



# Polygonal Meshes

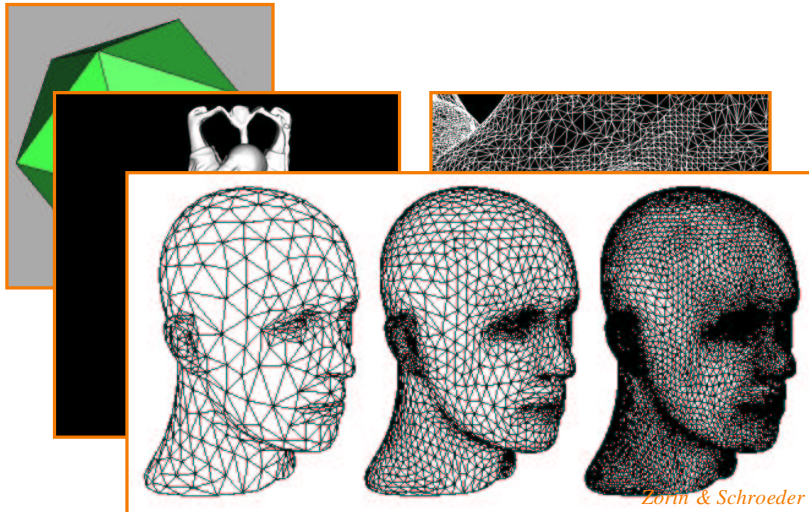
Thomas Funkhouser  
Princeton University  
COS 526, Fall 2002



## Outline

- Polygonal meshes
- Mesh data structures
- Mesh simplification

# Polygonal Meshes



## Outline

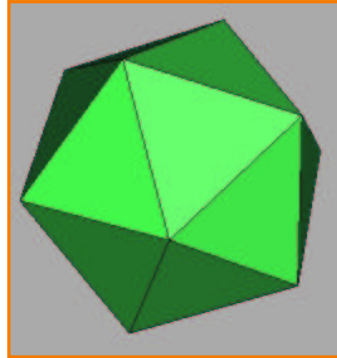


- Polygonal meshes
- Mesh data structures
- Mesh simplification

## Mesh Data Structures



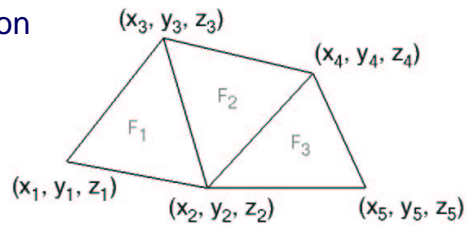
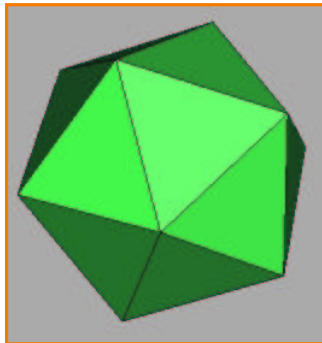
- Mesh Representations
  - Independent faces
  - Vertex and face tables
  - Adjacency lists
  - Winged-Edge
  - Triangle meshes



## Independent Faces



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

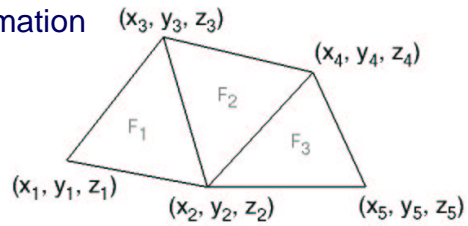
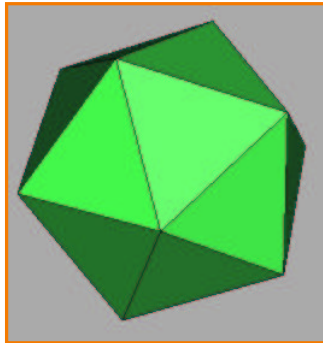


FACE TABLE			
F <sub>1</sub>	( $x_1, y_1, z_1$ )	( $x_2, y_2, z_2$ )	( $x_3, y_3, z_3$ )
F <sub>2</sub>	( $x_2, y_2, z_2$ )	( $x_4, y_4, z_4$ )	( $x_3, y_3, z_3$ )
F <sub>3</sub>	( $x_2, y_2, z_2$ )	( $x_5, y_5, z_5$ )	( $x_4, y_4, z_4$ )

