Computer Graphics

-- Implicit Representation

Junjie Cao @ DLUT

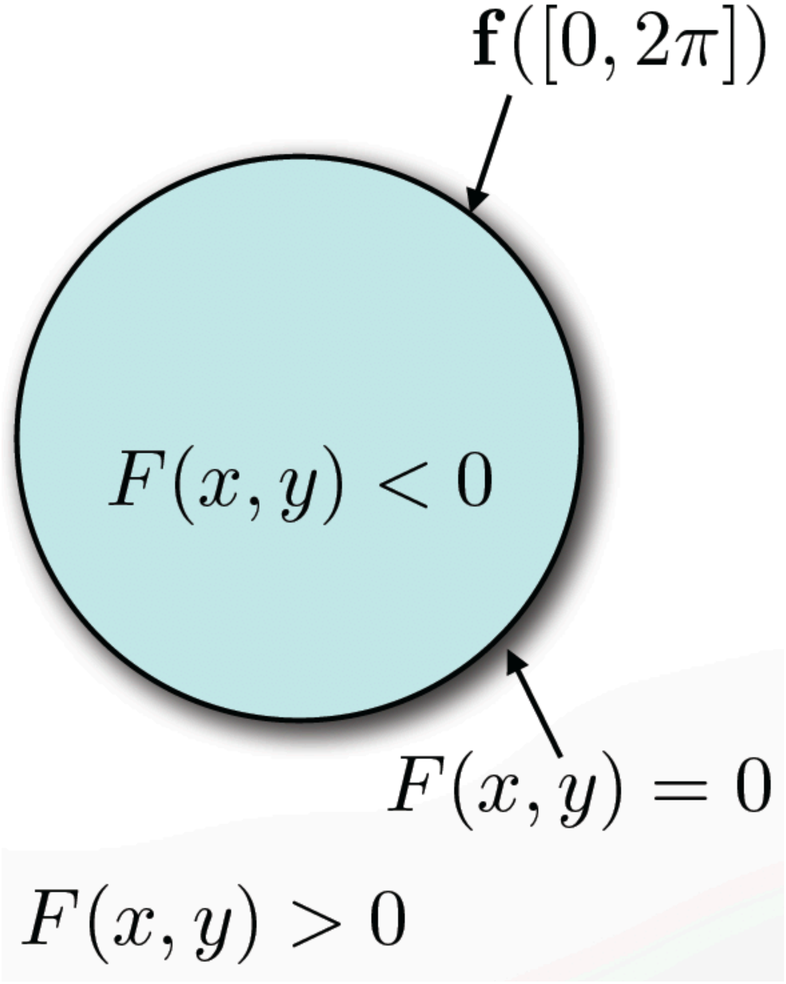
Spring 2019

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

"Implicit" Representations of Geometry

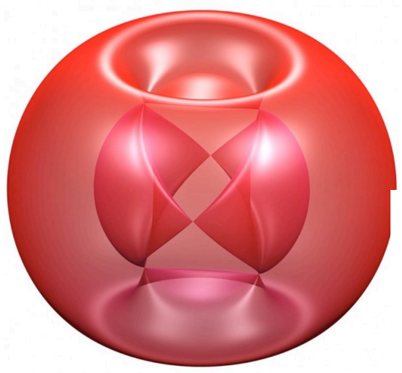
- Points aren't known directly, but satisfy some relationship
  - E.g., unit sphere is all points x such that **x^2+y^2+z^2=1**
  - More generally, **f(x,y,z) = 0**
- Represent a surface as the zero set of a (regular) function defined in R^3.

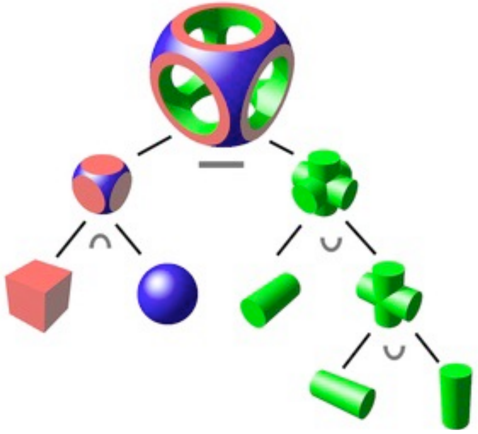
$$K = g^{-1}(0) = \{\mathbf{p} \in \mathrm{R}^3 : g(\mathbf{p}) = 0\}$$

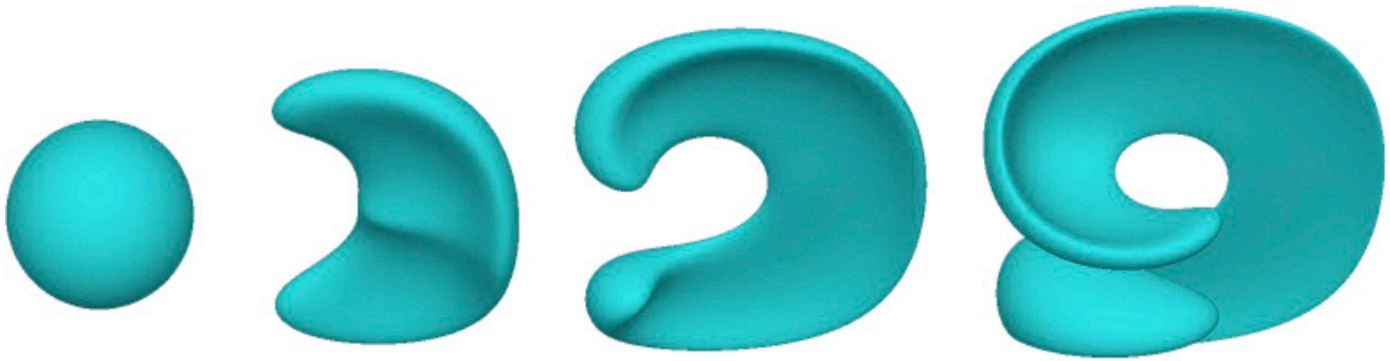


Many implicit representations in graphics

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



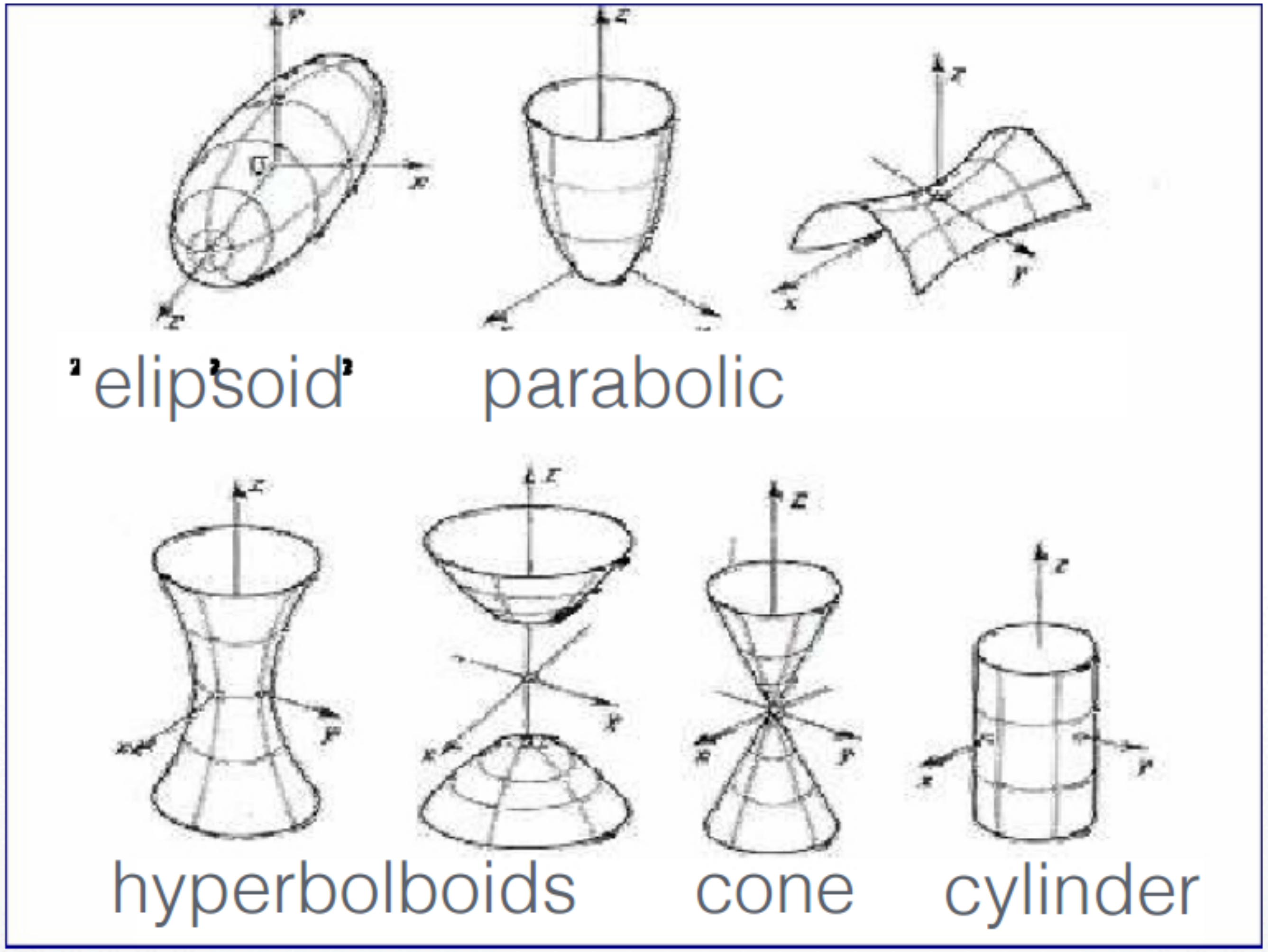




(Will see some of these a bit later.)

