Voxels and Voxelization Algorithms

Two methods of rasterization, raster specific operations

Ir. Pirouz Nourian
- Researcher, Urbanism Department, 3D Geo Information
- PhD candidate & Instructor, AE+T department
- chair of Design Informatics

Dr. Ing. Sisi Zlatanova
- Associate Professor, Urbanism Department, 3D Geo Information
Our proposed extended 3D representations in blue:

- **vector data model**: [data models] data that has discrete boundaries, such as country borders, land parcels, streets and building models.

  A representation of the world using:
  - Points
  - Lines (composed of points)
  - Polygons (composed of lines)
  - Meshes or Polyhedral Surfaces (composed of polygons), which can also be modelled as Breps (boundary representations)

- **raster data model**: [data models] data that varies continuously, as in an aerial photograph, a satellite image, a surface of chemical concentrations, an elevation surface or a continuous voxel model of underground masses.

  A representation of the world using:
  - Raster2D: a surface (bounding rectangle) divided into a regular grid of rectangular cells or pixels
  - Raster3D: a volume (bounding box) divided into a regular grid of cubic cells or voxels

Many new possibilities and also challenges: “3D is not just 2D + 1D; it is much more than that”!
Spatial Analysis Application Areas

What analytic operations can be enabled in 3D GIS?

Continuous models:
- Noise
- Moisture
- Temperature
- Composition
- E.g. Geological Analyses: obtaining iso-surface layers by querying various voxel properties
- Noise Analyses: noise pollution levels...

Discrete models:
- Visual Analyses: view shed analysis, sky view, surveillance and security
- Geodesic Analyses (shortest paths and navigation) for examples in case of drones
- Network Analyses: centrality analyses, graph traversals and searches
Spatial Analysis Application Areas

What analytic operations can be enabled in 3D GIS?

Continuous models:
- Noise
- Moisture
- Temperature
- Composition
- E.g. Geological Analyses: obtaining iso-surface layers by querying various voxel properties
- Noise Analyses: noise pollution levels...

Discrete models:
- Visual Analyses: view shed analysis, sky view, surveillance and security
- Geodesic Analyses (shortest paths and navigation) for examples in case of drones
- Network Analyses: centrality analyses, graph traversals and searches
Voxel Representation

In-depth Representation of 3D [Scalar] Fields

- Scientific & Medical Imagery
- Continuous 3D data
- Volumetric Pixels
Example Applications: Geological models

National model: Geological Formations

Hydrogeological model: more detail

Detailed model: the upper 30-50 meter
Example: GeoTOP

- 3D subsurface model of the Netherlands:
  - Nation-wide model (≈ 41,000 km²)
  - Upper 30 meters
  - Cell size 100 x 100 x 0.5 meters
  - Information per cell:
    - Stratigraphical unit + uncertainty
    - Lithology and sand grain-size classes + uncertainty
    - Hydrological, physical and chemical properties

- Model application:
  - Groundwater and pollution management
  - Land subsidence studies
  - Infrastructural issues

Courtesy TNO
Our Laboratory Prototype
Connecting MonetDB to McNeel’s Rhinoceros® & Grasshopper®

R&D lab setting for “RasterWorksTUD.DLL”

What can be done?

**MonetDB:**
a geospatial database to support 3D GIS operations

**ODBC connection**
Interface between

**MonetDB**
&
**Grasshopper**

**Rhino-GH:**
a parametric CAD environment working as a computational geometry lab & visualization environment

**RASTERWORKS.DLL**
an analytic engine for voxel/raster 3D operations

Topological Voxelization (Laine, 2013) NVIDIA, based on the concept of connectivity target, very elegant mathematically. Implemented √
Topological Rasterization with Semantics from a BIM model [original model]
Topological Rasterization with Semantics from a BIM model [0.5 by 0.5 by 0.5 M]
Topological Rasterization with Semantics from a BIM model [0.4 by 0.4 by 0.4 M]
Topological Rasterization with Semantics from a BIM model [0.2 by 0.2 by 0.2 M]
Topological Rasterization with Semantics from a BIM model [0.1 by 0.1 by 0.1 M]
Pixel & Voxel Adjacencies

For relative positioning and topology information

Why do we need connectivity information?

