Storing and analysing massive aerial LiDAR datasets in a DBMS

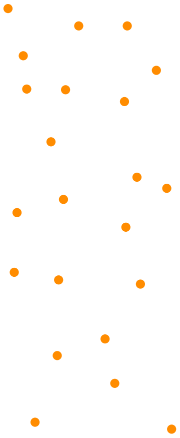
Hugo Ledoux

Technische Universiteit Delft

November 30 2010
ELMF, Den Haag

Introduction

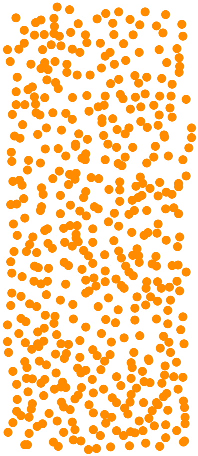
- Computers have many problems dealing with billions of points:
  - Storing is OK
  - Visualisation is still a challenge
  - **Processing** and **analysis** are very problematic
- Processing operations:
  - derivation of slope/aspect,
  - conversion to grid format,
  - calculations of area/volumes,
  - viewshed analysis,
  - creation of simplified DTM,
  - extraction of bassins,
  - etc.



Advances in technologies to collect data are far superior to our ability to process data.

Introduction

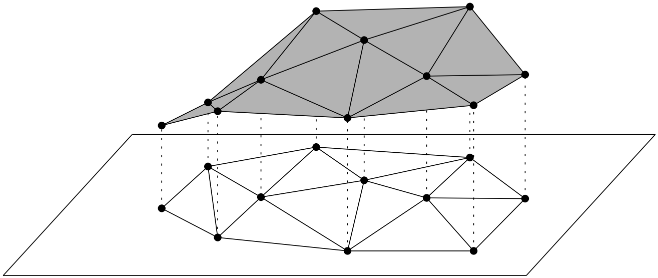
- Computers have many problems dealing with billions of points:
  - Storing is OK
  - Visualisation is still a challenge
  - **Processing** and **analysis** are very problematic
- Processing operations:
  - derivation of slope/aspect,
  - conversion to grid format,
  - calculations of area/volumes,
  - viewshed analysis,
  - creation of simplified DTM,
  - extraction of bassins,
  - etc.



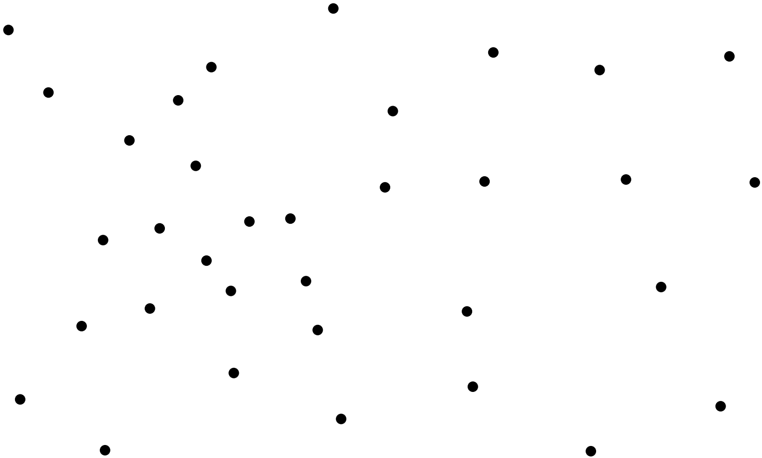
Advances in technologies to collect data are far superior to our ability to process data.

Point clouds are often "2.5D surface"

LiDAR datasets are formed by scattered points in 3D space, which are the samples of a surface that can be projected on the horizontal plan.

