Refinement and hierarchical coarsening schemes for triangulated surfaces

Jose P. Suárez
Department of Cartography and Graphic Engineering
University of Las Palmas.
Las Palmas de Gran Canaria (35017), Spain
jsuarez@dcegi.ulpgc.es

Angel Plaza
Department of Mathematics.
University of Las Palmas.
Las Palmas de Gran Canaria (35017), Spain
aplaza@dma.ulpgc.es

ABSTRACT

We present a refinement and a coarsening (also simplification or decimation) algorithm for the adaptive representation of bivariate functions. The algorithms have proved to be efficient tools in numerical methods such as finite element method or image processing, [Pla00, Sua01b]. In this paper we particularize the algorithms and apply to the generation of levels of detail of terrain models. The refinement algorithm is very simple and of linear complexity in the number of vertices, and proceeds uniformly or locally in triangular meshes. The coarsening algorithm shows a complexity of *O(logn)* and obtains an adaptive hierarchical representation of the input terrain. We provide the most important features of the algorithms as well as the application to generate levels of detail of regions in the Gran Canaria island, an island where the topography is of great irregularity. Several experimental data are presented, including times of the meshes generated, rendering times, error evolution, suitability of the meshes and size of the generated meshes. The algorithms have been tested for VRML visualization showing a real time generation of levels of detail, and this fact is showed in the numerical experiments.

Keywords
Algorithms, VRML, terrain modeling, Level of Detail

1. INTRODUCTION

The approximation of a bivariate function from a set of data points is of interest in a great variety of scientific applications, as for example finite element method, computer aided design, computer graphics, terrain modeling etc. The general problem can be stated as reconstructing a model of a surface by interpolating a finite set of points in the space belonging to it. Defining a convenient relation model for encoding the neighborhood relations among the data points is necessary to build an effective surface representation. A triangle surface model (also mesh or triangulation) is often used, because of the possibility of including surface features, and of the simplicity of the topological structure.

A related problem that is also of considerable interest in the approximation of a bivariate function is refinement of a mesh. 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 inverse problem, coarsening (also simplification or decimation), reduces the number of points in the mesh and its advantage is that it may remove insignificant vertices in the approximated model. A known application of both refinement and coarsening algorithms is in Finite Element method, where for example, a quasi-stationary problem is solved, see Fig. 1, [Pla92]. Note that this example uses a combination of refinement and coarsening based on a simple error estimation for each mesh and this guarantees high precision in the areas where the problem need accuracy.

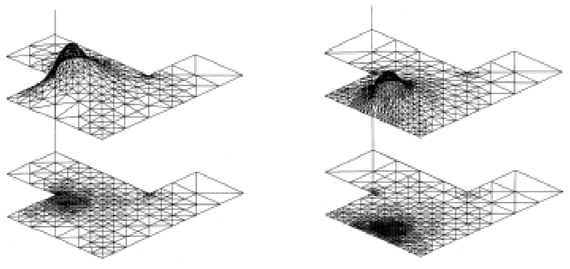
Figure 1. Meshes corresponding to times t=0 s. and t=0.00014 s. of a quasi-stationary problem

In terrain modeling, our major interest, there are mainly two methods for the arrangement of triangles in the terrain domain: a regular grid of digital

Journal of WSCG, Vol.11, No.1., ISSN 1213-6972
WSCG'2003, February 3-7, 2003, Plzen, Czech Republic.
Copyright UNION Agency – Science Press

elevation model (DEM) and a triangulated irregular networks (TIN). The simplest choice is a DEM. The values of the function (a topographic surface) represent the altitude or elevation, which are stored at regular intervals. Due to the uniformed spatial data at the mesh, that structure is not adaptive and so it can not capture the regions where irregularity is clear. Furthermore, the structure may produce a large of amount of data redundancy, especially where the topographic information is negligible. As an advantage, the data structure to store and manipulate the model is very simple and based on index operations.

Alternatively, triangulated irregular networks (TIN's) differ from the regular structure in the variability of the spatial points at the domain. The elevation points do not follow any regular pattern and so, the density may vary depending on the regions quality. The advantages of TIN's are clearly known: terrain features such as ridges or valleys may be detected and the accuracy to represent the features is greater than in a DEM, without and excessive load for the entire model. The drawback of a TIN is that it requires more storage for the same number of sampled points. The storage disadvantage becomes worse if one requires adjacency information for the triangulation.

