Building and Traversing a Surface at Variable Resolution

Leila De Floriani, Paola Magillo
Dipartimento di Informatica e Scienze dell'Informazione – Università di Genova *

Enrico Puppo
Istituto per la Matematica Applicata – Consiglio Nazionale delle Ricerche †

Abstract

We consider the *Multi-Triangulation*, a general model for representing surfaces at variable resolution based on triangle meshes. We analyse characteristics of the model that make it effective for supporting basic operations such as extraction of a surface approximation, and point location. An interruptible algorithm for extracting a representation at a resolution variable over the surface is presented. Different heuristics for building the model are considered and compared. Results on both the construction and the extraction algorithm are presented.

1 Introduction

Multiresolution geometric models support representation and processing of spatial entities at different levels of detail. Such representations have received recently a lot of attention because of their potential impact on applications, such as terrain modeling in geographic information systems, scientific data visualization, and virtual reality. A major approach to multiresolution modeling is based on a cellular decomposition of the spatial entity involved: the underlying idea is that the accuracy in the representation is somehow proportional to the number of cells in the subdivision. In this paper, we focus on multiresolution representations of surfaces, based on simplicial domain decomposition.

In the last few years, a number of multiresolution surface models have been proposed in the literature that support compact encoding and fast access to a high number of representations, whose level of detail is possibly variable across the surface domain [3, 5, 6, 10, 13, 16, 17]. Many such models were originally designed to represent terrain surfaces, though all of them, but [6, 16], can be generalized easily to free-form surfaces embedded in the three-dimensional space. In [7, 18], we have shown that existing multiresolution models can be viewed as special cases of the *Multi-Triangulation (MT)*, a general model introduced in [17], which will be reviewed, and extended to free-form surfaces in Section 2. Therefore, many general properties of multiresolution surface models can be studied on the MT, and the results can be extended to all other models.

Here, we consider two basic spatial operations on a multiresolution surface model, namely: extraction of a representation of minimum size at a resolution variable over the domain, according to a predefined threshold function; and point location at variable resolution. It is well understood that the efficiency of such operations depends on the characteristics of the model and, thus, on the underlying construction technique.

A major contribution of the paper is to propose, analyze and experiment different algorithms to construct an MT. The core of such an algorithm is the technique used to produce polyhedral approximations of the underlying surface based on suitably selected subsets of the data points. The general problem of finding a polyhedral surface of minimum size that approximates a dataset within a given threshold is known to be NP-hard [1]. In the literature, we can find heuristics, mostly based on an error-driven simplification approach, that are empirically good, i.e., they highly reduce the size of the mesh needed to achieve a given accuracy, while they do not guarantee good theoretical characteristics for the resulting structure, which are necessary to achieve good complexity bounds for the two basic operations. On the other hand, construction techniques that produce a theoretically efficient structure may need considerably more points to achieve a given accuracy.

Thus, we can see a trade-off between practical performance, as data compression tools, and theoretical efficiency. Here, we analyse some heuristics which try to overcome such a problem by combining an error-driven approach with a criterion that maintains the resulting structure well balanced. We analyze and compare such heuristics with techniques that try to optimize either only on the number of points necessary to achieve a certain accuracy, or only on the theoretical efficiency of the resulting multiresolution structure for extraction and searching.

We also describe a new algorithm for extracting a representation at variable resolution from an MT, which is interruptible, i.e., it converges towards an optimal solution by finding better and better approximations: this algorithm is especially suited for real time applications, since it allows an application to dinamically set the level of detail used in rendering according to time constraints.

2 The Multi-Triangulation

The *Multi-Triangulation* (MT), is a triangle-based multiresolution decomposition that has been introduced in [17]. In its original definition, an MT is based on a collection of local updates of an initial triangulation of a plane domain, arranged according to a partial order. Consistent subsets of the local updates can be combined to obtain triangulations at different resolutions. Here, we extend the model

*Address: Via Dodecaneso, 35 – 16146 Genova, ITALY. Email: {deflo,magillo}@disi.unige.it

†Address: Via de Marini, 6 (Torre di Francia) – 16149 Genova, ITALY. Email: puppo@ima.ge.cnr.it

to represent generic meshes of triangles embedded in three-dimensional space, and we give an equivalent graph-based definition that is more intuitive and directly reflects the data structure used to encode the model.

