Lecture 6:

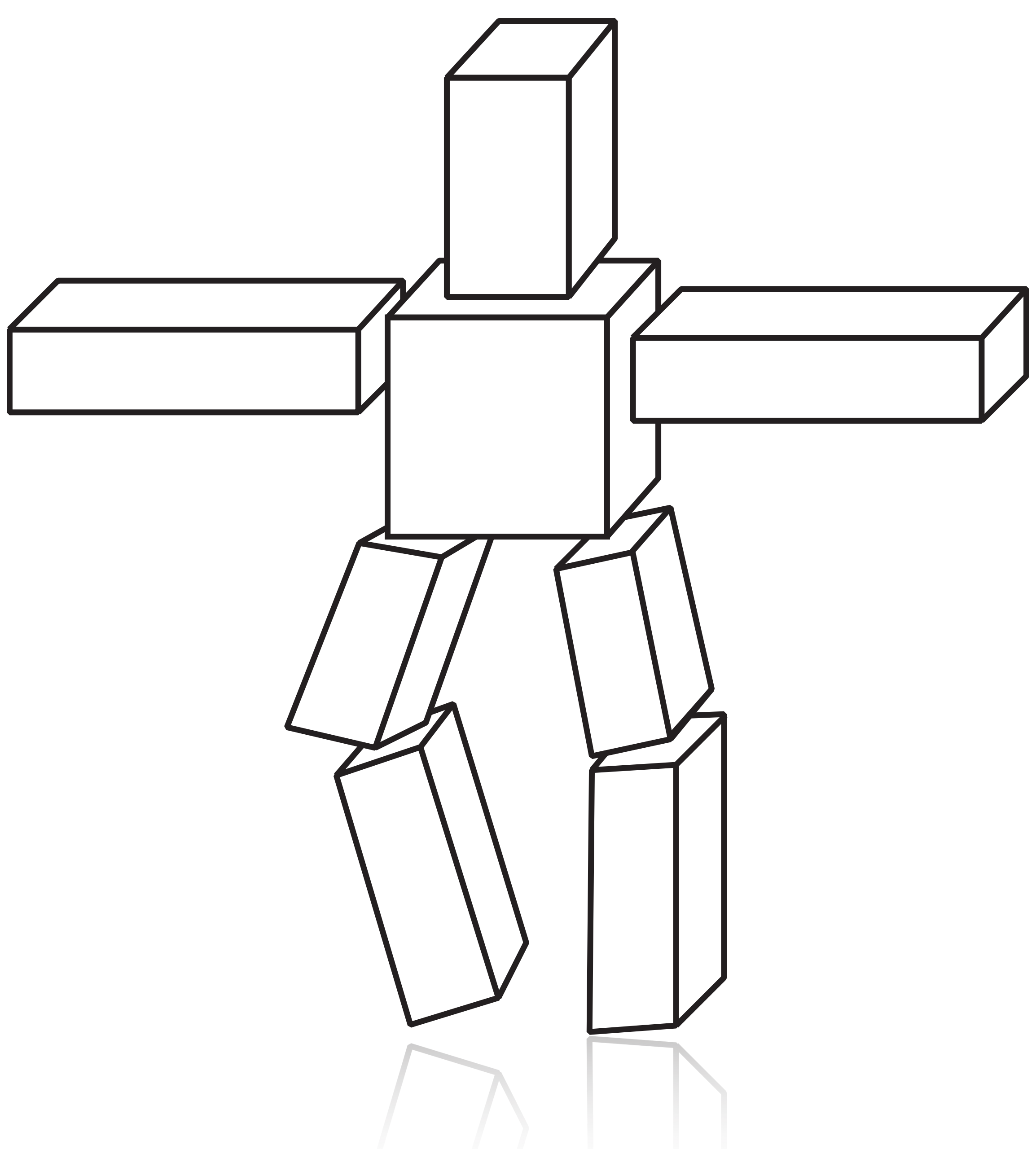
Introduction to Geometry

Computer Graphics

CMU 15-462/15-662, Fall 2015

Increasing the complexity of our models

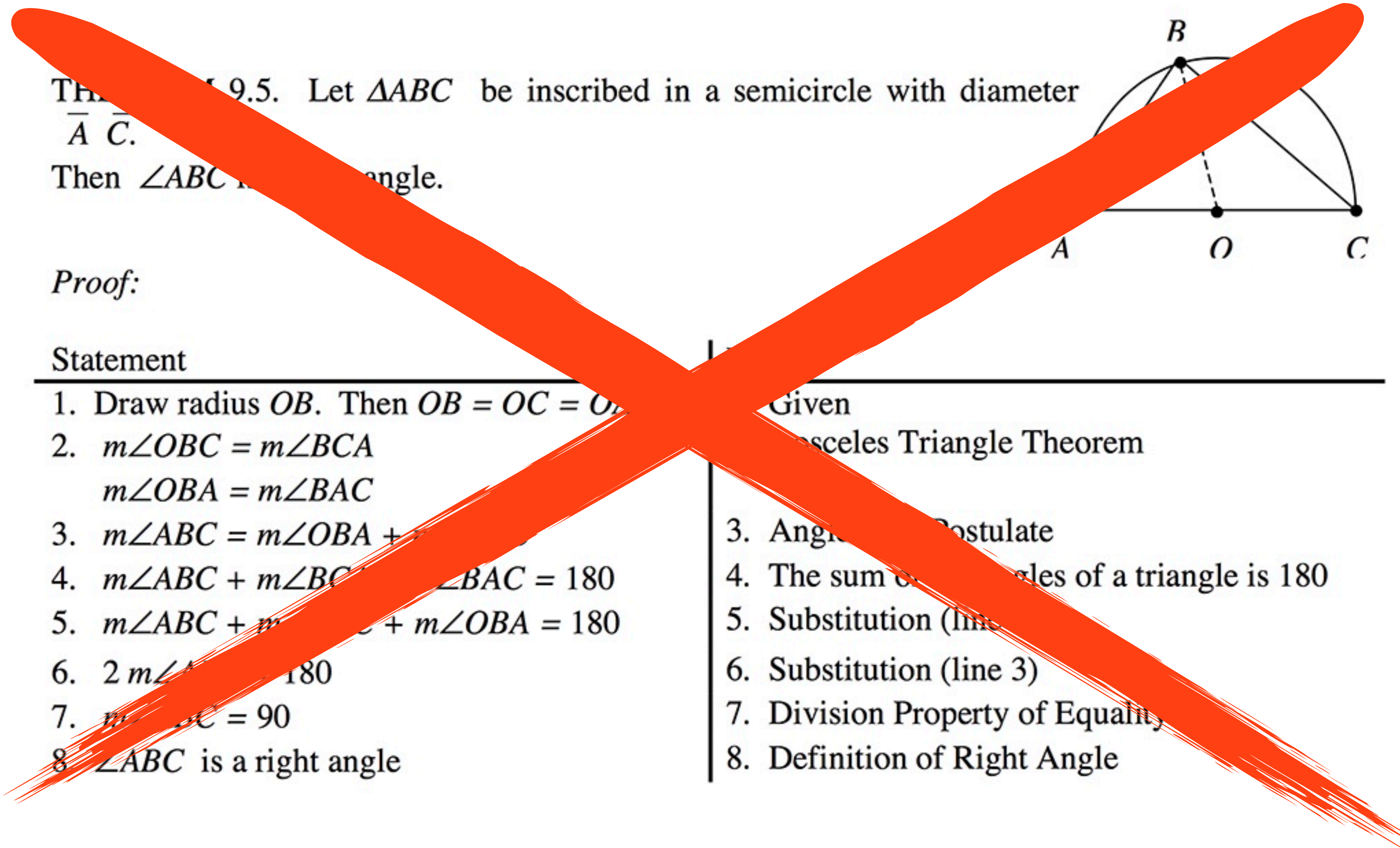
Transformations



Geometry

Materials, lighting, ...

What is geometry?



TH[illegible] 9.5. Let ΔABC be inscribed in a semicircle with diameter $\overline{A}\ \overline{C}$.
Then $\angle ABC$ [illegible] angle.

Proof:

Statement	[illegible]
1. Draw radius OB. Then $OB = OC = O$[illegible]	[illegible] Given
2. $m\angle OBC = m\angle BCA$ $m\angle OBA = m\angle BAC$	[illegible]sceles Triangle Theorem
3. $m\angle ABC = m\angle OBA +$ [illegible]	3. Angl[illegible]ostulate
4. $m\angle ABC + m\angle BC$[illegible]$BAC = 180$	4. The sum [illegible]gles of a triangle is 180
5. $m\angle ABC +$ [illegible] $+ m\angle OBA = 180$	5. Substitution (li[illegible]
6. $2\,m\angle$[illegible] 180	6. Substitution (line 3)
7. [illegible]$C = 90$	7. Division Property of Equalit[illegible]
8 $\angle ABC$ is a right angle	8. Definition of Right Angle

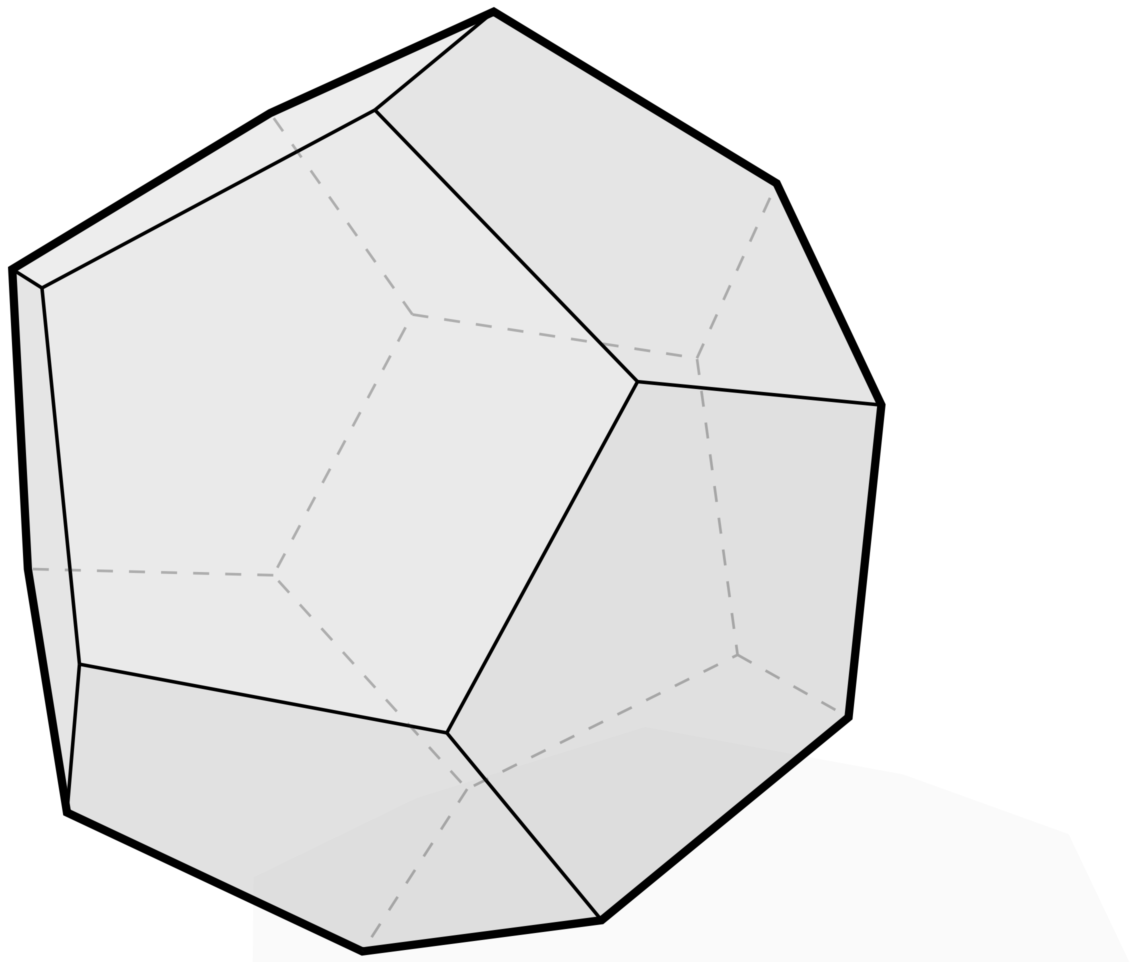
Ceci n'est pas géométrie.

What is geometry?

"Earth" "measure"

ge•om•et•ry /jēˈämətrē/ *n.*

1. The study of shapes, sizes, patterns, and positions.
2. The study of spaces where some quantity (lengths, angles, etc.) can be *measured*.

Plato: "...the earth is in appearance like one of those balls which have leather coverings in twelve pieces..."

How can we describe geometry?

IMPLICIT

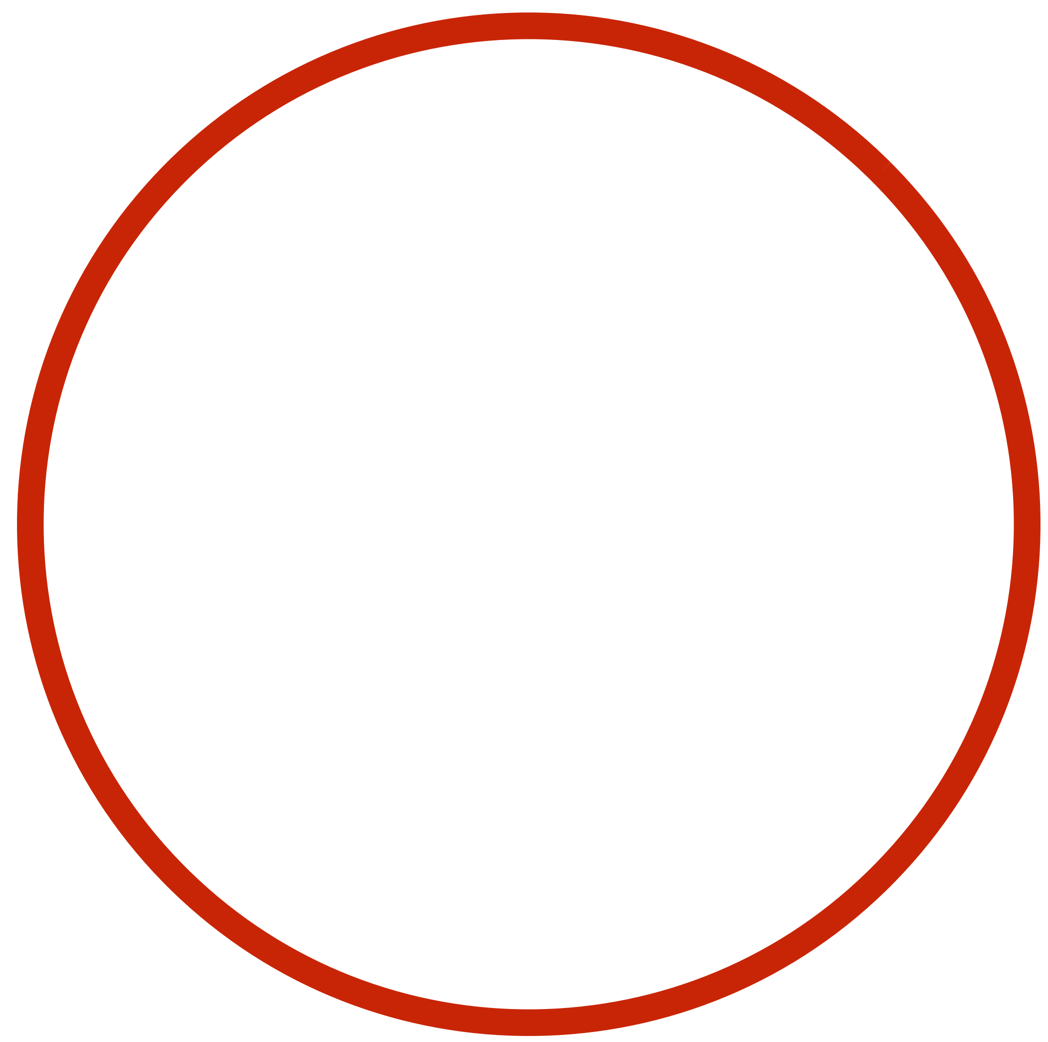
$$x^2 + y^2 = 1$$

LINGUISTIC

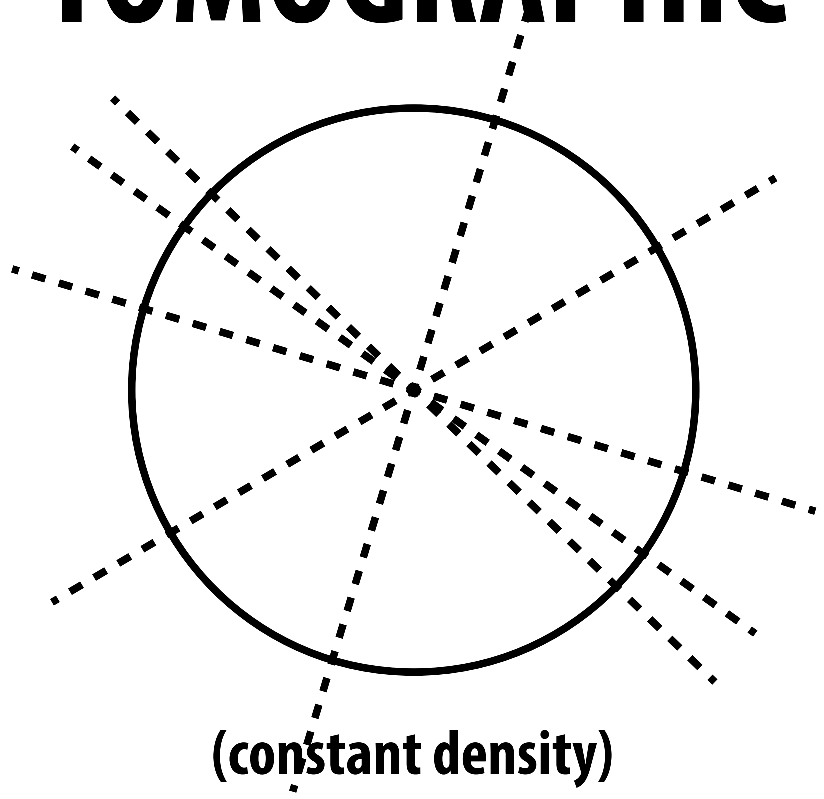
"unit circle"