## Vertex and Face Tables



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



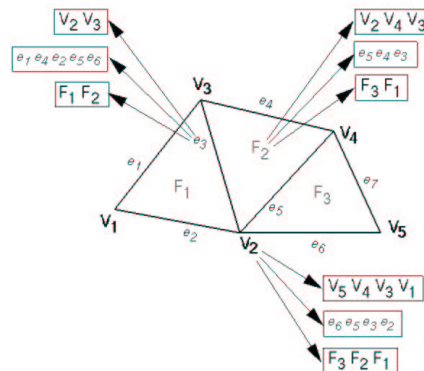
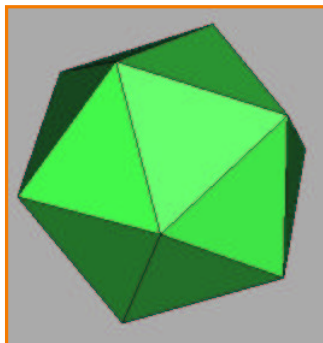
VERTEX TABLE			
V <sub>1</sub>	X <sub>1</sub>	Y <sub>1</sub>	Z <sub>1</sub>
V <sub>2</sub>	X <sub>2</sub>	Y <sub>2</sub>	Z <sub>2</sub>
V <sub>3</sub>	X <sub>3</sub>	Y <sub>3</sub>	Z <sub>3</sub>
V <sub>4</sub>	X <sub>4</sub>	Y <sub>4</sub>	Z <sub>4</sub>
V <sub>5</sub>	X <sub>5</sub>	Y <sub>5</sub>	Z <sub>5</sub>

FACE TABLE			
F <sub>1</sub>	V <sub>1</sub>	V <sub>2</sub>	V <sub>3</sub>
F <sub>2</sub>	V <sub>2</sub>	V <sub>4</sub>	V <sub>3</sub>
F <sub>3</sub>	V <sub>2</sub>	V <sub>5</sub>	V <sub>4</sub>

## Adjacency Lists



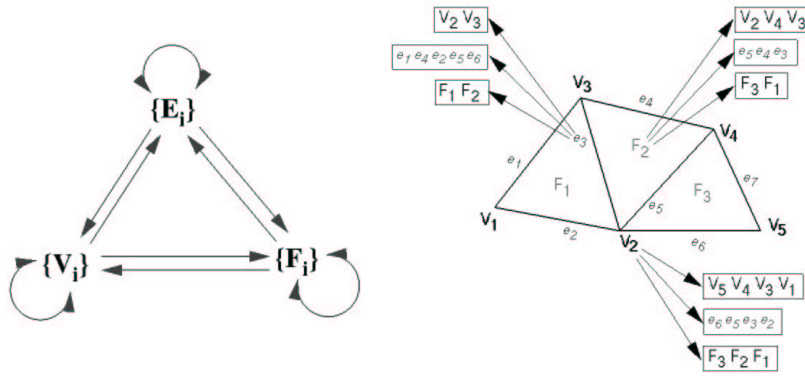
- Store all vertex, edge, and face adjacencies
  - Efficient topology traversal
  - Extra storage



## Partial Adjacency Lists



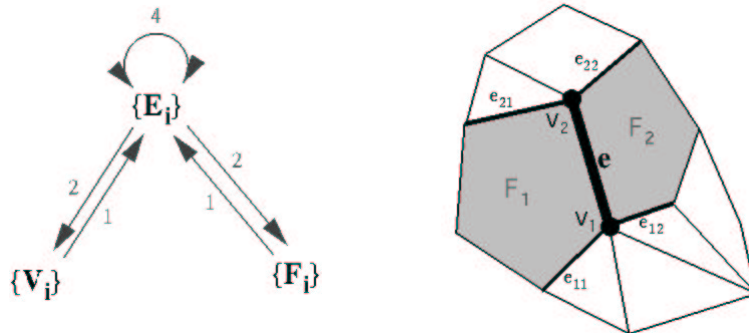
- Can we store only some adjacency relationships and derive others?



## Winged Edge



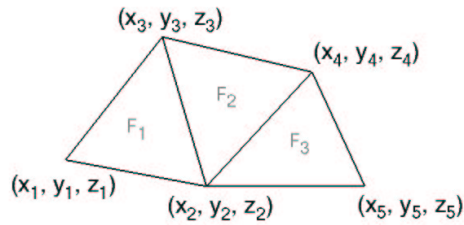
- Adjacency encoded in edges
  - All adjacencies in  $O(1)$  time
  - Little extra storage (fixed records)
  - Arbitrary polygons



## Winged Edge



- Example:



VERTEX TABLE				
V <sub>1</sub>	X <sub>1</sub>	Y <sub>1</sub>	Z <sub>1</sub>	e <sub>1</sub>
V <sub>2</sub>	X <sub>2</sub>	Y <sub>2</sub>	Z <sub>2</sub>	e <sub>6</sub>
V <sub>3</sub>	X <sub>3</sub>	Y <sub>3</sub>	Z <sub>3</sub>	e <sub>3</sub>
V <sub>4</sub>	X <sub>4</sub>	Y <sub>4</sub>	Z <sub>4</sub>	e <sub>5</sub>
V <sub>5</sub>	X <sub>5</sub>	Y <sub>5</sub>	Z <sub>5</sub>	e <sub>6</sub>

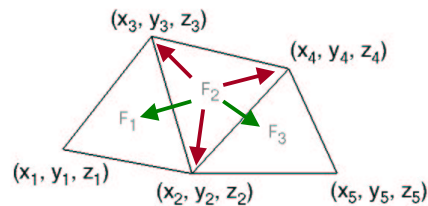
EDGE TABLE					11	12	21	22
e <sub>1</sub>	V <sub>1</sub>	V <sub>3</sub>	F <sub>1</sub>	e <sub>2</sub>	e <sub>2</sub>	e <sub>4</sub>	e <sub>3</sub>	
e <sub>2</sub>	V <sub>1</sub>	V <sub>2</sub>	F <sub>1</sub>	e <sub>1</sub>	e <sub>1</sub>	e <sub>3</sub>	e <sub>6</sub>	
e <sub>3</sub>	V <sub>2</sub>	V <sub>3</sub>	F <sub>1</sub> F <sub>2</sub>	e <sub>2</sub>	e <sub>5</sub>	e <sub>1</sub>	e <sub>4</sub>	
e <sub>4</sub>	V <sub>3</sub>	V <sub>4</sub>	F <sub>2</sub>	e <sub>1</sub>	e <sub>3</sub>	e <sub>7</sub>	e <sub>5</sub>	
e <sub>5</sub>	V <sub>2</sub>	V <sub>4</sub>	F <sub>2</sub> F <sub>3</sub>	e <sub>3</sub>	e <sub>6</sub>	e <sub>4</sub>	e <sub>7</sub>	
e <sub>6</sub>	V <sub>2</sub>	V <sub>5</sub>	F <sub>3</sub>	e <sub>5</sub>	e <sub>2</sub>	e <sub>7</sub>	e <sub>7</sub>	
e <sub>7</sub>	V <sub>4</sub>	V <sub>5</sub>	F <sub>3</sub>	e <sub>4</sub>	e <sub>5</sub>	e <sub>6</sub>	e <sub>6</sub>	

FACE TABLE	
F <sub>1</sub>	e <sub>1</sub>
F <sub>2</sub>	e <sub>3</sub>
F <sub>3</sub>	e <sub>5</sub>

## 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



## Outline



- Polygonal meshes
- Mesh data structures

Ø 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



*Stanford Graphics Lab*

## 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

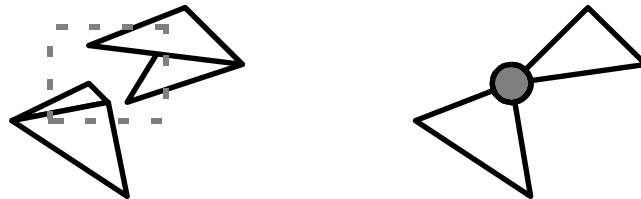


- General idea:
  - Each operations simplifies model by small amount
  - Apply many operations in succession
- Types of operations
  - Vertex cluster
  - Vertex remove
  - Edge collapse
  - Vertex pair

## Vertex Cluster



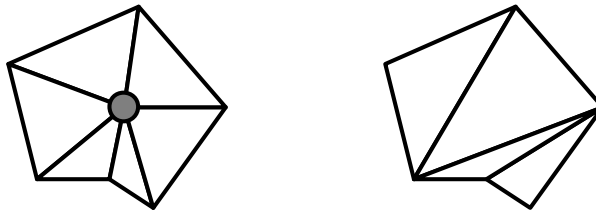
- Method
  - Merge vertices based on proximity
  - Triangles with repeated vertices become edge or point
- Properties
  - General and robust
  - Not usually attractive



## Vertex Remove



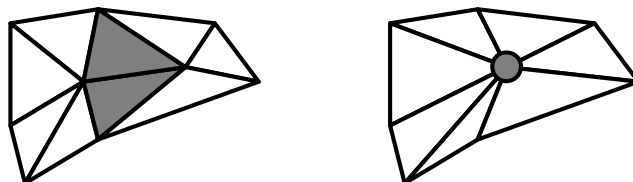
- Method
  - Remove vertex and adjacent faces
  - Fill hole with new triangles (reduction of 2)
- Properties
  - Requires manifold surface around vertex
  - Preserves local topological structure
  - Typically more attractive



## 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



## 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

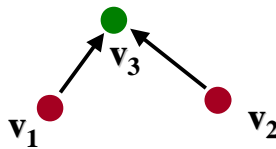


- Motivation
  - Promote accurate 3D shape preservation
  - Preserve screen-space silhouettes and pixel coverages
- Types
  - Vertex-Vertex Distance
  - Vertex-Plane Distance
  - Point-Surface Distance
  - Surface-Surface Distance

## Vertex-Vertex Distance



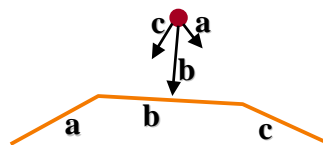
- $E = \max( \|v_3 - v_1\|, \|v_3 - v_2\| )$
- Appropriate during topology changes
  - Rossignac and Borrel 93
  - Luebke and Erikson 97
- Loose for topology-preserving collapses



## Vertex-Plane Distance



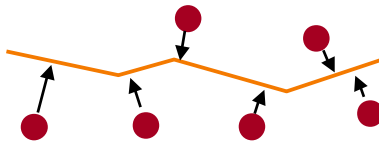
- Store set of planes with each vertex
  - Error based on distance from vertex to planes
  - When vertices are merged, merge sets
- Ronfard and Rossignac 96
  - Store plane sets, compute max distance
- Error Quadratics - Garland and Heckbert 96
  - Store quadratic form, compute sum of square distances



## Point-Surface Distance



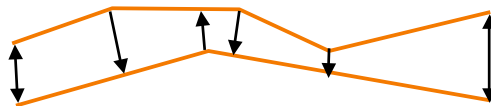
- Map point set to closest points on simplified surface
- Compute sum of square distances



## 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 $\neq$ Surface-Surface

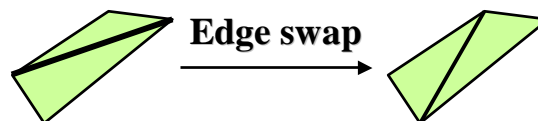


- Error is zero at vertices and exterior edges
- Error is non-zero everywhere else
  - not captured by vertex-vertex or vertex-plane metrics

## 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



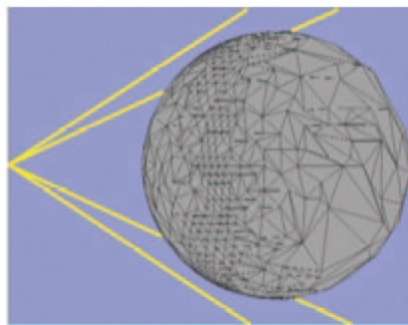
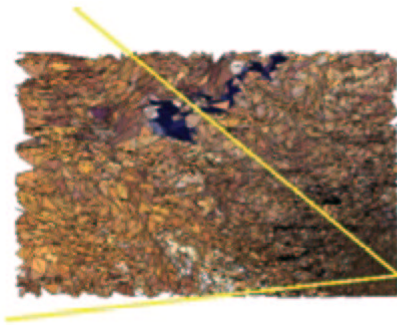
vertex-vertex  $\neq$  surface-surface

## Mesh Simplification Considerations

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

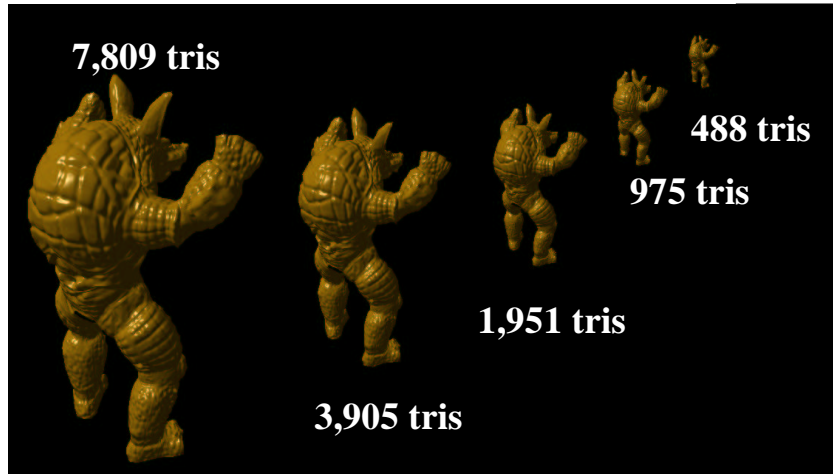
## View-Dependent Simplification

- Simplify dynamically according to viewpoint
  - Visibility
  - Silhouettes
  - Lighting



*Hugues Hoppe*

## Appearance Preserving Simplification



*Caltech & Stanford Graphics Labs  
and Jonathan Cohen*

## Course Projects