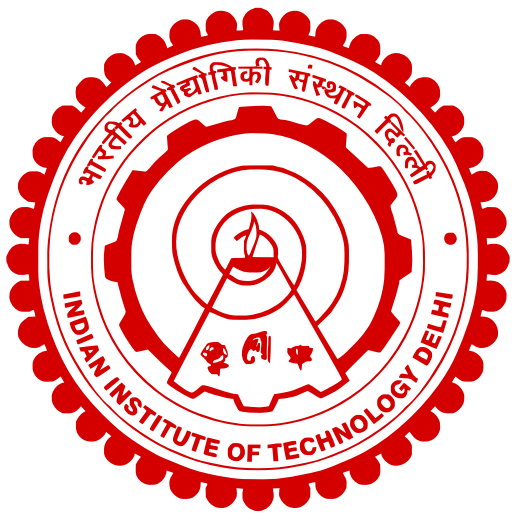


Computer Graphics

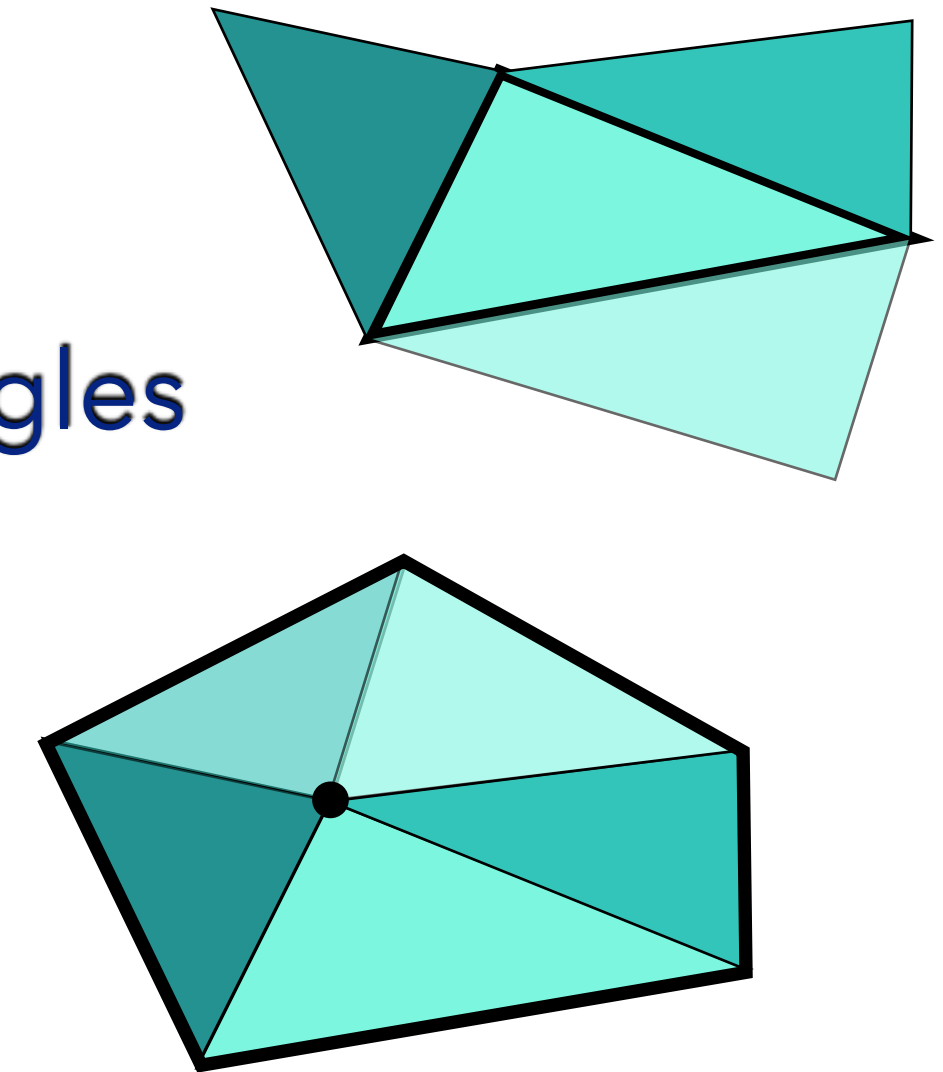
Modeling

Subodh Kumar, Dept of Computer Sc. & Engg., IIT Delhi

Physical Geometry is Volumetric



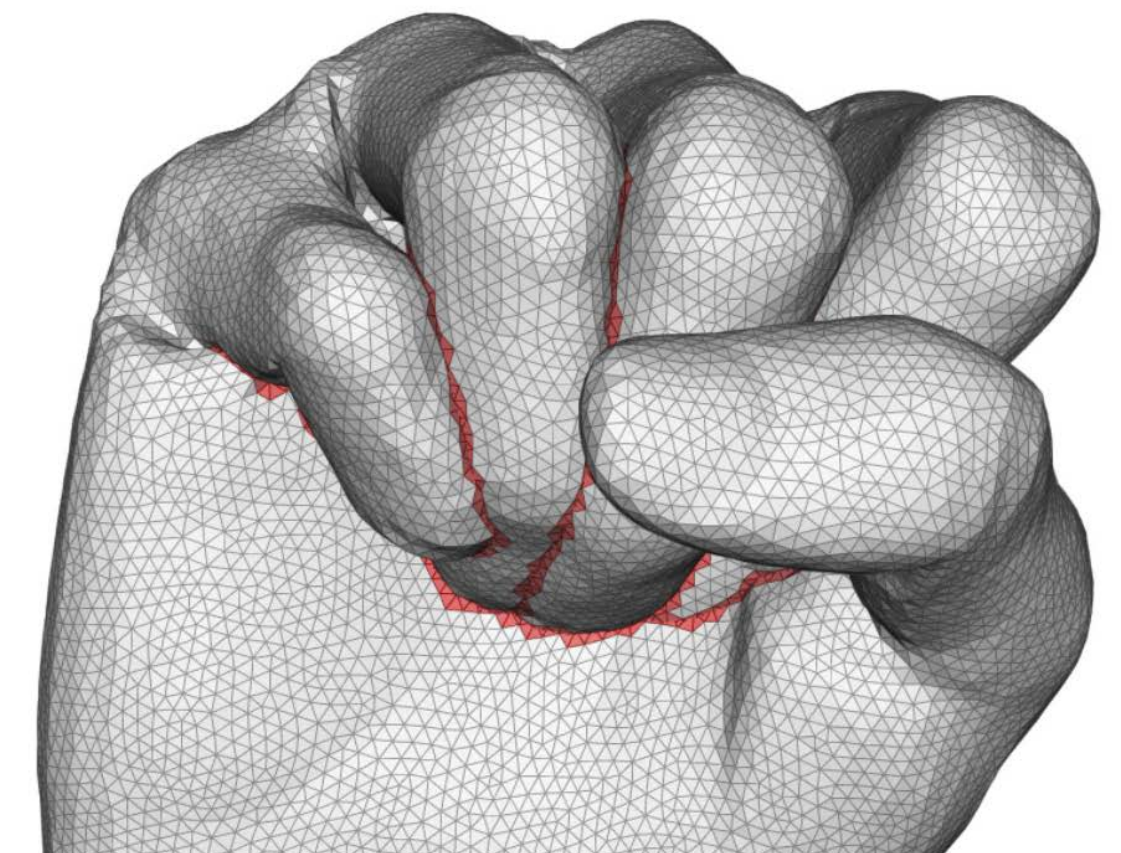
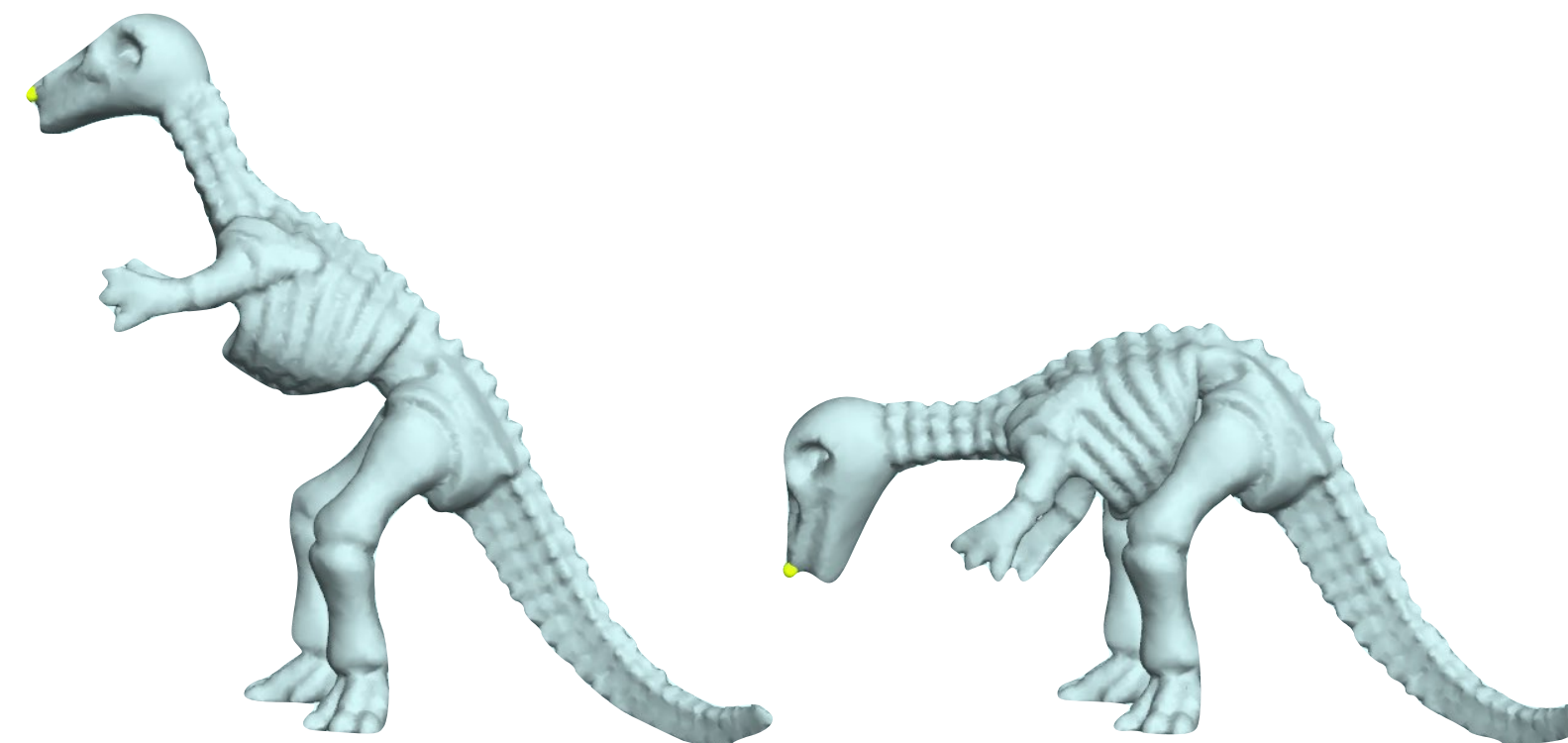
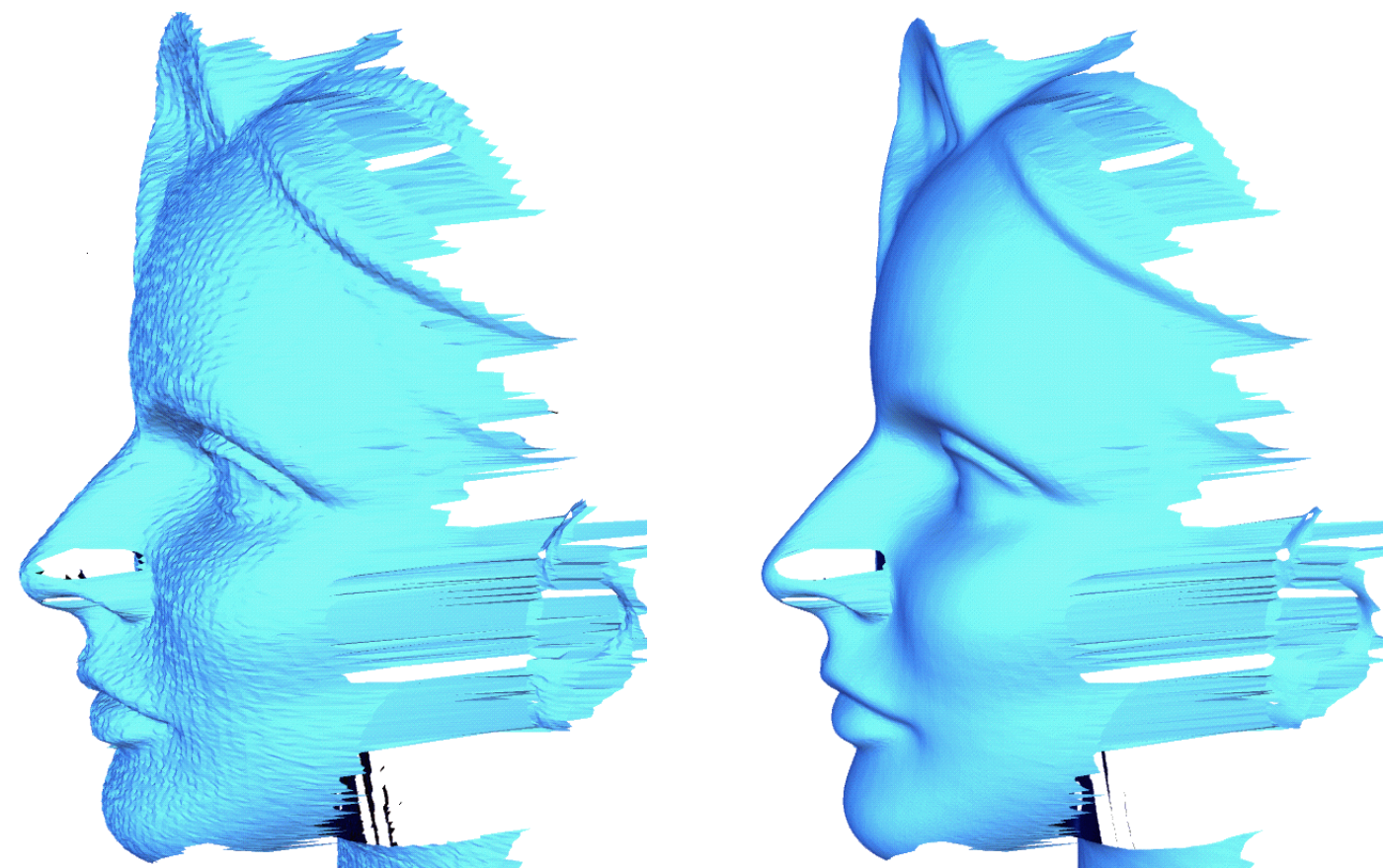
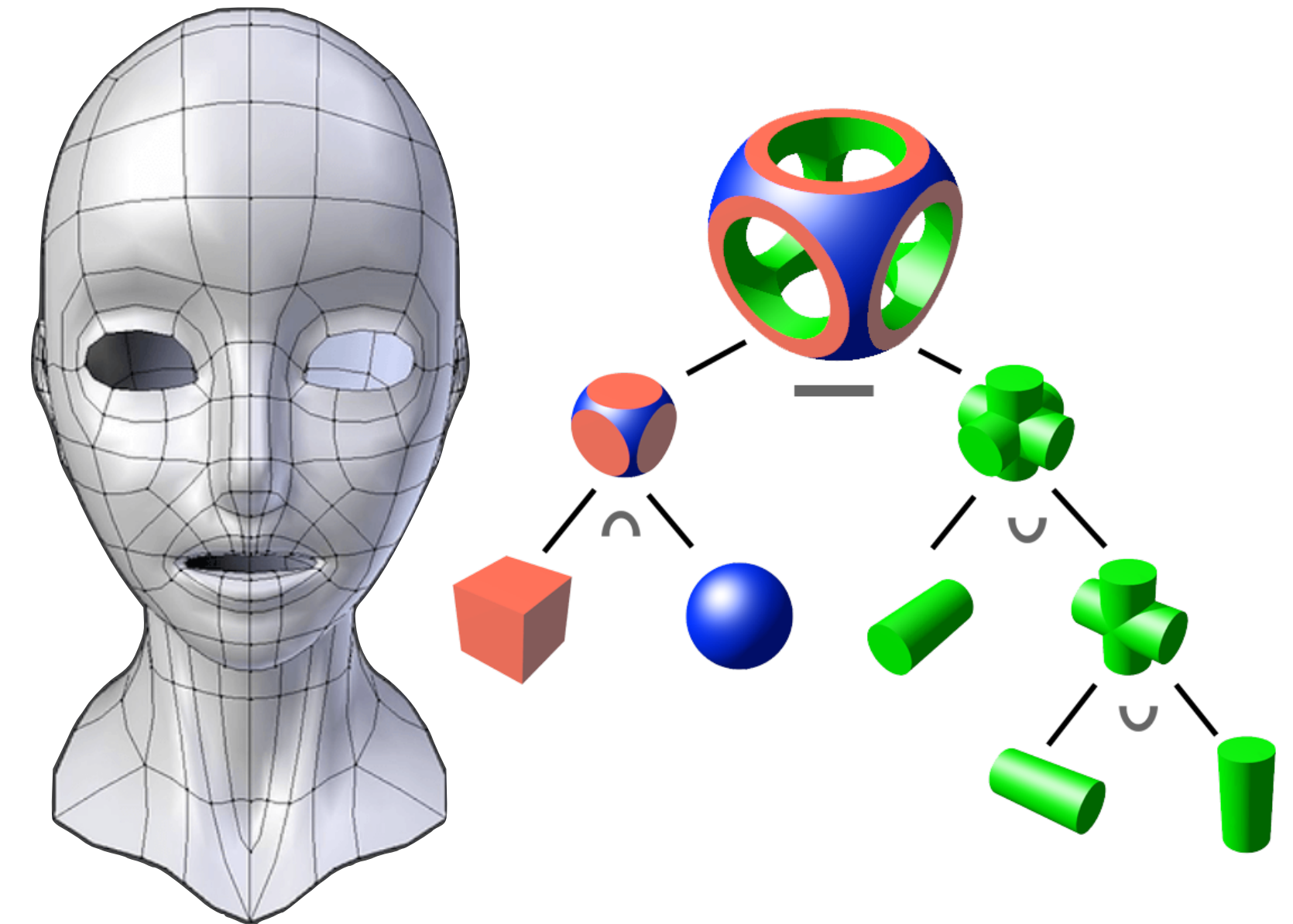
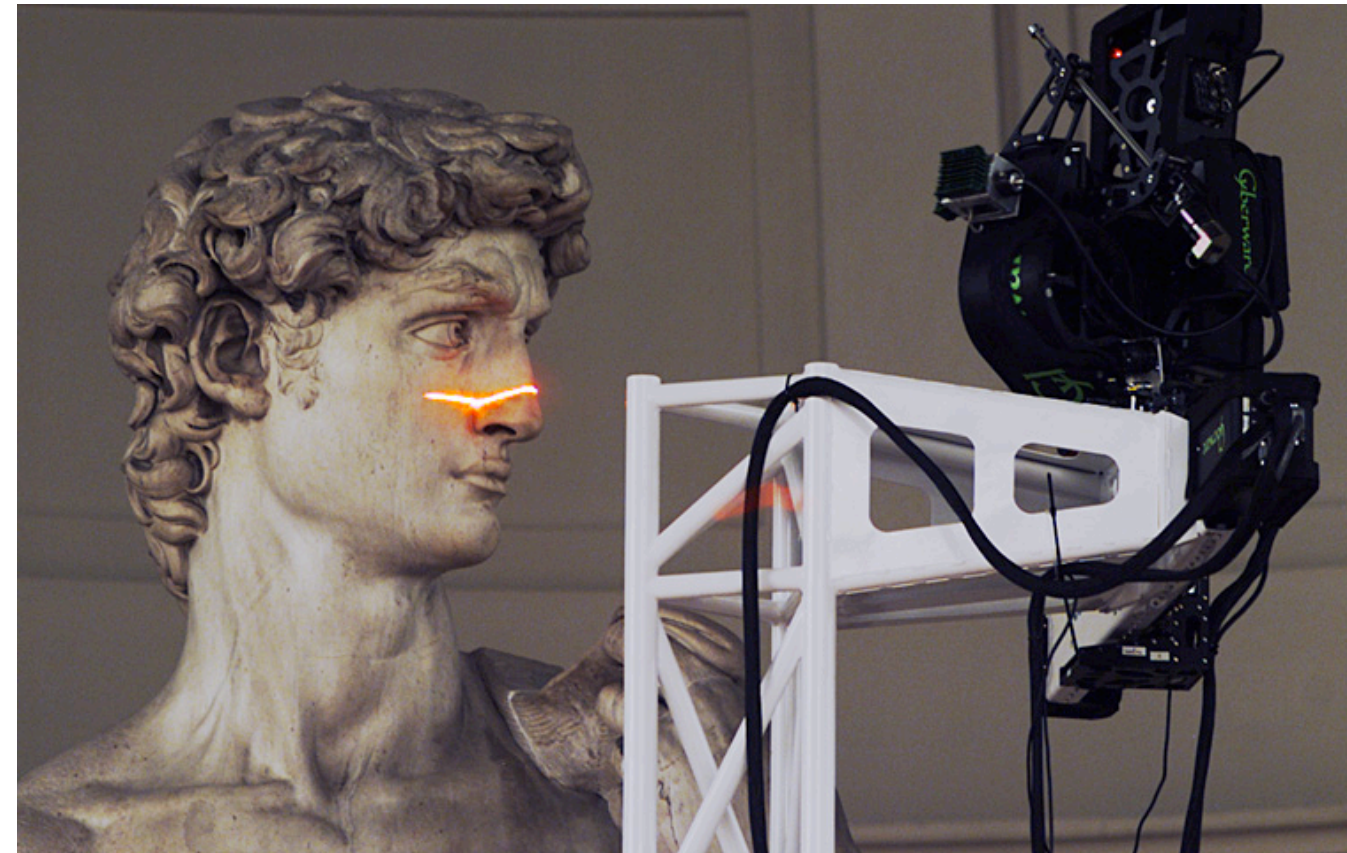
- Smooth volumes can be modeled by their bounding surfaces
- Many operations need only the boundary surface
 - ➔ Such surface is closed and have triangulation
 - ➔ Triangles should be non-intersection and have three adjacent triangles
 - ➔ Vertices have a ring of triangles around them
- Most operations also work with surfaces with boundary
 - ➔ Some triangles do have three neighbors
 - ➔ But the boundaries are closed curves
- "Thin" object: Surface without volume, Curve without area



Geometry Modeling



- Acquisition
- Ab-initio design
- Manipulation and editing

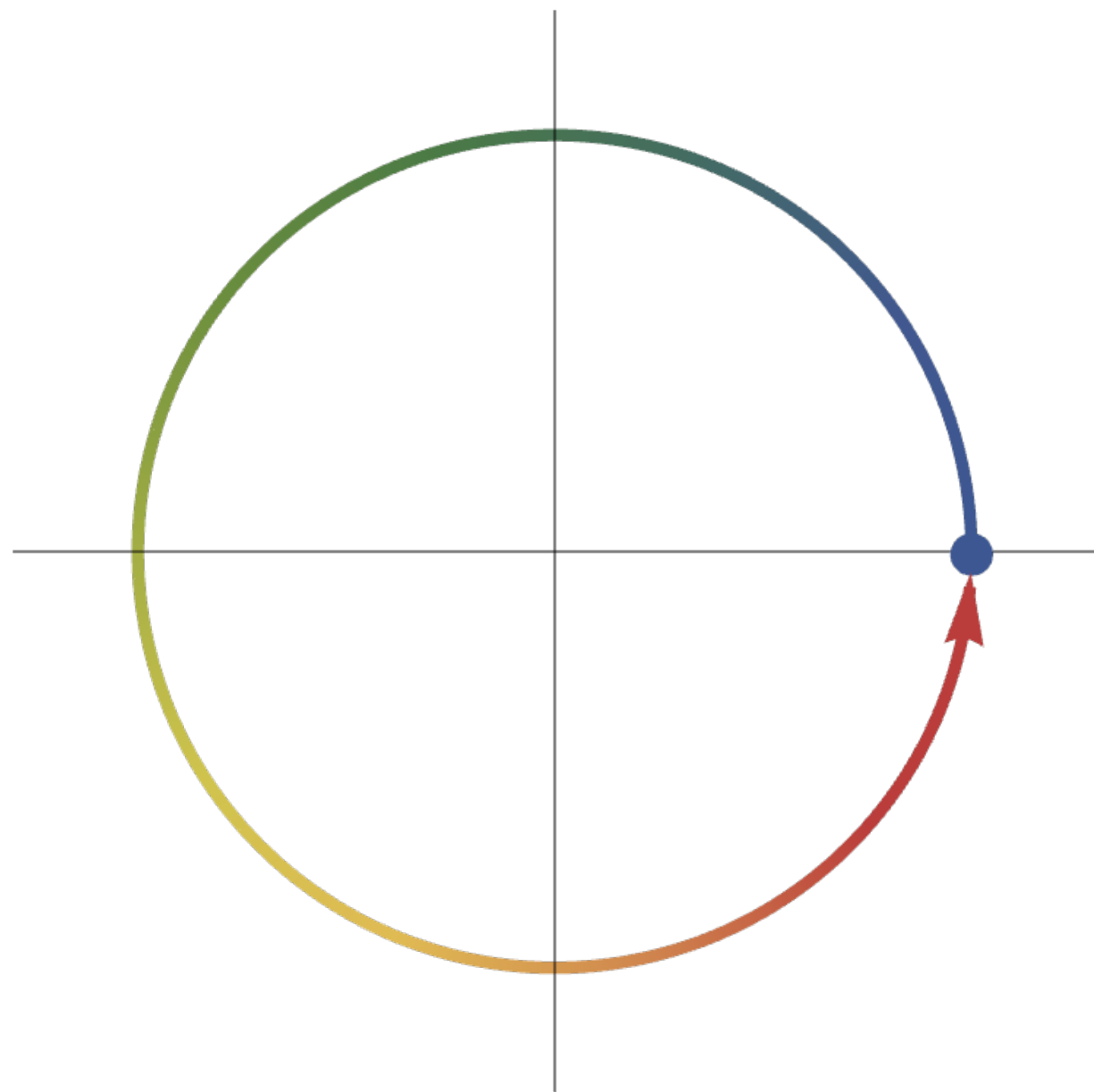




Define a unit circle (2D)

