Mathematical and Computer Modelling 49 (2009) 1012–1020

Contents lists available at ScienceDirect

Mathematical and Computer Modelling

journal homepage: www.elsevier.com/locate/mcm

Four-triangles adaptive algorithms for RTIN terrain meshes

J.P. Suárez [a,*], A. Plaza [b]

[a] Department of Cartography and Graphic Engineering, University of Las Palmas de Gran Canaria, 35017-Las Palmas de Gran Canaria, Spain
[b] Department of Mathematics, University of Las Palmas de Gran Canaria, Spain

ARTICLE INFO

Article history:
Received 18 January 2008
Accepted 27 February 2008

Keywords:
Longest edge
Terrain modeling
Meshes

ABSTRACT

We present a refinement and coarsening algorithm for the adaptive representation of Right-Triangulated Irregular Network (RTIN) meshes. The refinement algorithm is very simple and proceeds uniformly or locally in triangle meshes. The coarsening algorithm decreases mesh complexity by reducing unnecessary data points in the mesh after a given error criterion is applied. We describe the most important features of the algorithms and give a brief numerical study on the propagation associated with the adaptive scheme used for the refinement algorithm. We also present a comparison with a commercial tool for mesh simplification, *Rational Reducer*, showing that our coarsening algorithm offers better results in terms of accuracy of the generated meshes.

1. Introduction

Terrain data consist of a finite set of points in the plane with associated elevation values. In Geographic Information Systems, terrains are modeled by defining a decomposition of the height-field domain placing vertices at the data points, and a space of functions in order to piecewise interpolate elevations of the domain decomposition.

In this paper we present a simple and efficient refinement algorithm for triangle modeling of hierarchical terrain meshes. The refinement problem can be described as any technique involving the insertion of at least one additional vertex in order to produce meshes with increased accuracy. The *accuracy* of a terrain model refers to the error made in representing a terrain, and can be related to either geometric measurements (e.g., elevation, distance functions), or non-geometric attributes (e.g., soil type, land use). By hierarchical meshes we mean a sequence of nested triangular grids $T = \{\tau_1 < \tau_2 < \cdots < \tau_n\}$ where τ_1 is the initial mesh and τ_n the finest grid in the sequence and such that the nodes of τ_{n-1} are a subset of nodes in τ_n. Refinement algorithms for mesh-based surface modeling have been used in earth modeling applications. For example, the *Refinement Gridding* commercial surface modeling algorithm for the oil industry has been presented in [19] and a mesh refinement scheme for geological structures has been given in [18].

The inverse problem, coarsening (also simplification, decimation or derefinement), reduces the number of points in the mesh and its advantage is that it may remove non-significant vertices in the approximated model. We present a coarsening algorithm that combined with the refinement scheme may constitute a valuable tool for adaptive representation of digital elevation models. Adaptive algorithms consisting in the refinement/coarsening process are particularly useful to approximate terrains where a dynamic environment is considered. By a dynamic environment we mean a computer application dealing with approximated meshes that eventually depend on time, a prescribed observed-terrain distance, a numerical solution within the mesh etc. Such applications include Interactive Visualization and Level of Details techniques [6,8,12], Real-time rendering [6], and Finite Element computations, see [3,5,18]. For instance, in landscape visualization, the accuracy needed in the various parts of a terrain depends on the distance from the viewpoint [6,8].

* Corresponding author. Tel.: +34 928 457268; fax: +34 928 451872.
E-mail address: jsuarez@dcegi.ulpgc.es (J.P. Suárez).

0895-7177/$ – see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mcm.2008.02.012

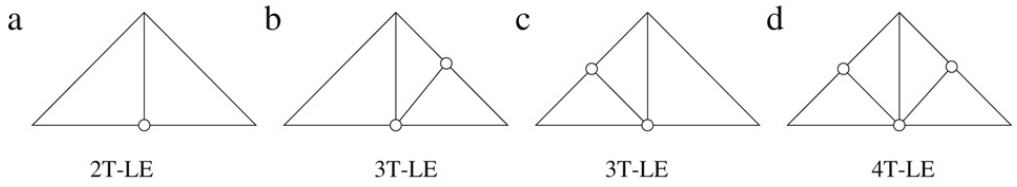


Fig. 1. Longest-edge triangle subdivisions.

A multigrid scheme that uses adaptive algorithms for the simulation of contaminant transport in 3D subsurfaces has been recently published in [5].

Our algorithms follow the Right-Triangulated Irregular Network (RTIN) scheme and differ from the algorithm of Evans et al. [4] in that we split target triangles in four instead of two, producing well graded and smooth meshes. Our splitting scheme is similar to the one by Velho et al. [17], although they use subdivision methods for adaptive tessellation of parametric or implicit surfaces.

The novel contributions of our present work are as follows: (i) original algorithms for mesh refinement and coarsening, that are improved versions compared to those published in [13], (ii) a numerical study regarding the propagation of longest-edge based refinement for RTIN meshes, (iii) a comparative study with a commercial package in which our coarsening algorithm is proven superior regarding accuracy of the generated meshes.

1.1. Related work

In terrain modeling there are mainly two data structures for the arrangement of triangles in the terrain domain: a regular grid of digital elevation model (DEM) and a Triangulated Irregular Network (TIN), see for instance [4,6,9,13]. The simplest choice is a DEM, where the values represent the elevation which is stored at regular intervals. This data structure is not adaptive and so it can not capture with accuracy the regions with higher irregularity, such as ridges or valleys.