- Sharing Edges: 4 Neighbours
- Sharing Vertices: 8 Neighbours

- Sharing Faces: 6 Neighbours
- Sharing Edges: 18 Neighbours
- Sharing Vertices: 26 Neighbours

Jian Huang, Roni Yagel, Vassily Filippov, and Yair Kurzion
Voxel Models: N-Adjacency & N-Paths

Representing closures

- Separability: What could be a closure?
- For defining 3D recognizable “objects”
Voxel Models: N-Adjacency & N-Paths

Representing closures for defining 3D recognizable “objects”

- **Connectivity:** how to represent a curve in pixels or voxels
- **Separability:** What could be a closure?
- **Thinness:** how do we ensure the minimality of a raster representation

Images courtesy of Samuli Laine, NVIDIA, 2013
Conclusion: A Pragmatic Point of View

Effectivity & Efficiency Achieved!

Images courtesy of Samuli Laine, NVIDIA, 2013

- Target results is 6-Connected Voxels; No need to intersect S against the full voxel; produces a thin voxelization as $S^d$; **Equivalent to rasterization in three projections! = Speed!**
Sparce Voxel Octree (SVO)

Adaptive Resolution: a more efficient way of representation

- Make smaller voxels if needed depending on the desired LOD
- The position of a voxel is inferred based upon its position relative to other voxels
- Connectivity graph of a voxel model

Octree and Quadtree representation

How do they work?

- **2D Quadtrees** (recursive subdivision of a bounding rectangle into 4 rectangles): divide the shape to *pixels* small enough to represent the required LOD, wherever needed, larger elsewhere.

- **3D Quadtrees** (recursive subdivision of a bounding box into 8 boxes): divide the shape to *voxels* small enough to represent the required LOD, wherever needed, larger elsewhere.
Voxel Representation vs. Brep

3D raster vs 3D vector representation

- Vector Graphics
  - Boundary representation
    - Brep
  - Vectorize
- Rasterize
- Raster Graphics
  - Voxel (Volume) representation
    - Vrep
- Vectorize
Rasterization Algorithms

How to convert vector data consistently to Raster2D or Raster 3D data

- Vector Data
  - Points
  - Lines
  - Polygons
  - Meshes

- Rasterize

- Raster Data
  - Raster 2D
  - Raster3D

Our current research agenda:
elegant systematic ways to convert point clouds, mesh models, polygons and lines into raster models

Current research question: sparse voxel Octree or regular voxel grid?
Vectorization Algorithms

How to convert raster data consistently to Vector2D or Vector3D data

- Vector Data
  - Points
  - Lines
  - Polygons
  - Meshes

- Raster Data
  - Raster 2D
  - Raster3D

- Profiling Algorithms
- Iso-Surface Algorithms
- Iso-Curve Algorithms
- Skeletonization Algorithms
- Segmentation Algorithms

Vectorize
Profiling Algorithms

How to construct Raster2D, Line and Mesh profiles from Raster3D models?

Image courtesy of Oregon State University

Courtesy TNO
Iso-Curve Algorithms

How to construct topography contour **lines** from elevation raster maps?

• Marching Squares_ Studied
• Marching Triangles_ To be prototyped

**Note:** Marching triangle does not bring about ambiguous topological situations, unlike marching squares. Therefore, curves produced by marching triangles are always manifold.
Iso-Surface Algorithms

How to construct contour surfaces from continuous Raster 3D field/function models?

• Marching Cubes (William E. Lorensen and Harvey E. Cline, 1985) _ Studied
• Marching Tetrahedrons (Doi and Koide, 1991) _ Prototyped

• Extracting boundaries using voxel attributes (measures, semantic etc.)

Note: Marching tetrahedrons does not bring about ambiguous topological situations, unlike marching cubes. Therefore, surfaces produced by marching tetrahedrons are always manifold.
Extracting an Isosurface from voxel data

Using the Marching Cubes algorithm

Image courtesy of Paul Bourke
Example: Numeric Query

How to construct contour **meshes** from continuous Raster 3D field/function models?

- Extracting boundaries using voxel attributes (measures, semantic etc.)

**Note:** Marching tetrahedrons does not bring about ambiguous topological situations, unlike marching cubes. Therefore, surfaces produced by marching tetrahedrons are always manifold.
Example: Semantic Query

How to construct contour meshes from continuous Raster 3D field/function models?

- Extracting boundaries using voxel attributes (measures, semantic etc.)