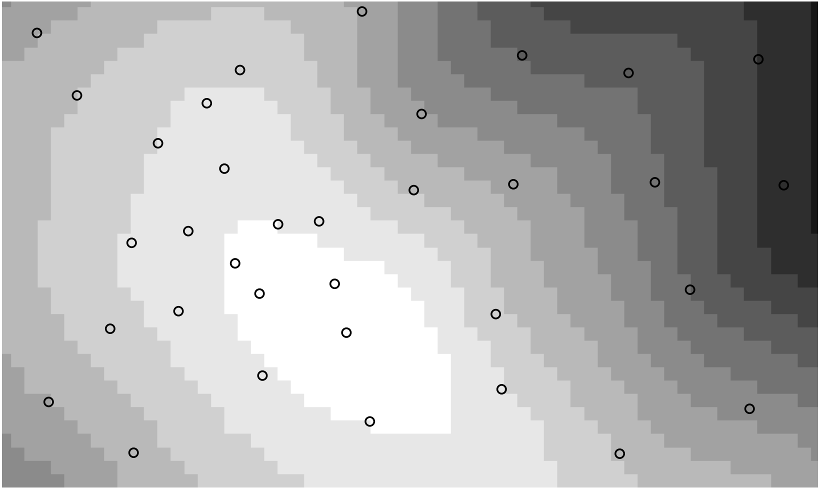


Reconstruction of the surface



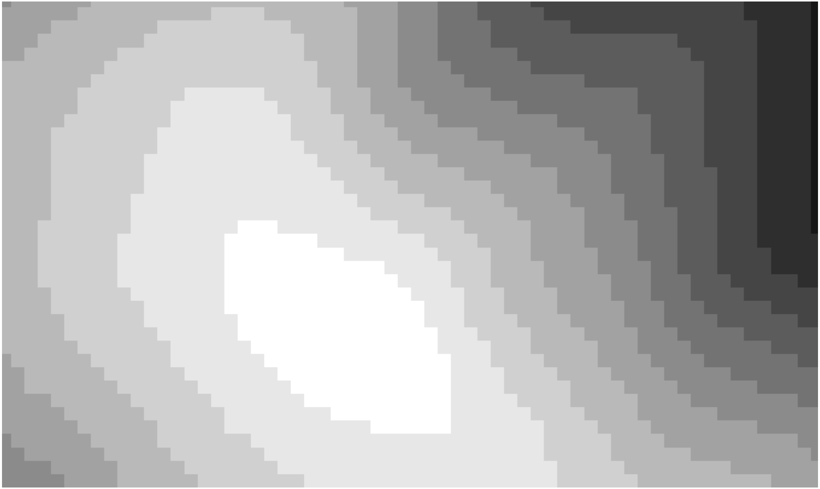
Original LiDAR points

Reconstruction of the surface



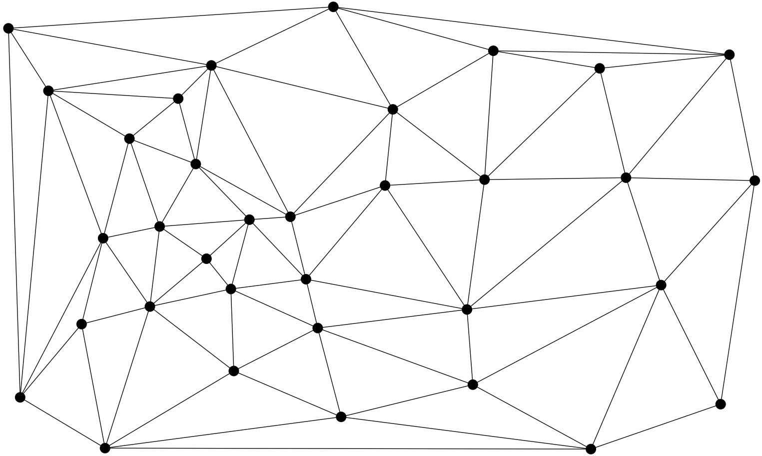
Raster representation

Reconstruction of the surface



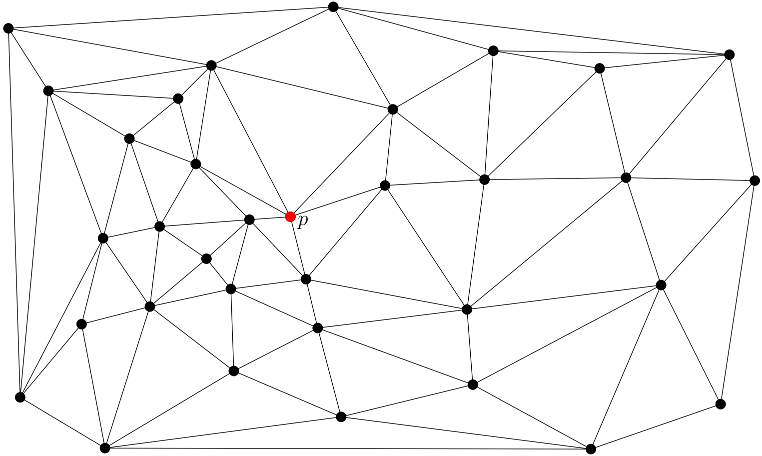
Raster representation

Reconstruction of the surface



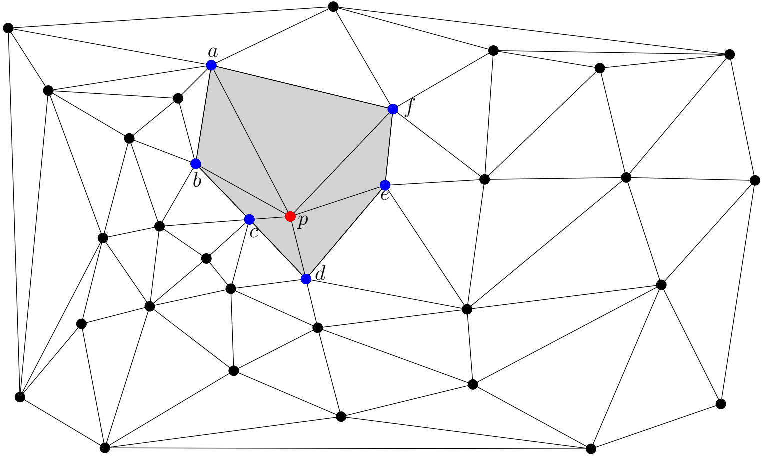
TIN (with Delaunay triangles)