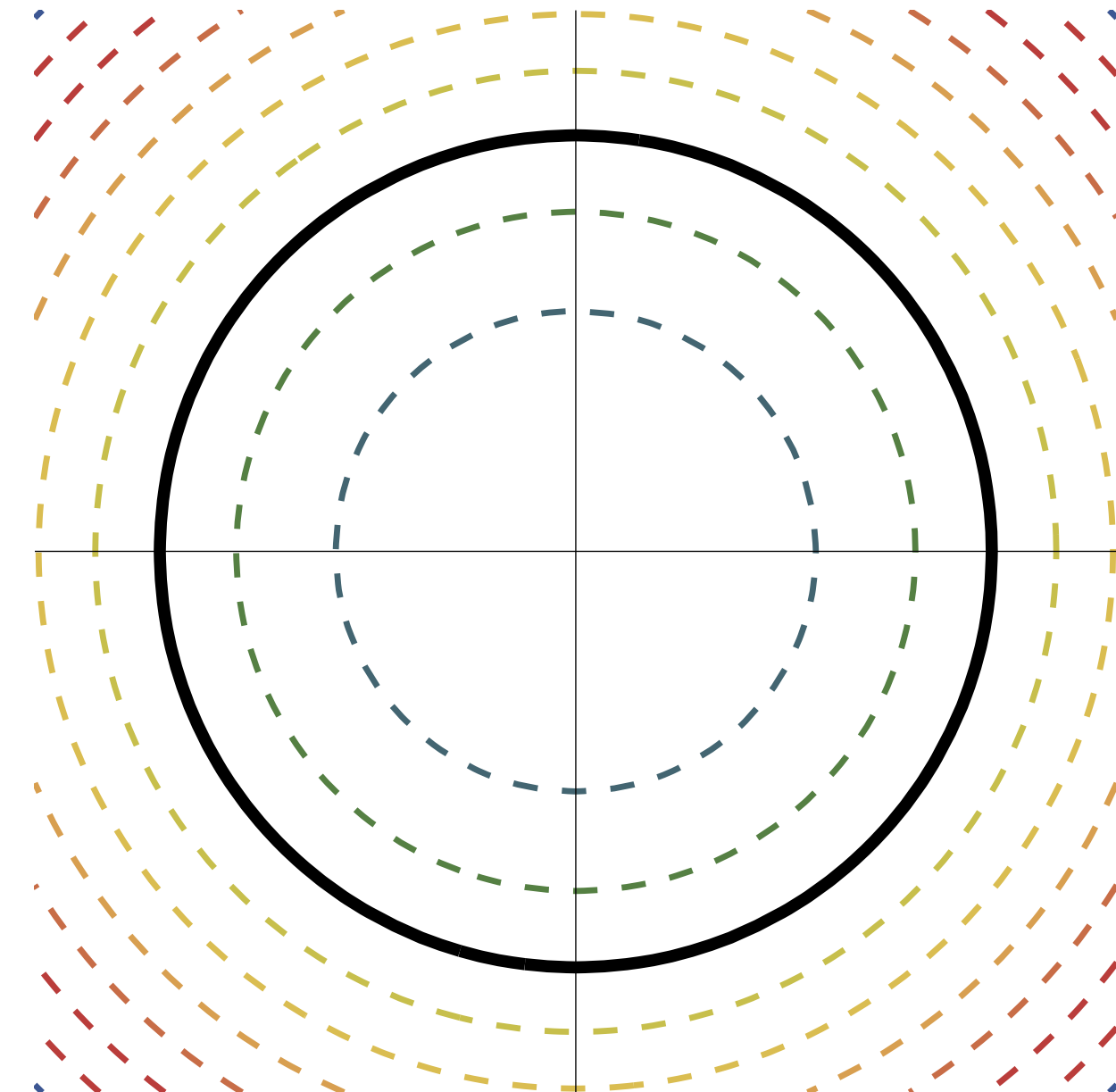
Explicit:

$$\{(\cos \theta, \sin \theta): 0 \leq \theta < 2\pi\}$$

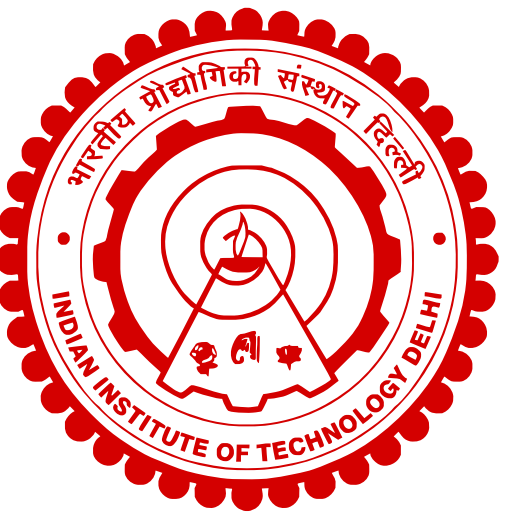


Implicit:

$$\{(x, y): x^2 + y^2 - 1 = 0\}$$

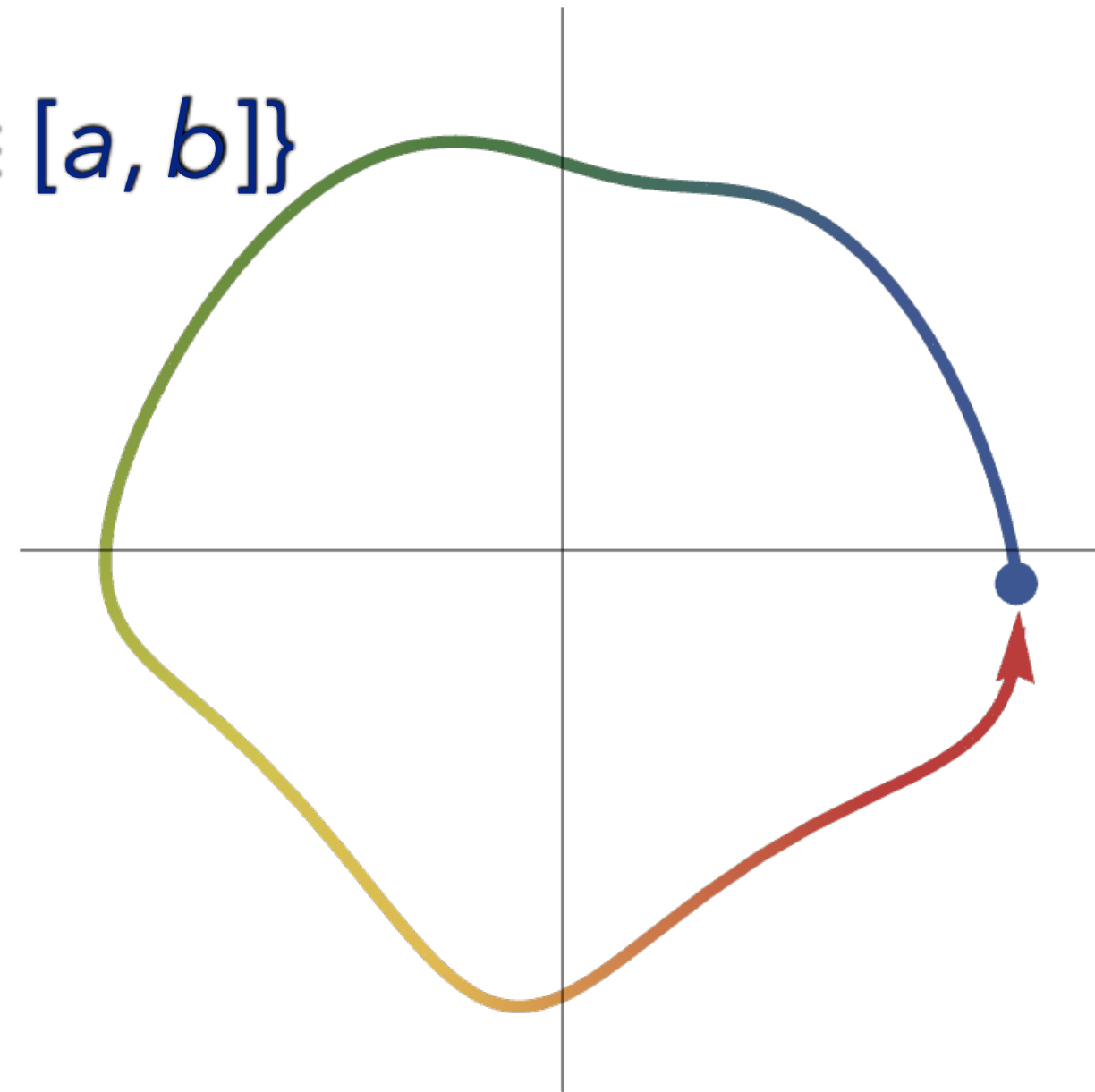


Explicit and Implicit Representations



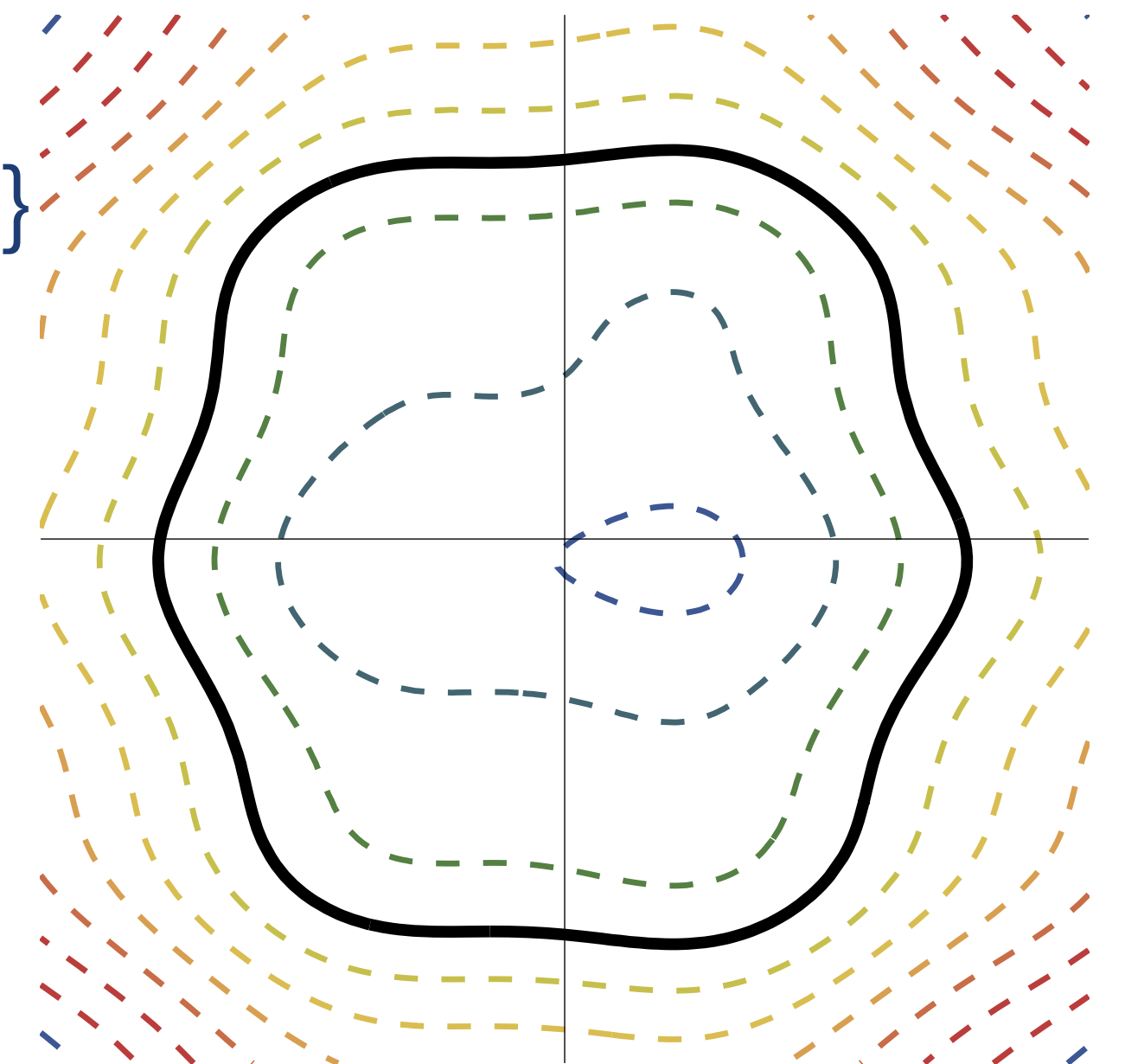
Parametric curve:

$$\rightarrow \{(x(t), y(t)): t \in [a, b]\}$$



Implicit curve:

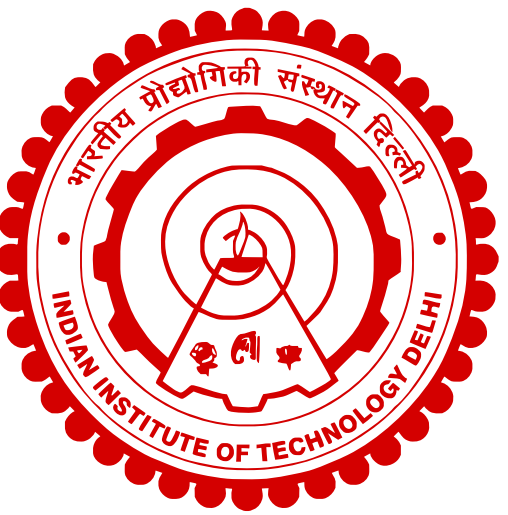
$$\rightarrow \{(x, y): f(x, y) = 0\}$$



Easier to **generate** points on the curve?

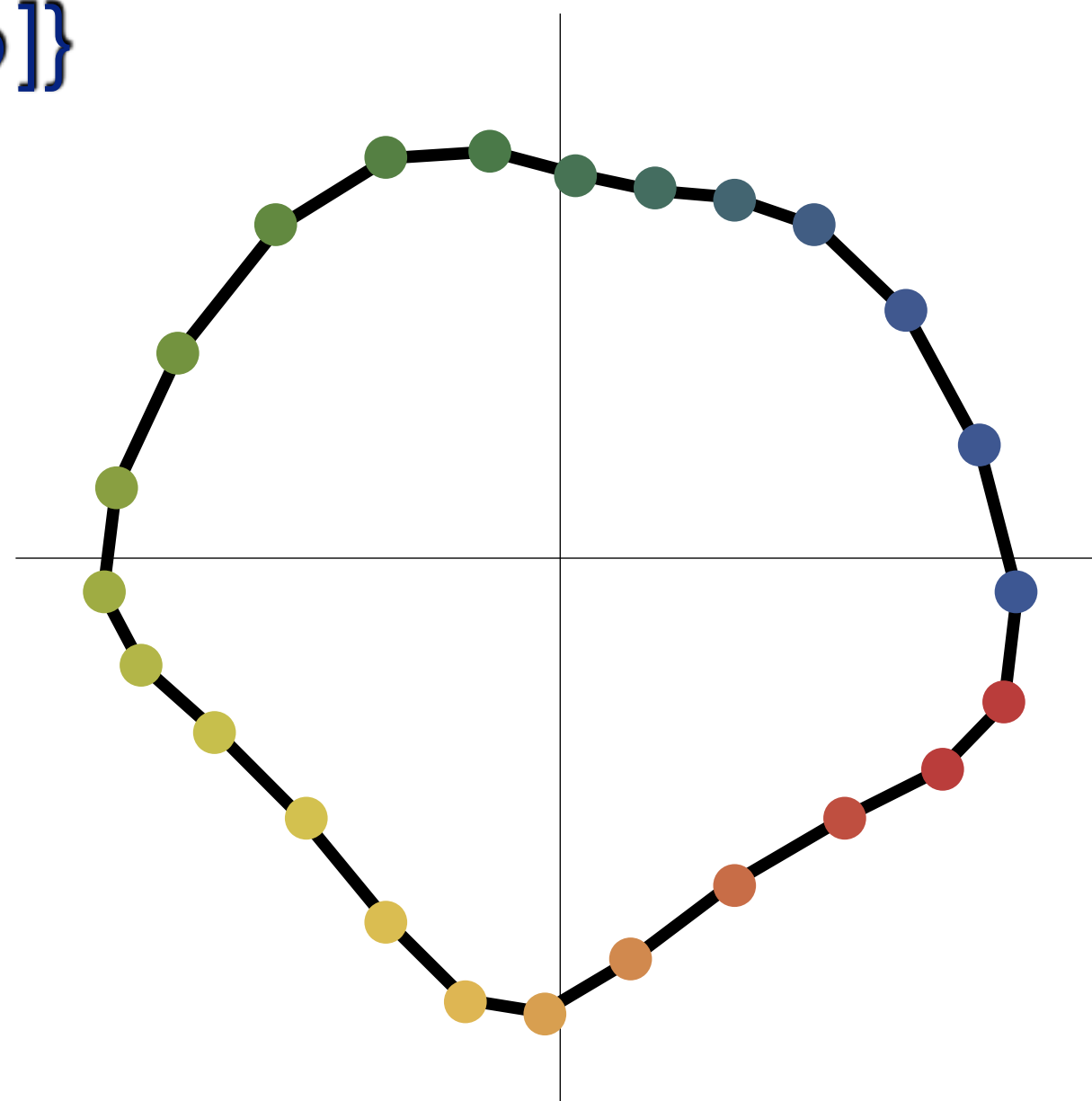
Easier to **test** if a given point lies on the curve?

Explicit and Implicit Representations

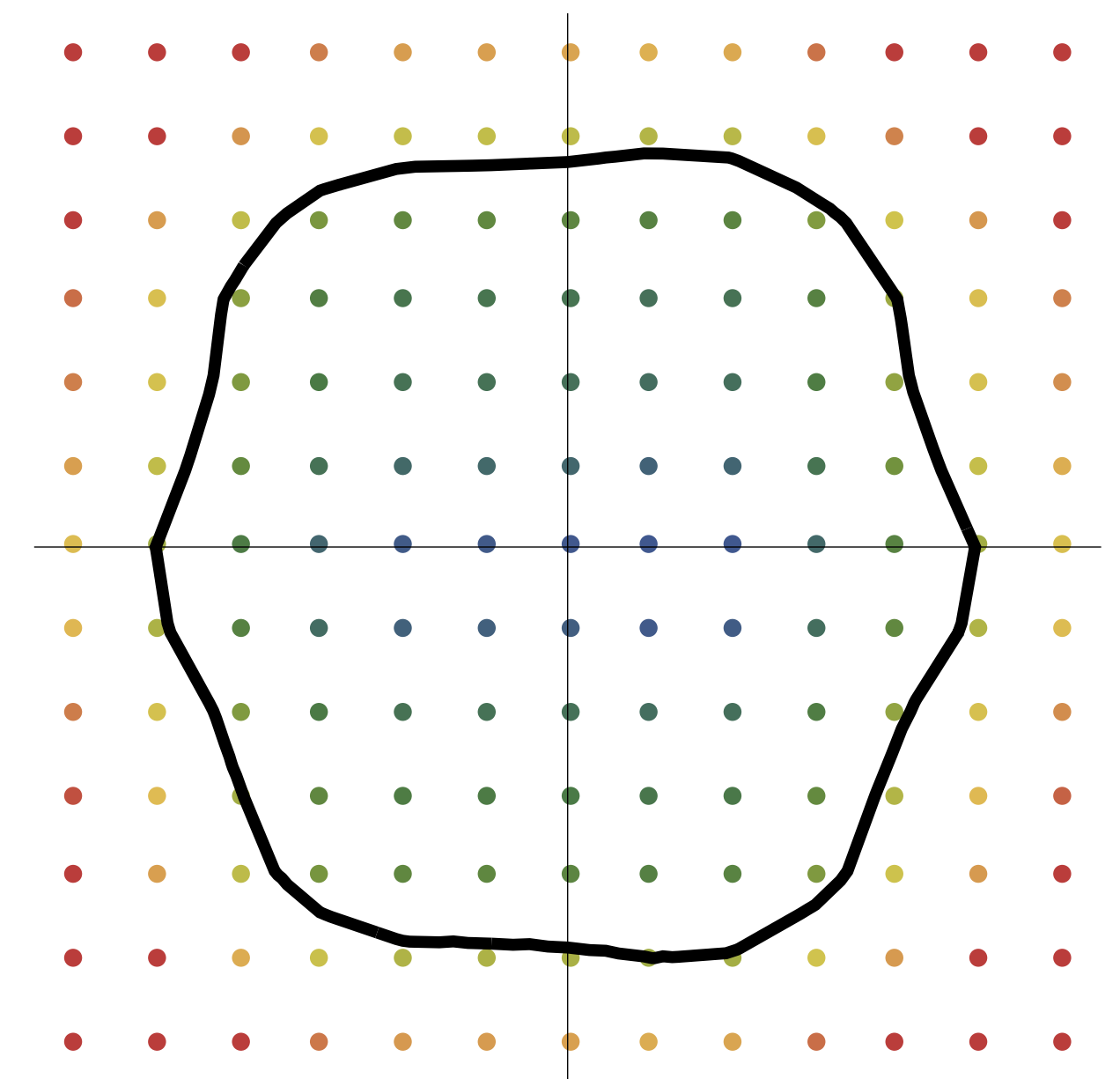


How to draw a curve given in one of these forms?

$$\{(x(t), y(t)): t \in [a, b]\}$$



$$\{(x, y): f(x, y) = 0\}$$



- Sample points at various values of t
- Connect by polyline

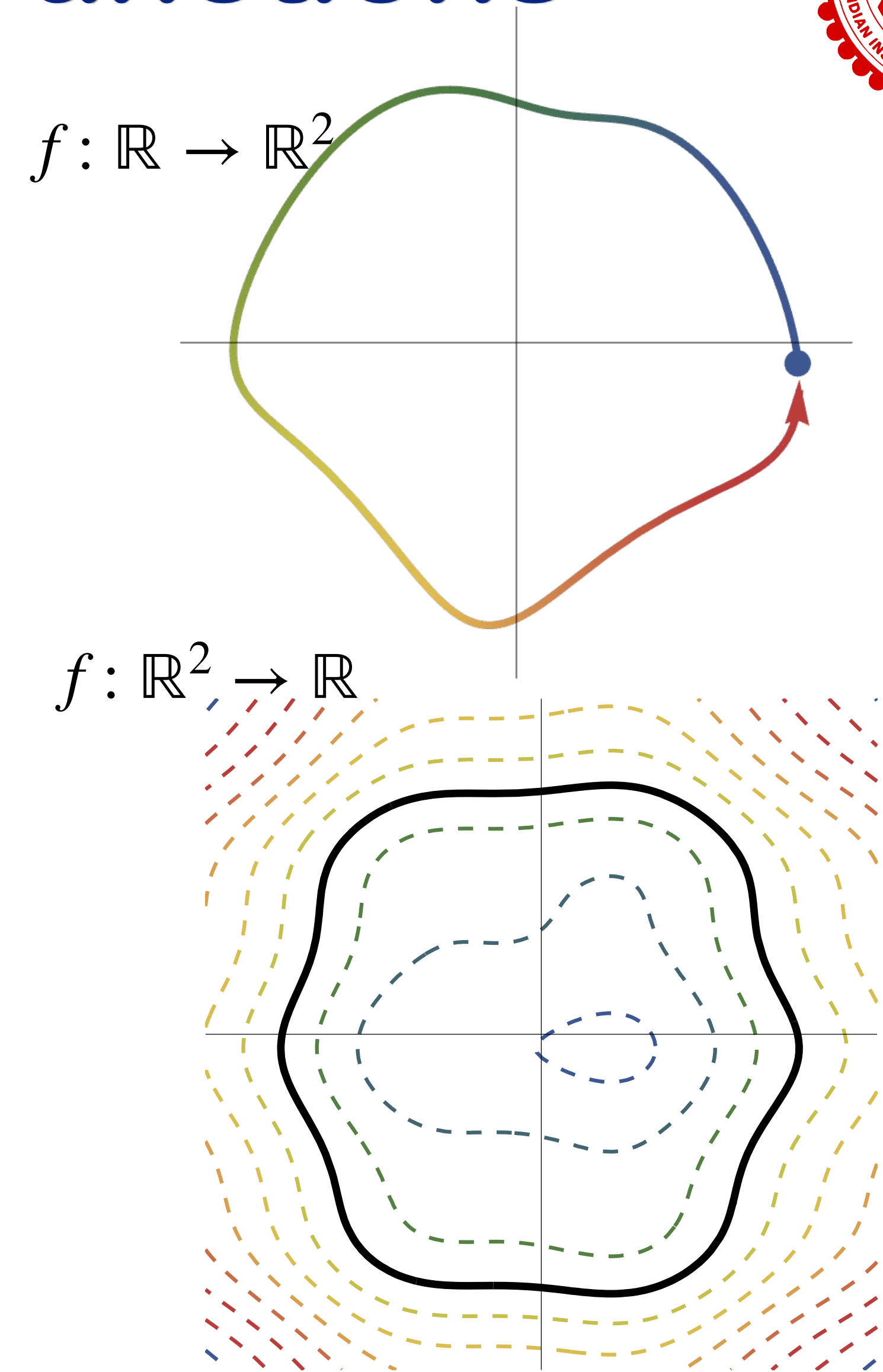
- Sample f at various points (x, y)
- Draw boundary between + and - points
➔ Interpolation