EXPLICIT

$$(\underbrace{\cos\theta}_{x}, \underbrace{\sin\theta}_{y})$$

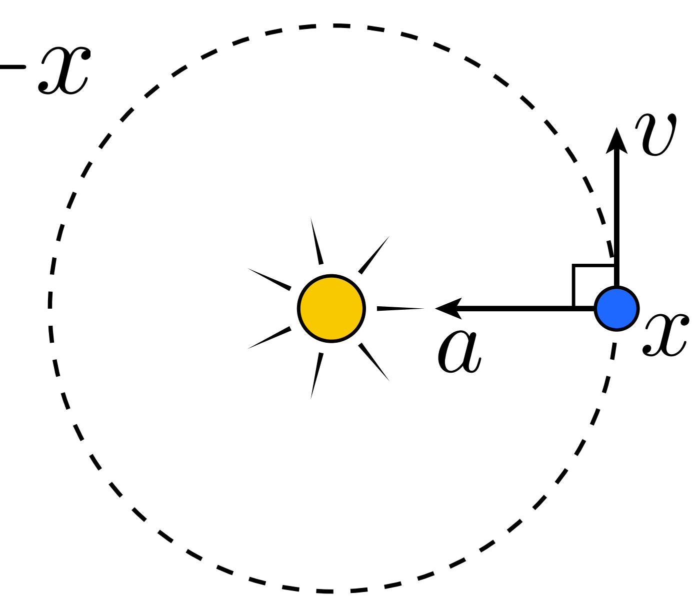


TOMOGRAPHIC

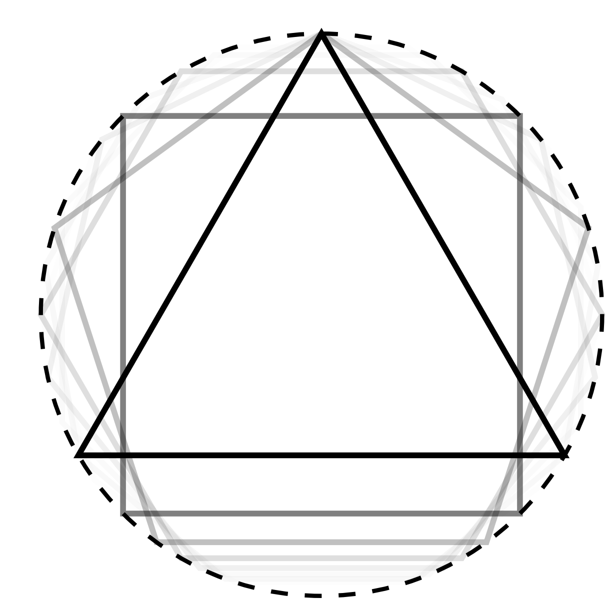


DYNAMIC

$$\frac{d^2}{dt^2}x = -x$$



DISCRETE

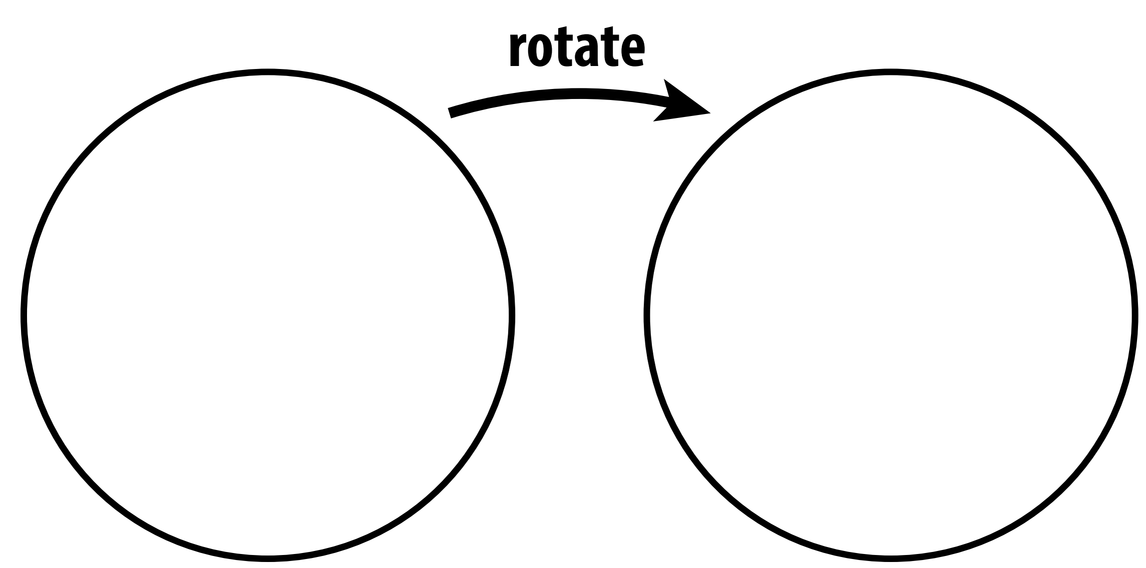


$n \to \infty$

CURVATURE

$$\kappa = 1$$

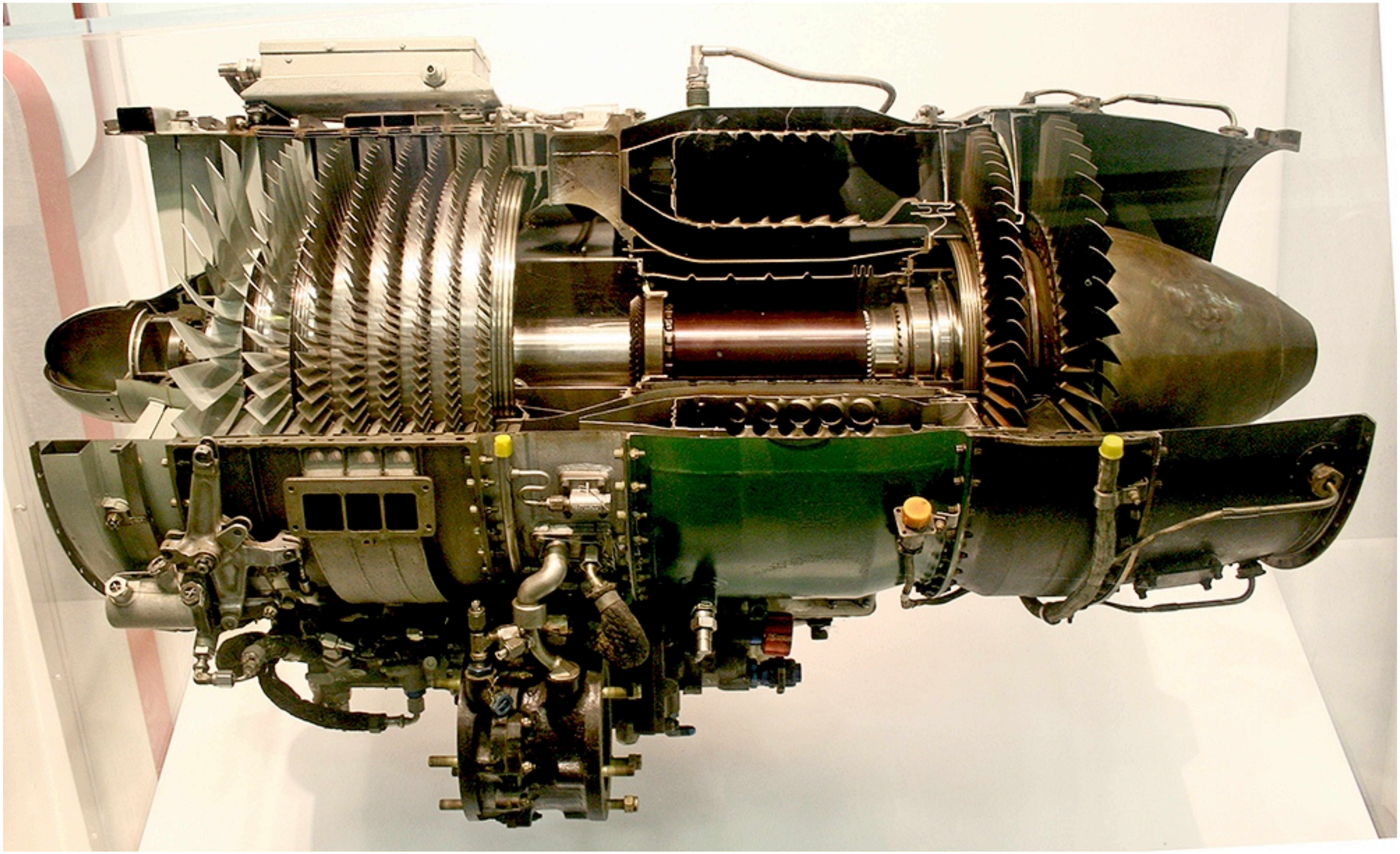
SYMMETRIC



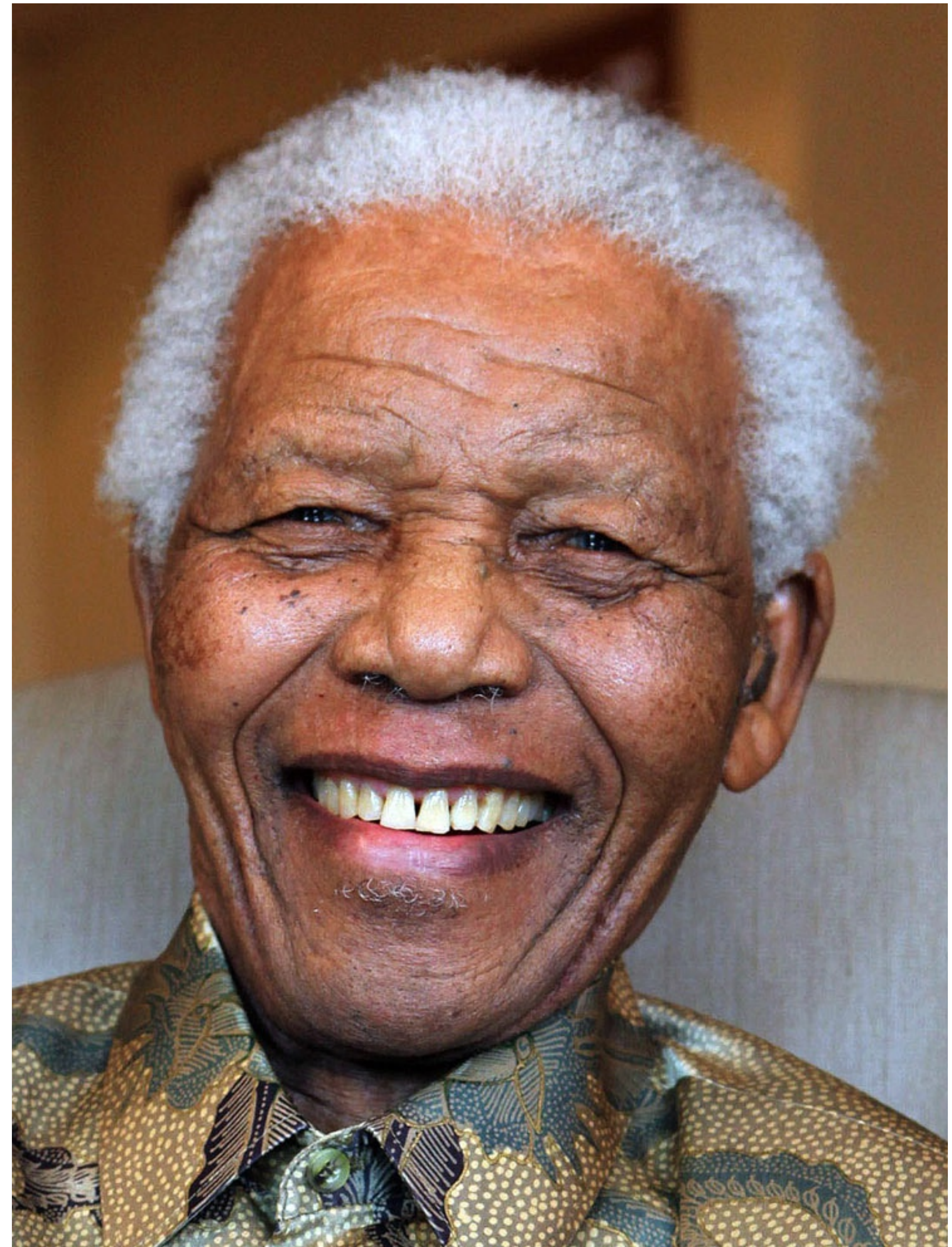
Given all these options, what's the best way to encode geometry on a computer?

Examples of geometry

Examples of geometry



Examples of geometry



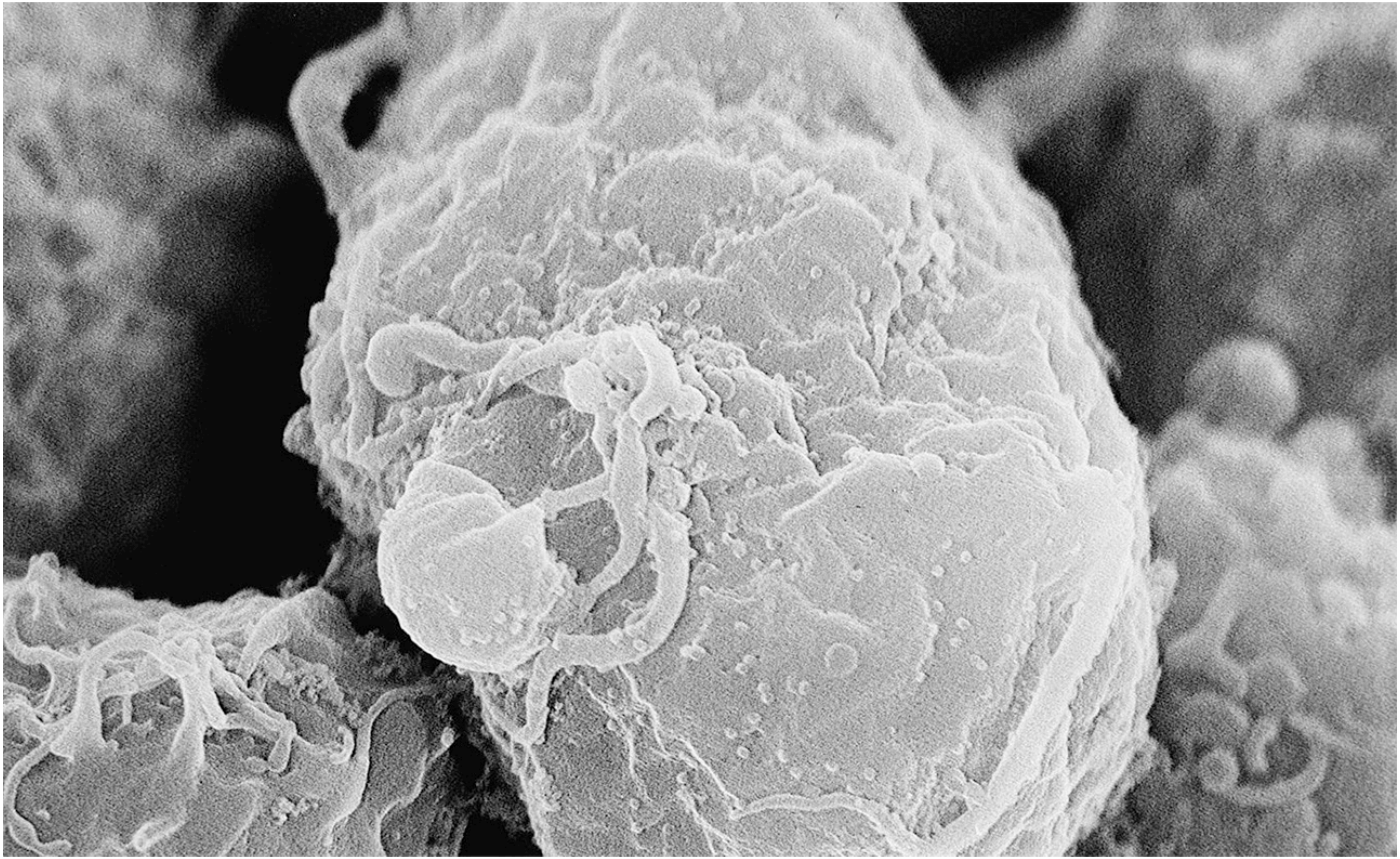
Examples of geometry

Examples of geometry

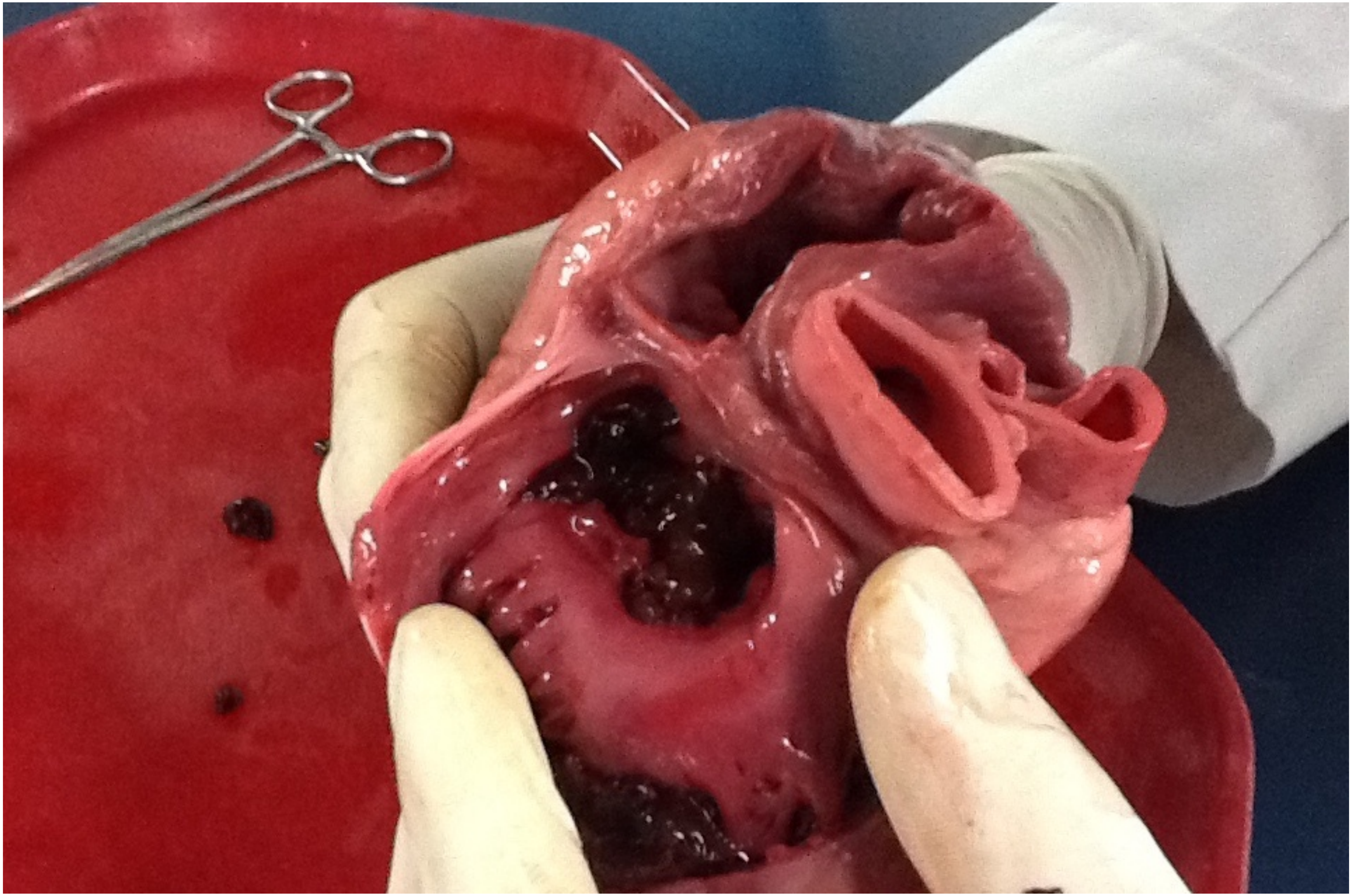
Examples of geometry

Examples of geometry

Examples of geometry



Examples of geometry



Examples of geometry

No one "best" choice—geometry is hard!

"I hate meshes.
I cannot believe how hard this is.
Geometry is hard."

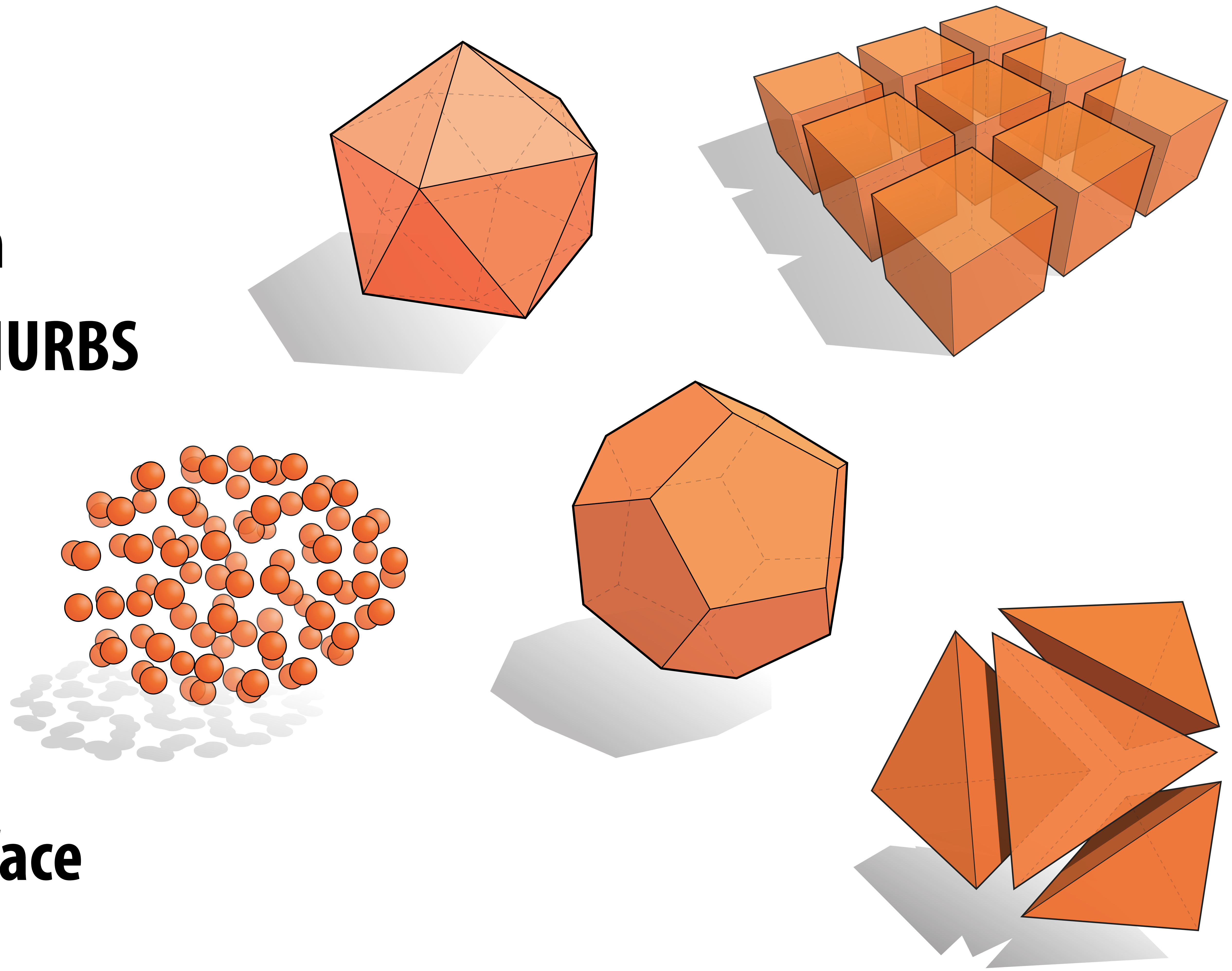
—David Baraff

Senior Research Scientist
Pixar Animation Studios

Slide cribbed from Jeff Erickson.

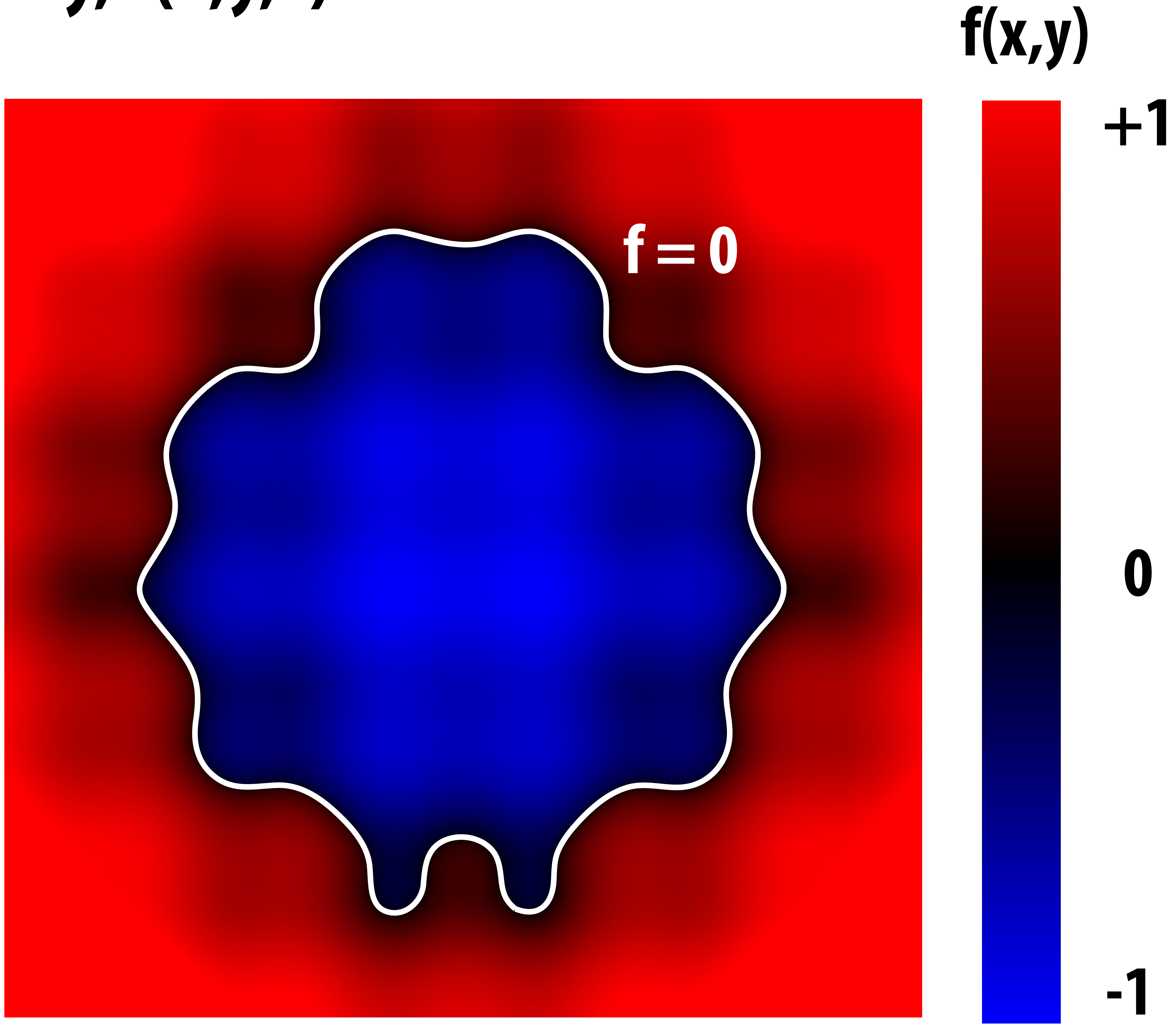
Many ways to digitally encode geometry

- **EXPLICIT**
  - point cloud
  - polygon mesh
  - subdivision, NURBS
  - L-systems
  - ...
- **IMPLICIT**
  - level set
  - algebraic surface
  - ...
- **Each choice best suited to a different task/type of geometry**



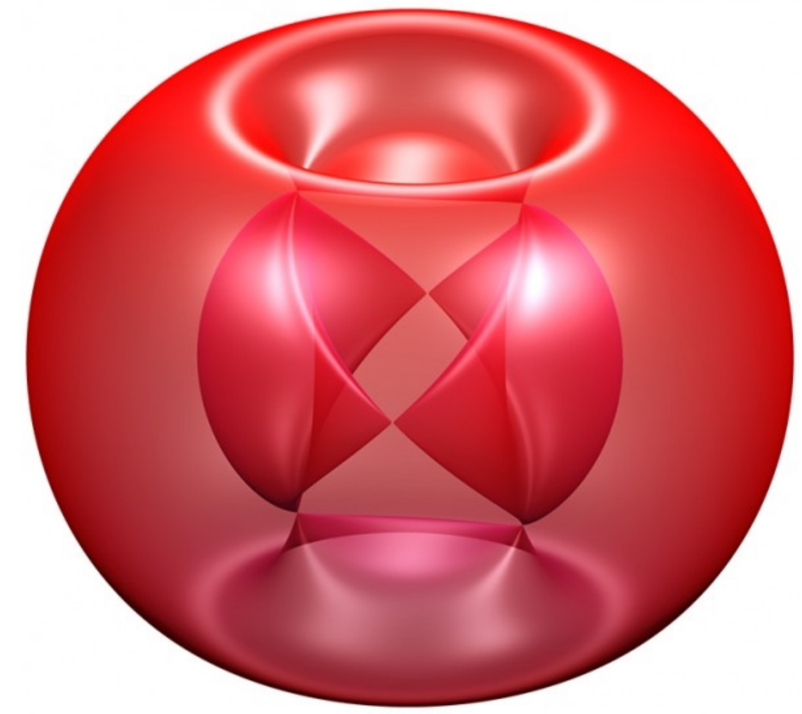
"Implicit" Representations of Geometry

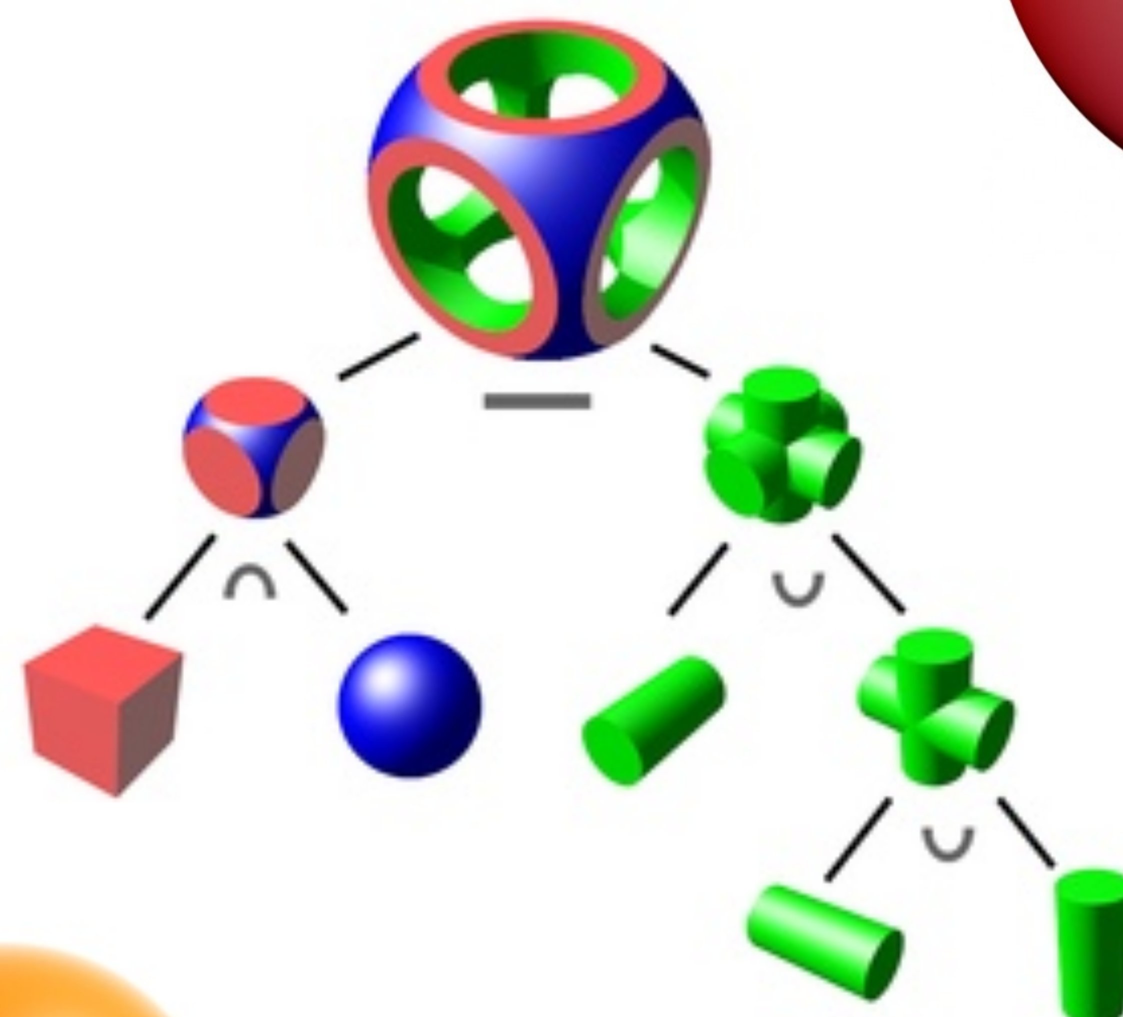
- Points aren't known directly, but satisfy some relationship
- E.g., unit sphere is all points x such that $x^2+y^2+z^2=1$
- More generally, $f(x,y,z) = 0$



Many implicit representations in graphics

- algebraic surfaces
- constructive solid geometry
- level set methods
- blobby surfaces
- fractals
- ...





