Computer graphics is not...

...only about modeling and image synthesis

How about acquiring geometry automatically?

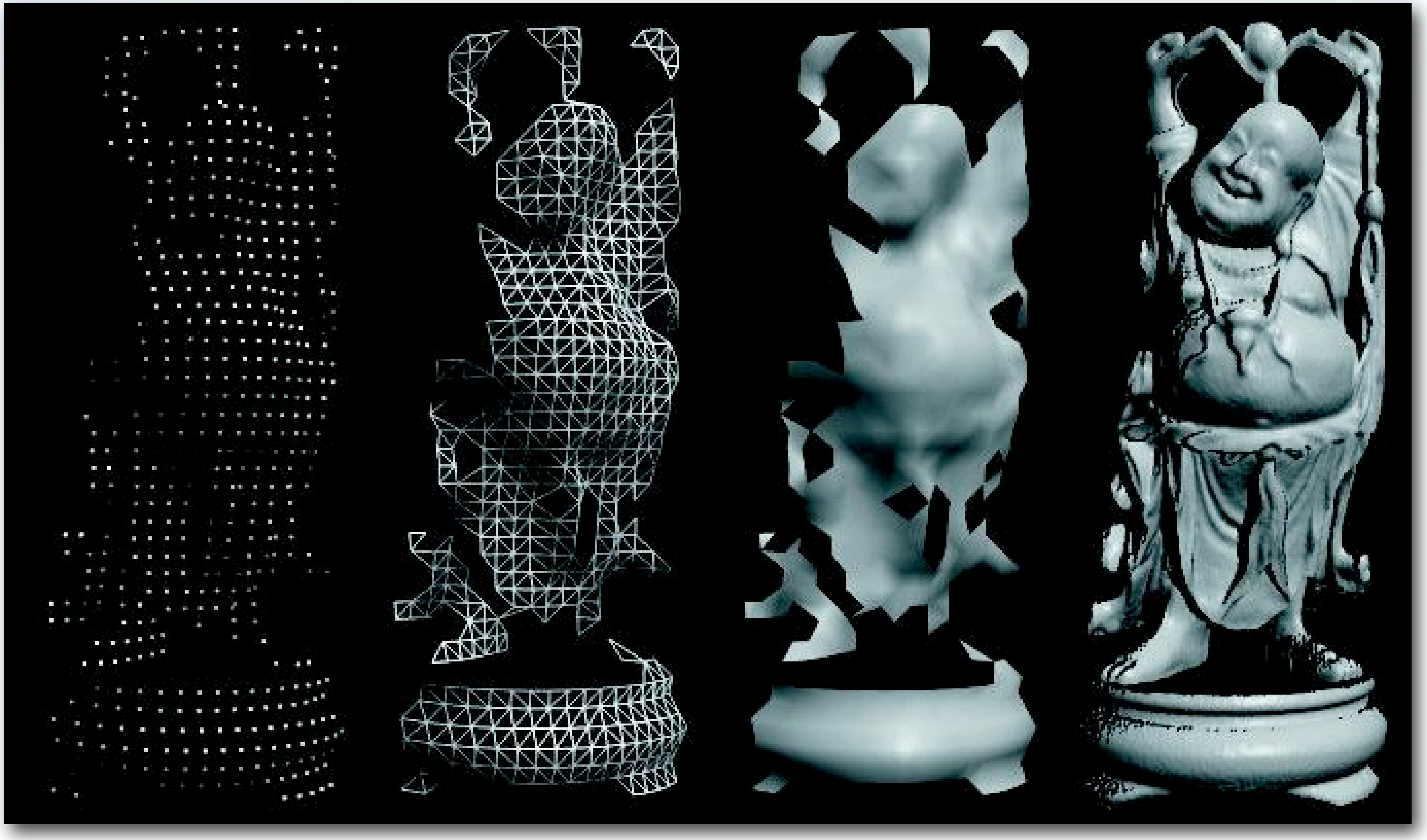


Image courtesy: Stanford University