The other alternative is the use of a TIN. The main difference is the variability of the point distribution in the terrain domain. The elevation points do not follow any regular pattern and therefore adaptivity is achieved when a greater amount of points are used in the regions of interest. However, a TIN requires more storage for the same number of sampled points.

Right Triangulated Irregular Networks, introduced by Evans et al. [4] are meshes of right-angled triangles. They are more regular structures than a TIN and more flexible than a regular grid. The main benefit of the RTIN approximation is not in providing a single approximation to a surface, but rather in providing a framework of many approximations at varying levels of detail. Moreover, the time and storage complexity is very competitive with the other approaches, considering that RTIN is a hierarchy based approximation. Pajarola et al. [9] presents QuadTIN, a novel approach to impose quadtree-based multiresolution triangulation hierarchy on arbitrary irregular sets of elevation points. Using a number of Steiner points this triangulation hierarchy allows fast view-dependent Level of Detail (LOD) triangulation, triangle strip generation and real-time rendering.

A complete review and comparison of mesh simplification algorithms can be found in [1]. Regarding commercial packages for reducing the number of polygons in a scene, *Rational Reducer* (RR) [7] is a TIN-based tool commonly used in 3D graphics, virtual scenes, terrain meshes etc. RR is developed by SIM (*System In Motion*). The fact that RR reduces the complexity of 3D models and scenes makes it particularly attractive to be used in real-time visualization environments, surface representation, etc. RR has a knowledge database obtained by running a genetic evolution system of different 3D models, thereby calculating the optimal set of parameters for its reduction algorithms for any type of model likely to be encountered. It works iteratively, removing the polygons which have the least visual impact on the 3D model or scene. When the angle between the outside normal vectors for two adjacent triangles is very small then the vertex points are combined to form one single vertex. This is performed by a recursive search through the triangular model structure, so that triangles can be recursively combined should their neighbor have a small angular separation (although steps are used to avoid flattening a sphere). *RR* uses a Delaunay approach to construct the triangle meshes. As a robust and fast tool for mesh simplification, we use Rational Reducer in this work to perform a comparison with our coarsening algorithm. Nevertheless, it should be pointed out that *RR* performs the reduction delivering at the end a quasi uniform organization of vertices. Conversely, our approach will follow an adaptive scheme that implies an unstructured organization of vertices in the simplified meshes.

Several approaches for partitioning triangles have been studied. They are important to triangle mesh modeling as far as terrain mesh quality and computational efficiency are concerned. The most simple approach is the Two Triangles Longest Edge (2T-LE) bisection that connects the midpoint of the Longest Edge to the opposite vertex, Fig. 1(a). The Four Triangles Longest Edge subdivision (4T-LE) [10,11], bisects a triangle into four subtriangles: the original triangle is first subdivided by its longest edge as before and then the two resulting triangles are bisected by joining the new midpoint of the longest edge to the midpoints of the remaining two edges of the original triangle, Fig. 1(d).

In [13] previous versions of 4T-LE algorithms for RTIN terrain meshes have been presented. In that work, an application to generate Level of Details of terrain meshes is illustrated and numerical experiments to prove their use in real time scenarios are provided. However, we present here new improved versions of the 4T-LE algorithms and show new studies regarding refinement propagation and accuracy of meshes.

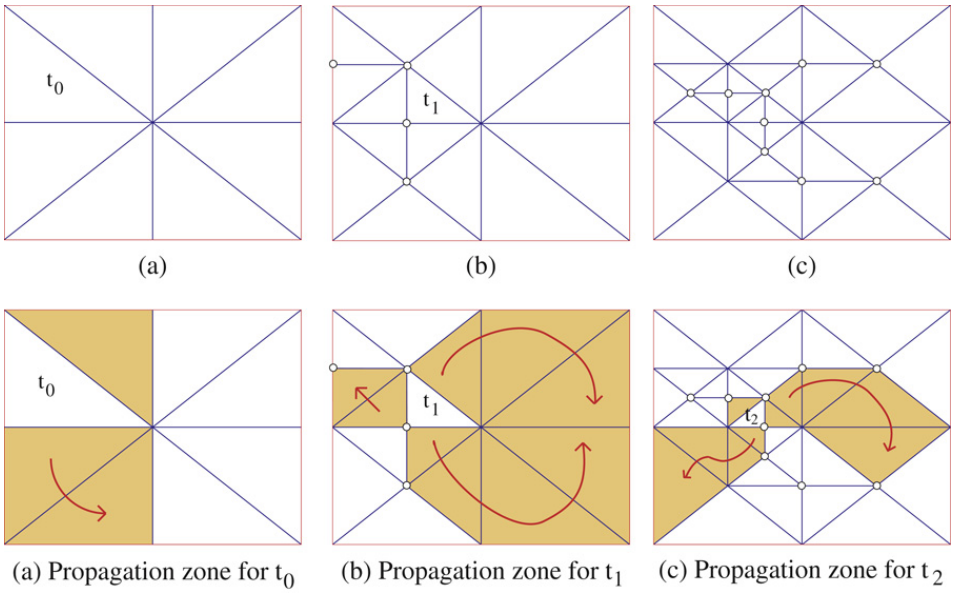


Fig. 2. 4T-LE Local refinement samples, (a) Initial triangle mesh where triangle t_0 is marked for refinement, (b) Refinement of t_0 and selection of triangle t_1 for a new refinement, (c) Final mesh with the refinements completed. Second row shows the propagation zone for different triangles.

