01 - Shape representation

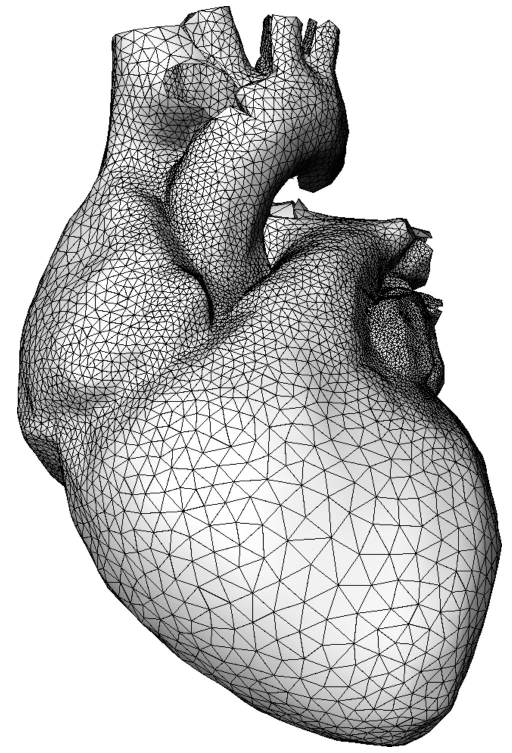
Acknowledgements: Daniele Panozzo

In this lecture

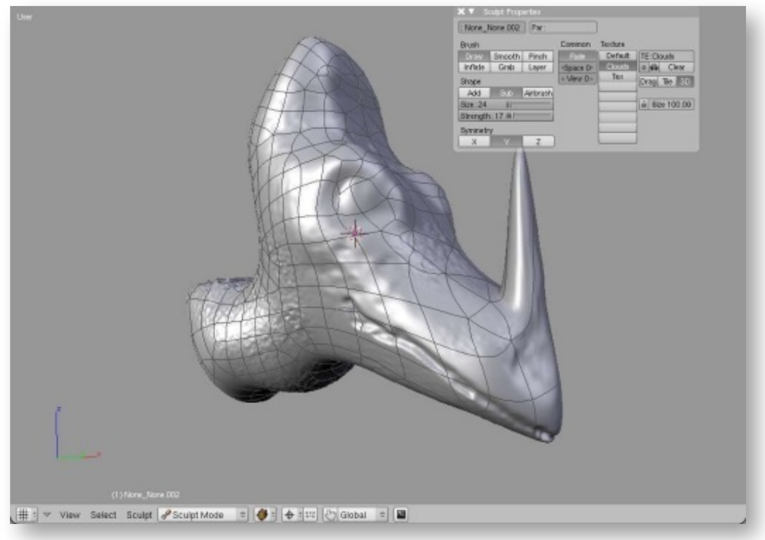
- Various ways to represent shapes
  - classification
  - overview only

Shape Representation: Origin- and Application-Dependent

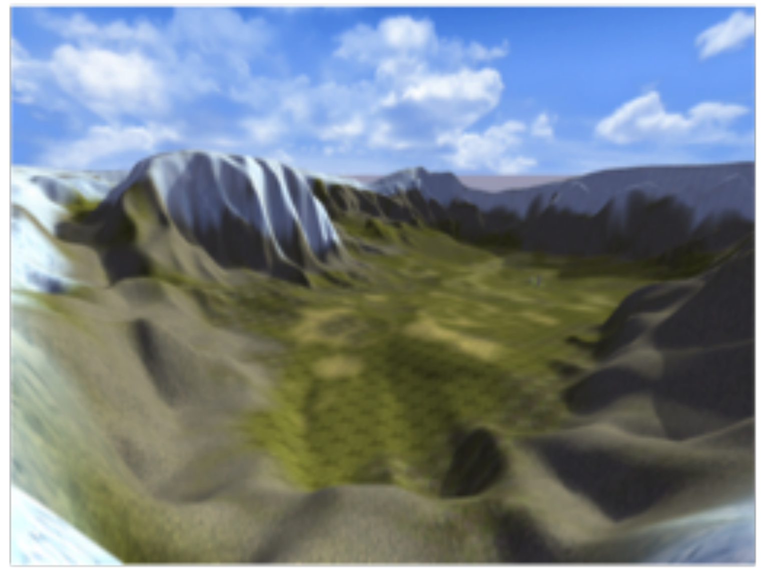
- Acquired real-world objects:
  - Discrete sampling
  - Points, meshes

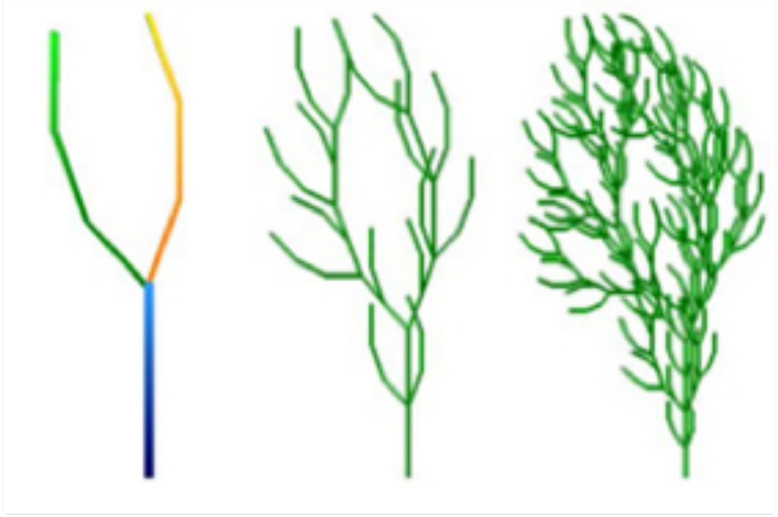


- Modeling "by hand":
  - Higher-level representations, amendable to modification, control
  - Parametric surfaces, subdivision surfaces, implicit surfaces



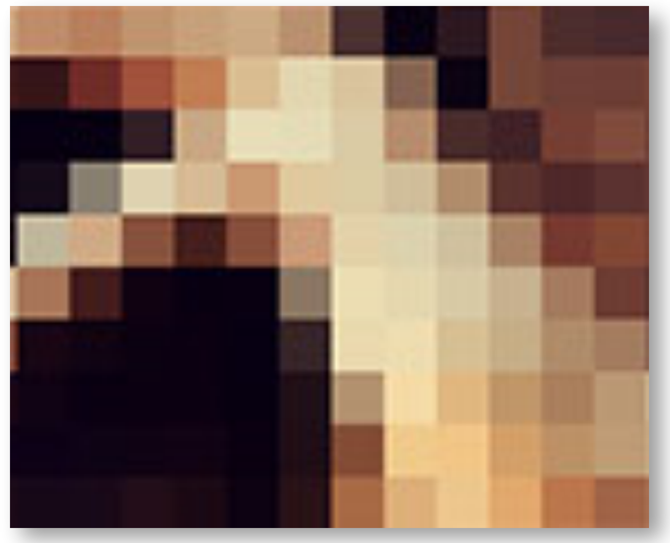
- Procedural modeling
  - Algorithms, grammars

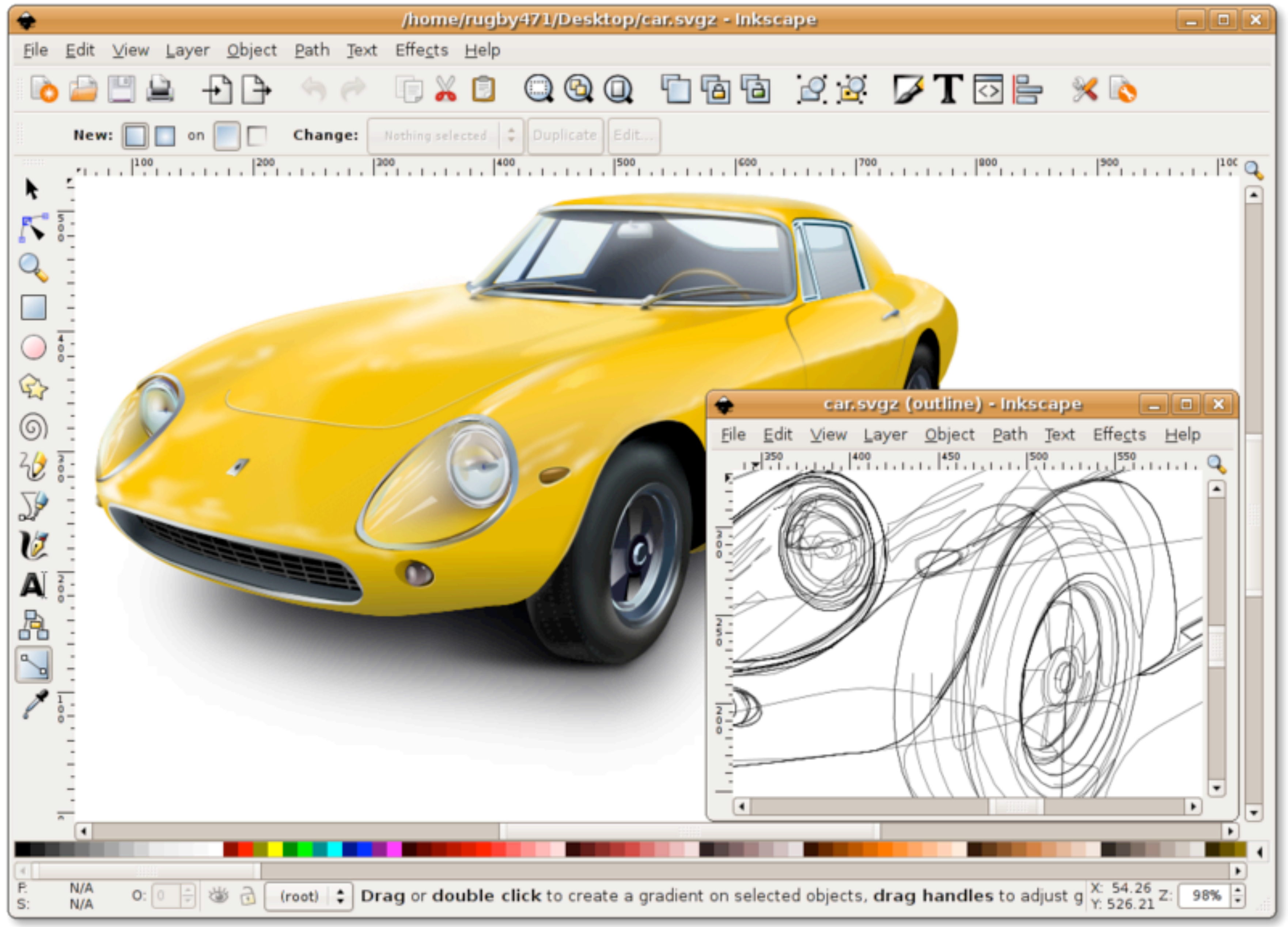




Similar to the 2D Image Domain

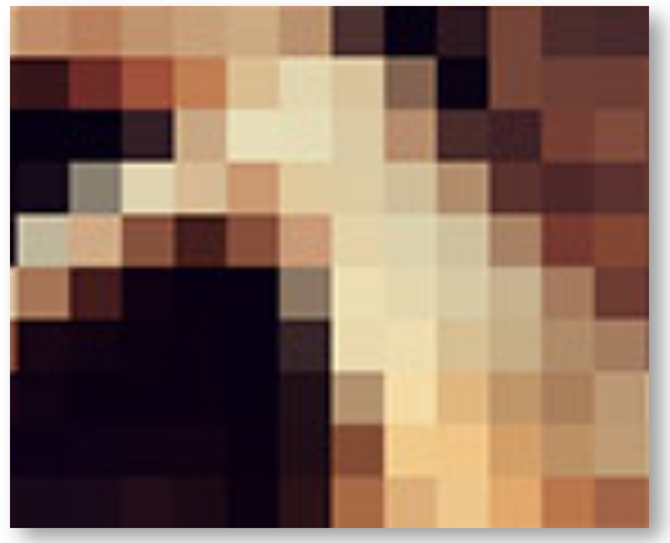
- Acquired digital images:
  - Discrete sampling
  - Pixels on a grid
- Painting “by hand”:
  - Strokes + color/shading
  - Vector graphics
  - Controls for editing



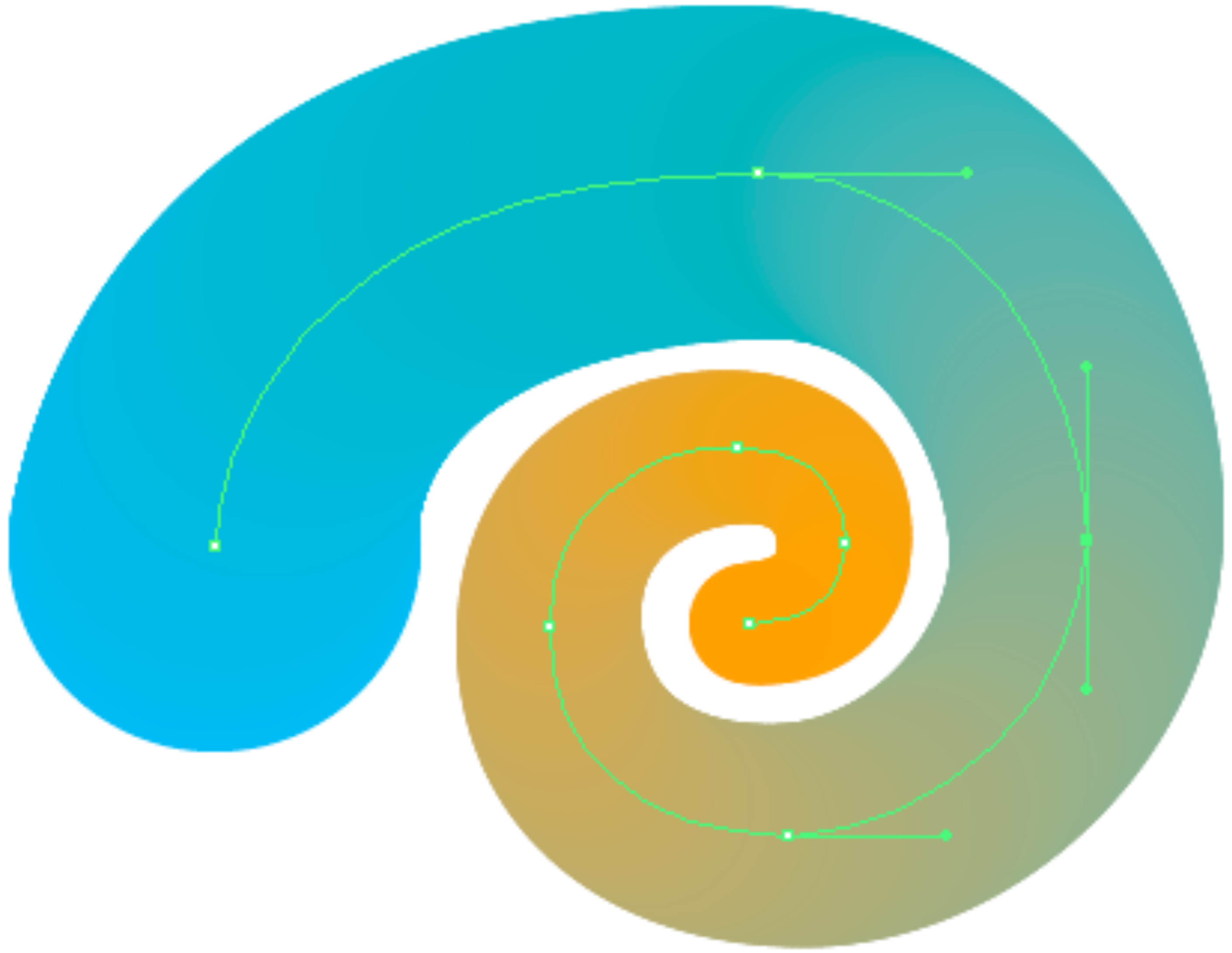


Similar to the 2D Image Domain

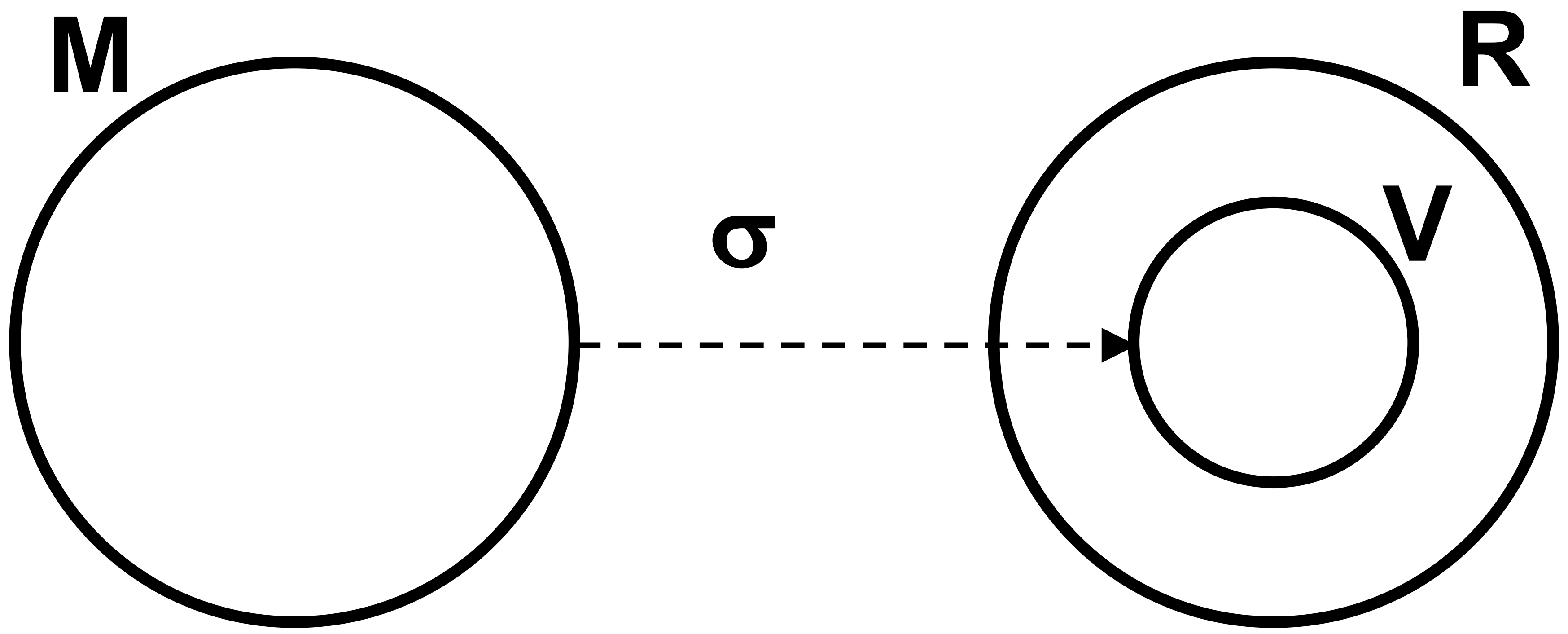
- Acquired digital images:
  - Discrete sampling
  - Pixels on a grid



- Painting “by hand”:
  - Strokes + color/shading
  - Vector graphics
  - Controls for editing

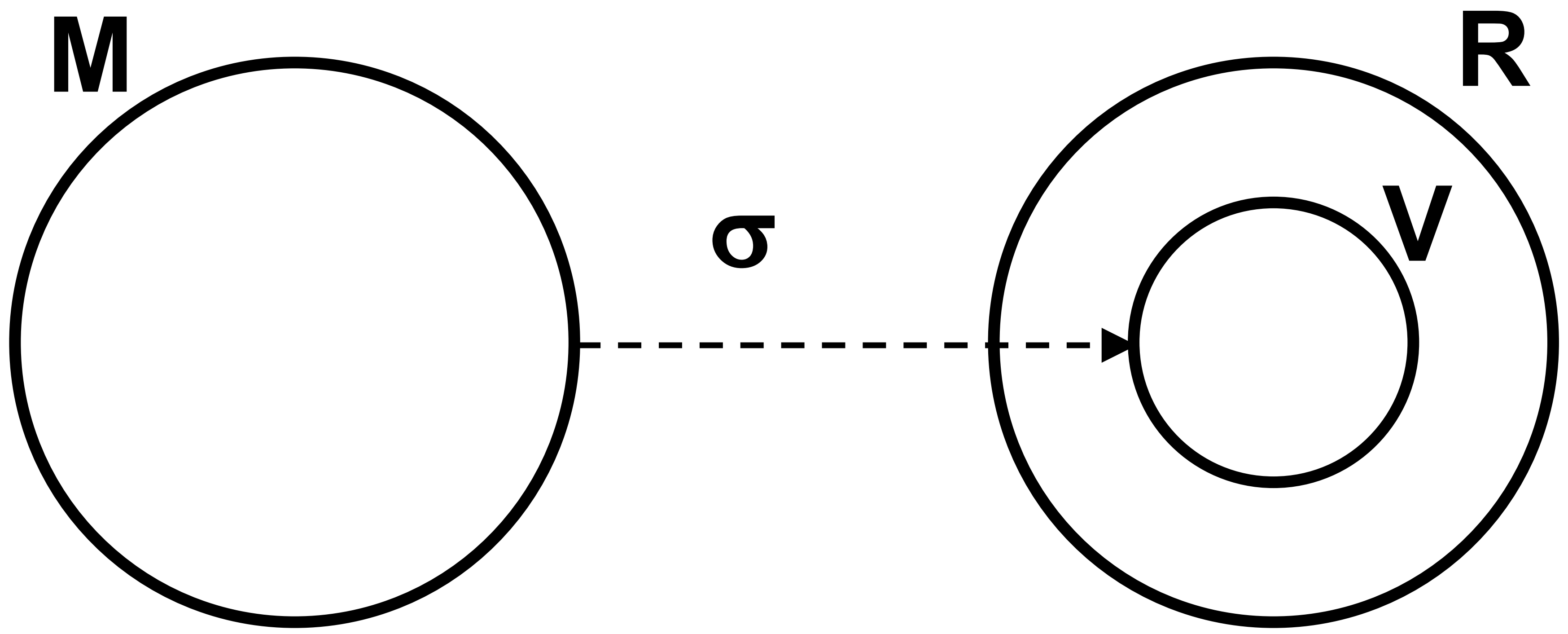


Representation: mathematical view



- Space of models **M**:
  - Physical, belongs to real world
  - Including just what we represent: *shape*
- Space of representations **R**:
  - Symbolic, belongs to virtual/computer world
  - Structured from syntax
- Space of *valid* representations **V**:
  - Image of **M** on **R** through **σ** representation function