A *triangle mesh* is a set Σ of triangles embedded in the three-dimensional space such that, for each pair of triangles $t_1, t_2 \in \Sigma$, either t_1 and t_2 are disjoint, or their intersection is an edge or a vertex common to t_1 and t_2, and such that the union of all such triangles form a manifold surface. The *size* of a mesh is the number of its triangles.
A *Multi-Triangulation* (MT) is described by a directed acyclic graph (DAG) $\mathcal{M} = (\mathcal{N}, \mathcal{A})$ such that:

1. there is a unique *source* node $n_S \in \mathcal{N}$ without incoming arcs, and a unique *drain* node $n_D \in \mathcal{N}$ without outgoing arcs;
2. every arc in $\mathcal{A}$ is labeled with a set of triangles;
3. for every node $n \in \mathcal{N}$, the unions of triangle sets which are labels of the arcs entering and emanating from n, respectively, form two connected triangle meshes with the same boundary;
4. the set of triangles corresponding to any cut of $\mathcal{M}$ that separates the source from the drain is a triangle mesh.

Property 4 is especially important because it relates triangle meshes at variable resolution to *cuts* in the directed acyclic graph. The mesh defined by the cut of arcs emanating from the source is called the *minimal* mesh of $\mathcal{M}$, and denoted by Σ_S. The mesh defined by the cut of arcs entering the drain is called the *maximal* mesh of $\mathcal{M}$, and denoted by Σ_D. Each internal node of $\mathcal{M}$ can be viewed as a local update operation that replaces triangles labeling its incoming arcs with triangles labeling its outgoing arcs. Figure 1 depicts the DAG describing an MT in the plane, the local updates corresponding to each node, a cut and the corresponding mesh. A cut separates the nodes of $\mathcal{M}$ into two connected components; for convenience, we call nodes *above* the cut, and nodes *below* the cut, the nodes of the component containing the source and the drain of $\mathcal{M}$, respectively.

An MT $\mathcal{M}$ is *increasing* if for any internal node of $\mathcal{M}$ the number of triangles labelong its outgoing arcs is larger than the number of triangles labeling its incoming arcs. In an increasing MT, the minimal mesh has the smallest size, and the maximal mesh has the largest size, among all meshes defined by cuts. In practice, the size of a mesh increases as the cut moves from the source to the drain. In the remainder of the paper, we always consider increasing MTs.

An increasing MT $\mathcal{M}$ has a *linear growth* if for each cut in $\mathcal{M}$, the ratio between the number of arcs in the component above the cut, and the size of the cut itself is bounded by a constant. An MT $\mathcal{M}$ has a *logaritmic height* if the maximum length of a path from the source to the drain is logarithmic in the number of the arcs of $\mathcal{M}$. An MT $\mathcal{M}$ has *bounded width* if the maximum number of arcs emanating from a single node is bounded by a constant. Linear growth, logaritmic height, and bounded width are important features for the efficiency of traversal algorithms on an MT.

3 Polyhedral Surfaces at Variable Resolution

We consider a *surface* as a compact, connected, orientable 2-manifold, possibly with boundary, embedded in $\mathbb{R}^3$. A *triangulated approximation* of a given surface $\mathcal{S}$ is a triangle mesh Σ, which preserves the topological type of $\mathcal{S}$ and is geometrically "close" to $\mathcal{S}$ with respect to some criterion for measuring distances.

The *distance* between a surface and its triangulated approximation may be measured in different ways, depending on the specific application. We considered meshes representing *terrains* built from a discrete set of sampled data points by defining a triangulation of their xy projections, and meshes representing *free form surfaces* acquired through scanner equipments. For terrains, a mesh interpolating all data points is assumed to have zero error, and the error of a generic mesh is evaluated as the maximum vertical distance between the mesh and a data point. For free form surfaces, the acquired mesh is considered to have zero error. The error of a generic mesh is estimated as follows: for each vertex v of the original mesh, its distance $d(v, \Sigma)$ from the approximating mesh Σ is evaluated as the distance between v and the nearest point of Σ to v; the error of Σ is defined as $\max_v d(v, \Sigma)$. An error value $Err(t)$ is also associated with each triangle t of a mesh Σ, obtained from a distance estimation restricted to t. For terrains, $Err(t)$ is computed by considering only those data points whose xy projection lies in t; for free form surfaces, $Err(t)$ is obtained by taking into account only those original vertices v such that the nearest point of Σ to v belongs to triangle t.

Given a *threshold function* $\tau : \mathbb{R}^3 \to \mathbb{R}$, an approximated mesh Σ for dataset $\mathcal{D}$ is said to satisfy τ if for every triangle t of Σ we have $\min_{x \in t} \tau(x) \geq Err(t)$. The simplest situation occurs when τ is constant; another interesting class of threshold functions, useful in the visualization of large objects (e.g., terrains, buildings, ships, planes, etc...) require an accuracy that decreases according to the distance from a given viewpoint.

