Large Scale Constraint Delaunay Triangulation for Virtual Globe Rendering

M. Christen and S. Nebiker

Abstract A technique to create a Delaunay triangulation for terrain visualization on a virtual globe is presented. This method can be used to process large scale elevation datasets with billions of points by using little RAM during data processing. All data is being transformed to a global spatial reference system. If grid based elevation data is used as input, a reduced TIN can be calculated. Furthermore, a level of detail approach for large scale out-of-core spherical terrain rendering for virtual globes is presented using the previously created TIN.

1 Introduction

Terrain Rendering with very large and dense height data sets have become very popular due to the progression of geospatial imaging sensors such as airborne LiDAR (Fowler et al. 1997; Shan and Toth 2009). Terrain point densities from such sensors can be in the range of one or several points per square meter leading to massive data sets when applied to entire states or countries. Even for small countries, such a LiDAR data set could easily comprise of several hundred billions of points or several TB of data respectively. Despite the wide-spread use of TIN-based elevation models in the GIS community, there are hardly any software solutions capable of efficiently processing such large data sets on the one hand of and also supporting the generation of levels of detail on the other. However, both capabilities are required to exploit such large TIN-based data sets in scalable and interactive 3D geoinformation environments. Virtual globes have a high impact on geographical 3D information systems and reality based games. They were mostly based on grid-based elevation data but will increasingly incorporate high-density elevation data sets in the multi-TB range. This paper shows an approach of how

M. Christen (✉) and S. Nebiker
Institute of Geomatics Engineering, University of Applied Sciences Northwestern Switzerland, Gründenstr. 40, 4132 Muttenz, Switzerland
e-mail: martin.christen@fhnw.ch, stephan.nebiker@fhnw.ch

T.H. Kolbe et al. (eds.), *Advances in 3D Geo-Information Sciences*,
Lecture Notes in Geoinformation and Cartography,
DOI 10.1007/978-3-642-12670-3_4,

very large irregulary spaced high-density elevation datasets can efficiently be pre-processed for ellipsoidal rendering on a virtual globe.

2 Previous Work

A virtual globe has been implemented at the University of Applied Sciences Northwestern Switzerland. This virtual globe is called i3D. Virtual globes can be implemented (Nebiker et al. 2007) using grid based geometry for the terrain. However, using regular grids has several drawbacks. For one the grid is limited to contain points aligned to the grid only: break-lines and spot heights are not supported since they are generally not grid-aligned. Another drawback is that when creating a polygonal representation from a grid representation many coplanar polygonneighbors may result, especially in flat areas like lakes, leading to expensive redundancy and unneeded geometry, which leads to wasted memory transfer between the CPU and GPU. Furthermore, if grid points are transformed from a local to a global spatial reference system the alignment of the grid changes to a new grid layout which requires height interpolations and thus leads to quality loss.

3 Related Work

Numerous interactive terrain rendering methods have been proposed. A thorough survey on different approaches was recently made by Pajarola and Gobbetti (2007).

Basically there are hierarchical level of detail approches using regular grids and others using irregular data sets (Hoppe et al. 1998; Pajarola et al. 2002).

Lately there was more focus on GPU optimized techniques (Livny et al. 2009) and related topics like data compression between the CPU and GPU (Dick et al. 2009; Lindstrom et al. 2010).

Most proposed works assume a planar reference – restricted to flat earth terrain rendering. Only few literature is considering spherical or even ellipsoidal planetary rendering or data processing, for example references (Gerstner et al. 1999; Szalay et al. 2007; Zhou et al. 2008).

For data preprocessing, a streaming Delaunay computation of very large height data sets was introduced by Isenburg et al. (2006). Their approach calculates Delaunay triangulations by using the spatial coherence of the dataset itself – without previously sorting the data. However, their approach does not consider storing the resulting triangulation in a spatial data structure suitable for the virtual globe and applying further processing steps to the resulting TIN, like thinning out and calculating level of detail.

Furthermore, an alternative way for visualizing large amount of LIDAR data in Google Earth has been presented by Isenburg et al. (2009). Their approach creates rasterized contour lines to visualize the data and not the actual geometry.

The Google Earth API (Google: Google Earth API Developer's Guide) currently doesn't have a mechanism to stream custom elevation data.

4 Virtual Globes

4.1 Overview of Virtual Globes

Virtual globes, sometimes also referred to as 3D geobrowsers, consist of virtual 3D environments capable of streaming and interactively displaying large amounts of geo-referenced spatial contents over the Internet (Nebiker et al. 2010). With the availability of the commercial products Google Earth (Google: Google Earth API Developer's Guide) and Microsoft: Bing Maps 3d virtual globes gained enormously in popularity. A virtual globe stores its data on servers that can be accessed by a client software. This software can either be a stand-alone executable file (Google Earth), or running from a browser plugin (Bing Maps, Google Earth).