Quadric Surfaces

$$F(x,y,z) = ax^2 + by^2 + cz^2 + 2fyz + 2hxy + 2px + 2qy + 2rz + d = 0$$



Algebraic Surfaces (Implicit)

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

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

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

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

Gradient of Algebraic Surfaces

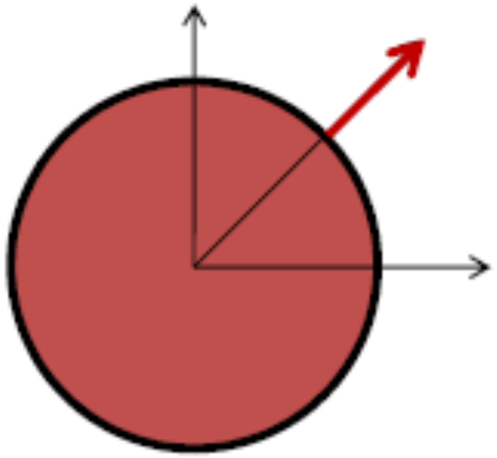
- Gradient is orthogonal to level set

$$\nabla g(x,y,z) = \left(\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}, \frac{\partial g}{\partial z} \right)^{\mathrm{T}}$$

- Example

$$g(x,y,z) = x^2 + y^2 + z^2 - r^2$$

$$\nabla g(x,y,z) = (2x, 2y, 2z)^{\mathrm{T}}$$

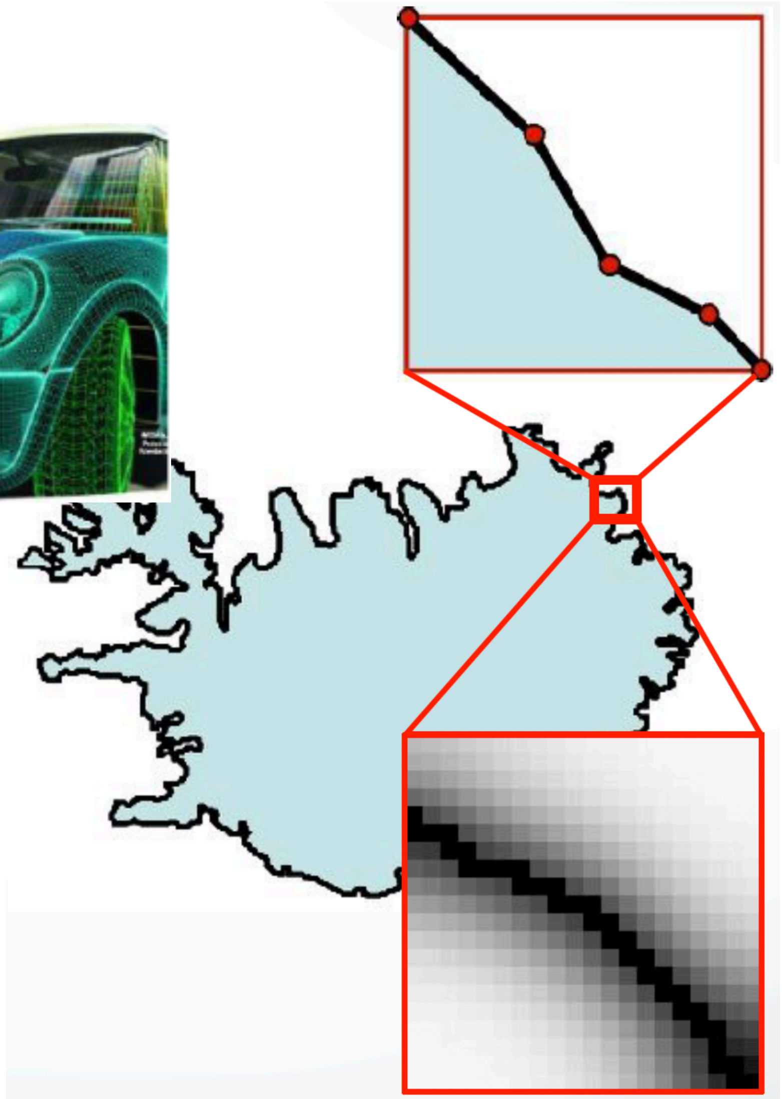


$$\nabla g(x,y,z) = (2,2,0)^{\mathrm{T}}$$

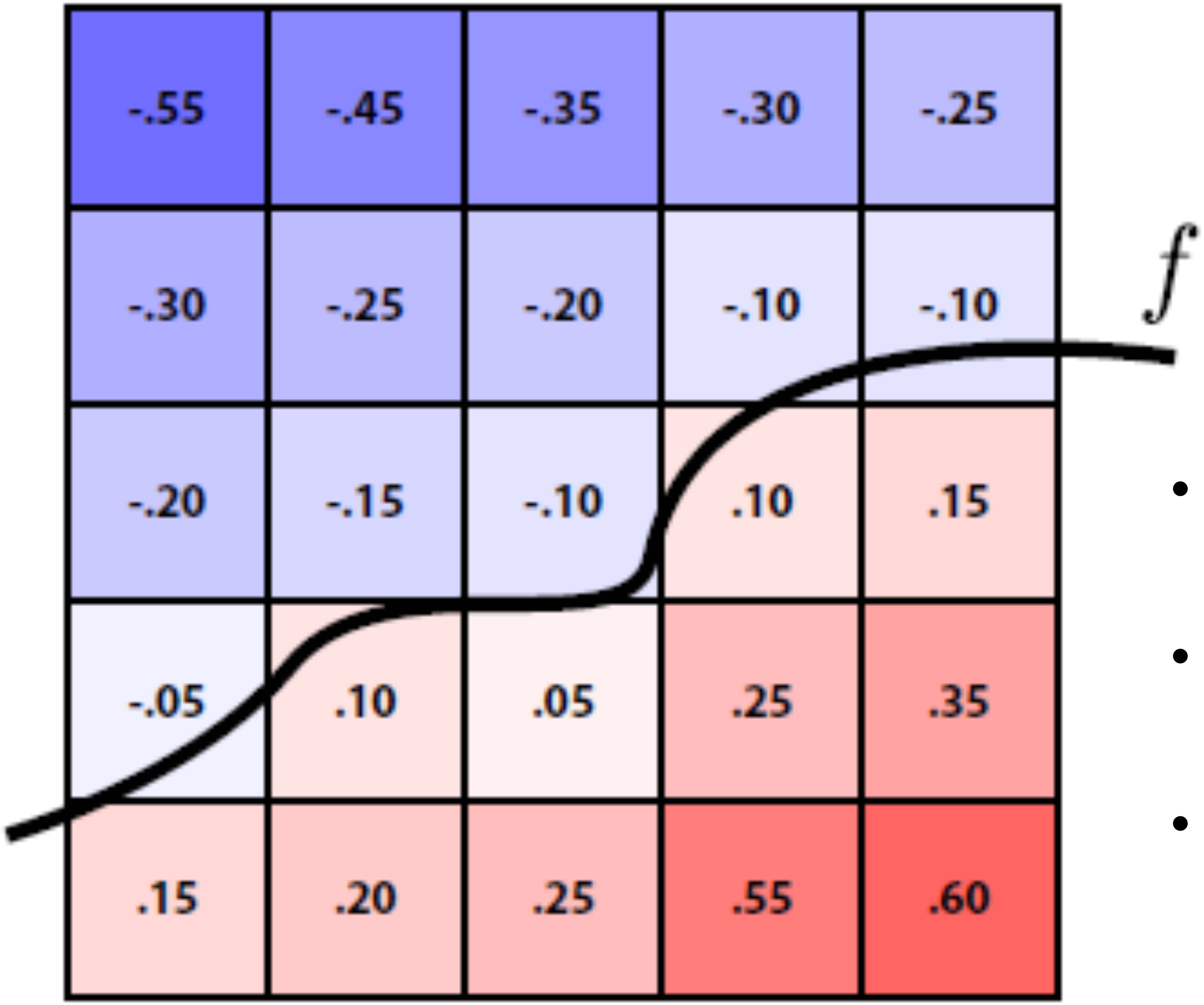
- => normal (1,1,0)

Algebraic Surfaces (Implicit)

- What about more complicated shapes?
- Very hard to come up with polynomials!



Level Set Methods (Implicit)

