Multiresolution Tetrahedral Meshes: an Analysis and a Comparison

Emanuele Danovaro, Leila De Floriani
Dipartimento di Informatica e Scienze dell'Informazione,
Università di Genova, Via Dodecaneso 35, 16146 Genova - Italy
{danovaro,deflo}@disi.unige.it

Michael Lee, Hanan Samet
Computer Science Department - Center for Automation Research
Institute for Advanced Computer Studies
University of Maryland, College Park, Maryland (USA) 20742
{magus,hjs}@cs.umd.edu

Abstract

The paper deals with the problem of analyzing and visualizing large-size volume data sets. To this aim, we consider multiresolution representations based on a decomposition of the field domain into tetrahedral cells. We compare two types of multiresolution representations that differ on the rule applied to refine an initial coarse mesh: one is based on tetrahedron bisection, and one based on vertex split. The two representations can be viewed as instances of a common multiresolution model, that we call a multiresolution mesh. Encoding data structures for the two representations are briefly described. An experimental comparison on structured volume data sets is presented.

1 Introduction

Several applications, including scientific visualization, medical imaging, and finite element analysis, deal with increasingly large volume data sets. A volume data set consists of a set of points in the three-dimensional Euclidean space where a value for a scalar field is associated with each point. Such data is often modeled by a mesh consisting of tetrahedral cells. A tetrahedral mesh is called regular when generated by a recursive decomposition on the points of a regular grid; it is called irregular otherwise.

In order to analyze volume data sets of large size and to accelerate rendering, multiresolution models have been proposed. Such models basically encode the steps performed by a refinement process applied to a coarse mesh, or by a decimation process applied to the mesh at full resolution (connecting the original data points) in a compact structure. In this way, a virtually continuous collection of simplified meshes at different Levels-Of-Detail (LODs) can be extracted. The resolution (i.e., the density of the cells) of an approximating mesh may vary in different parts of the field domain, or in the proximity of interesting field values. This enables the user not only to interactively explore large volume data using simplified approximations, but also to inspect specific areas of interest. The process of extracting meshes at a variable resolution from a multiresolution model is called *selective refinement*. It is the basic operation which must be efficiently supported by any multiresolution data structure.

In the computer graphics and finite element literature, much research has been devoted to nested tetrahedral meshes generated by recursive decomposition, which are suitable to deal with regularly distributed data points (usually called *structured data sets*). In the following, we will focus on nested meshes based on tetrahedron bisection, and we will use the term *Hierarchy of Tetrahedra (HT)* to describe them. Such meshes, introduced in finite element analysis, are an excellent basis for multiresolution representation of fields defined at the vertices of a regular grid, since several continuous linear approximations of the underlying field, with a level of detail varying in different parts of the domain, can be extracted from them.

Multiresolution models based on irregular tetrahedral meshes are desirable to deal with irregularly-distribuited data points (which define the so-called *unstructured data sets*) since they are adaptive and thus, they can capture the shape of the field domain accurately even at the lowest resolution. However, not much research has been performed on such models. To our knowledge, the only multiresolution model based on irregular meshes is the *Edge-based Multi-Tessellation* introduced in [2], which is built through an edge-collapse simplification process. This model specializes a continuous multiresolution model for representing k-dimensional spatial entities through simplicial complexes called a *Multi-Tessellation* (MT) [4].

The contribution of this paper is in analyzing and comparing regular and irregular multiresolution models based on tetrahedral decompositions. In particular, we focus on two models, the Hierarchy of Tetrahedra and the Edge-based Multi-Tessellation, and we discuss them within a common framework, that of multiresolution meshes, which allows dealing with the two models and analyzing them in a unifying way. We show that an HT and an edge-based MT are two instances of a multiresolution mesh. We perform an experimental comparisons of the two models based on a set of queries for analyzing and rendering a volume data set at a variable resolution.

We consider our implementation of the HT based on an ordering of the tetrahedra in such a way that it becomes possible to find not just the children and the parent of a given tetrahedron, but also neighboring tetrahedra using simple arithmetic and bit-wise operations [14]. This is the basic tool for performing selective refinement efficiently. We consider a compact implementation of an edge-based MT which is even more economical than encoding the mesh at full resolution [2].

The rest of this paper is organized as follows. Section 2 reviews some related work. Section 3 discusses some background notions. In Section 4, we discuss a general framework to describe multiresolution meshes, and we define the HT and the edge-based MT within such framework. In Sections 5.1 and 5.2, respectively, the encoding data structures for an edge-based MT and an HT are briefly described. In Section 6, an experimental comparison between the two models is presented. Concluding remarks are drawn in Section 7.

2 Related Work

Multiresolution models based on nested regular meshes. In the computer graphics literature, there has been a burst of research on nested tetrahedral meshes generated by recursive bisection of tetrahedra (see, for instance, [14, 16, 25]), or on the so-called red/green tetrahedron refinement (see, for instance, [10]). Such meshes have been introduced for domain decomposition in finite element analysis [11, 15, 19]. A nested decomposition of the underlying space is also at the heart of methods for isosurface extraction based on octrees (see for instance [21, 24]).

An important issue in using nested tetrahedral meshes is that if the domain is refined adaptively, the field associated with the extracted mesh (and, thus, the resulting isosurfaces) may present discontinuities in areas of transition. Different authors have proposed different solutions to this problem, including error saturation [25], re-meshing [10], insertion of points [21]. In [14], the continuity of the field associated with the extracted mesh is ensured by efficiently extracting meshes without cracks through worst-case constant time neighbor finding techniques. Hierarchical tetrahedral meshes based on tetrahedron bisection allow also an easy and space/time-efficient progressive encoding of isosurfaces at a variable resolution [17].

