北京航空航天大學

人工智能学院（人工智能研究院）

Computer Graphics
Lecture 10: Geometry Introduction

潘成伟 (Chengwei Pan)

Email: pancw@buaa.edu.cn

Office: Room B1021, New Main Building

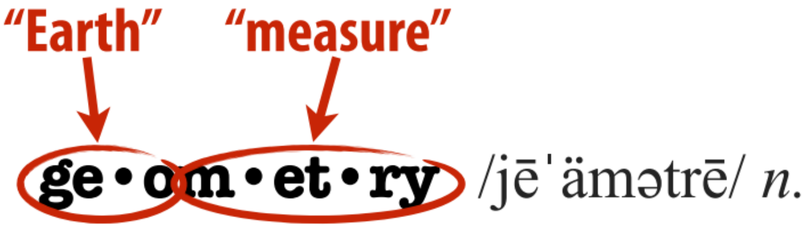
北京航空航天大学，人工智能学院

School of Artificial Intelligence, Beihang University

This Lecture

- Introduction to geometry
  - Examples of geometry
  - Various representations of geometry

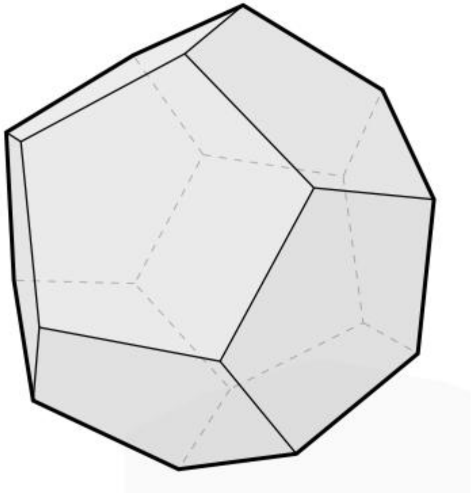
What is geometry?



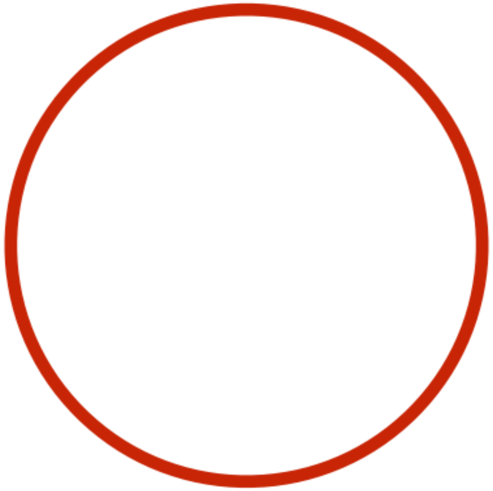
“Earth” “measure”

ge•om•et•ry /jēˈämətrē/ *n.*

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



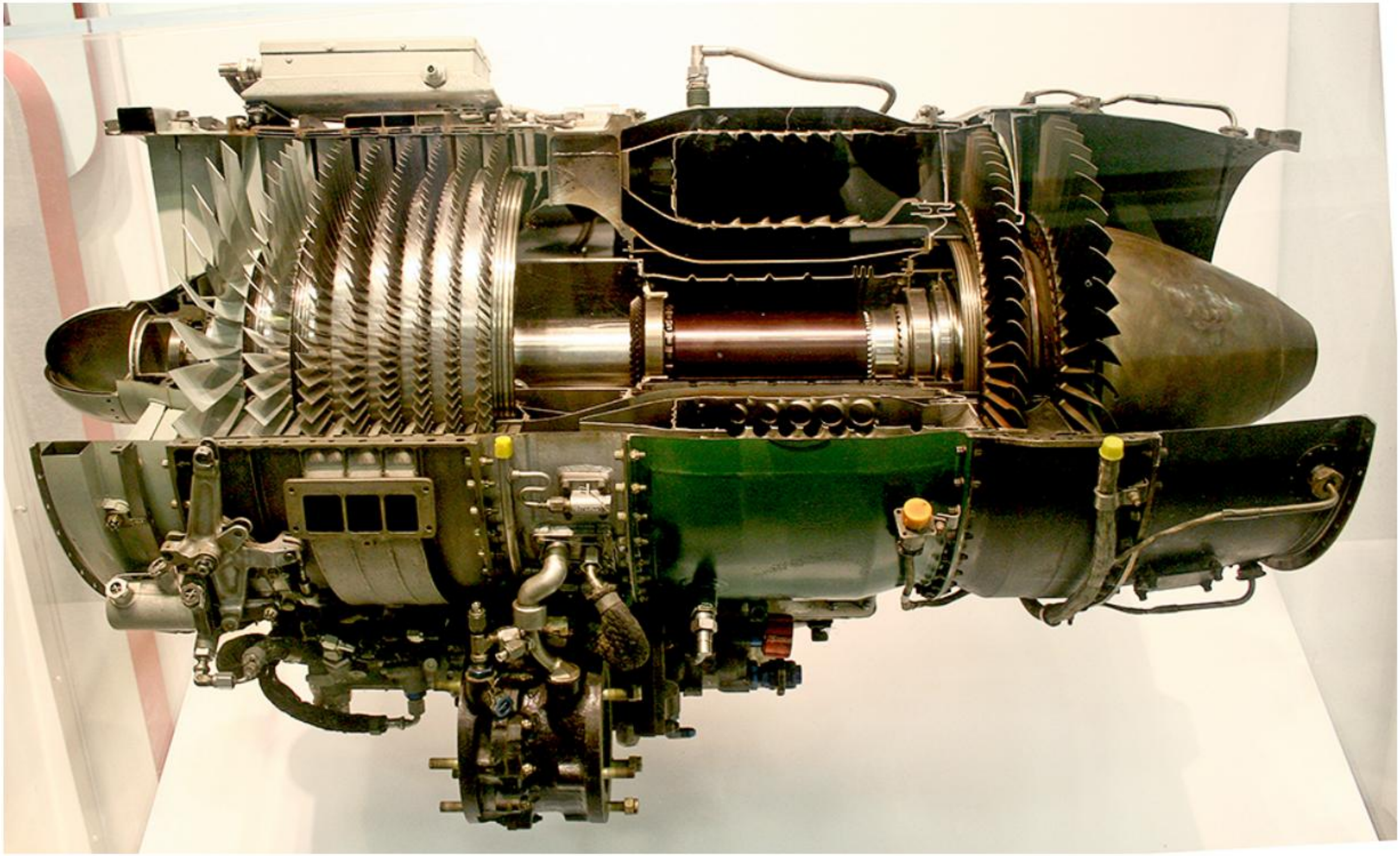
How can we describe geometry?



