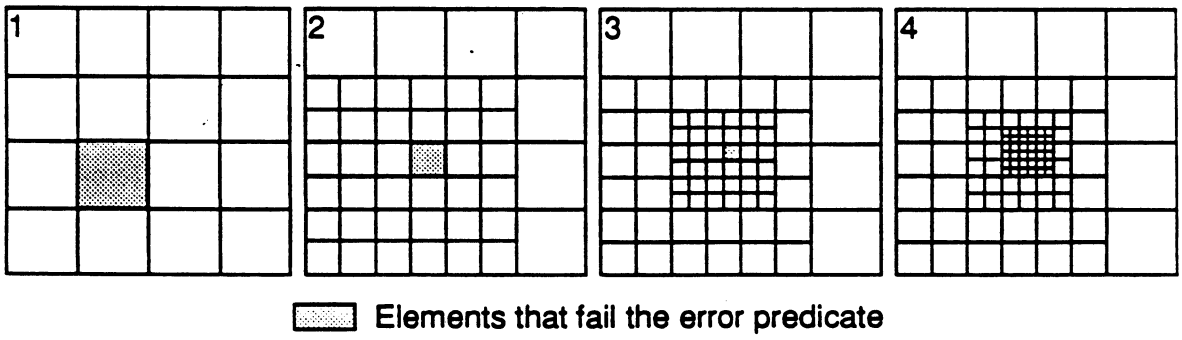

FIGURE 8: Several iterations of Buffered adaptive subdivision

Proof: The proof is by induction on k, the depth of the tree. Let B and C denote the buffer and core elements when the tree has depth k, and let D denote the rest of the elements. Let A denote the elements of B adjacent to elements of C. Finally, let A',B',C' and D' denote the corresponding sets in the depth $k+1$ tree. The inductive claim is that at the k^{th} step of the algorithm, T is a balanced quadtree, and that no element of A is adjacent to an element of D. Initially, take the core C as the single element of the tree. Then the tree is balanced, and the inductive claim holds.

In one refinement step of the Buffer method, elements of $A \cup C$ subdivide. No element that subdivides is adjacent to D, by the induction hypothesis. Thus, elements that subdivide have as neighbors only elements that also subdivide, and elements of $B - A$ that are the same size. The new elements that exist after subdivision have edges no less than half as long as the edges of some neighbors, so balance is preserved.

Because the error predicate is well behaved, the elements of C' are a subset of the children of elements of C. From the algorithm, $B' = \text{child}(C \cup A) - C'$. A child of an element $E \in A$ splits to form smaller elements, but none of the smaller elements is both adjacent to a child of an element of C and adjacent to an element of $B - A$. The elements of $B - A$ are elements of D', so none of the smaller elements is adjacent both to an element of C' and to an element of D'. □.

Each of the methods has strengths and weaknesses. The rebalancing method yields the minimum balanced quadtree contained in a given general quadtree. The Buffer method, by splitting extra elements to maintain the buffer, can produce a larger number of elements. Initial experience with implementations of each method indicates that, in graphics applications, the Buffer method typically creates $40 - 50\%$ more elements than the Rebalancing method. The Rebalancing method is also more flexible than the Buffer method, since it is can easily be extended to allow the subdivision of large elements as well as small ones. This could be helpful in the event that the data is ill-behaved and a particular element that satisfies the error predicate is split to form some elements that do not satisfy it. In such a circumstance, the decision not to subdivide an element must be reversible.

On the other hand, the Buffer method is local; an element subdivides based only on information about itself and its neighbors. This property can be useful for a parallel im-

plementation, since it limits interprocessor communication. Moreover, once all elements at depth k have been adaptively split the elements at depth $k-1$ or less are never subdivided again. This allows finite element computations, for example, to begin on coarser elements before the subdivision of the finest elements is complete. In some graphics applications, the portions of the quadtree at depth $k-1$ or less can be processed and discarded immediately.