Tetrahedral mesh simplification. The problem of simplifying an irregular mesh has been extensively studied for triangle meshes (see, e.g., [8] for a survey). Several methods are based on *incremental* techniques, which perform a sequence of atomic modifications on a given mesh by either removing details from a mesh at high resolution, or adding details to a coarse mesh. Some incremental techniques have been proposed in the three-dimensional case for simplification of tetrahedral meshes [3, 9, 20, 23]. Most of such techniques are based on edge collapse and differ in the way they control the error for producing a simplified mesh.

Multiresolution models based on irregular meshes. There are several proposals for multiresolution models based on irregular triangle meshes, in particular capable of supporting selective refinement, (i.e., of extracting meshes at a variable resolution). This capability derives from organizing updates according to a partial order based on a dependency relation, so that a virtually *continuous* set of representations, in which the resolution may vary in different parts of the domain, can be extracted (see [5] for a survey).

Less work has been done on multiresolution models based on irregular tetrahedral meshes. The simplest multiresolution models based on irregular meshes are the so-called *progressive models*. They encode a coarse mesh plus a linear sequence of updates that can be applied to such a mesh in order to progressively refine it [9, 18]. These models support the extraction of a mesh only at those intermediate resolutions which can be obtained by truncating the sequence of refinements at some point. In [1], a multilevel tetrahedral mesh representation has been defined, which encodes a pyramid of mesh approximations, but allows extracting only meshes at uniform resolution.

3 Background

In this Section, we review some concepts that we use throughout the paper. A *volume data set* S consists of a set V of points in the three-dimensional Euclidean space, and of a field value f associated with such points. The data points can be regularly, or irregularly spaced over the domain, and, thus, we talk about *structured* and *unstructured* data sets. The domain D of a volume data set is either the convex hull of V, or it can be of any arbitrary shape (as it is often the case for unstructured data sets), but we assume that it is a three-dimensional manifold.

A *tetrahedral mesh* Σ is a connected set of tetrahedra so that the union of all tetrahedra in Σ covers D and any two distinct tetrahedra have disjoint interiors. A tetrahedral mesh Σ is called a *conforming* mesh if the intersection of the boundaries of any two tetrahedra of Σ, which have a non-empty intersection, consists of lower dimensional simplexes that belong to the boundary of both tetrahedra. Conforming meshes have a well-defined combinatorial structure in which each cell is adjacent to exactly one cell along each of its faces. This is important when a tetrahedral mesh Σ is used as a decomposition of the domain of a volume data set S.

Although, theoretically, the number m of tetrahedra in a mesh Σ can be quadratic in the number n of vertices of Σ, in practice, we have $m \approx 6n$. Given a volume data set S, an *approximated* tetrahedral mesh is a mesh Σ' having m' ($m' < m$) tetrahedra and vertices at a subset V' of the original data set V, with n' ($n' < n$) points. A scalar field f' is defined on Σ', similarly to f, with the convention that values of f and f' are the same on each vertex that belongs to both V and V'. The approximation error associated with Σ' is the error that we perform in using Σ' instead of Σ for describing S. We consider as the *error* associated with each tetrahedral cell σ of Σ' (also called the *field error*) the maximum of the absolute value of the difference between the actual field value at the points of $V \setminus V'$ inside σ and the field value at the same points linearly interpolated within σ. When we deal with irregular meshes with a non-convex domain, the error associated with a tetrahedron σ takes into account the error performed at σ in approximating the field domain, that we call *domain error*. The domain error at a tetrahedron σ is computed as the maximum value of the one-sided Hausdorff distances of the points of the domain from tetrahedron σ, and it is not null only if σ is close to the boundary of Σ (see [2] for details).

4 Multiresolution Tetrahedral Meshes

The basic idea underlying any multiresolution model is to collect the updates performed on the mesh during top-down refinement, or bottom-up decimation, and to organize them by defining suitable dependency relations. Dependency relations drive the extraction of meshes at intermediate resolutions, possibly variable in space. On the other hand, the type of updates and the notion of dependency used in a model must obey some rules in order to ensure the topological correctness of the meshes that can be retrieved.

4.1 Updates in a Multiresolution Mesh

Given a mesh Σ, a *sub-mesh* Σ' of Σ is a mesh defined by any face-connected subset of the tetrahedral cells of Σ. The *(combinatorial) boundary* of a sub-mesh Σ' is the set of cells which are common to both $\Sigma \setminus \Sigma'$ and Σ'.