2. Longest-edge based refinement/coarsening algorithms

The local refinement of triangle meshes involves two main tasks. The first is the subdivision of the target triangles and the second is the refinement propagation to assure mesh conformity. We say that a mesh is *conforming* if any adjacent elements share an entire edge or a common vertex. Conversely, local coarsening follows the inverse tasks: (1) remove target vertices and (2) preserve conformity by removing additional triangles. Refinement/coarsening techniques can be used individually or combined as a general tool to get adaptive representations with meshes. The vast applications of such techniques include Finite Element Method, Level of Details, or Progressive Representations.

We follow the 4T-LE scheme for triangle subdivision throughout this work, see Fig. 1. Since a major step in local refinement is to ensure mesh conformity, additional subdivision patterns are needed to get a matching triangulation. See Fig. 1(a)–(c). We point out that a diagonal swap of the LE bisector in the 4T-LE yields the variant scheme 4T-Self Similar (4T-SS). Both schemes can be used indistinctly in RTIN as they always produce subtriangles congruent to the parents.

Local subdivision of a 'target' element by 4T-LE introduces new midedge nodes on the element boundary. In order to ensure conformity of the resulting mesh, the refinement is extended to additional triangles. Here we consider a progressive longest edge approach. These adjacent LE subdivision patterns always bisect the adjacent triangle by the midpoint of the longest-edge and then, if necessary, one or two of the resulting subtriangles are also bisected. We refer to the set of these additional triangles as the *propagation* zone.

Therefore, a local mesh refinement based on a longest edge strategy causes splits that propagate to growing edges, and thus to larger triangles or larger angles. This ensures the construction of graded and smooth irregular triangulations. In Fig. 2 two samples of the 4T-LE refinement and propagation are shown in which triangles t_0 and t_1 are split following 4T-LE subdivision. Although propagation seems to be an ominous problem that may affect refinement efficiency, it has been studied recently in [15] that asymptotically the propagation extends to just a few adjacent triangles. A typical number of these triangles to be refined has been observed to be around seven. We continue this study for RTIN meshes giving the corresponding propagation values.

Coarsening is the opposite of refinement; we coarsen locally by undoing a single edge bisection that can be seen as triangle merging. Unlike refinement, coarsening does not require propagation further into the mesh to maintain a conforming triangulation. Furthermore, a coarsening step may make other coarsenings possible.

2.1. Refinement algorithm at a glance

The benefit of the manner in which the 4T-LE refinement proceeds in RTIN meshes is that it produces only right-angled triangles. In [10,11] it has been proved that the 4T-LE scheme improves the shape of obtuse triangles when repeated refinement is applied to them. This is a remarkable feature in the refinement of general types of triangulations, e.g. TIN's.

We will consider the following statement of the problem: given a RTIN mesh τ and a set of triangles $R \subset \tau$ to be refined, perform the refinement process, which concerns the 4T-LE subdivision of triangles in R plus the propagation refinement in $\tau - R$. The refinement algorithm first subdivides target triangles by the 4T-LE and then the non-conforming triangles using the longest edge strategy. A non-conforming triangle is a triangle with some bisected edges that is not compatible for performing a longest-edge triangle subdivision as in Fig. 1.

Algorithm RTIN-Refinement (τ, R)
/* *Input: RTIN mesh τ. Set of triangles R to be refined*
/* *Output: new refined mesh τ*
1: Perform 4-LE subdivisions of triangles in R
2: S=list of non-conforming triangles in $\tau - R$
3: For each triangle $t_i \in S$ **do**
  4: While t_i is not conforming **do**
    /* *Mark the longest edge of triangle t_i to be refined:*
    Mark Longest-Edge of t_i
    /* *Update t_i to the triangle that is neighbor by the Longest-Edge:*
    t_i = LEneighbor(t_i)
  End
End
5: For each triangle t with some marked edges **do**
  /* *A subdivision pattern based on the Longest-edge is applied:*
  Perform LE subdivision of t
End

The algorithm above first subdivides the target triangles in R following the 4T-LE scheme, step 1. Step 2 constructs a list S with non-conforming triangles. These triangles contain some hanging nodes, hence a propagation of the refinement is needed to guarantee the conformity of the mesh. Step 3 activates the propagation refinement by selecting each non-conforming triangle and properly splitting neighboring triangles by the longest edge. An edge-marking in step 4 is performed that iteratively scans the neighboring triangles and marks edges to be subdivided at midpoint. This ensures the mesh conformity through the refinement process. As soon as the split propagation reaches the mesh boundary or two triangles sharing their common longest edge, the propagation stops. It should be noted that splitting a triangle t does not result in the splitting of a triangle smaller than t (in the sense of smaller Longest-Edge) and so the operation is guaranteed to terminate, see a proof in [16]. Depending on the number of marked edges, the triangles are adequately subdivided into two, three or four triangles, step 5 in above algorithm.

An example of the use of the refinement algorithm can be seen in Fig. 2, where triangles t_0 and t_1 are refined.

2.1.1. Data structure and error criterion for the RTIN-refinement algorithm

A graph data structure, edge-based, for the 4T-LE subdivision scheme that maintains an element dependency relation has been presented in [14]. The approach of this data-graph structure is similar to that in [8] although the latter uses vertex dependencies. In both cases the dependency relation in mesh elements implies a cost to update such element relations. Our data structure is based on a list of triangles given by their vertices (using a local numbering of vertices for each triangle) and a list of vertices. A local numbering for each triangle is considered so that vertices labeled with 1 and 2 determine the longest edge. It is worth noting that the basic adjacency relationship used in the above refinement algorithm is determined by the Longest Edge of each triangle. Hence, from the implementation point of view, we store in the first position of each triangle the two vertices that surround the longest edge (vertices 1 and 2) and this is consistently maintained in each refinement step. We remark here that no explicit neighboring function like that used in Evans et al. [4] or Pajarola et al. [9] needs to be used.

