Computer Graphics
- Geometry & Its Representation

Junjie Cao @ DLUT

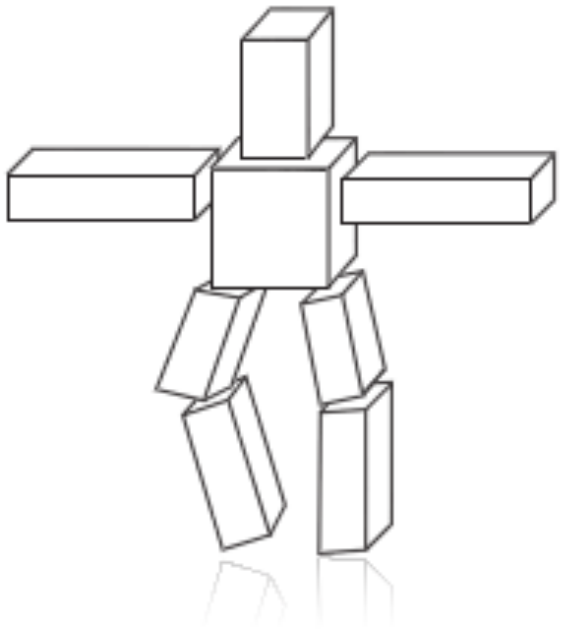
Spring 2019

http://jjcao.github.io/ComputerGraphics/

Music is dynamic, while score is static;
Movement is dynamic, while law is static.

Increasing the complexity of our models

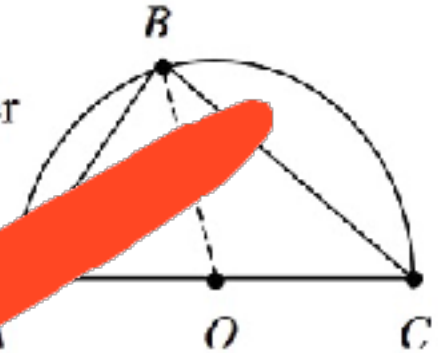
Transformations



Geometry

Materials, lighting, ...

What is geometry?



THEOREM 9.5. Let ΔABC be inscribed in a semicircle with diameter $\overline{A}\ \overline{C}$.

Then $\angle ABC$ [illegible] angle.

Proof:

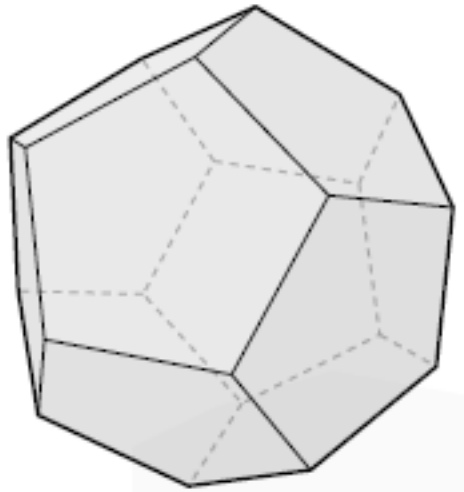
Statement	R[illegible]
1. Draw radius OB. Then $OB = OC = O$[illegible]	[illegible]iven
2. $m\angle OBC = m\angle BCA$ $m\angle OBA = m\angle BAC$	[illegible]osceles Triangle Theorem
3. $m\angle ABC = m\angle OBA + m$[illegible]	3. Angl[illegible] Postulate
4. $m\angle ABC + m\angle BCA$[illegible]$BAC = 180$	4. The sum o[illegible]ngles of a triangle is 180
5. $m\angle ABC + m$[illegible]$+ m\angle OBA = 180$	5. Substitution (lin[illegible]
6. $2\,m\angle AB$[illegible]80	6. Substitution (line 3)
7. $m\angle A$[illegible]$= 90$	7. Division Property of Equali[illegible]
8. $\angle ABC$ is a right angle	8. Definition of Right Angle

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..."

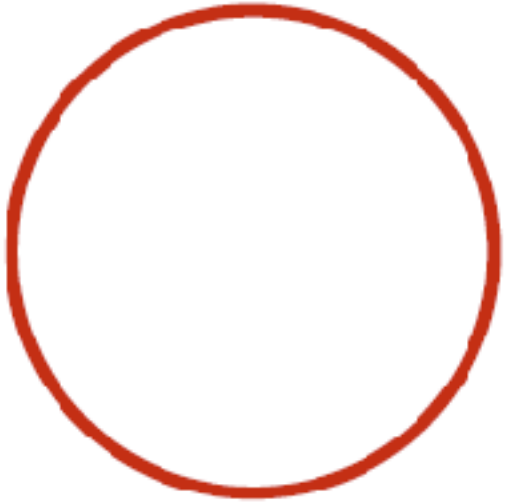
How can we describe geometry?

IMPLICIT

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

LINGUISTIC

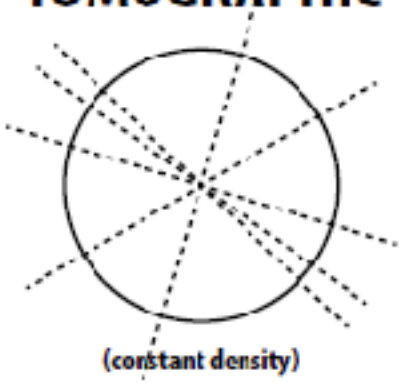
"unit circle"



EXPLICIT

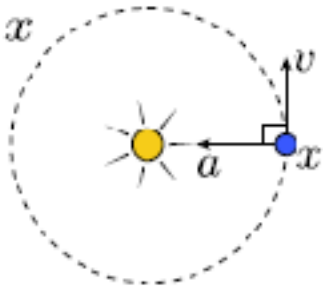
$$(\underbrace{\cos\theta}_{x}, \underbrace{\sin\theta}_{y})$$

