252-0538-00L, Spring 2017

Shape Modeling and Geometry Processing

Geometry Acquisition
Meshes

February 21, 2018

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Geometry Acquisition Pipeline

Scanning: results in range images ⇨ **Registration:** bring all range images to one coordinate system ⇨ **Reconstruction:** Integration of scans into a single mesh ⇨ **Postprocess:**
- Topological and geometric filtering
- Remeshing
- Compression

Geometry Acquisition Pipeline

Scanning: results in range images

Registration: bring all range images to one coordinate system

Reconstruction: Integration of scans into a single mesh

Postprocess:
- Topological and geometric filtering
- Remeshing
- Compression

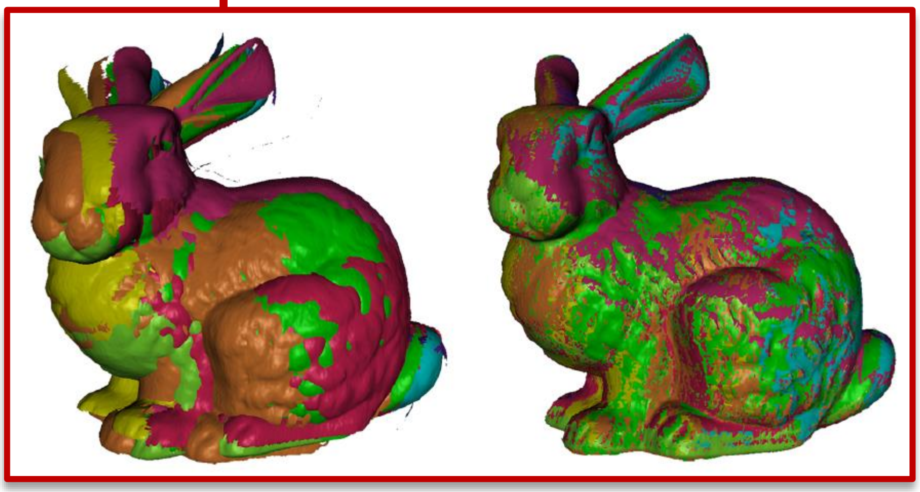
Geometry Acquisition Pipeline

Scanning: results in range images

Registration: bring all range images to one coordinate system

Reconstruction: Integration of scans into a single mesh

Postprocess:
- Topological and geometric filtering
- Remeshing
- Compression



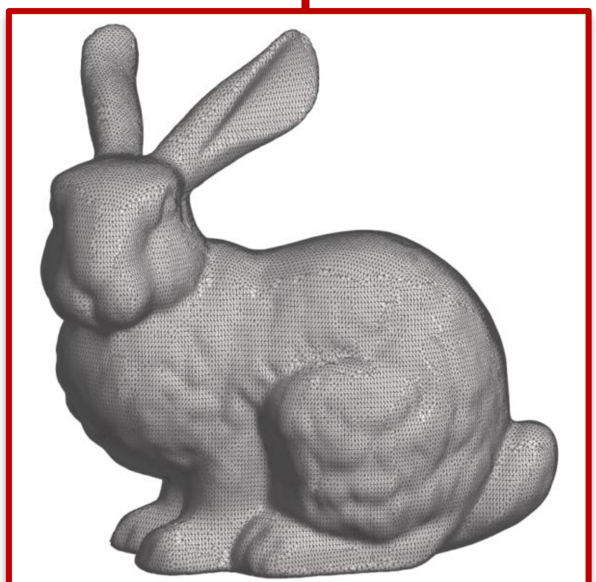
Geometry Acquisition Pipeline

Scanning: results in range images

Registration: bring all range images to one coordinate system

Reconstruction: Integration of scans into a single mesh

Postprocess:
- Topological and geometric filtering
- Remeshing
- Compression



Geometry Acquisition Pipeline

Scanning: results in range images

⇨

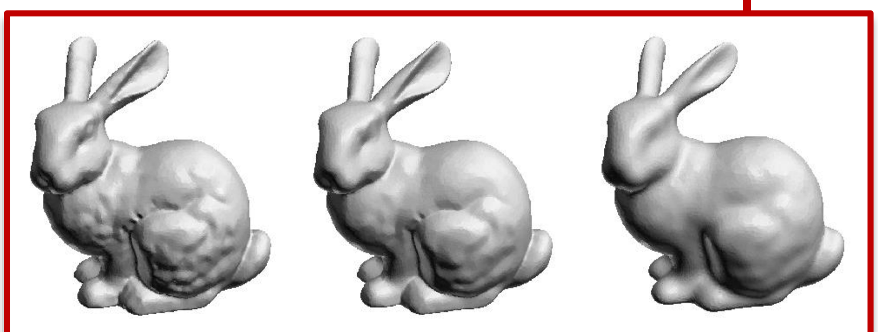
Registration: bring all range images to one coordinate system

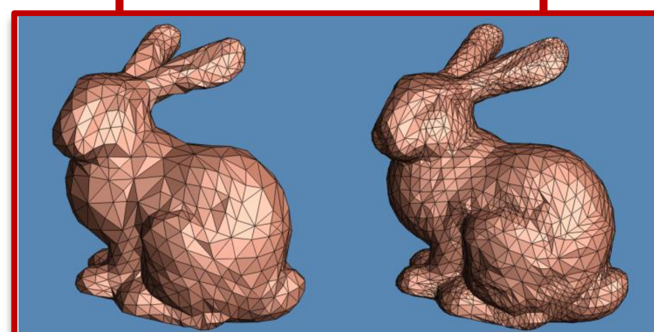
⇨

Reconstruction: Integration of scans into a single mesh

⇨

Postprocess:
- Topological and geometric filtering
- Remeshing
- Compression





Geometry Acquisition Pipeline

Scanning: results in range images

⇨

Registration: bring all range images to one coordinate system

⇨

Reconstruction: Integration of scans into a single mesh

⇨

Postprocess:
- Topological and geometric filtering
- Remeshing
- Compression

Touch Probes

ETH zürich

Touch Probes

- Physical contact with the object
- Manual or computer-guided
- Advantages:
  - Can be very precise
  - Can scan **any** solid surface
- Disadvantages:
  - Slow, small scale
  - Limited for fragile objects

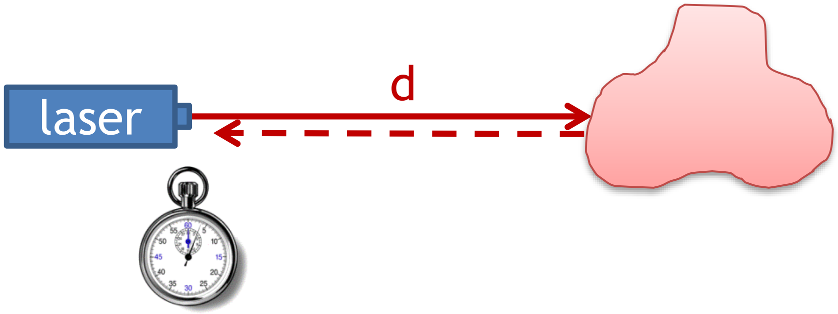
Optical Scanning

- Infer the geometry from light reflectance
- Advantages:
  - Less invasive than touch
  - Fast, large scale possible
- Disadvantages:
  - Difficulty with transparent and shiny objects

Optical scanning – active lighting

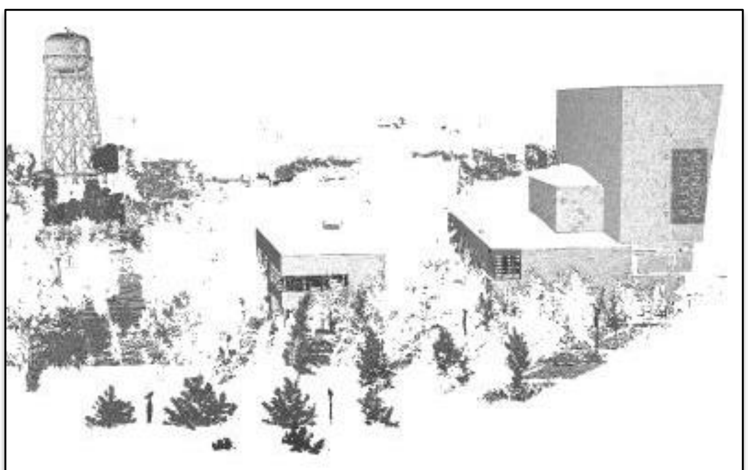
Time of flight laser

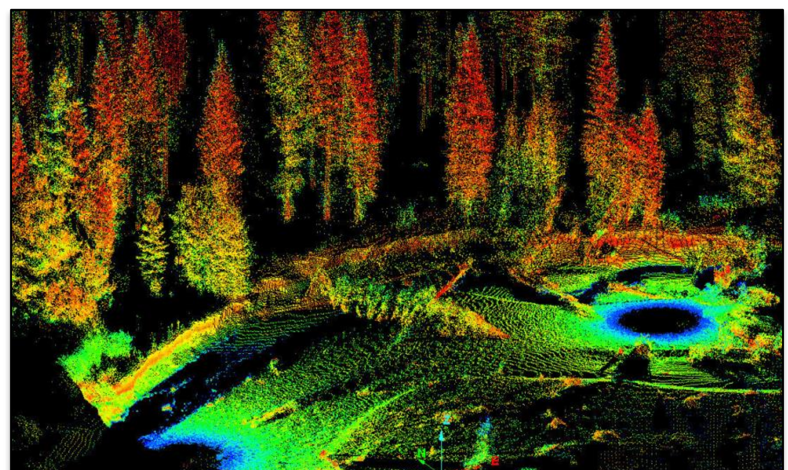
- A type of laser rangefinder (LIDAR)
- Measures the time it takes the laser beam to hit the object and come back



Optical scanning – active lighting

- Accommodates large range – up to several miles (suitable for buildings, rocks)
- Lower accuracy
  - objects move while scanning

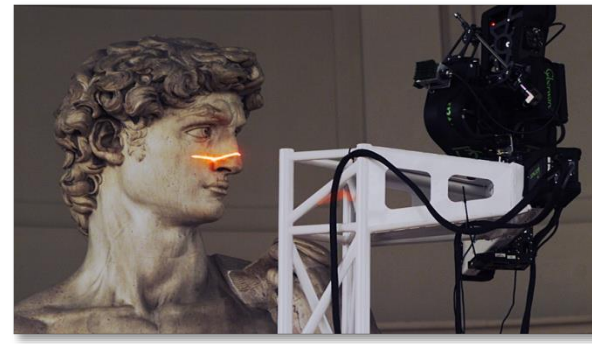


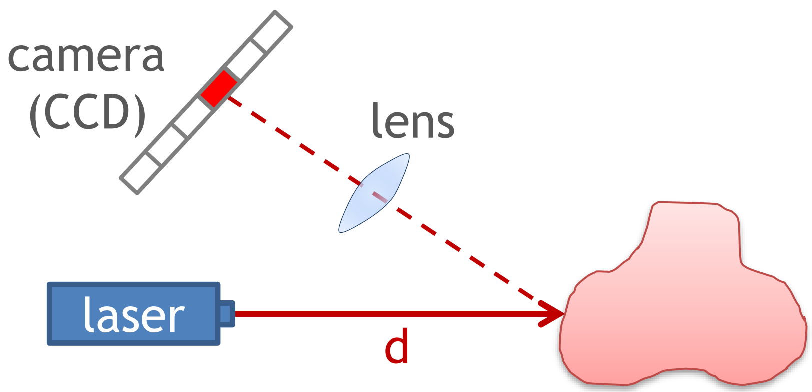


Optical scanning – active lighting

Triangulation laser

- Laser beam and camera
- Laser dot is photographed
- The location of the dot in the image allows triangulation: we get the distance to the object

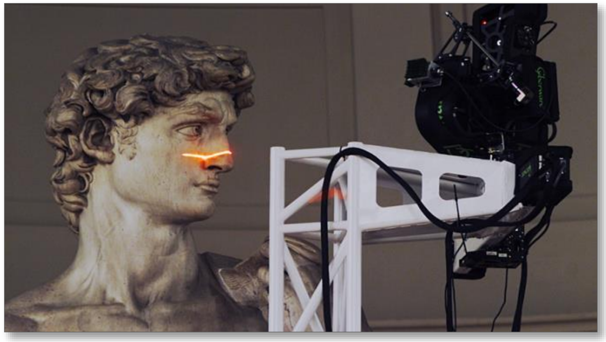


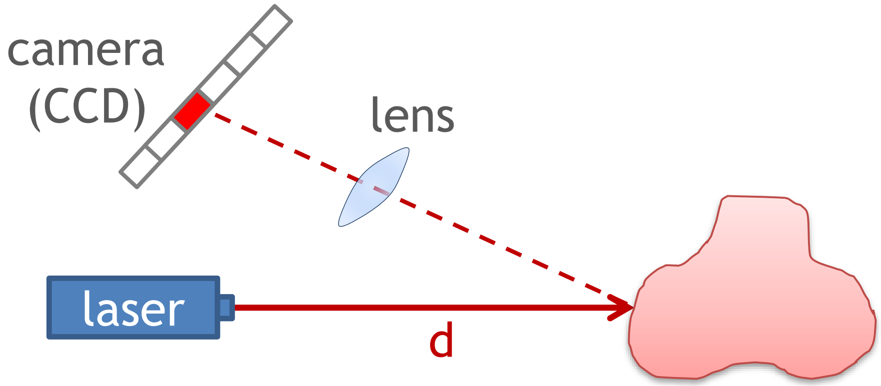


Optical scanning – active lighting

Triangulation laser

- Laser beam and camera
- Laser dot is photographed
- The location of the dot in the image allows triangulation: we get the distance to the object