There is little reliable information available on how these commercial products create their large scale elevation models. This may be one of the reasons, why adding large custom-built elevation models is not possible with these globes.

4.2 Preprocessing Data for Virtual Globes

A virtual globe can have several data categories like image data, elevation data, points of interest, vector data, and 3D objects. Before streaming over the Internet this data must be preprocessed. This preprocessing usually comprises a transformation from a local to a global reference system, creation of pyramid layers or level of detail, tiling of the data, and optionally compression and encryption of the data.

4.3 Virtual Globe Tile Systems

Bing Maps, Google Earth, and i3D tiles are indexed using quadtree keys (quadkeys). Each quadkey number identifies a single tile at a single zoom level (see Fig. 1).

Typically, the Mercator projection is used to map image and elevation data to a square area. The Mercator projection is mainly used to minimize distortions in processed images and elevation data. The meridians of the Mercator projection are vertical parallel equally spaced lines, cut at right angles by horizontal straight parallels which are increasingly spaced toward each pole so that conformality is preserved (Snyder 1987). The maximum latitude is chosen so that the resulting map fits into a square. For the spherical Mercator projection this maximum latitude is

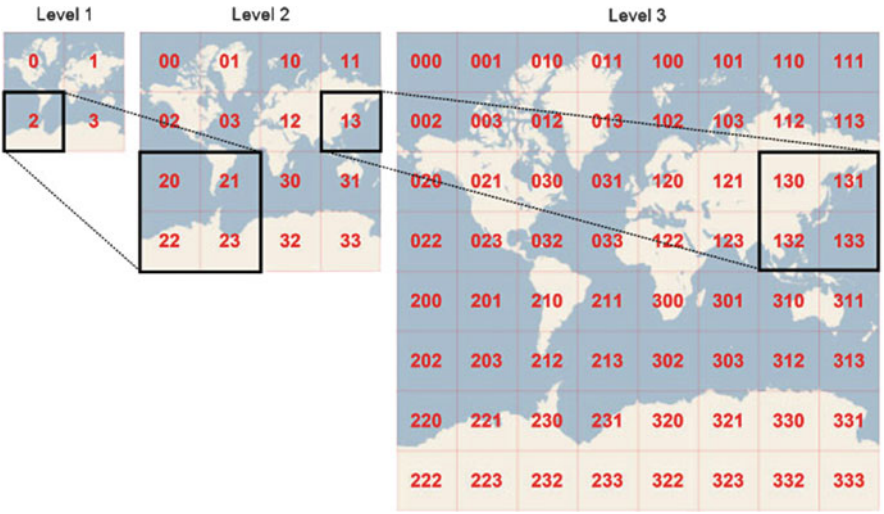

Fig. 1 Bing maps tile system (Microsoft: Bing Maps Tile System)

approximately 85.05° and for the ellipsoidal Mercator projection it is approximately 85.08°. The projection is normalized to values in the range (−1,−1) to (1,1) to ensure increased numerical stability during the triangulation and thinning process. Bing Maps 3D uses the spherical Mercator projection and i3D supports both the spherical and ellipsoidal Mercator projection. Such an ellipsoidal geodetic reference model is required, in order to minimise geometric transformation errors and to enable position accuracies within the Virtual Globe at the sub-meter level. Listing 1 shows the implementation used in i3D.

Points are transformed from the Mercator projection to WGS84 by using the respective algorithm described in Snyder (1987).

```
1 void EllipsoidToMercator (const double lngRad, const
double latRad, double& out_x, double& out_y, const double e)
2 {
3 const double a = 1.0 / M_PI;
4 const double lngRad0 = 0;
5
6 if (e == 0.0) // spherical case
7 {
8 out_x = a* (lngRad-lngRad0);
9 out_y = log(tan(M_PI/4.0 + latRad/2));
10 }
11 else // ellipsoidal case, first eccentricity not 0.
12 {
13 out_x = a* (lngRad-lngRad0);
```

```
  14 out_y = a*log (tan(M_PI/4.0+latRad/2.0)*pow((1.0-e*
sin( latRad))/(1.0 + e* sin (latRad)),0.5*e));
  15}
  16}
```

Listing 1 Transformation to normalized Mercator coordinates

5 Creating Large TIN for Virtual Globes

In the design of elevation support for a Virtual Globe a number of requirements must be taken into account, many of which are distinctly different from those applicable to DEM support in standard GIS environments. These DEM requirements in Virtual Globes include: the support for highly accurate regional to global elevation models with a preservation of the original precision at a global scale; the possibility to add spot heights and breaklines – either during preprocessing or at runtime; the adaptive data reduction or data thinning in order to optimise storage space, transmission performance and memory utilisation in the rendering process; the support for level of detail and view-dependent multiple resolutions; the need for highly efficient processing of very large terrain models supporting parallelisation approaches; a streamable delivery and rendering of terrain data, and finally, the capability of merging and displaying multiple DEMs, e.g., a global model and a high-resolution local model, at run-time. In this section we document an approach addressing these specific requirements.