(Will see some of these a bit later.)

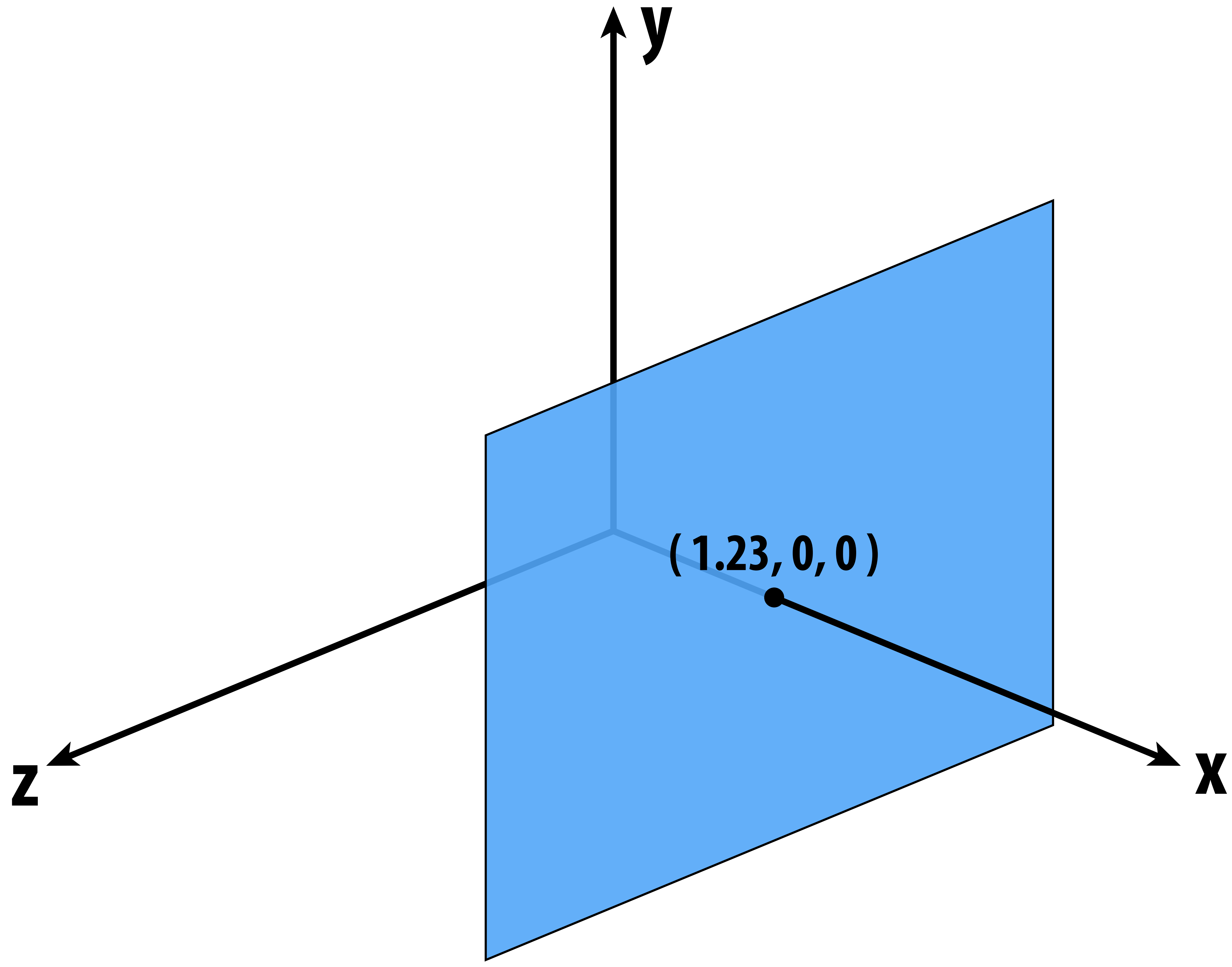
But first, let's play a game:

I'm thinking of an implicit surface $f(x,y,z)=0$.

Find *any* point on it.

Give up?

My function was f(x,y,z) = -1.23 (a plane):



Implicit surfaces make some tasks hard (like sampling).

Let's play another game.

I have a new surface $f(x,y,z) = x^2 + y^2 + z^2 - 1$

I want to see if a point is *inside* it.

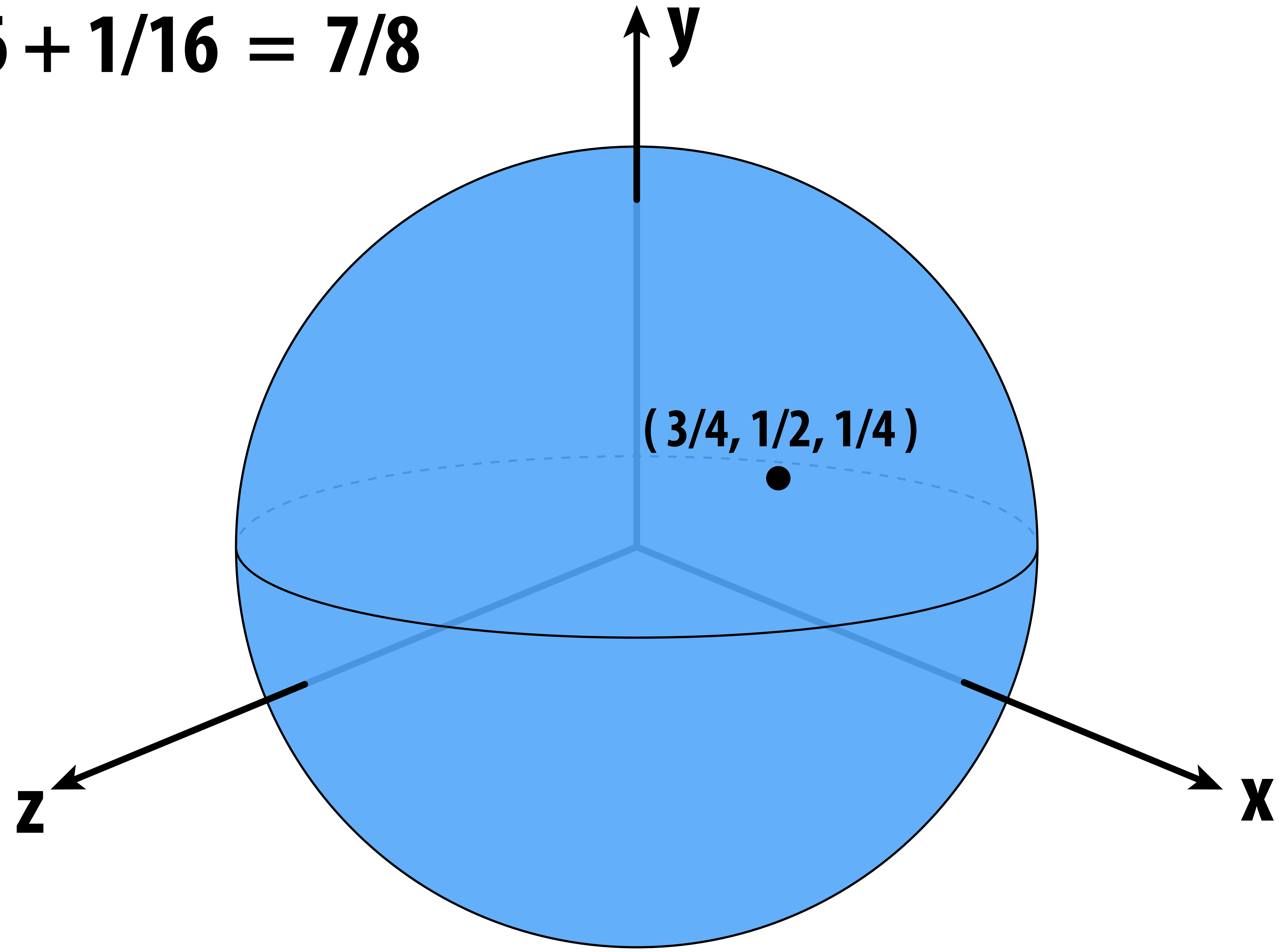
Check if this point is inside the unit sphere

How about the point (3/4, 1/2, 1/4)?

9/16 + 4/16 + 1/16 = 7/8

7/8 < 1

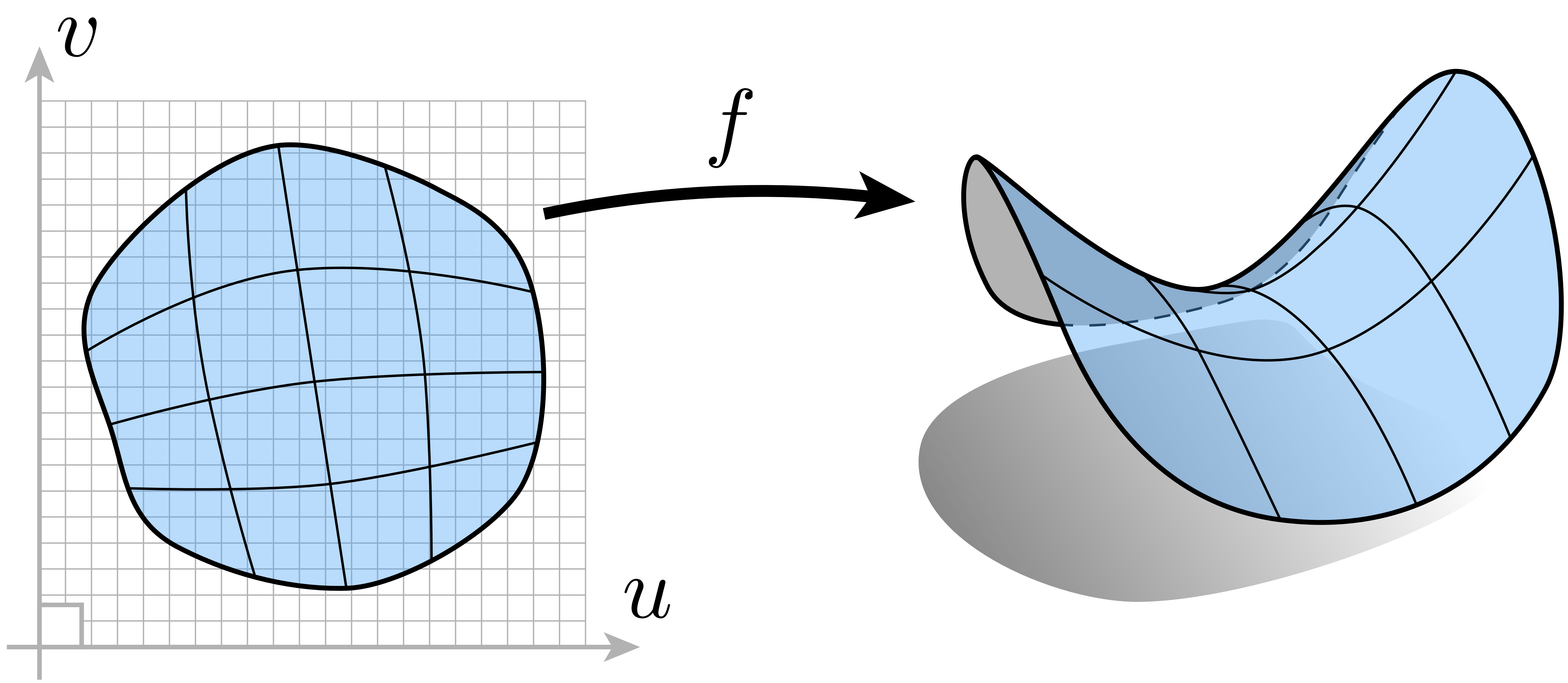
YES.



Implicit surfaces make other tasks easy (like inside/outside tests).

"Explicit" Representations of Geometry

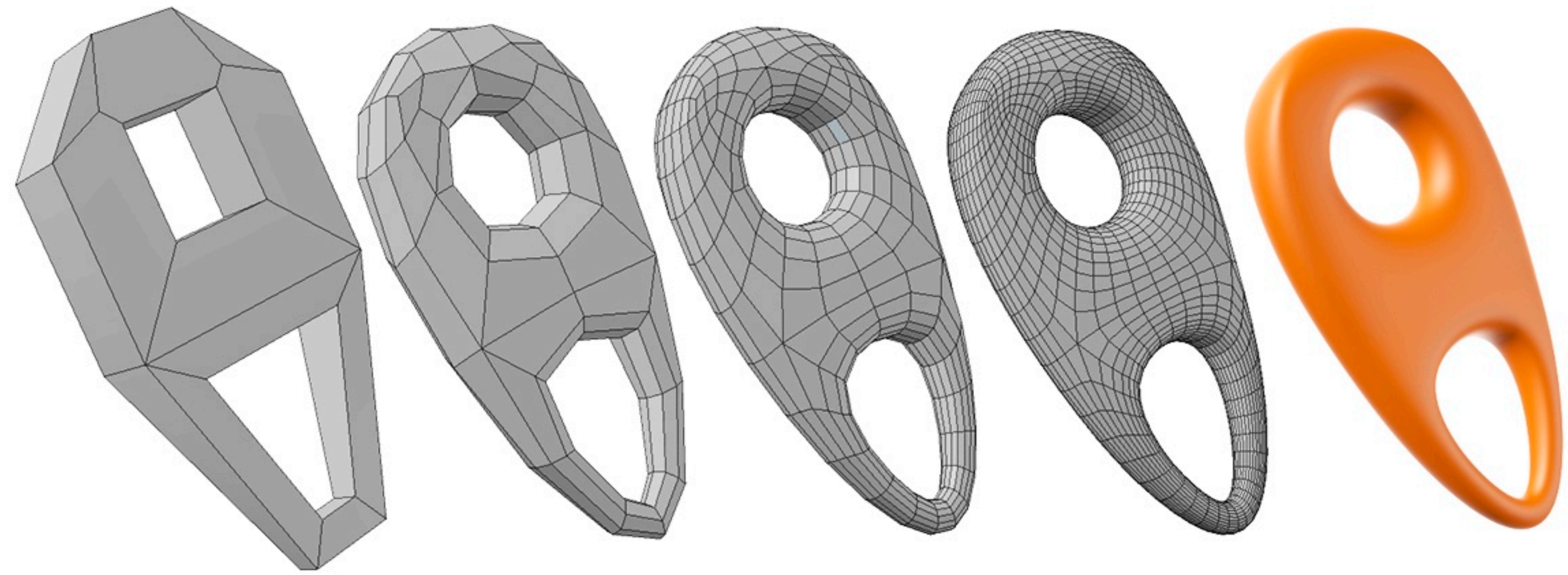
- **All points are given directly**
- **E.g., points on sphere are** $(\cos(u)\sin(v), \sin(u)\sin(v), \cos(v)),$
  $\text{for } 0 \leq u < 2\pi \text{ and } 0 \leq v \leq \pi$
- **More generally:** $f : \mathbb{R}^2 \to \mathbb{R}^3; (u, v) \mapsto (x, y, z)$

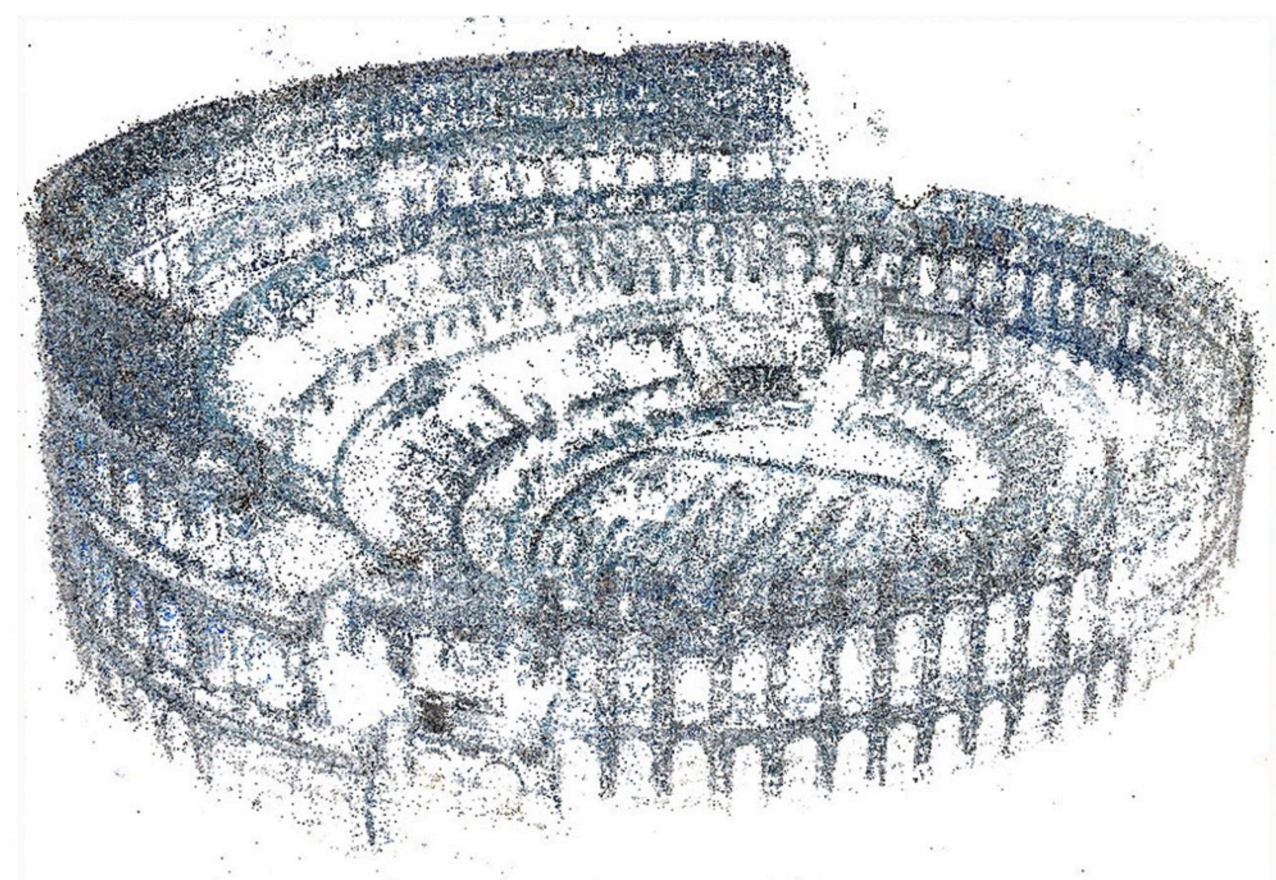


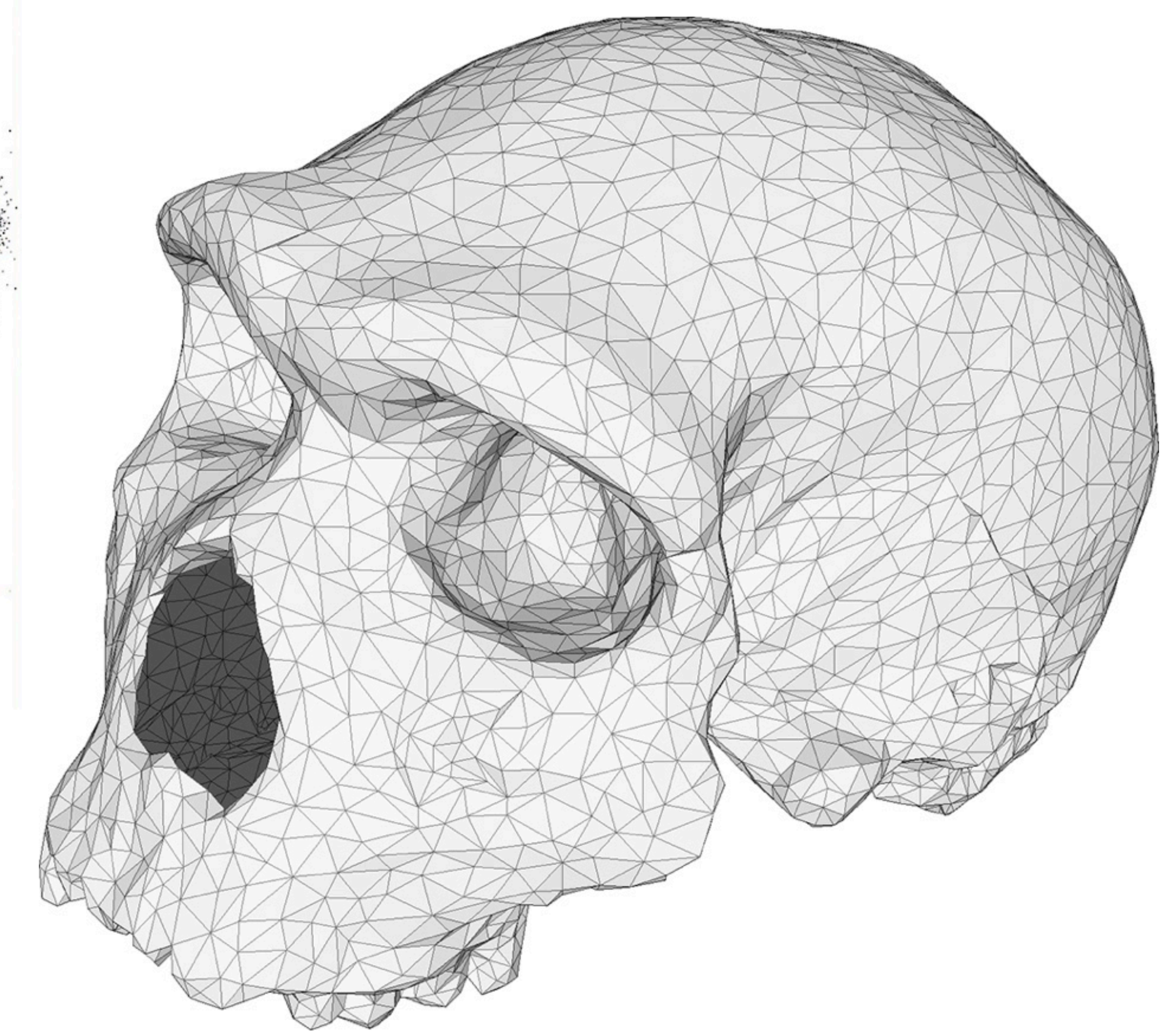
- **(Might have a bunch of these maps, e.g., one per triangle.)**

Many explicit representations in graphics

- triangle meshes
- polygon meshes
- subdivision surfaces
- NURBS
- point clouds
- ...







