

Visualization of TINs

Enylton Machado

Roberto Beauclair

{machado,tron}@visgrafimpa.br

Visualization of TINs

- GIS
- Basics in Visualization
 - clipping
 - projection
- Hidden-Surface Remove
 - image-space algorithms
 - object-space algorithms
- Levels of Details

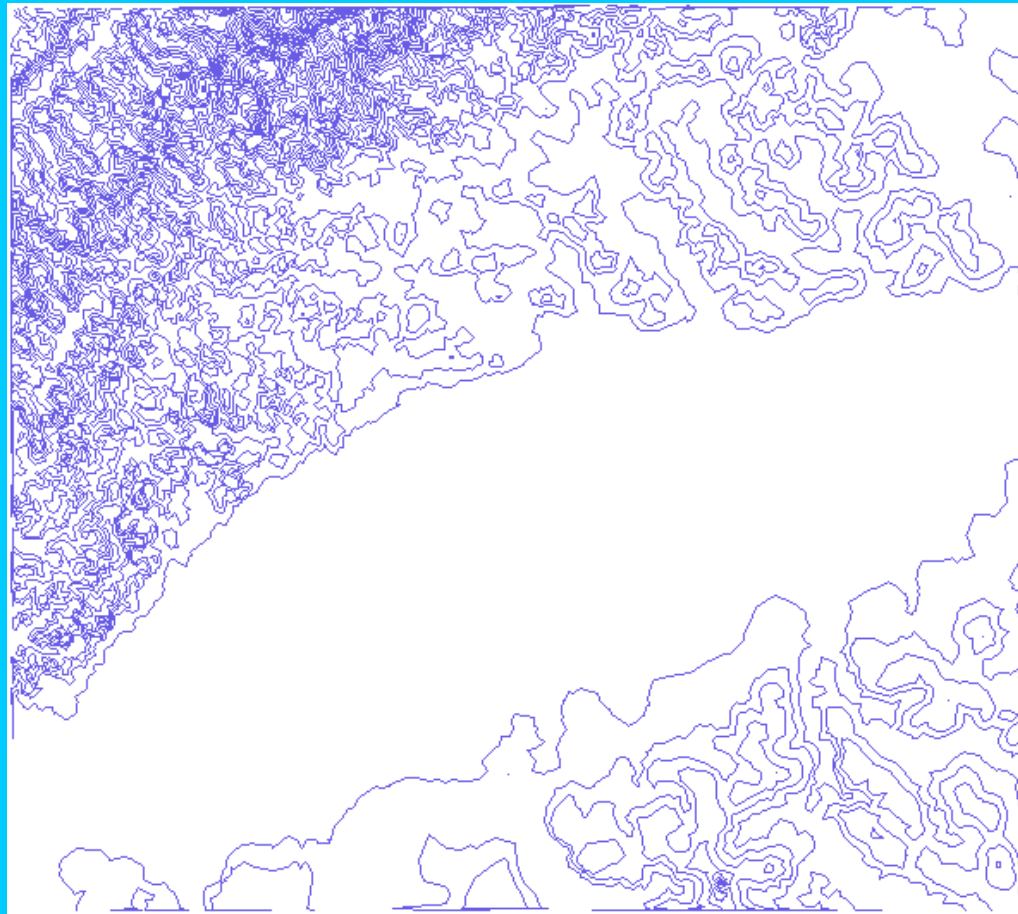
GIS

- Fundamental task:
 - *Visualization of geographic data*
- Three models:
 - Regular square grid
 - Contour-line model
 - **Triangulated irregular network**

Regular Square Grid

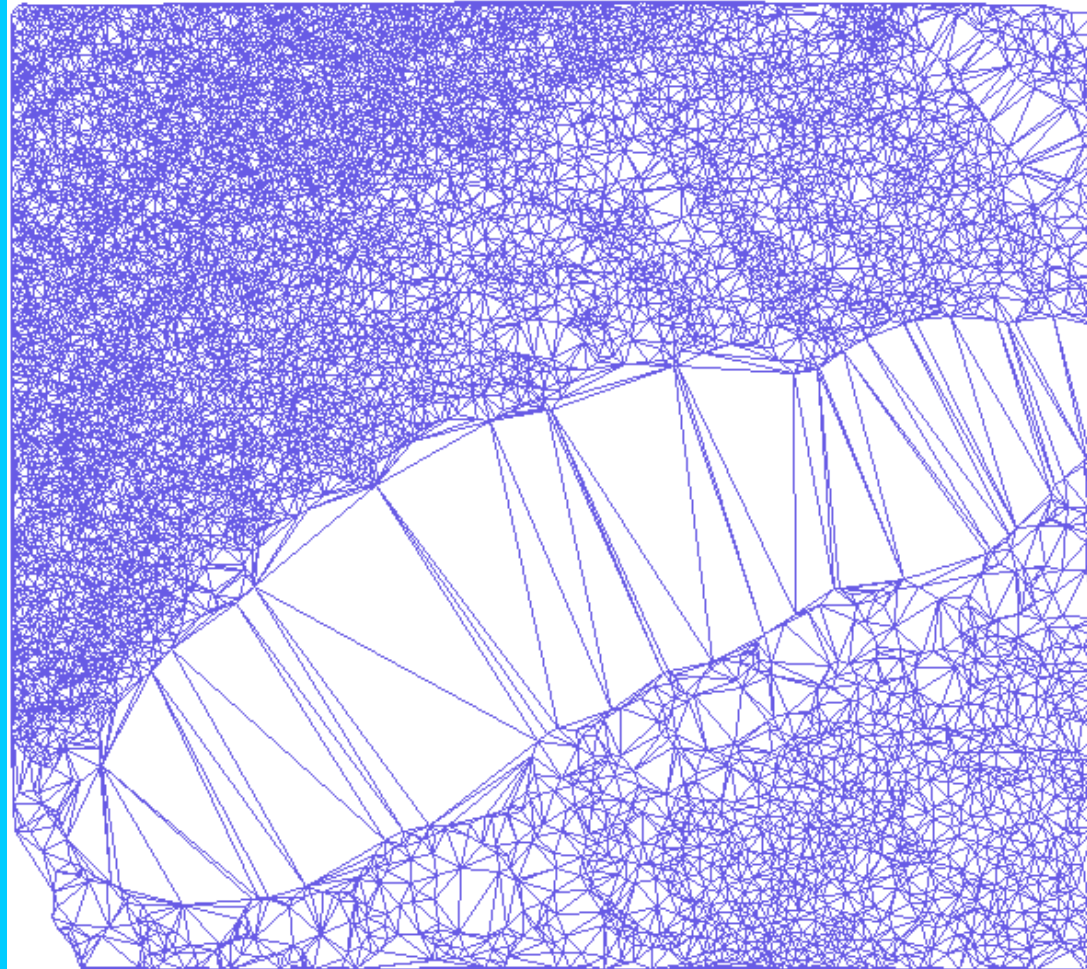
78.5	630.5	638.0	742.7	799.3	673.8	608.8	594.9	624.6	615.3	600.0	560.0
73.4	586.1	688.3	720.6	630.2	634.1	600.0	580.0	581.4	606.3	626.9	580.0
79.7	696.4	643.0	638.7	641.7	616.3	582.2	583.1	594.1	598.3	569.9	605.7
70.7	659.5	625.0	617.2	602.5	586.6	580.0	583.1	560.0	569.7	579.5	573.5
77.3	640.5	658.6	635.5	601.1	588.0	560.0	560.0	560.0	560.0	560.0	560.0
75.9	610.4	620.2	613.5	580.0	560.0	560.0	560.0	560.0	560.0	560.0	560.0
74.2	605.8	603.8	562.2	560.0	560.0	560.0	560.0	560.0	560.0	560.0	560.0
74.9	620.9	580.0	560.0	560.0	560.0	560.0	560.0	560.0	560.0	560.0	562.9
78.2	693.5	560.0	560.0	560.0	560.0	560.0	560.0	560.0	580.0	576.9	580.0
73.6	560.0	560.0	560.0	560.0	560.0	560.0	560.0	580.0	618.0	577.2	617.4
70.0	560.0	558.3	560.0	560.0	560.0	575.2	580.0	606.0	616.5	604.5	581.4
70.0	560.0	556.2	564.1	581.0	582.3	585.7	616.2	622.5	589.3	620.0	587.3
70.0	577.7	560.0	600.0	580.0	583.5	602.8	580.0	606.3	603.9	560.0	599.6

