



Polygonal Meshes

COS 426, Spring 2021

Princeton University

Felix Heide



DEADPOOL
20TH CENTURY FOX (2016)

3D Object Representations



- Points

- Range image
- Point cloud

- Surfaces

- Polygonal mesh

- Parametric
- Subdivision
- Implicit

- Solids

- Voxels
- BSP tree
- CSG
- Sweep

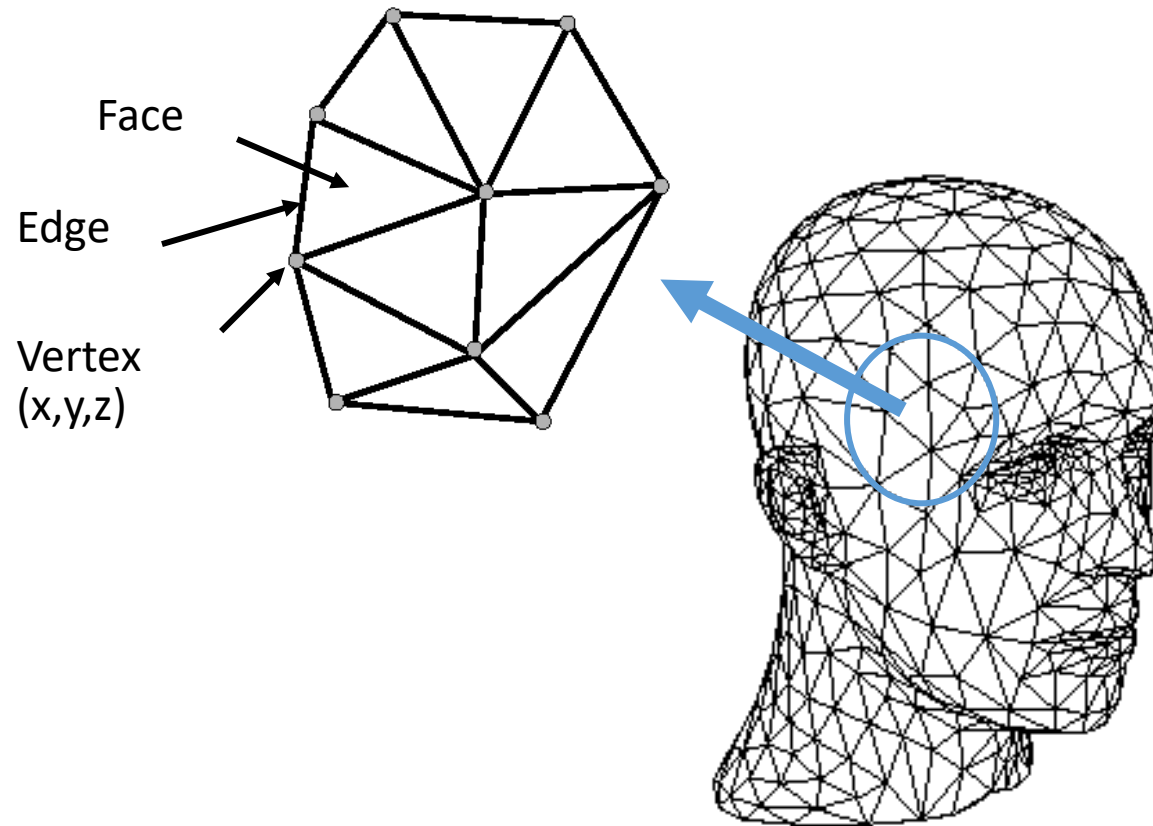
- High-level structures

- Scene graph
- Application specific

3D Polygonal Mesh



- Set of polygons representing a 2D surface embedded in 3D



3D Polygonal Mesh



- The power of polygonal meshes

3D Polygonal Mesh



- Set of polygons representing a 2D surface embedded in 3D

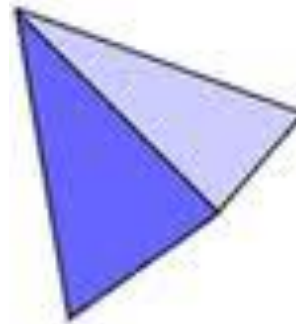
Platonic Solids



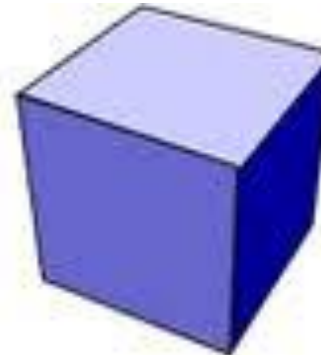
Dodecahedron



Icosahedron



Tetrahedron

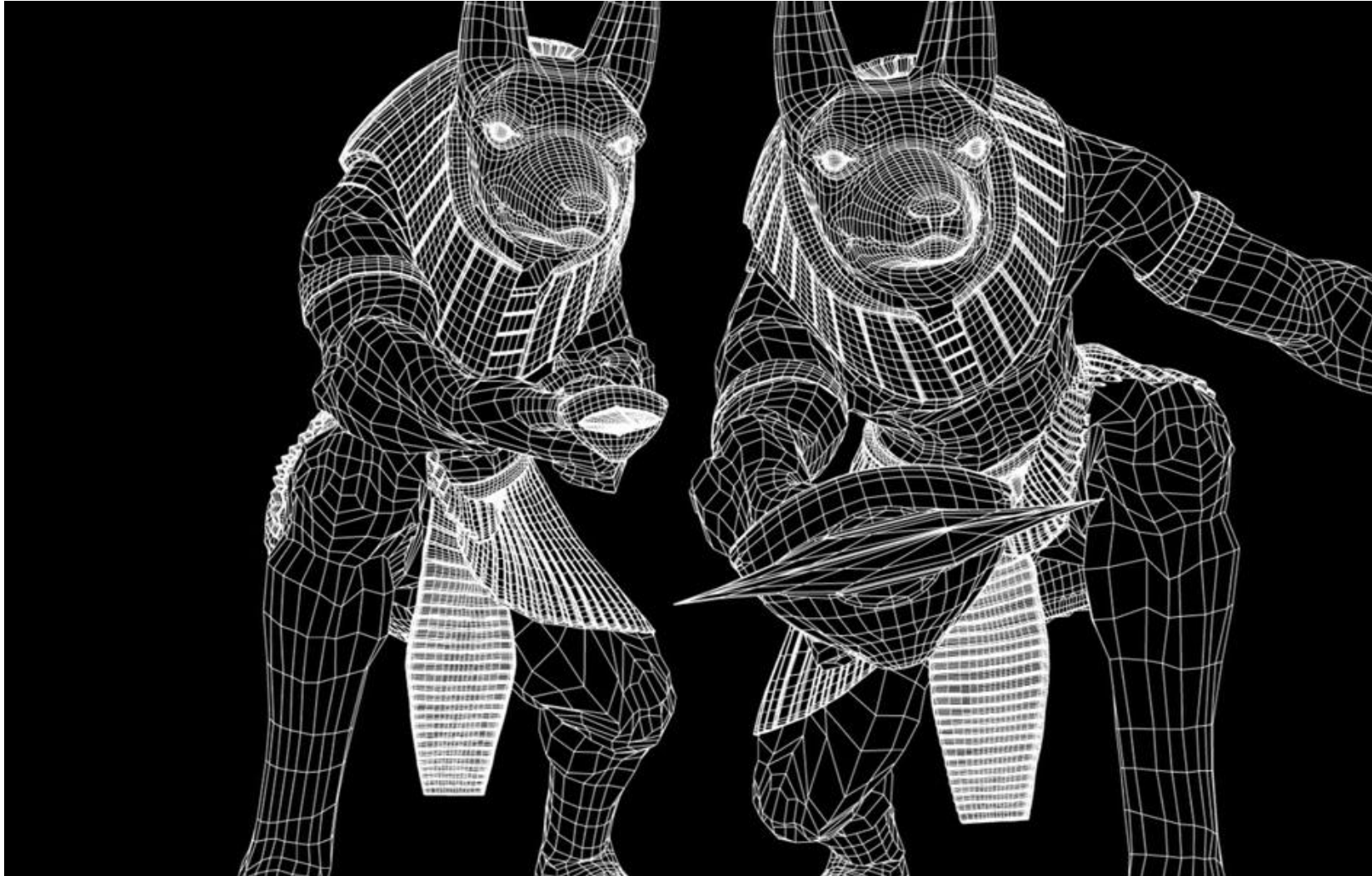


Cube

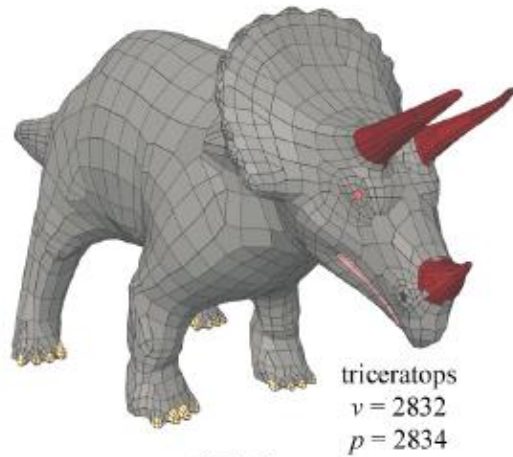


Octahedron

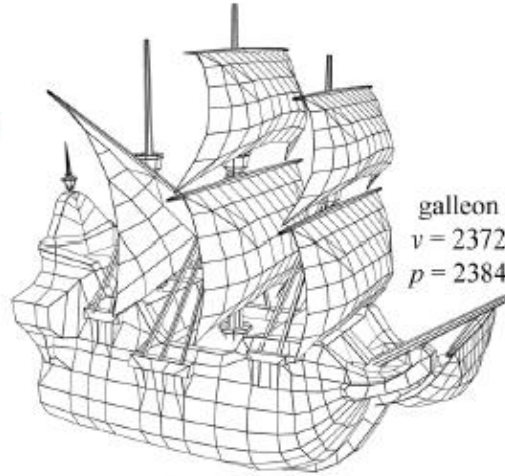
3D Polygonal Mesh



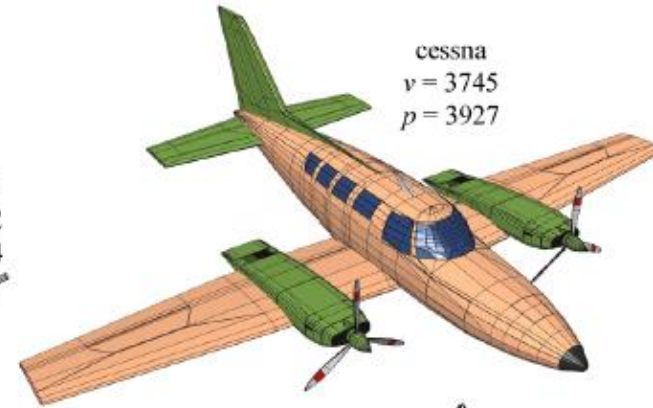
3D Polygonal Mesh



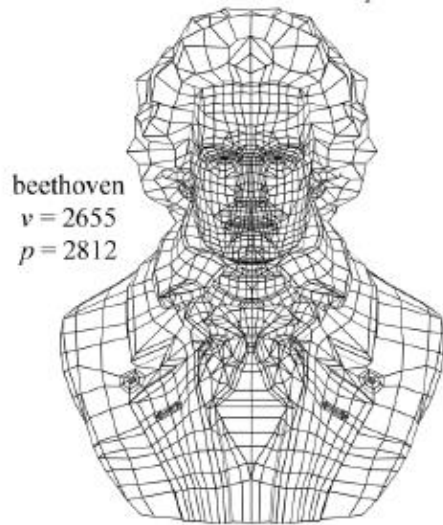
triceratops
 $v = 2832$
 $p = 2834$



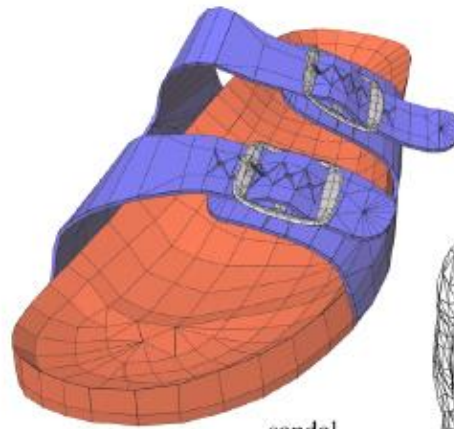
galleon
 $v = 2372$
 $p = 2384$



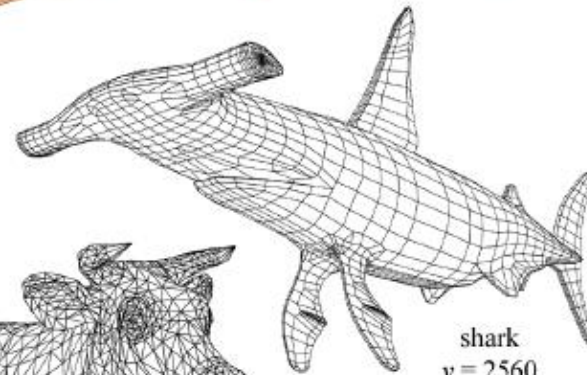
cessna
 $v = 3745$
 $p = 3927$



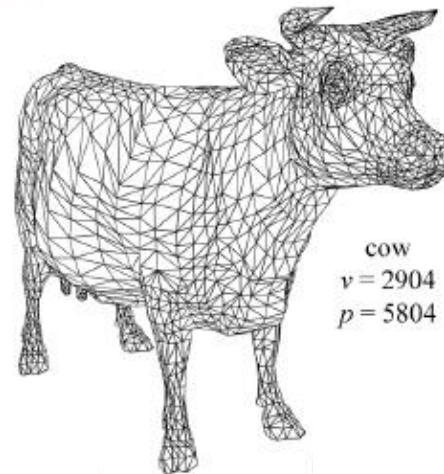
beethoven
 $v = 2655$
 $p = 2812$



sandal
 $v = 2636$
 $p = 2953$



shark
 $v = 2560$
 $p = 2562$



cow
 $v = 2904$
 $p = 5804$

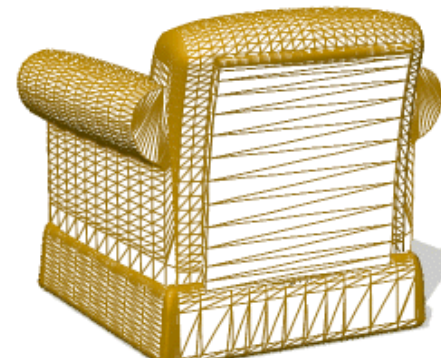
cow_poly
 $v = 2904$
 $p = 3263$

(the polygonal cow is not shown, it is the same cow model, but not fully triangulated)

3D Polygonal Meshes



- Why are they of interest?
 - Simple, common representation
 - Rendering with hardware support
 - Output of many acquisition tools
 - Input to many simulation/analysis tools



