Reconstruction of Voxels from Sensor Data

Geometric Modeling
COS 598d Fall 2000

Volume Representations

- Voxels = uniform, orthogonal grid
  - Binary (empty/full)
  - Float: density / color / distance to surface, etc.
    - $0^\text{th}$ (nearest) vs. $1^\text{st}$ order (tri-linear) interpolation
  - Can be RLE’d
- Octrees = hierarchical space subdivision
  - Cube nodes: black/ white/ gray (split 1-to-8)
Data acquisition / conversion

- MRI / CT
- Range Data
- Images (silhouettes)
- Polygons

Voxels / Octrees

Space carving
Scan conversion

Marching cubes → Mesh

3D “Scan-Conversion”

- Binary vs. anti-aliased
- Extension of scanline filling
- Scan-convert boundary, then flood fill
Problems

• “simple” and “separating”
• In 2D: 4/8-adjacency:
  neighbors share edges / edges OR vertices
  4-connected is 8-separating and vice-versa
• 3D: 6/18/26-adjacency

3D Antialiasing

• [Wang & Kaufman’93]
  – weighted integral of “stuff” inside ball
  – radius 3 voxels => filtering
  – weight function = cone (“hat”)
  – precompute for all primitive types & discrete
    set of distances
3D Antialiasing

[Oomes et al.’97]

- sample implicit function defined on volume
  - point => gaussian fall-off blob
  - line => “cylinder”, radial e^{-d} fall-off, erf for ends
  - triangle =>

\[
\int_{\Omega} \exp\left(-\frac{(x-x_0)^2+(y-y_0)^2+(z-z_0)^2}{2\sigma^2}\right) \left[ \text{erf}\left(\frac{\sqrt{2}z}{\sigma}\right) - \text{erf}\left(\frac{-\sqrt{2}z}{\sigma}\right) \right] dz
\]

- Expensive but solves some problems.

Space Carving

- Done hierarchically using octrees
- Cubes can be “maybe full” / mixed / empty
- Start w/ whole volume presumed full
- “Carve” away portions of “seen” space
- If cube partially empty, subdivide
- Carve 1 level using all views before subdiv.
Octrees from silhouettes

- [Szeliski ’93]
  - Map cube to image & see if full
  - Fast test on square bounding box (could use summed area tables too)
  - Subdivide after full rotation
  + Simple & fast
  - Result is “line hull”

Octrees from range data

- [Pulli et al. ’97] Project cubes to depth map

  - All points in hex. behind far cube vtx => empty
  - If points in hexagon in front => cube inside obj
  - Else, cube is boundary (subdivide)
  - Can accelerate using 2D min/max quadtrees.
  - Hint for voxel grid resolution
Marching Cubes

Cases
Matching up neighboring voxels