A FRAMEWORK FOR REAL-TIME TERRAIN VISUALIZATION WITH ADAPTIVE SEMI-REGULAR MESHES

Lourena Rocha, Sérgio Pinheiro, Marcelo B. Vieira, Luiz Velho
IMPA - Instituto Nacional de Matemática Pura e Aplicada Estrada Dona Castorina, 110, 22460 Rio de Janeiro, RJ, Brasil
{lourena, sergio, mbvieira, lvelho}@visgraf.impa.br

Abstract: This work introduces a framework for large terrain data visualization using a highly expressive semi-regular mesh. The main advantage of this structure is that it can represent most objects in real world uniquely and transparently. Stellar operations provide a simple interface for local adaptation such as simplification and refinement. This interface allows common procedures for terrain visualization such as culling, geomorphing and error function evaluation to be easily included into the adaptation algorithm. Results of the first stage of our developments show that the goal of achieving a complete framework for visualizing the whole Earth terrain is promising.

Key words: Real-time terrain visualization, semi-regular meshes, 4-8 mesh, stellar operations.

1. INTRODUCTION

Interactive terrain visualization is an important research area with applications in GIS, games, virtual reality, scientific visualization and flight simulators, besides having military use.

This is a complex and challenging problem considering that some applications require precise visualizations of huge data sets at real-time rates. In general, the size of data sets makes rendering at real-time difficult since the terrain data cannot fit entirely in memory. In such cases it needs to be partially loaded from a high capacity secondary memory. Moreover, the

renderer must process a large number of small triangles corresponding to distant terrain.

1.1 Motivation

In order to visualize terrain we employ real world modeling and visualization. Thus, it is expected that systems for terrain visualization are able to visualize the whole globe.

With recent advances in technology, such as satellites, high-capacity storage systems and internet, there are some systems that visualize the Earth or some specific regions, at different resolutions. Examples of such systems are *Keyhole* [Keyhole], *TerraServer* [TerraServer] and *TerrraFly* [TerraFly].

They are all based in aerial photography and satellite images, supplying 2D visualization. The exception is the *Keyhole* system, which possesses a layer of 3D visualization. However, the elevation data used is limited and it has the disadvantage of being commercial. Only *TerraServer* and *TerraFly* are available on the internet for free.

A complete system for terrain visualization must give support to the process of modeling, as well as the run-time management. As main abilities it is expected that it can:

- represent the entire Earth;
- deal with huge amount of data in an efficient way;
- supply visualization in real-time;
- have a flexible structure of modeling to be used in as many applications as possible.

To get interactive rates of visualization, a widely adopted solution is the use of some multiresolution adaptive scheme. In this case, a multiresolution representation of the terrain model is precomputed and used at run-time to construct a suitable terrain triangulation for each frame adaptively.

1.2 Contribution

Despite the maturity of the area of interactive terrain visualization, the main works are divided between those which represent terrain as height values sampled over a squared grid -- *regular triangulation* -- and those which perform the sampling over a triangular network -- *irregular triangulation*. However, another group is emerging whose works are based on hybrids methods exploring the advantages of regular and irregular representations.

In this work, we propose a system for real-time terrain visualization. Our goal is, at long term, to build a complete framework capable of dealing with huge amount of data, and supplying real-time visualization of images with

great visual quality. We use a hybrid, flexible and powerful structure of modeling which allows, in the specific case of terrain, the entire Earth modeling.

The proposed platform will have a real application in association with INPE - National Institute for Space Research - in the *TerraLib* [TerraLib] project. *TerraLib* [Camara et al., 2000, 2001] is a GIS classes and functions library, available from the internet as open source, allowing a collaborative environment for the development of multiple GIS tools. Its main goal is to support the development of a new generation of GIS applications, based on the technological advances of spatial databases.

This paper is organized in five sections. Section 2 presents a brief overview of related works and Section 3 shows the concepts and architecture of a general framework for real-time terrain visualization. Section 4 describes the hybrid structure -- *semi-regular 4-8 mesh* -- used in our platform and the implementation aspects of the first version of our system. Section 5 shows the results obtained up to now and Section 6 summarizes the main points of the work and gives directions of future research.