Digital Michelangelo Project

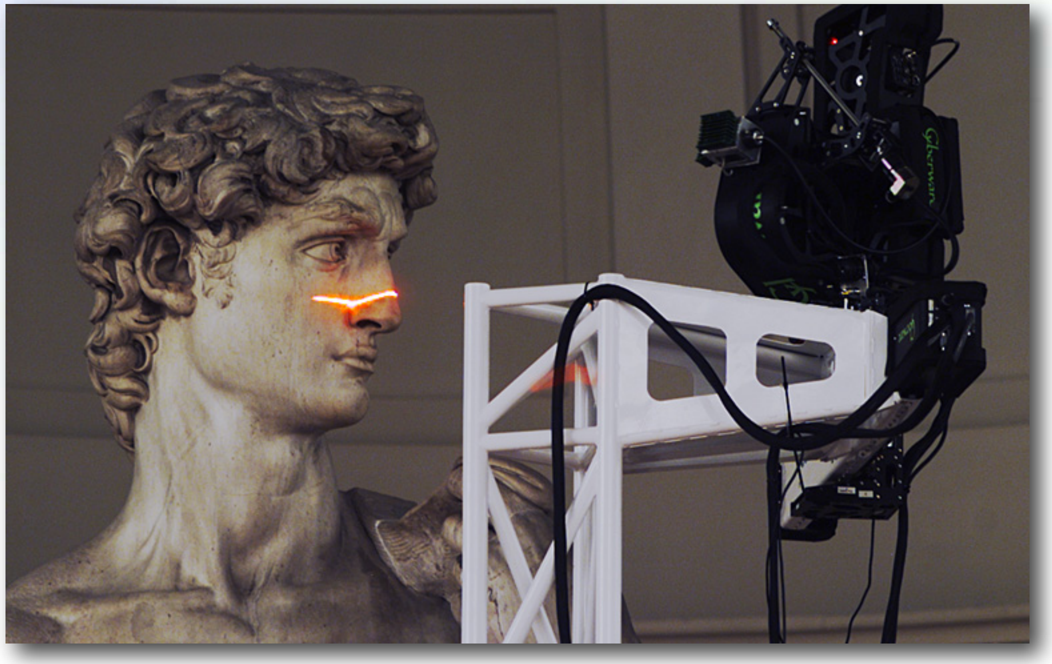


Image courtesy: Stanford University

Real-time 3-D acquisition

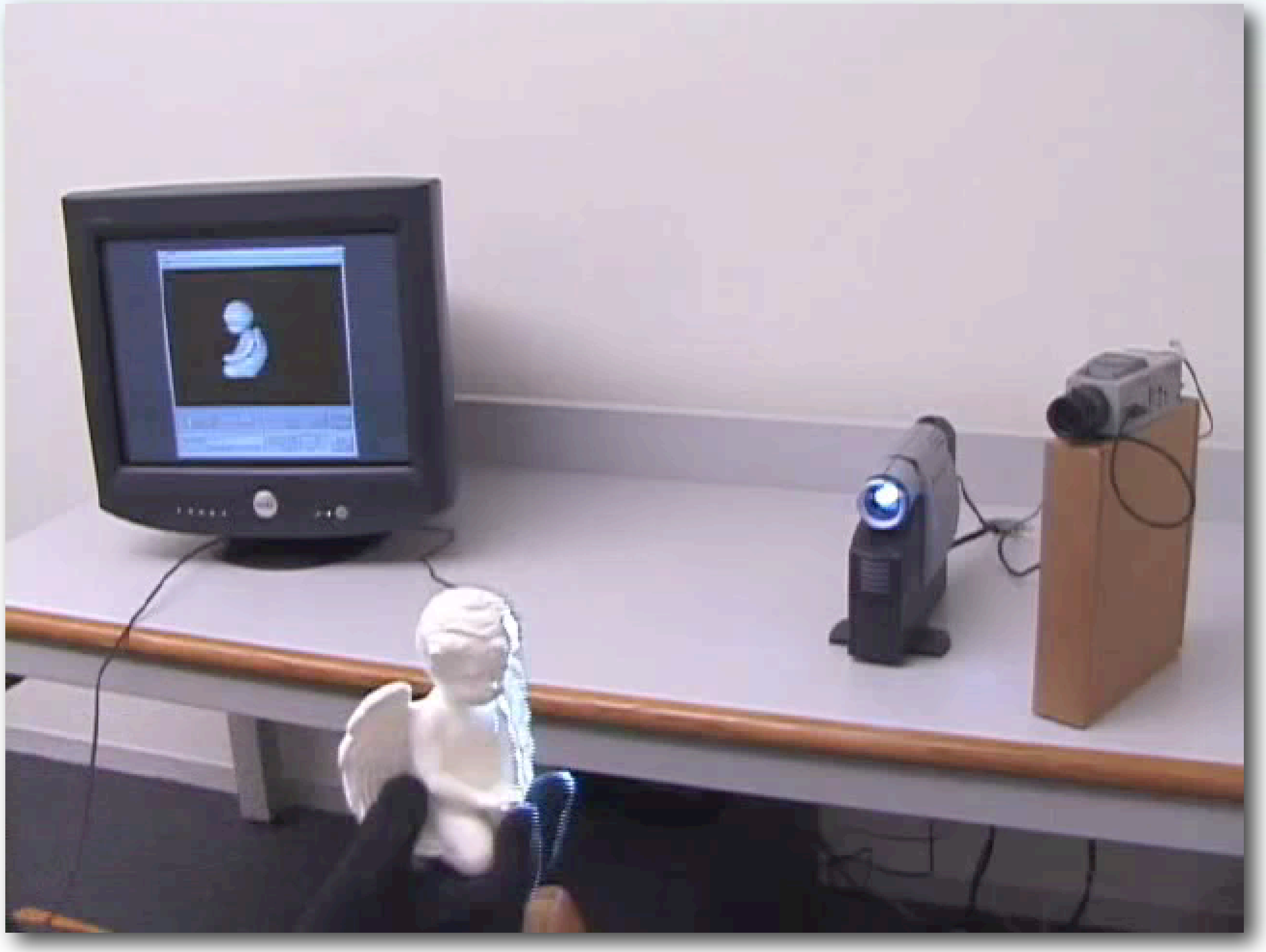
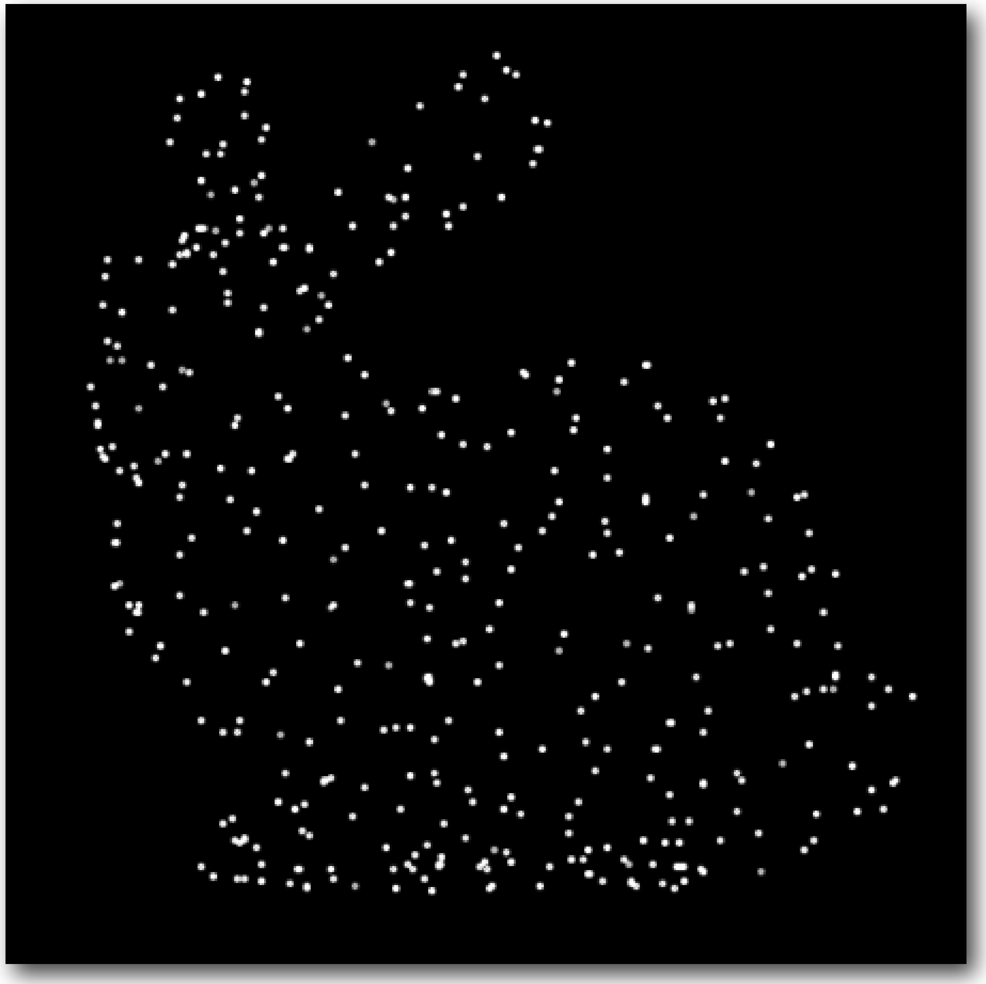


