Terrain Representation using Right-Triangulated Irregular Networks

Suddha Basu
UNC Chapel Hill
skbasu@cs.unc.edu

Jack Snoeyink
UNC Chapel Hill
snoeyink@cs.unc.edu

Abstract

A quadtree-based triangulation, sometimes called an RTIN, is a common hierarchical representation of terrain for 3D visualization. In this paper we focus on the memory/error bound trade-offs for using RTINs with planar and quadratic surface patches over triangles.

1 Introduction

Terrain data in Geographic Information Systems is often sampled on raster grids, but displayed as a TIN (Triangulated Irregular Network), which is a collection of triangles. A Right-Triangulated Irregular Network [5] is a special TIN using isosceles right-angled triangles, formed as a binary tree by splitting along the perpendicular bisector to the hypotenuse. Both RTINs and general TINs are irregular in structure, allowing non-uniform sampling of the terrain. Though an RTIN typically has more triangles for a given error-bound than other TIN variants, for gridded data it has advantages. The most important is that it is hierarchically structured like a quadtree [12], so it can avoid storing edge and neighbor information and derive this information from the hierarchy. In this paper, we explore the memory requirements for TIN vs RTIN representations, and compare using planar or curved triangles.

There has been much work in approximation surfaces, particularly on multi-resolution representation of spatial data in quadtree-like structures. RTINs have been independently developed in several large-scale terrain visualization systems [9, 4, 3, 8, 10]. Evans et al. [5], in particular, try to minimize memory and give code for computing neighbors in constant time. Triangular quadtrees based on equilateral triangles [6, 7] are essentially equivalent—each triangle is broken into four and triangles can be identified by integer `id`s, and neighbor indices can be computed in constant time. Other representations for curved triangles come from Akima [1, 2], who defined quintic bivariate polynomials to represent height fields over triangles, and Preusser [11], who defined C^2 bivariate Hermite polynomials over triangles.

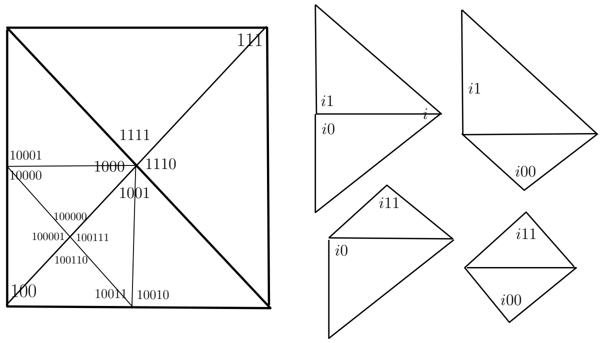

Figure 2: Portion of an RTIN and some neighbor cases. Triangle ids are at their right angles.

2 RTIN Representation

We save memory in the RTIN in a couple of ways. First, we need not store x and y coordinates or pointers to neighbor triangles because we can recover them from the hierarchy. We represent the mesh as a list of triangles, and a hash table with references to these triangles. Each triangle has a binary `id` that identifies whether the left or right triangle is chosen at each split. Second, since we assume we can obtain neighbors, we store only the north-west vertex with each triangle, and obtain the other vertices from the neighbors.

Since we rely on neighbor computation, we want it to be efficient. Lee et al. [6, 7] and Evans et al [5] give methods to compute neighbors of triangles from their indices in constant time, drawing on Schrack's work on linear quadtree indexing [13]. In the paper we show that it is sufficient to be able to include the same level neighbor across the hypotenuse, which we show can be done with $O(1)$ AND $(\cdot)$, exclusive-OR $(\oplus)$, multiplication by 2, and addition $(+)$ operations: HNEIGHBOR(T)

```
1. Assume $M = (10)^*$ is precomputed.
2. $D = (T \oplus 2T) \cdot M$.
3. $R = D + 2D$.
4. $C = R \oplus (R + 11)$.
5. Return $T \oplus C$.
```