2. RELATED WORK

In the last years, many algorithms and strategies to perform terrain visualization in real-time have been developed. In this section we present an overview of the main works classified by the triangulation structure used for dynamic multiresolution management.

2.1 Regular Triangulation

The regular schemes work on square grids of dimension $2^n + 1$. The vertex positions are restricted to regular positions on the grid. Examples of schemes using such structure are described at [Lindstrom et al., 1996], [Duchaineau et al., 1997] and [Lindstrom and Pascucci, 2001, 2002].

In general, these algorithms are easily implemented and the regular structure is pointerless resulting in low memory costs. However, since they store only geometric information, it is difficult to represent more complex terrain models.

The regular structure forces the use of saturated error metrics which is not optimal, and for some applications, it is a limitation. Moreover, the sampling regularity causes the extracted meshes to be generally suboptimal with respect to the number of triangles.

The triangulation structure proposed in this work stores the current mesh topology without high memory costs. This allows the use of different error metrics and makes the system more flexible for general applications.

2.2 Irregular Triangulation

The irregular schemes represent terrain as height values sampled over a triangulated irregular network (TIN). Works based on TINs are described at [Hoppe, 1997, 1998] and [Floriani et al., 1997, 2000a, 2000b].

These algorithms usually require less polygons than the regular ones to approximate the surface for a given error. These methods are also flexible regarding the use of different error metrics.

However, they tend to require more complex data structures to represent a multiresolution triangulation and dependency relations, leading to a higher computational overhead associated with simplification and refinement operations compared to regular hierarchical methods. The irregular structures have less expressive power [Puppo, 1998] than the regular ones to create adaptive meshes, and consequently do not exploit optimally the resolution levels feature.

Our triangulation scheme stores the current mesh structure with no associated cost, since it does not store the hierarchy pointers. Also, our variable resolution structure (see section 4) satisfies the criteria of effectiveness discussed by Puppo [1998].

2.3 Hybrid Methods

The hybrid schemes exploit the advantages of both approaches: regular and irregular. The main works that use this kind of structure are described at [Pajarola et al, 2002] and [Cignoni et al, 2003].

In [Pajarola et al., 2002], a regular hierarchical structure is used to approximate the irregular terrain data. Cignoni et al [2003] perform a fast VLOD extraction by considering a rectangle patch as the smallest primitive for simplification and refinement.

Although these methods are more advantageous than the regular and irregular approaches, they suffer drawbacks as sub-optimal mesh approximations and poor frustum culling.

The triangulation structure proposed in this work is hybrid as well. It is based on a triangulated quadtree and has the following features:

- is semi-regular;
- stores only the current terrain topology;
- does not store pointers;
- is flexible regarding the use of error metrics;

- satisfies the optimal criteria defined by Puppo [1998]: high expressive power, logarithmic depth and linear growth.

3. REAL-TIME ADAPTIVE TERRAIN VISUALIZATION

Real-time terrain visualization is still a challenge due to geometric and topological complexity of the terrain data and its massive volume. Complex calculations must be avoided and the amount of data processing must be reduced in order to maintain a minimum and constant frame rate.

Thus, the terrain visualization process should be divided in two stages. The first stage consists of formatting and extracting techniques to obtain information about data geometry and topology. This stage is called *preprocessing stage*. The second stage involves rendering techniques that exploit the data structures and other information generated in the preprocessing stage to accelerate the visualization process. It is called *runtime stage*.

3.1 Preprocessing Stage

The mathematical model used for terrain representation defines it as a rectangular area of the plane and associates a height and color information to each point inside that area. Thus, a terrain is defined by an *altimetry function* h: $\mathbf{U} \rightarrow |$ and by a *texture mapping* c: $\mathbf{U} \rightarrow |^n$ (Fig. 1).