Image courtesy: Stanford University

Surface Reconstruction

Problem

Given a set of points $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ with $\mathbf{p}_i \in \mathbb{R}^3$



Problem

Find a manifold surface $\mathcal{S} \subset \mathbb{R}^3$ which approximates $\mathcal{P}$

Two approaches

Explicit

Local surface connectivity estimation

Point interpolation

Implicit

Signed distance function estimation

Mesh approximation

Two approaches

Explicit

– Ball pivoting algorithm
– Delaunay triangulation
– Alpha shapes
– ...

– Image space triangulation

Implicit

– Distance from tangent planes
– SDF estimation via RBF
– ...

Implicit surface reconstruction

Given a set of points $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ with $\mathbf{p}_i \in \mathbb{R}^3$
Find a manifold surface $\mathcal{S} \subset \mathbb{R}^3$ which approximates $\mathcal{P}$

where $\mathcal{S} = \{\mathbf{x} \mid d(\mathbf{x}) = 0\}$ with $d(\mathbf{x})$ a signed distance function

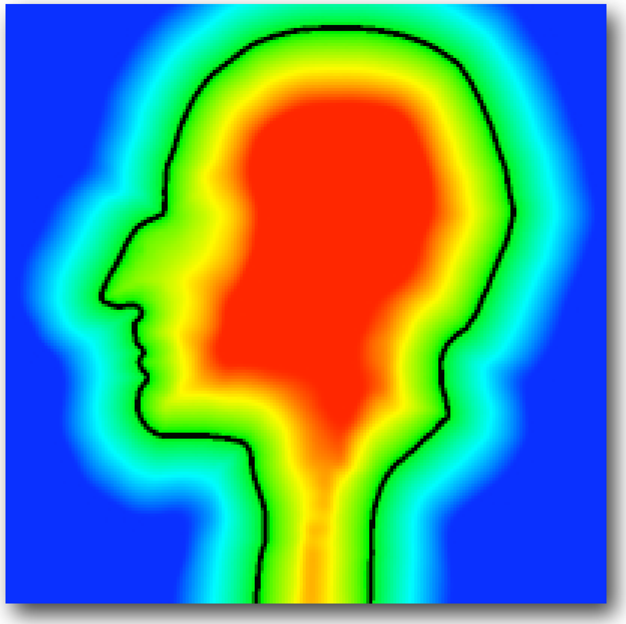


Image courtesy: MPI Saarbrücken

Data flow

Point cloud

Signed distance function estimation

$d(\mathbf{x}) \downarrow$

Evaluation of distances on uniform grid

$d(\mathbf{i}), \mathbf{i} = [i, j, k] \in \mathbb{Z}^3 \downarrow$

Mesh extraction via marching cubes

Mesh

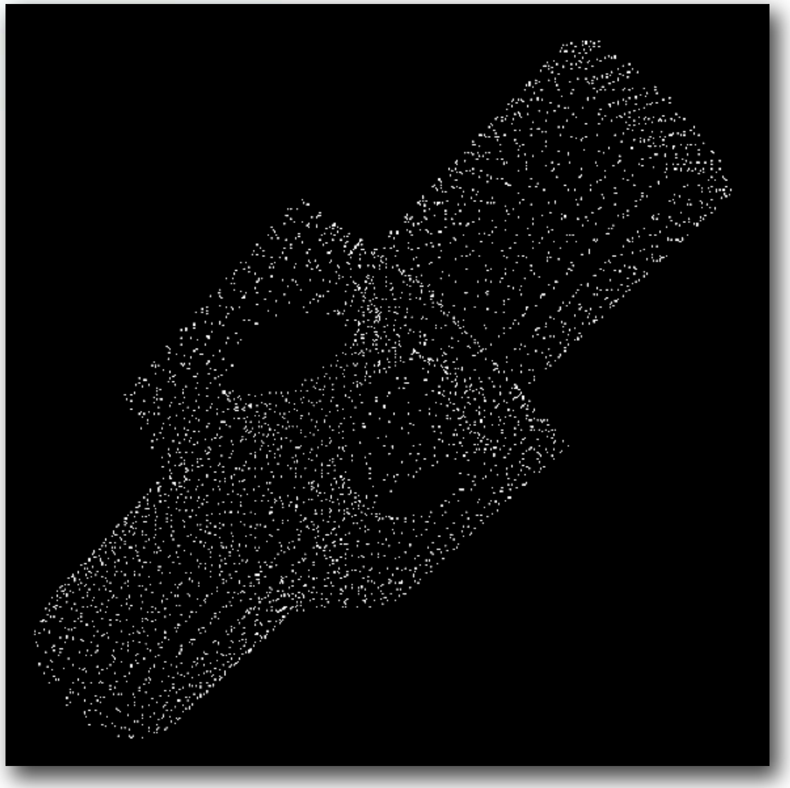
Implicit surface reconstruction methods

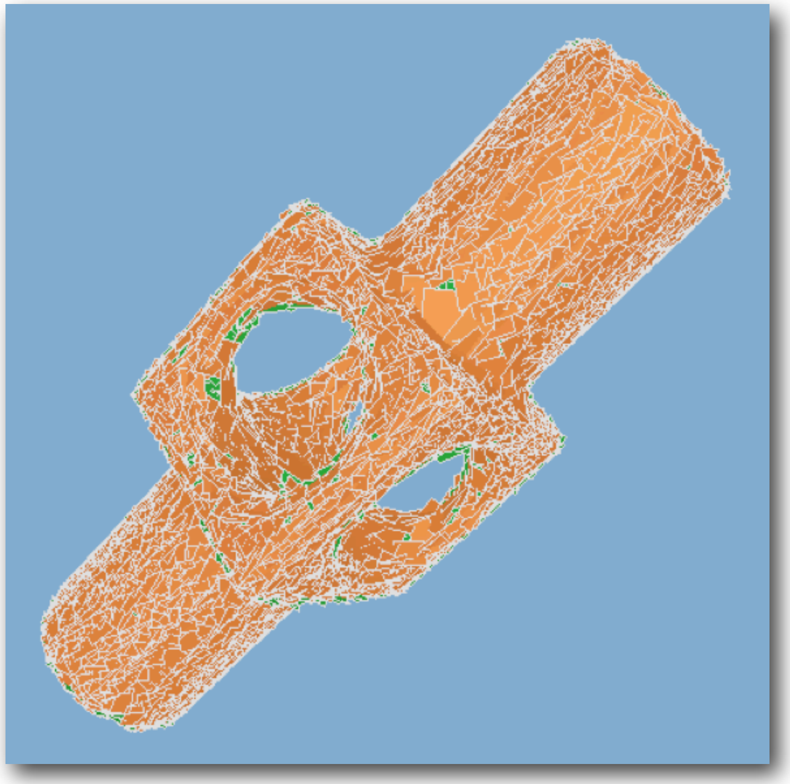
Mainly differ in their signed distance function