Reconstruction of the surface



TIN (with Delaunay triangles)

Reconstruction of the surface



TIN (with Delaunay triangles)

Related work

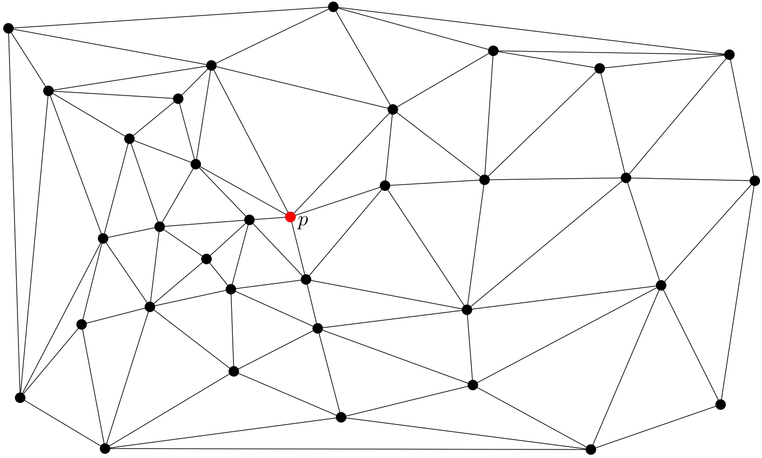
- Terrasolid: max is main memory
- ArcGIS: *Terrain* type (hierarchical structure)
- Oracle Spatial 11*g*: *Point Cloud* & *TIN* types
- External memory algorithms [AAD06, ADHZ06]
- Streaming of geometries of Isenburg *et al.* [ILSS06, ILS+06]

Storing triangles in a DBMS

1. Storing independently triangles ($\sim$ OGC)
2. Triangle-based data structure used by triangulation libraries [BDP$^+$02]
3. Edge-based data structure (e.g. half-edge [M$\ddot{8}$8])

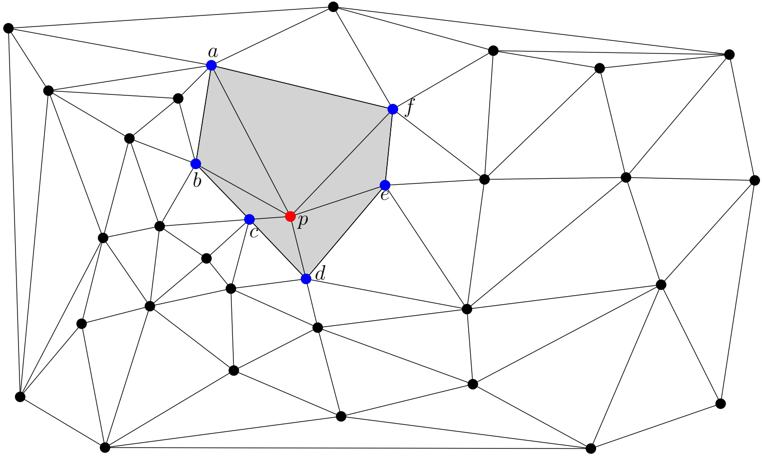
A star-based data structure

- Goes beyond the usual "store points and edges/triangles"
- Ideas come from data structures for compression of graphs [BBCK05]



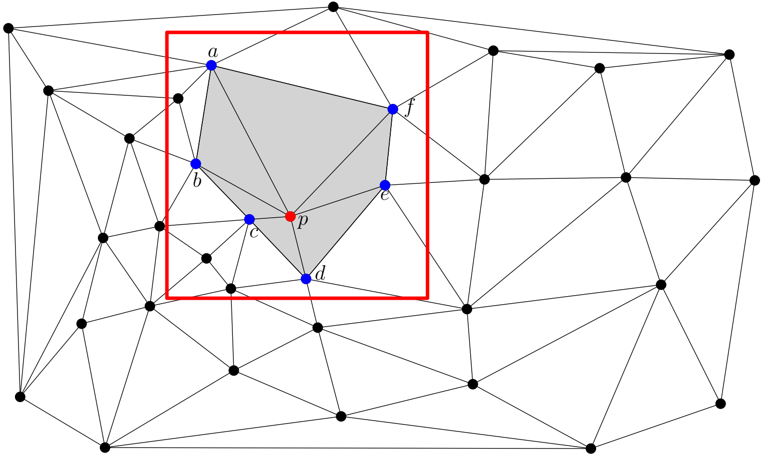
A star-based data structure

- Goes beyond the usual "store points and edges/triangles"
- Ideas come from data structures for compression of graphs [BBCK05]