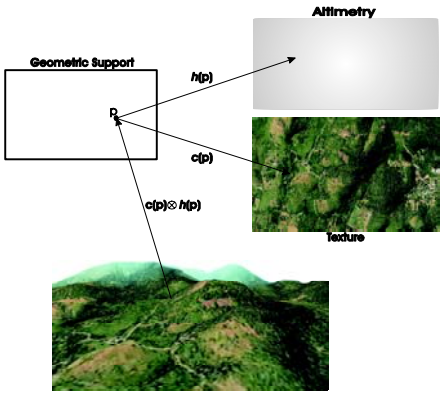


Figure 1. The terrain mathematical model

The discrete representation of function *h* is called *height field*. The matrix height field representation (uniform sampling) is named *digital elevation model* (DEM). The reconstruction operation of function *h* is obtained by linear interpolation of the DEM. The interpolating operation generates a piecewise linear surface. It is usually a triangulated mesh that can be regular or not.

Real-time mesh generation for terrain visualization can be seen as an optimization problem satisfying a set of restrictions. The error functions, named *adaptation functions*, can be defined by several criteria such as visual quality of the frame, geometric accuracy, etc. The restrictions typically used impose the maximum of: frame rate, number of triangles per frame, memory use, etc.

The criteria based on visual quality are named *view-dependent* since their error metrics are computed from camera parameters. Criteria based on geometric error only consider the geometry and topology properties of the mesh. They are named *view-independent*. In order to reach a good relationship between visual quality and geometric accuracy, the error function is composed by both criteria.

Greedy algorithms are used to find a mesh that minimizes the error function. These algorithms are based on the fact that the topological elements (vertices, edges and faces) can be ordered by the adaptation function.

The appropriate data structure to maintain elements always ordered is a priority queue. The priority of each element can change over time. This is important for incorporating view-dependent criteria in greedy algorithms.

The calculation of the geometric error is based on evaluating the maximum error of a terrain region. The evaluation process is made by generating the full resolution mesh. After that, in each step a vertex is removed from the mesh. The error is computed considering the disparity between the original mesh and the new one without the vertex.

The most used metric is based on the star vertex. The star vertex is the area of the mesh that is influenced when the vertex is removed (simplification). The vertex error is measured through the geometric disparity in its star. There are several techniques to evaluate the vertex error as height, area or volume variation. This variation is usually measured using the maximum error (e.g. the metric L^{∞}).

Note that in multi-resolution representation, the vertex neighborhood grows and include the other stars of vertices that were removed in previous steps. Intuitively, this means that the stars of vertices form nested areas. The error attributed to that last vertex is the maximum among the error evaluated in its star and evaluated in the nested stars. This process generates an error matrix with same dimensions of DEM.

These error metrics are used to accelerate processes, as culling and adaptation, which are essential for real-time mesh visualization. Nested volumes are used to accelerate culling. Their computation only depends on topological and geometric attributes. Therefore, it is possible to compute the geometric error for each topological element only once in the preprocessing stage.

The texture multi-resolution representation should be created in the preprocessing stage. It's not possible create distinct resolution levels in real-time. The texture multi-resolution representation is built through filtering and resampling operations that cannot be executed in real-time. The construction process generates a pyramid of images with different resolutions.

3.2 Run-Time Stage

The error based on view-dependent criterion is evaluated for each frame in the run-time stage. Every topological element is then associated to an error formed by combining the view-independent error, obtained in the preprocessing stage, and the view-dependent error.

The adaptation process starts with the base mesh. Only refinement operations are allowed in the initialization step. The refinement operation consists of adding a vertex in the mesh. It implies subdividing an edge and creating four faces when the vertex belongs to the mesh interior and two faces when it belongs to the border. When the mesh reaches the maximum resolution level, only simplification operations are allowed. The simplification operation is the inverse of the refinement operation. It removes a vertex and eliminates the edges and faces that were created in the refinement step. These operations are executed satisfying the topological constraints to guarantee that the new mesh is always valid.

The goal of the priority queue is to determine the order of the simplification and refinement operations. The implementation uses two priority queues, one for each operation. The simplification queue contains the vertices that should be removed from the mesh. The refinement queue contains the edges that should be subdivided.