(Will see some of these a bit later.)

But first, let's play a game:

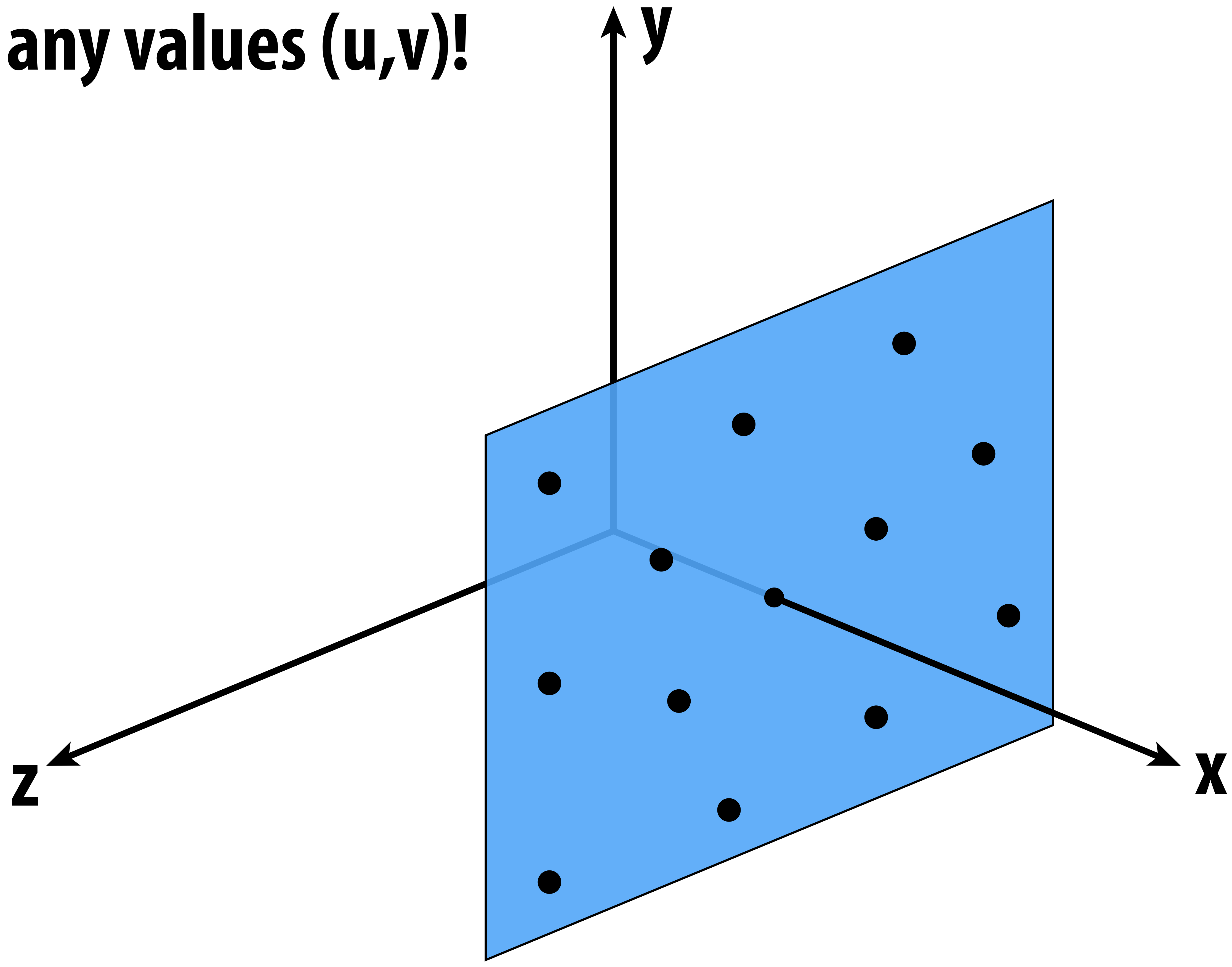
I'll give you an explicit surface.

You give me some points on it.

Sampling an explicit surface

My surface is f(u, v) = (1.23, u, v).

Just plug in any values (u,v)!



Explicit surfaces make some tasks easy (like sampling).

Let's play another game.

I have a new surface f(u,v).

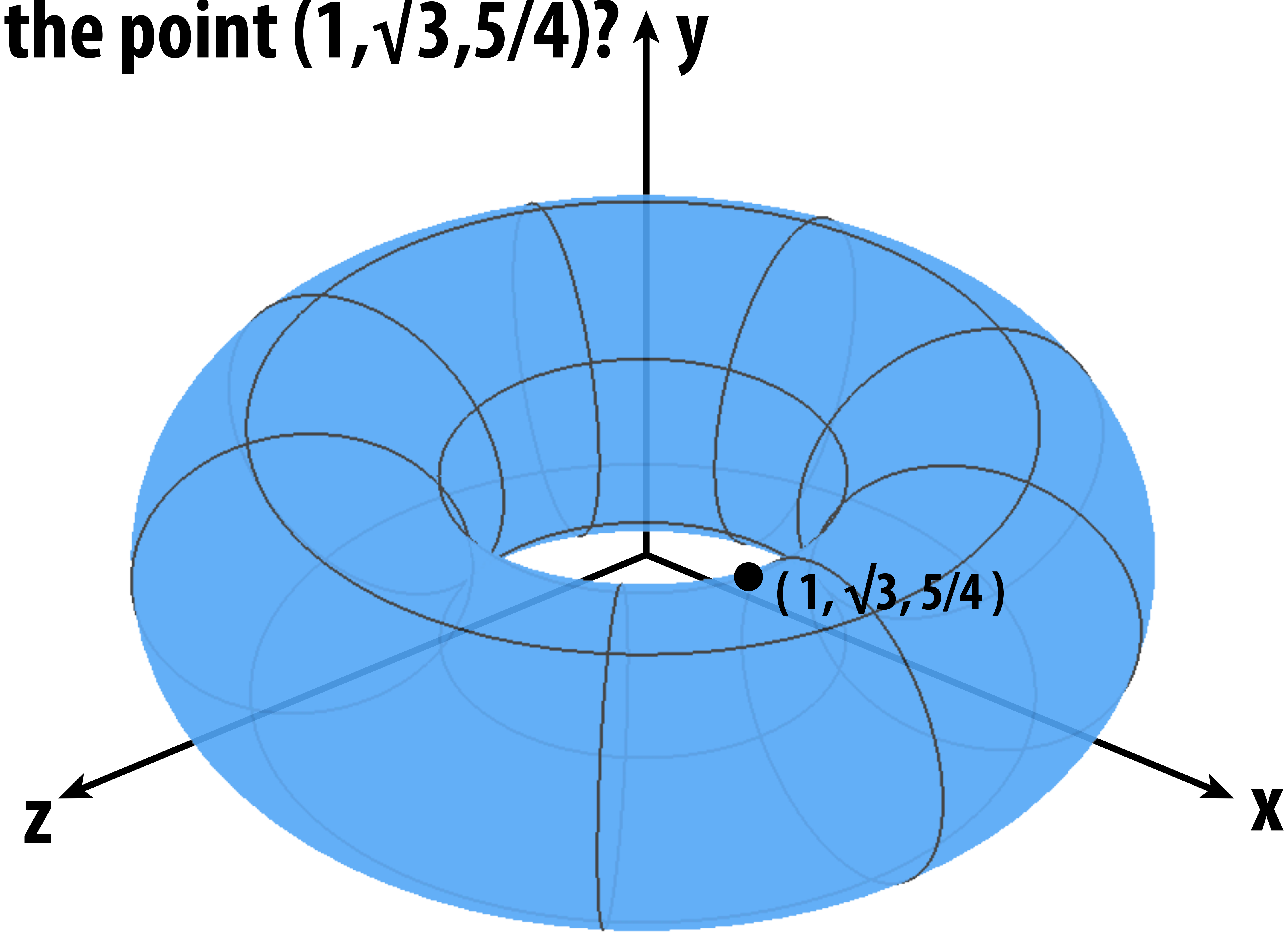
I want to see if a point is *inside* it.

Check if this point is inside the torus

My surface is $f(u,v) = (\,2+\cos(u))\cos(v),\, 2+\cos(u))\sin(v),\, \sin(u)\,)$

How about the point $(1,\sqrt{3},5/4)$?

...NO!



Explicit surfaces make other tasks hard (like inside/outside tests).

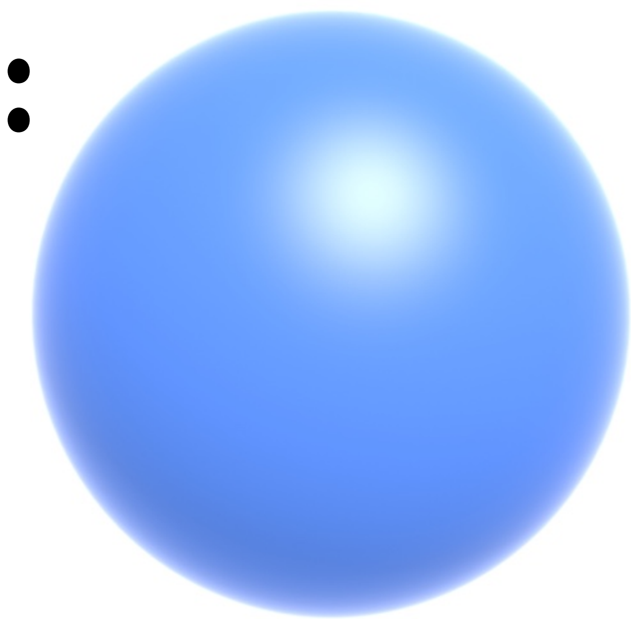
CONCLUSION:
Some representations work better than others—depends on the task!

Different representations will also be better suited to different types of geometry.

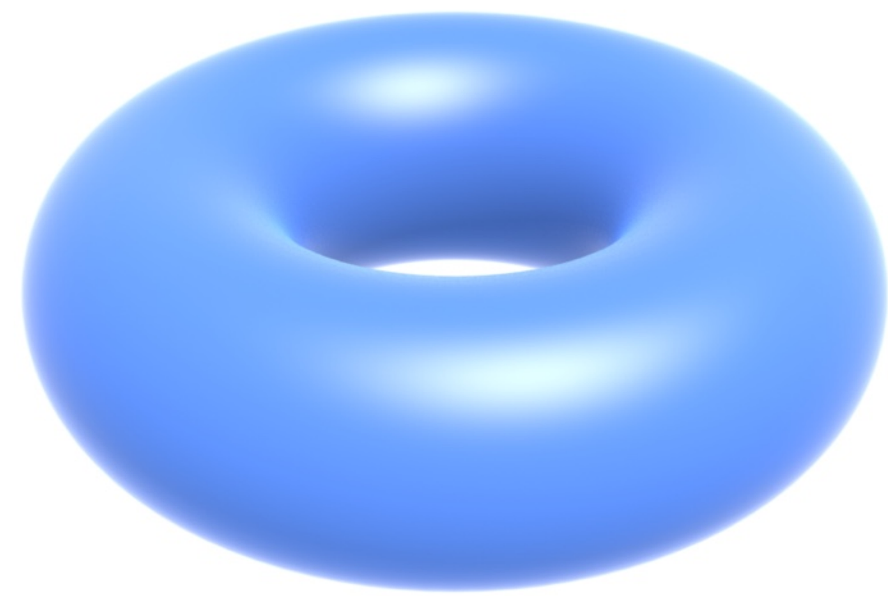
Let's take a look at some common representations used in computer graphics.

Algebraic Surfaces (Implicit)

- **Surface is zero set of a polynomial in x, y, z ("algebraic variety")**
- **Examples:**



$$x^2+y^2+z^2=1$$



$$(R-\sqrt{x^2+y^2})^2+z^2=r^2$$

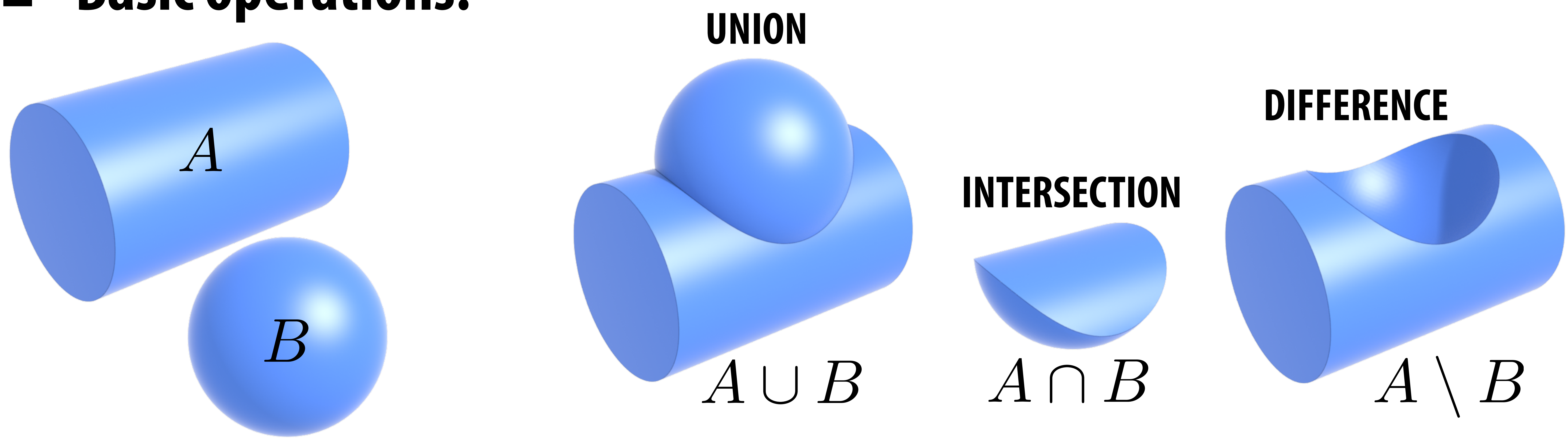
$$(x^2+\frac{9y^2}{4}+z^2-1)^3 = x^2z^3+\frac{9y^2z^3}{80}$$

- **What about more complicated shapes?**

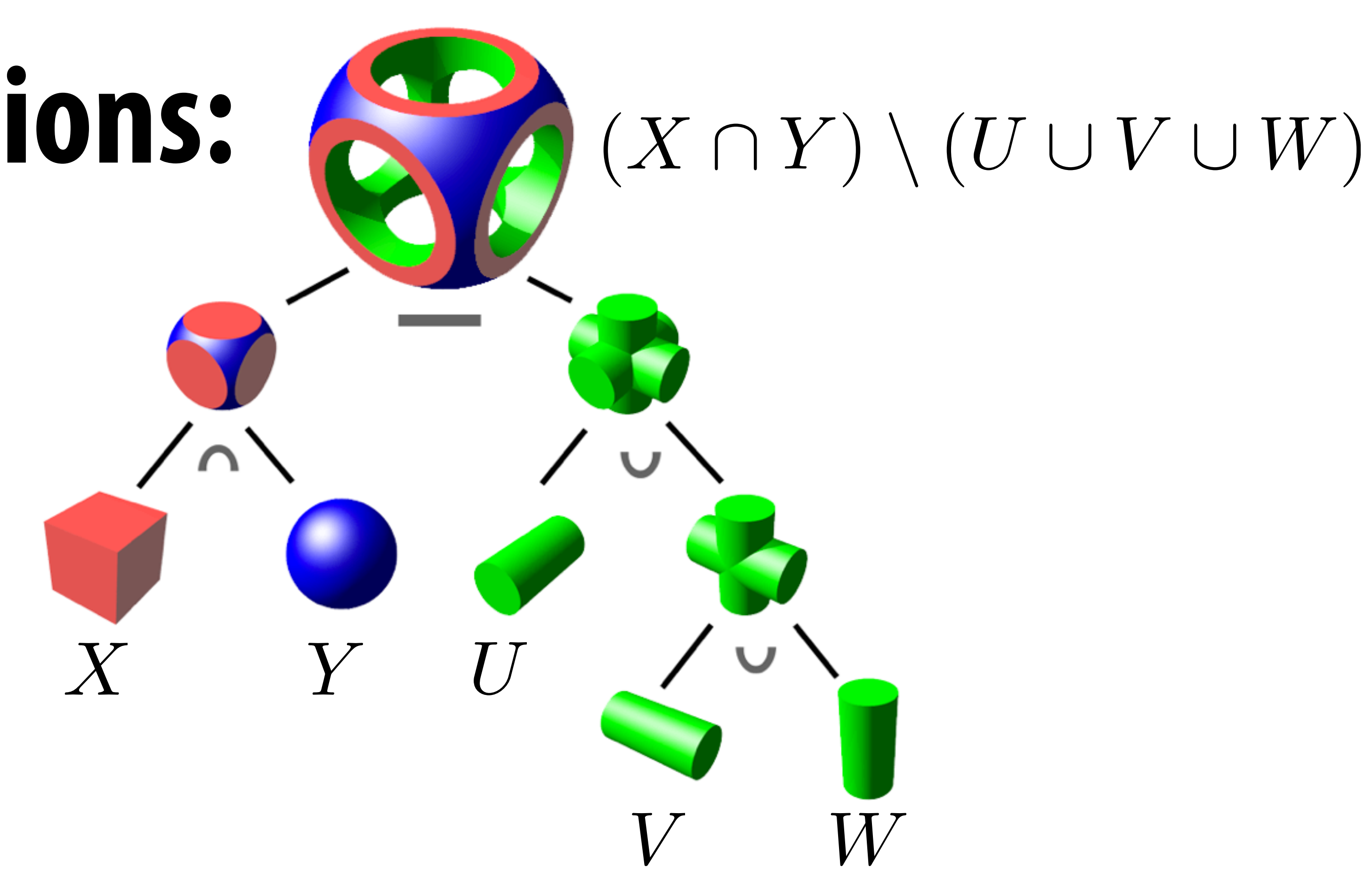
- **Very hard to come up with polynomials!**

Constructive Solid Geometry (Implicit)

- Build more complicated shapes via Boolean operations
- Basic operations:

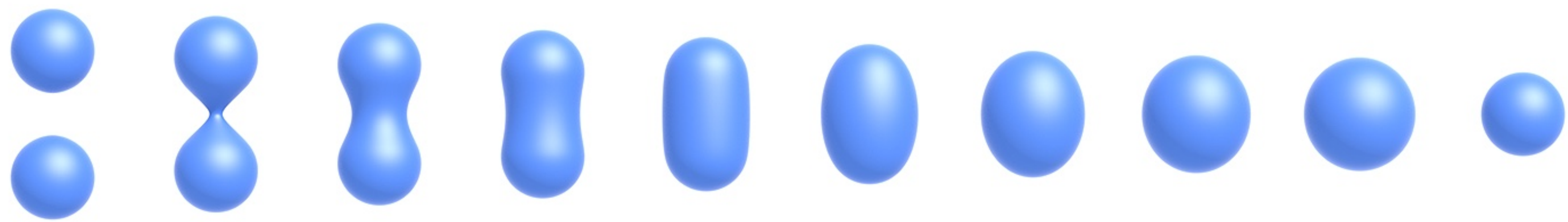


- Then chain together expressions:

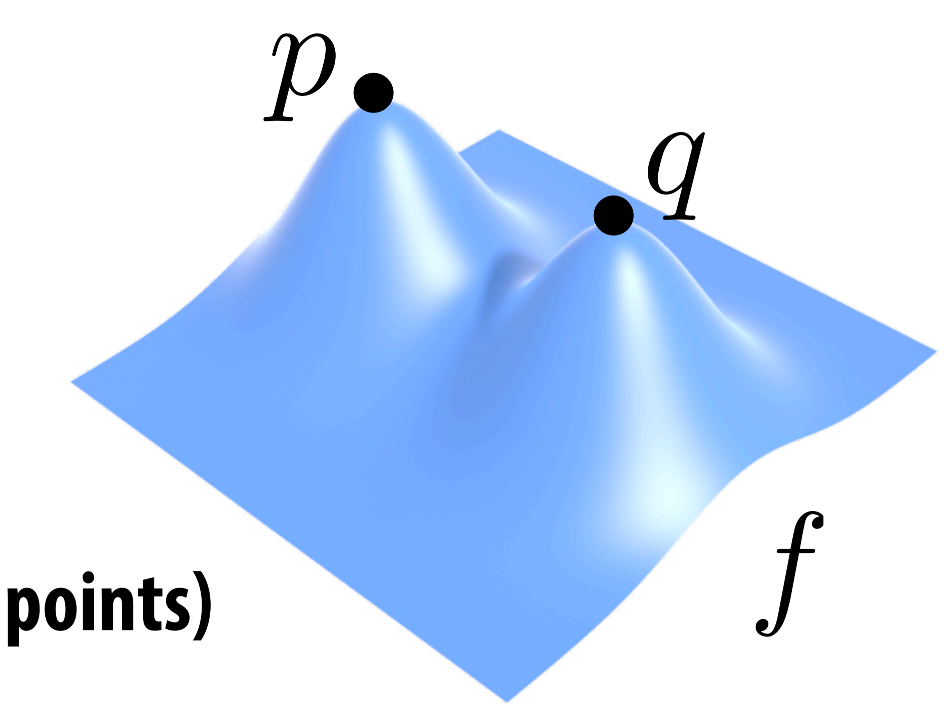


Blobby Surfaces (Implicit)

- **Instead of Booleans, gradually blend surfaces together:**

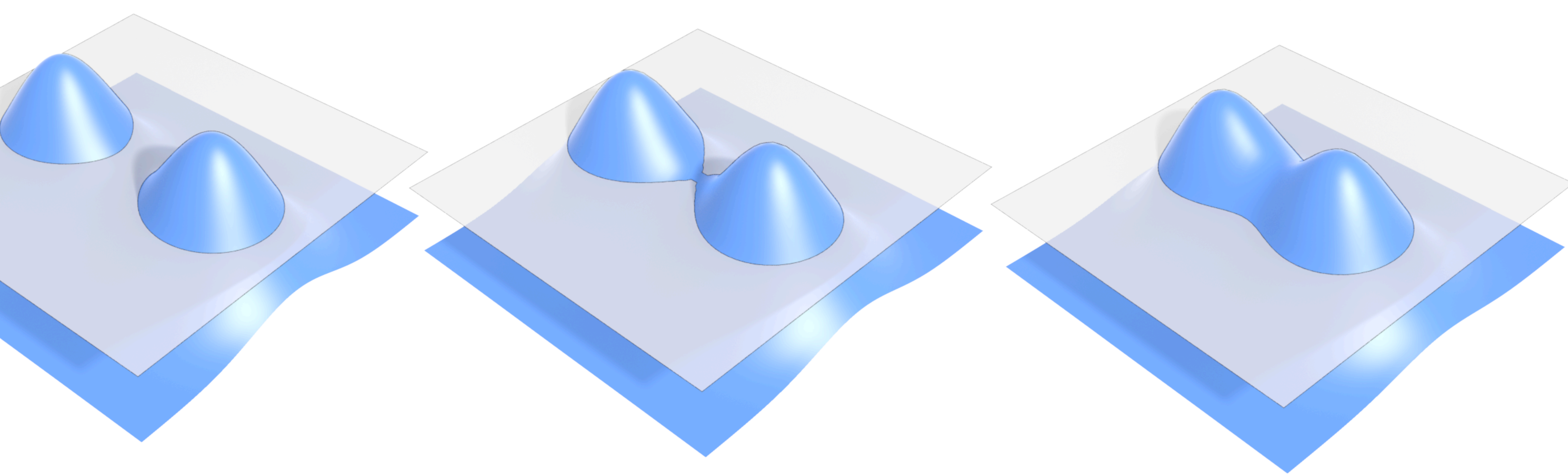


- **Easier to understand in 2D:**



$\phi_p(x) := e^{-|x-p|^2}$ **(Gaussian centered at p)**

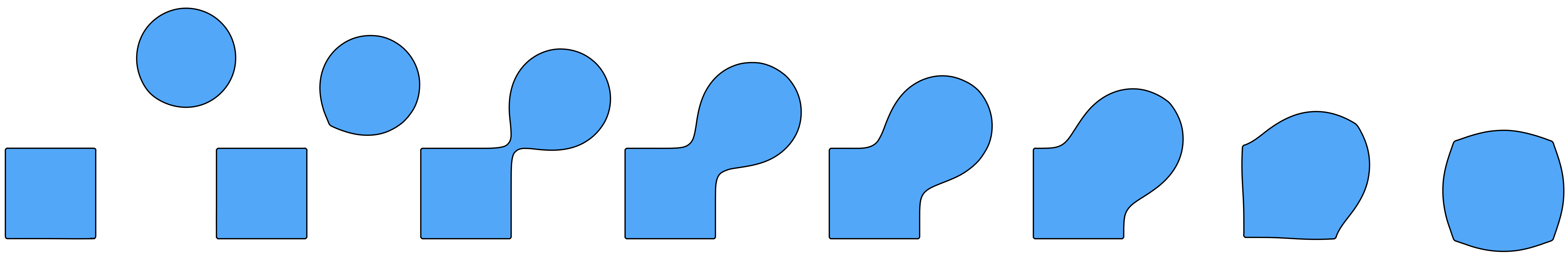
$f := \phi_p + \phi_q$ **(Sum of Gaussians centered at different points)**