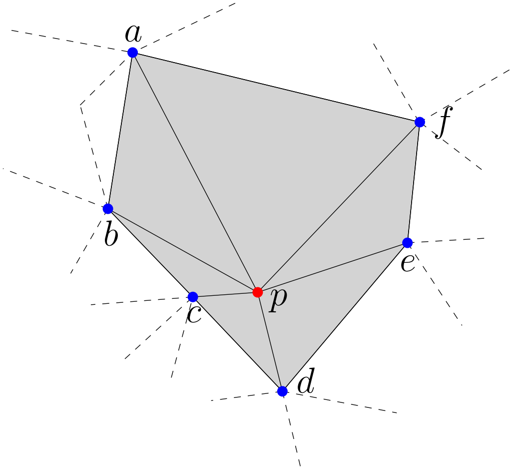
A star-based data structure

- Goes beyond the usual "store points and edges/triangles"
- Ideas come from data structures for compression of graphs [BBCK05]



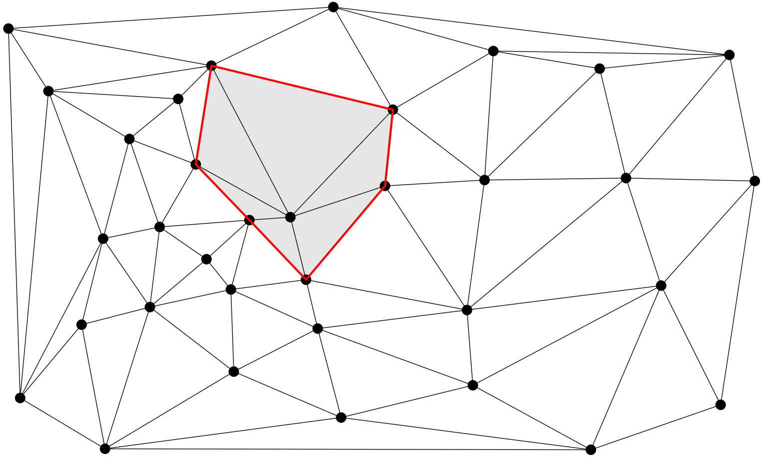
A star-based data structure

- Goes beyond the usual “store points and edges/triangles”
- Ideas come from data structures for compression of graphs [BBCK05]

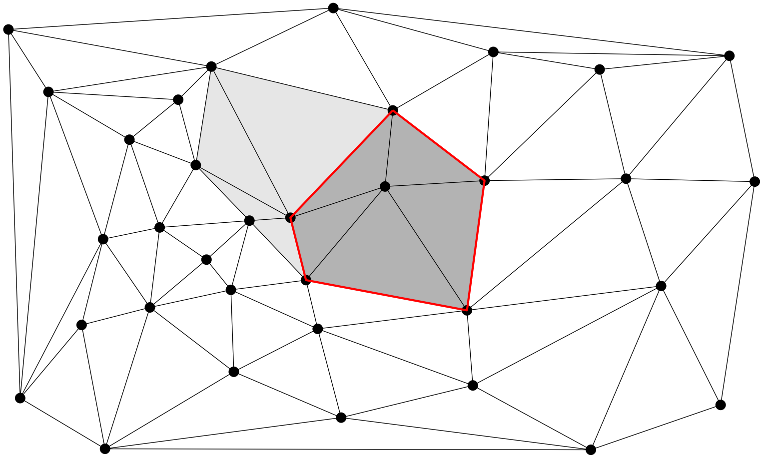


$$\boxed{star(p) = abcdef}$$

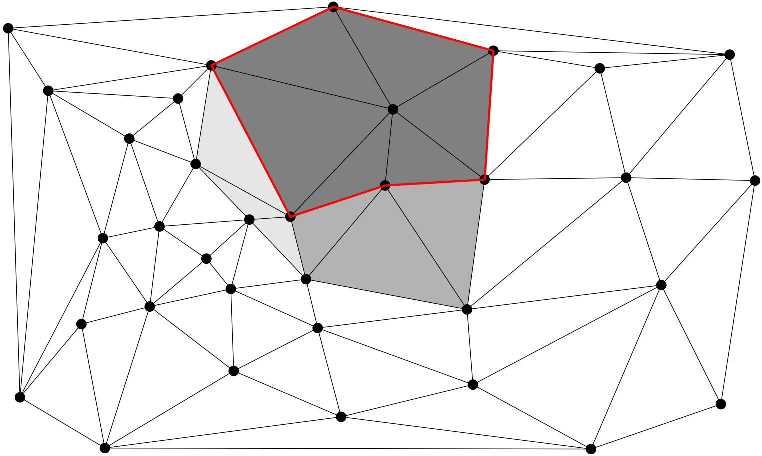
Every star(v) is stored → implicit triangles