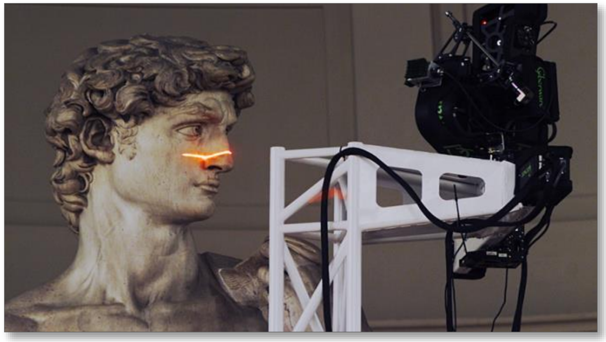


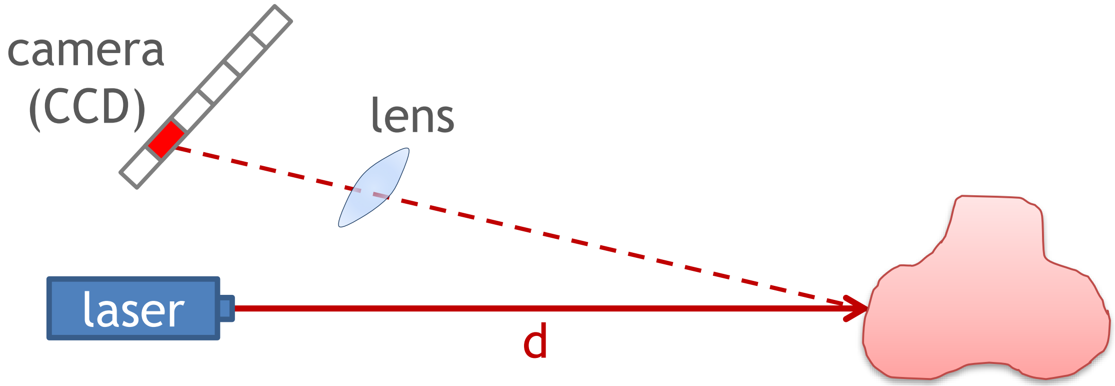


Optical scanning – active lighting

Triangulation laser

- Laser beam and camera
- Laser dot is photographed
- The location of the dot in the image allows triangulation:
  we get the distance to the object





Optical scanning – active lighting

Triangulation laser

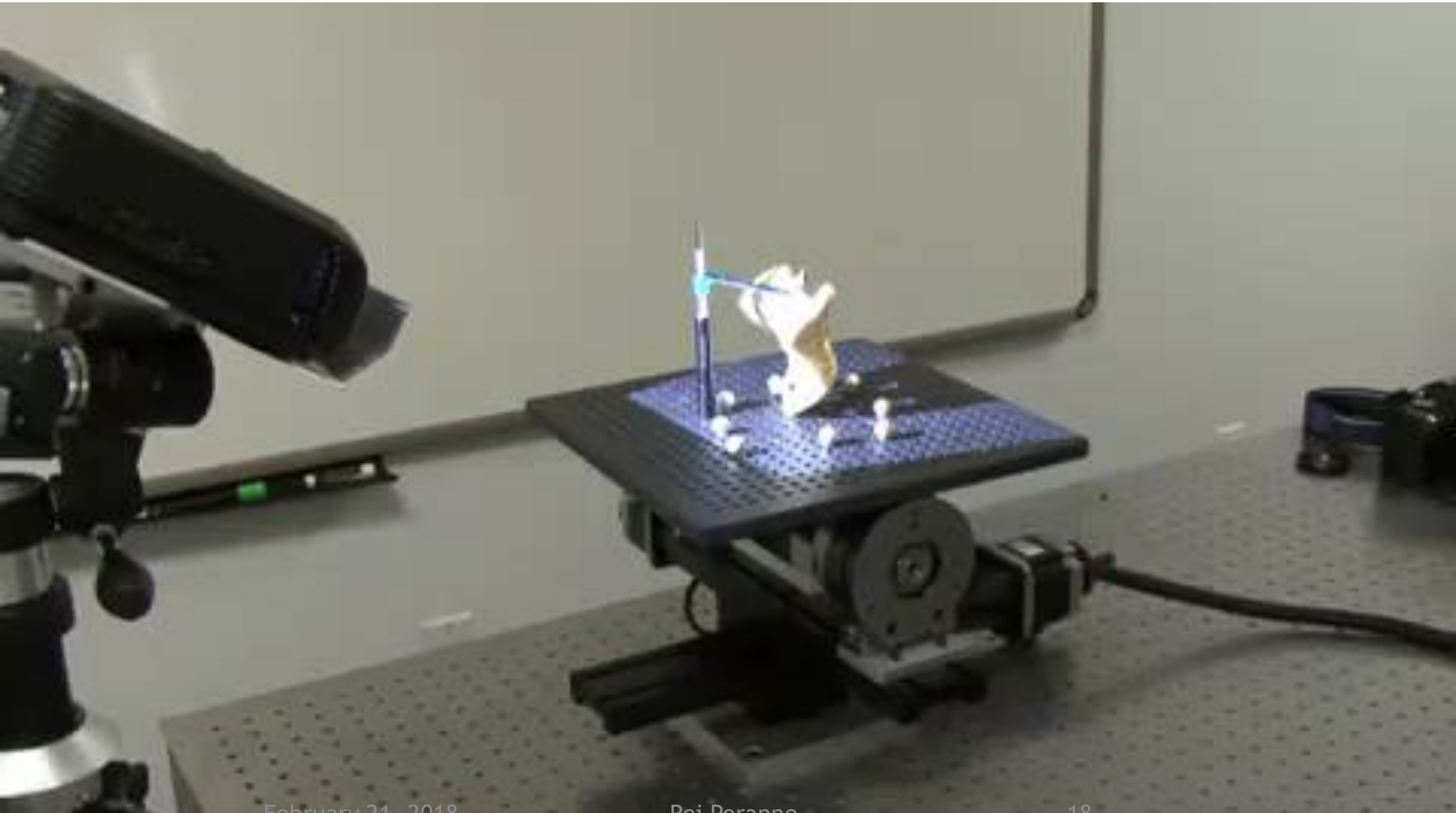
Optical scanning – active lighting

Triangulation laser

- Very precise (tens of microns)
- Small distances (meters)

Optical scanning – active lighting

Structured light



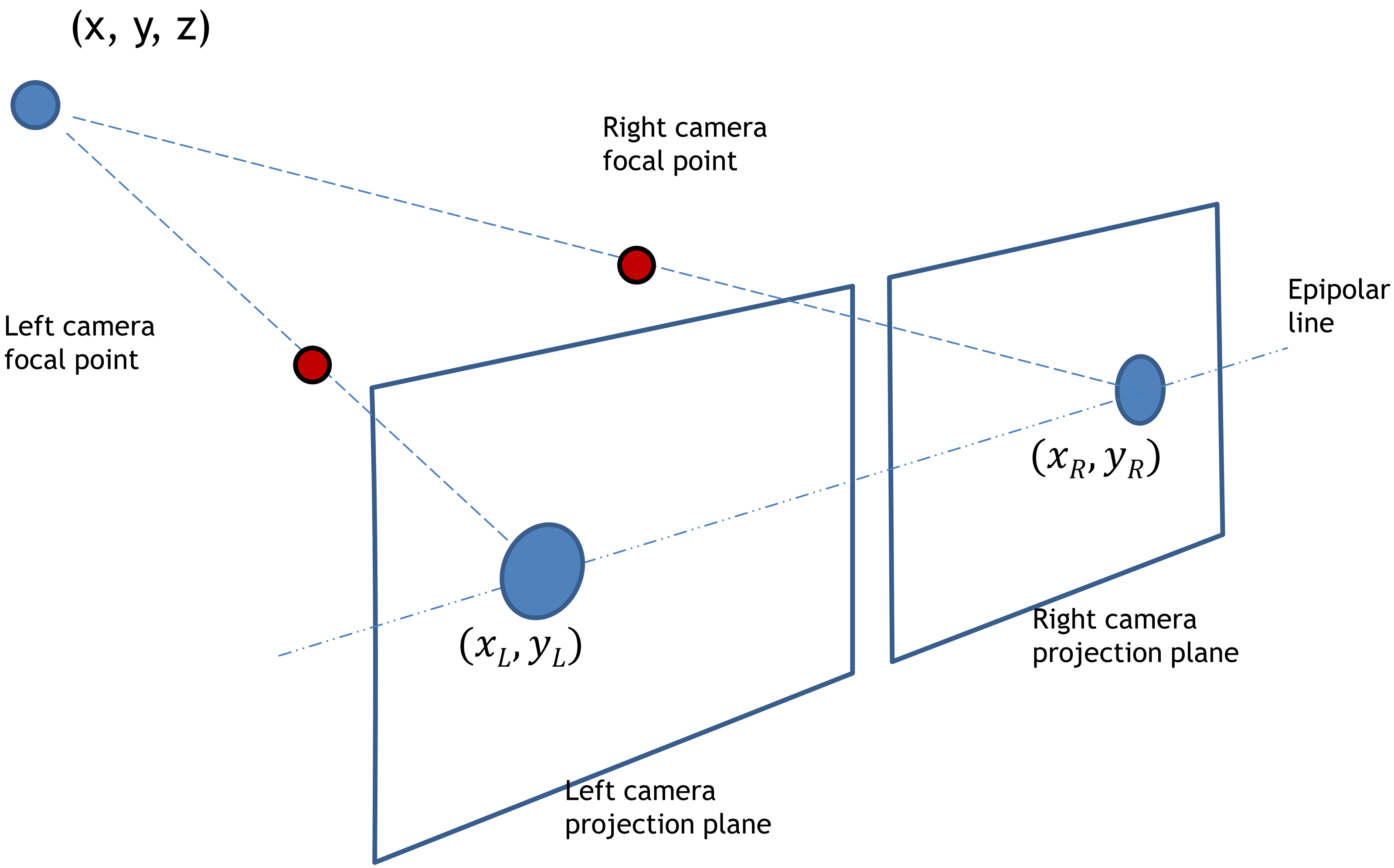
Optical scanning – active lighting

Structured light

- Very fast – 2D pattern at once
  - Used in Kinect, RealSense, Face ID
- Complex distance calculation, prone to noise

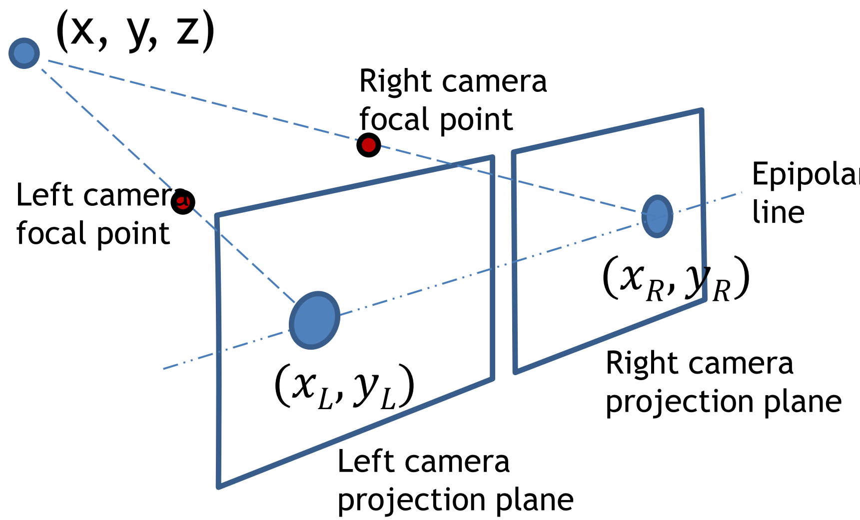
Optical scanning – passive

Stereo



Optical scanning – passive

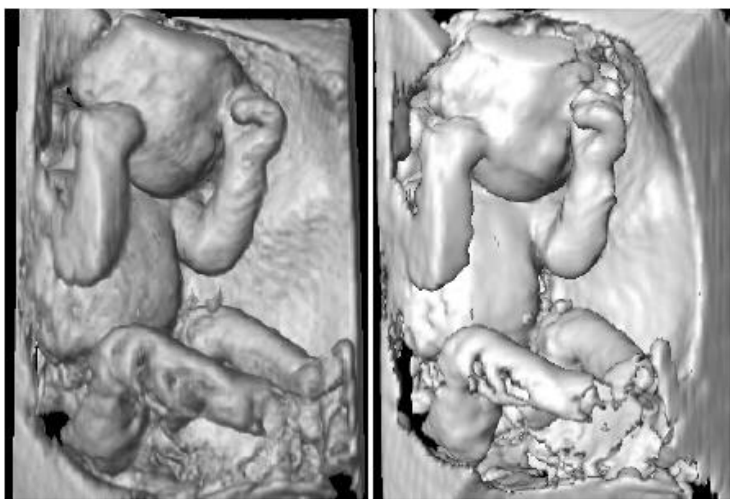
Stereo

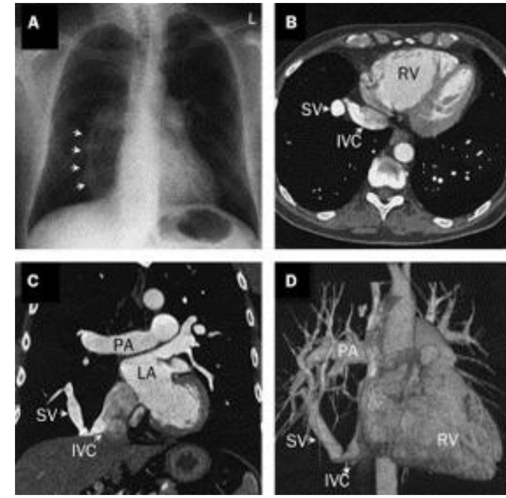


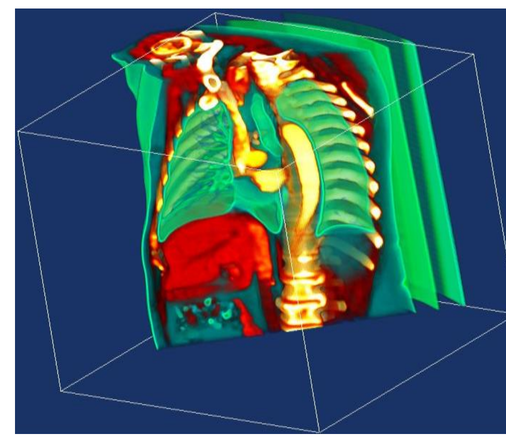
- No need for special lighting/radiation
- Requires two (or more) cameras
  - Feature matching and triangulation

Imaging

- Ultrasound, CT, MRI
- Discrete volume of density data
- First need to segment the desired object (contouring)



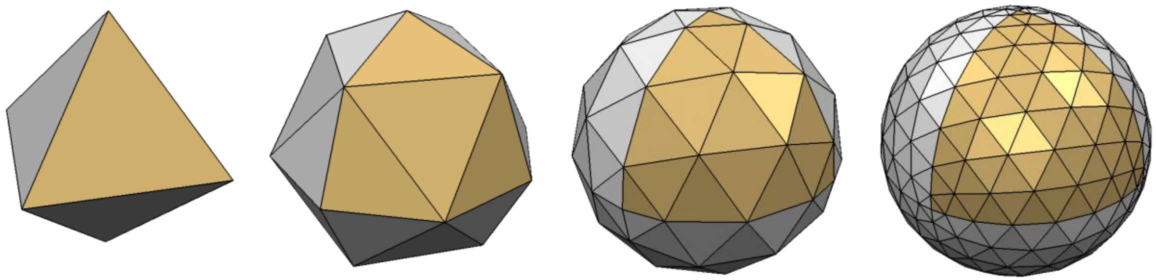


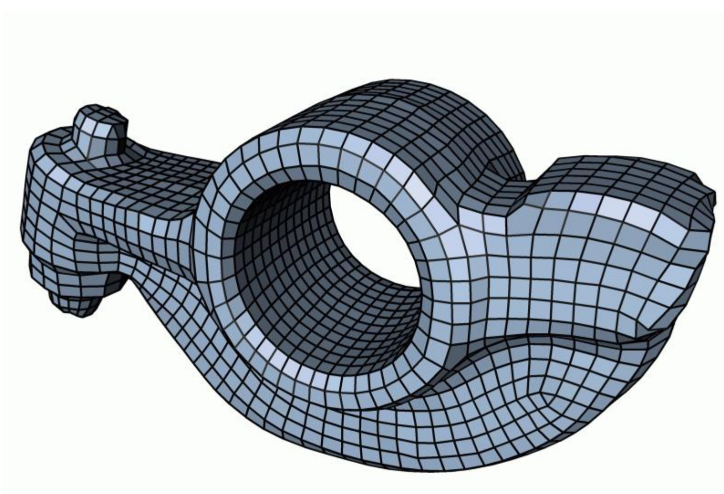


Meshes

Polygonal Meshes

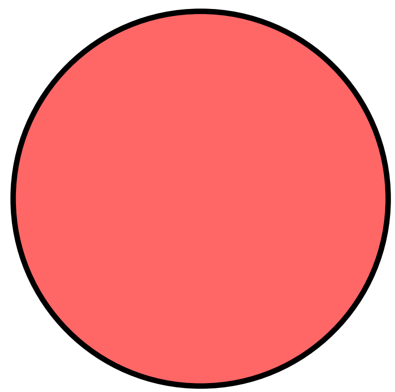
- Boundary representations of objects





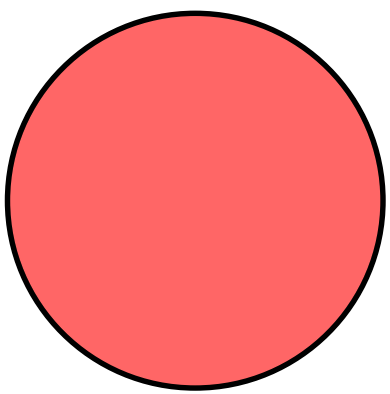
Meshes as Approximations of Smooth Surfaces

- Piecewise linear approximation
  - Error decreases with element size



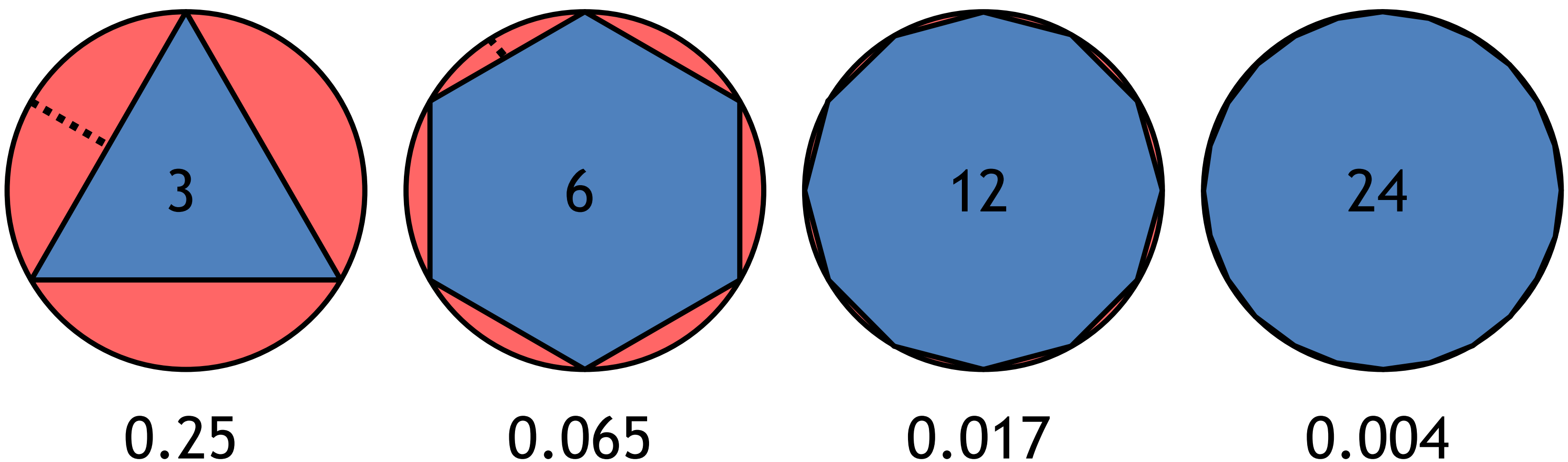
Meshes as Approximations of Smooth Surfaces

- Piecewise linear approximation
  - Error decreases with element size



Meshes as Approximations of Smooth Surfaces

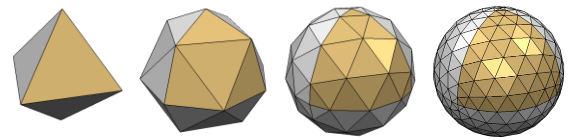
- Piecewise linear approximation
  - Error decreases with element size



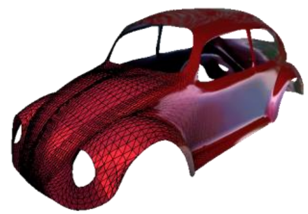
Polygonal Meshes

Polygonal meshes are a good representation

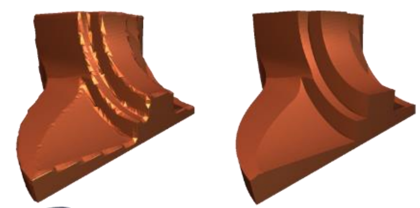
- approximation



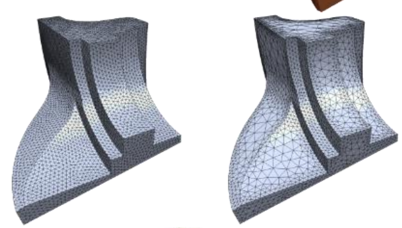
- arbitrary topology



- piecewise smooth surfaces



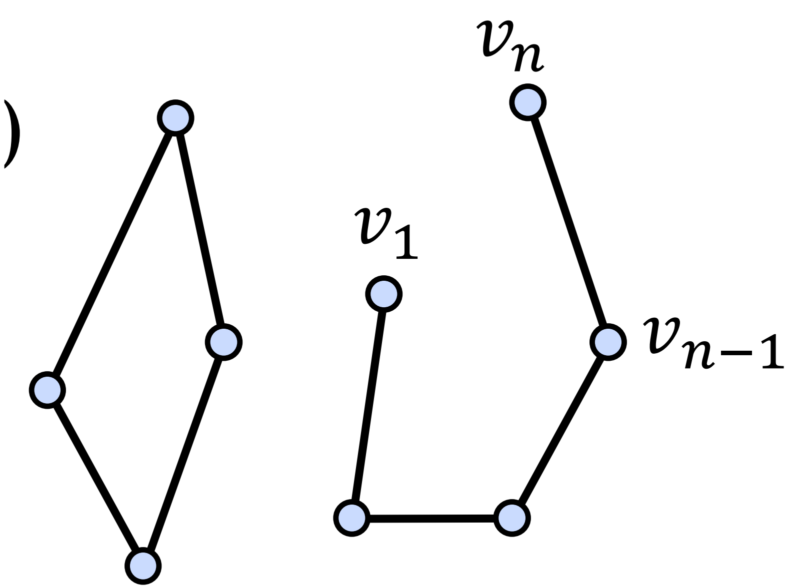
- adaptive refinement



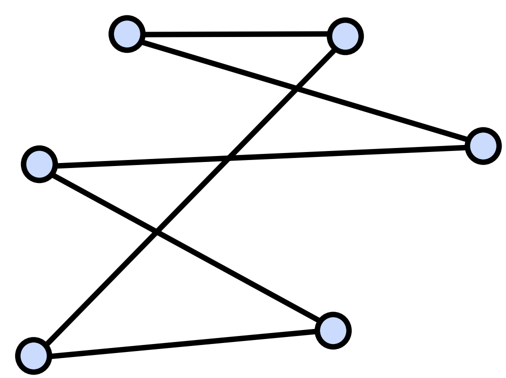
- efficient rendering

Polyline/Polygon

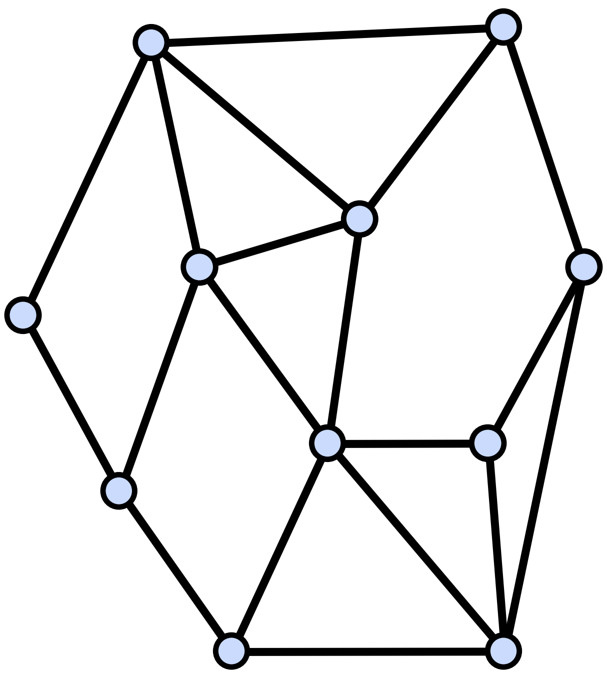
- Vertices: $v_1, v_2, \dots, v_n$
- Edges: $(v_1, v_2), \dots, (v_{n-1}, v_n)$



- Closed: $v_1 = v_n$
- Planar: all vertices on a plane
- Simple: not self-intersecting



Polygonal Mesh

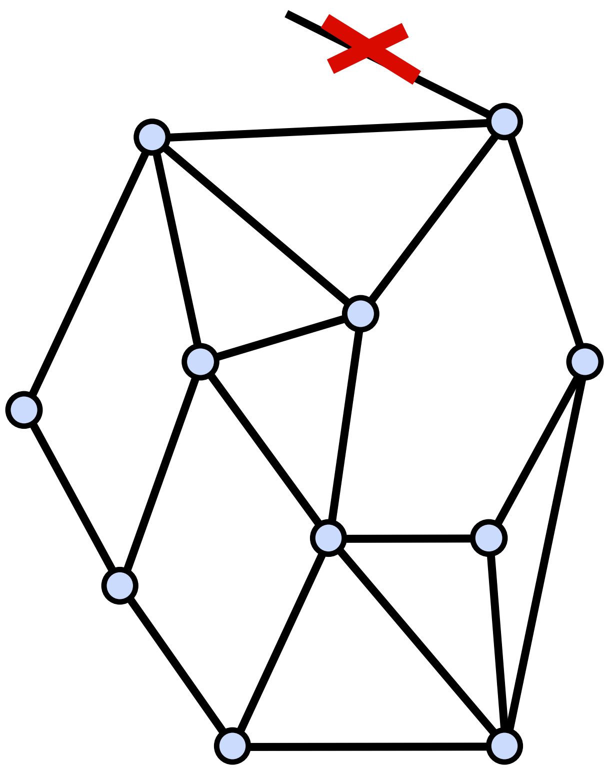


- A finite set M of closed, simple polygons f_j is a polygonal mesh

$$M = \langle V, E, F \rangle$$

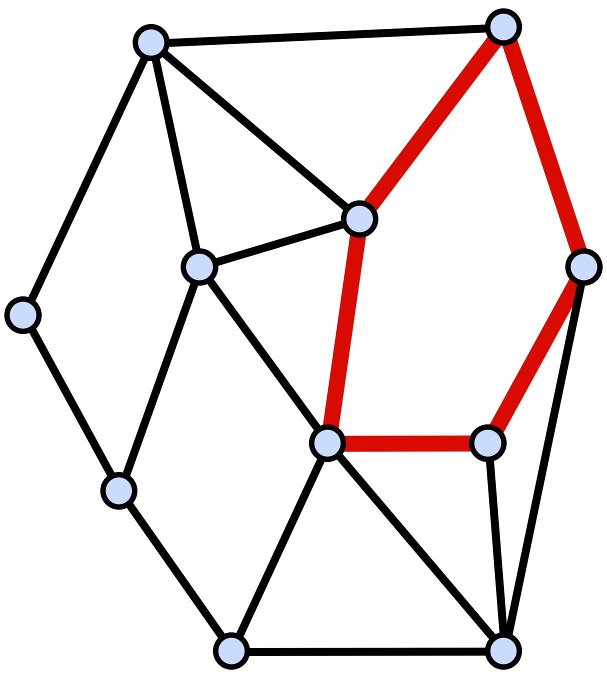
vertices edges faces

Polygonal Mesh



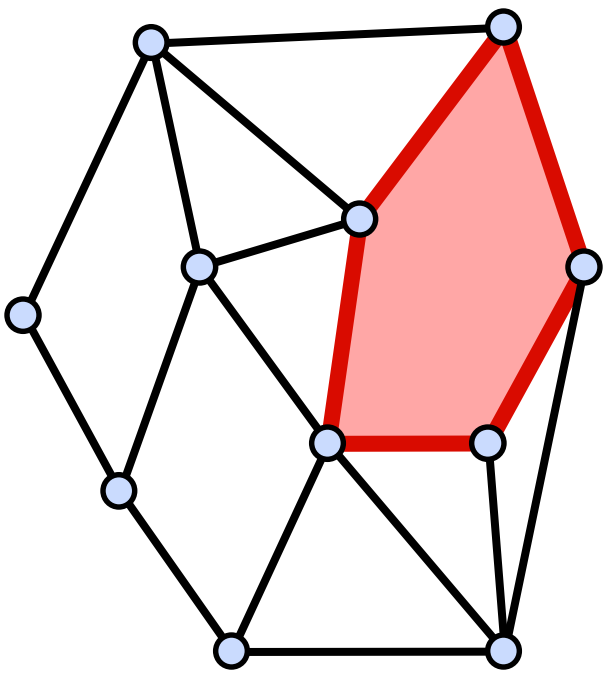
- A finite set M of closed, simple polygons f_j is a polygonal mesh
- Every edge belongs to at least one polygon

Polygonal Mesh



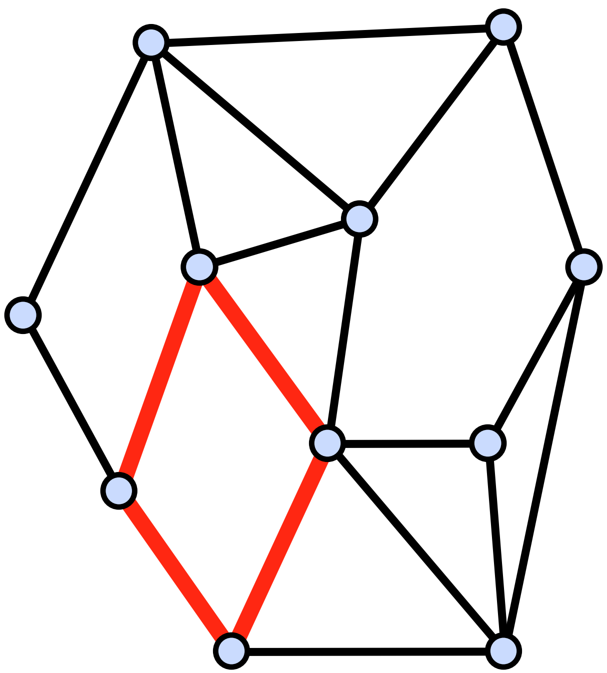
- A finite set M of closed, simple polygons f_j is a polygonal mesh
- Every edge belongs to at least one polygon
- Each f_j defines a **face** of the polygonal mesh

Polygonal Mesh



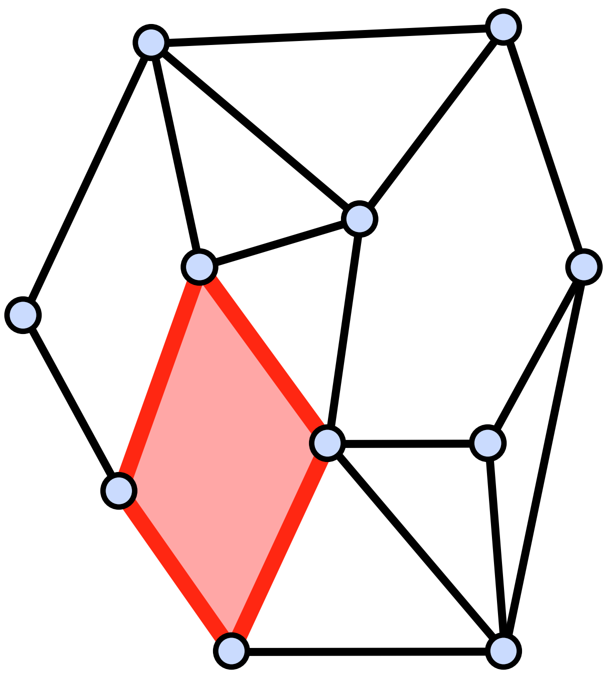
- A finite set M of closed, simple polygons f_j is a polygonal mesh
- Every edge belongs to at least one polygon
- Each f_j defines a **face** of the polygonal mesh

Polygonal Mesh



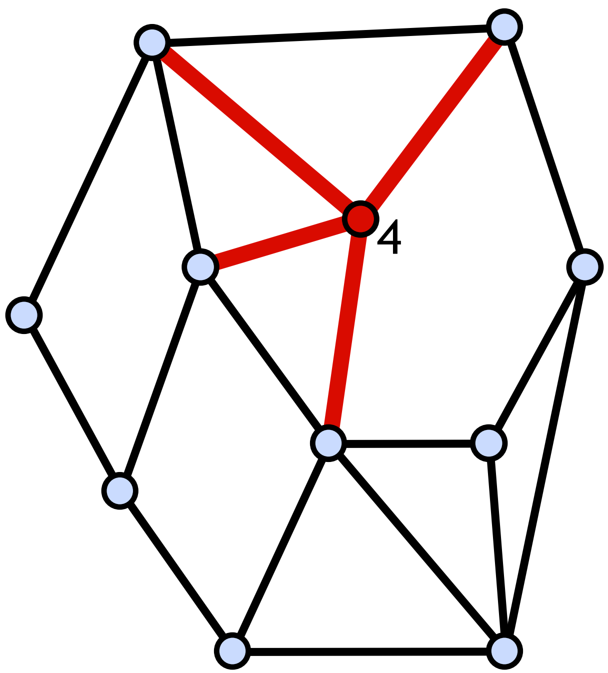
- A finite set M of closed, simple polygons f_j is a polygonal mesh
- Every edge belongs to at least one polygon
- Each f_j defines a **face** of the polygonal mesh

Polygonal Mesh



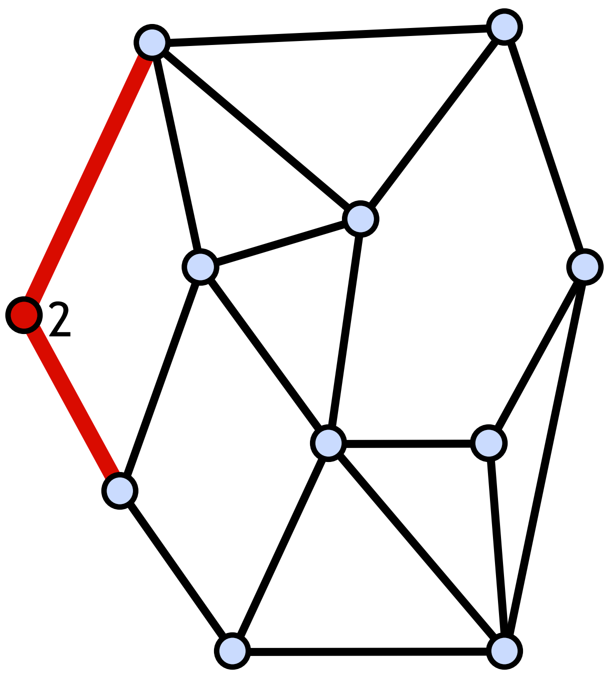
- A finite set M of closed, simple polygons f_j is a polygonal mesh
- Every edge belongs to at least one polygon
- Each f_j defines a **face** of the polygonal mesh

Polygonal Mesh



Vertex **degree** or **valence**
=
number of incident edges

Polygonal Mesh

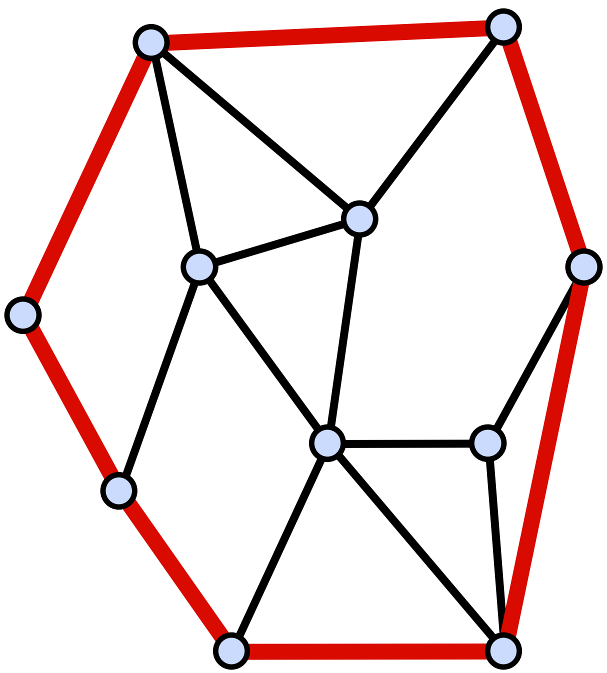


Vertex **degree** or **valence**

=

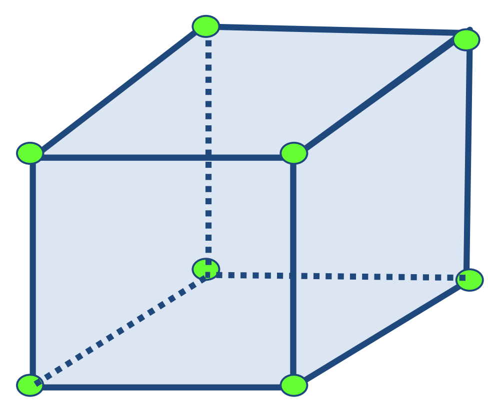
number of incident edges

Polygonal Mesh



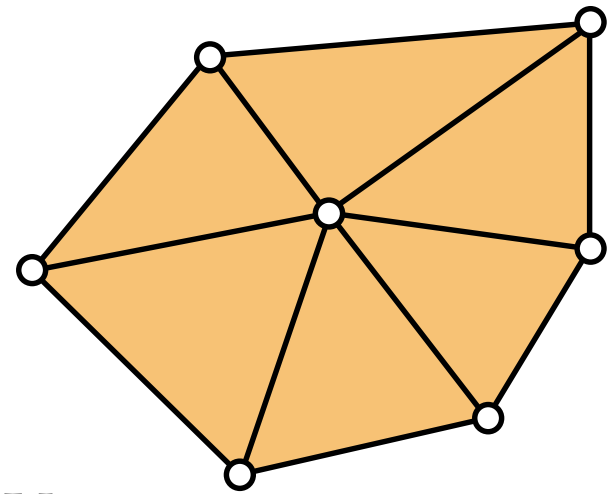
Boundary: the set of all edges that belong to only one face

- Either empty or forms closed loops
- If empty, then the polygonal mesh is closed



Triangle Meshes

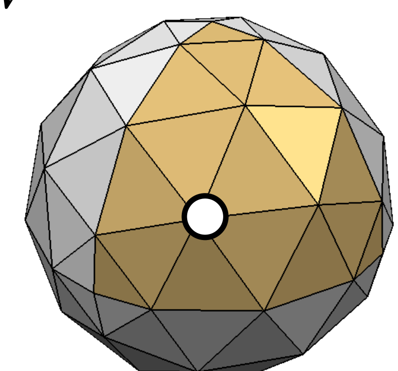
- Connectivity: vertices, edges, triangles
  - AKA Topology, Combinatorics
- Geometry: vertex positions



$$V = \{v_1, \ldots, v_n\}$$

$$E = \{e_1, \ldots, e_k\}, \quad e_i \in V \times V$$

$$F = \{f_1, \ldots, f_m\}, \quad f_i \in V \times V \times V$$



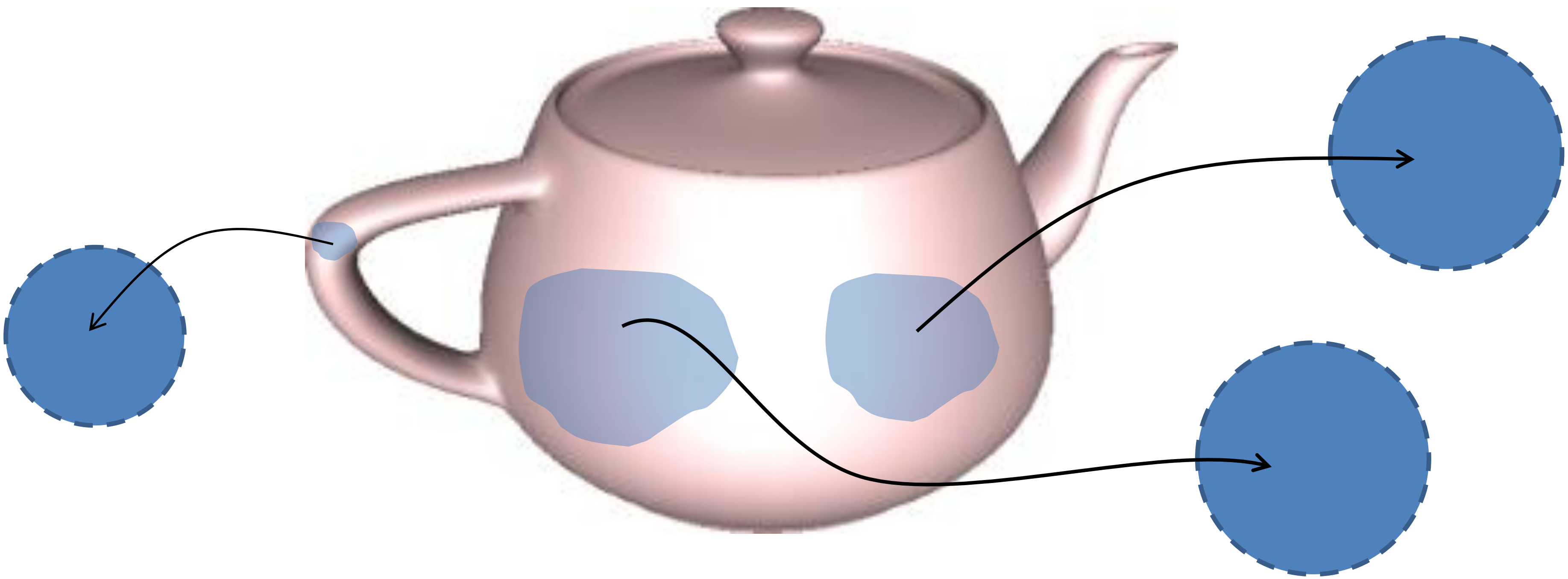
$$P = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}, \quad \mathbf{p}_i \in \mathbb{R}^3$$

ETH zürich

Manifolds

A surface is a closed **2D manifold** if it is everywhere locally **homeomorphic** to a disk

“Looks like”

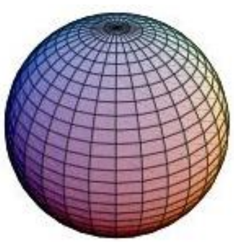


Manifolds

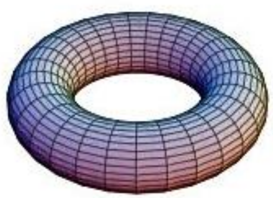
Boundary points are homeomorphic to a half-disks

Quiz

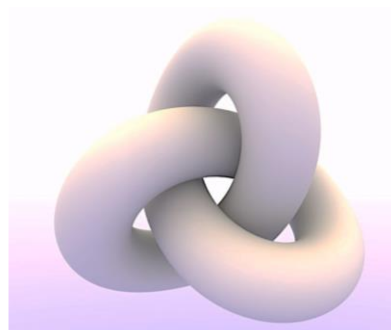
- For each case, decide if it is a 2-manifold (possibly with boundary) or not. If not, explain why not.



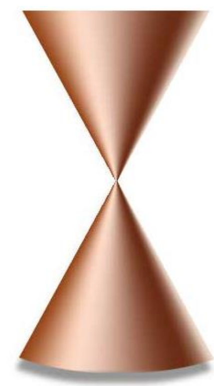
Case 1



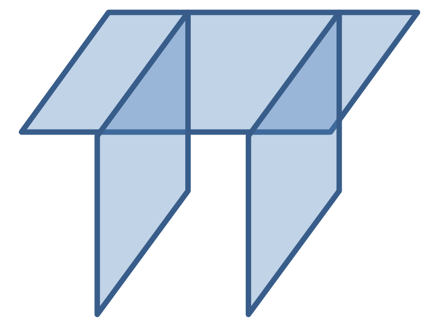
Case 2



Case 3



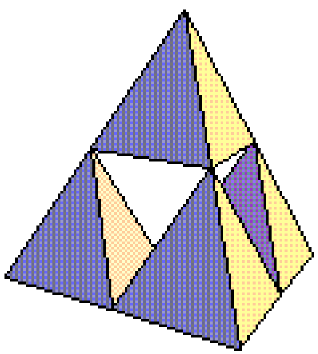
Case 4



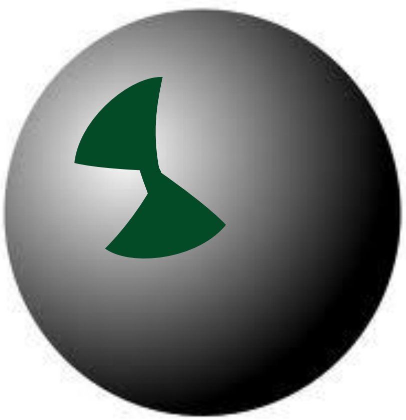
Case 5

Quiz

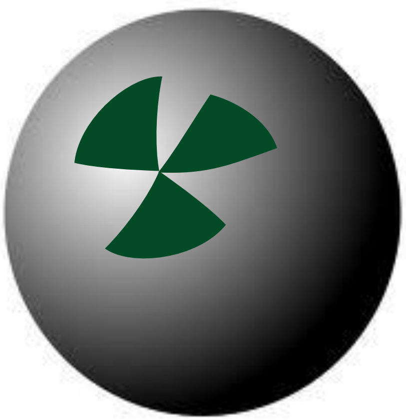
- For each case, decide if it is a 2-manifold (possibly with boundary) or not. If not, explain why not.



Case 6



Case 7



Case 8

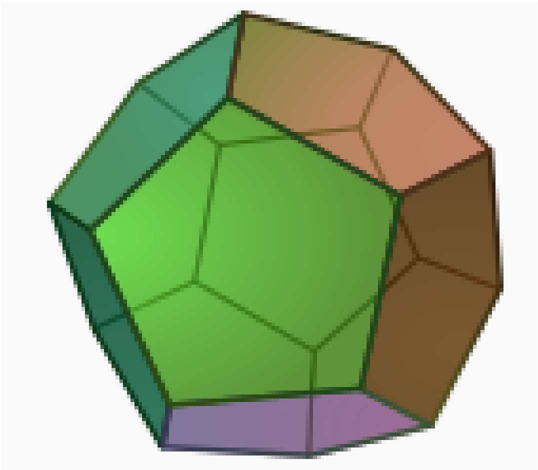
Manifold meshes

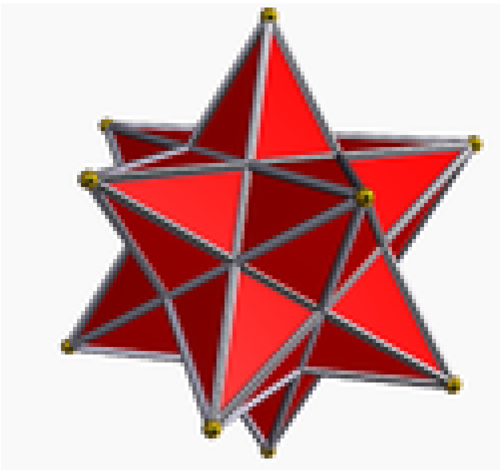
- In a manifold mesh, at most 2 faces share an edge
  - Boundary edges: have one incident face
  - Inner edges have two incident faces
- A manifold vertex has 1 connected **ring** of faces around it, or 1 connected half-ring (boundary)

✓ manifold ✗ non-manifold vertex ✗ non-manifold edge ✗ non-manifold vertex

Manifold meshes

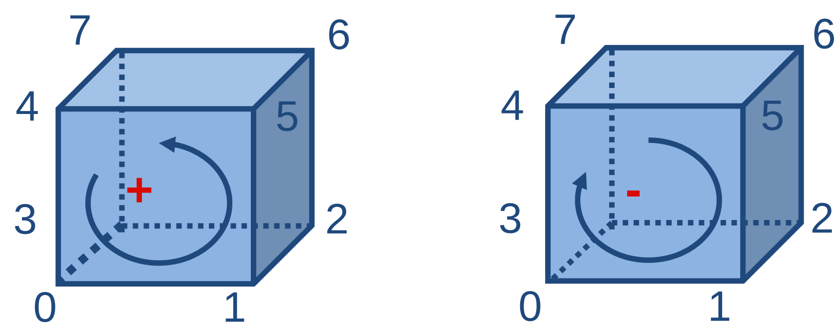
- If closed and not intersecting, a manifold divides the space into inside and outside
- A closed manifold polygonal mesh is called **polyhedron**





Orientation

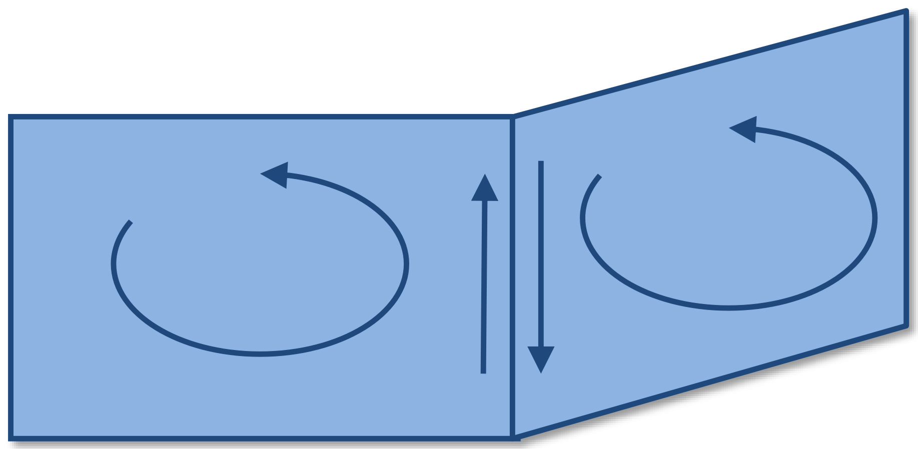
- Every face of a polygonal mesh is orientable
  - Clockwise vs. counterclockwise order of face vertices
  - Defines sign/direction of the surface normal



ETH zürich

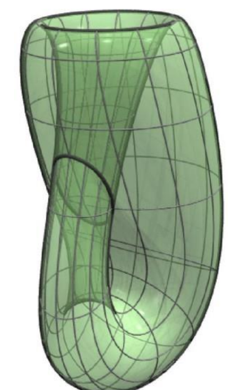
Orientation

- Consistent orientation of neighboring faces:



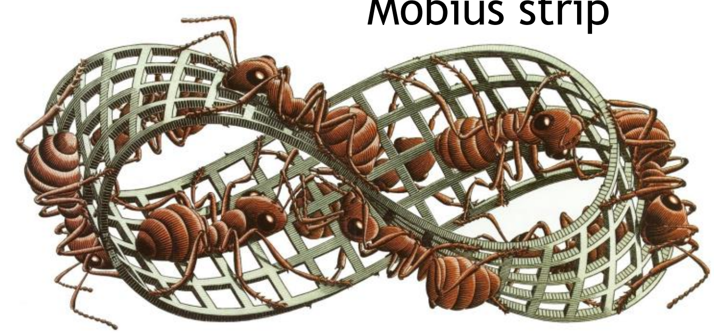
Orientability

- A polygonal mesh is orientable, if the incident faces to every edge can be consistently oriented
  - If the faces are consistently oriented for every edge, the mesh is oriented
- Notes
  - Every non-orientable closed mesh embedded in $\mathbb{R}^3$ intersects itself
  - The surface of a polyhedron is always orientable



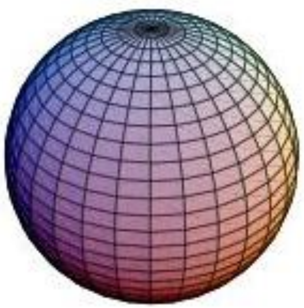
Klein bottle

Möbius strip

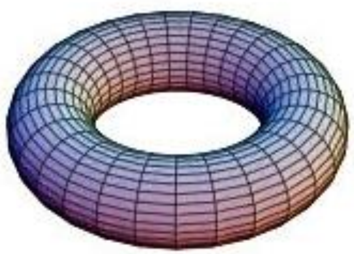


Global Topology of Meshes

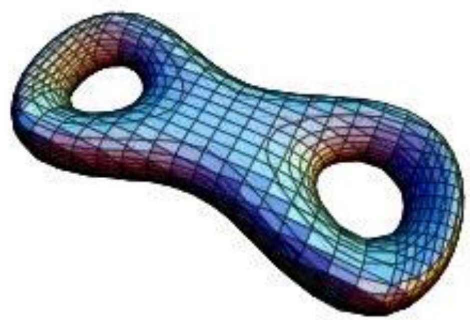
- **Genus:** ½ × the maximal number of closed paths that do not disconnect the graph.
  - Informally, the number of handles (“donut holes”).



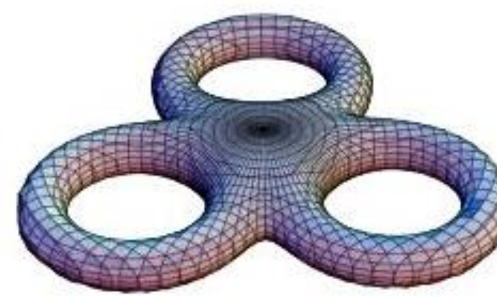
Genus 0



Genus 1



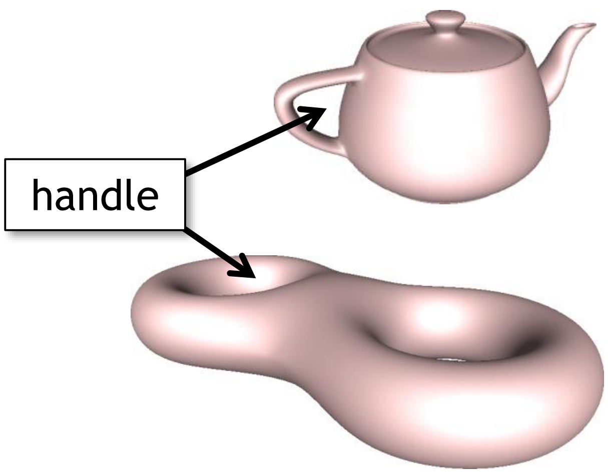
Genus 2

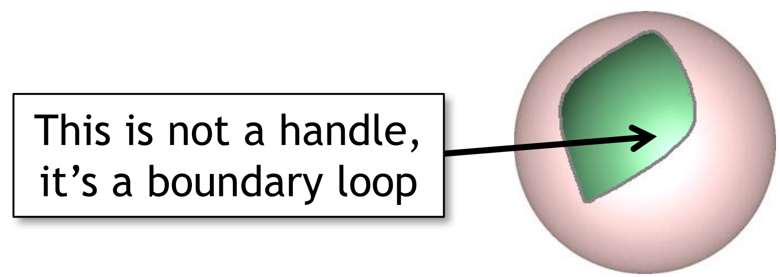


Genus 3

Global Topology of Meshes

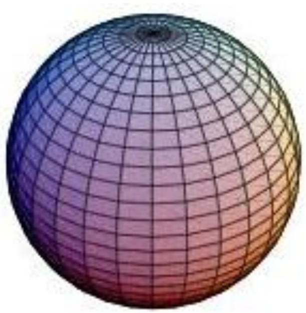
- **Genus:** ½ × the maximal number of closed paths that do not disconnect the graph.
  - Informally, the number of handles ("donut holes").



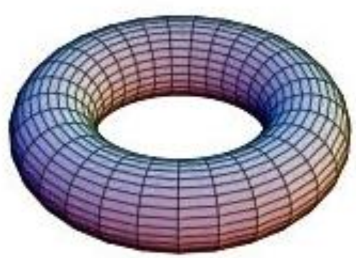


Global Topology of Meshes

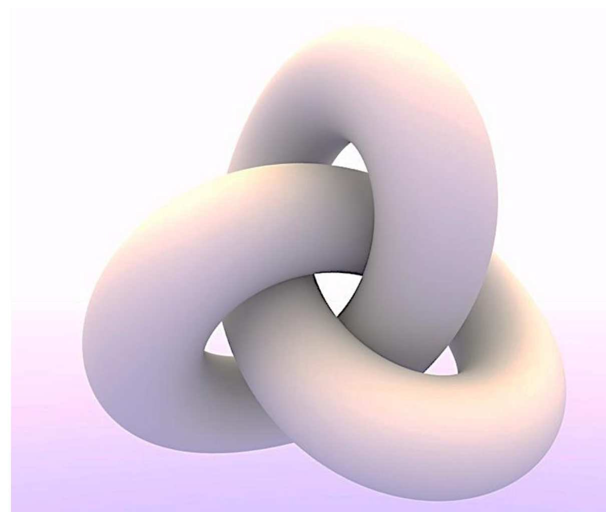
- **Genus:** ½ × the maximal number of closed paths that do not disconnect the graph.



Genus 0



Genus 1



?

Euler-Poincaré Formula

- Theorem (Euler): The sum

$$\chi(M) = v - e + f$$

is **constant** for a given surface topology, no matter which (manifold) mesh we choose.

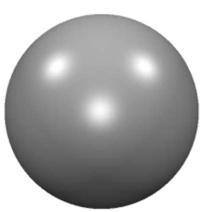
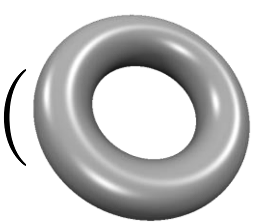
  - v = number of vertices
  - e = number of edges
  - f = number of faces

Euler-Poincaré Formula

- For orientable meshes:

$$v - e + f = 2(c - g) - b = \chi(M)$$

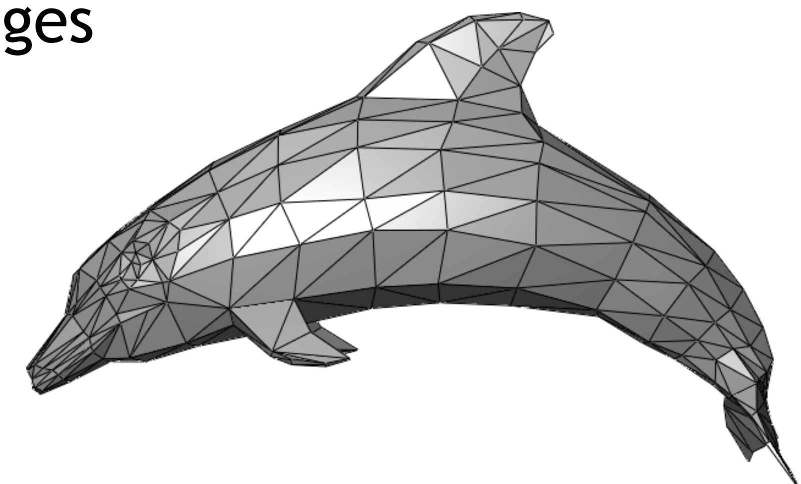
  - c = number of connected components
  - g = genus
  - b = number of boundary loops

$\chi($  $) = 2$ $\chi($  $) = 0$

Euler-Poincaré Formula

Count #edges and #faces in a **closed triangle mesh:**

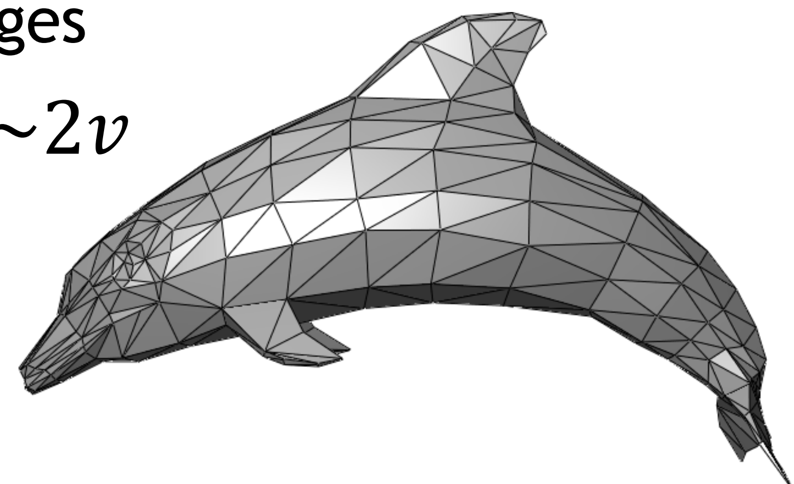
- Ratio of edges to faces: $2e = 3f \Rightarrow e = \frac{3}{2} f$
  - each edge belongs to exactly 2 triangles
  - each triangle has exactly 3 edges



Euler-Poincaré Formula

Count #edges and #faces in a **closed triangle mesh:**

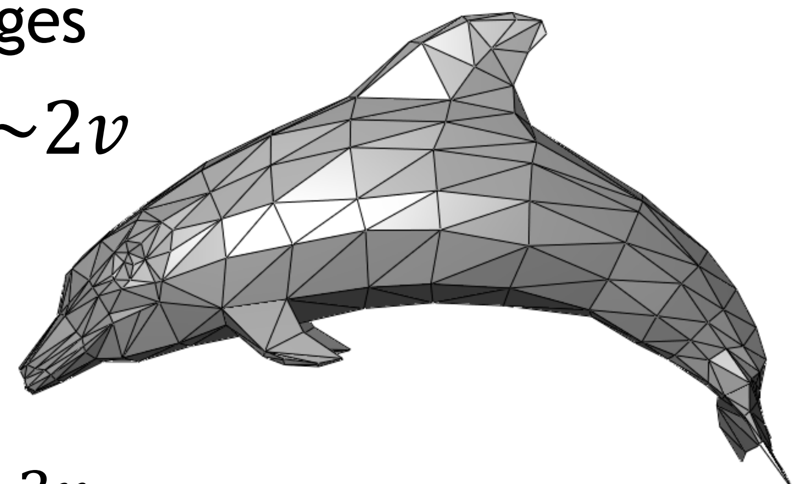
- Ratio of edges to faces: $2e = 3f \Rightarrow e = \frac{3}{2}f$
  - each edge belongs to exactly 2 triangles
  - each triangle has exactly 3 edges
- Ratio of vertices to faces: $f \sim 2v$
  - $2 = v - e + f = v - \frac{3}{2}f + f$
  - $2 + \frac{f}{2} = v$



Euler-Poincaré Formula

Count #edges and #faces in a **closed triangle mesh:**

- Ratio of edges to faces: $2e = 3f \Rightarrow e = \frac{3}{2}f$
  - each edge belongs to exactly 2 triangles
  - each triangle has exactly 3 edges
- Ratio of vertices to faces: $f \sim 2v$
  - $2 = v - e + f = v - \frac{3}{2}f + f$
  - $2 + \frac{f}{2} = v$
  - Ratio of edges to vertices: $e \sim 3v$
  - **Average degree of a vertex: 6**



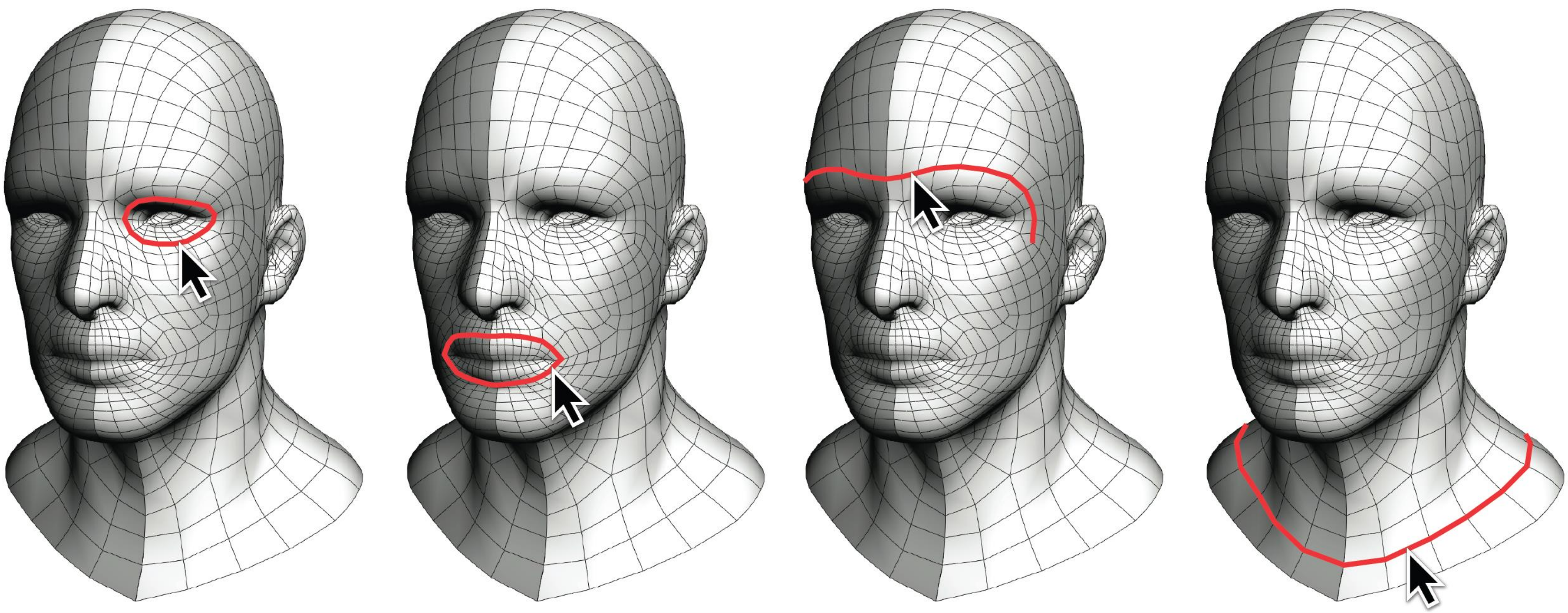
Regularity

- Triangle mesh: average valence = 6
- Quad mesh: average valence = 4

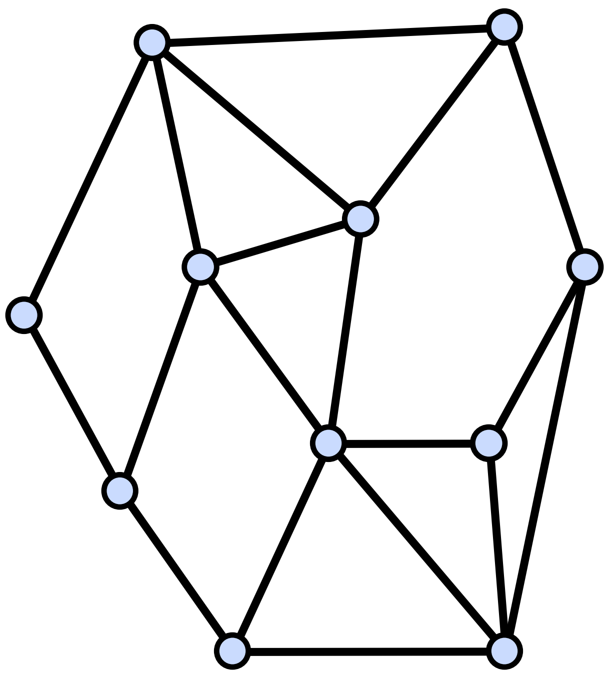
- **Regular mesh**: all faces have the same number of edges and all vertex degrees are equal
- **Quasi-regular mesh:**
  - Almost all of vertices have degree 6 (4).