Increasing relevance is taking an intermediate approach called Right-Triangulated Irregular (RTIN) Networks, [Eva01]. It is more regular than a TIN and more flexible than a regular grid. Like a TIN, it is a triangulated subset of the original data points, but, unlike the TIN, the subset is not arbitrary. The main benefit of the RTIN approx-imation is not in providing a single approximation to a surface, but rather in providing a framework of many approximations at varying level of detail. Moreover, the complexity time and storage is very competitive with the other approaches, considering that RTIN is a hierarchy based approximation.

An important issue in the coarsening of a bivariate function is the ability to bound the error. Given a triangulation M of a domain D and a function $f(x)$ defined over the triangulation, the coarse mesh can be called M' and the resulting function $f'(x)$. The measured error in a coarse mesh M' is commonly represented as: $E(M')=max\ \{|f(x)-f'(x)|,\ x \in D\}$.

1.1 Related work

A complete revision and comparison of mesh simplification algorithms can be found in [Cig98].

A number of coarsening techniques have been developed in order to reduce the number of triangles to a particular number or until a given error threshold is met: (i) *Vertex insertion/deletion* techniques are extensively used. These classes of simplification algorithms are based on successive applications of topological mesh operators, such as edge contraction or vertex deletion. The main application is in geographical information systems (GIS). (ii) *Region merging*. It basically consists on grouping faces which are nearly co-planar. Then, the interior points to the groups are removed and the resulted local geometry is re-triangulated. (iii) *Re-tiling*. New vertices are inserted at random on the original surface mesh, and then moved on the surface to be displaced over maximal curvature locations: the original vertices are then iteratively removed and a re-tiled mesh, built on the new vertices, is given in output, [Tur92]. (iv) *Energy Function optimization*. The mesh optimization approach defines an energy function which measures the 'quality' of each reduced mesh. Legal moves selection is driven by an optimization process of the energy function, [Hop93]. (v) *Vertex clustering*: Based on geometric proximity, this approach groups vertices into clusters, and for each cluster it computes a new representative vertex, [Ros93]. (vi) *Wavelet-based approaches*. It is based on the wavelet decomposition approach. It seems very promising for surface simplification. (vii) *Intermediate hierarchical representation.* An octree structure is adopted to automatically produce simplified representations, [And96].

A considerable interest is getting the coarsening algorithms using a parallel environment. This is especially recommended when large and complex datasets have to be managed, see for example [Fra00].

In this paper we present a vertex insertion/deletion simplification scheme which produces hierarchical Right-Triangulated Irregular Networks. The hierarchical approximation inherent to this scheme allows a natural succession of ordered level of detail meshes. To illustrate the proposed RTIN based coarsening scheme in terrain modeling we perform several level of details of a given terrain and include them into a practical VRML world where interactive visualization is tested.

2. THE REFINEMENT AND COARSENING PROBLEM

The manipulation of meshes in real time applications requires particular strategies to adaptively refine or simplify the domain. There are, essentially, two approaches to constructing an approximation from a hierarchy: bottom-up and top-down. This has lead to the improvement of algorithms to proceed with those operations. The (local) 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 (uniform) refinement problem differs from the local one in that the insertion and location of vertices in the refinement is the same over each triangle in the mesh. These are essentially bottom-up. The inverse problem, coarsening (also simplification or decimation), reduces the number of points in the mesh and its advantage is that it may remove insignificant vertices in the approximated model. Considerations about local or uniform coarsening are like in the refinement problem. Similarly, coarsening admits two different approaches, top-down and bottom-up. Top-down works well in cases where the approximation is close to the finest approximation. That process progressively unrefines a mesh until a desired approximation is achieved. Meanwhile, a bottom-up approach starts from the coarsest mesh and progressively refine local areas in the mesh. For that reason the bottom-up coarsening scheme can also be viewed as a bottom-up refinement scheme. The bottom-up coarsening is suitable for the case in which the expected approximation is much more coarse than detailed. It can be noted however that there is no difference between the two previous coarsening approaches if a complete approximation hierarchy is needed, going from the coarsest level to the finest one. This case is the situation in our work.

The local refinement of triangular meshes involves two main tasks. The first is the partition of the target triangles and the second is the propagation or extension to preserve the 'cracks' in the resulted mesh. However, the uniform refinement does not consider the propagation as the refinement is uniform in the mesh.

Several approaches for partitioning triangles have been studied. The simplest is *Bisection* into two subtriangles by connecting the midpoint of one of the edges to the opposite vertex, Fig. 2 (b). If the longest edge is chosen for the bisection, then this is called *Longest Edge Bisection*.

The *Four Triangles Longest Edge* partition (*4T-LE*) [Riv89], 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. 2 (d).