Note: Marching tetrahedrons does not bring about ambiguous topological situations, unlike marching cubes. Therefore, surfaces produced by marching tetrahedrons are always manifold.
Framework

Data Conversions

Vector Data
- 0D: Points
- 1D: Polylines
- 2D: Meshes
- 3D: Solids

Raster Data
- Raster 2D
- Raster3D

How to voxelize vector geometries while preserving topological relations and semantic properties?
Kaufman (Tel Aviv University, 1987)
Laine (NVIDIA, 2013)

How to reduce dimensionality and make some sense by making vector data models out of rasters?
- Iso-Surface
- [Skeletonization]
- [Segmentation]
- Surface Reconstruction

some other vector data

Challenge the future
# Raster Specific Functions

What a user would expect to find in a raster tool suite

## Topological
- Is Neighbour? (Boolean)
- N-Neighbourhood; Has N-Neighbours? (Voxel Set; Boolean)
- Outer-Layer (Voxel Set)
- Shortest Path (Voxel Set; possibly a polyline)

## Geometric
- Volume (Float)
- Distance [length of shortest path] (Float)
- Area [of the outer layer] (Float)

## Level Sets
- Plane Cut & Slice [X,Y,Z] (image/pixel collection/colored quadrangular mesh)
- Iso-Curve (Polyline)
- Iso-Surface (Mesh)

## Morphological
- Boolean [Union, Intersection, Difference] (Voxel Grid/3D Raster)
- Dilation & Erosion (Voxel Grid/3D Raster)
- Gradient (Voxel Grid/3D Raster)
- Opening & Closing (Voxel Grid/3D Raster)
- Skeleton [Medial Axis, Medial Surface]

## Approximations
- [Tetrahedral Tessellation]
- [Barycentric Interpolation on Lines, Triangles and Tetrahedrons]
- Interpolation & Extrapolation
Is voxelization trivial?

What would go wrong if we just intersect the voxels with the object?

- Extremely inefficient [processing] to consider the full voxel;
- The result will be an unnecessarily ‘thick’ voxel collection (inefficient storage);
- There is no direct control on connectivity (cohesion) level; thus no guarantee on full correctness of topological relations and semantic properties!
Open Problems

- How to rasterize (voxelize) big datasets?
- How to distinguish in and out in raster models?
- How to resolve conflicting semantics?

With the help of database? A storage issue?

Heuristics? Other datasets? User Input? Intelligence (pattern recognition)?

Rather philosophical!?
Rasterization* algorithm

- **3D Scan Conversion** (Kaufman, 1987), i.e. Extended 2D *scan conversion* into 3D
  - Very efficient but 6-Connectivity cannot be reached, therefore 26 separability not assured, difficult to generalize the method to different geometric primitives

- **Topological Voxelization** (Laine, 2013) NVIDIA, based on the concept of connectivity target, very elegant mathematically. Can be efficiently implemented.

**Rasterization** = Turn a continuous input in $\mathbb{R}^3$ into a discrete output in $\mathbb{Z}^3$ (Laine, 2013)

**Topological** instead of **geometrical** approach: Intuitively, things of Boolean nature: connectivity, separability, intersections, etc. (ibid)
To get an idea of 3D Scan Conversion

TU Delft, 3D Geo Info, after Laine NVIDIA
### Criteria for comparison

- Connectivity, Separability, Thinness, Semantics, Complexity

<table>
<thead>
<tr>
<th></th>
<th>Connectivity</th>
<th>Separability</th>
<th>Thinness</th>
<th>Semantics</th>
<th>Iteration Variable*</th>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kaufman, 1987</td>
<td>26-Connected</td>
<td>6-Separable</td>
<td>guaranteed</td>
<td>Based on vector primitives</td>
<td>Vector primitives</td>
<td>Different per input data types (triangle, line, etc.)</td>
</tr>
<tr>
<td>(3D Scan Conversion)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Laine, 2013</td>
<td>26-Connected</td>
<td>6-Spearable</td>
<td>guaranteed</td>
<td>Based on vector primitives</td>
<td>voxel</td>
<td>Only one for 1D input and one for 2D input</td>
</tr>
<tr>
<td>(Topological)</td>
<td>&amp; 6-Connected</td>
<td>&amp; 26-Separable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* This means that the theoretical performance of the method of Laine can be dependent on the number of voxels corresponding to the object; therefore, in contrast to the method of Kaufman, the performance is not affected by number of triangles and the method will never produce duplicate voxels and never misses any voxel.
Quality Criteria

Decisive Effectivity/Quality Criteria!

• Using Kaufmann’s method, we cannot opt for 6-connectivity (26-separability); this can be limiting in applications such as ray-tracing where a 26 connected ray is to intersect a rasterized object. If the object is not 26 separable we cannot guarantee a hit!

• Generalization to vector formats other than triangular mesh would not be easy as different algorithms will be needed for different primitive inputs!