You will implement two popular methods

Hoppe's distance from tangent planes

$\mathcal{C}^0$ surface





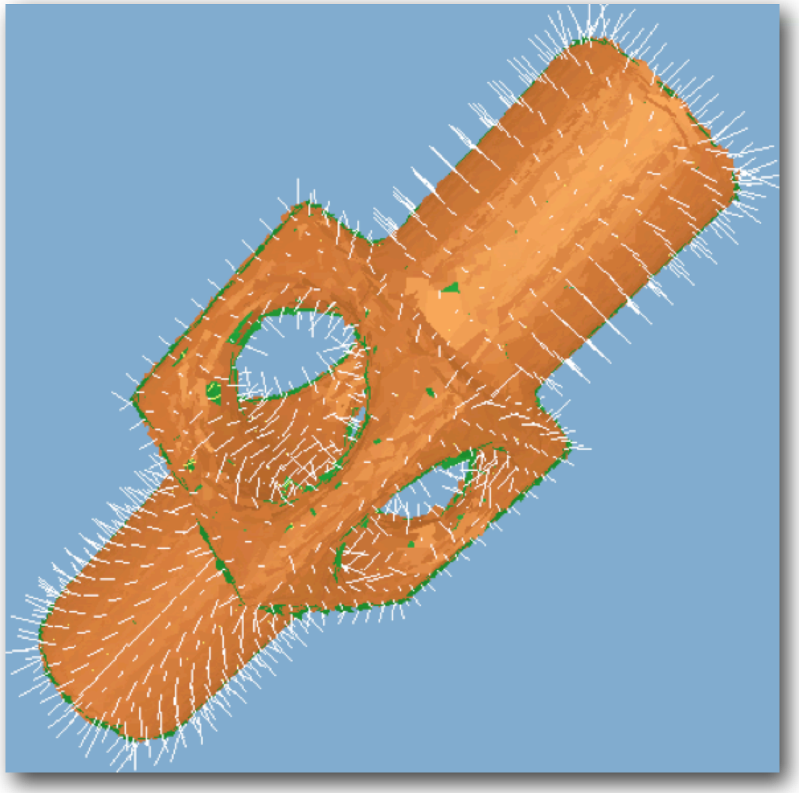
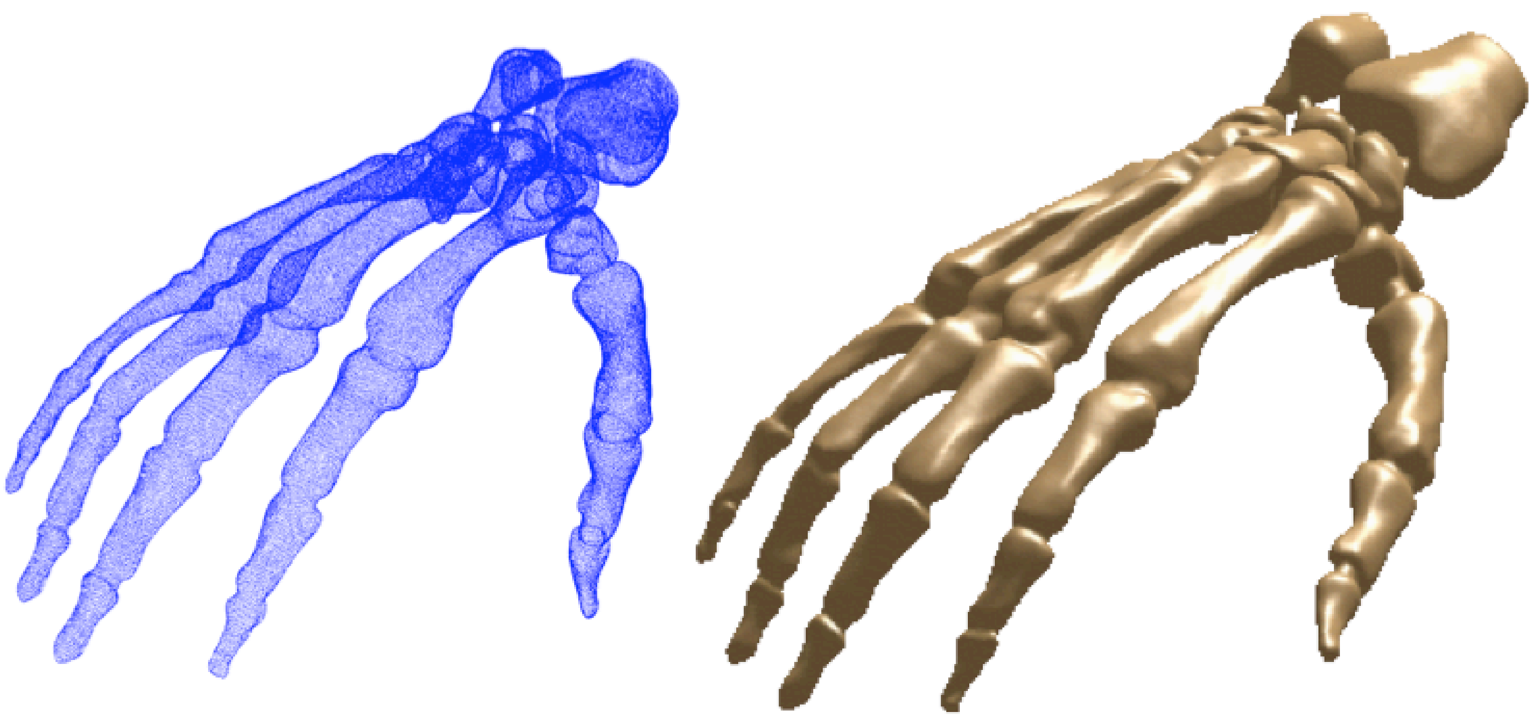