For practical purposes, it is important to be able to find efficiently an approximated model whose accuracy satisfies a given threshold, and which is likely to have a small size. In the remainder of the paper, we only consider triangulated approximations that correspond to cuts of an increasing MT, whose triangles have vertices at points of a dataset $\mathcal{D}$. In practice, the accuracy of a triangulated approximation increases with the number of triangles composing the mesh. Thus, approximations of the surface at increasing accuracies are obtained through cuts of an increasing MT located progressively farther from the source. The possibility of using arbitrary cuts allows us to locally adapt the resolution to an arbitrary threshold τ.

4 Spatial Operations on an MT

The power of a multiresolution model is related to its capability of supporting spatial operations at variable resolution. In [7] we outlined some of the basic operations that may be important in the applications. Among those, we consider here two fundamental operations on an increasing MT $\mathcal{M}$:

- *variable-resolution extraction*, i.e., finding the smallest mesh made of triangles of $\mathcal{M}$ that satisfies a given threshold function τ;
- *variable-resolution point location*, i.e., finding the coarsest triangle of $\mathcal{M}$ that contains a given query point p, and whose error is smaller than a given threshold $\tau(p)$.

4.1 Extraction of a triangulation at variable resolution

The problem of variable-resolution extraction is equivalent to finding a minimum cut of the MT, such that its corresponding triangulation is satisfies a threshold τ (see Figure