Examples of Geometry

Examples of Geometry

Examples of Geometry



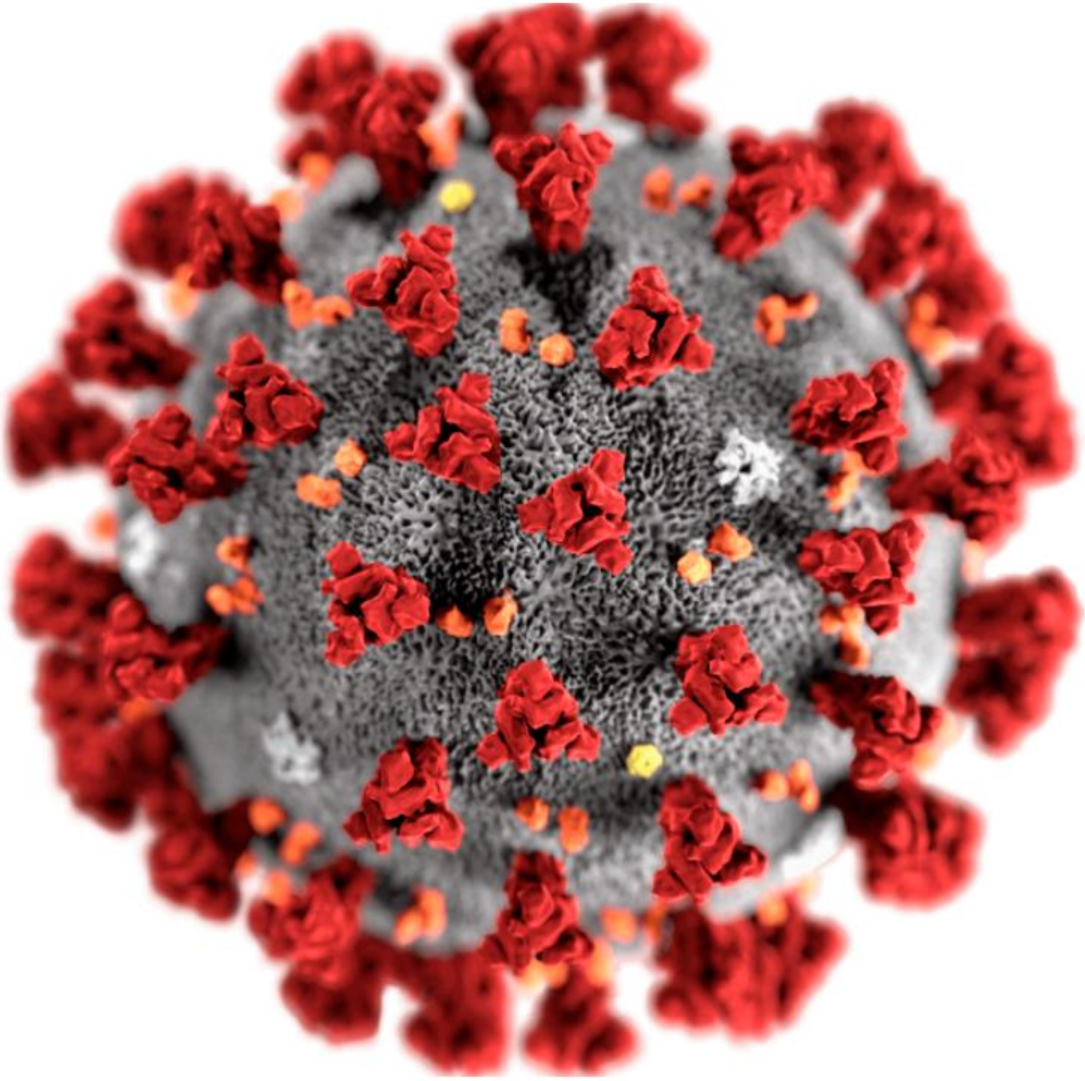
Example of Geometry

Examples of Geometry

Examples of Geometry

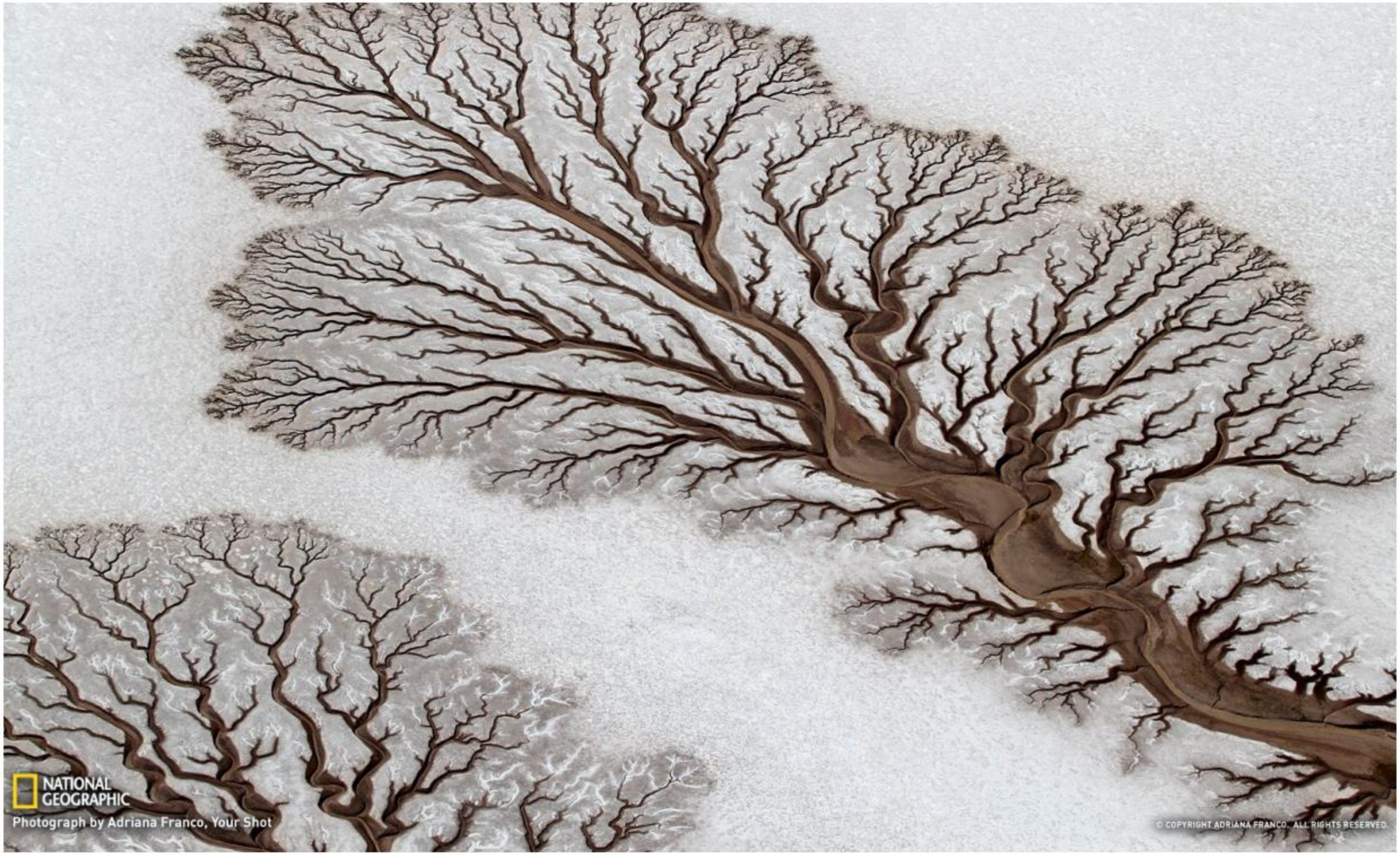
Examples of Geometry

Examples of Geometry



COVID-19

Examples of Geometry



NATIONAL GEOGRAPHIC

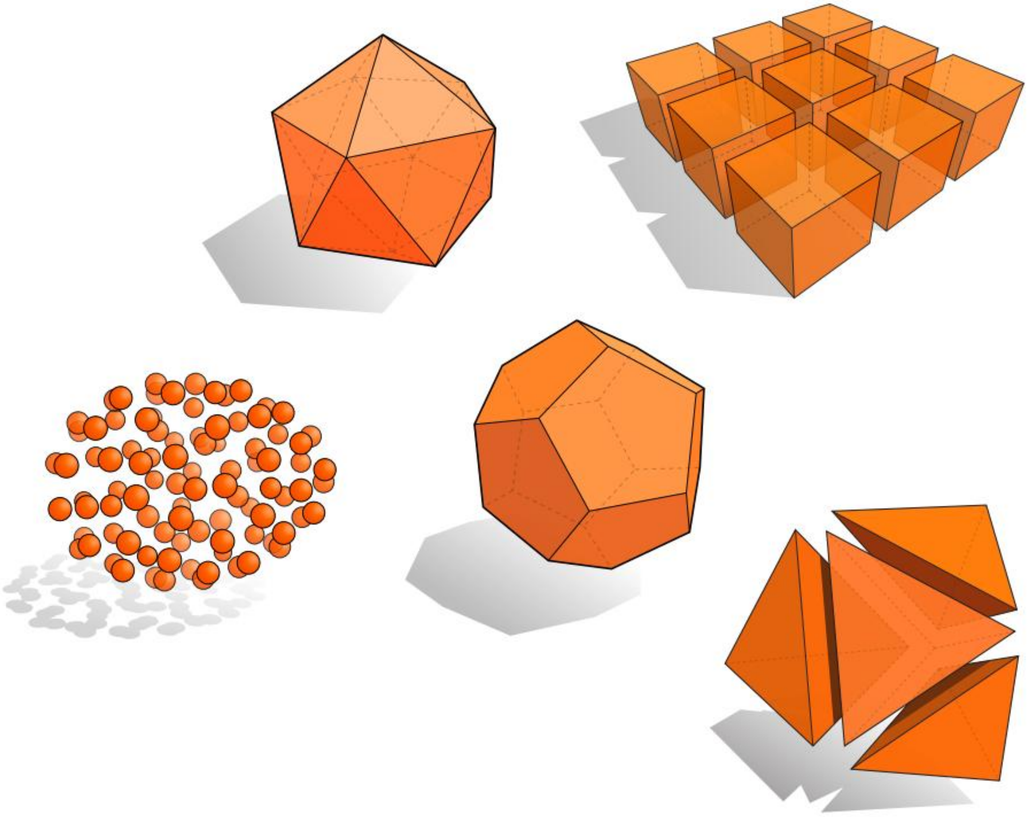
Photograph by Adriana Franco, Your Shot

This Lecture

- Introduction to geometry
  - Examples of geometry
  - Various representations of geometry

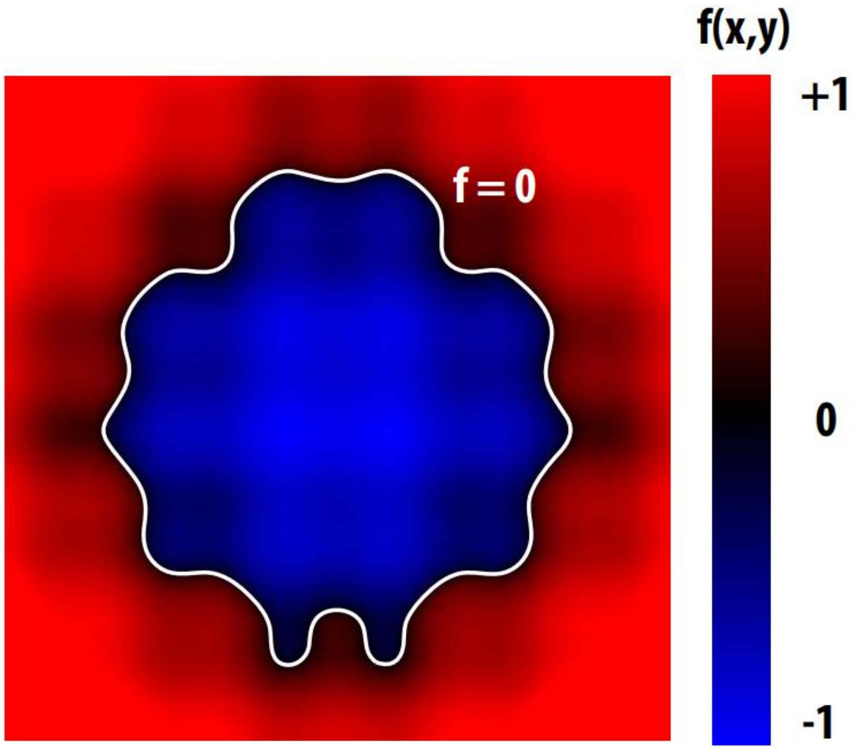
Many Ways to Represent Geometry

- **Implicit**
  - Algebraic surface
  - Level sets
  - Distance function
  - …
- **Explicit**
  - Point cloud
  - Polygon mesh
  - Subdivision, NURBS
  - …
- Each choice best suited to a different task/type of geometry



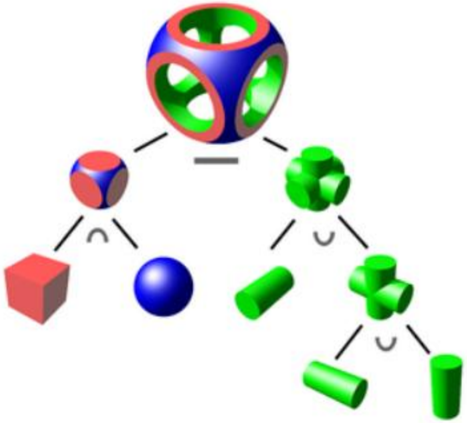
Implicit Representations of Geometry

- Points aren't known directly, but satisfy some relationship
- E.g., unit sphere is all points 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
- …



Implicit Surface – Sampling can be hard

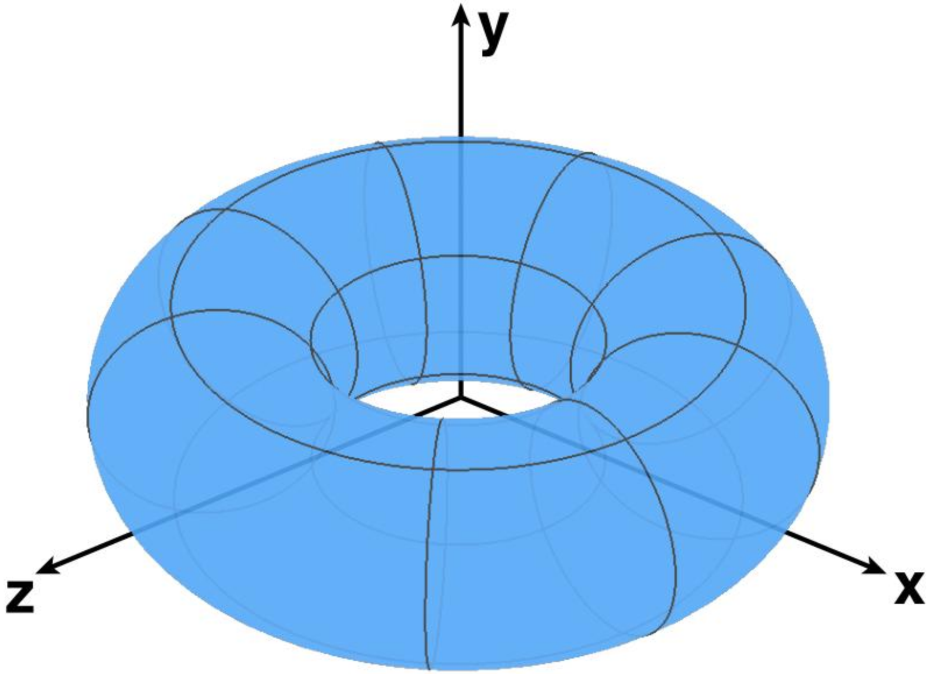
$$f(x, y, z) = (2 - \sqrt{x^2 + y^2})^2 + z^2 - 1$$

What points lie on $f(x, y, z) = 0$?

Implicit Surface – Sampling can be hard

$$f(x, y, z) = (2 - \sqrt{x^2 + y^2})^2 + z^2 - 1$$

What points lie on *f*(*x*, *y*, *z*) = 0 ?