An *update* of a mesh Σ can be viewed as a pair of meshes $u = (\Sigma_1, \Sigma_2)$, where Σ_1 is a sub-mesh of Σ, and Σ can be modified by replacing Σ_1 with Σ_2 in such a way that Σ_2 fills the hole left in Σ after the removal of Σ_1. We will refer to Σ_1 and Σ_2 as the *first* and the *second* component, respectively, of update u. An update $u = (\Sigma_1, \Sigma_2)$, is *conforming* when both Σ_1 and Σ_2 are conforming meshes and the combinatorial boundary of Σ_1 consist of the same set of cells

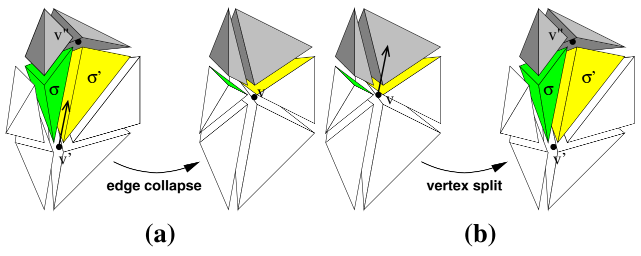


Figure 1. Exploded views of the result of an update of a tetrahedral mesh through edge collapse (a) and vertex split (b). In (a) the collapsing edge and its endpoints are marked and the yellow and green tetrahedra degenerate into triangles.

as that of Σ_2 (and, thus, of $\Sigma \setminus \Sigma_1$). Also, for the sake of brevity, below we consider only updates which increase the number of vertices and tetrahedra in the resulting mesh, i.e., updates $u = (\Sigma_1, \Sigma_2)$ in which the number of tetrahedra of Σ_2 is greater than the number of tetrahedra of Σ_1.

The two types of updates, on which the data structures we compare are based, are *edge collapse* and *tetrahedron bisection*. Edge collapse is the most common simplification operator for irregular tetrahedral meshes, because of the difficulties in dealing with a non-convex domain when vertex insertion is applied. Tetrahedron bisection is the most successful recursive subdivision operator for regular meshes because of its higher flexibility, with respect to other techniques, in producing variable resolution conforming meshes.

Edge collapse. Edge collapse consists of contracting an edge e, with endpoints v' and v'', to a point v, which can be one of the two endpoints of e, or an internal point. The mesh around e is deformed by replacing vertices v' and v'' with v. As a consequence, tetrahedra containing both v' and v'' collapse into triangles (see Figure 1a). The inverse operation of an edge collapse is a *vertex split*. A vertex split expands a vertex v into an edge e having its endpoints at v' and v''. The tetrahedra incident at v are partitioned into two subsets, which are separated by a fan of triangles incident at v. Tetrahedra of the two subsets are deformed to become incident at v' and v'' respectively. Triangles belonging to the fan become tetrahedra incident at edge e (see Figure 1a). An edge collapse and a vertex split are both conforming updates, since they do not split the simplexes in the combinatorial boundary of the update.

Tetrahedron bisection. The bisection rule for tetrahedra consists of replacing a tetrahedron σ with the two tetrahedra obtained by splitting σ through the middle point of its

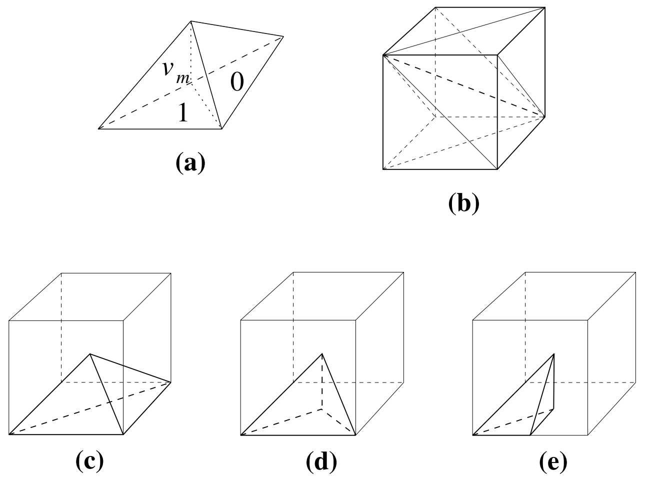


Figure 2. (a) An example of tetrahedron bisection. (b) Subdivision of the initial cubic domain into six tetrahedra. Examples of a 1/2 pyramid (c), of a 1/4 pyramid (d), and of a 1/8 pyramid (e).

longest edge and by the plane passing through such point and the opposite edge in σ (see Figure 2a). The bisection rule is applied recursively to an initial decomposition of the cubic domain into six tetrahedra (see Figure 2b). Regardless of the number of times the tetrahedra are bisected, the resulting shapes always fall into one of three cases. These shapes are actually cyclic in that every three levels of decomposition result in a geometrically similar shape. Splitting a tetrahedron in the initial cube subdivision results in two tetrahedra with a shape identical to that obtained by splitting a pyramid with a square base in half along the diagonal of its base. We call such shape a *1/2 pyramid* (see Figure 2c). Splitting a 1/2 pyramid along its longest edge results in two tetrahedra whose shape we call a *1/4 pyramid* (see Figure 2d). Finally, splitting a 1/4 pyramid along its longest edge results in two tetrahedra whose shape we call a *1/8 pyramid* (see Figure 2e). Each of the six initial tetrahedra also has a 1/8 pyramid shape.

Tetrahedron bisection is a non-conforming update. Only if a conforming update is applied to a conforming mesh, is the result still a conforming mesh. To this aim, we can cluster non-conforming updates to get conforming ones. Given a collection U of updates, we consider clusters of updates obtained by merging sets of updates $\{u_1 \dots u_k\}$ in U such that the union of $u_1 \dots u_k$ is a conforming update, and no other proper subset of $\{u_1 \dots u_k\}$ has this property. In the case of tetrahedron bisection, conforming updates are defined by clustering tetrahedra along their splitting edge. In each resulting conforming update $u = (\Sigma_1, \Sigma_2)$, we call Σ_1 a tetrahedral *cluster*, and Σ_2 the corresponding *split-set*. We have three types of clusters (and, thus, of updates) generated by the three basic tetrahedral shapes, that we call *axis-aligned*, *plane-aligned* and *non-aligned* clusters, respectively. They can be shown to be the smallest conforming updates that can be generated by a tetrahedron bisection:

- An *axis aligned* cluster is formed by eight 1/2 pyramids, as only 1/2 pyramids share an axis-aligned edge.
- A *plane-aligned* cluster is formed by four 1/4 pyramids, as only 1/4 pyramids share a plane-aligned edge, i.e., on an edge parallel to one of the coordinate planes.
- A *non-aligned* cluster is formed by twelve 1/8 pyramids, as only 1/8 pyramids share an edge which is not aligned to a coordinate axis or plane.