Contour-Line Model



24/03/99

Triangulated Irregular Network



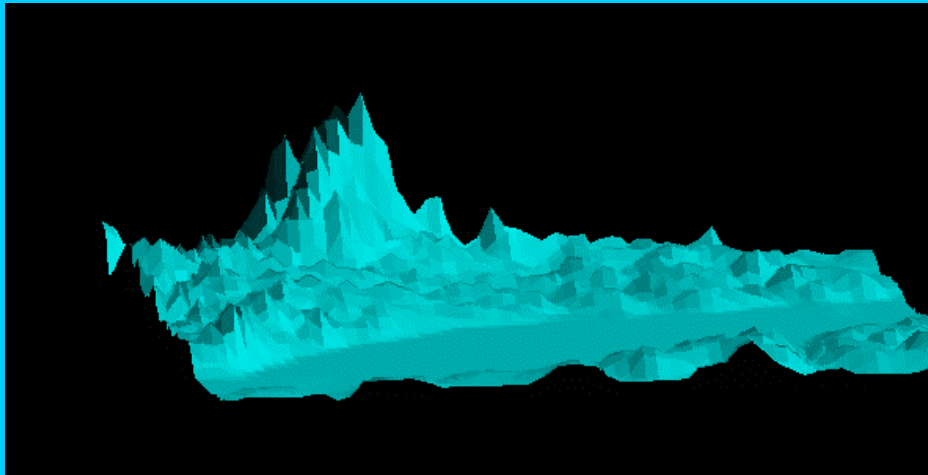
24/03/99

Triangulated Irregular Network

Definition:

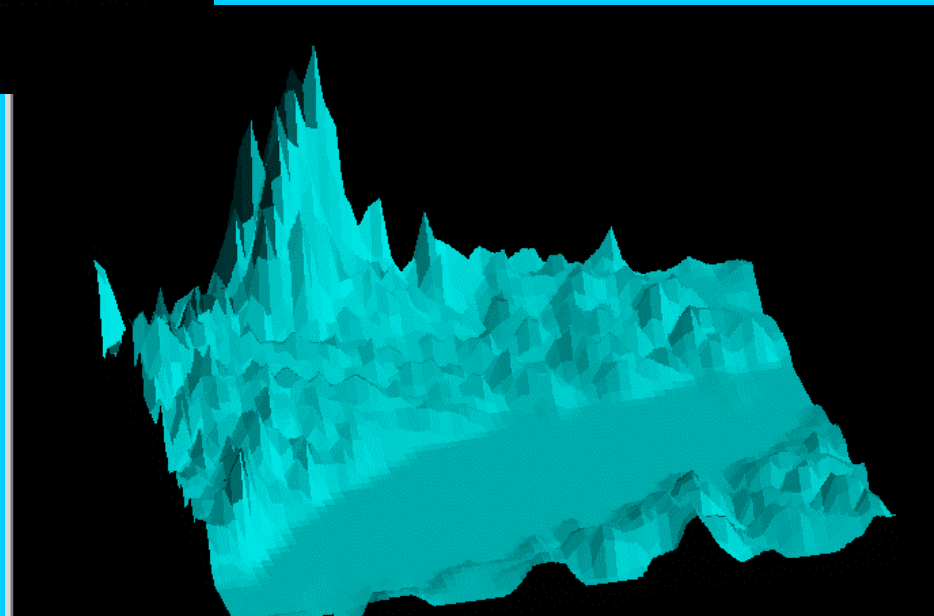
- A finite set of 2-dimensional data point is stored, together with the elevation.
- A collection of triangles in 3D space forming a connected surface which is z -monotone, that is, which is such that no two triangles intersect when projected vertically onto the xy -plane.

