

A hierarchical triangulation for distributed visualization of large scale terrain data

Rune Aasgaard
SINTEF Applied Mathematics
POBox 124
N-0314 Oslo
Norway
e-mail: rune.aasgaard@sintef.no

April 4, 2002

Key words: Visualization, Hierarchical data structures, Terrain models, Triangulations, Client-server

1 Introduction

Today large scale terrain data of high resolution are starting to be available. Data with a resolution of approximately 1km have been freely distributed for years, and 30-100m data can be purchased in most industrialized countries. Currently a 30m data set is under development by NASA for the regions between 60° north and 60° south: <http://www.jpl.nasa.gov/srtm/>.

In many occasions the terrain models will be used in an environment where the user has a small client machine, a PDA or an older PC. The machine may be connected to the network over a telephone line or a wireless communication channel of relatively narrow band with.

To provide interactive navigation of the terrain it is necessary to build a local view of the terrain database, a client terrain model. This is used for generating the graphical data structures that are needed by the rendering system. The client terrain model is continuously updated to have a sufficient accuracy for rendering. The approximation errors are projected into the image space and compared with a user specified tolerance number.

Areas outside of the view frustum are stored at the coarsest possible resolution to minimize the use of client resources. As the user moves around new data are requested, and data that have gone out of scope is dropped using a “least recently used” algorithm.

On the server the elevations are stored together with approximation error estimates. The error estimates indicates the contribution the vertex has on the accuracy of the terrain model.

2 The Virtual Globe

In this project we have developed a client-server application for visualizing global scale terrain using the principles described above. The client part is implemented as a Java Applet, and a demonstration, together with more documentation can be found at: <http://globe.sintef.no>. The project at an earlier stage was presented at ScanGIS'2001 [2].

2.1 Basic algorithms and data structures - Client

The client terrain model is here based on Binary Triangle Trees, a variation of the structure found in [4] or [5]. This structure uses a set of right isosceles triangles, connected by common edges to construct a continuous surface. The triangulation is refined by splitting a triangle in two, inserting a new point at the midpoint of the hypotenuse.

The neighbor triangle at the hypotenuse of the splitted triangle must also be split to preserve the continuity of the triangulation. If the neighbor meets with a short side against the hypotenuse the neighbor has to be split first. This forced neighbor splitting has to continue recursively until the triangulation is valid.

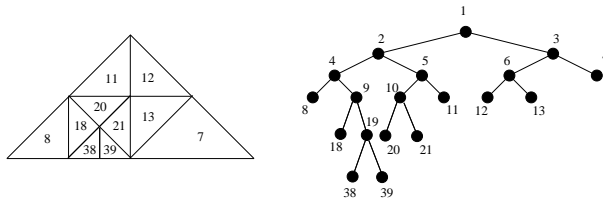


Figure 1: A triangulation and its binary triangle tree.

The forced splitting of triangles gives a partial ordering of vertices where the vertices that must be inserted before a given vertex are the ancestors of that vertex.

To handle the error estimates in relation to the view and camera parameters a variation of the sphere trees presented in [3] is used.

The greatest advantage of using BTTs for distributed LOD is the simplicity of the data structures. There is no transfer of complex object references or pointers as would be needed when using irregular triangulations. The data communication with the server is simply based on querying for elevations and elevation errors for points with given coordinates.

An alternative algorithm has been presented in [6] that makes the explicit storage of triangles unnecessary. This method also avoids the forced neighbor splitting, and simplifies the algorithms and data structures even more.

2.2 Basic algorithms and data structures - Server

The server stores elevation data and elevation error estimates. It has fast search methods for retrieving elevation data for a point given its position, and methods for integrating new data into the structure.

The elevations are stored in a variation of a quad tree. Like in the client triangle trees, the least detailed data are stored near the root and progressively more detail is added toward the leaves. Data that are near in both detail level and position is stored together to increase the probability of file cache hits when accessing data.