The complexity of the RTIN-Refinement algorithm is linear in the number of triangles — similar to that in [14]. It is clear that the cost of refining an element depends on the number of triangles taking part in the refinement propagation. Empirical results for the RTIN propagation studied in the next section show that the cost per element of the refinement algorithm approaches in average a fixed acceptable value ($M1 \approx 7$) as the number of refinement elements increases. In this regard, the cost per element can be considered constant in average.

The refinement criterion, i.e. whether to split a triangle or not, is based on whether the triangle approximates the corresponding part of the original high resolution mesh well enough. For a different scenario of terrain visualization a view-dependent refinement may be preferred, and so, the relative position and orientation of the viewer and the triangle must be considered. The error metric used to test RTIN-Refinement is the L_∞ norm of the vertical distance (terrain elevation dimension) of vertices to the triangulated surface. The set of vertices selected for the refinement is defined by the specified error criterion. For a complete reference of error criteria see [1]. We adopt here an error criterion based on a height threshold:

$$E(v) = \left| \frac{h(a) + h(b)}{2} - h(v) \right| \tag{1}$$

where a and b are the surrounding vertices of vertex v, h the height at each vertex and ϵ the threshold error. If for a given mesh τ_i, we have that $E(v) > \epsilon$, then vertex v will be included in the list of vertices to be refined.

This error metric, similar to that used in [9], provides a conservative LOD triangulation. The top-down vertex selection and triangle subdivision can be stopped at nodes below a given tolerance. Because this error metric is not guaranteed to be monotonic, cracks in the triangulation have to be avoided by propagating subdivision following the refinement patterns already shown in Fig. 1. Thus, propagated subdivisions must verify the error threshold as well. Note that the propagation strategy of the Longest Edge refinement avoids the cracks in the refined triangulation.

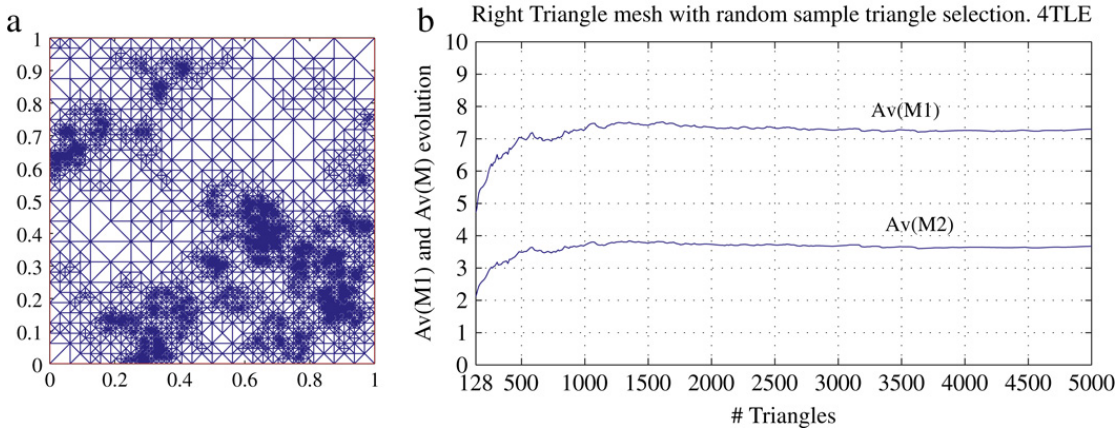


Fig. 3. Example of successive 4T-LE refinement and propagation evolution, $M1$ and $M2$.

2.1.2. Refinement propagation in the 4T-Longest Edge refinement algorithm

Local subdivision of triangles by 4T-LE introduces mismatches on the neighboring elements. Therefore, in order to ensure conformity of the resulting mesh, the refinement is propagated to additional triangles. Here we consider a progressive longest edge approach as in [4,11,15]. We are interested in the behavior of this propagation since mesh smoothness and algorithm efficiency may be compromised. Mesh smoothness is the feature that achieves an acceptable transition among elements such that no virtual artifacts or discontinuities in the terrain can be visualized.

In [4] a worst case for the split propagation termination is given by $O(\log\ n)$, n being the difference in levels of a generated RTIN quad-tree. In adaptive mesh refinement however, a small percentage of the elements is refined at each refinement step and several refinement stages are carried out during the terrain approximation which imply a recursive refinement of some elements. Also note that repeated local refinement stages are often limited to specific subregions, such as regions with high gradient.

The propagation problem has been studied in [15] by means of two metrics defined as follows: (1) *Conformity Neighborhood*: for refinement of a triangle t of a mesh τ, the set of triangles in $\tau - t$ that need to be refined due to the conformity process for t. For convenience, the size of Conformity Neighborhood, denoted by $M1(t)$, is used. (2) Since the conformity process extends at most over the three edges of a triangle t, the propagation defines at most three paths of ordered triangles. $M2(t)$ is the maximum number of triangles of the three resulting paths. Numerical experiments carried out in this work for RTIN meshes have shown that in average propagation approaches fixed values, $M1 = 7.3$ and $M2 = 3.8$, hence the longest-edge propagation performs much better than $O(\log\ n)$.

We give here an example in which a uniform RTIN mesh with 128 triangles is recursively refined following a random selection of triangles. Three hundred of 4T-LE splits are randomly performed to the initial right triangle mesh. The resulting mesh is shown in Fig. 3(a) and the evolution of $M1$ and $M2$ (in average) is shown in Fig. 3(b). Some other tests using different RTIN input meshes revealed analogous results.

