SuperTINs
New Line of Sight Algorithm
Renders Superlative TINs Superfluous

Bob Chamberlain  rgc@jpl.nasa.gov
Joseph Gonzalez  josephg@its.caltech.edu
TINs are piecewise planar models of a surface
- Triangulated Irregular Networks
- Used by Corps Battle Simulation since 1996
- For seven years I've been planning how to make better TINs
  » Finally, it was funded; Goal: “Superlative” TINs

Summer Research Project: Optimize the TINning Parameters
- First step: How do you know one TIN is better than another?
- Answer: Compare effectiveness re LOS assessment
  » Assume a regular triangulation of the full DEM is “ground truth”
- Then why not just use the regular triangulation?
  » Because ... and ... and ... — Why not, indeed!

The New O(log n) Line-of-Sight Algorithm
- Build a quadtree from quads of DEM posts, quads of quads, ...
- Store the quadtree on hard disk, use cache for access speed
- Use the quadtree to skip over many edges at once
First Story: Piece-wise Planar Terrain Skins

Digital Elevation Map (DEM)

Triangulated Irregular Networks (TINs)

Regular triangulations
State-of-the-Art TIN Construction in Military Simulations

1. **DEM**

2. **Slice for each 1° x 1° tile**

3. **Delaunay**

4. **Select VIPs (Very Important Points)**

5. **or**

6. **Starting TIN**

7. **Refine**

8. **or**

9. **Delaunay Triangulation for each 1° x 1° tile**

10. **Refine**

11. **Refine**

12. **Refine**

13. **Low Res TINs for each 1° x 1° tile**

14. **Med Res TINs**

15. **Refine**

16. **High Res TINs**
How to make better TINs

California Institute of Technology

ReadDEM
Extract playbox from NIMA CDs
Playbox DEM

DeduceRichlines
Find ridges, channels, shorelines, boundary profiles
"Rast稹zed" richlines with "good" endpoints

SimplifyLines
"Fat" richlines with "good" endpoints

PerformCDT
Use Faugeras' algorithm, then do the Delaunay triangulation
Constrained Delaunay Triangulation (with constrained "fat" lines marked)
CDT

AnnealTIN
Starting TIN

RefineTIN
A better TIN with more triangles

AnnealTIN (again)

BIGTIN
The highest resolution TIN covering the entire playbox

For details, see the paper, which will be submitted to Interfaces.
Improved TIN process has three dozen free parameters

- Tolerances, minimum lengths, budgets, weights, goals
- Reasonable guesses can be made, but it would be “cool” — and maybe a lot better — to optimize

Opportunity: Summer Undergraduate Research Fellowship

- Low risk, low cost, maybe a high payoff
- Five nibbles, two bites — one, clearly superior in several ways, suggested genetic optimization

First step in optimization: Understand the response surface

- Q. How do we know one TIN is better than another?
- A. Compare LOS results vs a regular triangulation of the DEM
And this led to the question...

Why not just use a regular triangulation of the DEM?
LOS evaluation would be as good as on the best possible TIN

• Performance
  • Quality demands all edges be considered
  • There would be too many edge crossings to evaluate

• Memory
  • CBS playbook DEMs would need up to 13 GB for the DEM
  • TINs are only allocated 350 MB
"LOS looks like a special case of ray tracing"

Algorithm
- Quadtrees (instead of octrees) allow test of LOS vs large areas
- Above the leaves, test intersections of LOS with bounding boxes
- Leaf nodes contain the terrain triangles defined by four DEM posts

Speed
- It's an $O(\log n)$ algorithm

Memory
- Higher limits now than in 1996
- Regular triangles can be implicit, compression may be possible
- SRTM has 1 arc-sec (31 m) spacing; DTED1 had 3 arc-sec
  » 6 arc-sec (185 m) spacing would fit the biggest playbook into 500 MB
- (Later:) May be able to use hard disk

Risk
- Very low due to simplicity
What Did We Do?