5 Producing conformal simplicial quadtrees

Either the two previous methods can be used to create balanced quadtrees in arbitrary dimensions. However, many applications require a mesh of conformal elements. This section describes a new algorithm for converting a balanced simplicial quadtree into a *conformal simplicial quadtree.* In a general quadtree of n dimensions, each non-leaf node has 2^n children, but in a conformal quadtree, a non-leaf node may have 2^m children, $m \leq n$, if all the children are leaves. The elements form a conformal mesh, so that the intersection of any two elements in the tree is a proper subface of each. The method is dimension independent.

The procedure is called *vertex marking*, and works as follows. Given a balanced quadtree, mark, for each element in the tree, any vertex of that element that coincides with a vertex of a smaller element. If $m+1$ vertices of an element E are marked, then these $m+1$ vertices define a m dimensional subface F of E. Next, recursively decompose F into 2^m subsimplices of dimension m by restricting the full recursive subdivision of E to F. Finally, decompose E into 2^m subsimplices of dimension n by projecting from the $n-m$ unmarked vertices of E to each of the m dimensional subsimplices. Figure 9 illustrates an example of this method applied to a balanced triangular quadtree.

Theorem 4 *Given a balanced simplicial quadtree, the vertex marking method produces a conformal quadtree.*

Proof: The proof is by induction on the depth k of the quadtree. We consider a sequential implementation of the method, in which all vertices of the smallest elements of the tree are marked, and elements of the next higher level are split according to those markings, then marked themselves. Initially, the subset of elements at depth k forms a conformal partition, and all such elements have marked vertices.