TOMOGRAPHIC



DYNAMIC

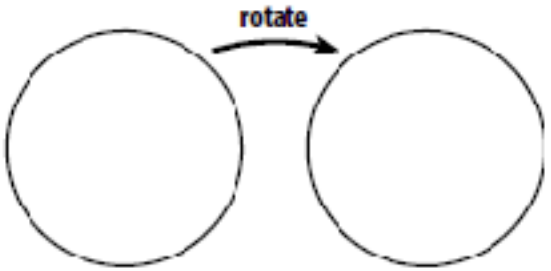
$$\frac{d^2}{dt^2}x = -x$$



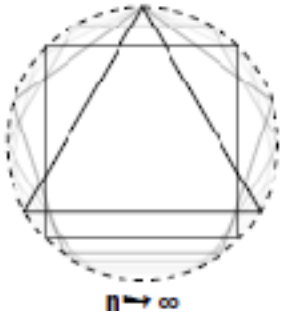
CURVATURE

$$\kappa = 1$$

SYMMETRIC

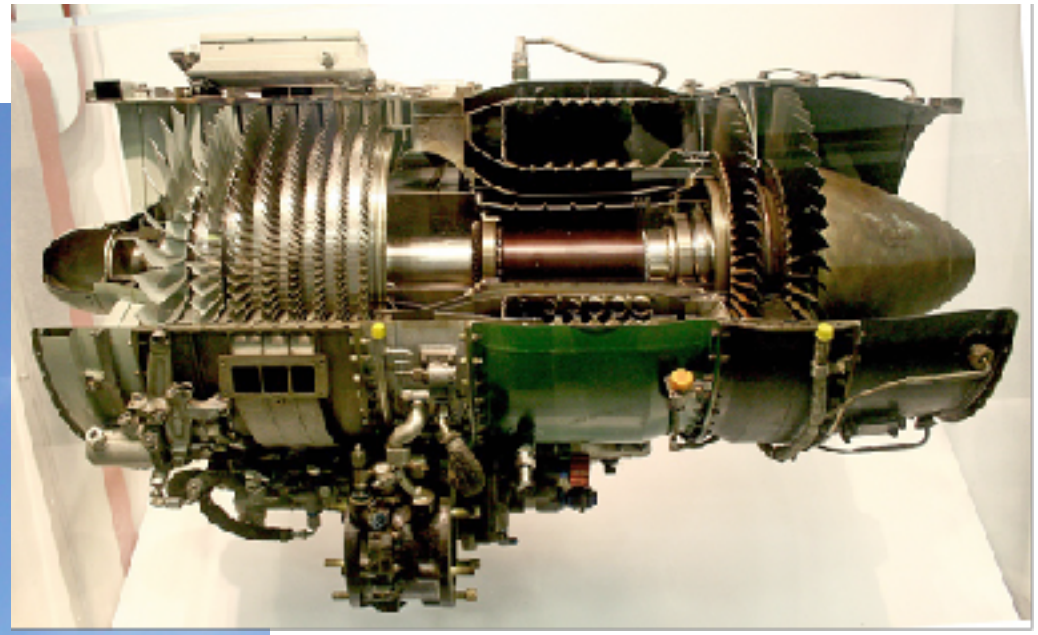


DISCRETE



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

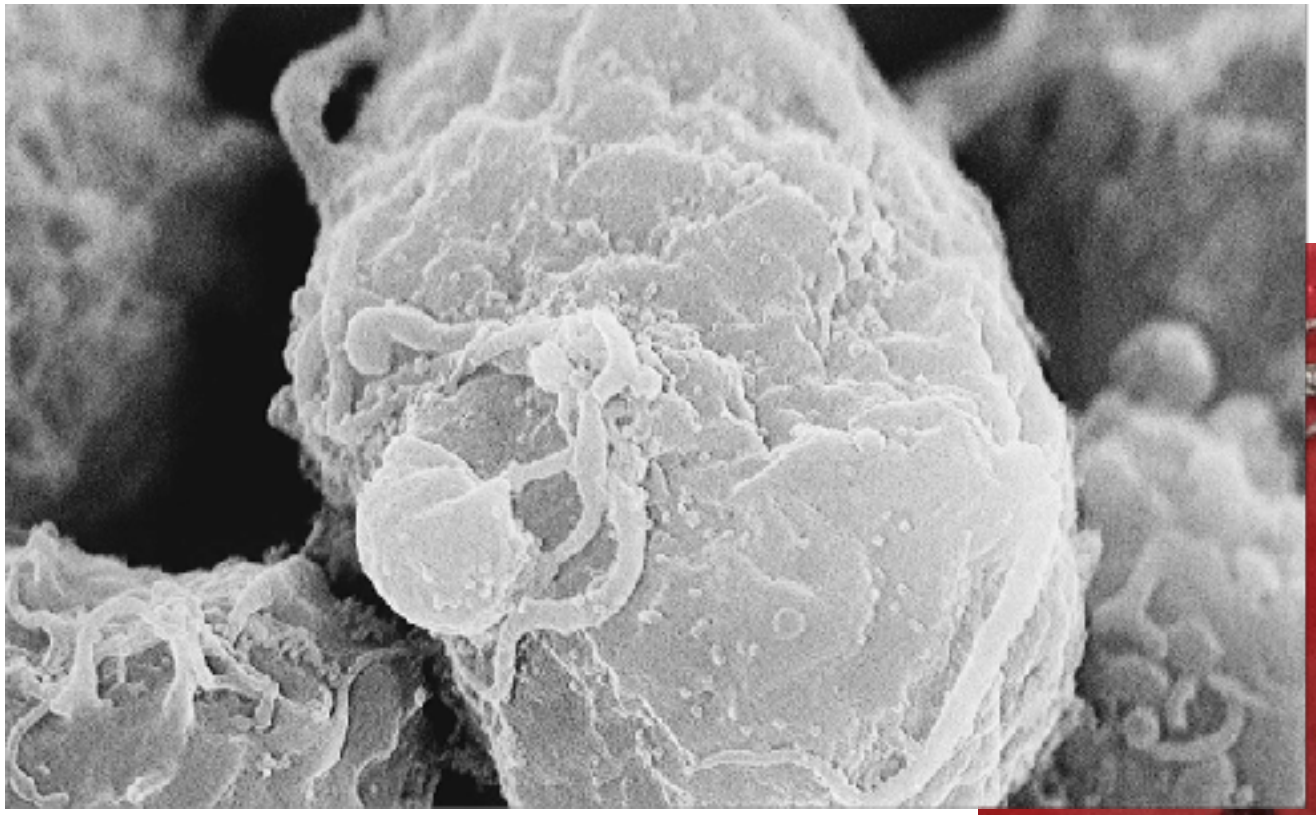
Examples of geometry

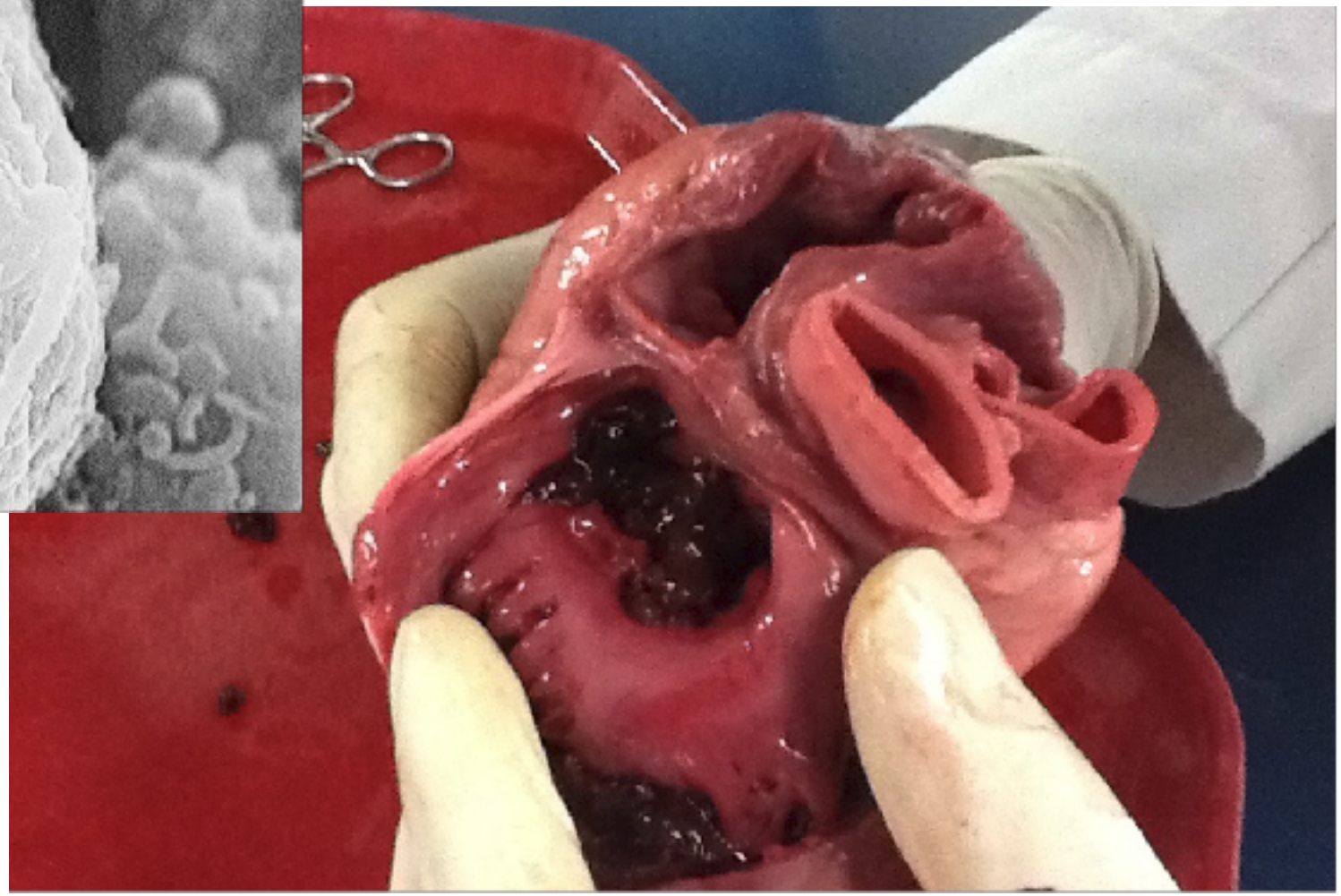


Examples of geometry

Examples of geometry

Examples of geometry

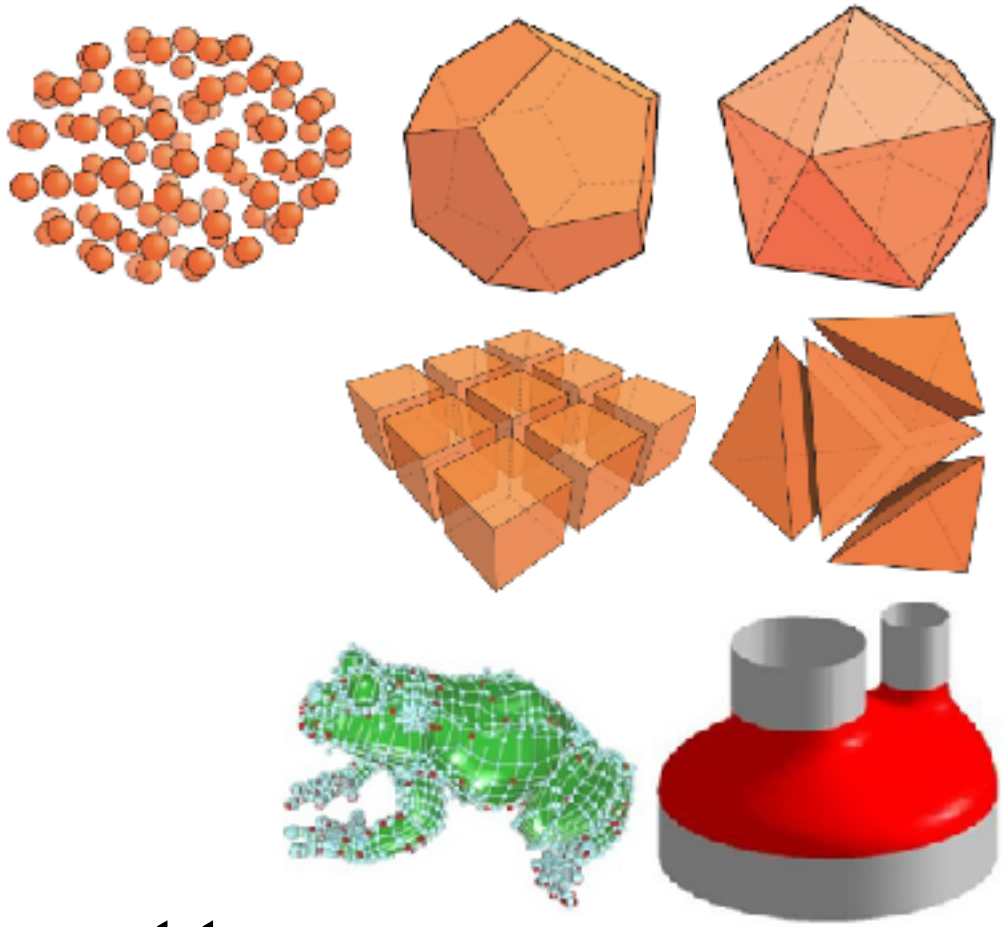




Examples of geometry

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**

Modeling Complex Shapes

- An equation for a sphere is possible, but how about an equation for a rabbit?
- **Complexity is achieved using simple pieces**
  - polygons, parametric surfaces, or implicit surfaces
- High Resolution or Large scenes

- Goals
  - Model anything with arbitrary precision (in principle)
  - Easy to build and modify
  - Efficient computations (for rendering, collisions, etc.)
  - Easy to implement (a minor consideration...)

What do we need from shapes in Computer Graphics?

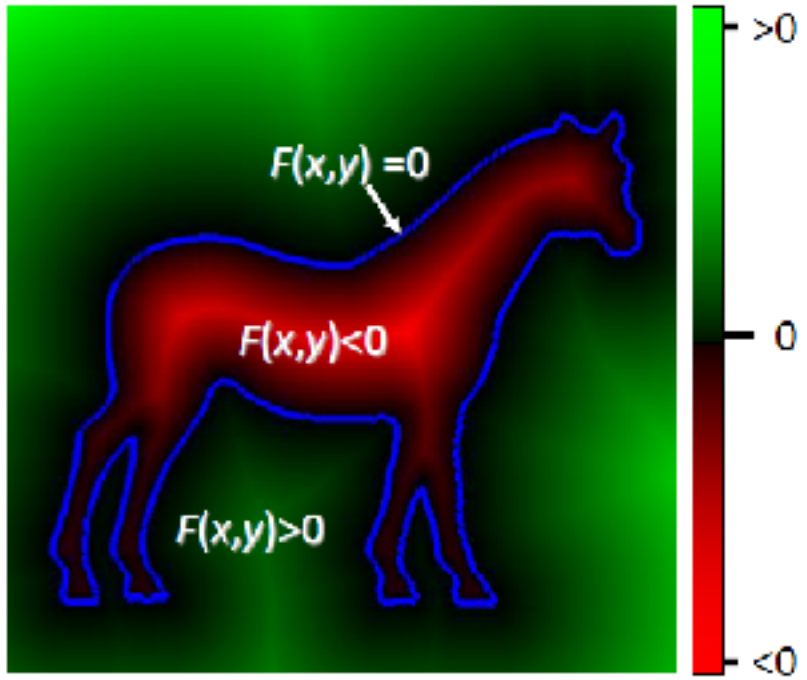
- Local control of shape for modeling
- Ability to model what we need
- Smoothness and continuity
- Ability to evaluate derivatives
- Ability to do collision detection
- Ease of rendering

- **No single technique solves all problems!**

“Implicit” Representations of Geometry

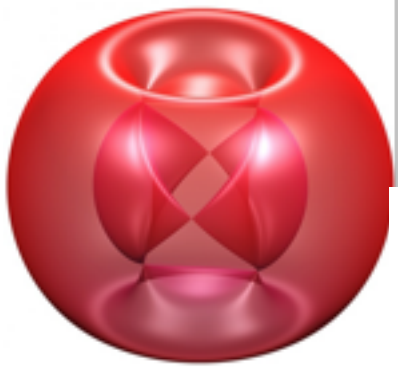
- Points aren’t known directly, but satisfy some relationship
  - E.g., unit sphere is all points x such that **x^2+y^2+z^2=1**
  - More generally, **f(x,y,z) = 0**
- Represent a surface as the zero set of a (regular) function defined in .