4.2 Dependencies in a Multiresolution Mesh

Given two updates u_1 and u_2, such that u_1 has been performed before u_2, we say that u_2 *directly depends* on u_1 if u_2 removes some tetrahedra inserted by u_1 and not yet removed (and re-inserted) in any other update in between. The transitive closure of the relation R of direct dependency is a *partial order*.

Thus, a *multiresolution tetrahedral mesh* $M = (\Sigma_b, U, R)$ is defined by an initial mesh Σ_b subdividing the domain, that we term the *base mesh*, a set of updates $U = \{u_1 \dots u_k\}$, and a relation R of direct dependency among updates. The mesh at the full resolution, that we term the *reference* mesh, can be obtained by applying all updates in U to the base mesh. We call a multiresolution mesh in which both the base mesh and all updates are conforming a *Multi-Tessellation (MT)*.

In the following, we will consider two instances of a Multi-Tessellation defined by two different construction process, as the *Edge-based Multi-Tessellation (MT)*, built through bottom-up decimation of the reference mesh, and the *Regular Multi-Tessellation* (also called a *Hierarchy of Tetrahedra (HT)*), built through top-down recursive refinement of the base mesh.

In an edge-based MT, the base mesh is an irregular coarse mesh covering the domain D, and all updates are vertex splits. The first component Σ_1 of an update u contains on average 27 tetrahedra, while the second component Σ_2 contains on average 32 tetrahedra. Thus, an update adds 5 tetrahedra on average.

In a hierarchy of tetrahedra, the base mesh is defined by the decomposition of the initial cubic domain into six 1/8 pyramids and the updates are defined by the three axis-aligned, plane-aligned and non-aligned clusters. In an HT, clusters (first components of an update) contain 4, 6 or 8 tetrahedra, while split-sets (second components of an update) contain 8, 12 or 16 tetrahedra. On average, six tetrahedra are added by an update.

A Multi-Tessellation $M = (\Sigma_b, U, R)$ provides a compact way of encoding all conforming meshes that can be obtained by using some of the updates performed during the process of mesh simplification [5]. Such meshes correspond to all subsets of updates in M which are closed with respect to the partial order. A subset U' of the updates in M is called *closed* if, for each update u in U', all predecessors of u with respect to the transitive closure of R belong to U'. A closed subset of updates corresponds to a set of updates that can be applied to the base mesh while satisfying the dependency relation. The collection of all closed sets of updates in a multiresolution mesh M defines the complete set of meshes which can be extracted from M. Moreover, since a Multi-Tessellation is a conforming multiresolution mesh, any of the extracted meshes is conforming.

5 Data Structures for Multiresolution Tetrahedral Meshes

5.1 Encoding an Edge-Based Multi-Tessellation

In this subsection, we briefly describe the data structure for encoding an edge-based MT (see [2] for more details). The algorithm for constructing an edge-based MT, that we use in the experiments shown in Section 6, is described in [3]. There are two basic ingredients in encoding an edge-based MT: encoding the direct dependency relation, and encoding the updates. The base mesh is encoded as an indexed structure with adjacencies.

The direct dependency relation is encoded by extending a technique proposed by El Sana and Varshney [7] for triangle meshes, which is based on a forest of binary trees of vertices, and on a suitable vertex enumeration mechanism. The leaf nodes of the forest correspond to the vertices of the reference mesh, the internal nodes to the vertices generated by the decimation process. Roots of the forest correspond to the vertices of the base mesh. The two children of each internal node v are the endpoints v' and v'' of the edge e created when splitting v. The vertex enumeration mechanism together with the arcs of the binary forest is an implicit way of encoding the dependency relation. Our implementation has a cost equal to $12n$ bytes, where n is the number of vertices of the reference mesh, since the number of internal nodes is basically equal to n.

An update u is implicitly encoded as an offset vector and an offset value used to find the positions and the field value of vertices v' and v'' from those of v, an error value, $\varepsilon(u)$, which provides an estimate of the approximation error associated with u (which is the maximum of the error associated with the tetrahedra forming u), and a bit mask used to partition the set of tetrahedra incident at v, when performing a vertex split. This bit mask contains one bit for each tetrahedron incident at v (see Figure 1). Note that to obtain a compact structure we only store the error associated with an update and not with each tetrahedra forming it.

In [2], we have shown that the storage cost for the information associated with a single update contributes for a cost of 18 bytes. Therefore, the total cost of the MT data structure (including the cost of encoding the direct dependencies) is equal to $30n$ bytes plus the cost of storing the base mesh, which is negligible.

The storage cost of the data structure for an edge-based MT is about 1/5 of the cost of storing the reference mesh in a data structure encoding both connectivity and face-adjacency information. If we store the reference mesh only with connectivity information (without face-adjacencies), the cost of the MT structure is still less than 2/5 than that of encoding the reference mesh.