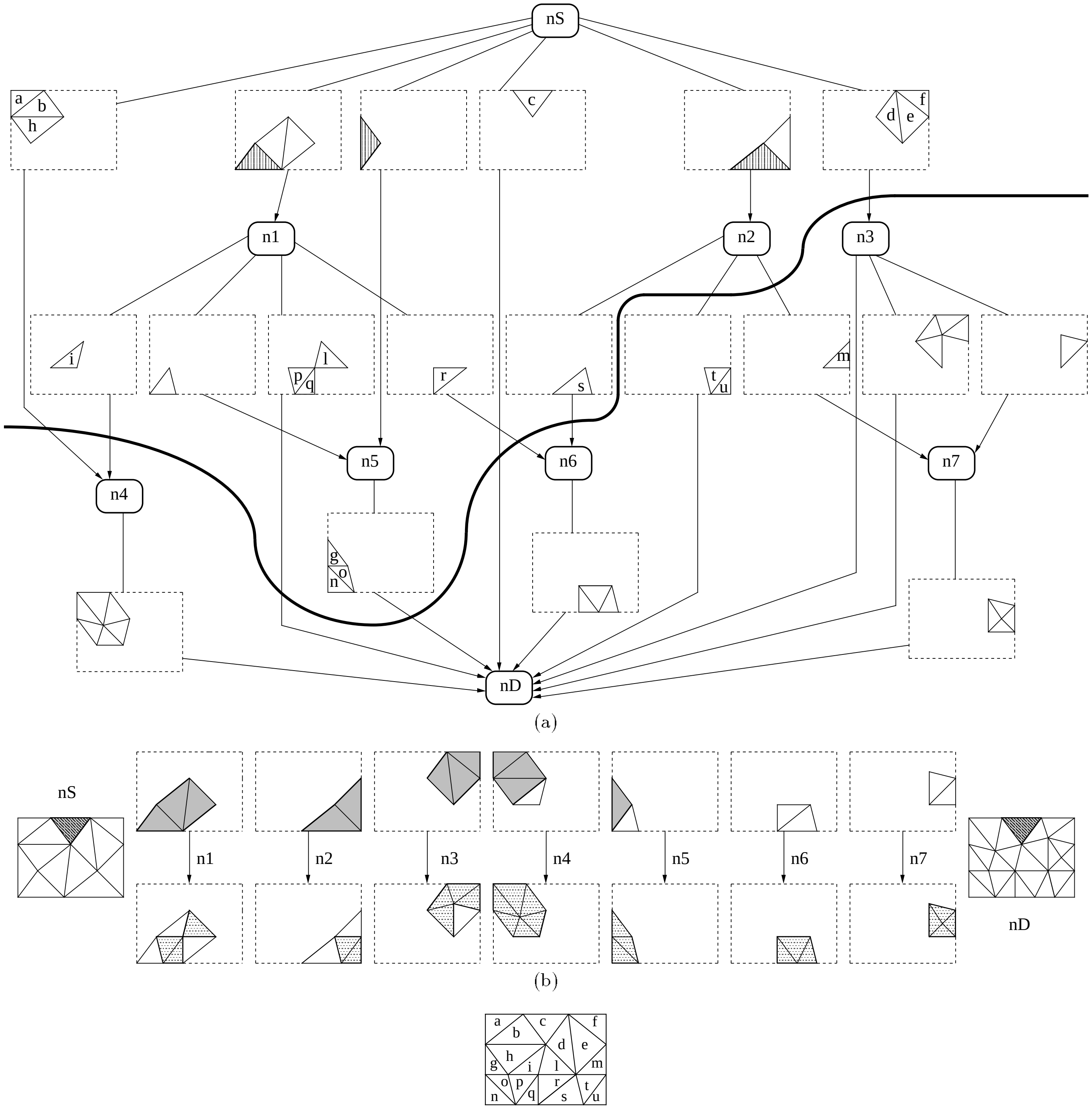

Figure 1: An increasing MT: (a) the DAG representing the model (the thick curve denotes a cut); (b) the minimal mesh corresponding to the source, the maximal mesh corresponding to the drain, and the local updates corresponding to each internal node (triangles belonging to the minimal mesh are shaded, triangles belonging to the maximal mesh are dotted; the dark grey triangle belongs to both the minimal and maximal mesh); (c) an extracted variable-resolution triangulation (the cut defining such triangulation is the one shown in (a), the dashed triangles in (a) do not satisfy the threshold).

1). An algorithm for such a problem has been proposed in [17] that achieves optimal time, i.e., linear in its output size, if the MT has linear growth. In the following, we describe an improved version of the algorithm that maintains the same time complexity, and, moreover, is interruptible.

An algorithm is called *interruptible* if it converges towards the exact solution by incrementally producing better and better approximations [20]. This characteristics is especially important in real-time contexts, where accuracy of the result might be traded for time, which is intended as a strict requirement. For instance, let us consider the rendering of a terrain in the context of a dynamic application such as a flight simulator, where the viewpoint is continuously changing. A different triangulation must be extracted and rendered at frame rate. Since the time for rendering is directly proportional to the size of the triangulation, an algorithm that always maintains a consistent (though approximated) solution can monitor at each instant how much time is left for completing extraction. If the time available expires before the optimal solution is found, then the algorithm can return the current solution.

Our algorithm is based on an breadth-first traversal of the DAG, like that proposed in [17]: starting from the source, it progressively visits all arcs and nodes above the solution. A queue of nodes that must be visited, and a list of triangles forming the current solution are maintained. The algorithm consists of a main loop that iterates until the queue is empty (or the extraction time expires). A node is added to the queue if either one of the triangles entering it is in the current solution, but it does not satisfy the required accuracy, or one of the nodes below it must be added to the queue. A triangle is added to the list when the node it emanates from is visited; a triangle is marked as being not part of the solution when the node it enters is visited. On exit from the loop, the list will contain all triangles of the solution, plus some extra (marked) triangles that are purged through a single scan.

The current solution is initialized with the triangles of the minimal triangulation Σ_S. For each such triangle t that does not satisfy the threshold function, the destination node of the arc containing t in its label is added to the queue. At each iteration, the node n_i which is the first element of the queue is visited and the current cut is advanced below n_i:

1. for every arc a incident into n_i, the triangles labeling a are eliminated from the current solution (i.e., they are marked);

2. for every arc a emanating from n_i, the triangles labeling a are added to the current solution; if some triangle t among those labeling a does not satisfy τ, then the destination node of a (if not visited) is added to the queue.

A special policy is used to insert elements into the queue: when a node n_j must be inserted into the queue, all unvisited sources of arcs entering n_j are recursively added to the queue *before* adding n_j. This policy ensures that a node is visited after its ancestors have been visited, i.e., only when all arcs entering it belong to the current cut. Thus, at each iteration the triangles of the current solution form a triangulation of the whole domain. It is not difficult to prove the correctness of the algorithm by showing the following facts: the set of output triangles corresponds to a cut, and any "smaller" cut defines a triangulation that does not satisfy the threshold. It is also easy to see that the time complexity is linear in the total number of triangles in the labels of arcs incident at nodes contained in the component above the cut. If the MT has linear growth, the time complexity is linear in the output size, hence optimal (see [17, 18] for details).

4.2 Point location at variable resolution

Locating a point on a surface is an important operation in all applications that need to evaluate local characteristics of the surface. Making a point location query at variable resolution on an MT means reporting the coarsest triangle of the MT that "contains" a given query point, and that satisfies a given threshold.

In general, the query point shall not be given by its coordinates in three-dimensional space, but rather by some means of *pointing at* the surface. This is possible, for instance, with surfaces that can be projected on a plane, like terrains. The query point is specified by its projection on the xy plane, while its elevation z is unknown: indeed, elevation is part of the output of the point location operation. The same can be done by using either cylindrical or spherical coordinates for surfaces that can be projected on either a cylinder or a sphere, respectively.