Triangulated Irregular Network

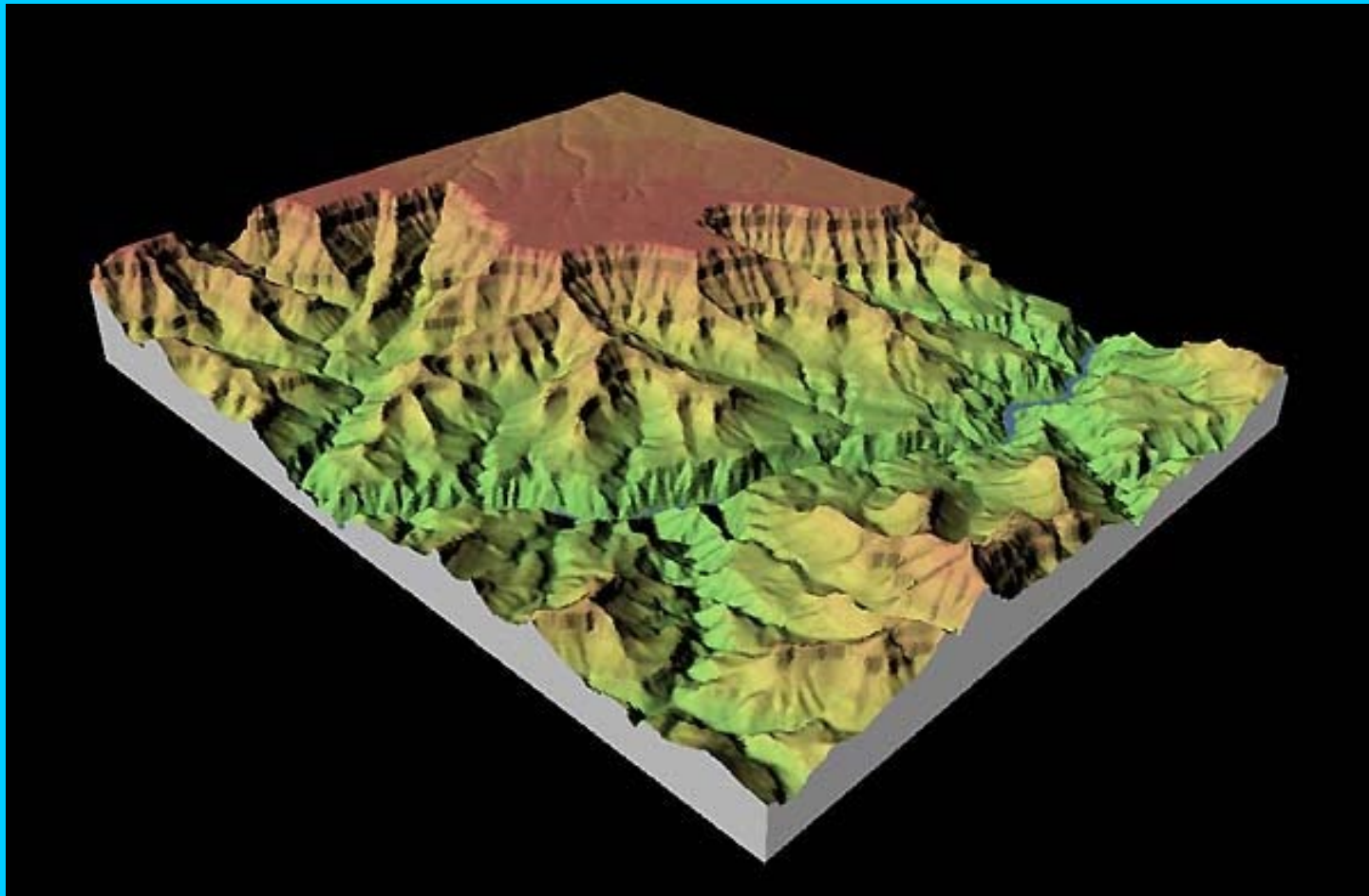


Z-scale = 1.0

Z-scale = 1.5



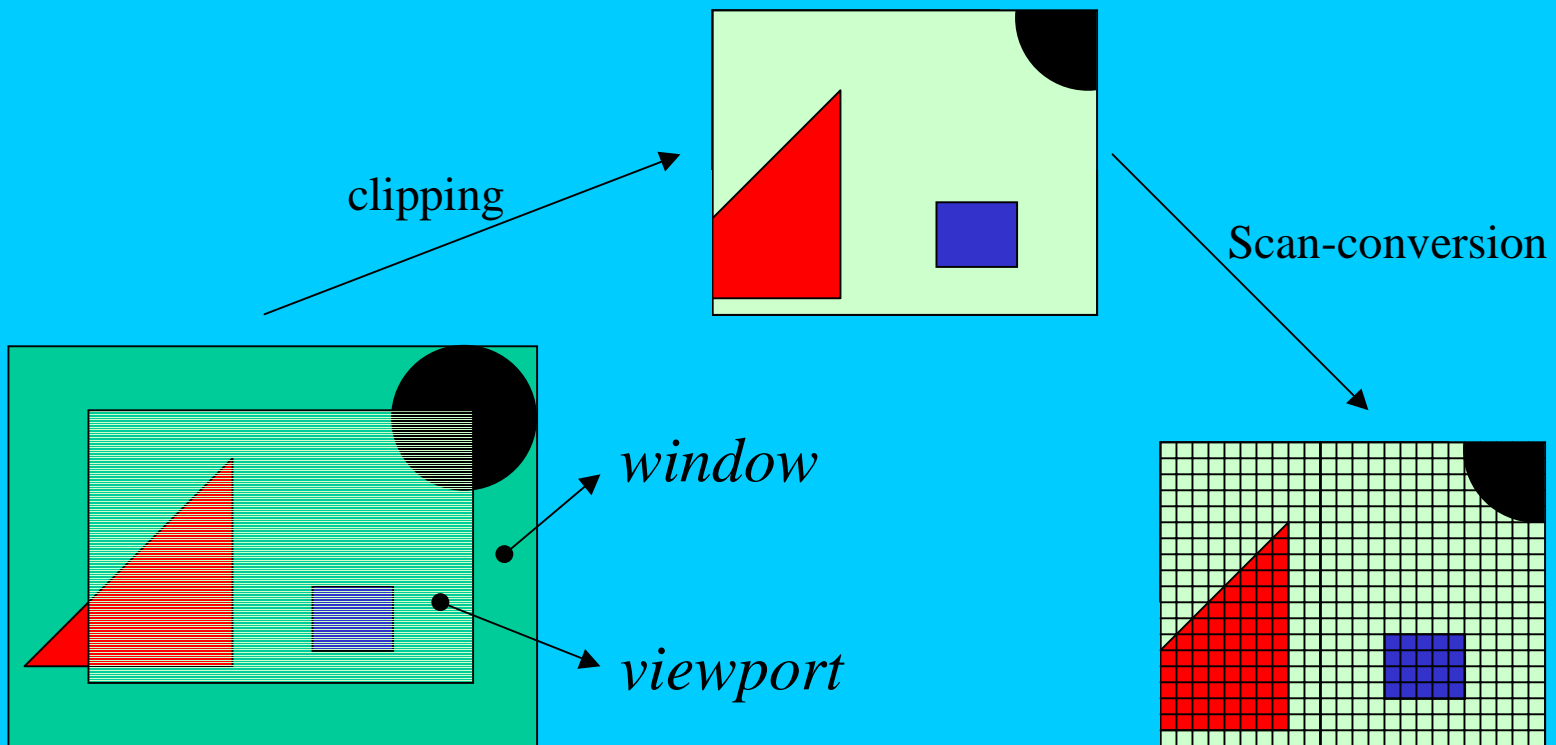
Triangulated Irregular Network



24/03/99

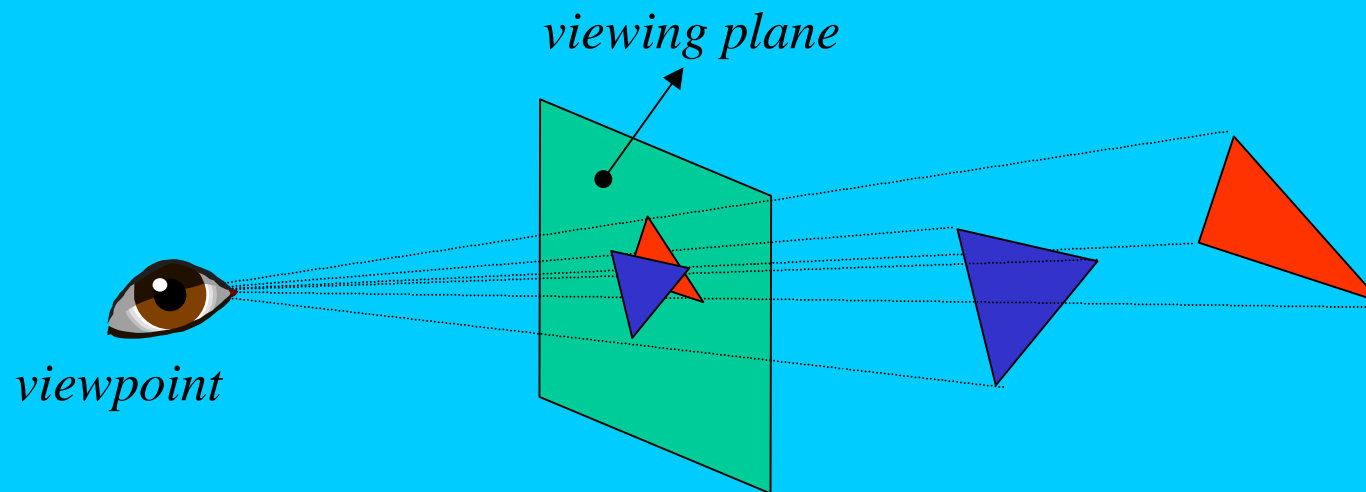
Basics in Visualization

- *Window clipping and scan-conversion.*



Basics in Visualization

- *Projection and hidden-surface removal*



visualizing a 3D scene

Hidden-Surface Removal

- Image-space algorithm
 - First project the objects, and decide during the scan-conversion for each individual pixel which object is visible at that pixel.
- Object-space algorithm