$f = \frac{1}{2}$

Blending Distance Functions (Implicit)

- A *distance function* gives distance to closest point on object
- Can blend any two distance functions d_1, d_2:



- Similar strategy to points, though many possibilities. E.g.,

$$f(x) := e^{-d_1(x)^2} + e^{-d_2(x)^2} - \frac{1}{2}$$

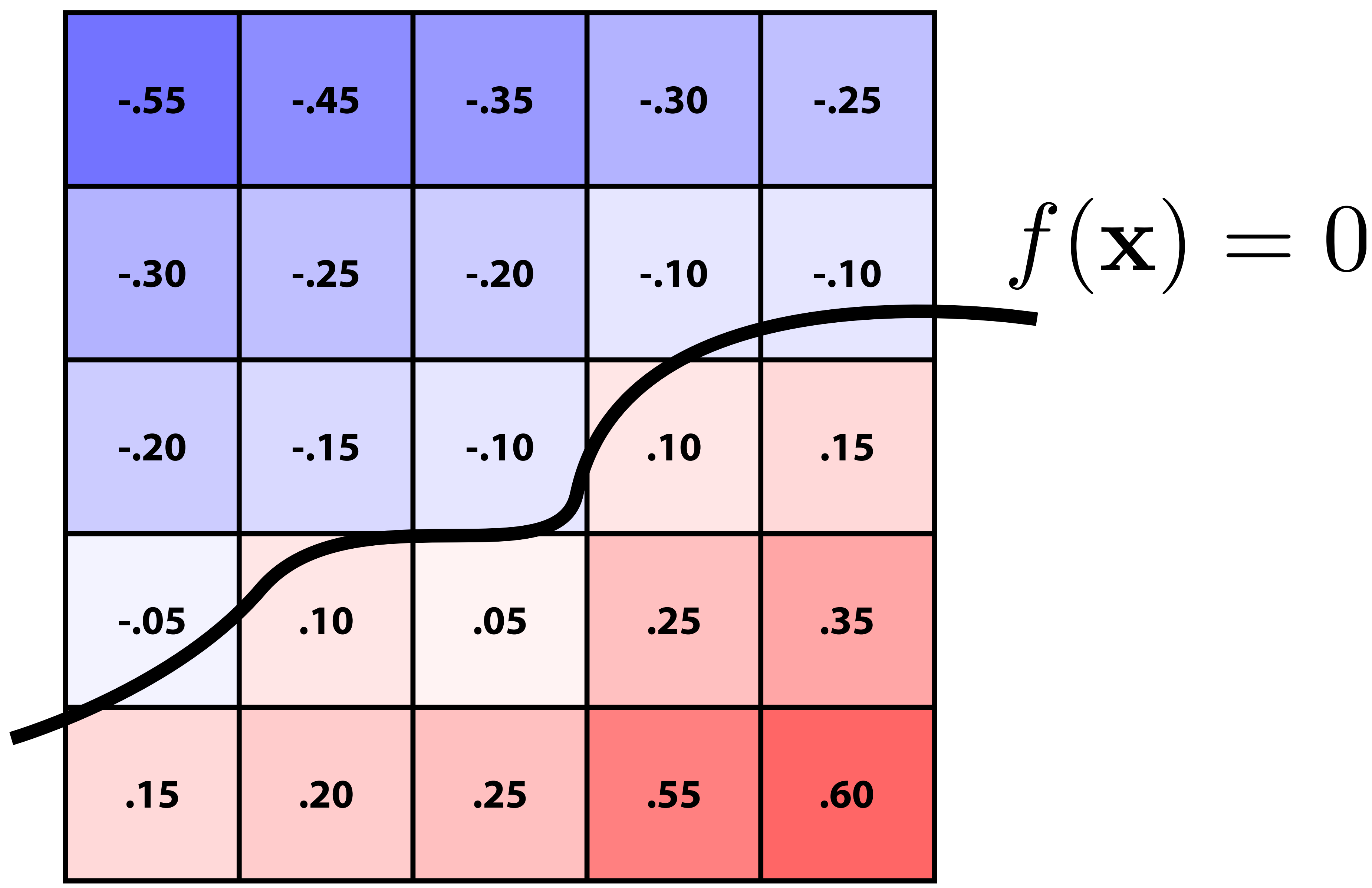
- Appearance depends on exactly how we combine functions
- Q: How do we implement a simple Boolean union?
- A: Just take the product: $f(x) := d_1(x)d_2(x)$

Scene of pure distance functions (not easy!)

See http://iquilezles.org/www/material/nvscene2008/nvscene2008.htm

Level Set Methods (Implicit)

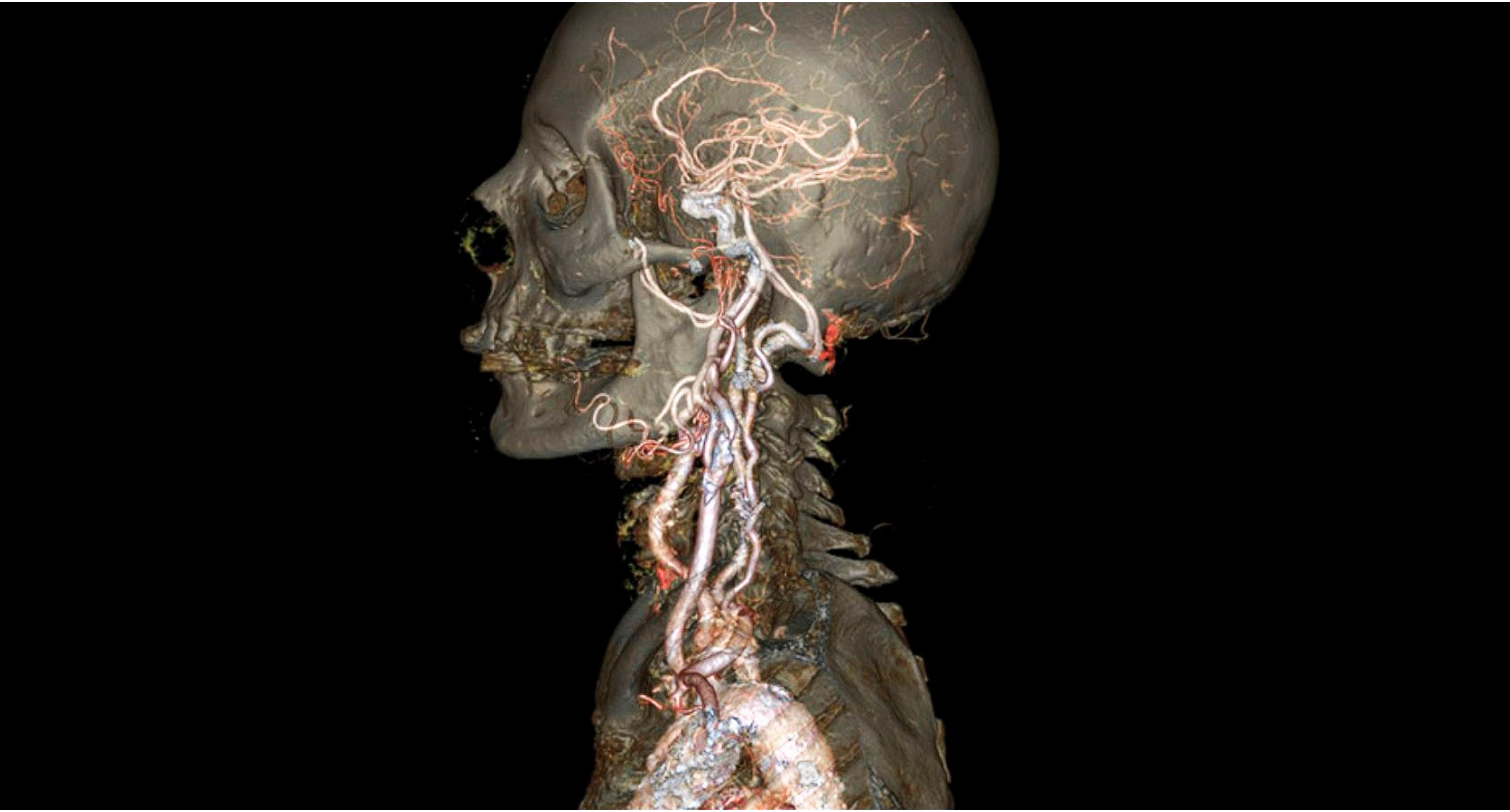
- Implicit surfaces have some nice features (e.g., merging/splitting)
- But, hard to describe complex shapes in closed form
- Alternative: store a grid of values approximating function



- Surface is found where *interpolated* values equal zero
- Provides much more explicit control over shape (like a texture)

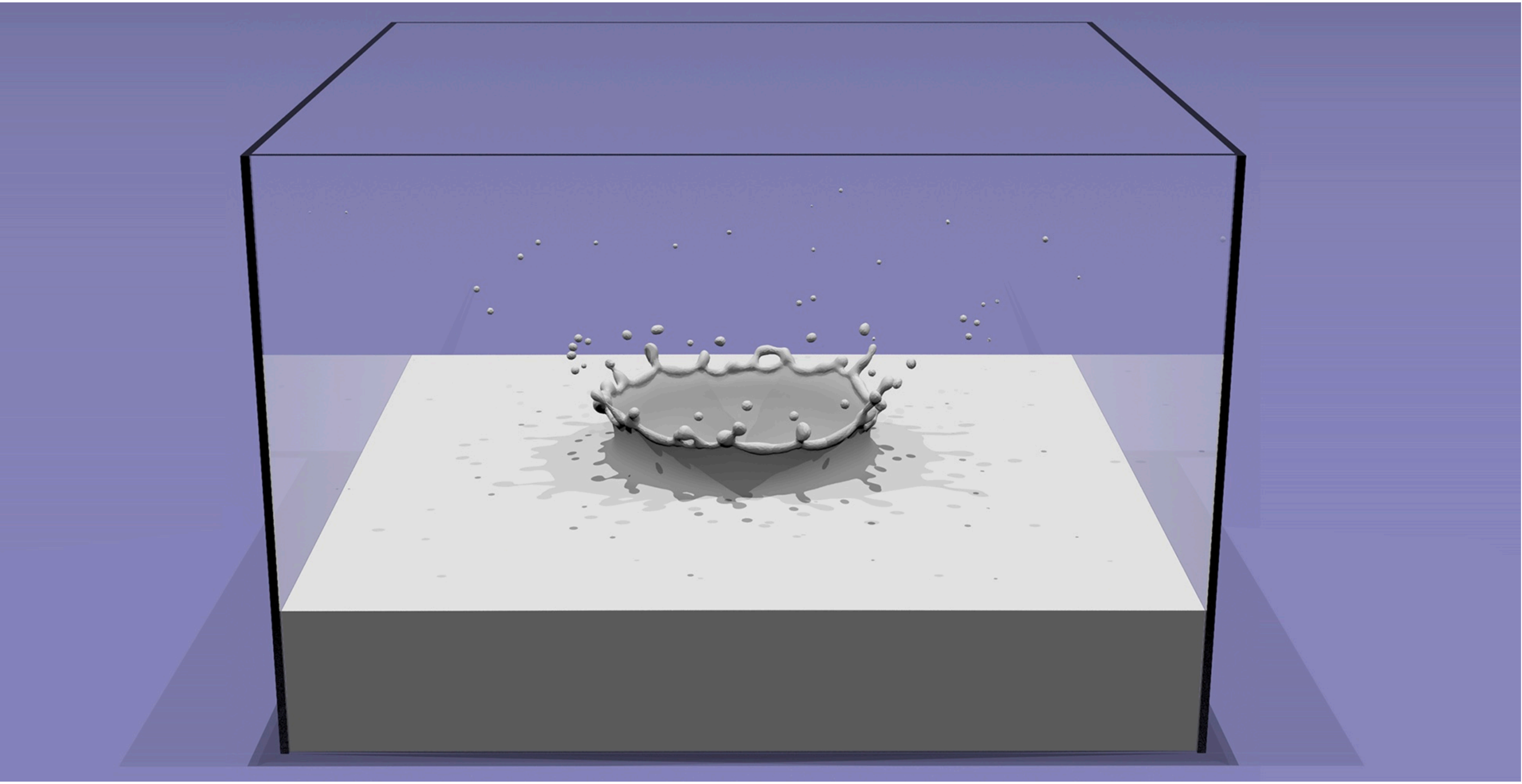
Level Sets from Medical Data (CT, MRI, etc.)

- Level sets encode, e.g., constant tissue density



Level Sets in Physical Simulation

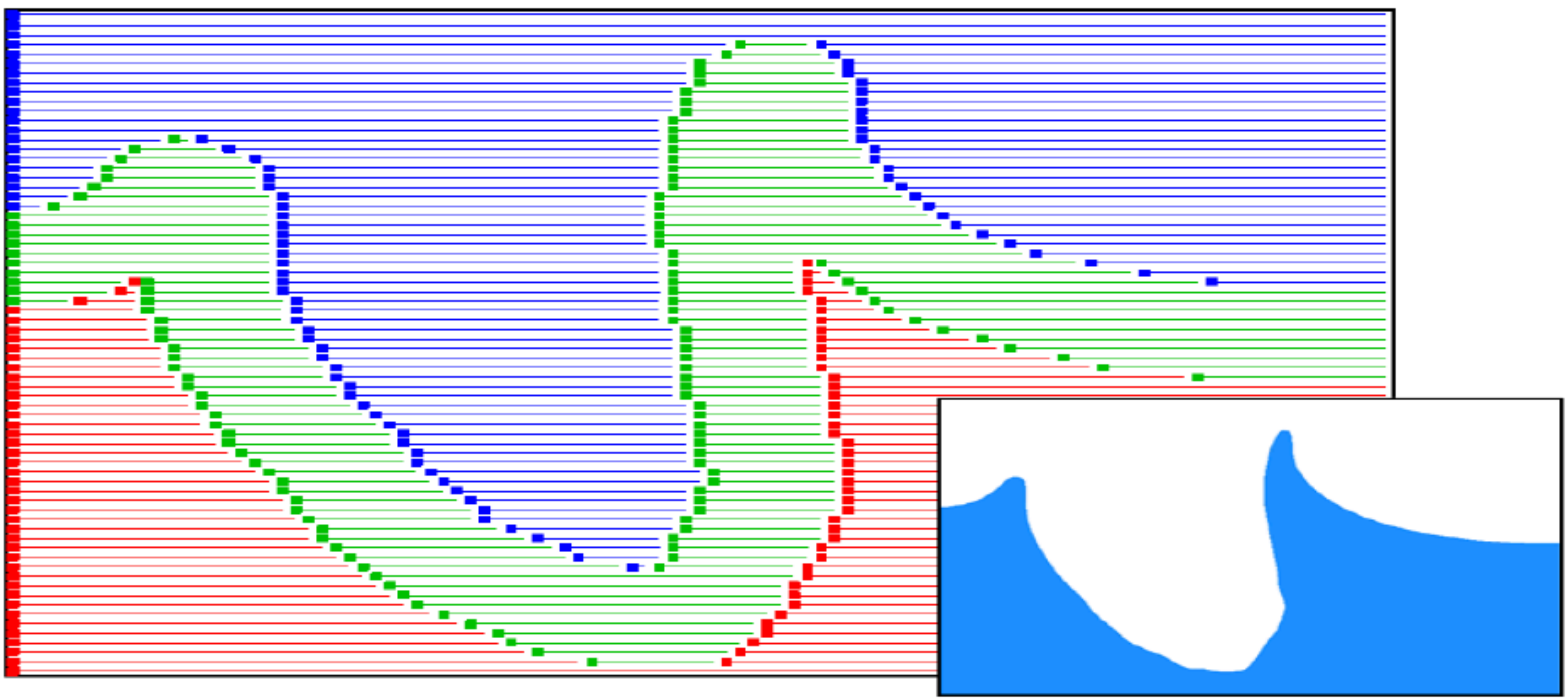
- Level set encodes distance to air-liquid boundary



See http://physbam.stanford.edu

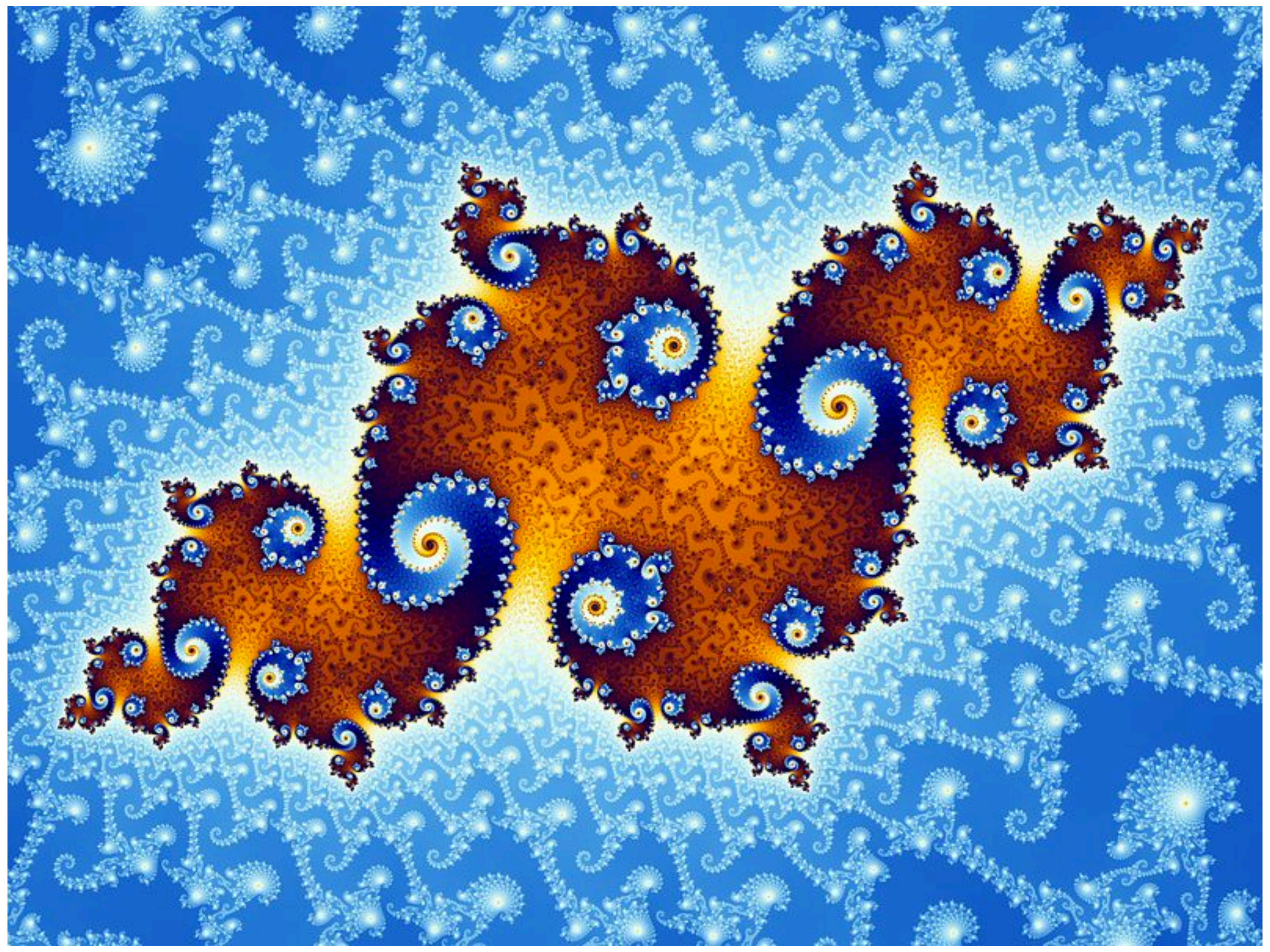
Level Set Storage

- Drawback: storage for 2D surface is now $O(n^3)$
- Can reduce cost by storing only a narrow band around surface:



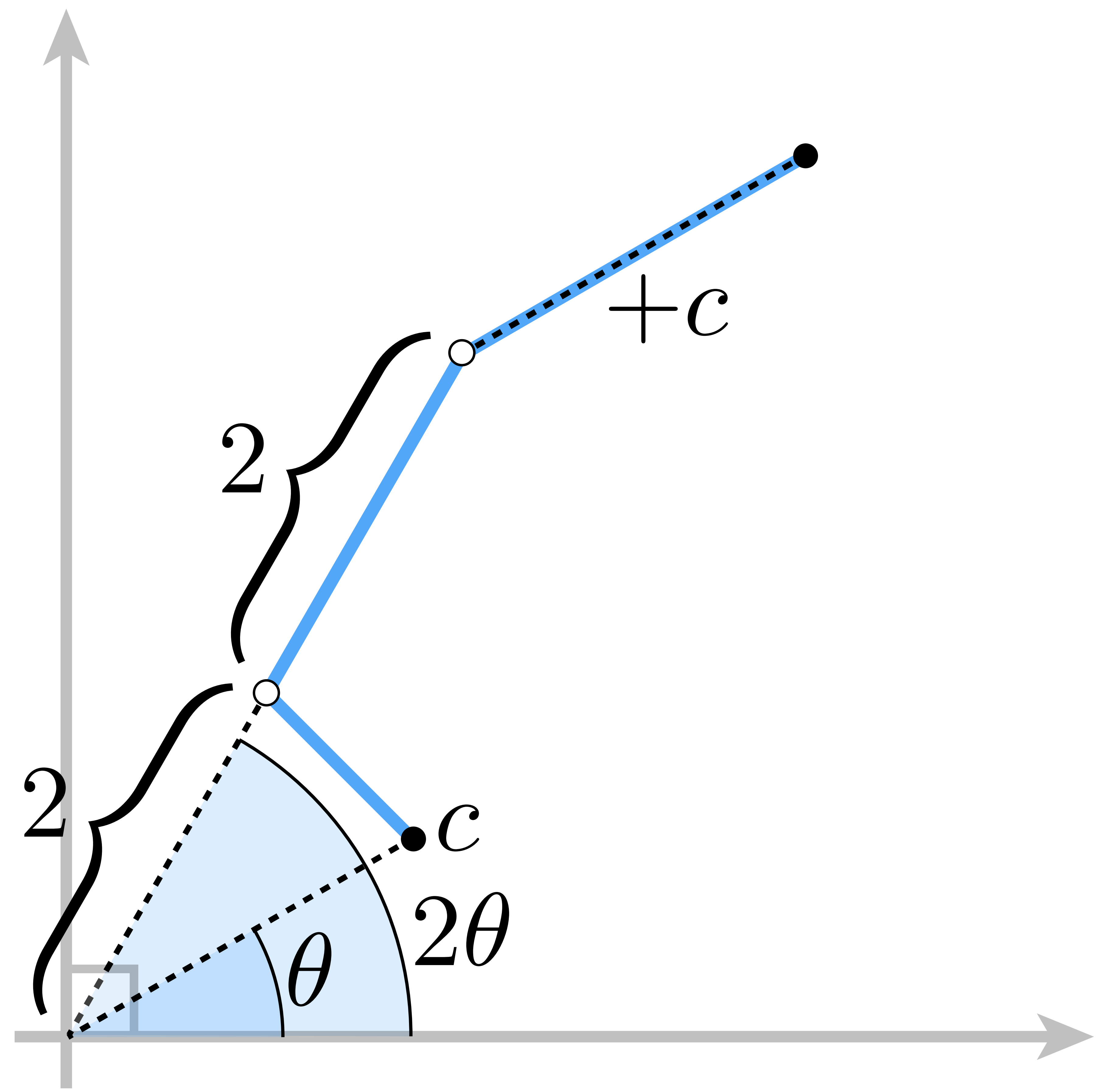
Fractals (Implicit)

- No precise definition; exhibit self-similarity, detail at all scales
- New "language" for describing natural phenomena
- Hard to control shape!