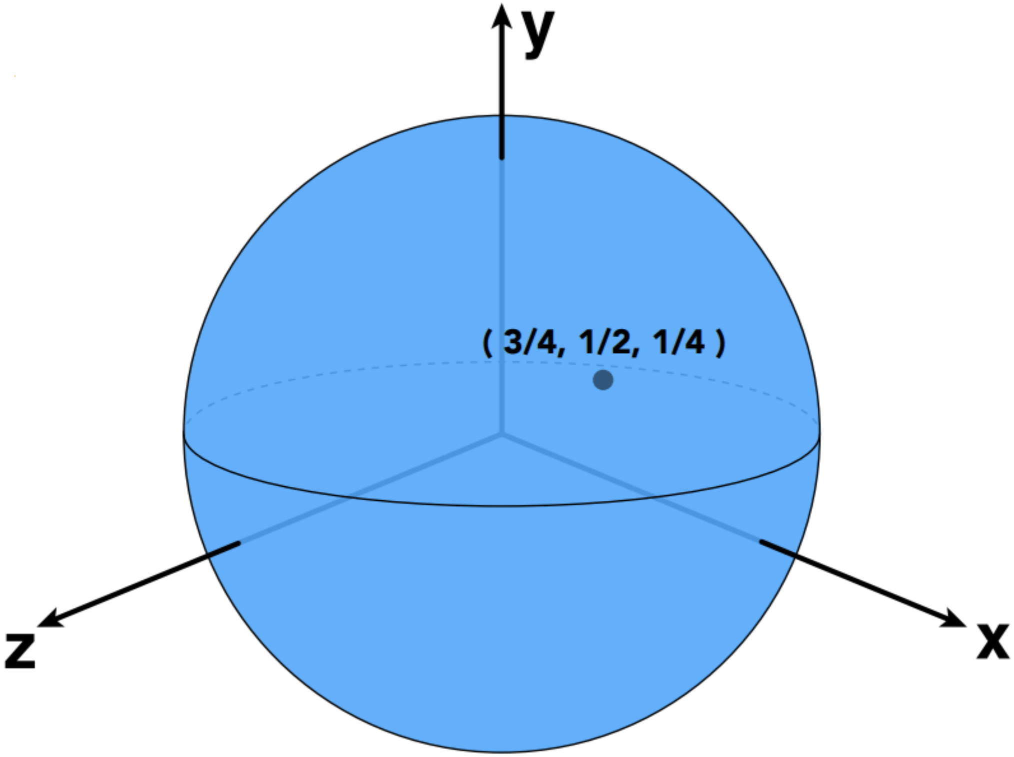
Some tasks are hard

with implicit representations

Check inside or outside

$$f(x, y, z) = x^2 + y^2 + z^2 - 1$$

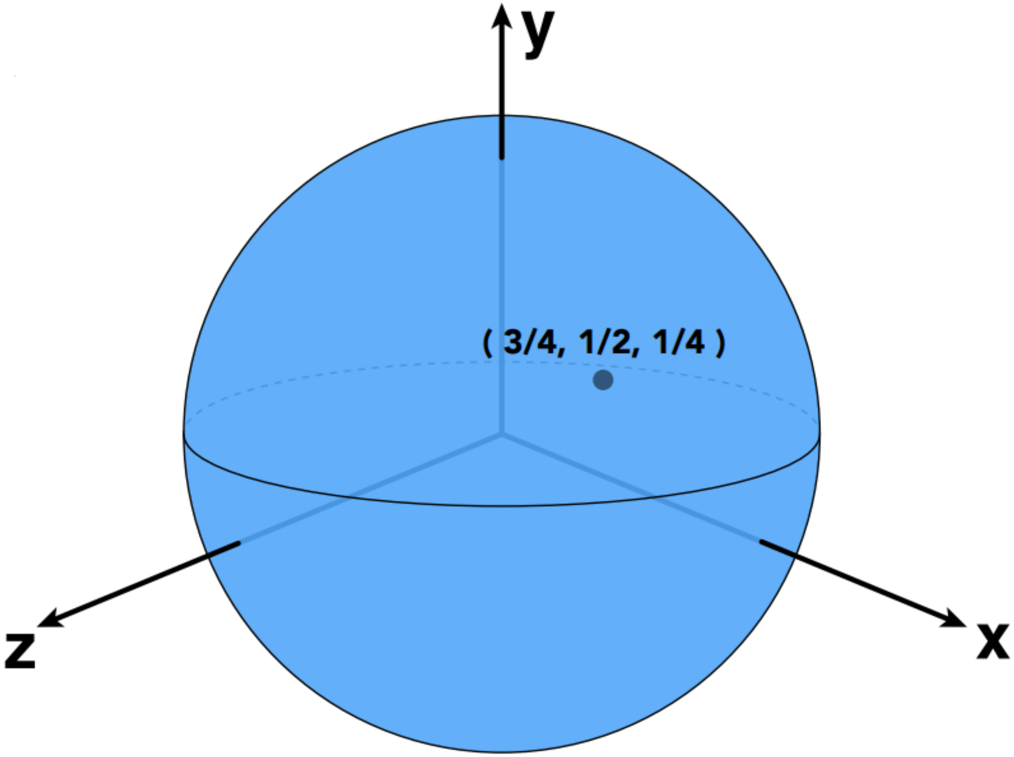
Is (3/4, 1/2, 1/4) inside?



Check inside or outside

$$f(x, y, z) = x^2 + y^2 + z^2 - 1$$

Is (3/4, 1/2, 1/4) inside?



Just plug it in:

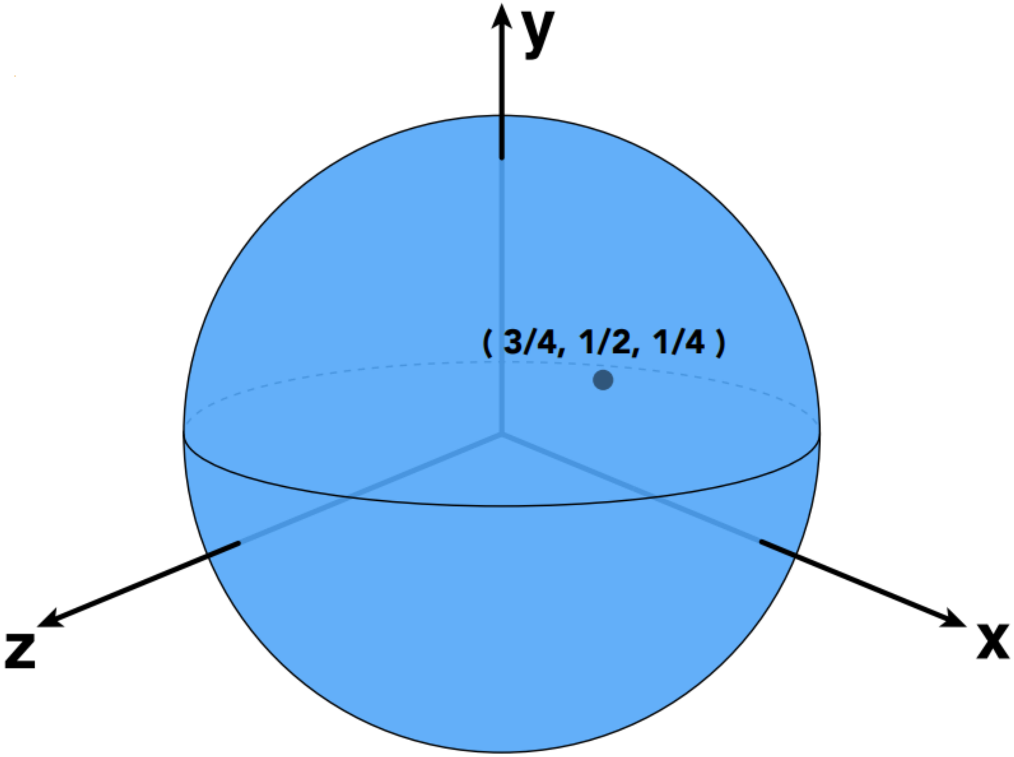
$f(x,y,z) = -1/8 < 0$

inside

Check inside or outside

$$f(x, y, z) = x^2 + y^2 + z^2 - 1$$

Is (3/4, 1/2, 1/4) inside?



Just plug it in:

$f(x,y,z) = -1/8 < 0$

inside

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

Explicit Representations of Geometry

- All points are given directly
- E.g. points on sphere are

$$(\cos(u)\sin(v), \sin(u)\sin(v), \cos(v)),$$

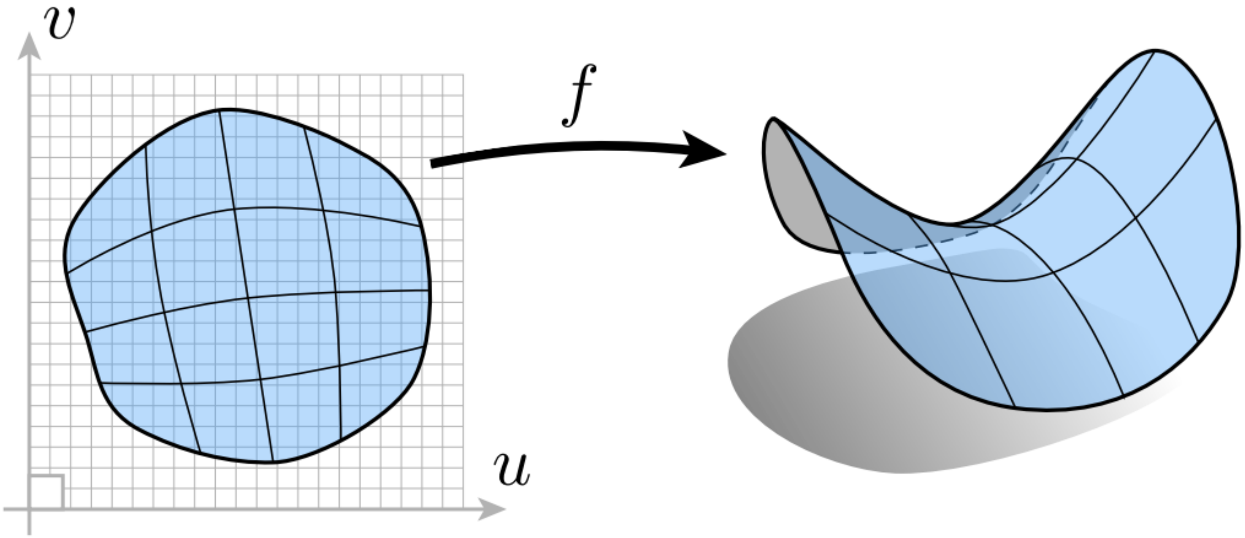
$$\text{for } 0 \leq u < 2\pi \text{ and } 0 \leq v \leq \pi$$

Explicit Representations of Geometry

- All points are given directly
- E.g. points on sphere are

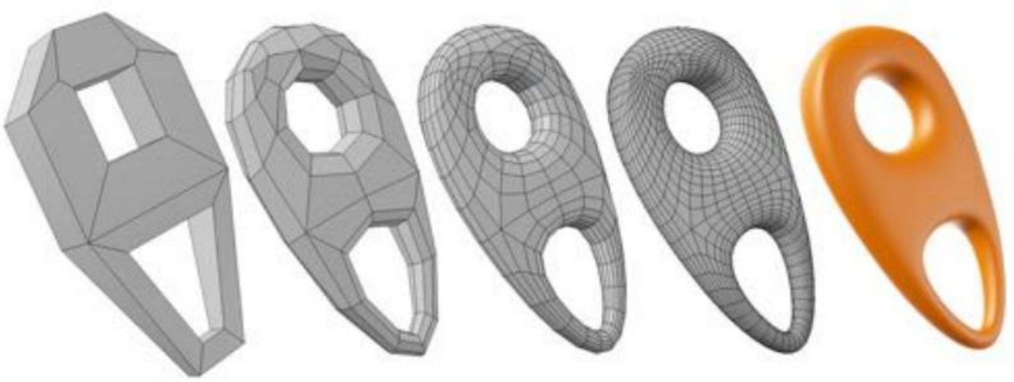
$$(\cos(u)\sin(v), \sin(u)\sin(v), \cos(v)),$$
$$\text{for } 0 \le u < 2\pi \text{ and } 0 \le v \le \pi$$

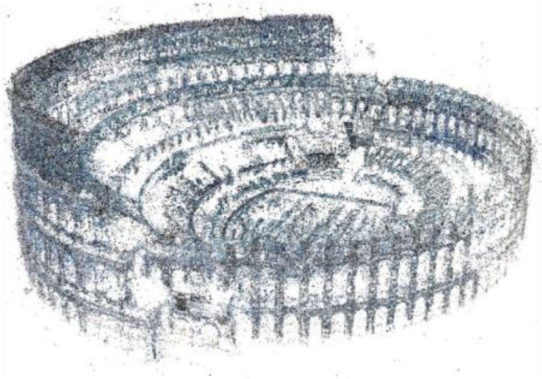
- More generally: $f : \mathbb{R}^2 \to \mathbb{R}^3; (u, v) \mapsto (x, y, z)$