 - Determine which part of each object is visible, and then project and scan-convert only the visible parts.

Image-Space Algorithm

- Z-buffer

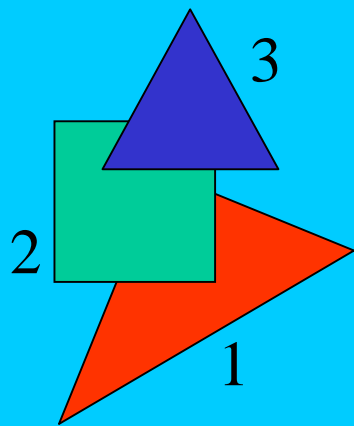
- Store a z-coordinate for each pixel in a buffer.
- Easy to implement, and any graphics workstation provides it, often in hardware.

```
if (  $z_t(x,y) < \text{Zbuf}[x,y]$  ) then
  {
     $\text{FB}[x,y] = \text{color}_t$ ;
     $\text{Zbuf}[x,y] = z_t(x,y)$ ; }
else /* anything to do */;
```

Image-Space Algorithm

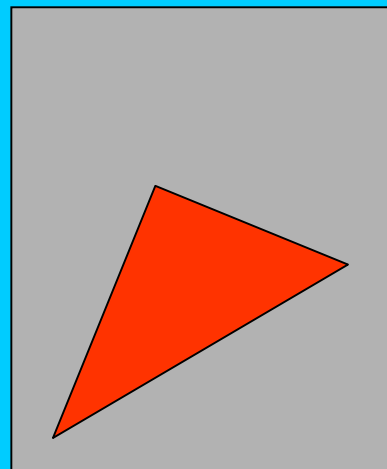
- Z-sort

- scan-convert the objects in a back-to-front order.
- saves memory and don't test visibility.

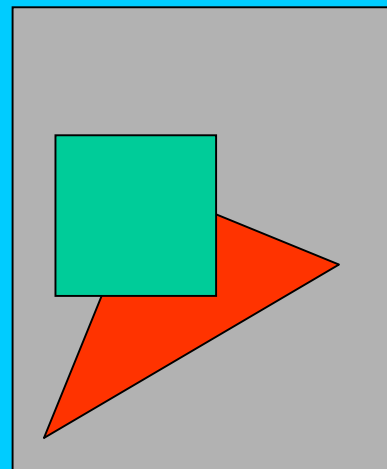


Painter's algorithm

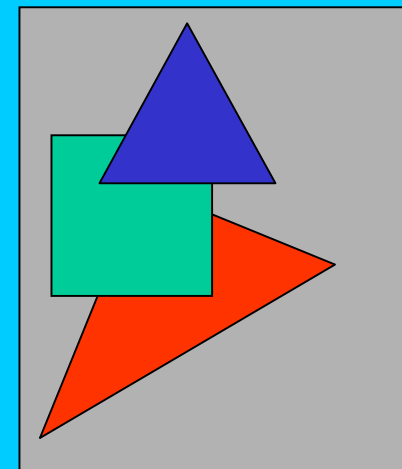
24/03/99



Scan-convert 1



Scan-convert 2

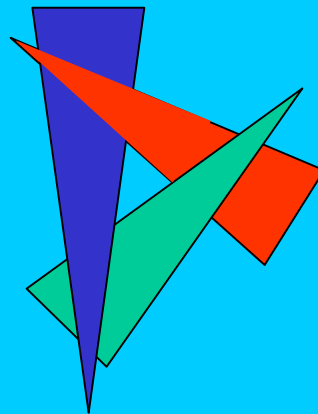


Scan-convert 3

Image-Space Algorithm

- Z-sort problem
 - Avoid the visibility test in scan-conversion phase, but it adds an extra preprocessing step: a back-to-front order, or *depth order*, must be computed.

Cyclically
overlap



Object-Space Algorithm

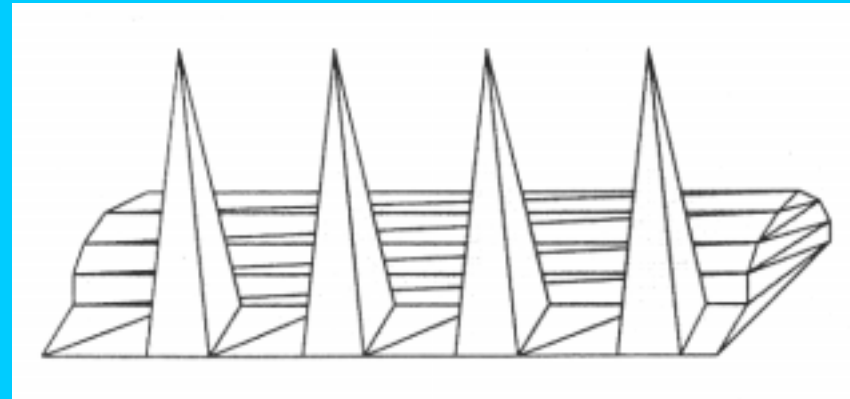
- Image-space methods compute the view of a scene pixel by pixel. This means that the “structure” of the view is lost.
- Object-space algorithms compute a combinatorial representation of the view, computing a *visibility map* of the given objects as a collection of (polygonal) faces.

Object-Space Algorithm

- Object-space hidden surface removal methods tend to be slower than image-space methods such *z-buffer* algorithm, because they “cannot be” implemented in hardware.
- Object-space algorithms can :
 - display the invisible parts dashed
 - compute shadows in a scene
 - allow *viewshed analysis* (given a view point)

Object-Space Algorithm

- The complexity of the *visibility map* can be as high as $\theta(n^2)$, $\Omega(n^2)$ in the worst case.



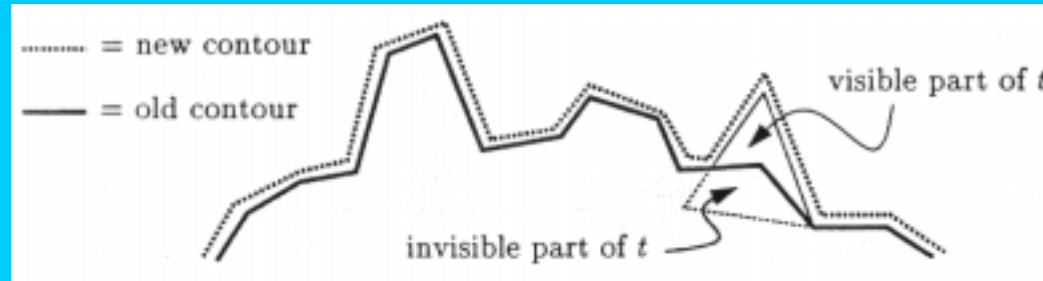
- There are algorithms with $O(n^2)$, which are thus optimal in the worst case. But the worst case usually does not occur in practice.
 - Try and find algorithms whose running time not only depends on n , the number of triangles, but also on k , the complexity of the output.