Parametric/Implicit Functions



Both methods require an arbitrary function f

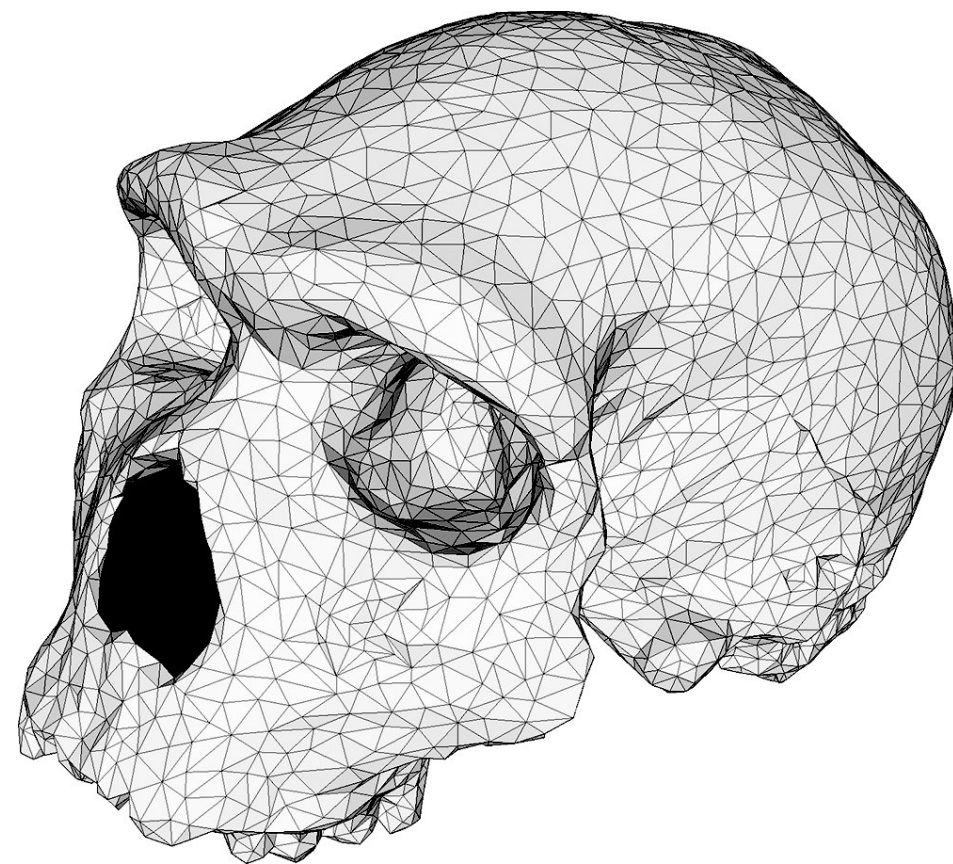
- Analytical
- Basis function expansion
 - ➔ e.g. polynomials, Fourier series, ...
 - ➔ Piecewise polynomials
- Sampling and interpolation/approximation
(often can also be expressed via basis functions)
- Neural networks



3D Geometry Representation

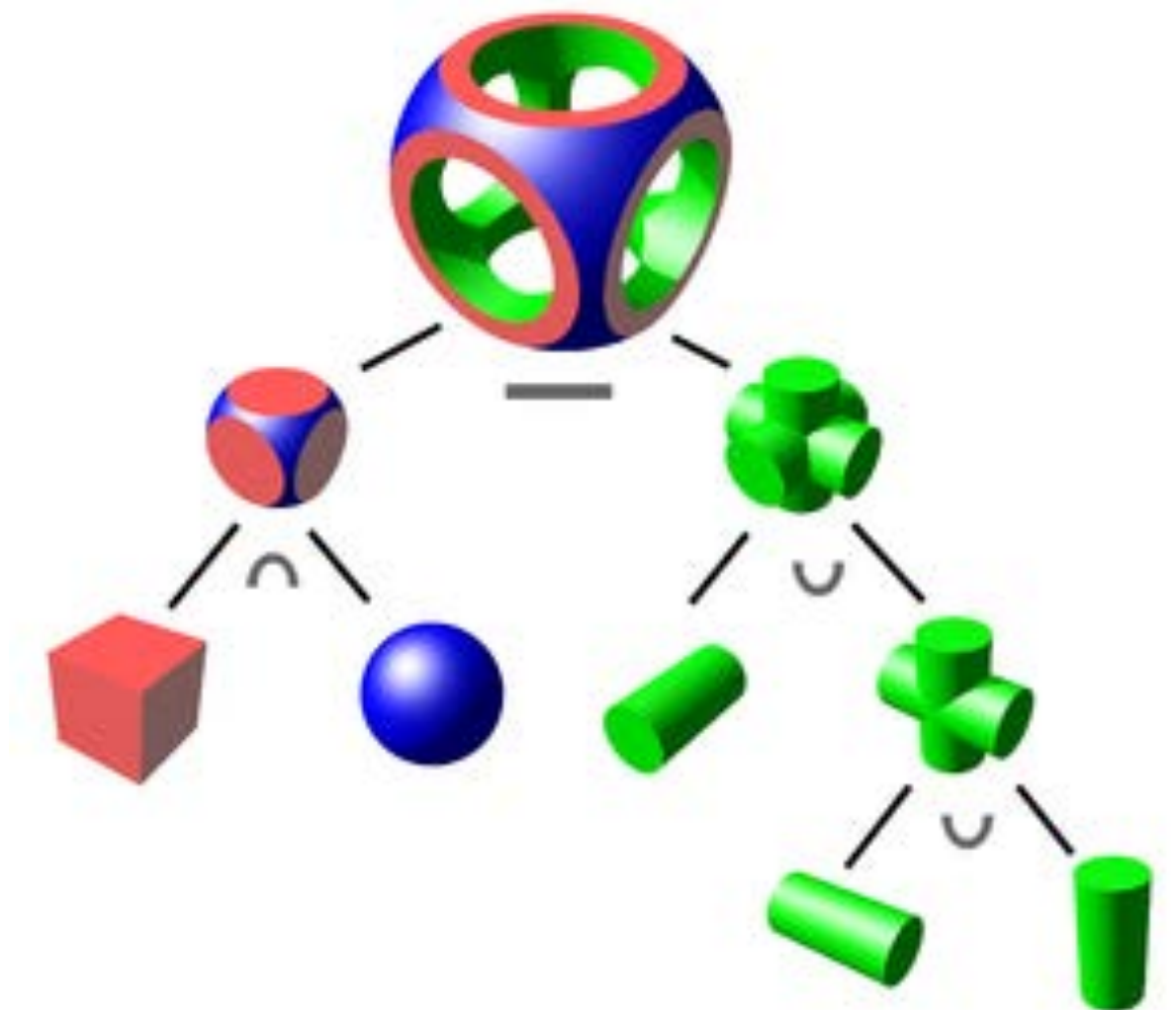
Explicit:

- Polygon meshes
- Parametric curves and surfaces
- Subdivision surfaces
- Point clouds



Implicit:

- Algebraic surfaces, distance fields
- Constructive solid geometry
- "Blobby" surfaces
- Level sets



Implicit surfaces

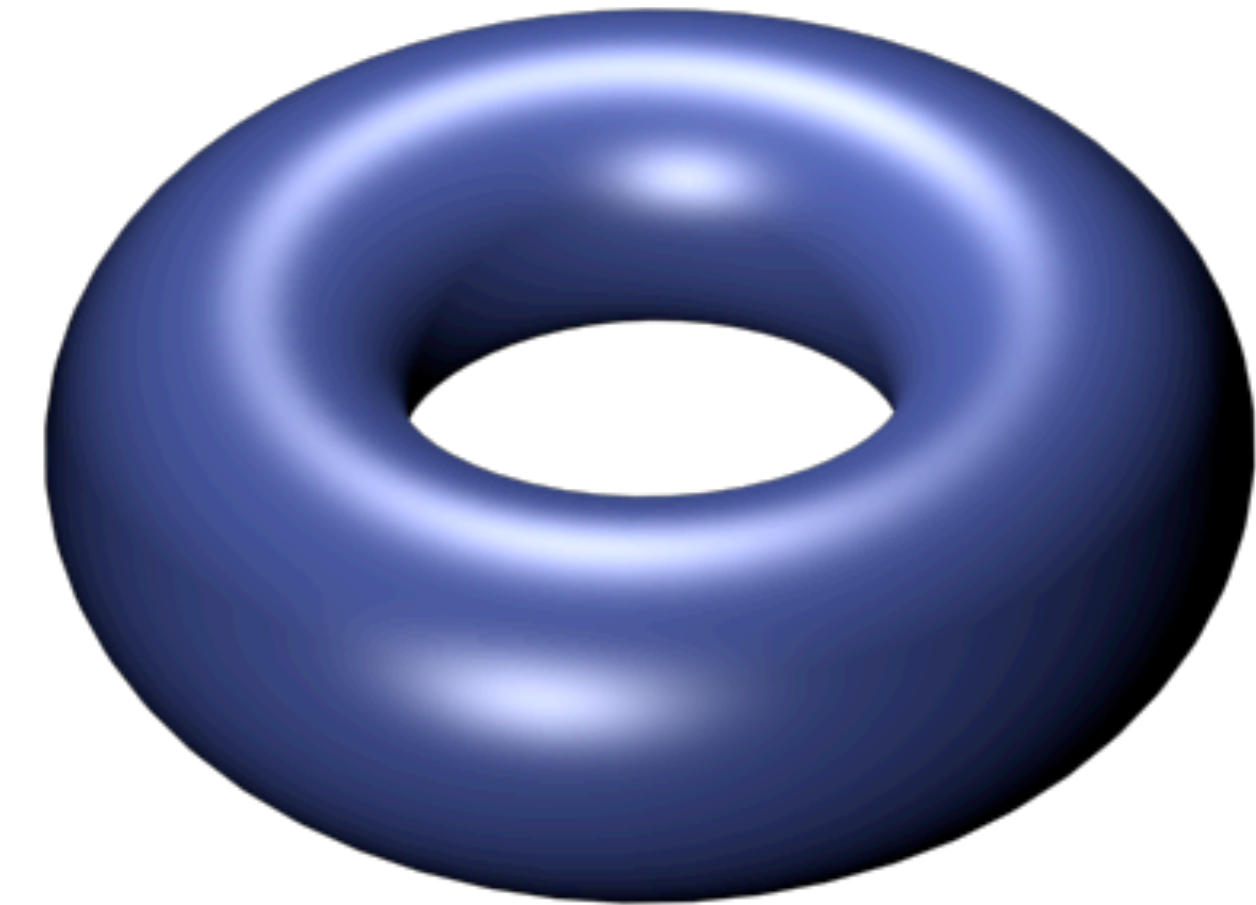
- Defined as the zero level set of a given function

$$S = \{(x, y, z) : f(x, y, z) = 0\}$$

- Algebraic surface: f is a polynomial
- Signed distance field:

$$f(p) = \begin{cases} \text{dist}(\mathbf{p}, S) & \text{if } \mathbf{p} \text{ is outside } S, \\ -\text{dist}(\mathbf{p}, S) & \text{if } \mathbf{p} \text{ is inside.} \end{cases}$$

- Simple formulas only exist in very special cases...



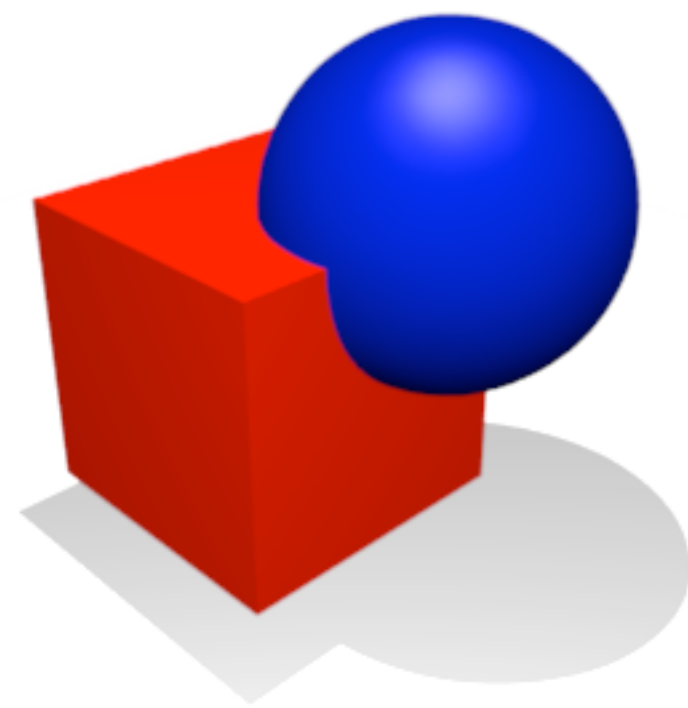
$$(x^2 + y^2 + z^2 + R^2 - r^2)^2 - 4R^2(x^2 + y^2) = 0$$



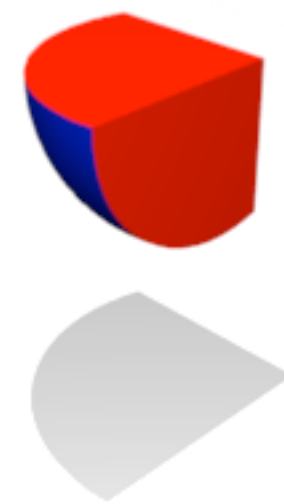
Constructive Solid Geometry

- An implicit representation defines both a surface, $f(\mathbf{p}) = 0$, and its enclosed volume, $f(\mathbf{p}) \leq 0$.

So we can do set operations on the volume



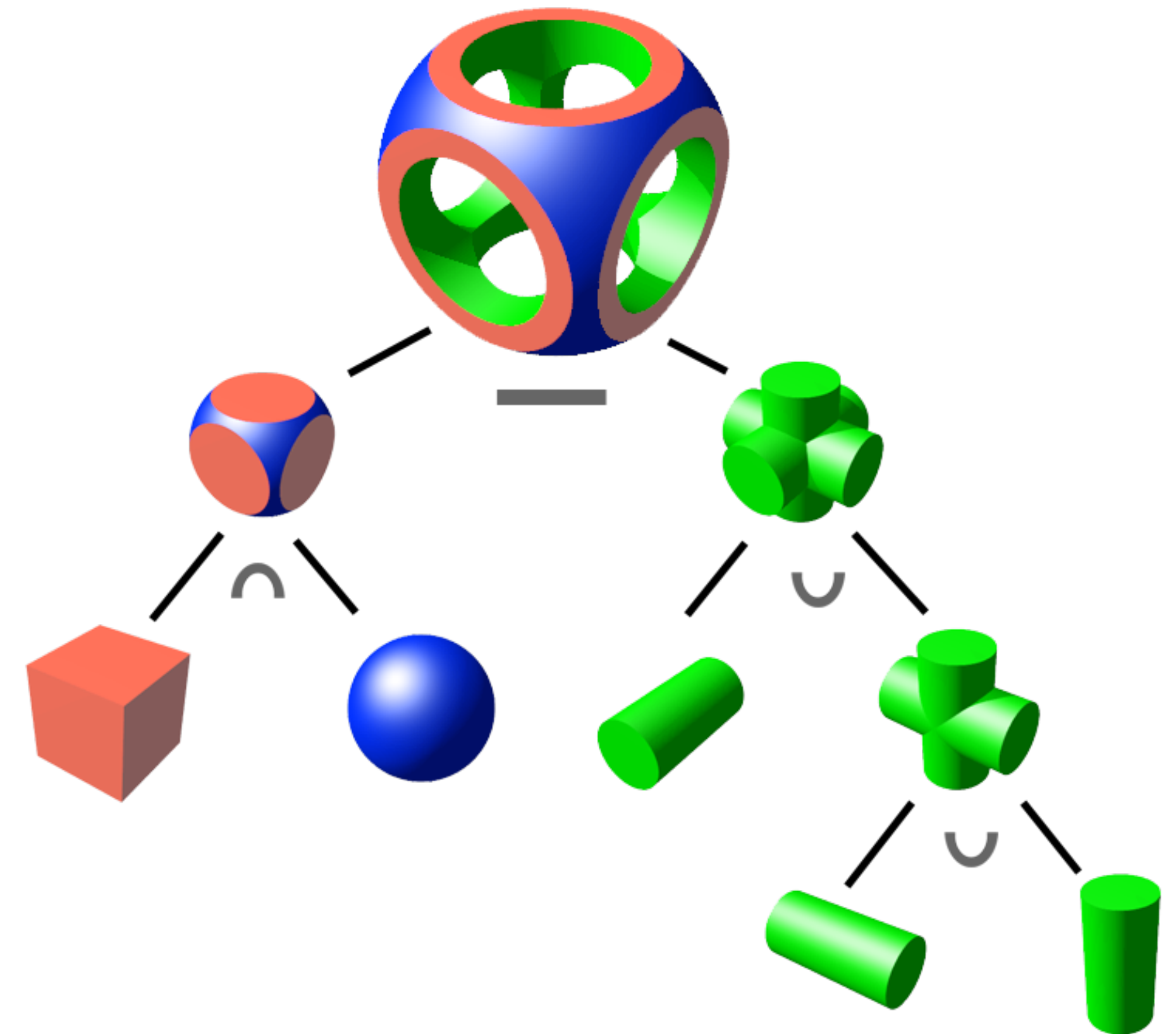
Union



Intersection



Difference



Smooth implicit modeling



Instead of a Boolean operation, blend together the implicit functions of two surfaces.

e.g. $f_i(\mathbf{p}) = \exp(-\|\mathbf{p} - \mathbf{c}_i\|^2 / r_i^2)$

$$S = \{\mathbf{p} : \sum f_i(\mathbf{p}) = 0.5\}$$

a.k.a. metaballs, blobbies, soft objects, ...

Choice of blending operation can give useful effects:



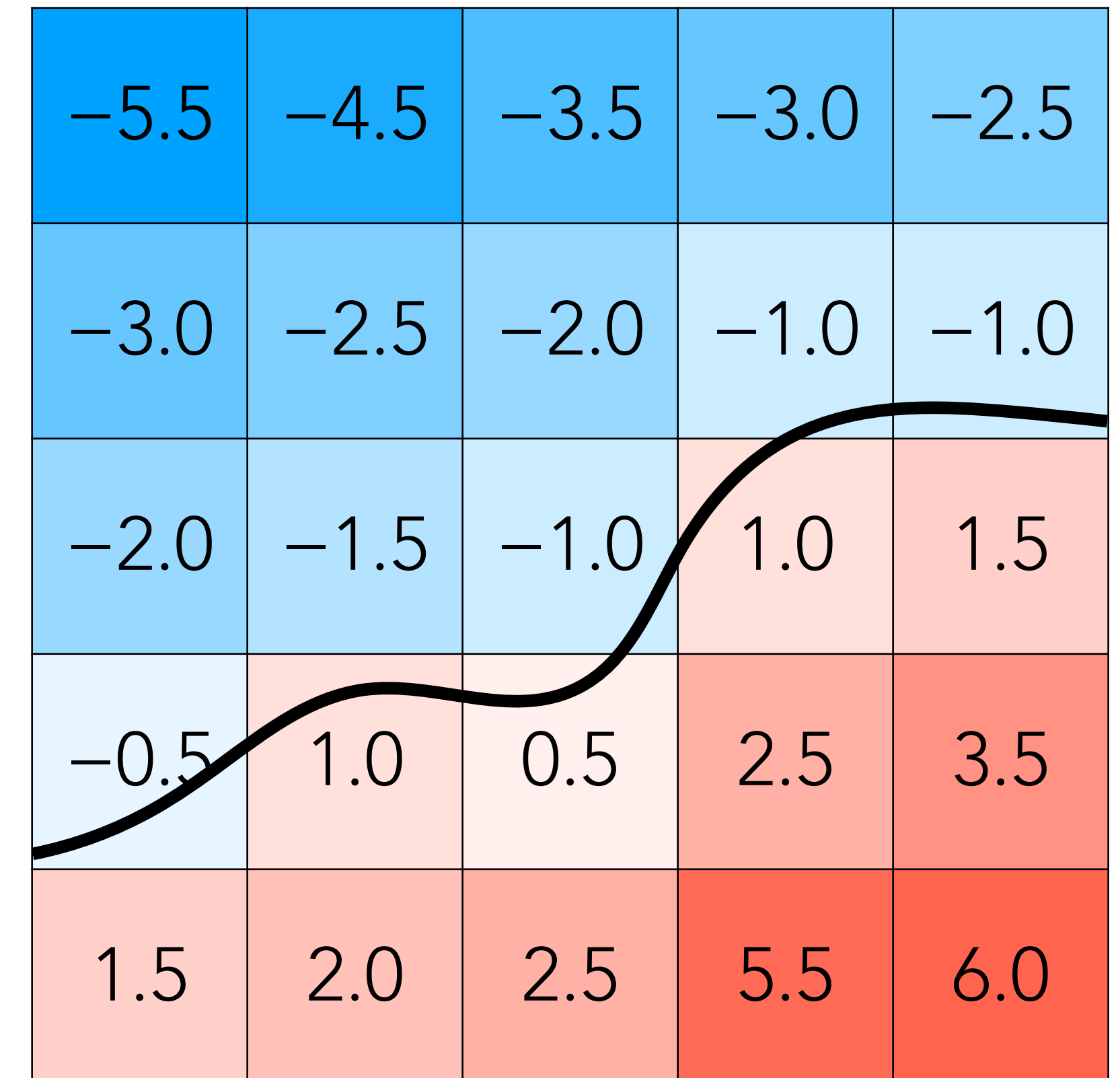
Angles et al. 2017



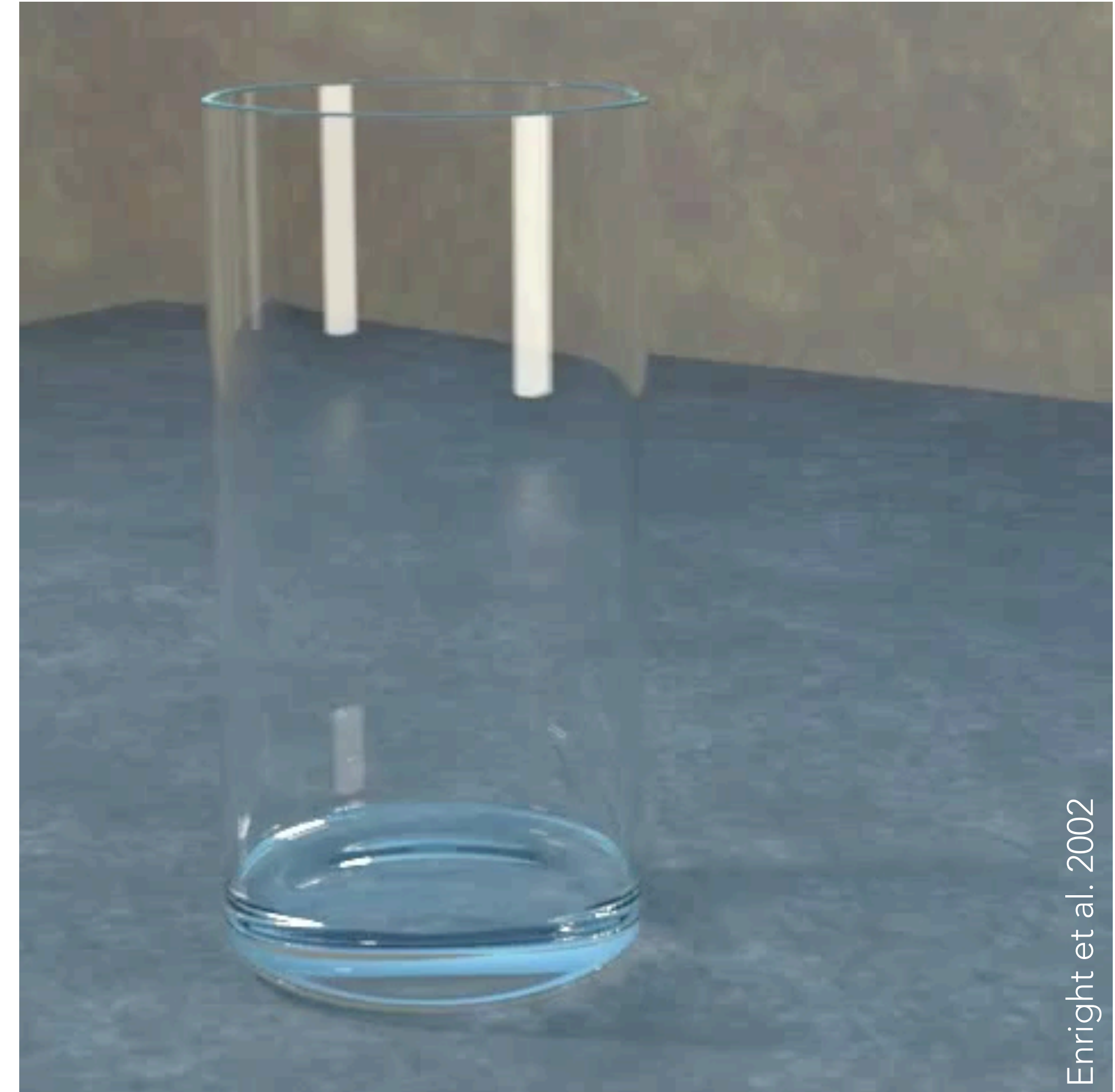
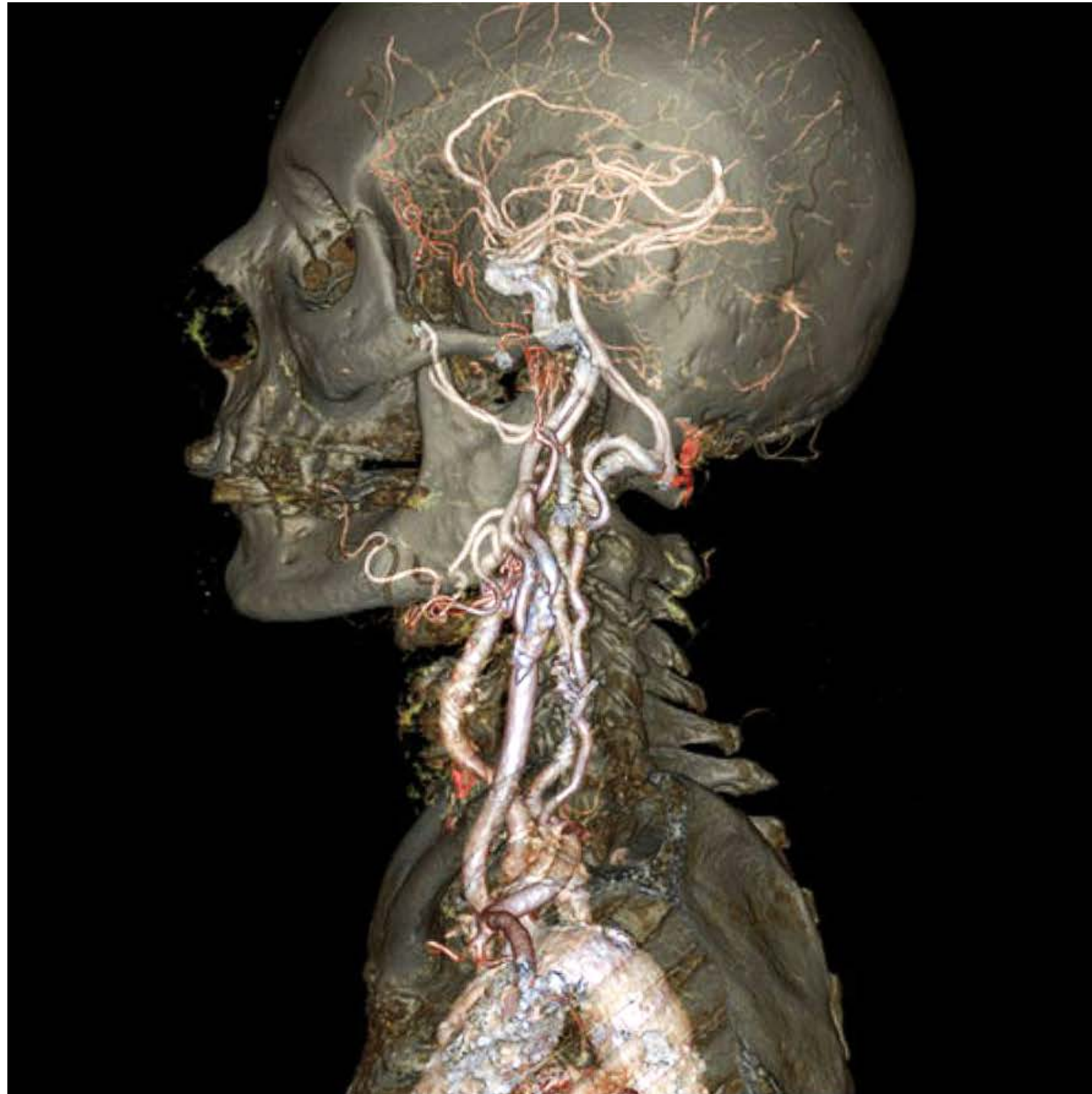
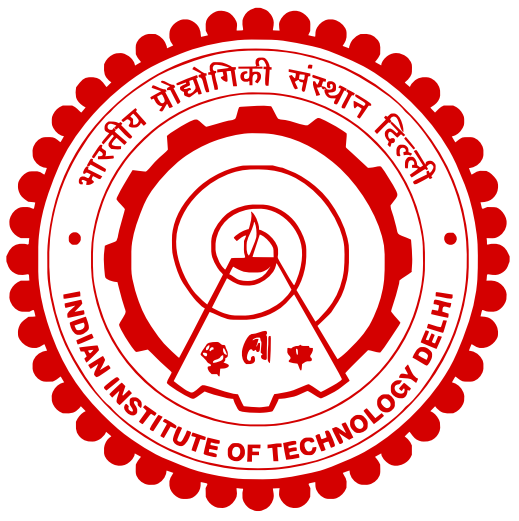
Level sets on grids

- Implicit representations are useful for changing topology (merging / splitting), but usually no closed form for $f(x, y, z)$
 - ➔ Just store sampled values on a grid!
- Surface is wherever **interpolated** value is 0
- Modify surface by changing values on the grid

See: Marching Cubes algorithm

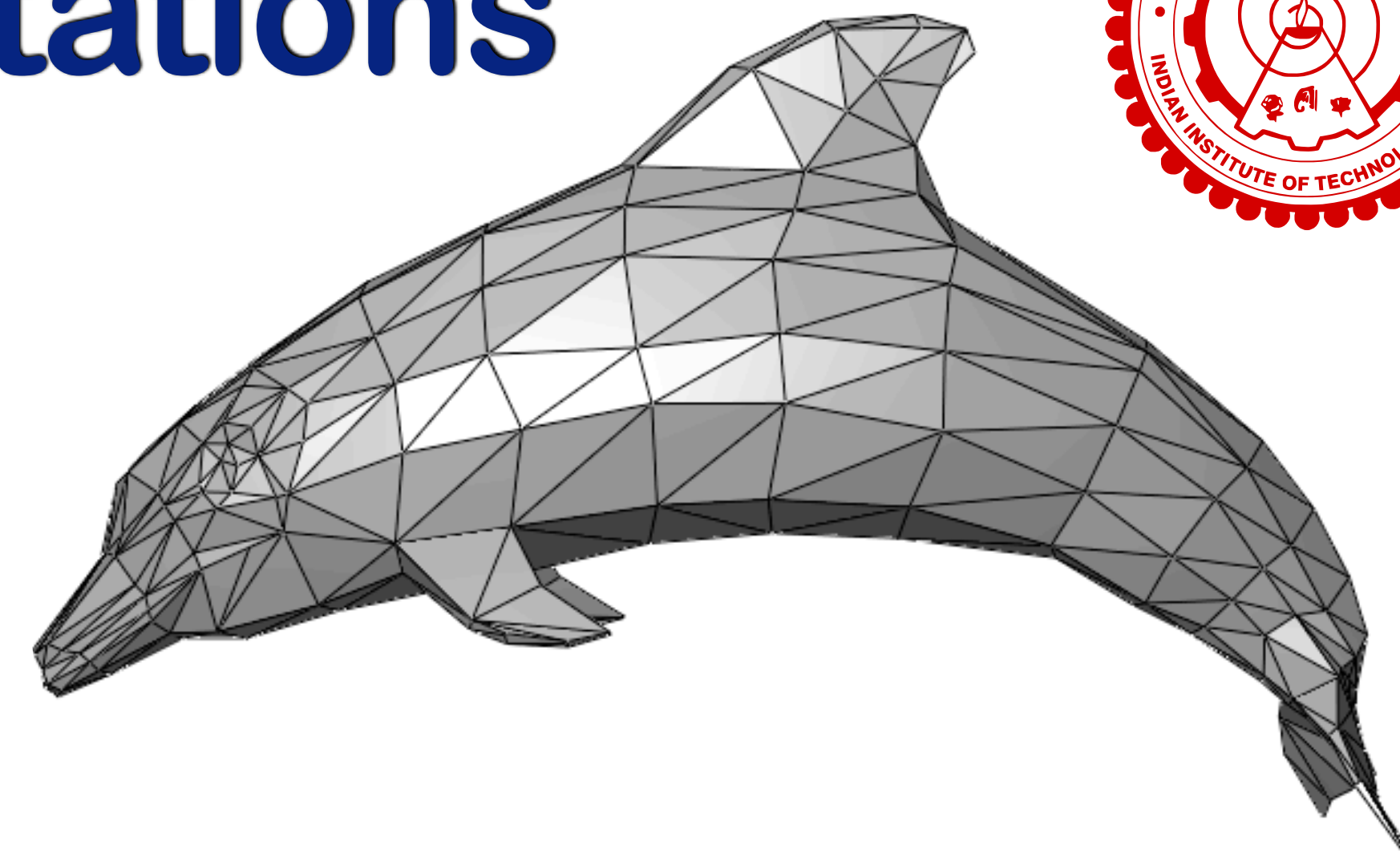
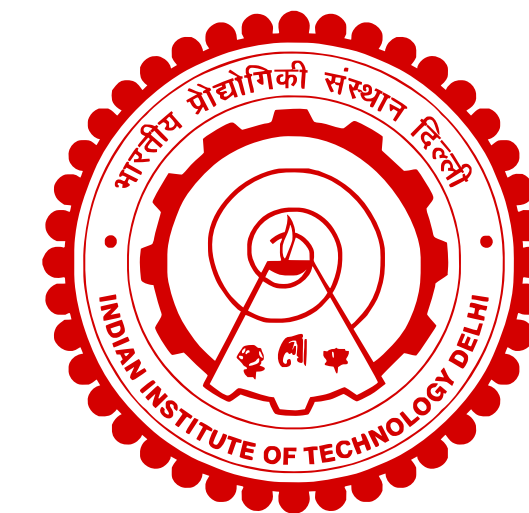


Level sets on grids



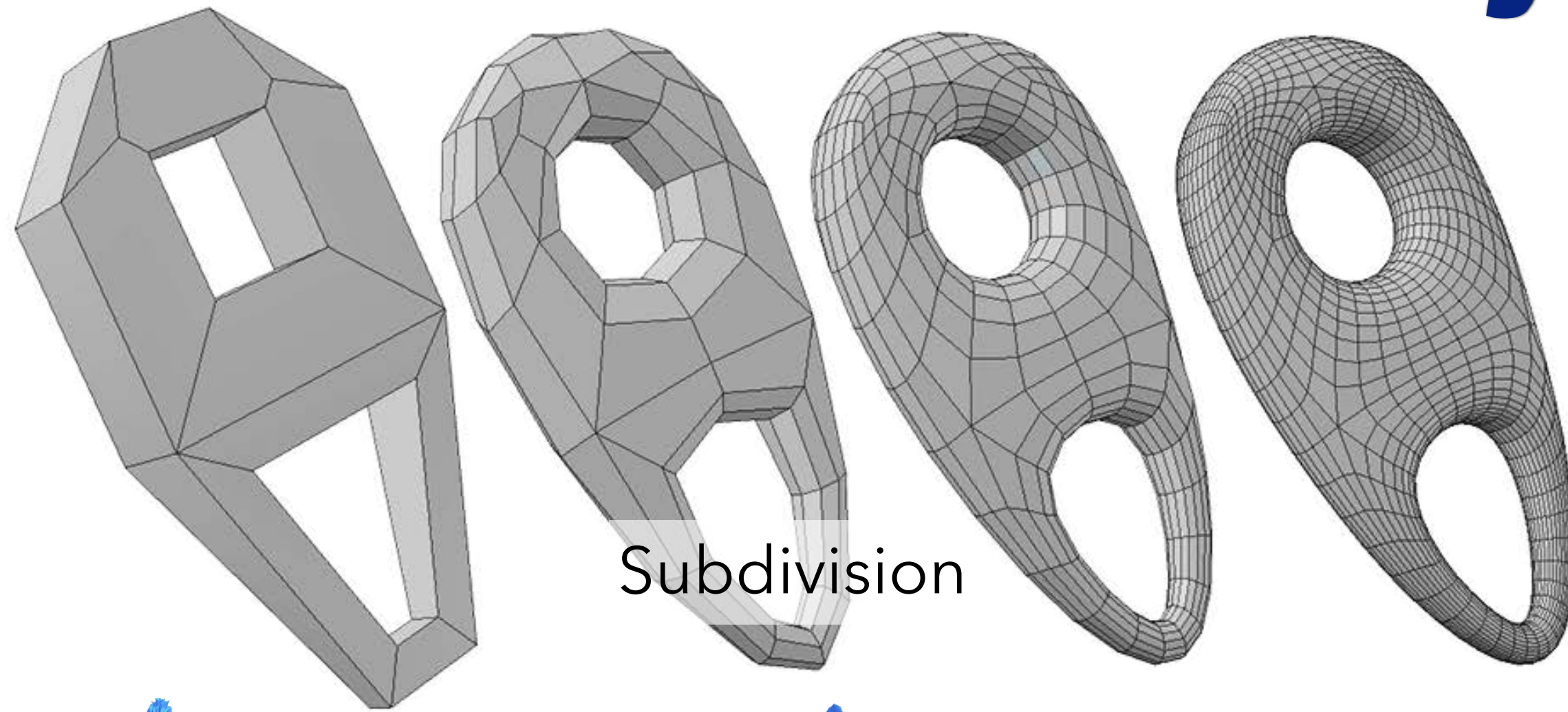
Enright et al. 2002

Explicit Representations

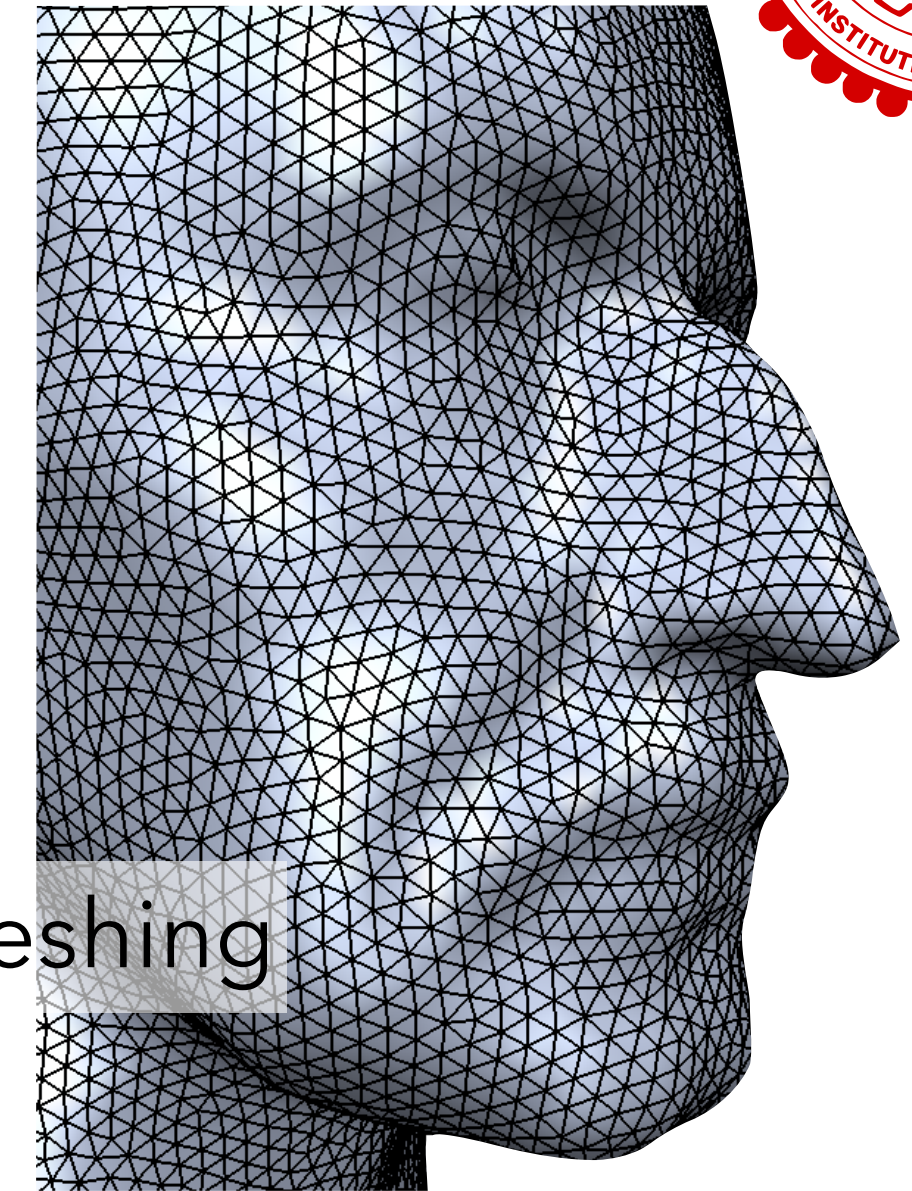
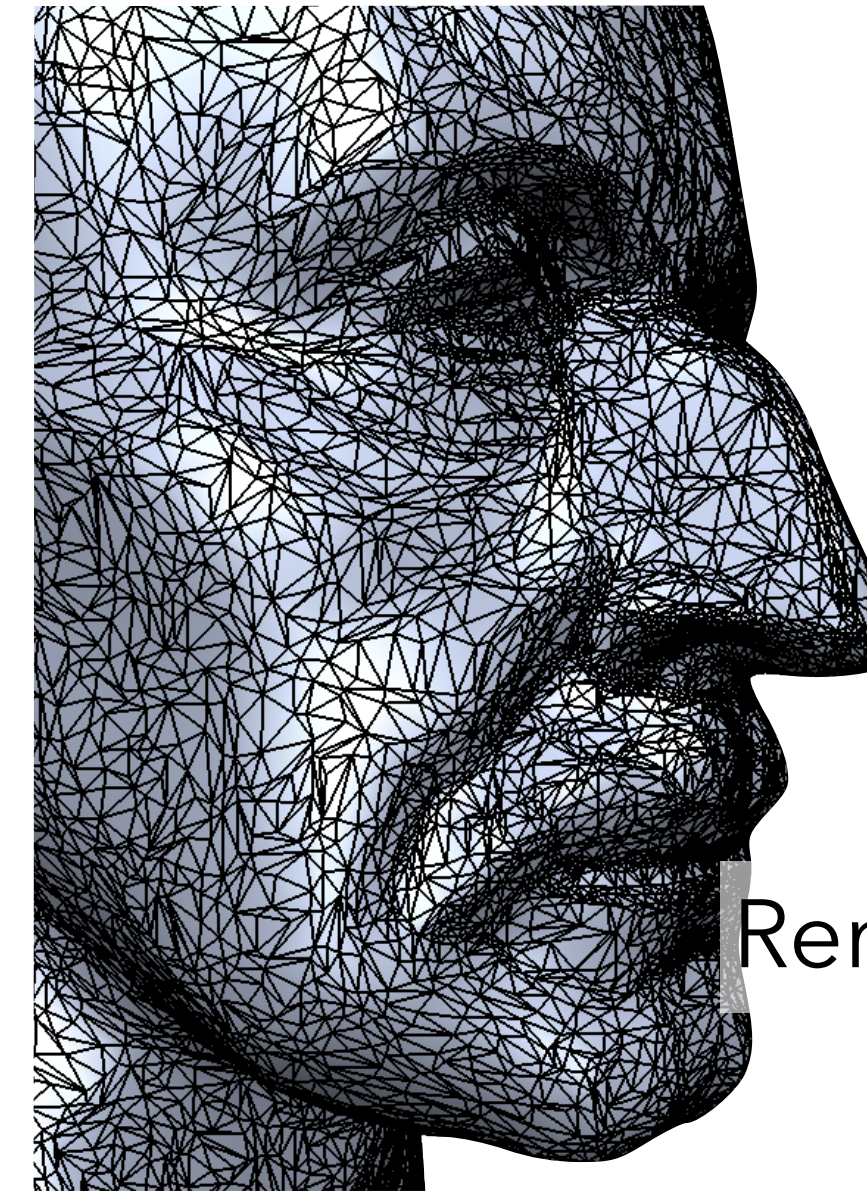


- We've already seen polygons
 - ➔ Efficient to render
 - ➔ Simple to edit and refine
- Vertices $(x, y, z) \in \mathbb{R}^3$
- Triangles stored via vertex indices $(i, j, k) \in \mathbb{N}^3$
 - ➔ Can also allow arbitrary polygons $(i_1, i_2, i_3, \dots)$
- How to determine if a point lies on the surface?