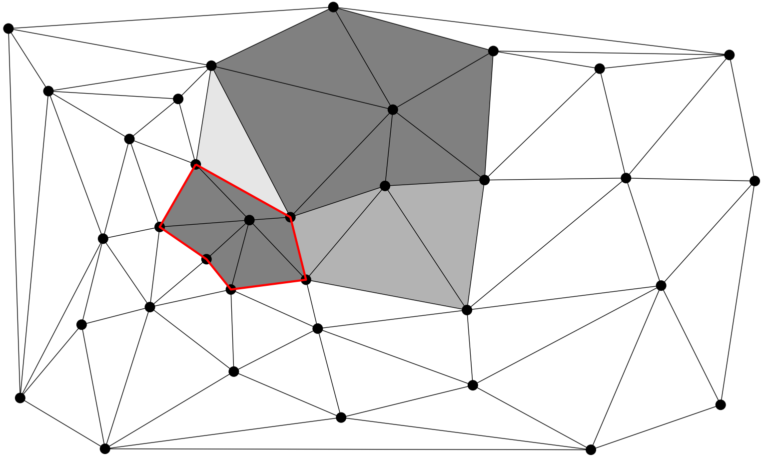
Every star(v) is stored → implicit triangles



Every star(v) is stored → implicit triangles



Every star(v) is stored $\rightarrow$ implicit triangles



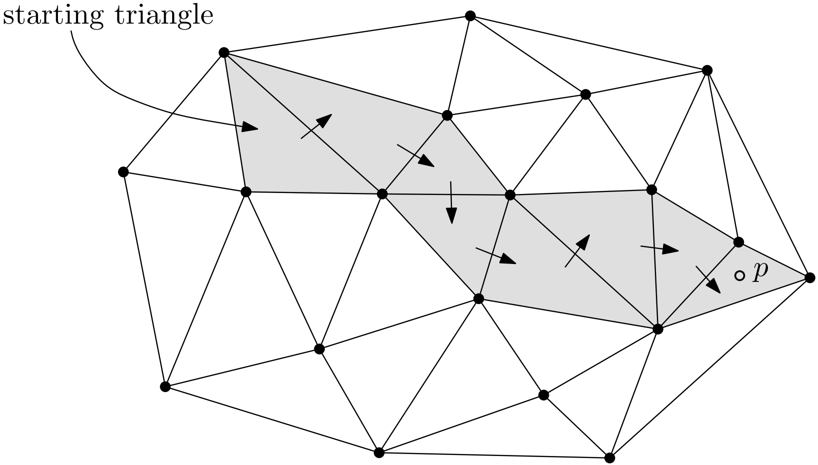
Stars in a DBMS

ID	x	y	z	star[]
1	3.21	5.23	2.11	[2,44,55,61,23]
2	5.19	29.01	4.55	[7,98,111,233,222]
3	22.43	15.99	8.19	[99,101,73,23]
...	...	...	...	...
5674	221.19	15.23	37.81	[309,802,793,1111]

Advantages:

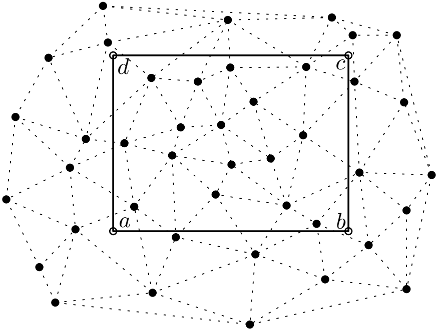
1. Only **one** table with $id - x - y - z - star$
2. No spatial index needed: fetching of triangles based on "walking"
3. Star column need not be filled ($\sim$ Simple Features)
4. Local updates are possible (insertion and removals)

Point Location = "Walking" in the triangulation

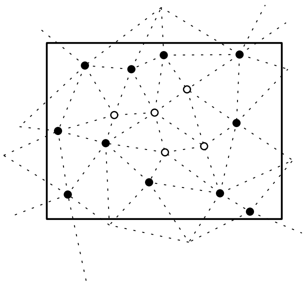


(Can be made efficient with some tricks [MSZ99])

Range Queries: also uses the triangulation



(a)



(b)

One problem: how to create that DT in the first place?