5.1 Choosing a Data Structure for Large Scale Triangulation

There are many possible data structures for representing triangulations. Thus, a triangulation data structure should be chosen in view of the needs and requirements of the actual application (Hjelle and Daehlen 2006). Popular data structures for triangulations are: triangle-based data structure with neighbors, vertex-based data structure with neighbors, half-edge data structure (Weiler 1985), and the quad-edge data structure (Guibas et al. 1983).

For our large scale implementation of the Delaunay triangulation a triangle-based data structure is used. One of the reasons for that is that the triangle neighbor must support a mechanism to temporarily remove neighbor triangles from memory to simplify our large scale triangulation and to calculate level of detail which is easier to implement in a triangle based structure.

The main characteristic of a triangle-based data structure is that edges are not stored directly – only vertices and triangles which are always stored in counter-clockwise orientation as we can see in Listings 2 and 3.

The triangle structure is defined by using two structures, one for the vertex (also called elevation point) and a second structure for the triangle.

```
1 struct Vertex
2 {
3 double x, y; // position in mercator projection
4 double elv; // elevation above WGS84 ellipsoid [m]
5 double weight; // weight/importance of this point
6 };
```

Listing 2 Definition of an elevation point

All values of the vertex structure are held as double precision floating point to support precision in the millimeter range. Besides the position parameters a weight parameter is used which can be used to define the importance of the point. This is useful in the preservation of characteristic mountain peaks when calculating the level of detail.

```
1 struct Triangle
2 {
3 Vertex* pVertex0 ;
4 Vertex* pVertex1 ;
5 Vertex* pVertex2 ;
6 Triangle* pTriangle0 ;
7 Triangle* pTriangle1 ;
8 Triangle* pTriangle2 ;
9 } ;
```

Listing 3 Definition of a triangle

The triangle structure holds pointers to vertices and neighbor triangles as shown in Fig. 2. Because vertex-pointers and triangle-pointers are shared among different triangles a reference count mechanism is implemented.

Because the triangulation must be large scale, it is necessary to remove triangles in a special way by deleting them from memory without affecting the Delaunay condition. This can be done by deleting unused triangles from memory and by setting the appropriate triangle neighbor pointers to 0. This allows creating disconnected areas with islands in the triangle structure as shown in Fig. 3.

Furthermore for thinning out data a mechanism must be available for removing points from the triangulation, for example by using the removal algorithm described in Mostafavi et al. (2003).

The Delaunay triangulation itself is created using an incremental approach. In summary, the supported basic operations of the triangulation must be:

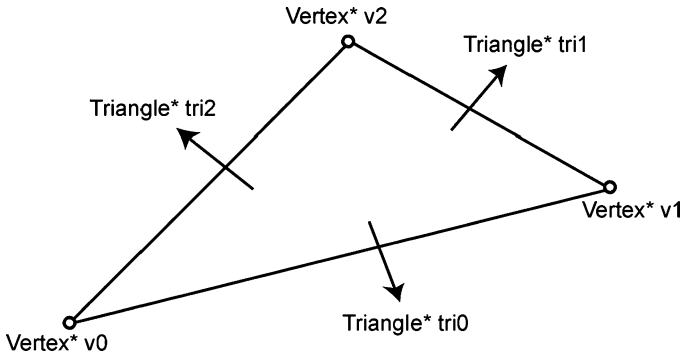

Fig. 2 Triangle data structure

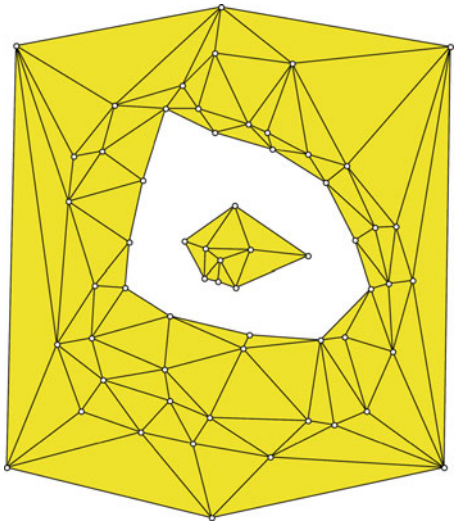

Fig. 3 Disconnected area with island

- Insert a new point into triangulation (for incremental construction)
- Remove a triangle from memory (for large scale support)
- Remove a point from triangulation (for thinning out data)

5.2 Incremental Delaunay Construction

Incremental Delaunay trigulation has been presented by Guibas and Stolfi (1985). For reasons of clarity it is described briefly below.