Image courtesy: University of Washington

You will implement two popular methods

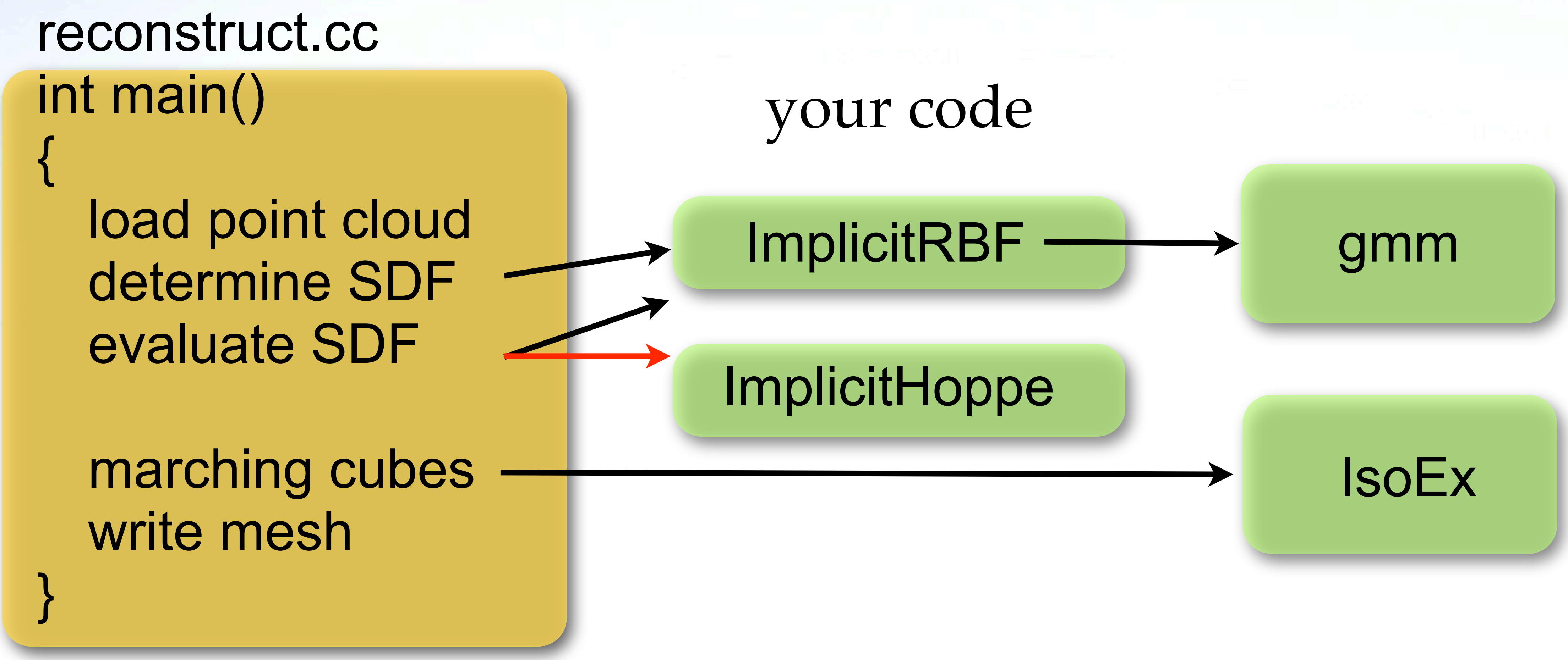
Signed distance function via triharmonic RBFs