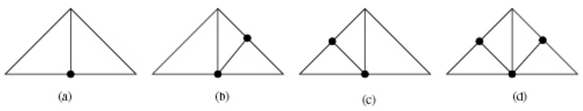

Figure 2. Subdivision patterns in 4T-LE

The second task in the local refinement problem is to ensure the conformity of the mesh. It is necessary to determine additional irregular patterns, Fig. 2 (a-c) which makes it possible to extend the refinement to neighbor triangles. One can connect to the midpoint of the longest edge and complete the subdivision by connecting this new midpoint to the opposite vertex (Fig. 3).

2.1 Consideration about the approximated meshes

Our mesh representation contains subdivision patterns corresponding to the 4T-LE partition. Starting from a coarse two dimensional square domain, the most detailed uniform mesh is structured as a grid of size $(2^k+1)\times(2^k+1)$ for some integer k, the level of the refinement. The benefit of the manner in which the 4T-LE refinement proceeds is that it produces only right-angled, isosceles triangles, as in RTIN [Eva01]. Alternatively other irregular meshes, for example see Fig. 3, are also possible and the partition guarantees a finite number of similarity classes when successively refinement is applied. A bound and exact number of similarity classes triangles in the refinement of meshes is also of particular interest in Finite Element computations, [Pla92, Riv89], or in the geometrical aspects of triangulations. A recent work by the authors solves that problem and will be next published.

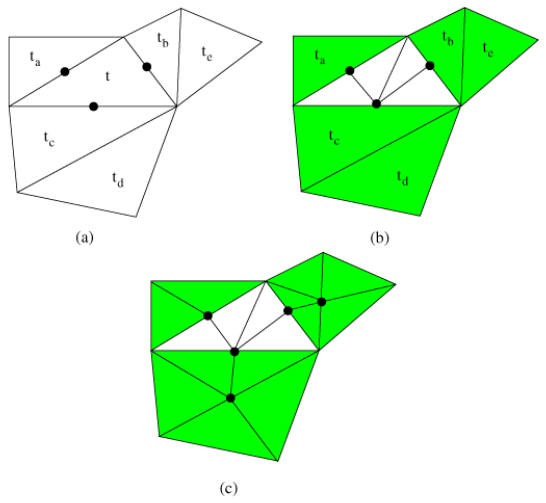

Figure 3. Edge bisection for refining triangle t (b) 4T-LE refinement of t and propagation refinement area, (c) refinement of triangles

As desired geometric features of the meshes produced by both, refinement and coarsening techniques, we have: (i) *non-degeneracy* of the elements, (ii) *conformity* and (iii) *smoothness*. Mesh degeneracy corresponds to the occurrence of thin triangles with large aspect ratio and can lead to undesirable behavior affecting numerical stability and producing visual artifacts, [Ber90]. Mesh conformity refers to the requirement that the intersection of non-disjoint triangles is either a common vertex or a common side. A vertex in the interior of an edge is *non-conforming* and it is sometimes called a 'crack'. [Paj98] presents a model that avoids cracks with a restricted selection of points according to a dependency graph defined between vertices. Our scheme to avoid cracks is similar to that but we use a longest edge based inclusions of points that yields the same result. A different approach is used in [Aba00] that minimizes extra triangles: they use ‘fictitious points’ instead of complete dependency paths and this reduces extra triangles in the conformity process.

Mesh smoothness implies that the transition between small and large elements should be gradual. Smoothness is directly connected with the propagation problem in the local refinement of meshes. A recent study by the authors finds an asymptotic behavior of the propagation when longest edge based refinement is applied to triangular meshes, see [Sua02], which is summarized as follows: since the conformity process extends at most by the three edges of a triangle, the propagation defines at most three lists of ordered triangles. If we call *M2(t)* the maximum number of triangles of the three resulting lists, the main result concerns that when uniform iteratively 4T-LE refinement is performed to a mesh, *M2(t)* tends to 2 when the refinement level *n* goes to infinity. Therefore, this result also applies for our refinement algorithms here.

3. RTIN-REFINEMENT AND RTIN-COARSENING ALGORITHMS

In order to describe our refinement (RTIN-refinement) algorithm we assign a local numeration to each triangle in a mesh. The numeration is counter-clockwise from 1 to 3, being the vertex of right angle numerated as 3, see Fig. 4. Note that from the implementation point of view, it is sufficient for a triangle to store as the first vertex the right angle vertex. For the new generated triangles the right angle vertex is easily derived from the partition applied.

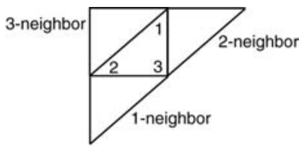

Figure 4. i-neighbors for a single triangle

Definition 1. (i-neighbor) The *i-neighbor* (i=1,2,3) of a triangle is the neighbor that does not share the triangle vertex *i.*

Of particular interest in the longest edge based local refinement is the 3-neighbor, which can be viewed as the longest edge neighbor of a triangle *t*. That adjacency relationship is crucial for the local refinement since the refinement propagates on that triangle over the mesh.

RTIN-refinement algorithm proceeds uniformly or locally in a mesh. We will consider the following statement of the problem: given a RTIN mesh τ and a set of triangles $t_0 \in \tau$ to be uniformly refined by the 4T-LE partition, perform the refinement process, which concerns the uniform refinement of τ_0 plus the propagation refinement in $\tau - \tau_0$, see Fig. 5.

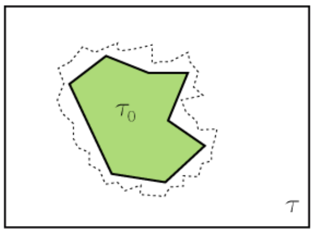

Figure 5. Uniform refinement on τ_0 and propagation (area between dashed line and solid line) of refinement

The algorithm does not take into account the computed error, instead, it is provided the set of triangles τ_0 not satisfying the error criterion and that set is the input triangles to be refined. The algorithm for the refinement process is as follows:

```
Algorithm RTIN-Refinement(τ,τ0)
/* Input: τ mesh, τ0 set of triangles to refine
/* Output: τ new mesh
1) Perform uniform 4T-LE refinement of τ0
2) S=list of non-conforming triangles in τ-τ0
3) For each triangle Si ∈ S do
        t=Si
        4) While t is not subdivided do
             Subdivide t
             t=3-neighbor(t)
           end
   end
End.
```

The procedure that computes the error in a mesh operates on the triangles as follows: if for a given mesh *j*, see Fig. 6, we have that $|h(a)-h(b)|<\varepsilon$, being edge *ab* the longest edge of the triangle *abc*, *h* the height at a point and ε the fixed error, then the supported triangle *abc* is not refined; otherwise, the 4T-LE partition is applied and four new triangles are generated at mesh level *j+1.* The error checking is done one time for every refinement level.

The coarsening algorithm we provide here is similar to that in [Pla92, Pla00] where the application area concentrated on the Finite Element Method. Here we concern the application for RTIN meshes in terrain approximation and as different features we remark: the data structures are simplified (see Section 3.1), the level of details benefit from the hierarchical meshes and the algorithm is proved in a real time VRML world.

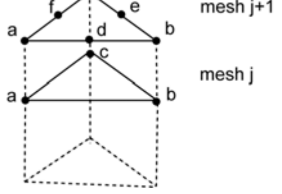

Figure 6. Level *j* and *j+1* in the refinement for a given triangle ε

The input of the algorithms is a $(2^k+1)\times(2^k+1)$ regular spaced grid of heights. That represents k levels in the hierarchy corresponding to k different uniformly refined meshes $\{\tau_1, \tau_2, \tau_3, \ldots, \tau_k\}$.

During the inspection within a level, a vertex is selected if its error exceeds the tolerance threshold ε, or if the point is marked from a dependency relation. Additionally, the two related vertices are marked accordingly.

After that process, the algorithm progressively reconstructs the k approximations starting from the coarsest level to the finest one. At the end, the result is a sequence of approximations $\{\tau'_1, \tau'_2, \tau'_3, \ldots, \tau'_k\}$.

```
Algorithm RTIN-Coarsening(T, ε)
/* Input: T={$\tau_1$, $\tau_2$, $\tau_3$,…, $\tau_k$}, ε=error
/* Output: T'={$\tau'_1$, $\tau'_2$, $\tau'_3$,…, $\tau'_k$}
1) For i=k to 2 do
      For each vertex $v_j \in \tau_i$
            hull($v_j$,a,b)
            1.1) if abs(h(a)-h(b))<e then mark($v_j$)
            1.2) Check local-conformity on unmarked nodes
            1.3) if a or b is marked then
            unmarK(a);unmark(b)
      end
   end
2) For j=2 to k do
            Redefine mesh $\tau_j$ with the marked vertices
   end
End.
```

The function *hull(v_j,a,b)* provides for a given vertex $v_j \in \tau_i$, $i>1$, the surrounding vertices a and b belonging to τ_{i-1}, i>1 such that v_j is the midpoint of the edge *ab*. Note that *mark* and *unmark* are simple functions that assign or remove a bit flag that temporally represents the status of each vertex: *mark* means remove and *unmark* means remain.