5.2 Encoding a Hierarchy of Tetrahedra

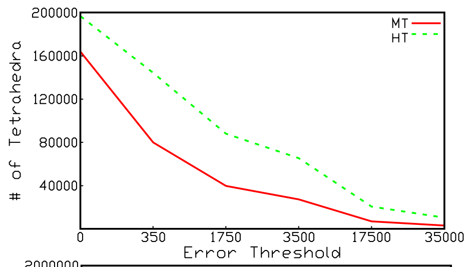
The data structure for the HT does not encode the dependency relation and the updates of the multiresolution mesh directly, but it encodes a binary tree which describes the nested structure of the subdivision. It makes use of a linear representation for the hierarchy instead of a pointer-based tree structure. The data structure consists of:

- A table containing the field values at the n data points.
- A forest of six almost full binary trees, containing the errors associated with the tetrahedra encoded as an array. The trees describe the complete subdivision of the initial cube, except for the last level which corresponds to the tetrahedra of the reference mesh, that have a null error.

Location codes, one for each tetrahedron σ, are used to index the field table. In each location code, the first number is the level of σ in the tree, and the second number indicates the path from the root of the tree to σ. The location code is defined on the basis of a labeling scheme for the children of a tetrahedron and for the vertices of these children in the hierarchy. The forest can be traversed using only local computations to determine where we are in space: the only computation is finding the midpoint of the longest edge. All other vertices can be obtained directly from the vertices of the parent tetrahedron.

The dependency relation in the Multi-Tessellation is implicitly encoded by the forest which describes the nesting structure of the tetrahedra in the subdivision. The clusters defining the updates are efficiently computed when extracting a mesh by using location codes and constant-time neighbor finding (see [14]).

In a data set with n points, there are $6n$ tetrahedra in the reference mesh, and, thus, $6n$ internal tetrahedra. This yields a storage cost of $12n$ bytes for the error values plus

(a)


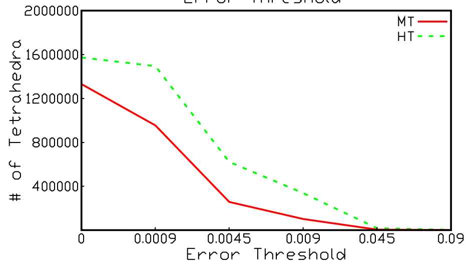
(b)


Figure 3. Number of tetrahedra for a mesh at a uniform LOD with different error thresholds for the (a) Smallbucky and (b) Plasma data sets.

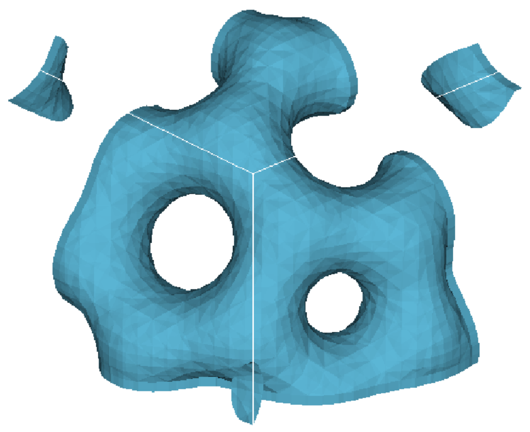


Figure 4. Uniform LOD extraction: error threshold equal to 0.1% of the field range. The isosurface for a field value equal to 105.000 is shown (see the corresponding color plate).

$2n$ bytes for the field table, assuming two bytes per error and field value, leading to a total cost of $14n$ bytes. Thus, the storage cost of an HT is a little less than 1/2 of the cost of an edge-based MT.

6 Experiments and Comparisons

In order to compare the HT and the edge-based MT, we used two volume data sets with rather different sizes:

- The first data set, termed *Smallbucky*, is a portion (1/8) of the well-known bucky-ball data set (courtesy of AVS inc.), with 32,768 vertices. In the case of full resolution with 0% error tolerance, the MT consists of 163,840 tetrahedra, while the HT consists of 196,608 tetrahedra. The difference arises because the MT starts with a reference mesh obtained by splitting each cube formed by eight data points with five tetrahedra, while the HT splits the initial cube into six tetrahedra.
- The second data set, termed *Plasma*, is a large synthetic data set (courtesy of the Visual Computing Group at the Italian National Research Council) with 262,144 vertices. In the case of full resolution with 0% error tolerance, the MT (HT) consists of 1,310,703 (1,572,864) tetrahedra.

Thus, in terms of the number of tetrahedra, we see that Plasma is about 10 times as complex as Smallbucky.

Our comparison is in terms of the number of tetrahedra: this quantity is directly related to the complexity of the queries as the execution time of the selective refinement algorithms depends on such parameter.

The first comparison that we performed was for a uniform LOD query, where we extracted meshes having an approximation error below a certain error threshold over the whole domain. Figure 4 shows (through an isosurface) an example of a mesh at a uniform LOD in which the error threshold is equal to 0.1% of the field range. Figure 3 shows the number of tetrahedra in the extracted mesh for different error thresholds. From such graphs we see that the HT has more tetrahedra than the MT. The main reason for this is that in the MT there is more flexibility in choosing which tetrahedra are split (e.g., when applying a vertex split operation), while in the HT these are determined by the fixed recursive decomposition rule. Also, note that, when we perform a vertex split in the MT, we introduce 5 tetrahedra on average, and when we perform a conforming update in the HT, we introduce 6 tetrahedra on average.