$\mathcal{C}^2$ surface

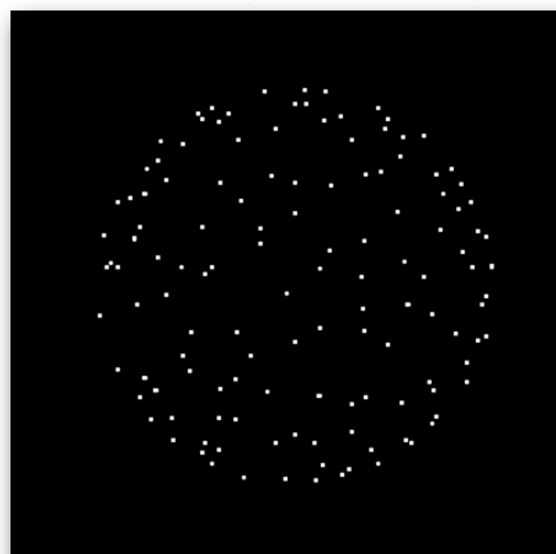
Image courtesy: University of Canterbury

Your new framework



Input data

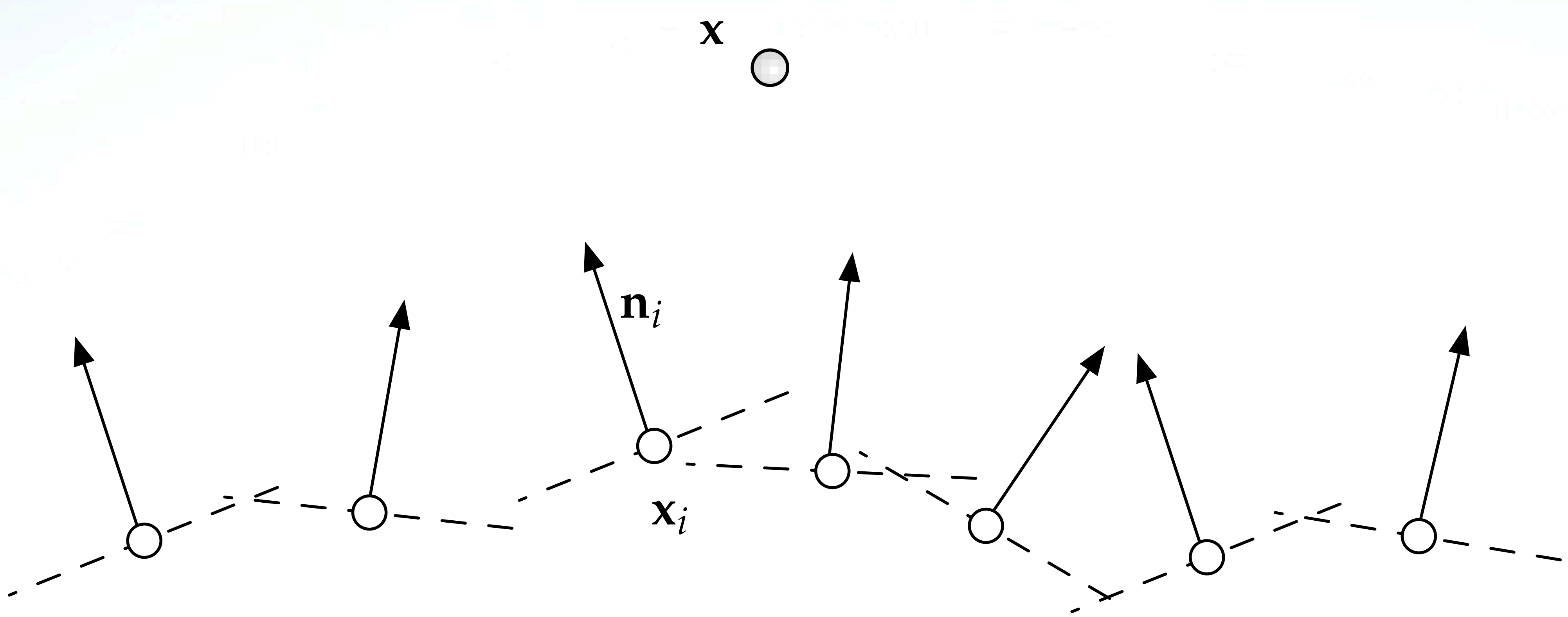
bunny-500.pts