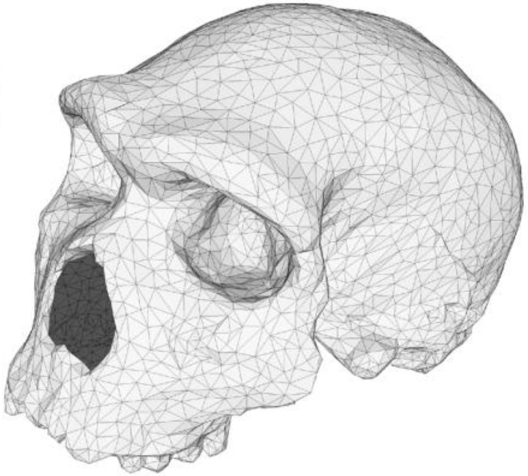


Many explicit representations in graphics

- Triangle meshes
- Polygon meshes
- Subdivision surfaces
- NURBS
- Point clouds
- …

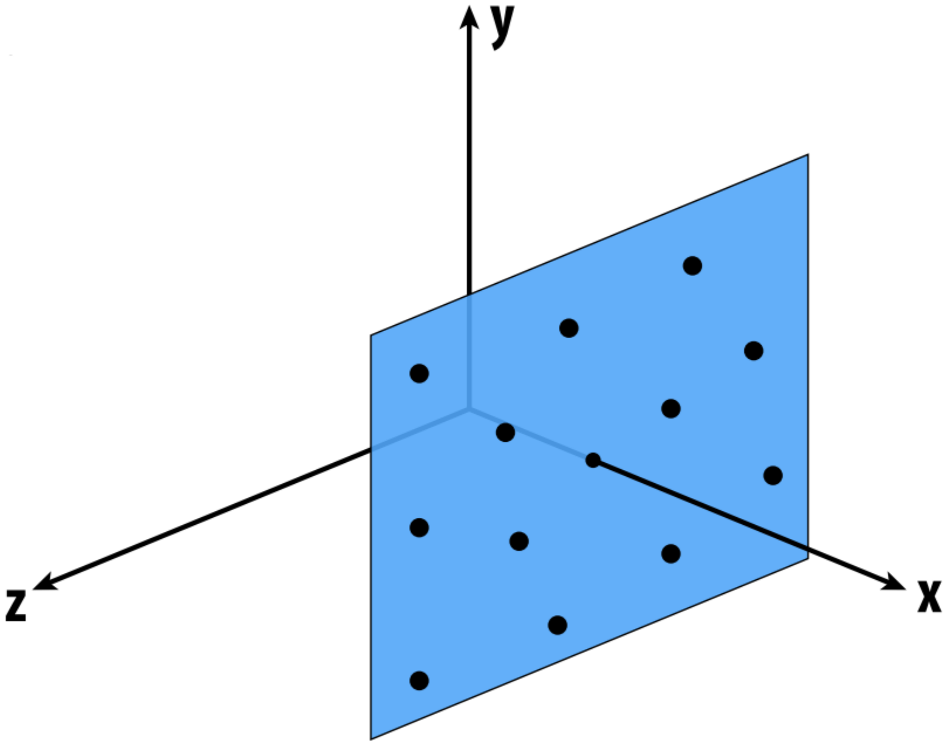






Sampling an explicit surface

- The surface is $f(u, v) = (1.23, u, v)$
- Just plug in any values u, v!



Explicit representations make some tasks easy

Check inside or outside

- The surface is

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

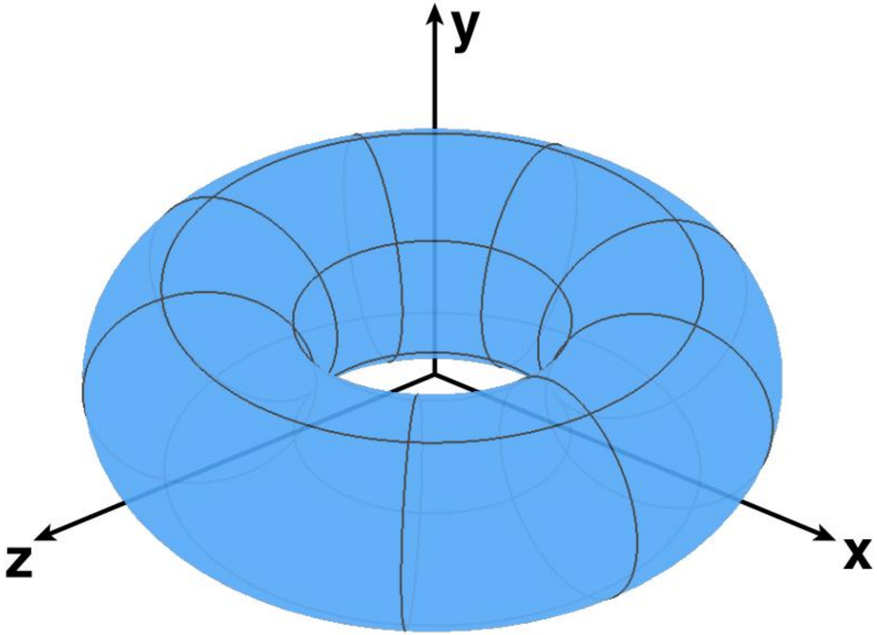
- How about the point (1.96, -0.39, 0.9)?

Check inside or outside

- The surface is

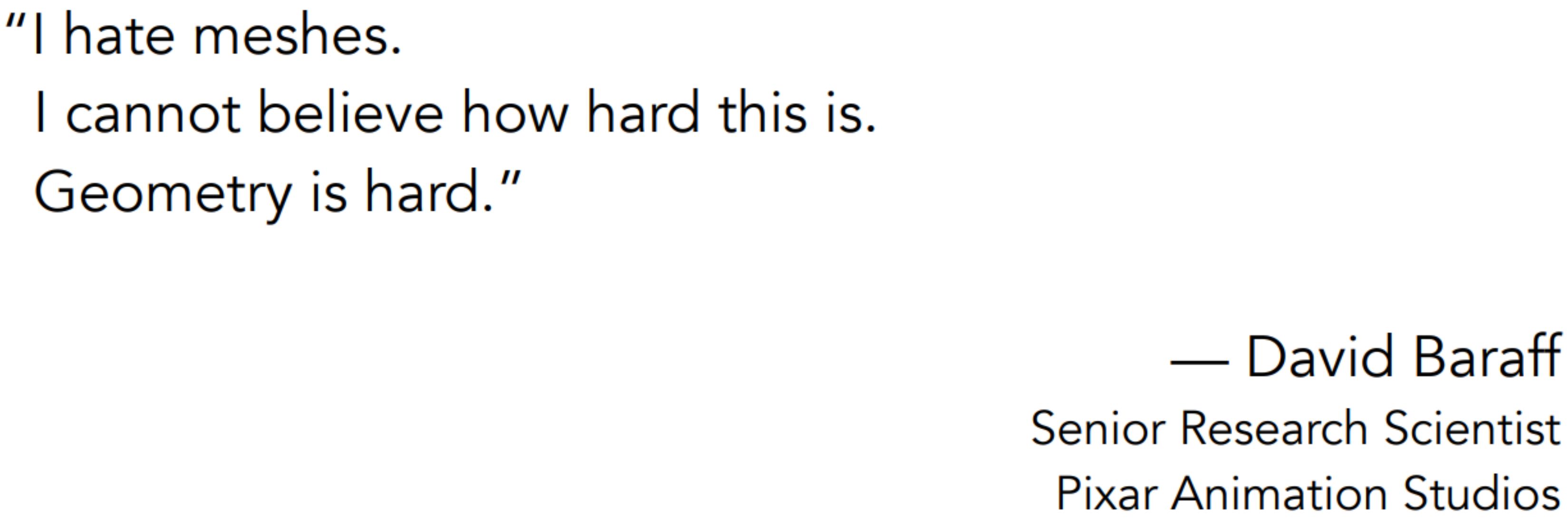
$$f(u, v) = ((2+\cos u)\cos v, (2+\cos u)\sin v, \sin u)$$

- How about the point (1.96, -0.39, 0.9)?



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

No “Best” Representations – Geometry is Hard!



> “I hate meshes.
> I cannot believe how hard this is.
> Geometry is hard.”

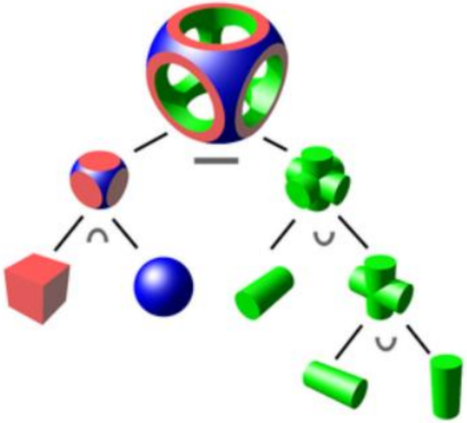
— David Baraff
Senior Research Scientist
Pixar Animation Studios

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.

Many implicit representations in graphics

- Algebraic surfaces
- Constructive solid geometry
- Level set methods
- Blobby surfaces
- Fractals
- …



Algebraic Surfaces (Implicit)

- Surface is zero set of a polynomial in *x*, *y*, *z*
- Examples:

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

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

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

Algebraic Surfaces (Implicit)

- Surface is zero set of a polynomial in *x*, *y*, *z*
- Examples:

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

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

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

Algebraic Surfaces (Implicit)

- Surface is zero set of a polynomial in *x*, *y*, *z*
- Examples:

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

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

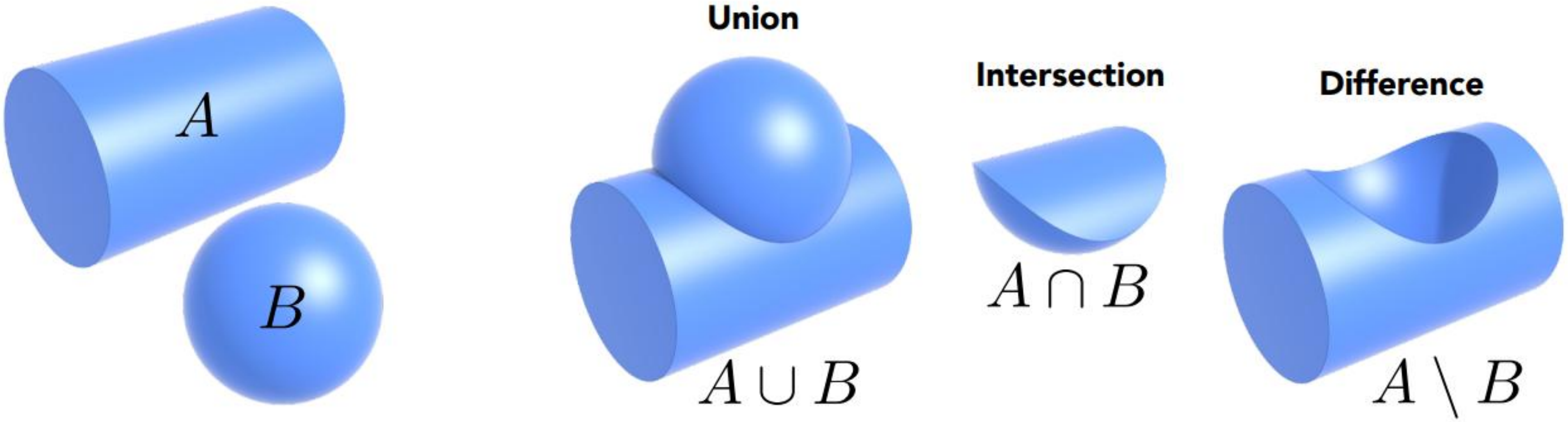
$$(x^2+\frac{9y^2}{4}+z^2-1)^3 = x^2z^3+\frac{9y^2z^3}{80}$$

- What about more complicated shapes?

- Very hard to come up with polynomials!