We tabled TIN implementation though it was 80% complete
- Not resumed because the Quadtree LOS Algorithm works!
- Documented in the paper, which we will submit to Interfaces

We looked for other possible "show stoppers"
- LOS and elevations are the only CBS uses of TINs
- The user interface ("the workstation") had been using the TINs to find elevations to draw LOS fans
- Quadtrees can easily produce interpolated DEM elevations

We developed a prototype that runs with the main simulation engine as a proof of concept
- Tested on 65,000 LOS's drawn from a CBS exercise
- LOS tests on the old TINs were right almost 81% of the time
- Achieved same speed (96%), using TCP/IP for interprocess communication; improvement is likely
which brings us to the third story...

Line of Sight Algorithms
State-of-the-Art LOS Assessment in Military Simulations

**Edge Traversal**
- demands efficient TINs

**Periodic Sampling**
- flexible at the expense of accuracy
The Quadtree LOS Algorithm

As with the TIN algorithm, there are two steps:

- **Pre-exercise preparation:**
  - Instead of a TIN, build a quadtree

- **During the exercise:**
  - Apply the quadtree algorithm to evaluate LOS
  - Interpolate to determine elevations
Constructing an LOS Quadtree
First, the DEM (elevation posts)
— The Leaves of the Tree —
Boxes containing 4 posts each

height of the box = height of tallest post it contains

The triangulation is implicit

rgc 14
The Leaves of the Tree
Boxes containing 4 posts each

height of each box = height of tallest post it contains

(when used, there is an implicit adjustment for the curvature of the Earth)
— The Leaves of the Tree —
Boxes containing 4 posts each

height of each box = height of tallest post it contains

16 boxes
— The Leaves of the Tree —
Boxes containing 4 posts each

height of each box = height of tallest post it contains

256 boxes (in this example)
The Lowest Nodes in the Tree
Combine quads of boxes

height of each quad = height of tallest post it contains

4 quads
The Lowest Nodes in the Tree
Combine quads of boxes

height of each quad = height of tallest post it contains

64 quads
The Next Level of Nodes
Combine quads of quads

height of each quad = height of tallest post it contains
The Next Level of Nodes
Combine quads of quads

height of each quad = height of tallest post it contains

16 quads of quads
And so on...
quads of quads of quads

height of each quad = height of tallest post it contains

4 quads of quads of quads
Finally (in this example), a quad of quads of quads of quads

height of the top of this quad = height of tallest elevation post anywhere in the playbox (plus an adjustment for the curvature of the Earth)

Next, a one-dimensional illustration
A One-Dimensional Illustration
now looking from the side instead of downward

Make the tree
First, read the elevation data (the DEM)
Make the tree

First, read the elevation data (the DEM)

Drape a skin over the DEM posts
Make the tree

First, read the elevation data (the DEM)

Drape a skin over the DEM posts

Collect the posts in pairs (instead of quads)
Make the tree
  First, read the elevation data (the DEM)
  Drape a skin over the DEM posts
  Collect the posts in pairs (instead of quads)
  Make pairs of pairs . . . and so on

. . . until done

Next, an illustration of an LOS assessment
Now, apply the algorithm

LOS is not obstructed

- This is a relatively difficult case: LOS grazes the ground surface
- 9 box queries at a distance of 20 posts (about 40 edge crossings in 3-D)
LOS is unobstructed

Next, an obstructed case
An Illustration of an Obstructed LOS

On the same terrain: LOS is obstructed

- LOS still close to grazing
- Also 9 box queries, plus two sets of leaf queries, at a distance of 20 posts (about 40 edge crossings in 3-D)
Consider the following...
An Easy Case

Different terrain: LOS is unobstructed
Consider the following...

An Easy Case

Different terrain: LOS is unobstructed

- 1 box query, still at a distance of 20 posts (about 40 edges)
An Easy Case

Different terrain: LOS is unobstructed

- 1 box query, still at a distance of 20 posts (about 40 edges)
Different terrain: LOS is unobstructed
- 1 box query, still at a distance of 20 posts (about 40 edges)

LOS is unobstructed
SURF'S UP!