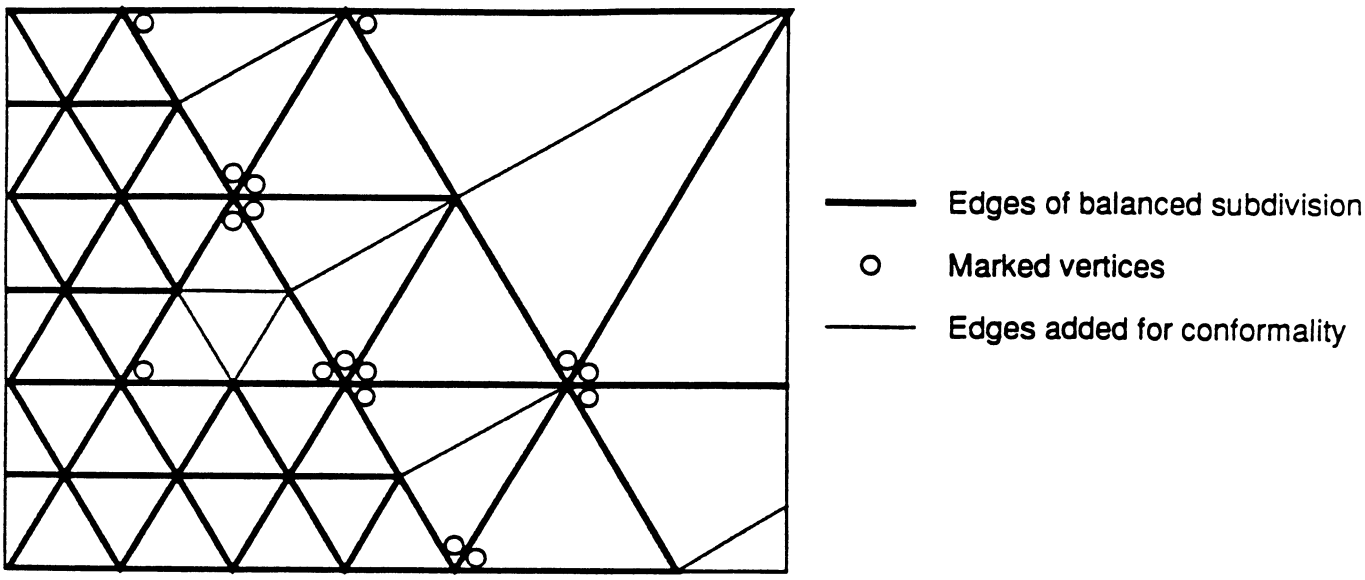

FIGURE 9: Conformality using vertex marking

Suppose, then, that the elements at depths greater than i form a conformal partition, that all vertices of those elements are marked, and that elements at depths up to i form a balanced partition. Let F be an m-dimensional simplex that is a face of at least one element E at depth $i-1$ and properly contains m-dimensional faces of elements $G_1, \ldots, G_p$ at depth i. Since all the vertices of G_i are marked, all the vertices of F are marked as well. These elements G_i are children of elements that contained F as a subface. Vertex marking subdivides F into 2^m subfaces. Because the decomposition of a simplex and its subfaces is symmetric, the subdivision of F produces the same decomposition now as it did when the elements G_i were created, and the children of E meet the corresponding G_i conformally.

No vertex at level $i-2$ is marked at this stage, because level $i-1$ as a whole has not been marked. Of the level $i-1$ vertices, only those also at level i have been marked, and the balance of the partition prevents a vertex from being part of level i and $i-2$ simultaneously. Therefore, levels 0 to $i-1$ of the partition remain balanced.

Finally, after marking all the vertices of all the level $i-1$ elements, the conditions necessary to apply vertex marking to the next level of the quadtree are satisfied and the part already processed forms a conformal partition. □.

The vertex marking method subdivides any element of dimension m into at most 2^m

subelements during conformality conversion. Methods that introduce extra vertices at the centroids of subfaces and project from centroids to lower dimensional subfaces often generate many more elements. For example, the method of [HW88] may subdivide a triangle into as many as five subtriangles and a tetrahedron into as many as 20 subtetrahedra during conformality conversion. The vertex marking method also produces very few distinct shapes, as figure 10 illustrates.

6 Conclusions and Future Work

This paper has described dimension-independent methods for generating subdivisions of space. Although the primary focus has been upon simplicial subdivisions, some of the methods are shape-independent as well. That is, any recursively decomposable shape can serve as the standard element. We believe that practical, efficient procedures to automatically generate good meshes result from the application of these ideas.

A number of mathematical questions arise from this work. For example, what families of simplices besides the ones discussed are recursively decomposable? If other such families exist, do any include simplices with better aspect ratios? We conjecture that no other

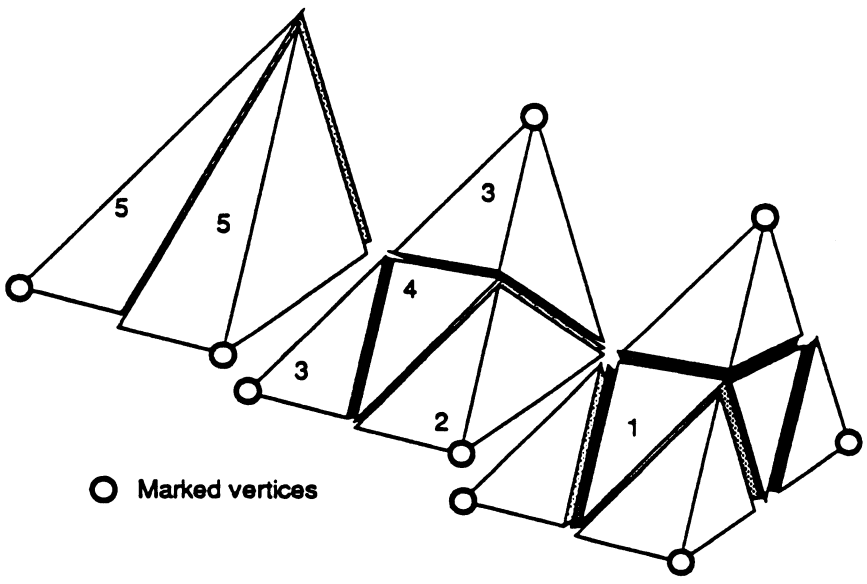

FIGURE 10: Space can be adaptively triangulated with five shapes.

families of recursively decomposable simplices exist in dimensions higher than two, but a mathematical proof of this conjecture would be most satisfying.

It would also be interesting to study the relationship between the shape of an arbitrary simplex and the shapes of its components when it is subdivided by one of our algorithms. There is more than one way to map an arbitrary simplex to a Q-simplex, and one of these mappings is likely to be better than the others. An algorithm to determine the best such mapping would be a worthy goal.