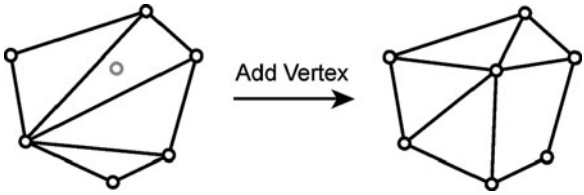

Fig. 4 Adding a new vertex to triangulation

The construction of an incremental Delaunay triangulation consists of adding new vertices to an already existing triangulation. When adding a new vertex to the triangulation, there are some cases to be distinguished (Fig. 4):

1. The new vertex is inside an existing triangle. In this case three new triangles are created.
2. The new vertex is on an edge of an existing triangle. In this case the triangles containing this edge are removed and four new triangles are created.
3. The new vertex has the same position as an existing vertex in the triangulation. In this case the new vertex is rejected.
4. The new vertex is outside a previously defined boundary. In this case the vertex is rejected.

To maintain the Delaunay condition when inserting new points, invalid edges must be flipped, so that the circumcircle of a triangle never contains another point of the triangulation (Fig. 5).

5.3 Numerical Stability in a Global Spatial Reference System

Because the triangulation is done in a global spatial reference system, numerical stability is an important issue, as triangles may become very small in relation to the global system. When inserting a new vertex into the triangulation, cases 2 and 3 of the incremental Delaunay construction described above bear numerical limitations. If all points of the triangles are almost collinear ("skinny triangle"), numerical problems will occur when using standard floating point arithmetic. Exact computation of the incircle and orientation predicates makes the algorithm robust, but slow. More speed can be gained by using arithmetic filtering approaches (Shewchuk 1996; Devillers et al. 2003) (Fig. 5).

5.4 Large Scale Triangulation

Our algorithm is divided into three passes:

1. Geodetic transformation
2. Spatial subdivision
3. Triangulation and cell split

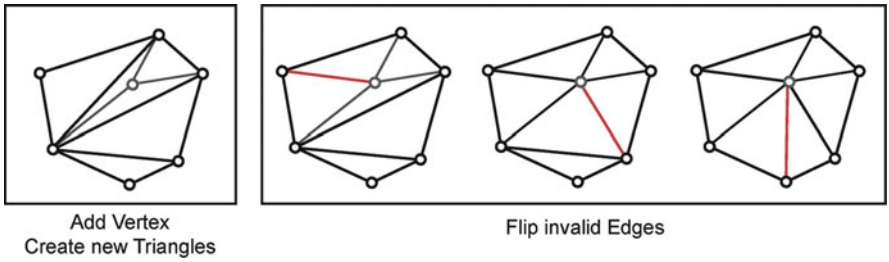

Fig. 5 Flipping edges to satisfy the Delaunay condition

First Pass: The first pass transforms points from the source spatial reference system to a normalized Mercator projection using any custom rotational ellipsoid or sphere as described in Listing 1. During this pass a new file containing all transformed points in Mercator projection is created. In addition, the axis aligned bounding box (in Mercator projection) of the dataset is calculated.

In summary, at the end of the first pass, the total number of points and the axis aligned bounding box is known, and all transformed points are stored in a binary file.

Second Pass: The second pass creates a spatial subdivision of the points. This spatial subdivision is the same global quadtree of the Mercator projection which is later used for out-of-core rendering of the terrain. All points are stored in tiles of the lowest level of detail of this dataset. The maximum number of points per tile can be specified and according to that number the lowest level of detail is calculated. At the end of the second pass the original dataset is spatially subdivided and for each tile a file exists. The filename of the tile is generated using its quad-code – a unique identifier for each cell in the quadtree (Fig. 6).

Third Pass: The third pass actually calculates the triangulation. By traversing the quadtree structure in a Morton order(Morton 1966), the triangulation is calculated step by step.

When creating tiles, new points are added to the triangulation, the border points of the tiles (see Fig. 7). These points are stored in an edge file. “Edge points” are usually shared by two neighbor tiles and “corner points” are shared by up to four neighbor tiles. Every tile has at least four corner points. If a corner is not part of the triangulation, a point with elevation 0 will be added and marked as “no data value”, otherwise points are calculated by doing a linear interpolation of the elevation points at the corresponding position in the triangle containing the point. These points are treated as a constraint in the Delaunay triangulation, so that a rectangular tile border exists for all tiles.

Tiles are stored by saving corner points, divided into north edge points, east edge points, south edge points, west edge points, and interior points separately. This way the Delaunay triangulation can be recreated for every tile and therefore additional points or break-lines may be added at a later time or even interactively during visualization.

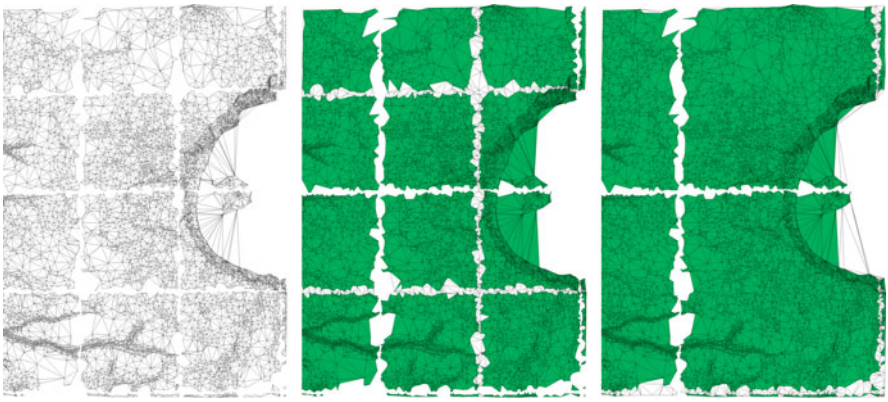

Fig. 6 Calculating triangulation: output of processing three levels. (Only *white triangles* of max. four tiles are kept in memory)

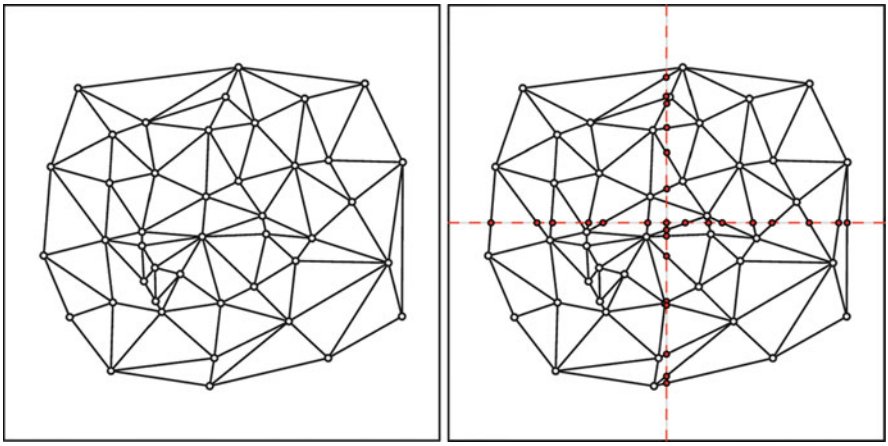

Fig. 7 Calculating "tile edges" and "tile corner points"

5.5 Creating Level of Detail Tiles

Once all tiles are stored, the level of detail for all remaining pyramid layers can be calculated. The maximum number of points per tile must be maintained through all levels of detail. The algorithm used is an adapted version of the mesh simplification method presented by Heller (1990). For each level of detail tile all points of the four parent tiles are loaded and merged together in a new tile. The interior of the new tile and the corner and edge points of the parent tiles are disregarded. The new tile is being triangulated and thinned out by using the algorithm described by Lee (1989): for each point in the new triangulation an error value is calculated. This error value

holds the elevation difference which results when the point is removed from the triangulation. Points with small error are removed first. This procedure is repeated until the number of points in the tile is smaller than or equal to the maximum allowed number of points per tile. If the error value is still smaller than a previously defined epsilon value the points may even be thinned out more, which allows thinning out flat areas like lakes in an early stage of the level of detail.

6 Rendering the Virtual Globe

Because visualization may consist of terabytes of orthophoto and elevation data, out-of-core rendering with a level of detail approach must be used. Data can be streamed over the Internet, a local network or a local hard drive. This is done by using download threads as shown in Fig. 8. Using a least recently used caching approach (LRU), the most recently requested data remains in the most expensive and fastest memory layer.

For the level of detail approach a quadtree is used. The quadtree can be mapped to the WGS84 ellipsoid and an error metric adapted to ellipsoids decides which resolution is suitable for the current view. Depending on the error per pixel, a new resolution is requested from another data storage layer. The data itself is stored in tiles based on a quadtree.

Tiles with neighbors of a different level of detail must be stitched together to avoid visualization artifacts. This usually involves updating the geometry of the tile with higher resolution to match the neighbor resolution, or using a curtaining method to hide stitching problems in a more efficient manner as it is not necessary to figure out the resolution of neighbors and to avoid retriangulations during visualisation.

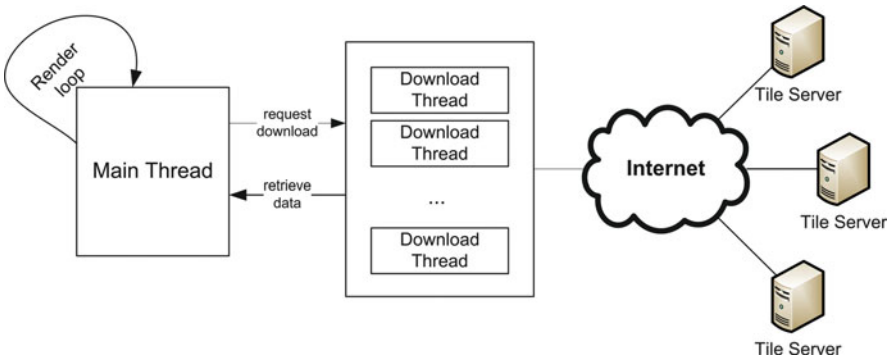

Fig. 8 Elevation tile data is being downloaded while the rendering thread remains running at constant speed

7 Results

The presented approach for large scale constraint Delauney triangulation has been implemented in our own i3D virtual globe technology.

7.1 Streaming Data

The propoposed algorithm produces tiled data that can be streamed from a harddisk, local area network, or from the Internet. Depending on the number of cores of the client machine, up to eight download threads are created to retrieve the data while the main thread is responsible for rendering data available in best resolution as shown in Fig. 8.

7.2 Quality

The error metric allows to change a quality parameter in real-time. This can be used to force level of detail to a specific setting, as shown in Fig. 9.

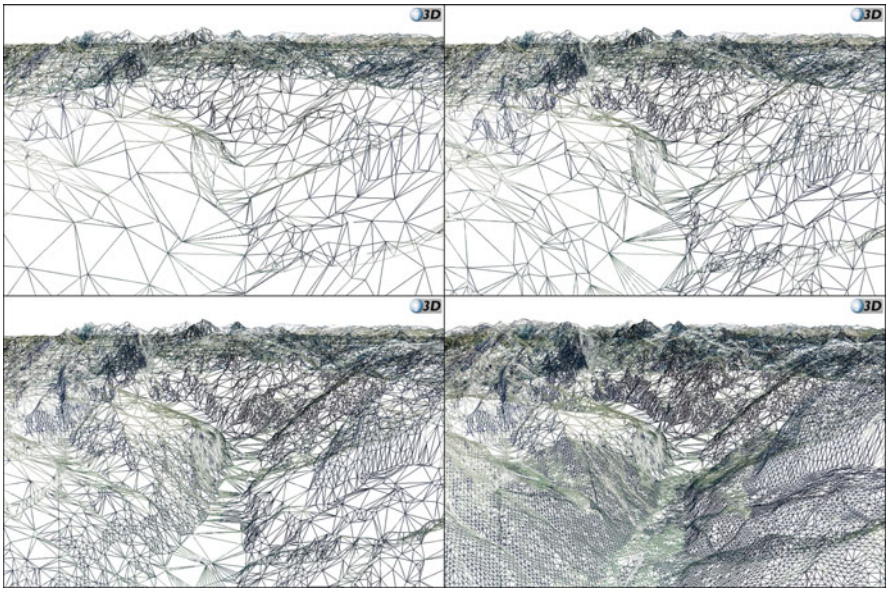

Fig. 9 Interactive change of quality parameters. Using swisstopo DHM25 dataset. Base data © swisstopo (JA100071)

Computers or mobile devices with lower memory or low-end graphics hardware can be set to a lower quality while maintaining a certain quality. Lower quality is also preferred in a slower network as it results in faster streaming.

7.3 Combining Image Data

The produced elevation tiles can be combined with image data, the result is shown in Fig. 10.

Image data is being tiled using the same quadtree tiling structure as the elevation data. Image and elevation tiles are not stored together to be independent. It is possible to exchange image tiles without updating elevation tiles.

In Fig. 11 elevation tiles and image tiles are blended together to see the resolution difference between image and elevation data.

8 Conclusion

An approach for creating large scale elevation data processing has been introduced. With this approach it is possible to process global and local elevation datasets for visualization on a virtual globe. It permits the efficient processing of very large scale regularly and irregularly spaced terrain data sets on standard computer

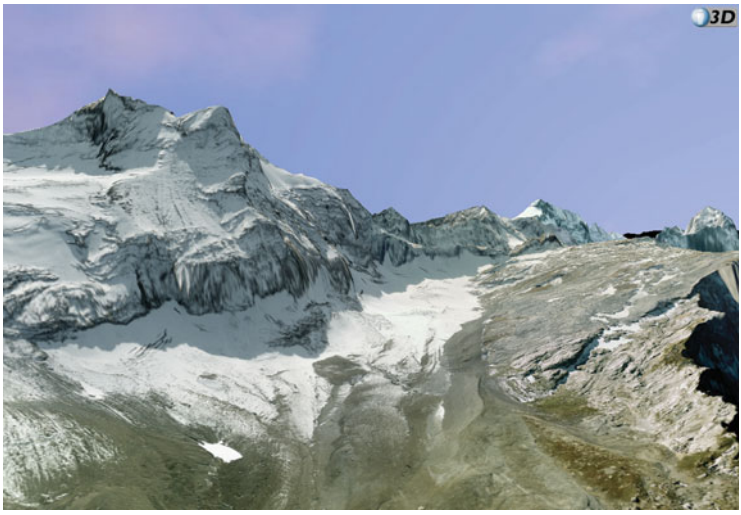

Fig. 10 Rendering a TIN (original DEM point spacing of 25 m) on the virtual globe with and areal imagery with 50 cm per pixel resolution. Using SWISSIMAGE and DHM25 datasets. Base data © swisstopo (JA100071)

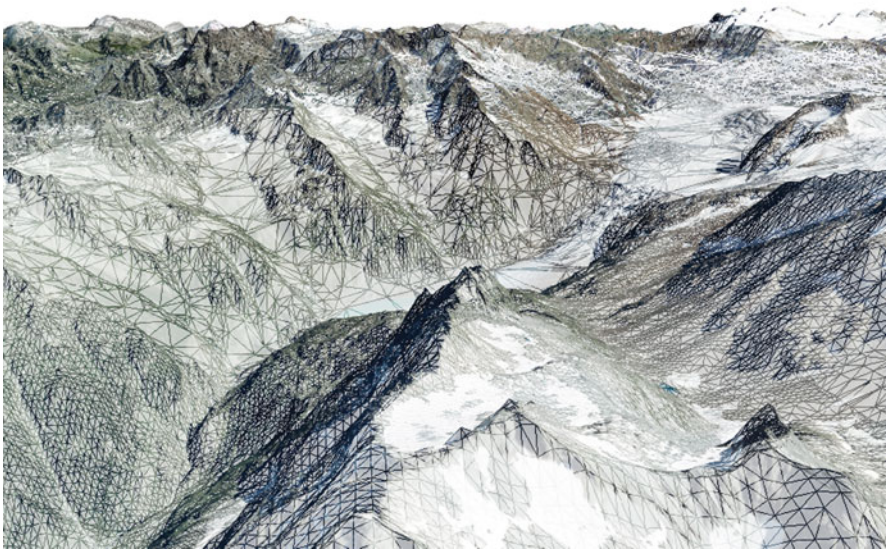

Fig. 11 TIN wireframe and image data blended together. Using SWISSIMAGE and DHM25 datasets. Base data © swisstopo (JA100071)

hardware. The presented triangulation approach is well suited for a future parallelisation of the triangulation of national to continental high-resolution elevation data sets. The presented TIN-based approach offers a number of advantages over grid-based terrain rendering in earlier versions of virtual globes, namely a significantly improved terrain representation with a simultaneous dramatic reduction in data size and streaming performance. The approach also supports the on-the-fly integration of spot heights and break-lines at runtime. This on-the-fly integration of break-lines, including special geospatial features such as Terrain Intersection Curves (TIS) of the CityGML standard (2010), will be a prerequisite for enabling future applications in Virtual Globes requiring more accurate and higher fidelity representations of urban environments (Nebiker et al. 2010).

References

Devillers, O., Devillers, Pion, S., Pion, S., Prisme, P.: Efficient exact geometric predicates for Delaunay triangulations. In: Proceedings of the 5th Workshop Algorithm Engineering and Experiments. pp. 37–44. Baltimore (2003)

Dick, C., Schneider, J., Westermann, R.: Efficient geometry compression for GPU- based decoding in realtime terrain rendering. Comp. Graph. Forum 28(1), 67–83 (2009)