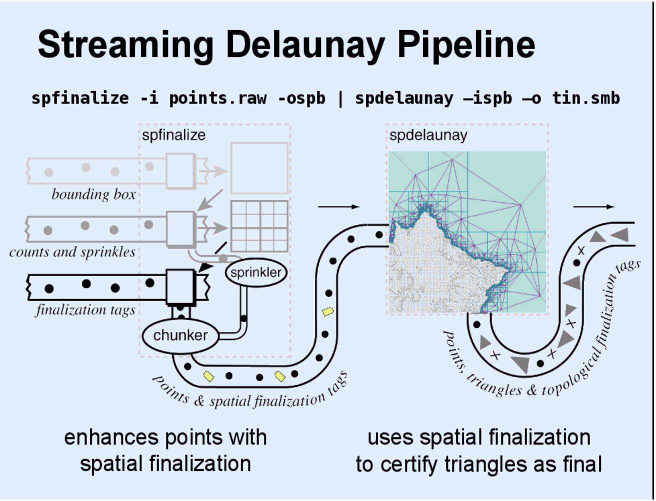
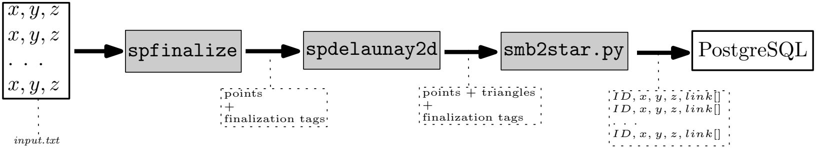
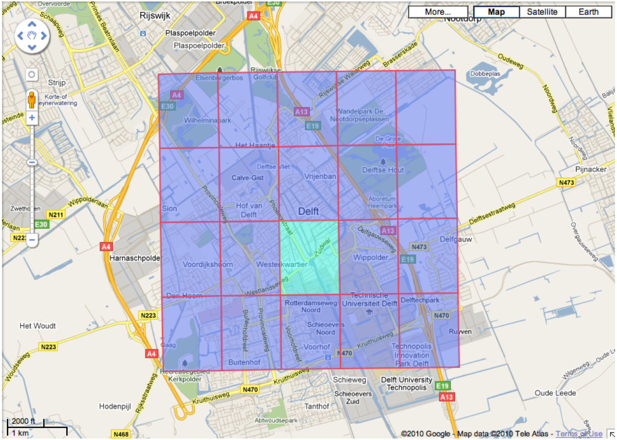


Figure from Martin Isenburg's presentation at GIScience 2006 [ILSS06]

Streaming of geometries to construct massive TINs

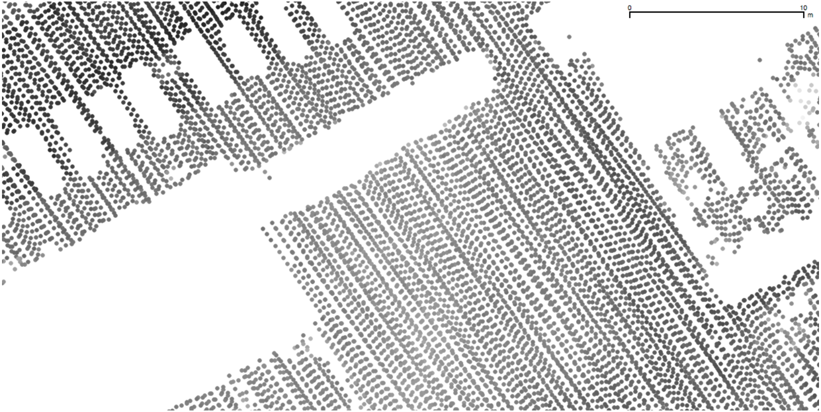


Experiments with AHN2 datasets



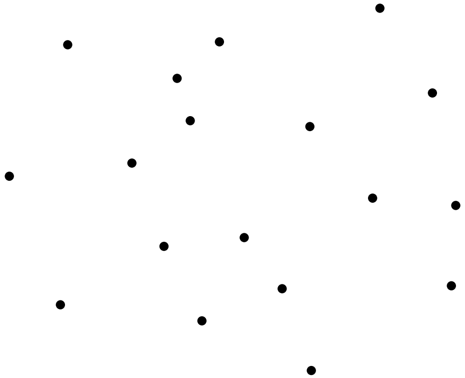
	# pts	# triangle	$degree_{avg}$	$degree_{max}$
20tiles	281 884 687	563 768 199	6.00	63
g37en1_15	8 605 090	17 201 289	6.00	39