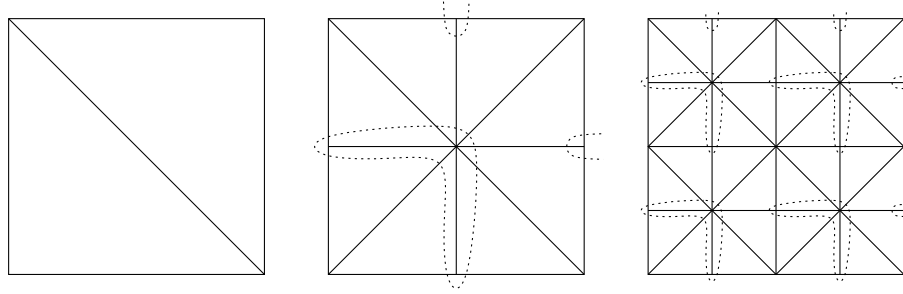


Figure 2: Triangulation related to the quad tree. Vertices added in each quad tree level are marked.

The server quad tree is constructed by a recursive procedure that follows the same split pattern as the binary triangle trees, where also approximation errors are computed and stored. Approximation errors are computed such that the error for a parent vertex is always larger than or equal to the error of its children. When the approximation error is found to be less than a given threshold the tree can be truncated to save storage space in flat areas.

2.3 Coordinate system

The triangulation and the quad tree works on a large virtual grid. For global applications this grid has to be projected to a sphere. To keep the grid cells nearly square in shape, while still having a finite grid a new projection has been developed. It will be presented on SDH 2002 [1].

In the current implementation the grid has the dimensions $0 \leq x < 2^{30}$, $-2^{30} \leq y \leq 2^{30}$. This gives a maximum resolution of approximately 4cm at the equator, better at higher latitudes.

The x value uses modular arithmetics and wraps around from $2^{30} - 1$ to 0 when the x value is incremented with 1, and in the triangulation the “rightmost” triangles are connected to the “leftmost” ones to create a cylinder topology. There are however discontinuities at the poles that can usually be hidden by assuring that the north and south rim of the grid has identical elevations. The

coarse base triangulation is based on a 4×8 grid to avoid ambiguities in the computation of midpoints.

3 Current status and ongoing work

The current java implementation of the “Virtual Globe” is not sufficiently efficient. It runs well on a reasonably modern PC with a fast network connection, but must be optimized more for slow connections and small hand-held computers. The ultimate goal is to use interactive distributed terrain visualization in mobile tourist applications.

Several steps are taken to accomplish this:

- Optimizing the client code.
- Establishing mirror servers in other parts of the world.
- Using persistent client side disk cache where possible.
- Adding a HTTP based communication protocol that also works through firewalls.

All of those tasks are worked on, and some may be completed before the workshop.

This work has been supported by the Norwegian Research Council through the “DynaMap” project and the European Union through the “TellMaris” project.

References

- [1] Aasgaard R: Projecting a regular grid onto a sphere or ellipsoid. *Spatial Data Handling 2002*, 2002.
- [2] Aasgaard R and Sevaldrud T: Distributed Handling of Level of Detail Surfaces with Binary Triangle Trees. *ScanGIS'2001*, 2001, <http://www.nlh.no/conf/scangis2001/papers/27.pdf>
- [3] Blow J: Terrain Rendering at High Levels of Detail. in Proc. *Game Developers Conference 2000*, 2000.
- [4] Duchaineau M, Wolinsky M, Sigeti DE, Miller MC, Aldrich C and Mineev-Weinstein MB: ROAMing Terrain: Real-Time Optimally Adapting Meshes, in Proc. *Visualization '97*, 1997.
- [5] Lindstrom P, Koller D, Hodges LF, Ribarsky W, Faust N and Turner G: Real-Time, Continuous Level of Detail Rendering of Height Fields, *Computer Graphics (Proc. SIGGRAPH '96)*, 1996.
- [6] Lindstrom P, Pascucci V: Visualization of Large Terrains Made Easy *Proceedings of IEEE Visualization 2001*, October 2001, pp. 363-370, 574.