Fowler, R.E., Samberg, A., Flood, M., Greaves, T.J.: Modeling mobile terrestrial LiDAR to vector based models. In: Maune, D. F. (ed.) Digital Elevation Model Technologies and Applications: The DEM Users Manual, chap. Topographic and Terrestrial Lidar. pp. 199–252. American Society of Photogrammetry and Remote Sensing, Bethesda (1997)
Gerstner, T.: Multiresolution visualization and compression of global topographic data. Tech. rep., GeoInformatica (1999)
Google: Earth, http://earth.google.com
Google: Google Earth API Developer's Guide, http://code.google.com/apis/earth/documentation/
Guibas, L.J., Stolfi, J.: Primitives for the manipulation of general subdivisions and the computation of voronoi diagrams. In: STOC' 83: Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing. pp. 221–234. ACM, New York (1983)
Guibas, L.J., Stolfi, J.: Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. ACM Trans. Graph. 4, 74–123 (1985)
Heller, M.: Triangulation algorithms for adaptive terrain modeling. In: Proceedings of the 4th International Symposium on Spatial Data Handling. pp. 163–174. Zurich, Switzerland (1990)
Hjelle, O., Daehlen, M.: Triangulations and Applications (Mathematics and Visualization). Springer-Verlag New York, Secaucus, NJ (2006)
Hoppe, H.: Smooth view-dependent level-of-detail control and its application to terrain rendering. In: VIS '98: Proceedings of the Conference on Visualization '98. pp. 35–42. IEEE Computer Society Press, Los Alamitos (1998)
Isenburg, M., Liu, Y., Shewchuk, J., Snoeyink, J.: Streaming computation of Delaunay triangulations. ACM Trans. Graph. 25(3), 1049–1056 (2006)
Isenburg, M., Shewchuk, J.: Visualizing LIDAR in Google Earth. In: Proceedings of the 17th International Conference on Geoinformatics. Fairfax (2009)
Lee, J.: A drop heuristic conversion method for extracting irregular networks for digital elevation models. In: Proceedings of the GIS/LIS '89. pp. 30–39. Orlando (1989)
Lindstrom, P., Cohen, J.D.: On-the-fly decompression and rendering of multiresolution terrain. In: I3D '10: Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. pp. 65–73. ACM, New York (2010)
Livny, Y., Kogan, Z., El-Sana, J.: Seamless patches for GPU-based terrain rendering. Vis. Comput. 25(3), 197–208 (2009)
Microsoft: Bing Maps 3d, http://www.bing.com/maps
Microsoft: Bing Maps Tile System, http://msdn.microsoft.com/en-us/library/bb259689.aspx
Morton, G.: A computer oriented geodetic data base and a new technique in file sequencing. Tech. Rep. IBM Ltd., Ottawa, Ontario, Canada (1966)
Mostafavi, M.A., Gold, C., Dakowicz, M.: Delete and insert operations in Voronoi/Delaunay methods and applications. Comput. Geosci. 29(4), 523–530 (2003)
Nebiker, S., Christen, M., Eugster, H., Flückiger, K., Stierli, C.: Integrating mobile geo sensors into collaborative virtual globes – design and implementation issues. Paper presented at the Mobile Mapping Technologies Symposium MMT 2007, Padua (2007)
Nebiker, S., Bleisch, S., Christen, M.: Rich point clouds in virtual globes a new paradigm in city modeling? Computers, Environment and Urban Systems (June 2010), http://dx.doi.org/10.1016/j.compenvurbsys.2010.05.002
Open Geospatial Consortium, Inc: OpenGIS® city geography markup language (CityGML) – encoding standard (ogc 08-007r1). (p. 218): Open Geospatial Consortium Inc. (2010)
Pajarola, R., Gobbetti, E.: Survey of semi-regular multiresolution models for interactive terrain rendering. Vis. Comput. 23(8), 583–605 (2007)
Pajarola, R., Antonijuan, M., Lario, R.: Quadtin: quadtree based triangulated irregular networks. In: VIS '02: Proceedings of the Conference on Visualization '02. pp. 395–402. IEEE Computer Society, Washington, DC (2002)
Shan, J., Toth, C.: Topographic laser ranging and scanning. CRC Press, Boca Raton (2009)
Shewchuk, J.R.: Adaptive precision floating-point arithmetic and fast robust geometric predicates. Discrete Comput. Geometry 18, 305–363 (1996)

Snyder, J.P.: Map Projections: A Working Manual. U.S. Geological Survey Professional Paper 1395, U.S. Geological Survey, http://pubs.er.usgs.gov/usgspubs/pp/pp1395 (1987)
Szalay, A.S., Gray, J., Fekete, G., Kunszt, P.Z., Kukol, P., Thakar, A.: Indexing the sphere with the hierarchical triangular mesh. CoRR abs/cs/0701164 (2007)
Weiler, K.: Edge-based data structures for solid modeling in curved-surface environments. IEEE Comput. Graph. Appl. 5(1), 21–40 (1985)
Zhou, Q., Lees, B., Tang, G.A.: Lecture Notes in Geoinformation and Cartography, chap. A Seamless and Adaptive LOD Model of the Global Terrain Based on the QTM. pp. 85–103. Springer Berlin Heidelberg, New York (2008)