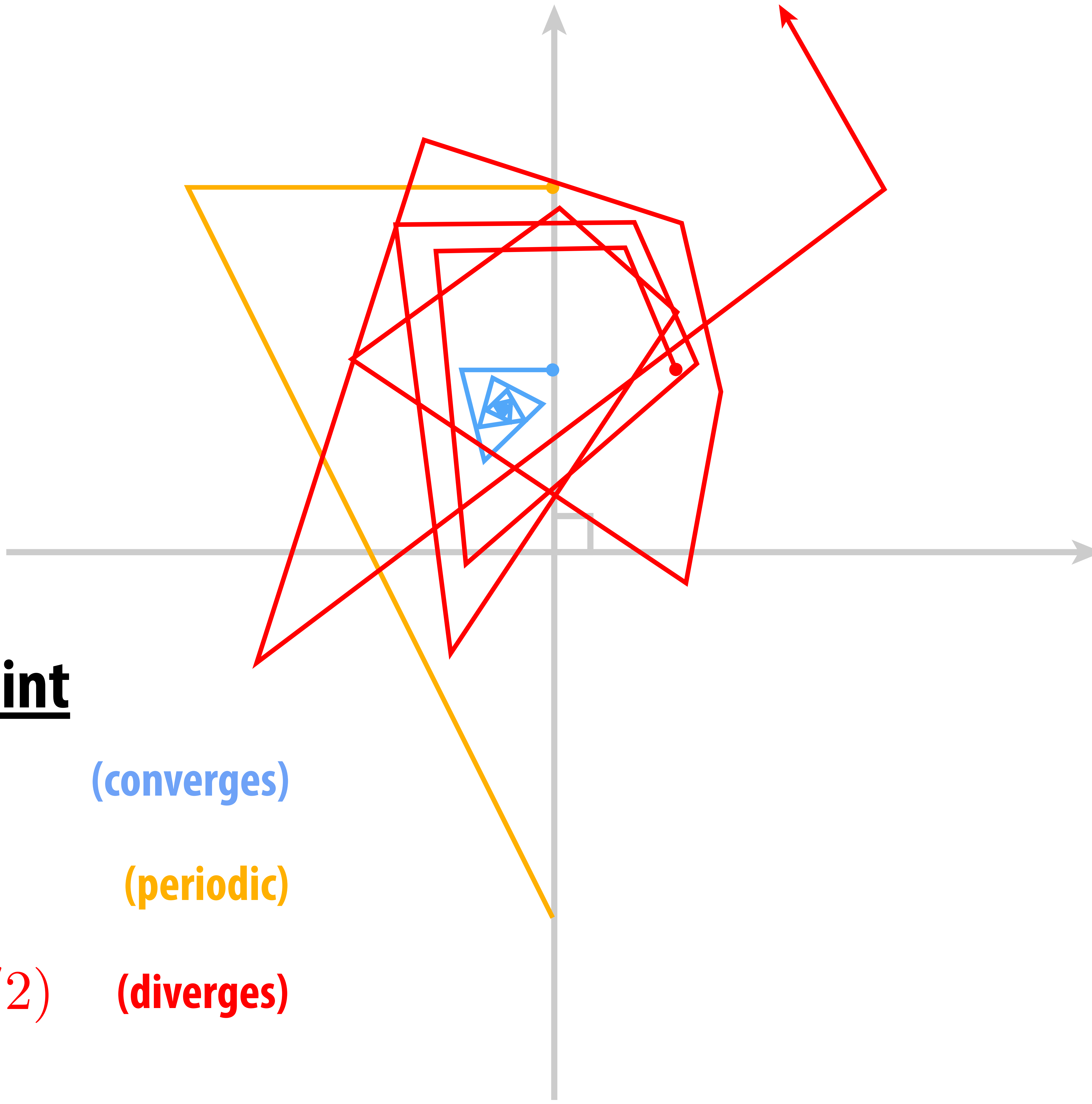
Mandelbrot Set - Definition

- For each point c in the plane:
  - double the angle
  - square the magnitude
  - add the original point c
  - repeat



If the point remains bounded (never goes to ∞), it's in the set.

Mandelbrot Set - Examples



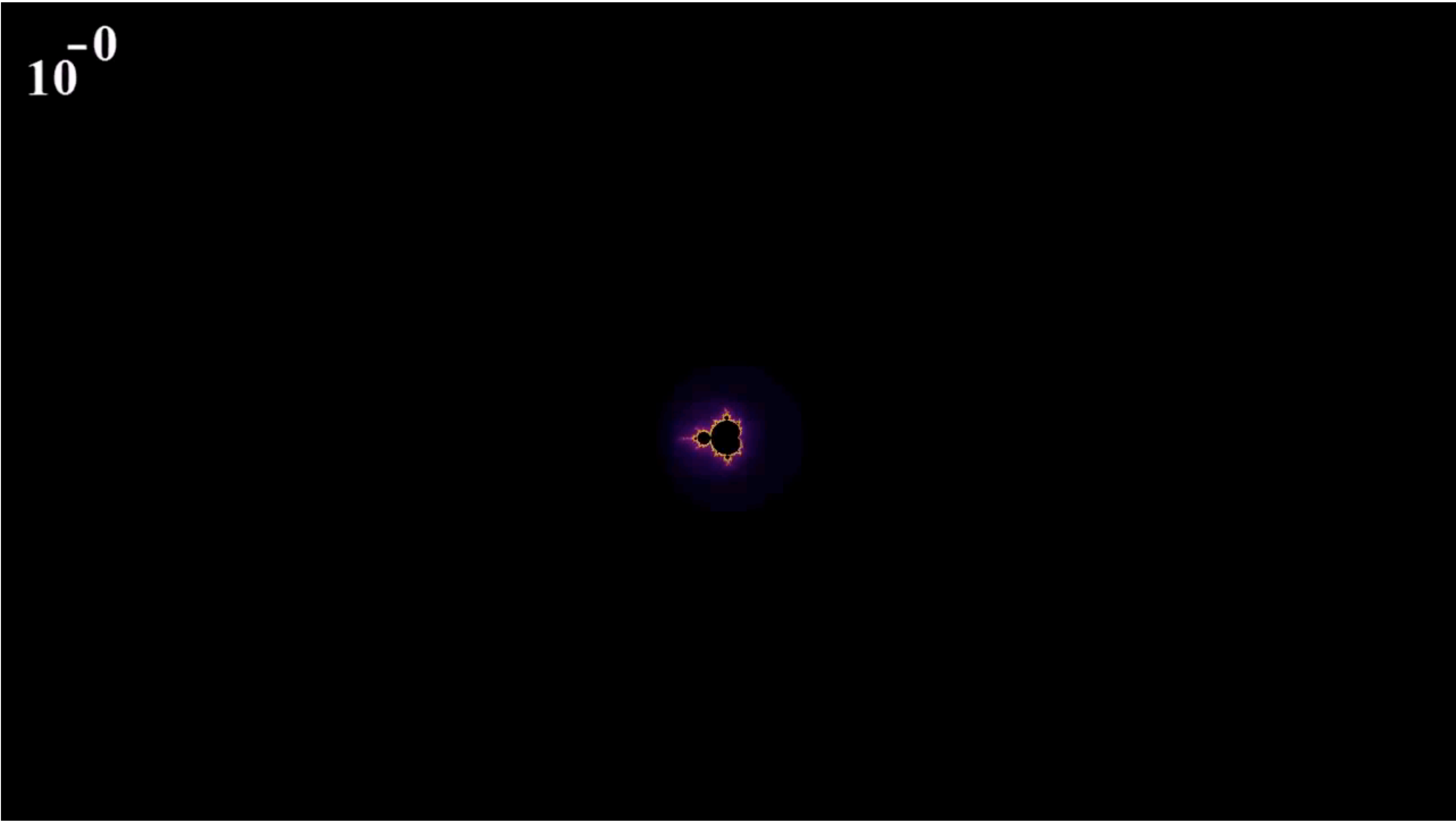
starting point

$(0, 1/2)$ **(converges)**

$(0, 1)$ **(periodic)**

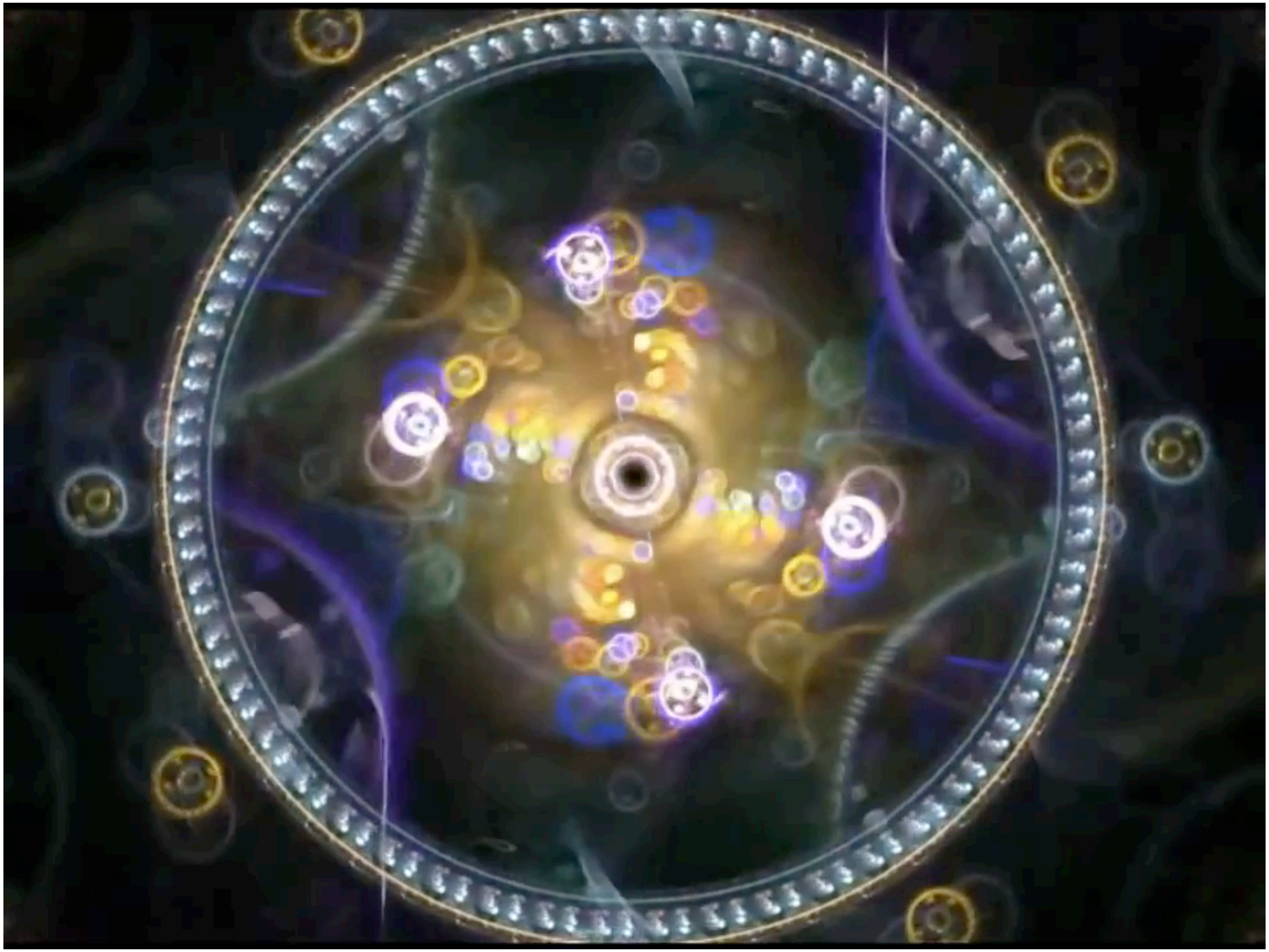
$(1/3, 1/2)$ **(diverges)**

Mandelbrot Set - Zooming In



(Colored according to how quickly each point diverges/converges.)

Iterated Function Systems



Scott Draves (CMU Alumn) - see http://electricsheep.org

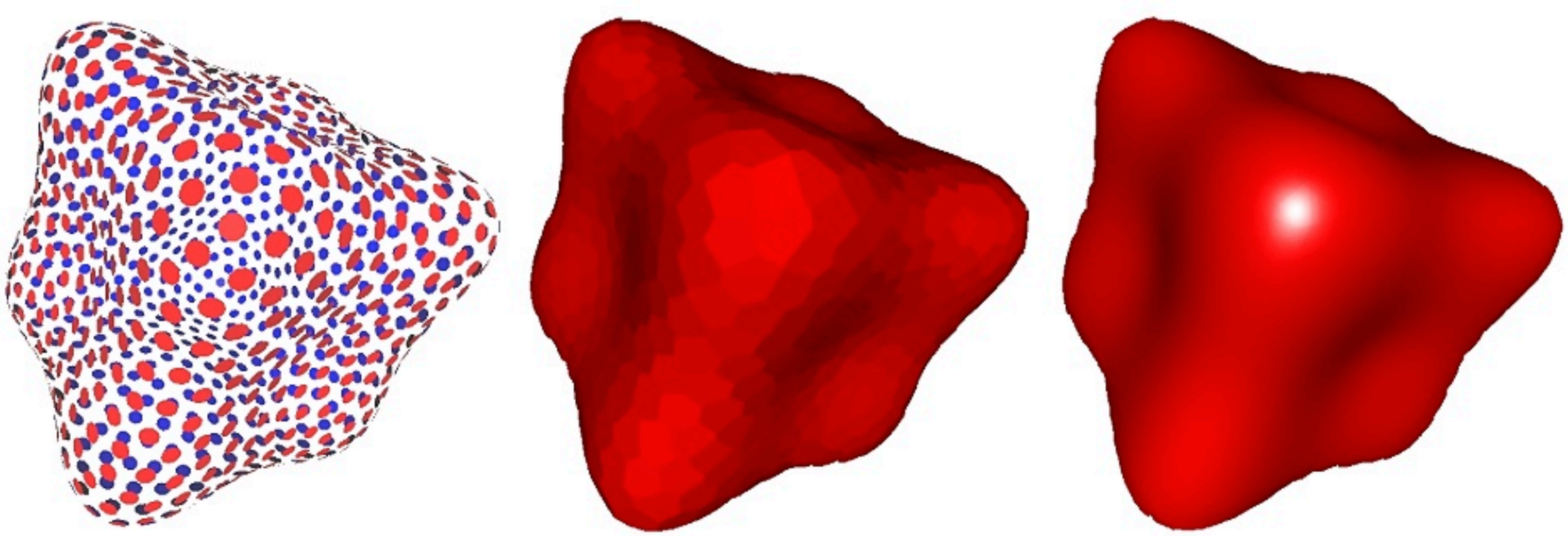
Implicit Representations - Pros & Cons

- **Pros:**
  - **description can be very compact (e.g., a polynomial)**
  - **easy to determine if a point is in our shape (just plug it in!)**
  - **other queries may also be easy (e.g., distance to surface)**
  - **for simple shapes, exact description/no sampling error**
  - **easy to handle changes in topology (e.g., fluid)**
- **Cons:**
  - **expensive to find all points in the shape (e.g., for drawing)**
  - ***very difficult to model complex shapes***

What about explicit representations?

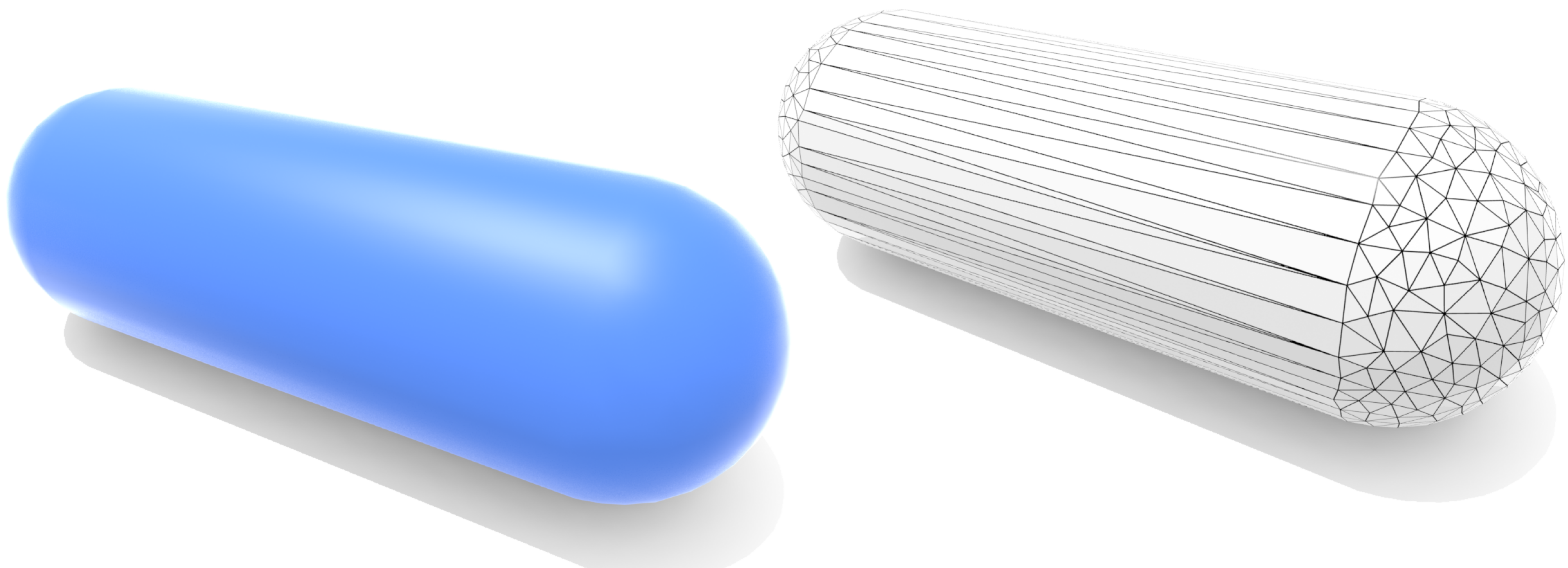
Point Cloud (Explicit)

- Easiest representation: list of points (x,y,z)
- Often augmented with *normals*
- Easily represent any kind of geometry
- Useful for LARGE datasets (>>1 point/pixel)
- Difficult to draw in undersampled regions
- Hard to do processing / simulation



Polygon Mesh (Explicit)

- Store vertices *and* polygons (most often triangles or quads)
- Easier to do processing/simulation, adaptive sampling
- More complicated data structures
- Perhaps most common representation in graphics

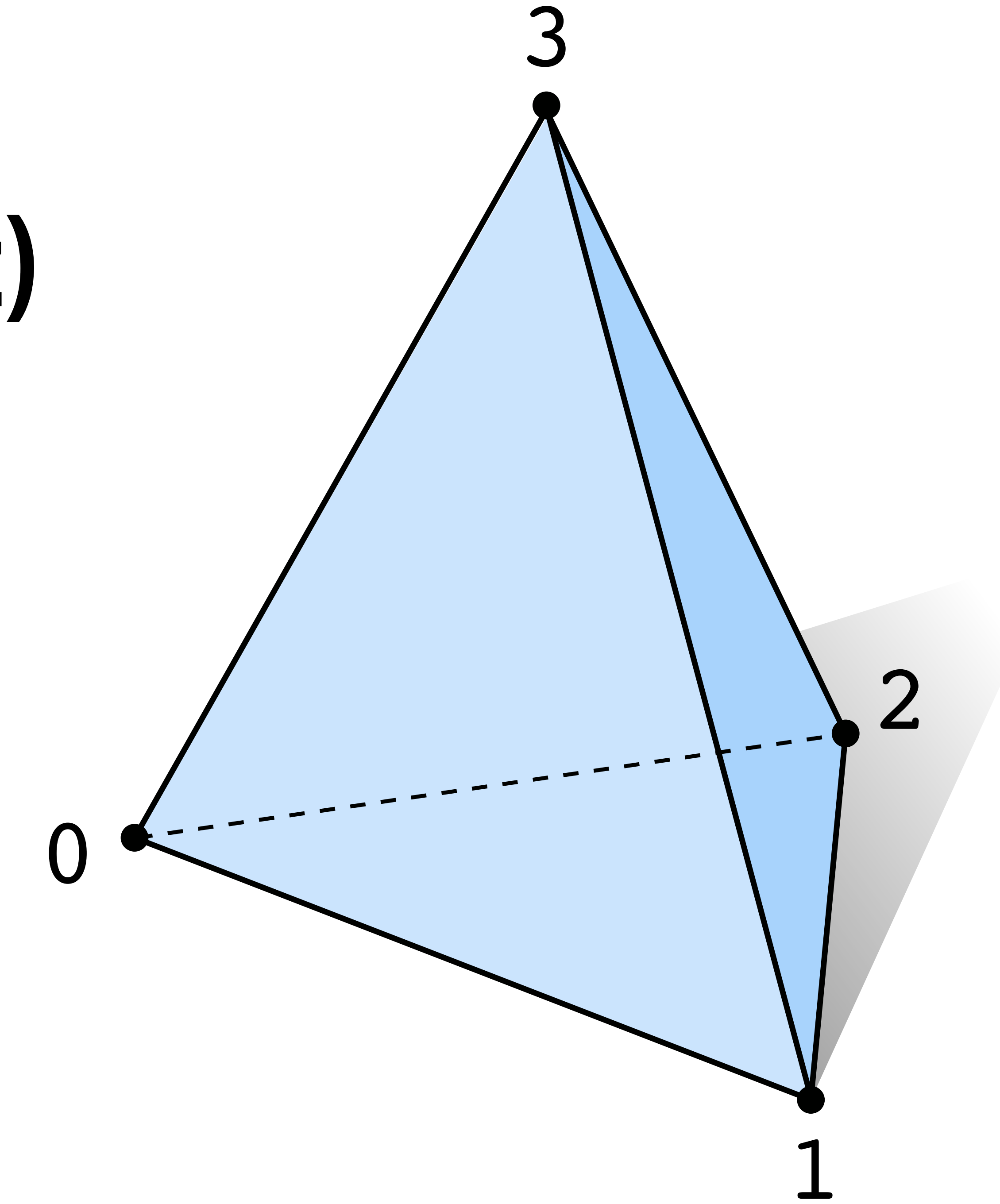


(Much more about polygon meshes in upcoming lectures!)