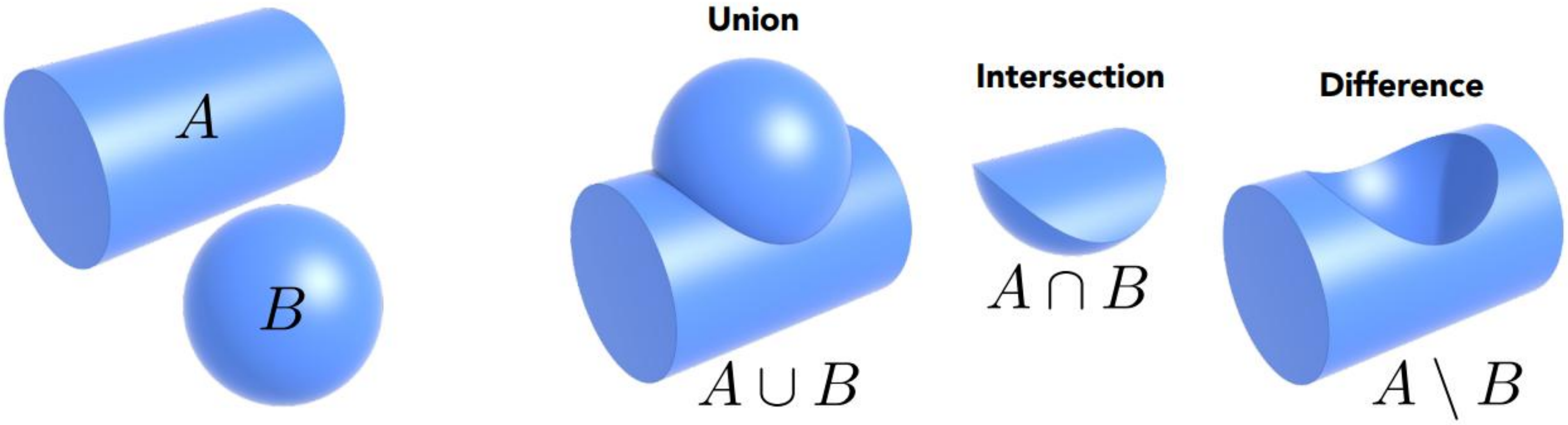
Constructive Solid Geometry (Implicit)

- Build more complicated shapes via Boolean operations
- Basic operations

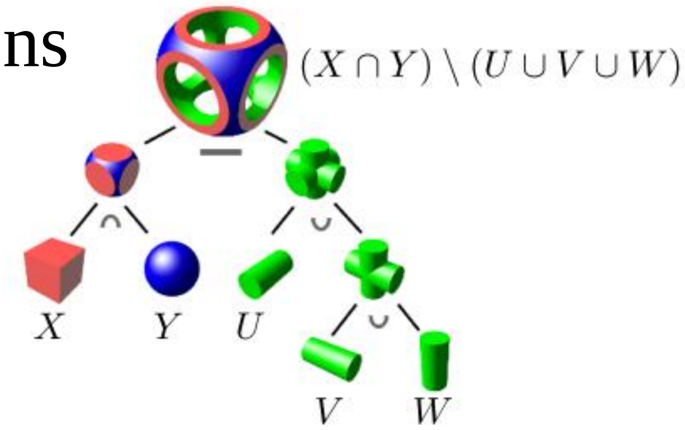


Constructive Solid Geometry (Implicit)

- Build more complicated shapes via Boolean operations
- Basic operations

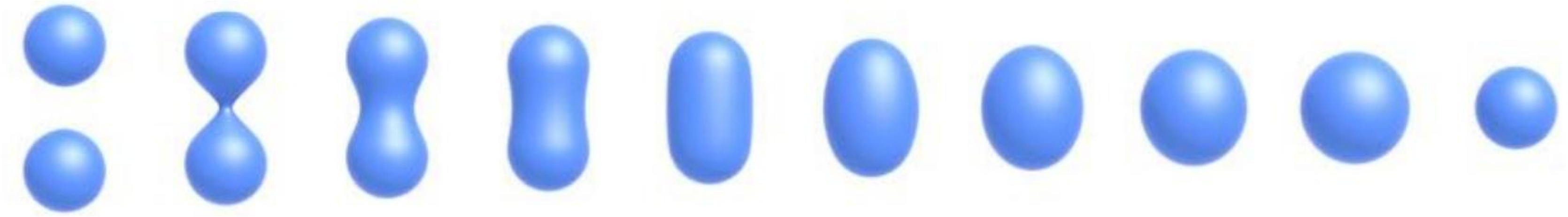


- Then chain together expressions



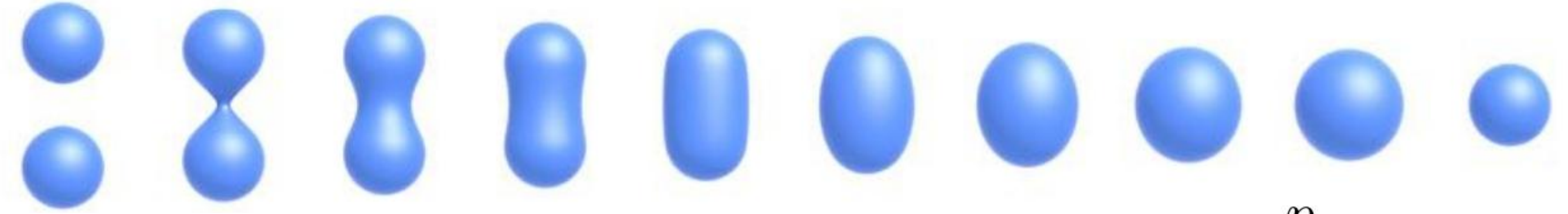
Blobby Surfaces (Implicit)

- Instead of Booleans, gradually blend surfaces together using distance functions:
  - Giving minimum distance (could be signed distance) from any where to object

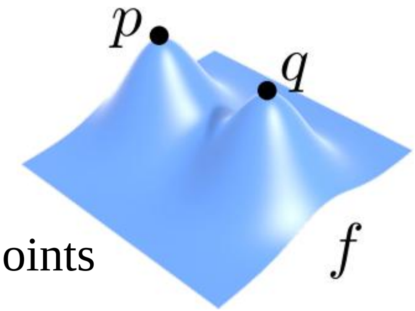


Blobby Surfaces (Implicit)

- Instead of Booleans, gradually blend surfaces together using distance functions:
  - Giving minimum distance (could be signed distance) from any where to object

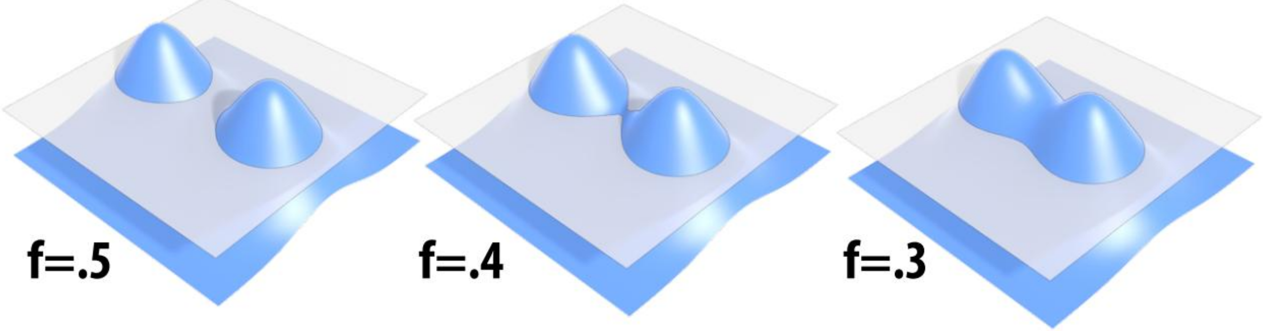


- Easier to understand in 2D



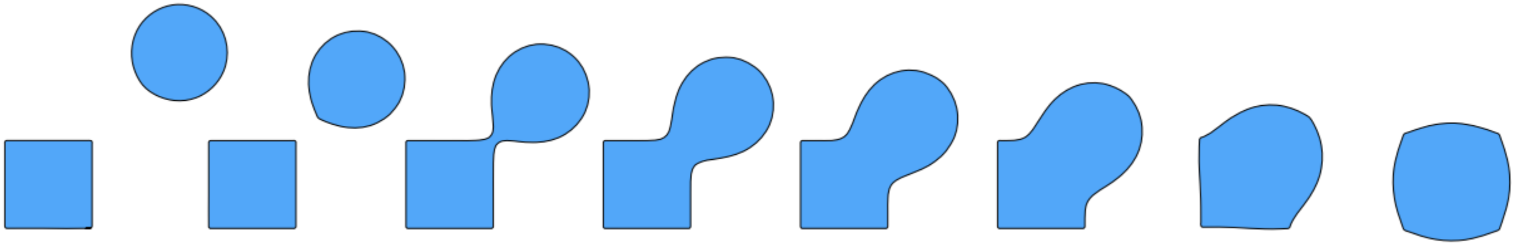
$\phi_p(x) := e^{-|x-p|^2}$ Gaussian centered at *p*

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



Blending Distance Functions (Implicit)

- A distance functions gives distance to closet point on object
- Can blend any two distance functions d_1, d_2



- Similar strategy to points, though many possibilities

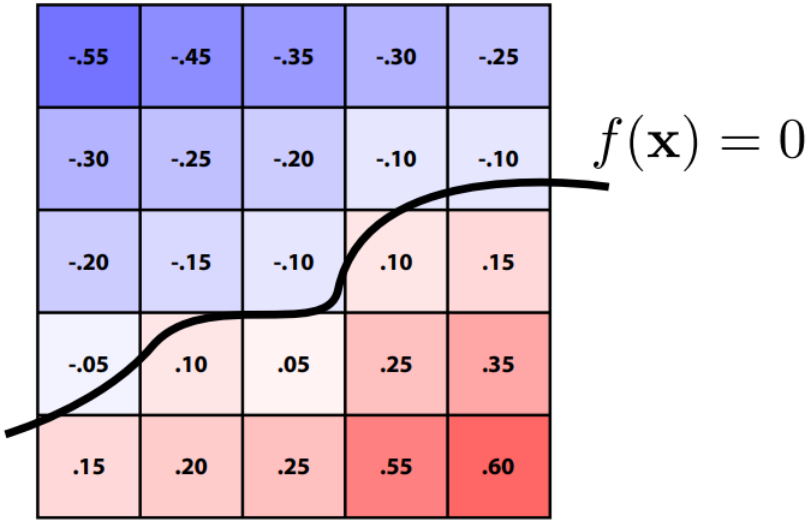
$$f(x) := e^{d_1(x)^2} + e^{d_2(x)^2} - \frac{1}{2}$$

Scene of Pure Distance Functions

see https://iquilezles.org/articles/raymarchingdf/

Level Set Methods (Implicit)