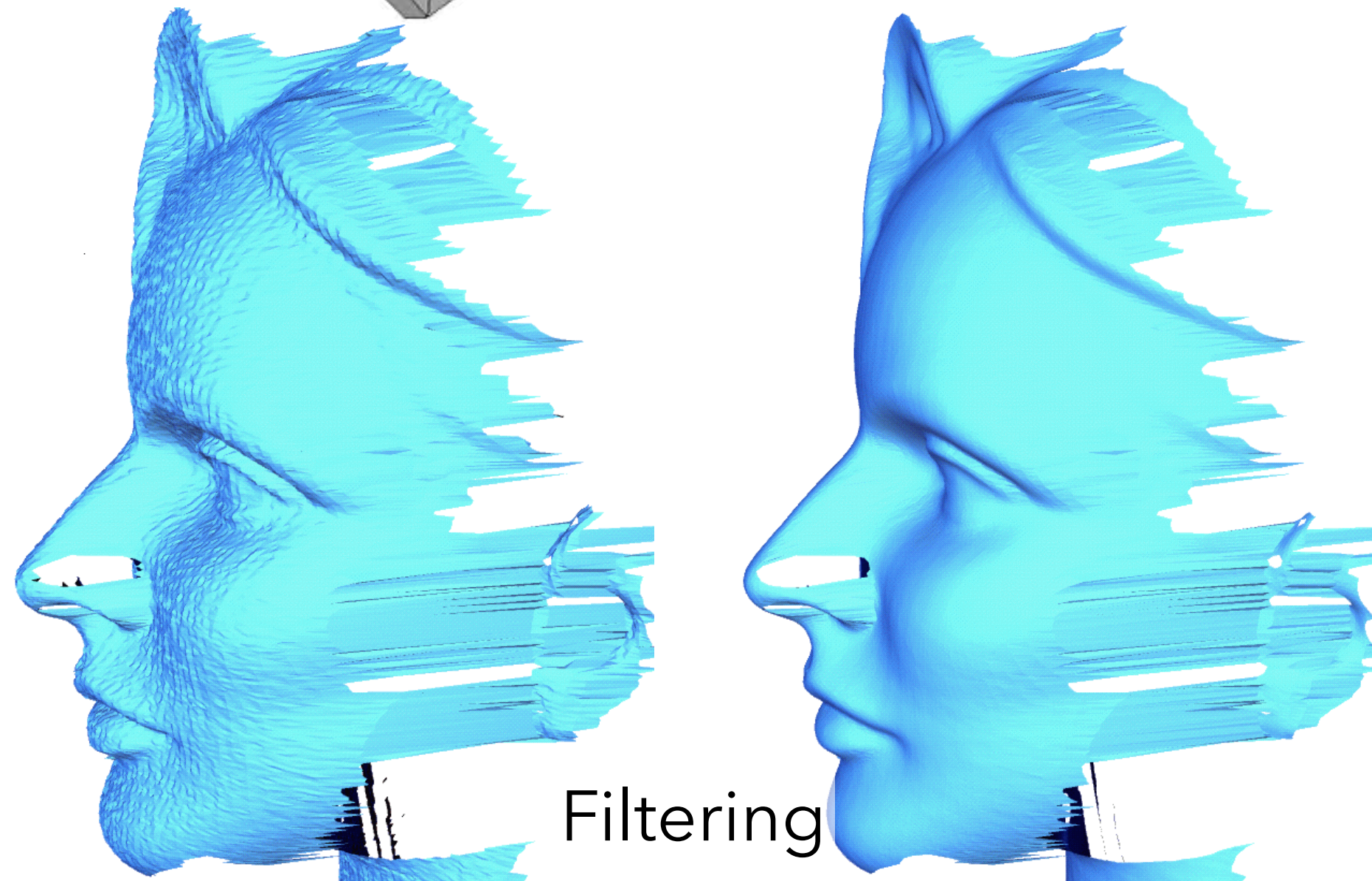
Polygons



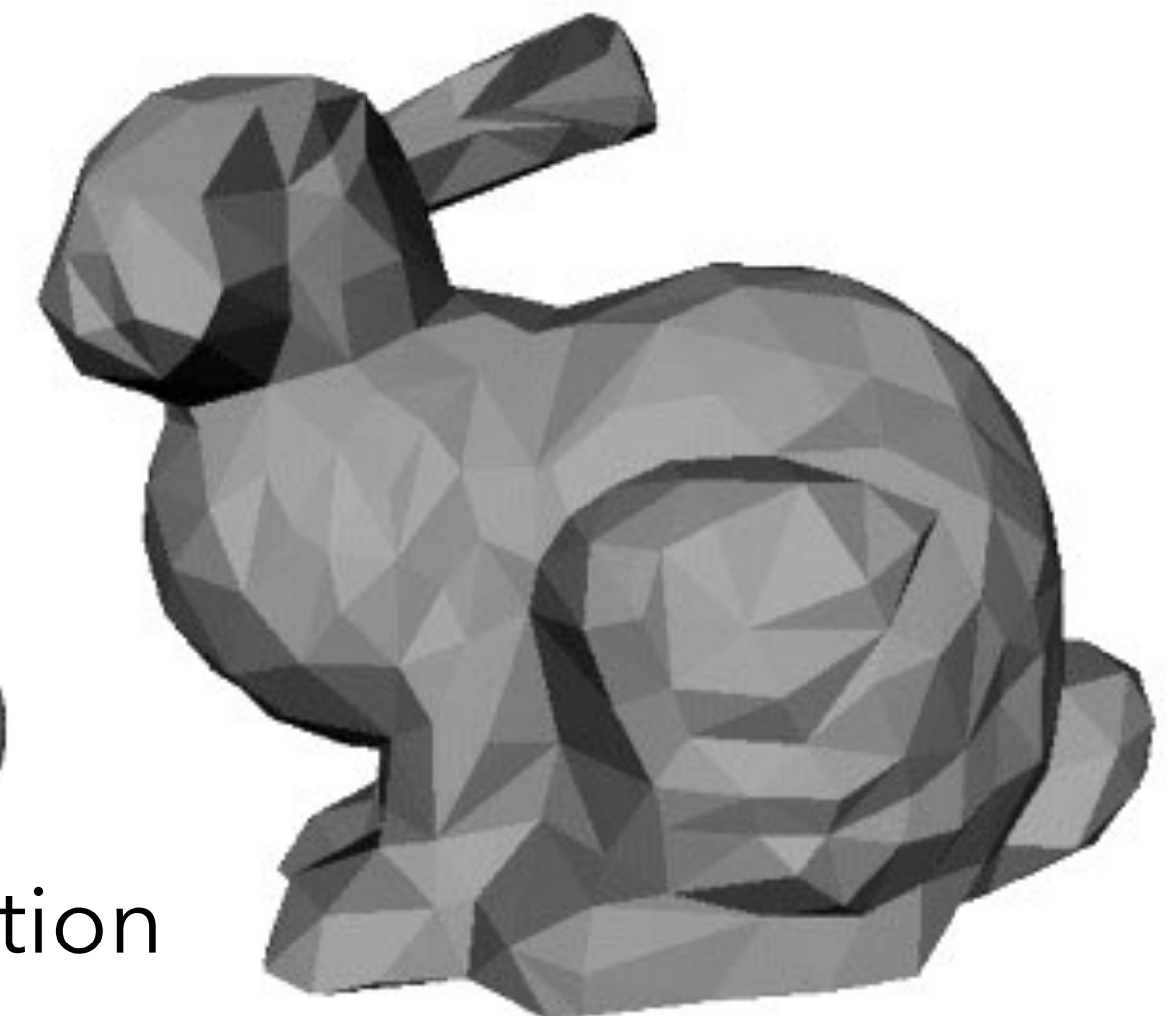
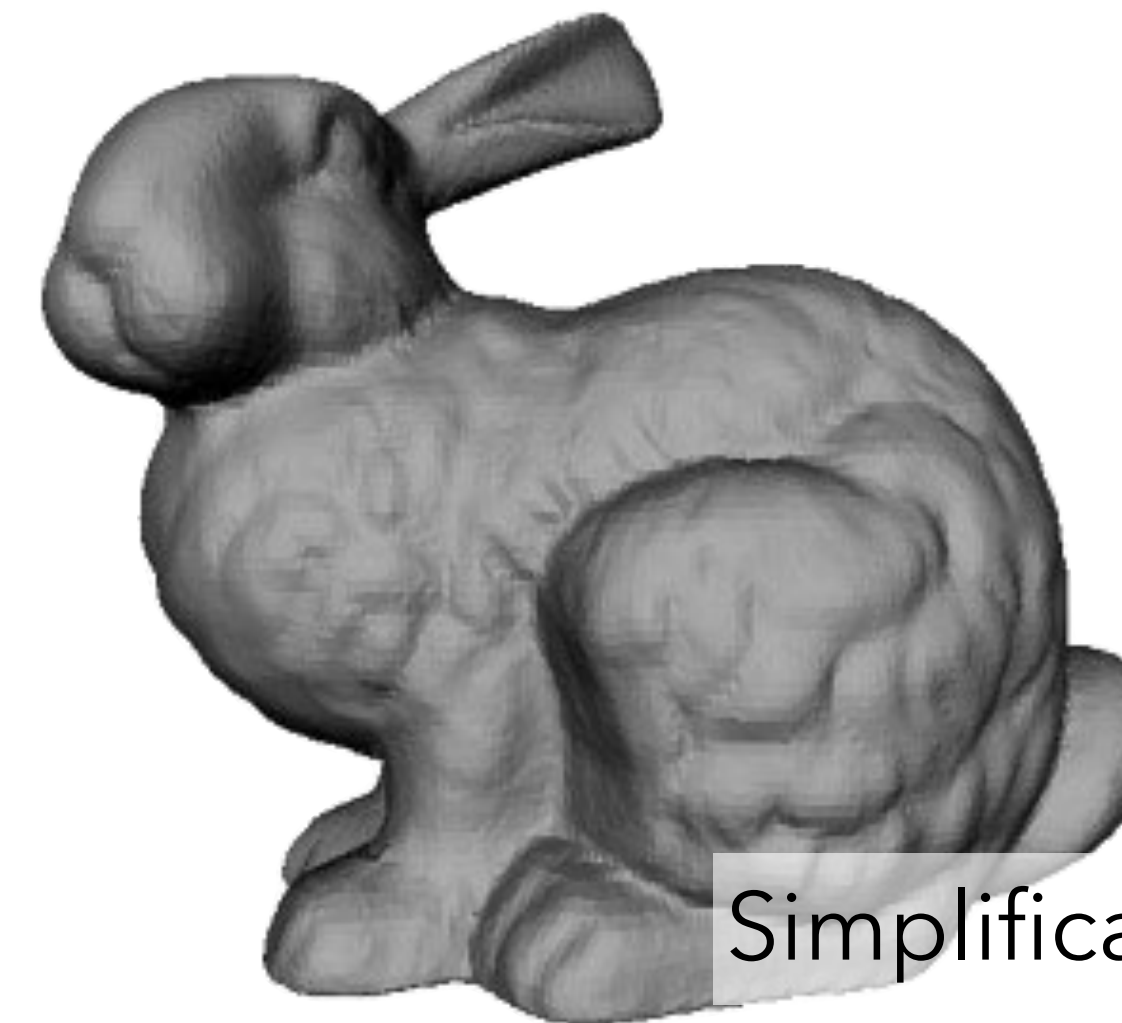
Subdivision



Remeshing



Filtering



Simplification



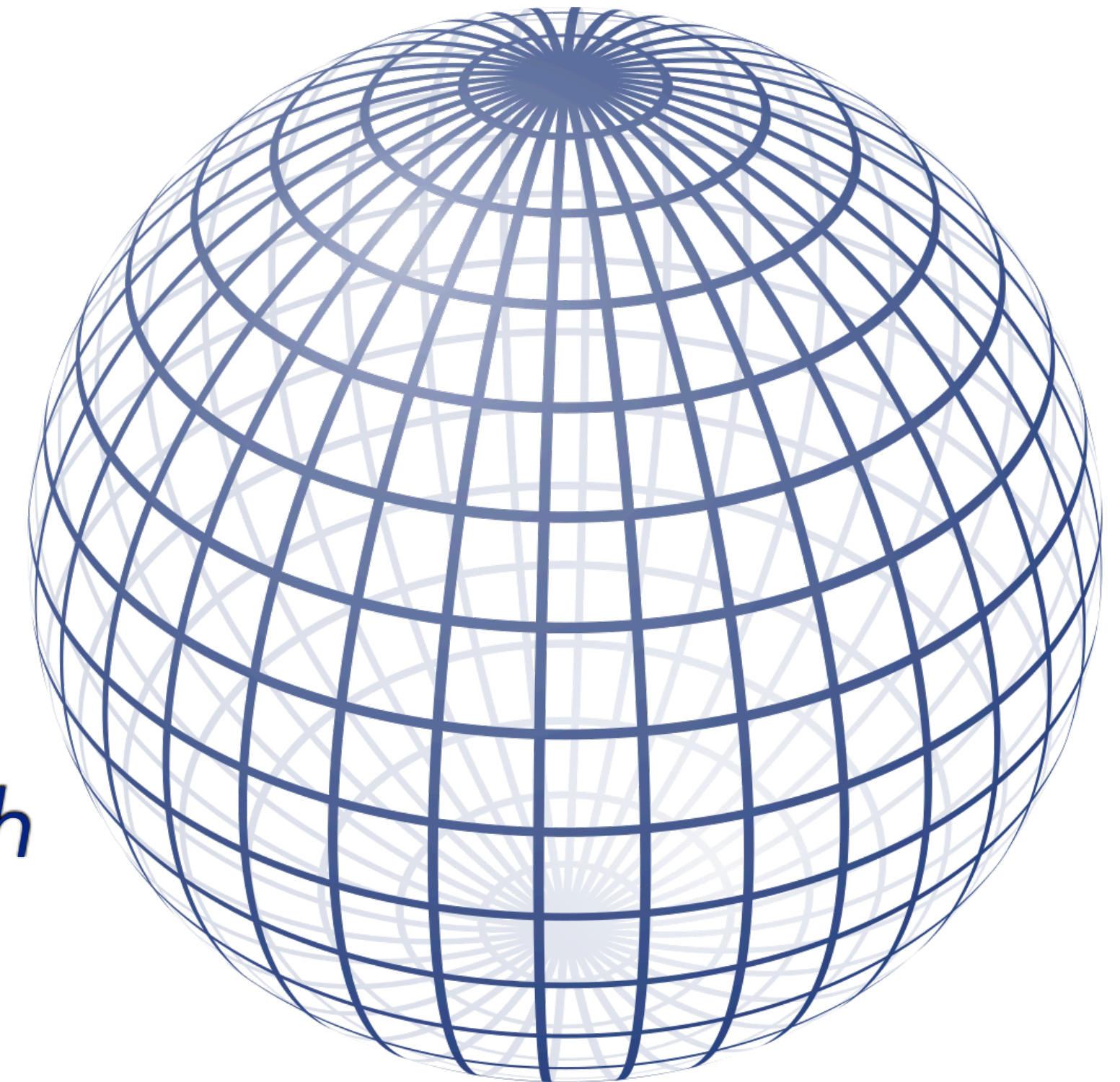
Parametric Surfaces

- Given by a map from (some subset of) $\mathbb{R}^2$ to $\mathbb{R}^3$.

$$\begin{aligned}x &= f(u, v), \\y &= g(u, v), \\z &= h(u, v)\end{aligned}$$

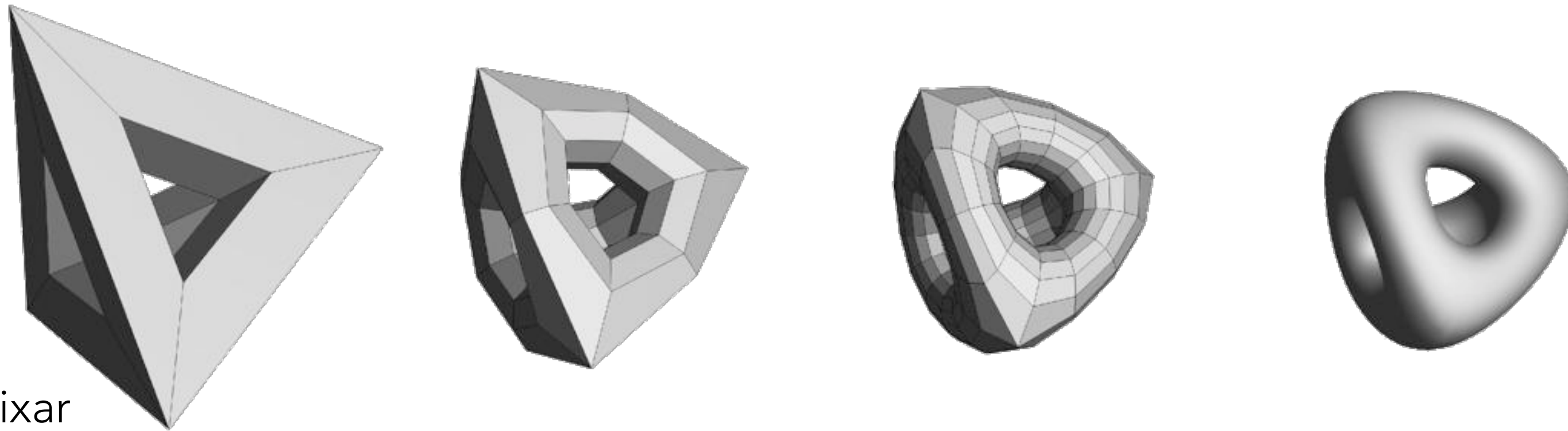
➔ e.g. a sphere is $(\cos u \cos v, \sin u \cos v, \sin v)$

- For modeling general shapes, we usually take f, g, h to be piecewise polynomial functions a.k.a. **splines**



Subdivision Surfaces

- Another way to define a smooth surface: Take a coarse polygon mesh and repeatedly subdivide and smooth it.



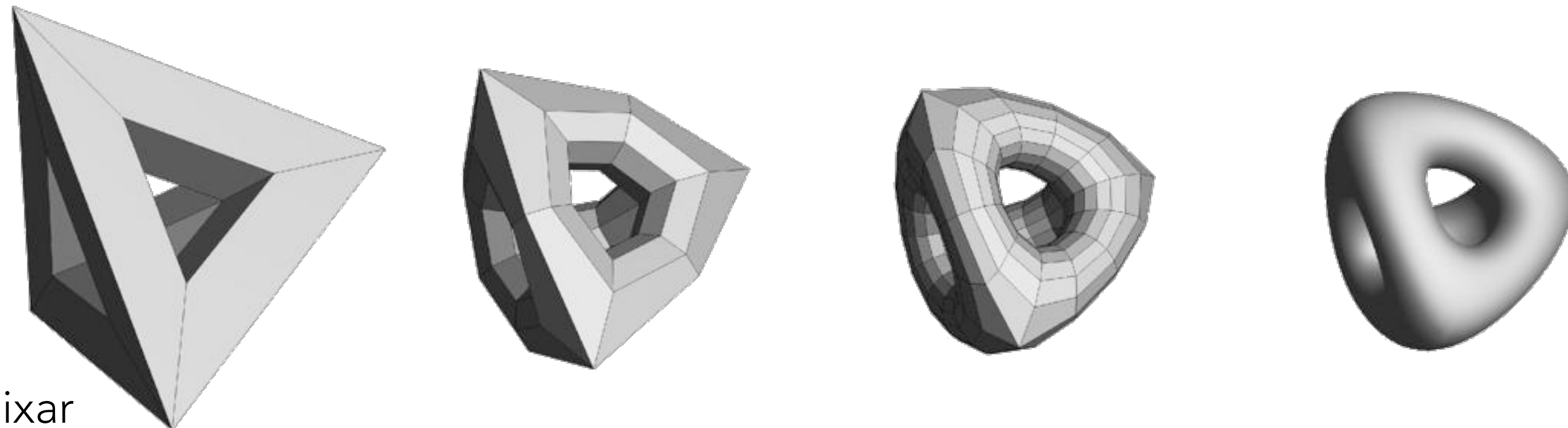
Pixar

- Various smoothing rules for triangle and polygon meshes

Subdivision Surfaces

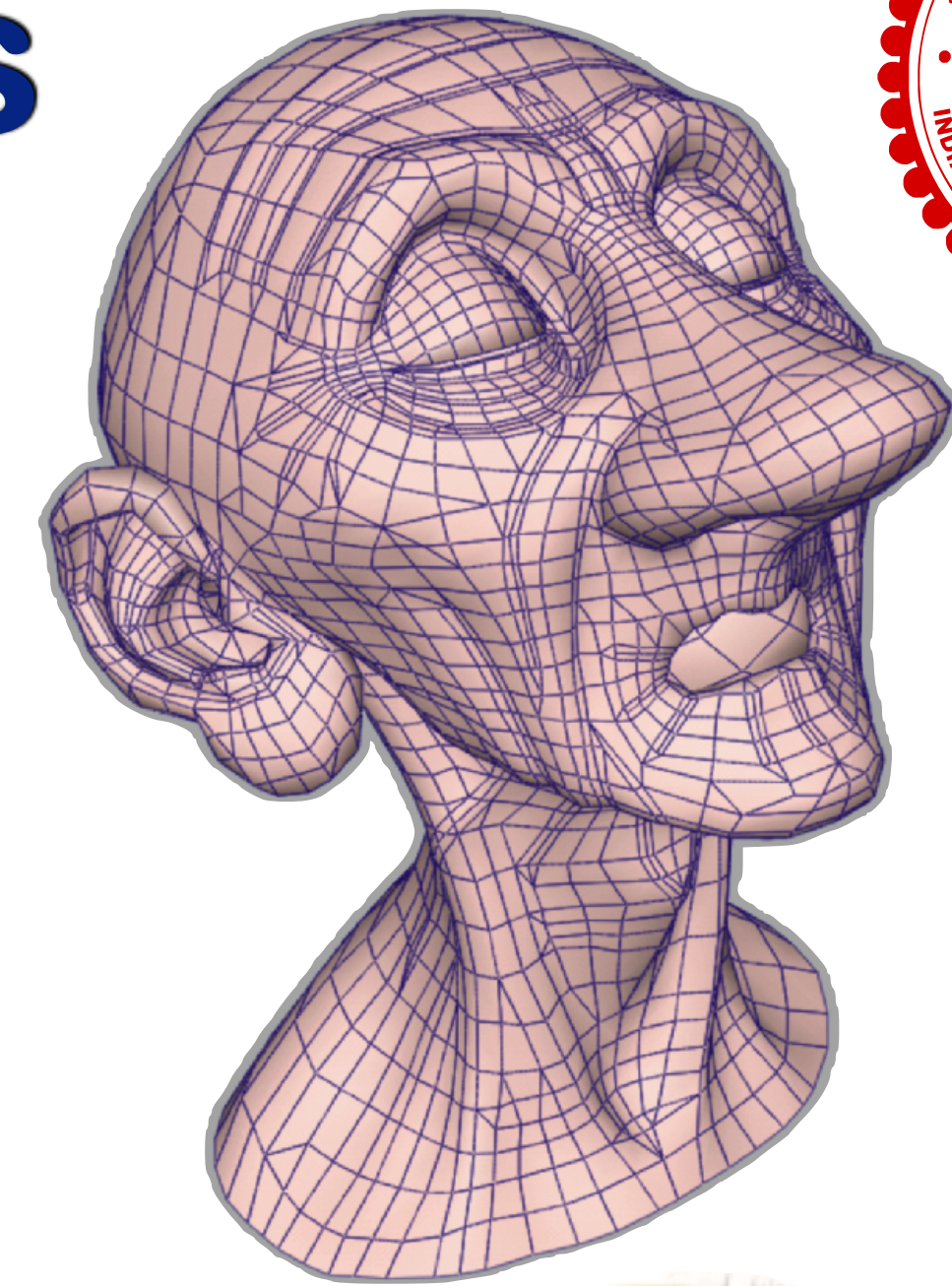


- Another way to define a smooth surface: Take a coarse polygon mesh and repeatedly subdivide and smooth it.



Pixar

- Various smoothing rules for triangle and polygon meshes
- Widely used in practice for modeling characters

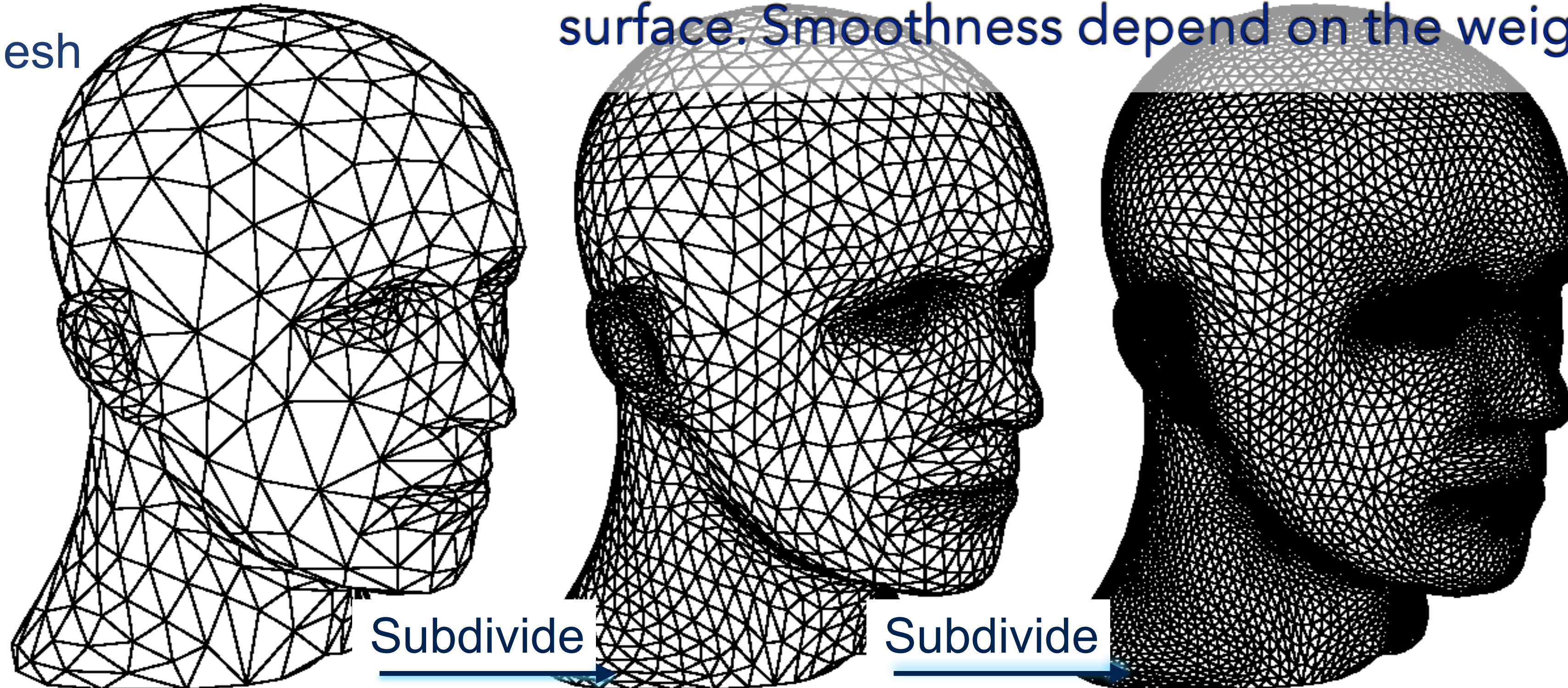


Pixar

Subdivision Surface

Can repeat 'indefinitely', Converges to a smooth surface. Smoothness depend on the weights.

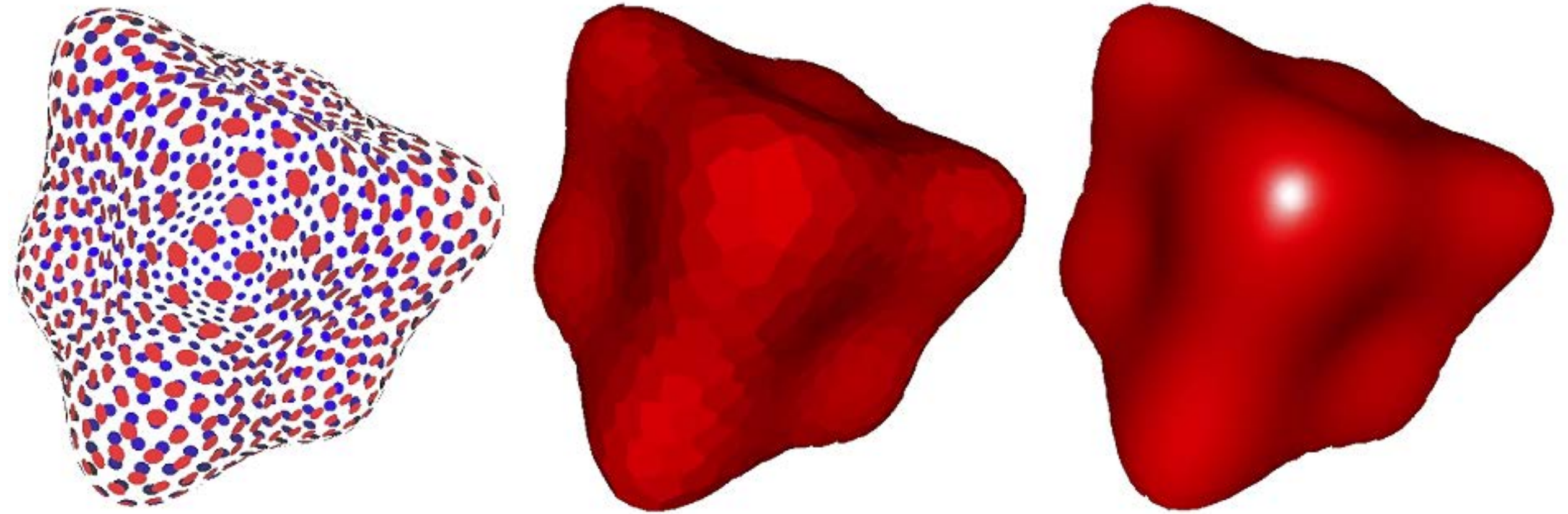
Triangle Mesh



- Introduce new vertices
- Positions are weighted average of old vertices
- The old vertices may also be shifted similarly
- 'Smoothes' edges and vertices (but sharp edges can also be retained)

Point clouds

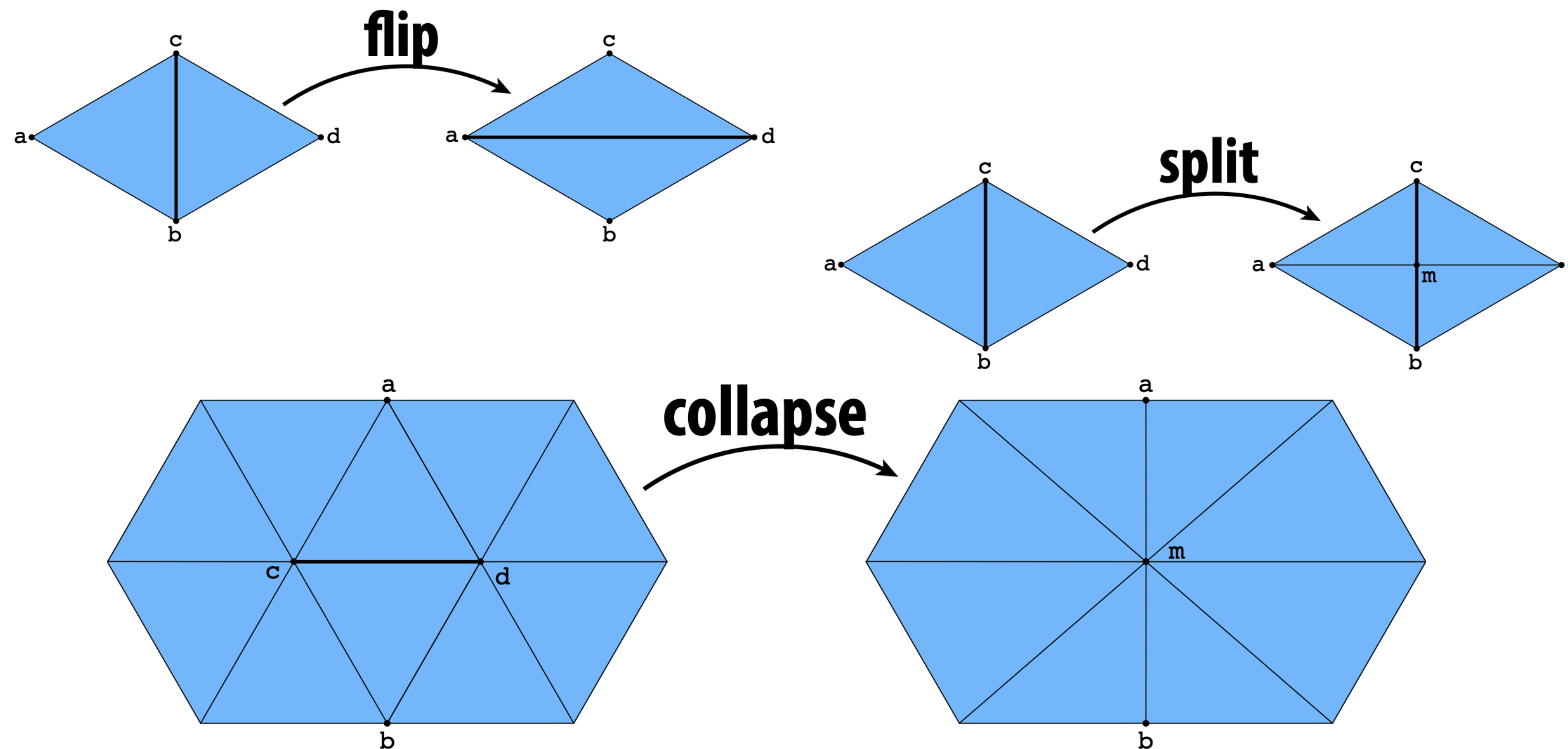
What if you just store a finite set of points (x, y, z) from the surface?
(Optionally including normals)



- Very flexible representation
- Various schemes to **reconstruct** surface between sampled points
- Harder to do processing, editing, simulation, ...

Triangle mesh editing

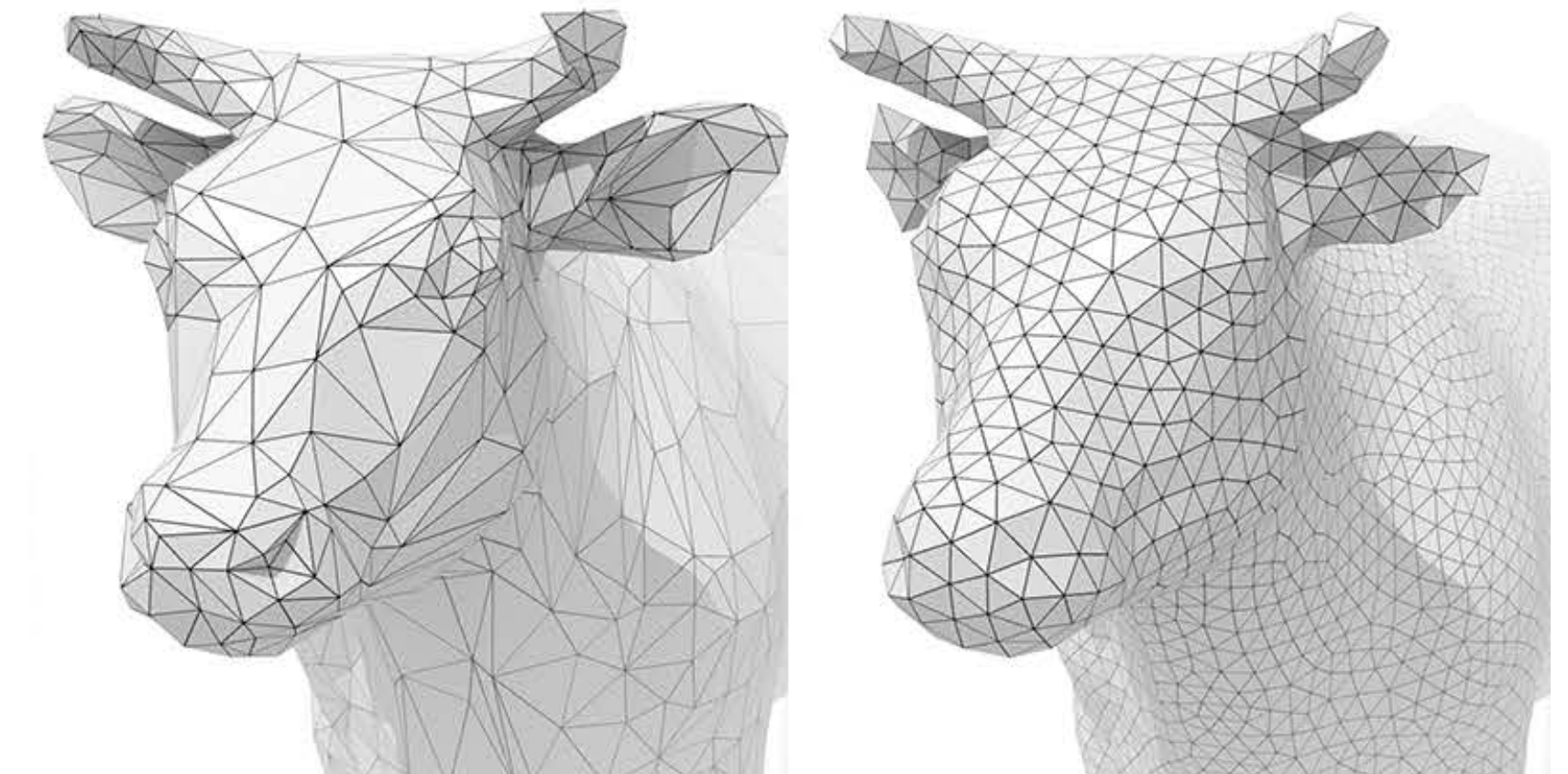
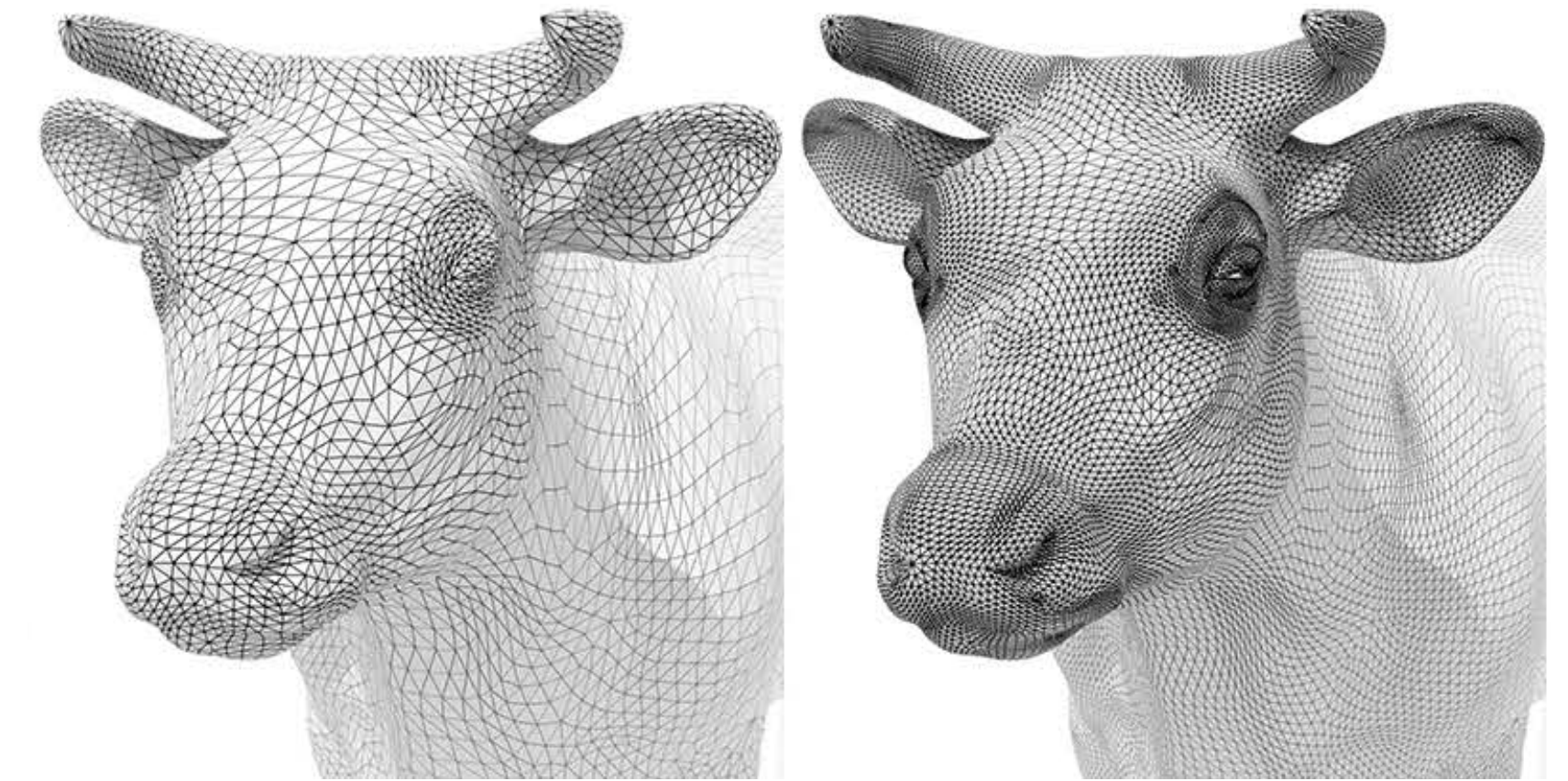
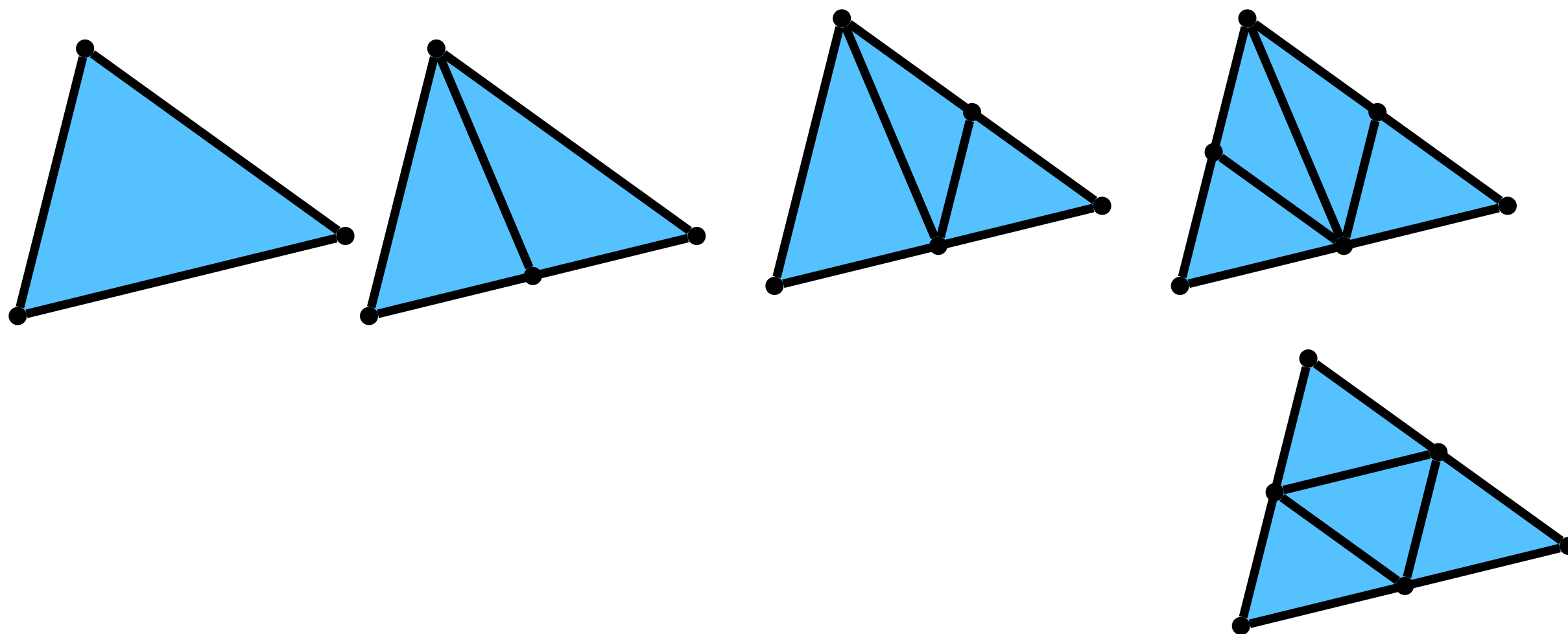
Atomic mesh operations that preserve surface topology (don't allow tear or stitch)



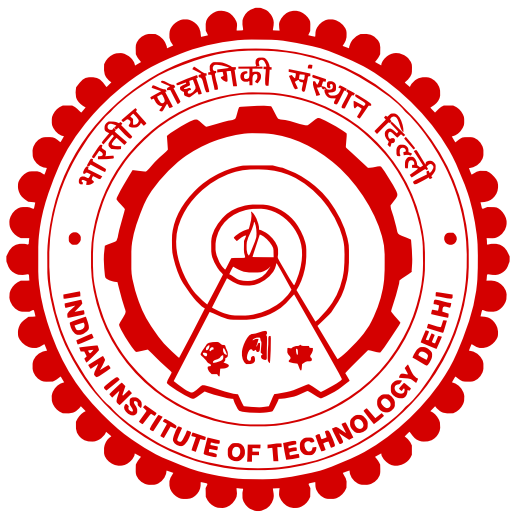
- Must assign positions, texture coordinates, and other attributes of new vertices
- Other data, e.g. normals, of all neighbours may need to be recomputed

Mesh Editing

- **Subdivision:** split all edges
- **Simplification:** collapse unimportant edges
- **Regularization:** split long edges, collapse short edges
- **Fattening:** Make shapes nicer



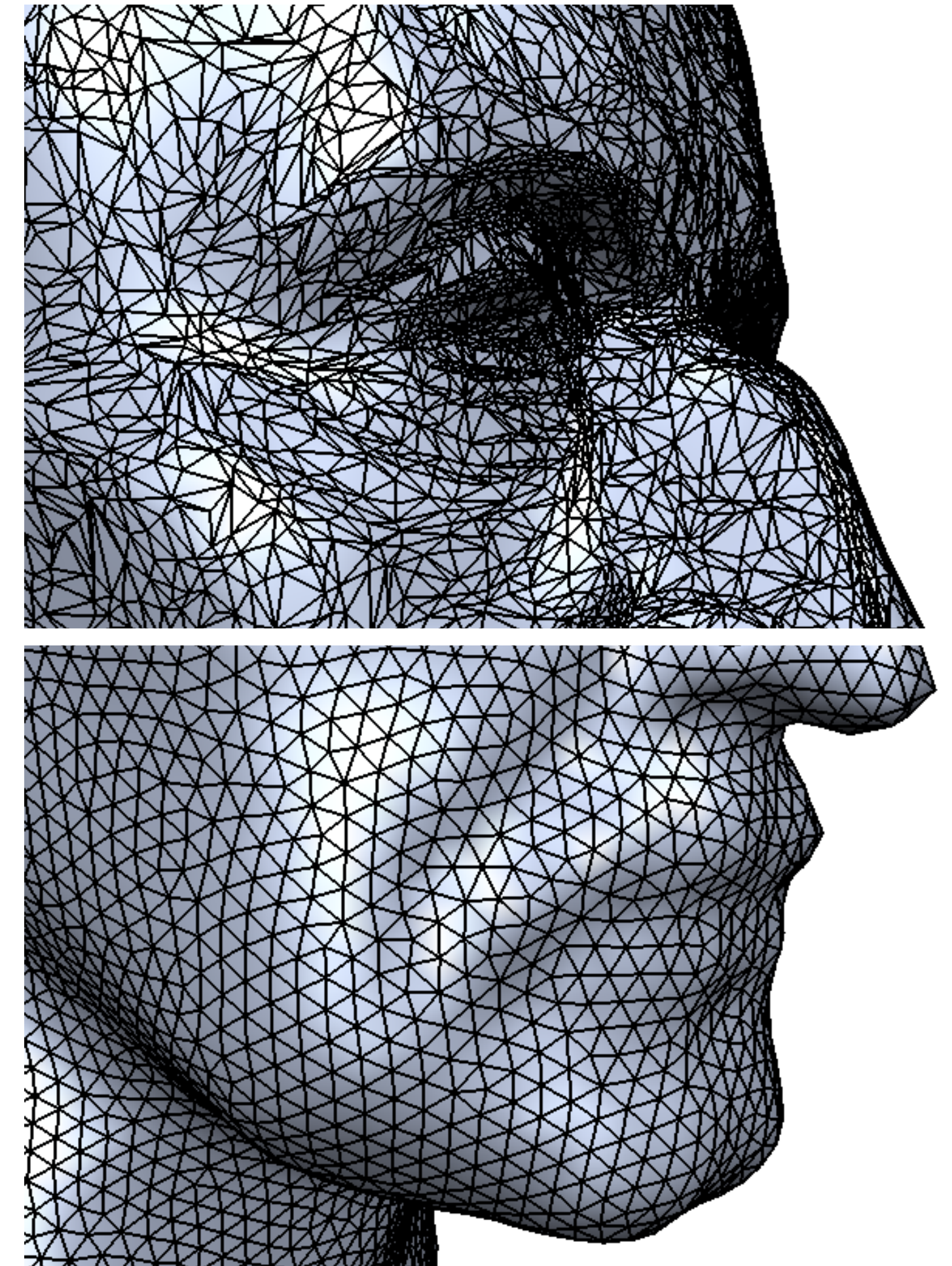
Example: Isotropic Remeshing

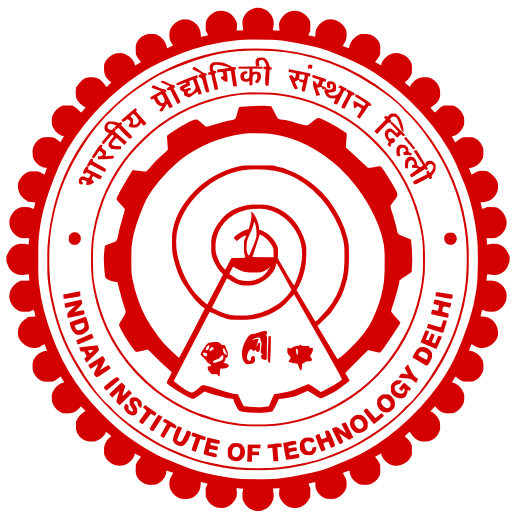


Choose desired edge length L

1. Split all edges longer than L
2. Collapse all edges shorter than L
3. Flip edges to bring vertex degrees closer to 6
4. Move vertices tangentially towards average of neighbours
5. Repeat steps 1-4 several times

[Botsch & Kobbelt 2004]





Mesh Quality Metrics

- Skew
 - ➔ Closeness to ideal shape (e.g., equilateral triangle)
 - ➔ Minimum angle, Parametric distortion
- Aspect Ratio
 - ➔ Ratio of the longest side to the shortest side
- Dihedral angle
 - ➔ Angle between adjacent triangles