Viewpoint

Outline



- Acquisition
- Representation
- Processing



Polygonal Mesh Acquisition

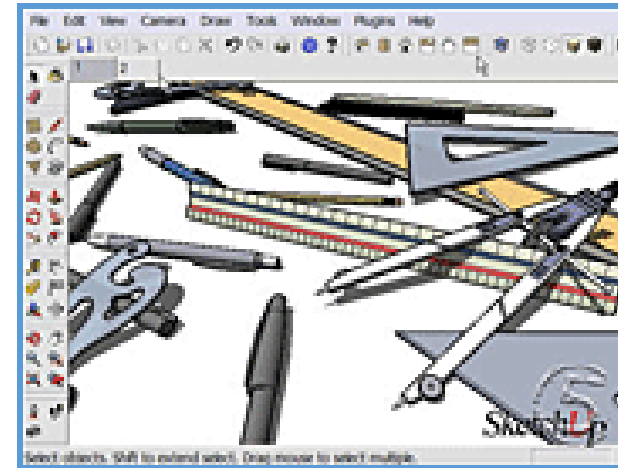


- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations

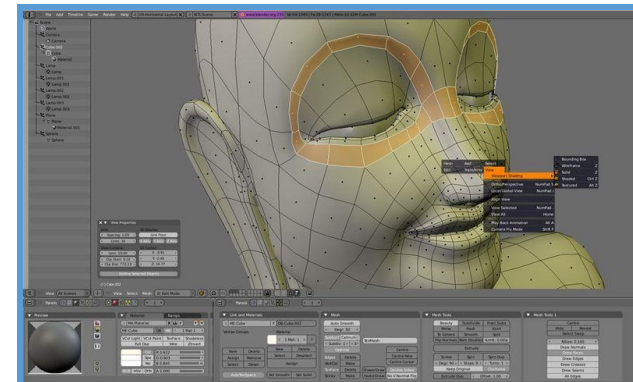
Polygonal Mesh Acquisition



- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations



Sketchup

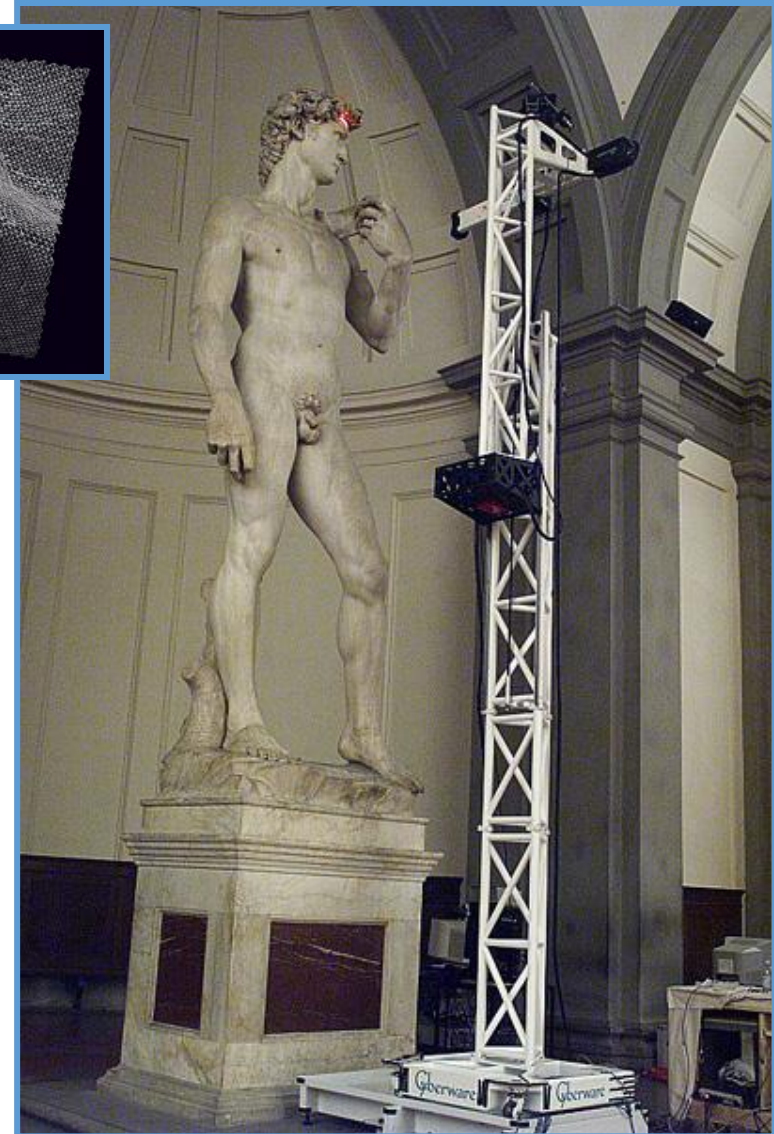
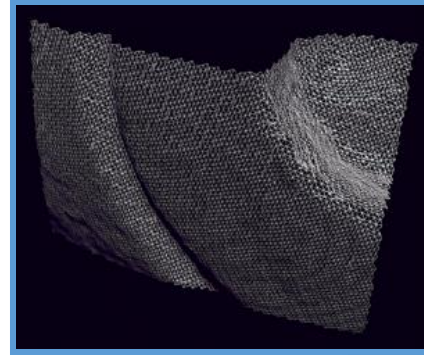


Blender

Polygonal Mesh Acquisition



- Interactive modeling
- **Scanners**
- Procedural generation
- Conversion
- Simulations

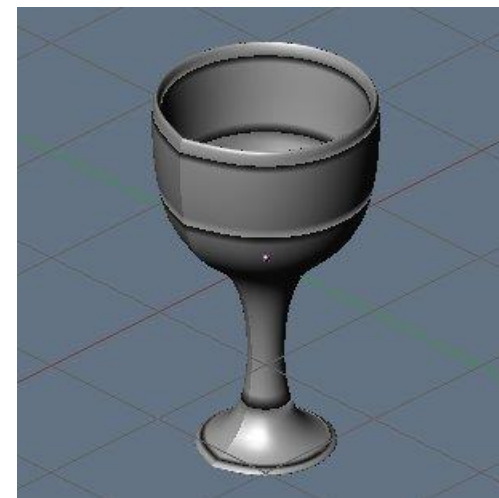
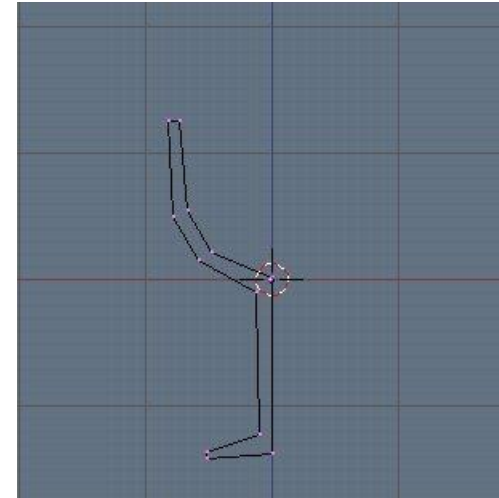


Digital Michelangelo Project
Stanford

Polygonal Mesh Acquisition



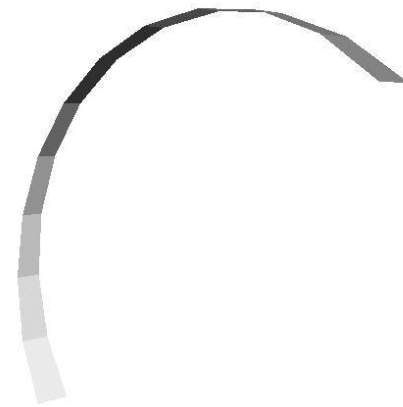
- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations



Polygonal Mesh Acquisition



- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations



MakeAGIF.com

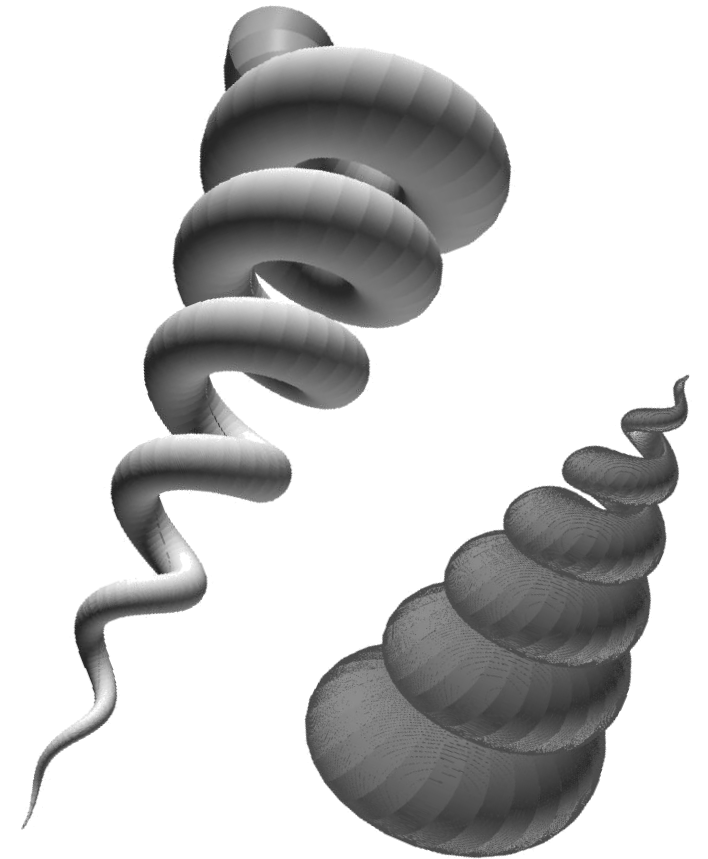
Polygonal Mesh Acquisition



- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations



Fowler et al., 1992

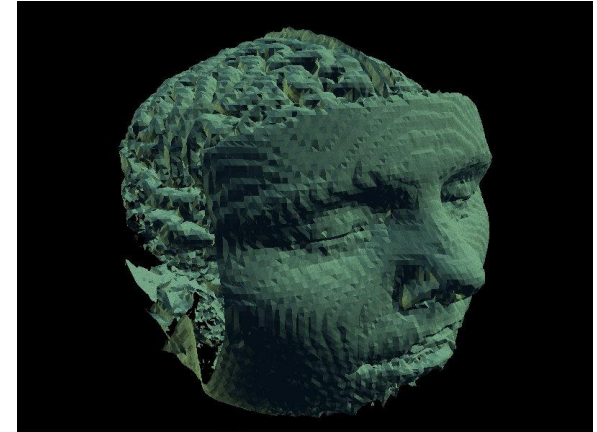
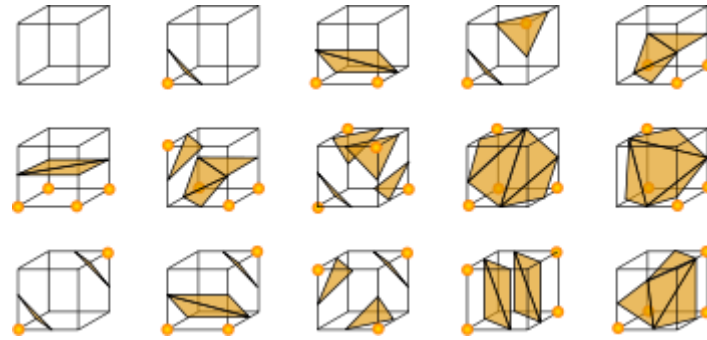


Peter Maag, COS 426, 2010

Polygonal Mesh Acquisition



- Interactive modeling
- Scanners
- Procedural generation
- **Conversion**
- Simulations



Marching cubes

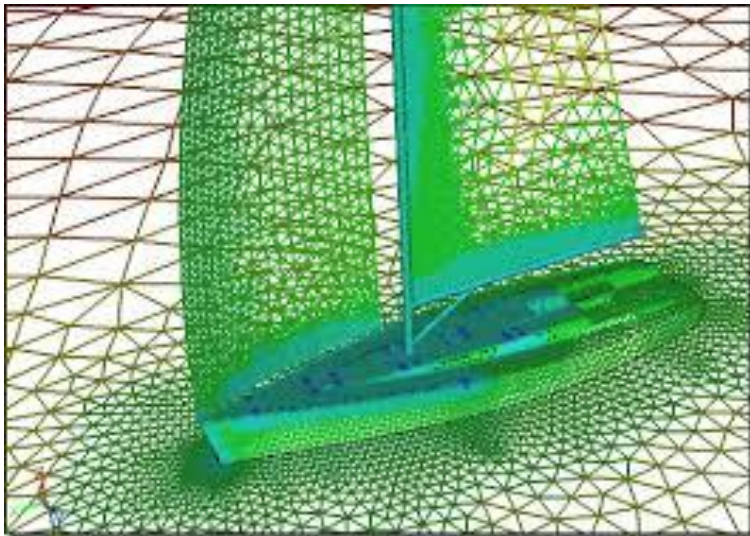


Jose Maria De Espona

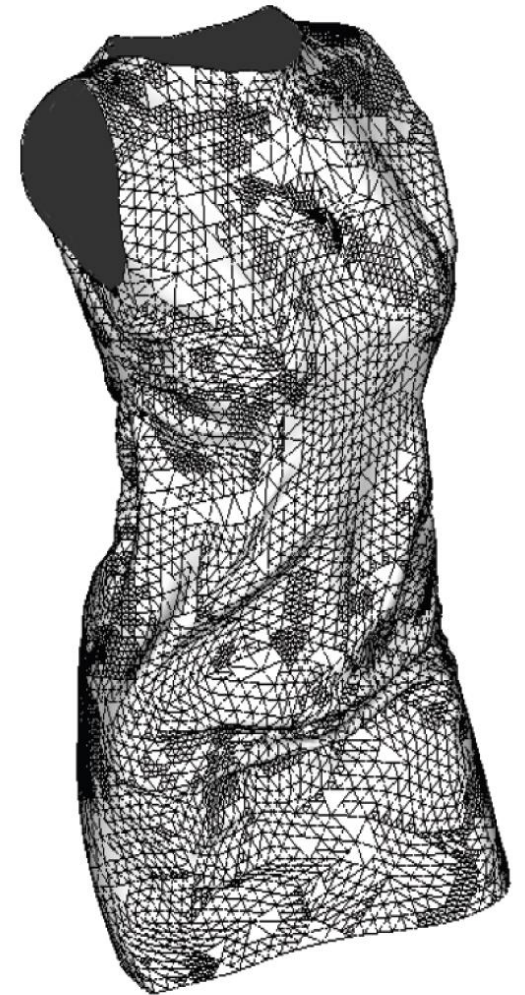
Polygonal Mesh Acquisition



- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations



sym scape



Lee et. al 2010

Outline

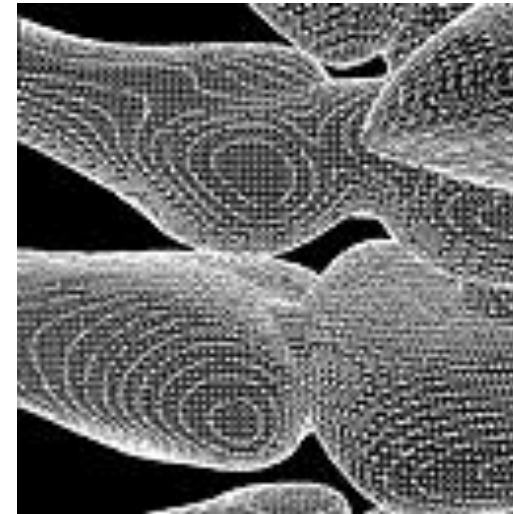
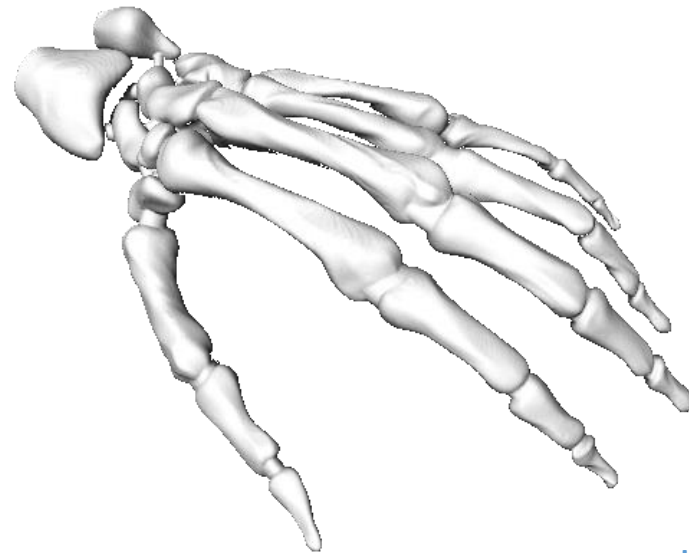


- Acquisition
- Representation ←
- Processing

Polygon Mesh Representation



- Important properties of mesh representation?
 - Efficient traversal of topology
 - Efficient use of memory
 - Efficient updates

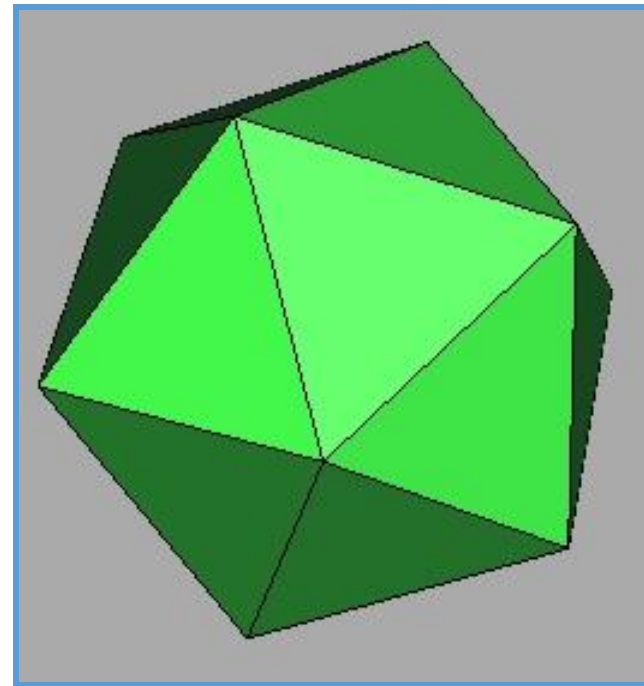


Large Geometric Model Repository
Georgia Tech

Polygon Mesh Representation



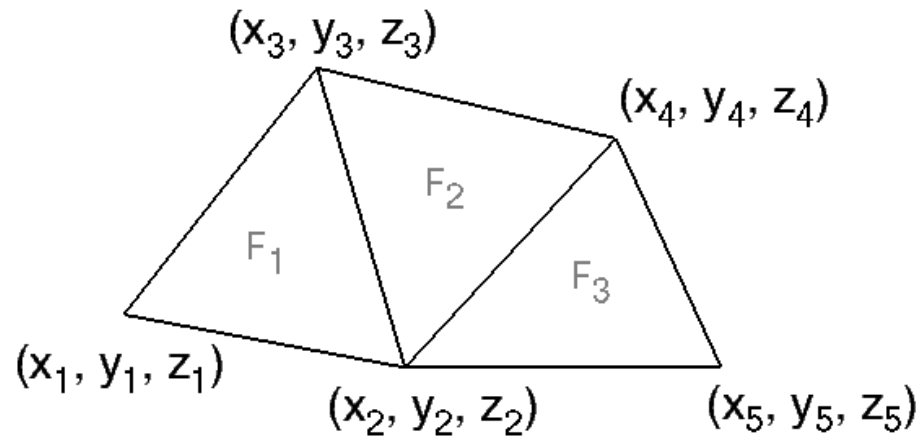
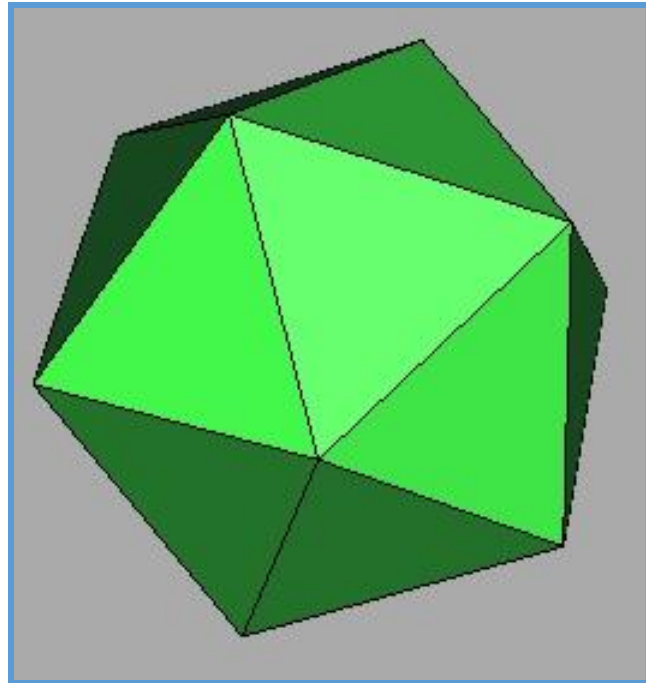
- Possible data structures