Experiments with AHN2 datasets



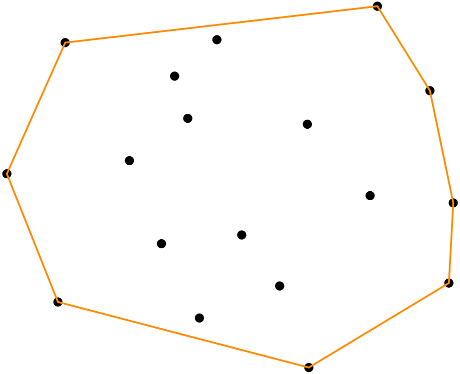
Examples of queries: statistics about convex hull

```
20tiles=# select count(id) from points where is_convexhull(star) is true
count
-------
1173
(1 row)
Time: 333050.861 ms
```



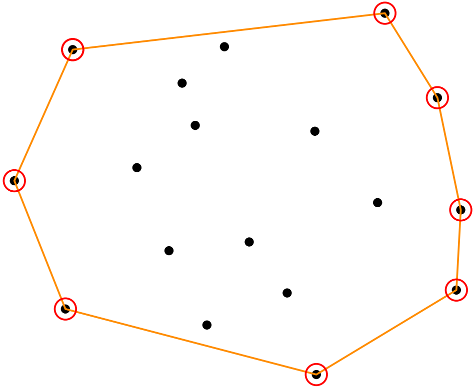
Examples of queries: statistics about convex hull

```
20tiles=# select count(id) from points where is_convexhull(star) is true
count
-------
1173
(1 row)
Time: 333050.861 ms
```



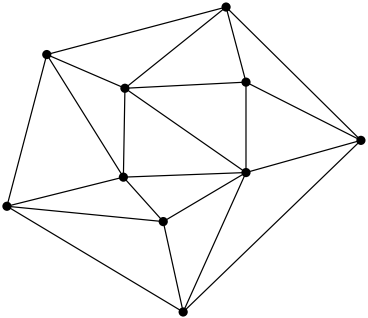
Examples of queries: statistics about convex hull

```
20tiles=# select count(id) from points where is_convexhull(star) is true
count
-------
1173
(1 row)
Time: 333050.861 ms
```



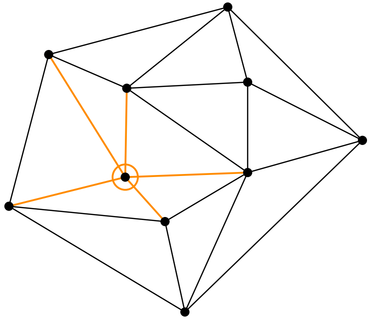
Examples of queries: statistics about degree of vertices

```
20tiles=# select avg(degree(star)) from points;
       avg
--------------------
 5.9999958142601850
(1 row)
Time: 332265.041 ms
```



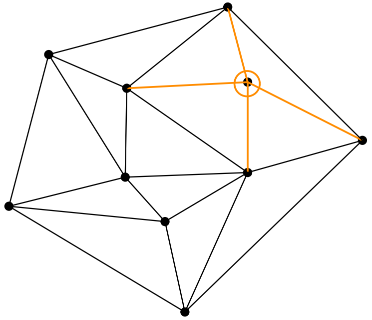
Examples of queries: statistics about degree of vertices

```
20tiles=# select avg(degree(star)) from points;
       avg
--------------------
 5.9999958142601850
(1 row)
Time: 332265.041 ms
```



Examples of queries: statistics about degree of vertices

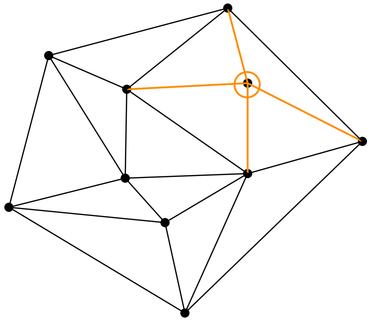
```
20tiles=# select avg(degree(star)) from points;
       avg
--------------------
 5.9999958142601850
(1 row)
Time: 332265.041 ms
```



Examples of queries: statistics about degree of vertices

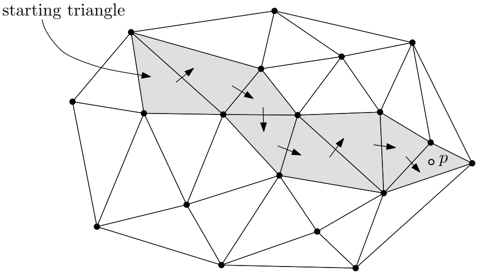
```
g37en1_15=# select degree(star) as degree, count(id) from points group
by degree order by degree;

 degree |  count
--------+---------
      3 |   65620
      4 |  844625
      5 | 2277911
      6 | 2484212
      7 | 2005407
      8 |  698540
      9 |  170214
     10 |   37534
     11 |    9587
     12 |    3395
     13 |    1552
     14 |     772
     15 |     456
... (truncated) ...
     37 |       3
     39 |       1
(35 rows)
Time: 39722.017 ms
```