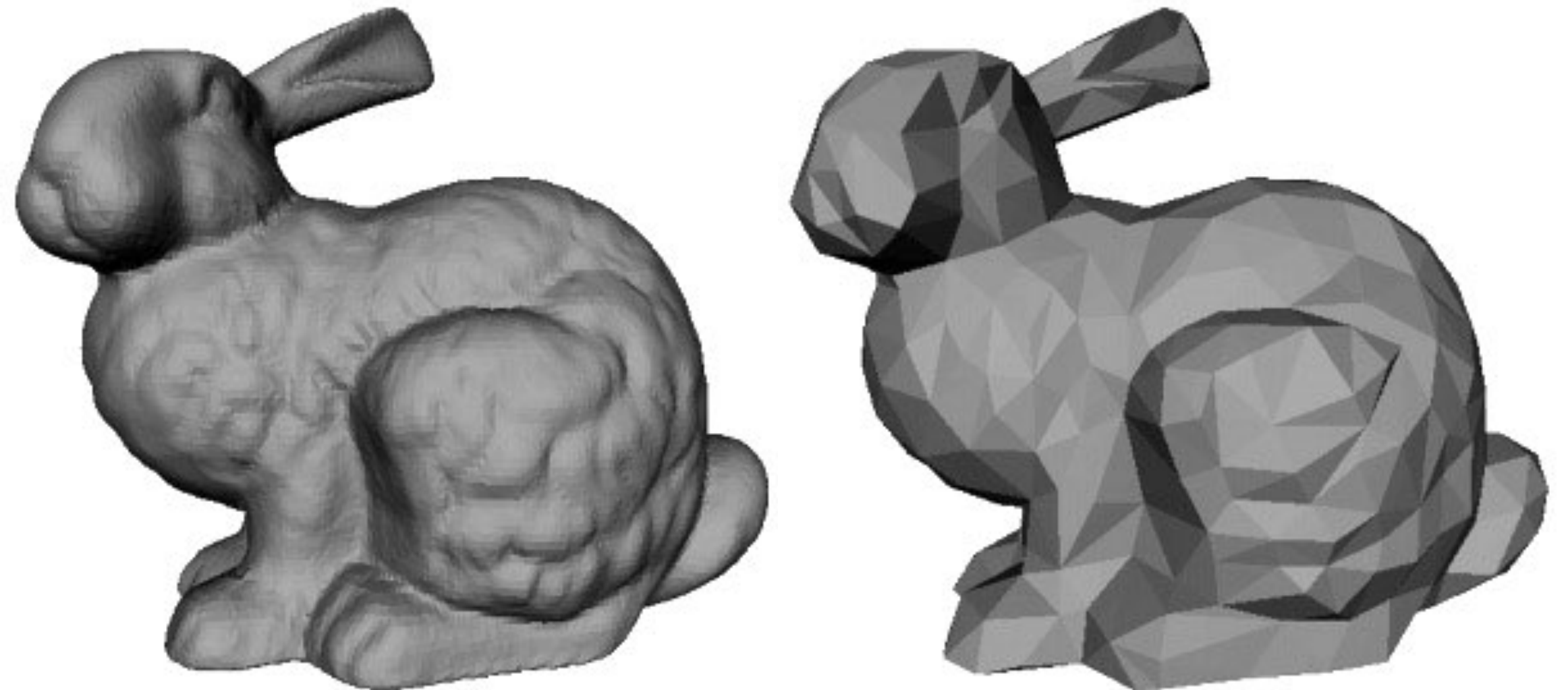
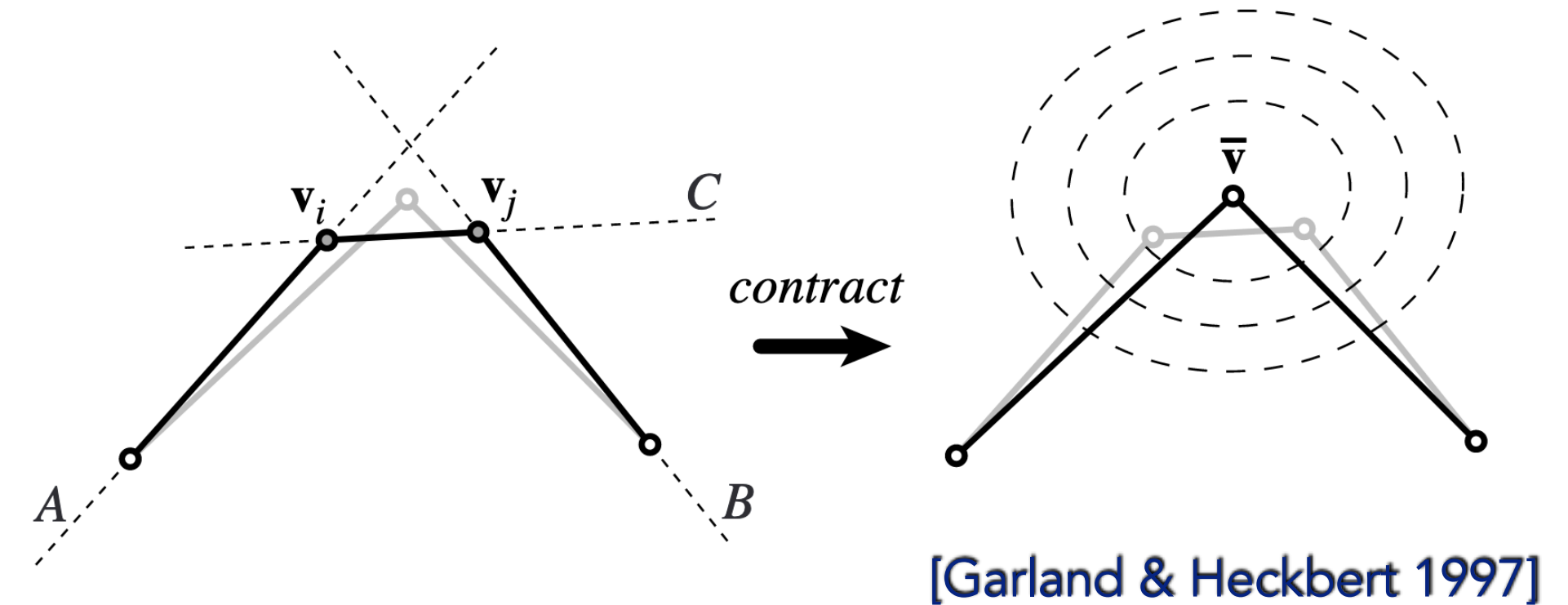


Example: Mesh simplification

- Define cost of collapsing an edge $\approx$ deviation of new vertex from original surface
- Maintain priority list of candidate edges

1. Collapse edge with lowest cost
2. Update costs
3. Repeat until desired number of vertices remain

Simple, greedy, works quite well!



Measuring Geometric Distance



- Surface-surface
 - ➔ Hausdorff distance: max of closest distances
- Vertex-surface
- Vertex-plane
- Vertex-vertex
- Average distance or Max distance



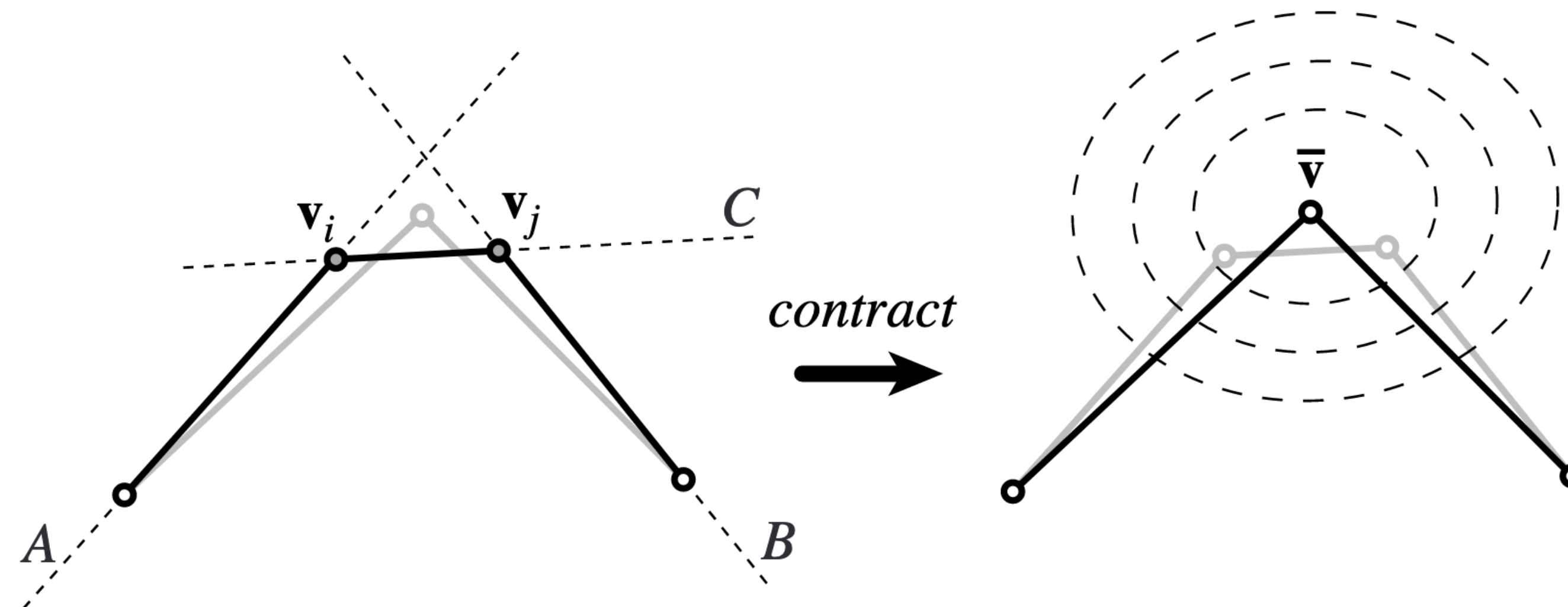
Quadric Error

For each face $\mathbf{f}$, error of a point $\mathbf{p} = \text{dist}(\mathbf{p}, \text{plane}(\mathbf{f}))^2 = (\mathbf{n} \cdot (\mathbf{p} - \mathbf{p}_0))^2$

For each vertex, "quadric error" $\Delta(\mathbf{p}) = \sum_{\text{adj. face } \mathbf{f}_i} \text{dist}(\mathbf{p}, \text{plane}(\mathbf{f}_i))^2$ is **quadratic** in $\mathbf{p}$

Cost of collapsing edge $(\mathbf{v}_i, \mathbf{v}_j)$ to new vertex $\bar{\mathbf{v}}$: $\Delta(\bar{\mathbf{v}})$

$\bar{\mathbf{v}}$ is unknown, find one that minimizes cost





Quadric Error Matrix

Plane Equation: $Ax + By + Cz + D = 0$

$$P^T v = 0, \quad P = [A \ B \ C \ D]^T$$

Distance to vertex $v, [x, y, z, 1]^T = P^T v$

$$\text{Quadric error of vertex } v, \Delta v = \sum_{v \in P_i} (P_i^T v)^2 = v^T \underbrace{\left(\sum (P_i P_i^T) \right)}_{Q_v} v$$

$$\text{Recall, } PP^T = \begin{bmatrix} A^2 & AB & AC & AD \\ AB & B^2 & BC & BD \\ AC & BC & C^2 & CD \\ AD & BD & CD & D^2 \end{bmatrix}$$

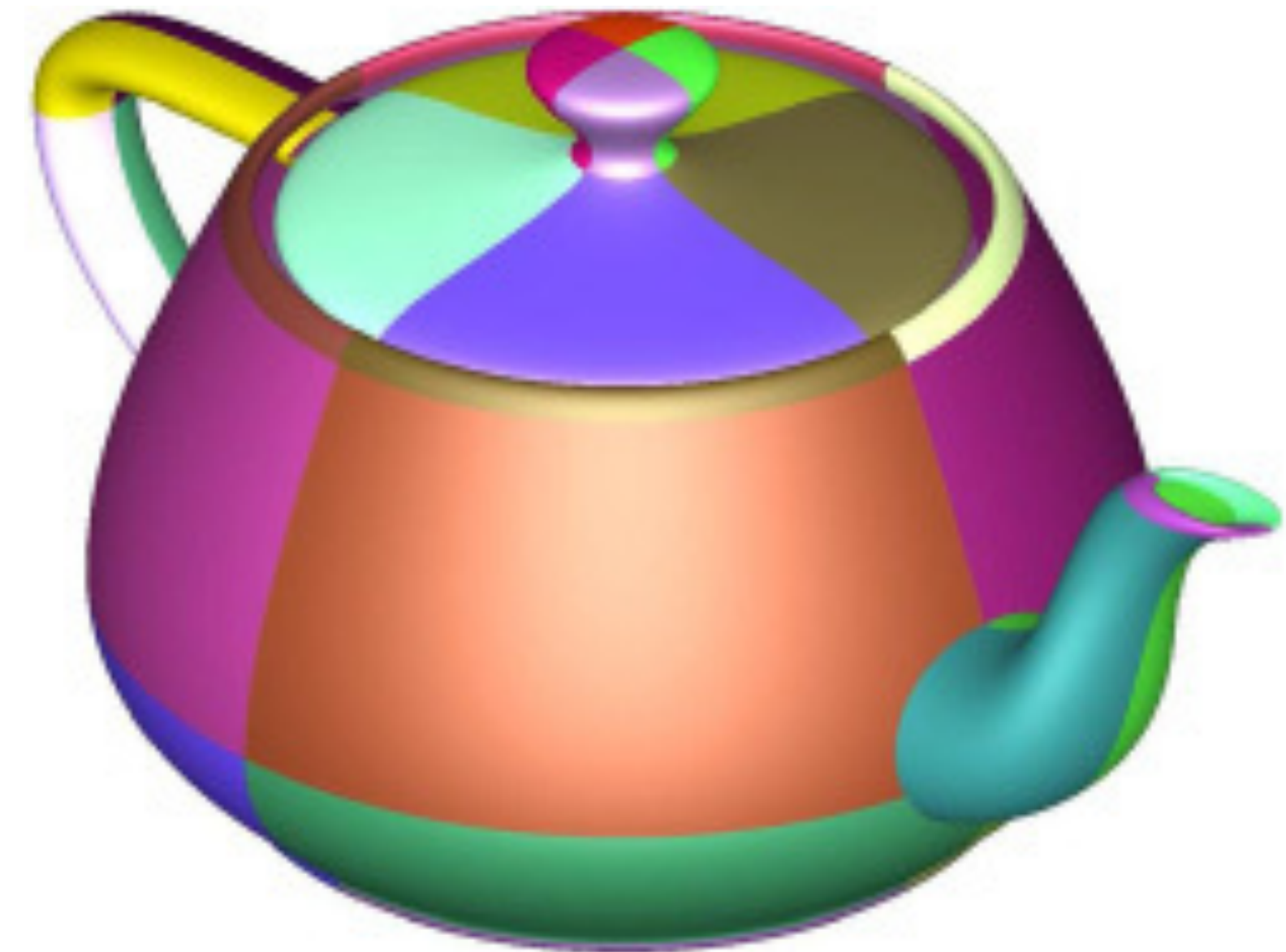
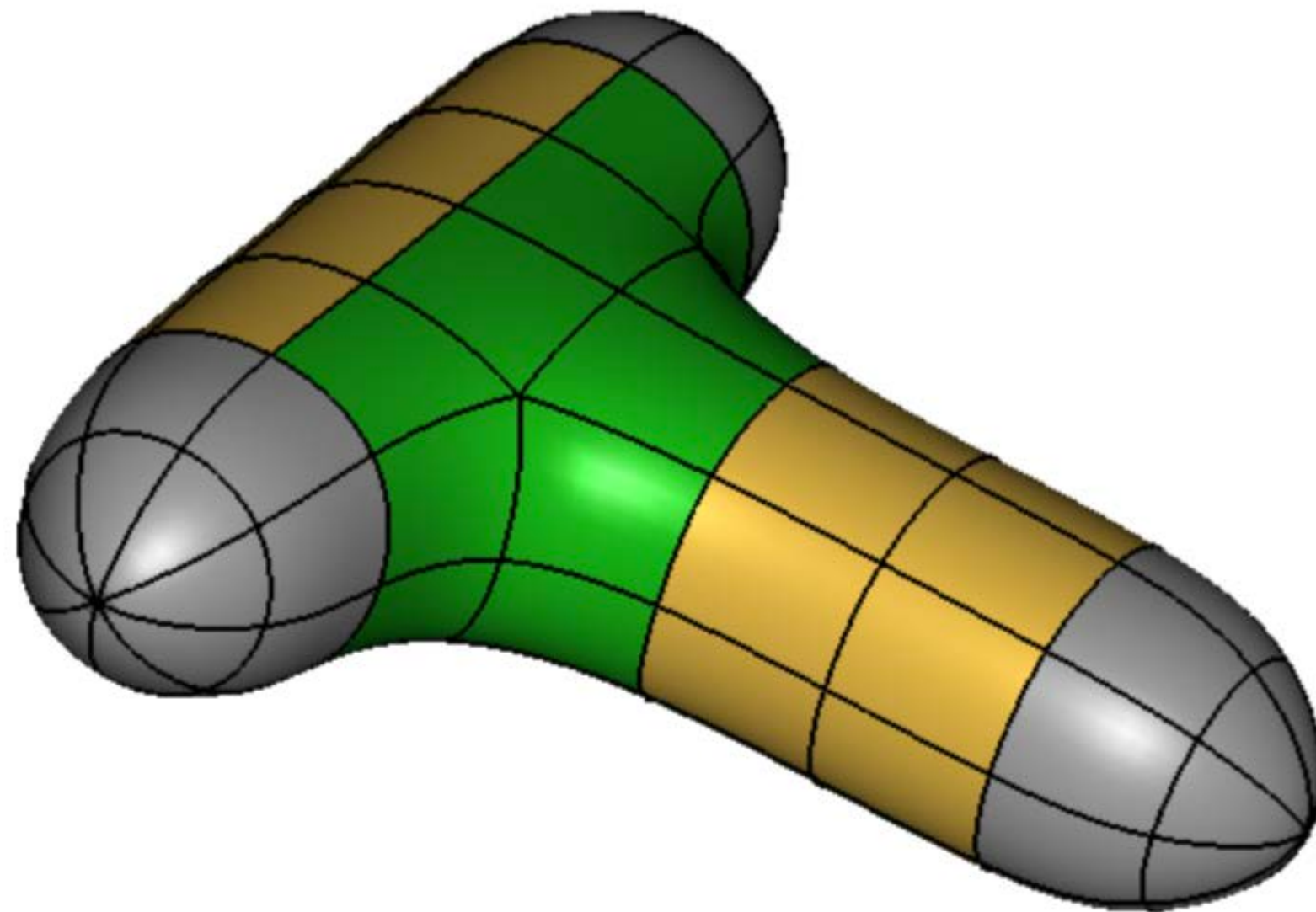
Minimize $\Delta v \Rightarrow$ Derivatives are 0:

$$\text{Solve } \begin{bmatrix} Q_{00} & Q_{01} & Q_{02} & Q_{03} \\ Q_{10} & Q_{11} & Q_{12} & Q_{13} \\ Q_{20} & Q_{21} & Q_{22} & Q_{23} \\ A0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

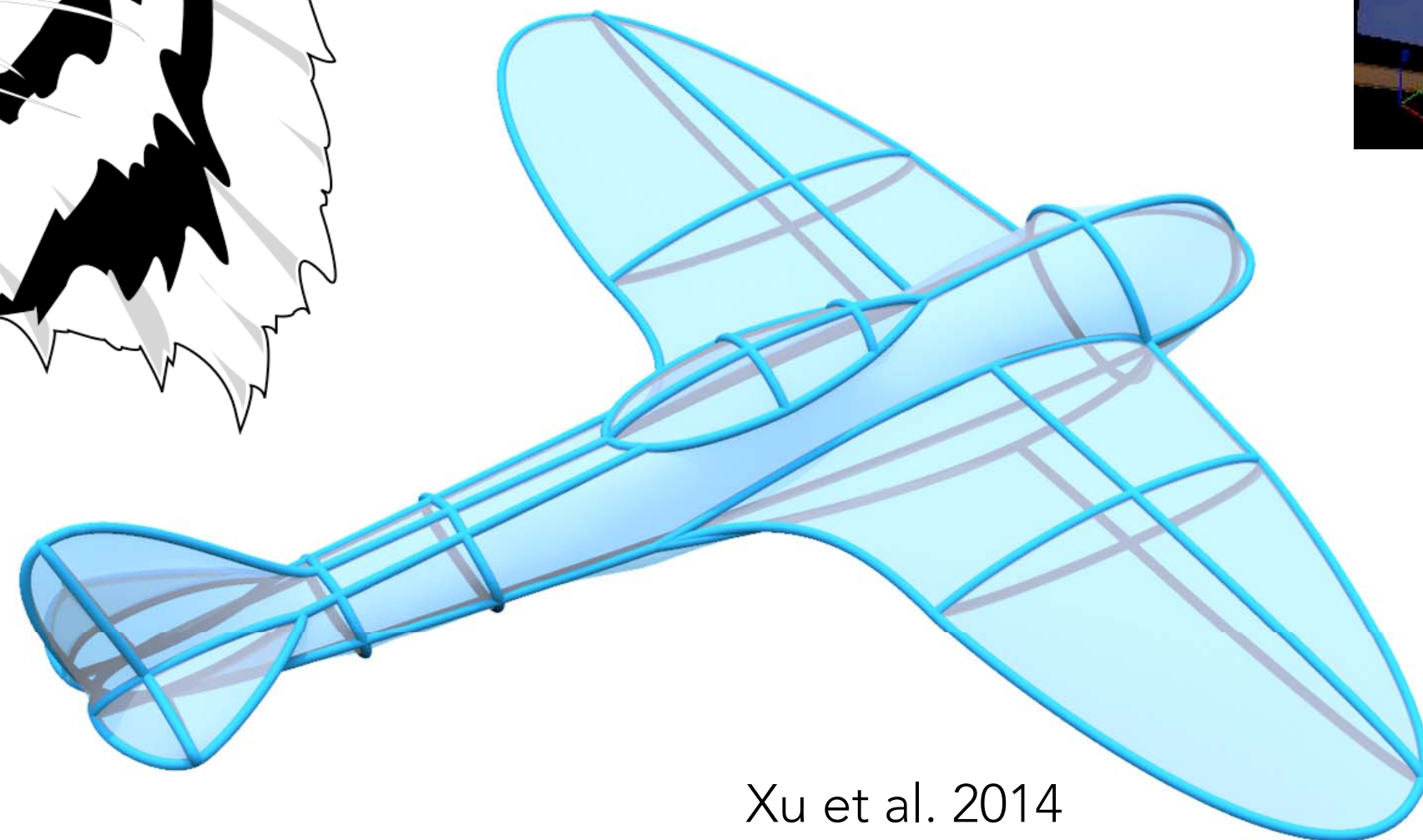
On collapsing edge (v_i, v_j) to new vertex v : $Q_v = Q_{v_i} + Q_{v_j}$