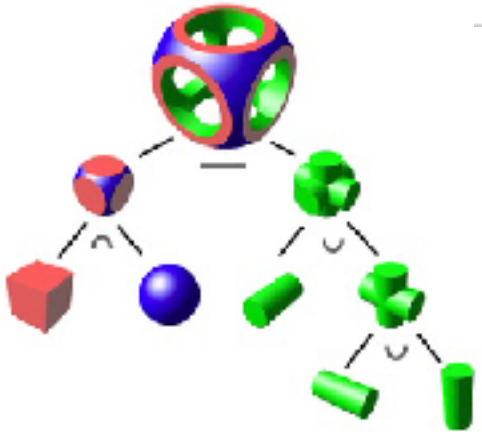
$$K = g^{-1}(0) = \{\mathbf{p} \in \mathrm{R}^3 : g(\mathbf{p}) = 0\}$$



Many implicit representations in graphics

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





(Will see some of these a bit later.)

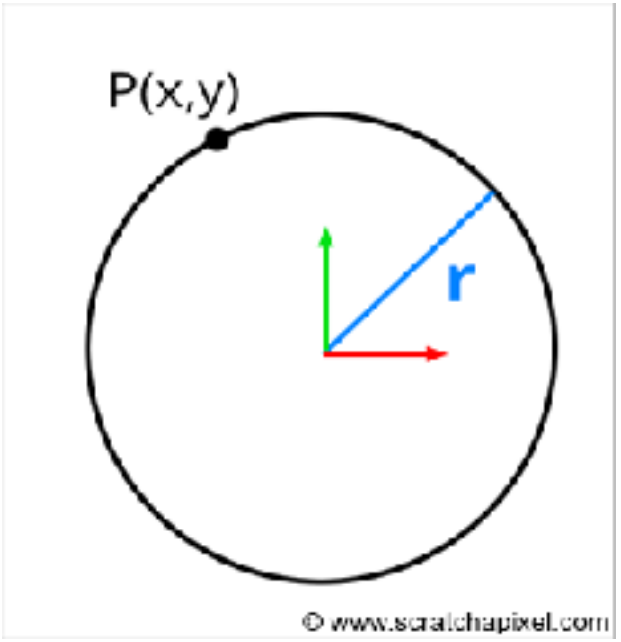
But first, let's play a game:

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

Find *any* point on it.

Give up?

- **My function was** $g(x,y,z) = x^2 + y^2 + z^2 - r^2$



- **Implicit surfaces make some tasks hard (like sampling).**

Let’s play another game.

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

I want to see if a point is inside it.

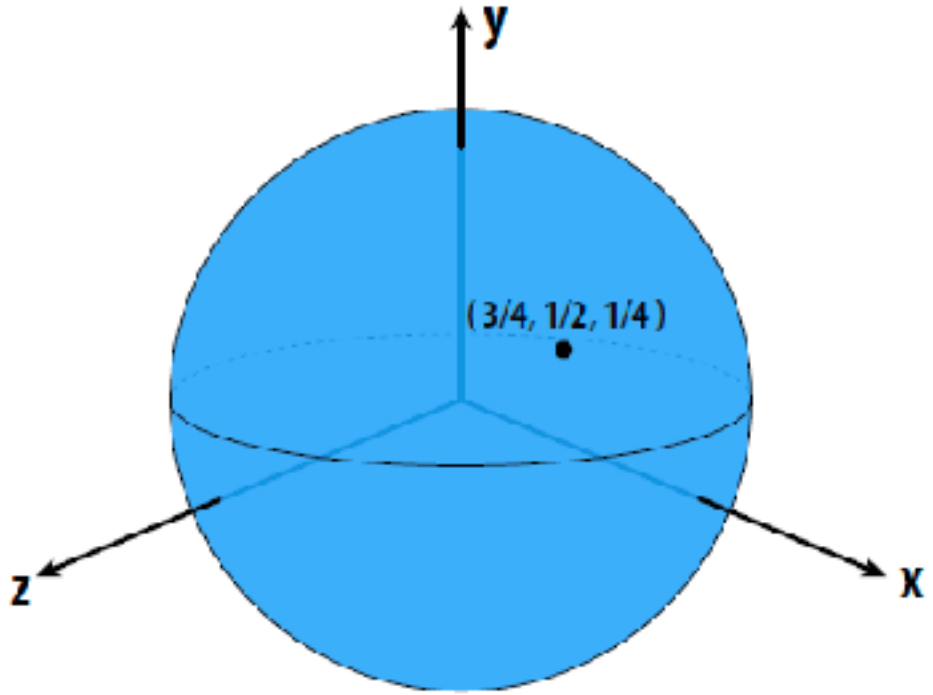
Check if this point is inside the unit sphere

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

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

7/8 < 1

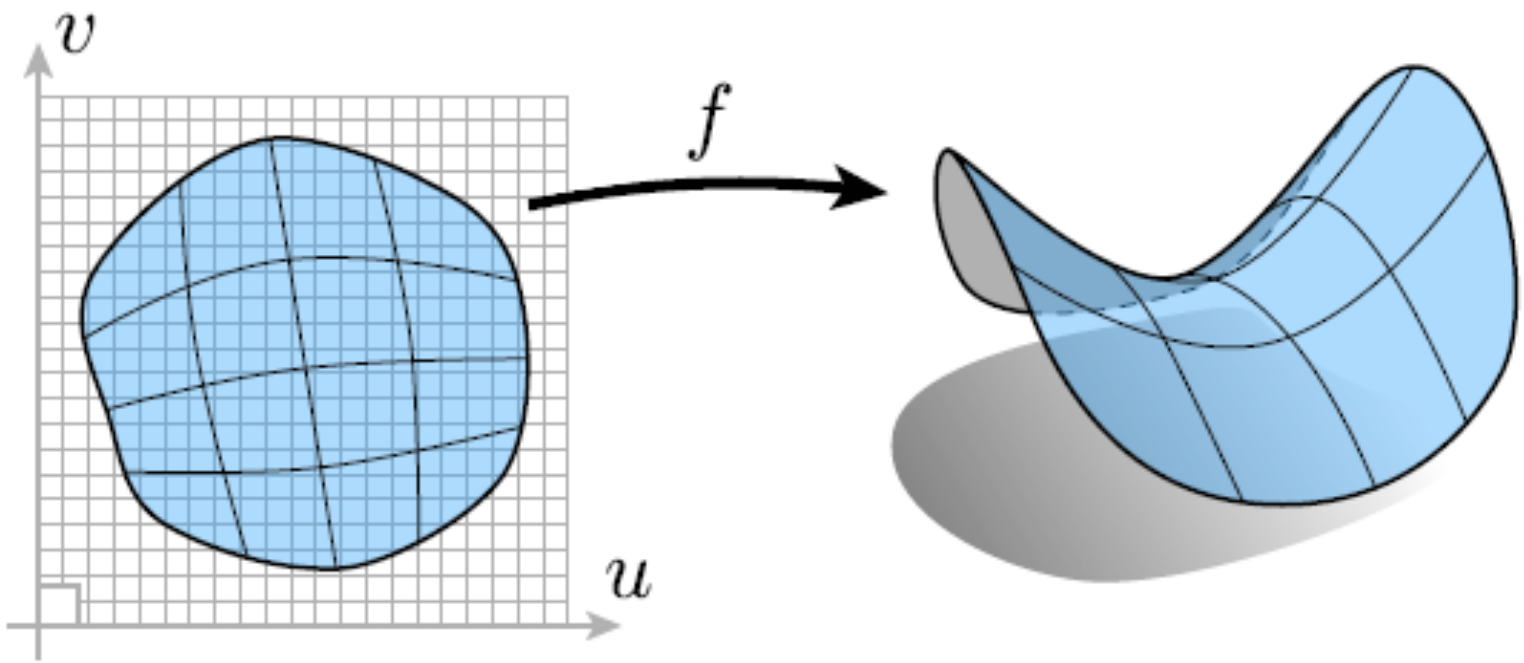
YES.



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

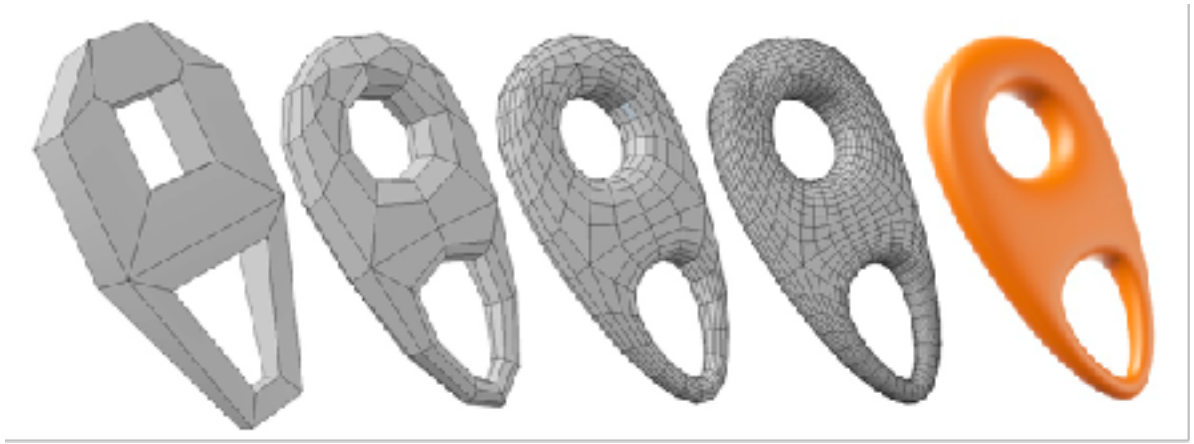
“Explicit” Representations of Geometry

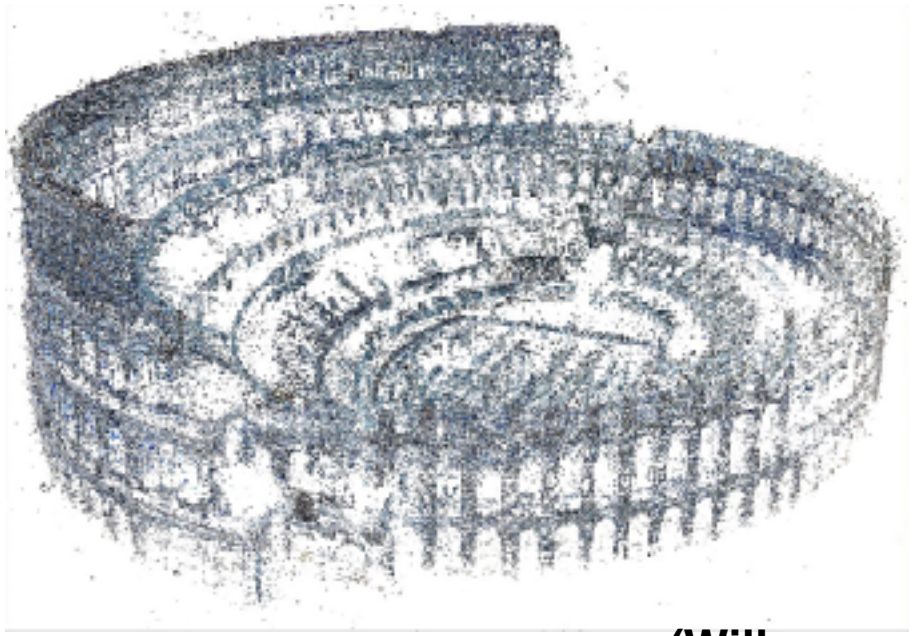
- **All points are given directly**
- **E.g., points on sphere are** $(\cos(u)\sin(v), \sin(u)\sin(v), \cos(v))$, for $0 \leq u < 2\pi$ and $0 \leq v \leq \pi$
- **More generally:** $f : \mathbb{R}^2 \to \mathbb{R}^3; (u, v) \mapsto (x, y, z)$

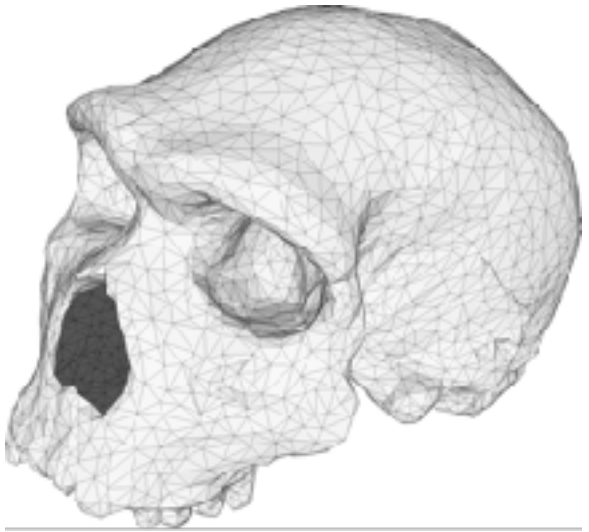


Many explicit representations in graphics

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







(Will see some of these a bit later.)