Object-Space Algorithm

Output-sensitive algorithms:

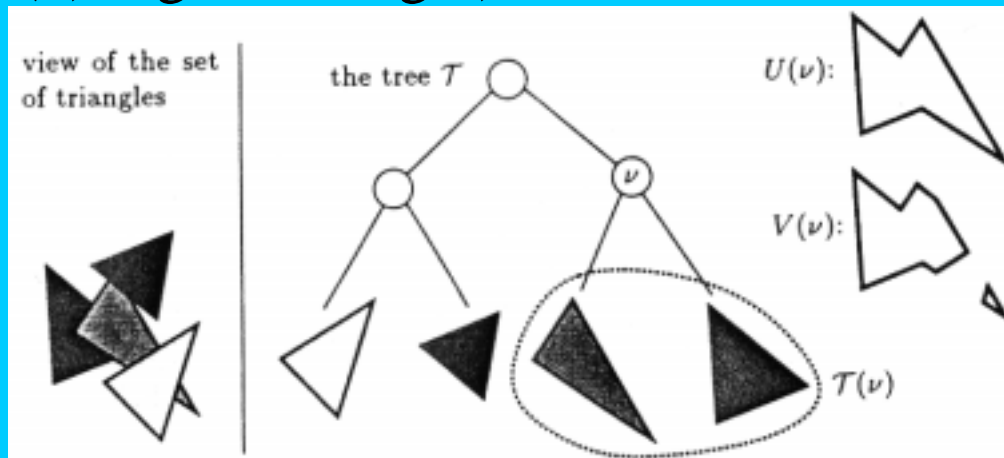
– $O((n+k) \log n \log \log n)$

front-to-back, maintain the contour



– $O(n\alpha(n) \log n + k \log n)$

$\alpha(n)$ - Ackermann's function



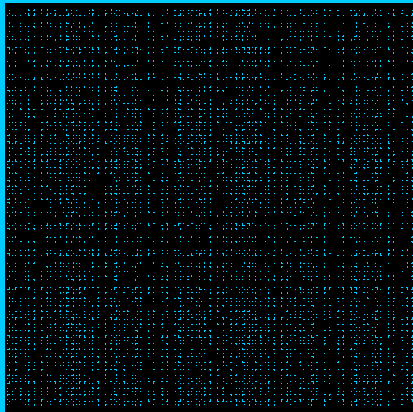
Levels of Detail

- **Purpose:** To reduce the amount of polygons rendered at a time.
- **Based on:** Details perceived only when object is close.
- **How?:** Create a Multi-Resolution Model.

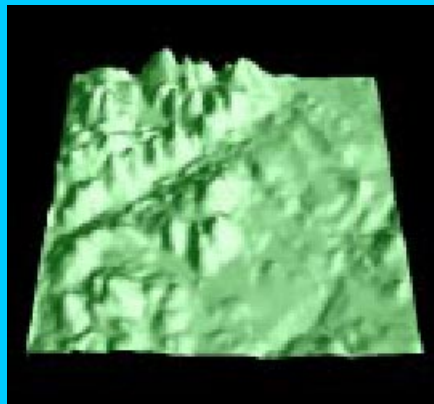
Levels of Detail

- Required Properties:
 - Subtle differences between levels.
 - Use as few levels as possible.
 - Combine neatly to create a variable-resolution representation.

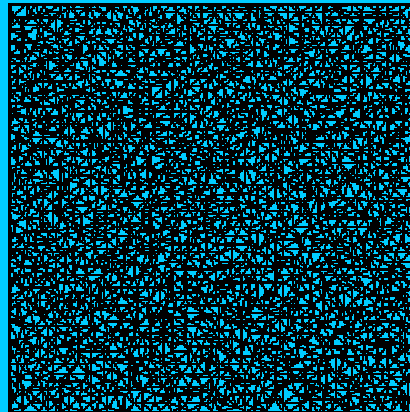
Multi-Resolution



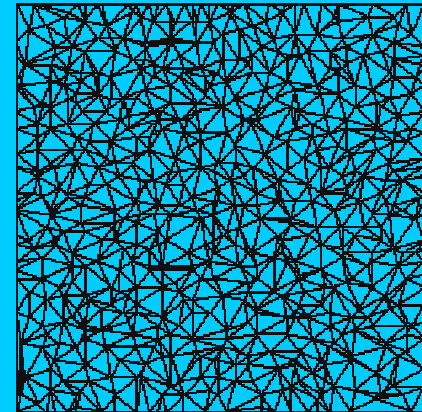
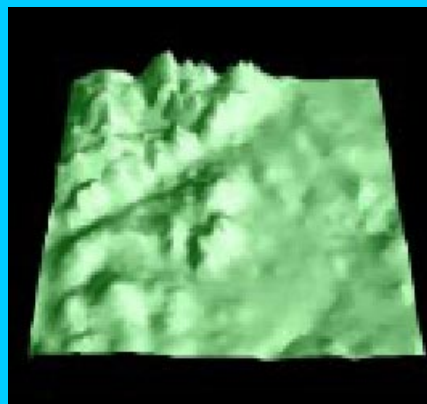
Level 0