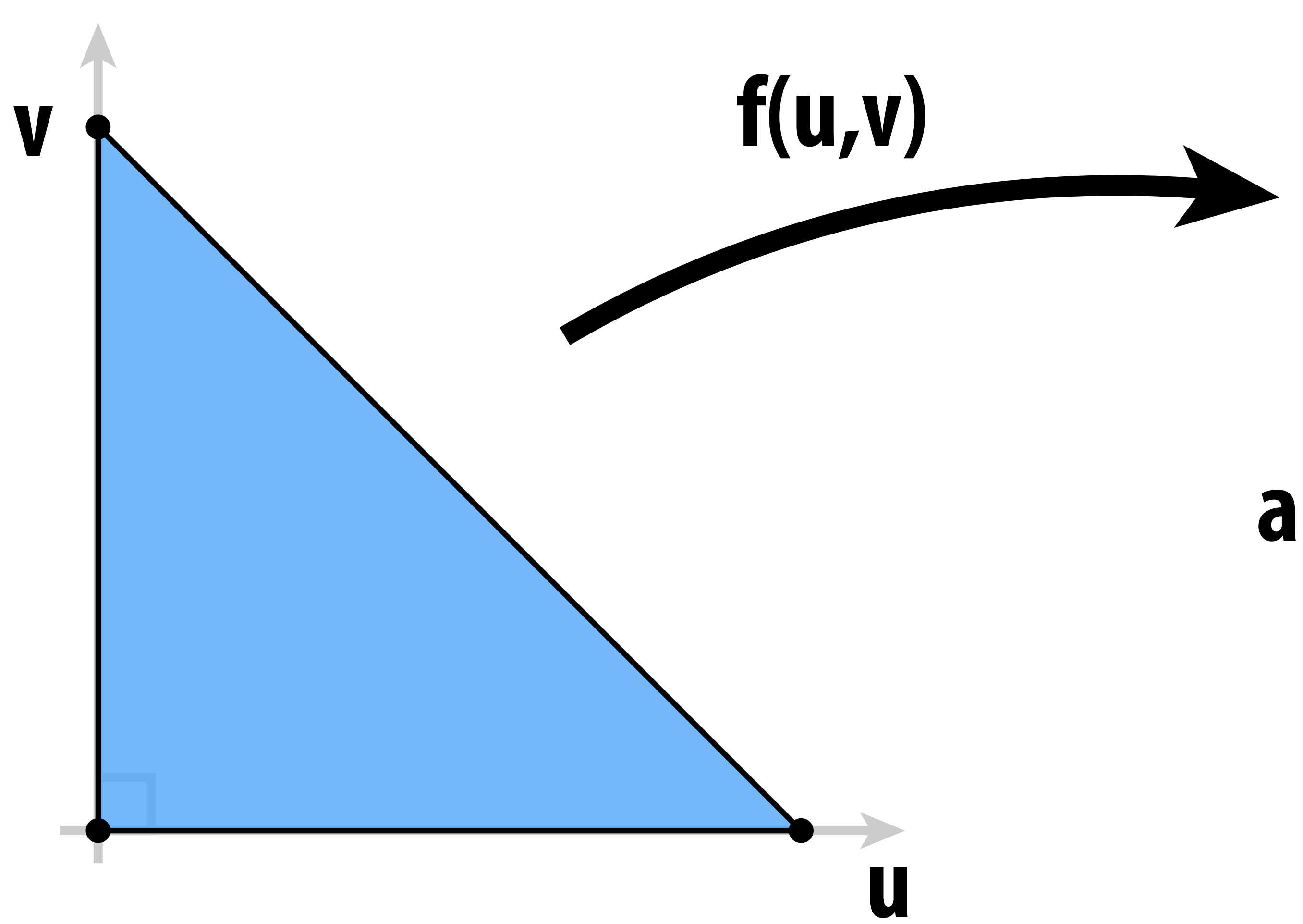
Triangle Mesh (Explicit)

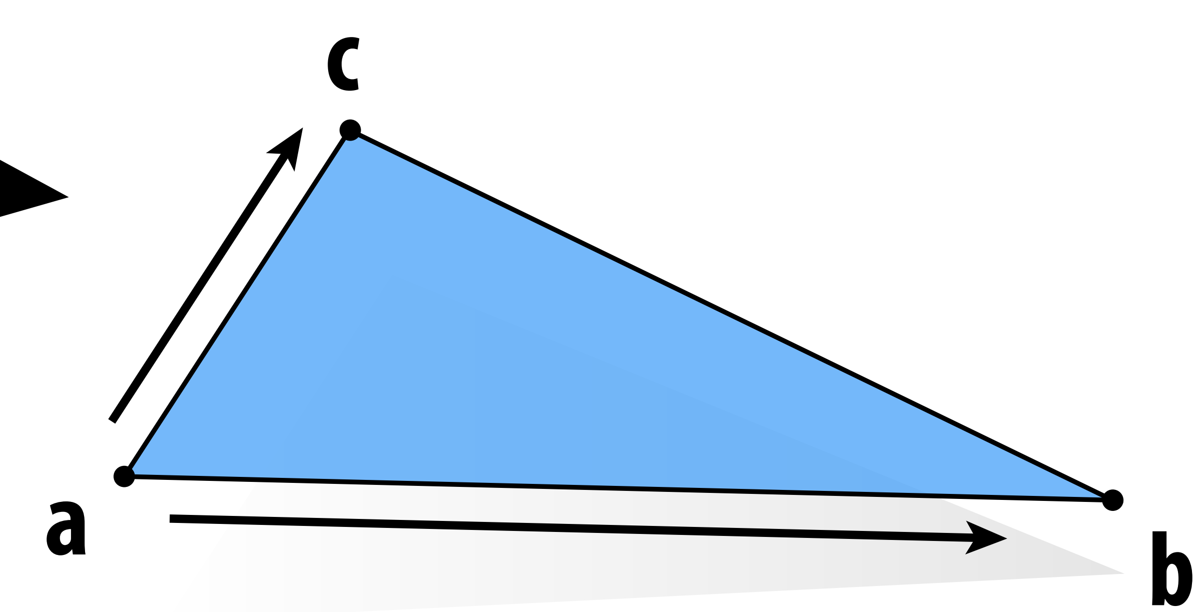
- Store vertices as triples of coordinates (x,y,z)
- Store triangles as triples of indices (i,j,k)
- E.g., tetrahedron:

	VERTICES			TRIANGLES		
	x	y	z	i	j	k
0:	−1	−1	−1	0	2	1
1:	1	−1	1	0	3	2
2:	1	1	−1	3	0	1
3:	−1	1	1	3	1	2



- Can think of triangle as *affine* map from plane into space:





$$f(u,v) = a + u(b-a) + v(c-a)$$

Bernstein Basis

- Why limit ourselves to just affine functions?
- More flexibility by using higher-order polynomials
- Instead of usual basis (1, x, x^2, x^3, ...), use Bernstein basis:

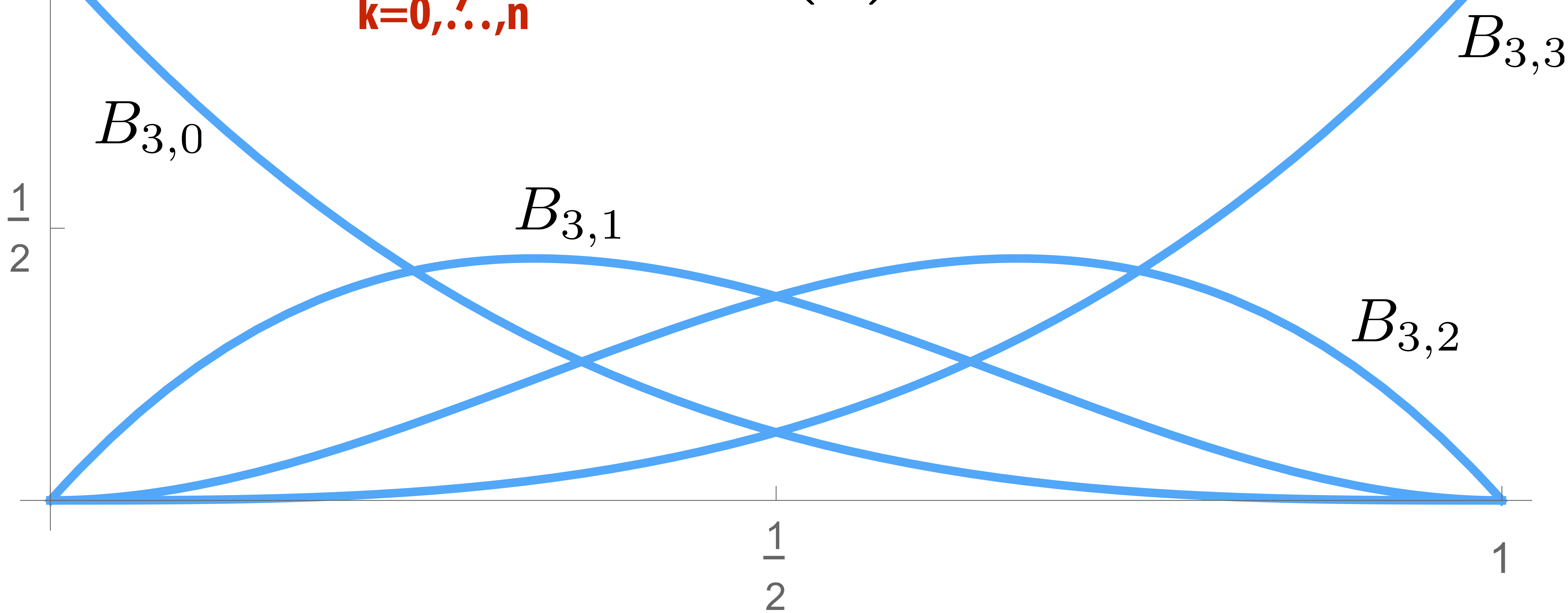
degree

0≤x≤1

"n choose k"

$$B_{n,k}(x) := \binom{n}{k} x^k (1-x)^{n-k}$$

k=0,...,n



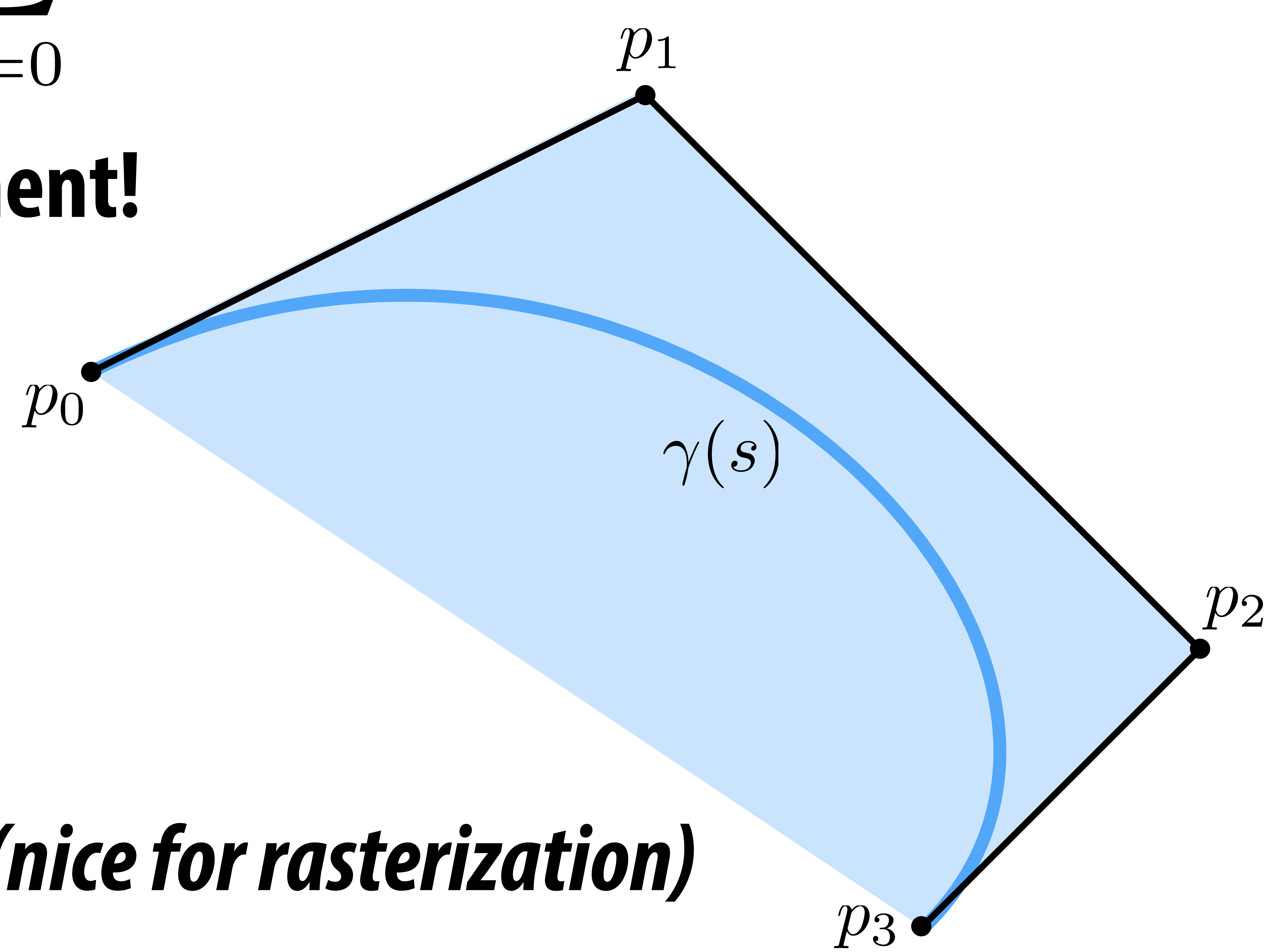
Bézier Curves (Explicit)

- **A Bézier curve is a curve expressed in the Bernstein basis:**

$$\gamma(s) := \sum_{k=0}^{n} B_{n,k}(s)p_k$$

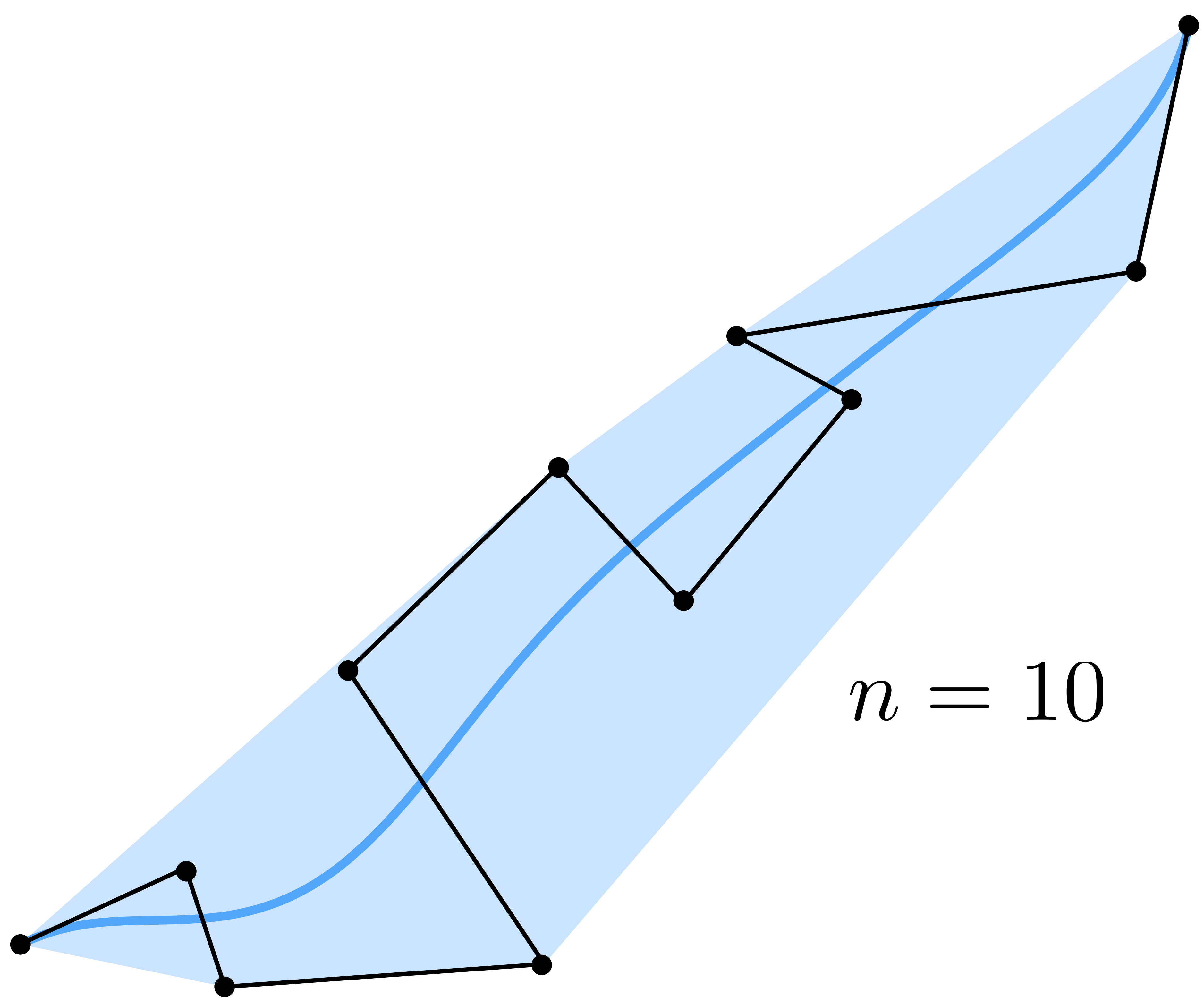
control points

- **For n=1, just get a line segment!**
- **For n=3, get "cubic Bézier":**
- **Important features:**
  1. **interpolates endpoints**
  2. **tangent to end segments**
  3. **contained in convex hull *(nice for rasterization)***



Higher-order Bézier Curves?

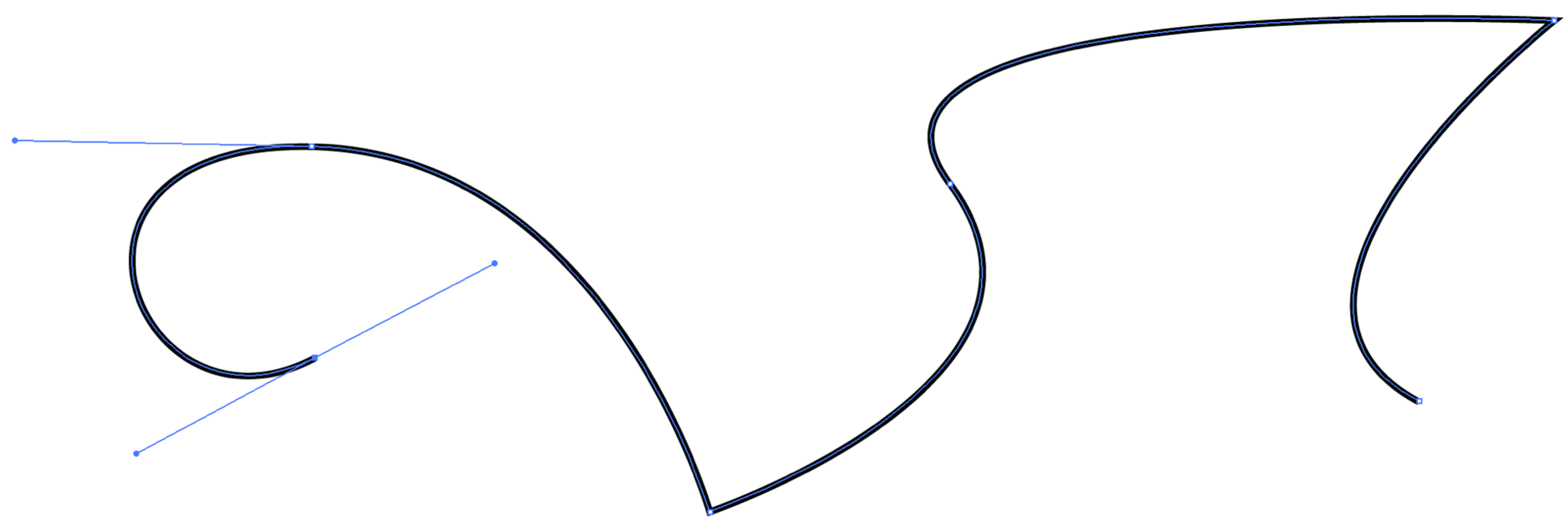
- What if we want a more interesting curve?
- High-degree Bernstein polynomials don't interpolate well:



Very hard to control!

B-Spline Curves (Explicit)

- **Instead, use many low-order Bézier curve (B-spline)**
- **Widely-used technique in software (Illustrator, Inkscape, etc.)**



- **Formally, piecewise Bézier curve:**

B-spline

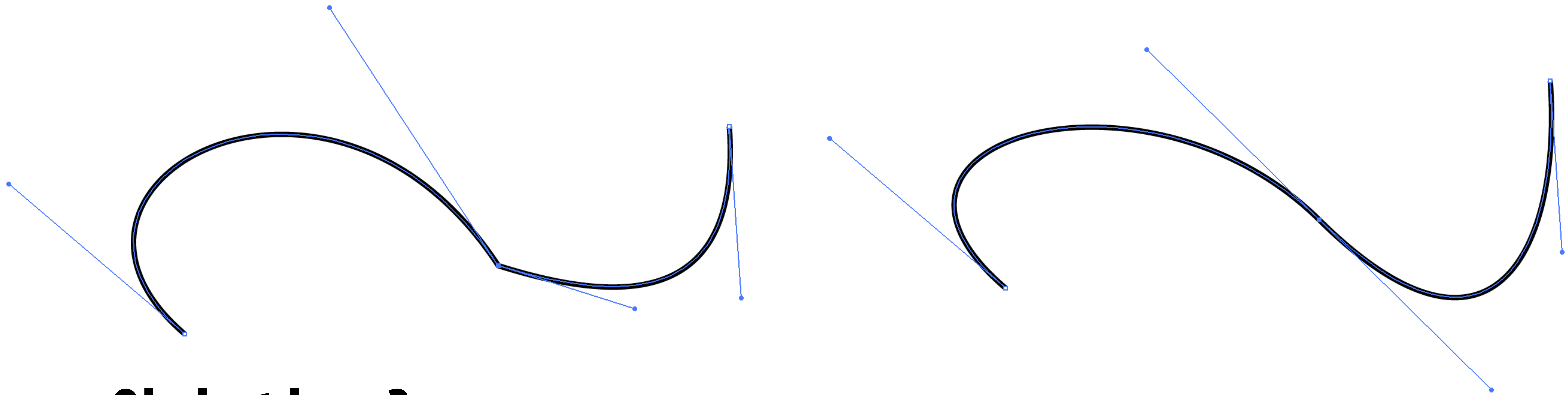
$$\gamma(u) := \gamma_i\left(\frac{u-u_i}{u_{i+1}-u_i}\right), \qquad u_i \leq u < u_{i+1}$$

Bézier

- **Location of u_i parameters are called "knots"**

B-Splines — tangent continuity

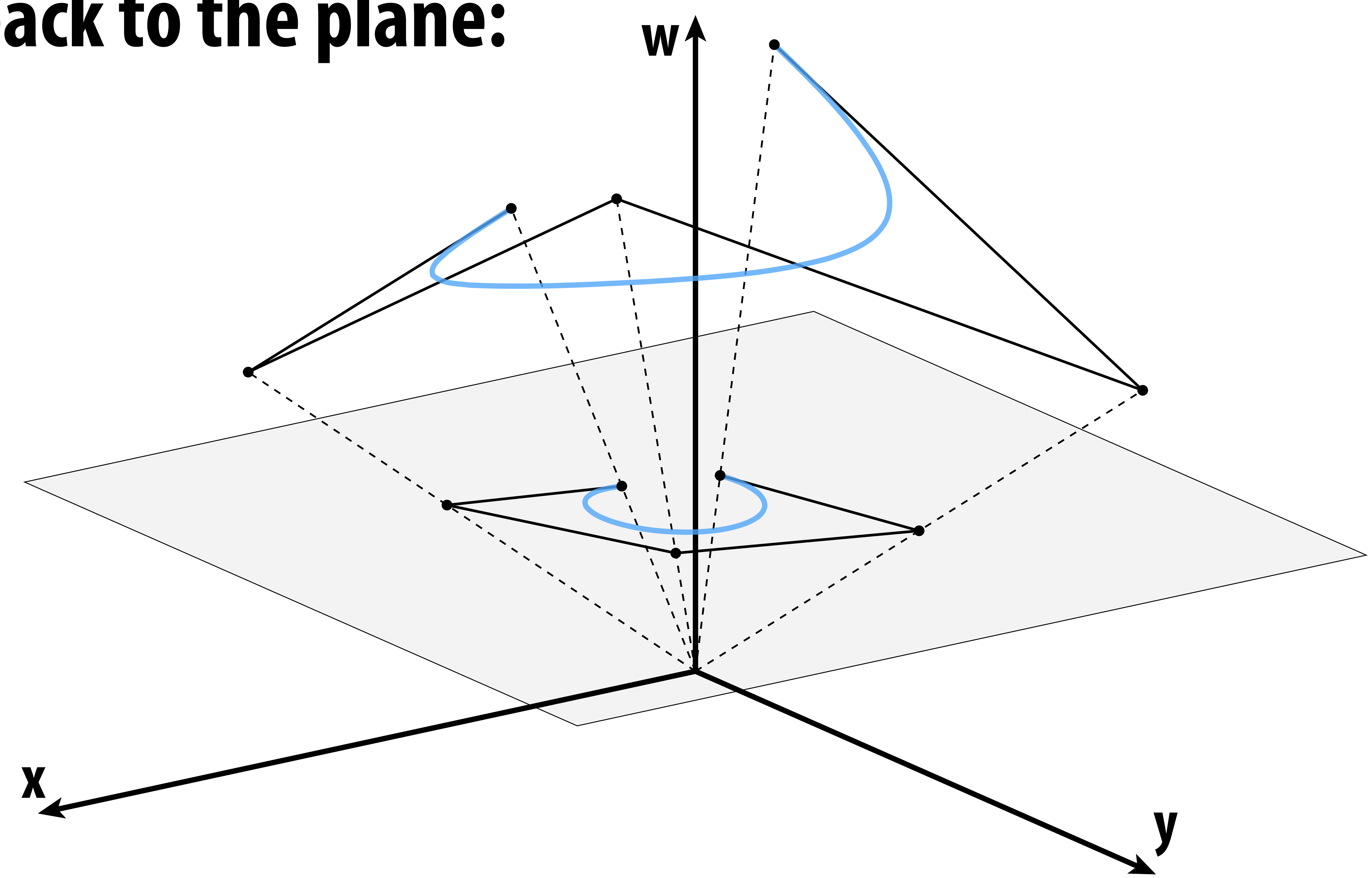
- To get "seamless" curves, want *tangents* to line up:



- Ok, but how?
- Each curve is cubic: $u^3p_0 + 3u^2(1-u)p_1 + 3u(1-u)^2p_2 + (1-u)^3p_3$
- Q: How many degrees of freedom in a single cubic Bézier
- Tangents are difference between first two & last two points
- Q: How many degrees of freedom per B-spline segment?
- Q: Could you do this with quadratic Bézier? Linear Bézier?