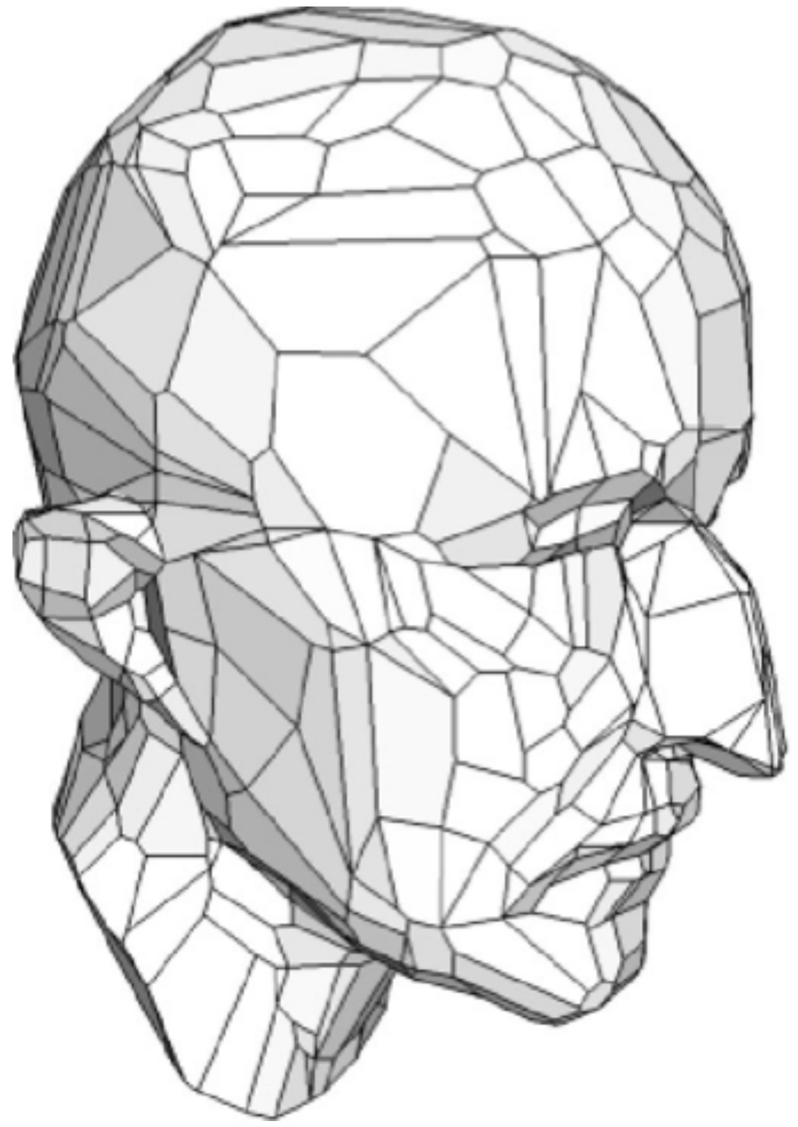
Representation: mathematical view



- (Our view of) physical world is analogical
  - Elements of **M** are *ideal* and *exact* objects
- Virtual/computer world is digital
  - Elements of **V** provide *approximate* representations of elements of **M**
- Function **σ** is meant to formalize the process of model creation
  - From analogical to digital
  - From *ideal* and *exact* to *approximate*
- Not all elements of **R** represent valid objects

Representation Considerations

- How should we represent geometry?
  - Needs to be stored in the computer
  - Creation of new shapes
    - Input metaphors, interfaces...
  - What operations do we apply?
    - Editing, simplification, smoothing, filtering, repair, deform...
  - How to render it?
    - Rasterization, raytracing...

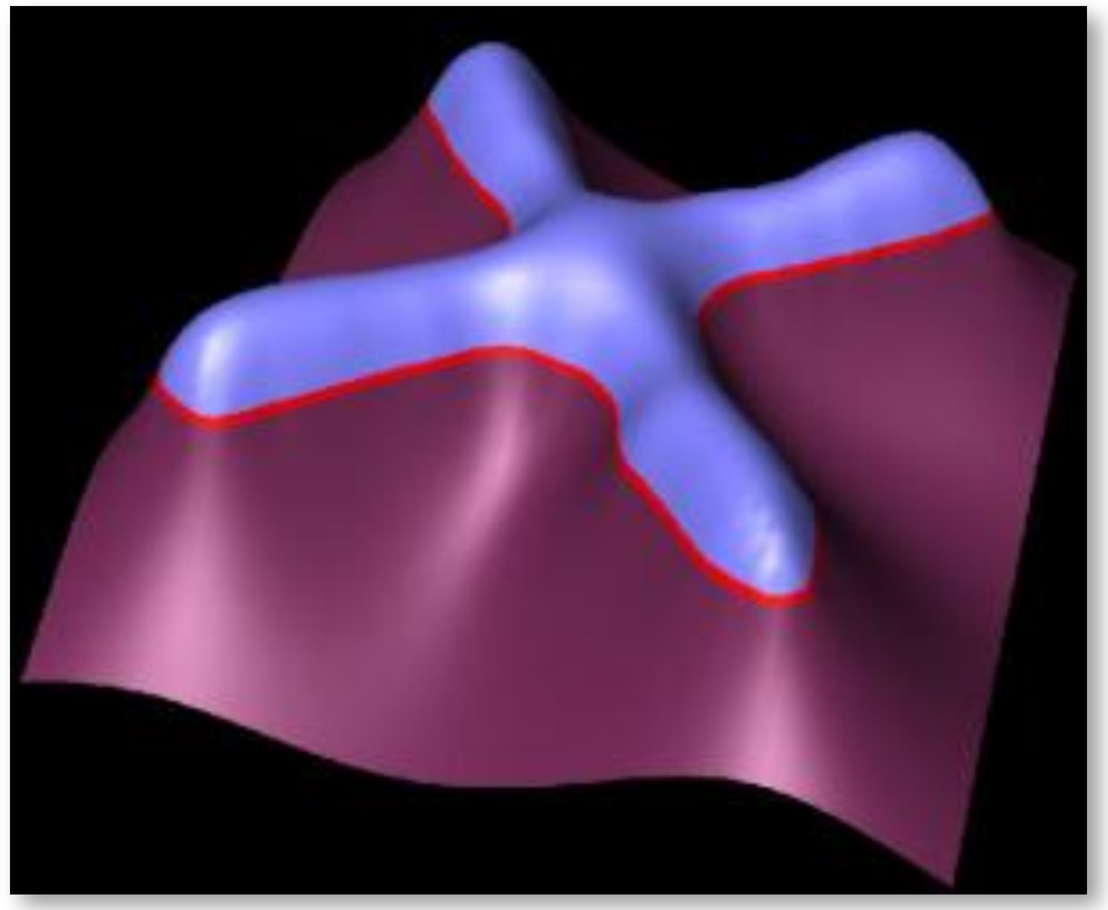


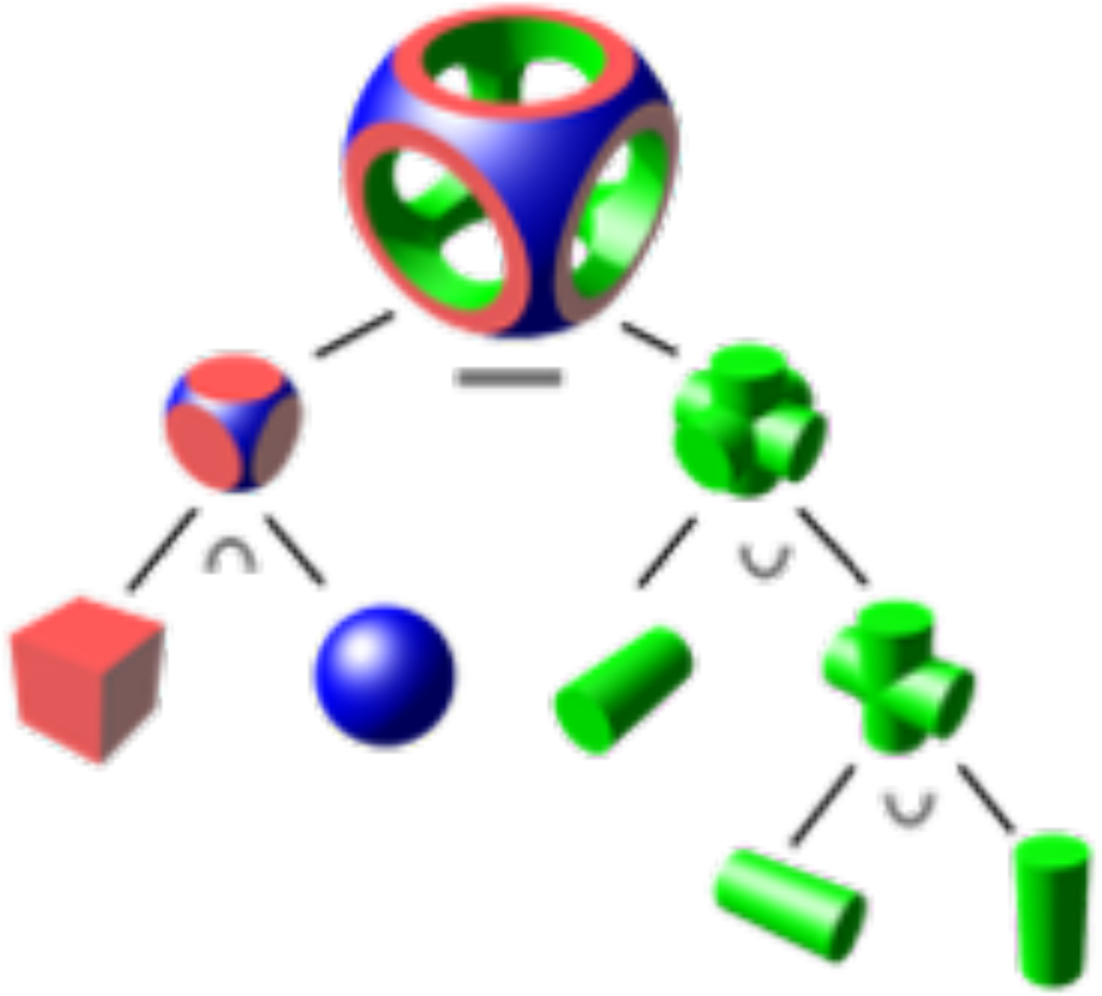
Variational Shape Approximation

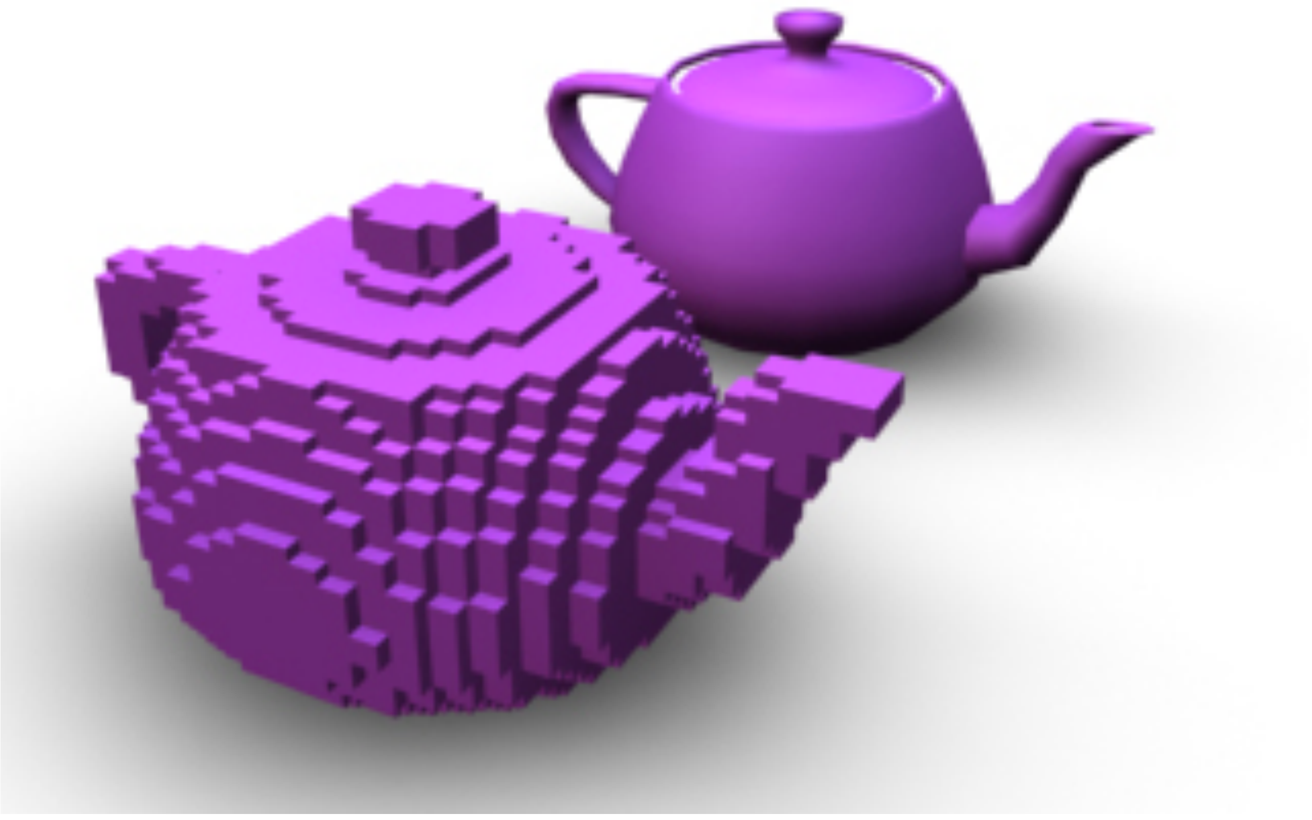
Representation Schemes

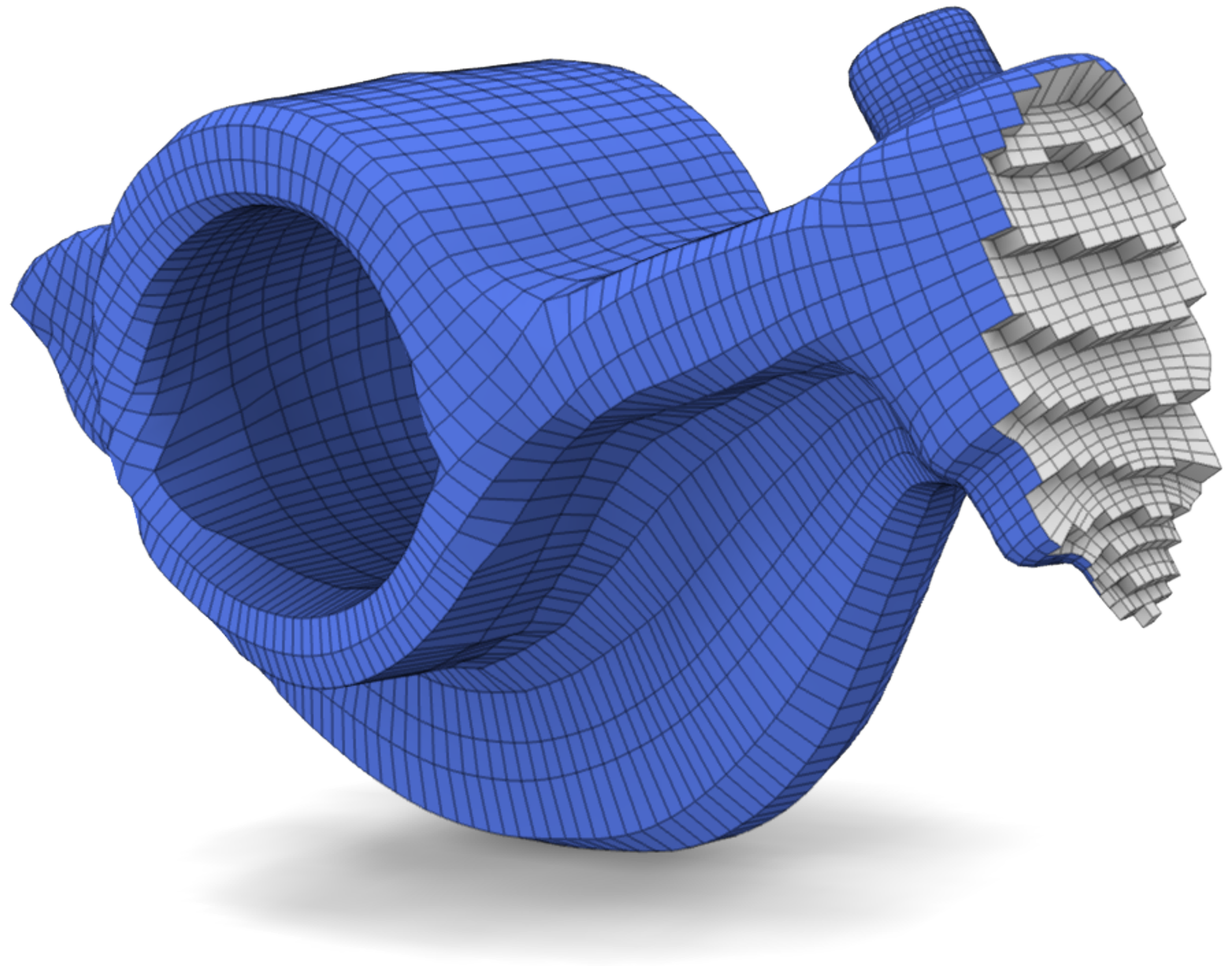
Represent object through the space it occupies:

- interior/exterior/ boundary
  - Object-based: Implicit functions, CSG, volume meshes
  - Space-based: Voxels





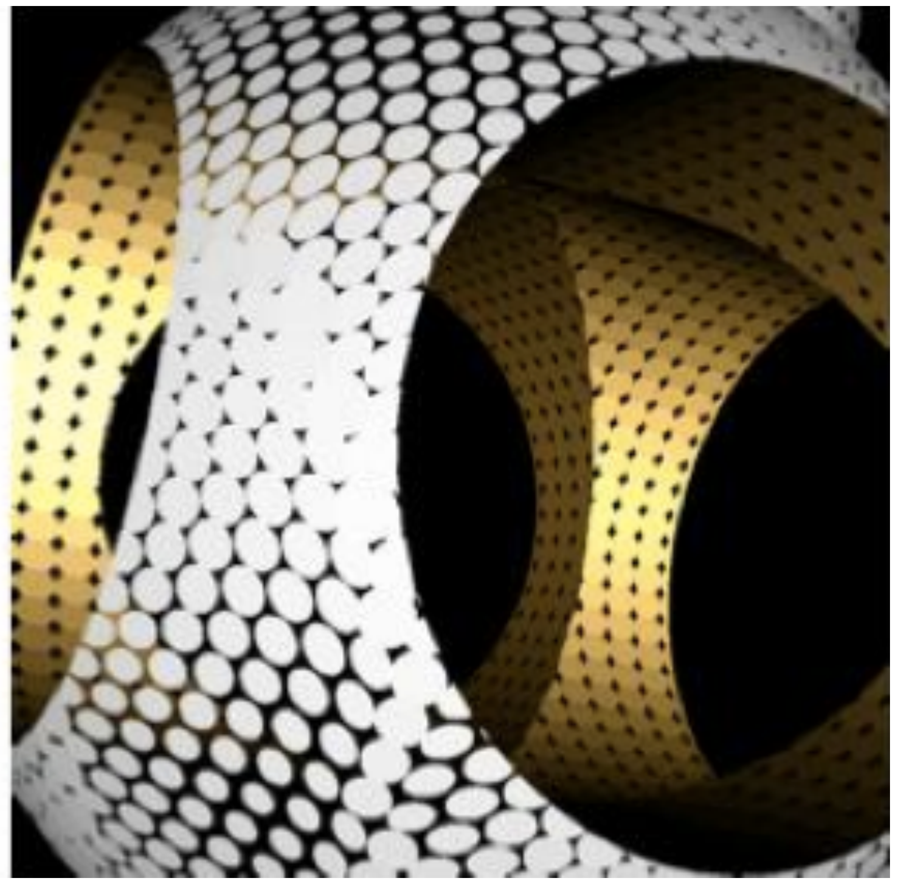


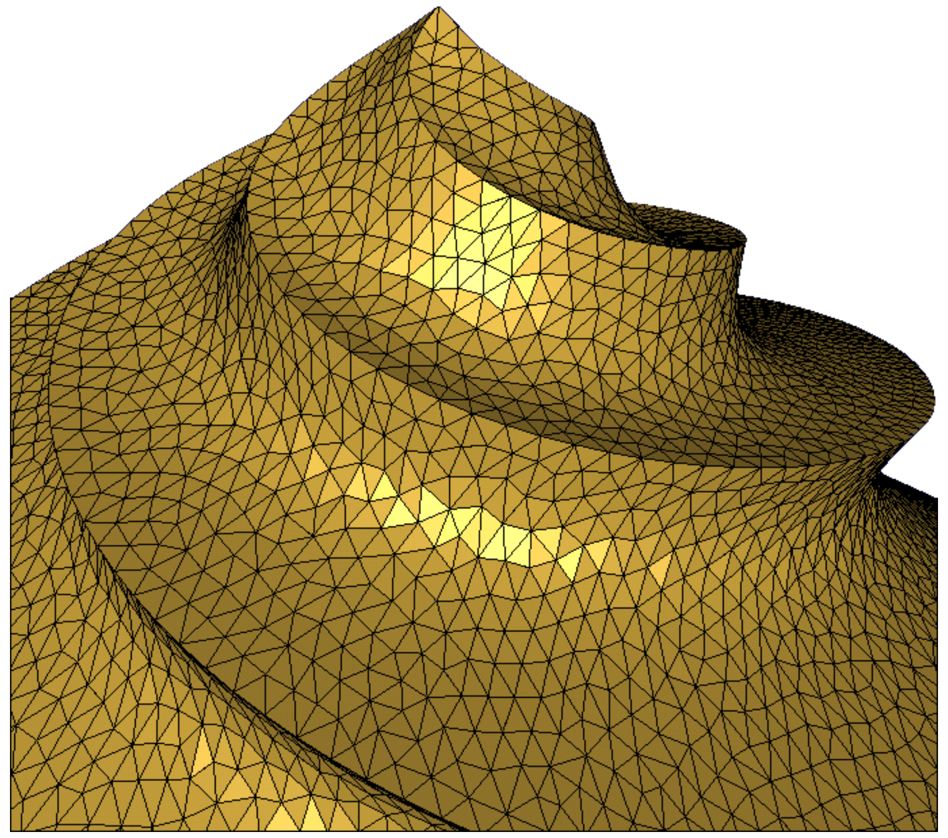


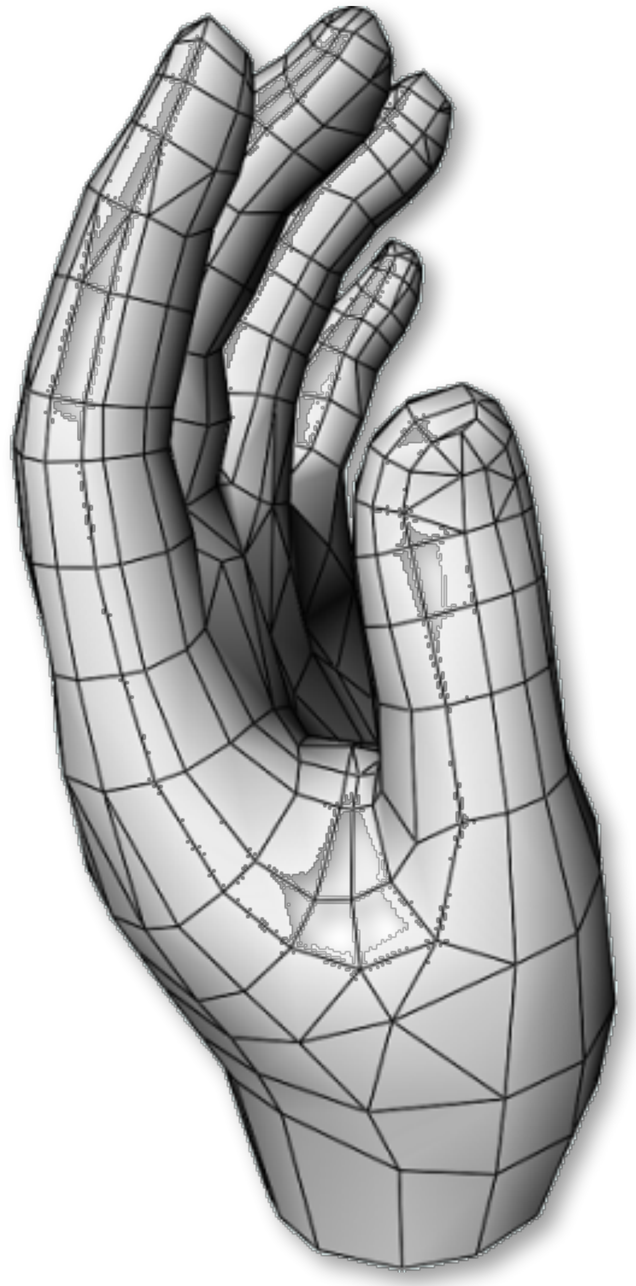
Representation Considerations

Represent object through its surface:

- just the boundary! interior/exterior are implicit
  - Points (discrete)
  - Polygonal meshes
  - Parametric & Subdivision surfaces





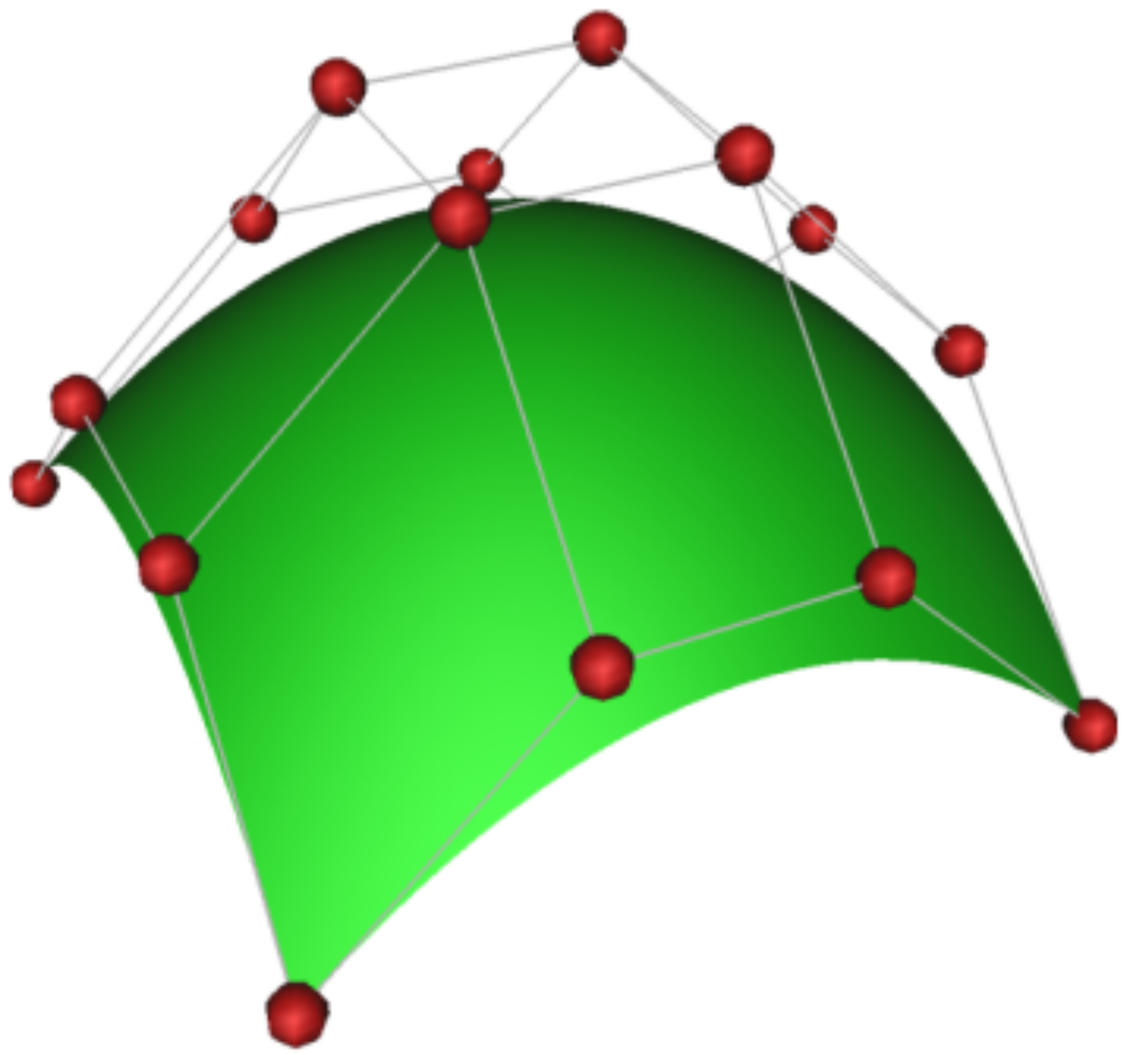


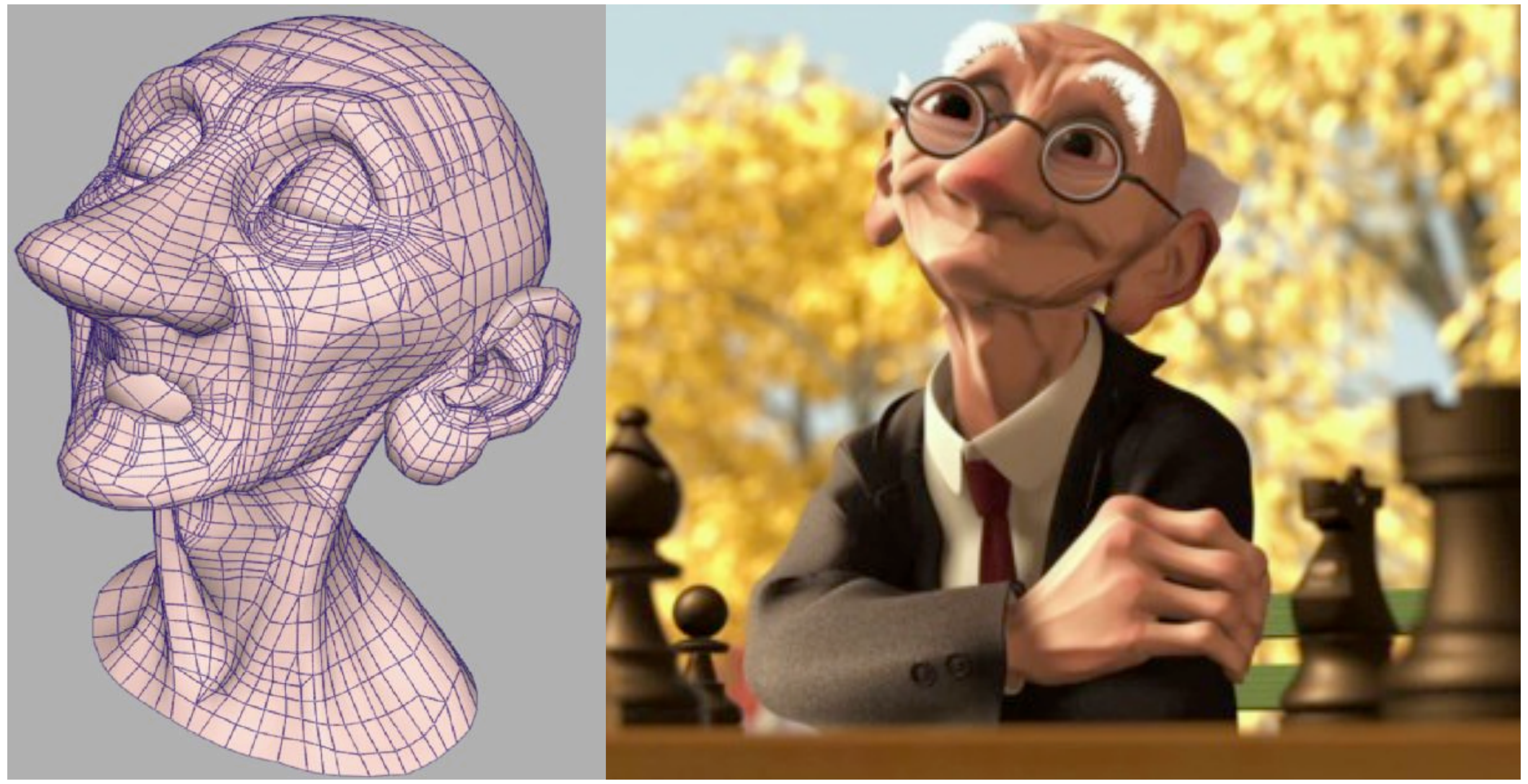
Anisotropic Polygonal Remeshing

Representation Considerations

Represent object through its surface:

- just the boundary! interior/exterior are implicit
  - Points (discrete)
  - Polygonal meshes
  - Parametric & Subdivision surfaces

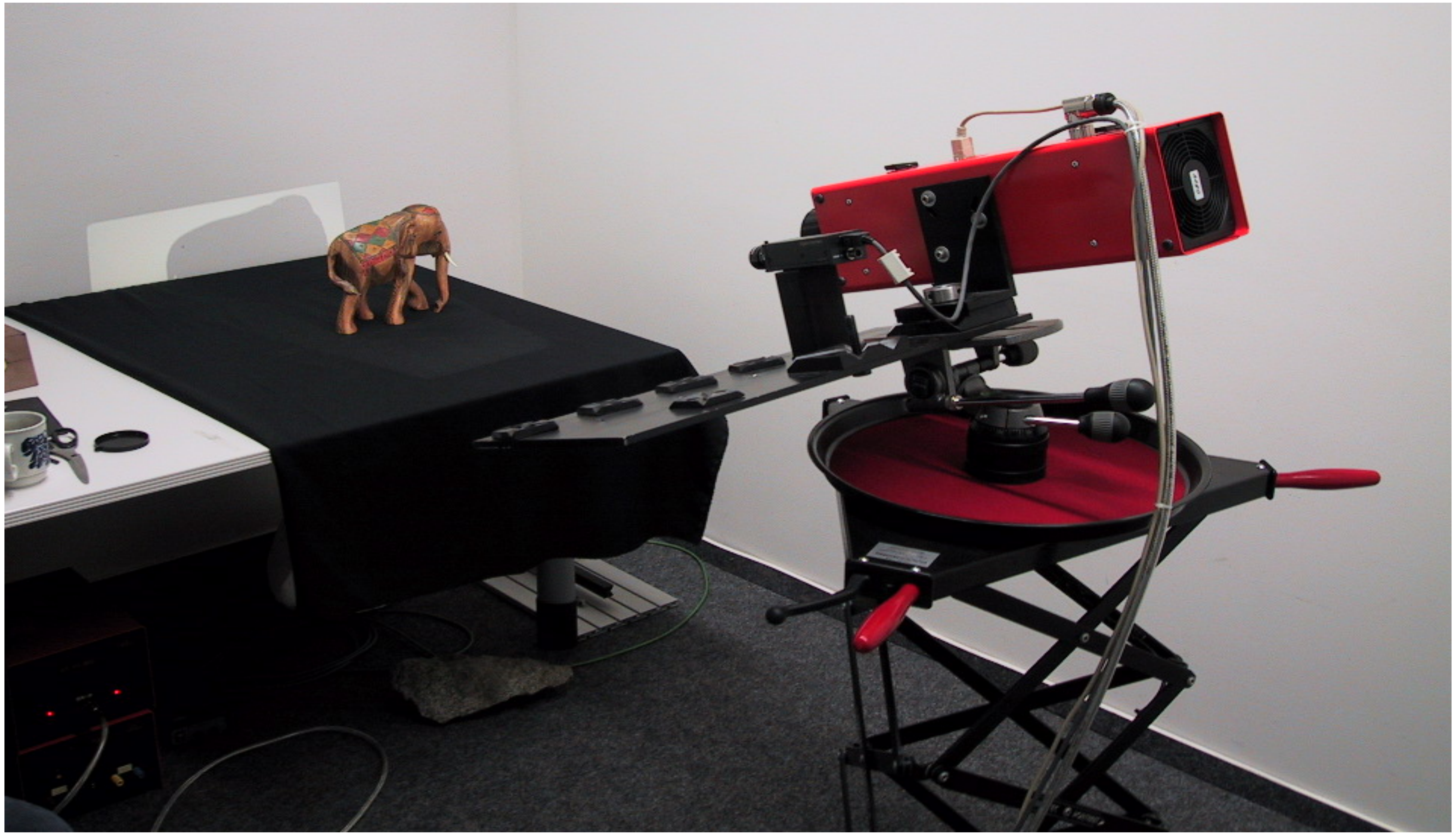




Geri's game - Pixar

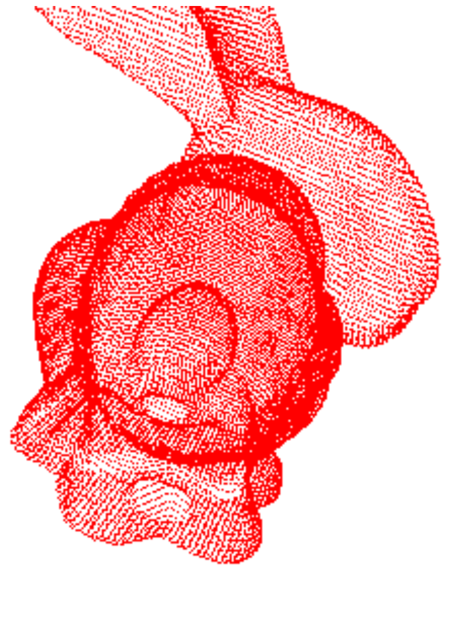
Points

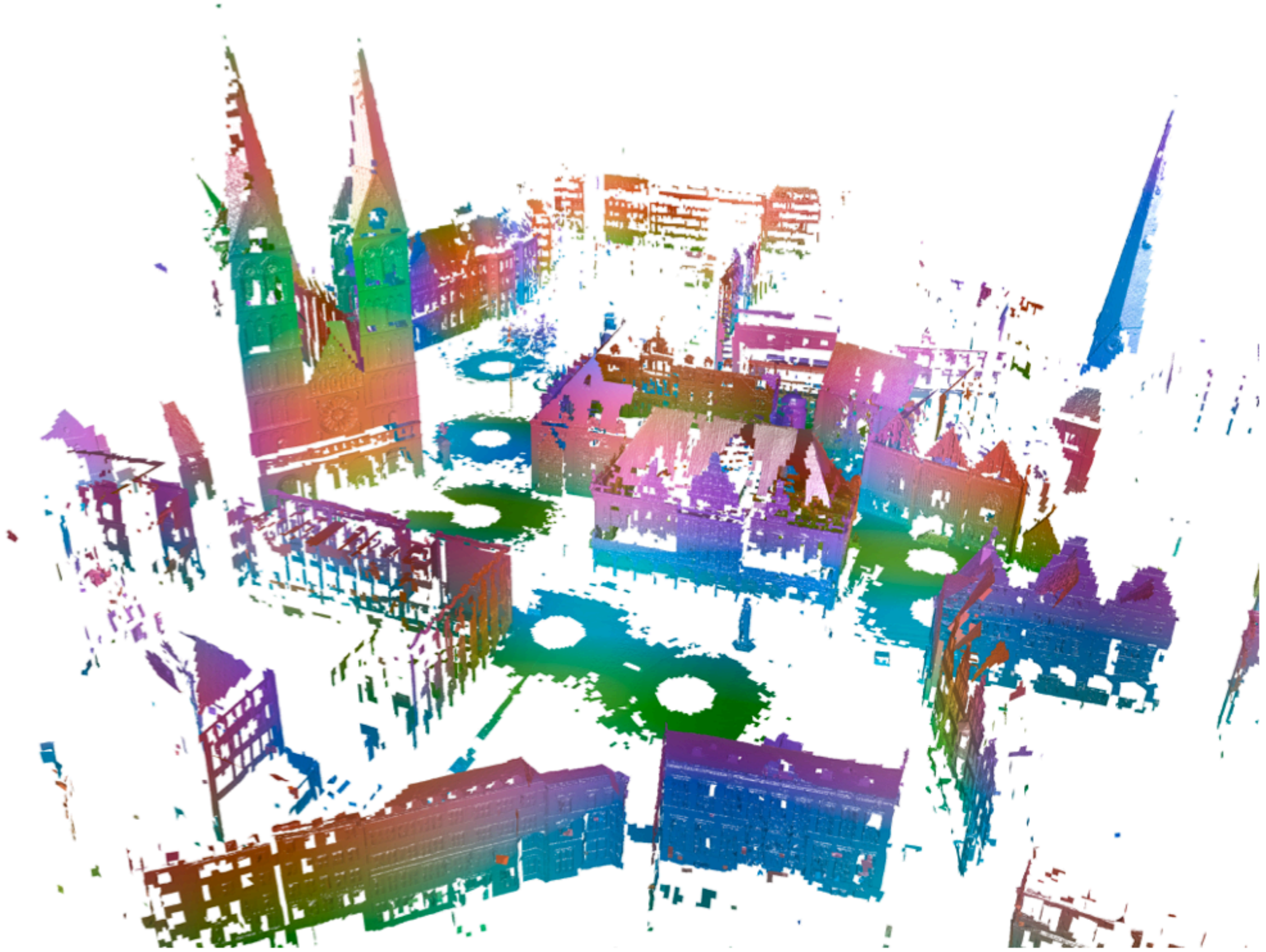
Output of Acquisition



Points

- Standard 3D data from a variety of sources
  - Often results from scanners
  - Potentially noisy

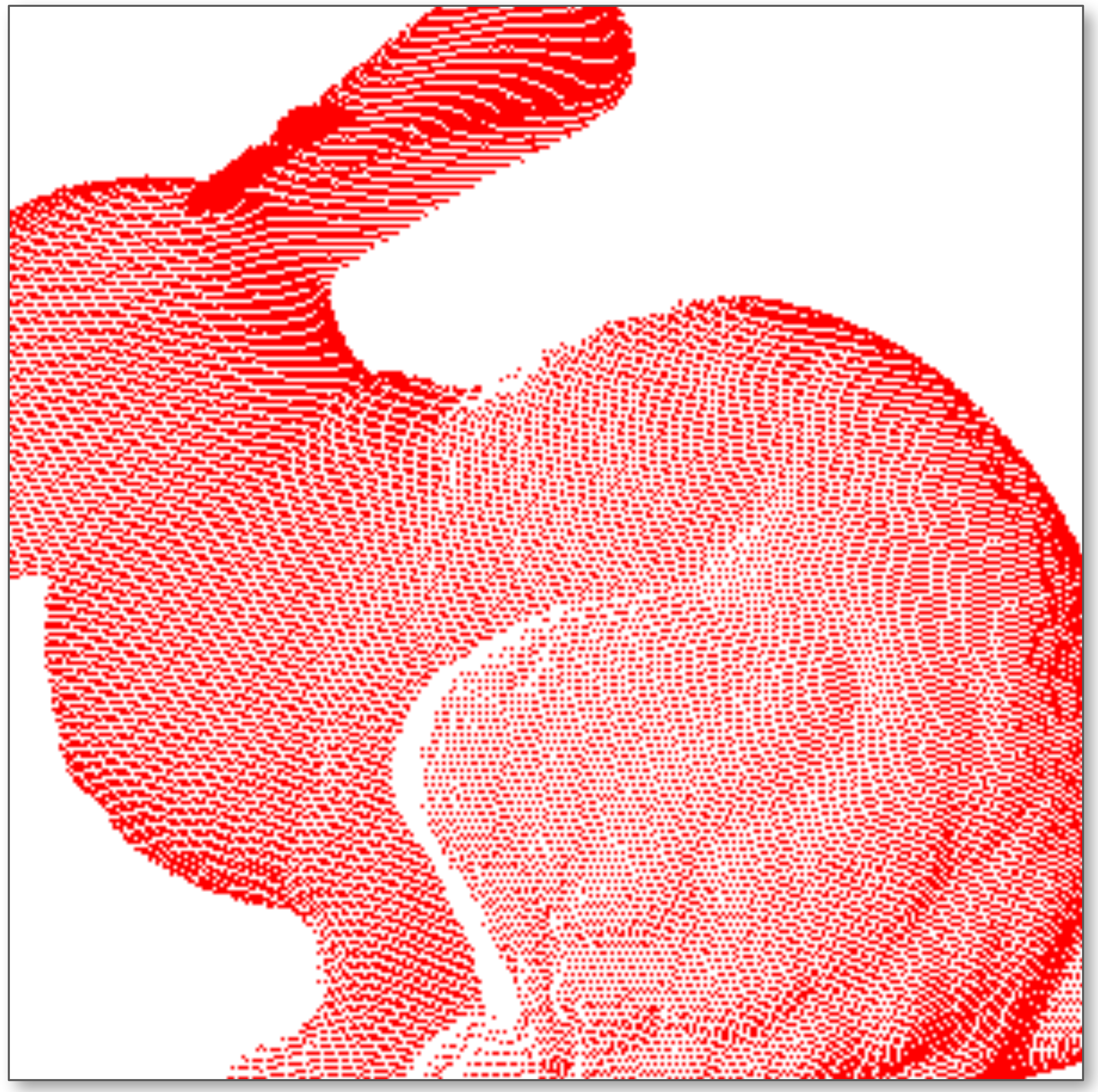




  - Depth imaging
  - Registration of multiple images

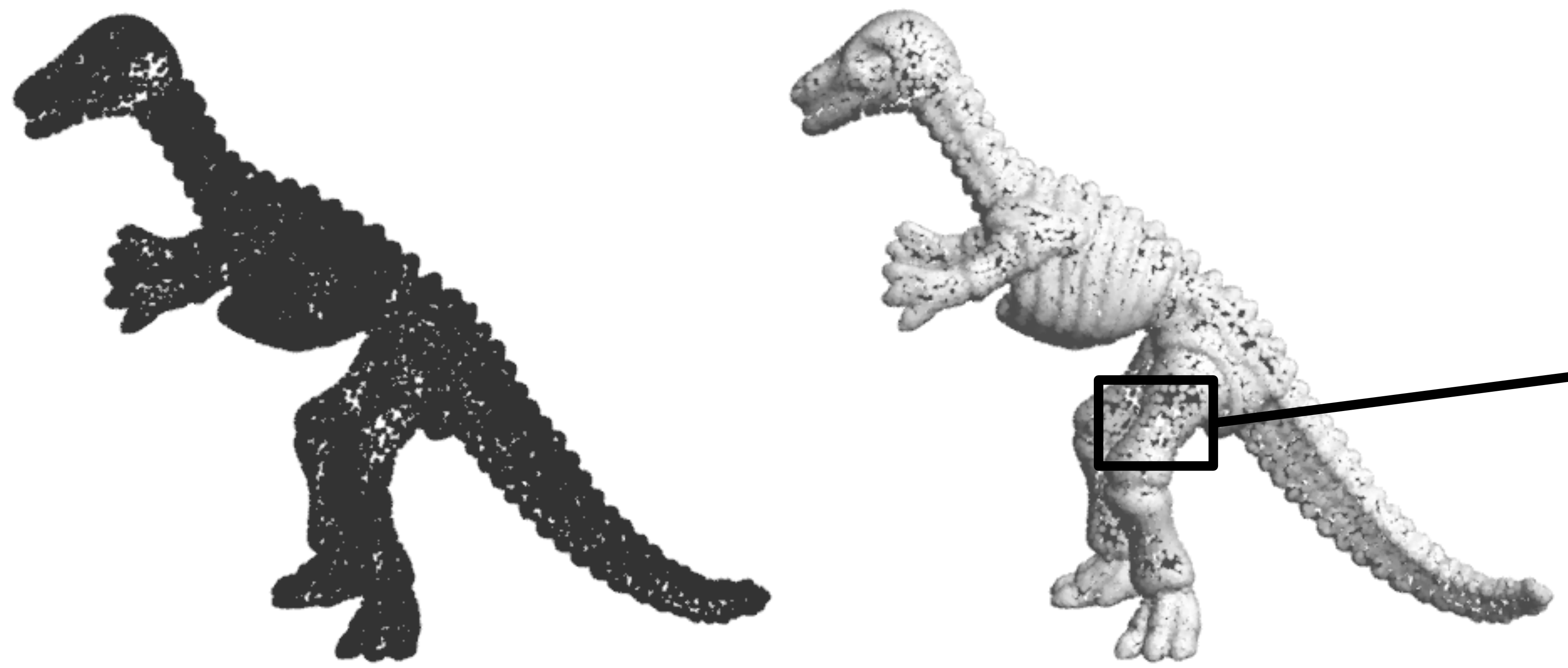
Points

- Points = unordered set of 3-tuples
  - sometimes: point = differential of surface
  - Not just a position: normal and curvature, too
- Often converted to other representations
  - Meshes, implicit functions, parametric surfaces
  - Easy to encode, process, edit and render
- Efficient point processing and modeling requires a spatial partitioning data structure to figure out neighborhoods
  - Query: which are the neighbors (on the surface) of a given point?

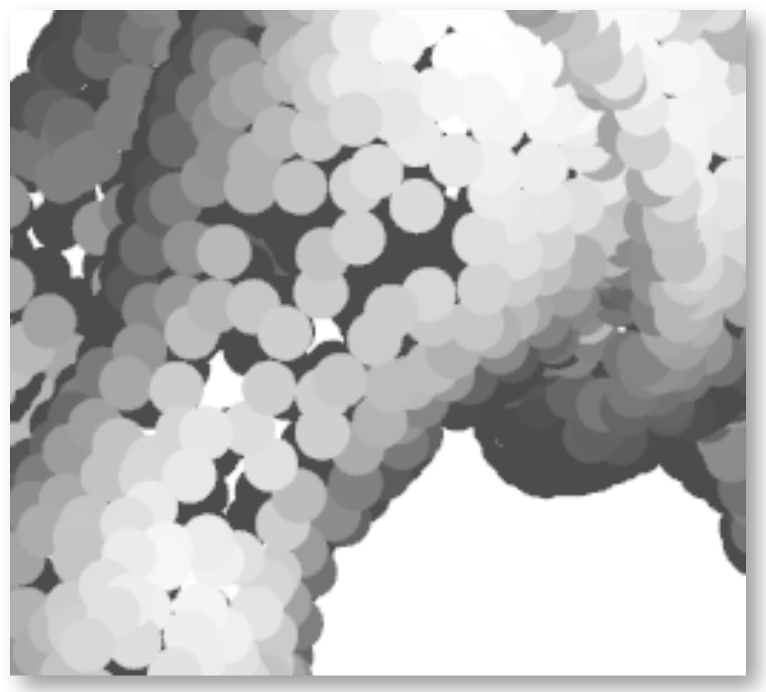


Points: Neighborhood information

- Why do we need neighbors?



need normals (for shading)



upsampling –
need to estimate density

- Need sub-linear-time implementations of
  - k-nearest neighbors to point $\mathbf{x}$
  - In-radius search $\|\mathbf{p}_i - \mathbf{x}\| < \varepsilon$

Parametric Curves and Surfaces

(introduction; specific classes later on)

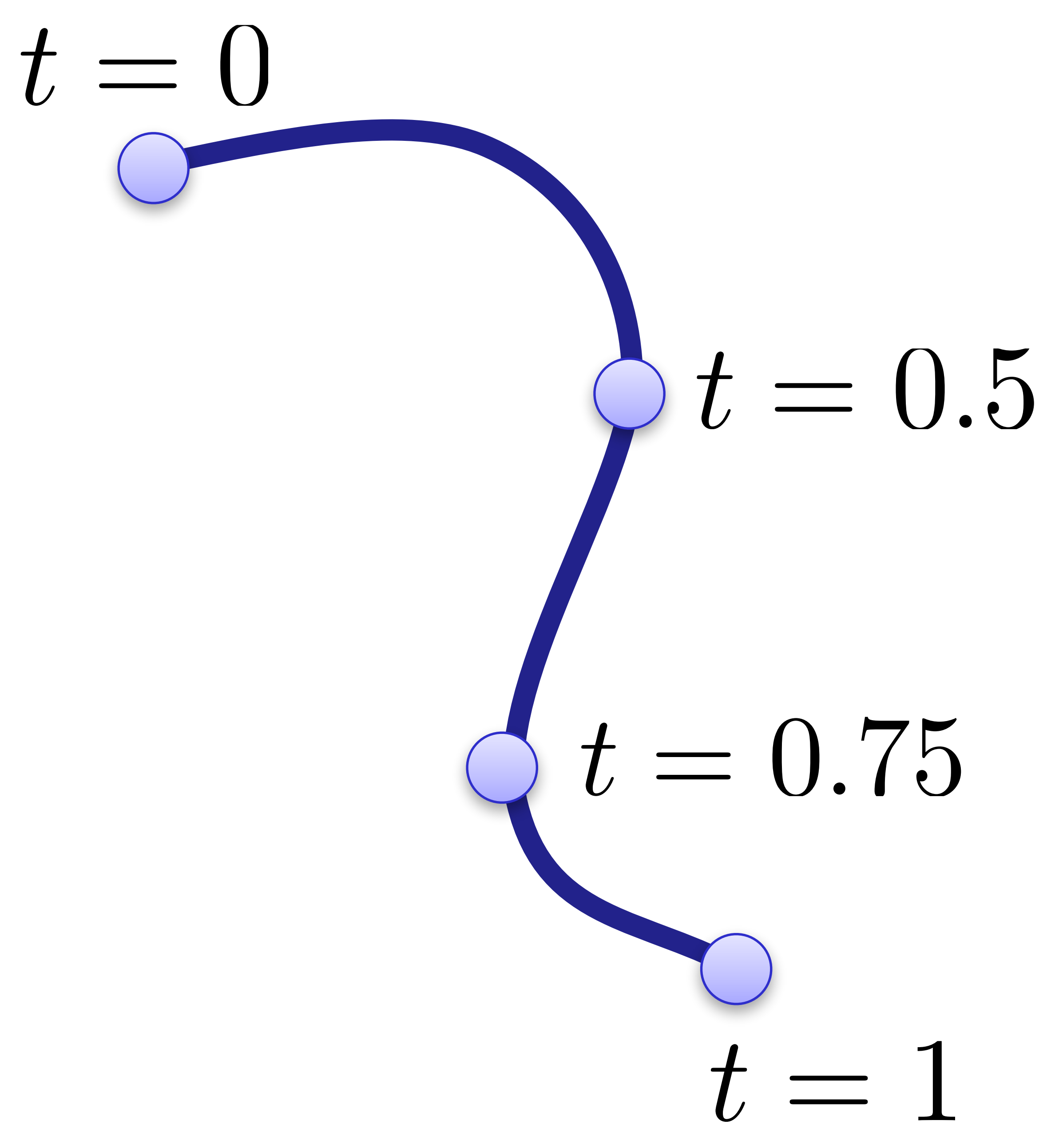
Parametric Representation

- Range of a function $f : X \to Y, X \subseteq \mathbb{R}^m, Y \subseteq \mathbb{R}^n$
  - Planar curve: $m = 1, n = 2$

$$s(t) = (x(t), y(t))$$

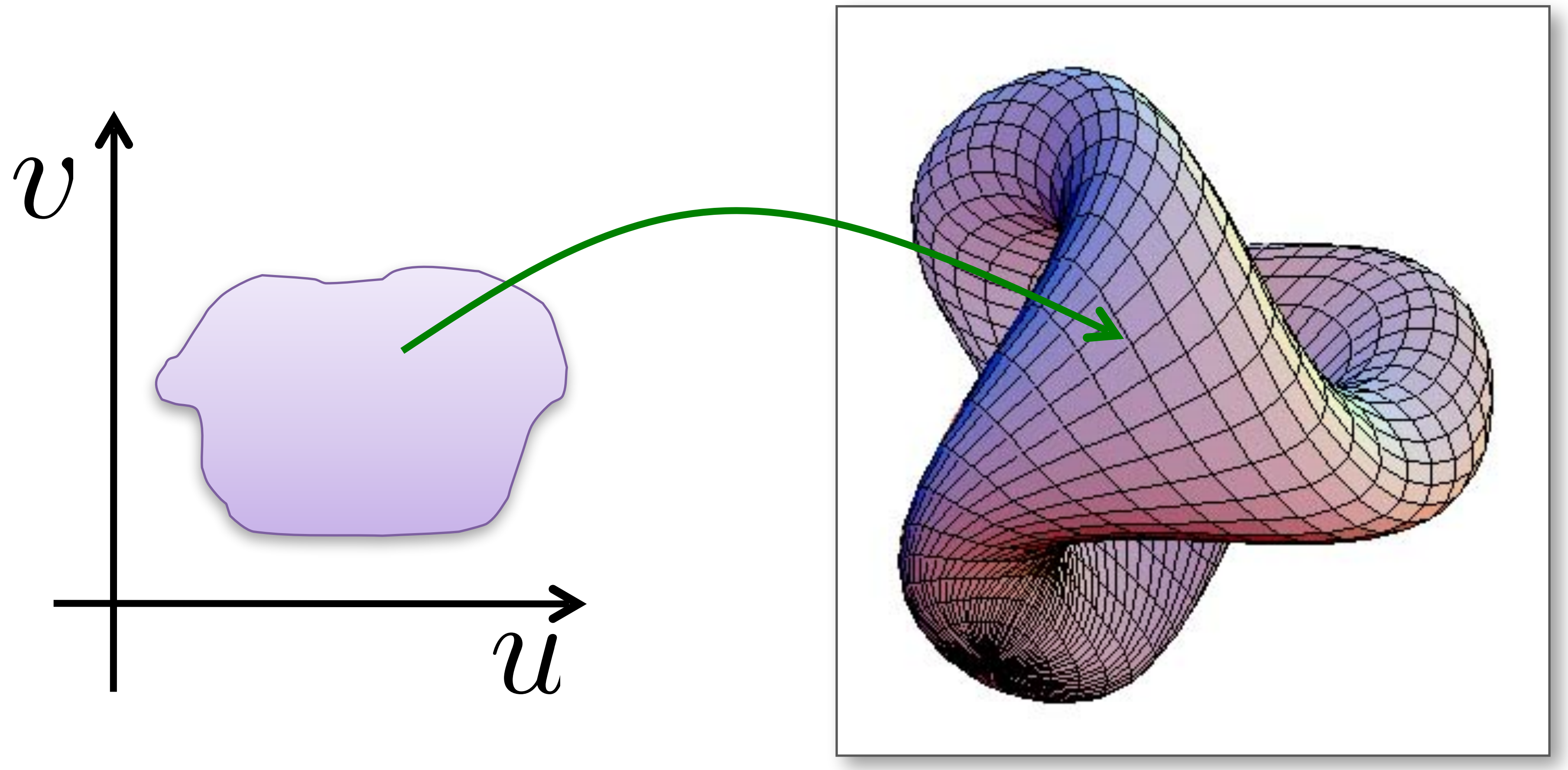
  - Space curve: $m = 1, n = 3$

$$s(t) = (x(t), y(t), z(t))$$



Parametric Representation

- Range of a function $f : X \to Y, X \subseteq \mathbb{R}^m, Y \subseteq \mathbb{R}^n$
  - Surface in 3D: $m = 2, n = 3$



$$s(u, v) = (x(u, v), y(u, v), z(u, v))$$

Parametric Curves

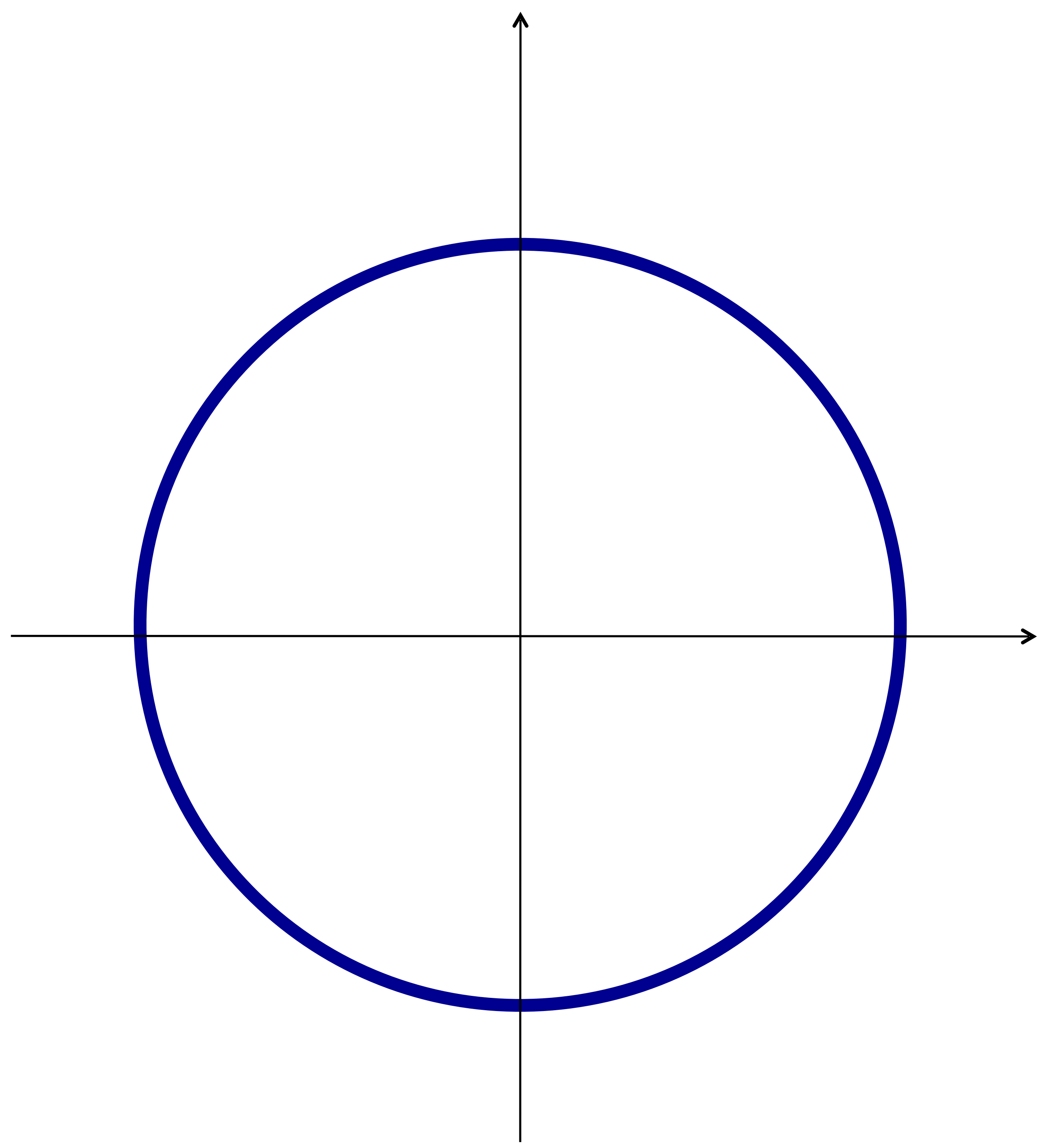
- Explicit curve/circle in 2D

$$\mathbf{p} : \mathbb{R} \to \mathbb{R}^2$$

$$t \mapsto \mathbf{p}(t) = (x(t), y(t))$$

$$\mathbf{p}(t) = r\,(\cos(t), \sin(t))$$

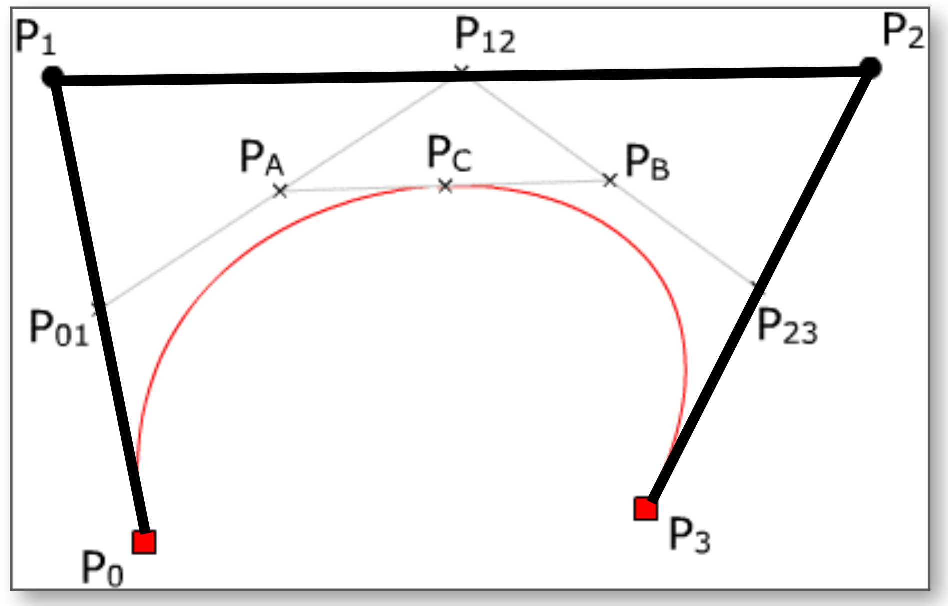
$$t \in [0, 2\pi)$$



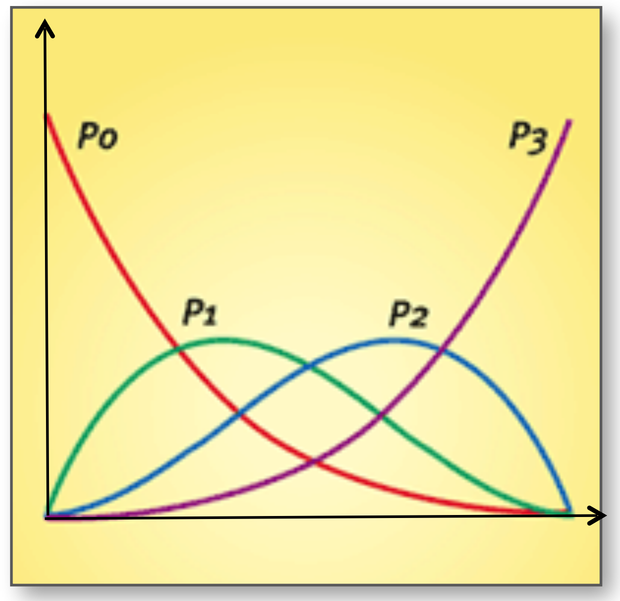
Parametric Curves

- Bezier curves, splines

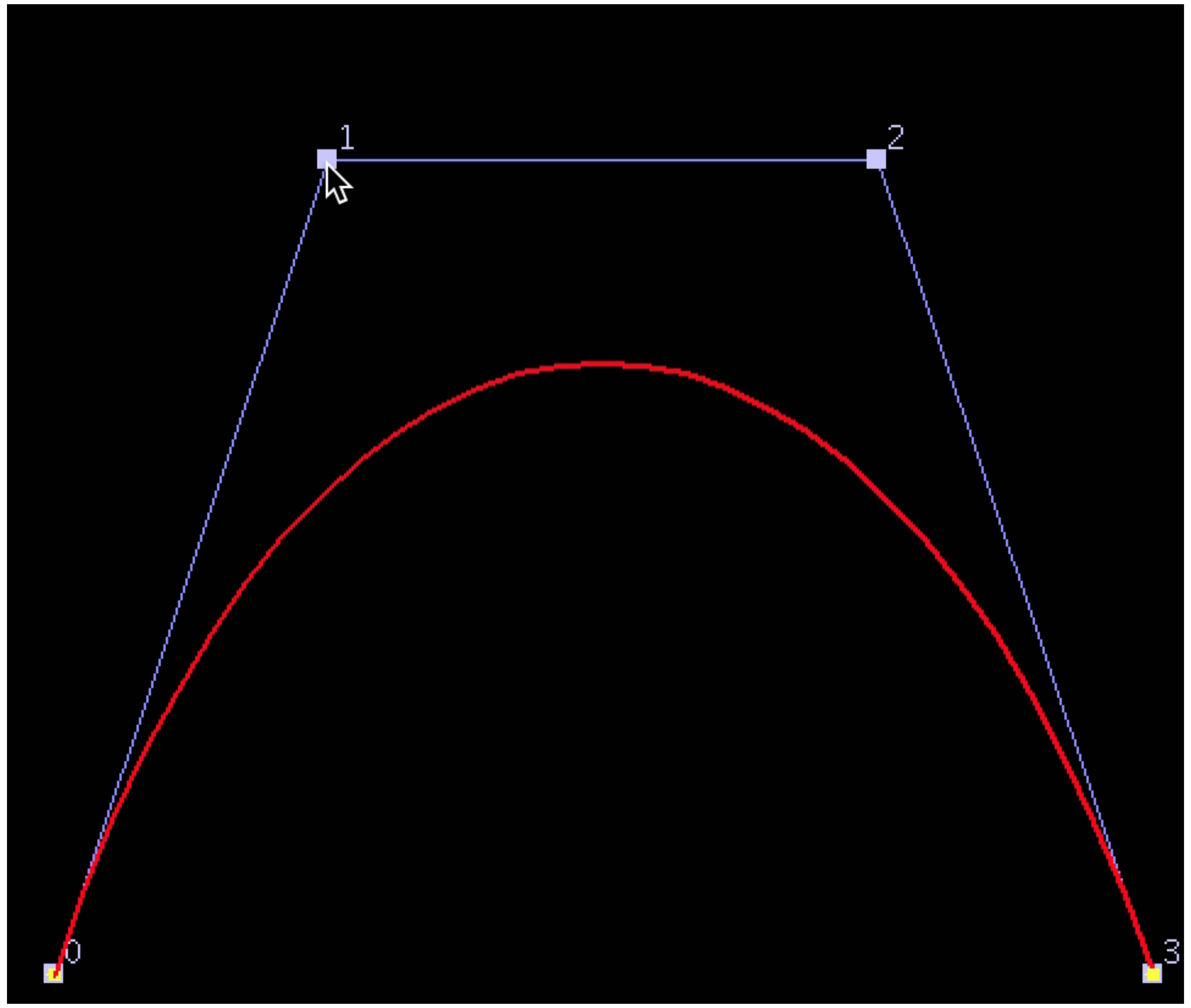
$$s(t) = \sum_{i=0}^{n} \mathbf{p}_i B_i^n(t) \qquad B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$



Curve and control polygon



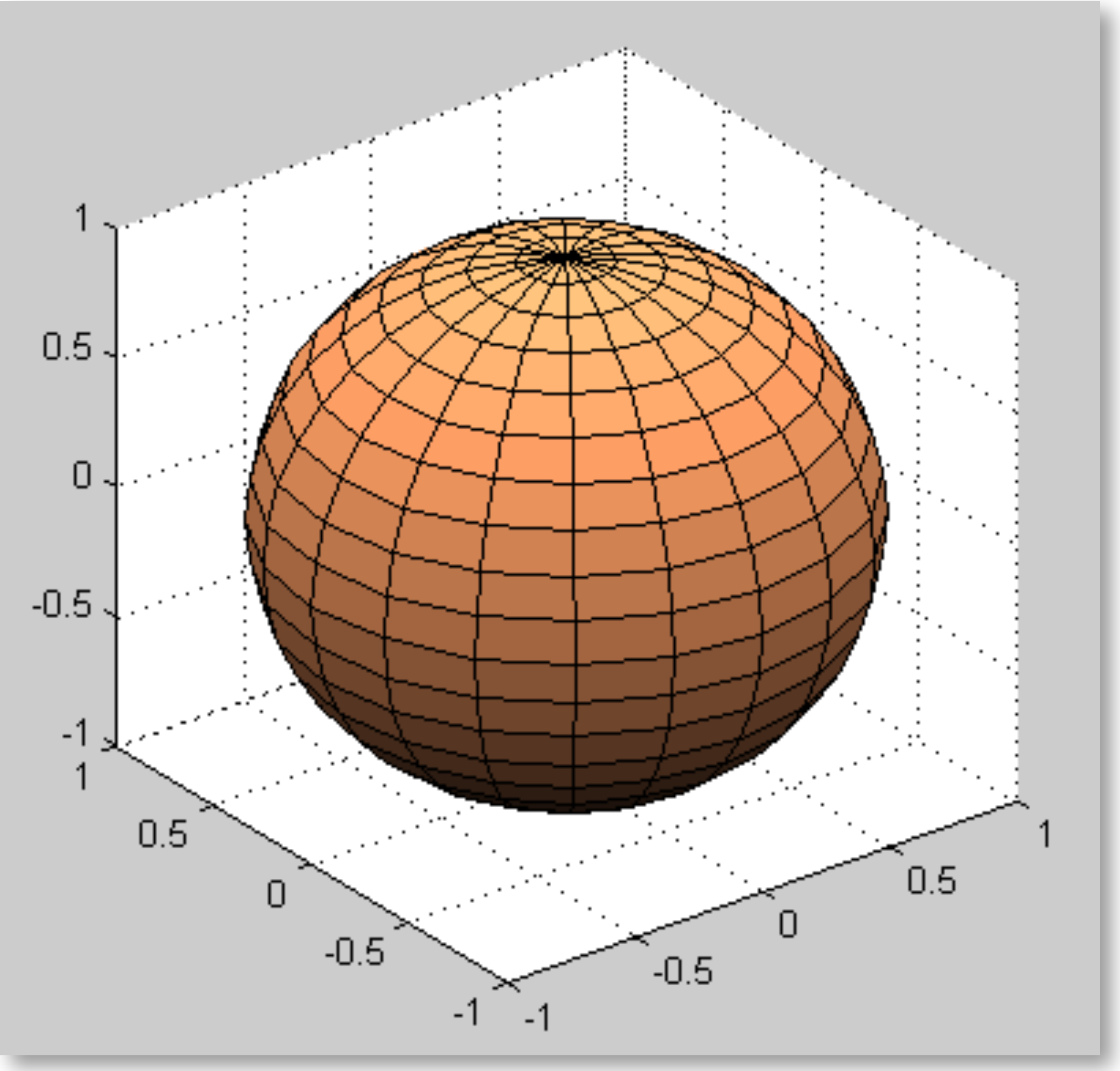
Basis functions



Parametric Surfaces

- Sphere in 3D

$$s : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$



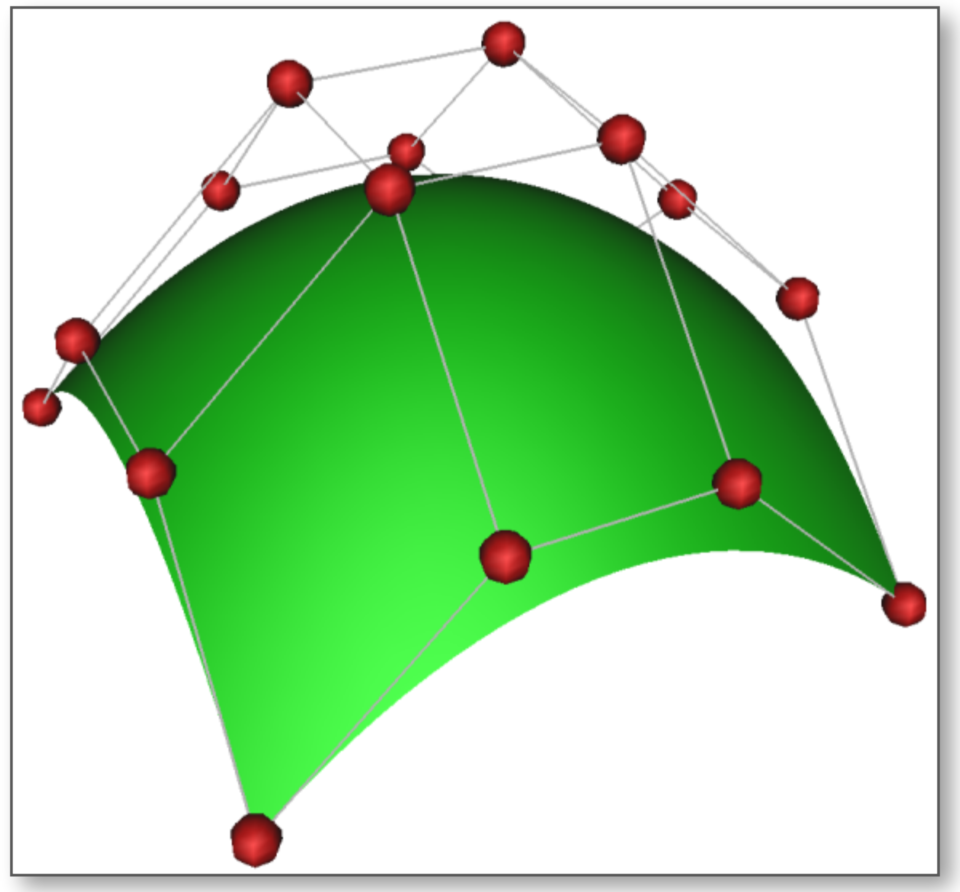
$$s(u,v) = r\left(\cos(u)\cos(v), \sin(u)\cos(v), \sin(v)\right)$$

$$(u,v) \in [0, 2\pi) \times [-\pi/2, \pi/2]$$

Parametric Surfaces

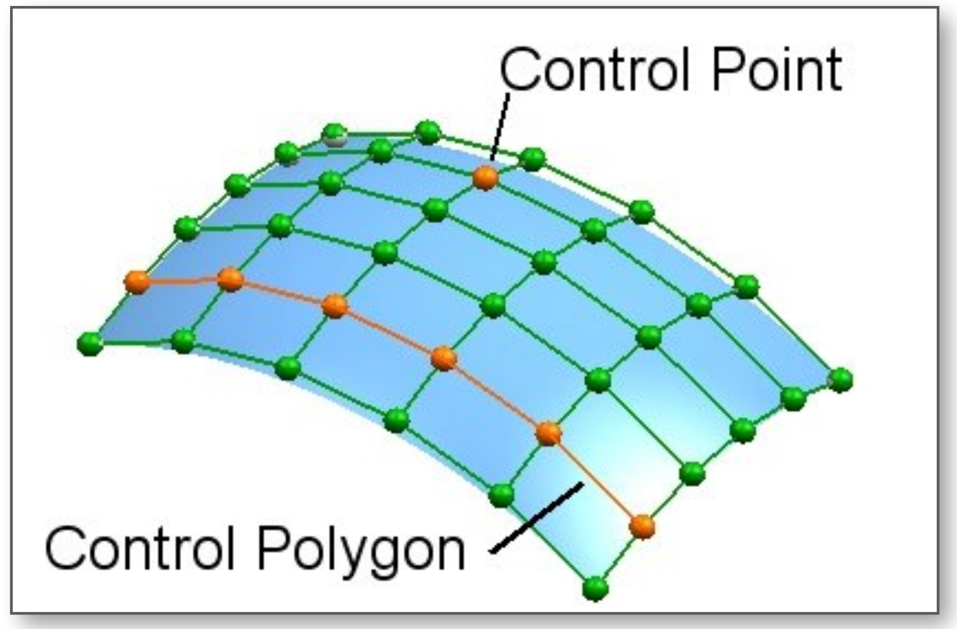
- Curve swept by another curve

$$s(u,v) = \sum_{i,j} \mathbf{p}_{i,j} B_i(u) B_j(v)$$



- Bezier surface:

$$s(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{p}_{i,j} B_i^m(u) B_j^n(v)$$

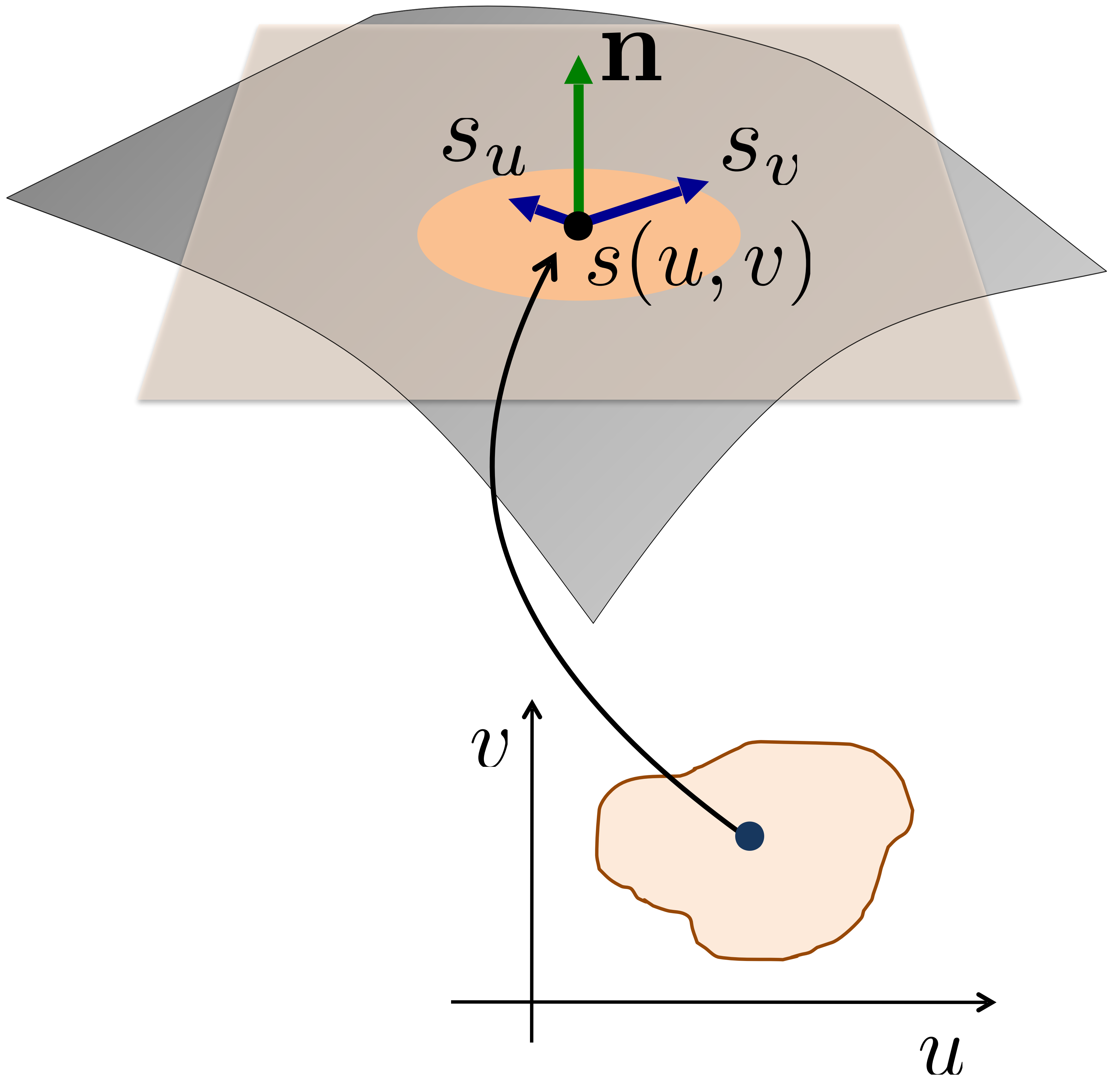


Tangents and Normal

$$s_u = \frac{\partial s(u,v)}{\partial u}$$

$$s_v = \frac{\partial s(u,v)}{\partial v}$$

$$\mathbf{n} = \frac{s_u \times s_v}{\|s_u \times s_v\|}$$



Tangent plane is normal to $\mathbf{n}$

Parametric Curves and Surfaces

- Advantages
  - Easy to generate points on the curve/surface
  - Separates x/y/z components

- Disadvantages
  - Hard to tell inside/outside
  - Hard to tell if a point is on the curve/surface
  - Hard to join patches at boundaries that match exactly

Implicit Curves and Surfaces

Implicit Curves and Surfaces

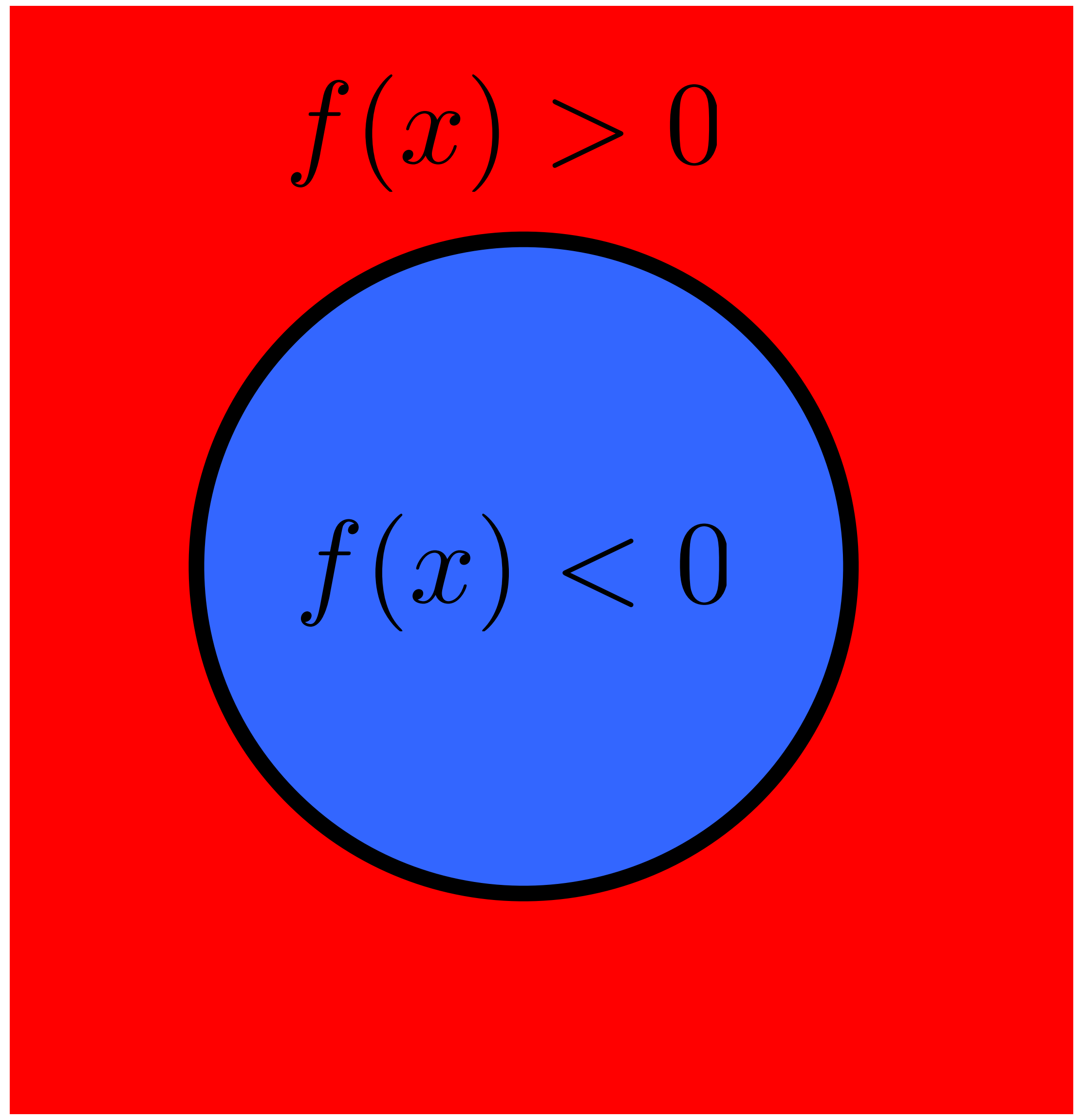
- Kernel of a scalar function $f : \mathbb{R}^m \rightarrow \mathbb{R}$
  - Curve in 2D: $S = \{x \in \mathbb{R}^2 | f(x) = 0\}$
  - Surface in 3D: $S = \{x \in \mathbb{R}^3 | f(x) = 0\}$

- Space partitioning

$\{x \in \mathbb{R}^m | f(x) > 0\}$ Outside

$\{x \in \mathbb{R}^m | f(x) = 0\}$ Curve/Surface

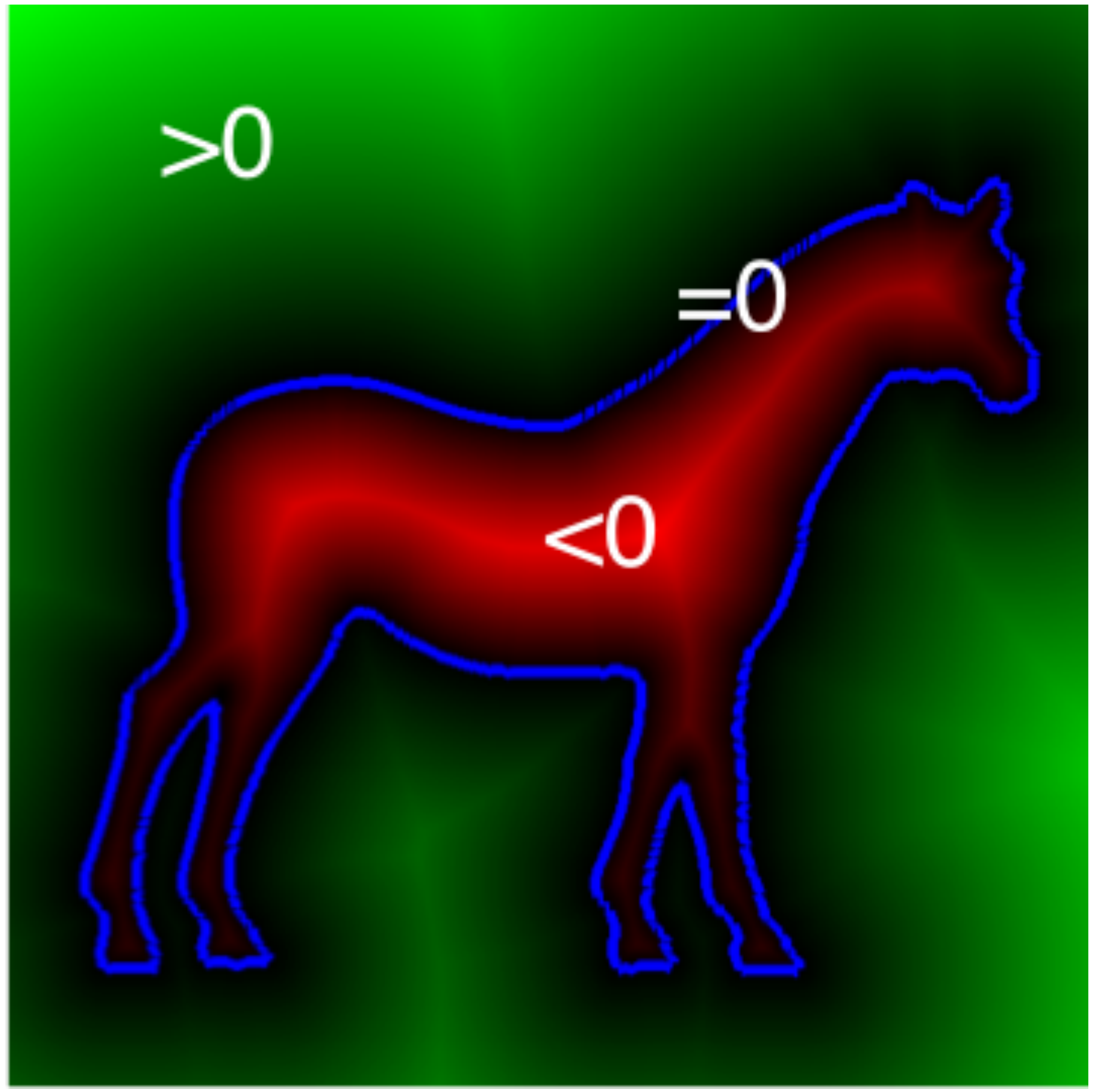
$\{x \in \mathbb{R}^m | f(x) < 0\}$ Inside



Implicit Curves and Surfaces

- Kernel of a scalar function $f : \mathbb{R}^m \rightarrow \mathbb{R}$
  - Curve in 2D: $S = \{x \in \mathbb{R}^2 | f(x) = 0\}$
  - Surface in 3D: $S = \{x \in \mathbb{R}^3 | f(x) = 0\}$

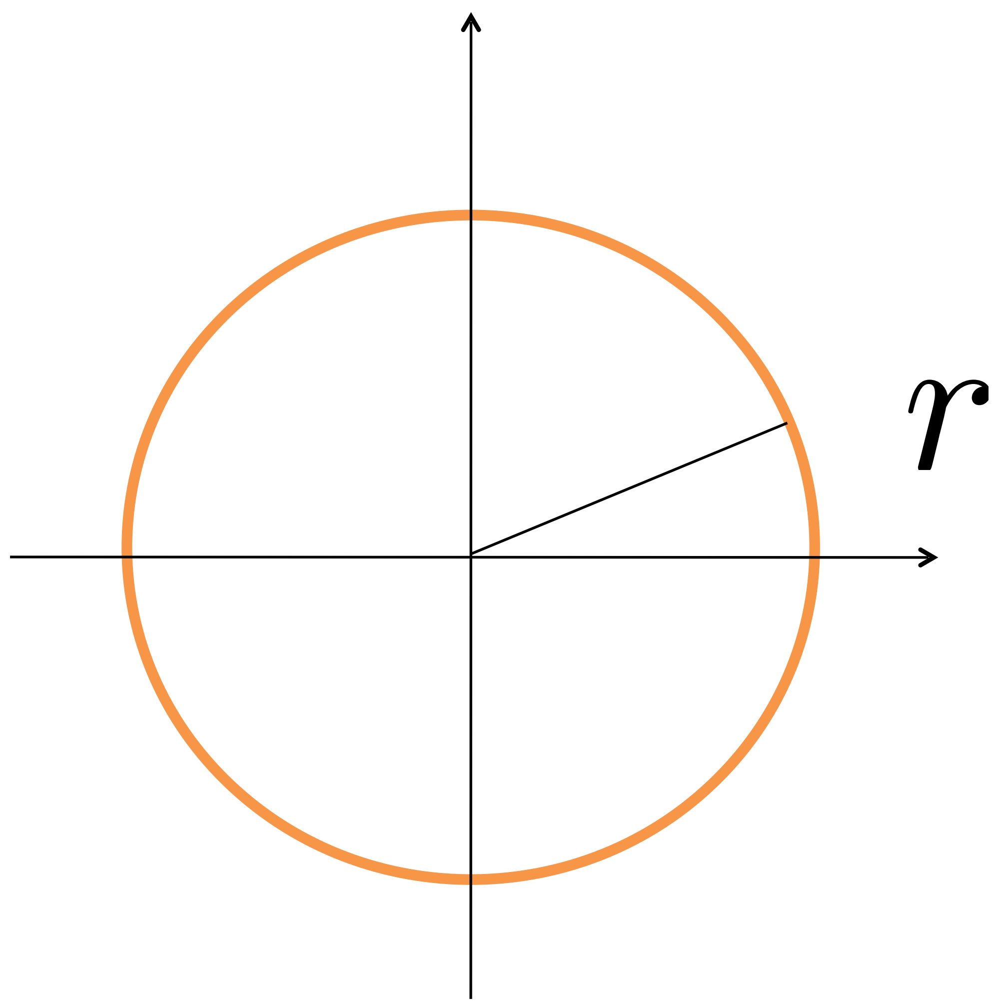
- Zero level set of signed distance function

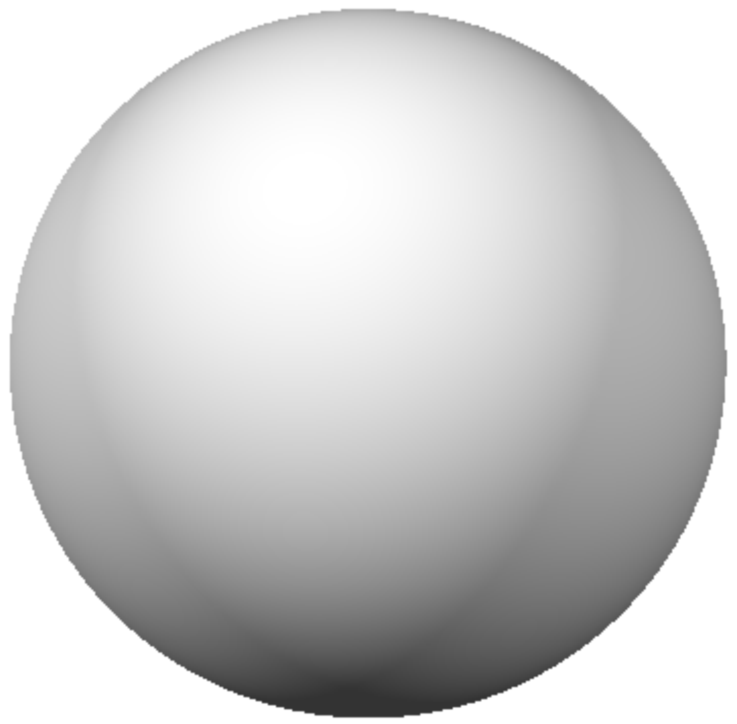


Implicit Curves and Surfaces

- Implicit circle and sphere

$$f(x,y) = x^2 + y^2 - r^2 \qquad f(x,y,z) = x^2 + y^2 + z^2 - r^2$$





Implicit Curves and Surfaces

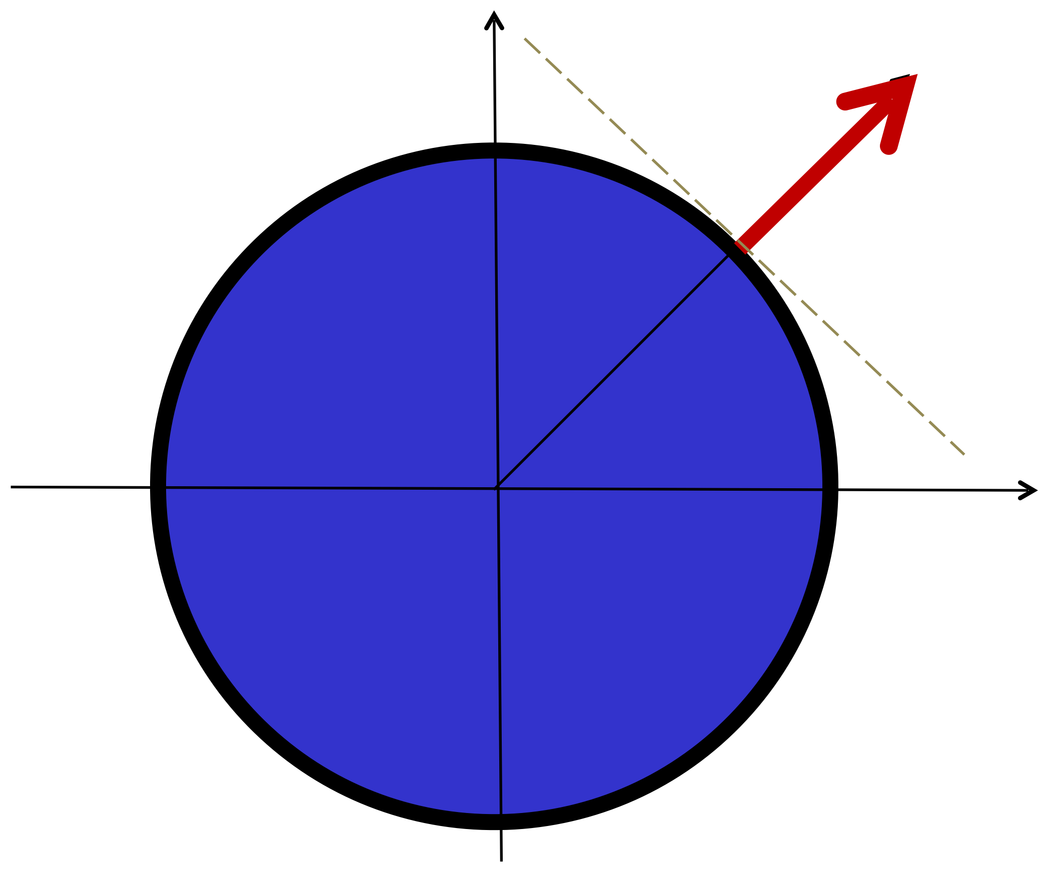
- The normal direction to the surface (curve) is given by the gradient of the implicit function

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

- Example

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

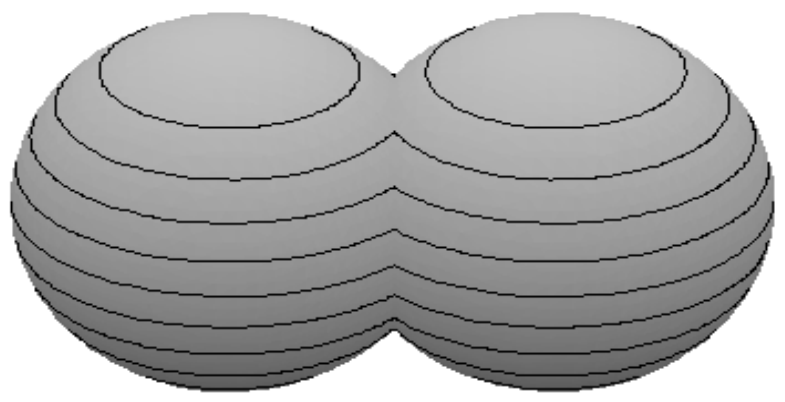
$$\nabla f(x, y, z) = (2x, 2y, 2z)^T$$

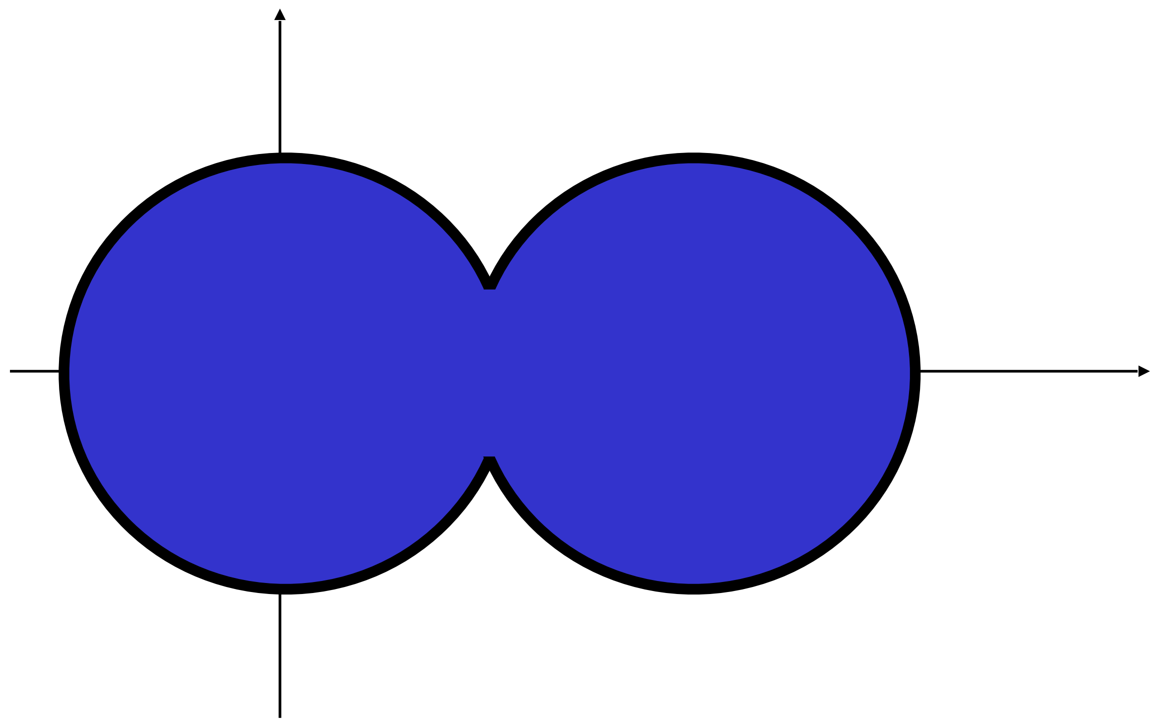


Boolean Set Operations

- Union:

$$\bigcup_i f_i(x) = \min f_i(x)$$





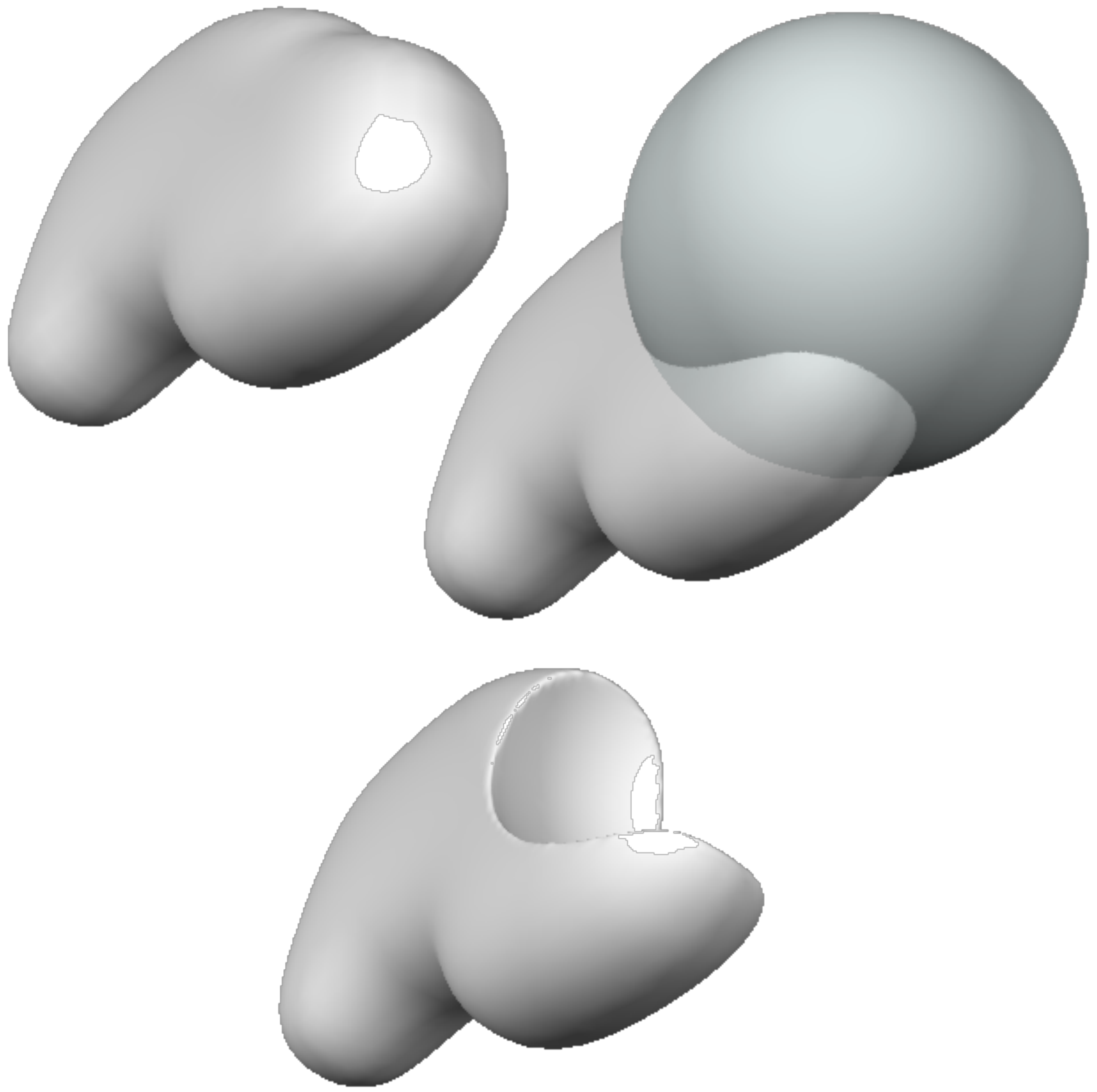
- Intersection:

$$\bigcap_i f_i(x) = \max f_i(x)$$

Boolean Set Operations

- Positive = outside, negative = inside
- Boolean subtraction: $h = max(f, -g)$

	$f > 0$	$f < 0$
$g > 0$	$h > 0$	$h < 0$
$g < 0$	$h > 0$	$h > 0$



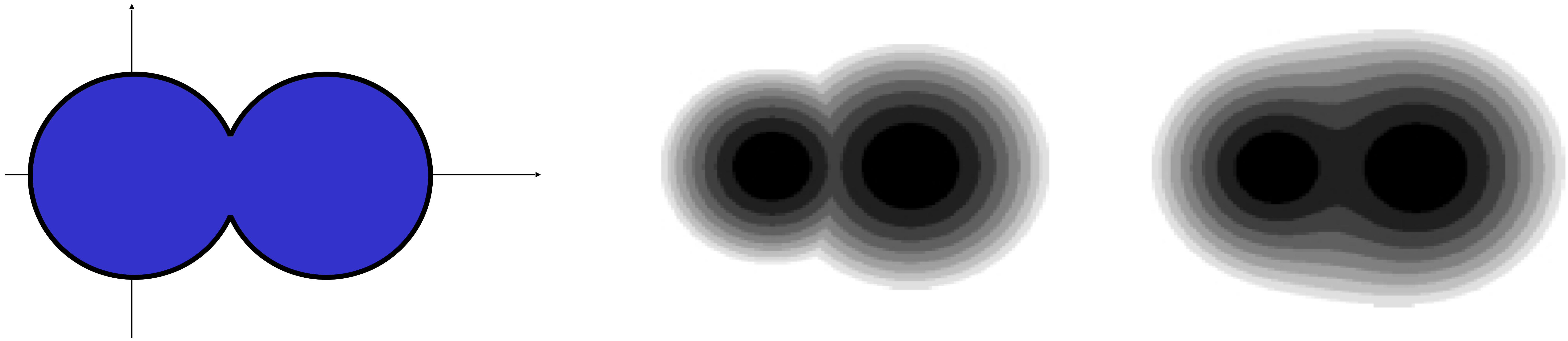
- Much easier than for parametric surfaces!

Smooth Set Operations

- In many cases, smooth blending is desired
  - [Pasko and Savchenko, Blending operations for the functionally based constructive geometry 1994]

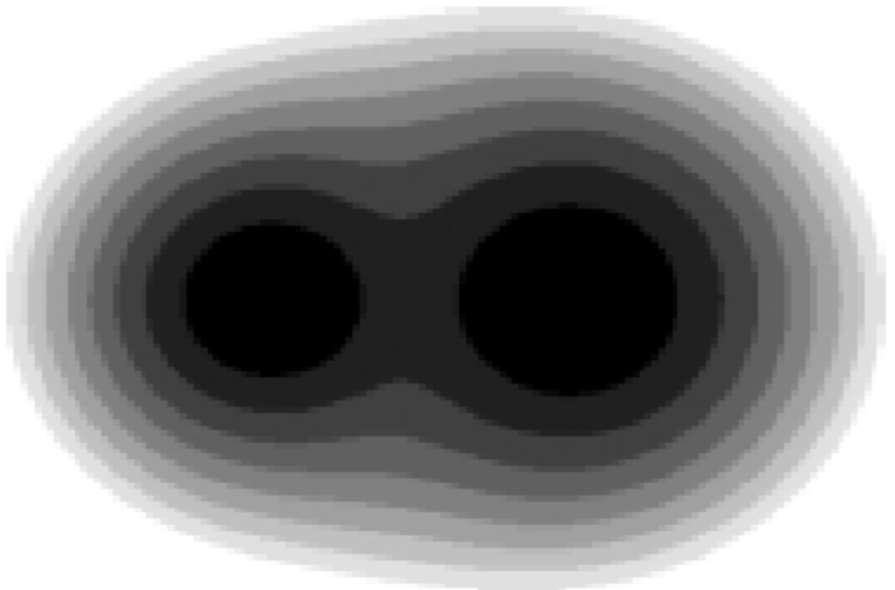
$$f \cup g = \tfrac{1}{1+\alpha}\left(f + g - \sqrt{f^2 + g^2 - 2\alpha f g}\right)$$

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

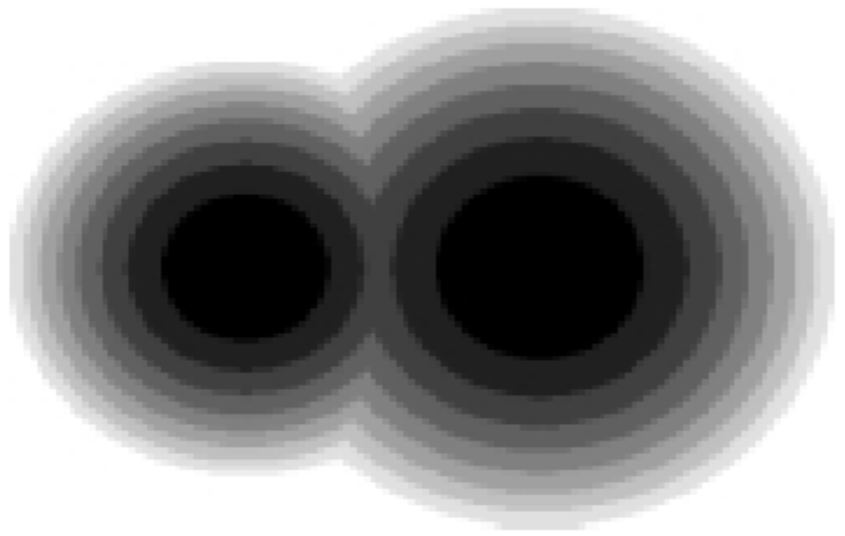


Smooth Set Operations

- Examples



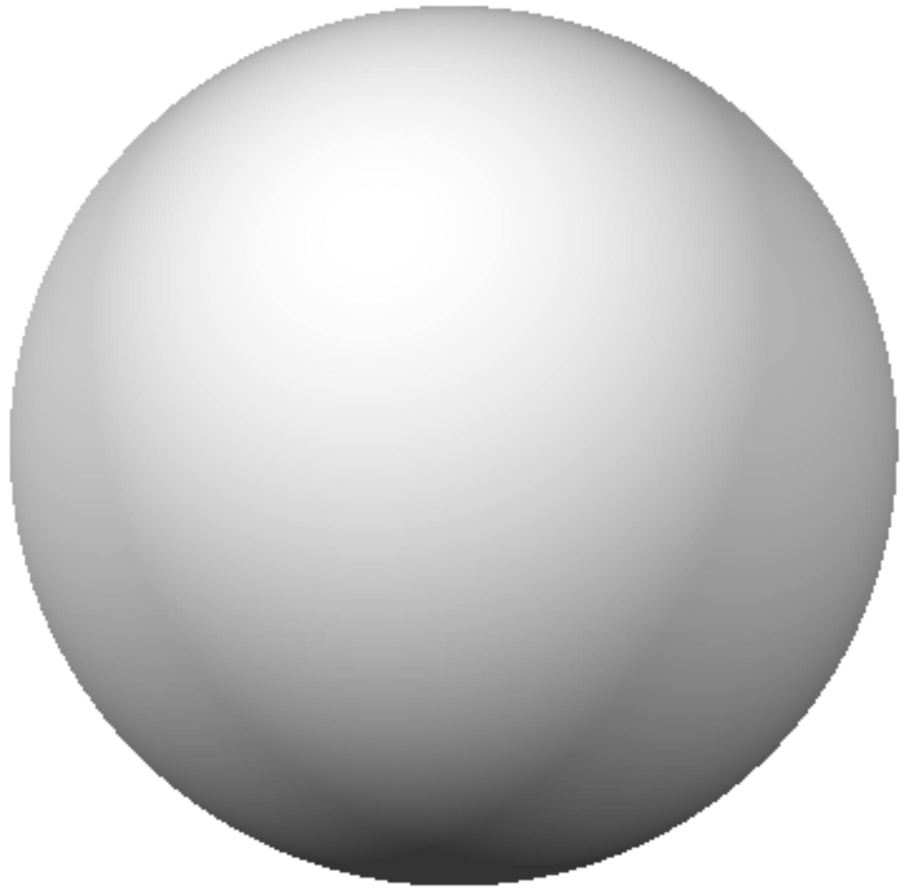
$\alpha = 0$



$\alpha = 1$

Designing with Implicit Surfaces

- Zero set (or level set) of a function:



$$f(\mathbf{p}) = \|\mathbf{p}\|^2 - r^2$$

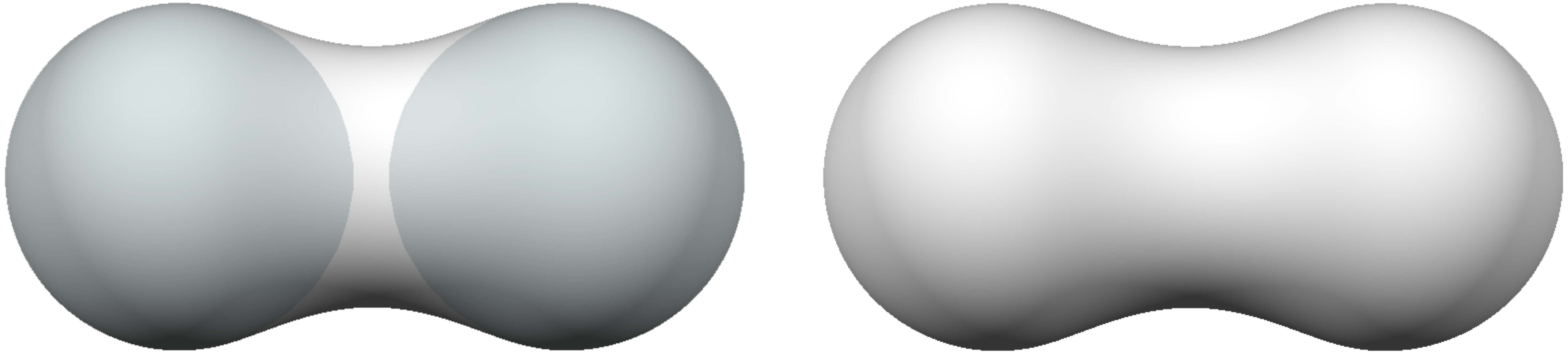
- But also a level set at value e^{-1} of this function:

$$f(\mathbf{p}) = e^{-\|\mathbf{p}\|^2/r^2} \text{ at } e^{-1}$$

Designing with Implicit Surfaces

- With smooth falloff functions, adding implicit functions generates a blend:

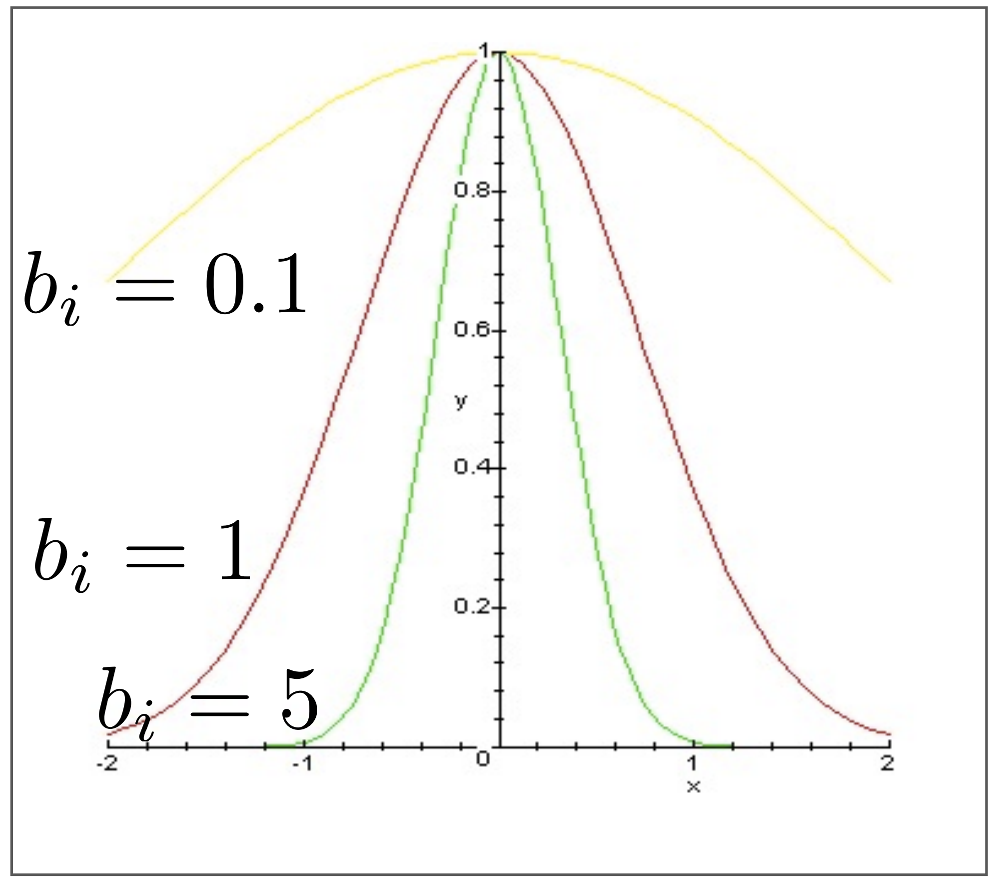
$$f(\mathbf{p}) = e^{-\|\mathbf{p}-\mathbf{p}_1\|^2} + e^{-\|\mathbf{p}-\mathbf{p}_2\|^2}$$

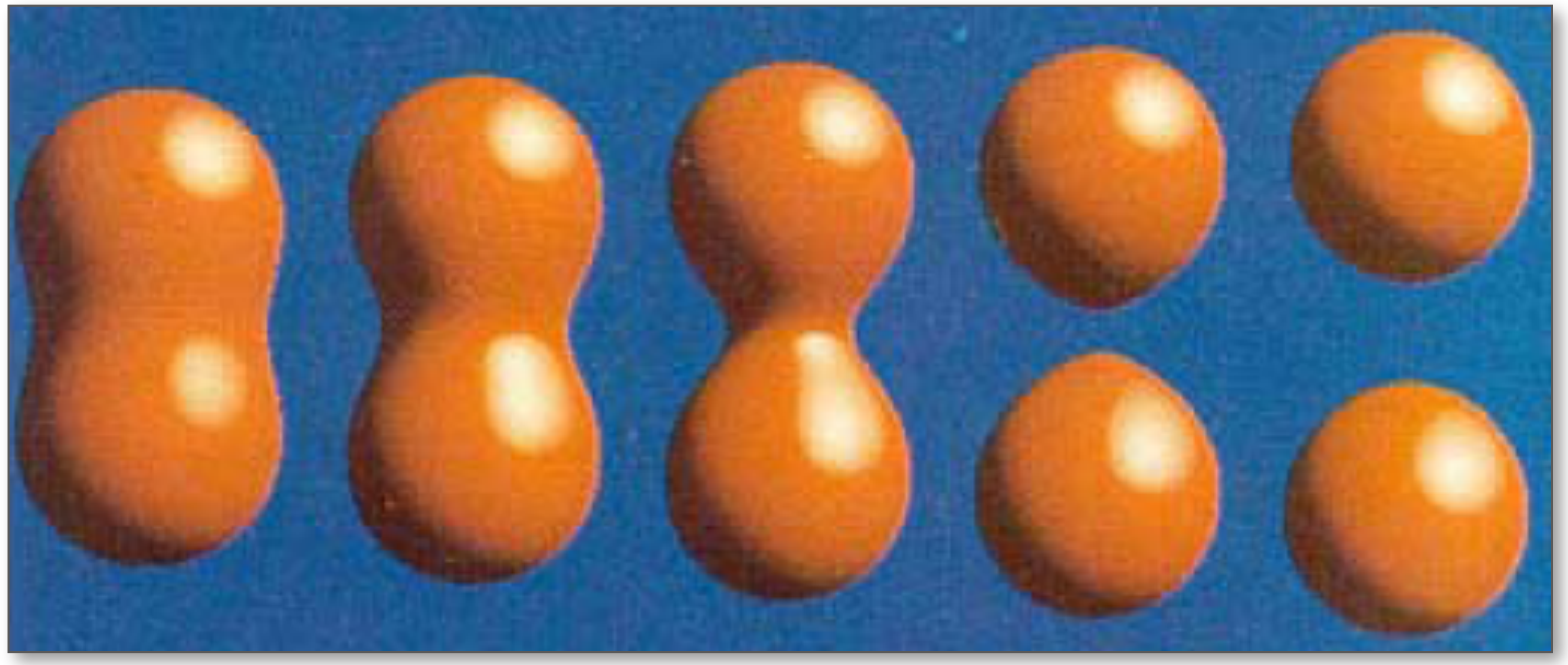


- Called “Metaballs” or “Blobs”

Blobs

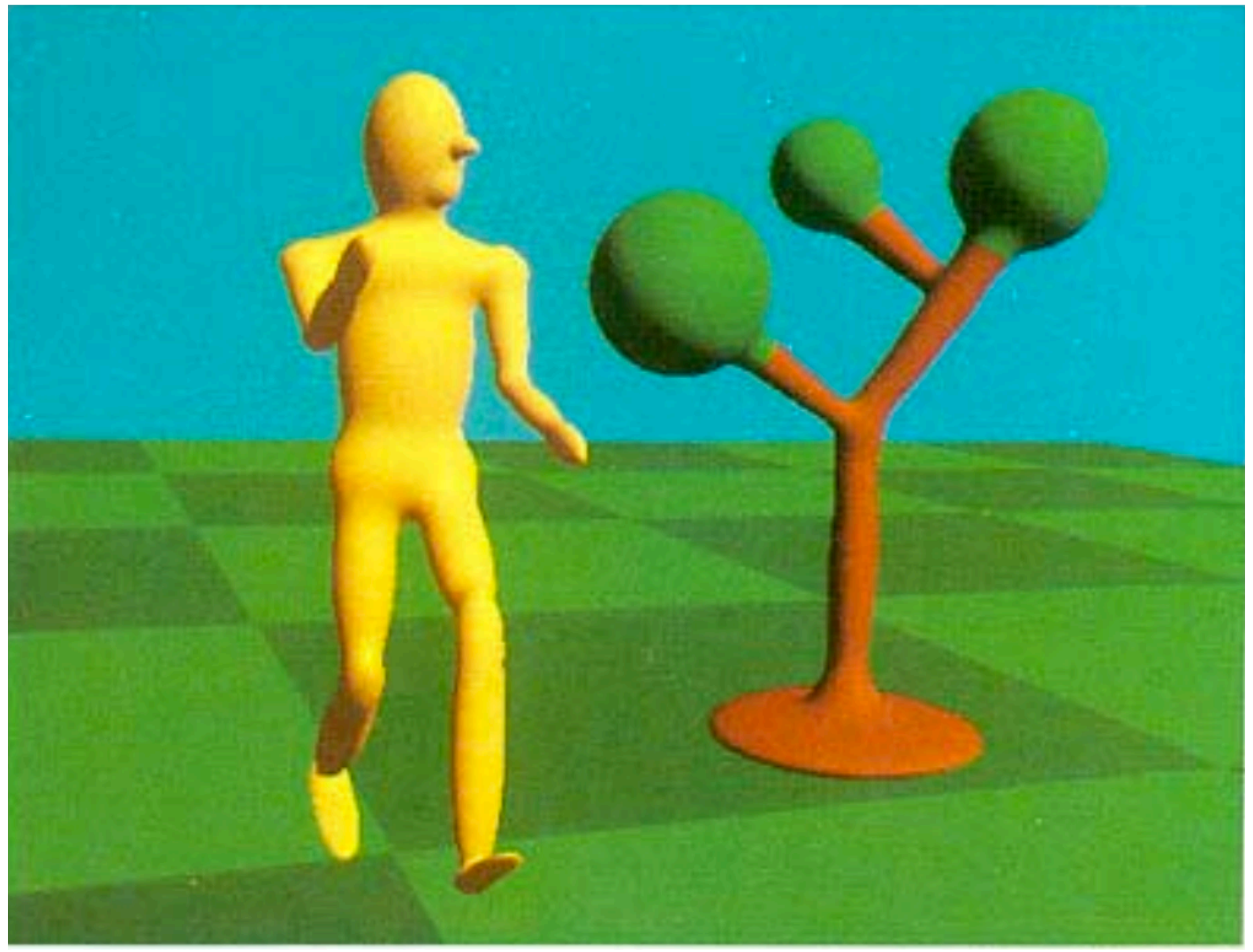
- Suggested by Blinn [1982]
  - Defined implicitly by a potential function around a point $\mathbf{p}_i$:
$$f(\mathbf{p}) = a_i e^{-b_i \|\mathbf{p}-\mathbf{p}_i\|^2}$$
  - Set operations by simple addition/subtraction



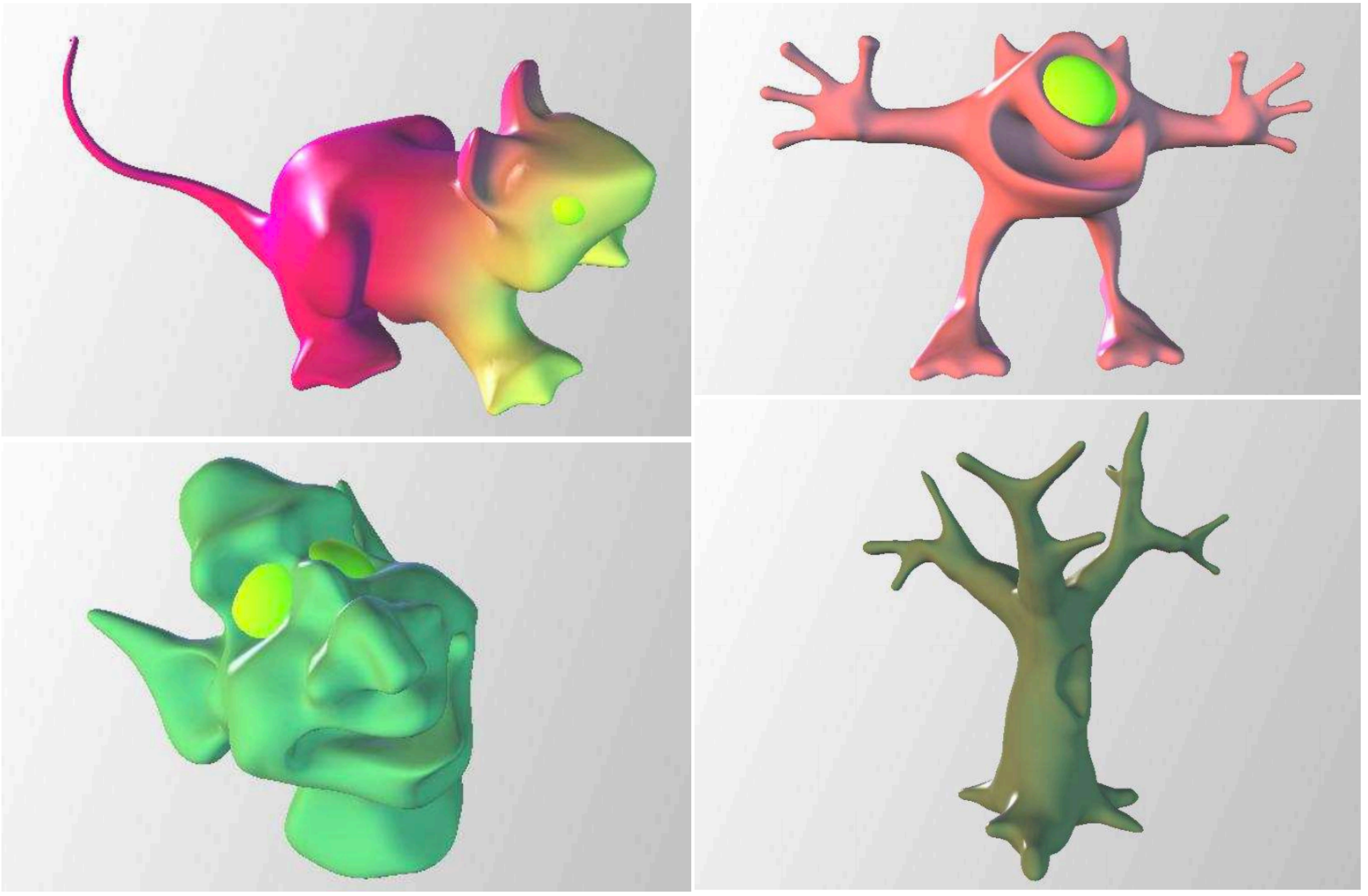


J. Blinn, "A Generalization of Algebraic Surface Drawing", ACM Transactions on Graphics, Vol. 1, No. 3, pp. 235-256, July, 1982.

Blobs



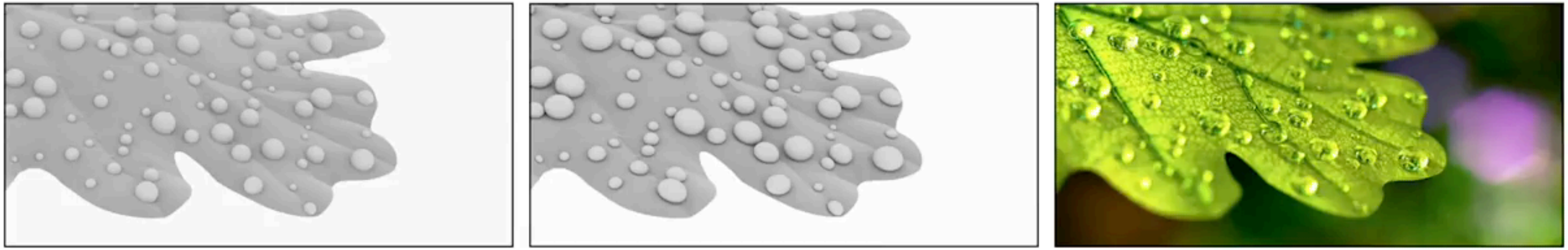
J. Blinn, "A Generalization of Algebraic Surface Drawing", ACM Transactions on Graphics, Vol. 1, No. 3, pp. 235-256, July, 1982.



Angelidis et al., “Swirling-Sweepers: Constant-Volume Modeling”, Pacific Graphics 2004

Sketch-Based Implicit Blending

Baptiste Angles[1,2], Marco Tarini[3], Brian Wyvill[1], Loïc Barthe[2], Andrea Tagliasacchi[1]

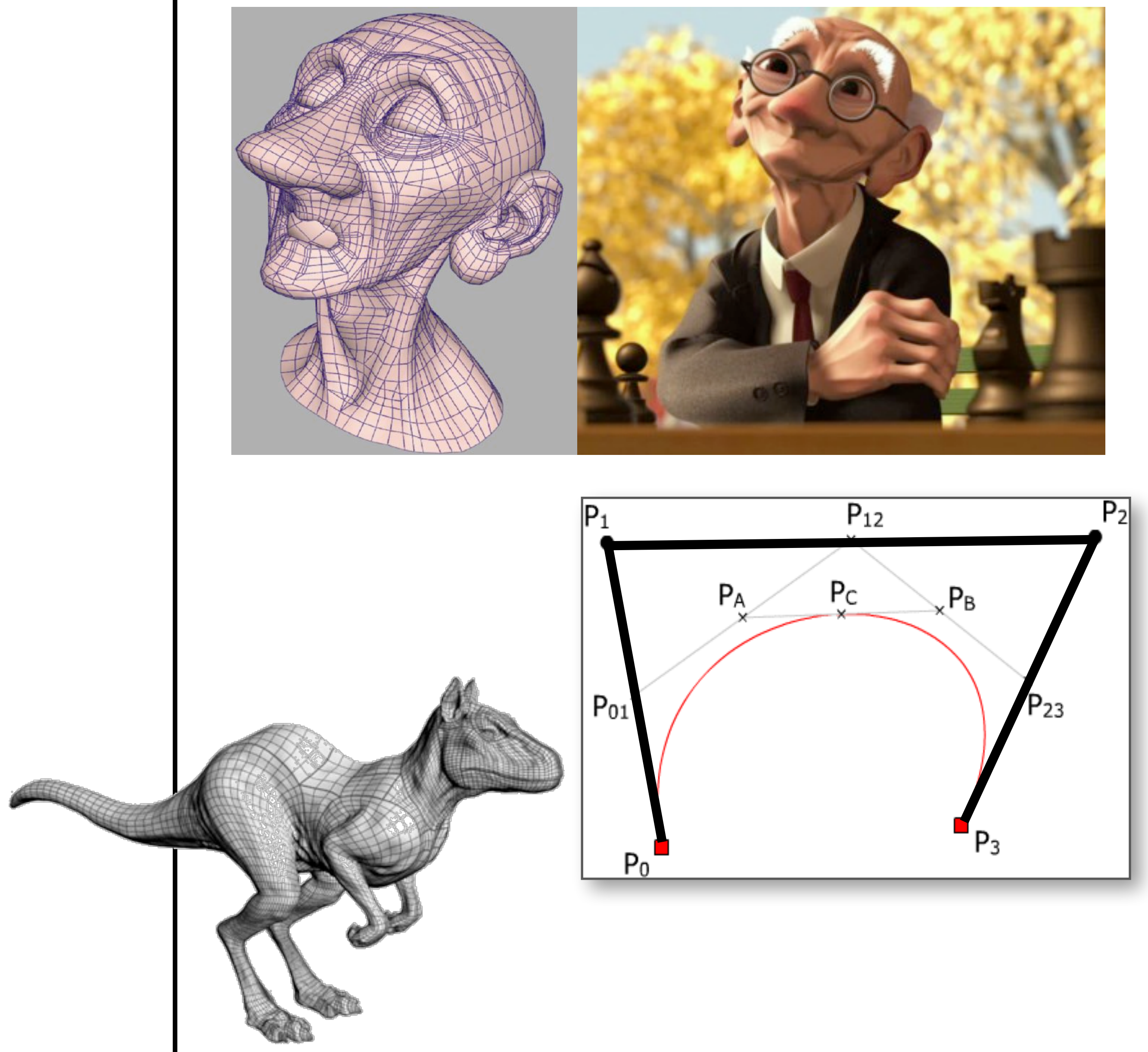
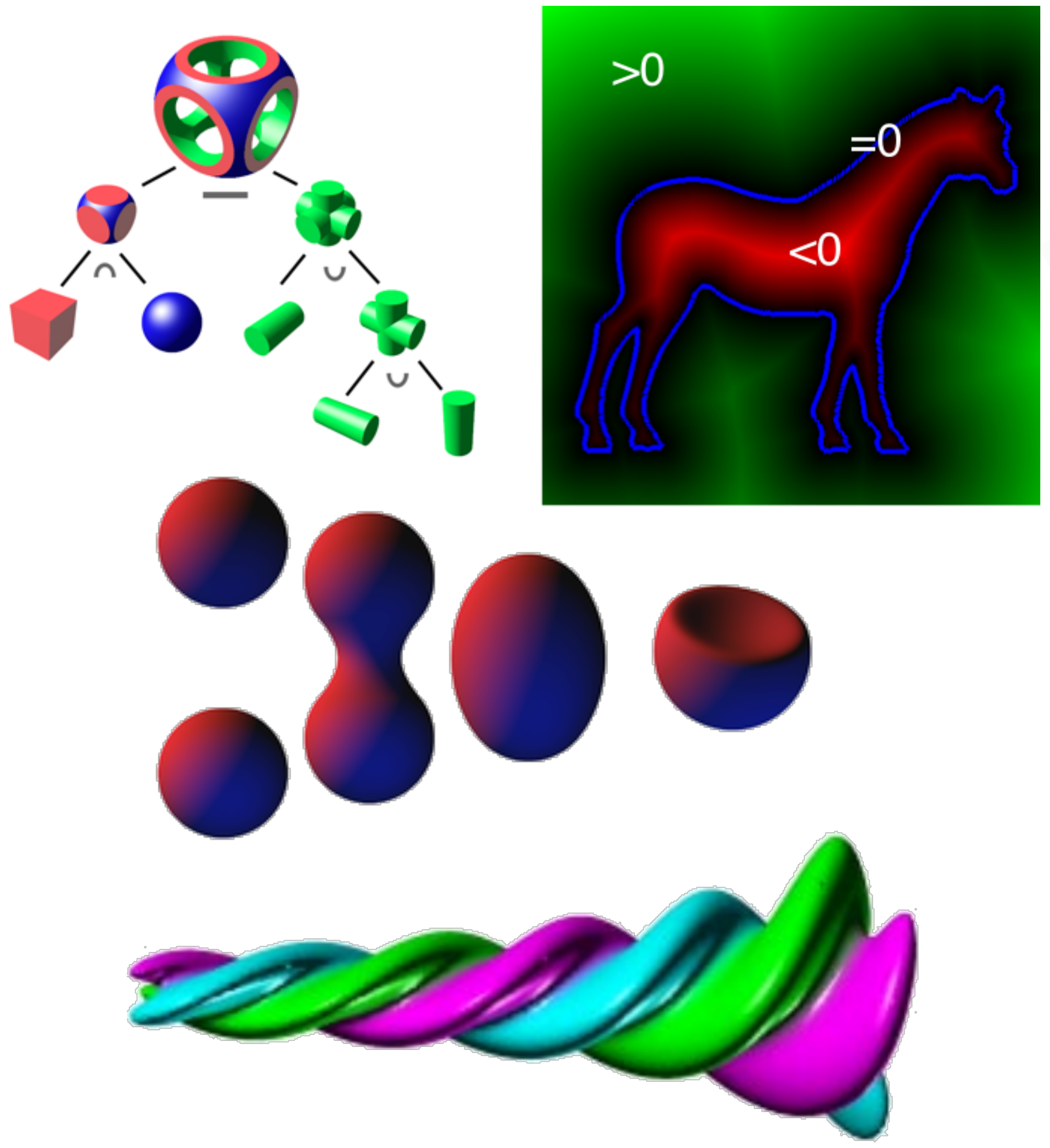
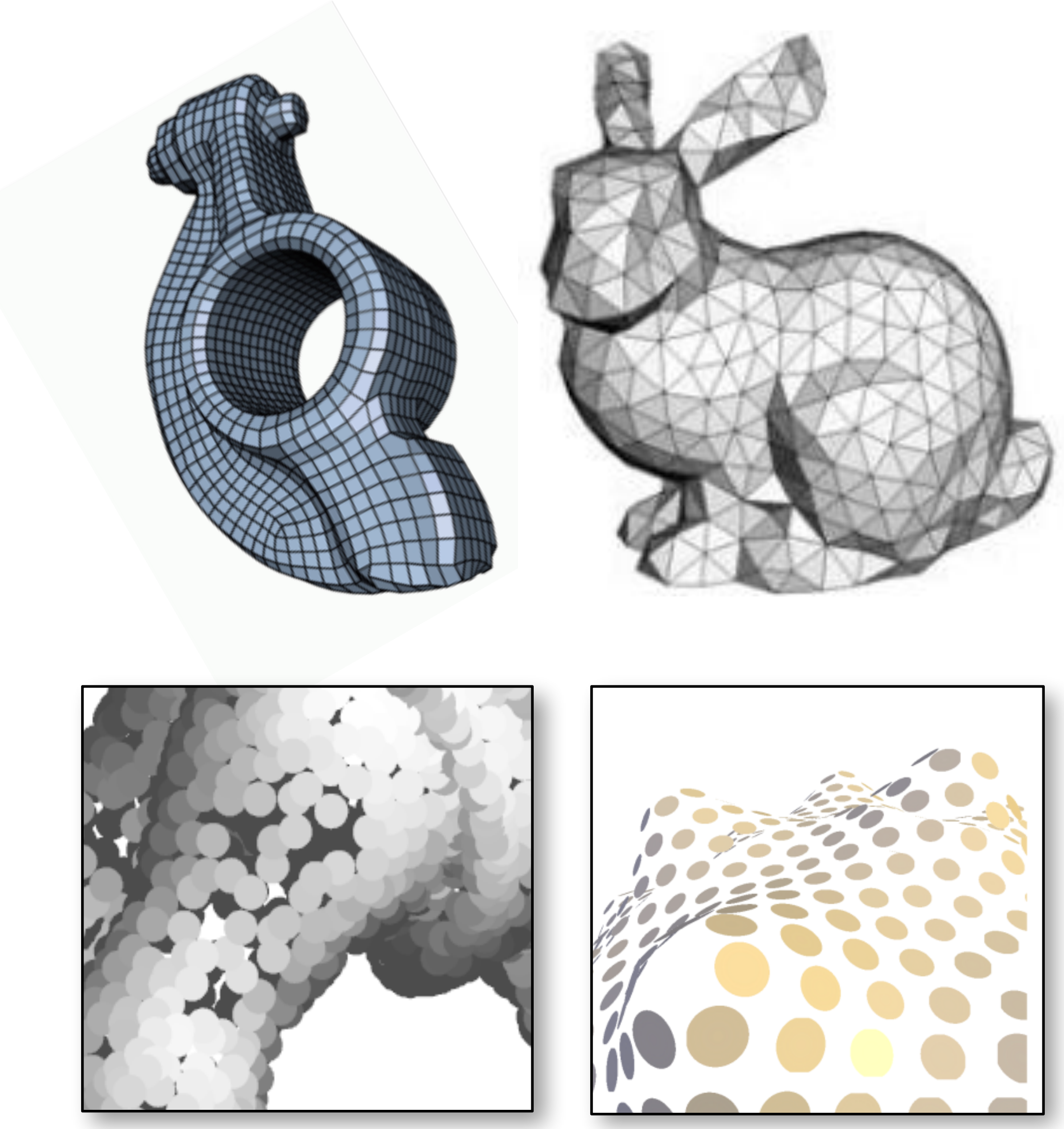


1. University of Victoria
2. Université de Toulouse, IRIT/CNRS
3. Università dell'Insubria, ISTI / CNR

Implicit Curves and Surfaces

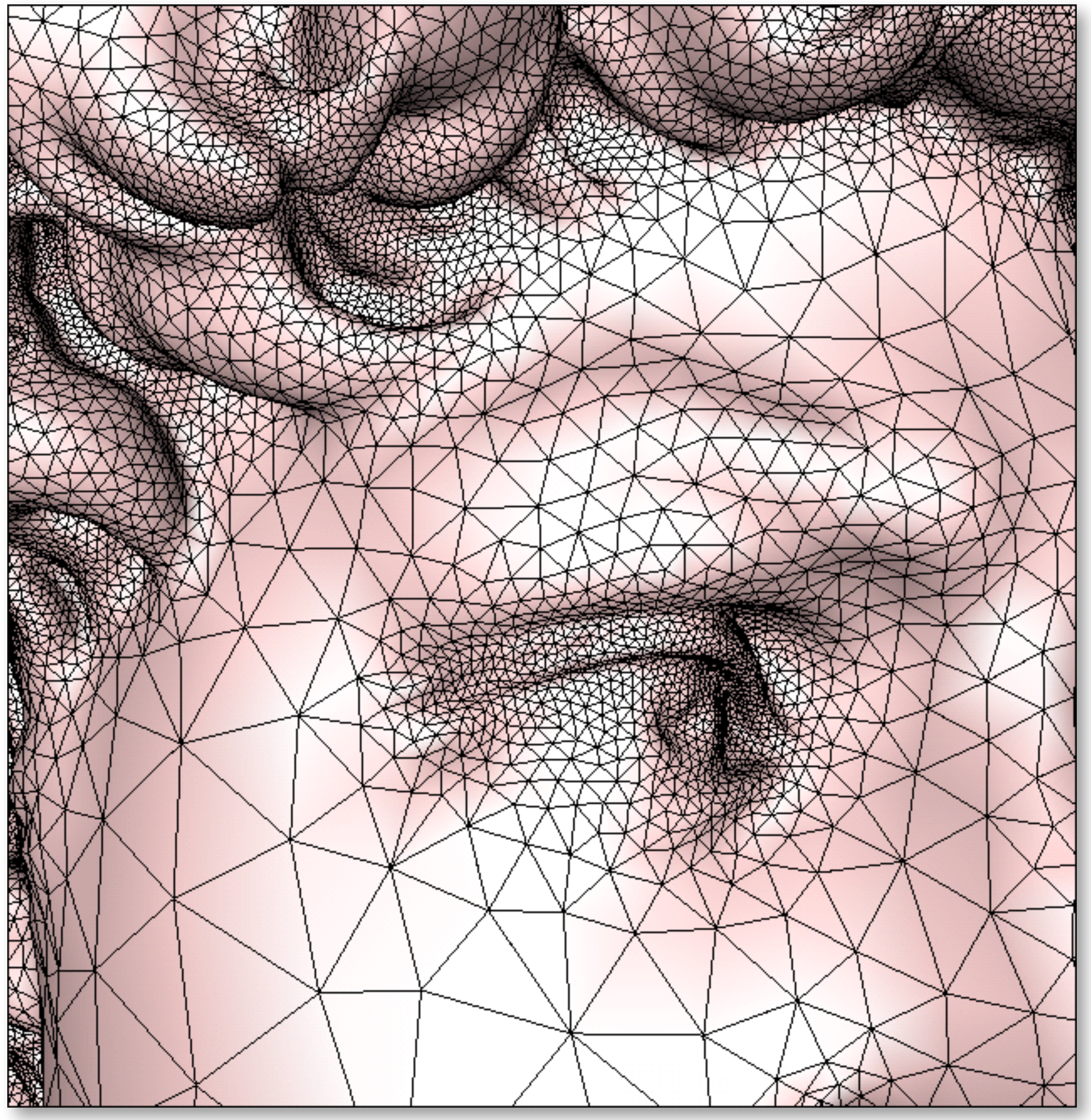
- Advantages
  - Easy to determine inside/outside
  - Easy to determine if a point is **on** the curve/surface
- Disadvantages
  - Hard to generate points on the curve/surface
  - Does not lend itself to (real-time) rendering

Summary

<table>
<tr><th>Parametric</th><th>Implicit</th><th>Discrete/Sampled</th></tr>
<tr>
<td>
</td>
<td>
</td>
<td></td>
</tr>
<tr>
<td>
Splines, tensor-product surfaces
Subdivision surfaces
</td>
<td>
Metaballs/blobs
Distance fields
CSG
</td>
<td>
Meshes
Point set surfaces
</td>
</tr>
</table>

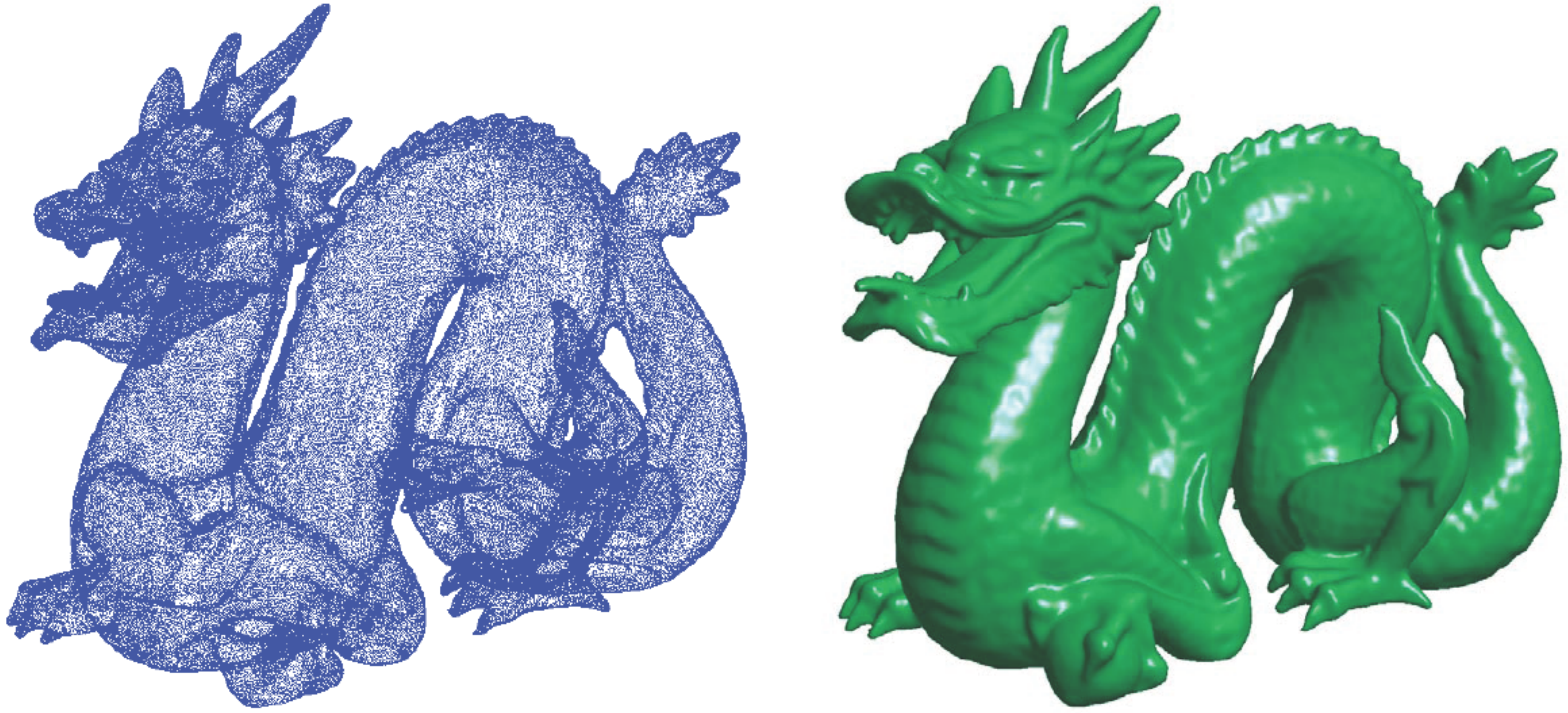
In the Next Lectures

- All About Meshes



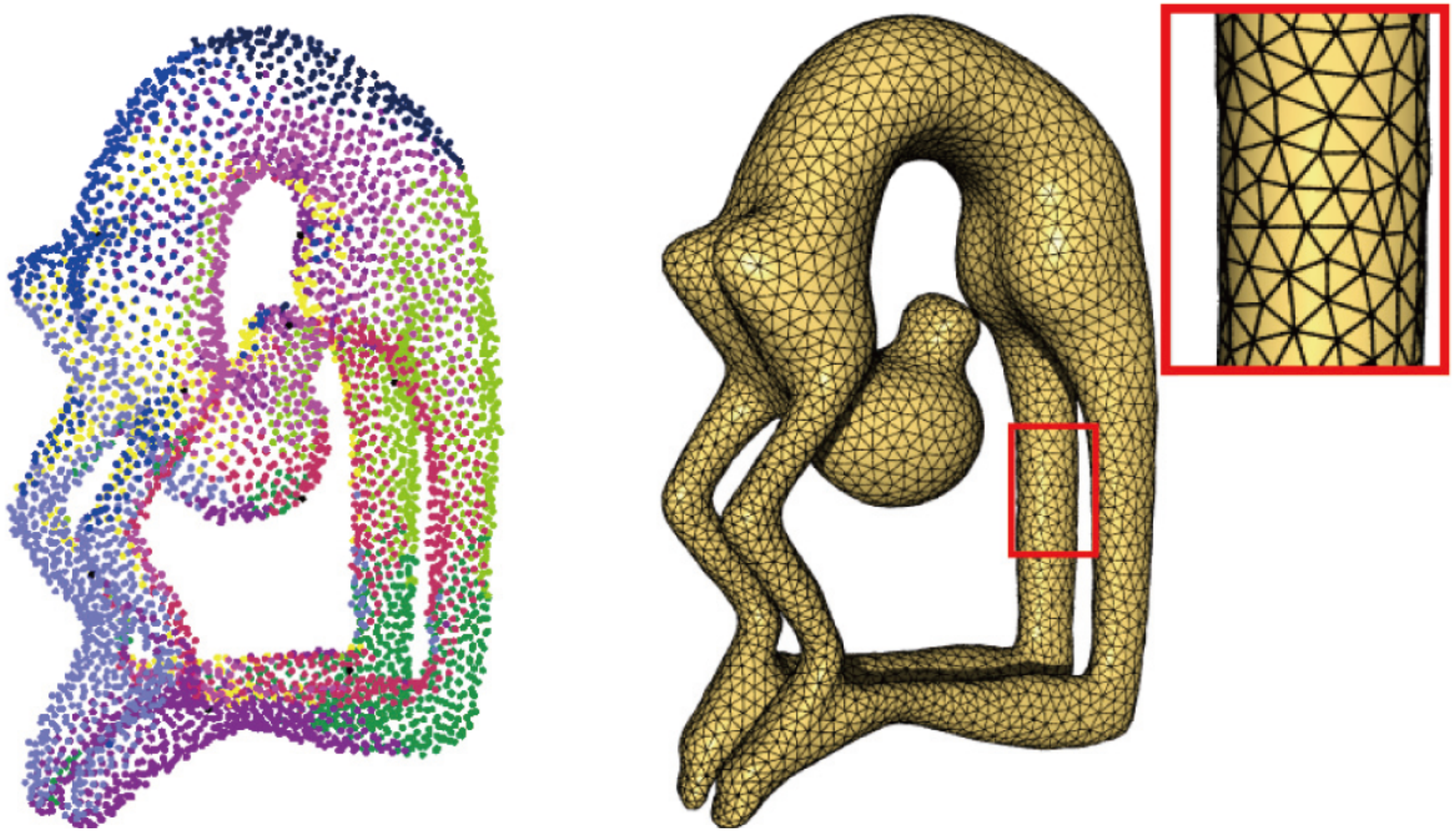
In the Next Lectures

- From points to surfaces:
  - a point is in fact a differential of surface
  - what can we say about it?
  - how can we fill the gaps?



In the Next Lectures

- How to get a nice, watertight surface mesh from a sampled point set



- The most popular way:
  points➔implicit function➔surface mesh

Thank you