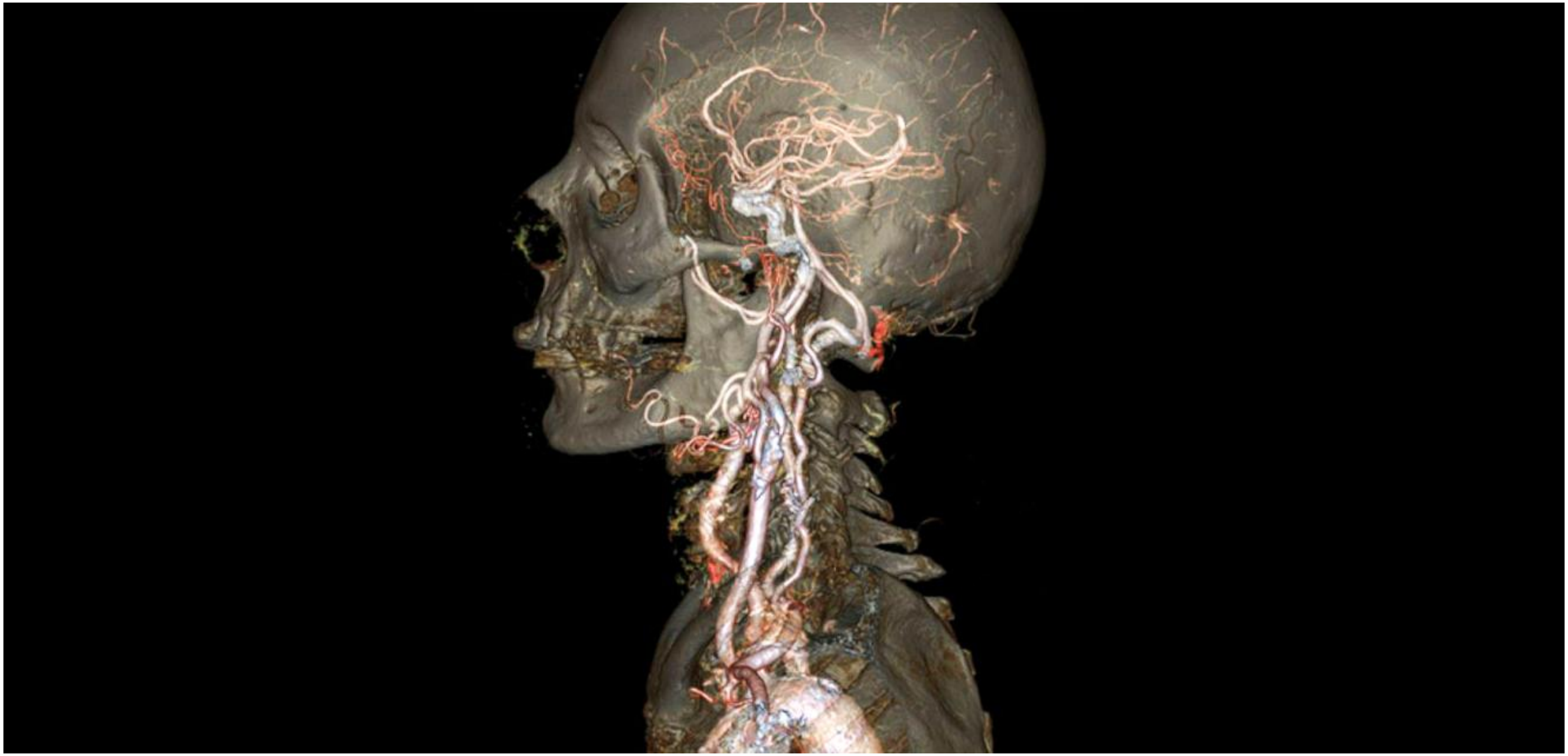
- Closed-form equations are hard to describe complex shapes
- Alternative: store a grid of values approximating function
- Surface is found where interpolated values equal zero
- Provides much more explicit control over shape (like a texture)
- Unlike closed-form expression, run into problems of aliasing!



-.55	-.45	-.35	-.30	-.25
-.30	-.25	-.20	-.10	-.10
-.20	-.15	-.10	.10	.15
-.05	.10	.05	.25	.35
.15	.20	.25	.55	.60

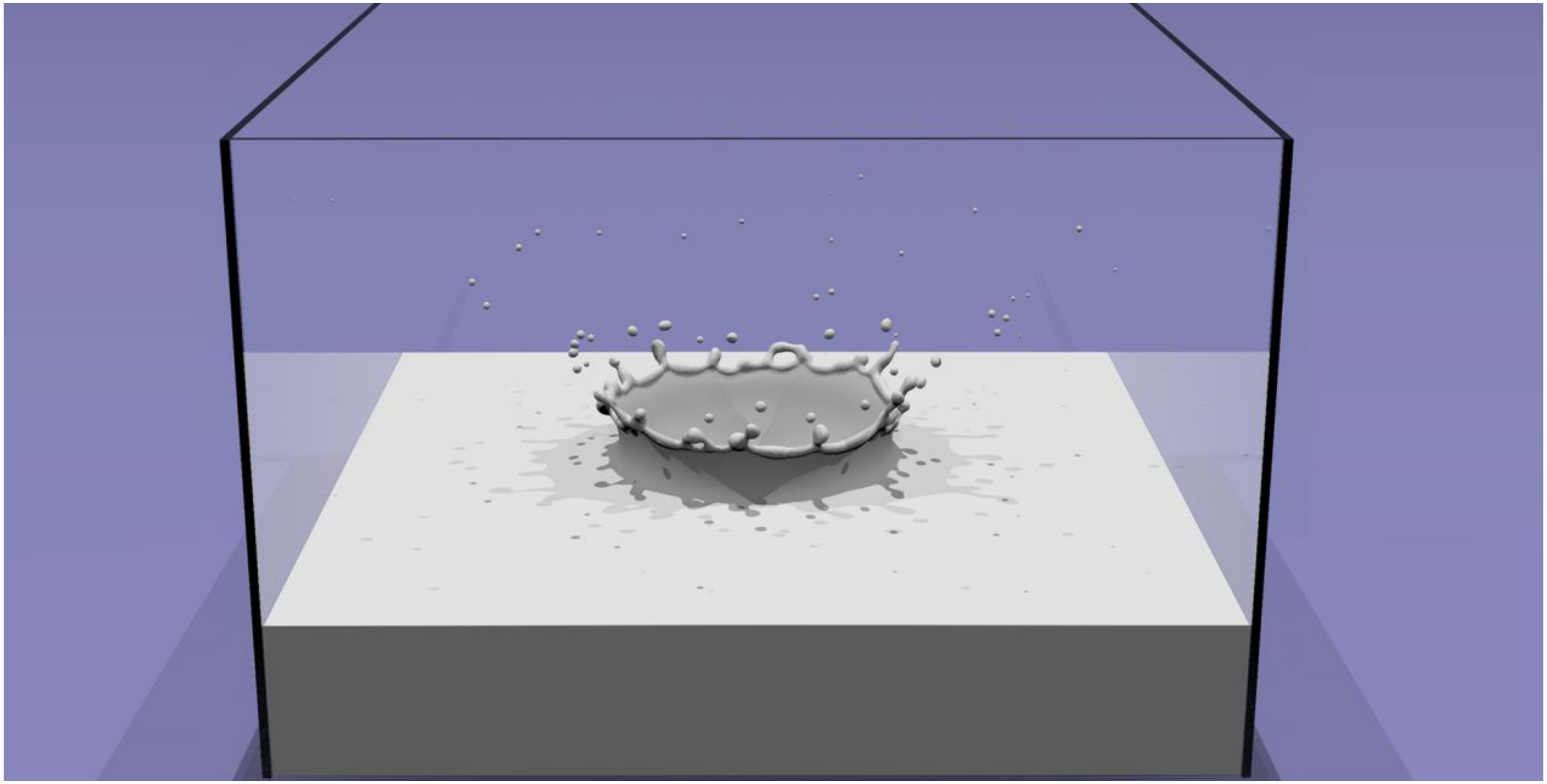
Level Sets from Medical Data (CT, MRI, etc.)

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



Level Sets in Physical Simulation

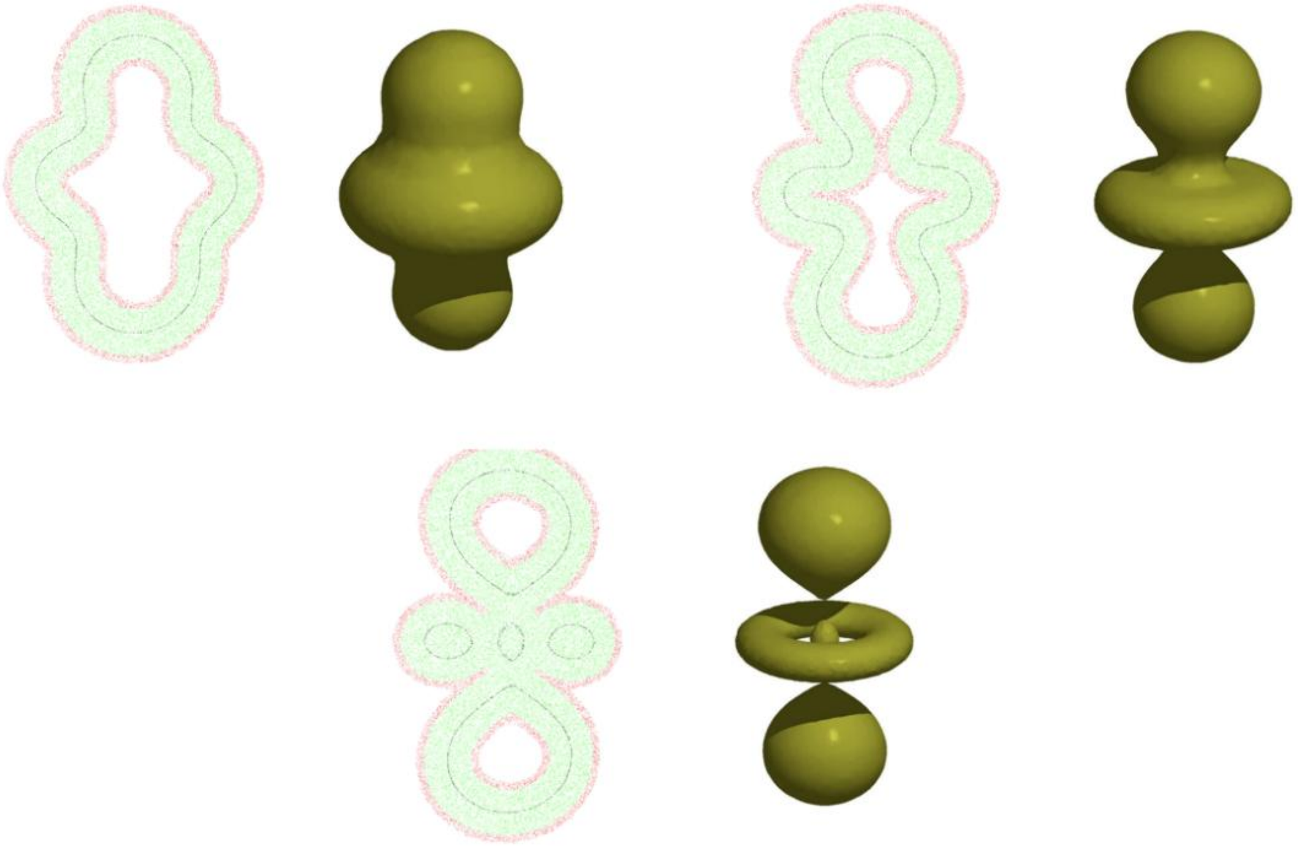
Level set encodes distance to air-liquid boundary



see http://physbam.stanford.edu

Level Set Storage

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



Fractals (Implicit)

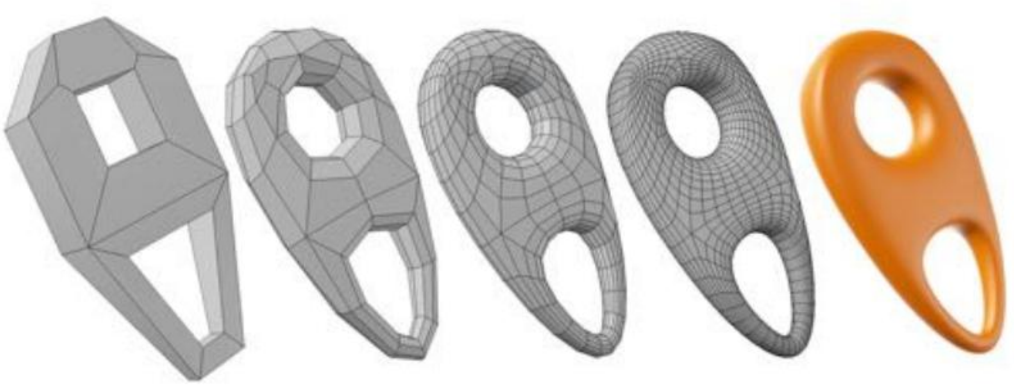
- No precise definition; exhibit self-similarity, detail at all scales
- New “language” for describing natural phenomena
- Hard to control shape!

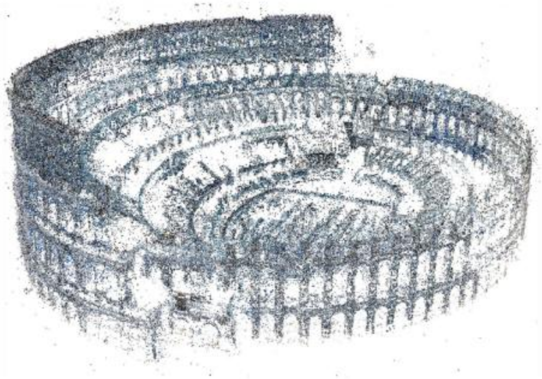
Implicit Representations - Pros & Cons

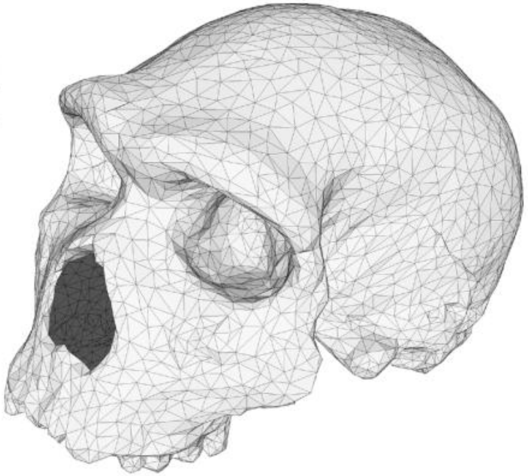
- Pros:
  - description can be very compact (e.g., a polynomial)
  - easy to determine if a point is in our shape (just plug it in!)
  - other queries may also be easy (e.g., distance to surface)
  - for simple shapes, exact description/no sampling error
  - easy to handle changes in topology (e.g.,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

Many explicit representations in graphics

- Triangle meshes
- Polygon meshes
- Subdivision surfaces
- NURBS
- Point clouds
- …

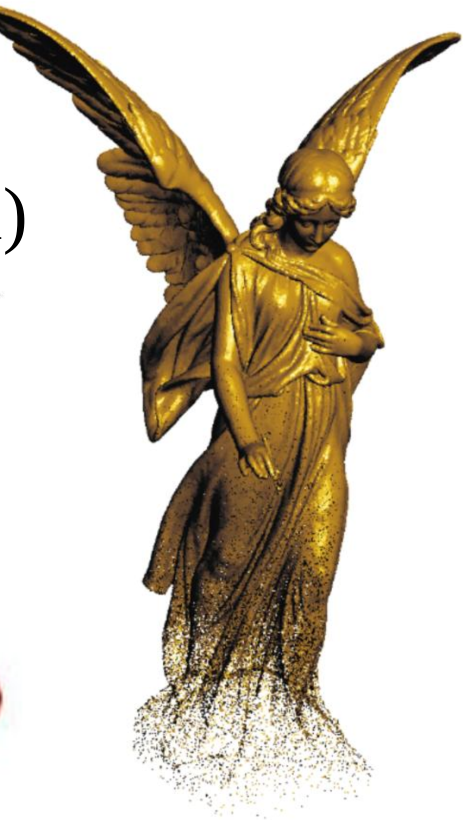


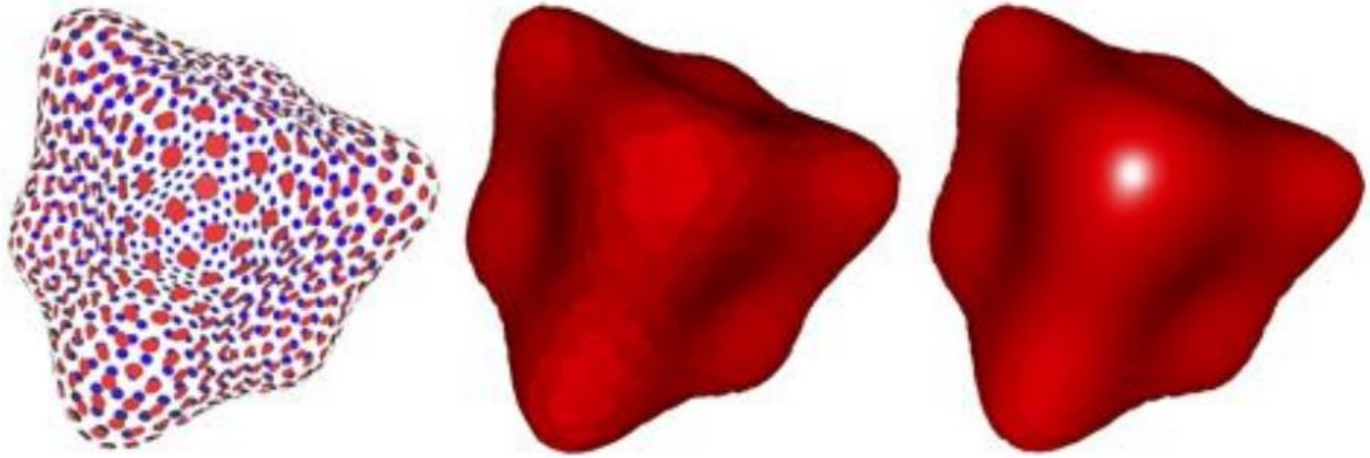




Point Cloud (Explicit)

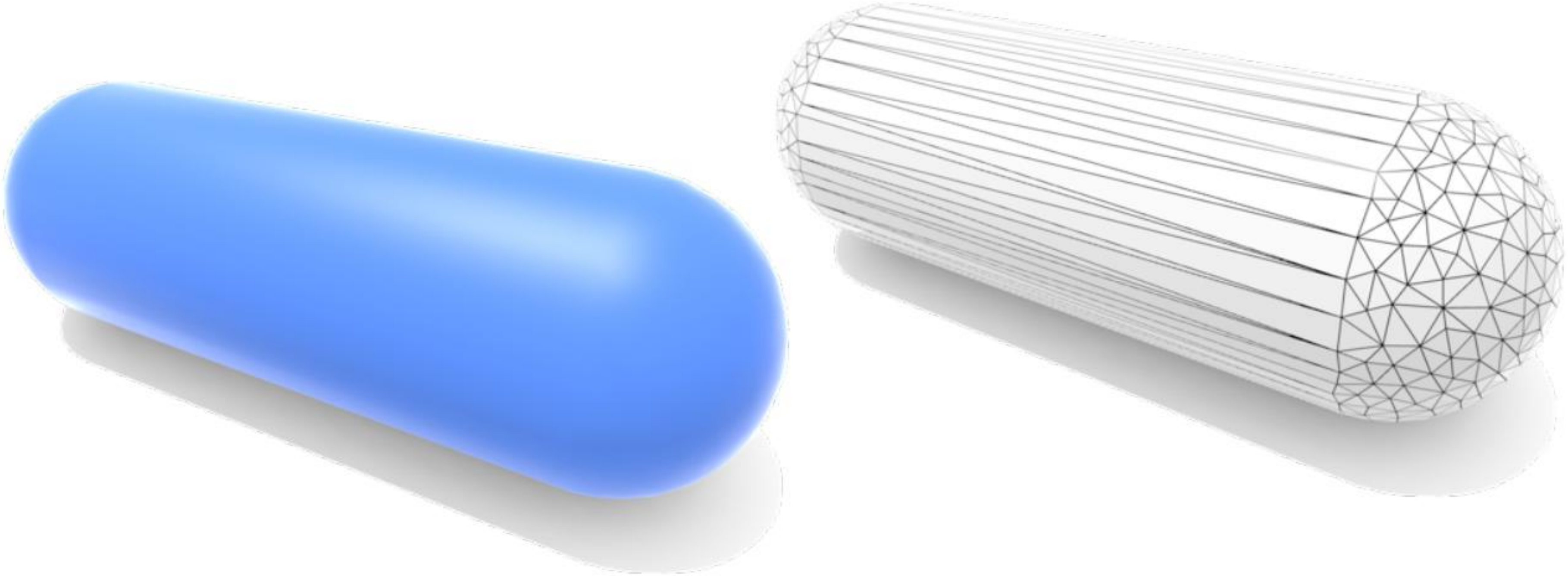
- Easiest representation: list of points (*x*, *y*, *z*)
- Often converted into polygon mesh
- Easily represent any kind of geometry
- Easy to draw dense cloud (>>1 point/pixel)
- Hard to interpolate undersampled regions





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

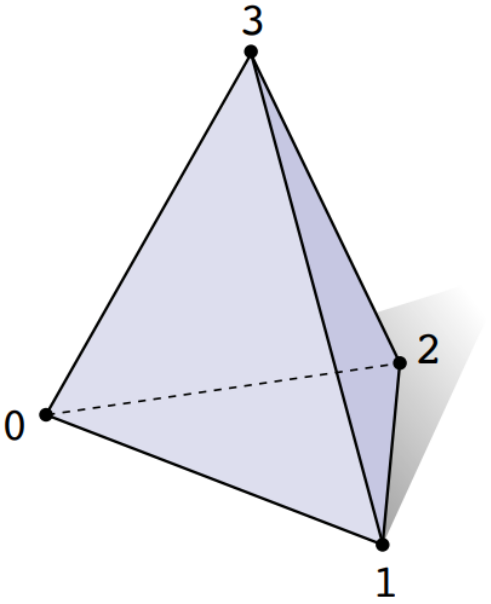


Triangle Mesh (Explicit)

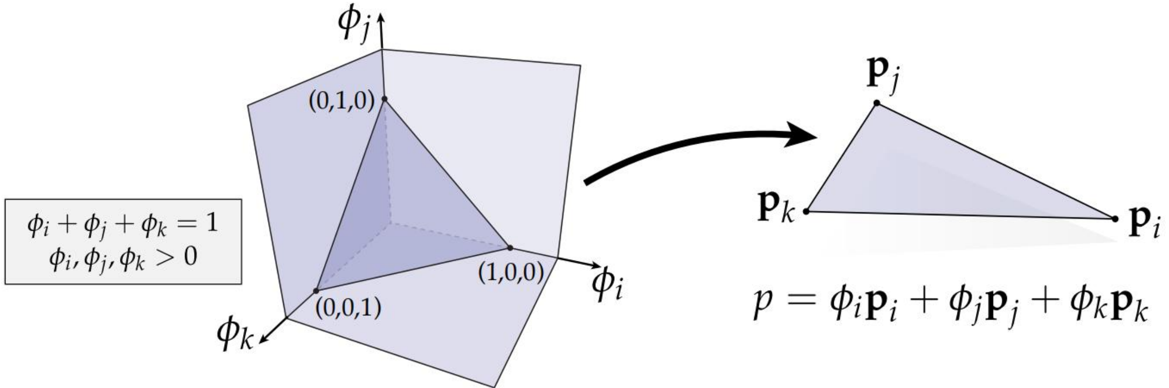
- Store vertices as triples of coordinates (*x*,*y*,*z*)
- Store triangles as triples of indices (*i*,*j*,*k*)
- E.g., tetrahedron vertices triangles

	x	y	z
0:	−1	−1	−1
1:	1	−1	1
2:	1	1	−1
3:	−1	1	1

i	j	k
0	2	1
0	3	2
3	0	1
3	1	2



- Use barycentric interpolation to define points inside triangles

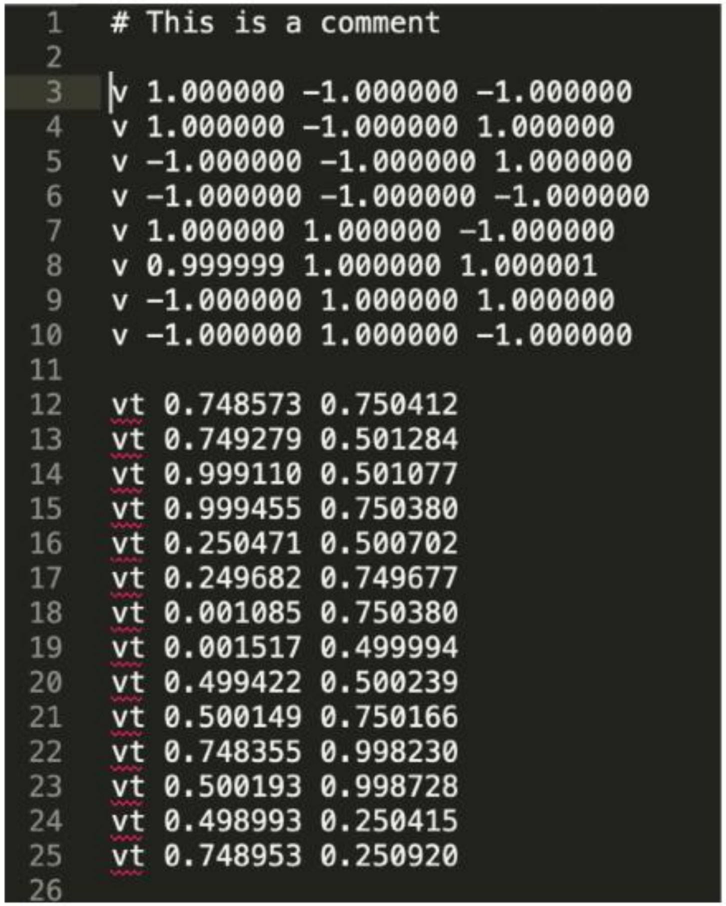


$$\phi_i + \phi_j + \phi_k = 1$$
$$\phi_i, \phi_j, \phi_k > 0$$

$$p = \phi_i \mathbf{p}_i + \phi_j \mathbf{p}_j + \phi_k \mathbf{p}_k$$

The Wavefront Object File(.obj) Format

- Commonly used in Graphics research
- Just a text file that specifies vertices, normals, texture coordinates and their connectivities