The second comparison that we performed was extracting a mesh at variable LOD in a region of space, i.e. a certain approximation error is allowed inside a region of interest, while a larger error is allowed elsewhere. For our tests, we chose an axis-aligned box as region of interest. Figure 6 shows (through an isosurface) a mesh extracted with an error threshold equal to 0.1% of the field range in the selected box, and arbitrary large outside. We used a number of different boxes at different positions and recorded the average number of tetrahedra in the resulting meshes, while varying the error threshold for the field values inside the boxes (see Figure 5). Note that the number of tetrahedra in the HT is less than in the MT. This is due to the use of regular decomposition in the HT which means that the tetrahedra

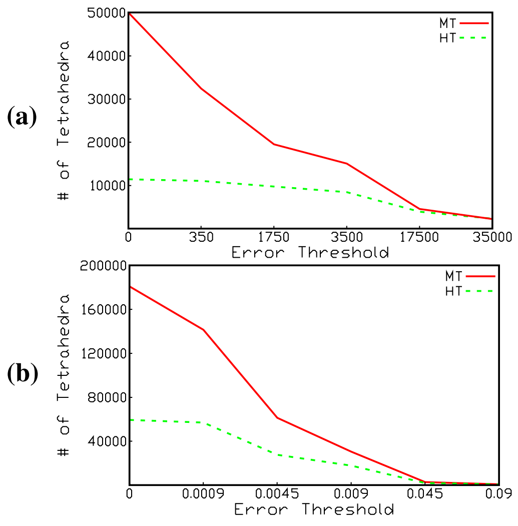


Figure 5. Number of tetrahedra for a mesh extracted at variable resolution in a box with different error thresholds for the (a) Smallbucky and (b) Plasma data sets.

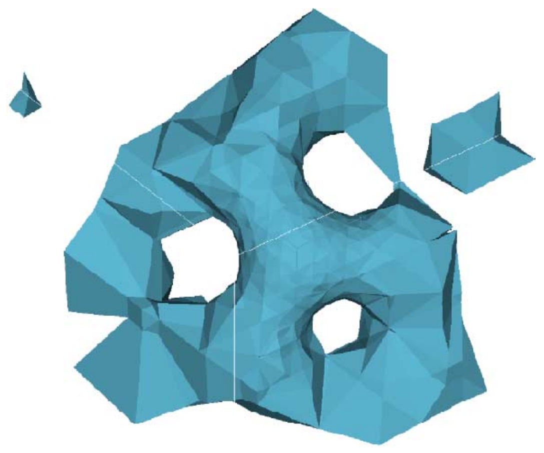


Figure 6. Variable LOD based on a region in space: mesh extracted with an error threshold equal to 0.1% of the field range in selected box, and arbitrary large outside. The isosurface for a field value equal to 105.000 is shown (see the corresponding color plate).

are aligned with the faces of the box (i.e., they are axis-parallel). In particular, we know that each of the tetrahedra in the HT has at least one face that is axis-parallel. Thus, as soon as we cross the boundary of a box, we can start having tetrahedra with larger faces, thereby having fewer tetrahedra than in the MT where the tetrahedra may be forced to intersect the boundary of the box, thereby delaying the merging process.

The third comparison that we performed was for a query that extracts a mesh at variable LOD based on the field value (i.e., on the isosurface). A given error threshold is allowed in the tetrahedra intersected by the isosurface while a larger error threshold is allowed elsewhere. Figures 7 and 8 indicate the resulting number of tetrahedra in the extracted mesh and faces in the isosurface for the Smallbucky and Plasma data sets. As in the box query, the number of tetrahedra in the mesh is less for the HT than the MT, while the number of faces is greater for the HT than the MT. Observe that the fact that the HT has less tetrahedra in the mesh than the MT but more faces than the MT means that a higher percentage of tetrahedra in the HT intersect the isosurface. Thus, the HT is better than the MT at pruning the irrelevant tetrahedra from the isosurface field value that forms the query. Of course, it could also be argued that in the MT, the same error value is associated with all of the tetrahedra in the conforming update, while in the HT, an error value is associated with each tetrahedron. Thus we could obtain better performance with the MT if we were to store an error value with each tetrahedron, but this would increase the storage cost of an edge-based MT from $30n$ to $49n$ bytes, i.e., of about 5/3.

For comparison purposes, we also implemented another technique for extracting conforming meshes from a hierarchy of tetrahedra based on *error saturation* [16, 25]. First, all tetrahedra belonging to the same clusters are assigned the same error value, which is equal to the maximum of their original error values. Moreover, the approximation error associated with each tetrahedron is saturated to be greater than or equal to the error associated with its children. This implies that, during mesh extraction, if a tetrahedron is refined, then all tetrahedra of the same cluster are refined. Thus, all meshes extracted at uniform LOD are guaranteed to be conforming, while consistency is not guaranteed when they are extracted at a variable LOD. Experimental comparisons that we have performed on the basis of uniform LOD queries have shown that the meshes extracted from a saturated HT have slightly more tetrahedra than those extracted by our method. On the other hand, the computing times of our depth-first algorithm are the same as those of the algorithm which extracts from a saturated HT (which simply performs a top-down traversal of the hierarchy without any neighbor finding).

Another important comparison can be performed between the two models on the basis of parameters which characterize the shape of the tetrahedra in the extracted meshes. As discussed in [22], a widely used and elegant measure for analyzing the shape of the tetrahedra in a mesh is the *circumradius-to-shortest-edge* ratio r of a tetrahedron. The circumradius is the radius of its circumsphere. One would like this ratio to be as smallest as possible. In an HT, r is equal to 0.91 on average, where the minimum value of r (equal to 0.75) is for the 1/2 pyramid, while its

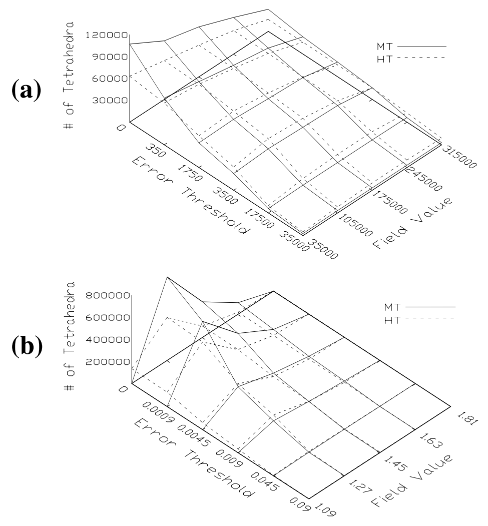


Figure 7. Number of tetrahedra for a mesh extracted at a variable LOD based on a different field values with different error thresholds for the (a) Smallbucky and (b) Plasma data sets.

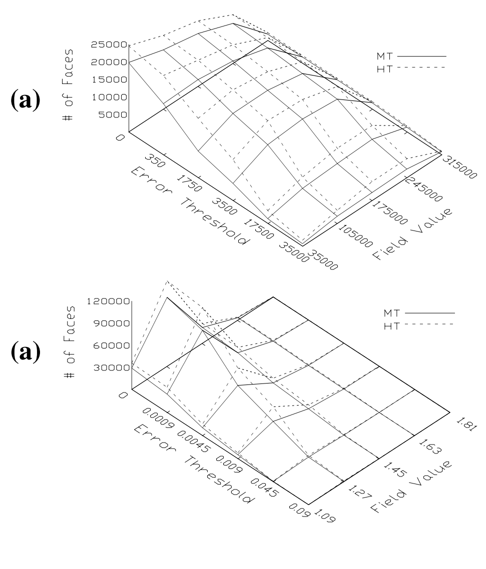


Figure 8. Number of faces in the isosurface for a mesh extracted at a variable LOD based on different field values with different error thresholds for the (a) Smallbucky and (b) Plasma data sets.

maximum value (equal to 1.19) is for the 1/4 pyramid. In an edge-based MT, our experiments have shown a value of r equal to 1.38 on average.

7 Concluding Remarks

A comparison of two multiresolution representations for large volume data sets based on decompositions of the field domain into tetrahedral cells has been presented. The models differ on the basis of the rule applied to refine an initial coarse mesh (or, to decimate an initially fine mesh). The HT model is based on a refinement of a coarse mesh by tetrahedral bisection, while the MT model is based on a decimation of an initially fine mesh via a vertex split.

An MT can deal with both structured and unstructured data sets, while the HT is specific for structured ones. The meshes extracted from an HT satisfy the Delaunay criterion, and the better shape of the tetrahedra forming it, captured by the circumradius-to-shortest edge ratio, is reflected also in the visual quality of the isosurfaces extracted. The HT is obviously more economical than an MT since it does not store the topology of the mesh explicitly. On the other hand, encoding an edge-based MT is considerably more economical than encoding the mesh at full resolution.

The experiments on the queries that we performed showed that the HT performed better than the MT in terms of the size of the extracted mesh, in that there were fewer tetrahedra for the HT than for the MT, except when using a uniform level of detail.

Finally, an important common feature of both the HT and the edge-based MT is that from both of them we extract meshes with connectivity and adjacency information at no extra cost. Such information is fundamental in applications involving geometric computations and navigation over the extracted mesh.

In the case of very large data sets, *out-of-core* solutions can be very effective. Even when a multiresolution approach is adopted, out-of-core approaches can be useful in order to process the data at the maximum resolution. Moreover, the multiresolution representations of a huge data set can exceed the available in-core memory, and therefore the extraction of LOD models should be implemented out-of-core. This latter problem is an open research problem that we plan to investigate in the near future.

Acknowledgements This work has been supported in part by the National Science Foundation under grants EIA-99-00268 and ITS-00-86162, by a University of Maryland International Travel Fund Grant, by a project funded by INDAM National Group on Scientific Computing (GNCS) on “Scientific Data Visualization” and by a University of Genova travel grant.

References

[1] P. Cignoni, L. De Floriani, C. Montani, E. Puppo, and R. Scopigno. Multiresolution modeling and visualization of volume data based on simplicial complexes. In *Proceedings 1994 Symposium on Volume Visualization*, pages 19–26, Washington, DC, October 1994.

