A Hierarchical Triangulation for Multiresolution Terrain Models

M. J. Abásolo[1], A. De Giusti[2], J. Blat[3]

Abstract

Interactive visualisation of triangulated terrain surfaces is still a problem for virtual reality systems. A polygonal model of very large terrain data requires a large number of triangles. The main problems are the representation rendering efficiency and the transmission over networks. The major challenge is to simplify a model while preserving its appearance. A multiresolution model represents different levels of detail of an object. We can choose the preferable level of detail according to the position of the observer to improve rendering and we can make a progressive transmission of the different levels. We propose a multiresolution triangulation scheme that eliminates the restrictions of the restricted quadtree triangulation and obtains better results.

Keywords: *multiresolution model, terrain model, level of detail, quadtree subdivision, hierarchical triangulation*

1. INTRODUCTION

3D terrain models are a useful tool for interpreting terrain data such as in terrain analysis, territorial planning, and geographical information systems. Besides they are used in simulation systems and virtual reality [Casillas99], [Hernández99] where real time is needed and the size of the model has to be seriously considered.

Triangles are the most popular drawing primitive in computer graphics. They can represent any model approximately and specialised graphics hardware can render them very quickly. Unfortunately, accurately representing a three-dimensional model often requires a large number of polygons. For every computer graphic hardware there exists a model complex enough to get an unacceptable performance. Polygonal simplification is the act of transforming a three-dimensional polygonal model into a simpler version [Erikson96]. It reduces the number of polygons needed to represent a model while trying to retain a good approximation to the original shape and appearance. Polygonal simplification not only provides the benefit of increasing rendering performance, but it also reduces the amount of storage and helps quicken the transmission over networks.

Multiresolution models provide different level-of-detail (LOD) representations of the modelled object. The appropriate resolution can be used to display the model depending on viewing parameters, like screen size of the object, distance from the viewpoint and view direction. The appropriate model is the coarsest level that looks the same as the finest level progressive mesh.

In section 2 we present a multiresolution triangulation model for terrain databases, that:

- Is adaptive to the terrain structure, so regions with high frequency elevation changes are modelled with more triangles per area unit than low frequency surface regions;

- Provides means to extract surface representation at variable precision. This fact enables visualisation using multiples LODs, or for progressive transmission.

[1] *Conicet* fellowship, UNICEN. e-mail: abasolo@exa.unicen.edu.ar
[2] LIDI, Universidad Nacional de La Plata. 50 y 115 (1900) La Plata, Argentina.
e-mail:degiusti@lidi.info.unlp.edu.ar
[3] Universidad de las Islas Baleares. Cra. de Valldemossa km7.5 (Palma, España)
e-mail: josep.blat@iua.upf.es

The triangulation model tries to use as many triangles as absolutely necessary, that's why it eliminates the restrictions of the restricted quadtree triangulation that causes unneeded triangles used to prevent cracks [Pajarola98].

In section 3 we describe the steps following in a multiresolution model: level of detail selection, points selection and model updating. In section 4 we describe an efficient implementation with the *triangle strip* structure. Finally, in section 5 we presents the results comparing them with a similar scheme called restricted quadtree triangulation.

2. UNRESTRICTED HIERARCHICAL TRIANGULATION

2.1. Hierarchy and Triangulation

A terrain model is generally a height field, that is a matrix of points that are distributed regularly on a two-dimensional grid. The triangulation model presented here is an adaptive hierarchical triangulation for height field.

Other regular meshes triangulation schemes are based on restricted quadtree triangulation (Figure 1). The basic quadrant consists of four points and two triangles, as shown in Figure 1-a, or its 90º rotated equivalent. The first step of the refinement is achieved by adding the mid-point of the quadtree block. This step splits the initial two triangles into four as shown in Figure 1 b). Adding the mid points of the quadrant's boundary edges and splitting the triangles accordingly, as shown in Figure 1 c) completes the second step of the refinement. Four identical quadtree blocks are obtained, and every one can be recursively subdivided as the initial quadtree block.

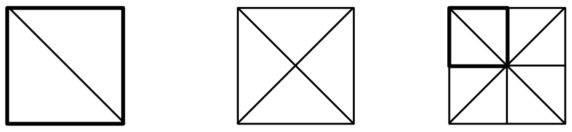


FIGURE 1. Quadtree subdivision

Our triangle hierarchy has only one stage of recursive subdivision of a triangle block (Figure 2). The basic pattern consists of a triangle, as shown in Figure 2-a, that is recursively split by adding the mid-point of its diagonal. Two adjacent triangular patterns form the initial quadtree block of the restricted quadtree as is shown in Figure 1-a. Figure 2-c shows how we can recursively subdivide each triangular pattern obtaining the same results as with quadtree subdivision in Figure 1-c.