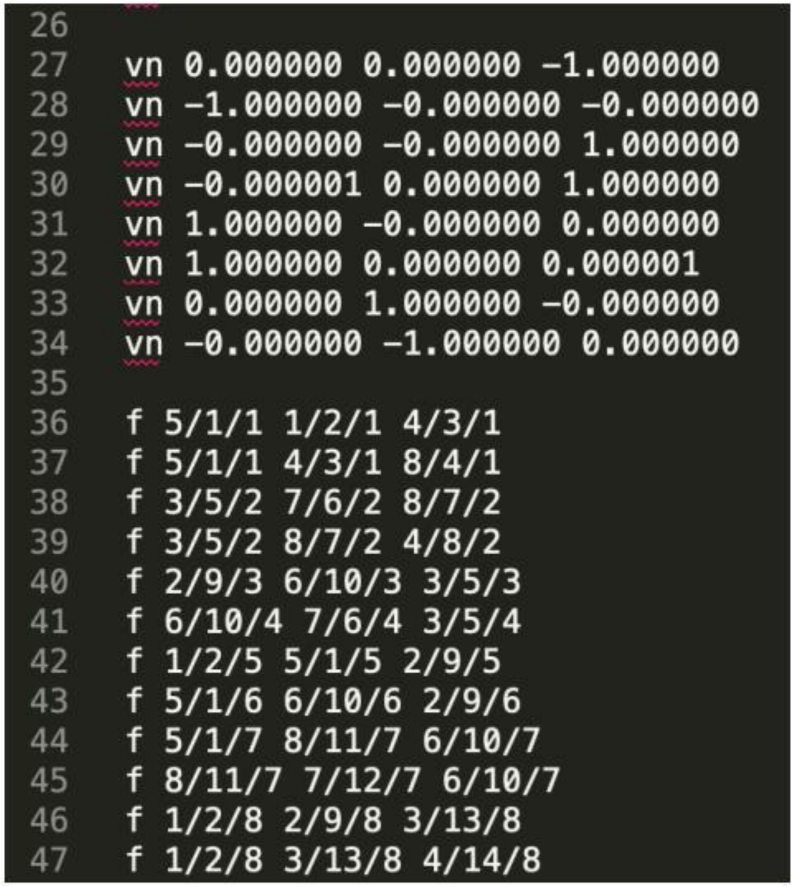


```
# This is a comment

v 1.000000 -1.000000 -1.000000
v 1.000000 -1.000000 1.000000
v -1.000000 -1.000000 1.000000
v -1.000000 -1.000000 -1.000000
v 1.000000 1.000000 -1.000000
v 0.999999 1.000000 1.000001
v -1.000000 1.000000 1.000000
v -1.000000 1.000000 -1.000000

vt 0.748573 0.750412
vt 0.749279 0.501284
vt 0.999110 0.501077
vt 0.999455 0.750380
vt 0.250471 0.500702
vt 0.249682 0.749677
vt 0.001085 0.750380
vt 0.001517 0.499994
vt 0.499422 0.500239
vt 0.500149 0.750166
vt 0.748355 0.998230
vt 0.500193 0.998728
vt 0.498993 0.250415
vt 0.748953 0.250920

```



```

vn 0.000000 0.000000 -1.000000
vn -1.000000 -0.000000 -0.000000
vn -0.000000 -0.000000 1.000000
vn -0.000001 0.000000 1.000000
vn 1.000000 -0.000000 0.000000
vn 1.000000 0.000000 0.000001
vn 0.000000 1.000000 -0.000000
vn -0.000000 -1.000000 0.000000

f 5/1/1 1/2/1 4/3/1
f 5/1/1 4/3/1 8/4/1
f 3/5/2 7/6/2 8/7/2
f 3/5/2 8/7/2 4/8/2
f 2/9/3 6/10/3 3/5/3
f 6/10/4 7/6/4 3/5/4
f 1/2/5 5/1/5 2/9/5
f 5/1/6 6/10/6 2/9/6
f 5/1/7 8/11/7 6/10/7
f 8/11/7 7/12/7 6/10/7
f 1/2/8 2/9/8 3/13/8
f 1/2/8 3/13/8 4/14/8
```

Bézier Curves (Explicit)

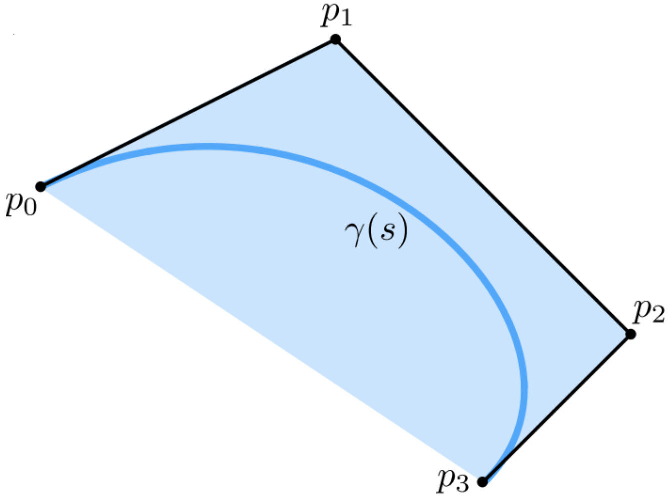
- A Bézier curve is a curve expressed in the Bernstein basis:

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

control points

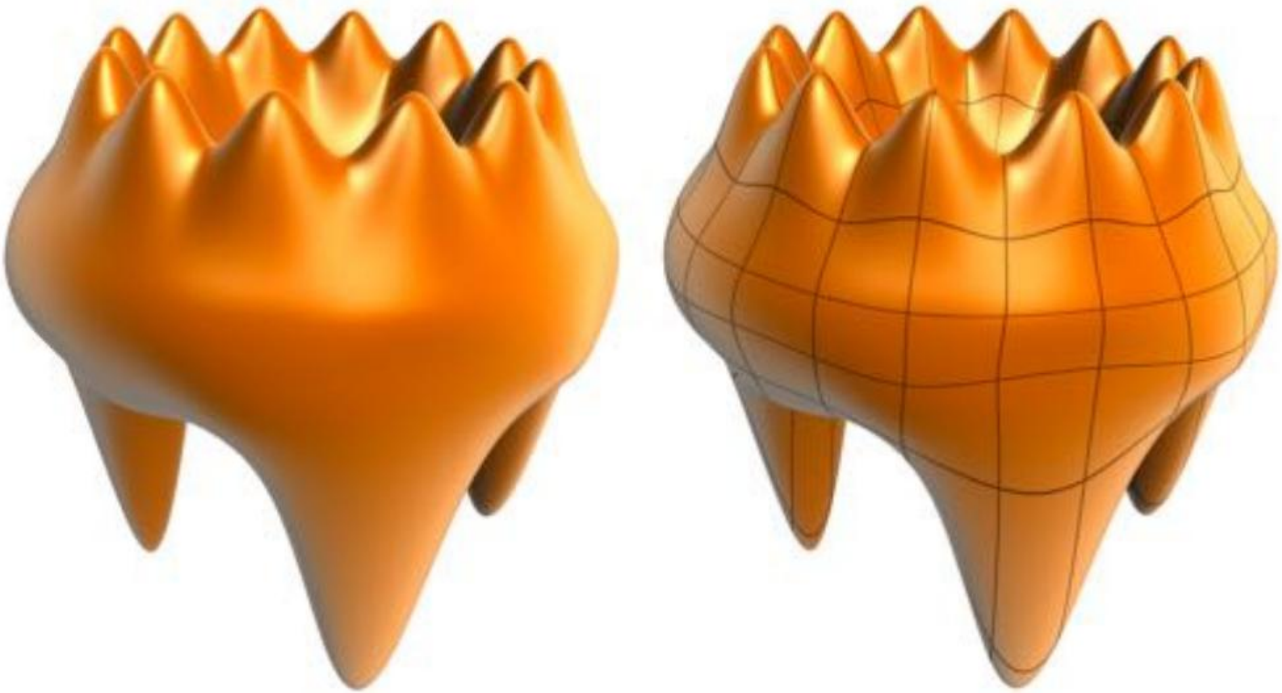
p_1

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



Bézier Surface

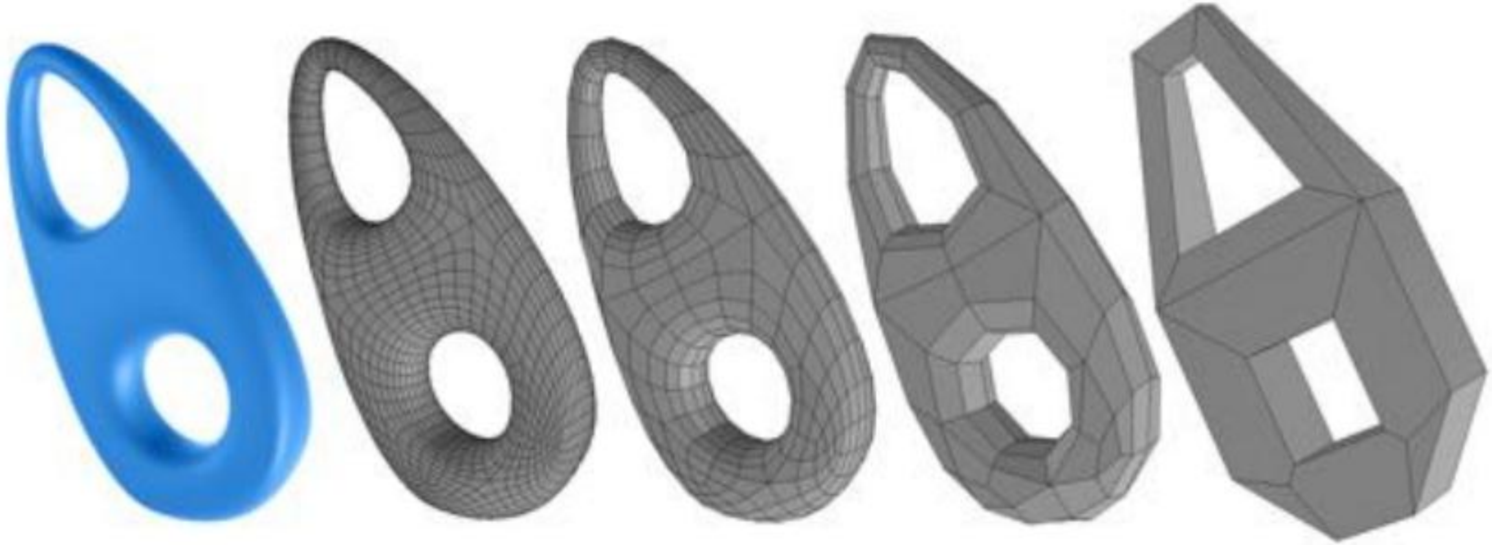
- Just as we connected Bézier curves, can connect Bézier patches to get a surface:



- Very easy to draw: just dice each patch into regular (*u*,*v*) grid!

Subdivision Surfaces (Explicit)

- Start with coarse polygon mesh
- Subdivide each element
- Update vertices via local averaging



谢谢

北京航空航天大学人工智能学院

SCHOOL OF ARTIFICIAL INTELLIGENCE, BEIHANG UNIVERSITY

北京航空航天大學

人工智能学院(人工智能研究院)