Modeling Curved Shapes

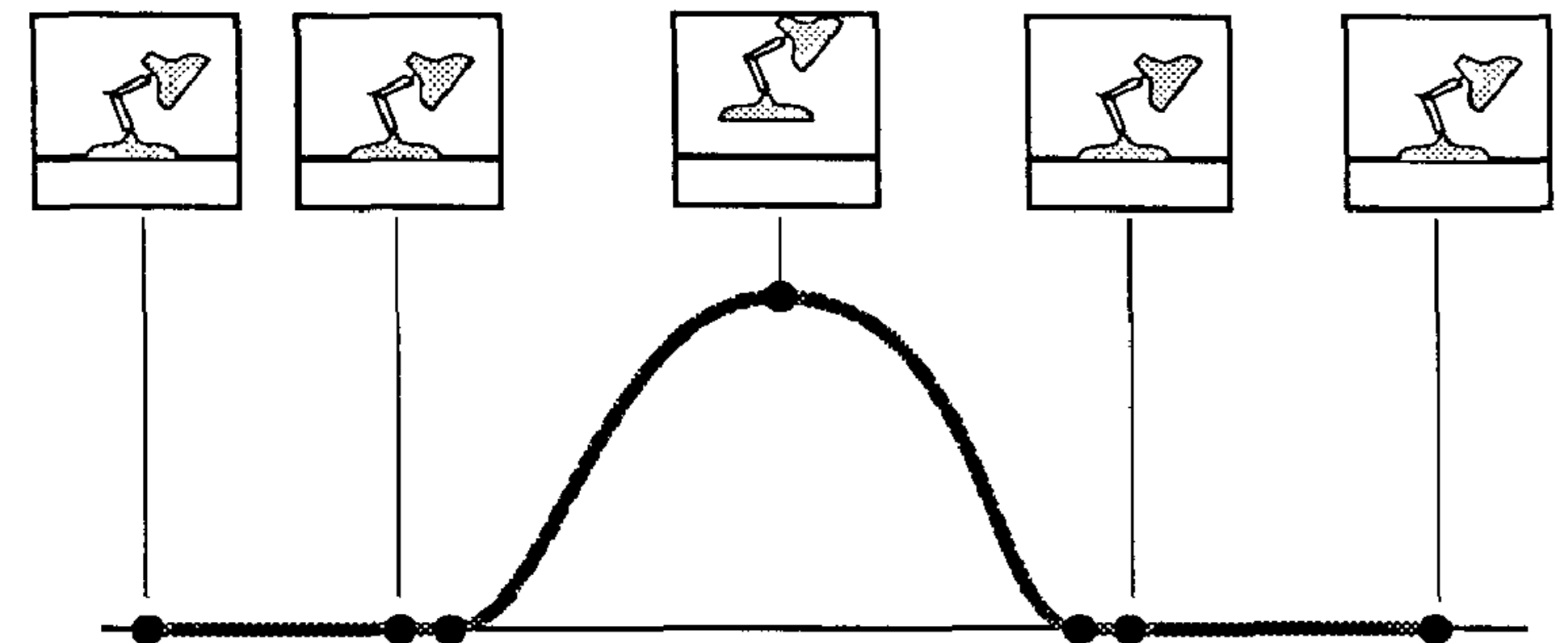
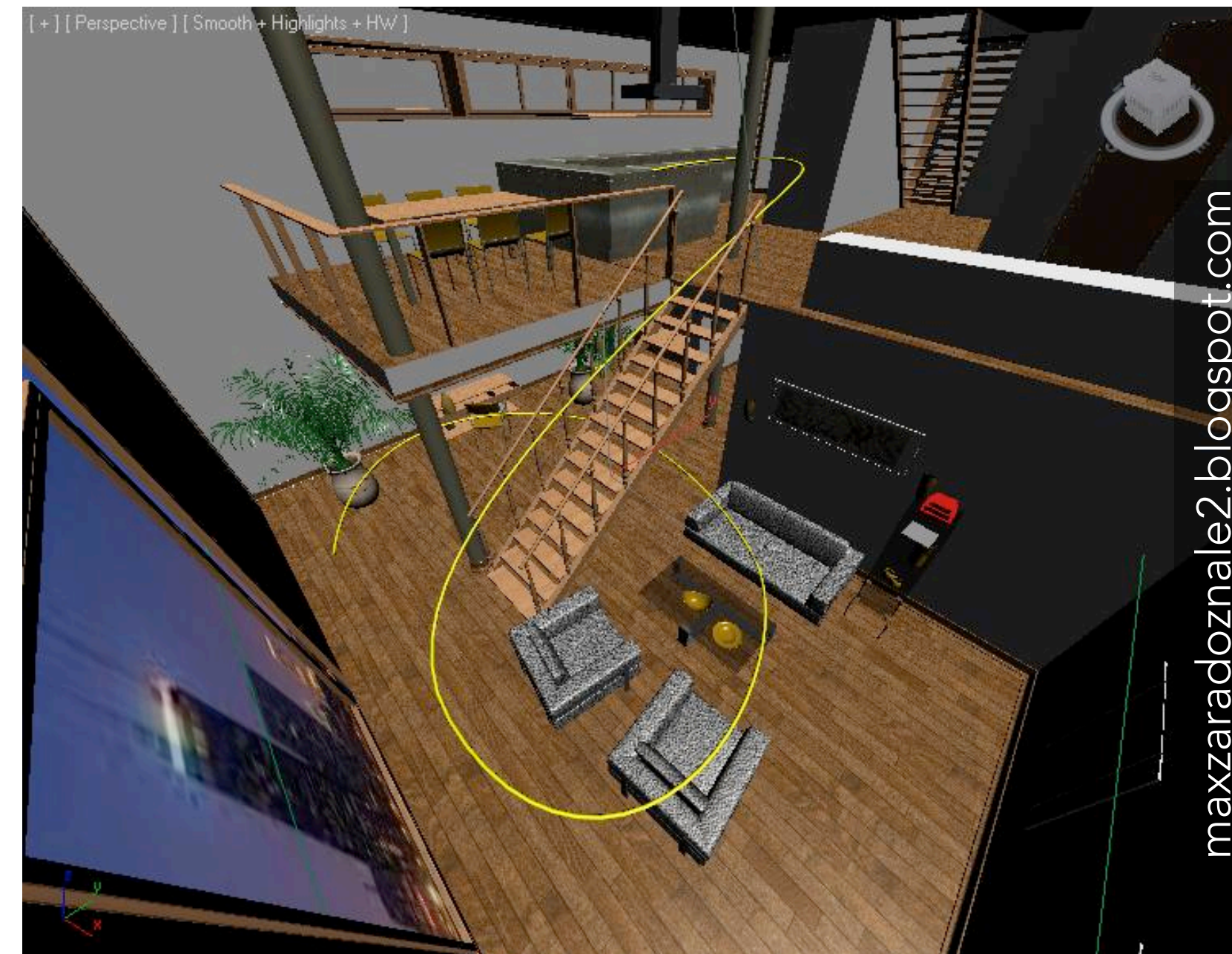
- With meshes we can only approximate smooth shapes, by adding lots of vertices and triangles.
- How can we directly model a smooth shape?



Curves



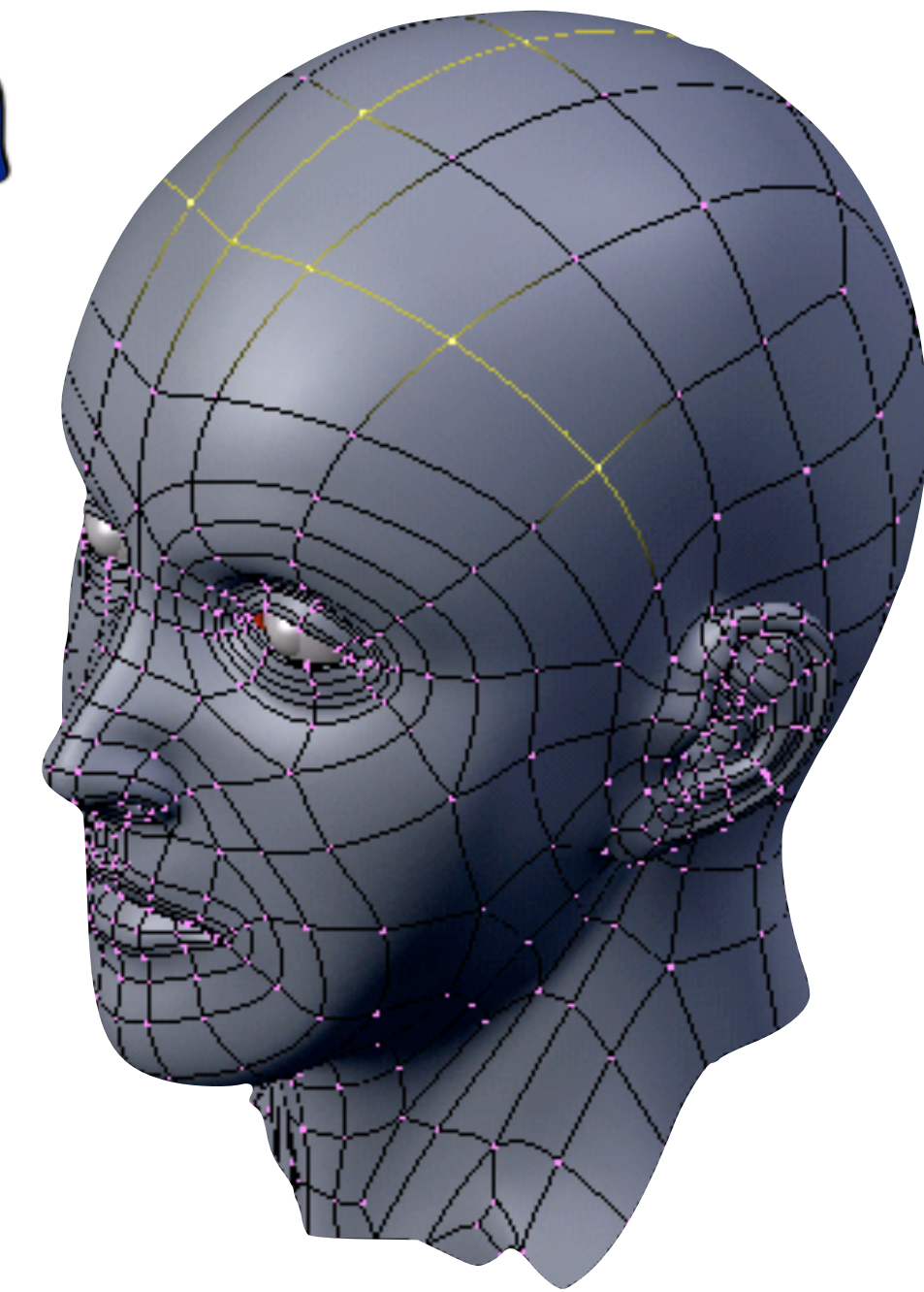
Xu et al. 2014



Lasseter 1987

Curves to Fit Data

(Surfaces)



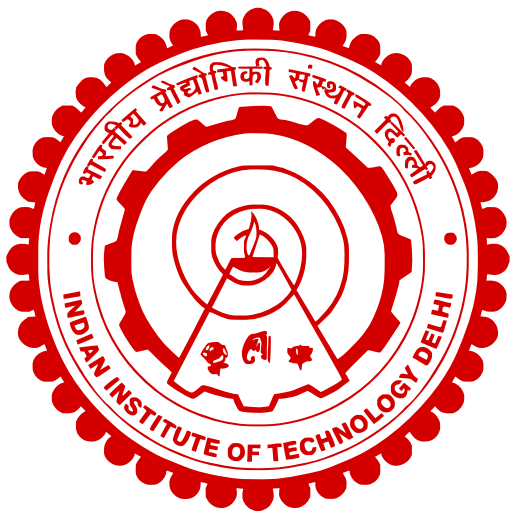
- Analytic Function
 - ➡ Parametric form: $C(t)$
 - ➡ Implicit form: $F(t) = 0$
- Design based on known or desired points
 - ➡ Curve should pass near or through these

Approximating
curve



known points
(aka data-points)

curve interpolates
this data point



Representation Desirables

- Stable
 - ➡ Small changes to data points should induce small change to the curve
- Smooth
 - ➡ Tangent should be continuous
 - ➡ Or, second derivative should be continuous
 - ➡ Second derivative is related to curvature
 - Curvature indicates how fast curve turns
- Easy to evaluate
 - ➡ including derivatives, curvature, etc.
- Interpolate or approximate data



Parametric Form

Suppose curve C is a polynomial in t
i.e., Point $P(t)$ on curve C is computed as follows:

$$P(t) = \sum_{k=0}^n (c_k t^k)$$

c_k are vector coefficients
(with x, y, z coordinates)

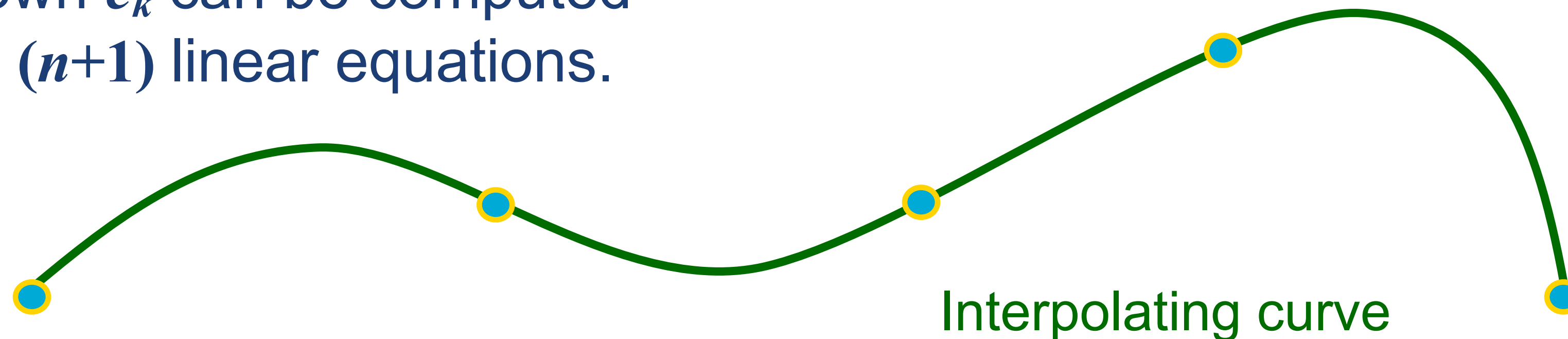
Other forms are possible, we will see.
This one is called power basis.

All data points satisfy the above equation.

If the number of data points is $(n+1)$,

$\Rightarrow (n+1)$ unknown c_k can be computed
using $(n+1)$ linear equations.

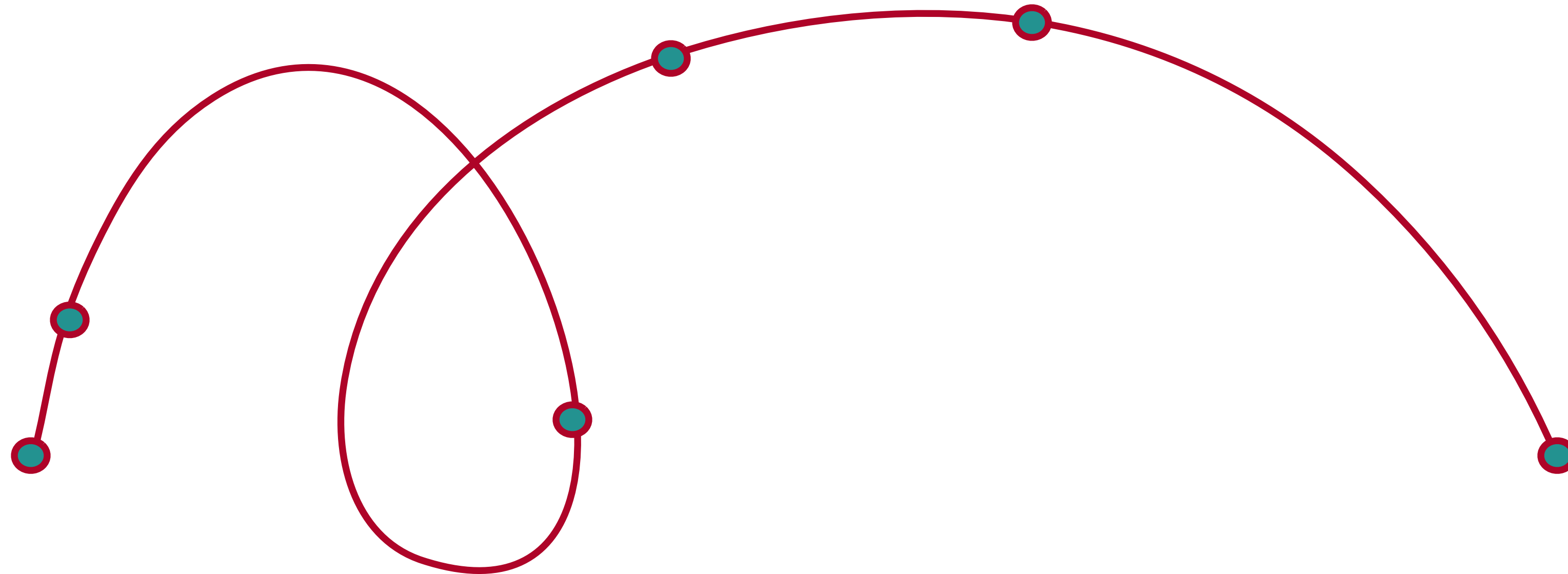
(For each coordinate)



Interpolating Curve



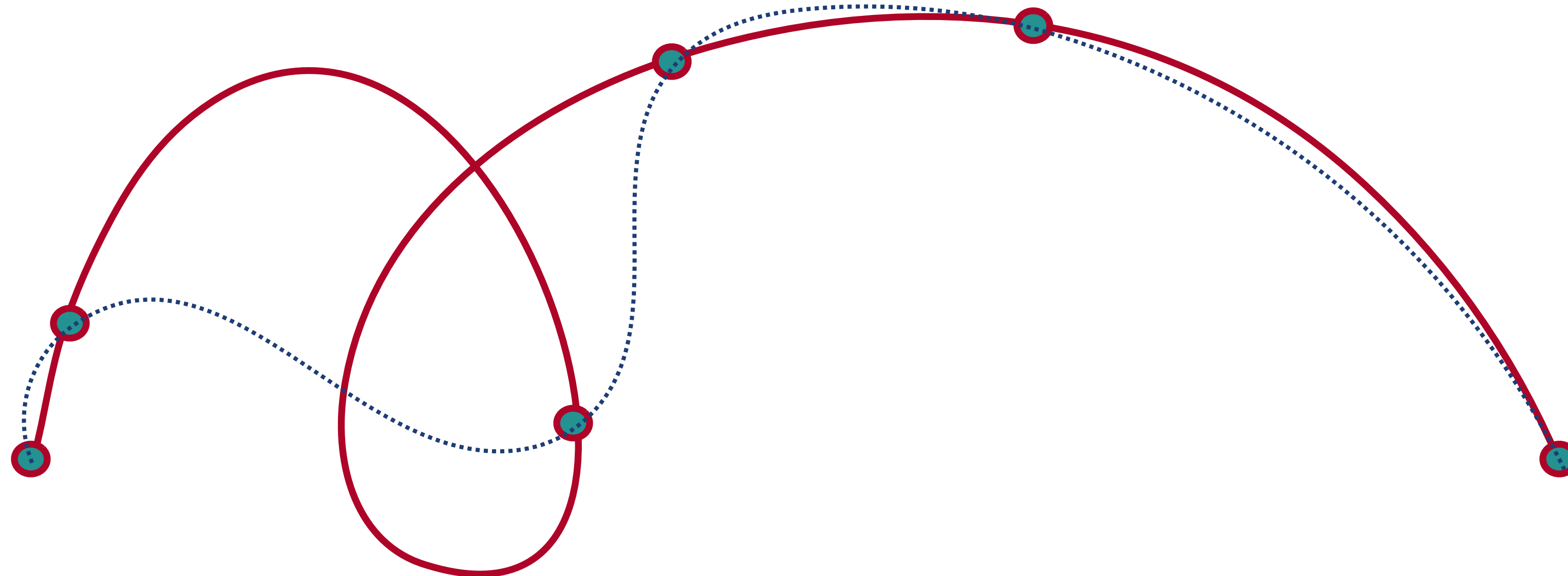
High degree can lead to unstable and unnatural curves





Interpolating Curve

High degree can lead to unstable and unnatural curves



But a low degree curve does not have sufficient degrees of freedom ..
too few coefficients and too many points to match

Can we piece together lower degree curves?



Matrix Form for Cubics, e.g.

Consider cubic curve, $n = 3$: $P(t) = \sum_{k=0}^3 (c_k t^k)$ (Power basis)

Let,

$$[c] = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

$$[t] = \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

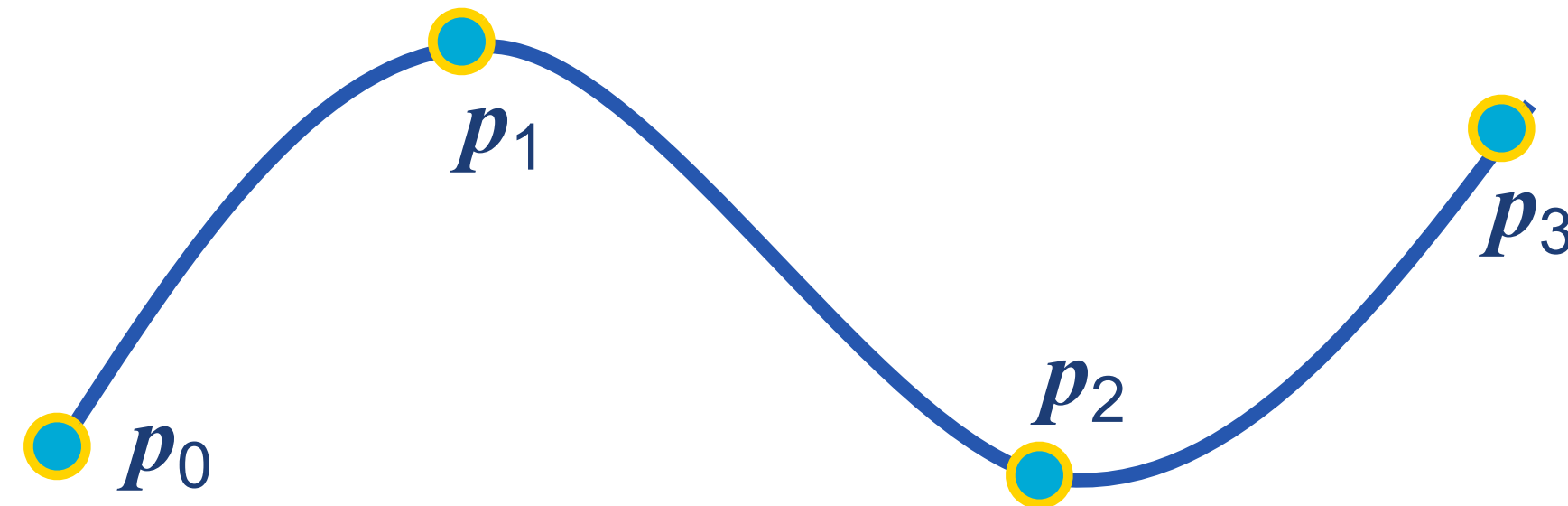
$$\Rightarrow P(t) = [t]^T [c] = [c]^T [t]$$



Interpolating Cubic Curve

Find cubic $P(t)$ interpolating p_i , i.e., find $[c] = [c_0, c_1, c_2, c_3]^T$

Eq1



Assume interpolation at $t = 0, \frac{1}{3}, \frac{2}{3}, 1$ (call these $t_0..t_3$)

(This is called uniform parameterization)

Other parameterizations are possible.

$$p_0 = P(0) = c_0$$

$$p_1 = P(\frac{1}{3}) = c_0 + (\frac{1}{3}) c_1 + (\frac{1}{3})^2 c_2 + (\frac{1}{3})^3 c_3$$

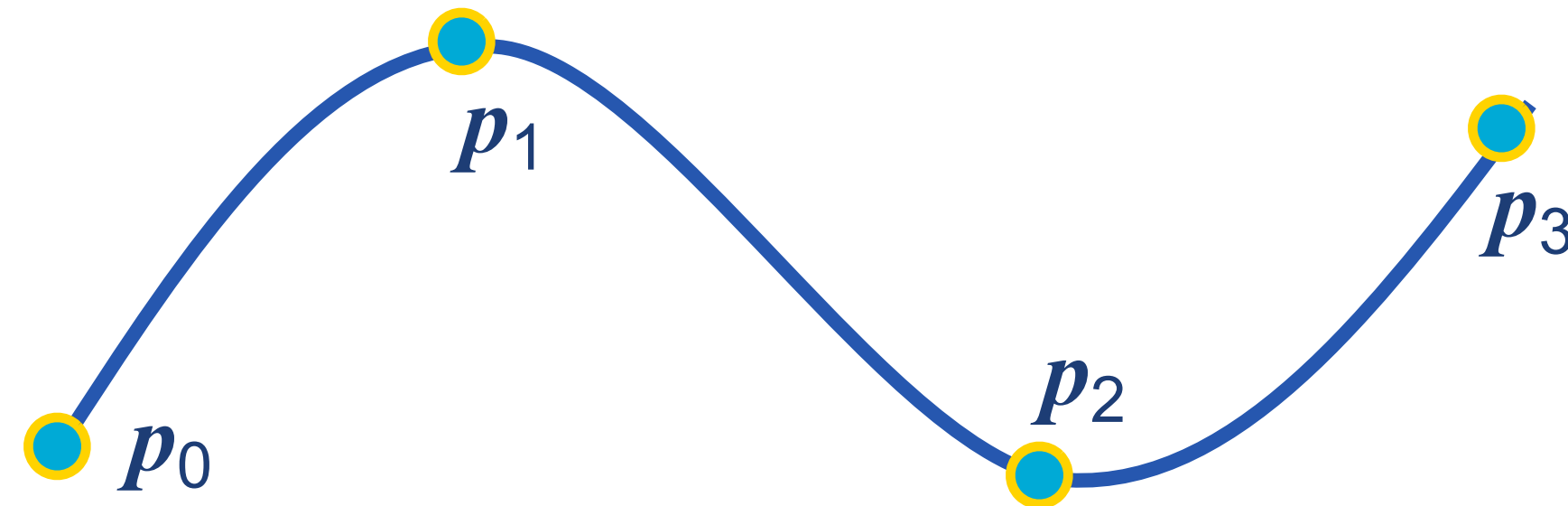
$$p_2 = P(\frac{2}{3}) = c_0 + (\frac{2}{3}) c_1 + (\frac{2}{3})^2 c_2 + (\frac{2}{3})^3 c_3$$

$$p_3 = P(1) = c_0 + c_1 + c_2 + c_3$$



Interpolating Cubic Curve

Find cubic $P(t)$ interpolating p_i , i.e., find $[c] = [c_0, c_1, c_2, c_3]^T$



Assume interpolation at $t = 0, \frac{1}{3}, \frac{2}{3}, 1$ (call these $t_0..t_3$)

(This is called uniform parameterization)

Other parameterizations are possible.

$$\begin{aligned} p_0 &= P(0) = c_0 \\ p_1 &= P(\frac{1}{3}) = c_0 + (\frac{1}{3})c_1 + (\frac{1}{3})^2 c_2 + (\frac{1}{3})^3 c_3 \\ p_2 &= P(\frac{2}{3}) = c_0 + (\frac{2}{3})c_1 + (\frac{2}{3})^2 c_2 + (\frac{2}{3})^3 c_3 \\ p_3 &= P(1) = c_0 + c_1 + c_2 + c_3 \end{aligned}$$

Matrix Form: $[p] = [A][c]$ Eq1

Solve for unknown $[c]$

where, $[p] = \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix}$

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \left(\frac{1}{3}\right) & \left(\frac{1}{3}\right)^2 & \left(\frac{1}{3}\right)^3 \\ 1 & \left(\frac{2}{3}\right) & \left(\frac{2}{3}\right)^2 & \left(\frac{2}{3}\right)^3 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$