Finally, we hope soon to extend vertex marking to n-cubes. Many applications use cubical meshes and they will certainly remain popular. A dimension-independent way to irregularly divide an n-cube into smaller topologically equivalent elements, bounded by multilinear surfaces, could be valuable in those applications that manipulate higher dimensional data.

The authors are grateful to Ron Goldman for his careful reading and constructive criticism of an earlier draft of this paper.

Appendix - The subdivision of simplices

Theorem 5 *The aspect ratio of a Q-simplex in n dimensions is given by*

$$AR_n = \frac{\sqrt{2n}}{n-1+\sqrt{2}}.$$

In particular the Q-tetrahedron has an aspect ratio near 0.7174.

In the following, we assume that the Q-simplex Σ^n is defined over the n-cube $[-1,1]^n$, rather than over $[0,1]^n$.

Proof: The circumcenter of Σ^n is the origin and its circumradius is $r_{circum} = \sqrt{n}$. The incenter $(c_0, \ldots, c_{n-1})$ lies equidistant from each of the hyperplanes that bound the Q-simplex, and that distance is the inradius r_{in} of the Q-simplex. Formally, the constraints are

$$\begin{aligned} r_{in} &= 1 + c_{n-1} \\ &= (c_i - c_{i+1})/\sqrt{2}, \forall i \in 0, \ldots, n-2 \\ &= 1 - c_0 \end{aligned}$$

which suggests that the c_i form an arithmetic sequence symmetric about zero. The substitution $c_i = (i - (n-1)/2)d$ reduces the constraints to two,

$$\begin{aligned} r_{in} &= 1 + d(n-1)/2 \\ r_{in} &= -d/\sqrt{2} \end{aligned}$$

and leads to the conclusion that $r_{in} = \frac{\sqrt{2}}{n-1+\sqrt{2}}$. Dividing r_{in} by r_{circum} and multiplying the result by n yields $\frac{\sqrt{2n}}{n-1+\sqrt{2}}$, as stated. □.

Stretching or shrinking a Q-simplex in the direction of its longest edge changes its shape, and thus its aspect ratio, but does not change its recursive decomposability. That is because all the similar subsimplices have longest edges parallel to the longest edge of their parent, from the construction. Some particular amount of stretching maximizes the aspect ratio and yields much improved recursively decomposable simplices. In particular, the Q-tetrahedron may be compressed by half along its longest edge to form a particularly well-shaped tetrahedron, a tetragonal disphenoid [Cox73] with aspect ratio near 0.949.

Theorem 6 *Let $s_n = (\alpha - 1)/n$. Let $S_n(\alpha)$ denote the $n \times n$ matrix*

$$\begin{pmatrix} 1+s_n & s_n & \dots & s_n & s_n \\ s_n & 1+s_n & \dots & s_n & s_n \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ s_n & s_n & \dots & 1+s_n & s_n \\ s_n & s_n & \dots & s_n & 1+s_n \end{pmatrix}.$$

For any vector v, $S_n(\alpha)v$ is a translation of v parallel to $(1\ 1\ \dots\ 1)$, and $S_n(\alpha)(1\ 1\ \dots\ 1) = \alpha(1\ 1\ \dots\ 1)$.

Proof: For any vector v, $S_n(\alpha)v - v = s_n(\sum_i v_i)(1\ 1\ \dots\ 1)$. The second part of the theorem follows from simple linear algebra. □.

The matrix $S_n(\alpha)$ is the stretching transformation described above. It remains to choose the best value of α for a particular dimension.