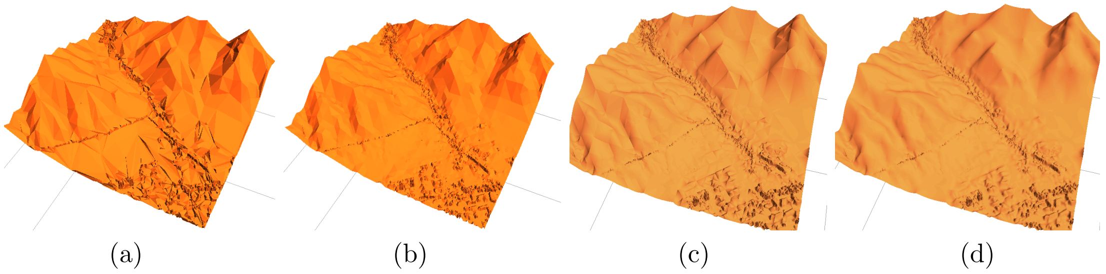

Figure 1: Surface representation using (a) regular TIN (9,909 triangles), (b) RTIN with piecewise linear triangles (40,759 triangles), (c,d) RTIN with piecewise quadratic triangles (10,794 triangles; (d) has smoothed normals).

3 Quadratic surface patches

In a triangulated terrain, each triangle typically approximates a region covering several grid points. We were interested in knowing whether, for a given error bound, the overhead of storing coefficients for representing quadratic patches instead of linear patches would be offset by the reduction in the number of triangles. A quadratic patch over a triangle could be represented by the standard basis:

$$z = c_0x^2 + c_1y^2 + c_2xy + c_3x + c_4y + c_5, \qquad (1)$$

but we observe some benefits of using a basis from barycentric coordinates, which is up to constants, the multivariate Bernstein basis.

The six coefficients can be determined in several ways. We use two: (1) by fixing vertices and tangents at edge midpoints and (2) using a best-fitting polynomial to the N sample points that lie within the triangle. Fixing vertices and edge-midpoints does not attempt to fit any of the interior points to the polynomial surface, but is a simple way to ensure that curves match along triangle edges. A best-fitting polynomial should fit the interior points to the surface more tightly, with the overhead of having to minimize a linear system with 6 variables and N constraints, but causes cracking between triangles. Fixing vertices and using best fit for edges, easily done in barycentric coordinates, seems to work well.

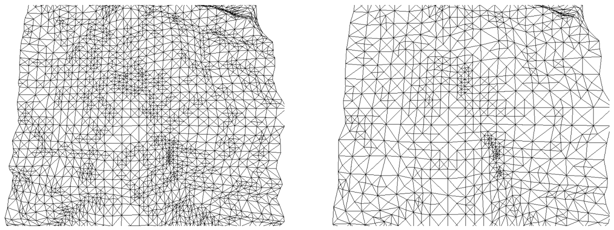

Figure 3: Meshes for planar (3,837 triangles) and quadratic (1,481 triangles) representations that have similar error.

SNo.	No. Triangles	Memory	Total Error
		in KB	$\times 10^6$ m
(a)	3,535	27.62	6.567
(b)	4,657	36.38	5.990
(c)	5,688	44.44	5.471
(d)	7,384	57.69	4.929
(e)	9,766	76.11	4.554
(f)	12,972	101.34	4.013
(g)	17,939	140.15	3.626
(h)	40,759	318.43	2.825
(i)	91,485	714.73	2.077

Table 1: Error Vs. Memory usage from experiments with the planar surface representation on RTINs using University of Utah grid. (Refer to Figure 4(a)-(i).)

4 Experiments and Results

Our goal was to compare memory vs. accuracy for different representations. We used test data from different sensors and landforms, downloaded from the USGS Seamless Data Distribution System (http://seamless.usgs.gov). Additional tables are given in the full paper; in this abstract we simply include one table, one graph, and some illustrations for different parameter values.