Quad vs. Triangle Meshes

- Edge loops are great for editing



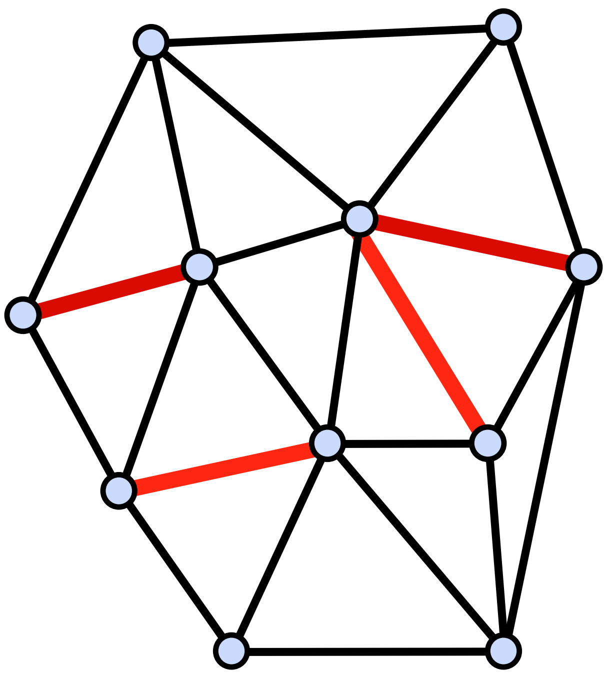
Triangulation



- Every face is a triangle

- Simplifies data structures
- Simplifies rendering
- Simplifies algorithms
- Each face planar and convex
- Any polygon can be triangulated

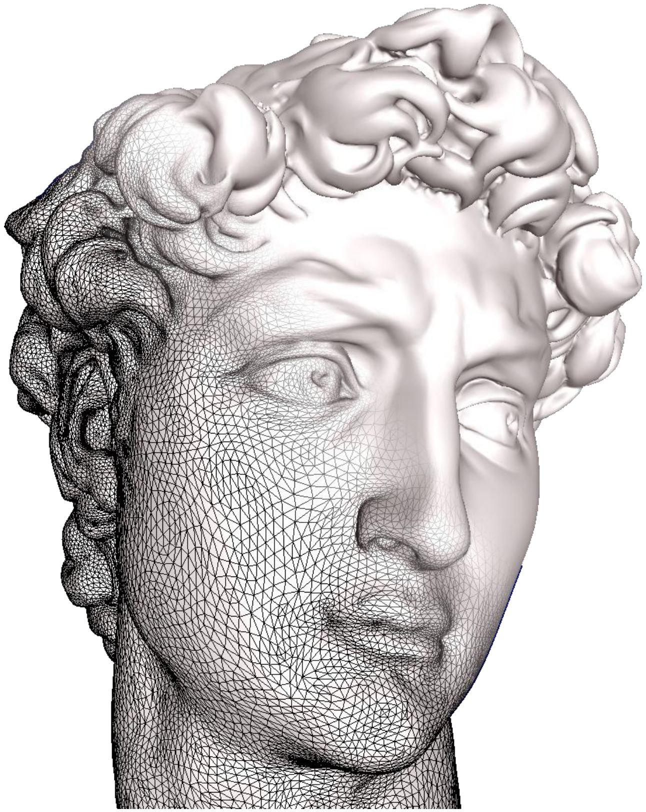
Triangulation



- Every face is a triangle

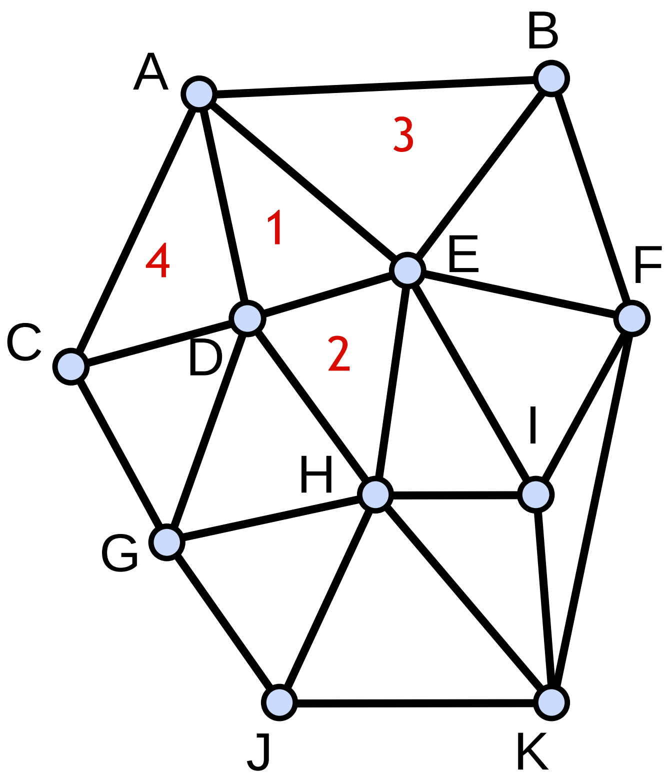
- Simplifies data structures
- Simplifies rendering
- Simplifies algorithms
- Each face planar and convex
- Any polygon can be triangulated

Data Structures



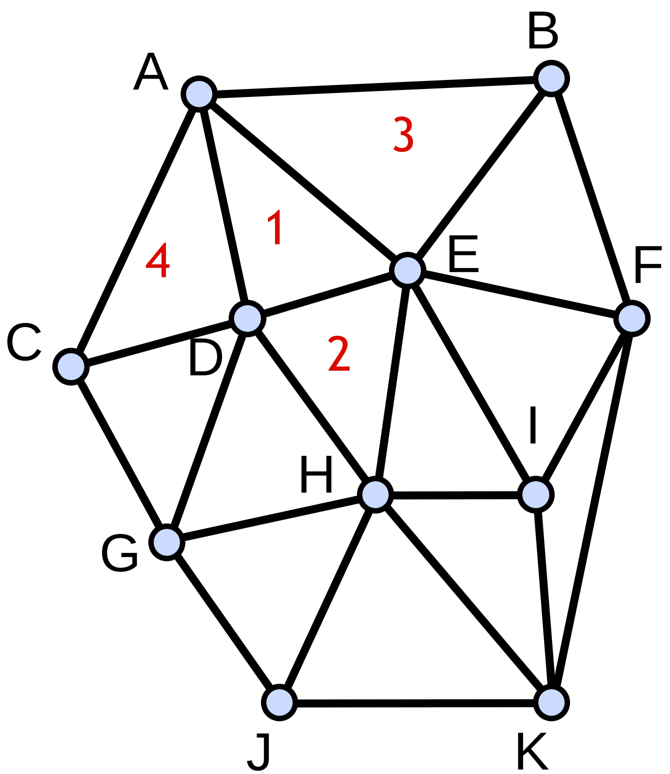
- What should be stored?
  - Geometry: 3D coordinates
  - Connectivity
    - Adjacency relationships
  - Attributes
    - Normal, color, texture coordinates
    - Per vertex, face, edge

Data Structures



- What should be supported?
  - Rendering
  - Geometry queries
    - What are the vertices of face #2?
    - Is vertex *A* adjacent to vertex *H*?
    - Which faces are adjacent to face #1?
  - Modifications
    - Remove/add a vertex/face
    - Vertex split, edge collapse

Data Structures



- How good is a data structure?
  - Time to construct
  - Time to answer a query
  - Time to perform an operation
  - Space complexity
  - Redundancy
- Criteria for design
  - Expected number of vertices
  - Available memory
  - Required operations
  - Distribution of operations

Triangle List

- STL format (used in CAD)
- Storage
  - Face: 3 positions
- No connectivity information

Triangles			
0	x0	y0	z0
1	x1	x1	z1
2	x2	y2	z2
3	x3	y3	z3
4	x4	y4	z4
5	x5	y5	z5
6	x6	y6	z6
…	…	…	…

Indexed Face Set

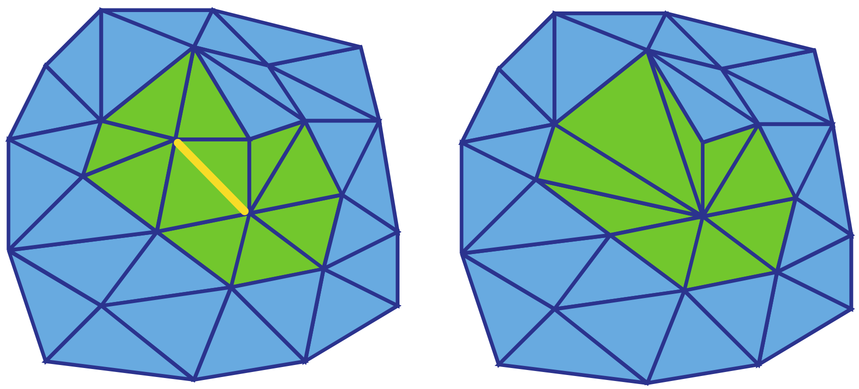
- Used in many formats
- Storage
  - Vertex: position
  - Face: vertex indices
- No *explicit* neighborhood info

Vertices			
v0	x0	y0	z0
v1	x1	x1	z1
v2	x2	y2	z2
v3	x3	y3	z3
v4	x4	y4	z4
v5	x5	y5	z5
v6	x6	y6	z6
...	...	...	...

Triangles			
t0	v0	v1	v2
t1	v0	v1	v3
t2	v2	v4	v3
t3	v5	v2	v6
...	...	...	...

Indexed Face Set: Problems

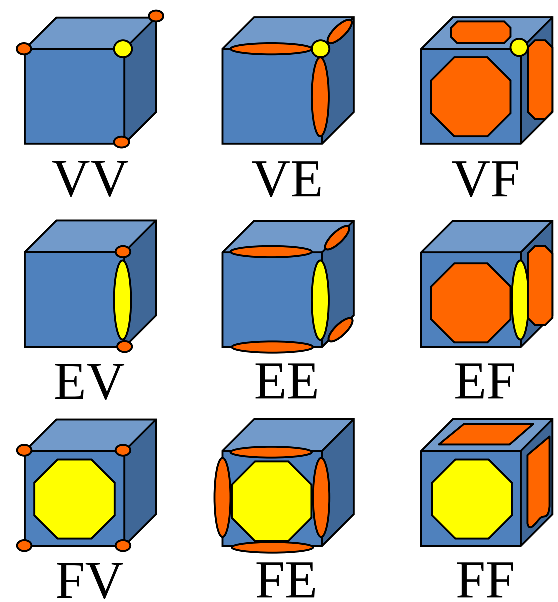
- Information about neighbors is not explicit
  - Finding neighbors vertices/edges/faces is $O(\#V)$
  - Local mesh modifications cost $O(V)$

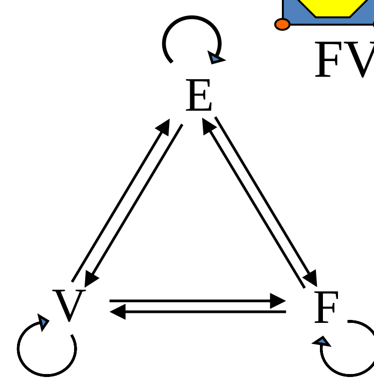


  - Breadth-first search costs $O(k*\#V)$ where k = # found vertices

Neighborhood Relations

- All possible neighborhood relationships:
  1. Vertex - Vertex VV
  2. Vertex - Edge VE
  3. Vertex - Face VF
  4. Edge - Vertex EV
  5. Edge - Edge EE
  6. Edge - Face EF
  7. Face - Vertex FV
  8. Face - Edge FE
  9. Face - Face FF