Under a stretching transformation, the constraints that determine the incenter change only slightly. All the planes that bound the simplex remain unchanged, except that $1+c_0 = 0$

becomes

$$1 + c_0 - \frac{\alpha - 1}{\alpha n} \sum_{i=0}^{n-1} c_i = 0$$

and $1 - c_{n-1} = 0$ becomes

$$1 - c_{n-1} + \frac{\alpha - 1}{\alpha n} \sum_{i=0}^{n-1} c_i = 0.$$

As a result, the constraints that determine the incenter and inradius are

$$\begin{aligned} r_{in} &= (1 - c_{n-1} + \frac{\alpha - 1}{\alpha n} \sum_{i=0}^{n-1} c_i)/\sqrt{1 - 1/n + 1/\alpha^2 n} \\ &= (c_i - c_{i+1})/\sqrt{2}, \forall i \in 0, \ldots, n-2 \\ &= (1 + c_0 - \frac{\alpha - 1}{\alpha n} \sum_{i=0}^{n-1} c_i)/\sqrt{1 - 1/n + 1/\alpha^2 n}, \end{aligned}$$

and again the coordinates c_i form a arithmetic sequence symmetric about the origin. The substitution $c_i = (i - (n-1)/2)d$ gives

$$\begin{aligned} r_{in} &= (1 + d(n-1)/2)/\sqrt{1 - 1/n + 1/\alpha^2 n} \\ r_{in} &= -d/\sqrt{2}, \end{aligned}$$

which has the solution $r = \frac{\sqrt{2}}{n-1+\sqrt{2(1-1/n+1/\alpha^2 n)}}$.

Having determined the inradius for the stretched Q-simplex, it remains to calculate the circumradius. The circumcenter c is given by $c_i = (1-\alpha^2)(1-(2i+1)/n)$ and circumradius by $\sqrt{(1-\alpha^2)^2(n/3 - 1/3n) + \alpha^2 n}$. The maximizing choice $\alpha = 1/\sqrt{1+n}$ gives a simplex with aspect ratio $\sqrt{\frac{6n}{(n+1)(n+2)}}$. The proof of these statements follows from elementary, if tedious, algebra and calculus, and is omitted. Although the aspect ratio does fall toward zero as the dimension grows, the aspect ratios for the first few dimensions are quite good.

Uniqueness of the decomposition

The decomposition of an element has the "corner-chopping" property when any vertex of that element is a vertex of exactly one of its descendents. The symmetric decomposition of the Q-simplex has the corner-chopping property, as can be seen by considering the triangulation of each of the n simploids that form the decomposition. In each simploid, a copy of standard Q-simplex Σ^n appears, and in the j-th simploid, the j-th vertex of Σ^n coincides with a vertex of the larger Q-simplex.

Corner-chopping is a symmetry property of the subdivision. A final theorem asserts the uniqueness of such symmetric subdivisions of tetrahedra.

Theorem 7 *The set of stretched Q-tetrahedra contains all the tetrahedra which can be recursively divided by corner-chopping into tetrahedra that are equivalent under translation, rotation and reflection. Among the stretched Q-tetrahedra, only a rhombic tetrahedron can be recursively divided by corner-chopping into tetrahedra that are equivalent under translation and rotation.*

Proof: Label the vertices of a tetrahedron $V_0, \ldots, V_3$, and let E_{ij} denote half the length of the edge from V_i to V_j. Let V_{ij} denote the midpoint of the edge V_iV_j. The interior of the tetrahedron can be divided into four similar *outer* tetrahedra, with edge lengths $E_{ij}, 0 \leq i < j \leq 3$, and an octahedron. Figure 11 illustrates this decomposition.

The volume of this octahedron is equal to four times the volume of an outer tetrahedron, so a recursive decomposition based on corner-chopping must divide the octahedron into four *inner* tetrahedra. One way to divide the octahedron is to pick two nonadjacent vertices, V_{02} and V_{13} for example, and consider each of the edges of the parallelogram $V_{01}V_{12}V_{23}V_{03}$. Each pair of points at the ends of an edge of the parallelogram, together with V_{02} and V_{13}, lies at the vertices of a tetrahedron, and the four tetrahedra so constructed fill the interior of the octahedron. Either of the other pairs of nonadjacent vertices could be picked instead, but the three resulting subdivisions of the octahedron are topologically identical. No other subdivision into as few as four similar tetrahedra is possible, because there is no way to pick four vertices of the octahedron to make a nondegenerate tetrahedron, except the ways that induce one of the three triangulations already mentioned.