Our triangle representation requires only an index and a vertex pointer per triangle. Using quadratic polynomials adds 5 coefficients. An ordinary linear TIN has a similar cost of about 60 bytes per vertex, or 30 bytes per triangle. Figure 3 shows the triangle meshes obtained from the planar and polynomial representations. The polynomial representation has fewer triangles, as expected. Considering the fact that each triangle in the polynomial representation needs 3.5 times more storage than the planar triangles, the reduction in the number of triangles still does not allow us to easily choose between the two representations. However, some applications need to store more information with the triangles (e.g., vegetation, water flow, or data statistics) and the observed reduction can significant.

Figures 4 and 5 show the variation of absolute error

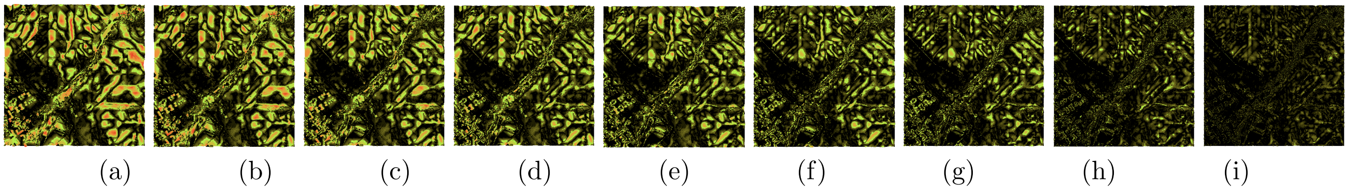
(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 4: Absolute error grids from the planar surface representation on RTINs, University of Utah data. As the absolute error increases, the color changes from black to green to yellow to red.

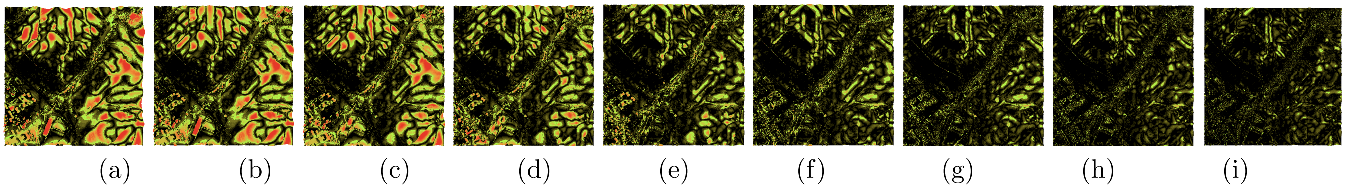
(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 5: Absolute error grids from the quadratic surface representation on RTINs, University of Utah data.

over the entire region. The number of triangles in the RTINs and vertices in the regular TINs are chosen so that the total memory utilization varies over a common range (about 25 KB to 750 KB). For each such representation, we record the sum of absolute errors over all grid points.

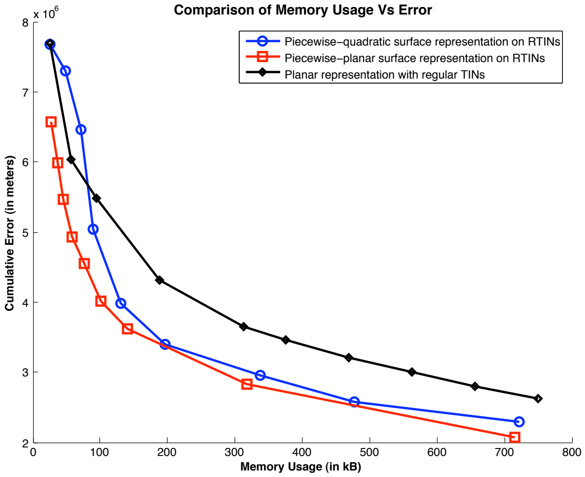

Figure 6: Comparison of memory usage vs. accuracy between various surface representations

Figure 6 shows plots of memory usage vs. cumulative error over the entire grid, for three representations of the terrain. Note that although the planar representation is slightly better than the polynomial representation using RTINs, it might not be so in applications where each triangle in the structure must also contain additional terrain information (such as type of land form or land use, drainage, texture or satellite image coordinates). In such applications, could realize a significant benefit from the reduction in the number of triangles required by a polynomial representation.