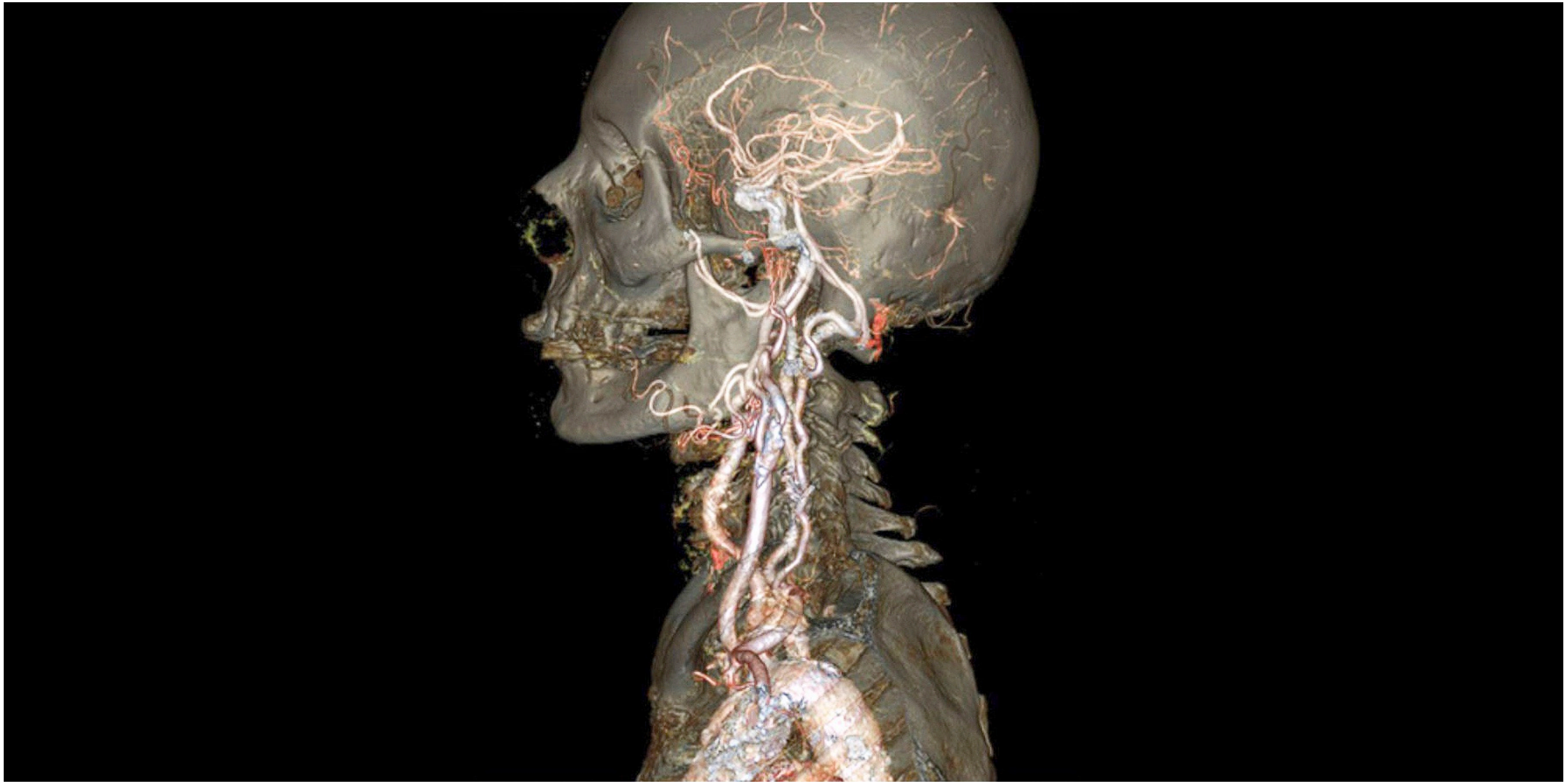


$$f(\mathbf{x}) = 0$$

- **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)**

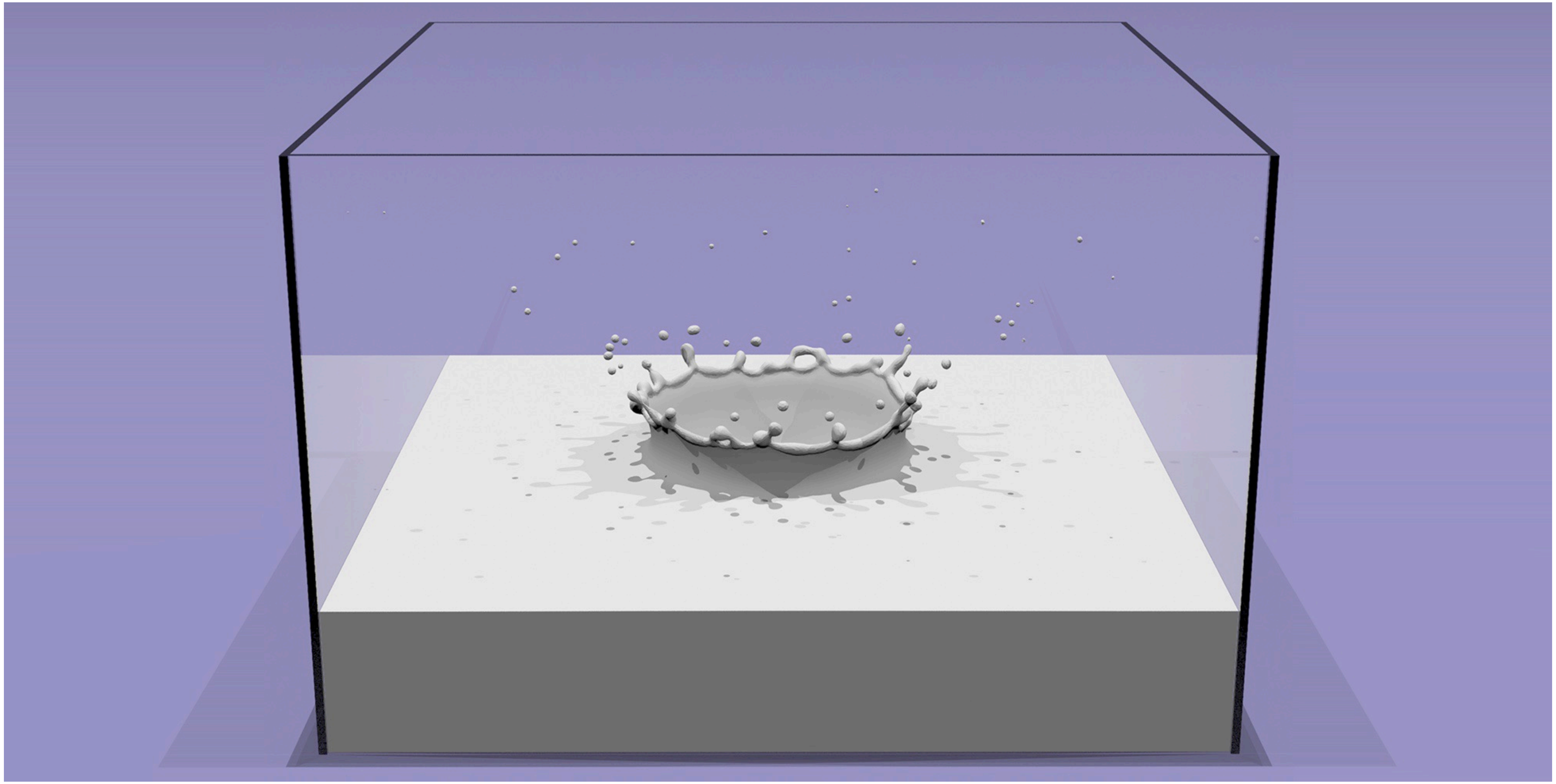
Level Sets from Medical Data (CT, MRI, etc.)

- **Level sets encode, e.g., constant tissue density**
- Natural representation for volumetric data: CT scans, density fields, etc.



Level Sets in Physical Simulation

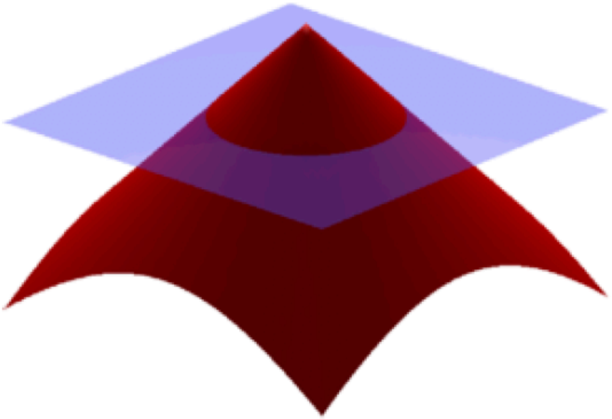
- **Level set encodes distance to air-liquid boundary**
- Advantageous when modeling shapes with complex and/or changing topology (e.g., fluids)



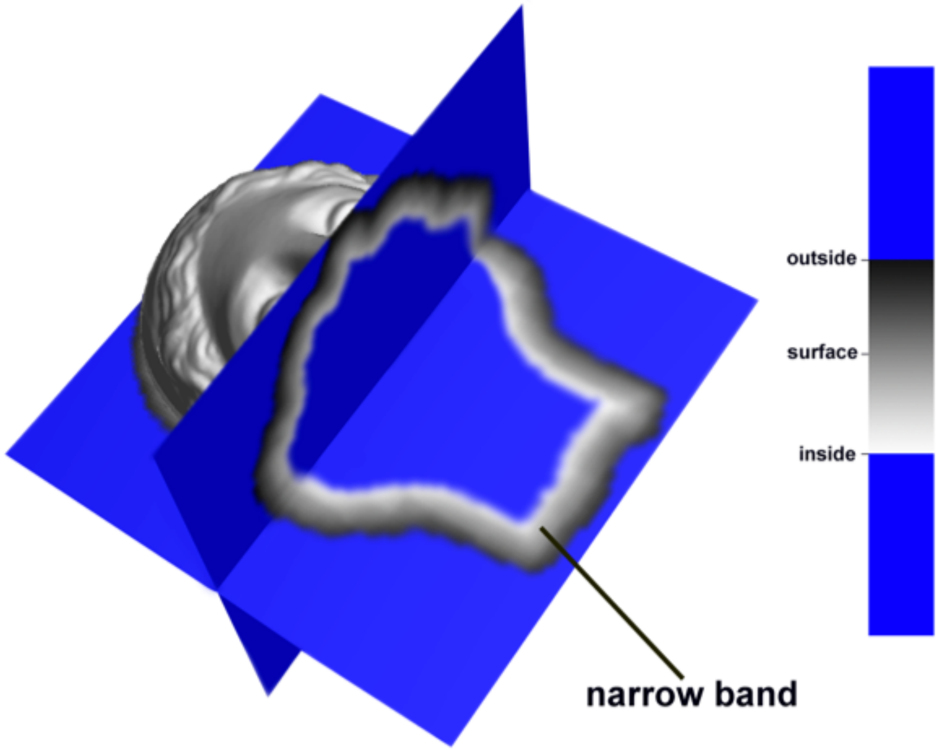
See http://physbam.stanford.edu

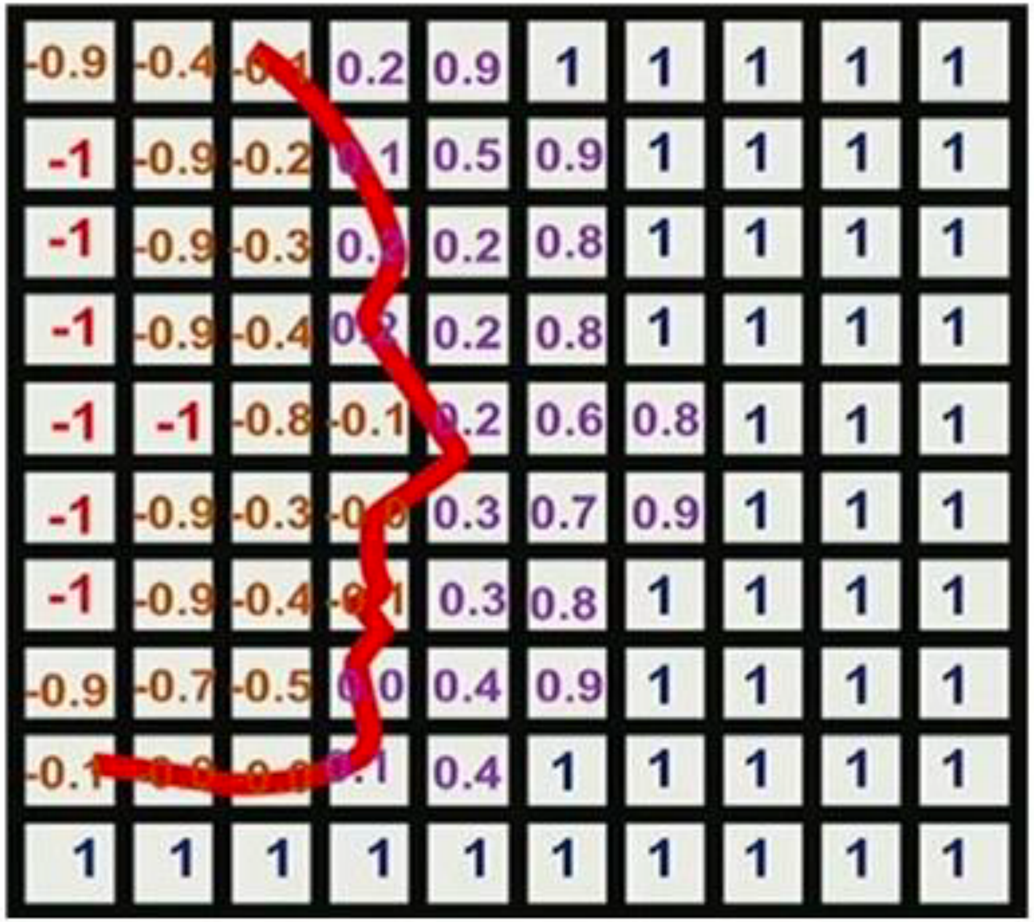
Signed Distance Function

- SDF of a circle?