It is worth noting that the favorable case for propagation is to consider a uniform RTIN at the beginning of the 4T-LE refinement. This can be seen in Fig. 3(b) for the initial mesh with 128 triangles. In this case $M1 = 5$ and $M2 = 2$. Initially, for a few refinement steps the metric values progressively increase. As refinement increases, the averages of $M1$ and $M2$ reach the values 7.3 and 3.8, respectively.

The propagation values for RTIN meshes are similar to the asymptotic values known for arbitrary meshes ($M1 \approx 7.2$ and $M2 \approx 3.6$) [15]. The 4T-LE subdivision used in this work makes a natural propagation of the refinement around the three edges of a given triangle, and affects up to seven triangles in average. This promotes mesh smoothness and gives also an idea of the quality of the refinement propagation for terrain meshes.

2.2. The Coarsening algorithm for RTIN meshes

The mesh input for the algorithm is a $(2^k + 1) \times (2^k + 1)$ regular spaced grid of heights. This represents k levels in the hierarchy corresponding to k different RTIN meshes $\{\tau_1, \tau_2, \ldots, \tau_k\}$.

The coarsening procedure performs vertex removal based on a specified error criterion. For a reference of error criteria see [1]. We adopt here an error criterion based on the height threshold given in Eq. (1): if for a given mesh τ_i, see Fig. 4(b), we have that $E(v) < \epsilon$, then vertex v is to be removed. For clarity, we separate in one task the error checking and in the other the coarsening itself. In the error checking task, a Mark/Unmark matrix C is constructed that associates with each vertex a value/flag that indicates whether a vertex should be removed (Mark) or preserved (Unmark). It should be noted that the algorithm itself does not depend on the coarsening condition, so that matrix C can be computed elsewhere – outside the algorithm – and then used as an input for the coarsening algorithm. In spite of the simplicity of this error criterion it is satisfactory for most of the cases. Our experience has shown that a careful selection of the threshold allows to preserve visual features in the terrains; see numerical results in Section 3.

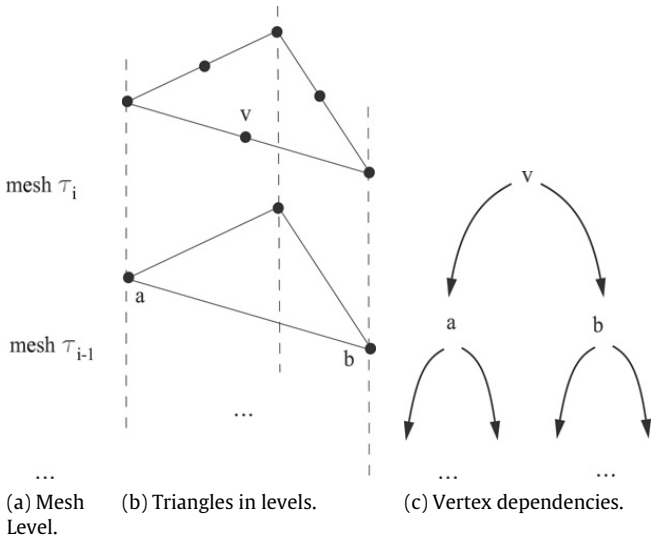


Fig. 4. Mesh levels with vertices dependency tree.

A brief summary of the coarsening algorithm is presented:

(1) A hierarchy of the RTIN meshes is provided, together with a matrix with vertices to be removed.
(2) A backward process that iterates from the finest mesh τ_k to the mesh τ_2 is initiated to perform vertices removal.
(3) Edge re-connection is performed to generate a new output mesh.

The coarsening algorithm proceeds as follows: In step 1, each mesh level $\tau_i \in T$, for $i = k, k-1, \ldots, 2$, is processed. In each iteration i of that loop, only vertices that belong to mesh τ_i are checked (called *proper* vertices). It should be noted that the context of vertices a and b in step 2 is mesh level τ_{i-1}. Step 3 ensures that matching vertices be removed for the sequence of hierarchical meshes, so that consistency between levels τ_i and τ_{i-1} is guaranteed. That is, if a vertex v is not marked for removal, then vertices surrounding it are not removed. This can be viewed as preserving a dependency relation between vertices in the mesh hierarchy. In Fig. 4(c) that dependency relation is shown through a binary tree.

Step 4 performs the final re-triangulation of the remaining nodes. This is a local process implying the triangles in mesh τ_i only. In this step, matrix C is examined so that only unmarked vertices will take part in the re-triangulation. More precisely, the re-triangulation step checks the vertices for each triangle and according to whether they are Unmarked or Marked, a triangle will be eliminated or not from the mesh. At the end, a new conformal sequence of coarsened meshes is obtained. See algorithm below.