In these cases, the DAG encoding the MT naturally provides a search structure. The query point is located first in the mesh formed by triangles emanating from the root, and the DAG is traversed following a path of arcs labelled by triangles that contain the query point, until the current triangle satisfies the threshold function [5]. The algorithm involves a search for the triangle containing the query point in each visited node. The time complexity depends on the number of arcs leaving each of the visited nodes, and on the length of the traversed path.

The cost of answering a point location query is, in the worst case, proportional to the product of height and width of the MT. If the MT has logarithmic height and bounded width, point location is performed in optimal logarithmic time in the worst case. In some situations, where logarithmic height and/or bounded width is not warranted, amortized analysis can be applied to show that logarithmic time can be achieved in the expected case, though the cost in the worst case can be higher. These cases occur for structures built by some of the algorithms described in the following Section.

5 Building a multiresoluton model

It follows from the arguments presented in the previous section that an MT, in order to be effective, should have the following characteristics, which are to be taken into account in designing a construction algorithm:

- good compression ratio: the size of the mesh extracted to achieve a given accuracy should be relatively small;

- small overhead factor: the number of triangles traversed to extract a given mesh should be not much higher than the size of the output; this property is implied by linear growth;

- bounded width: the number of arcs visited at each step to direct a path thorugh the DAG should be small;

- logarithmic height: a path on the DAG should always be short.

An MT can be easily built by dynamic algorithms that perform successive local modifications on an initial mesh Σ. A *local modification* is an operation that replaces a group of triangles of Σ with another group of triangles approximating the same portion of surface. There is a flourishing literature on mesh simplification algorithms which can be used to this purpose (see, e.g., [2, 4, 9, 10, 11, 19]). Such algorithms rely on bottom-up strategies that start from a mesh at full resolution, and successively reduce resolution, by local modifications that substitute groups of triangles with other groups of smaller size. Opposite strategies that start from a mesh at coarse resolution and progressively refine it are popular in terrain approximation [8], but cannot be adapted easily to general surfaces. All such algorithms follow a *greedy scheme* that can be summarized as: *perform local modifications one after the other; each time, select the modification that is likely to give the smallest increase in the approximation error.*

Greedy heuristics give effective solutions in practice: they can produce MTs with good compression ratios. On the other hand, they may have theoretical drawbacks because many of them do not warrant linear growth, logarithmic height, and bounded width. Different construction techniques, like the one proposed in [5], produce theoretically efficient structures with linear growth, logarithmic height, and bounded width, but they might result less effective in practice, since they give worse compression ratios.

In the following, we describe and analyze four variants of a construction strategy based on vertex decimation. The general scheme for the construction of an MT $\mathcal{M}$ from a dataset $\mathcal{D}$ of n sites is valid for all variants, and it can be described as follows:

1. start from a mesh Σ_S having vertices at all points of $\mathcal{D}$; initialize $\mathcal{M}$ by creating its source and its drain, and joining them with one arc for each triangle of Σ_S;
2. while vertices can be removed from the current minimal triangulation Σ_S of $\mathcal{M}$ do the following:
   - select a set V of vertices to be removed;
   - find the holes left in Σ_S by removing all triangles incident at vertices of V;
   - for each such hole:
     – create a new node n;
     – every triangle t inside the hole labels an arc a emanating from the source: remove t from the label of a (removing a if its label becomes empty) and add t to the label of a new node a' emanating from n and having the same destination as a;
     – re-triangulate the hole;
     – create an arc from the source to n labeled with newly created triangles (see Figure 2).

For the re-triangulation of holes we adopt the method of [2] for general surfaces, and the Delaunay triangulation of the plane projection [14] for terrain surfaces.

Variants of the bottom-up technique depend on which vertices are selected for deletion at each iteration. The selection mechanism adopted to obtain set V effects not only the internal structure and the time complexity of each specific algorithm, but also the characteristics of the MT produced.

Variant A1. This algorithm was proposed by de Berg and Dobrindt [5] to build their hierarchical triangulation, which is a special case of an MT. Set V is selected at each iteration as an arbitrary maximal independent set of vertices of bounded degree (at most b) from the current minimal triangulation Σ_S. At each iteration, a new node is created for each vertex of V.

The resulting MT has linear growth, bounded width, and logarithmic height. The computational complexity of the algorithm (apart from computation of the initial triangulation) is $O(n)$, but a further $O(n \log n)$ time is needed to evaluate the accuracies of all triangles in the MT.