The "bitriangle" or "bitree" subdivision differs from quadtree subdivision in the hierarchy of triangles that can be divided independently from the adjacent ones. Figure 3 shows some possible results of subdivision with this scheme that are not possible with the restricted quadtree scheme.

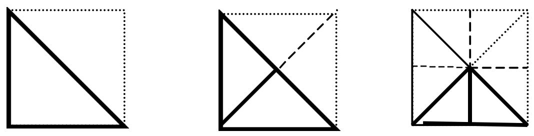


FIGURE 2. Bitree subdivision

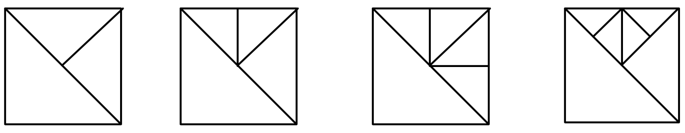


FIGURE 3. Bitree subdivision variations

Figure 4 shows the assignment of points on the grid levels in the triangle hierarchy. L_0 is the top level in the hierarchy and it denotes the two-root of the triangle bitree initially formed by two triangles of level L_0. A triangle of level L_i is subdivided into two triangles of Level L_{i+1} by the diagonal mid-point that is assigned to level L_{i+1}.

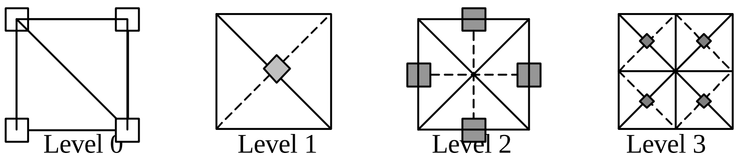


FIGURE 4. Hierarchy Levels

Because the triangulation is hierarchically defined, every triangle is recursively subdivided. The triangulation is computationally very efficient because it is given implicitly. There is no need of geometric computations like in-circle tests.

2.2. Avoiding *cracks*

The presence of cracks is undesirable in any triangulation scheme. When rendered from certain viewing angles, even small gaps become very noticeable. A solution may be to fill in a gap with a polygon as in [DeHaemer91], but it produces extra polygons while trying to minimise the number of them. Besides it breaks the hierarchy of triangles in our scheme. Figure 5 shows how a nonrestricted quadtree triangulation may produce cracks. [Herzen87] presents a restricted quadtree triangulation model with the requirement that adjacent quadrants, or quadtree blocks, must differ by at most one level in the quadtree hierarchy. In addition, a triangulation rule says that every quadtree block is triangulated by two triangles per boundary edge unless the edge borders a larger block.

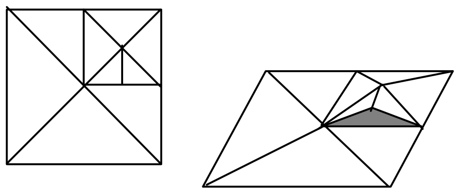


FIGURE 5 Presence of cracks with nonrestricted quadtree triangulation

[Pajarola98] presents a model based on the restricted quadtree triangulation that avoids *cracks* with a restricted selection of points according to a dependency graph defined between the vertices. The centre-vertex of a block depends on two diagonally opposite vertices of the block (Figure 6-a). The non-centre-vertices depend on two center-vertices of the adjacent blocks vertically or horizontally aligned (Figure 6-b). At the top level, the four corner vertices of the quadtree block are mutually dependent from each other.

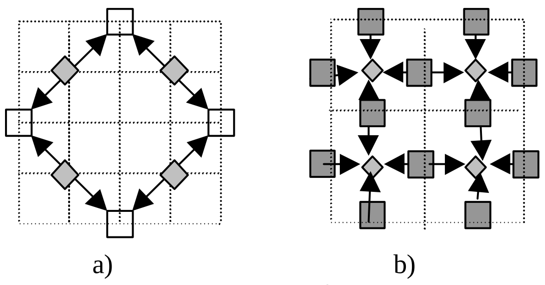


FIGURE 6. Vertex dependencies of restricted quadtree

The rule of the restricted quadtree subdivision is such that if a vertex is selected for triangulation the related dependencies must be selected too. At the same time, this related vertices have other dependencies that must be selected too, and so on. That rule finally restrict the quadtree in a way that adjacent blocks must differ by at most one level in the quadtree hierarchy.

The presence of *cracks* (Figure 5) can be avoided by adding points following the dependencies. This warranties a matching triangulation but produces extra triangles as it is shown in Figure 7 in dotted line.

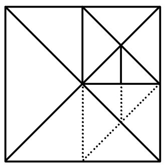


FIGURE 7. Avoiding *cracks* with restricted quadtree

In the hierarchical triangulation presented in this paper, every reoccurrence of the triangle pattern at a smaller scale can be replaced by two triangles. In the examples shown in Figure 3 we can see that there is a risk of cracks. To avoid cracks appearances we add points called "fictitious" that have interpolated height. This denotes points, which are not selected to satisfy the approximation tolerance (section 2.3), of the corresponding terrain patch but required to build a triangulation without cracks between adjacent triangles of different levels.