5 Discussion

In this paper we compare memory requirements for RTINs using planar and quadratic surface patches. (We also simplify code for fast neighbor computations in RTINs.) RTINs provide a more useful representation than general TINs for gridded data, since they provide a hierarchy of levels of detail, instead of just a single level. Thus, we believe their larger number of triangles, offset by the small per-triangle size, makes them a better choice.

Using polynomial surface patches depends on how much per-triangle or per-vertex data is to be stored. Since with curved patches one can use fewer triangles, with more data per triangle the advantage swings to polynomial patches. More should be said about the best way to fit interior grid points tightly to the polynomial surface. We have used only quadratic surfaces, but it might be useful to try cubic surfaces too. Using higher degree polynomials have the overhead of storing more and more coefficients, not necessarily guaranteeing a tighter fit to the grid points. In fact, polynomial patches of degree 5 or higher tend to be noisy.

Acknowledgment

This research was supported in part by NSF and NGA/Darpa.

References

[1] Hiroshi Akima. Bivariate interpolation and smooth surface fitting based on local procedures. *Commun. ACM*, 17(1):26–31, 1974.

[2] Hiroshi Akima. A method of bivariate interpolation and smooth surface fitting for irregularly distributed data points. *ACM Trans. Math. Softw.*, 4(2):148–159, 1978.

[3] Leila de Floriani and Enrico Puppo. Hierarchical triangulation for multiresolution surface description. *ACM Trans. Graph.*, 14(4):363–411, 1995.

[4] Mark A. Duchaineau, Murray Wolinsky, David E. Sigeti, Mark C. Miller, Charles Aldrich, and Mark B. Mineev-Weinstein. ROAMing terrain: real-time optimally adapting meshes. In *IEEE Visualization*, pages 81–88, 1997.

[5] W. S. Evans, G. Kirkpatrick, and G. Townsend. Right-triangulated irregular networks. *Algorithmica, vol. 30, no. 2, pp. 264-286*, 2001.

[6] Michael Lee and Hanan Samet. Navigating through triangle meshes implemented as linear quadtrees. *ACM Transactions on Graphics*, 19(2):79–121, 2000.

[7] Michael Lee, Hanan Samet, and Leila de Floriani. Constant-time neighbor finding in hierarchical tetrahedral meshes. In *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*, pages 286–295, Washington, DC, USA, 2001. IEEE Computer Society.

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

[9] Peter Lindstrom, David Koller, William Ribarsky, Larry F. Hodges, Nick Faust, and Gregory A. Turner. Real-time, continuous level of detail rendering of height fields. *Computer Graphics (SIGGRAPH 96)*, pages 109–118, 1996.

[10] Renato Pajarola, Marc Antonijuan, and Roberto Lario. Quadtin: Quadtree based triangulated irregular networks. *Proceedings of IEEE Visualization 2002*, 00, 2002.

[11] Albrecht Preusser. Algorithm 684: C1- and c2-interplation on triangles with quintic and nonic bivariate polynomials. *ACM Trans. Math. Softw.*, 16(3):253–257, 1990.

[12] Hanan Samet. The quadtree and related hierarchical data structures. *ACM Comput. Surv.*, 16(2):187–260, 1984.

[13] Günther Schrack. Finding neighbors of equal size in linear quadtrees and octrees in constant time. *CVGIP: Image Underst.*, 55(3):221–230, 1992.