3D Object Representation

Michael Kazhdan

(600.457)
3D Objects

How can this object be represented in a computer?
3D Objects

This one?

H&B Figure 10.46
How about this one?

Stanford Graphics Laboratory
3D Objects

This one?

H&B Figure 9.9
3D Objects

This one?
3D Object Representations

• Raw data
  ○ Point cloud
  ○ Range image
  ○ Polygon soup

• Surfaces
  ○ Mesh
  ○ Subdivision
  ○ Parametric
  ○ Implicit

• Solids
  ○ Voxels
  ○ BSP tree
  ○ CSG
  ○ Sweep

• High-level structures
  ○ Scene graph
  ○ Skeleton
  ○ Application specific
Point Clouds

• Unstructured set of 3D point samples
  ◦ Acquired from random sampling, particle system implementations, etc.
Range Images

- An image storing depth instead of color
  - Acquired from 3D scanners

Range Image  Tesselation  Range Surface
Polygon Soups

- Unstructured set of polygons
  - Created with interactive modeling systems, combining range images, etc.
3D Object Representations

• Raw data
  ○ Point cloud
  ○ Range image
  ○ Polygon soup

• Surfaces
  ○ Mesh
  ○ Subdivision
  ○ Parametric
  ○ Implicit

• Solids
  ○ Voxels
  ○ BSP tree
  ○ CSG
  ○ Sweep

• High-level structures
  ○ Scene graph
  ○ Skeleton
  ○ Application specific
(Manifold) Meshes

- Connected set of polygons (usually triangles)
Subdivision Surfaces

• Coarse mesh & subdivision rule
  ◦ Define smooth surface as limit of sequence of refinements
Parametric Surfaces

- Tensor product spline patches
  - Careful use of constraints to maintain continuity

FvDFH Figure 11.44
Implicit Surfaces

- Points satisfying: \( F(x, y, z) = 0 \)
3D Object Representations

• Raw data
  ○ Point cloud
  ○ Range image
  ○ Polygon soup

• Surfaces
  ○ Mesh
  ○ Subdivision
  ○ Parametric
  ○ Implicit

• Solids
  ○ Voxels
  ○ BSP tree
  ○ CSG
  ○ Sweep

• High-level structures
  ○ Scene graph
  ○ Skeleton
  ○ Application specific
Voxels

- Uniform grid of volumetric samples
  - Acquired from CT, MRI, etc.
BSP Trees

- Binary space partition with solid cells labeled
  - Constructed from polygonal representations
Constructive Solid Geometry (CSG)

• Hierarchy of boolean set operations (union, difference, intersect) applied to simple shapes

FvDFH Figure 12.27

H&B Figure 9.9
Sweep Surfaces

- Solid swept by curve along trajectory
3D Object Representations

- **Raw data**
  - Point cloud
  - Range image
  - Polygon soup

- **Surfaces**
  - Mesh
  - Subdivision
  - Parametric
  - Implicit

- **Solids**
  - Voxels
  - BSP tree
  - CSG
  - Sweep

- **High-level structures**
  - Scene graph
  - Skeleton
  - Application specific
Scene Graphs

- Union of objects at leaf nodes

Bell Laboratories

avalon.viewpoint.com
Skeletons

- Graph of curves with radii
Application Specific

Apo A-1
(Theoretical Biophysics Group,
University of Illinois at Urbana-Champaign)

Architectural Floorplan
Computational Differences

• Efficiency
  ◦ Combinatorial complexity
  ◦ Space/time trade-offs
  ◦ Numerical accuracy/stability

• Simplicity
  ◦ Ease of acquisition
  ◦ Hardware acceleration
Surfaces

• What makes a good surface representation?
  ○ Concise
  ○ Local support
  ○ Affine invariant
  ○ Arbitrary topology
  ○ Guaranteed smoothness
  ○ Natural parameterization
  ○ Efficient display
  ○ Efficient intersections

H&B Figure 10.46
Surfaces

- What makes a good surface representation?
  - Concise
  - Local support
  - Affine invariant
  - Arbitrary topology
  - Guaranteed smoothness
  - Natural parameterization
  - Efficient display
  - Efficient intersections

Not Local Support
Surfaces

• What makes a good surface representation?
  ○ Concise
  ○ Local support
  ○ Affine invariant
  ○ Arbitrary topology
  ○ Guaranteed smoothness
  ○ Natural parameterization
  ○ Efficient display
  ○ Efficient intersections
Surfaces

• What makes a good surface representation?
  ○ Concise
  ○ Local support
  ○ Affine invariant
  ○ Arbitrary topology
  ○ Guaranteed smoothness
  ○ Natural parameterization
  ○ Efficient display
  ○ Efficient intersections

Topological Genus Equivalences
Surfaces

• What makes a good surface representation?
  ◦ Concise
  ◦ Local support
  ◦ Affine invariant
  ◦ Arbitrary topology
  ◦ Guaranteed smoothness
  ◦ Natural parameterization
  ◦ Efficient display
  ◦ Efficient intersections
Surfaces

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

A Parameterization (not necessarily natural)
Surfaces

• What makes a good surface representation?
  ◦ Concise
  ◦ Local support
  ◦ Affine invariant
  ◦ Arbitrary topology
  ◦ Guaranteed smoothness
  ◦ Natural parameterization
  ◦ Efficient display
  ◦ Efficient intersections
Subdivision Surfaces

• Properties:
  ◦ Concise
  ◦ Local support
  ◦ Affine invariant
  ◦ Arbitrary topology
  ◦ Guaranteed continuity
  ◦ Natural parameterization
  ◦ Efficient display
  ◦ Efficient intersections
Subdivision

• How do you make a smooth curve?

We want to “smooth out” severe angles
Subdivision

- How do you make a smooth curve?

We want to “smooth out” severe angles
Subdivision Surfaces

• Coarse mesh & subdivision rule
  ◦ Define smooth surface as limit of sequence of refinements

Note that the subdivision surface does not have to interpolate the original vertices.
Key Questions

• How to subdivide the mesh?
  ◦ Aim for properties like smoothness

• How to store the mesh?
  ◦ Aim for efficiency of implementing subdivision rules
General Subdivision Scheme

• How to subdivide mesh?
  
  Two parts:
  
  » Refinement:
    – Add new vertices and connect (topological)
  
  » Smoothing:
    – Move vertex positions (geometric)
Loop Subdivision Scheme

• How to subdivide mesh?

  Refinement:
  » Subdivide each triangle into 4 triangles by splitting each edge and connecting new vertices

Zorin & Schroeder
SIGGRAPH 99
Course Notes
Loop Subdivision Scheme

• How to subdivide mesh:
  Refinement
  Smoothing:
    » Existing Vertices: Choose new location as weighted average of original vertex and its neighbors

Existing vertex being moved from one level to the next
Loop Subdivision Scheme

• General rule for moving existing interior vertices:

New\_position = (1 - k\beta)original\_position + \text{sum}(\beta \times each\_original\_vertex)

What about vertices that have more or less than 6 neighboring faces?
Loop Subdivision Scheme

• General rule for moving existing interior vertices:

$$\text{New}_{\text{position}} = (1 - k\beta) \text{original}_{\text{position}} + \sum \beta \text{each}_{\text{original}}$$

$$0 \leq \beta \leq \frac{1}{k}$$:

• As $\beta$ increases, the contribution from adjacent vertices plays a more important role.

• If $\beta = 0$, the subdivision is interpolatory.
Where do existing vertices move?

- How to choose $\beta$?
  - Analyze properties of limit surface
  - Interested in continuity of surface and smoothness
  - Involves calculating eigenvalues of matrices

  » Original Loop
  \[
  \beta = \frac{1}{2} \left( \frac{5}{8} - \left( \frac{3}{8} + \frac{1}{4} \cos \frac{2\pi}{k} \right)^2 \right)
  \]

  » Warren
  \[
  \beta = \begin{cases} 
  \frac{3}{8k} & k > 3 \\
  \frac{3}{16} & k = 3 
  \end{cases}
  \]
Loop Subdivision Scheme

• How to subdivide mesh:

  Refinement

  Smoothing:

  » Inserted Vertices: Choose location as weighted average of original vertices in local neighborhood

New vertex being inserted

Zorin & Schroeder
SIGGRAPH 99
Course Notes
Boundary Cases?

What about *boundary vertices and boundary edges*:?

- Existing vertex adjacent to a missing triangle
- New vertex bordered by only one triangle
Boundary Cases?

- Rules for *boundary vertices* and *boundary edges*:
Loop Subdivision Scheme

Limit surface has provable smoothness properties!
Loop Subdivision Scheme

Geri’s Game, *Pixar*