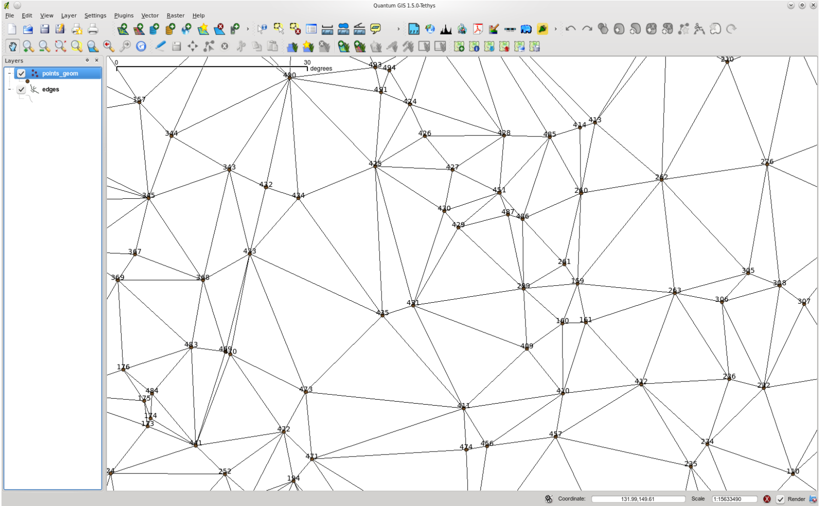


Examples of queries: point location

```
g37en1_15=# select point_location(84111, 446666, 0, 0.2);
    WARNING:  #of samples checked: 24
    WARNING:  start distance is 49.843357
    WARNING:  # of triangles visited is 222
          point_location
    ---------------------------
     (3672278,3695197,3695256)
    (1 row)
    Time: 191.903 ms
```

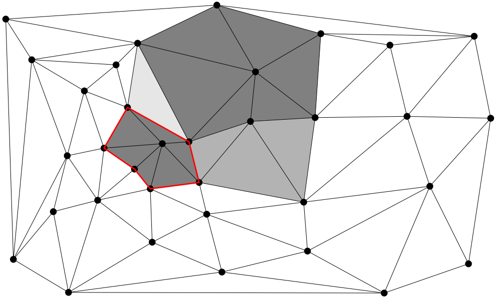


Visualisation with a GIS



Thanks for your attention

Hugo Ledoux
h.ledoux@tudelft.nl



References

P. K. Agarwal, L. Arge, and A. Danner.
From point cloud to grid DEM: A scalable approach.
In A. Reidl, W. Kainz, and G. Elmes, editors, *Progress in Spatial Data Handling—12th International Symposium on Spatial Data Handling*. Springer, 2006.

L. Arge, A. Danner, H. Haverkort, and N. Zeh.
I/O-efficient hierarchical watershed decomposition of grid terrain models.
In A. Reidl, W. Kainz, and G. Elmes, editors, *Progress in Spatial Data Handling—12th International Symposium on Spatial Data Handling*, pages 825–844. Springer-Verlag, 2006.

Daniel K. Blandford, Guy E. Blelloch, David E. Cardoze, and Clemens Kadow.
Compact representations of simplicial meshes in two and three dimensions.
International Journal of Computational Geometry and Applications, 15(1):3–24, 2005.

Jean-Daniel Boissonnat, Olivier Devillers, Sylvain Pion, Monique Teillaud, and Mariette Yvinec.
Triangulations in CGAL.
Computational Geometry—Theory and Applications, 22:5–19, 2002.

Martin Isenburg, Yuanxin Liu, Jonathan Richard Shewchuk, Jack Snoeyink, and Tim Thirion.
Generating raster DEM from mass points via TIN streaming.
In *Geographic Information Science—GIScience 2006*, volume 4197 of *Lecture Notes in Computer Science*, pages 186–198, Mnster, Germany, 2006.

Martin Isenburg, Yuanxin Liu, Jonathan Richard Shewchuk, and Jack Snoeyink.
Streaming computation of Delaunay triangulations.
ACM Transactions on Graphics, 25(3):1049–1056, 2006.

Martti Mäntylä.
An introduction to solid modeling.
Computer Science Press, New York, USA, 1988.

Ernst P. Mücke, Isaac Saias, and Binhai Zhu.
Fast randomized point location without preprocessing in two- and three-dimensional Delaunay triangulations.
Computational Geometry—Theory and Applications, 12:63–83, 1999.

Populating the database

	time pipepline	time B-tree	time tr	time `lasblock`	width_{max}
20tiles	178.3	24.65	36.8	*crashed*	53 003
g37en1_15	4.7	0.3	1.1	4.3	7 601