Algorithm RTIN-Coarsening (τ_k, C)
/* *Input:* τ_k = *regular spaced mesh with height data,*
/* $C = (2^k + 1) \times (2^k + 1)$ *matrix with vertices marked to be removed*
/* *Output: Coarsened mesh* τ_k
/* *Main loop in the mesh level k:*
1: For $i = k$ to 2 **do**
  2: For each *proper* vertex $v \in \tau_i$ **do**
    Let a, b be vertices $\in \tau_{i-1}$ that surround v
    /* *If a vertex v is not chosen to be removed, neither do a,* $b \in \tau_{i-1}$*:*
    3: If v is unmarked in C **then**
      Unmark(a) and Unmark(b); /* *Update matrix C*
    End
  End
/* *Mesh reconnection using longest edge in those triangles with hanging nodes:*
4: Re-triangulate mesh level τ_i
End

We now illustrate in Fig. 5 the use of the coarsening algorithm applied to a terrain area in the north of Gran Canaria, Spain. In this example, the input RTIN is a mesh with 4225 vertices and 8192 triangles ($k = 6$, six mesh levels). The topographic surface extends over a square grid of 990 m with a cell step of 15 m and with a maximum height of 253 m, starting from sea level. In this example, we set the threshold error to 15 m to produce a simplified mesh of 1588 vertices and 3063 triangles. See in Fig. 5(a) a view in the 2D plane and in (b) a perspective of the coarsened terrain. The coarsest area in this example corresponds to the sea, modeled with flat triangles.

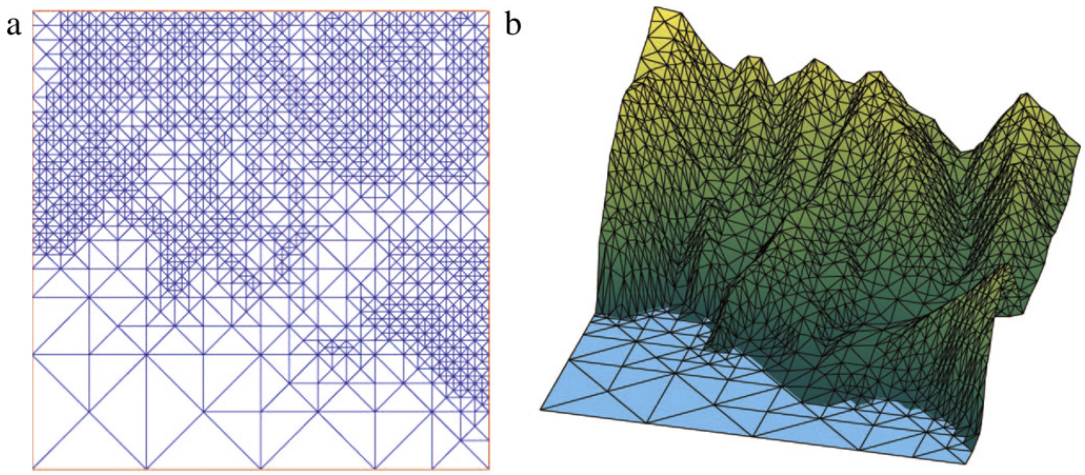


Fig. 5. Coarsening example of a terrain mesh.

Table 1
RTIN-Coarsening

Error (m)	# Vertices	RMS	Max	Mean	Hausdorff
RTIN-30	5750	0.006734	0.048659	0.003678	0.048659
RTIN-60	1975	0.011615	0.088699	0.006760	0.127386
RTIN-90	1094	0.018781	0.124696	0.011200	0.173783

Mesh statistics with Metro. Error of altitude in meters.

3. Numerical tests and results for the coarsening algorithm

The conjecture under study in this section is whether a mesh obtained with our coarsening algorithm is a better approximation than a mesh obtained with RR. To address the matching process between RTIN-Coarsening and RR we compare terrains with a reasonable close number of data points. We are interested in the accuracy of the produced meshes. The mesh model constructed by RR is of Delaunay type that follows a TIN structure.

Many approaches to evaluate simplified mesh accuracy are possible. A uniform and general tool for the evaluation of approximation precision is Metro [1,2]. Metro compares the difference between a pair of surfaces (e.g. a RTIN mesh and its simplified representation). This tool uses an approximate approach based on surface sampling and the computation of point-to-surface distances. Three sampling methods are available in the tool, Monte Carlo, Subdivision, and Similar Triangles samplings [2].