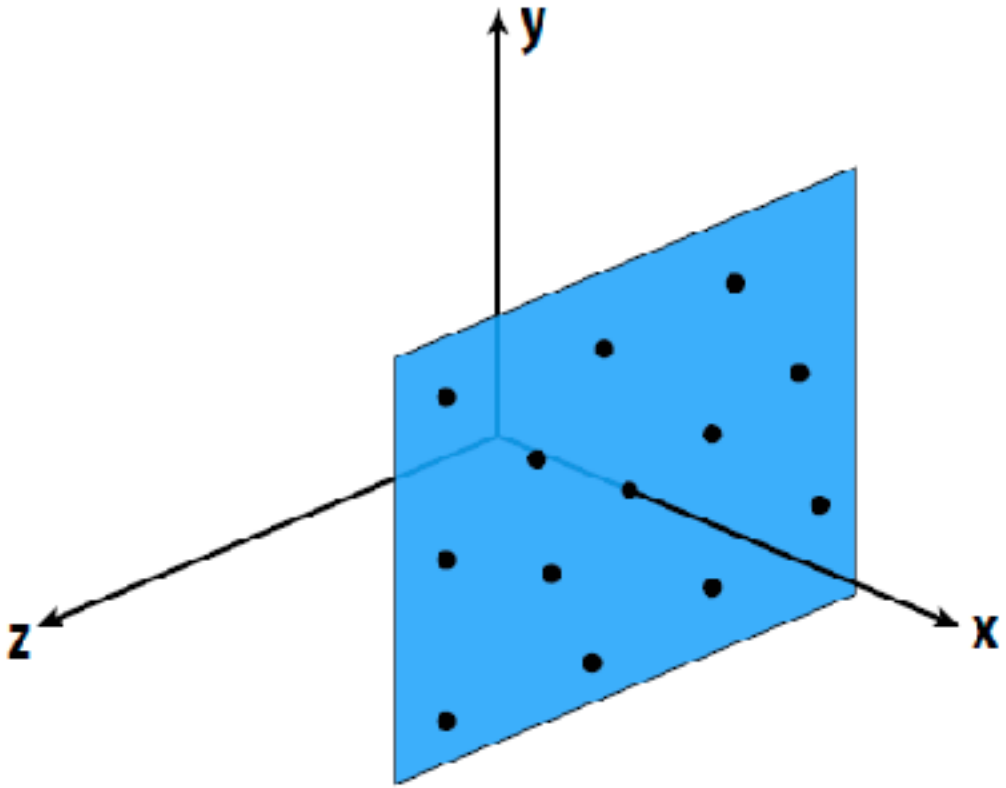
But first, let's play a game:

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).

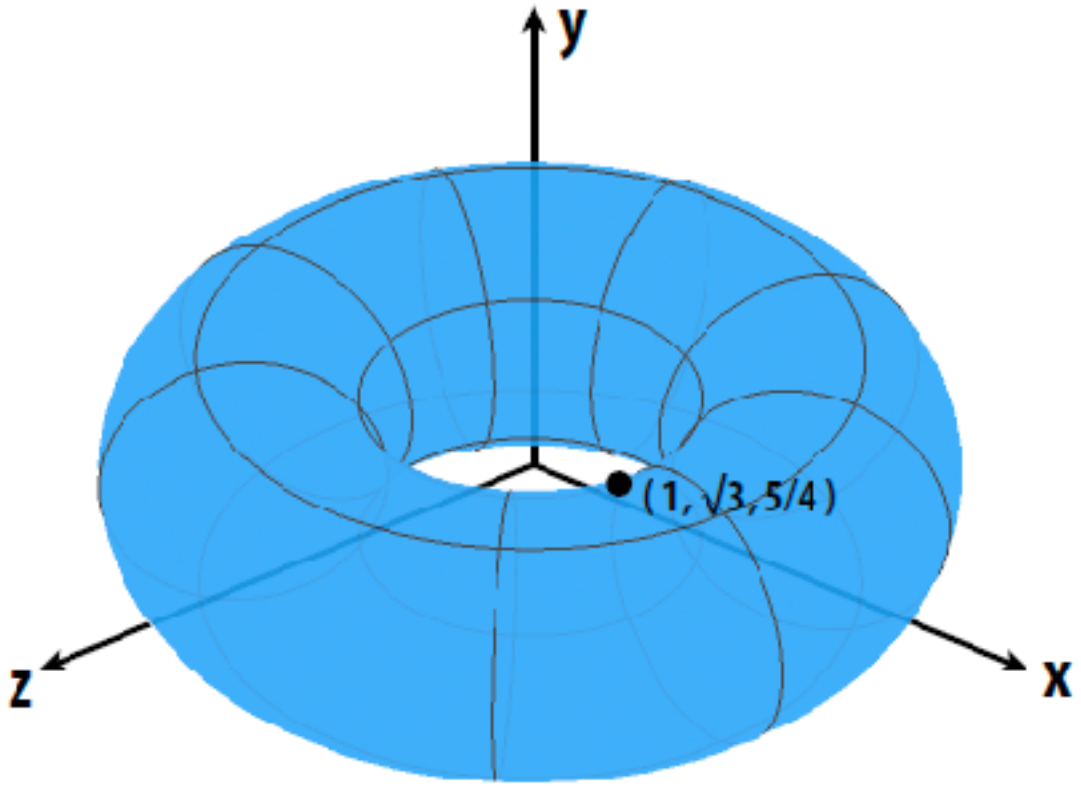
I want to see if a point is inside it.

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,√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:

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

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

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

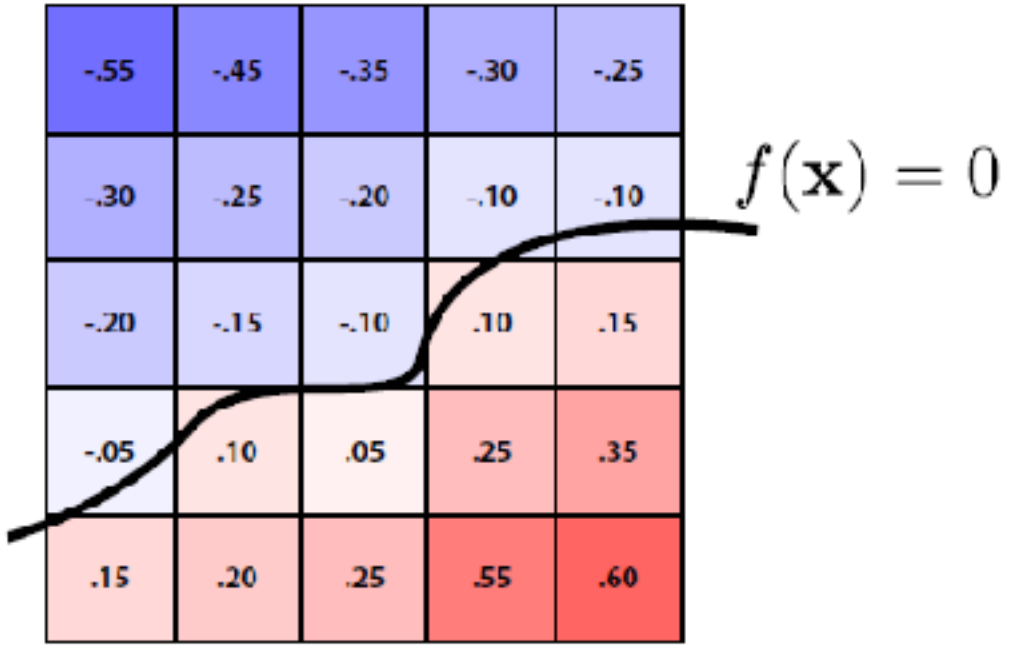
$x^2z^3+\frac{9y^2z^3}{80}$

- What about more complicated shapes?

- Very hard to come up with polynomials!

Level Set Methods (Implicit)