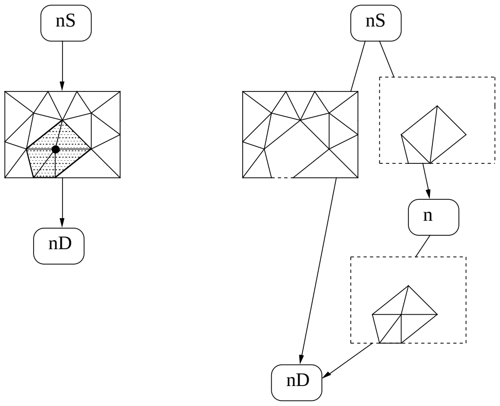

Figure 2: One step in building an MT through mesh simplification: n is the new node corresponding to the removal of the black vertex from the current mesh.

Variant A2. Same as Variant A1, but V is built with a greedy technique as follows: initialize V with the vertex whose elimination causes the smallest increase in error among those that have a degree smaller than b; add other vertices iteratively, until possible: at each iteration, select the vertex causing the smallest possible error, among those that have a degree smaller than b, and are independent of the vertices already in V.

Since V is a maximal independent set of vertices with bounded degree, as in Variant A1, the resulting MT has the same characteristics. Moreover, the additional heuristics, which makes the selection depend on the error caused by removing a vertex, helps maintaining "important" vertices close to the source, while less relevant vertices are gradually discarded while moving towards the drain.

The greedy selection of vertices requires that error caused by removing a vertex of the current minimal triangulation is known at each iteration. The error can be evaluated at each vertex by simulating its deletion each time its surrounding triangles change. In order to achieve efficiency, we use a bucketing technique similar to that used in [12] for dynamic triangulations, i.e., we associate with each triangle t in the minimal triangulation the list of vertices already deleted that lie inside t. Moreover, we maintain vertices of the minimal triangulation in a priority queue that supports fast selection of the vertex causing the smallest error. It can be shown that, by using such techniques, the computational complexity of building an MT through Variant A2 is $O(n \log n)$, i.e., the same as with Variant A1. Technical details of the analysis are omitted for brevity.

Variant A3. At each iteration, V consists of just one vertex, namely the vertex that causes the smallest error increase, among those having degree smaller than b. In this case, the resulting MT still has linear growth and bounded width, while it might not have logarithmic height. Therefore, point location on this structure can take as much as $O(n)$ time in the worst case. However, following [14], it is easy to show through amortized analysis that the expected cost of point location is still $O(\log n)$.

The elimination of important feature points is further delayed, since selection is performed on the basis of a criterion that is only related to error increase caused by each deletion. Indeed, in Variant A2, the need of having a maximal independent set can force the algorithm to delete points that cause a high error increase at early stages of the construction.

Error evaluation is performed as in Variant A2, hence both bucketing and priority queue are necessary, and the same analysis applies. Thus, the time complexity of Variant A3 is again $O(n \log n)$.

Variant A4. Same as variant A3, but the vertex causing the minimum error increase is selected, regardless of its degree: this corresponds to the criterion adopted in [2]. This variant adopts a selection strategy that is completely error-driven, therefore the greedy heuristic approach, i.e., trying to discard less important points at early stages, is stronger than in any other previous variant. However, the resulting MT in the worst case may have none of the desired properties in terms of growth, height, and width.

The same tools used in the previous two cases (bucketing and priority queue) lead to a time complexity of $O(n^2 \log n)$ in the worst case. The increase of an n factor is due to the fact that the growth of the structure might be quadratic. This means that this variant takes longtime to run if and only if it produces an MT that is qualitatively poor. It should be remarked that this worst case behavior is not likely to happen in practical cases.

6 Experimental Results

All construction algorithms described in this abstract have been implemented and tested for the case of terrain surfaces; algorithm A4 has been implemented also for general surfaces. The algorithm for extracting a mesh at a given resolution has also been impemented and tested both on terrain surfaces and on generic surfaces.

In order to compare the different variants of the construction algorithm, we have run experiments on different terrain datasets. Here, we show some results obtained on two datasets made available from the U.S. Geological Survey: Sen Bernardino is a regular grid of 128×128 elevation data, while dataset Mount Marcy is a regular grid of 160×160 elevation data.

Table 1 summarizes the results of running the four bottom-up construction algorithms. Variants A1-2-3 have been applied by using a bound $b = 11$ for the maximum degree of vertices that can be eliminated at each step. For each variant, we list (a) the size of the maximal triangulation (obtained by subdividing each pixel square of the data grid into two right triangles), (b) the total size of the resulting MT, (c) its absolute height, (d) the ratio between its height and the logarithm of its size, and (e) its maximum width.

Variant A4 gives the best result in terms of space occupancy (size of the MT), though the gain with respect to Variant A1 (the worst one) is only about $7-8\%$. Variants A1 and A2 give much better results in terms of the maximum height of the MT, while the width is fixed for variants A1-2-3, and it is slightly larger for variant A4. On the basis of these results, we can expect that a point location algorithm runs about three to four times faster on an MTs built with Variants A1 or A2, than on an MT built with Variants A3 or A4.