- General shapes

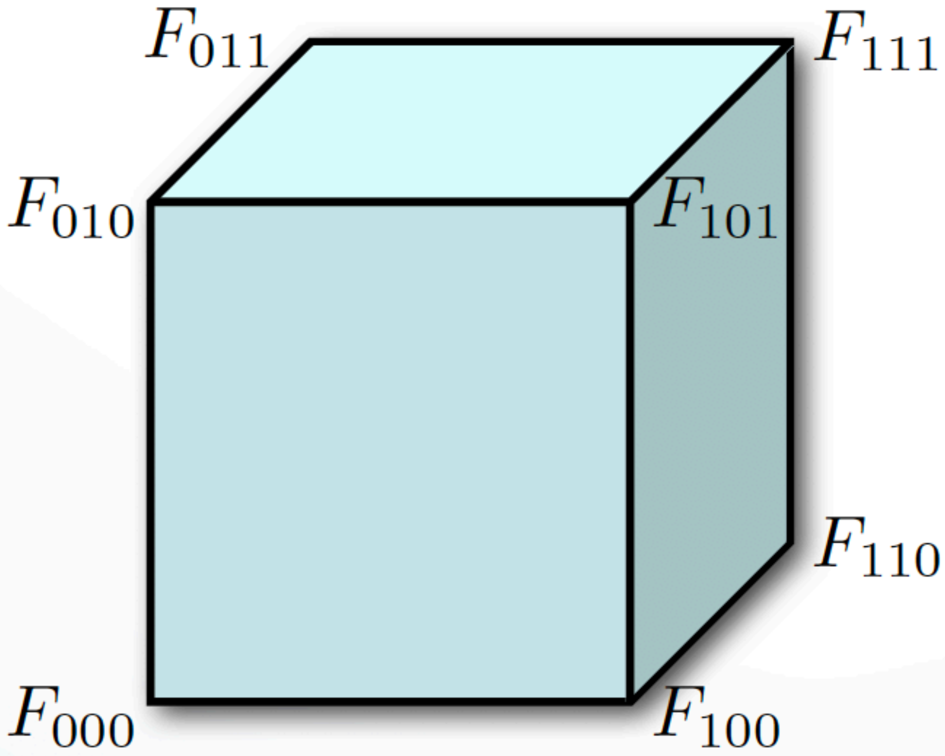




Truncated signed distance field (TSDF):
Less memory than SDF

SDF Discretization

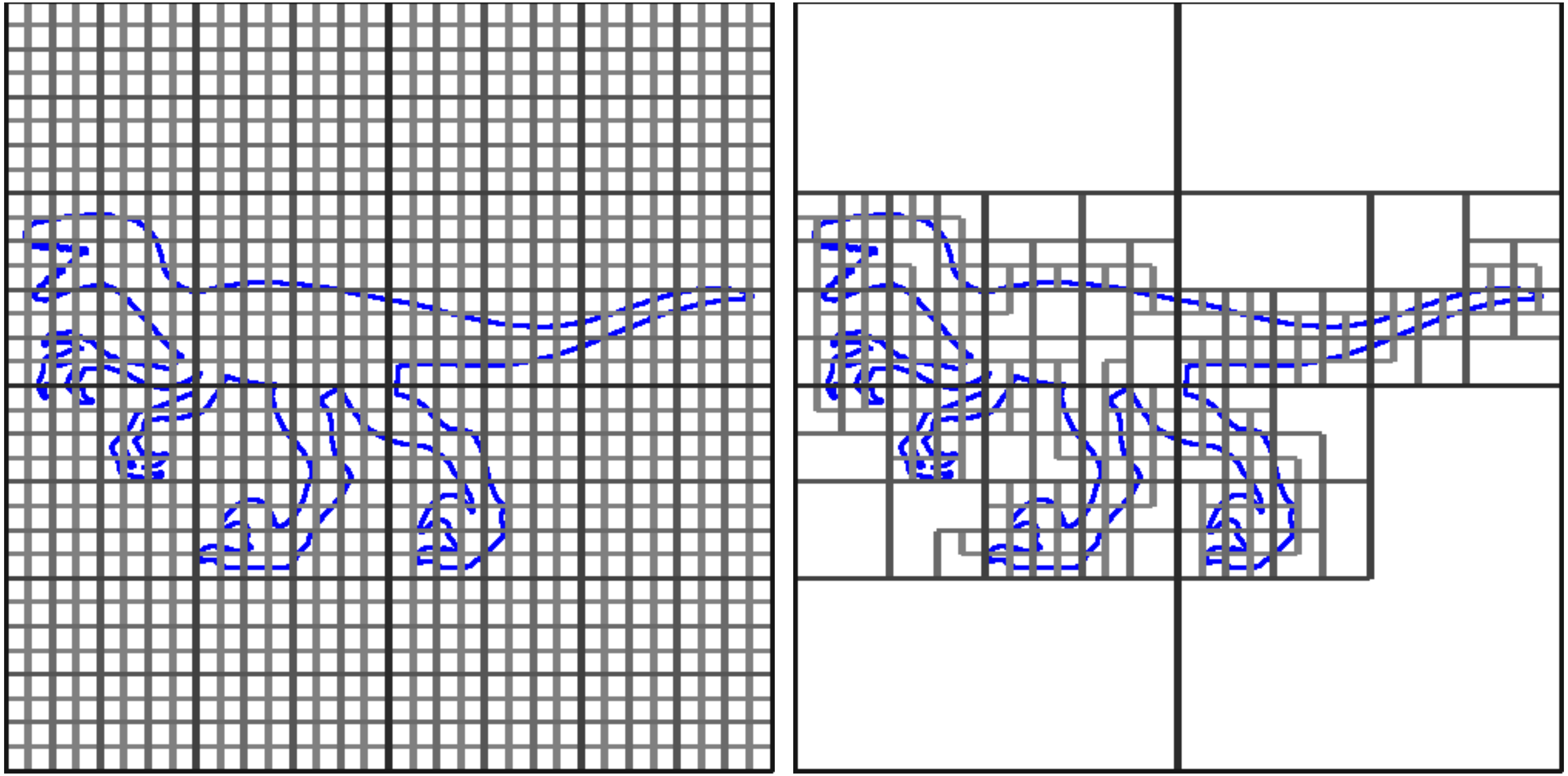
- Regular cartesian 3D grid
  - Compute signed distance at nodes
  - Tri-linear interpolation within cells



$$
\begin{array}{lrrrl}
F_{000} & (1-u) & (1-v) & (1-w) & + \\
F_{100} & u & (1-v) & (1-w) & + \\
F_{010} & (1-u) & v & (1-w) & + \\
F_{001} & (1-u) & (1-v) & w & + \\
\vdots & & & & \\
F_{111} & u & v & w &
\end{array}
$$

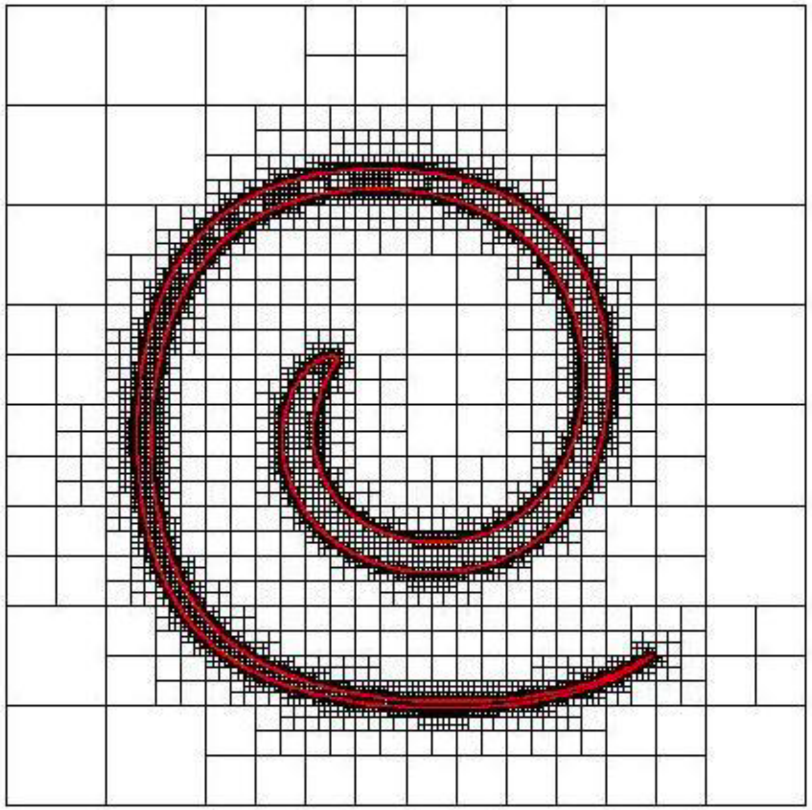
Implicit Representations

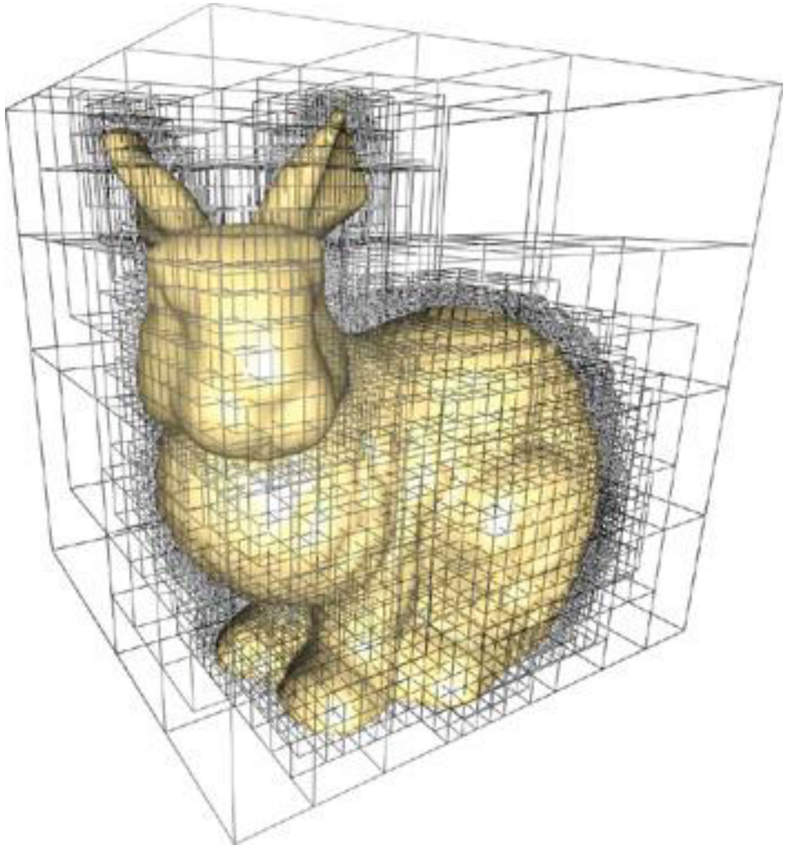
- **Level Set Storage: storage for 2D surface is now O(n3)**
- **Reduce cost by storing only a narrow band around surface --** Adaptive Grids
  - Quadtree



Octree

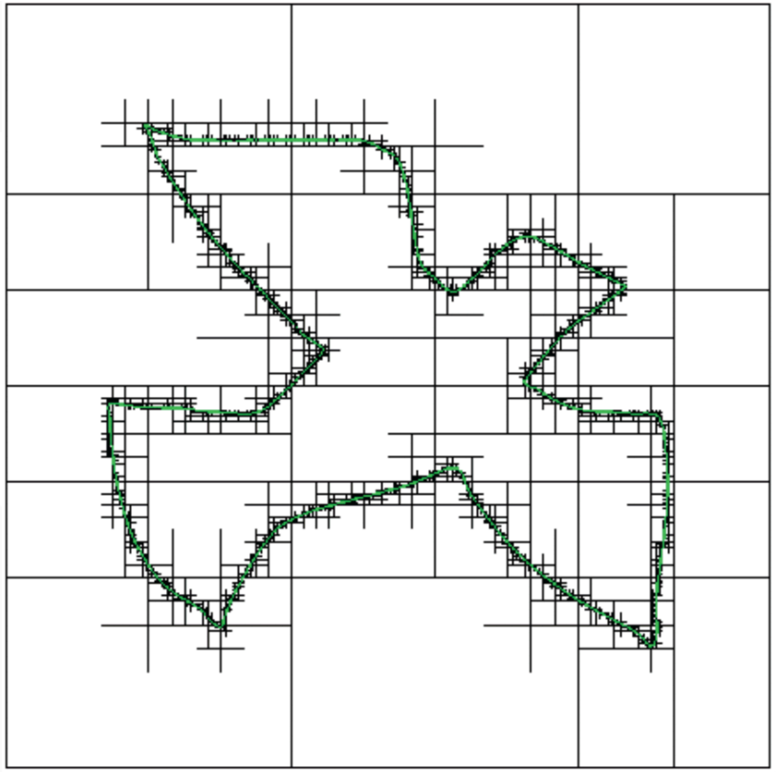
- A hierarchical tree build by sequential subdivision (8) of a occupied cells.
- Adaptive, i.e. only splits when too many points in cell
- Widely used for complicated scenes that need faster processing
  - E.g. Collision detection in real-time simulation or animation



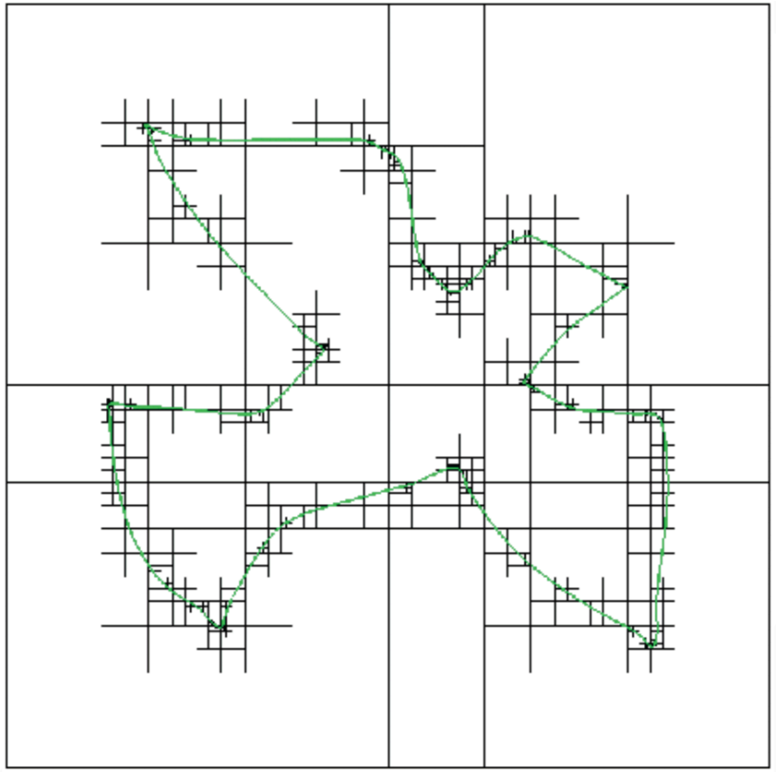


Paper: Adaptive O-CNN: A Patch-based Deep Representation of 3D Shapes

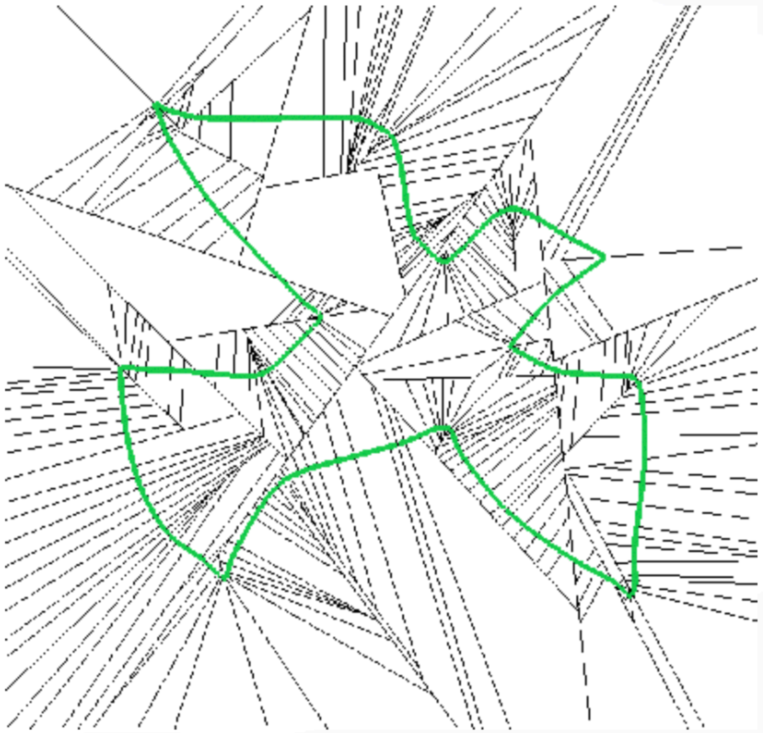
Adaptively Sampled Distance Fields



12040 cells



895 cells



254 cells

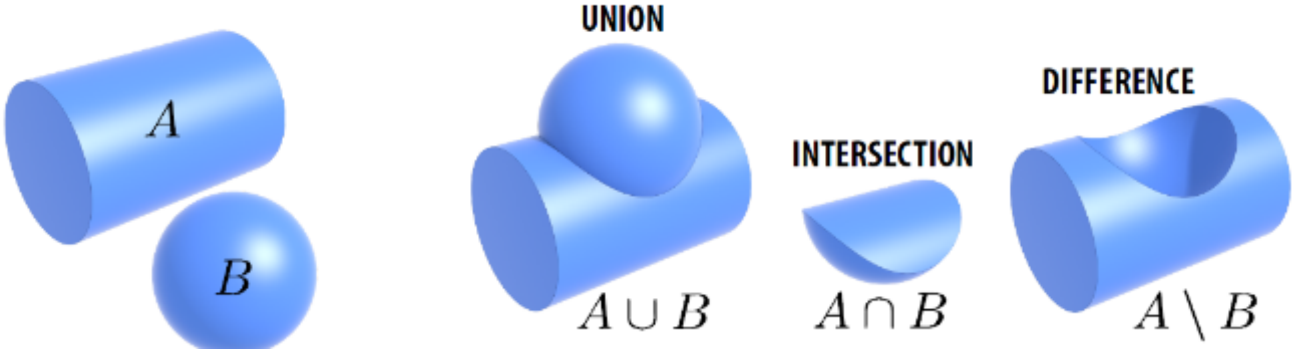
- Adaptively Sampled Distance Fields: A general representation of shape for computer graphics, SIGGRAPH 2000
- Piecewise Linear Approximation of Signed Distance Fields, VMV 2003

Implicit surface discretizations

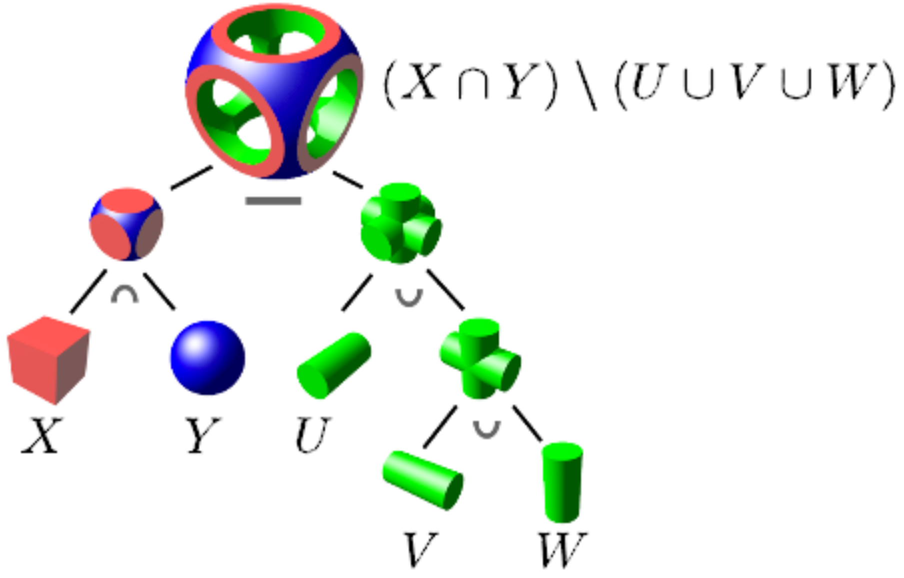
- Uniform, regular voxel grids $O(h^{-3})$
- Adaptive, 3-color octrees
  - Surface-adaptive refinement $O(h^{-2})$
  - Feature-adaptive refinement $O(h^{-1})$
- Irregular hierarchies
  - Binary space partition (BSP) $O(h^{-1})$

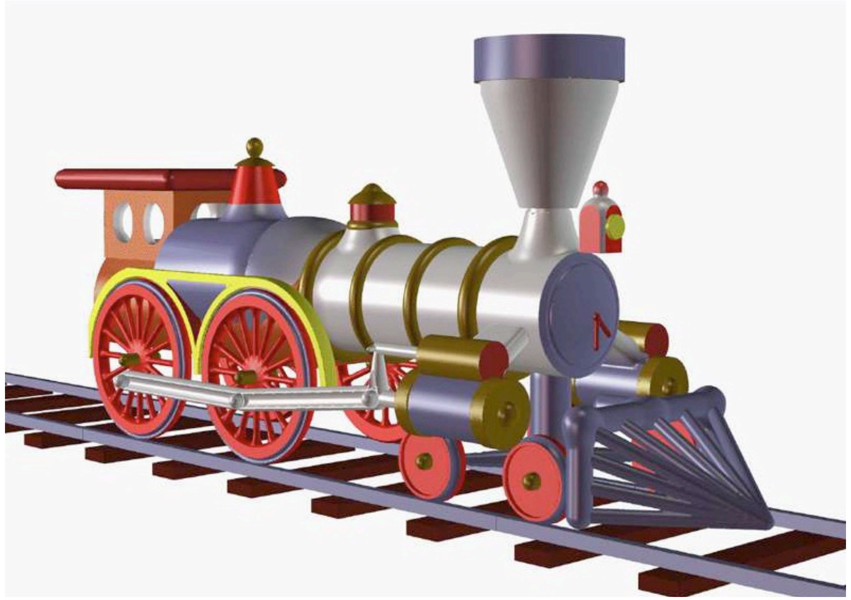
Constructive Solid Geometry (Implicit)

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



- Then chain together expressions:

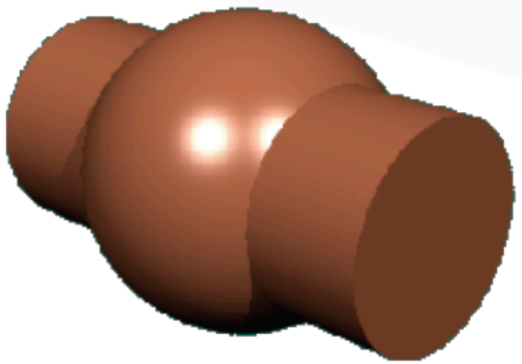




CSG Examples

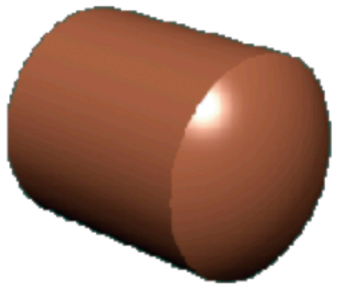
Union

$$F_{C \cup S}(\cdot) = \min\left\{F_C(\cdot)\,, F_S(\cdot)\right\}$$



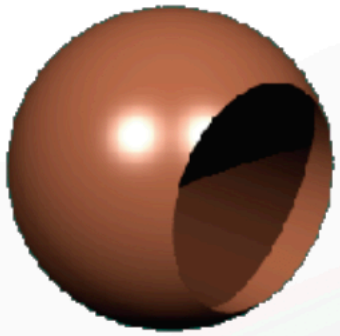
Intersection

$$F_{C \cap S}(\cdot) = \max\left\{F_C(\cdot)\,, F_S(\cdot)\right\}$$



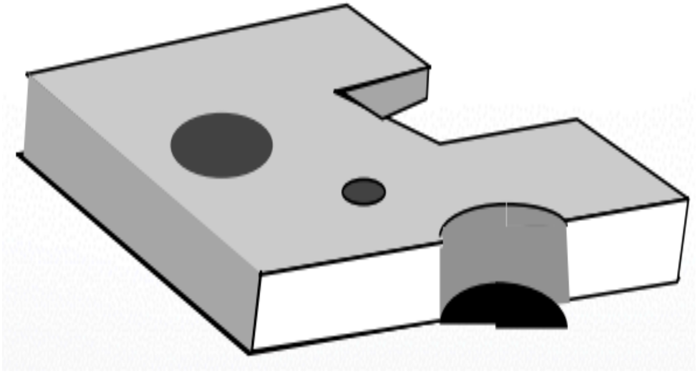
Difference

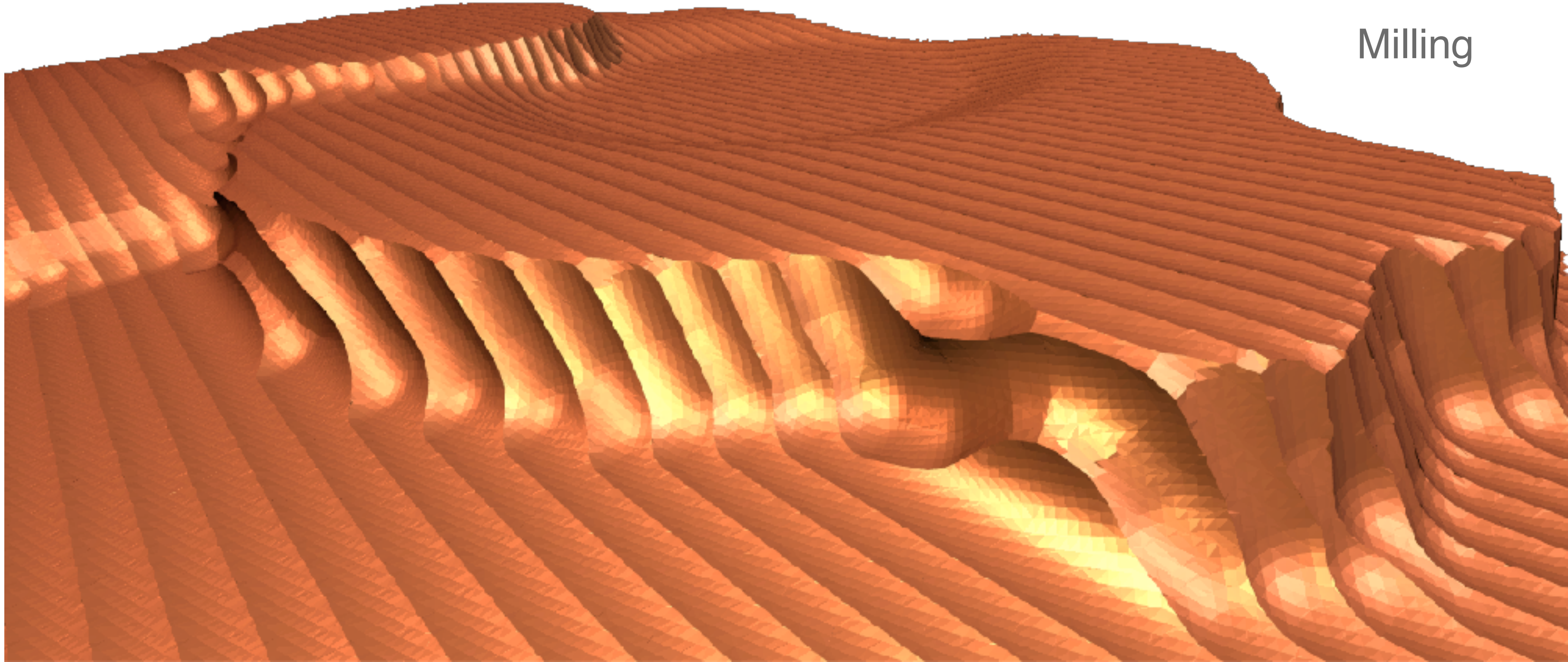
$$F_{S \setminus C}(\cdot) = \max\left\{-F_C(\cdot)\,, F_S(\cdot)\right\}$$



Constructive Solid Geometry (CSG)

- Machine an object - saw parts off, drill holes, glue pieces together
- This is sensible for objects that are actually made that way (human-made, particularly machined objects)

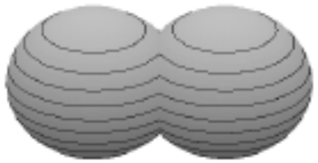




Milling

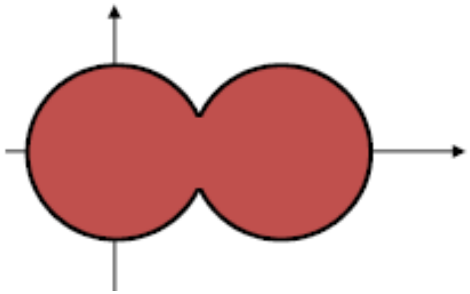
Implicit Surfaces -- Smooth set operation

- Standard operations: union and intersection



$$\bigcup_i g_i(\mathbf{p}) = \min g_i(\mathbf{p})$$

$$\bigcap_i g_i(\mathbf{p}) = \max g_i(\mathbf{p})$$

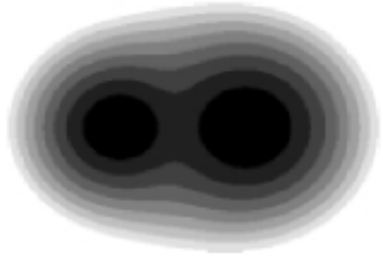


- In many cases, smooth blending is desired
  - Pasko and Savchenko [1994]

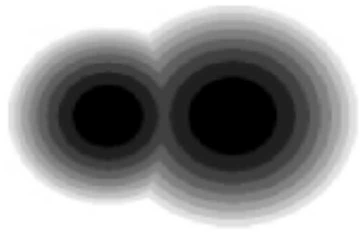
$$g \cup f = \tfrac{1}{1+\alpha}\left(g + f - \sqrt{g^2 + f^2 - 2\alpha g f}\right)$$

$$g \cap f = \tfrac{1}{1+\alpha}\left(g + f + \sqrt{g^2 + f^2 - 2\alpha g f}\right)$$