   **Topological Voxelization is superior to 3D Scan Conversion!**
Control on Connectivity Level

Stanford Bunny, scaled 0.1: Mesh (V:34834  F:69664)

6-Connected Stanford Bunny, 1265 voxels (0.8,0.8,0.8)

26-Connected Stanford Bunny, 755 voxels (0.8, 0.8, 0.8)
26-Connected Voxels of various XYZ dimensions: 0.8x0.8x0.1
6-Connected Voxels of various XYZ dimensions: 0.8x0.8x0.8
Topological Voxelization (Laine, 2013) [from NVIDIA research]
C# and VB.NET implementation by Pirouz Nourian

1. Input Triangular Mesh
2. Bounding Box Voxels
3. R-Tree Applied
4. What happens to a mesh triangle?
5. ‘targets’ formed for those which ‘possibly’ intersect with the triangle
Topological Voxelization (Laine, 2013) [from NVIDIA research] C# and VB.NET implementation by Pirouz Nourian

1. Input Triangular Mesh
2. Bounding Box Voxels
3. R-Tree Applied
4. What happens to a mesh triangle?
5. ‘targets’ formed for those which ‘possibly’ intersect with the triangle
1. Input Triangular Mesh
2. Bounding Box Voxels
3. R-Tree Applied
4. What happens to a mesh triangle?
5. ‘targets’ formed for those which ‘possibly’ intersect with the triangle

Topological Voxelization (Laine, 2013) [from NVIDIA research]
C# and VB.NET implementation by Pirouz Nourian
Challenge the future

1. Input Triangular Mesh
2. Bounding Box Voxels
3. R-Tree Applied
4. one triangle and its candidate voxels
5. ‘targets’ formed for those which ‘possibly’ intersect with the triangle

Topological Voxelization (Laine, 2013) [from NVIDIA research]
C# and VB.NET implementation by Pirouz Nourian
Topological Voxelization (Laine, 2013) [from NVIDIA research]
C# and VB.NET implementation by Pirouz Nourian

1. Input Triangular Mesh
2. Bounding Box Voxels
3. R-Tree Applied
4. one triangle and its candidate voxels
5. ‘targets’ formed for those which ‘possibly’ intersect with the triangle
“RasterWorksTUD.DLL”
Contents of “RasterWorksTUD.DLL”

The current structure

Questions:

Voxel=Point3D + Measure + Semantic? (as point-cloud item)

Voxel=Point3I + Bounding Box + Measure + Semantic?

Voxel=Boolean + Bounding Box + Measure + Semantic?

SQL Create Index? (list indices to i,j,k)
Rubroek, Rotterdam

- Rubroek 10 Centimetre resolution, empty space excluded,
- 59654004 voxels, out of 13229 mesh objects, 857x643x50 Meters
- Obtained from CityGML LOD2
- Semantics represented in DXF layers
- Colours represent semantics: roof, walls, ground surface
Rubroek, Rotterdam

- Rubroek 40 Centimetre resolution,
- empty space excluded
- Obtained from CityGML LOD2
- Semantics represented in DXF layers
- Colours represent semantics: roof, walls, ground
OTB Building, Delft, Rasterized from BIM

Empty space voxelized

- 2D floor plans processed in ArcGIS; later converted to DXF
- Semantics represented in DXF layers
- Semantics stored in voxel representation
- Resolution 0.1 Meter in XYZ directions
- Triangulated first, voxelized next;

<table>
<thead>
<tr>
<th>OTB Building, X Section</th>
<th>OTB Building, Y Section</th>
<th>OTB Building, Z Section</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
</tbody>
</table>
Quality Criteria

Coarse resolution models have conflicting semantics in edges!

Rasterized from CityGML file of Agniesebuurt: 0.4 M and 0.1 M
**Raster Queries and SQL commands**

**Tested with MonetDB**

**Bounding Box Selection**

```
CREATE TABLE voxels_within_box (x double, y double, z double, m double, s int);
INSERT INTO voxels_within_box SELECT * FROM Rubroek_voxels
WHERE X BETWEEN 92981 AND 93131 AND Y BETWEEN 438112 AND 438260 AND
Z BETWEEN 0 AND 46; //takes almost all voxels in the collection Runroek-Voxels
```

**Semantic Selection**

```
CREATE TABLE semantic_collection (x double, y double, z double, m double, s int);
INSERT INTO semantic_collection SELECT * FROM Rubroek_voxels WHERE s=3
AND Z>3; // s=3 has the meaning of ‘ground’ semantic label, therefore this query
returns all voxels pertaining to ground objects
```

**Update Entries**

```
UPDATE Rubroek_voxels SET m = ROUND(Z/3,0); this command sets a measure for
every voxel as to its height; for a flat neighborhood it says at which floor the voxel
is located, provided that h=0 is ground
```

**Insert Entries**

```
How to generate empty voxels on demand? Empty voxels are needed for producing
slices and level sets from raster 3D models.
```
References: