Polygonal Meshes

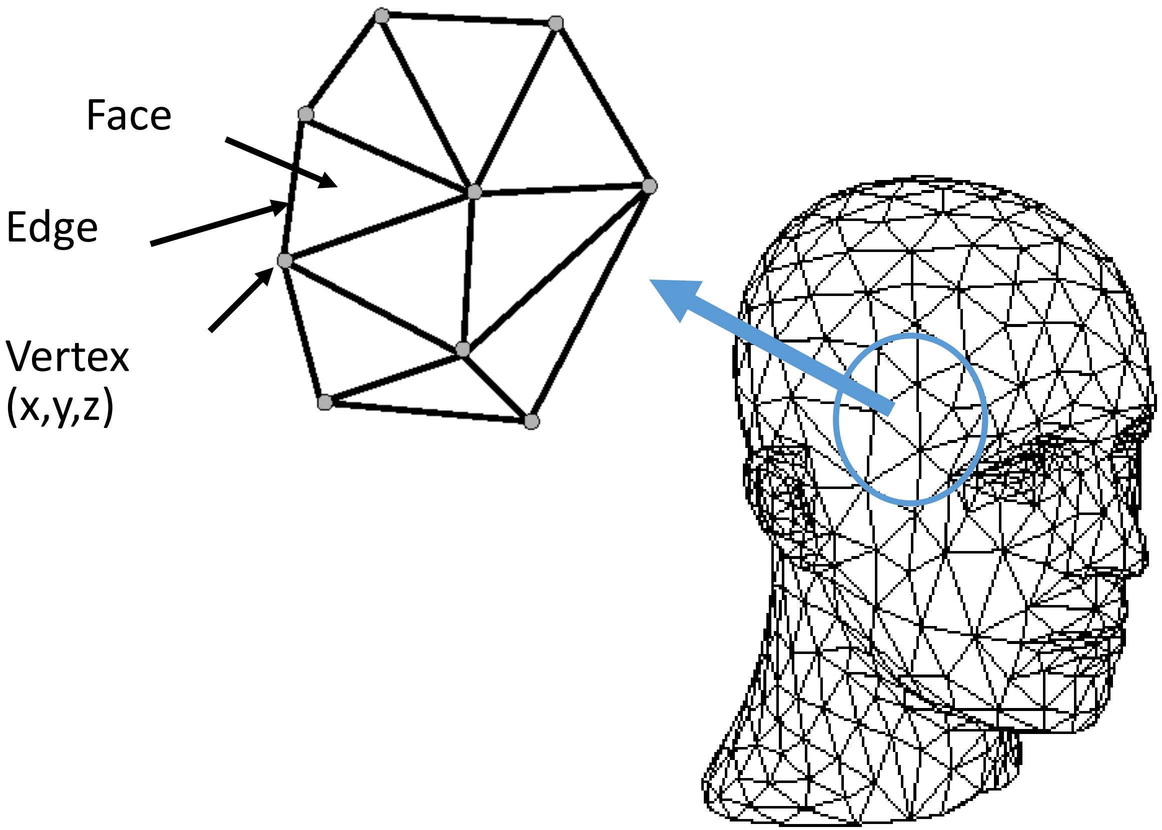
COS 426, Spring 2017
Princeton University
Amit Bermano

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



Zorin & Schroeder

3D Polygonal Mesh

- The power of polygonal meshes

DEADPOOL

20TH CENTURY FOX (2016)

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

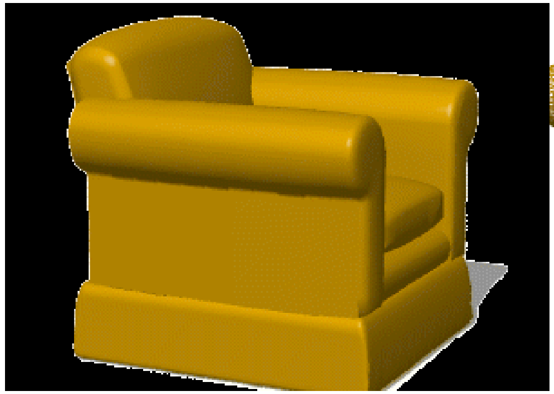
Viewpoint

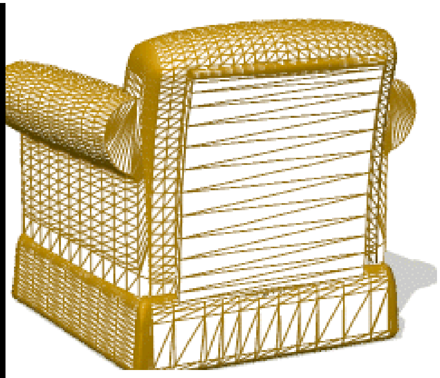
3D Polygonal Meshes

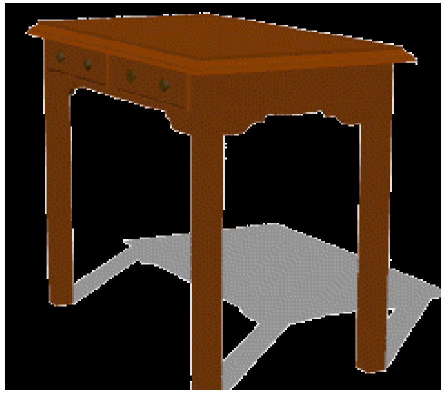
Meshlab Demo

3D Polygonal Meshes

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







Viewpoint

Outline

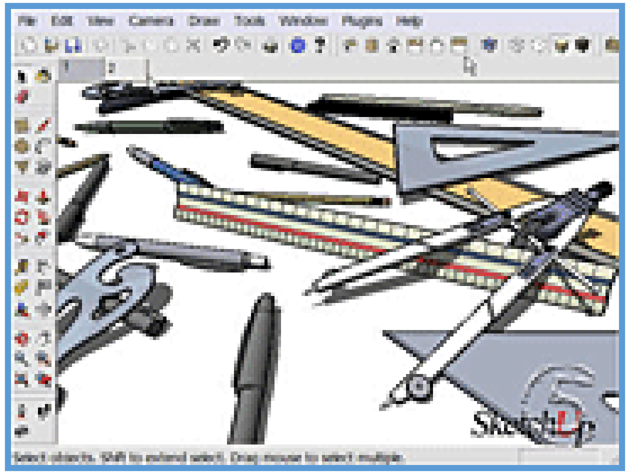
- Acquisition ←
- Representation
- Processing

Polygonal Mesh Acquisition

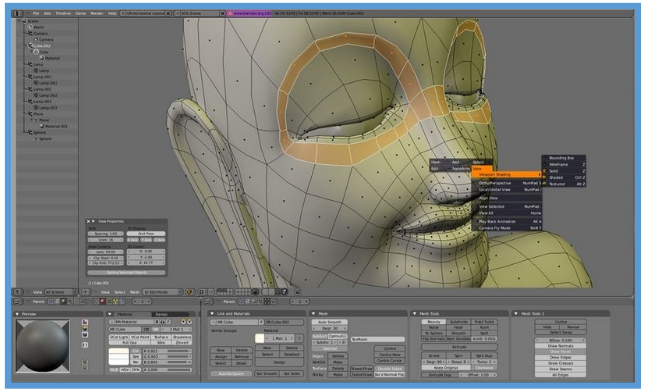
- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations

Polygonal Mesh Acquisition

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



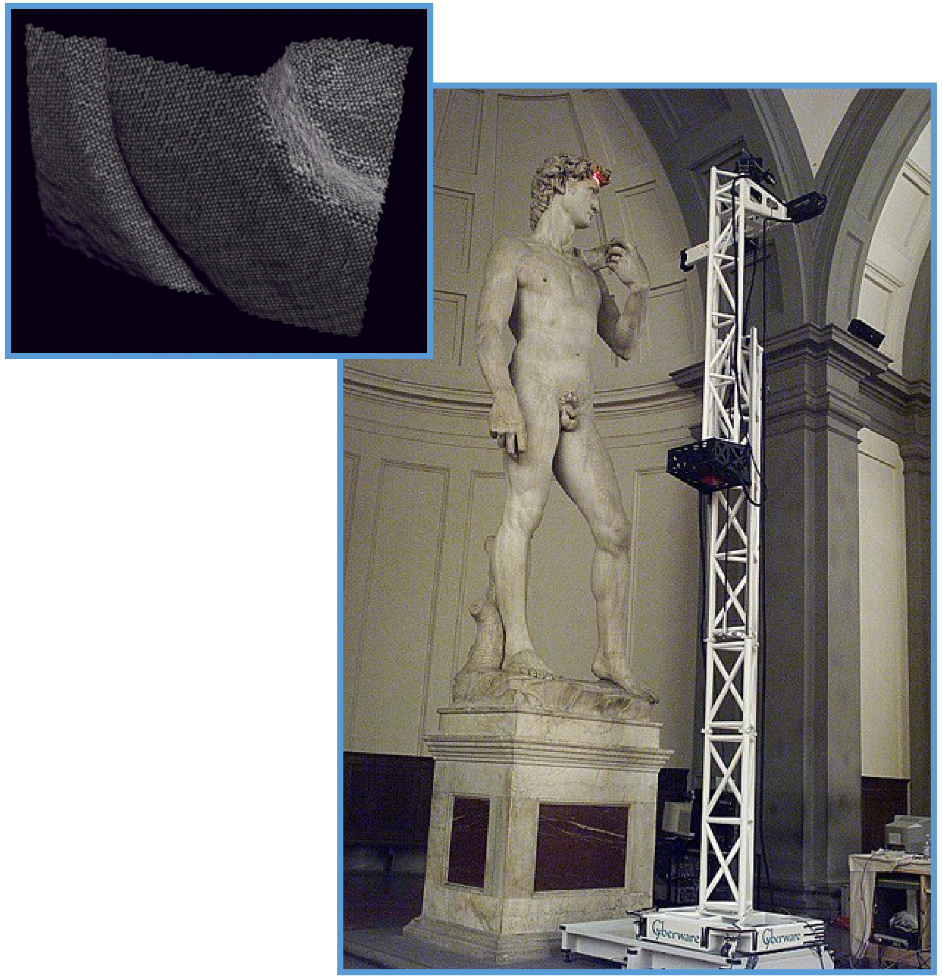
Sketchup



Blender

Polygonal Mesh Acquisition

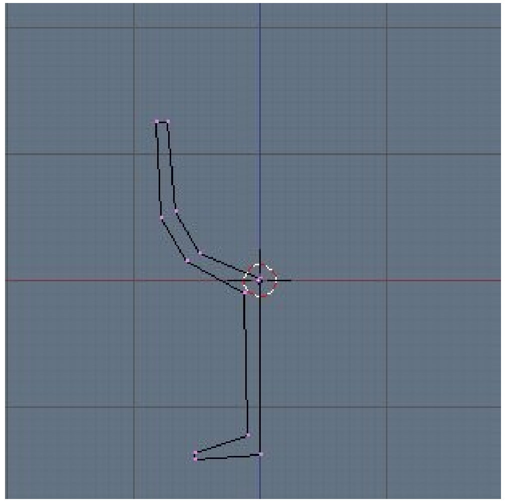
- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations

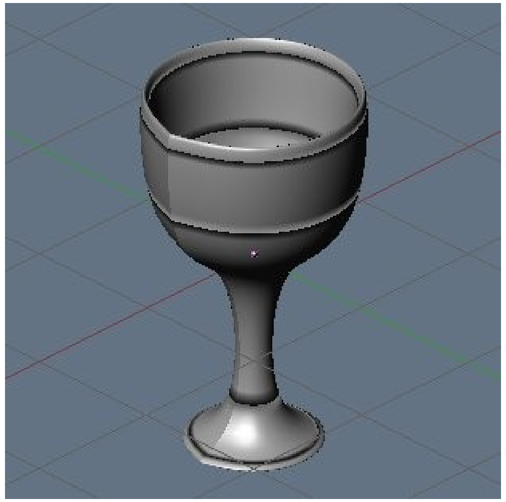


Digital Michelangelo Project
Stanford

Polygonal Mesh Acquisition

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





sphynx.co.uk

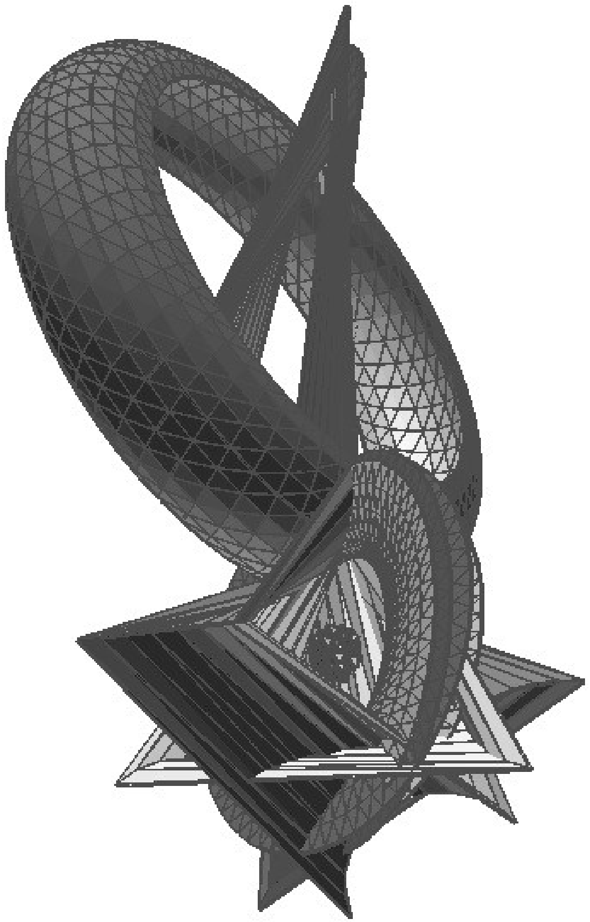
Polygonal Mesh Acquisition

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

Nicky Robinson, COS 426, 2014

Polygonal Mesh Acquisition

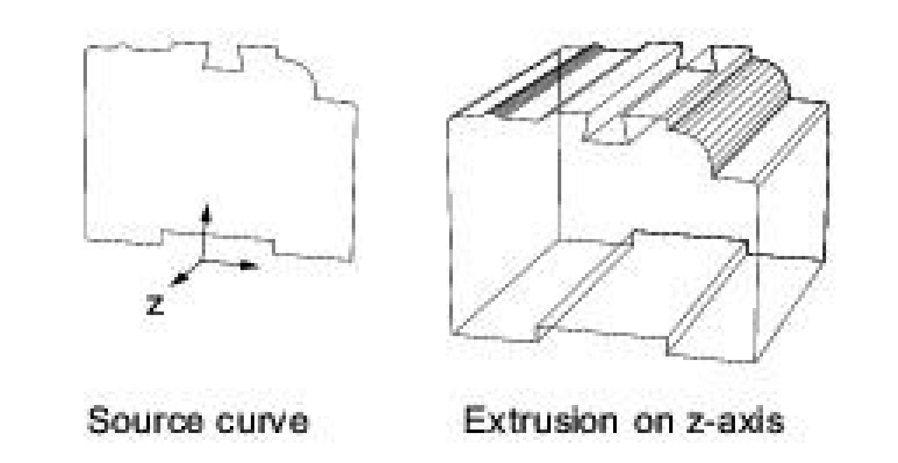
- Interactive modeling
- Scanners
- Procedural generation
- Conversion
- Simulations

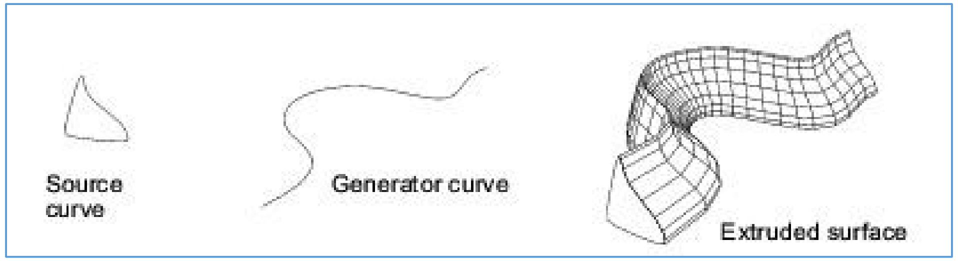


Alejandro Van Zandt-Escobar, COS 426, 2014

Polygonal Mesh Acquisition

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



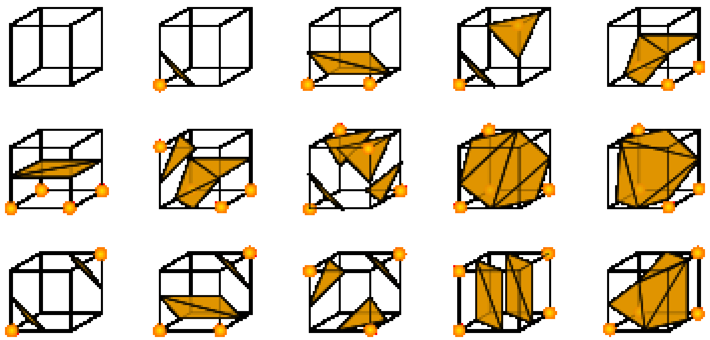


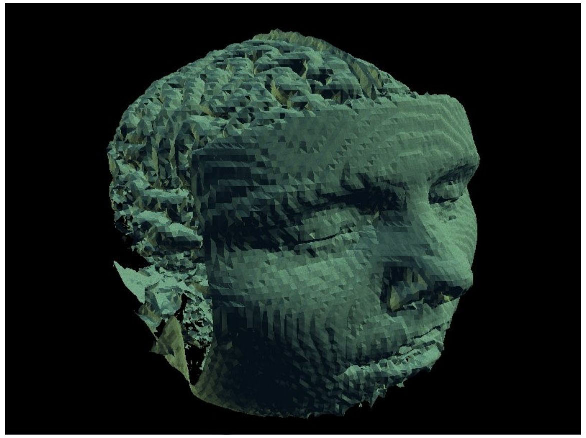
Polygonal Mesh Acquisition

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

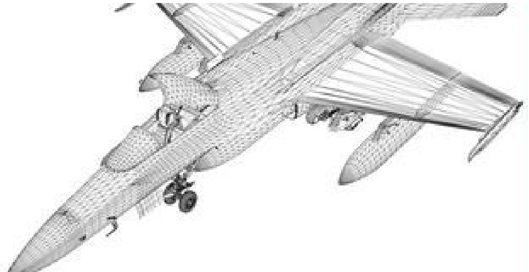
Polygonal Mesh Acquisition

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





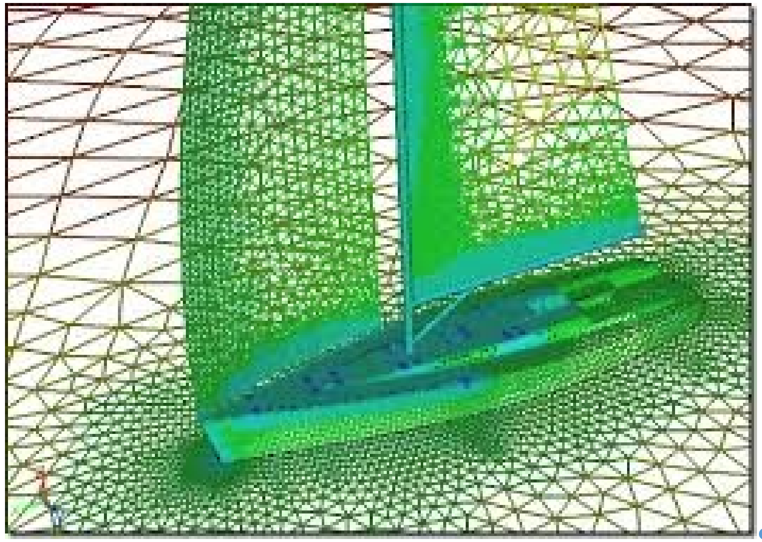
Marching cubes



Jose Maria De Espona

Polygonal Mesh Acquisition

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



symscape

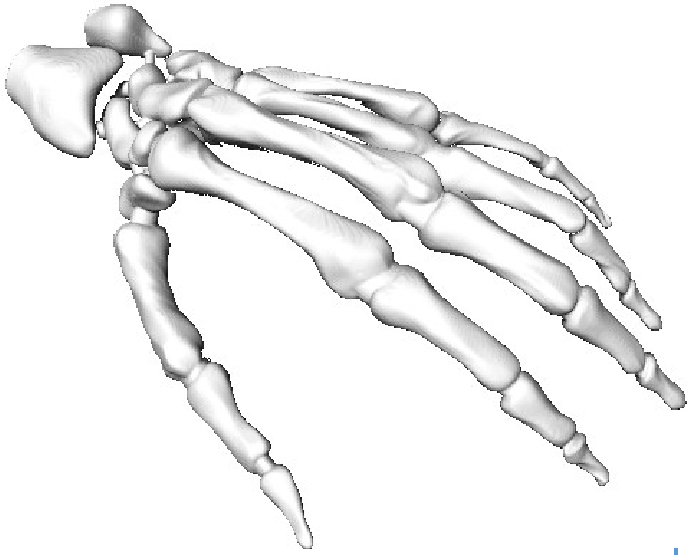
Lee et. al 2010

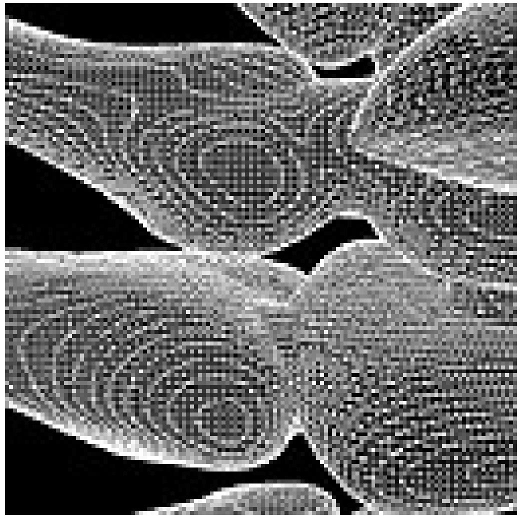
Outline

- Acquisition
- Representation ←
- Processing

Polygon Mesh Representation

- Important properties of mesh representation?

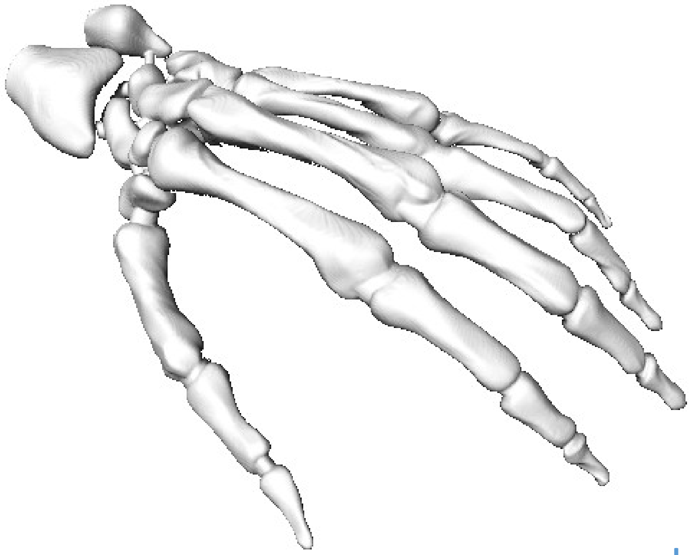


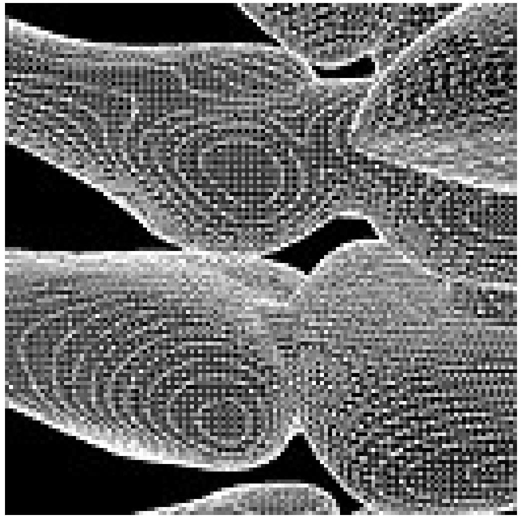


Large Geometric Model Repository
Georgia Tech

Polygon Mesh Representation

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

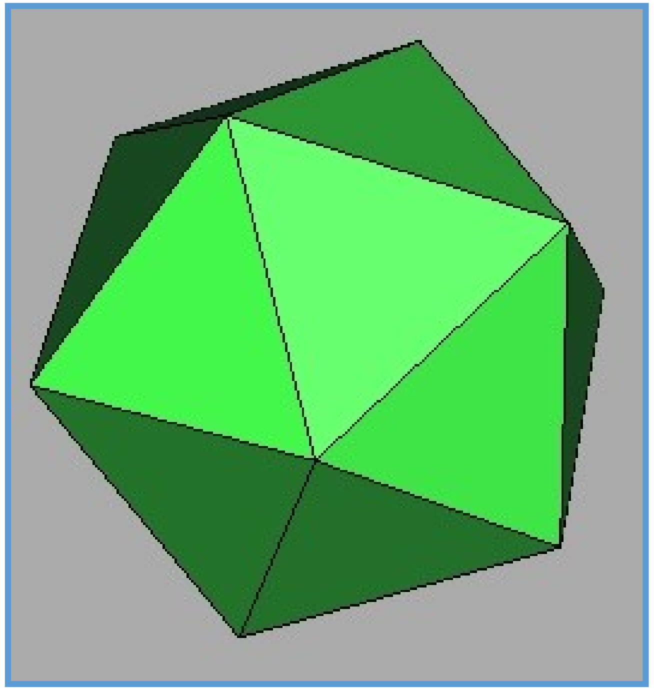




Large Geometric Model Repository
Georgia Tech

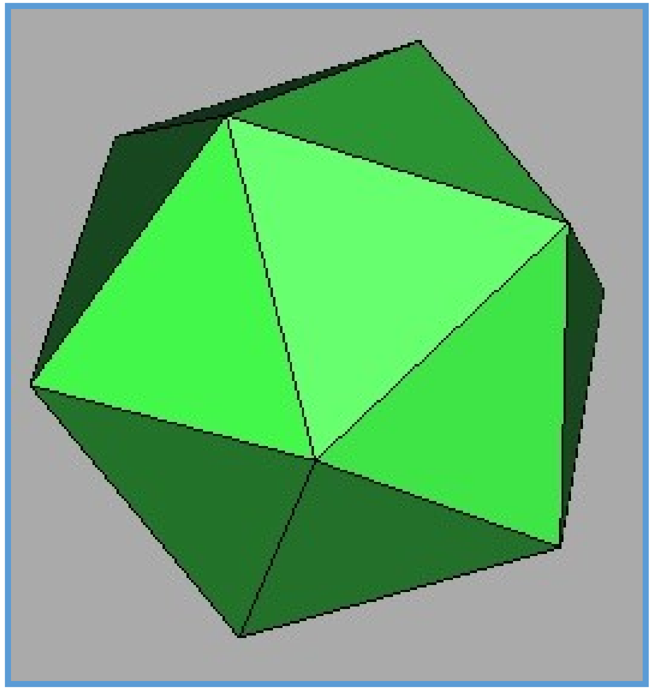
Polygon Mesh Representation

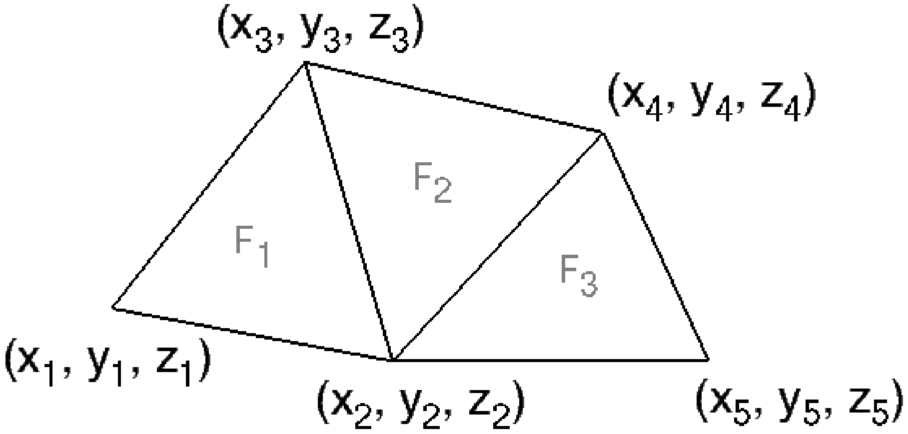
- Possible data structures



Independent Faces

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

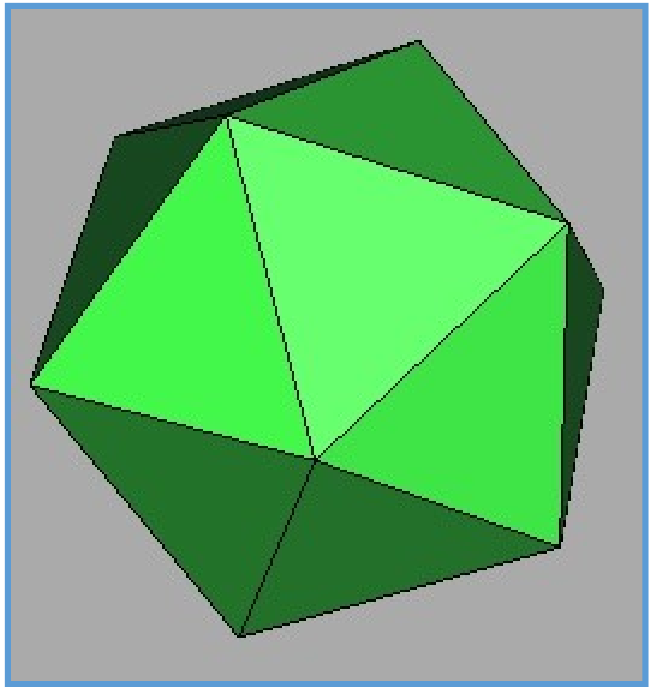


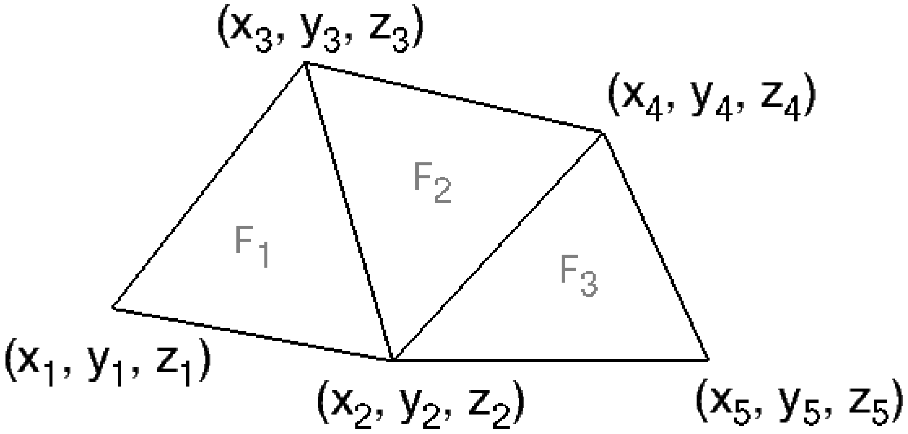


FACE TABLE	
F_1	(x_1, y_1, z_1) (x_2, y_2, z_2) (x_3, y_3, z_3)
F_2	(x_2, y_2, z_2) (x_4, y_4, z_4) (x_3, y_3, z_3)
F_3	(x_2, y_2, z_2) (x_5, y_5, z_5) (x_4, y_4, z_4)

Vertex and Face Tables (Indexed Vertices)

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



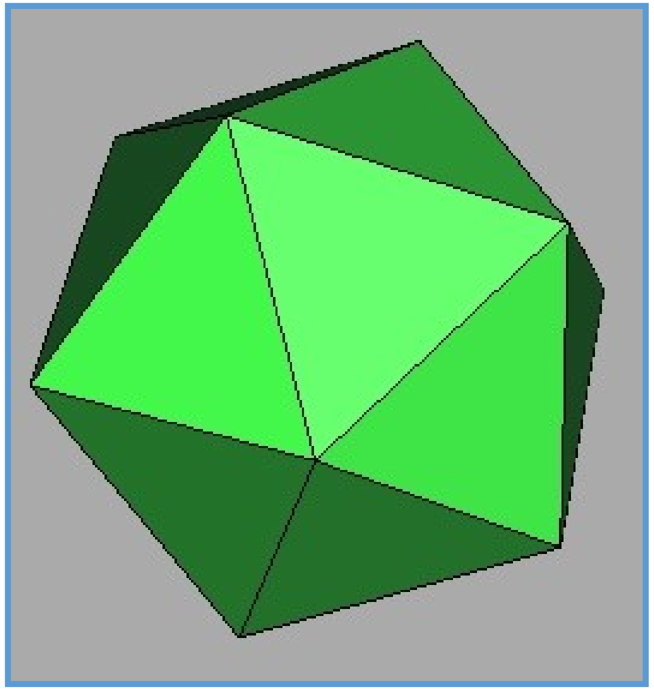


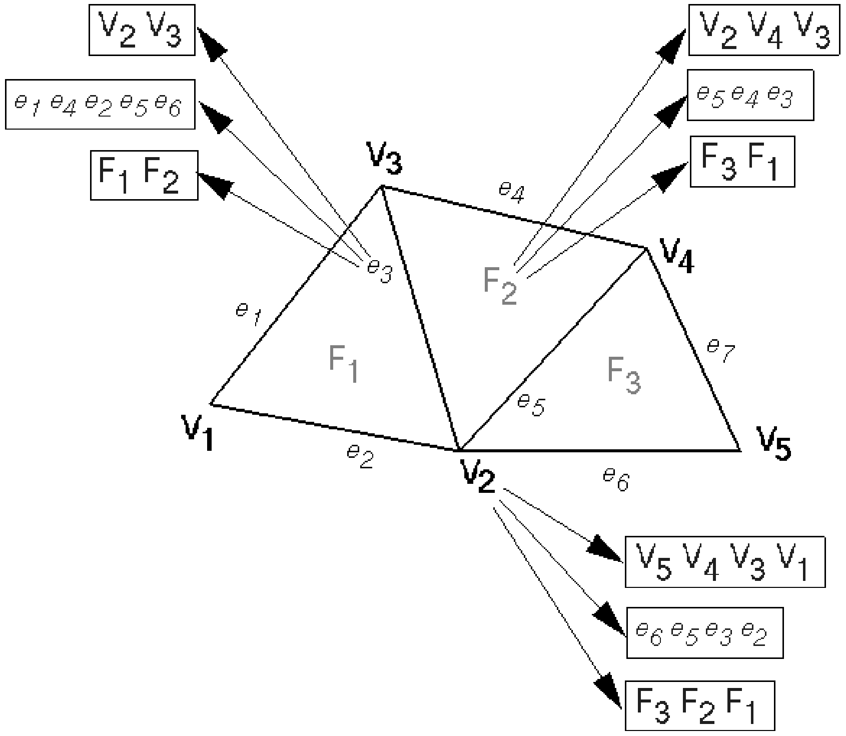
VERTEX TABLE			
V_1	X_1	Y_1	Z_1
V_2	X_2	Y_2	Z_2
V_3	X_3	Y_3	Z_3
V_4	X_4	Y_4	Z_4
V_5	X_5	Y_5	Z_5

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

Adjacency Lists

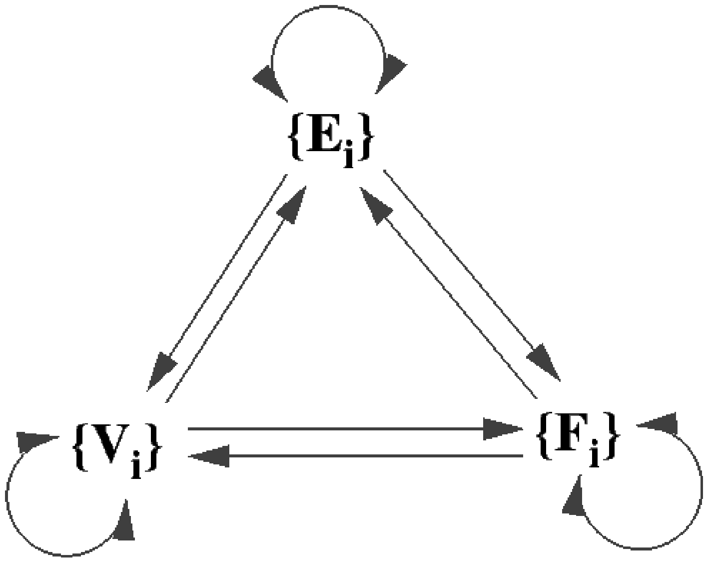
- Store all vertex, edge, and face adjacencies
  - Efficient adjacency traversal
  - Extra storage

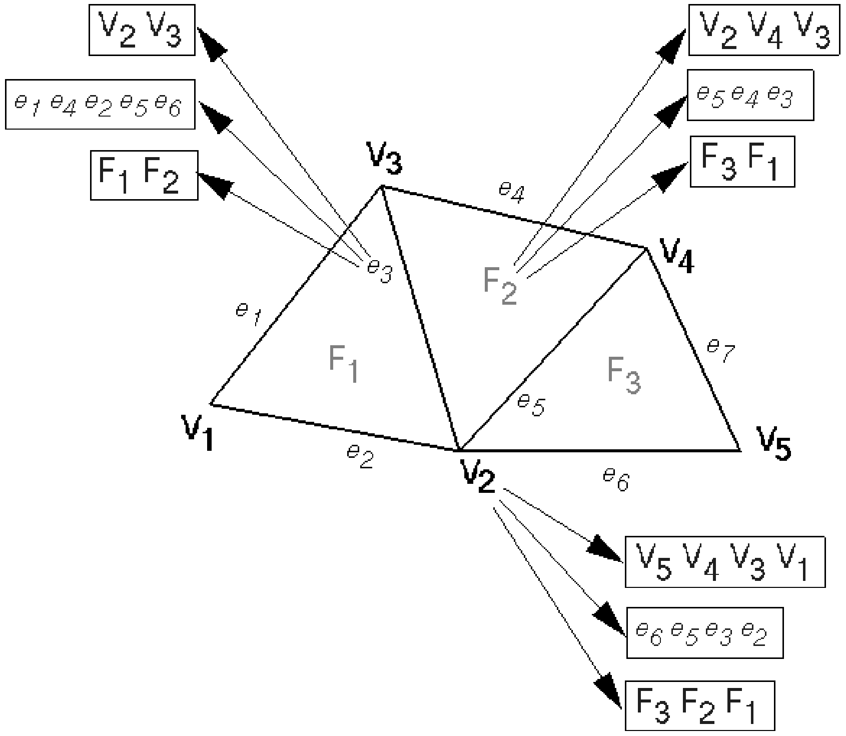




Partial Adjacency Lists

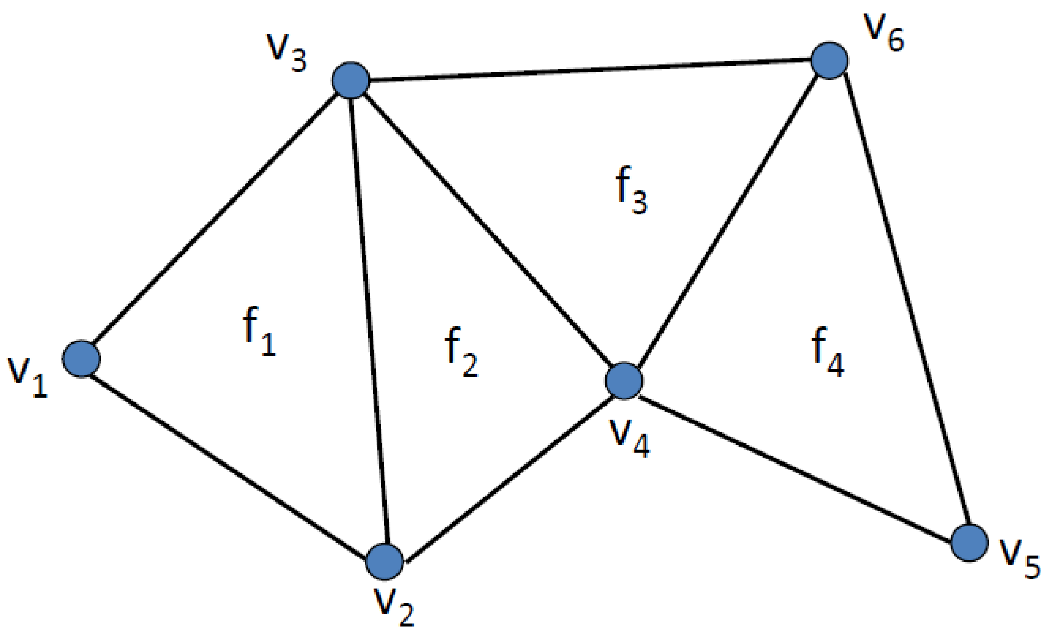
- Can we store only some adjacency relationships and derive others?





Adjacency Matrix

- If there is an edge between v_i & v_j then $A_{ij} = 1$
- Cons:
  - No connection between a vertex and its adjacent faces
  - A lot of storage (should use sparse matrices)

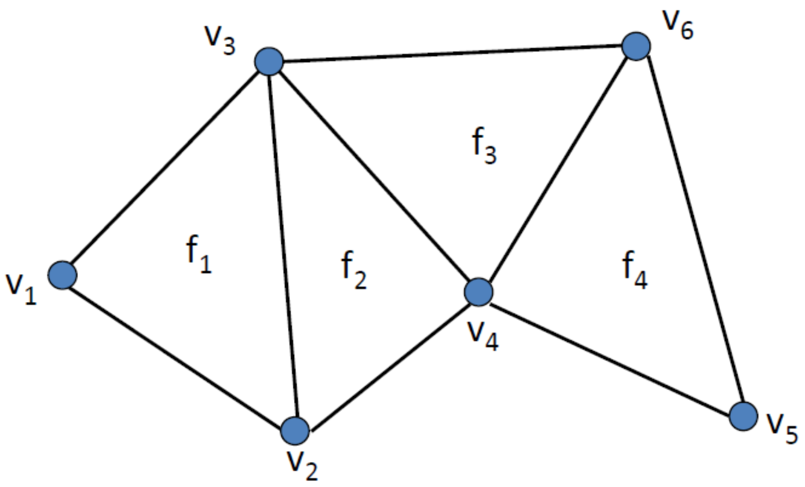


A =

	v_1	v_2	v_3	v_4	v_5	v_6
v_1		1	1			
v_2	1		1	1		
v_3	1	1		1		1
v_4		1	1		1	1
v_5				1		1
v_6			1	1	1	

Adjacency Matrix

- If there is an edge between v_i & v_j then $A_{ij} = 1$
- Pros:
  - No restrictions on mesh topology
  - Mathematical properties:
    - $(A^n)_{ij}$ = # paths of length n from v_i to v_j
    - Eigenvectors are meaningful

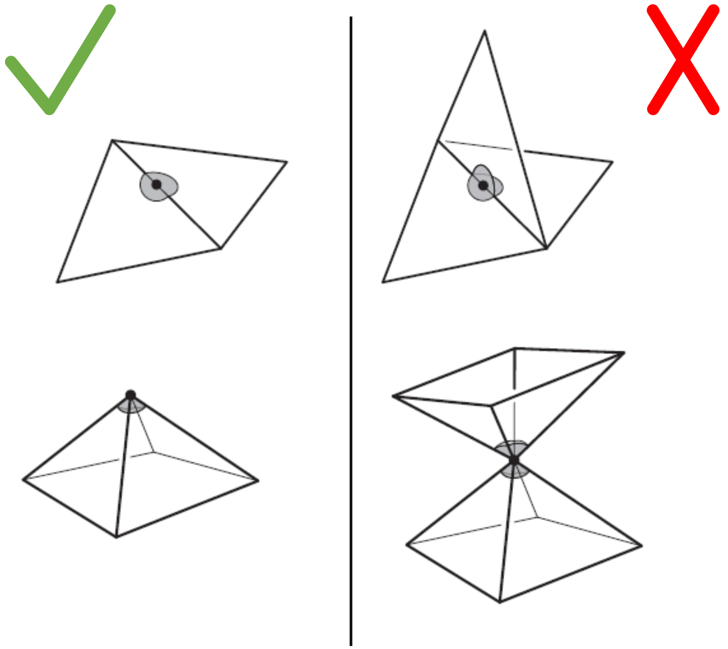


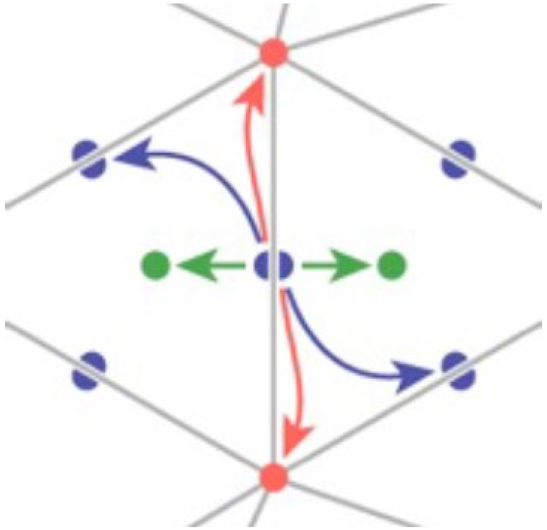
A =

	v_1	v_2	v_3	v_4	v_5	v_6
v_1		1	1			
v_2	1		1	1		
v_3	1	1		1		1
v_4		1	1		1	1
v_5				1		1
v_6			1	1	1	

Half Edge

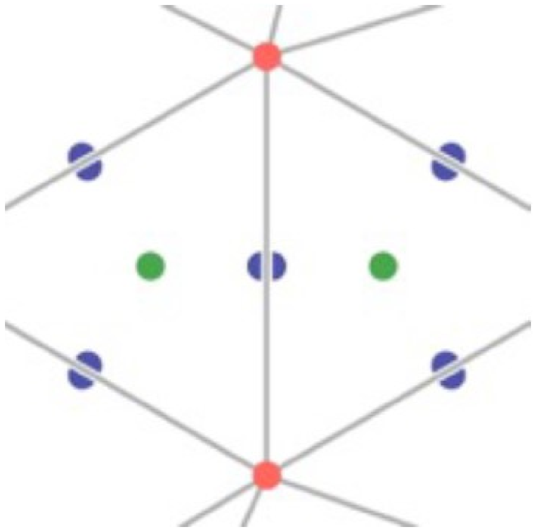
- Adjacency encoded in edges
  - All adjacencies in O(1) time
  - Little extra storage (fixed records)
  - Arbitrary polygons
  - **Assumes 2-Manifold surfaces**





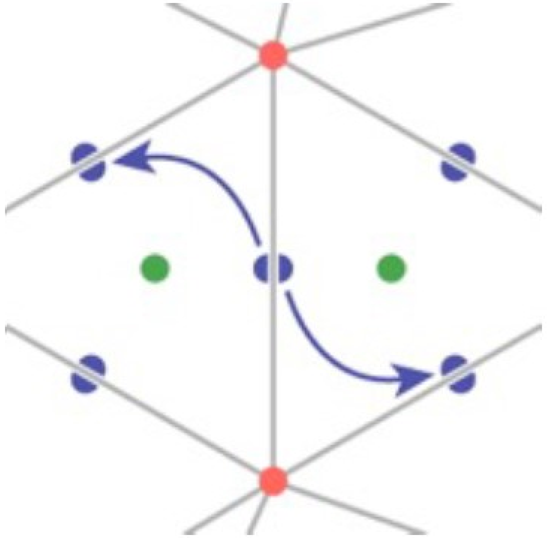
Half Edge

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



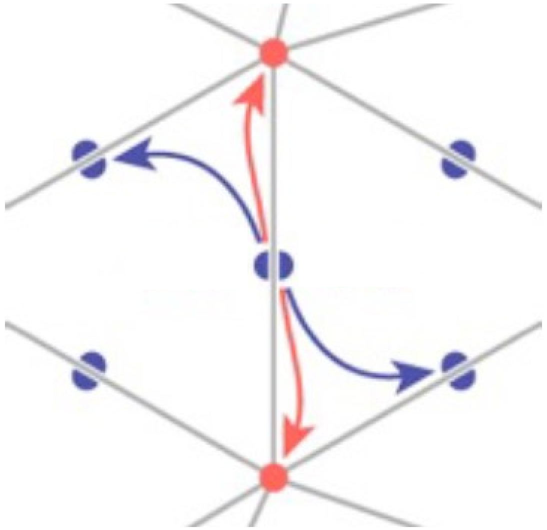
Half Edge

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



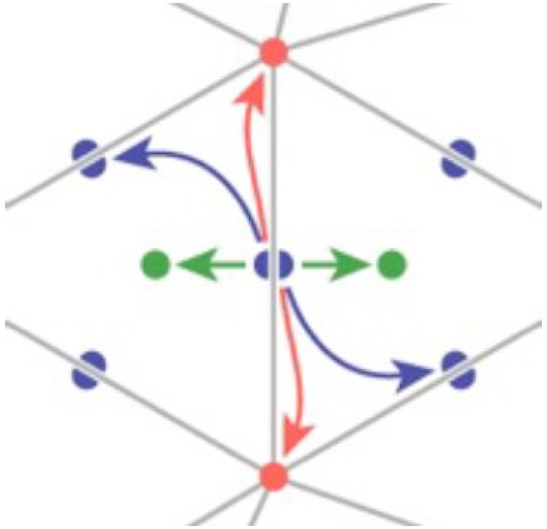
Half Edge

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



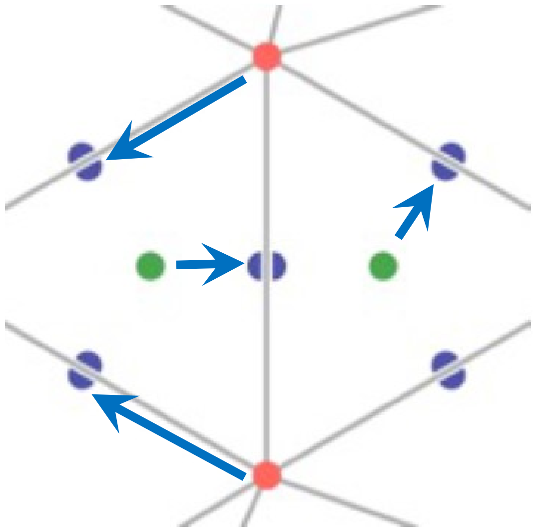
Half Edge

- 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?

Outline

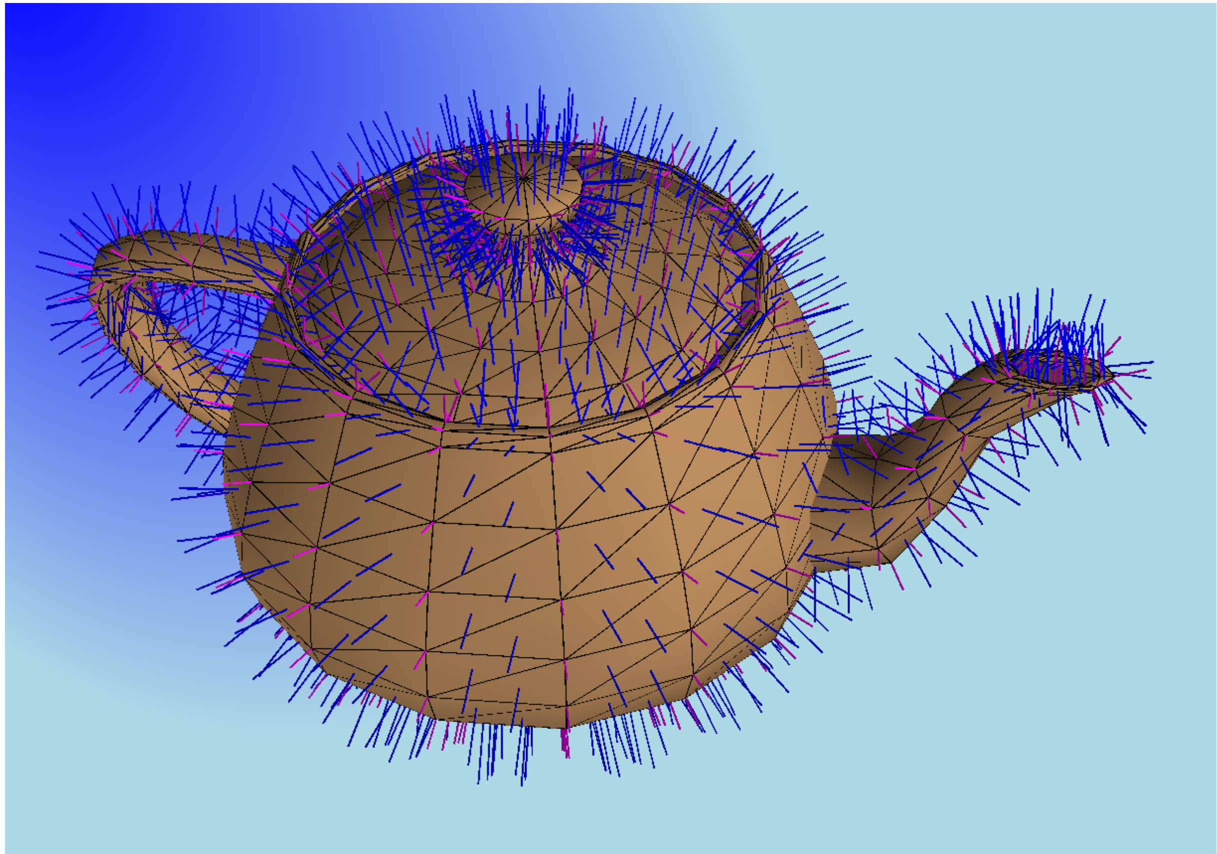
- Acquisition
- Representation
- Processing ←

Polygonal Mesh Processing

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

Polygonal Mesh Processing

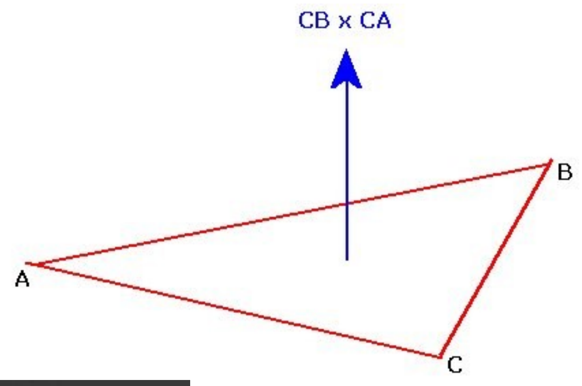
- Analysis
  - ➢ Normals
  - Curvature
- Warps
  - Rotate
  - Deform
- Filters
  - Smooth
  - Sharpen
  - Truncate
  - Bevel

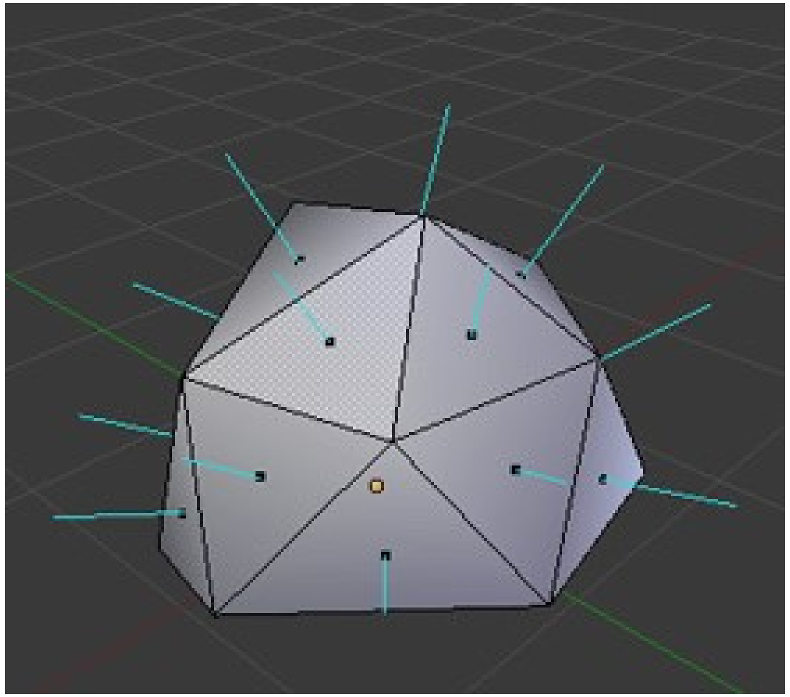


Polygonal Mesh Processing

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

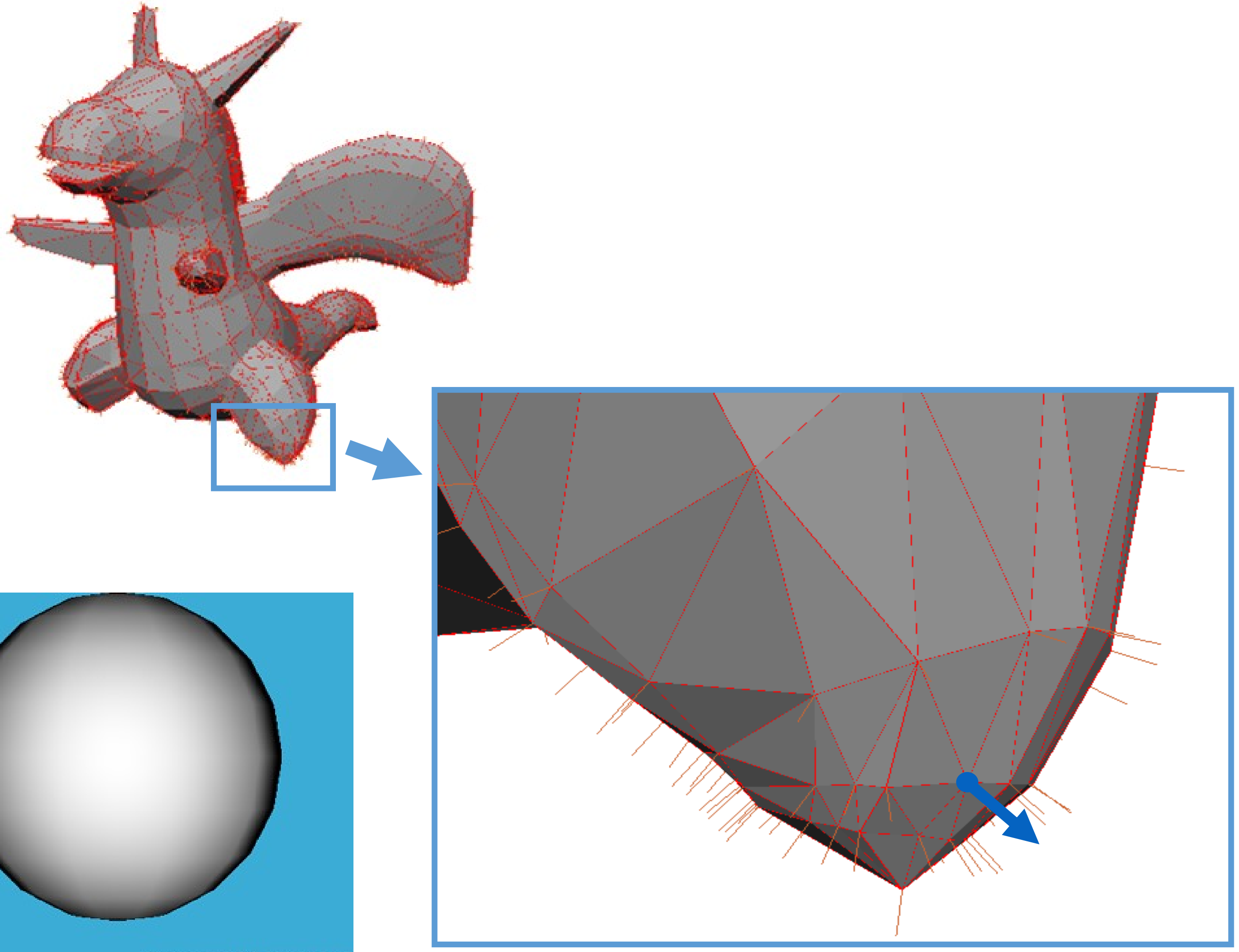
Face normals:

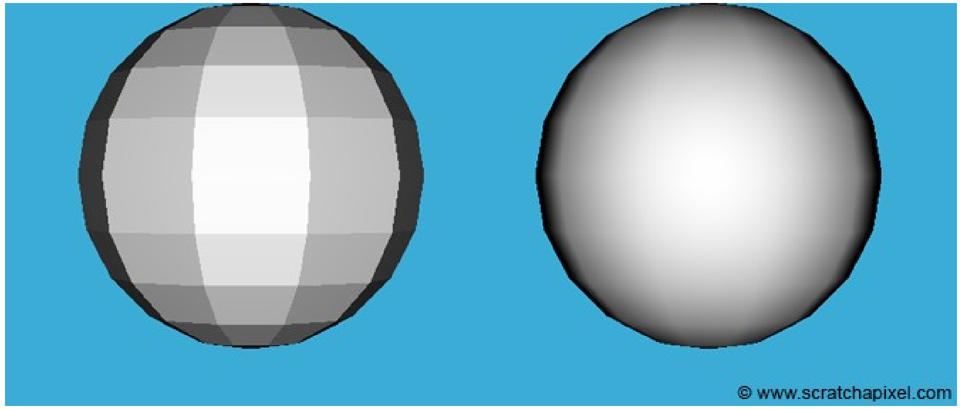




Polygonal Mesh Processing

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

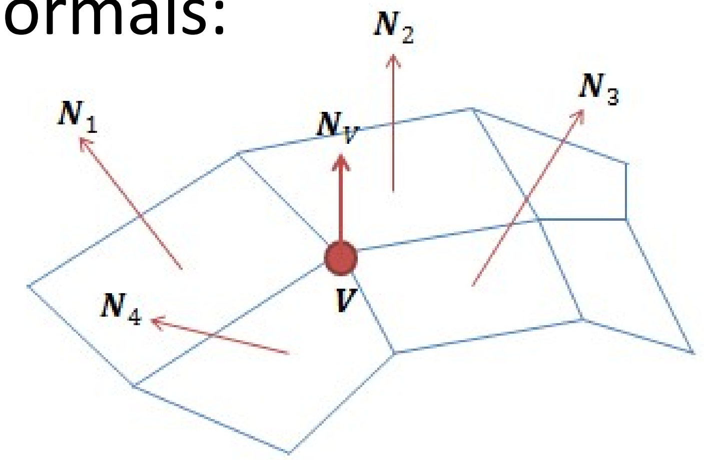




Polygonal Mesh Processing

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

Vertex normals:



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

for each face
-calculate face normal
-add normal to each connected vertices normal

for each face normal
-normalize

for each vertex normal
-normalize

Polygonal Mesh Processing

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

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

Lucas Mayer, COS 426, 2014

Polygonal Mesh Processing

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

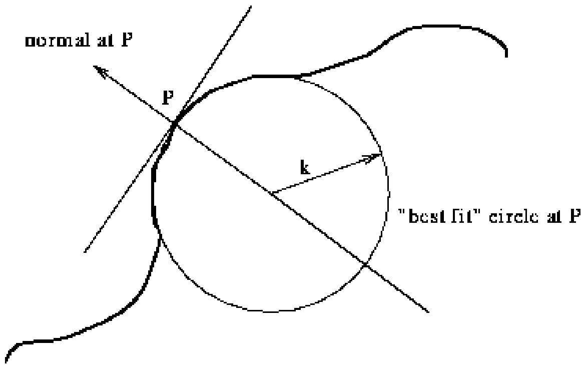
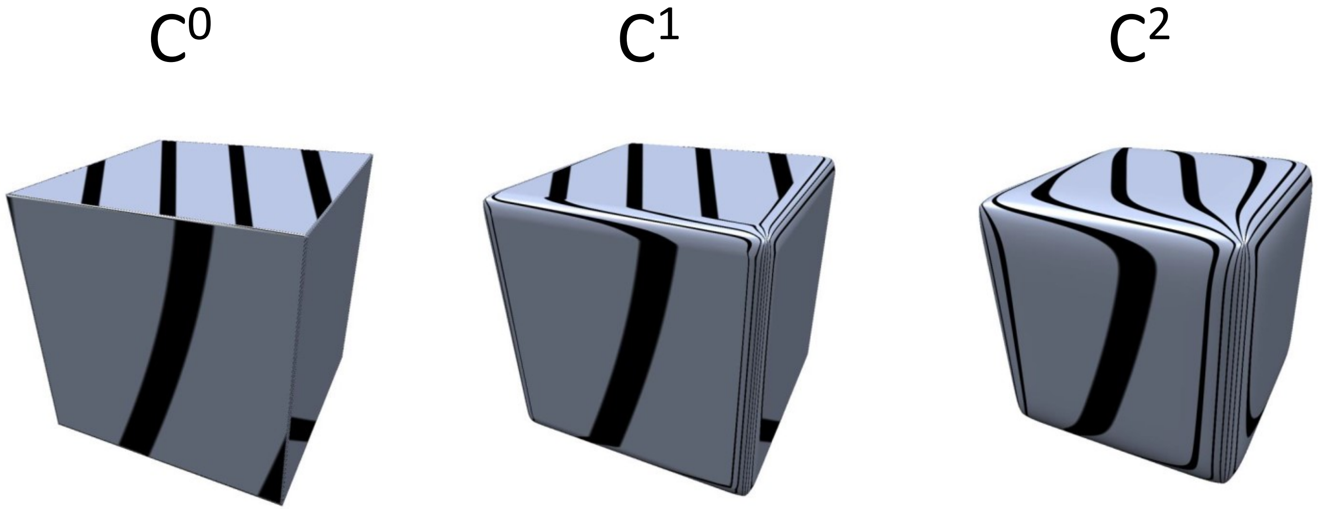


Figure 32: curvature of curve at P is $1/k$

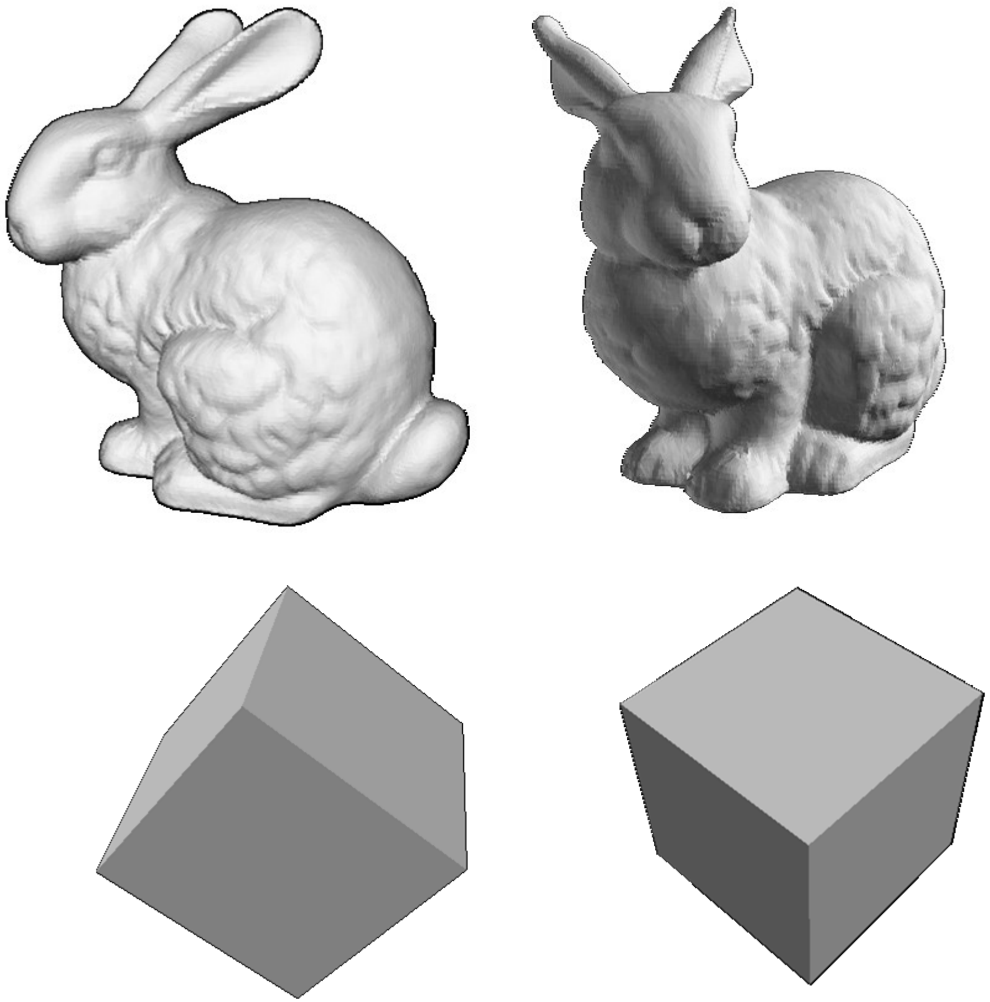
Polygonal Mesh Processing

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



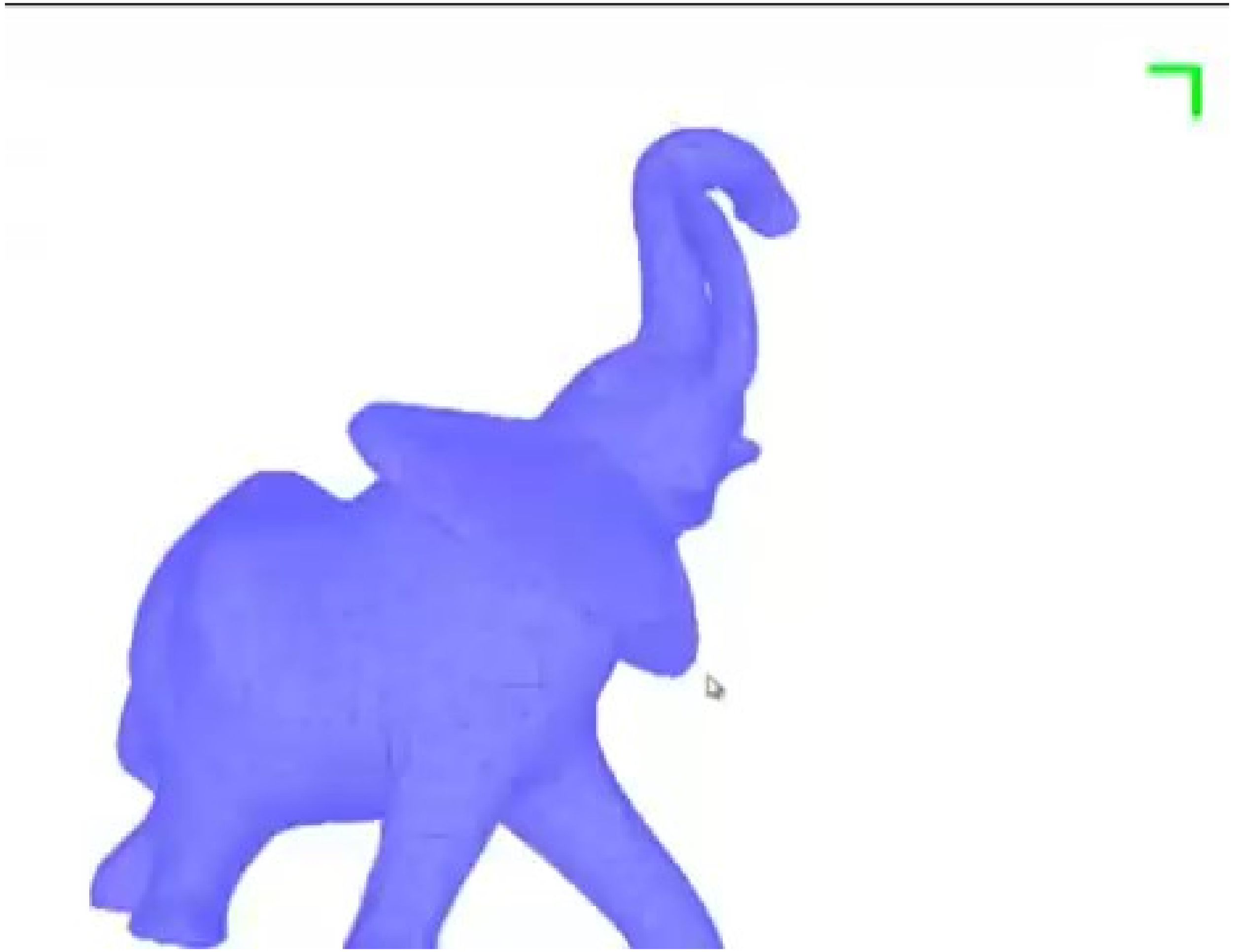
Polygonal Mesh Processing

- 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



Sorkine

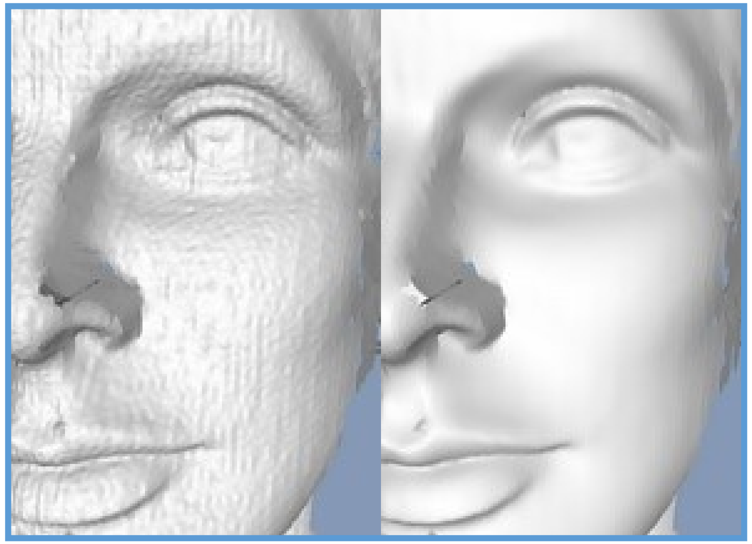
Polygonal Mesh Processing

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

Brendan Chou, COS 426, 2014

Polygonal Mesh Processing

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



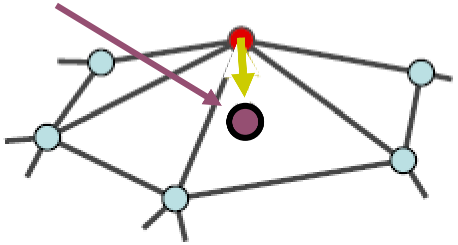
Thouis "Ray" Jones

How?

The Laplacian Operator

- Mesh formulation: Average of Neighboring Vertices

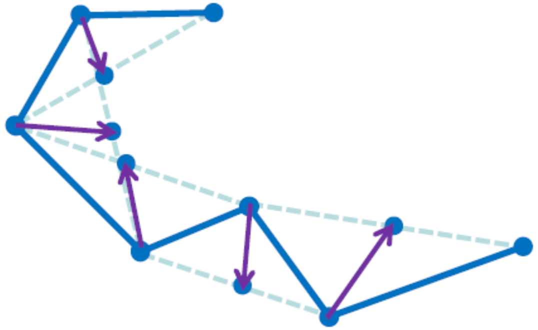
$$p_i = \frac{\Sigma_{j \in 1ring_i} p_j}{\#1ring_i}$$



Olga Sorkine

- Curve (Γ) formulation:

$$p_i = \frac{p_{i+1} + p_{i-1}}{2}$$



$$0 = \frac{(p_{i+1} - p_i) - (p_i - p_{i-1})}{2} \Rightarrow 0 = \Gamma''(p_i)$$

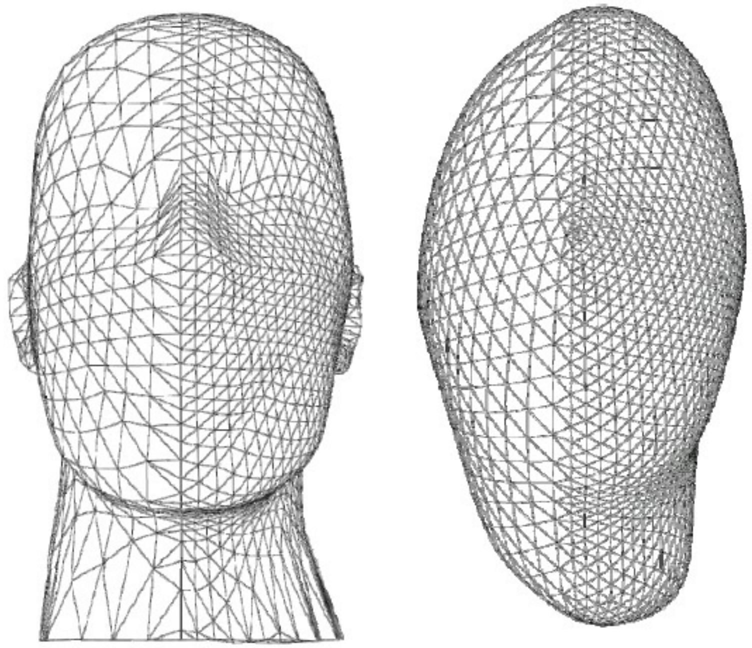
$\Gamma'(p_{i+1})$ $\Gamma'(p_i)$

The Laplacian Operator

- Lo and behold: The Laplacian operator Δ

$$L(p_i) = \Delta(p_i) = \frac{\Sigma_{j \in 1_{ring_i}} p_j - p_i}{\# 1_{ring_i}}$$

- However, Meshes are irregular



The Laplacian Operator

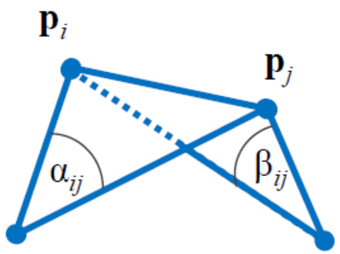
- Lo and behold: The Laplacian operator Δ

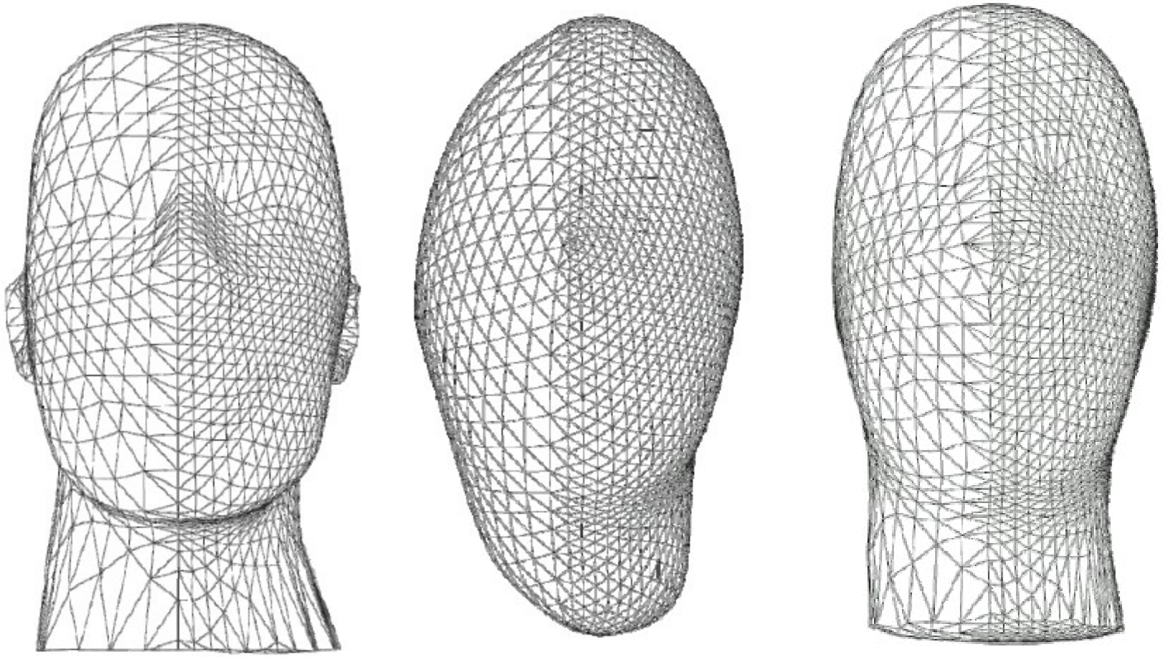
$$L(p_i) = \Delta(p_i) = \frac{\sum_{j \in 1ring_i} p_j - p_i}{\#1ring_i}$$

- However, Meshes are irregular
  - Cotangent weights:

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

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

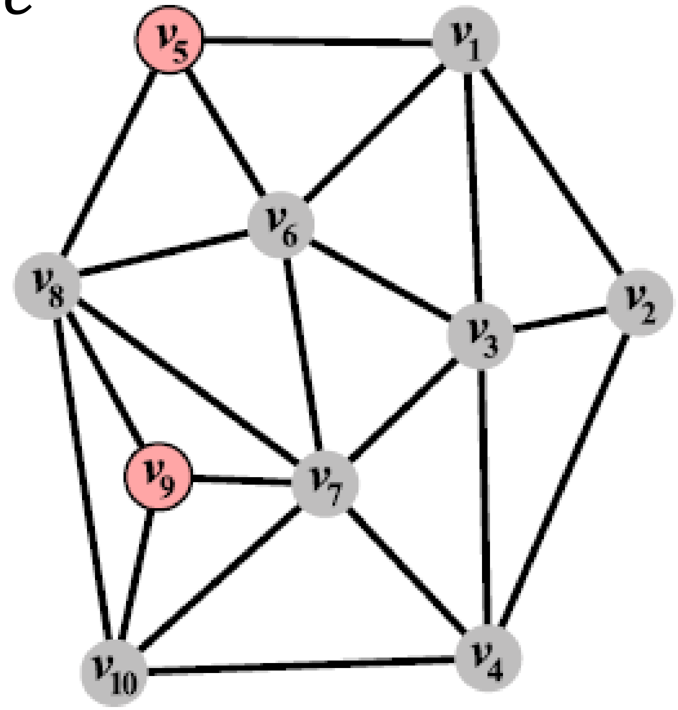




The Laplacian Operator

- In matricial form:

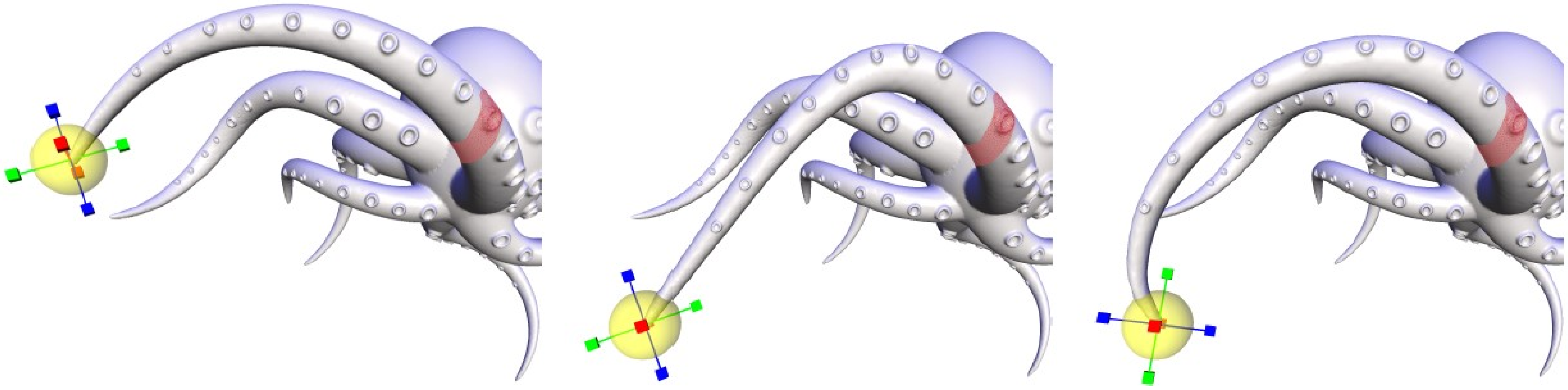
$$L_{ij} = \begin{cases} -w_{ij} & i \neq j \\ \Sigma_{j \in 1_{ring_i}} w_{ij} & i = j \\ 0 & else \end{cases}$$



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			4	−1	−1		
		−1	−1		−1	6	−1	−1	−1
				−1	−1	−1	6	−1	−1
						−1	−1	3	−1
			−1			−1	−1	−1	4

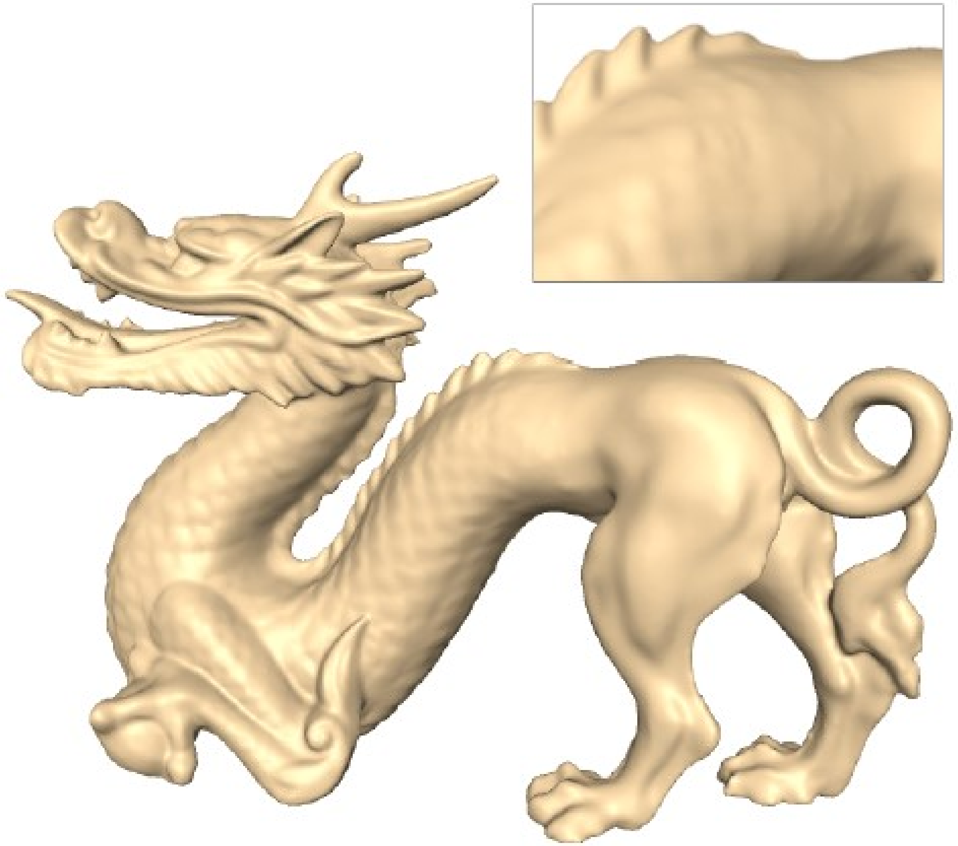
The Laplacian Operator

- In matricial form
- Applicable to:
  - Deformation, by adding constraints



The Laplacian Operator

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

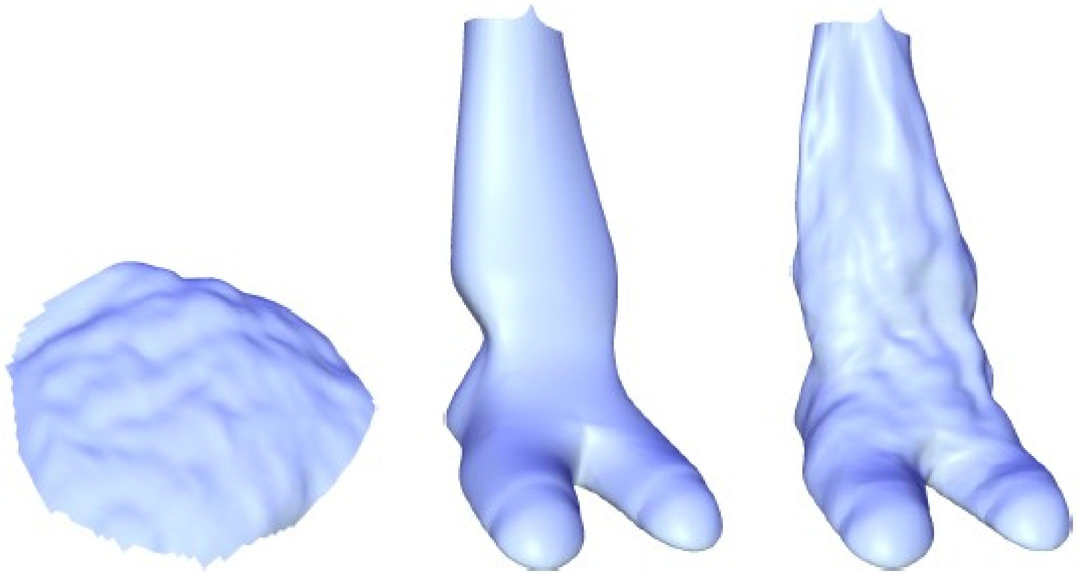


The Laplacian Operator

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

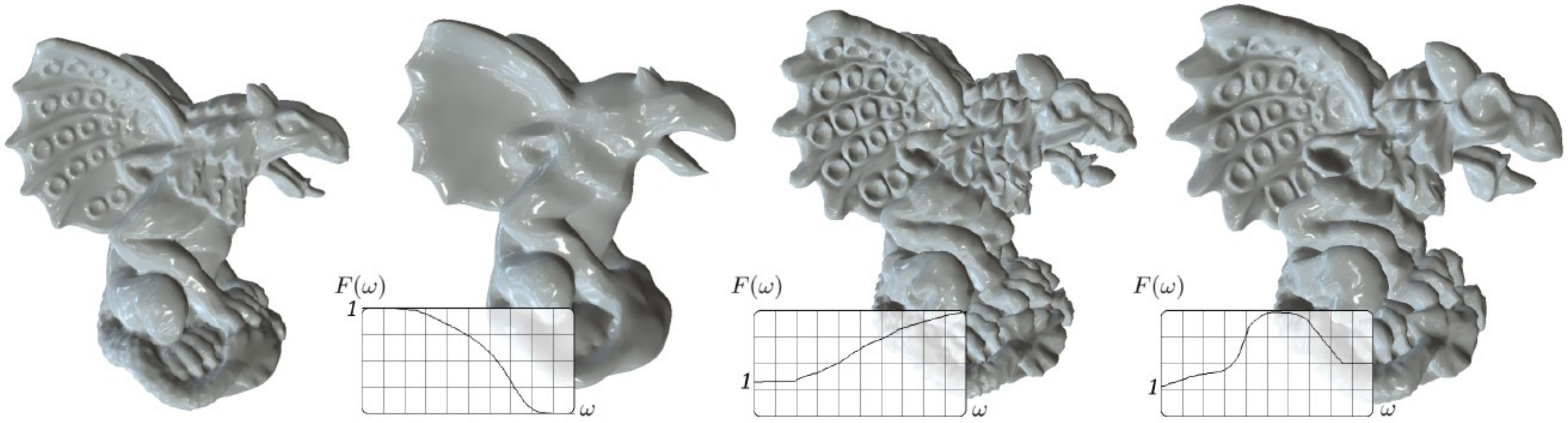
The Laplacian Operator

- In matricial form
- 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)



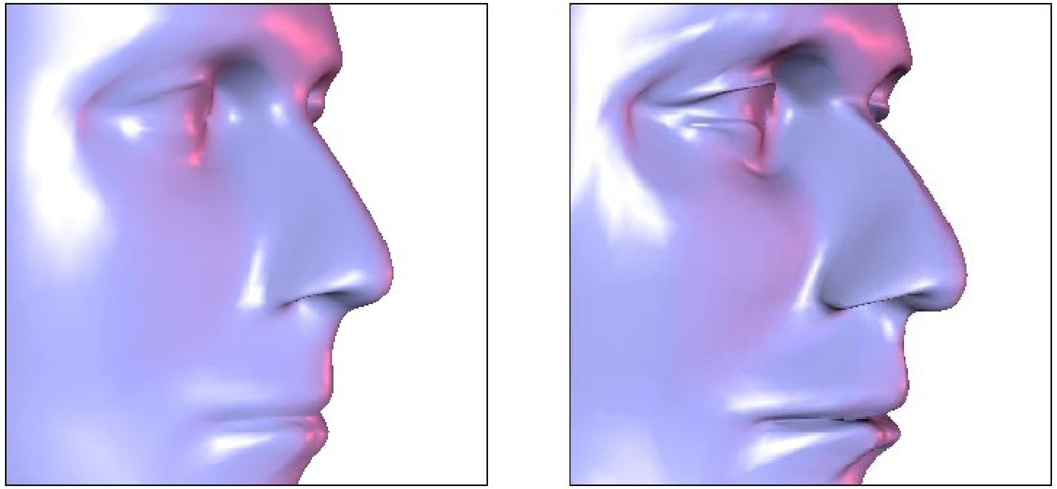
The Laplacian Operator

- In matricial form
- 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

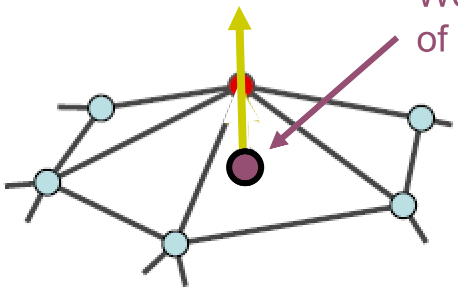


Polygonal Mesh Processing

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



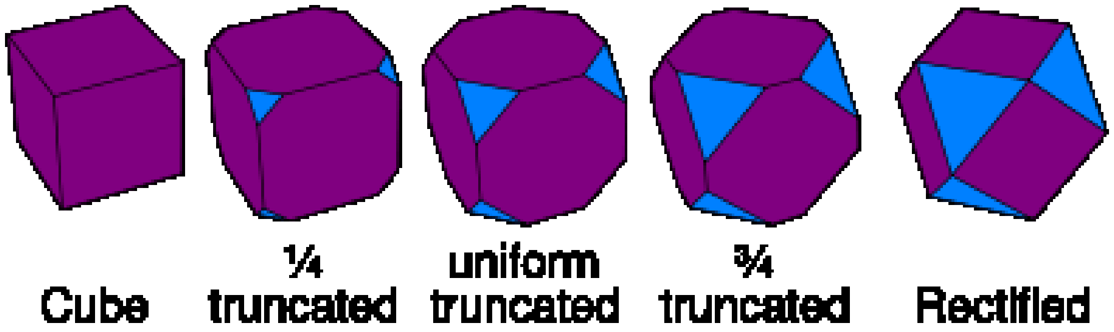
Desbrun

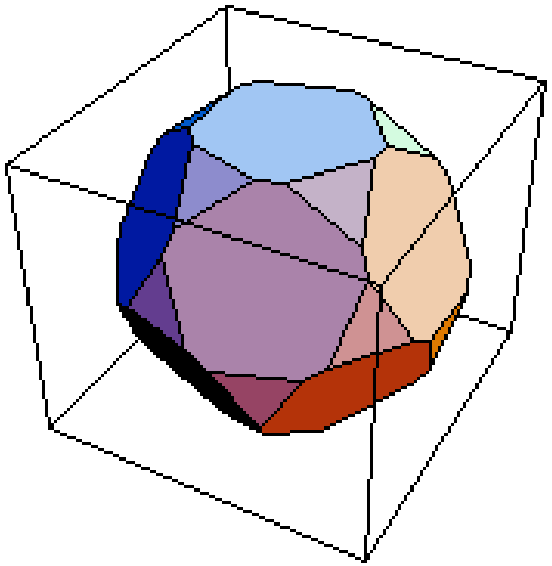


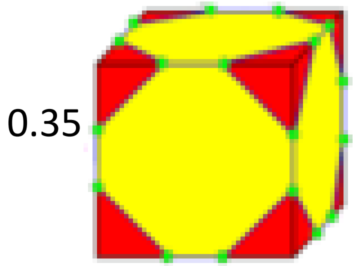
Olga Sorkine

Polygonal Mesh Processing

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







Conway

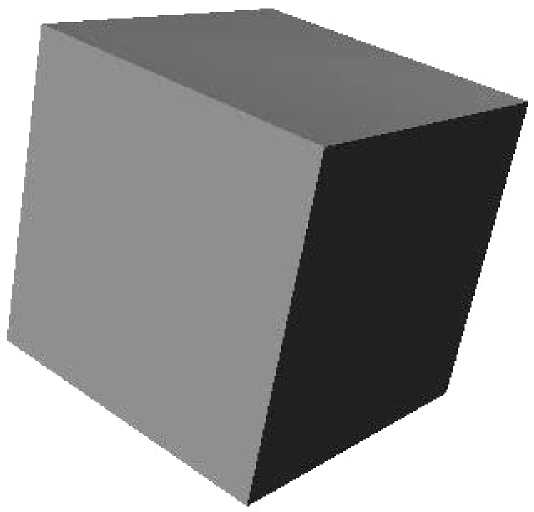
Polygonal Mesh Processing

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

Riley Thomasson, COS 426, 2014

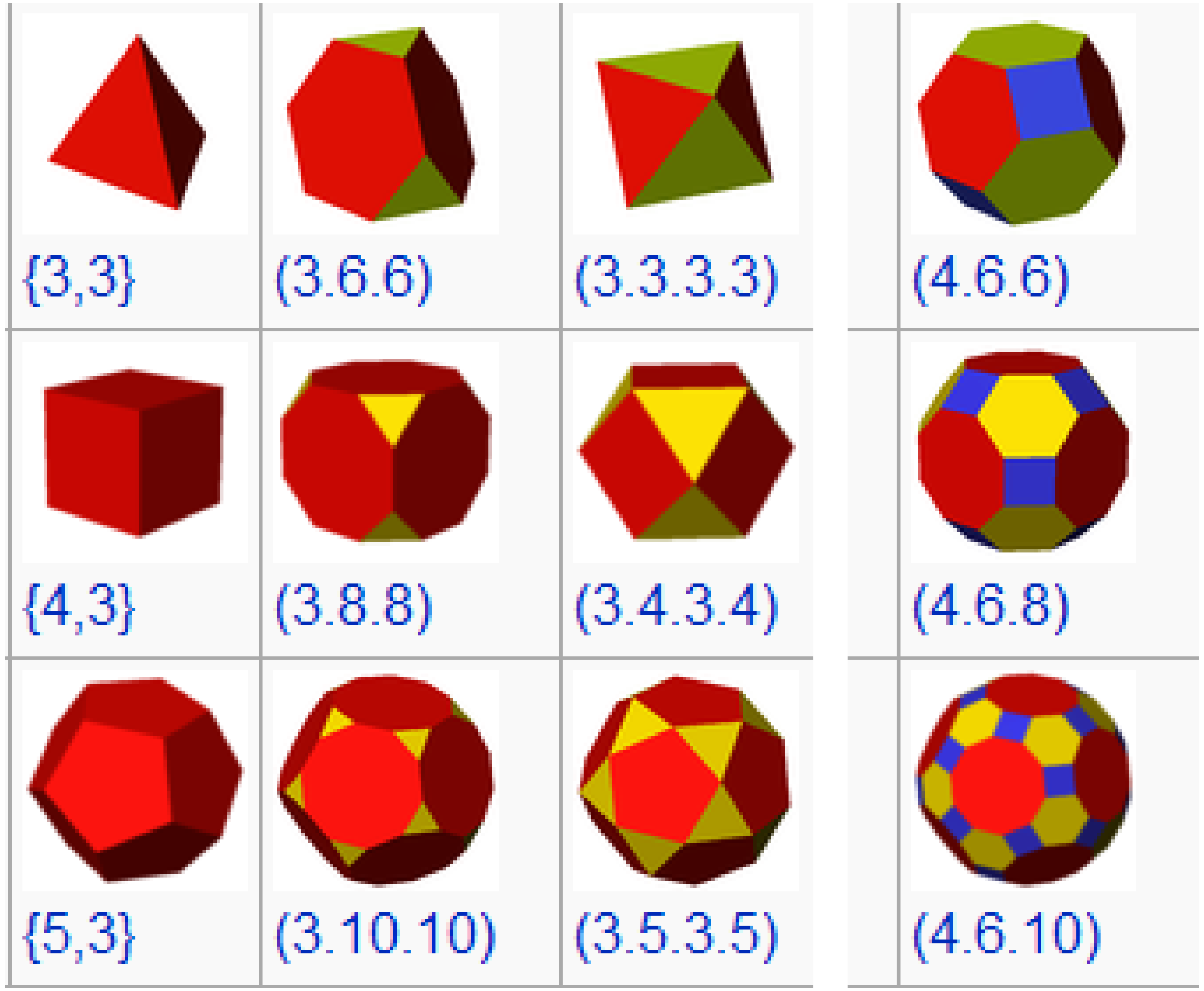
Polygonal Mesh Processing

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



Polygonal Mesh Processing

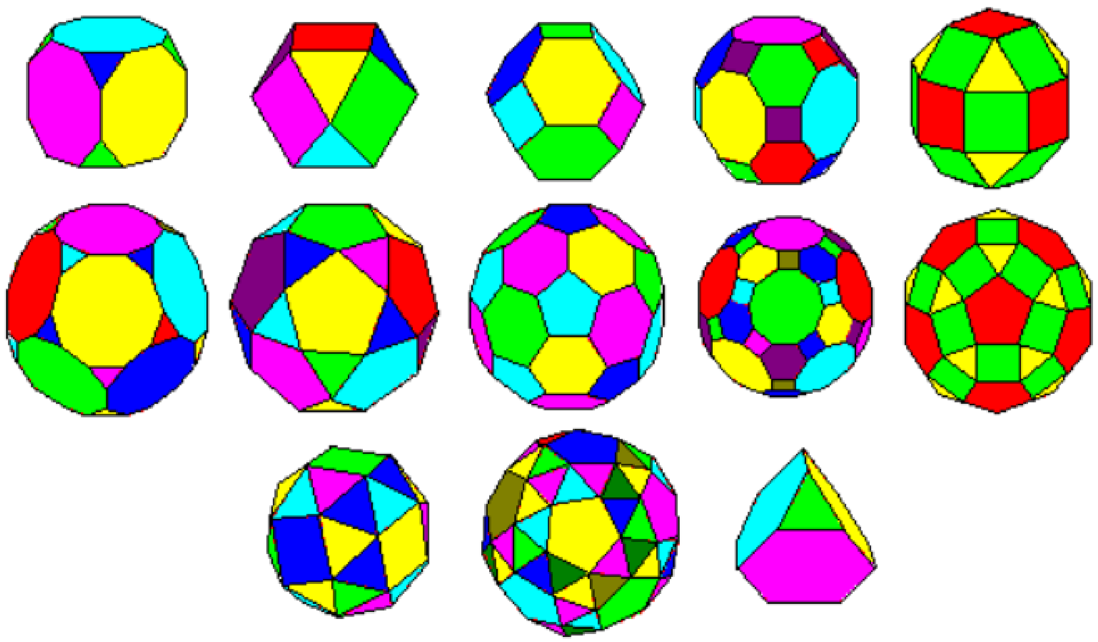
- Analysis
  - Normals
  - Curvature
- Warps
  - Rotate
  - Deform
- Filters
  - Smooth
  - Sharpen
  - ➢ Truncate
  - Bevel



Wikipedia

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

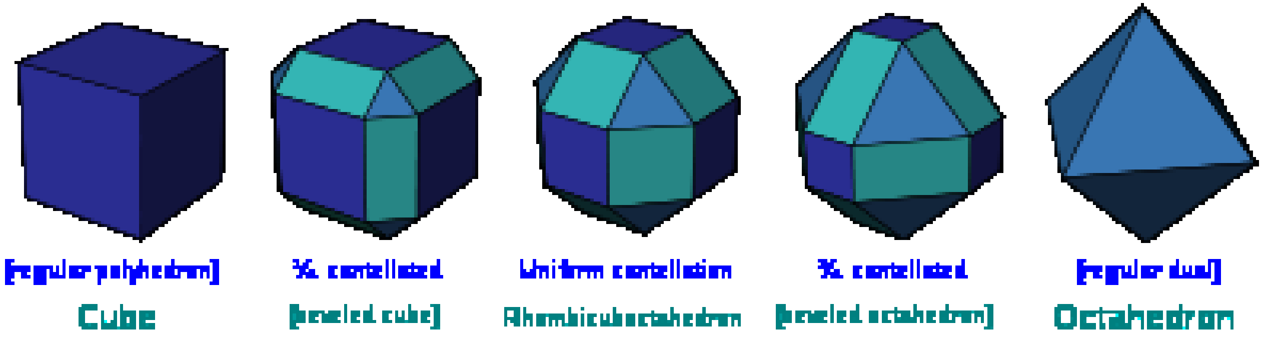
- Analysis
  - Normals
  - Curvature
- Warps
  - Rotate
  - Deform
- Filters
  - Smooth
  - Sharpen
  - ➢ Truncate
  - Bevel

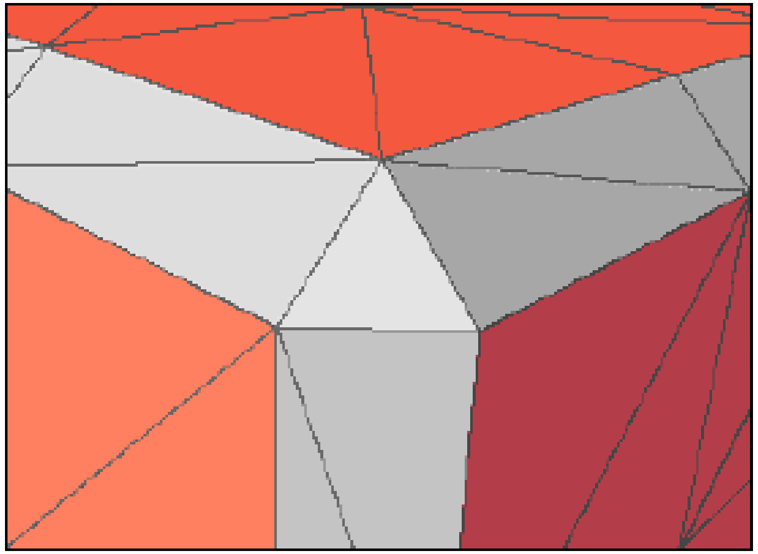
Carlo Sèquin

Polygonal Mesh Processing

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

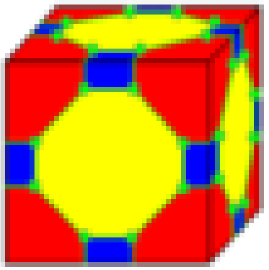
Wikipedia





Jarek Rossignac

0.40



Conway

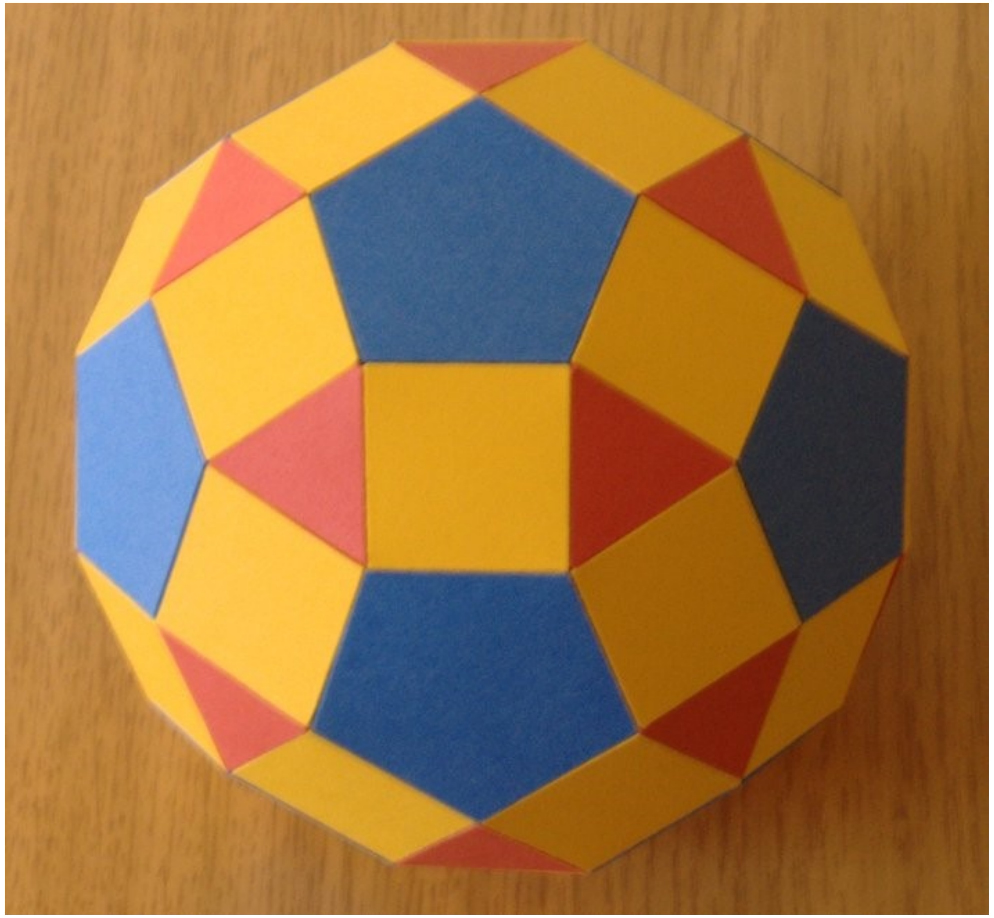
Polygonal Mesh Processing

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

www.srcf.ucam.org

Polygonal Mesh Processing

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



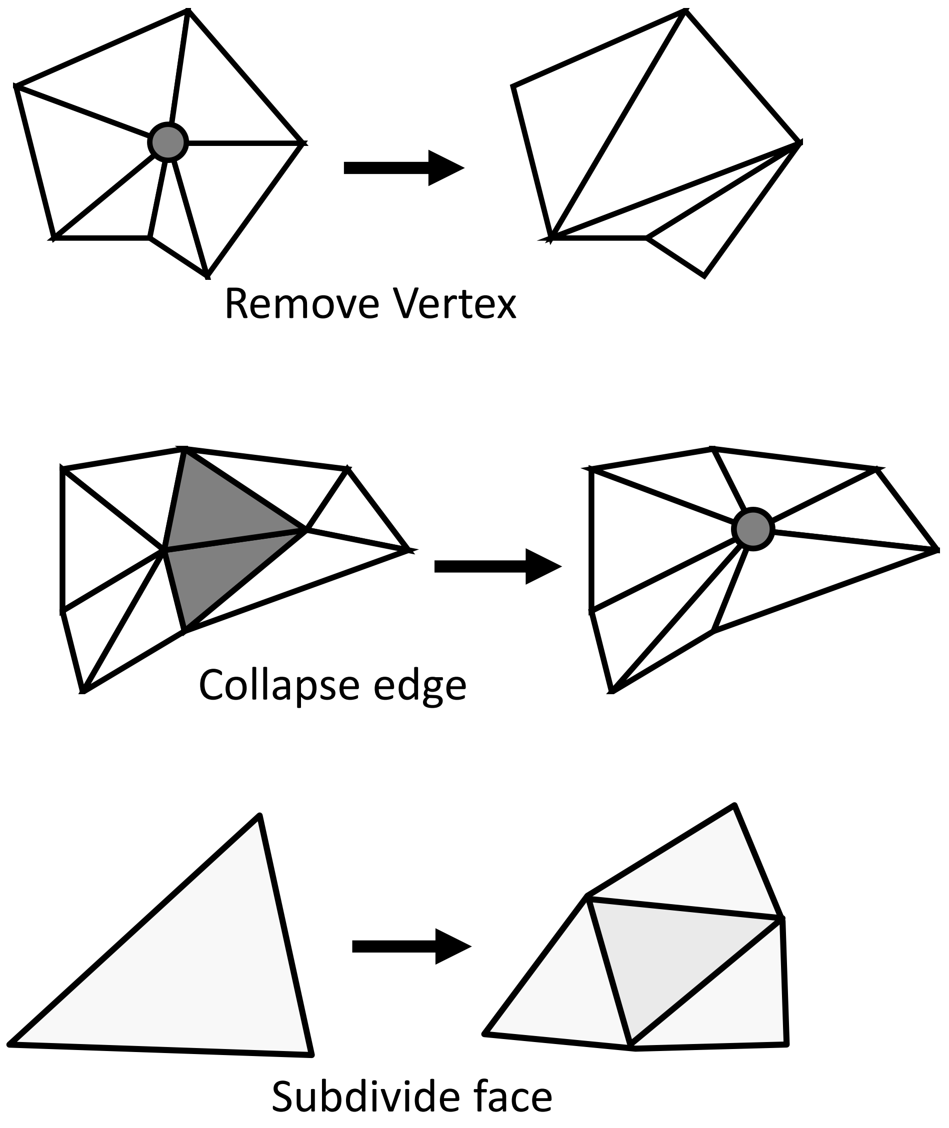
www.srcf.ucam.org

Polygonal Mesh Processing

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

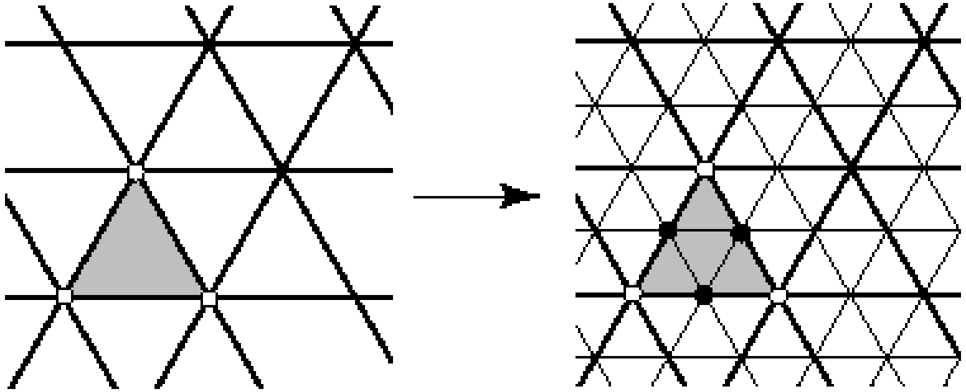
Polygonal Mesh Processing

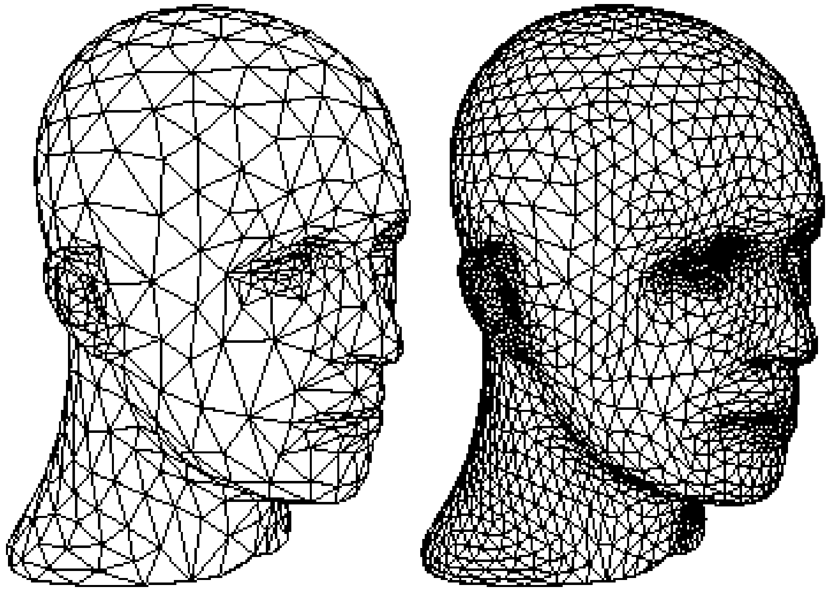
- Remeshing
  - Subdivide
  - Resample
  - Simplify
- Topological fixup
  - Fill holes
  - Fix self-intersections
- Boolean operations
  - Crop
  - Subtract



Polygonal Mesh Processing

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





Zorin & Schroeder

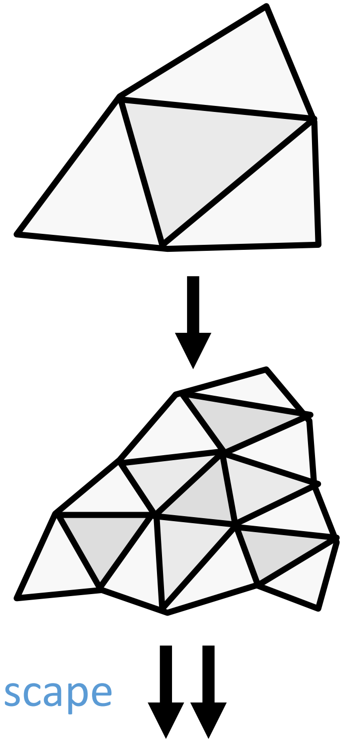
Polygonal Mesh Processing

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

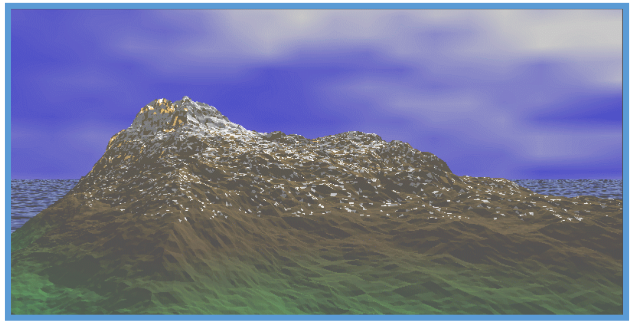
Matt Matl, COS 426, 2014

Polygonal Mesh Processing

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



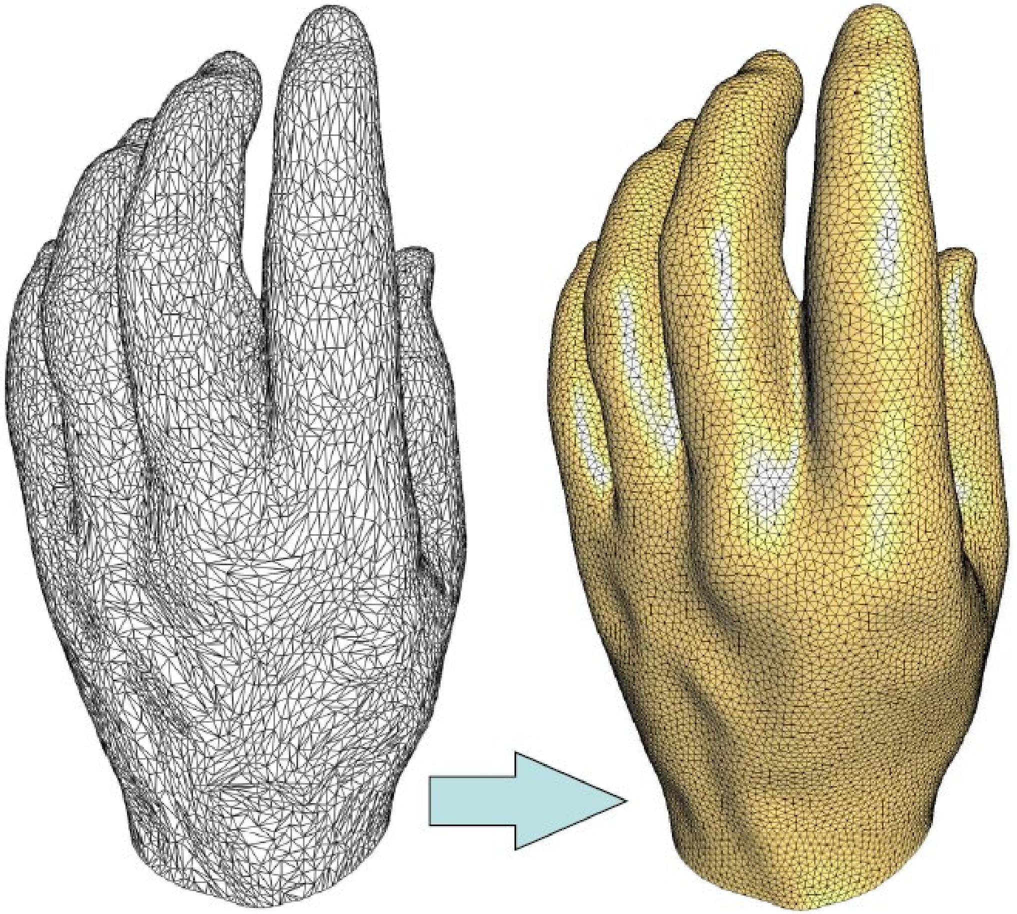
Fractal Landscape



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

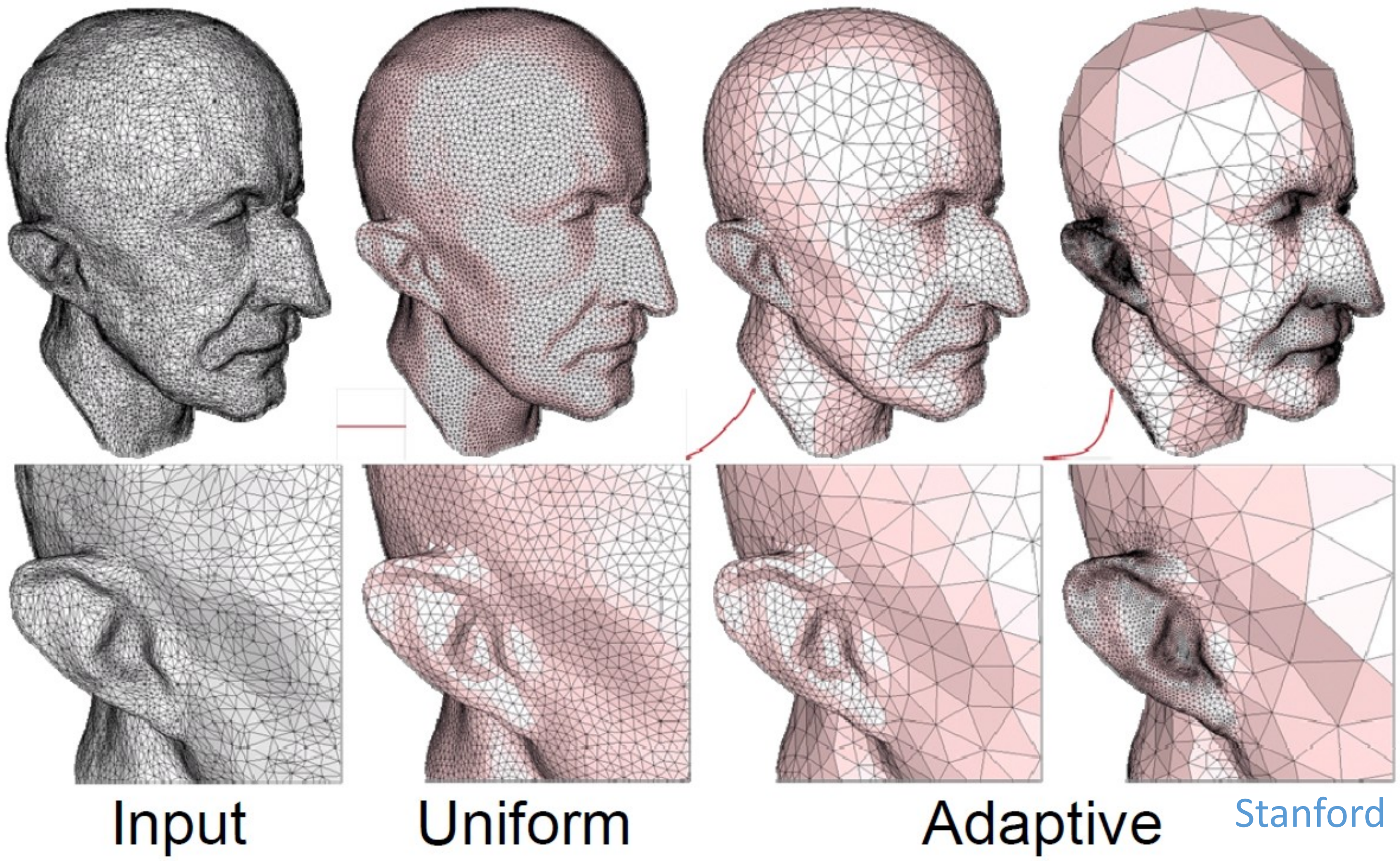


Stanford

- more uniform distribution
- triangles with nicer aspect

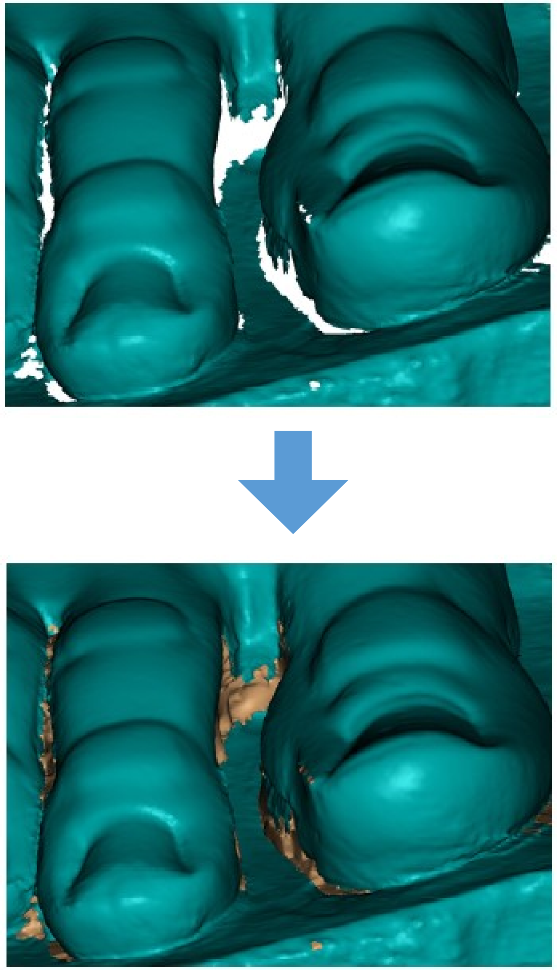
Polygonal Mesh Processing

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



Polygonal Mesh Processing

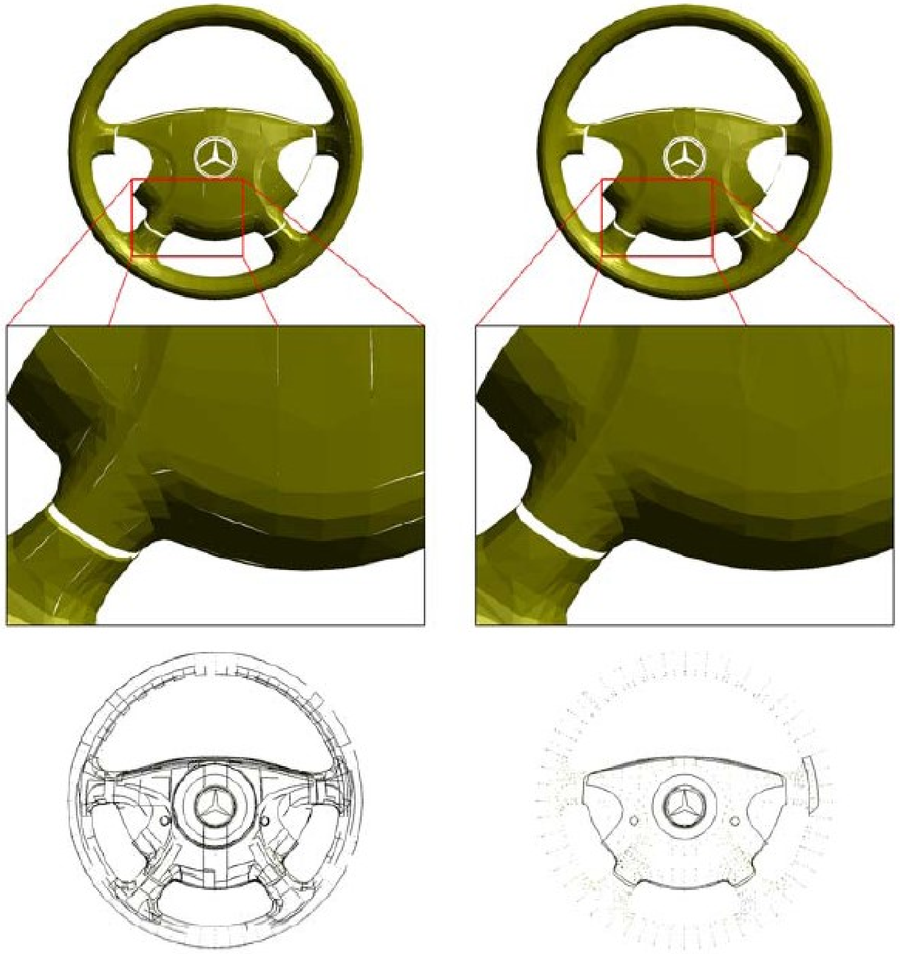
- Remeshing
  - Subdivide
  - Resample
  - Simplify
- Topological fixup
  - ➢Fill holes
  - Fix self-intersections
- Boolean operations
  - Crop
  - Subtract



Podolak

Polygonal Mesh Processing

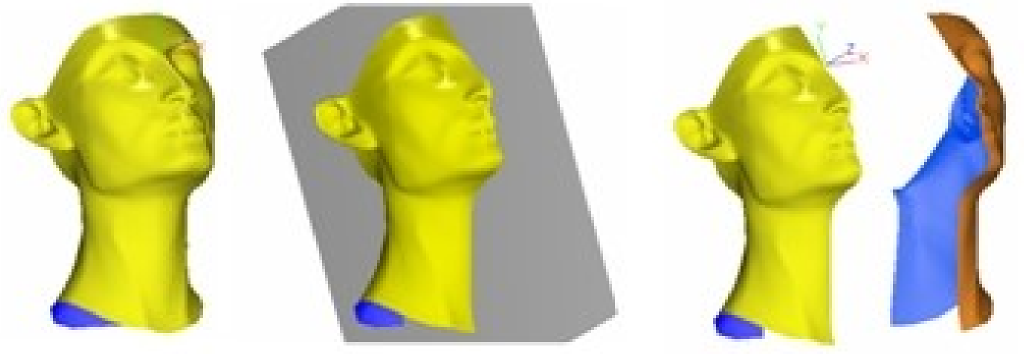
- Remeshing
  - Subdivide
  - Resample
  - Simplify
- Topological fixup
  - Fill holes
  - ➢Fix self-intersections
- Boolean operations
  - Crop
  - Subtract



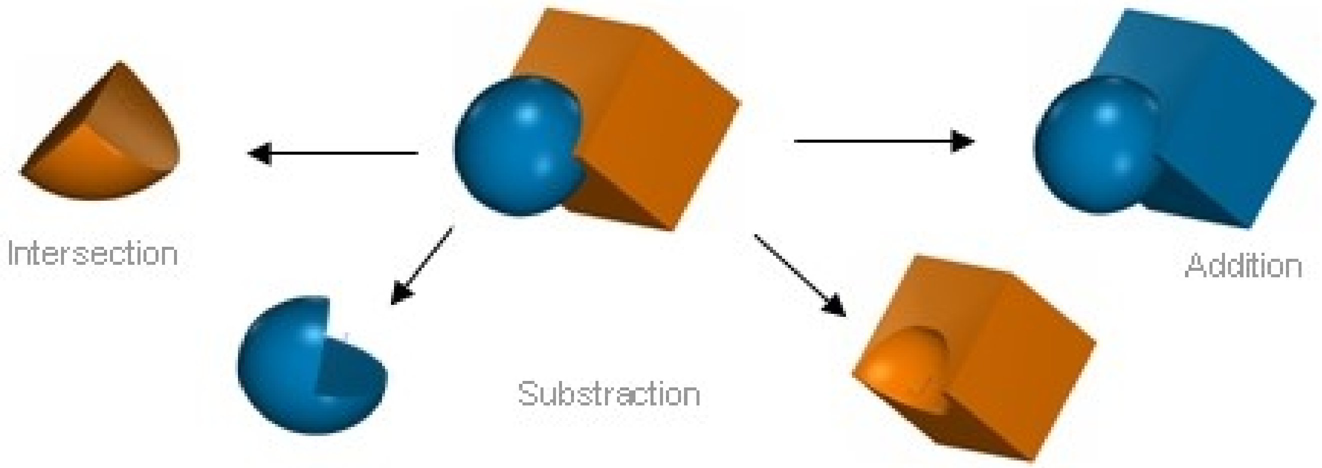
Borodin

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.



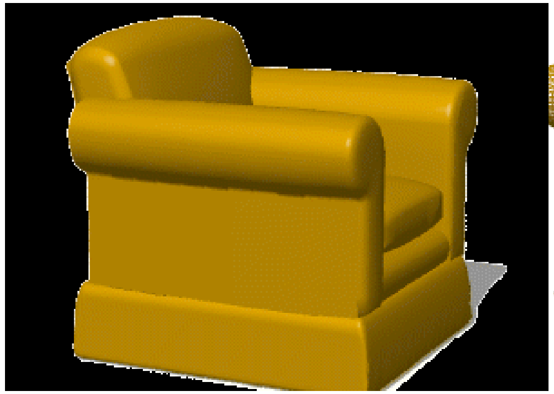
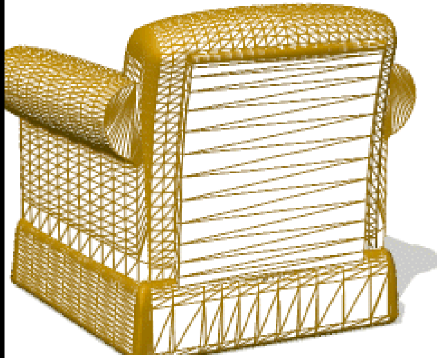
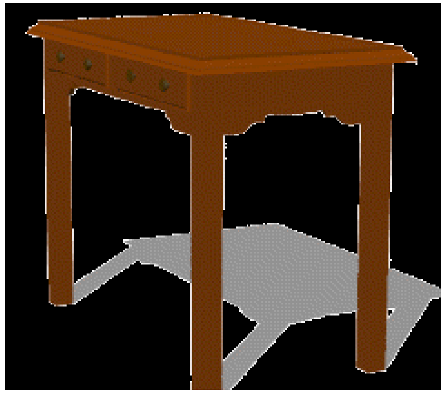
Several Boolean operations with 3DReshaper®

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

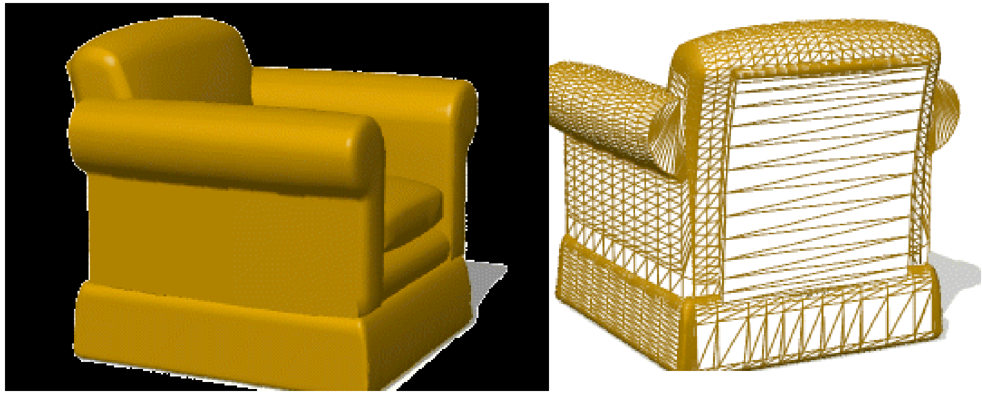
- Properties
  - ? Efficient display
  - ? Easy acquisition
  - ? Accurate
  - ? Concise
  - ? Intuitive editing
  - ? Efficient editing
  - ? Efficient intersections
  - ? Guaranteed validity
  - ? Guaranteed smoothness
  - ? etc.

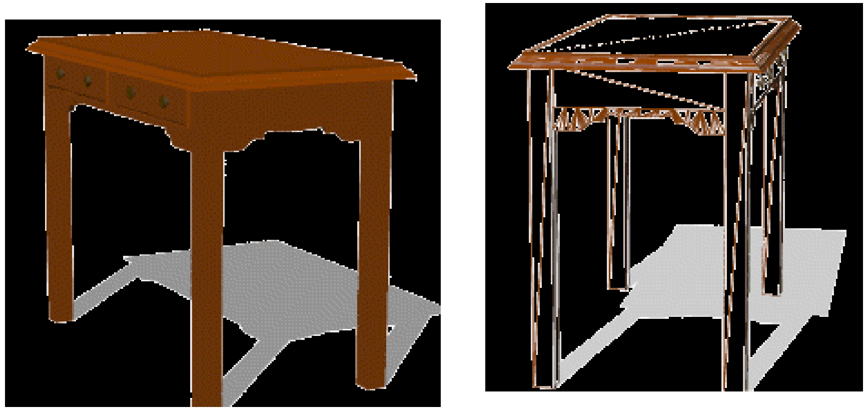





Viewpoint

3D Polygonal Meshes

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





Viewpoint