We redefine vertex dependencies reducing the quadtree dependency chain, and as a consequence less points and triangles are added to avoid *cracks*.

The points of the top level L_0, can't be fictitious, and are mutually dependents from each other. For every point p_i of level L_i ($i>0$), mid-point of the diagonal of two adjacent triangles T^1_{i-1} and T^2_{i-1} of level L_{i-1}, (if the diagonal belongs to a border edge of the initial square block it only belongs to one triangle), we redefine the dependencies used in restricted quadtree triangulation (Figure 6) as follows:

- If p_i is selected (non-fictitious), it depends on the points p^1_{i-1} and p^2_{i-1} of level L_{i-1} that are the vertex of triangles T^1_{i-1} and T^2_{i-1} that don't belong to the diagonal. Figure 8 shows the dependencies for all the pattern orientations. Note that this dependencies are the same as that ones shown in Figure 6, but now there is no need to differentiate between centre and non-centre vertices as the restricted quadtree does.

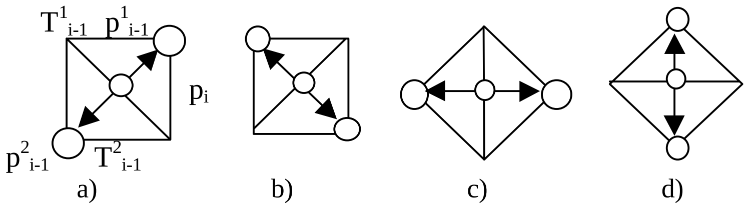


FIGURE 8. Selected vertex dependencies

- If p_i is fictitious it only depends on only one point p^k_{i-1} of level L_{i-1} that is the vertex of triangle T^k_{i-1} (k=1 o k=2) that needs p_i. That means that exist at least one child triangle of T^k_{i-1}, called T^{kj}_i whose diagonal mid point p^k_{i-1} depends on p_i. Figure 9 shows the dependency of p_i and the triangle that depends on p_i is the shadowed.

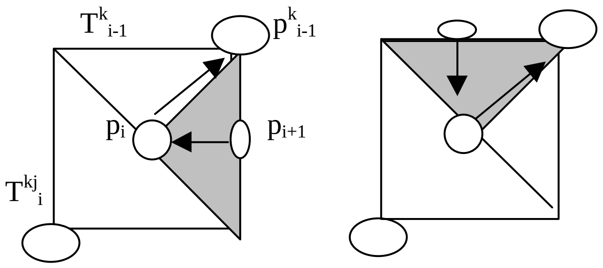


FIGURE 9. Fictitious vertex dependencies

We modify the restricted quadtree rule in such a way that if a vertex is selected to satisfy an approximation criterion the related dependencies don't need to be selected too but can be replaced with a fictitious point. A fictitious point only has a dependency vertex, so this fact reduces the vertex dependency chain, and in consequence fewer triangles will finally result. There are no level restrictions between adjacent triangles and *cracks* produced by this differences are avoided with the fictitious points.

Finally, there is no need of an auxiliary structure with the relations between points because the hierarchy is implicitly defined and the dependencies can be deduced according the point position.

3. MULTIRESOLUTION MODEL

In the last chapter we describe a hierarchical triangulation for a height field. This structure provides means to extract surface representation at variable precision. Different LODs don't have to be precomputed and stored, since the different LODs must be efficiently extractable at any desired resolution. As we see, different regions of the bitriangle structure model can be extracted at different resolution. That why we say that there are a finite number of levels limited by the number of points, but that can be almost unlimited combined according to different varying regions.

Besides it supports an adaptive model, so regions with high frequency elevation changes are modelled with more triangles per area unit than low frequency surface regions. Triangles added to avoid *cracks* are local to the critical region.

The terrain model can be used in visualisation using multiples LODs or for progressive transmission. We can generalise three steps to follow:

- Determine the level of detail of a model or a region of it;
- Select points according to the error tolerance determined from the selected level of detail;
- Update the model by adding points if it is a refinement or by deleting points if it is a simplification.

$$\begin{tikzcd}
\boxed{\text{Determine LOD of a model / area}} \arrow[r, "\textit{tolerance}"'] & \boxed{\text{Select points}} \arrow[r, "\textit{points}"'] & \boxed{\text{Update model}}
\end{tikzcd}$$

3.1. Determining LOD

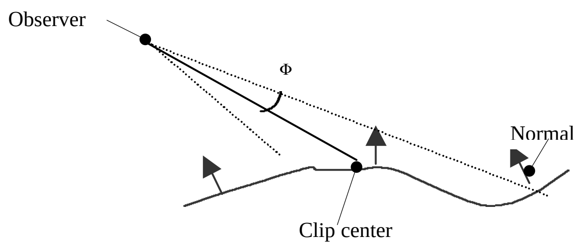
When we visualize a model, we can dynamically determine to show an area with a lower resolution obtaining the same visual result as with the finest level. We have to maximize a benefit function while minimizing the resolution based on the relative position of the observer and the area. The position is defined with:

- distance between the area and the point of view;

- angle Φ between the view direction and the line from the observer to the area;

- angle between the normal of the area and the view direction.

Areas that are far from the observer or from the view direction or areas with a great inclination from the view direction are visualized with a low resolution because we don't obtain a better result with a higher resolution.



According to the level of detail we determine an error tolerance or a similar parameter to control the selection of points (section 3.2).

3.2. Selection of points

The goal of a simplification surface process is to obtain a reduced model that uses fewer polygons that approximates the original surface in a certain degree. It is desirable that the simplification process be adaptive to the terrain structure. The local curvature of a region is an approximate measure of how fast the geometry changes at that area. A region of high curvature abruptly changes direction, and has to be modeled with more triangles per area unit than a low curvature region that remains relatively stable.

The selection of a point to contribute to a triangulation, may be done by a criterion based in an object-space error measure of the approximation, or may be done based on a criterion over a parametric representation of the surface. In the object-space, we can select a point accordingly its characteristics, such as:

- Local curvature.

- Euclidean distance from the point to the projection ver the simplified model;

- Normal of the triangles which use the vertex.

- Etc.

An alternative is to have a parametric representation of the surface and triangulate it using points that were selected in the parameter space. In [Lounsbery97] and [Gross95] the initial surface is decomposed by means of a Wavelet transform. The selection of point is done according to the resulting coefficients. [Gross95] computes the partial energy of the coefficients in regions surface. If the energy is low, the approximation of the surface can be performed with larger triangles, and that means small number of points.

To control the accuracy of the surface approximation, we can use an object-space characteristic, such as:

- Error tolerance in terms of Euclidean distance: the simplified model is within a distance from the original. It is the most common way to control an approximation, and there are numerous simplification algorithms that use it [DeHaemer91] [Schroeder92] [Varshney94] [Pajarola98].

- Error tolerance in terms of normal angular error: a triangle can replace two child triangles if its normal is within some angular tolerance. [Hinker93] eliminates over-tessellated coplanar surfaces or polygons, by using this criterion.

- Number of geometric elements such as vertices or triangles: [Hamann94] removes a percentage of triangles from the model. It uses the local curvature as the selection criteria to remove first the triangles with lower average vertex curvature.

One important difference between bitree and restricted quadtree is that in our scheme the criterion of selection of points is independent from the triangulation. The hierarchical bitree decomposition may be directly applied to the height field data in the object space, or we can have a parametric representation, select the points in the parameter space and later triangulates it. Once the points are selected with a certain criterion the model is completed with fictitious points and triangulated.

Because there is no need of a balanced hierarchy an adaptive surface triangulation can be performed, and we can locally control the approximation error.

Particularly, we use a criterion based on the Euclidean distance. For every triangle T_i of level L_i we compute its error measure $e(T_i)$ as the maximal Euclidean distance of all points whose xy-projection belongs to the domain of the triangle in the xy-projection. The computation of its error consider the child points p_k of all the k levels from level L_i to the maximal resolution level L_n.

3.3. Model updating

After selecting the points according the approximation criterion and error tolerance, a set of points is added if it is a refinement of the model or is deleted if it is a simplification. After that, fictitious points are added to complete a bitree hierarchy without cracks. Section 4 describes an implementation and a method to systematically add or delete points from a model.

When using different level of detail of a model, we have to consider the continuity of different regions at different LOD (spatial continuity) and a continuous switching between different LODs of a region (temporal continuity). With the bitree structure different regions of the bitree structure model can be extracted at different resolution. Avoiding cracks with fictitious points (section 2.2) solves the discontinuity between different regions. If we divide a very large model in tiles and represent every tile with a bitree structure, the cracks between tiles may be avoided in the same way that the cracks inside by using fictitious points.

Model updating must be done point by point or it can be all updated in one step. The first solution gives softer transitions and the second one gives a better time of global update. When we add or delete a point we can interpolate between the selected point and its correspondent fictitious to produce a continuous switching between different LODs.

Progressive Transmission

Progressive transmission or meshing [Hoppe96] denotes showing progressively better approximations to the model (for example, when a mesh is transmitted over a network, or simply to display it progressively). As most hierarchical triangulation, the bitriangle hierarchy supports progressive meshing.

We can transmit the points level by level, starting with level 0 and complete the structure with fictitious points to form a bitriangle hierarchy without cracks. A better alternative is doing an incrementally updating based on an approximation error. We initially form a bitriangle hierarchy with a selected error threshold, and transmit it. After that we refine the structure whit a smaller threshold, and transmit the added points. This process continues until all the points are transmitted. The receptor completes the bitree structure with fictitious points to avoid cracks. Note that a fictitious point doesn't need to be transmitted, since it may be calculated.