Independent Faces



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



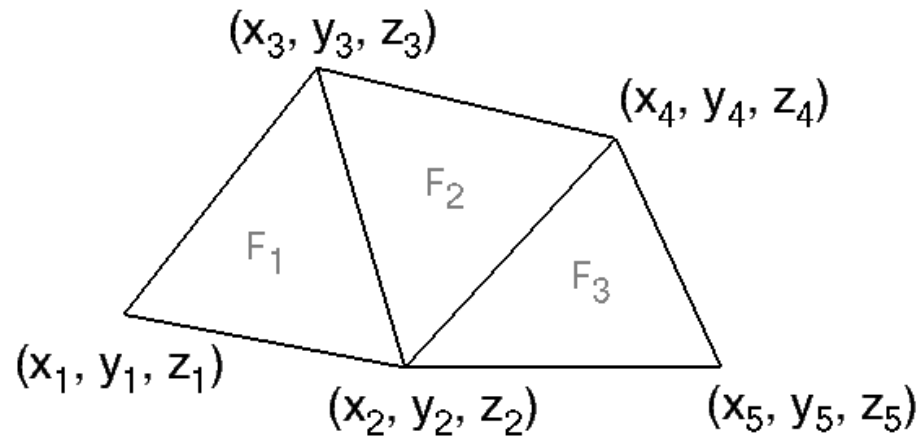
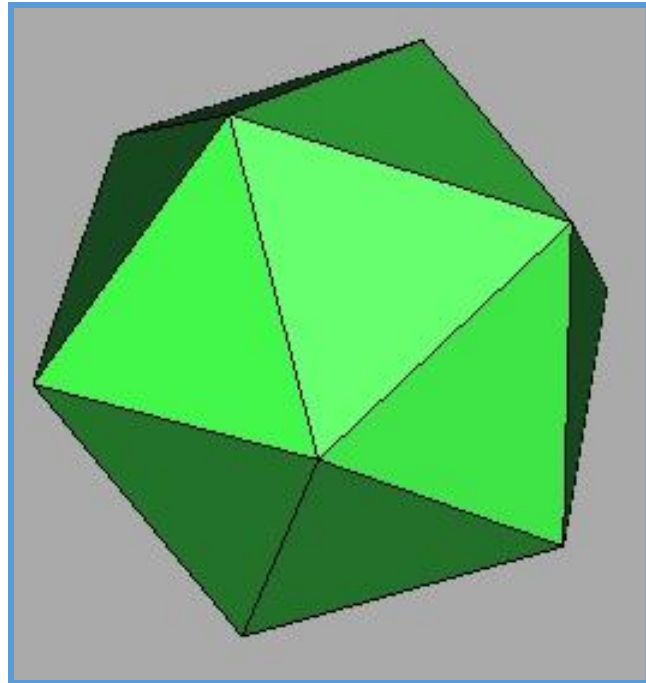
FACE TABLE

F_1	(x_1, y_1, z_1)	(x_2, y_2, z_2)	(x_3, y_3, z_3)
F_2	(x_2, y_2, z_2)	(x_4, y_4, z_4)	(x_3, y_3, z_3)
F_3	(x_2, y_2, z_2)	(x_5, y_5, z_5)	(x_4, y_4, z_4)

Vertex and Face Tables (Indexed Vertices)



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



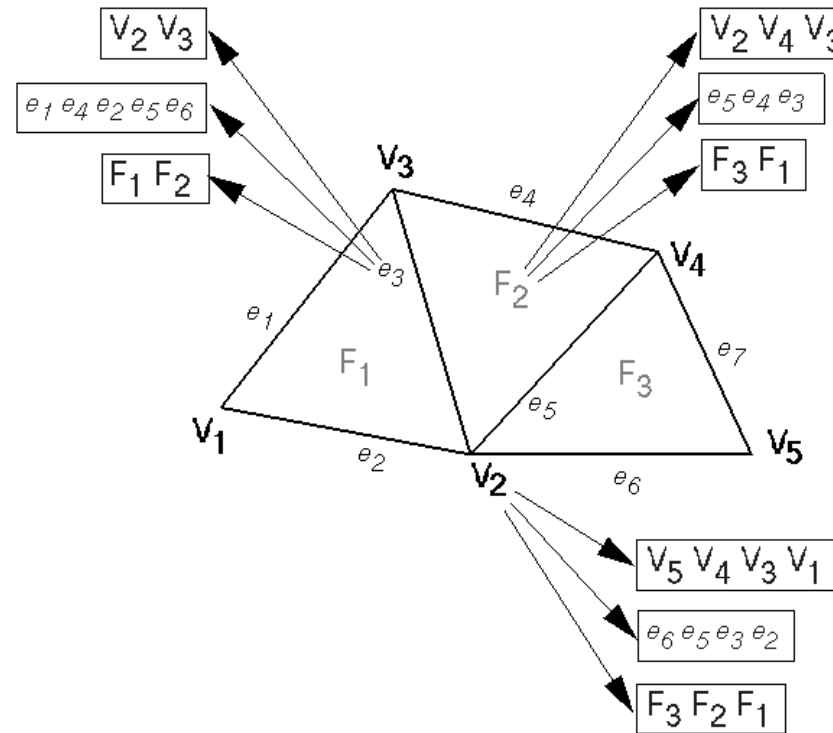
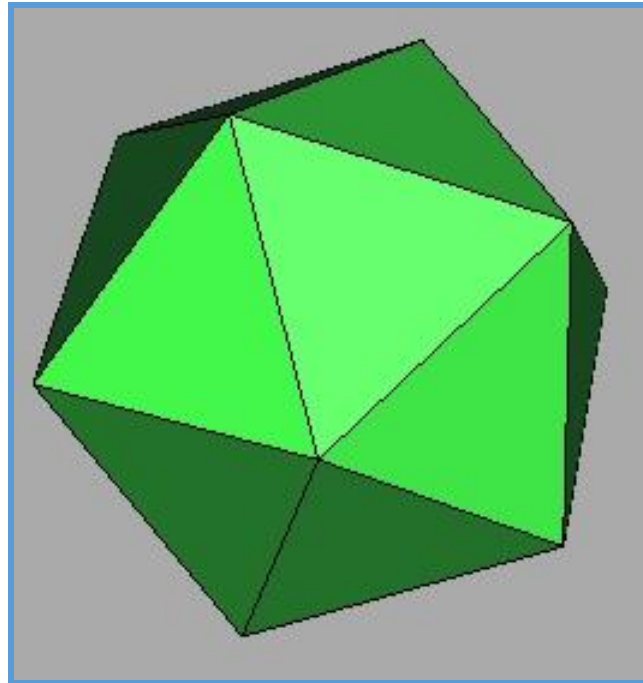
VERTEX TABLE			
V ₁	X ₁	Y ₁	Z ₁
V ₂	X ₂	Y ₂	Z ₂
V ₃	X ₃	Y ₃	Z ₃
V ₄	X ₄	Y ₄	Z ₄
V ₅	X ₅	Y ₅	Z ₅

FACE TABLE			
F ₁	V ₁	V ₂	V ₃
F ₂	V ₂	V ₄	V ₃
F ₃	V ₂	V ₅	V ₄

Full Adjacency Lists



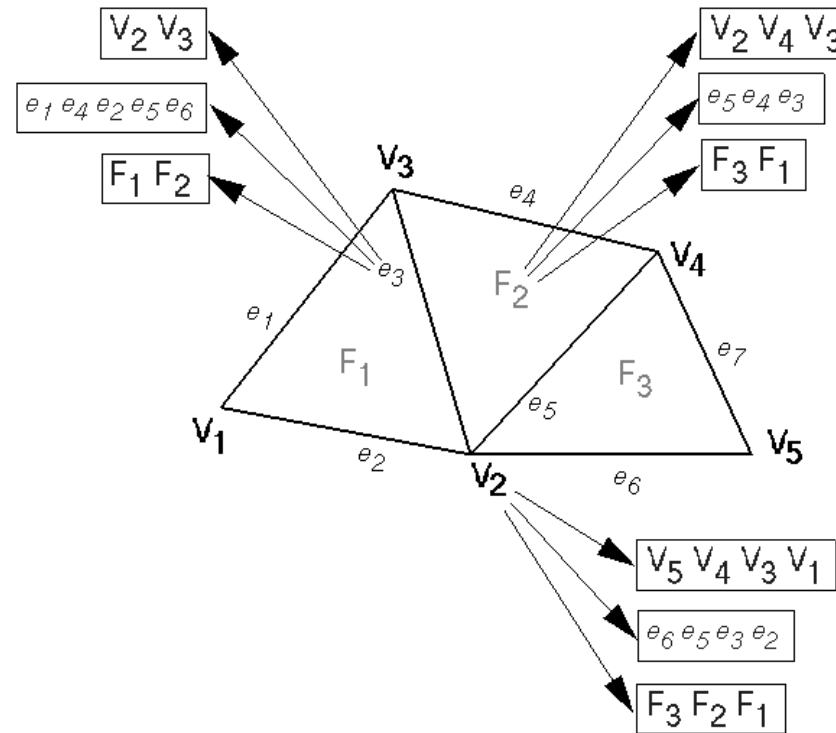
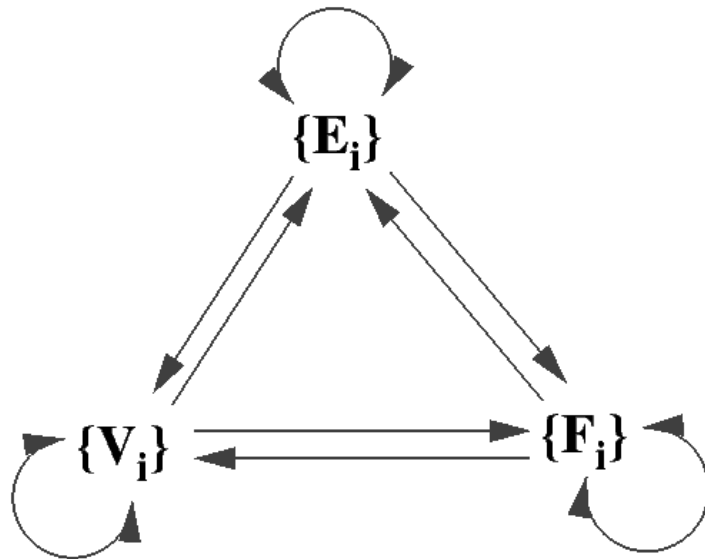
- Store all vertex, edge, and face adjacencies
 - Efficient adjacency traversal
 - Extra storage



Full Adjacency Lists



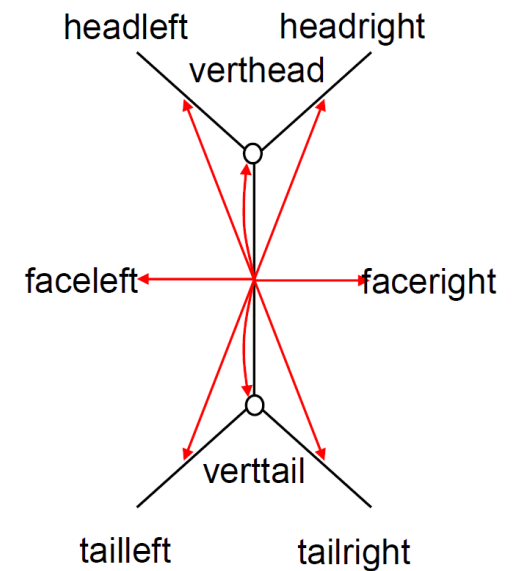
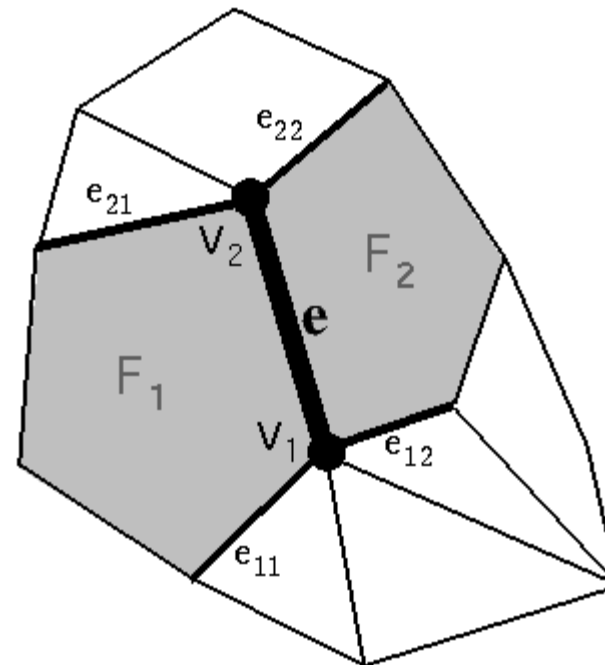
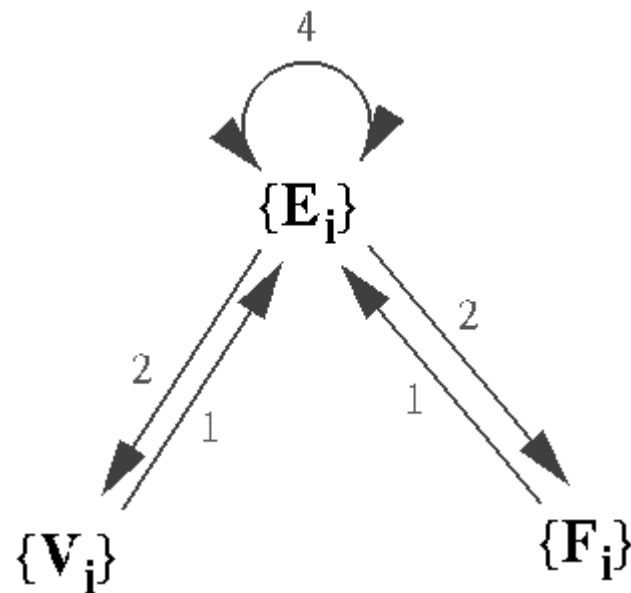
- Can we store only some adjacency relationships and derive others?



Partial Adjacency - Winged Edge



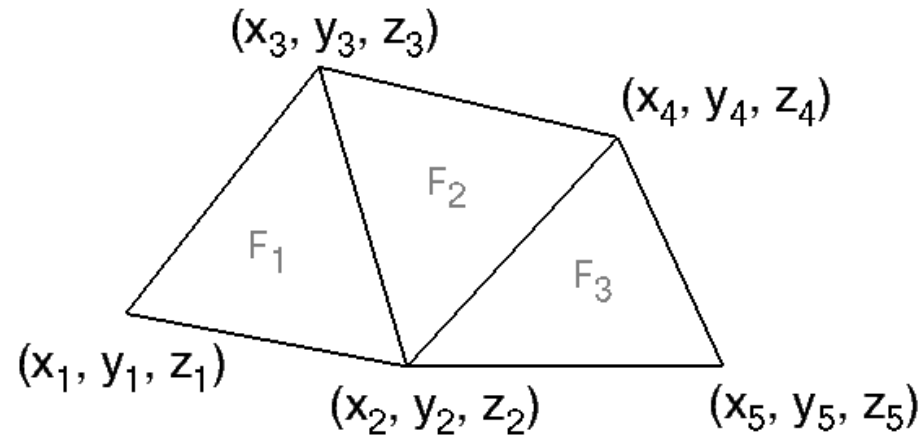
- Adjacency encoded in **edges**
 - All adjacencies in $O(1)$ time
 - Little extra storage (fixed records)
 - Arbitrary polygons



Winged Edge



- Example:



VERTEX TABLE				
V ₁	X ₁	Y ₁	Z ₁	e ₁
V ₂	X ₂	Y ₂	Z ₂	e ₆
V ₃	X ₃	Y ₃	Z ₃	e ₃
V ₄	X ₄	Y ₄	Z ₄	e ₅
V ₅	X ₅	Y ₅	Z ₅	e ₆

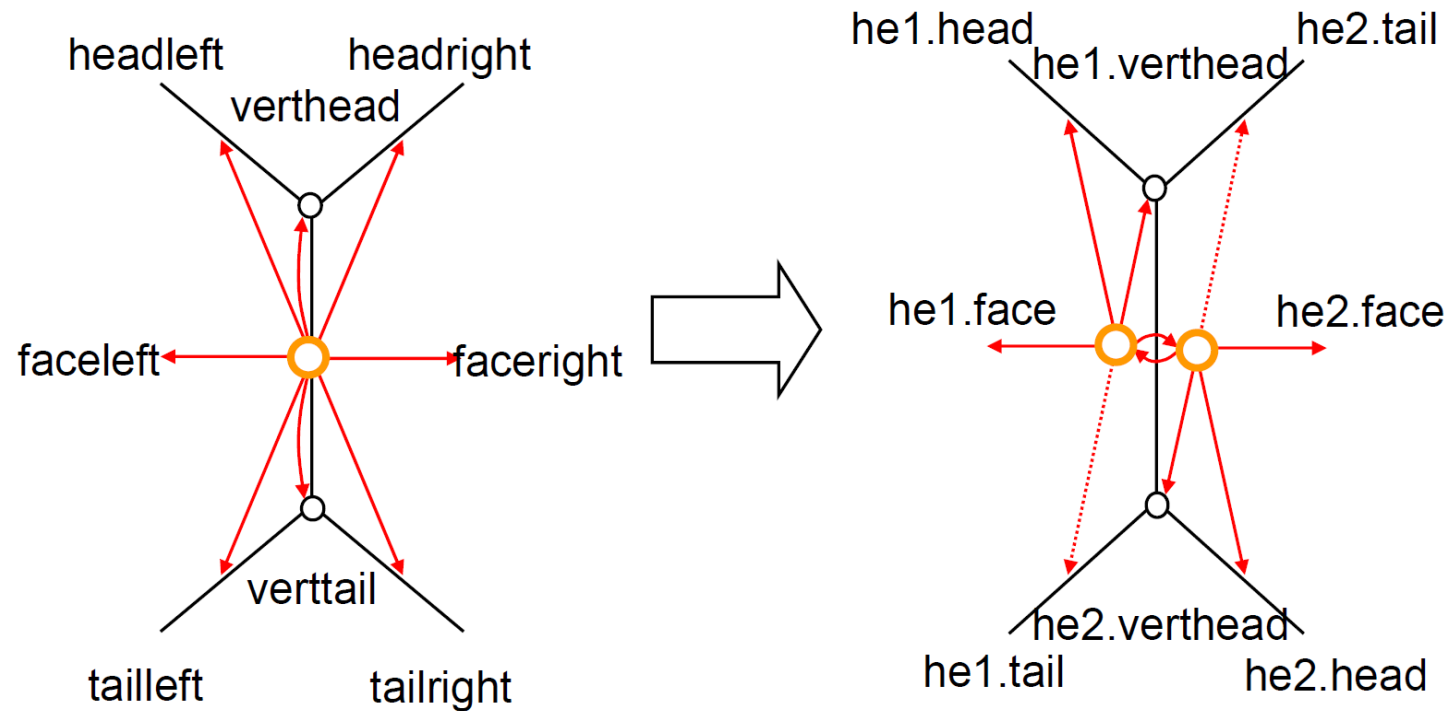
EDGE TABLE					11	12	21	22
e ₁	V ₁	V ₃	F ₁		e ₂	e ₂	e ₄	e ₃
e ₂	V ₁	V ₂	F ₁		e ₁	e ₁	e ₃	e ₆
e ₃	V ₂	V ₃	F ₁	F ₂	e ₂	e ₅	e ₁	e ₄
e ₄	V ₃	V ₄		F ₂	e ₁	e ₃	e ₇	e ₅
e ₅	V ₂	V ₄	F ₂	F ₃	e ₃	e ₆	e ₄	e ₇
e ₆	V ₂	V ₅	F ₃		e ₅	e ₂	e ₇	e ₇
e ₇	V ₄	V ₅		F ₃	e ₄	e ₅	e ₆	e ₆

FACE TABLE	
F ₁	e ₁
F ₂	e ₃
F ₃	e ₅

Half Edge



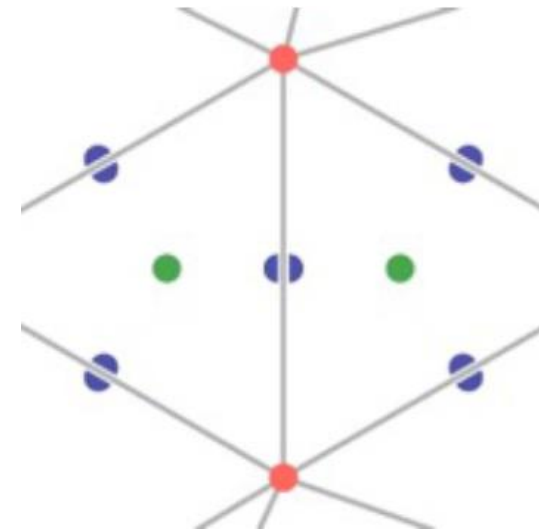
- traversals do not require “ifs” in code
- consistent orientation



Half Edge ... in more detail



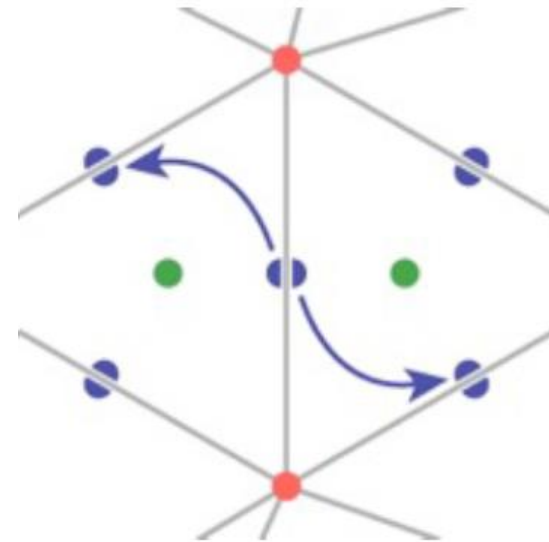
- Each **half-edge** stores:
 - Its twin half-edge



Half Edge



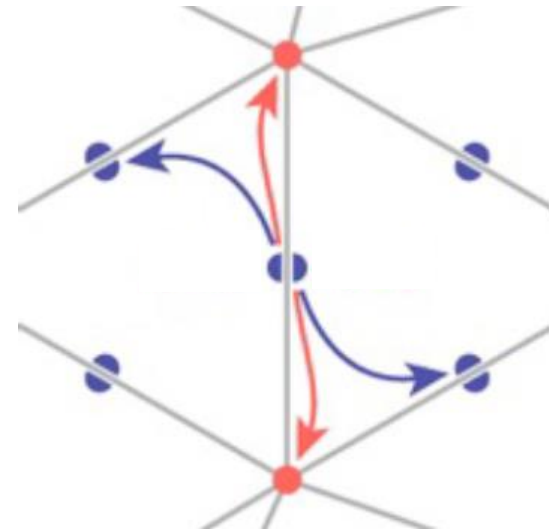
- Each **half-edge** stores:
 - Its twin half-edge
 - The next half-edge



Half Edge



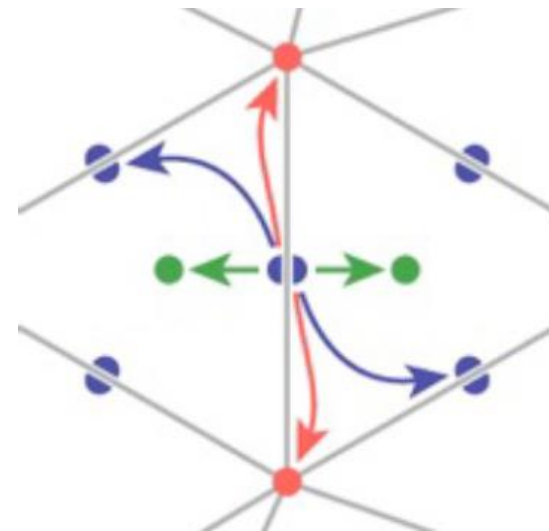
- Each **half-edge** stores:
 - Its twin half-edge
 - The next half-edge
 - **The next vertex**



Half Edge



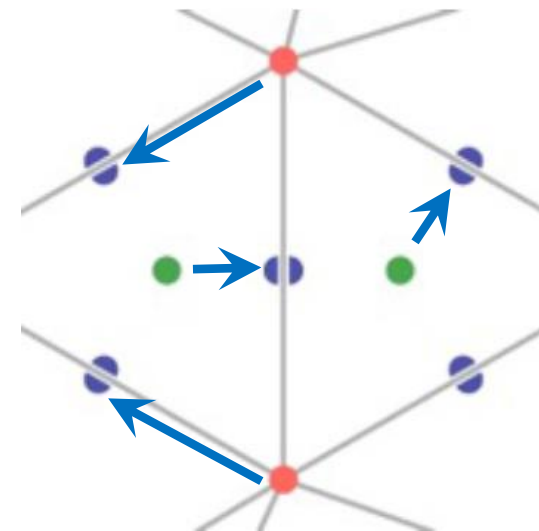
- Each **half-edge** stores:
 - Its twin half-edge
 - The next half-edge
 - **The next vertex**
 - **The incident face**



Half Edge



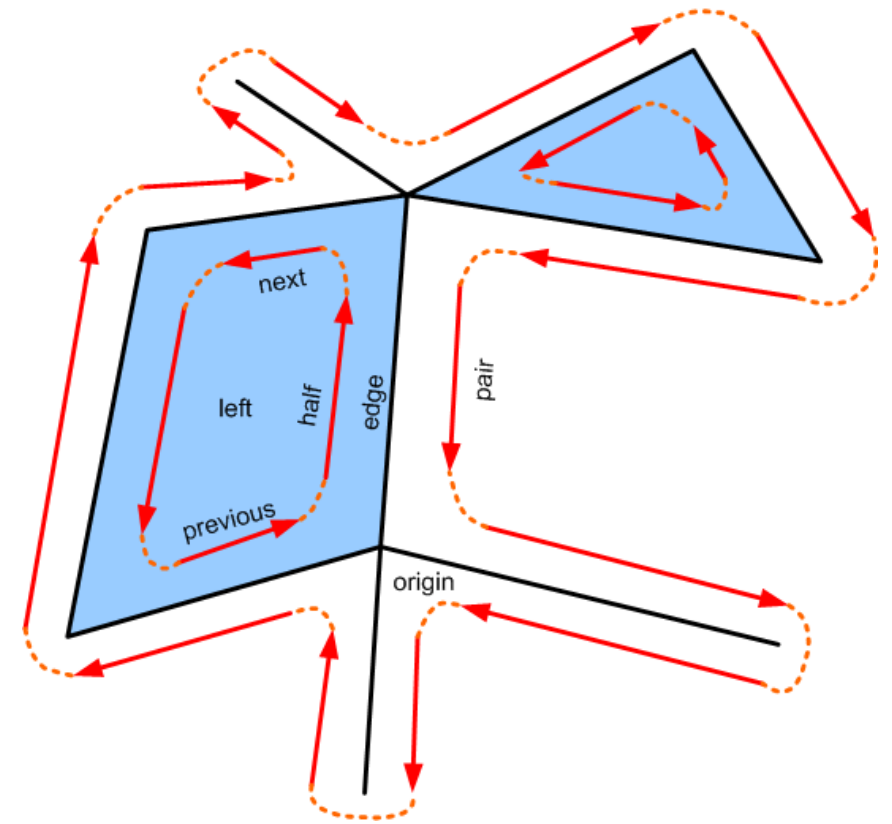
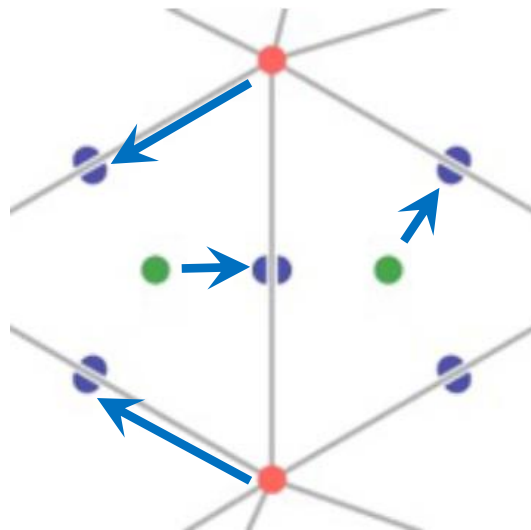
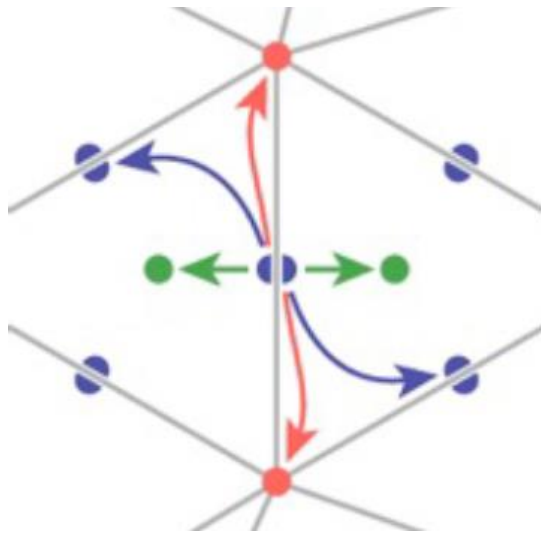
- Each **half-edge** stores:
 - Its twin half-edge
 - The next half-edge
 - **The next vertex**
 - **The incident face**
- Each face stores:
 - 1 adjacent half-edge
- Each vertex stores:
 - 1 outgoing half-edge



Half Edge



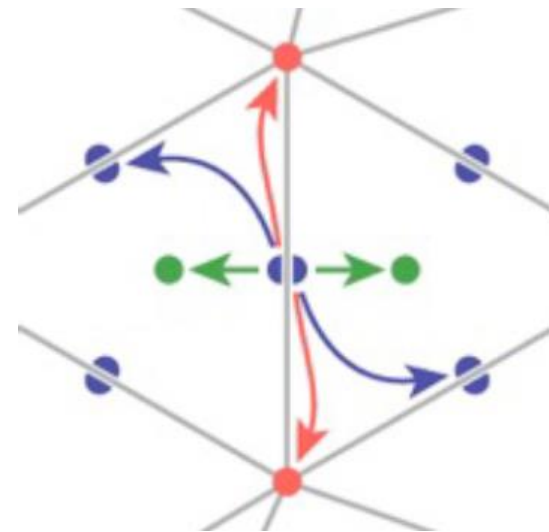
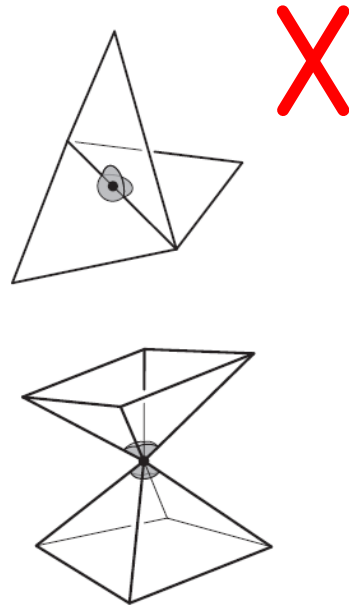
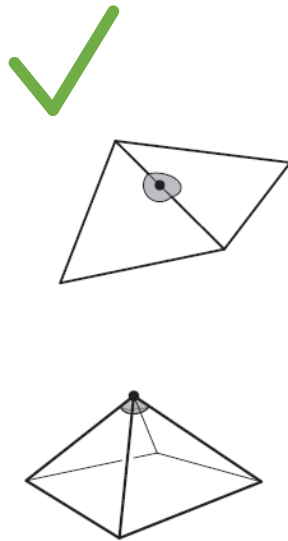
- Queries. How do you find:
 - All faces incident to an edge?
 - All vertices of a face?
 - All faces incident to a face?
 - All vertices incident to a vertex?



Half Edge



- Adjacency encoded in edges
 - All adjacencies in $O(1)$ time
 - Little extra storage (fixed records)
 - Arbitrary polygons
 - **Assumes 2-Manifold surfaces**



Outline



- Acquisition
- Representation
- Processing



Polygonal Mesh Processing

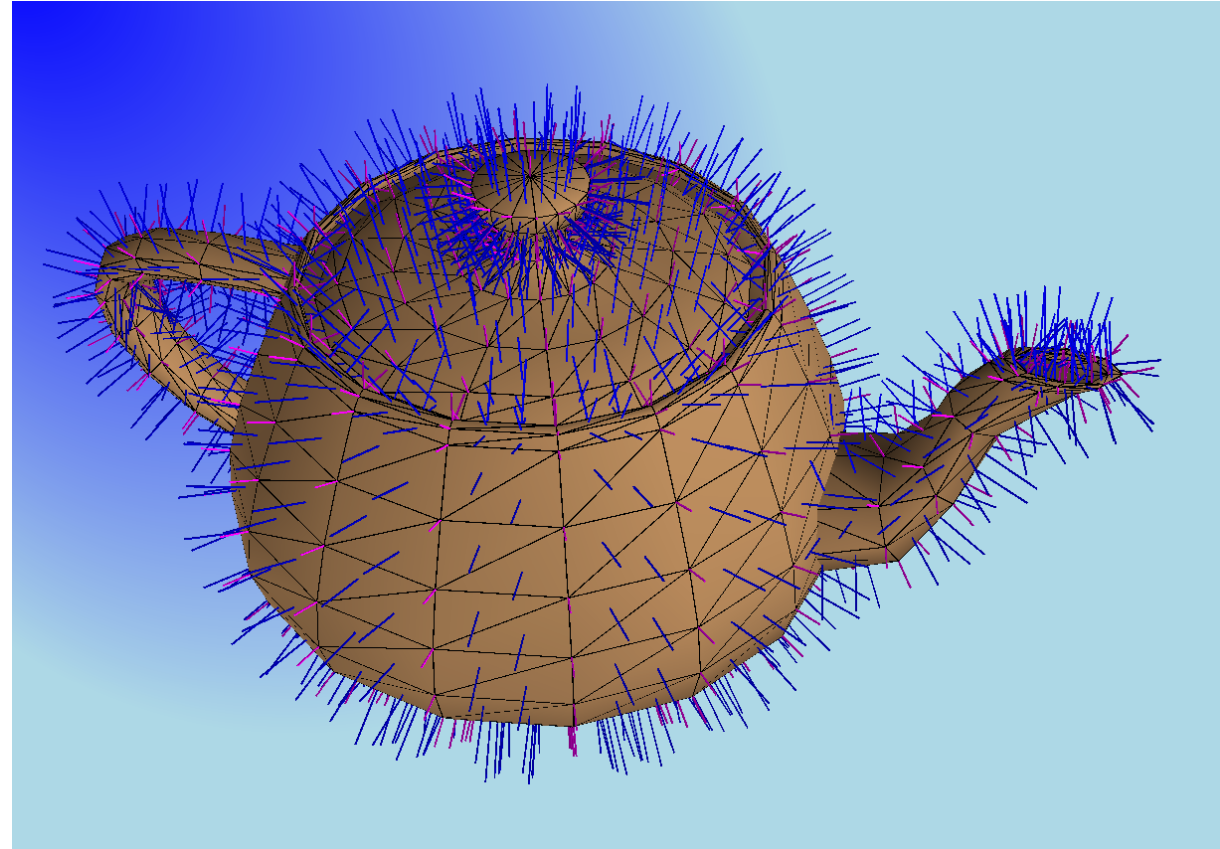


- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

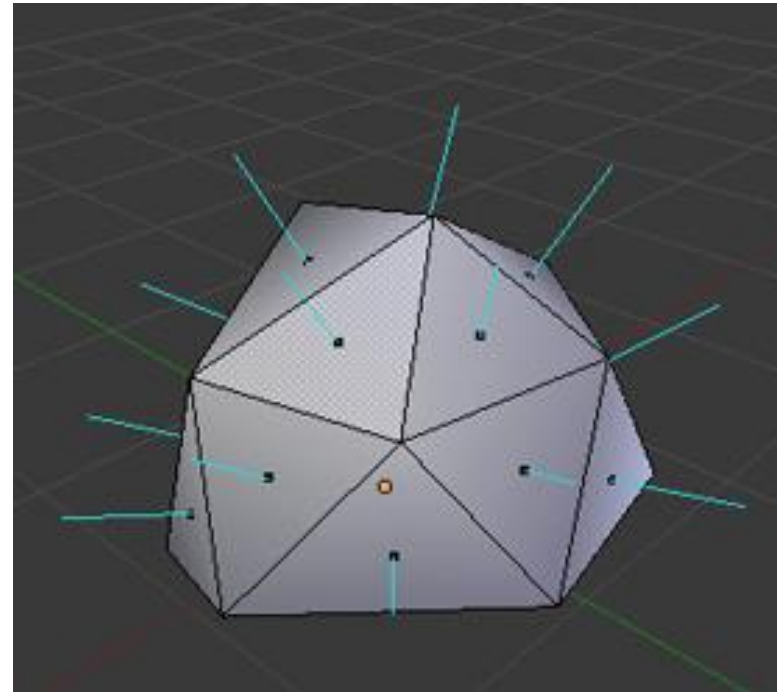
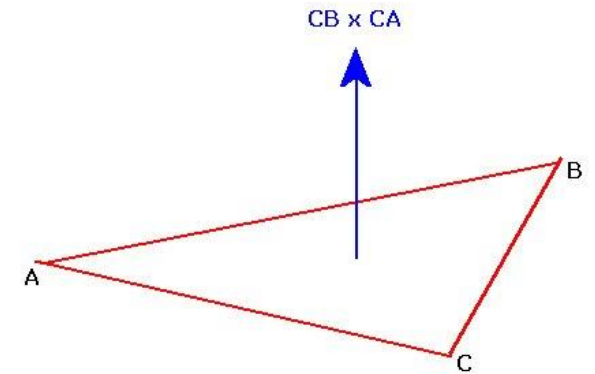


Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

Face normals:

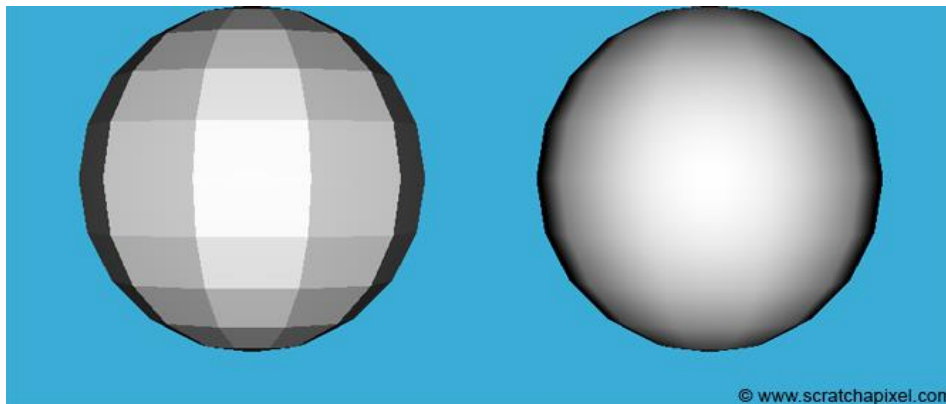
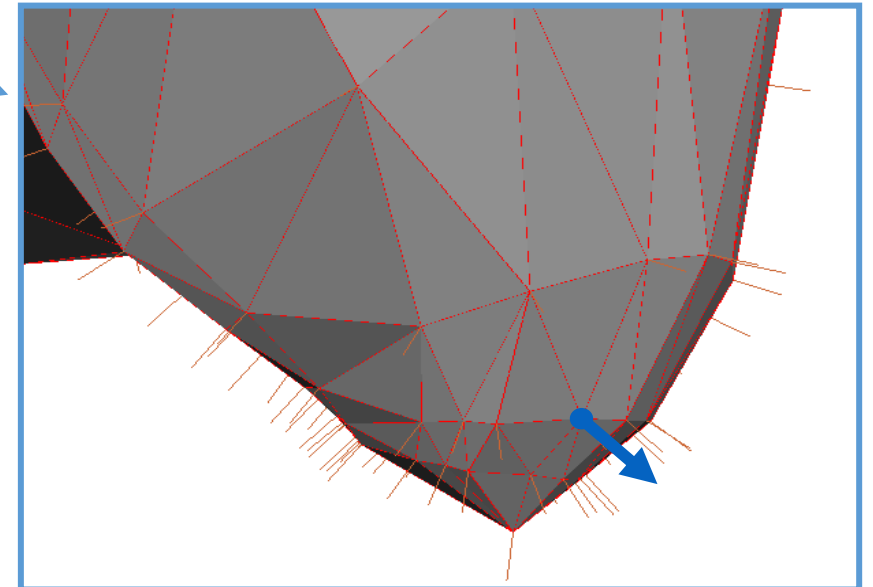
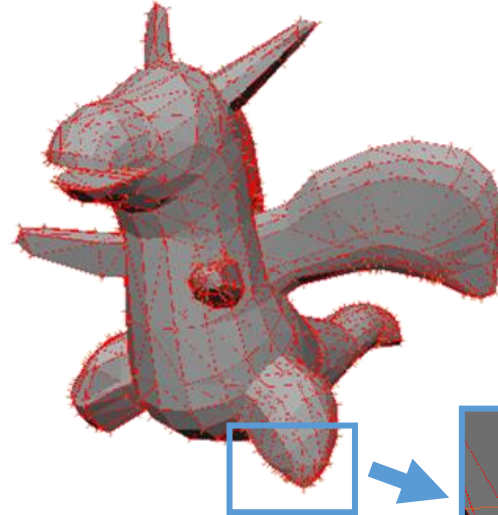


Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

Vertex normals:

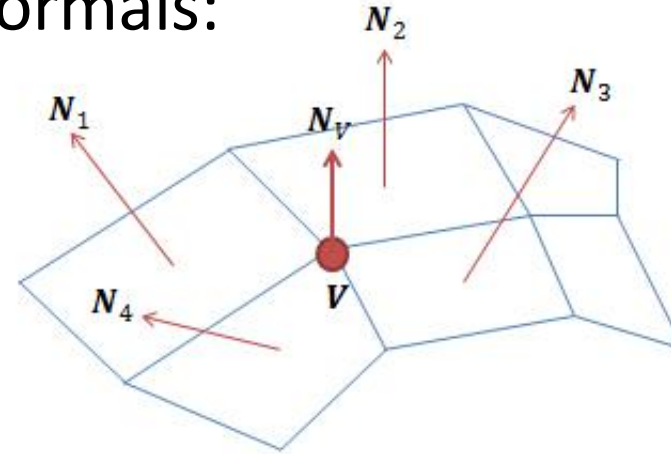


Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

Vertex normals:



$$N_V = \frac{\sum_{k=1}^n N_k}{|\sum_{k=1}^n N_k|}$$

for each face

-calculate face normal

-add normal to each connected vertex normal

for each vertex normal

-normalize

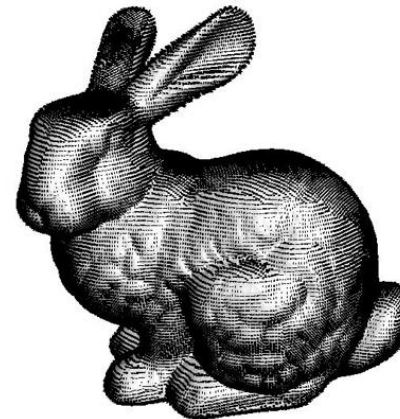
Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

NORMAL VERTEX

presents



The Next Dual

“The bunny with normal vertices shown.
Reminded me of an album cover so I made it into one.”

Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

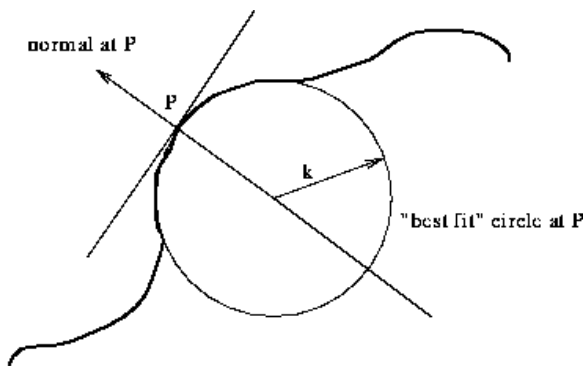
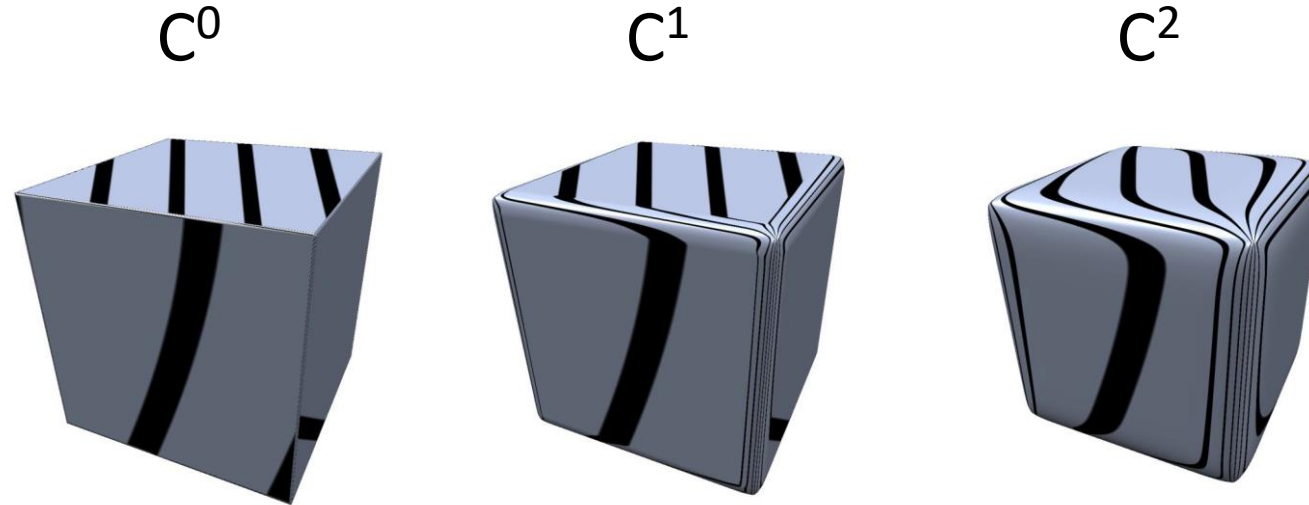


Figure 32: curvature of curve at P is $1/k$

Polygonal Mesh Processing



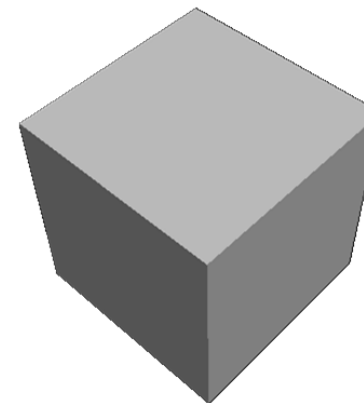
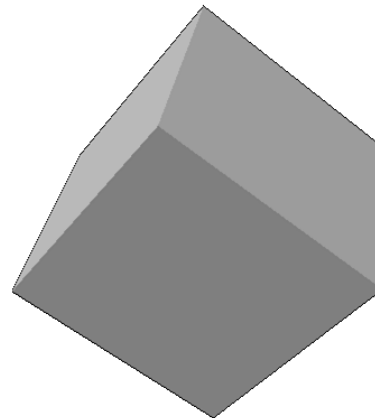
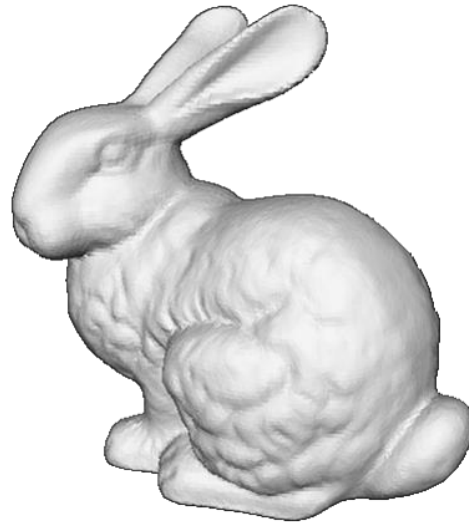
- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel



Polygonal Mesh Processing



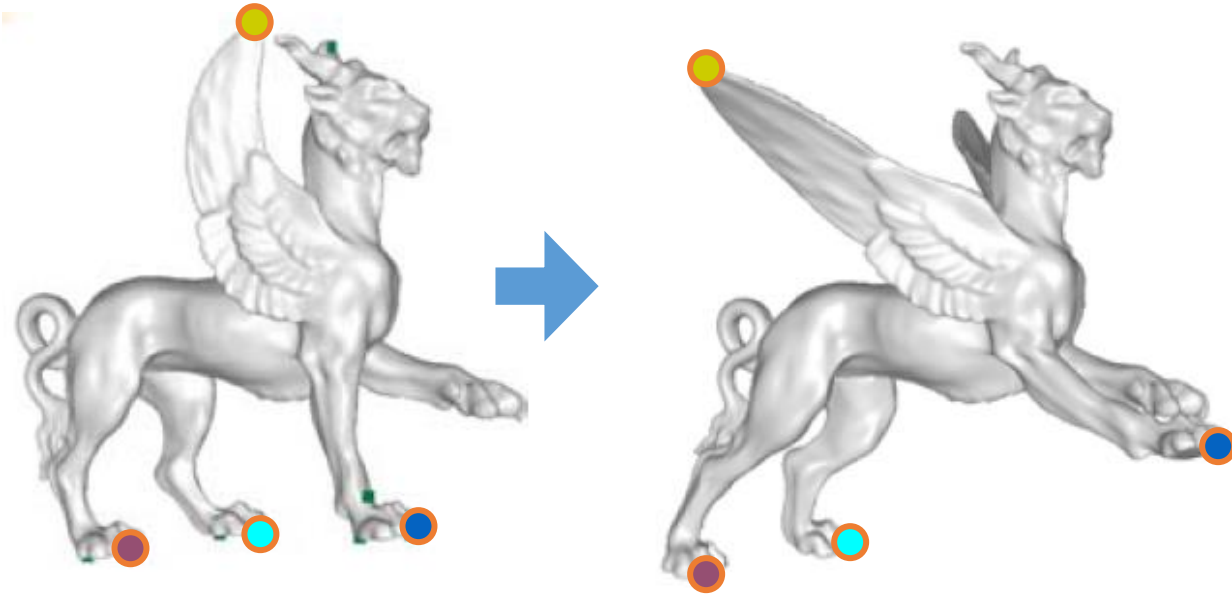
- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel



Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel



Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel



Thouis "Ray" Jones

How?