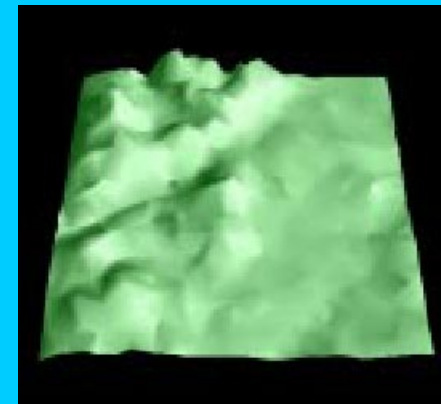
24/03/99



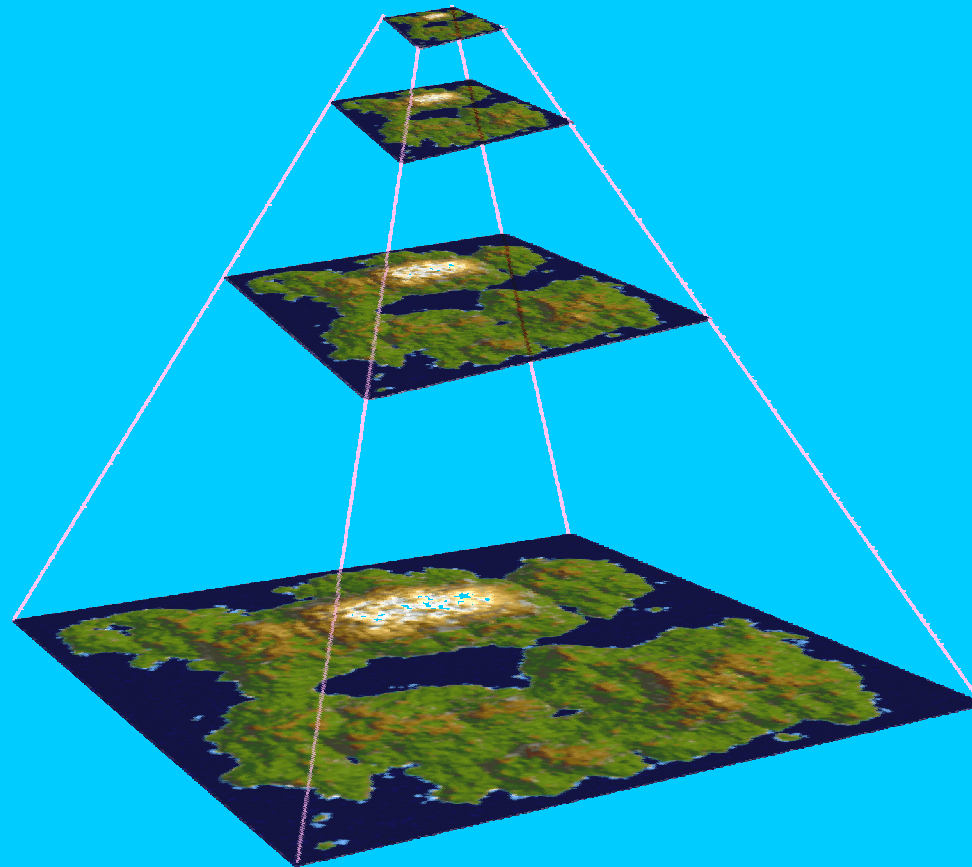
Level 1



Level 2

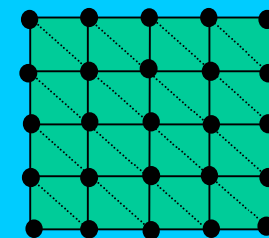
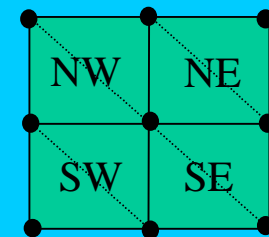
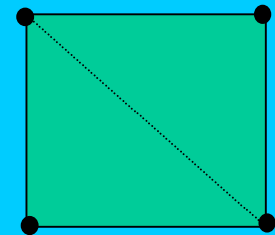
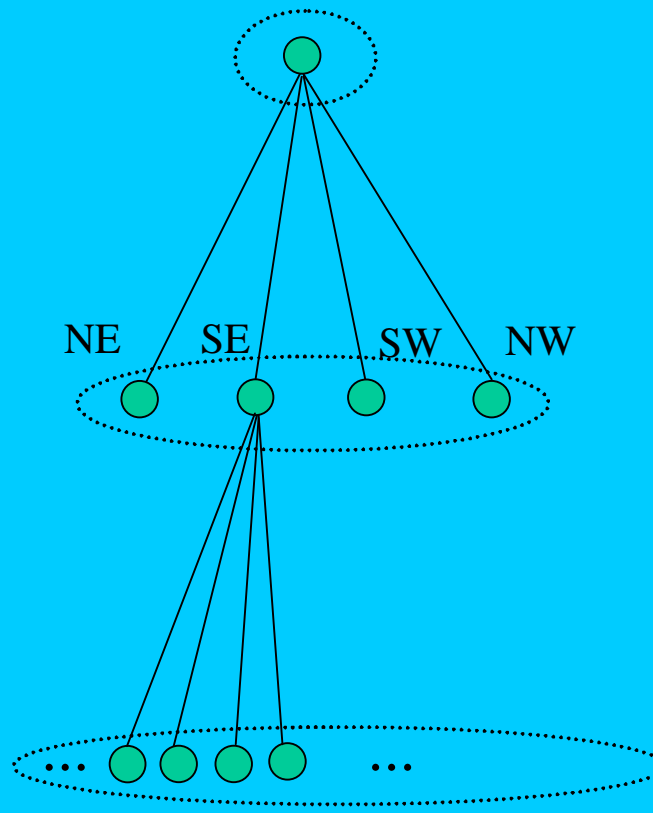


Multi-Resolution Pyramid

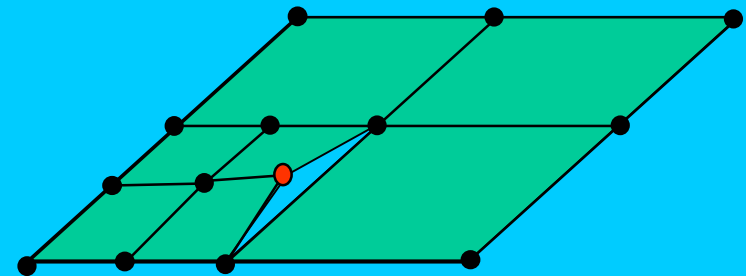
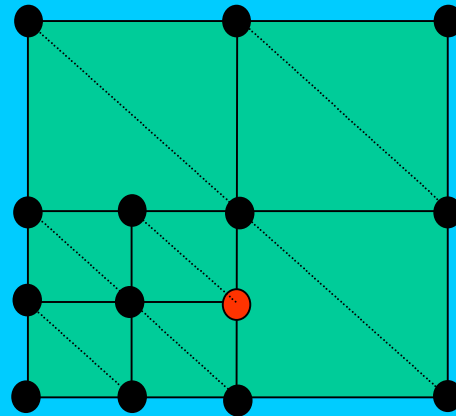
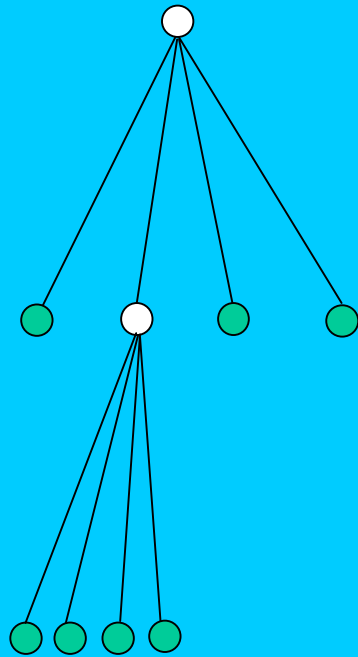