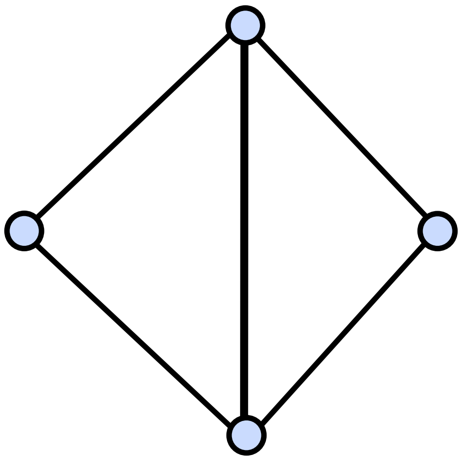




We'd like $O(1)$ time for queries and local updates of these relationships

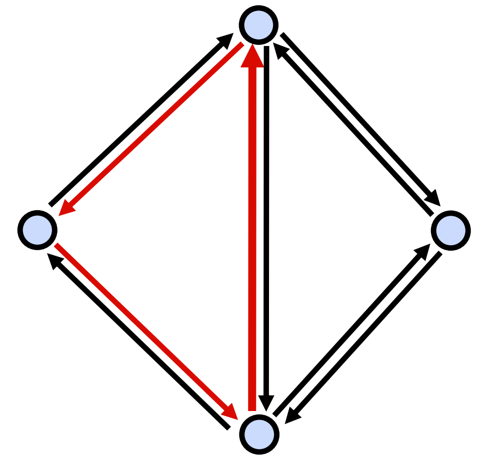
Halfedge data structure

- Introduce orientation into data structure
  - Oriented edges



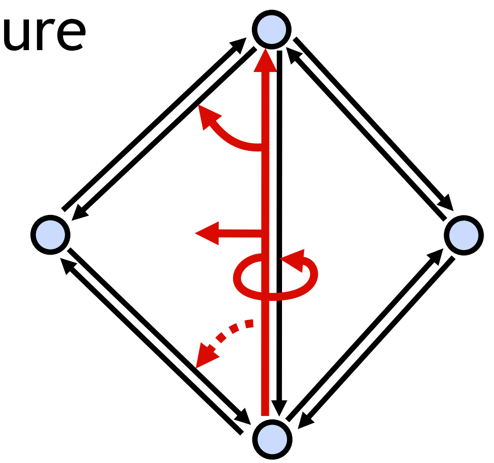
Halfedge data structure

- Introduce orientation into data structure
  - Oriented edges



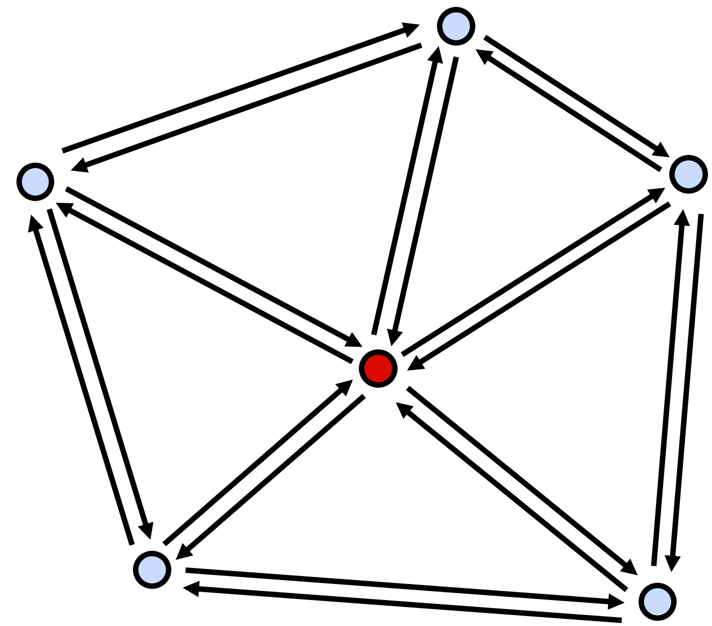
Halfedge data structure

- Introduce orientation into data structure
  - Oriented edges
- Vertex
  - Position
  - 1 outgoing halfedge index
- Halfedge
  - 1 origin vertex index
  - 1 incident face index
  - 3 next, prev, twin halfedge indices
- Face
  - 1 adjacent halfedge index
- Easy traversal, full connectivity



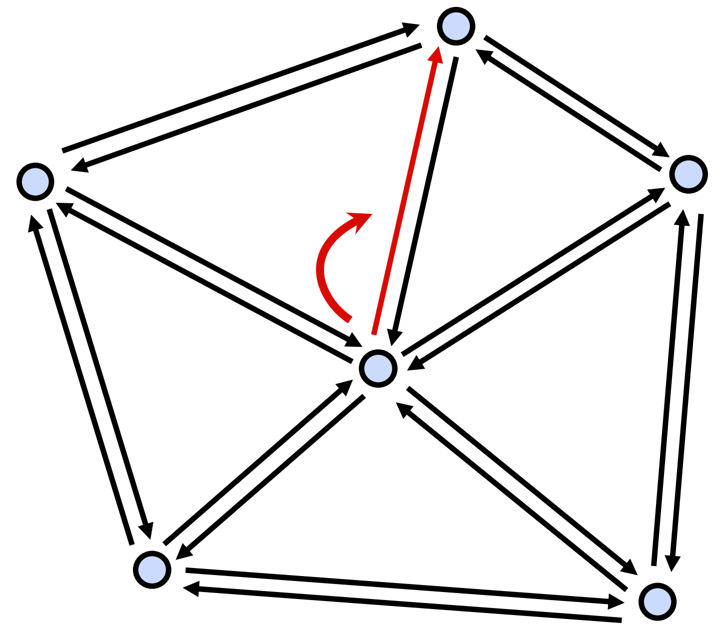
Halfedge data structure

- One-ring traversal
  - Start at vertex



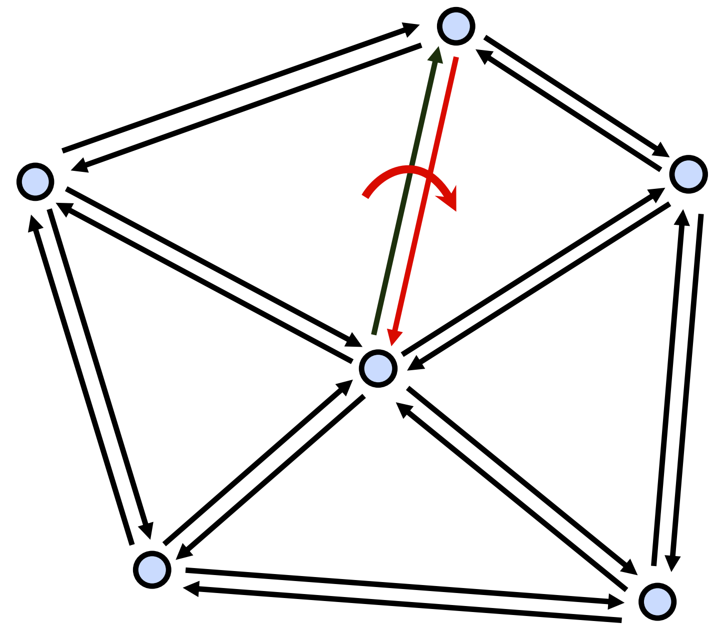
Halfedge data structure

- One-ring traversal
  - Start at vertex
  - Outgoing halfedge



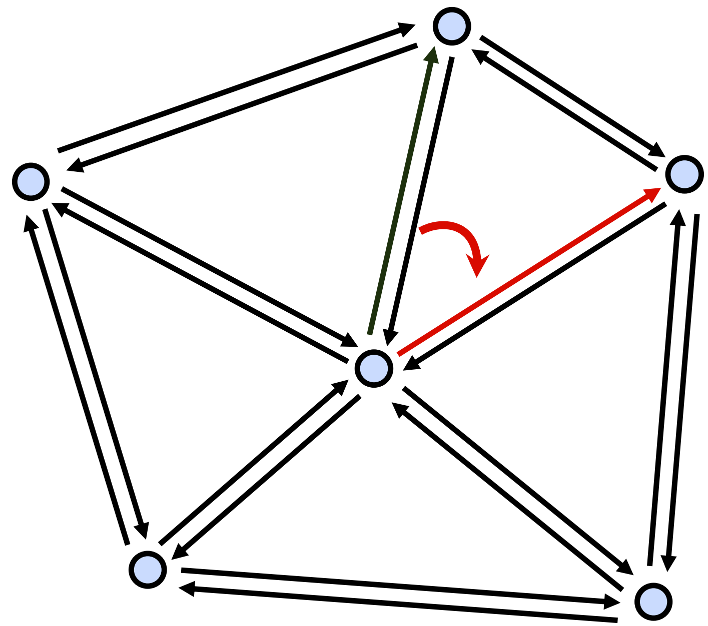
Halfedge data structure

- One-ring traversal
  - Start at vertex
  - Outgoing halfedge
  - Twin halfedge



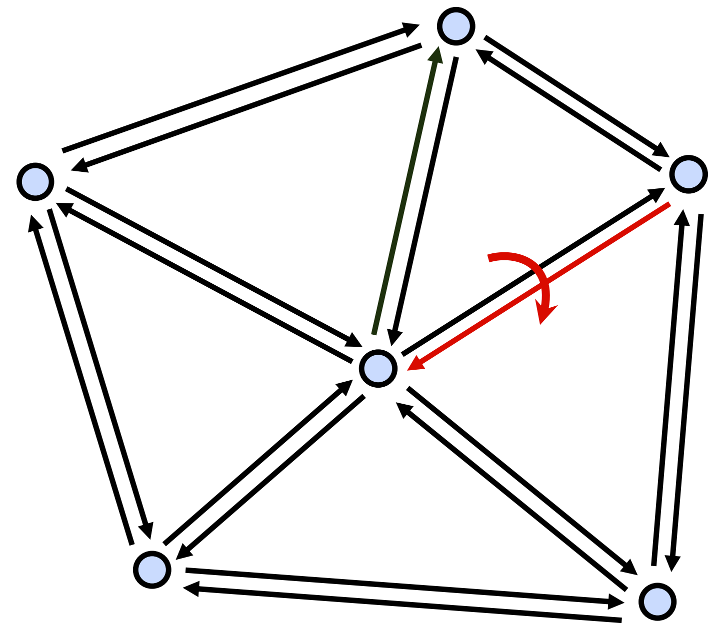
Halfedge data structure

- One-ring traversal
  - Start at vertex
  - Outgoing halfedge
  - Twin halfedge
  - Next halfedge



Halfedge data structure

- One-ring traversal
  - Start at vertex
  - Outgoing halfedge
  - Twin halfedge
  - Next halfedge
  - Twin ...



Halfedge data structure

- Pros: *(assuming bounded vertex valence)*
  - $O(1)$ time for neighborhood relationship queries
  - $O(1)$ time and space for local modifications (edge collapse, vertex insertion…)
- Cons:
  - Heavy – requires storing and managing extra pointers
  - Not as trivial as Indexed Face Set for rendering with OpenGL / Vertex Buffer Objects

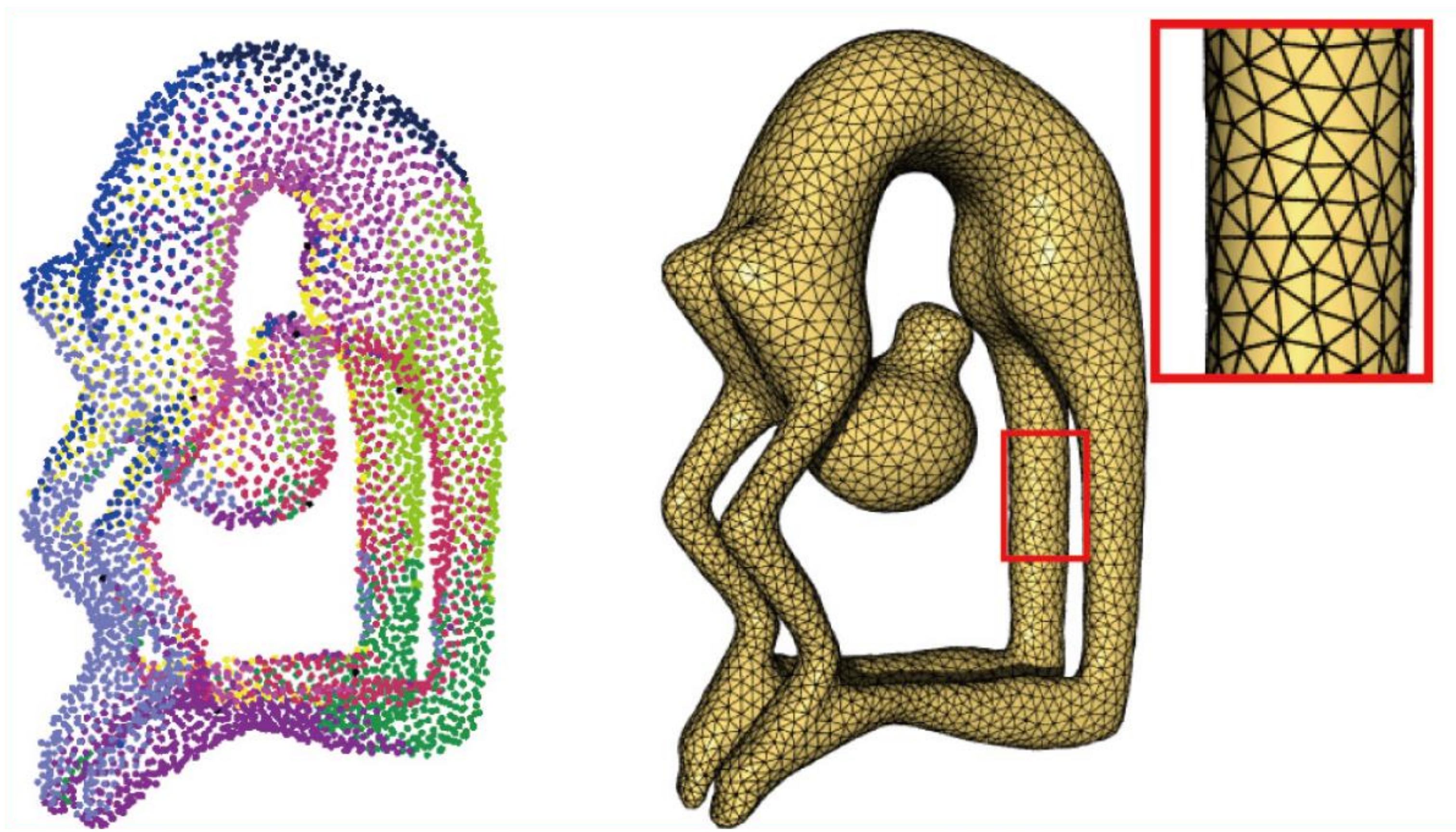
Halfedge Libraries

- CGAL
  - www.cgal.org
  - Computational geometry
- OpenMesh
  - www.openmesh.org
  - Mesh processing

- We will not implement a half-edge data structure in the class. Instead we will work with Indexed Face Set and augment it to have fast queries.

Next time…

Surface reconstruction



Thank you