The Laplacian Operator

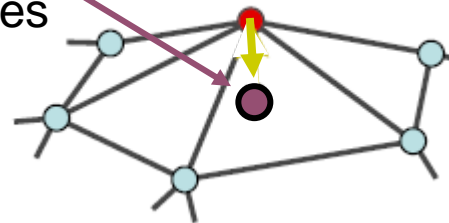


- Mesh formulation:

$$\delta_i = \frac{1}{d_i} \sum_{j \in N(i)} (\mathbf{v}_i - \mathbf{v}_j)$$

$d_i = |N(i)|$ is the number of neighbors.

Average of
Neighboring
Vertices



Olga Sorkine

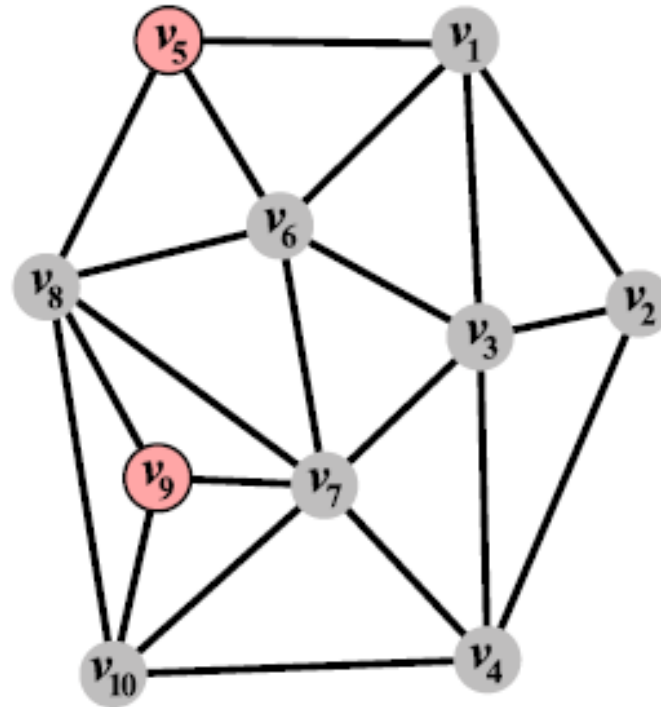
The Laplacian Operator

- The Laplacian operator Δ

$$L(v_i) = \Delta(v_i) = \frac{\sum_{j \in 1_{ring_i}} v_j - v_i}{\#1_{ring_i}}$$

- In matrix form:

$$L_{ij} = \begin{cases} -w_{ij} & i \neq j \\ \sum_{j \in 1_{ring_i}} w_{ij} & i = j \\ 0 & \text{else} \end{cases}$$



4	-1	-1	-1	-1					
-1	3	-1	-1						
-1	-1	5	-1	-1	-1				
	-1	-1	4		-1				-1
-1				3	-1		-1		
-1		-1		-1	5	-1	-1		
		-1	-1		-1	6	-1	-1	-1
				-1	-1	-1	5	-1	-1
						-1	-1	3	-1
			-1			-1	-1	-1	4

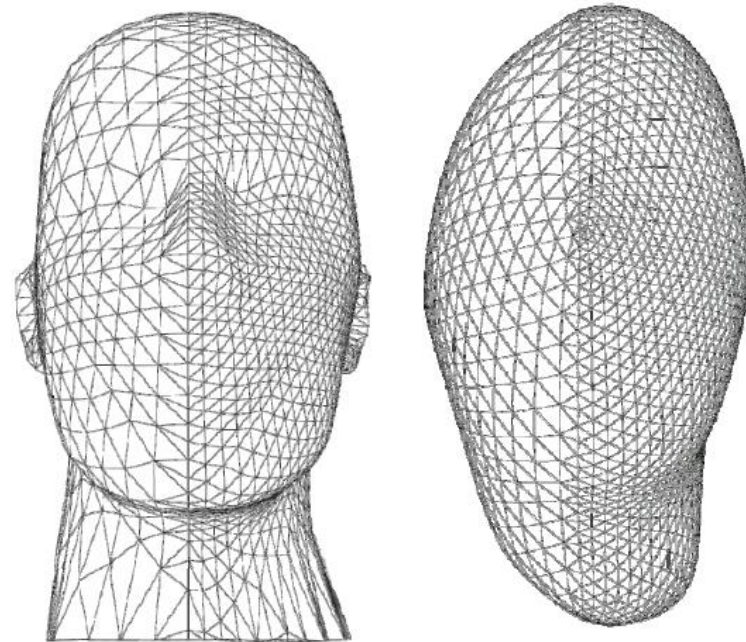
The Laplacian Operator



- The Laplacian operator Δ

$$L(v_i) = \Delta(v_i) = \frac{\sum_{j \in 1ring_i} v_j - v_i}{\#1ring_i}$$

- However, Meshes are irregular



The Laplacian Operator



- The Laplacian operator Δ

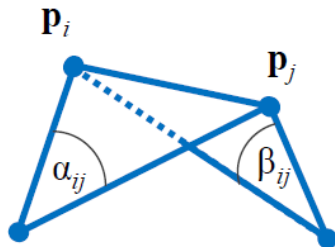
$$L(v_i) = \Delta(v_i) = \frac{\sum_{j \in 1ring_i} v_j - v_i}{\#1ring_i}$$

- However, Meshes are irregular

- Cotangent weights:

$$L(p_i) = \frac{\sum_{j \in 1ring_i} w_{ij} p_j}{\sum_{j \in 1ring_i} w_{ij}} - p_i$$

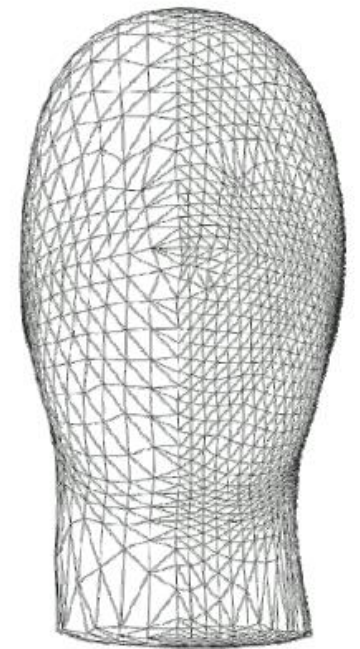
$$w_{ij} = \frac{\cot(\alpha_{ij}) + \cot(\beta_{ij})}{2}$$



Solve:

$$\left(\begin{array}{c|c} L & \\ \hline \omega I_{m \times m} & 0 \end{array} \right) \mathbf{x} = \left(\begin{array}{c} \delta^{(x)} \\ \omega c_{1:m} \end{array} \right)$$

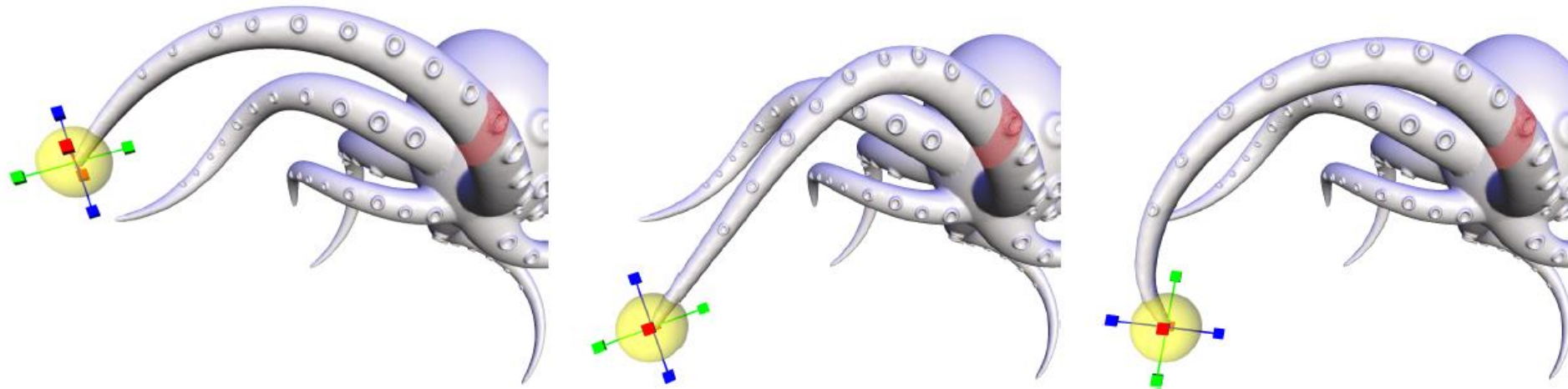
$$\tilde{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \left(\|L\mathbf{x} - \delta^{(x)}\|^2 + \sum_{j \in C} \omega^2 |x_j - c_j|^2 \right)$$



The Laplacian Operator



- Applicable to:
 - Deformation, by adding constraints



Polygonal Mesh Processing



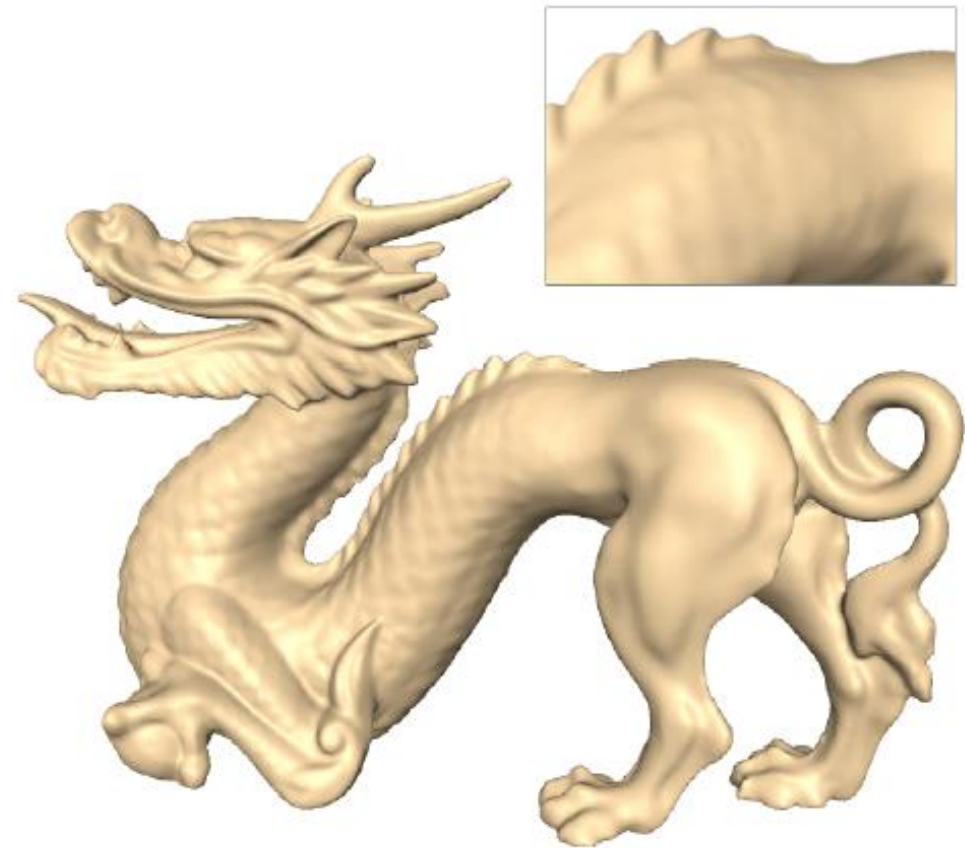
Deformation



The Laplacian Operator



- Applicable to:
 - Deformation, by adding constraints
 - Blending, by concatenating rows



The Laplacian Operator



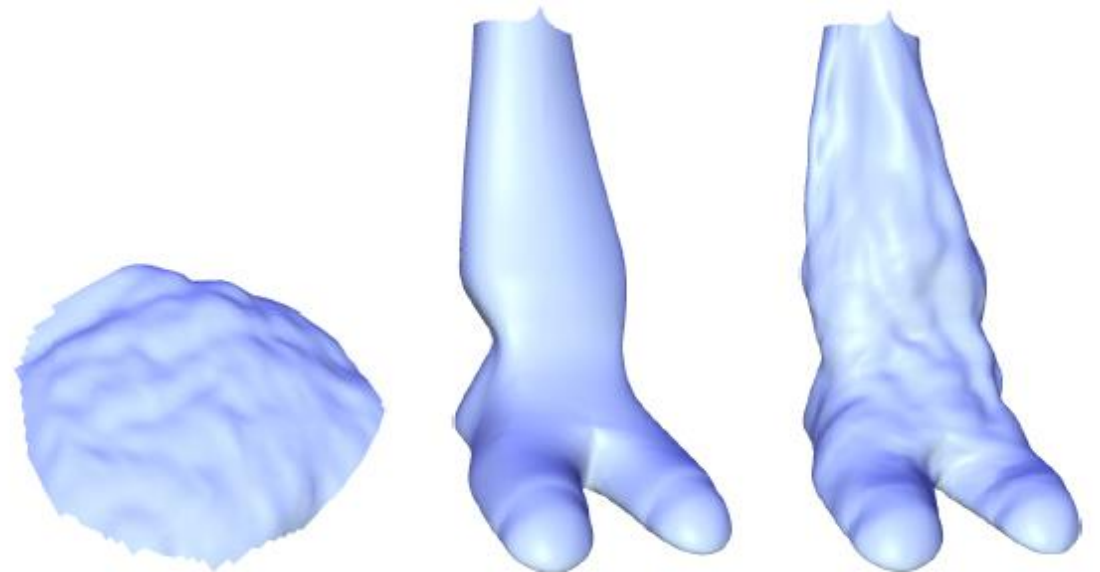
- Applicable to:
 - Deformation, by adding constraints
 - Blending, by concatenating rows
 - Hole filling, by 0's on the RHS



The Laplacian Operator



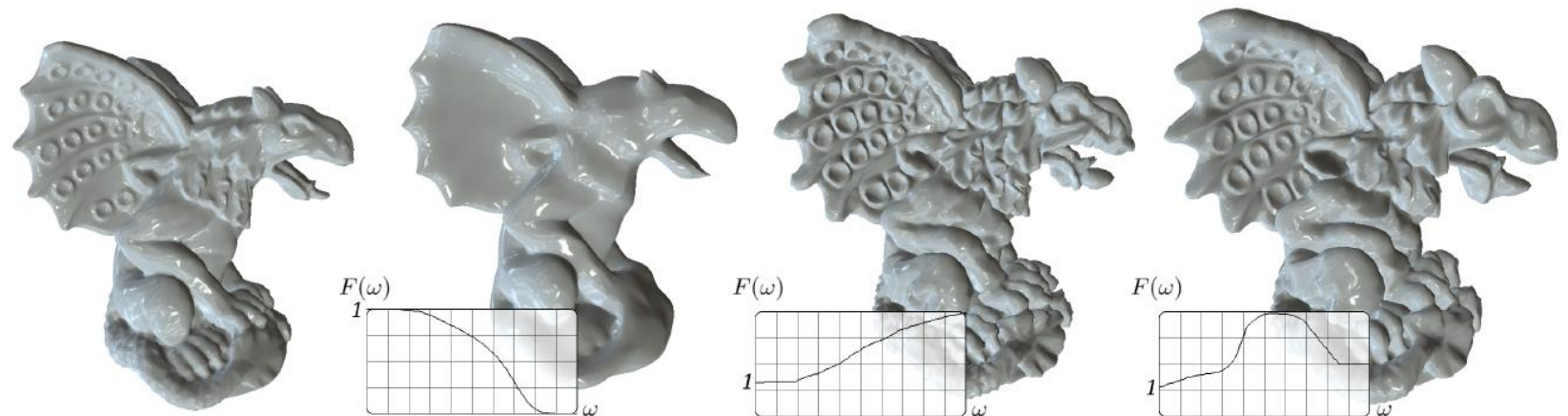
- Applicable to:
 - Deformation, by adding constraints
 - Blending, by concatenating rows
 - Hole filling, by 0's on the RHS
 - Coating (or detail transfer), by copying RHS values (after filtering)



The Laplacian Operator



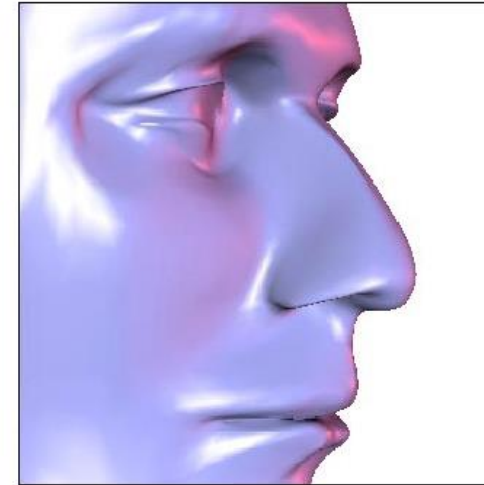
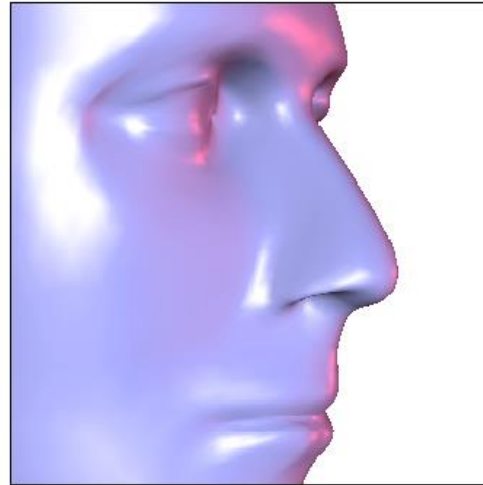
- Applicable to:
 - Deformation, by adding constraints
 - Blending, by concatenating rows
 - Hole filling, by 0's on the RHS
 - Coating (or detail transfer), by copying RHS values (after filtering)
 - Spectral mesh processing, through eigen analysis