Quadtree

- Root
 - Least Detailed Level
- Next Levels
 - 4 Children:
1 for each
Quadrant of the
Previous Level

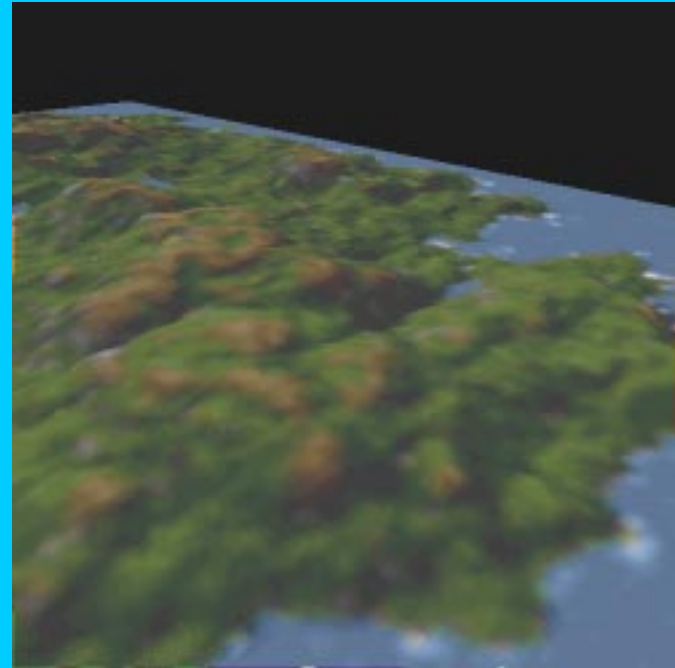
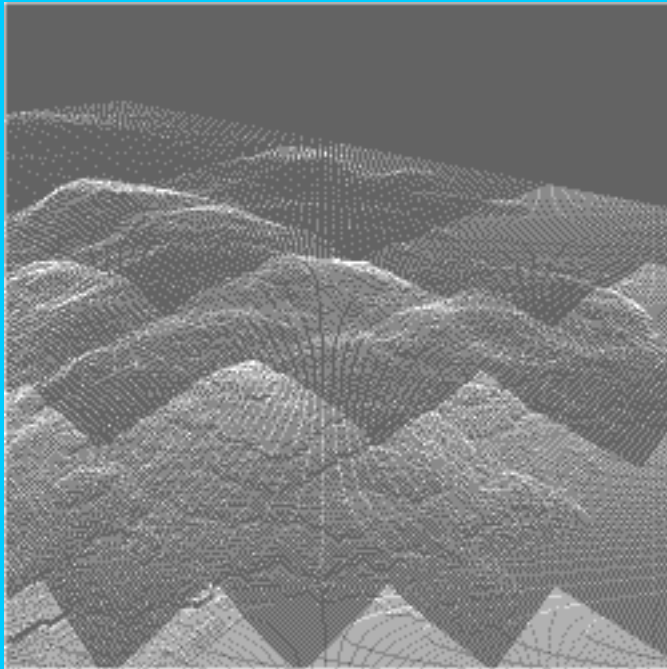


Variable Resolution



sliver problem

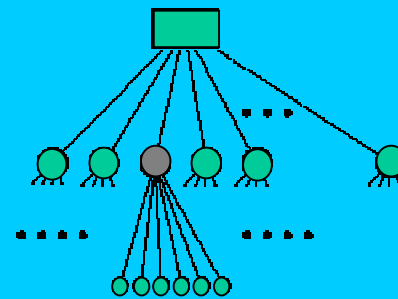
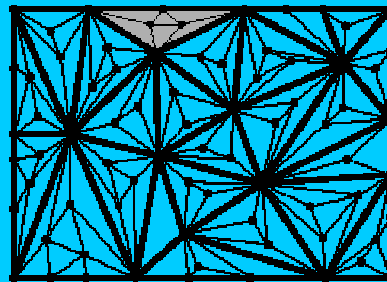
Variable-Resolution



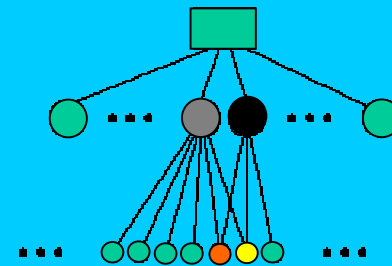
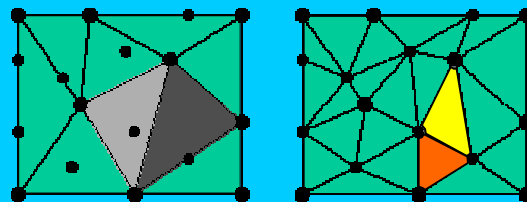
Multi-Resolution for TINs

- Two categories:

- Tree-like hierarchies:



- Delaunay triangulation at all levels.



Tree-like hierarchies

- Construction:
 - Start: Triangulation of a subset of data points.
 - Iteration:
 - Insert data points inside triangles.
 - Calculate new triangulation.
 - Stop:
 - All data points added.
 - Precision criterion met.

Tree-like hierarchies

- Can be modeled as a tree.
(Node=Triangle, Arc=Contained)
- Pros:
 - easy to combine different levels.
- Cons:
 - Triangles at higher levels are very skinny.

Delaunay

- **Construction:**
 - Delaunay triangulation of the set of data points at every level.
- **Represented as directed acyclic graphs.**
(Node=Triangle, Arc=Intersection)
- **Pros:**
 - Maximizes minimum angle.
 - Reduced robustness and aliasing problems.
- **Cons:**
 - Variable resolution representation is not easy.
(Triangles cannot be refined independently.)

Variable Resolution for TINs