Table 2 shows results related to the extraction of a triangulation at thresholds 1%, 2%, 5%, 10%, and 20% of the height range in the dataset, respectively, where the threshold is assumed constant over the whole domain: for each variant, we list the ratio between the size of the output and the size of the maximal triangulation (compression factor), and the ratio between the triangles visited by the extraction algorithm and the number of triangles in the output (overhead factor).

San Bernardino

Variant	(a)	(b)	(c)	(d)	(e)
A1	32258	91642	29	2.60	11
A2	32258	86181	27	2.38	11
A3	32258	89170	90	7.89	11
A4	32258	84971	89	7.84	13

Mount Marcy

Variant	(a)	(b)	(c)	(d)	(e)
A1	50562	144082	30	2.68	11
A2	50562	139523	28	2.36	11
A3	50562	142017	98	8.26	11
A4	50562	134565	123	10.42	12

Table 1: Parameters of the MT built through the four bottom-up variants.

In terms of compression, Variants A2-3-4 are almost equivalent, and perform better than variant A1, especially at low resolution. Note that, on the other hand, the overhead factor is almost the same for all variants and thresholds (on average, performance is slightly better for cases A2 and A4). On the basis of these experiments, we can conclude that variant A2 gives the best overall performance.

In Figure 3 we show a mesh representing Mount Marcy at variable resolution, extracted through the algorithm described in Section 4.1 from an MT built with algorithm A4. The resolution is linearly decreasing with distance from a viewpoint. In Figure 4 we show variable resolution representations of some free-form surfaces: the resolution is exponentially decreasing with distance from a focus point. The numbers of original and extracted triangles, and the extraction times are shown; times are obtained on a SGI Indigo2 10000.

7 Conclusions

Several multiresolution models of triangle meshes have been developed in the literature because of their wide applicability. The Multi-Triangulation is a comprehensive model that generalizes over all other models based on triangle decompositions. See [7, 18] for a discussion of how existing multiresolution models can be interpreted as special cases of MTs. We have considered algorithms for the extraction of representations at variable resolutions, and for point location on an MT; an interruptable algorithm for extracting representations at variable resolution from the MT has been presented. Four variants of a construction technique based on decimation have been considered on terrain modeling. The impact of the technique adopted to build the model on the performance of the algorithms for the two basic operations has been evaluated both from a theoretical and an experimental point of view. The best overall performance is achieved by a method that progressively removes independent set of vertices of small degree, selecting with an error-based criterion. Although decimation is a standard technique in the literature, this variant was never adopted previously.

We are currently developing a library for building and manipulating a multiresolution surface description in con-

San Bernardino - compression factor

Error	A1	A2	A3	A4
0%	1.0	1.0	1.0	1.0
1%	1.0	1.0	1.0	1.0
2%	1.0	1.0	0.999	0.995
5%	0.99	0.98	0.99	0.97
10%	0.79	0.72	0.78	0.73
20%	0.30	0.29	0.8	0.6

San Bernardino - overhead factor

Error	A1	A2	A3	A4
0%	2.88	2.71	2.80	2.63
1%	2.88	2.63	2.81	2.93
2%	2.88	2.72	2.80	2.93
5%	2.76	2.74	2.80	2.67
10%	2.85	2.60	2.77	2.62
20%	2.83	2.73	2.75	2.72

Mount Marcy - compression factor

Error	A1	A2	A3	A4
0%	1.0	1.0	1.0	1.0
1%	1.0	1.0	1.0	0.98
2%	0.99	0.99	0.97	0.96
5%	0.60	0.56	0.11	0.13
10%	0.47	0.07	0.048	0.035
20%	0.27	0.02	0.014	0.015

Mount Marcy - overhead factor

Error	A1	A2	A3	A4
0%	2.88	2.79	2.84	2.69
1%	2.75	2.85	2.89	2.71
2%	2.77	2.93	2.80	2.69
5%	2.96	2.75	2.82	2.80
10%	2.86	2.73	2.77	2.78
20%	2.84	2.69	2.65	2.67

Table 2: Results for the extraction of a triangulation at various error thresholds.

nection with various reconstruction problems: reconstruction from a CAD description of the boundary surface of a solid object, reconstruction from a set of measurements on a free-form surface, when a set of points, or a set of contours, or a dense triangle mesh are given. We have also extended the multiresolution triangulation to arbitrary dimensions, and we are currently developing a prototype for encoding and manipulating volume data at different LODs, based on a 3D extension of the data structures and algorithms described in this paper.