  - alpha



$\alpha = 0$

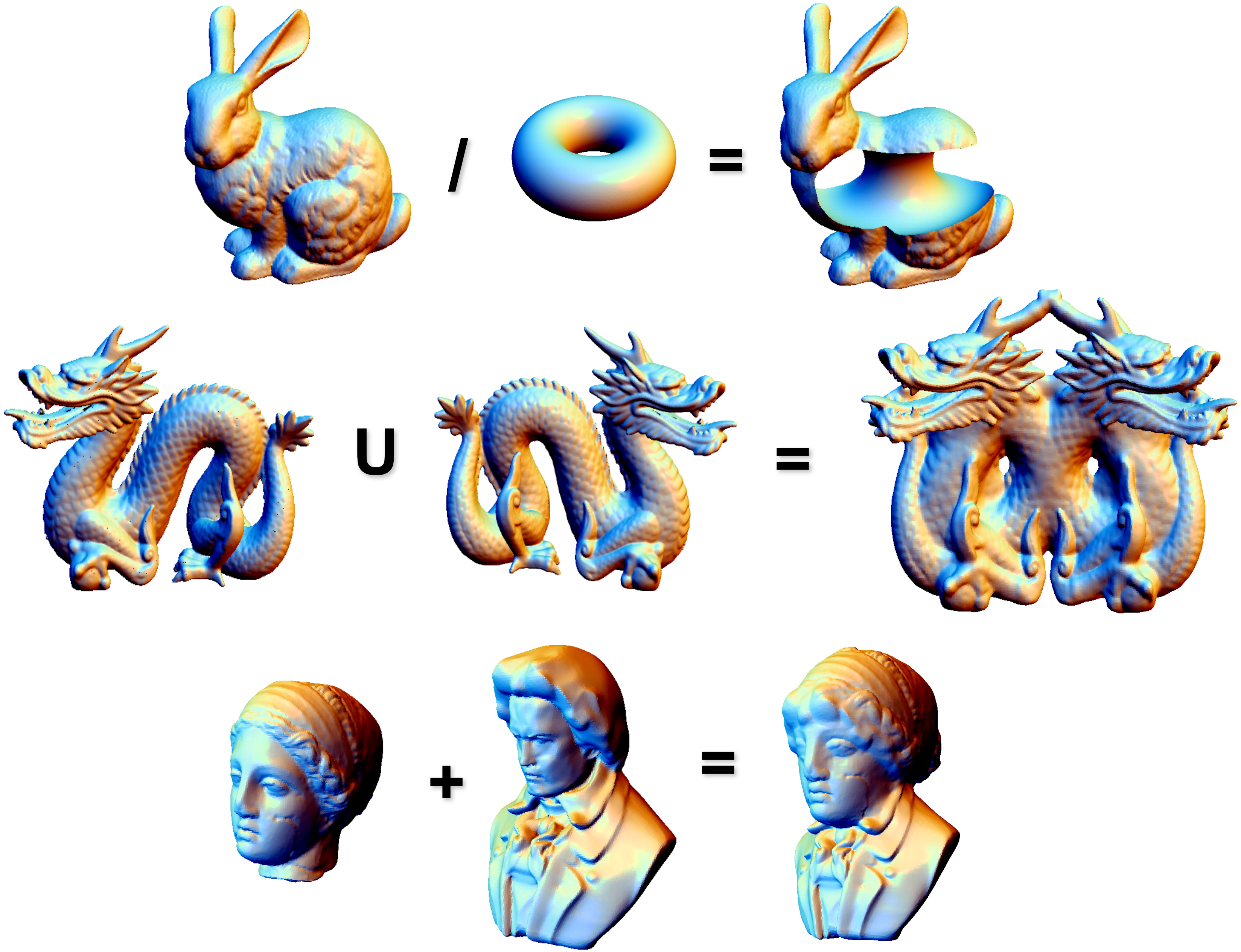


$\alpha = 1$

$$\lim_{\alpha \to 1} g \cup f = \tfrac{1}{2}\left(g + f - \sqrt{(g-f)^2}\right) = \frac{g+f}{2} - \frac{|g-f|}{2} = \min(g, f)$$

$$\lim_{\alpha \to 1} g \cap f = \tfrac{1}{2}\left(g + f + \sqrt{(g-f)^2}\right) = \frac{g+f}{2} + \frac{|g-f|}{2} = \max(g, f)$$

Shape Modeling with Implicits



Shape Modeling with Implicits

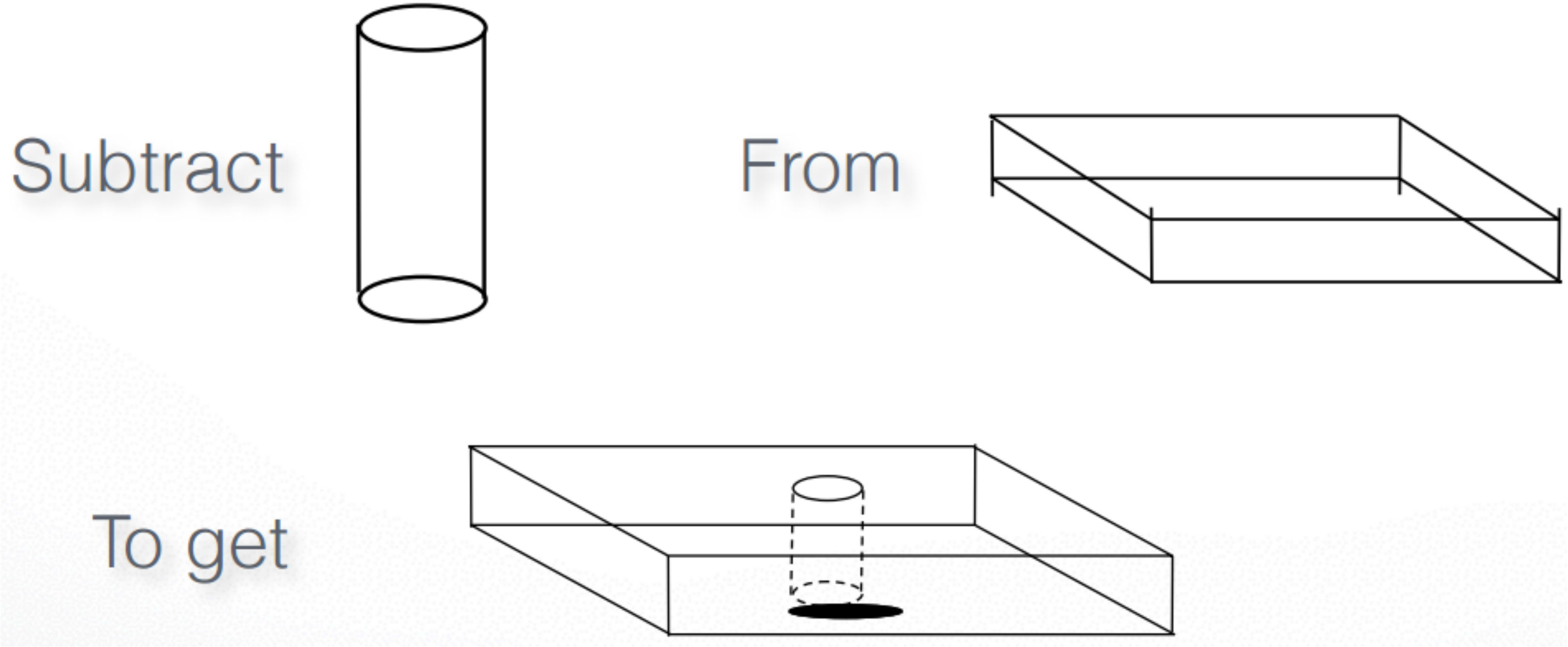
$$\boxed{f(\boldsymbol{x}) = (1-t)\, f_1(\boldsymbol{x}) + t\, f_2(\boldsymbol{x})}$$

$f_1(\boldsymbol{x}) = 0$

$f_2(\boldsymbol{x}) = 0$

Negative Objects

- Use point-by-point boolean functions
  - remove a volume by using a negative object
  - e.g. drill a hole by subtracting a cylinder



- Inside(BLOCK-CYL) = Inside(BLOCK) And Not(Inside(CYL))

Set Operations

- UNION: Inside(A) || Inside(B)

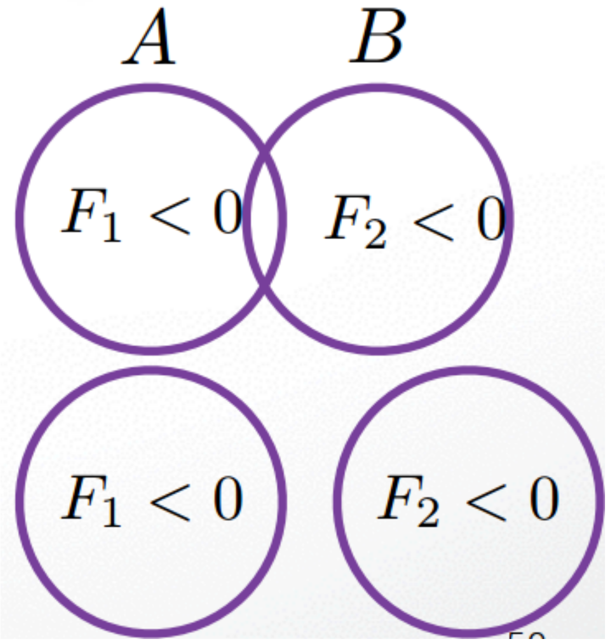
  Join A and B
- INTERSECTION: Inside(A) && Inside(B)

  Chop off any part of A that sticks out of B
- SUBTRACTION: Inside(A) && (! Inside(B))

  Use B to Cut A
- Examples:
  - Use cylinders to drill holes
  - Use rectangular blocks to cut slots
  - Use half-spaces to cut planar faces
  - Use surfaces swept from curves as jigsaws, etc

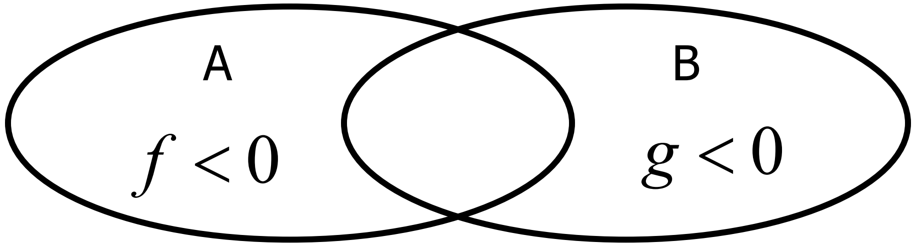
Implicit Functions for Booleans

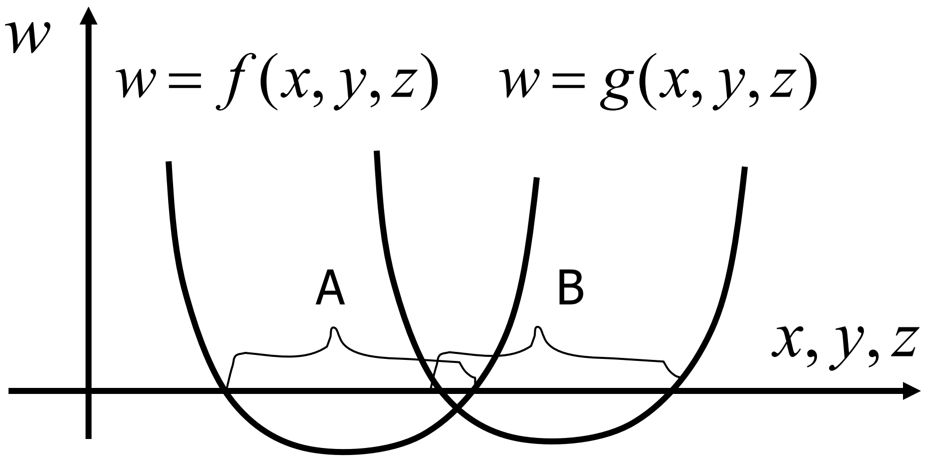
- Recall the implicit function for a solid: F(x,y,z)
- Boolean operations are replaced by arithmetic
  - MAX replaces And (intersection)
  - MIN replaces OR (union)
  - MINUS replaces NOT(unary subtraction)
- Thus
  - F(Intersect(A,B)) = MAX(F(A),F(B))
  - F(Union(A,B)) = MIN(F(A),F(B))
  - F(Subtract(A,B)) = MAX(F(A), -F(B))