The elements are classified in the queue in decreasing order of priorities determined by the adaptation function. The greedy algorithm decides which queue is accessed at a time (order of the refinement and simplification operations). The queue top elements are processed until the top element of the other queue has greater priority. The simplification and refinement operations are executed until the mesh reaches the target error. This target error can be defined statically or dynamically.

The static error is specified by the user during run-time. The advantage of this approach is fast implementation. The disadvantages are the difficulty of specifying a error value that gives a good relationship between visual quality and performance and the dependence between error specification and hardware technology.

The dynamic error is determined automatically by the system in order to satisfy restrictions as frame rate, number of triangles per frame or the user's point of view. In this case the system tries to evaluate the maximum error that satisfies the restrictions determined by the user. The advantages of using the dynamic error are that it is more intuitive for the user to specify the restrictions parameters and allows the system to easily adapt itself to different run-time configurations.

The static and dynamic error techniques can be combined in the visualization process. The visualization mode is called *interactive* when the camera parameters are changing over time, otherwise the visualization mode is called *progressive* (the camera is stopped). The usage of dynamic error is appropriate in interactive mode and the static error is suitable in progressive mode.

Massive volumes of data should be loaded in high-speed memory (texture and RAM) to be visualized. In general, these memories have small storage capacity. This problem is solved by using a memory management system.

The memory management model most used is the paging system. This model splits the storage device in blocks of same size. It is suitable to manage data that can be uniformly splitted and has low implementation complexity.

The visual quality of the visualization process is closely related to the memory system performance. The memory system efficiency is related to its ability to fetch the amount of data possible and necessary to generate a frame. Predictive loading can be used to achieve this requirement. Prediction exploits spatial and temporal coherence to determine the data that should be loaded in advance [Pinheiro and Velho, 2002].

4. IMPLEMENTATION

In this section we discuss the implementation aspects of the first version of our system. The method implemented is view-dependent and performs view frustum culling and geomorphing. At this point, we are interested in address just the flexibility and power of the hybrid structure used in our framework: the semi-regular 4-8 mesh (see section 4.1).

Frustum culling, geomorphing, as well as the calculus of geometric error and bounding spheres radius performed at the preprocessing stage are similar to the ones described at [Lindstrom and Pascucci, 2002]. Section 4.2 summarizes the implemented top-down algorithm and shows what would be the incremental algorithm - still under development.

4.1 Semi-Regular 4-8 Meshes

The 4-8 mesh is a particular case of 4-*k* meshes [Velho and Gomes, 2000], which are a specialization of the general variable resolution structure introduced independently by De Berg et al [1995] and by Puppo et al [1998]. The semi-regular 4-8 mesh, a particular case of 4-8 mesh, is a tessellation which contains isolated extraordinary vertices with valence different than 4 or 8.

This mesh structure is created from a coarse irregular mesh, with a triangulated quadtree structure [Velho, 2000], by applying a semi-regular 4-8 refinement method that introduces only regular vertices [Velho and Gomes, 1999]. In that way, as the mesh is refined, extraordinary vertices from the initial mesh are surrounded by regular vertices with valence 4 or 8.

The variable resolution structure of a semi-regular 4-8 mesh has optimal properties, according to the three criteria used to analyze a variable resolution discussed in Puppo et al [1998]: *high expressive power*, *logarithmic depth* and *linear growth*.

Note that the semi-regular 4-8 mesh can be encoded by a pointerless structure, which implies in a compact representation.

In our framework, we use the A48 library [A48] described at [Velho, 2004]. A48 is a dynamic adaptive mesh library which maintains a conforming triangulation of time-varying surfaces using the variable resolution structure of a semi-regular 4-8 mesh. The user supplies an initial mesh, a surface sampling procedure and a set of adaptation criteria. The dynamic mesh is automatically modified in order to conform to user defined characteristics. The mesh has also an underlying semi-regular multiresolution structure.

