Marching Cubes

Sara McMains

ME 290-R
Marching Cubes

Motivation

• Visualization for medical apps

Input

• Regular grid of points
• Density values at each point
Data acquisition

- MRI (Magnetic Resonance Imaging)
  - Excitation of water molecules
- CT scan (Computer Tomography)
  - Absorption of x-rays
- Ultrasound
  - Backscatter strength
Goal

Display “iso-surface”

- Surface of constant density
  - Assumption: sampling from a continuous such surface
- Medical visualization
  - Bone, flesh, organ densities differ
  - Operator selects desired density
Display

Outputs triangles

- Graphics hardware optimized for triangles
  - Today, everyone has graphics cards
  - Back in the “dark ages” had to do it in software

Surface normal for each vertex

- Improve rendered appearance via Gouraud shading
Gouraud Shading

A simple, effective computer graphics hack
Make low-detail surfaces appear smooth
Standard Flat Shading

material + lighting + viewing angle -> color

• Same color for whole triangle
Phong Shading

Pretend surface normal varies across triangle

- Implementation
  - Store a surface normal at each vertex
  - Interpolate them to calculate each pixel color
Gouraud Shading

Simplification of Phong shading

- Interpolate vertex colors instead of normals
Gouraud Shading
Back to Marching Cubes

Basic idea:

• Look at one “cube” of 8 samples at a time
• Determine if each corner inside or outside volume
  • Density above threshold => 0 label
  • Density below threshold => 1 label
• Pattern of labels tells topology of intersection

*Calculation of topology and geometry separated*
Cases

256 (2^8) cases total

- Build a look-up table
  - Index: ordered 8-bits of in/out labels from cube corners
  - Output: which edges intersected, triangles formed
Cases

Only 15 patterns
Cases

Only 15 patterns

- Use symmetry
Cases

Only 15 patterns

- Use symmetry
Geometry calculations

Edge intersection positions

- Linear interpolation of density values at corners

Isosurface density 5
Geometry calculations

Edge intersection positions
  - Linear interpolation of density values at corners

Label = 1 (out)

Label = 0 (in)
Geometry calculations

Edge intersection positions

- Linear interpolation of density values at corners

Label = 1 (out)

Intersection position for isosurface density 5

Label = 0 (in)
Geometry Calculations

Vertex normals

- Want to set to normal to iso-surface
- Gradient direction = normal direction for iso-surfaces
- Calculate gradient at each corner
  - Look at 6 neighbors
- Interpolate to edge intersections
Efficiency

Internal cube edges shared

- 12 edges/cube
- 4 cubes/edge

\{ 3 new edges per internal cube \}
Booleans

Boolean evaluation also uses in/out labeling (and on)

- Implement Booleans on cube index
- Use truth tables to combine 0, 255, “in-between”
- For in-between/in-between, clip triangles
Memory Usage

Just need four slices of data in memory
  • No problem for normal data sets

Small memory footprint good indicator of possible parallelism
Topology Problems

E.g. non-manifold output
Topology Problems

E.g. holes in output
Topology Problems

E.g. holes in output
Ambiguity of configurations

2D Example
Fixes

Simple approaches

• Consider adjacent cubes
  • Helps some, not all cases
• Divide into Tetrahedra
  • Make consistent with adjacent elements
  • Correctness not guaranteed
Fixes

Interpolation approaches calculate additional vertices or values

• Add center point & tetrahedralize
• Subdivision
  • Down to pixel resolution, e.g.
  • Dividing Cubes [Cline et al. 1988]
• Fit higher order curves
  • Curve orientation disambiguates
Questions?