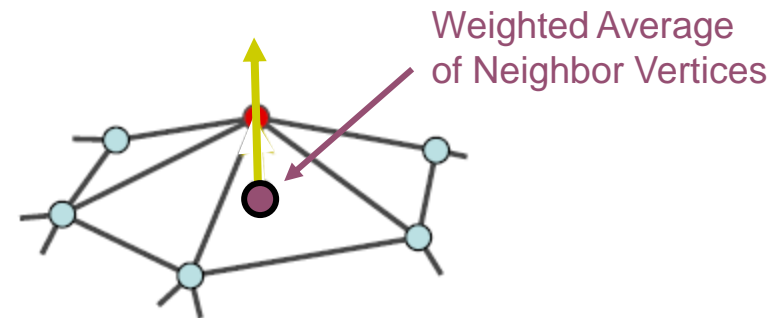
Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel



Desbrun

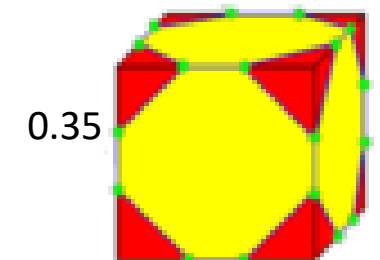
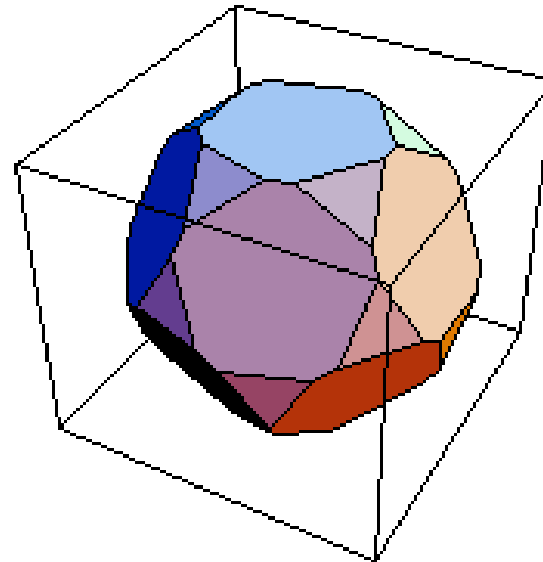
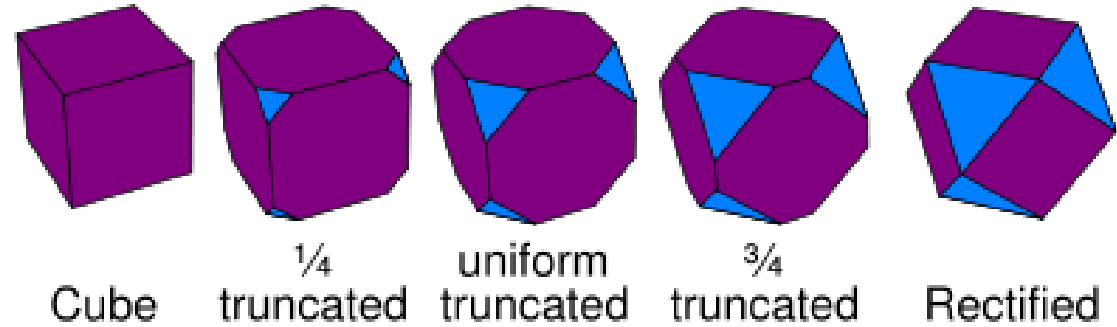


Olga Sorkine

Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

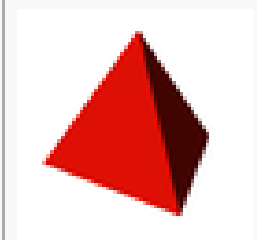
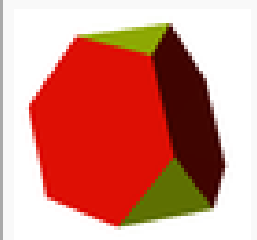
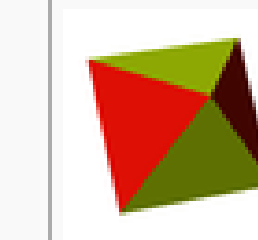
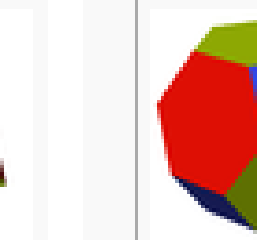
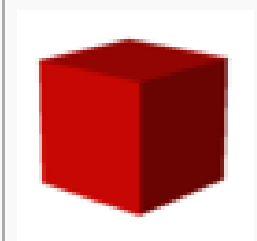
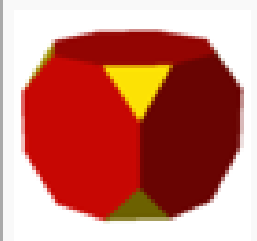
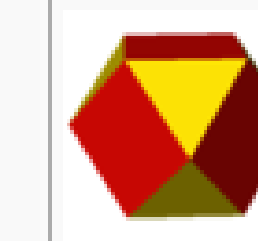
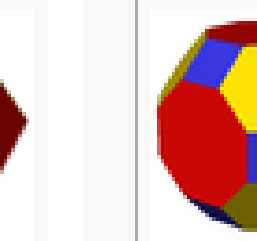
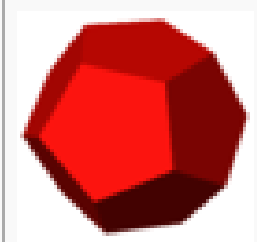
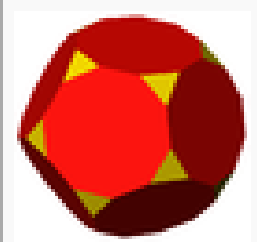
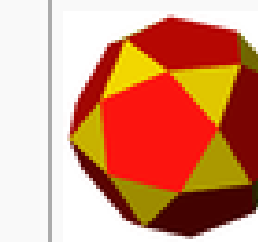
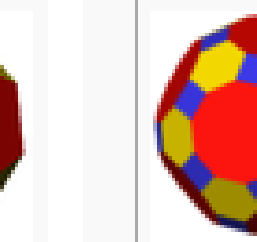


Conway

Polygonal Mesh Processing



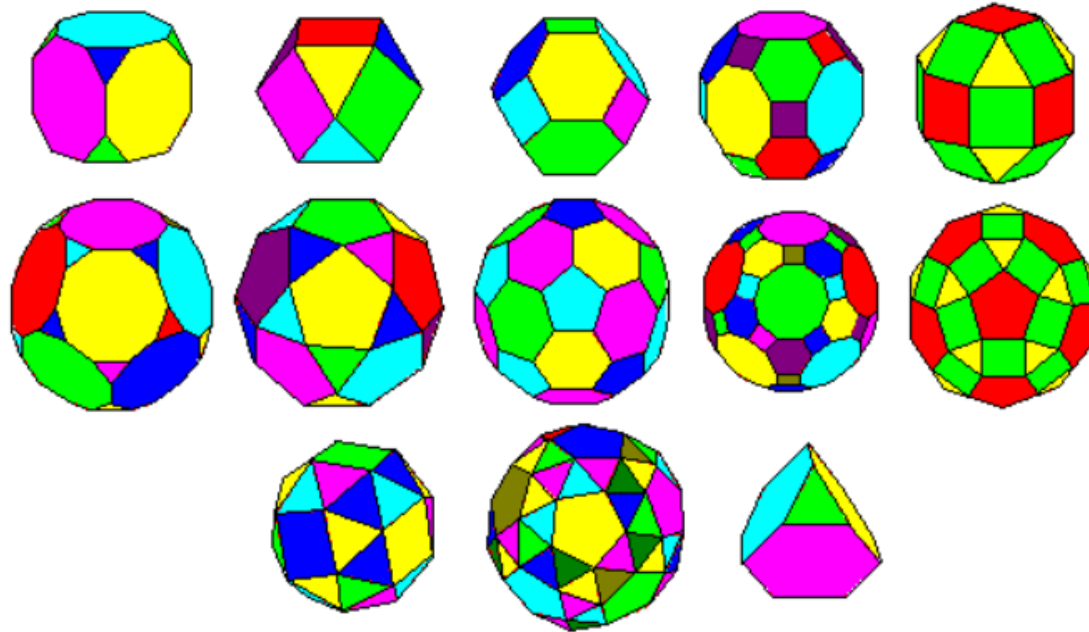
- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - **Truncate**
 - Bevel

 {3,3}	 (3.6.6)	 (3.3.3.3)	 (4.6.6)
 {4,3}	 (3.8.8)	 (3.4.3.4)	 (4.6.8)
 {5,3}	 (3.10.10)	 (3.5.3.5)	 (4.6.10)

Polygonal Mesh Processing



- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - **Truncate**
 - Bevel



Archimedean Polyhedra

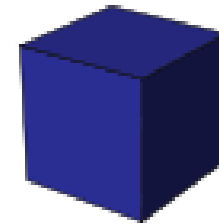
<http://www.uwgb.edu/dutchs/symmetry/archpol.htm>

Polygonal Mesh Processing

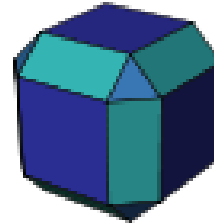


- Analysis
 - Normals
 - Curvature
- Warps
 - Rotate
 - Deform
- Filters
 - Smooth
 - Sharpen
 - Truncate
 - Bevel

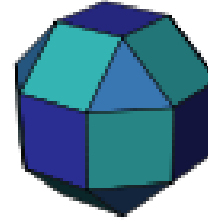
Wikipedia



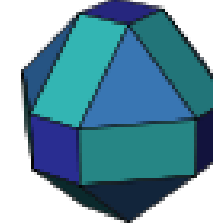
(regular polyhedron)
Cube



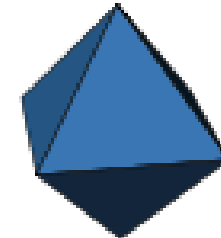
1/2 cantellated
(beveled cube)



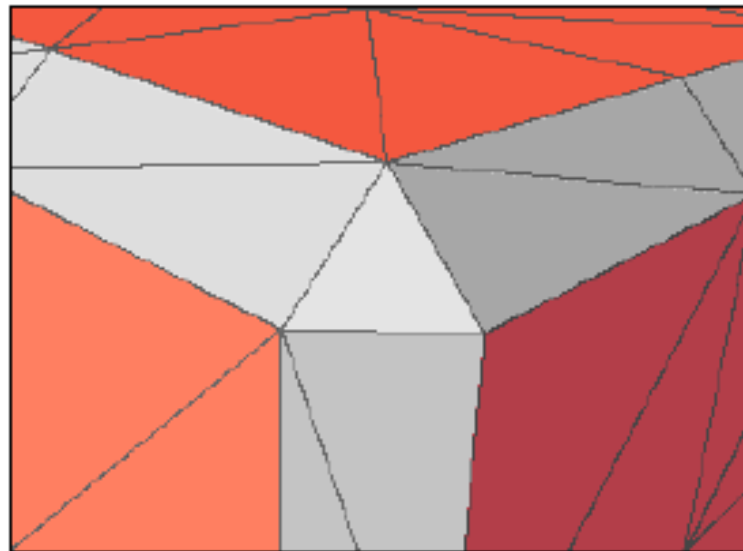
Uniform cantellation
Rhombicuboctahedron



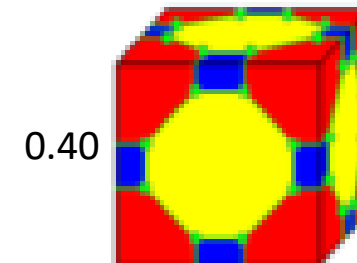
3/4 cantellated
(beveled octahedron)



(regular dual)
Octahedron



Jarek Rossignac



0.40

Conway

Polygonal Mesh Processing



- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract

Polygonal Mesh Processing



- **Remeshing**

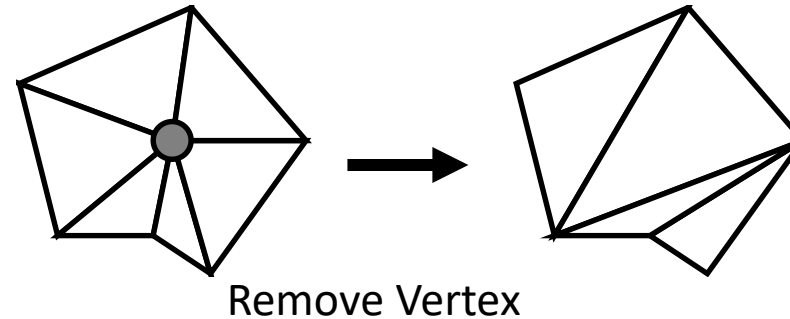
- Subdivide
- Resample
- Simplify

- Topological fixup

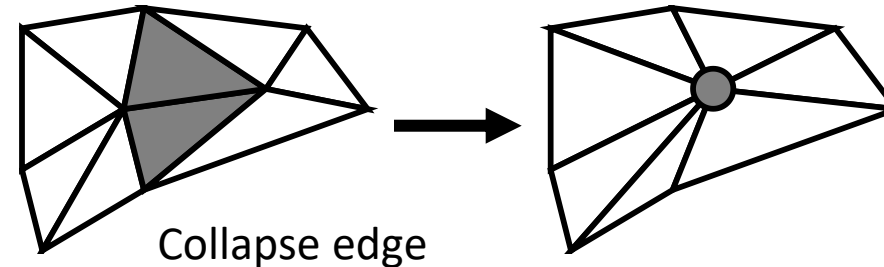
- Fill holes
- Fix self-intersections

- Boolean operations

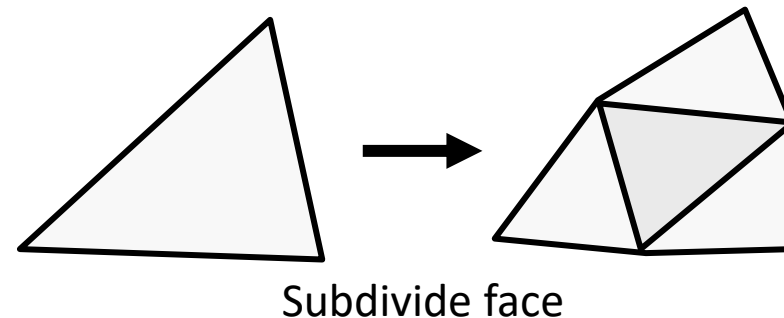
- Crop
- Subtract



Remove Vertex



Collapse edge

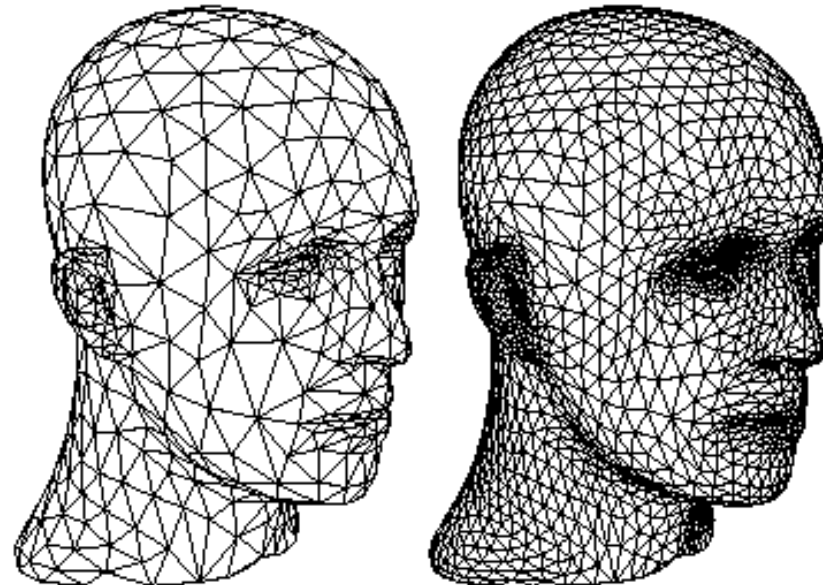
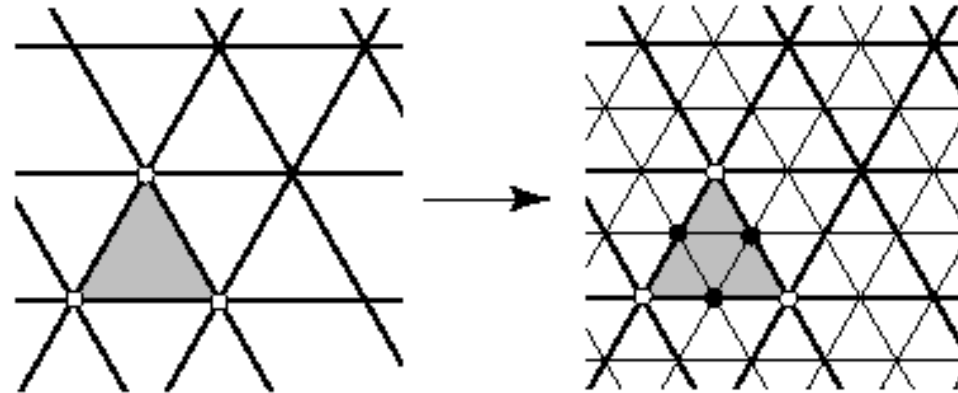