For each of the four inner tetrahedra, five of its edge lengths are known to be E_{ij} for some i and j, and the sixth is the length of the edge $V_{02}V_{13}$. Each of the four has edges of length E_{01}, E_{12}, E_{23} and E_{03}, but two have a fifth edge specified to be of length E_{02}, while the known length in the other two is E_{13}. From this, it follows that the inner and outer tetrahedra can be identical only if the length of the edge $V_{02}V_{13} = E_{02} = E_{13}$. In what follows, we call this constant edge length b.

The constraint above is sufficient to ensure that each of the inner tetrahedra has edge

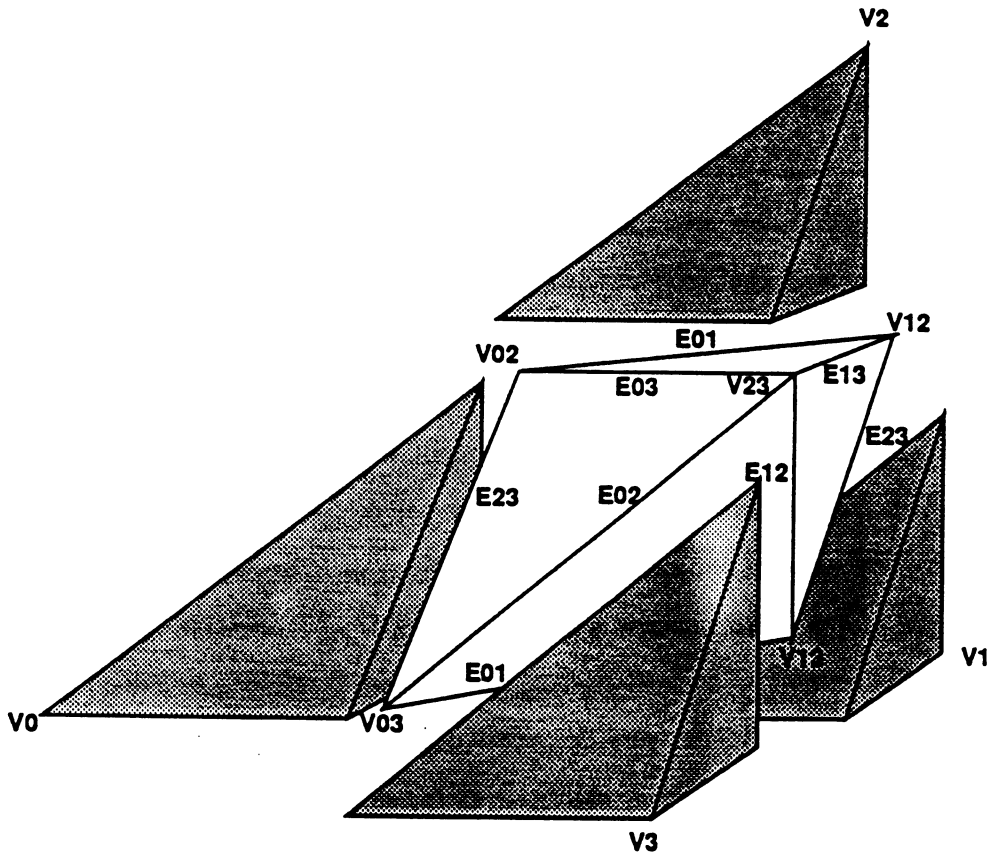

FIGURE 11: Tetrahedron = 4× tetrahedron + octahedron

lengths $\{E_{ij}\}, 0 \leq i < j \leq 3$, but it is not sufficient to guarantee that such a tetrahedron is similar to the outer tetrahedra. In the outer tetrahedra, the edges of length b are not adjacent, and the edges of lengths $E_{01}, E_{12}, E_{23}, E_{03}$ appear in cyclic order to form a skew quadrilateral. In the inner tetrahedra, the edges of length b are again not adjacent, but the other edge lengths appear in either of two different cyclic orders around a skew quadrilateral. Those orders are $E_{23}, E_{01}, E_{12}, E_{03}$ and $E_{12}, E_{23}, E_{01}, E_{03}$. These three distinct cyclic orders must describe identical quadrilaterals if the subdivision is to produce identical tetrahedra.

Let c denote the value E_{03} and let c be distinct from the other three edge lengths. The other three values must be indistinguishable, because each appears as the edge length opposite c in one of the cyclic orderings of the edge lengths. Let a denote this common edge length. Figure 12 reflects the new edge labeling, based on the conclusions reached so far.

The set of tetrahedra we have left to consider has three degrees of freedom; one represents

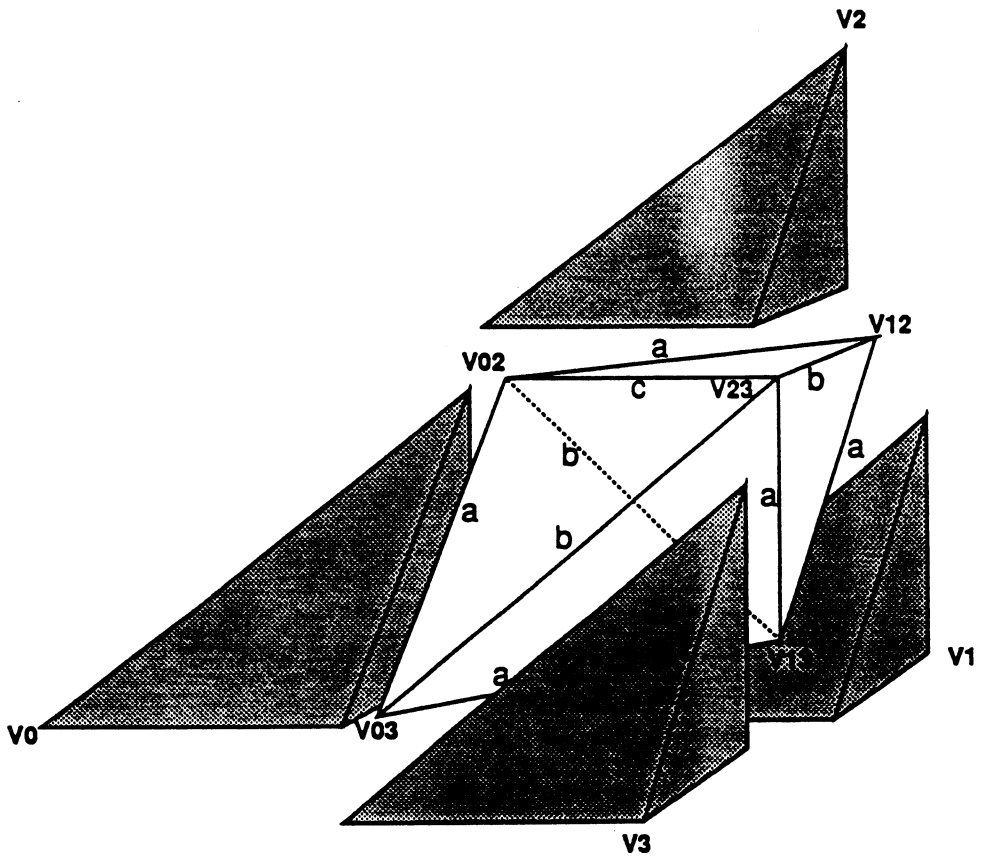

FIGURE 12: An octahedron dividing into tetrahedra

a uniform scaling factor and disappears if we set $a = 1$. Fixing a value for another, b for example, uniquely determines c. Therefore, there is only one degree of freedom left to choose a recursively decomposable tetrahedron. The stretched Q-tetrahedra form a one-parameter family of recursively decomposable tetrahedra, which includes tetrahedra with all possible aspect ratios. It follows that these two families of tetrahedra are the same.

Geometers who have studied the question of which tetrahedra fill space distinguish between those that require reflections to map one element to another, and those which do not [Sen81]. The tetrahedra that satisfy the constraints developed above can only be said to fill space in the weaker sense.

Suppose, for example, that the edge lengths a, b, and c are distinct. In that event, each tetrahedron has two faces that are scalene triangles. An examination of figure 12 reveals that a traversal of the edges of such a triangle in the order abc can be a clockwise traversal, when viewed from outside the tetrahedron, or a counterclockwise traversal. However, on any one tetrahedron the traversals of the scalene triangles are oriented similarly. That tetrahedron cannot be affixed face to face with an identical tetrahedron along that face, only with a tetrahedron that has a similar triangle of opposite orientation. Therefore, a recursively decomposable tetrahedron with a scalene face cannot fill space in the stricter sense.

That leaves only the possibility that a recursively decomposable tetrahedron with isoceles faces can fill space in the strict sense. The rhombic tetrahedron has isoceles faces and is therefore identical to its reflection. Thus, it is the unique, space-filling recursively decomposable tetrahedron. □.

References

[Ban90] R. E. Bank. *PLTMG User's Guide.* SIAM, 1990.

[BEG90] Marshall Bern, David Eppstein, and John Gilbert. Provably good mesh generation. In *31st IEEE Symposium on Foundations of Computer Science.* IEEE, 1990.

[Ben75] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, September 1975.

[BGR88] B. S. Baker, E. Grosse, and C. S. Rafferty. Nonobtuse triangulation of polygons. *Discrete and Computational Geometry*, 3:147–168, 1988.

[CFF85] J. C. Cavendish, David Field, and W. B. Frey. An approach to automatic three dimensional finite element mesh generation. *International Journal for Numerical Methods in Engineering*, 21:329–347, 1985.

[Cox73] H. S. M. Coxeter. *Regular Polytopes.* Dover, New York, third edition, 1973.

[DM82] Wolfgang A. Dahmen and Charles A. Micchelli. On the linear independence of multivariate B-splines, I. triangulations of simploids. *SIAM Journal of Numerical Analysis*, 19(5):993–1012, October 1982.

[FF85] David A. Field and W. H. Frey. Automation of tetrahedral mesh generation. Technical Report GMR-4967, General Motors Research Laboratories, Warren, Michigan, 1985(?).

[Fie86] David A. Field. Implementing watson's algorithm in three dimensions. In *Second ACM Symposium on Computational Geometry*, pages 246–259, Yorktown Heights, New York, 1986. ACM.

[FKN80] H. Fuchs, Z. M. Kedem, and B. F. Naylor. On visible surface generation by a priori tree structures. *Computer Graphics*, 14(3):124–133, July 1980.

[HB87] B. Herzen and A. Barr. Accurate triangulations of deformed intersecting surfaces. *Computer Graphics*, 21(4):103–110, 1987.

[HW88] Mark Hall and Joe Warren. Adaptive tessellation of implicitly defined surfaces. Technical Report TR-88-84, Rice University, Department of Computer Science, 1988.

[Joe84] B. Joe. *Finite Element Triangulation of Complex Regions using Computational Geometry.* PhD thesis, University of Waterloo, 1984.

[MW90] Doug Moore and Joe Warren. Adaptive mesh generation II: Packing solids. Technical Report TR 90-139, Rice University, Department of Computer Science, 1990.

[Riv87] M-C Rivara. A grid generator based on 4-triangles conforming mesh-refinement algorithms. *International Journal for Numerical Methods in Engineering*, 24:1343–1354, 1987.

[Sam90] Hanan Samet. *The Design and Analysis of Spatial Data Structures.* Addison Wesley, 1990.

[Sen81] Marjorie Senechal. Which tetrahedra fill space? *The Mathematics Magazine*, 54(5):227–243, November 1981.

[Smi88] W. D. Smith. *Studies in Discrete and Computational Geometry.* PhD thesis, Princeton University, 1988.

[TW88] Robert E. Tarjan and Christopher J. Van Wyk. An $O(n \log \log n)$-time algorithm for triangulating a simple polygon. *SIAM Journal on Computing*, 17(1):143–178, February 1988.

[YS84] M. A. Yerry and M. S. Shephard. Automatic three-dimensional mesh generation by the modified octree technique. *International Journal for Numerical Methods in Engineering*, 20:1965–1990, 1984.