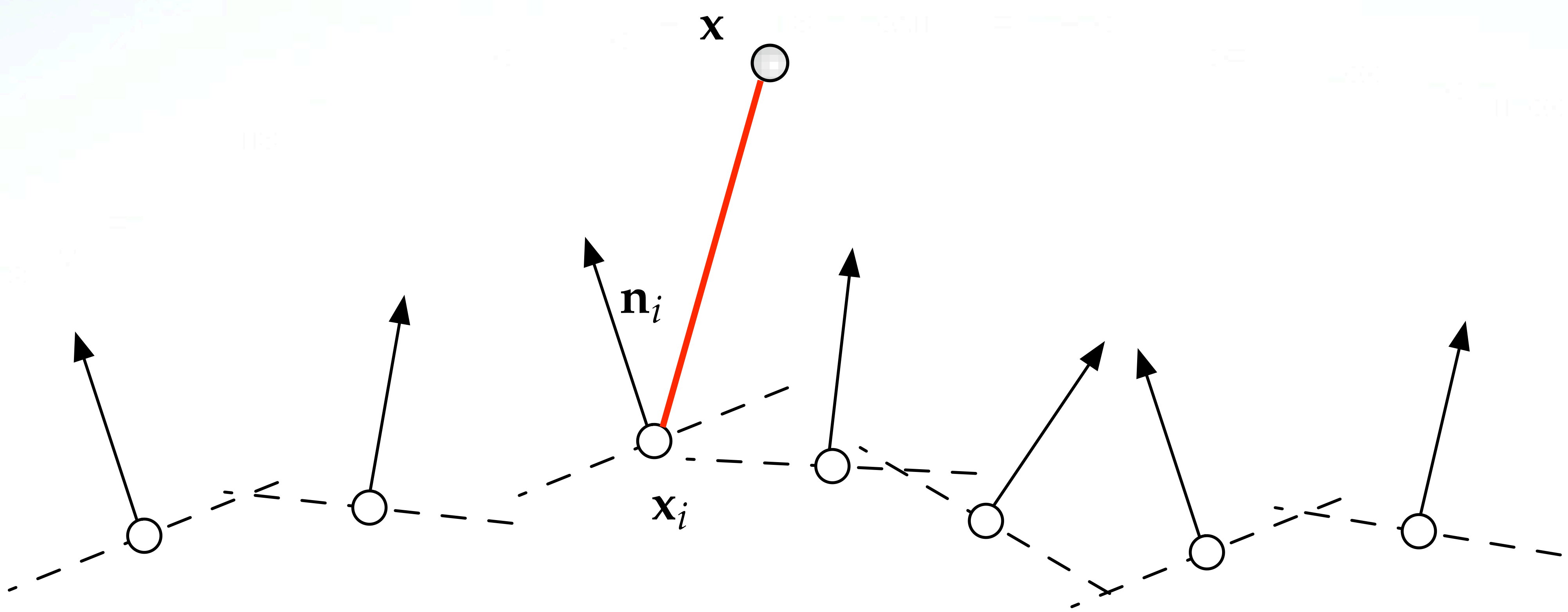
sphere.pts

– ASCII file format
– List of point and normal coordinates

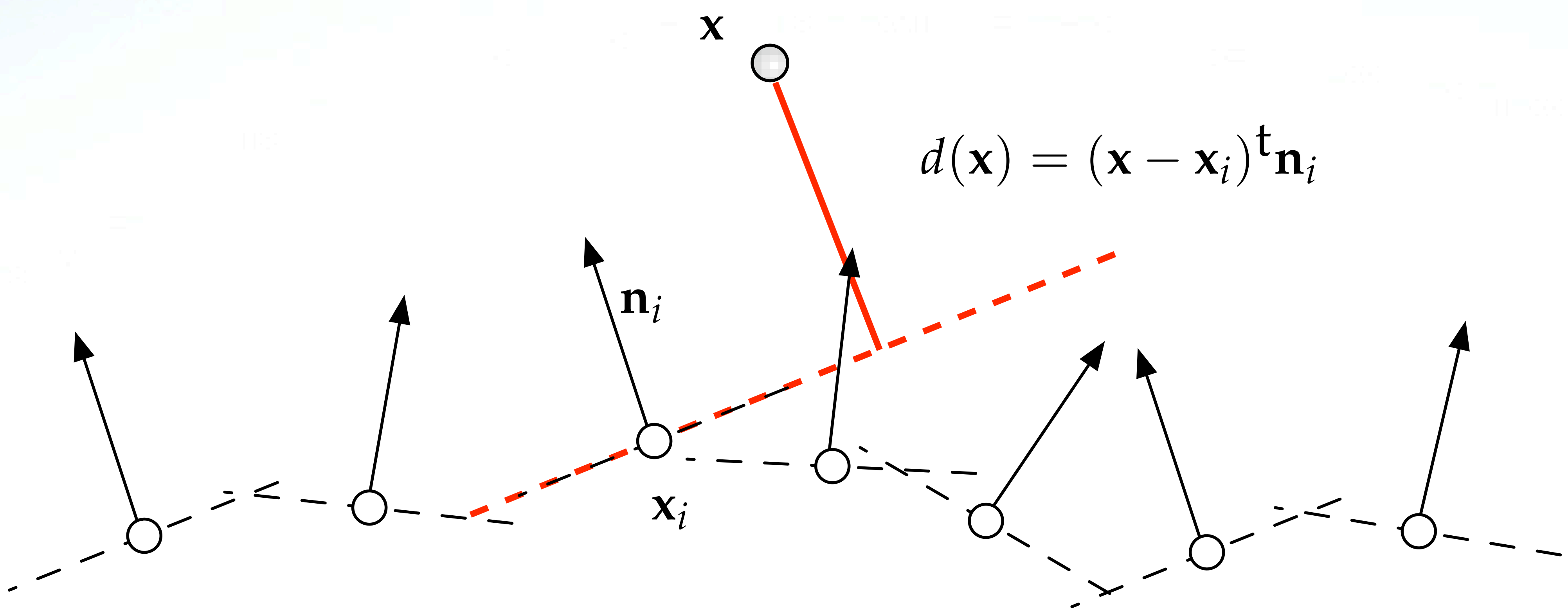
Exercise 2.1:
Distance from tangent planes