Subdivide face

Polygonal Mesh Processing



- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract



Polygonal Mesh Processing



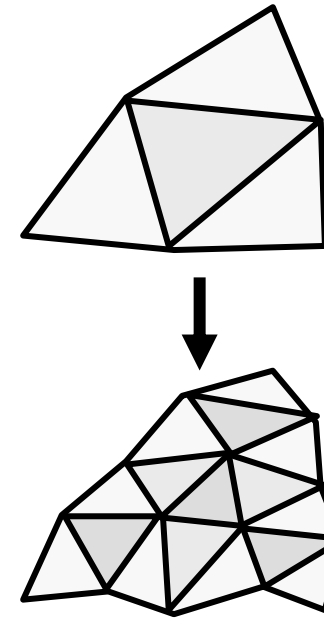
- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract



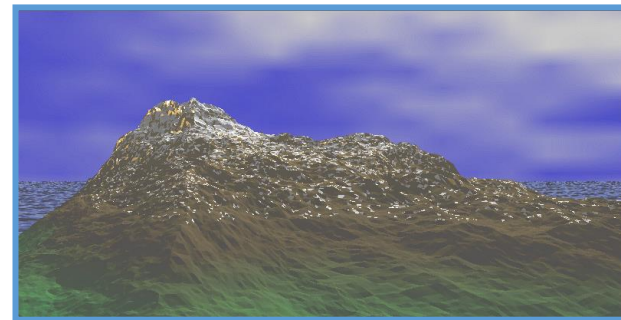
Polygonal Mesh Processing



- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract



Fractal Landscape

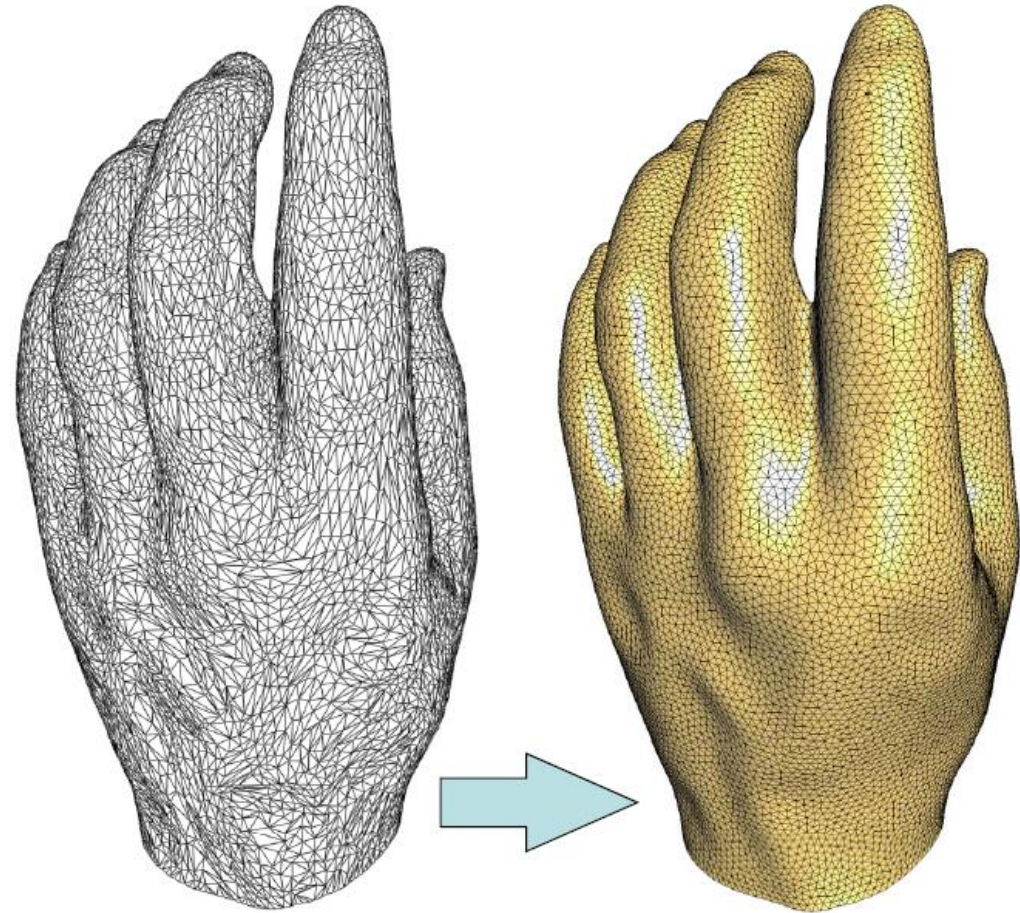


*Dirk Balfanz, Igor Guskov,
Sanjeev Kumar, & Rudro Samanta,*

Polygonal Mesh Processing



- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract



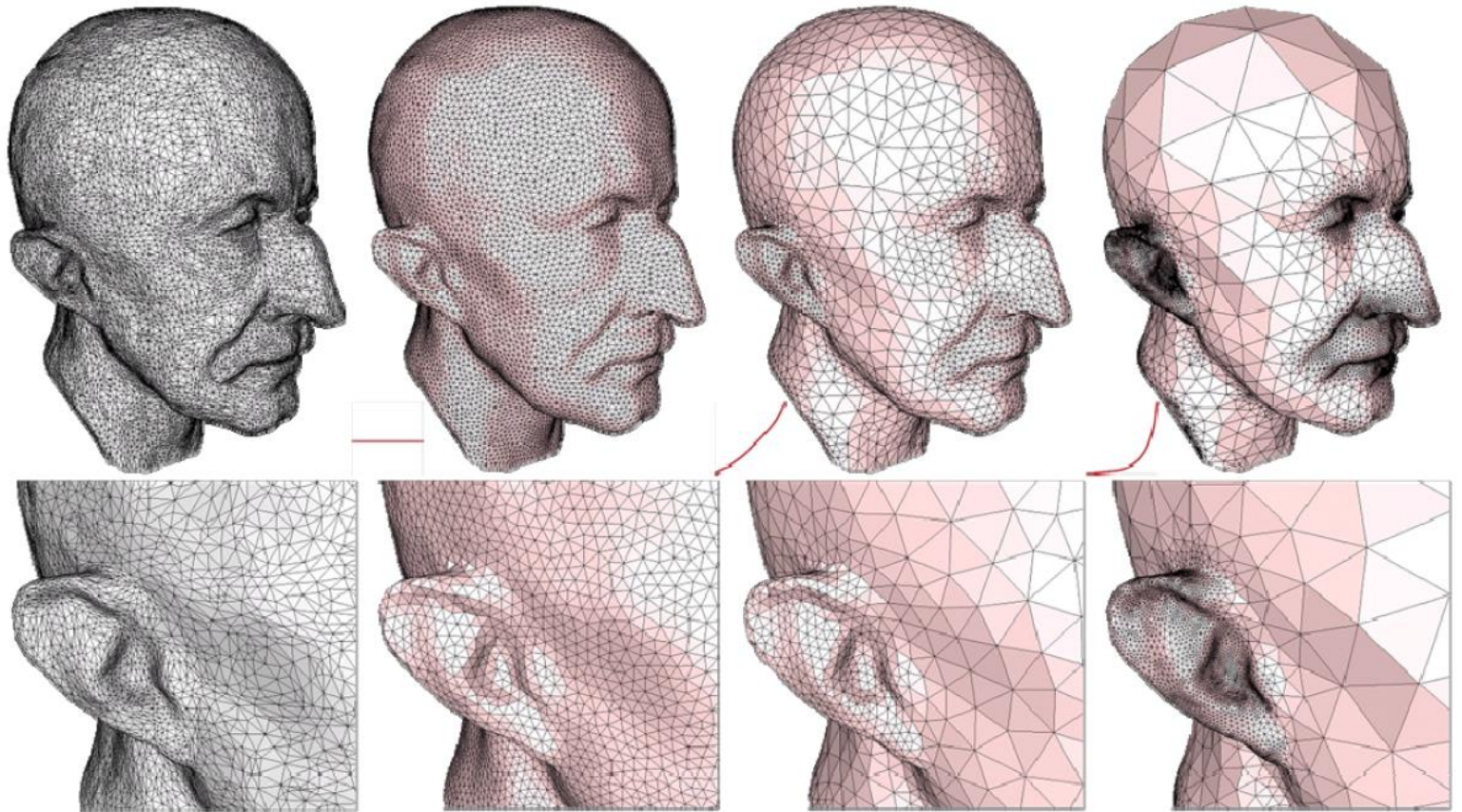
- more uniform distribution
- triangles with nicer aspect

Stanford

Polygonal Mesh Processing



- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract



Input

Uniform

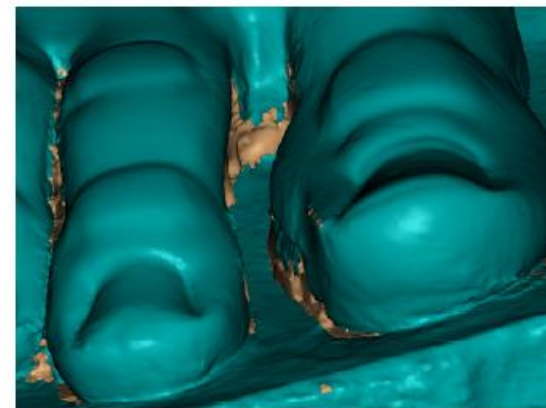
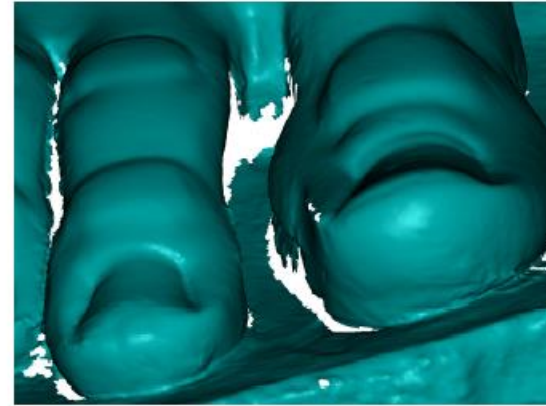
Adaptive

Stanford

Polygonal Mesh Processing



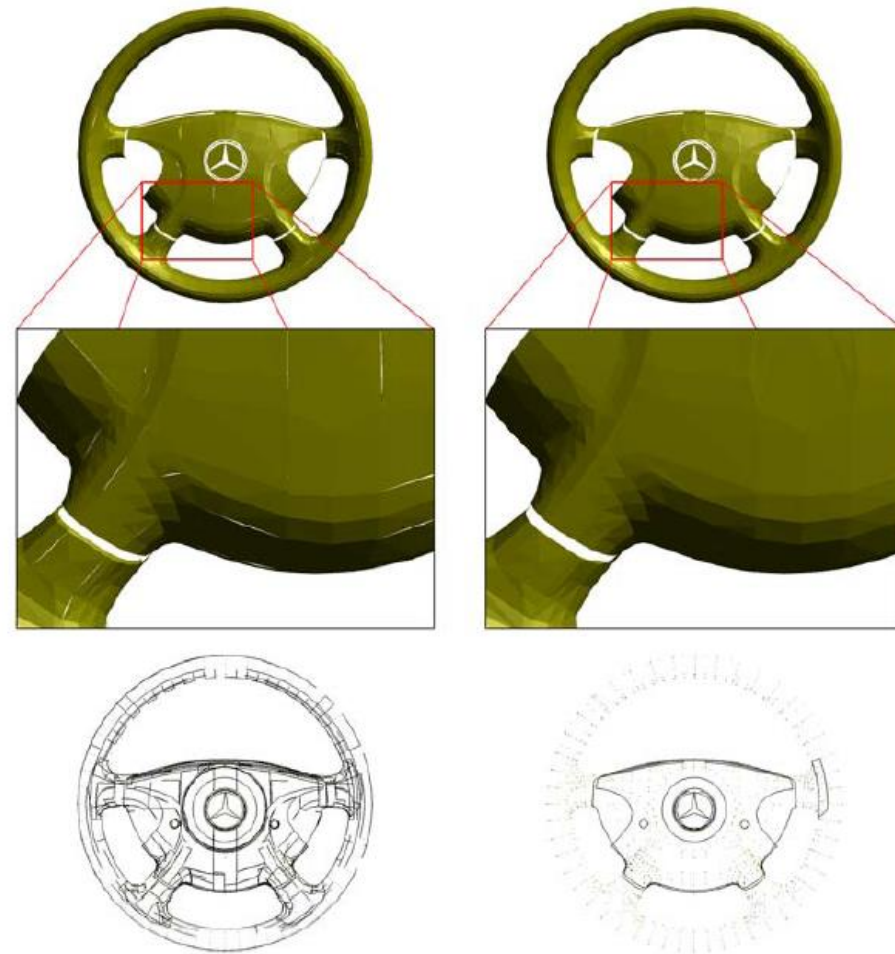
- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract



Polygonal Mesh Processing



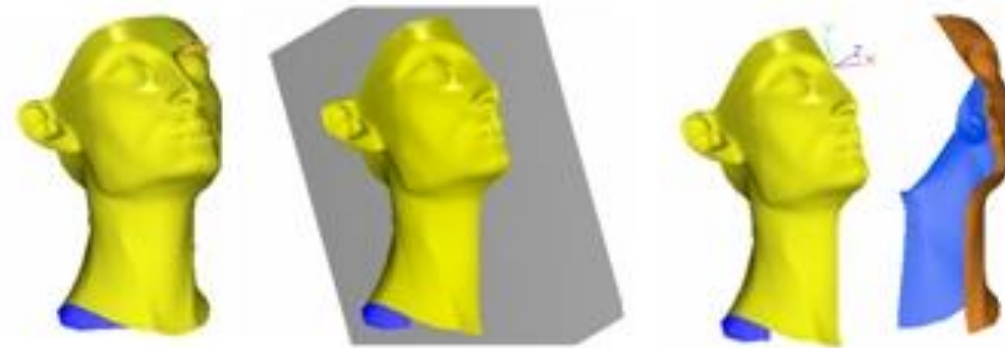
- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract



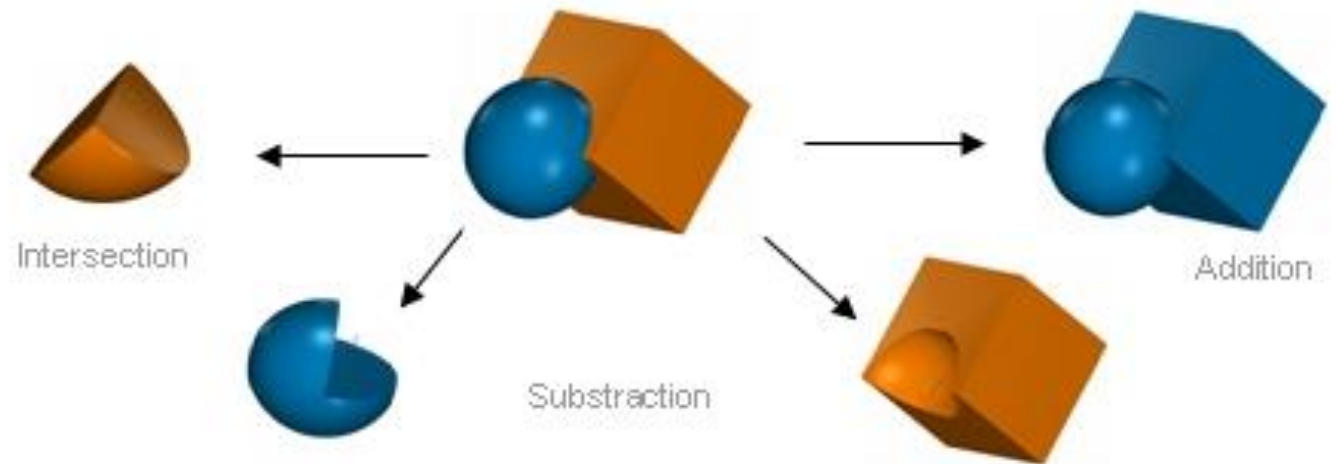
Polygonal Mesh Processing



- Remeshing
 - Subdivide
 - Resample
 - Simplify
- Topological fixup
 - Fill holes
 - Fix self-intersections
- Boolean operations
 - Crop
 - Subtract
 - Etc.



Mesh separation processed by a boolean operation.



Several Boolean operations with 3DReshaper®

Summary



- Polygonal meshes
 - Most common surface representation
 - Fast rendering
- Processing operations
 - Must consider irregular vertex sampling
 - Must handle/avoid topological degeneracies
- Representation
 - Which adjacency relationships to store depend on which operations must be efficient

3D Polygonal Meshes



- Properties

- ? Efficient display
- ? Easy acquisition
- ? Accurate
- ? Concise
- ? Intuitive editing
- ? Efficient editing
- ? Efficient intersections
- ? Guaranteed validity
- ? Guaranteed smoothness
- ? etc.



Viewpoint

3D Polygonal Meshes



- Properties

- ☺ Efficient display
- ☺ Easy acquisition
- ☹ Accurate
- ☹ Concise
- ☹ Intuitive editing
- ☹ Efficient editing
- ☹ Efficient intersections
- ☹ Guaranteed validity
- ☹ Guaranteed smoothness



Viewpoint