Subdivision Surfaces

Thomas Funkhouser
Princeton University
COS 426, Spring 2004

Surfaces

• What makes a good surface representation?
  ◦ Accurate
  ◦ Concise
  ◦ Intuitive specification
  ◦ Local support
  ◦ Affine invariant
  ◦ Arbitrary topology
  ◦ Guaranteed continuity
  ◦ Natural parameterization
  ◦ Efficient display
  ◦ Efficient intersections

H&B Figure 10.46

Modeling

• How do we ...
  ◦ Represent 3D objects in a computer?
  ◦ Construct 3D representations quickly/easily?
  ◦ Manipulate 3D representations efficiently?

Different representations for different types of objects

3D Object Representations

• Raw data
  ◦ Voxel
  ◦ Point cloud
  ◦ Range image
  ◦ Polygons

• Solids
  ◦ Octree
  ◦ BSP tree
  ◦ CSG
  ◦ Sweep

3D Object Representations

• Raw data
  ◦ Voxel
  ◦ Point cloud
  ◦ Range image
  ◦ Polygons

• Solids
  ◦ Octree
  ◦ BSP tree
  ◦ CSG
  ◦ Sweep

• Surfaces
  ◦ Mesh
  ◦ Subdivision
  ◦ Parametric
  ◦ Implicit

• High-level structures
  ◦ Scene graph
  ◦ Skeleton
  ◦ Application specific

• High-level structures
  ◦ Scene graph
  ◦ Skeleton
  ◦ Application specific

H&B Figure 10.46

Surfaces

H&B Figure 10.46
### Subdivision
- How do you make a smooth curve?

![Image](image1)

### Key Questions
- How refine mesh?
  - Aim for properties like smoothness
- How store mesh?
  - Aim for efficiency for implementing subdivision rules

![Image](image2)

### Loop Subdivision Scheme
- How position new vertices?
  - Choose locations for new vertices as weighted average of original vertices in local neighborhood

![Image](image3)

### Subdivision Surfaces
- Coarse mesh & subdivision rule
  - Define smooth surface as limit of sequence of refinements

![Image](image4)

### Loop Subdivision Scheme
- How refine mesh?
  - Refine each triangle into 4 triangles by splitting each edge and connecting new vertices

![Image](image5)

### Loop Subdivision Scheme
- Rules for extraordinary vertices and boundaries:
  - a. Mark for odd vertices
  - b. Mark for even vertices

![Image](image6)
Loop

• How to choose $\beta$?
  - Analyze properties of limit surface
  - Interested in continuity of surface and smoothness
  - Involves calculating eigenvalues of matrices
  - Original Loop
    $$\beta = \frac{1}{2} \left( \frac{1}{n} - \frac{1}{4} \left( \frac{1}{n} + \frac{1}{2} \cos \frac{2\pi}{n} \right)^2 \right)$$
  - Warren
    $$\beta = \left\{ \begin{array}{ll}
    \frac{1}{n} & n > 3 \\
    \frac{1}{3} & n = 3 
  \end{array} \right.$$
Key Questions

• How refine mesh?
  • Aim for properties like smoothness

• How store mesh?
  • Aim for efficiency for implementing subdivision rules

Polygon Meshes

• Mesh Representations
  • Independent faces
  • Vertex and face tables
  • Adjacency lists
  • Winged-Edge

Independent Faces

• Each face lists vertex coordinates
  • Redundant vertices
  • No topology information

Vertex and Face Tables

• Each face lists vertex references
  • Shared vertices
  • Still no topology information

Adjacency Lists

• Store all vertex, edge, and face adjacencies
  • Efficient topology traversal
  • Extra storage

Partial Adjacency Lists

• Can we store only some adjacency relationships and derive others?
**Triangle Meshes**

- Relevant properties:
  - Exactly 3 vertices per face
  - Any number of faces per vertex

- Useful adjacency structure for Loop subdivision:
  - Do not represent edges explicitly
  - Faces store refs to vertices and neighboring faces
  - Vertices store refs to adjacent faces and vertices

**Assignment 3**

- Edit coarse mesh while display subdivided mesh

**Assignment 3**

- Store hierarchy of meshes
  - Full triangle mesh at every level
  - Vertices references to counterparts
  - One level up and one level down
  - Enables efficient re-positioning of mesh vertices after interactive dragging
Summary

- Advantages:
  - Simple method for describing complex surfaces
  - Relatively easy to implement
  - Arbitrary topology
  - Local support
  - Guaranteed continuity
  - Multiresolution

- Difficulties:
  - Intuitive specification
  - Parameterization
  - Intersections