The main features of the library are:

- simple dynamic adaptive mesh library based on the half-edge, a standard topological data structure. The implementation of this new adaptive multiresolution functionality does not require any extra storage in the representation. Also, because the half-edge is widely adopted, it should be easy to apply our system in many applications.
- topological elements are vertices, edges and faces.

- a conformal mesh structure that dynamically changes its resolution based on user defined criteria. This makes the associated adaptation capabilities very general and powerful.
- an effective mechanism for refinement and simplification of semi-regular meshes that maintains a restricted multiresolution structure. This mechanism is based on the concept of a restricted binary multi-triangulation and stellar theory.

The definition of stellar operations makes possible the development of a simple interface for the mesh adaptation operations: simplification and refinement. This interface complements the traditional topological query operators; it consists of only a few functions and allows an easy incorporation of culling and geomorphing operations, besides error computations.

4.2 Algorithm

As explained in Section 3, some computations must be done in a preprocessing stage in order to minimize the amount of data to be processed at runtime. From this stage we obtain as input for the runtime stage the geometric error and the bounding-sphere radius maps [Lindstrom and Pascucci, 2002].

Our algorithm is greedy and considers that topological elements - vertices and edges - can be sorted using the error function. The data structure used to perform this task is the priority queue. Vertices are stored in the heap of simplification and edges in the heap of refinement. Then, starting with a coarse mesh, the adaptive algorithm removes vertices from the mesh and put edges in the correspondent heap - simplification operation - or, removes edges from the heap and put vertices in the mesh - refinement operation.

We summarize below the top-down algorithm, which uses only the priority queue for refinement. Then, we describe an approach, still under development, for the incremental algorithm. This last one uses one priority queue for refinement operation and other for simplification operation.

Top-Down Algorithm:

1. Read input files: height map, geometric error map, bounding-sphere radius map.
2. For each frame:
   a) Initialize the base mesh;
   b) Read the camera parameters which are the view-dependent parameters;

c) Fill the refinement queue using the geometric error, the distance between the observer and the vertex position and the bounding-sphere radius. Thus, edges with higher errors will be refined first.
d) Edges in the refinement queue are refined if they satisfy the culling test and the adaptation criterion. Then, the new vertex is added to the mesh and its coordinate *z* the real world is determined using the geomorphing criterion.
e) Render the current mesh.

Incremental Algorithm:

1. Read input files: height map, geometric error map, bounding-sphere radius map.
2. Fill the refinement queue using the same criterion as above e make an initial refinement. Each refined edge produces a new vertex, which is inserted at the simplification queue.
3. For each frame:
   a) Read the camera parameters which are the view-dependent parameters;
   b) Recompute priorities in the refinement and simplification queue.
   c) Vertices are simplified according to adaptation criterion. Then, they are removed of the simplification queue and the new edges produced are inserted at the refinement queue.
   d) Next, edges at the refinement queue satisfying the adaptation criterion are refined and the produced vertices are inserted at the other queue as described above.
   e) Render the current mesh.

5. RESULTS

The experiments were done using a DEM of 4097x4097 points and a texture of 4096x4096 pixels. The terrain was visualized using the static error based on geometric error (Fig. 2a) and the dynamic error based on number of triangles per frame.

In both cases the adaptation function was defined by combining the geometric error (view-independent) and the distance of the topological element to the viewer (view-dependent) (Fig. 2b). Frustum culling was performed using the nested spheres map defined by Lindstrom (Fig. 2c). Figure 3 shows the final results using all static error functions and frustum culling.

The computer used in the experiments was a Pentium4 2.4Ghz with 1GByte of RAM memory and the graphic device was a GeForce4 Ti.

6. CONCLUSIONS

This section concludes the paper with a review of the results and a discussion of on-going work.

6.1 Overview

The main goal of this work was to introduce a framework for interactive terrain visualization. The results show the first stage of our developments that emphasizes the use of a semi-regular mesh for terrain representation. The advantage of using semi-regular meshes is that it allows representing most objects of the real world.

The construction method of the mesh is based on the refinement and simplification stellar operators. These operations are based on stellar operators which allow to adapt the mesh locally. Stellar operations have allowed to easily incorporate frustum culling, geomorphing and error function to the mesh adaptation algorithm.

The error function employed is based on a saturated error metric. This error metric is appropriate to build the mesh using top-down search. Therefore, for each frame it is necessary start from the base mesh. Following, the base mesh should be refined to obtain the adapted mesh defined by the error function.

Although we have not used the incremental method, the system performance has been very satisfactory. The system has guaranteed a frame rate of 30fps in cases where the mesh has a complex geometry.

The results obtained in the initial system implementation can be improved achieving a better performance of the visualization system, for instance using triangle strip. On the other hand, note that we have not explored the fact that the topology of the mesh is explicitly stored. The topological information can be used to develop an incremental adaptation algorithm, completely different of the top-down adaptation that has been used to generate the results. The topological information can also be used to provide other kinds of processing, for instance normal and curvature evaluation of the mesh.

6.2 On-going Work

Due to the spatial and temporal coherence existent in the visualization process, the meshes generated in a frame sequence are very similar. The incremental adaptation algorithm would adapt regions that are out of error tolerance, only. Therefore, the development of such algorithm is essential to increase the system visualization performance.

Nowadays, it is possible to obtain terrain data and satellite image of the entire Earth. The real-time visualization of this massive volume of data is possible only by developing a memory management system. This system should support multi-resolution and predictive loading [Pinheiro and Velho, 2002].

This work is under development, thus it has not been possible to compare our technique with other representation and visualization techniques of terrain data. Also, a detailed case studying will be made to measure the performance of our system in comparison with other works.

ACKNOWLEDGEMENTS

This work was partially supported by CNPq under grant number 552040/02-9 as part of the TerraLib Project.

REFERENCES

A48 mesh library. http://w3.impa.br/ lvelho/a48

Keyhole. http://www.keyhole.com

Terrafly. http://terrafly.com

Terralib. http://terralib.dpi.inpe.br

Terraserver. http://terraserver.microsoft.com/

Camara, G., de Souza, R. C. M., Pedrosa, B. M., Vinhas, L., Monteiro, A. M. V., Paiva, J. A., de Carvalho, M. T., and Gattass, M. (2000). Terralib: Technology in support of gis innovation. In *II Workshop Brasileiro de Geoinformatica, GeoInfo*.

Camara, G., Vinhas, L., Souza, R. C., Paiva, J., Monteiro, A. M., de Carvalho, M. T., and Raoult, B. (2001). Design patterns in gis development: The terralib experience. In *III Workshop Brasileiro de Geoinformatica, GeoInfo*.

Cignoni, P., Ganovelli, F., Gobbetti, E., Marton, F., Ponchio, F., and Scopigno, R. (2003). BDAM – batched dynamic adaptive meshes for high performance terrain visualization. *Computer Graphics Forum*, 22(3):505–514.

De Berg, M. and Dobrindt, K. (1995). On level of details in terrains. In *Proc. 11th Annual ACM Symp. on Computational Geometry*, Vancouver, B.C.

Duchaineau, M. A.,Wolinsky, M., Sigeti, D. E., Miller, M. C., Aldrich, C., and Mineev-Weinstein, M. B. (1997). ROAMing terrain: real-time optimally adapting meshes. In *IEEE Visualization*, pages 81–88.

Floriani, L. D., Magillo, P., Morando, F., and Puppo, E. (2000a). Dynamic view-dependent multiresolution on a client-server architecture. *Computer-Aided Design*, Volume 32(Issue 13):805–823.

Floriani, L. D., Puppo, E., and Magillo, P. (1997). A formal approach to multiresolution modeling. In *Geometric Modeling: Theory and Practice*, pages 302–323. Springer-Verlag. Floriani, L. D., Puppo, E., and Magillo, P. (2000b). VARIANT: A system for terrain modeling at variable resolution. *GeoInformatica*, 4(3):287–315.

Hoppe, H. (1997). View-dependent refinement of progressive meshes. *Computer Graphics*, 31(Annual Conference Series):189–198.

Hoppe, H. (1998). Smooth view-dependent level-of-detail control and its application to terrain rendering. In Ebert, D., Hagen, H., and Rushmeier, H., editors, *IEEE Visualization '98*, pages 35–42.

Lindstrom, P., Koller, D., Ribarsky, W., Hodges, L., Faust, N., and Turner, G. (1996). Real time continuous level of detail rendering of height fields. *Proceedings of SIGGRAPH'96*, pages 109–118.

Lindstrom, P. and Pascucci, V. (2001). Visualization of large terrains made easy. In *Proceedings of the Conference on Visualization '01*, pages 363–371. IEEE Computer Society.

Lindstrom, P. and Pascucci, V. (2002). Terrain simplification simplified: A general framework for view-dependent out-of-core visualization. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):239–254.

Pajarola, R., Antonijuan, M., and Lario, R. (2002). Quadtin: Quadtree based triangulated irregular networks. In *Proceedings of the Conference on Visualization '02*, pages 395–402. IEEE Computer Society.

Pinheiro, S. and Velho, L. (2002). A virtual memory system for real-time visualization of multiresolution 2d objects. In *Proceendings of WSCG*.

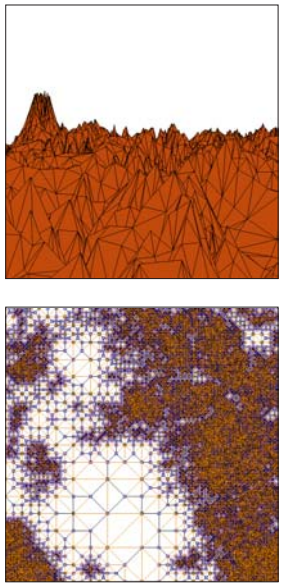
Puppo, E. (1998). Variable resolution triangulations. *Computational Geometry Theory and Apllications*, 11(3-4):219–238.

Velho, L. (2000). Quandrangulation using 4-8 clustering. In Vaz, L. E., editor, *21st Iberian Latin American Congress on Computational Methods in Engineering (CILAMCE 2000)*, IMPA, Rio de Janeiro.

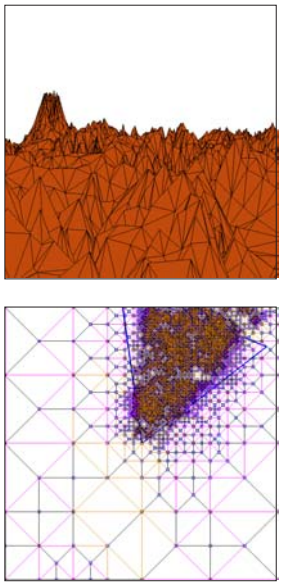
Velho, L. (2004). A dynamic adaptive mesh library based on stellar operators. *Journal of Graphics Tools*, 9(2):1–29.

Velho, L. and Gomes, J. (1999). Semi-regular 4-8 refinement and box spline surfaces. Preprint.

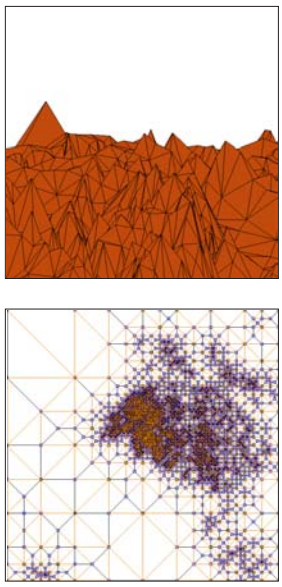
Velho, L. and Gomes, J. (2000). Variable resolution 4-k meshes: Concepts and applications. *Computer Graphics Forum*, 19(4):195–214.