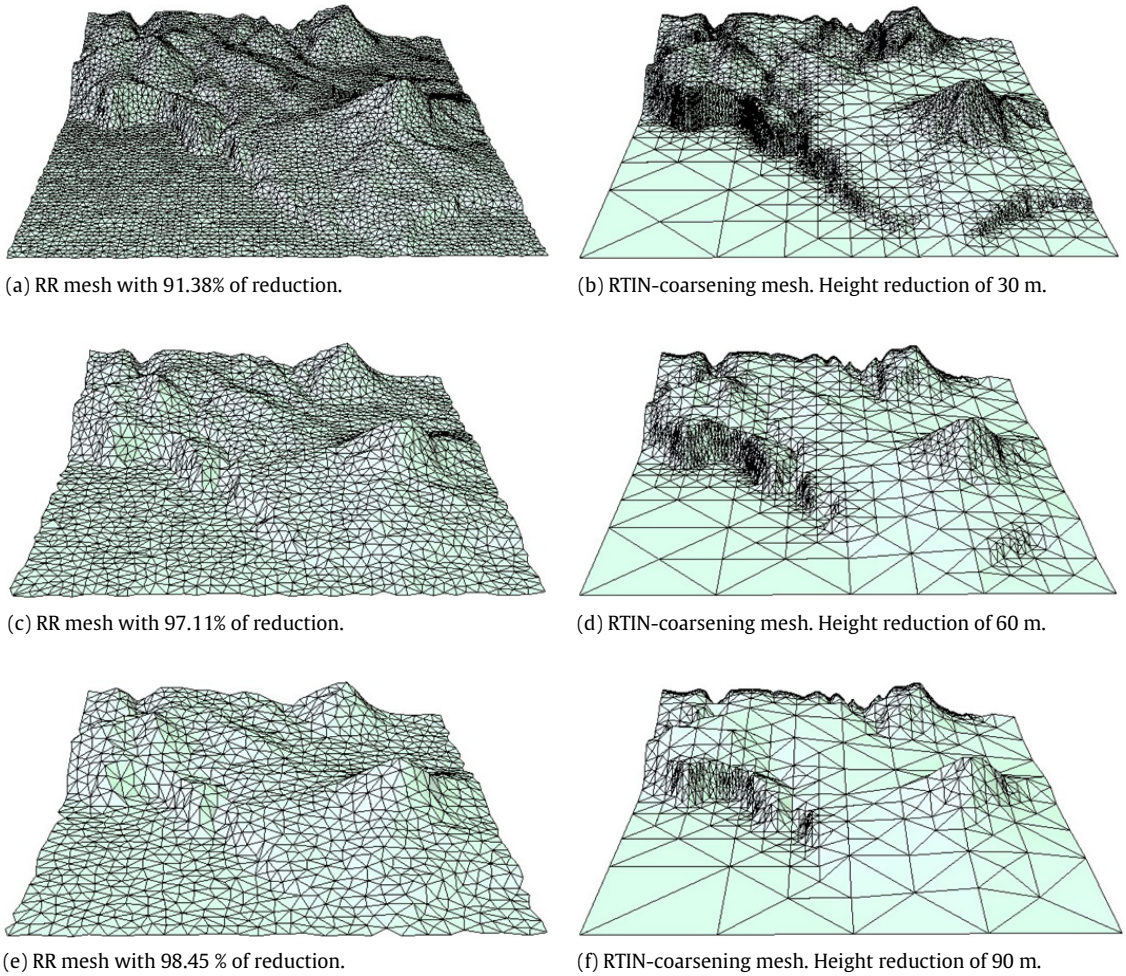
To approximate the error between two meshes we use the distance between corresponding sections of the meshes. Metro uses four types of distances as follows:

(1) Given a point p and a surface S, the distance $e(p, S) = \min_{p' \in S} d(p, p')$, where d is the Euclidean distance between two points. The one-sided distance between two surfaces S_1, S_2 is then defined as: $E(S_1, S_2) = \max_{p \in S_1} e(p, S_2)$.
(2) A two-sided distance (Hausdorff distance) is then obtained by taking the maximum of $E(S_1, S_2)$ and $E(S_2, S_1)$.
(3) The mean distance E_m between two surfaces is the surface integral of the distance divided by the area of S_1: $E_m(S_1, S_2) = \frac{1}{A_{S1}} \int_{S1} e(p, S_2) \mathrm{d}s$.
(4) The standard Root Mean Square from two meshes (RMS).

We used Metro to calculate the accuracy in the simplified meshes obtained by the RTIN-Coarsening algorithm and Rational Reducer. We test the north area of Gran Canaria Island (Spain) where the coast delimits the Atlantic Ocean and the terrain irregularity in the region is significant. The original sample terrain (131072 triangles and 66049 vertices) extends over a square grid of 4884 m with a cell step of 19 m and with a maximum height of 445 m above sea level.

RTIN-Coarsening was tested on the input mesh with the coarsening criterion defined by the altitude error threshold of 30, 60 and 90 m respectively. Mesh accuracy of the simplified terrains are reported using Subdivision sampling in Metro (Montecarlo and Similar Triangle sampling were also tested showing analogous results). In each case we tested mesh accuracy comparing the simplified meshes with the original terrain mesh. The results for the three simplified meshes are shown in Table 1 as RTIN-30, RTIN-60 and RTIN-90, where the RMS, Max, Mean and Hausdorff distances are reported.

Analogously, *RR* is used to reduce approximately the same amount of vertices as with the RTIN-Coarsening algorithm. This leads to use a reduction factor (in percentage) of 91.38%, 97.11% and 98.45% vertices respectively. The results are reported in Table 2. Fig. 6 shows the reduced meshes for approximating the initial terrain, using Rational Reducer (left part of the figure) and RTIN-Coarsening scheme (right part), respectively. In general, we obtained better accuracy results using the RTIN-Coarsening algorithm.



(a) RR mesh with 91.38% of reduction.

(b) RTIN-coarsening mesh. Height reduction of 30 m.

(c) RR mesh with 97.11% of reduction.

(d) RTIN-coarsening mesh. Height reduction of 60 m.

(e) RR mesh with 98.45 % of reduction.

(f) RTIN-coarsening mesh. Height reduction of 90 m.

Fig. 6. Gran Canaria mesh. Rational Reducer meshes (a), (c) and (e) & RTIN-coarsening meshes (b), (d) and (f).

Table 2
Rational Reducer

Error (%)	# Vertices	RMS	Max	Mean	Hausdorff
RR-91.38%	5741	0.489856	0.906838	0.421390	0.906838
RR-97.11%	1978	0.251238	0.586906	0.221660	0.906827
RR-98.45%	1079	0.250870	0.581874	0.221626	0.906798

Mesh statistics with Metro. Error equal % of triangles removed.

Table 3
Time comparison (in seconds) in constructing the simplified meshes

Terrain aprox.	RTIN-Coarsening	Rational reducer
Sample 1	5.312	1.289
Sample 2	2.232	2.150
Sample 3	1.443	4.203

Table 3 provides the *CPU* times required to compute the reduced meshes using an AMD Athlon 2.2 Ghz processor with 1 Gb RAM. It should be noted that the computational times to perform the simplification in the *RR* software is just in reverse order to the RTIN-Coarsening software. Constructing the coarsest mesh in *RR* takes more *CPU* time (4.203 s) than using RTIN-Coarsening (1.443 s). To the contrary, meshes with less reduction factor will be easier for the *RR* software (1.289 s) than for RTIN-Coarsening (5.312 s).