References

[1] P.K. Agarwal, S. Suri, 1994, Surface Approximation and Geometric Partitions, *Proceedings ACM-SIAM Symposium on Discrete Algorithms*, pp. 24–33.

[2] A. Ciampalini, P. Cignoni, C. Montani, R. Scopigno, 1996, Multiresolution Decimation Based on Global Error, *Technical Report* TRC96-021, CNUCE-CNR, Pisa, Italy. (To appear on *The Visual Computer*).

[3] P. Cignoni, E. Puppo, R. Scopigno, 1995, Representation and Visualization of Terrain Surfaces at Variable Resolution, *Proceedings International Symposium on Scientific Visualization*, R. Scateni, Ed., World Scientific, Cagliari (Italy), September 1995, 50–68. (Extended version to appear on *The Visual Computer*).

[4] J. Cohen, A. Varshney, D. Manocha, G. Turk, H. Weber, P. Agarwal, F. Brooks, W. Wright, 1996, Simplification Envelopes, in *Computer Graphics* (SIGGRAPH '96), pp.119-128.

[5] M. de Berg, M., K.T.G. Dobrindt, 1995, On the Levels of Detail in Terrains, *Proceedings 11th ACM Symposium on Computational Geometry*, c26–c27. Also available in long version as: *Technical Report* UU-CS-1995-12, Utrecht University, Dept. of Computer Science, April 1995. Available as ftp: //ftp.cs.ruu.nl/ pub/RUU/CS/techreps/ CS-1995/1995-12.ps.gz.

[6] L. De Floriani, E. Puppo, 1995, Hierarchical Triangulation for Multiresolution Surface Description, *ACM Transactions on Graphics*, 14, 4, pp. 363-411.

[7] L. De Floriani, E. Puppo, P. Magillo, 1996, A Formal Approach to Multiresolution Modeling, *Theory and Practice of Geometric Modeling*, W. Straßer, R. Klein, R. Rau (Editors), Springer-Verlag.

[8] Fowler, R.J., Little, J.J., 1979, Automatic Extraction of Irregular Network Digital Terrain Models, *Computer Graphics* (SIGGRAPH'79), 13-3, pp. 199-207.

[9] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuelzle, 1993, Mesh Optimization, *Computer Graphics Proceedings* (SIGGRAPH '93), pp. 19-26.

[10] H. Hoppe, 1996, Progressive Meshes, *Computer Graphics Proceedings* (SIGGRAPH '96), pp.99-108.

[11] A.D. Kalvin, R. Taylor, 1996, Superfaces: Polygonal Mesh Simplification with Bounded Error, *IEEE Computational Geometry and Applications*, 16(3), 64-77.

[12] T. Kao, D. Mount, A. Saalfeld, 1991, Dynamic Maintenance of Delaunay Triangulations, *Proceedings Auto-Carto 10*, pp.219-233.

[13] R. Klein, W. Straßer, 1996, Generation of Multiresolution Models from CAD Data for Real Time Rendering, *Theory and Practice of Geometric Modeling*, W. Straßer, R. Klein, R. Rau (Editors), Springer-Verlag.

[14] Knuth, D.E., Guibas, L.J., Sharir, M., 1992, Randomized Incremental Construction of Delaunay and Voronoi Diagrams, *Algorithmica*, 7 (4), pp. 381-413.

[15] J. Lee, 1991, Comparison of Existing Methods for Building Triangle Irregular Network Models of Terrain from Grid Digital Elevation Models, *International Journal of Geographical Information Systems*, 5,3, pp.267-285.

[16] P. Lindstrom, D. Koller, W. Ribarsky, L.F. Hodges, N. Faust, G.A. Turner, 1996, Real-time, Continuous Level of Detail Rendering of Height Fields, *Computer Graphics Proceedings (SIGGRAPH'96)*, pp.109-118.

[17] E. Puppo, 1996, Variable Resolution Terrain Surfaces, *Proceedings Eight Canadian Conference on Computational Geometry*, pp.202-210.

[18] E. Puppo, 1996, Variable Resolution Triangulations, *Technical Report* N.12/96, Istituto per la Matematica Applicata, C.N.R., Genova, Italy (submitted for publication). Available as http: //www.ima.ge.cnr.it/ STAFF/ puppo-en.html.

[19] W.J. Schroeder, J.A. Zarge, W. Lorensen, 1992, Decimation of Triangle Mesh, *ACM Computer Graphics* (Proceedings SIGGRAPH '92), 26, 2, pp.65-70.

[20] R. Tamassia and fifteen others, 1996, Strategic Directions in Computational Geometry Working Group Report, *ACM Computing Surveys*, 28, 4.