Rational B-Splines (Explicit)

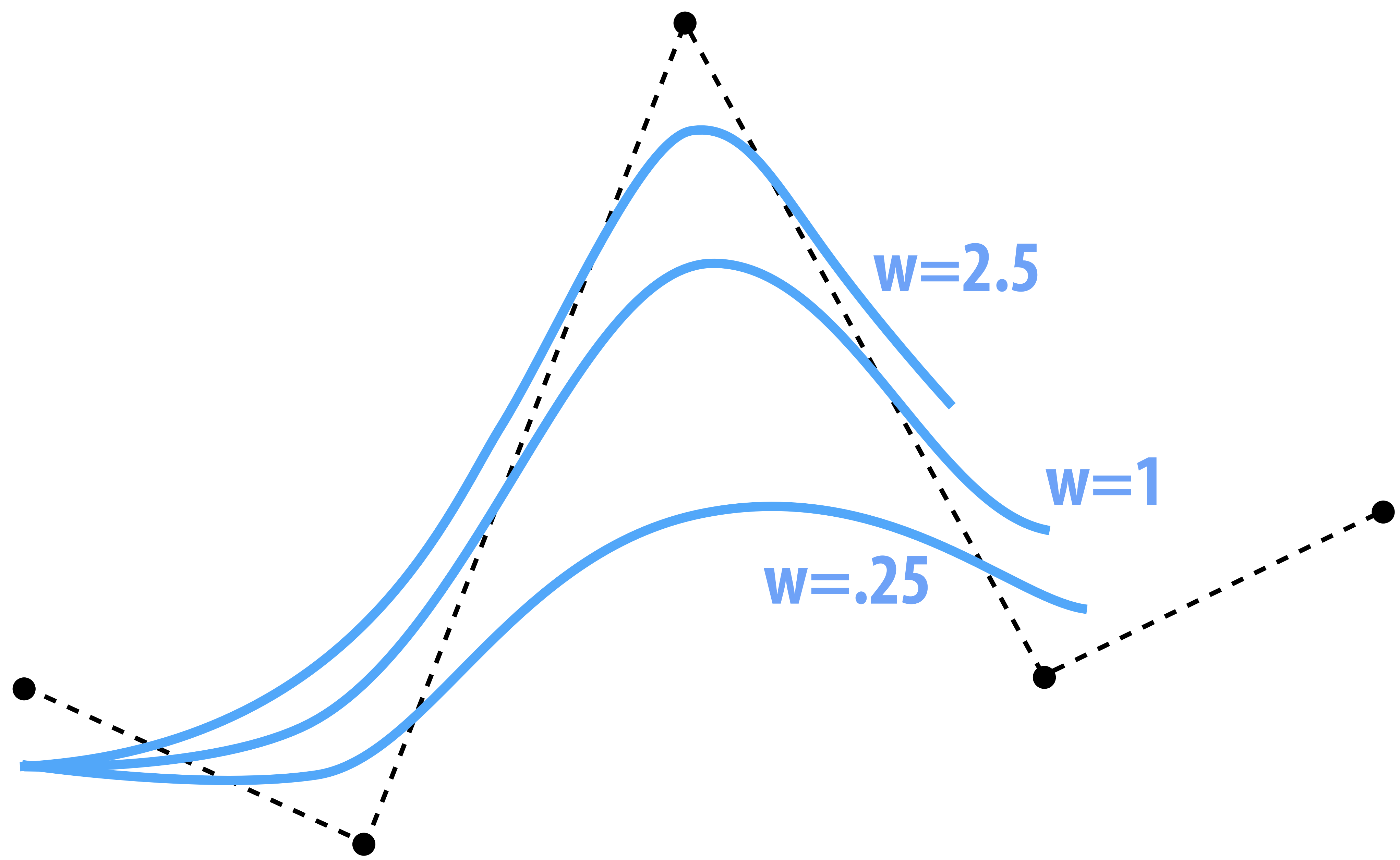
- B-Splines can't exactly represent *conics*—not even the circle!
- Solution: interpolate in homogeneous coordinates, then project back to the plane:



Result is called a *rational* B-spline.

NURBS (Explicit)

- (N)on-(U)niform (R)ational (B)-(S)pline
  - knots at arbitrary locations (non-uniform)
  - expressed in homogeneous coordinates (rational)
  - piecewise polynomial curve (B-Spline)
- Homogeneous coordinate w controls "strength" of a vertex:

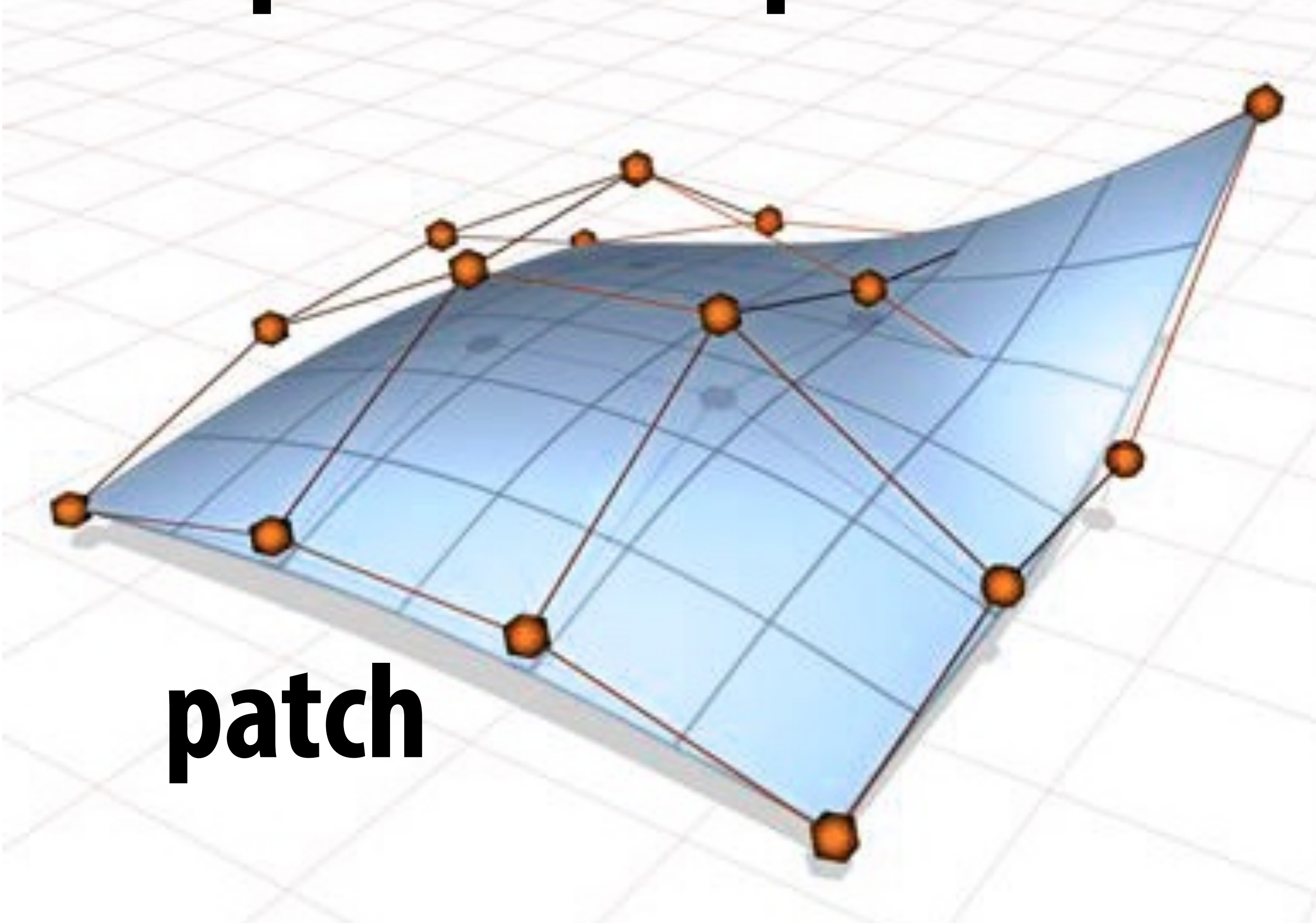


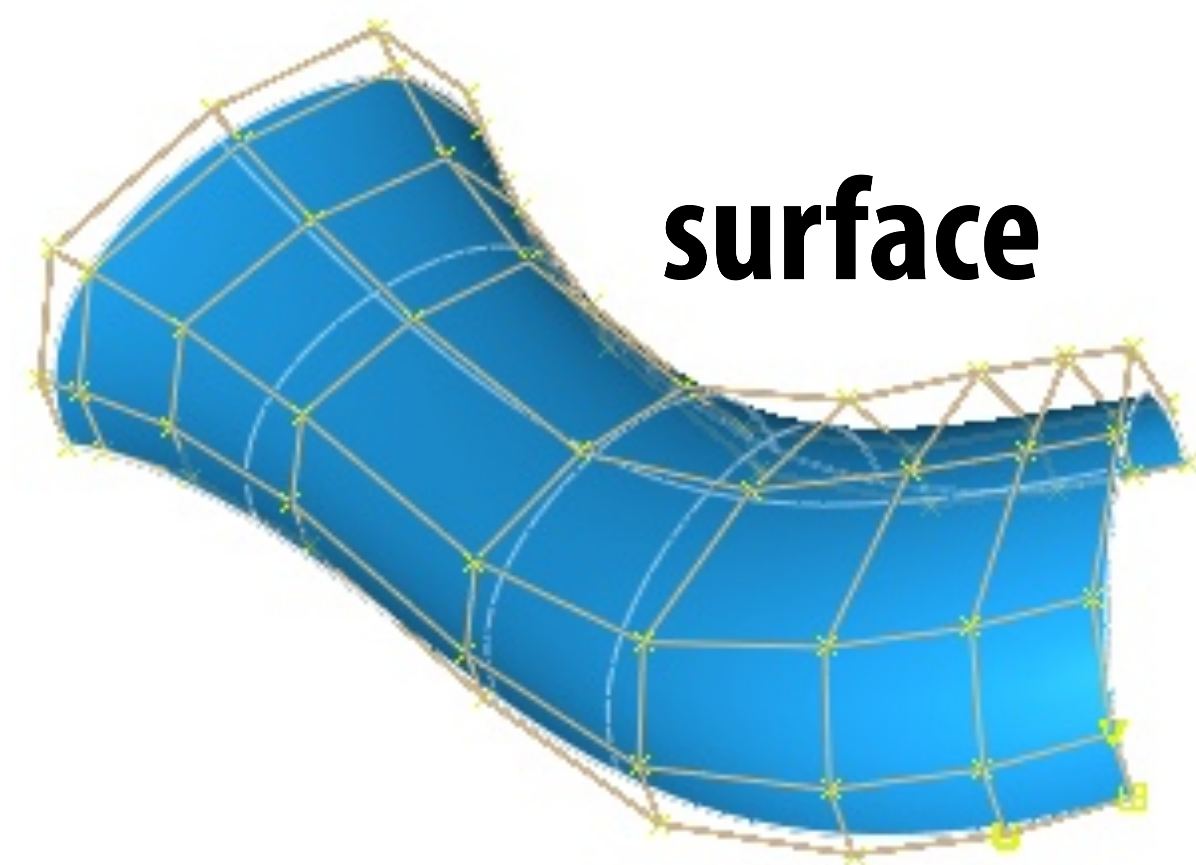
NURBS Surface (Explicit)

- Much more common than using NURBS for curves
- Use *tensor product* of scalar NURBS curves to get a patch:

$$S(u,v) := N_i(u)N_j(v)p_{ij}$$

- Multiple NURBS patches form a surface

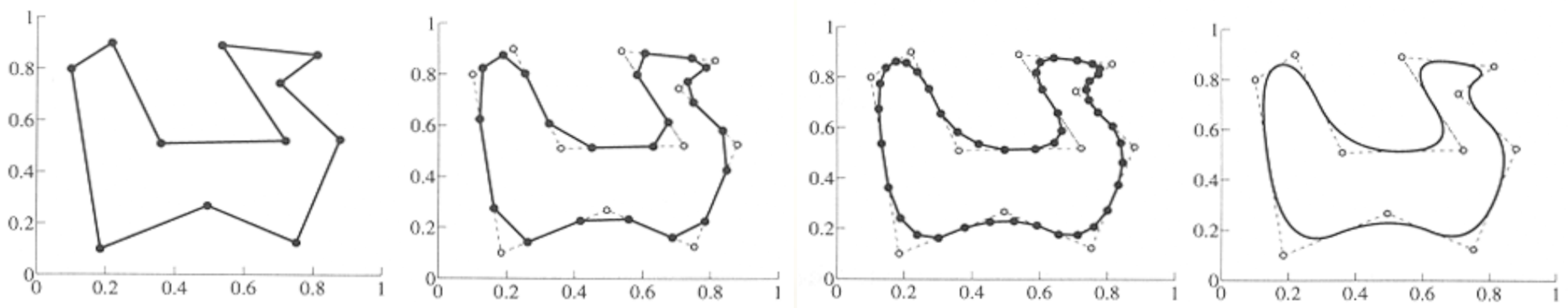




- Pros: easy to evaluate, exact conics, high degree of continuity
- Cons: Hard to piece together patches, hard to edit (many DOFs)

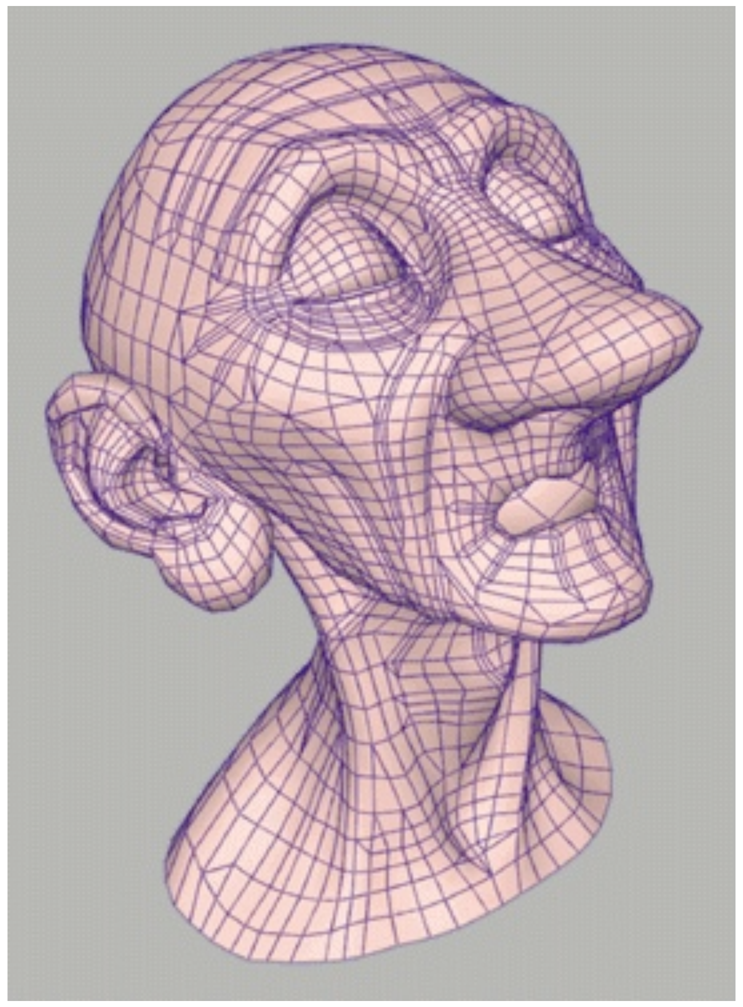
Subdivision (Explicit)

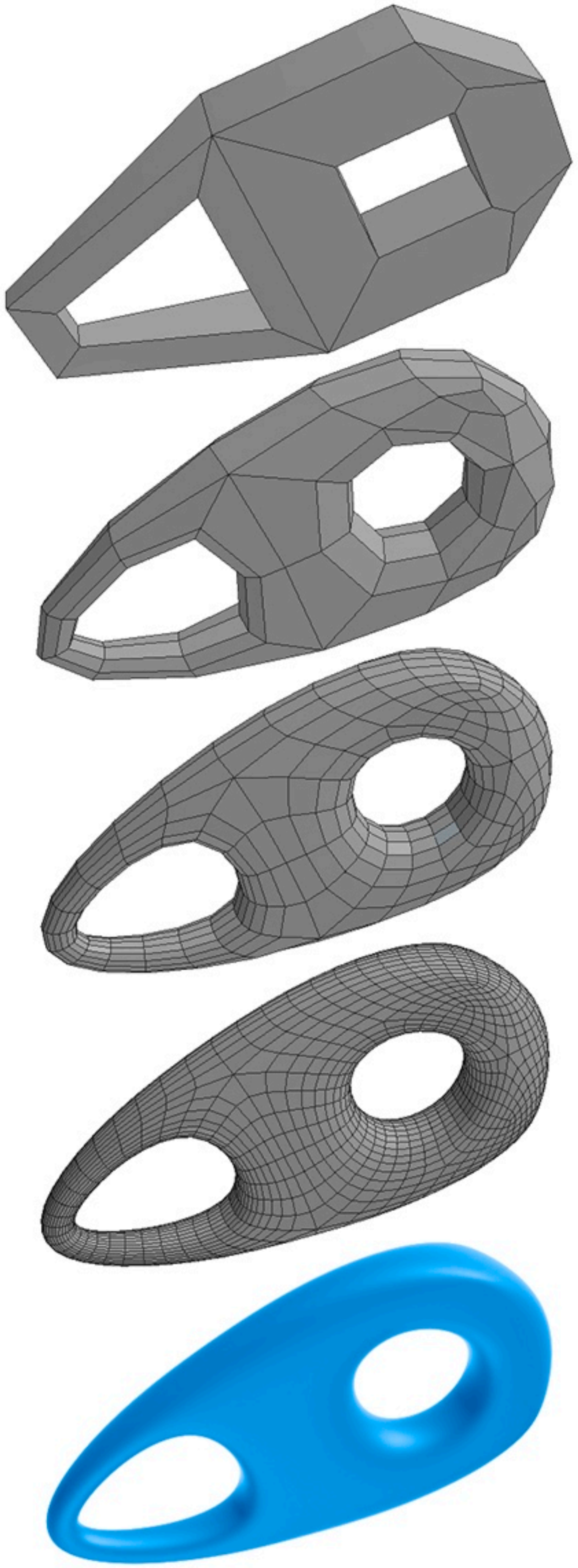
- **Alternative starting point for B-spline curves: *subdivision***
- **Start with control curve**
- **Insert new vertex at each edge midpoint**
- **Update vertex positions according to fixed rule**
- **For careful choice of averaging rule, yields smooth curve**
  - **Average with "next" neighbor (Chaikin): quadratic B-spline**
  - **Lane-Riesenfeld: B-spline curve of any degree**



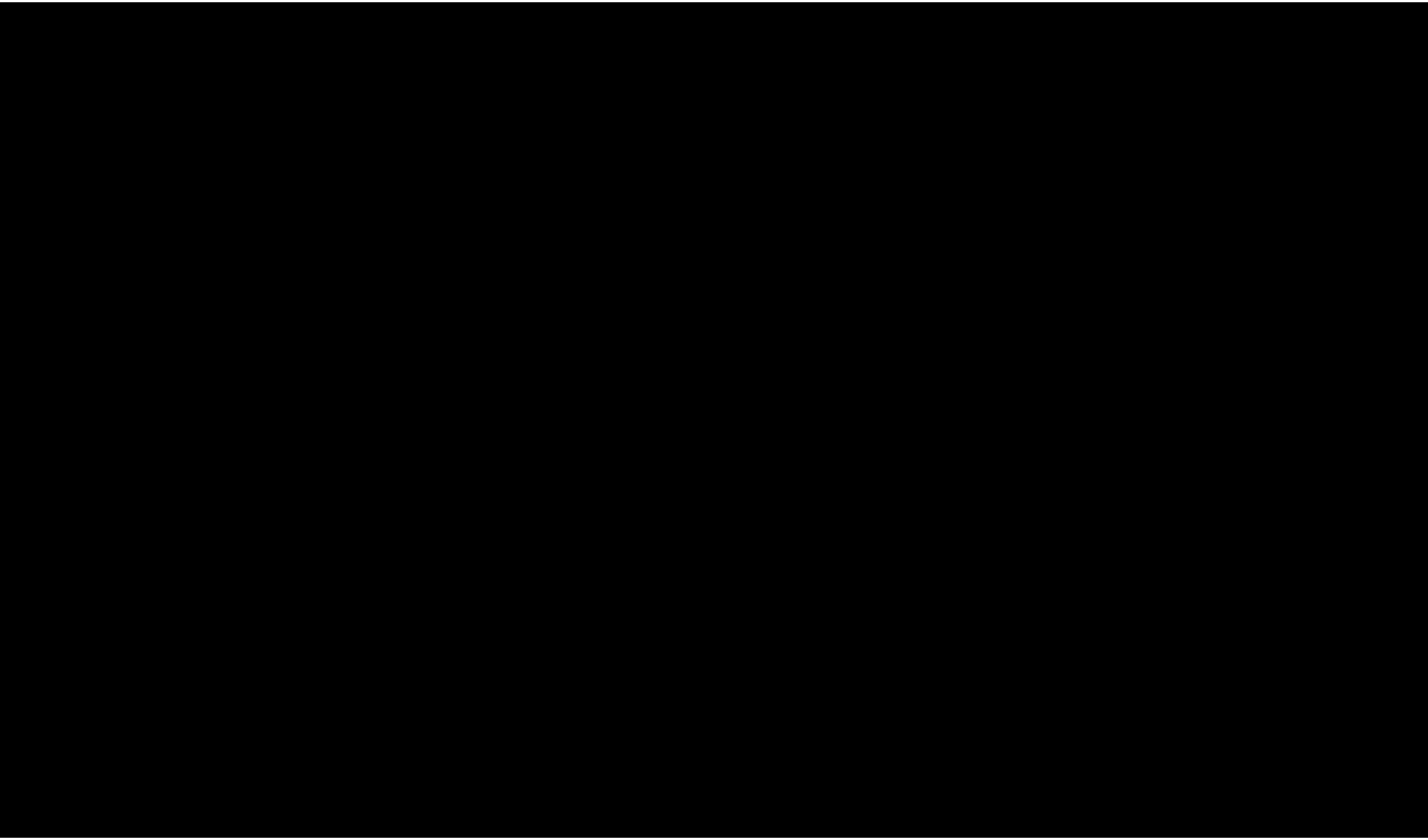
Subdivision Surfaces (Explicit)

- Start with coarse polygon mesh ("control cage")
- Subdivide each element
- Update vertices via local averaging
- Many possible rule:
  - Catmull-Clark (quads)
  - Loop (triangles)
  - ...
- Common issues:
  - interpolating or approximating?
  - continuity at vertices?
- Easier than NURBS for modeling; harder to guarantee continuity





Subdivision in Action (Pixar's "Geri's Game")



Next time: Curves, Surfaces, & Meshes