Meanwhile the refinement algorithm operates on the triangles of a given mesh τ, the coarsening algorithm operates on the vertices for each intermediate mesh τ_i of an input sequence T. The algorithms rely on the particular error indicator used to guide the process. We choose a point based error as follows: if for a given mesh $j+1$, a triangle as in Fig. 6 we have that $|h(a)-h(b)|<\varepsilon$, then the supported triangle *abc* is coarse, marking the mid vertex d to be removed. At this point, it is checked the next two conditions in order to preserve conformity and hierarchical representation respectively: (1) does the removal of d produce a non conformity mesh? if affirmative, then vertex b must remain, and (2) have the surrounding vertices a and b been marked for being removed previously? In affirmative case, it is required to unmark both vertices a and b.

3.1 Data structures

Many advances on data structures for triangular meshes have been made in last years. We presented in [Sua01a, Sua01b] an efficient graph based data structure (processing and storage) that fits the need of the refinement and coarsening algorithms for non-hierarchical meshes. For hierarchical meshes it is usually used a tree based data structure, with many variants. A brief review of quadtree based data structures can be found in [Bal00], where it is also proposed a new quadtree data structure called *semi-linear quadtree* with efficient navigation and compact storage. Contributions by Evans *et al.* [Eva01], De Floriani *et al.* [DeF84], Lindstrom *et al.* [Lin96], Pajarola [Paj98] and [Aba00] also gives tree based data structures for coarsening algorithms.

Note that a quadtree data structure on a $(2^k+1)\times(2^k+1)$ mesh can be implemented using index operations instead of using pointers and nodes, which is often called an *implicit quadtree* defined on a mesh. That is the approach used in our algorithms. A stored mesh consists of a list of vertices, a list of triangles given by the three vertices and a extra bit per vertex used to mark or unmark vertices for the RTIN-Coarsening algorithm. Note also that no special neighbor-finding technique is needed for our algorithms. The only adjacency used by the algorithms is the 3-neighbor, which is explicitly obtained if the first vertex stored for each triangle is just the right angle vertex.

3.2 Approximation quality

In this section we describe two methods to measure the quality of the approximated meshes when RTIN algorithms are used in terrains. The first method is based on contour lines for topographic surfaces and the second one uses a metric called Signal Noise Relation.

A topographic surface or terrain is the graph of a bivariate function *f(x,y)* over a connected and compact domain *Ω*. For example,

$$\sigma = \{(x,y), f(x,y) = (x,y) \in \Omega\}$$

Equation 1

is the graph of a function f representing a terrain. Given a real number q, the contour lines set $C_{\sigma,q}$ for a terrain σ is defined as follows:

$$C_{\sigma,q} = \{(x,y) \in \Omega, f(x,y) = q\}$$

Equation 2

We can compare the contour lines sets of the sequence of terrain meshes resulted in the RTIN-coarsening process, and then we have an useful measure of the relative difference between the terrains. More exactly, it should be compared the contour lines of the original terrain (the $(2^k+1)\times(2^k+1)$ source vertices) with a coarse mesh. We compute the area closed by each two contour lines $C_{\sigma i,q}$, $C_{\sigma j,q}$ for two different terrains σ_i, σ_j. Therefore, the area computed represents the deviation of the respective meshes in terms of the heights at the locality expressed by the contour lines. When this value is big, a clear poor approximation is derived. Obviously, the given area under two contour lines of different approximations is related to the error threshold used to coarse the meshes, and it will be bigger as bigger is the error threshold chosen for the approximation.

The second method relies on a different way to measure the quality of the approximation of two meshes We use the *Signal Noise Ratio* (*SNR*), [Fis94]. SNR compares the original signal data and the noise in the approximated signal. In other words, SNR provides the relation between the sources heights of the terrain *s(n)***,** and the error induced when vertices are removed *e(n)*, see Equation 3. The range of SNR goes from zero (maximum error) to infinity (no error).

$$SNR = 10\log\frac{\sum_{n=1}^{N} s^2(n)}{\sum_{n=1}^{N} e^2(n)}$$

Equation 3

In order to give some experimental data of the behaviour of both methods, we apply the RTIN-coarsening algorithm to a region of the Gran Canaria island, Galdar city, where the coast delimits the Atlantic ocean and the terrain irregularity in the city is notorious, see Fig. 7.

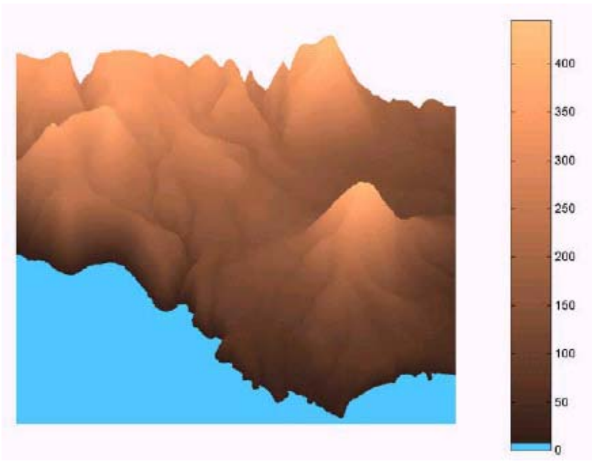

Figure 7. Coast of Galdar, Gran Canaria island. Rendered image with 131072 triangles and 66049 vertices

The computations were made at an *AMD* 1.5 Ghz-256Mb RAM computer. The algorithms have been implemented in *Matlab* 6.5®. The same computation issues apply for Section 4. The error threshold goes from 0 to 100 meters, and a fixed value is chosen each 5 meters. The evolution of error compared to the vertices in the resulted meshes is presented in Fig. 8. Fig. 9 shows the evolution of SNR in relation to the vertices. According to the contour lines method, Fig. 10 shows the overlapped contour lines of the original terrain (131072 triangles and 66049 vertices) and an approximation taking 30 meters as threshold error (11300 triangles and 5750 vertices). It is worth to note the deviation of each contour lines in the representation.

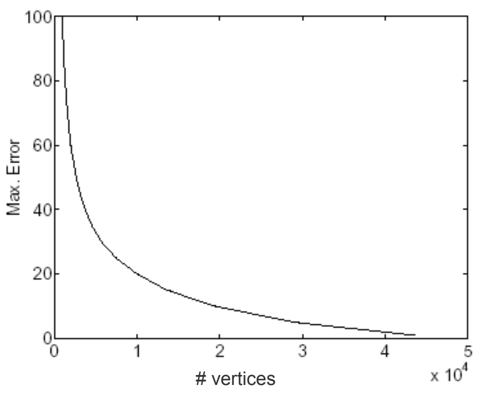

Figure 8. Error in meter versus the number of vertices in the coarse meshes

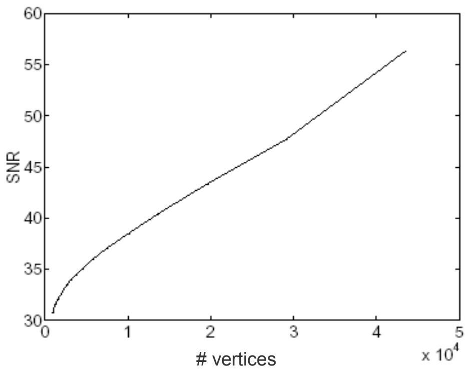

Figure 9. SNR in Db. versus the number of vertices in the coarse meshes

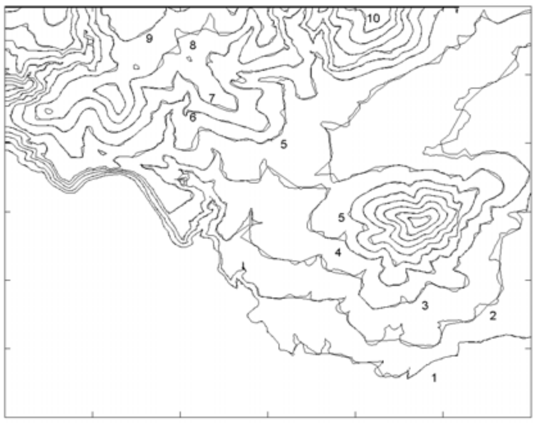

Figure 10. Overlapped contour lines: Error=0 m. (131072 triangles and 66049 vertices) & Error=30 m. (11300 triangles and 5750 vertices)

4. APPLICATION: LEVEL OF DETAILS IN VRML WORLDS

For the purpose of illustrating the coarsening algorithm in a practical real time application, we provide the generation of *Level Of Details* (*LOD*) for *VRML* worlds. Varying detail with distance reduces upfront download time, and increases drawing speed. A LOD function is known as a function *L(d)* defined over a real discretized space $d \in [R_0, R_n]$ and $\{L_0, L_1, L_2, \dots, L_{i+1}, \dots, L_{n-1}\}$ is the ordered set of meshes which are the LOD representations:

$$L(d) = \begin{cases} L_0 & \text{if } d < R_0 \\ L_{i+1} & \text{if } R_i \le d < R_{i+1} \quad \text{for } -1 < i < n-1 \\ L_{n-1} & \text{if } d \ge R_{n-1} \end{cases}$$

Equation 4

VRML manages levels of detail by defining a specific node:

LOD { center 0.0 0.0 0.0
range [. . .]
level [. . .]}

where *center* is the *x,y,z* coordinate of the shape center, *range* is a list of level switch ranges and *level* a list of shape levels. We present in Table I experimental data of 4 LOD's each of one of 0, 20, 40 and 60 meters of error.

Figures 11 and 12 show LOD's 2 and 4 for approximations of 20 and 60 meters respectively. It can be noted the considerable reduction of triangles and vertices which also produces reduction in the rendering time and the mesh size. The resulted LOD's have been included in the VRML syntax and tested in a VRML browser (*Cosmoplayer*®) showing an interactive real-time navigation.

LOD	1	2	3	4
Error (m.)	0	20	40	60
Triangles	131072	10096	3755	1975
Vertices	66049	19962	7333	3789
Rend. Time (sg.)	1.310	0.220	0.110	0.060
Mesh Size (kb.)	3653	406	156	86

Table I. Generation of 4 LOD's. Galdar terrain

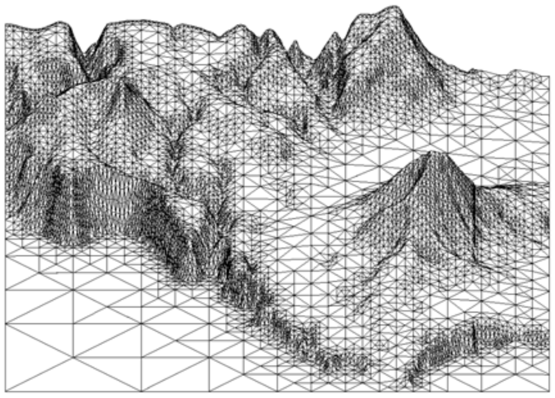

Figure 11. Approximation with error of 20 m., 19962 vertices, 10096 triangles

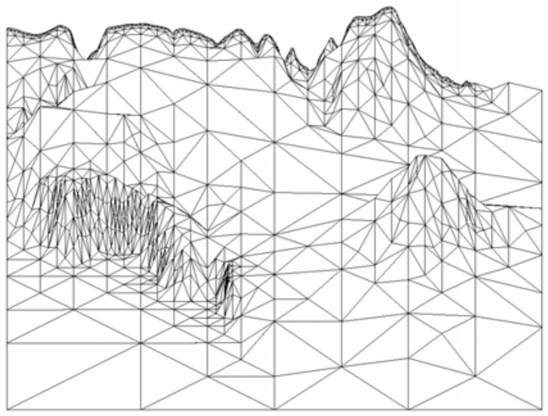

Figure 12. Approximation with error of 60 m., 3789 vertices, 1975 triangles

5. CONCLUSIONS

Simple and efficient hierarchical algorithms for refinement and coarsening triangular meshes with complexity of *O(n)* and *O(logn)* respectively have been presented here. Although the timing of our algorithms is similar to those by [Eva01], De Floriani *et al.* [DeF84], Lindstrom *et al.* [Lin96] and Abásolo *et al.* [Aba00], we emphasize the following points in this paper: the types of mesh subdivisions produces right isosceles similar triangles, and the algorithms admit both very simple description and implementation. As the data structures are also very simple, the estimated time for preparing the input models for the algorithms is short.

Moreover, we have provided two methods to measure the quality of the approximated meshes and finally give experiments including the resulted approximations in a practical and interactive VRML world. A future work may consider the implementation of the algorithms in parallel environment for dealing with large and complex datasets.

6. ACKNOWLEDGMENTS

This research work has been supported by Project UNI2002/22 (Universidad de Las Palmas de Gran Canaria).

7. REFERENCES

[Aba00] Abásolo, M.J., Blat, J., and De Giusti, A. A Hierarchical Triangulation for Multiresolution Terrain Models. The Journal of Computer Science and Technology (JCS&T), 1, 3, 2000.

[And96] Andujar, C., Ayala, D., Brunet, P., Joan – Arinyo, R. and Sole, J. Automatic generation of multiresolution boundary representations. Computer Graphics Forum (Eurographics'96 conf. proc.), 15,3, pp.87-96, 1996.