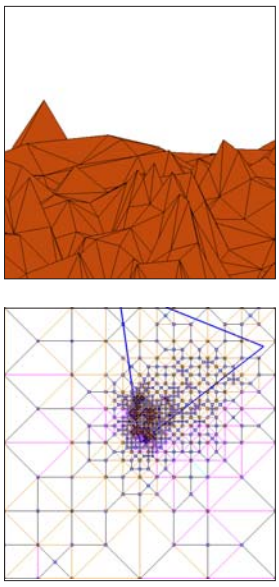
(a) View-independent



(b) Frustum culling using nested spheres



(c) View-dependent



(d) View-dependent and frustum culling

Figure 2. Terrain mesh results considering different error functions and adaptation criteria.

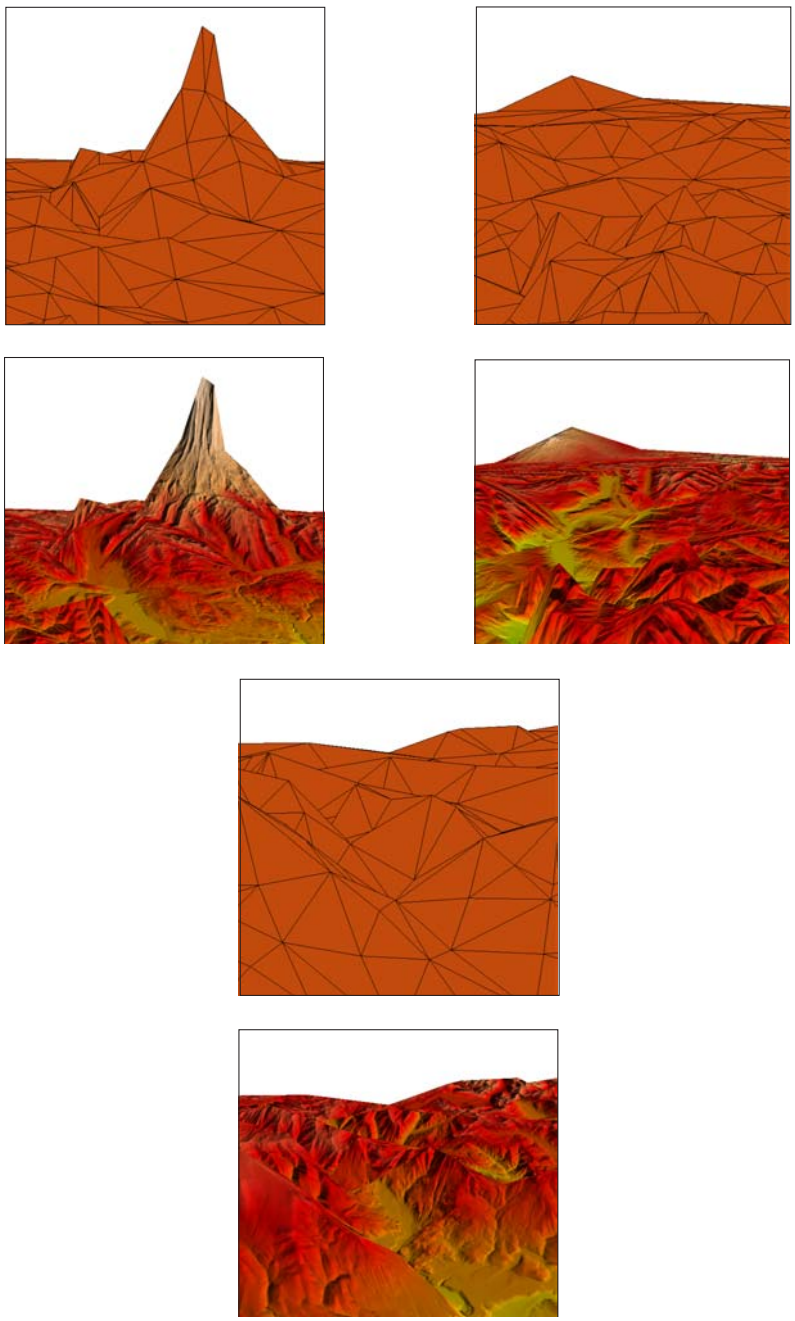


Figure 3. Terrain visualization from different point of view using a composition of geometric and view dependent errors and frustum culling.