Polygonal Meshes

Thomas Funkhouser
Princeton University
COS 526, Fall 2002

Outline

- Polygonal meshes
- Mesh data structures
- Mesh simplification
Polyhedral Meshes

Outline

- Polygonal meshes
  - Mesh data structures
- Mesh simplification
Mesh Data Structures

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

Independent Faces

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

<table>
<thead>
<tr>
<th>FACE TABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
</tr>
<tr>
<td>F2</td>
</tr>
<tr>
<td>F3</td>
</tr>
</tbody>
</table>
**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?

Winged Edge

- Adjacency encoded in edges
  - All adjacencies in O(1) time
  - Little extra storage (fixed records)
  - Arbitrary polygons
Winged Edge

- Example:

![Winged Edge Example Diagram]

<table>
<thead>
<tr>
<th>VERTEX TABLE</th>
<th>EDGE TABLE</th>
<th>FACE TABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1 X1 Y1 Z1 e1</td>
<td>e1 V1 V3 F1</td>
<td>F1 e1 e3 e5</td>
</tr>
<tr>
<td>V2 X2 Y2 Z2 e6</td>
<td>e2 V1 V3 F1</td>
<td>F1 e1 e3 e5</td>
</tr>
<tr>
<td>V3 X3 Y3 Z3 e3</td>
<td>e3 V1 V3 F1</td>
<td>F1 e1 e3 e5</td>
</tr>
<tr>
<td>V4 X4 Y4 Z4 e5</td>
<td>e4 V2 V3 F2</td>
<td>F2 e2 e4 e6</td>
</tr>
<tr>
<td>V5 X5 Y5 Z5 e6</td>
<td>e5 V2 V3 F2</td>
<td>F2 e2 e4 e6</td>
</tr>
</tbody>
</table>

Triangle Meshes

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

- Easy adjacency structure
  - Faces store refs to vertices and neighboring faces
  - Can find most adjacencies in constant time

![Triangle Meshes Example Diagram]
Outline

• Polygonal meshes
• Mesh data structures

Mesh simplification

Mesh Simplification

Triangles:
41,855
27,970
20,922
12,939
8,385
4,766

Division, Viewpoint, Cohen
Mesh Simplification Goals

- Reduce number of polygons
  - Faster rendering
  - Less storage
  - Simpler manipulation

- Desirable properties
  - Generality, efficiency, scalability
  - Produces “good” approximation

Simplification Algorithms

- Measure cost of possible decimation operations according to error measure
- Place operations in queue according to error
- Perform operations in queue successively
  - After each operation, re-evaluate error metrics
Mesh Simplification Operations

- General idea:
  - Each operation simplifies model by small amount
  - Apply many operations in succession

- Types of operations
  - Vertex cluster
  - Vertex remove
  - Edge collapse
  - Vertex pair

Vertex Cluster

- Method
  - Merge vertices based on proximity
  - Triangles with repeated vertices become edge or point

- Properties
  - General and robust
  - Not usually attractive
**Vertex Remove**

- **Method**
  - Remove vertex and adjacent faces
  - Fill hole with new triangles (reduction of 2)

- **Properties**
  - Requires manifold surface around vertex
  - Preserves local topological structure
  - Typically more attractive

![Vertex Remove Diagram](image)

**Edge Collapse**

- **Method**
  - Merge two edge vertices to one
  - Delete degenerate triangles

- **Properties**
  - Requires manifold surface around vertex
  - Preserves local topological structure
  - Typically more attractive
  - Allows smooth transition

![Edge Collapse Diagram](image)
Operation Considerations

• Topology considerations
  ◦ Attention to topology promotes better appearance
  ◦ Allowing non-manifolds increases robustness and ability to simplify

• Operation considerations
  ◦ Collapse-type operations allow smooth transitions
  ◦ Vertex remove affects smaller portion of mesh than edge collapse

Geometric Error Metrics

• Motivation
  ◦ Promote accurate 3D shape preservation
  ◦ Preserve screen-space silhouettes and pixel coverages

• Types
  ◦ Vertex-Vertex Distance
  ◦ Vertex-Plane Distance
  ◦ Point-Surface Distance
  ◦ Surface-Surface Distance
**Vertex-Vertex Distance**

- \( E = \max( ||v_3-v_1||, ||v_3-v_2|| ) \)
- Appropriate during topology changes
  - Rossignac and Borrel 93
  - Luebke and Erikson 97
- Loose for topology-preserving collapses

![Vertex-Vertex Distance Diagram](image)

**Vertex-Plane Distance**

- Store set of planes with each vertex
  - Error based on distance from vertex to planes
  - When vertices are merged, merge sets
- Ronfard and Rossignac 96
  - Store plane sets, compute max distance
- Error Quadrics - Garland and Heckbert 96
  - Store quadratic form, compute sum of square distances

![Vertex-Plane Distance Diagram](image)
Point-Surface Distance
- Map point set to closest points on simplified surface
- Compute sum of square distances

Surface-Surface Distance
- Bound maximum distance between input and simplified surfaces
  - Tolerance Volumes - Guéziec 96
  - Simplification Envelopes - Cohen/Varshney 96
  - Hausdorff Distance - Klein 96
  - Mapping Distance - Bajaj/Schikore 96, Cohen et al. 97
Vertex-Vertex ≠ Surface-Surface

- Error is zero at vertices and exterior edges
- Error is non-zero everywhere else
  - not captured by vertex-vertex or vertex-plane metrics

Geometric Error Observations

- Vertex-vertex and vertex-plane distance
  - Fast
  - Low error shown after-the-fact, but not guaranteed by metric
- Surface-surface distance
  - Required to guarantee errors

vertex-vertex ≠ surface-surface
Mesh Simplification Considerations

- Type of input mesh?
- Modifies topology?
- Continuous LOD?
- Speed vs. quality?

View-Dependent Simplification

- Simplify dynamically according to viewpoint
  - Visibility
  - Silhouettes
  - Lighting
Appearance Preserving Simplification

Course Projects