4. IMPLEMENTATION: *"TRIANGLE STRIP"* STRUCTURE

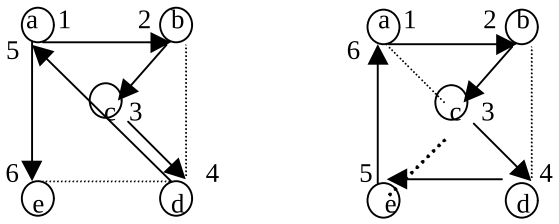
Triangle strips are a construction efficiently supported by hardware rendering engines and graphics. They take less space than the implementation with separate triangles.

Instead of defining a triangle with three points, a *triangle strip* constructs triangles from an ordered list of points a, b, c, d, e… by grouping by three consecutive points, such as: abc, bcd, cde… In a triangle strip, successive triangles must always share an edge.

If we want to change the orientation of a triangle, invisible *line triangles* may be used. For example, if we want cbd instead of bcd, the sequence of points has to be a, b, c, b, d, e, … and the triangles will be abc, bcb, cbd,… with a line triangle bcb. Line triangles don't generate calculation errors and take less rendering time than a regular triangle.

In a bitree structure, if we have two adjacent triangles that share an edge, these triangles can be queued up next to each other. If we have two adjacent triangles that don't share an edge, that is if they have different levels, they can't be queued up next to each other, so a line triangle has to be inserted in the mid of them.

Figure 10 shows an example where we want to generate the triangles **abc**, **bcd** and **dae**. Figure 10-a shows a right sequence where the line triangle **cda** is added, but Figure 10-b shows a wrong sequence that generates the undesirable triangle **cde**. Every adjacent edge must be explicitly generated, such as **cd** and **da** in Figure 10-a.



Sequence: a, b, c, d, a, e
Triangles: abc, bcd, *cda*, dae
a) Right sequence

Sequence: a, b, c, d, e, a
Triangles: abc, bcd, **cde**, dea
b)Wrong sequence

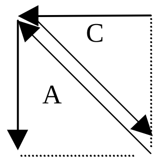
FIGURE 10. *Triangles strips* sequence

We define a rule to generate a desirable *triangle strip* sequence from recursively traverse a bitree hierarchy, that says:

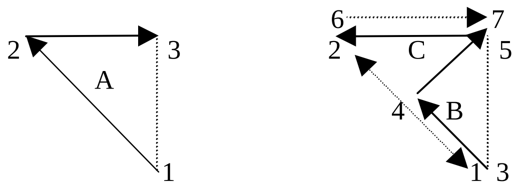
1. Two adjacent triangles that share an edge completely will be queued in the sequence and the shared edge will be explicit.

2. Two adjacent triangles that don't share an edge completely will be queued in the sequence and both edges will be consecutive and explicit in the sequence. A line triangle will be generated between the two triangles.

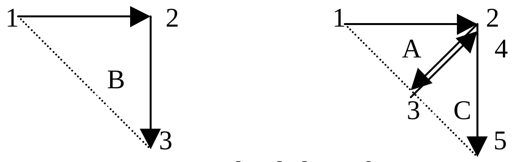
Starting with the sequence of Figure 11-a, we classify the triangles sequences and show modify it when the triangle is split (Figure 11-b, 11-c, 11-d).



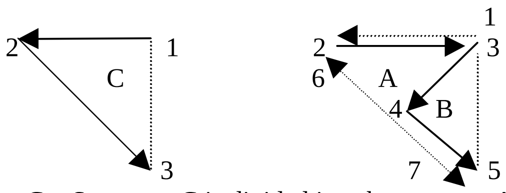
a) Initial sequence



b) Sequence A – Sequence A is divided in subsequences B and C



c) Sequence B – Sequence B is divided in subsequences A and C



d) Sequence C – Sequence C is divided in subsequences A and B

FIGURE 11. Sequence classification and division

Let’s see the subdivision of a 3 x 3 grid in Figure 12. Figure 12-a shows the initial bitriangle and the sequences C and A; Figure 12-b shows the sequence C subdivision in sequences A and B; Figure 12-c shows the sequence A subdivision in sequences B and C; Figure 12-d, 12-e and 12-f go on with this subdivision.

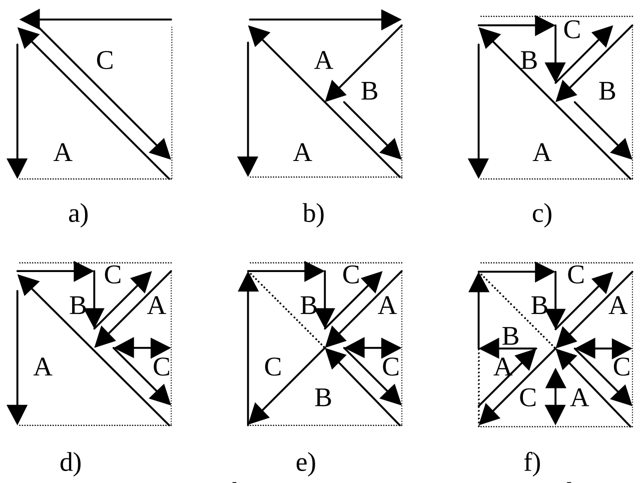


FIGURE 12. *Triangle strip* generation in a 3x3 grid

Besides the efficiency of rendering a triangle mesh, a strip triangle also offers an economic use of space. In a full 3x3 mesh there are 8 triangles. The number of vertices used for independent triangles is 24, while using the strip shown in Figure 12-f, are needed only 15. In the following chapter the results obtained with a real case are exposed.

5. COMPARISON OF RESULTS

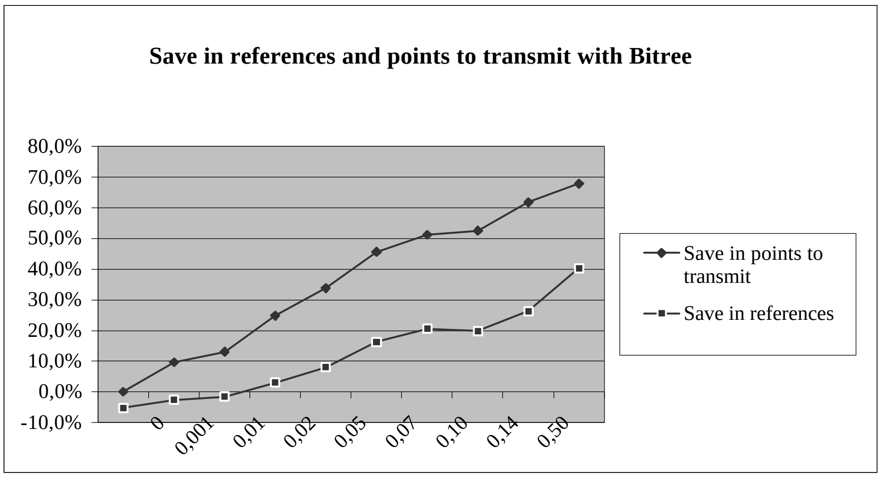
We evaluate our scheme in a 65 x 65 grid with terrain data of a zone with variable topography, with heights from 0 to 1445 meters. Figure 13-a shows the geometry and a mapped texture of an area. Figure 13-b shows the triangle mesh with the full set of points. Figure 13-c shows a simplified model where the redundant points were eliminated. This model uses 65% of the triangles and 65% of the points, and it has a zero error tolerance. Figures 13-d and 13-e show two simplified models with different no null error tolerances.

Figure 14 shows compares how bitree triangulation adds less triangles (Figure 14-a) than restricted quadtree (Figure 14-b) from the same set of selected points. We can see that restricted quadtree triangulation adds more non-local triangles to avoid cracks.

Table 1 compares bitree triangulation with the restricted quadtree triangulation scheme presented in [Pajarola98]. It shows the points added to avoid cracks, the points to transmit over network (fictitious points are excluded because they are deducible), number of triangles and number of references of the *triangle strip*.

Error Tol.	**Points Added**			**Points to transmit**			**Triangles**			**References**		
	BT	**QT**		**BT**	**QT**		**BT**	**QT**		**BT**	**QT**	
Orig.	0	0		4225	4225	0%	8192	8192	0%	13655	12972	-5%
0%	215	272	21%	2533	2805	10%	5296	5843	3%	8685	8467	-2%
1%	344	407	15%	1235	1642	25%	2899	3184	9%	4643	4788	3%
2%	281	360	22%	704	1064	34%	1736	2041	15%	2803	3049	8%
7%	161	241	33%	229	470	51%	647	884	27%	1037	1307	20%
10%	144	202	29%	182	384	53%	527	715	26%	851	1061	20%
14%	89	142	37%	88	230	62%	268	418	36%	443	601	26%
50%	10	19	47%	9	28	68%	23	41	44%	37	62	40%

TABLE 1. Comparison of results obtained with Bitree and Quadtree



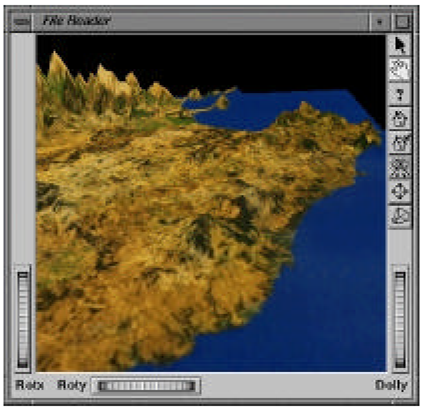
GRAPHIC 1. Save in references and points to transmit with Bitree

Graphic 1 shows the saving with Bitree in the number of points to transmit and in the number of references in the *triangle strip*. We can observe the greater the error tolerance the greater the saving. That means that bitree works much better than quadtree in simplified models. In a high resolution model bitree presents a low overhead in the number of references because its sequence is prepared for a possible jump between levels in a future highly simplified model.

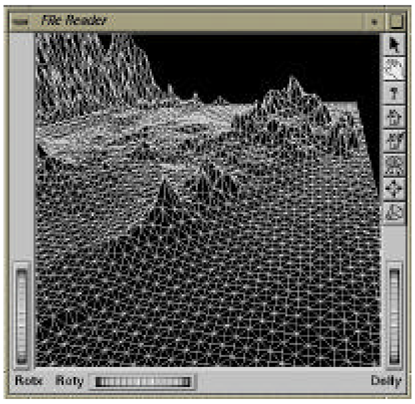
Table 2 compares the *strip triangle* with the separate triangles implementation. We can see that s*trip triangles* adds line triangles, but finally it saves a mean of 44% in the number of references.

Error Tolerance	**Triangles**	**References Separate tr.**	**Line tr.**	**References *tr.strip***
Original Mesh	8192	24576	5461	13655
0%	5296	15888	3446	8685
2%	1736	5208	1172	2803
7%	647	1941	441	1037

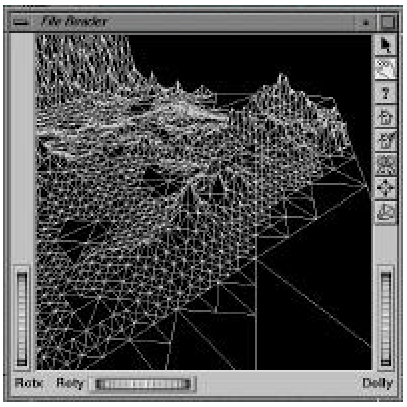
TABLE 2. Comparison between *triangle strips* and separate triangles implementation



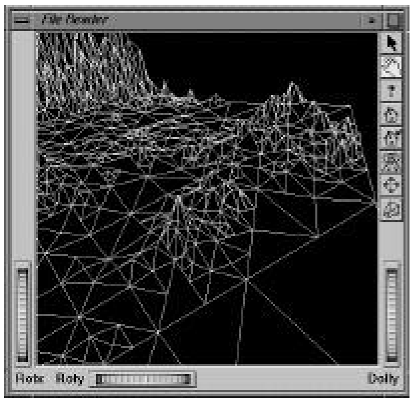
a) Original mesh with a texture mapped



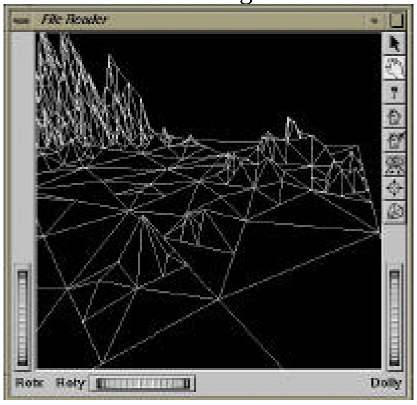
b) Original mesh:
100% points selected
100% triangles



c) Error tolerance: 0%
Same appearance than original model
60% points selected + 5% Fictitious
65% triangles

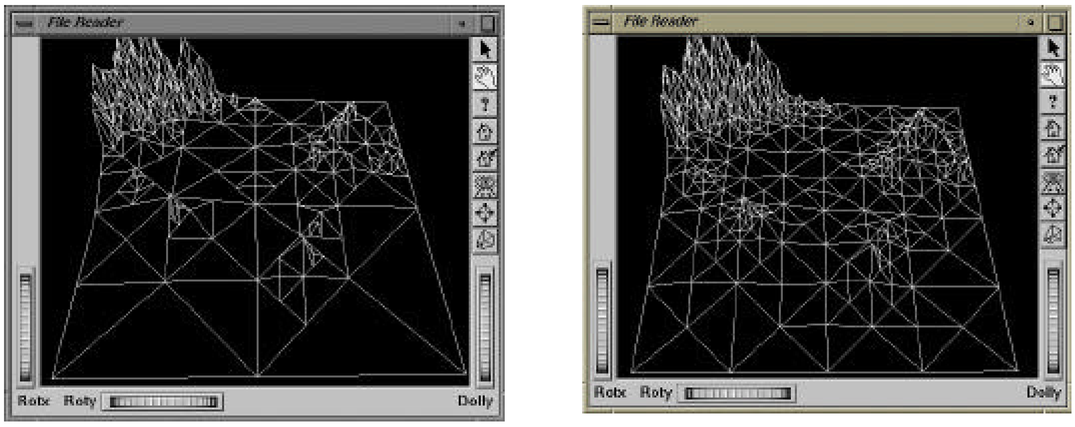


d) Error tolerance: 2%
17% points selected + 7% Fictitious
21% triangles



e) Error tolerance: 7%
5% points selected + 4% Fictitious
8% triangles

FIGURE 13. Bitree triangulation according the error tolerance



a) Bitree

b) Quadtree restringido

FIGURE 14. Points added to avoid *cracks*

6. CONCLUSION

We have proposed a multiresolution model based on non-restricted triangle hierarchy free of cracks. We can use any approximation criterion for the selection of points independently from the hierarchy. The structure is completed to avoids cracks in a way that minimise extra triangles. As a consequence, it simplifies in a higher degree than other restricted model as such presented in [Pajarola98].

This model provides a triangulation data structure that supports adaptive and multiresolution triangulation. That's why it can be integrated in a visualisation system as the core part. Besides, it addresses well to typical problems in computer graphics, as it supports mesh simplification, selective refinement, mesh compression, continuous LOD and progressive meshing [Hoppe96].

As future work we propose a *framework* that considers not only a geometry model but also a multiresolution model of textures. As other improvements we can add different data type from a geographic information system such as routes, bridges, buildings, etc.

Acknowledgments

Our acknowledge to the Geographic Information System Laboratory of the University of the Balearic Islands for the terrain data of the Balearic Islands used in this work.

REFERENCES

[Casillas99] J.Casillas, J.Sevilla, J.Blasco, C.Pardo, C.Romero, J.Fernandez. "Implementación de un sistema visual para un simulador de helicóptero". Congreso Español de Informatica Grafica, CEIG 99, Jaen, España, junio de 1999

[DeHaemer91] M.DeHaemer, M.Zyda. "Simplification of Objects Rendered by Polygonal Approximations". *Computer & Graphics*, 15(2): pages 175-184, 1991.

[Erikson96] C.Erikson. "Polygonal Simplification: An Overview". TR96-016, *Department of Computer Science, University of North Carolina - Chapel Hill.* USA.

[Gross95] M. Gross, R. Gatti, O. Staadt. “ Fast Multiresolution Surface Meshing”. *Internal report no. 230. Institute for Information Systems. ETH Swiss Federal Institute of Technology.* Zürich, Switzerland.

[Hamann94] B.Hamann. “A Data Reduction Scheme for Triangulated Surfaces”. *Computer Aided Geometric Design,* 11(2): pages 197-214, 1994.

[Heckbert94] P.Heckbert, M.Garland. “Multiresolution Modeling for Fast Rendering”. *Proceedings of Graphics Interface ´94, Banff, Alberta, Canada, May 1994.*

[Hernández99] L.Hernández, J.Taibo, A.Seoane. "Una aplicación para la navegación en tiempo real sobre grandes modelos topográficos". Congreso Español de Informatica Grafica, CEIG 99, Jaen, España, junio de 1999

[Herzen87] B.Von Herzen and A.Barr. “ Accurate triangulations of deformed, intersecting surfaces”. *Computer Graphics, (SIGGRAPH´87 Proceedings),* pages 103-110, 1987.

[Hinker93] P.Hinker, C.Hansen. “Geometric Optimization”. *Proceedings of Visualisation,* pages 189-195, 1993.

[Hoppe93] H.Hoppe, T.DeRose, T.Duchamp, J.McDonald, W.Stuetzle. “Mesh Optimization”. *Computer Graphics, (SIGGRAPH´93 Proceedings),* pages 19-26, 1993.

[Hoppe96] H.Hoppe. “Progressive Meshes”. *Computer Graphics, (SIGGRAPH´96 Proceedings),* pages 99-108, 1996.

[Hoppe97] H.Hoppe. “View-Dependent Refinement of Progressive Meshes”. *Computer Graphics, (SIGGRAPH´97 Proceedings),* pages 189-198, 1997.

[Lounsbery97] M.Lounsbery, T.DeRose, J.Warren. “Multiresolution Analysis for Surfaces of Arbitrary Topological Type”. *ACM Transactions on Graphics,* v.16, no.1, 1997.

[Pajarola98] R.Pajarola. “Large scale Terrain Visualisation using the Restricted Quadtree Triangulation”. Internal report.292 *Institute of Theorical Computer Science. ETH Swiss Federal Institute of Technology.* Zürich, Switzerland, 1998.

[Schroeder92] W.Schroeder, J.Zarge, W.Lorensen. “Decimation of Triangle Meshes”. *Computer Graphics, (SIGGRAPH´92 Proceedings),* 26(2): pages 65-70, 1992.

[Varshney94] A.Varshney. “Hierarchical Geometric Approximations”. PhD Thesis. *Department of Computer Science, University of North Carolina-Chapell Hill, USA,* 1994.