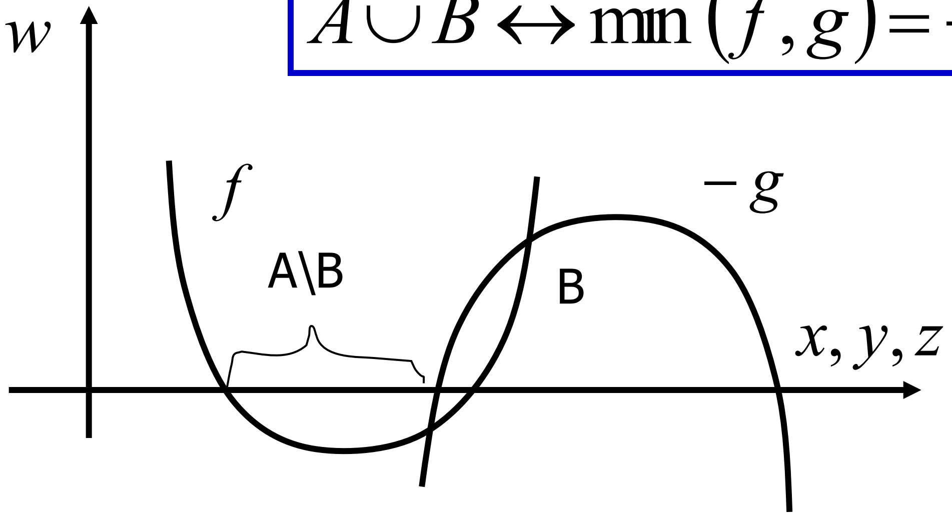
Boolean Operations with Implicits

$w = f(x,y,z) \quad w = g(x,y,z)$





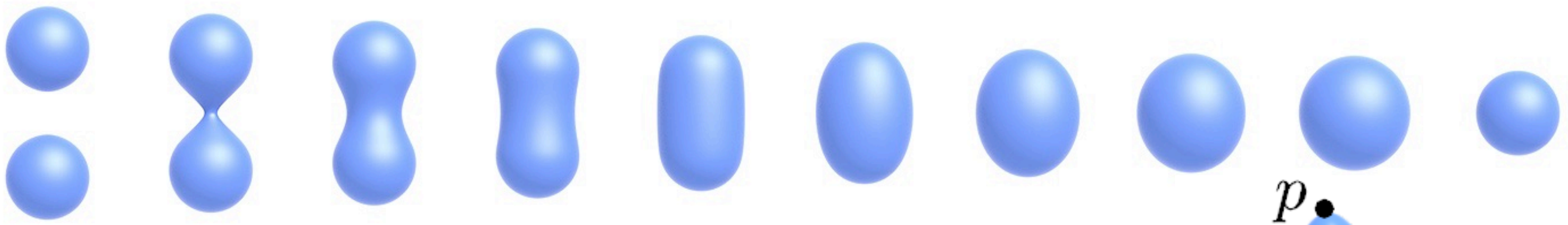
$$\boxed{A \cup B \leftrightarrow \min(f, g) = -\max(-f, -g)}$$



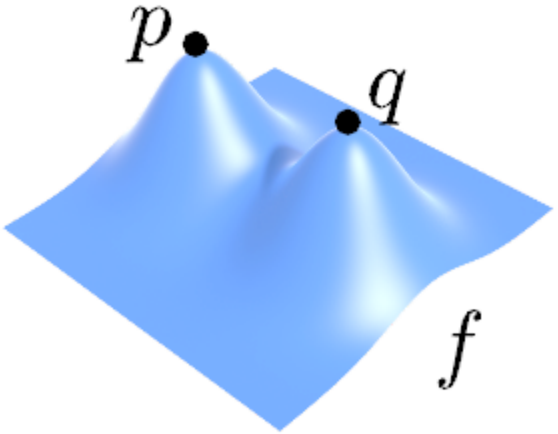
$$\boxed{A \setminus B \leftrightarrow \max(f, -g)}$$

Blobby Surfaces (Implicit)

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

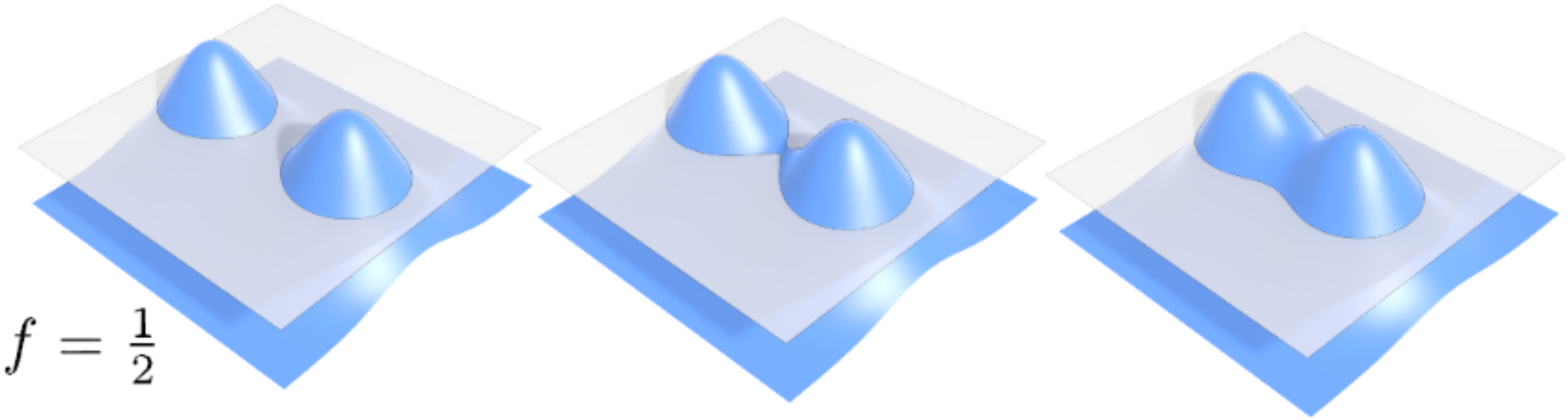


- **Easier to understand in 2D:**



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

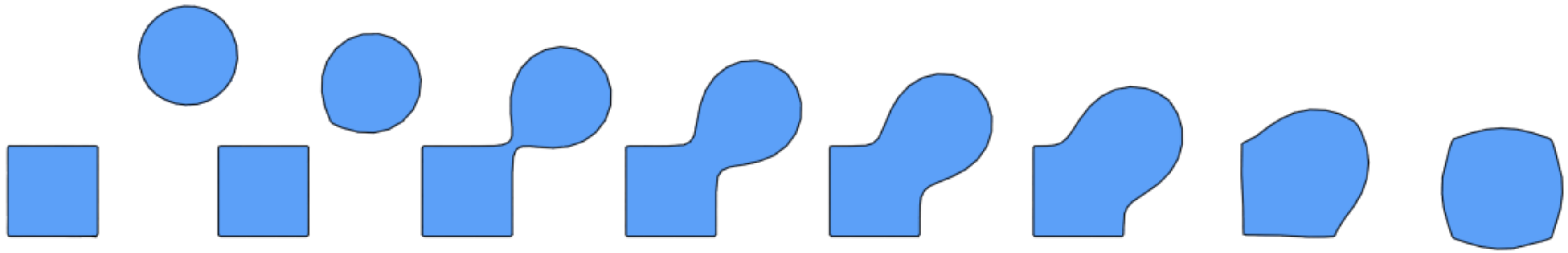
$$f := \phi_p + \phi_q$$ **(Sum of Gaussians centered at different points)**



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

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.,

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

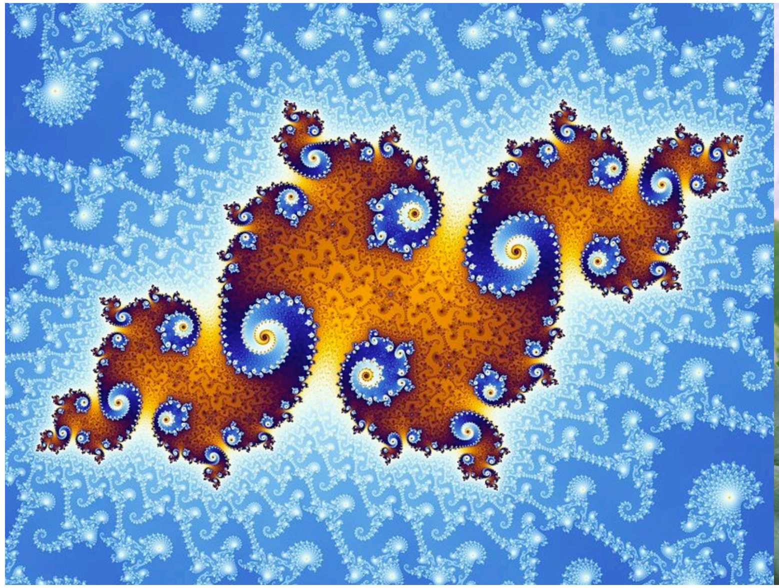
- Appearance depends on exactly how we combine functions
- Q: How do we implement a simple Boolean union?
- A: Just take the product: $f(x) := d_1(x)d_2(x)$

Scene of pure distance functions (not easy!)

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

Fractals (Implicit)

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



How to draw implicit surfaces?

- It’s easy to ray trace implicit surfaces
  - because of that easy intersection test
- Volume Rendering can display them

- Convert to polygons: the Marching Cubes algorithm
  - Divide space into cubes
  - Evaluate implicit function at each cube vertex
  - Do root finding or linear interpolation along each edge
  - Polygonize on a cube-by-cube basis

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***
    - Level set is easy to represent complex shapes, but still difficult to model …

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***