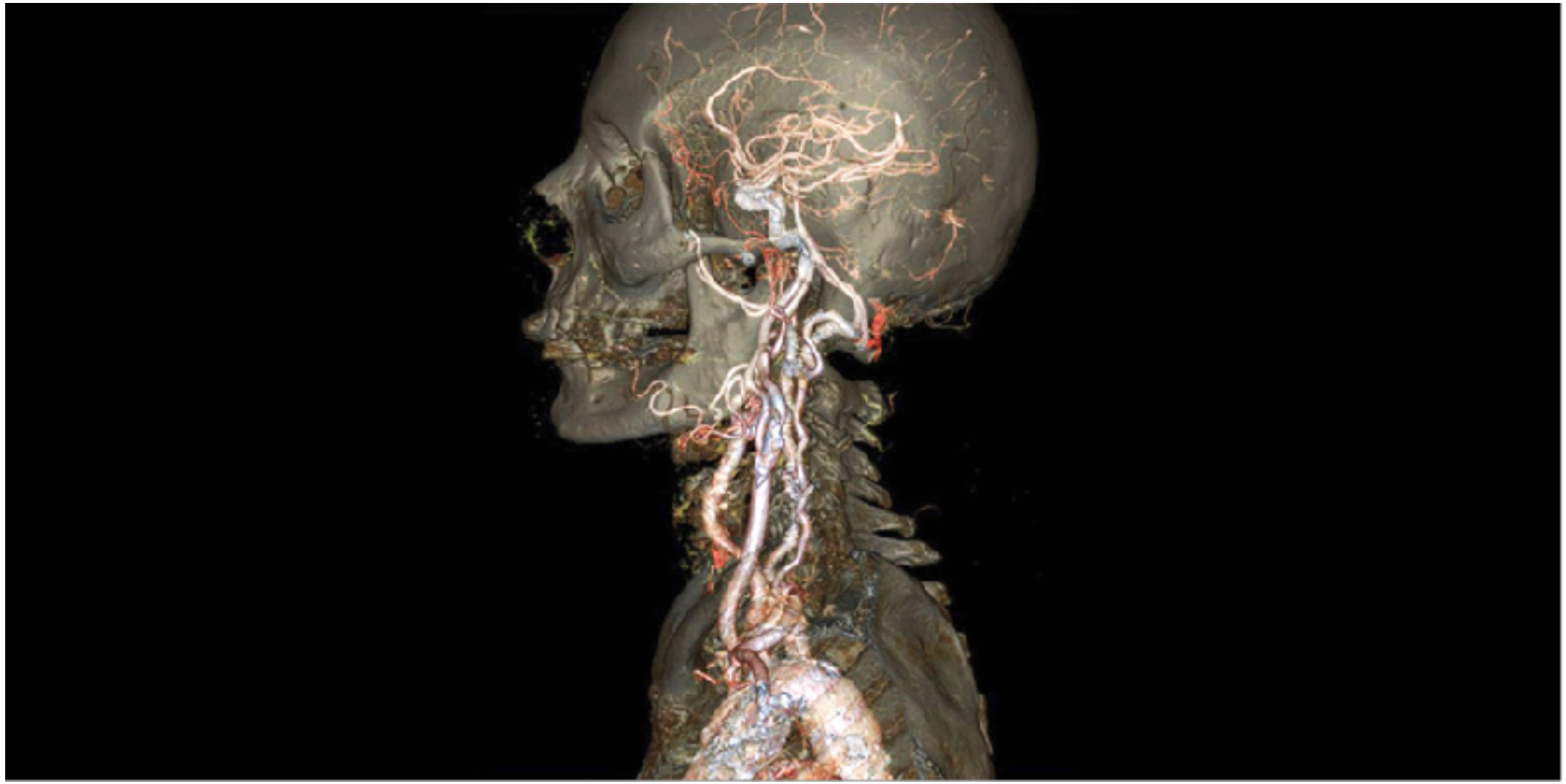
- Implicit surfaces have some nice features (e.g., merging/splitting)
- But, hard to describe complex shapes in closed form
- Alternative: store a grid of values approximating function



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

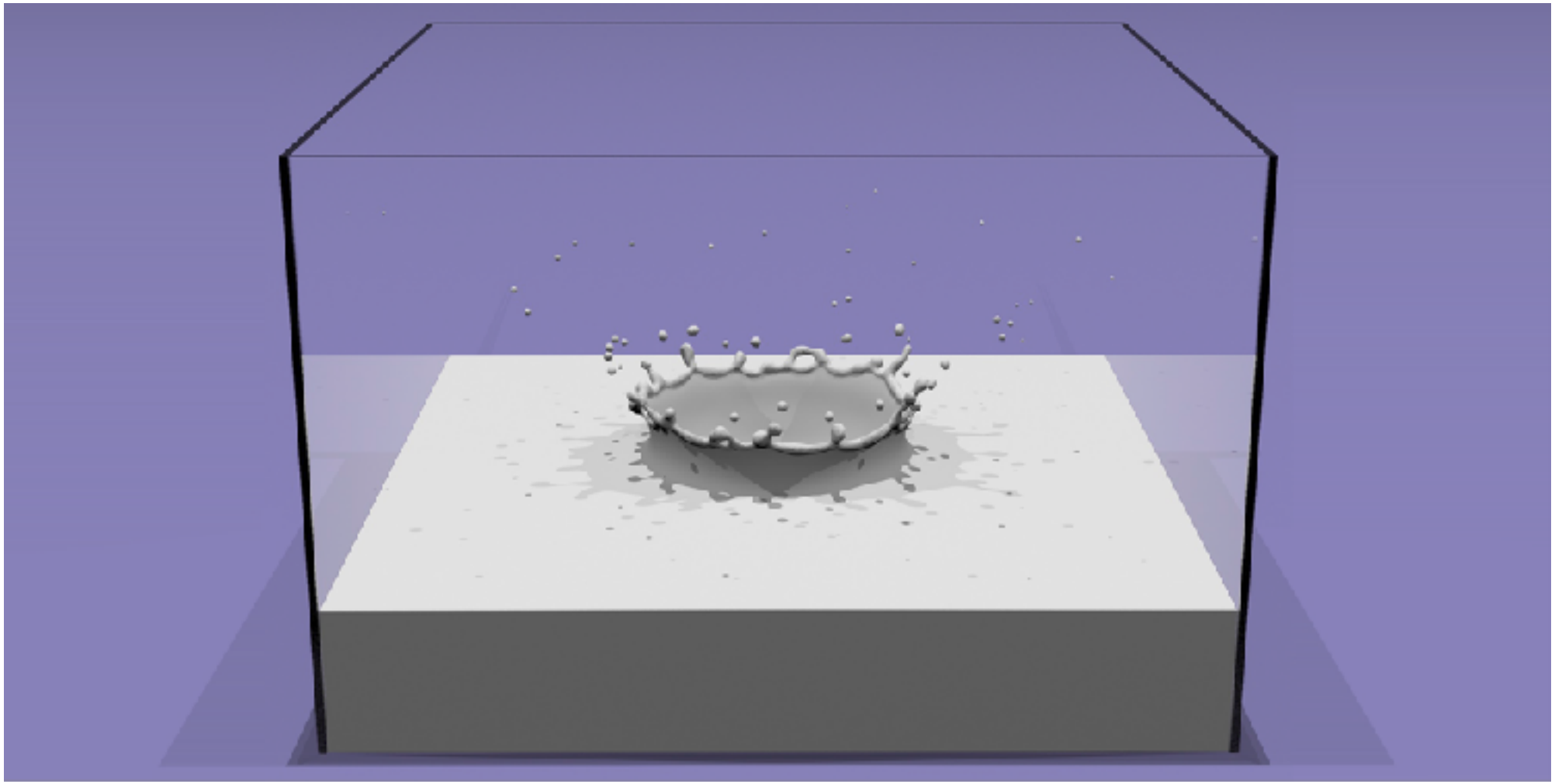
Level Sets from Medical Data (CT, MRI, etc.)

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



Level Sets in Physical Simulation

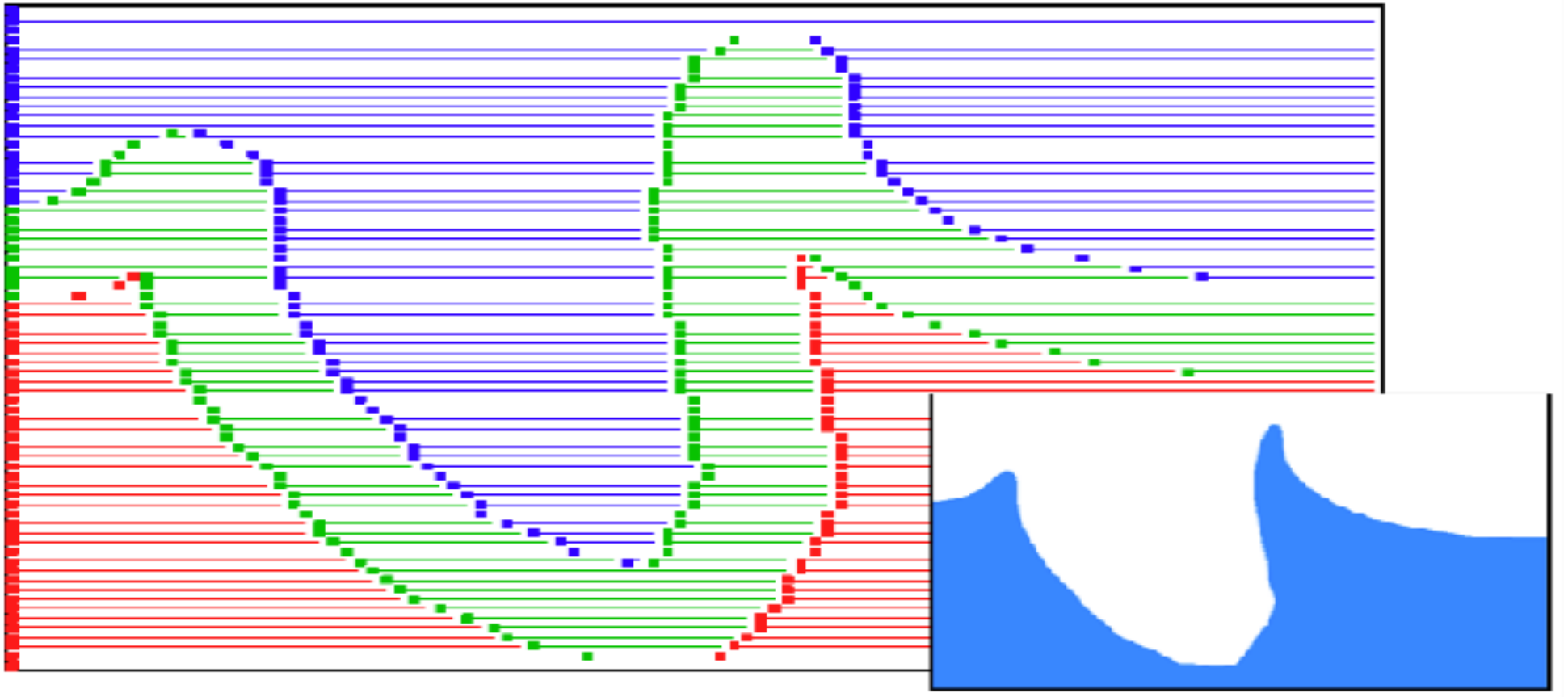
- **Level set encodes distance to air-liquid boundary**



See http://physbam.stanford.edu

Level Set Storage

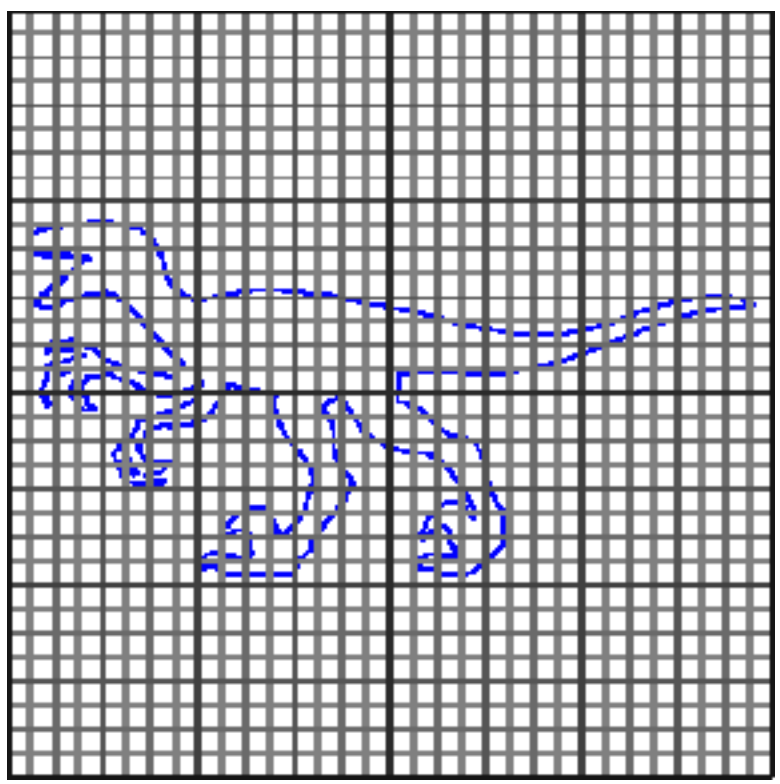
- Drawback: storage for 2D surface is now O(n3)
- Can reduce cost by storing only a narrow band around surface:

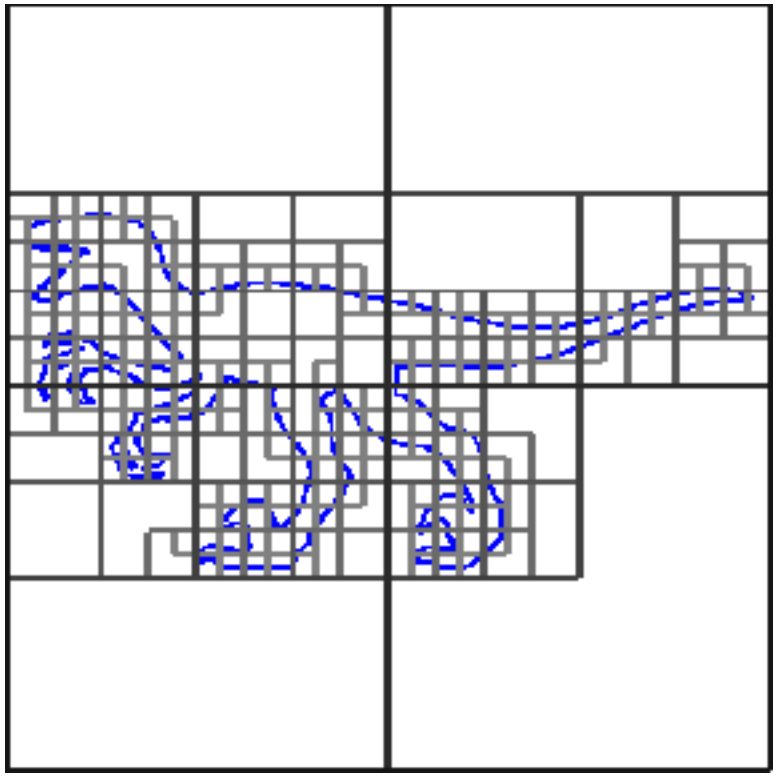


Implicit Representations

Adaptive Grids: Quadtree

In practice, we may only need a high-precision representation of the implicit function near the surface, so represent the function over an adaptive grid.

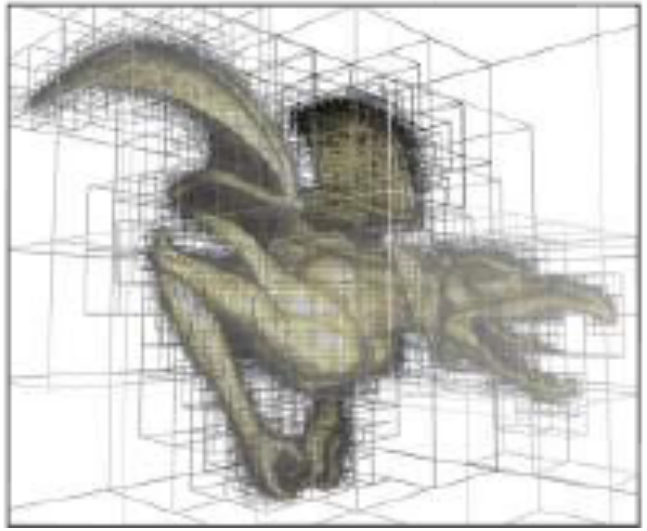


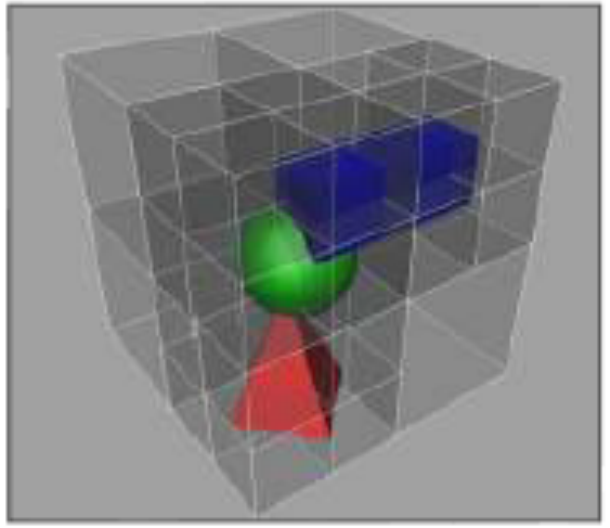


Octree

- A hierarchical tree build by sequential subdivision (8) of a occupied cells.
- Adaptive, i.e. only splits when too many points in cell
- Proximity search by (recursive) tree traversal and distance to neighboring cells
- Tree might not be balanced
- Widely used for complicated scenes that need faster processing and lower accuracy

E.g. Collision detection in real-time simulation or animation





Implicit Representations - Pros & Cons

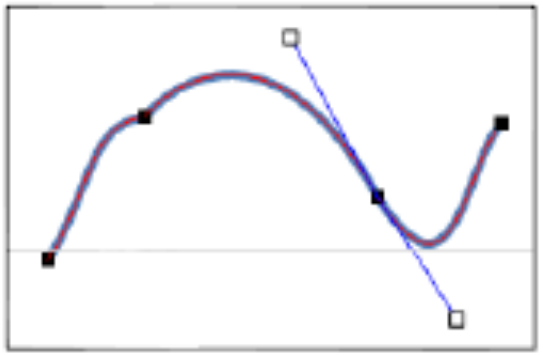
- **Pros:**
  - **description can be very compact (e.g., a polynomial)**
  - **easy to determine if a point is in our shape (just plug it in!)**
  - **other queries may also be easy (e.g., distance to surface)**
  - **for simple shapes, exact description/no sampling error**
  - **easy to handle changes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

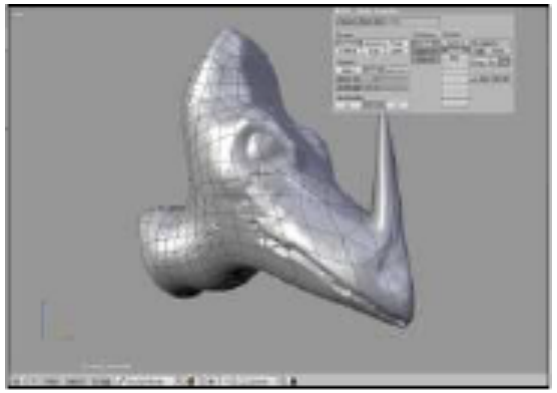
Shape representation

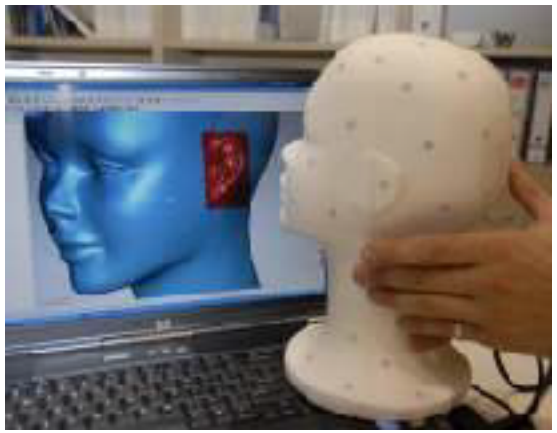
Where does the shape come from?

- **Modeling “by hand”**
  - Higher-level representations,
  - Amenable to modification, control

- **Acquired real-world objects**
  - Discrete sampling
  - Points, meshes







Shape Acquisition

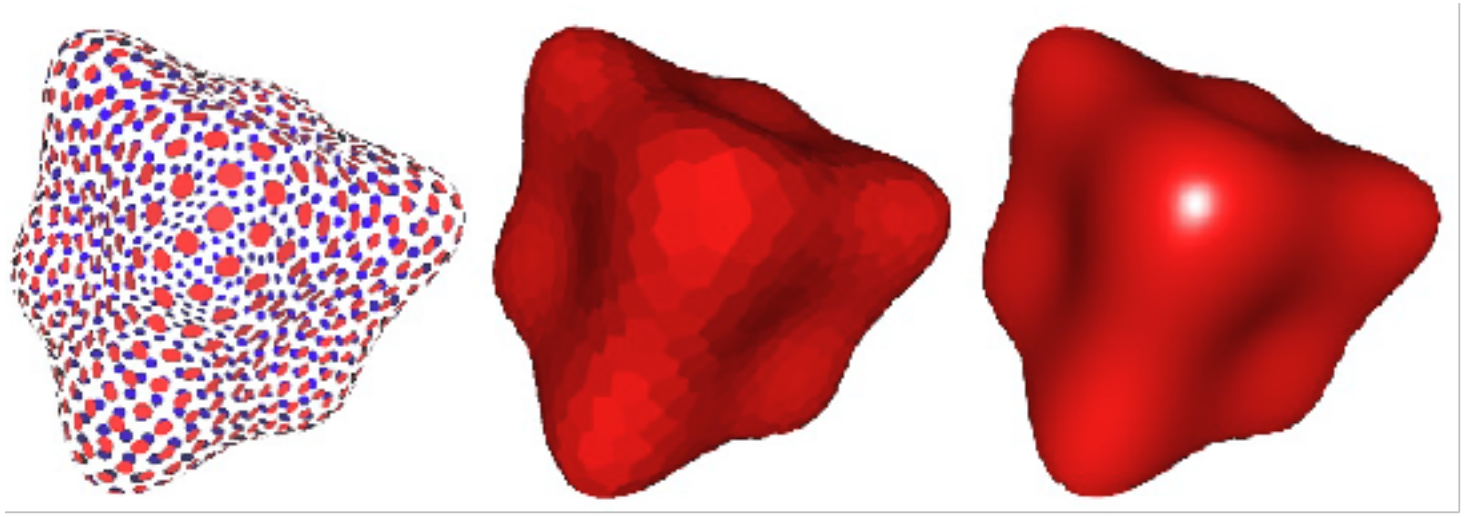
Sampling of real world objects

- Scanners
  - Laser
  - Depth imaging
- Properties & Operations
  - Potentially noisy, with outliers
  - Registration of multiple images
  - Non-uniform sampling, sparse, holes

180 frames per second (From David Gu)

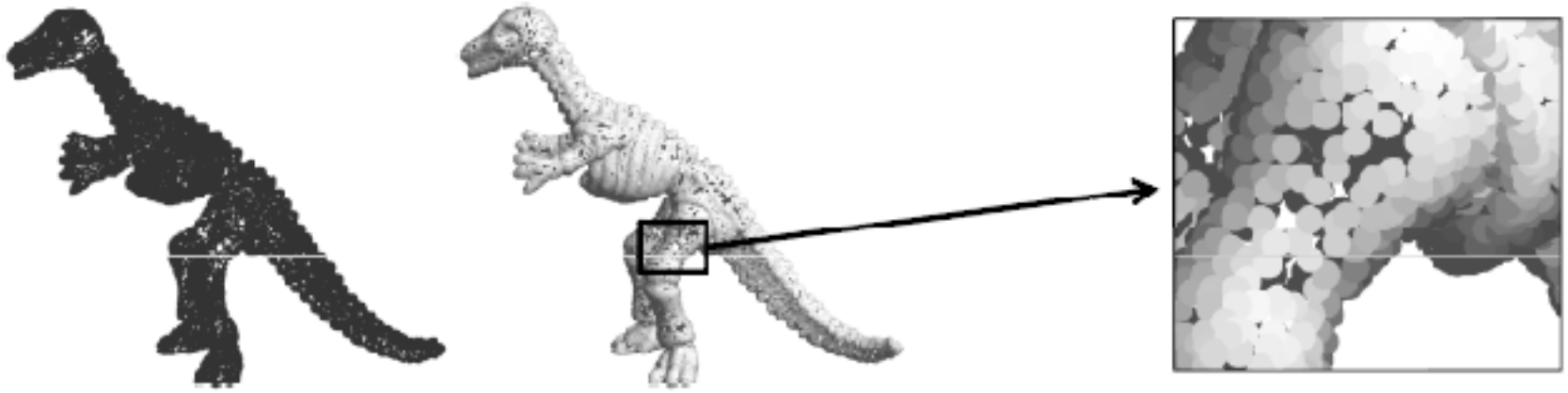
What about explicit representations?

- 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



Points: Neighborhood information

- Efficient point processing & modeling requires a spatial partitioning data structure
  - To figure out **neighborhoods**
- Why do we need neighbors?



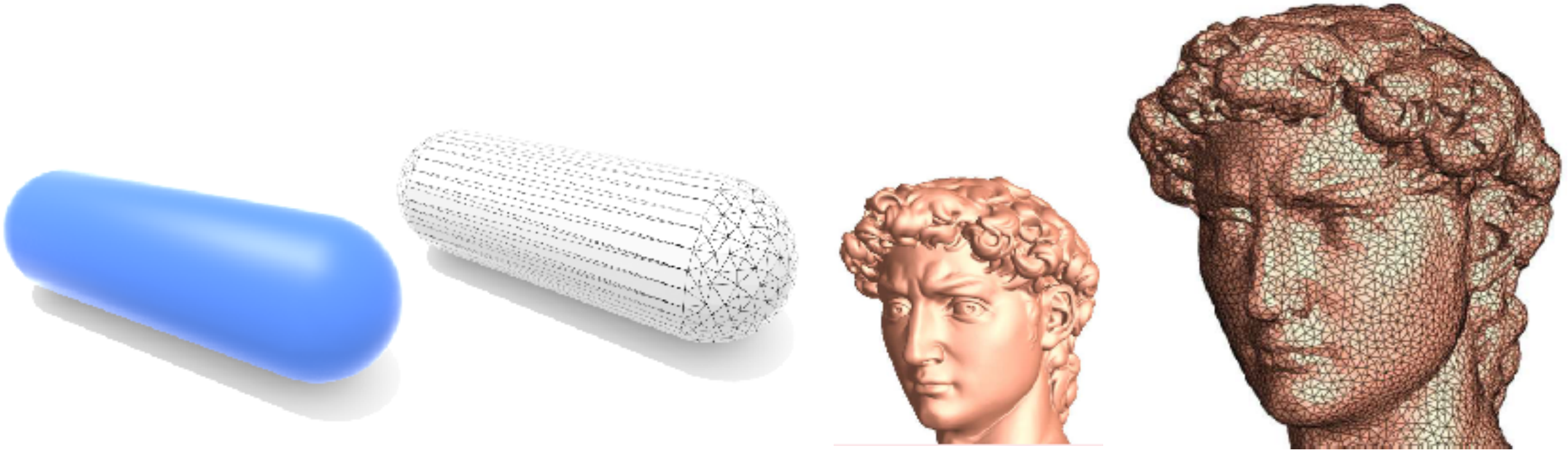
need normals (for shading)

upsampling – need to count density

- Need sub-linear implementations of
  - K-nearest neighbors to point x (knn)
  - In radius search

Polygon Mesh (Explicit)

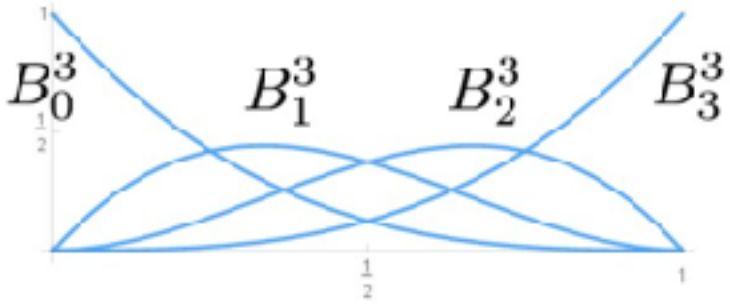
- Store vertices *and* polygons (most often triangles or quads)
- Easier to do processing/simulation, adaptive sampling
- More complicated data structures
- Perhaps most common representation in graphics



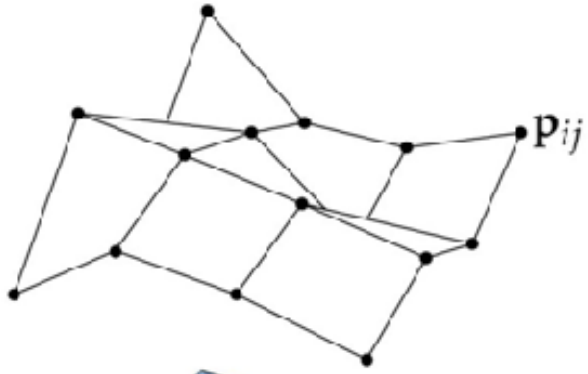
Spline & NURBS

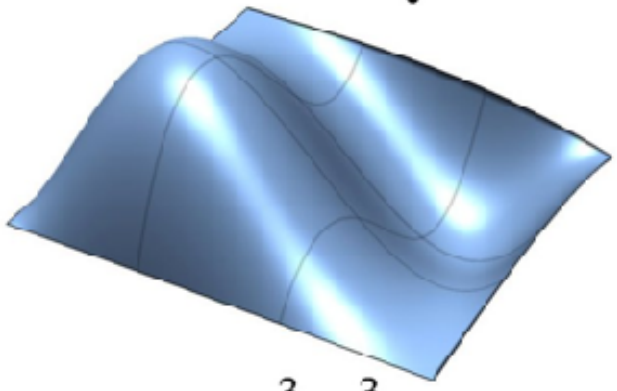
- Extract analytical rep.
- Support interactive shape editing
- Compact rep.

- Major modeling techniques in CAD



$$B_{i,j}^3(u,v) := B_i^3(u)B_j^3(v)$$

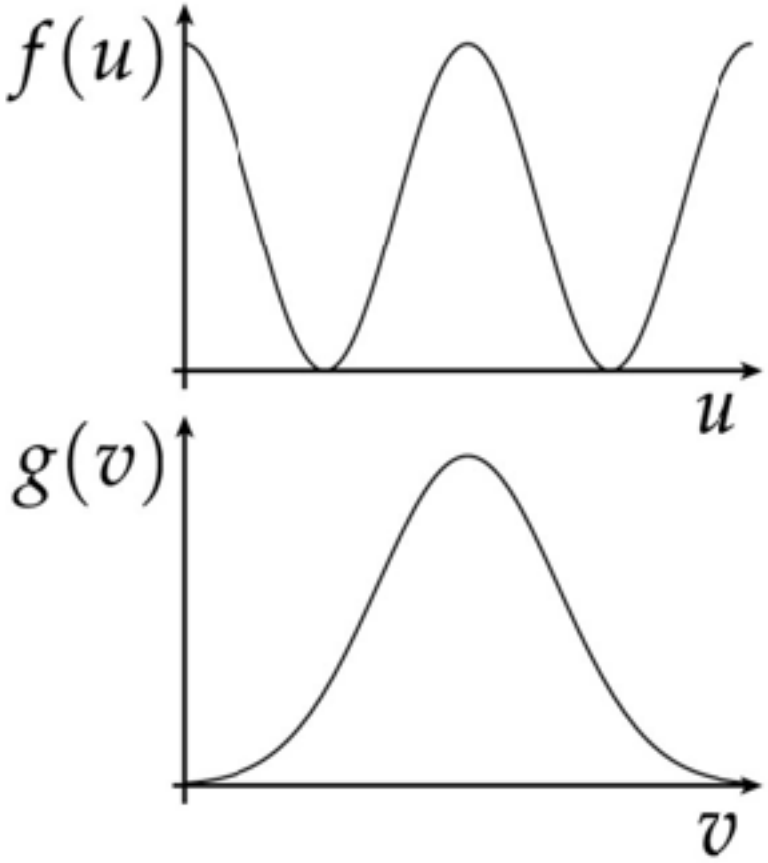


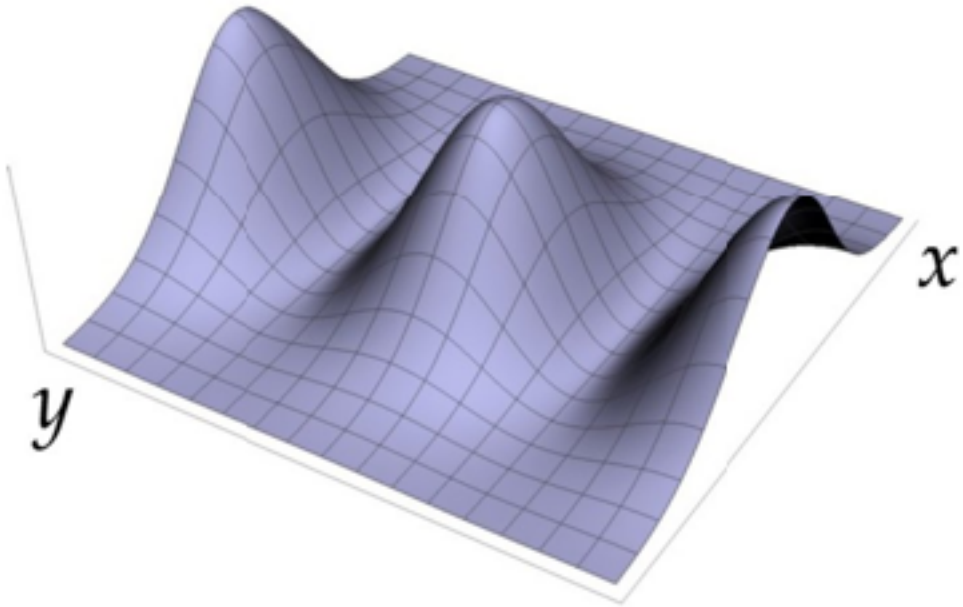


$$S(u,v) := \sum_{i=0}^{3}\sum_{j=0}^{3} B_{i,j}^3(u,v)\mathbf{p}_{ij}$$

Tensor product

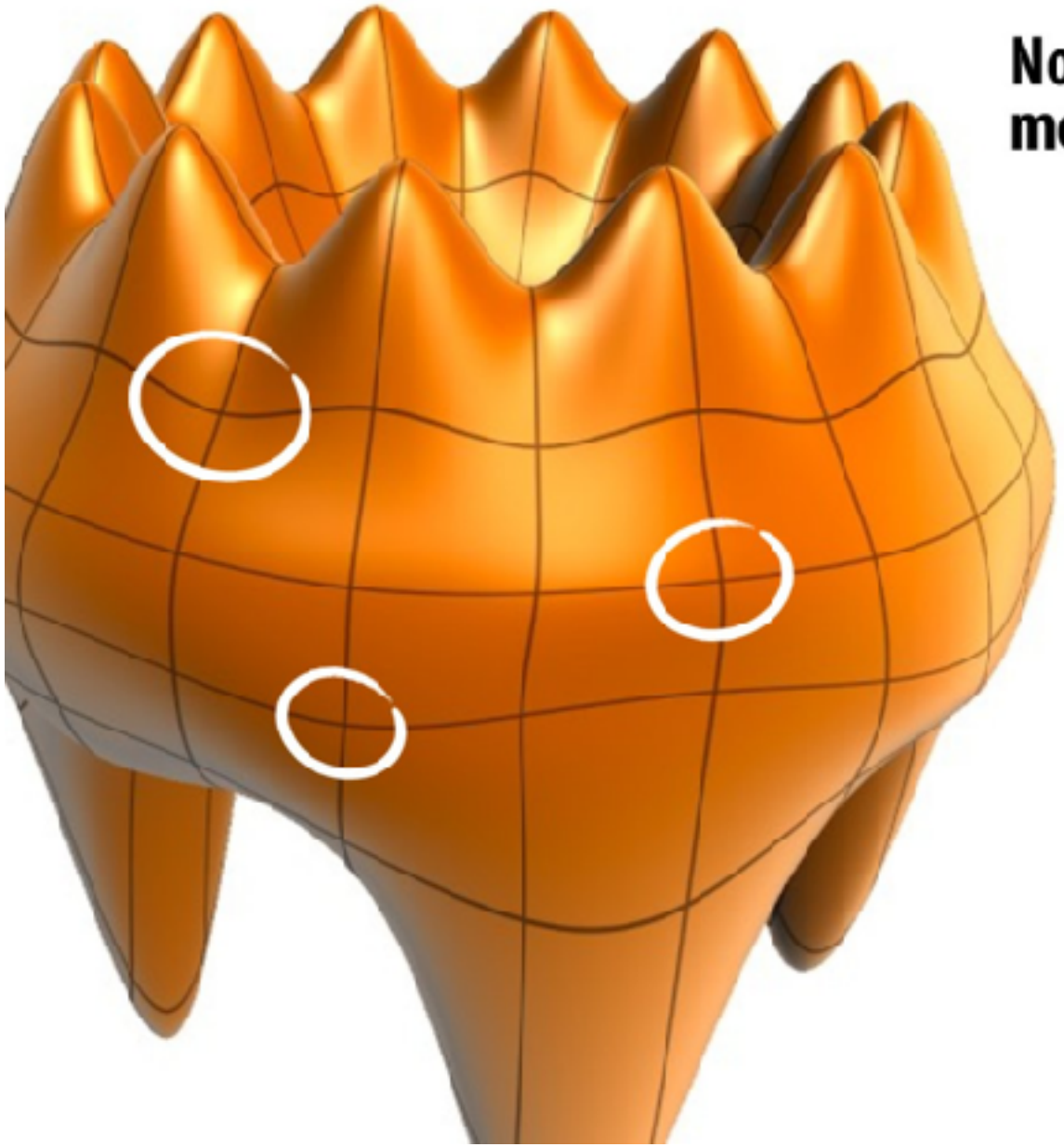
- Use a pair of curves to get a surface
- Value at any point (u,v) given by product of a curve f at u and a curve g at v





$$(f \otimes g)(u,v) := f(u)g(v)$$

Bezier patches are too simple



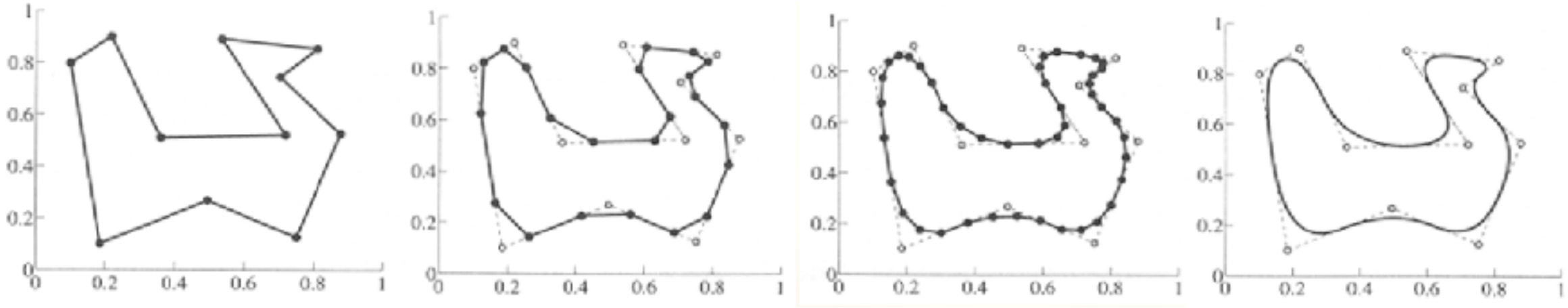
Notice that exactly four patches meet around *every* vertex!

In practice, far too constrained.

To make interesting shapes (with good continuity), we need patches that allow more interesting connectivity...

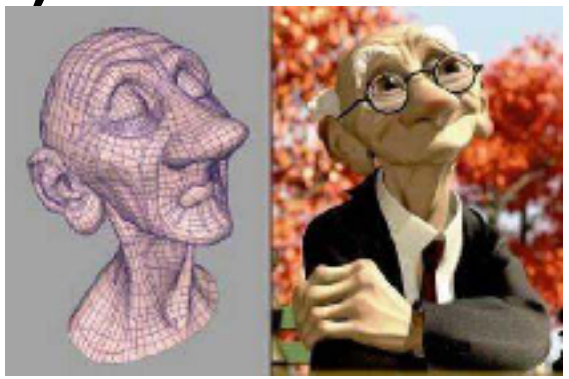
Subdivision (Explicit)

- Alternative starting point for 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
  - Some subdivision schemes correspond to well-known spline schemes!



Subdivision Surfaces (Explicit)

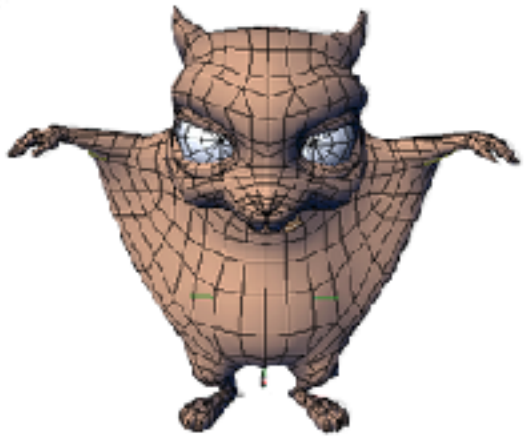
- Start with coarse polygon mesh (“control cage”)
- Subdivide each element
- Update vertices via local averaging
- Many possible rule:
  - Catmull-Clark (quads)
  - Loop (triangles)
  - ...



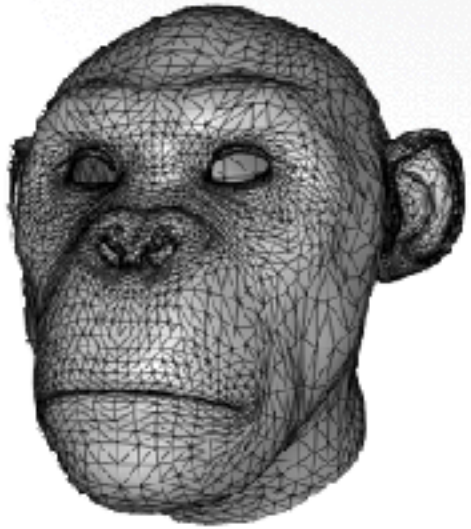
- Common issues:
  - interpolating or approximating?
  - continuity at vertices?
- Easier than splines for modeling; harder to guarantee continuity

Summary

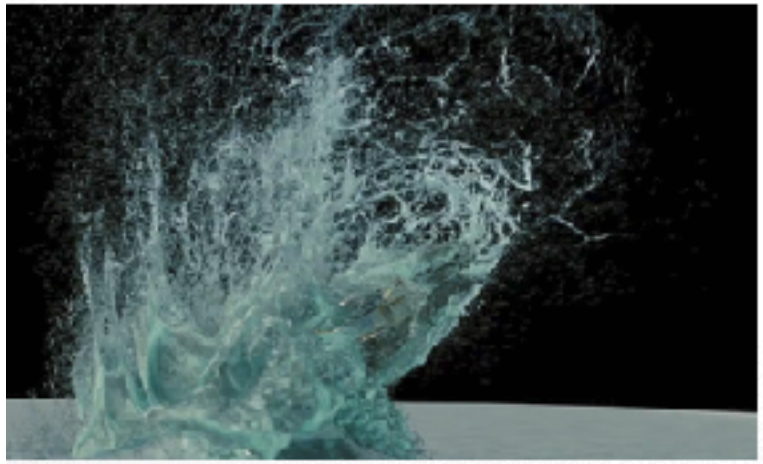
point based



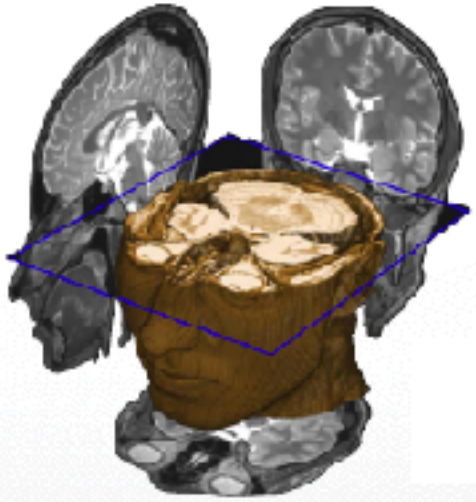
quad mesh



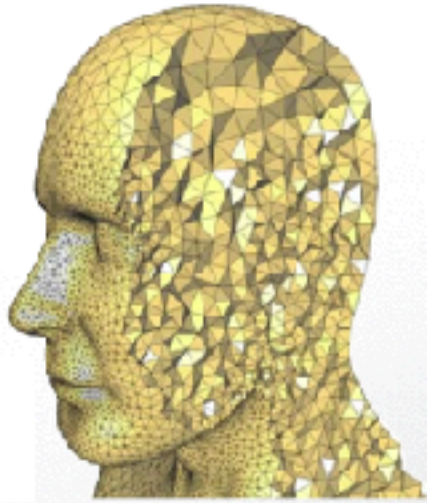
triangle mesh



implicit surfaces / particles



volumetric



tetrahedrons

What do we need from shapes in Computer Graphics?

- Local control of shape for modeling
- Ability to model what we need
- Smoothness and continuity
- Ability to evaluate derivatives
- Ability to do collision detection
- Ease of rendering

No single technique solves all problems!

Resources

- read & display a mesh: jjcao_plot/eg_trisurf.m
- Read & display a huge point set (100k to 1 million points)
  - PC_processing_1.0
  - jjcao_code/tools/pcd_viewer