Some other terrain cases were tested, and similar results to the ones obtained in the previous example were observed. Both tools avoid getting visual artifacts and do not destroy any substantial information as long as the choice of the coarsening criterion remains reasonable (below 90 m). In addition, singularity zones of the terrain such as rifts and valleys are visually better preserved using RTIN-Coarsening which is a clear advantage of adaptive algorithms.

4. Conclusions

We have presented and discussed two algorithms for adaptively refine/coarse RTIN terrain meshes that recursively subdivide triangles following a 4-Triangles Longest Edge scheme. Adaptive algorithms can be used in Interactive Visualization, Level of Details or Finite Element computations where a new terrain mesh is needed in response to any event such as time, numerical solution, observer-terrain distance etc. In such cases, a further study on the error criterion that guides the adaptive process is needed.

For the refinement algorithm we provide numerical experiments showing that the propagation associated with the longest-edge based refinement in RTIN meshes (strategy extensively used also in [4,6,8,11]) only refines a few layers of elements surrounding the element to be refined. For the coarsening algorithm, we compare the algorithms against a commercial reduction software (Rational Reducer) and discuss the accuracy of the approximated meshes as well as the computational cost. Straightforward pseudocodes of the algorithms are provided in order to facilitate their implementation.

Acknowledgments

This work has been partially supported by Project UNI-2003/16 of the University of Las Palmas de Gran Canaria, by Project PI2003/35 of the Gobierno de Canarias, and by CICYT Project number MTM2005-08441-C02-02 from Ministerio de Educación y Ciencia of Spain.

References

[1] P. Cignoni, C. Montani, A. Varshney, A comparison of mesh simplification algorithms, Comput. Graph. 22 (1) (1998) 37–54.
[2] P. Cignoni, C. Rocchini, R. Scopigno, Metro: measuring error on simplified surfaces, Comput. Graph. Forum 17 (2) (1998) 167–174.
[3] E. Bellenger, P. Coorevits, Adaptive mesh refinement for the control of cost and quality in finite element analysis, Finite Elem. Anal. Des. 41 (15) (2005) 1413–1440.
[4] W. Evans, D. Kirkpatrick, G. Townsend, Right-triangulated irregular networks, in: Algorithms for Geographical Information Systems, Algorithmica 30 (2) (2001) 264–286 (special issue).
[5] M.H. Li, H.P. Cheng, G.T. Yeh, An adaptive multigrid approach for the simulation of contaminant transport in the 3d subsurface, Comput. Geosci. 31 (8) (2005) 1028–1041.
[6] P. Lindstrom, D. Koller, W. Ribarsky, L. Hodeges, N. Faust, G. Turner, Real-time, continuous level of detail rendering of height fields, in: ACM Computer Graphics, Conf. Proc. Annual Conference Series, SIGGRAPH'96, 1996.
[7] System in motion. Rational reducer. http://www.rational-reducer.com.
[8] R. Pajarola, Large scale terrain visualization using the restricted quadtree triangulation, in: Proceedings IEEE Visualization, 1998, pp. 19–26.
[9] R. Pajarola, M. Antonijuan, R. Lario, Quadtin: Quadtree based triangulated irregular networks, in: Proceedings IEEE Visualization, 2002, pp. 395–402.
[10] A. Plaza, J.P. Suárez, M.A. Padrón, S. Falcón, D. Amieiro, Mesh quality improvement and other properties in the four-triangles longest-edge partition, Comput. Aided Geome. Design 21 (4) (2004) 353–369.
[11] M.C. Rivara, G. Iribarren, The 4-triangles longest-side partition of triangles and linear refinement algorithms, Math. Comp. 65 (216) (1996) 1485–1502.
[12] J. Rossignac, P. Borrel, Multiresolution 3D approximation for rendering complex scenes, in: Geometric Modeling in Computer Graphics, Springer Verlag, 1993, pp. 455–465.
[13] J.P. Suárez, A. Plaza, Refinement and hierarchical coarsening schemes for triangulated surfaces, Journal of WSCG 11 (2003) WSCG'2003, Feb. 3-7, Plzen, Czech Republic.
[14] J.P. Suárez, A. Plaza, G.F. Carey, Graph based data structures for skeleton based refinement algorithms, Commun. Numer. Methods Eng. 17 (12) (2001) 903–910.
[15] J.P. Suárez, A. Plaza, G.F. Carey, Propagation of longest-edge mesh pattern in local adaptive refinement, Commun. Numer. Methods Eng. (2007) doi:10.1002/cnm.956.
[16] J.P. Suárez, A. Plaza, M.A. Padrón, Mesh graph structure for longest-edge refinement algorithms, 7th International Meshing Roundtable, Michigan, USA. SAND 98-2250, Sandia National Laboratories, 1998.
[17] L. Velho, D. Zorin, 4-8 subdivision, Computer-Aided Geometric Design 18 (5) (2001) 397–427. Special issue on subdivision techniques.
[18] Y. Wang, J.-P. Renaud, M.G. Anderson, C.B. Allen, A boundary and soil interface conforming unstructured local mesh refinement for geological structures, Finite Elem. Anal. Des. 40 (2004) 1429–1443.
[19] S. Zoraster, A surface modeling algorithm designed for speed and ease of use with all petroleum industry data, Comput. Geosci. 29 (2003) 1175–1182.