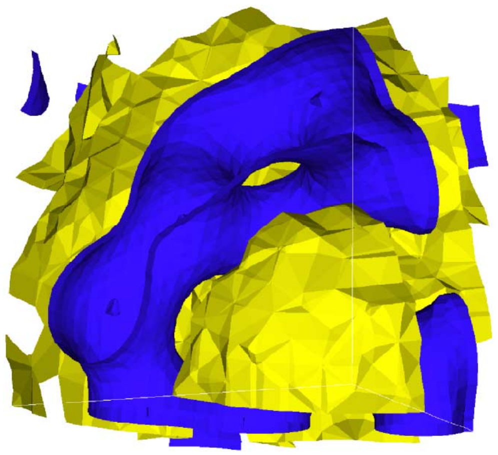


Figure 9. Variable LOD based on field value: a mesh extracted with an error threshold equal to 0.1% of the field range on the tetrahedra intersected by the isosurface with field value equal to 105.000 (shown in blue), and an error threshold arbitrary large outside. The second isosurface, with a field value equal to 10.000 (shown in yellow), illustrates the lower resolution of the mesh in the region formed by the tetrahedra that do not intersect the selected isosurface (see the corresponding color plate).

[2] P. Cignoni, L. De Floriani, P. Magillo, E. Puppo, and R. Scopigno. TAn2 - Visualization of Large Irregular Volume Data sets. Department of Computer and Information Science, University of Genova (Italy), DISI-TR-00-07, 2000.

[3] P. Cignoni, D. Costanza, C. Montani, C. Rocchini, and R. Scopigno. Simplification of tetrahedral volume with accurate error evaluation. In *Proceedings IEEE Visualization 2000*, pages 85–92, Salt Lake City, UT, October 2000.

[4] L. De Floriani, P. Magillo, and E. Puppo. A formal approach to multiresolution hypersurface modeling. In *Geometric Modeling: Theory and Practice*, W. Strasser, R. Klein, and R. Rau, eds., pages 302–323. Springer-Verlag, Berlin, Germany, 1997.

[5] L. De Floriani and P. Magillo. *Multiresolution Meshes*. Principles of Multiresolution in Geometric Modeling - PRIMUS summer school, Munich, Germany, August 2001.

[6] B.Hamann, E. LaMar, and K. I.Joy. High-quality rendering of smooth isosurfaces. *The Journal of Visualization and Computer Animation*, 10(2):79–90, April-June 1999.

[7] J. El-Sana and A. Varshney. Generalized view-dependent simplification. *Computer Graphics Forum*, 18(3):83–94, September 1999.

[8] M. Garland. Multiresolution modeling: Survey & future opportunities. In *Eurographics '99 – State of the Art Reports*, pages 111–131, 1999.

[9] M.H. Gross and O.G. Staadt. Progressive tetrahedralizations. In *Proceedings IEEE Visualization '98*, pages 397–402, Research Triangle Park, NC, October 1998.

[10] R. Grosso, C. Luerig, and T. Ertl. The multilevel finite element method for adaptive mesh optimization and visualization of volume data. In *Proceedings IEEE Visualization '97*, pages 387–394, Phoenix, AZ, October 1997.

[11] D. J. Hebert. Symbolic local refinement of tetrahedral grids. *Journal of Symbolic Computation*, 17:457–472, 1994.

[12] I. Ihm and S. Park. Wavelet-based 3D compression scheme for very large volume data. In *Proceedings Graphics Interface*, pages 107–116, Banff, Canada, May 1998.

[13] T. Kim and Y. Shin. An efficient wavelet-based compression method for volume rendering. In *Proceedings of 7^{th} Pacific Conference on Computer Graphics and Applications*, pages 147–157, Seoul, Korea, October 1999.

[14] M. Lee, L. De Floriani and H. Samet. Constant-time neighbor finding in hierarchical tetrahedral meshes. In *Proceedings Shape Modeling International 2001*, pages 286–295, Genova, Italy, May 2001.

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

[16] M. Ohlberger and M. Rumpf. Hierarchical and adaptive visualization on nested grids. *Computing*, 56(4):365–385, 1997.

[17] V. Pascucci and C. L. Bajaj. Time critical isosurface refinement and smoothing. In *Proceedings IEEE Symposium on Volume Visualization*, pages 33–42, Salt Lake City, UT, October 2000.

[18] J. Popovic and H. Hoppe. Progressive simplicial complexes. In *ACM Computer Graphics Proceedings, Annual Conference Series, (SIGGRAPH '97)*, pages 217–224, 1997.

[19] M. Rivara and C. Levin. A 3D refinement algorithm for adaptive and multigrid techniques. *Communications in Applied Numerical Methods*, 8:281–290, 1992.

[20] K.J. Renze and J.H. Oliver. Generalized unstructured decimation. *IEEE Computer Graphics & Applications*, 16(6):24–32, November 1996.

[21] R. Shekhar, E. Fayyad, R. Yagel, and J. Cornhill. Octree-based decimation of marching cubes surfaces. In *Proceedings IEEE Visualization '96*, pages 335–344, San Francisco, CA, October 1996.

[22] J.R. Shewchuck. Tetrahedral mesh generation by Delaunay refinement. In *Proceedings 14th Annual Symposium on Computational Geometry*, pages 86–95, Minneapolis, MN, June 1998.

[23] I.J. Trotts, B. Hamann, and K.I. Joy. Simplification of tetrahedral meshes with error bounds. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):224–237, July-September 1999.

[24] J. Wilhelms and A. Van Gelder. Multi-dimensional trees for controlled volume rendering and compression. In *Proceedings of the 1994 Symposium on Volume Visualization*, pages 17–18, Washington, DC, October 1994.

[25] Y. Zhou, B. Chen, and A. Kaufman. A multiresolution tetrahedral framework for visualizing regular volume data. In *Proceedings IEEE Visualization '97*, pages 135–142, Phoenix, AZ, October 1997.