[Bal00] Balmelli, L. Rate-distorsion optimal mesh simplification for communications. Ph.D. Thesis 2260, Ecole Polytechnique Federale de Laussane (EPFL), Switzerland, 2000.

[Baj98] Bajaj, C.L. and Schikore, D.R. Topology preserving data simplification with error bounds. Comp. & Graph., 22, 1, pp.3-12, 1988.

[Ber90] Bern, M., Eppstein, D. and Gilbert, J. Provably good mesh generation. Proc. 31st IEEE Symposium on Fundations of Computer Science, pp.231-241, 1990.

[Cig98] Cignonni, P, Montani, C. and Varshney, A. A comparison of mesh simplification algorithms. Comp. & Graph., 22, 1, pp. 37-54, 1998.

[DeF84] De Floriani, L., Falcidieno, B., Nagy, G. and Pienovi, C. Hierarchical structure for surface approximation. Comp. & Graph., 8, 2, pp.183-193, 1984.

[DeF00] De Floriani, L., Magillo, P. and Puppo, E. Applications of computational geometry to geographic information systems, in Handbook of Computational Geometry. Sack, J.-R. and Urrutia, J. (eds.), Elsevier, pp.333-388, 2000.

[Eva01] Evans, W., Kirkpatrick, D. and Townsend, G. Right-Triangulated Irregular Networks. Algorithmica: Special Issue on Algorithms for Geographical Information Systems, 30, 2, pp.264-286, 2001.

[Fis94] Fisher, Y., Fractal Image Compression. Theory and Applications, Springer Verlag, 1994.

[Fra00] Franc, M. and Skala, V. Triangular mesh decimation in parallel environment. EUROGRAPHICS Workshop on Computer Graphics and Visualization, Girona, Spain, pp.39-52, 2000.

[Hop93] Hoppe, H., De Rose, T., Duchamp, T. McDonald, J. and Stuetzle, W. Mesh optimization. In ACM Computer Graphics conf. proc. Annual Conference Series (SIGGRAPH'93), pp. 19-26, 1993.

[Lin96] Lindstrom, P., Koller, D., Ribarsky, W., Hodeges, L., Faust, N. and Turner, G. Real-Time, continuous level of detail rendering of height fields, in conf. proc. of SIGGRAPH'96, 1996.

[Paj98] Pajarola, R. Large scale Terrain Visualization using the Restricted Quadtree Triangulation, in conf. proc. IEEE Visualization '98, pp. 19-26 & 515, 1998.

[Ped00] Pedrini, H. An improved refinement and decimation method for adaptive terrain surface approximation, in conf. proc. of the 8th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, 2000.

[Pla00] Plaza, A. and Carey, G.F. Local refinement of simplicial grids based on the skeleton. Appl. Num. Math., 32, 2, pp.195-218, 2000.

[Pla92] Plaza, A, Ferragut L. and Montenegro, R. Derefinement algorithms of nested meshes, in Algorithms, Software, Architecture, J. van Leeuwen (eds.), Elsevier Science Publishers B.V. (North-Holland), pp. 409-415, 1992.

[Rib00] Ribelles, J., López, A., Belmonte, O., Remolar, I., Chover, M. Variable resolution level-of-detail of multiresolution ordered meshes, in conf. proc. of the 8th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, 2000.

[Riv89] Rivara, M.C. Selective refinement/derefinement algorithms for sequences nested triangulations. Int. J. Num. Meth. Eng., 28, pp. 2889-2906, 1989.

[Ros93] Rossignac, J. and Borrel, P. Multiresolution 3D approximation for rendering complex scenes. In Geometric Modeling in Computer Graphics, Falcidieno B. and Kunii T.L. (eds.). Springer Verlag, pp.455-465, 1993.

[Sua01a] Suárez, J.P., Carey, G.F. and Plaza, A. Graph based data structures for skeleton based refinement algorithms. Comm. Num. Meth. Eng., 17, pp.903-910, 2001.

[Sua01b] Suárez, J.P. Estructuras de datos en los algoritmos de refinamiento y desrefinamiento basados en la bisección por el lado mayor. Aplicaciones. Ph.D. Thesis (in Spanish), University of Las Palmas de Gran Canaria, Spain, 2001.

[Sua02] Suárez, J.P. and Plaza, A., The propagation problem in longest-edge based refinement algorithms, IUMA Institute Report-1/02, Universidad de Las Palmas de Gran Canaria, 2002.

[Tur92] Turk, G. Re-tiling polygonal surfaces, in ACM Computer Graphics (SIGGRAPH'92 conf. proc.), 26, E. E. Catmull (ed.), pp. 55-64, 1992.