Find Closest point



Distance to tangent plane



Distance function is explicitly defined
→ no need to be estimated, evaluation is sufficient

Exercise 2.2: Signed distance function via RBFs

$$d(\mathbf{x}) = \sum_i w_i \phi(\|\mathbf{x} - \mathbf{c}_i\|)$$

contribution weights

radial basis function (kernel)

centers

Must be estimated using contraints
→ Data interpolation problem

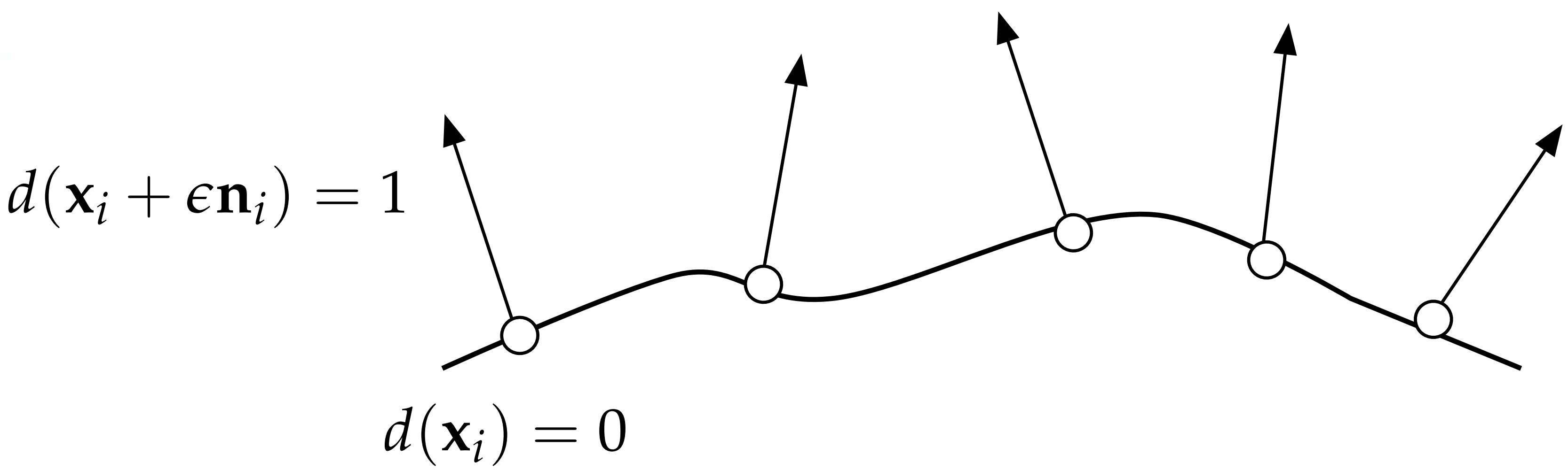
Estimate the signed distance function

$$d(\mathbf{x}) = \sum_i w_i \phi(\|\mathbf{x} - \mathbf{c}_i\|)$$

- Centers are chosen as contraint points
- Values contrained to points where distances are known
- Kernel is a triharmonic RBF: $\phi(x) = x^3$
- Weights have to be determined

On- and off-surface constraints

$$d(\mathbf{x}) = \sum_i w_i \phi(\|\mathbf{x} - \mathbf{c}_i\|)$$



$2n$ point contraints with known distances

Linear system for weights

$$d(\mathbf{x}) = \sum_i w_i \phi(\|\mathbf{x} - \mathbf{c}_i\|)$$

$$d(\mathbf{x}_i + \epsilon \mathbf{n}_i) = 1$$

$$d(\mathbf{x}_i) = 0$$

$$\wr$$

$$\begin{bmatrix} \phi(\|\mathbf{x}_1 - \mathbf{x}_1\|) & \dots & \phi(\|\mathbf{x}_1 - (\mathbf{x}_n + \epsilon \mathbf{n}_n)\|) \\ \vdots & \ddots & \vdots \\ \phi(\|(\mathbf{x}_n + \epsilon \mathbf{n}_n) - \mathbf{x}_1\|) & \dots & \phi(\|(\mathbf{x}_n + \epsilon \mathbf{n}_n) - (\mathbf{x}_n + \epsilon \mathbf{n}_n)\|) \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_{2n} \end{bmatrix} = \begin{bmatrix} d_1 \\ \vdots \\ d_{2n} \end{bmatrix}$$

Solve system using the gmm library

Linear system for weights

$$d(\mathbf{x}) = \sum_i w_i \phi(\|\mathbf{x} - \mathbf{c}_i\|)$$

$$d(\mathbf{x}_i + \epsilon \mathbf{n}_i) = 1$$

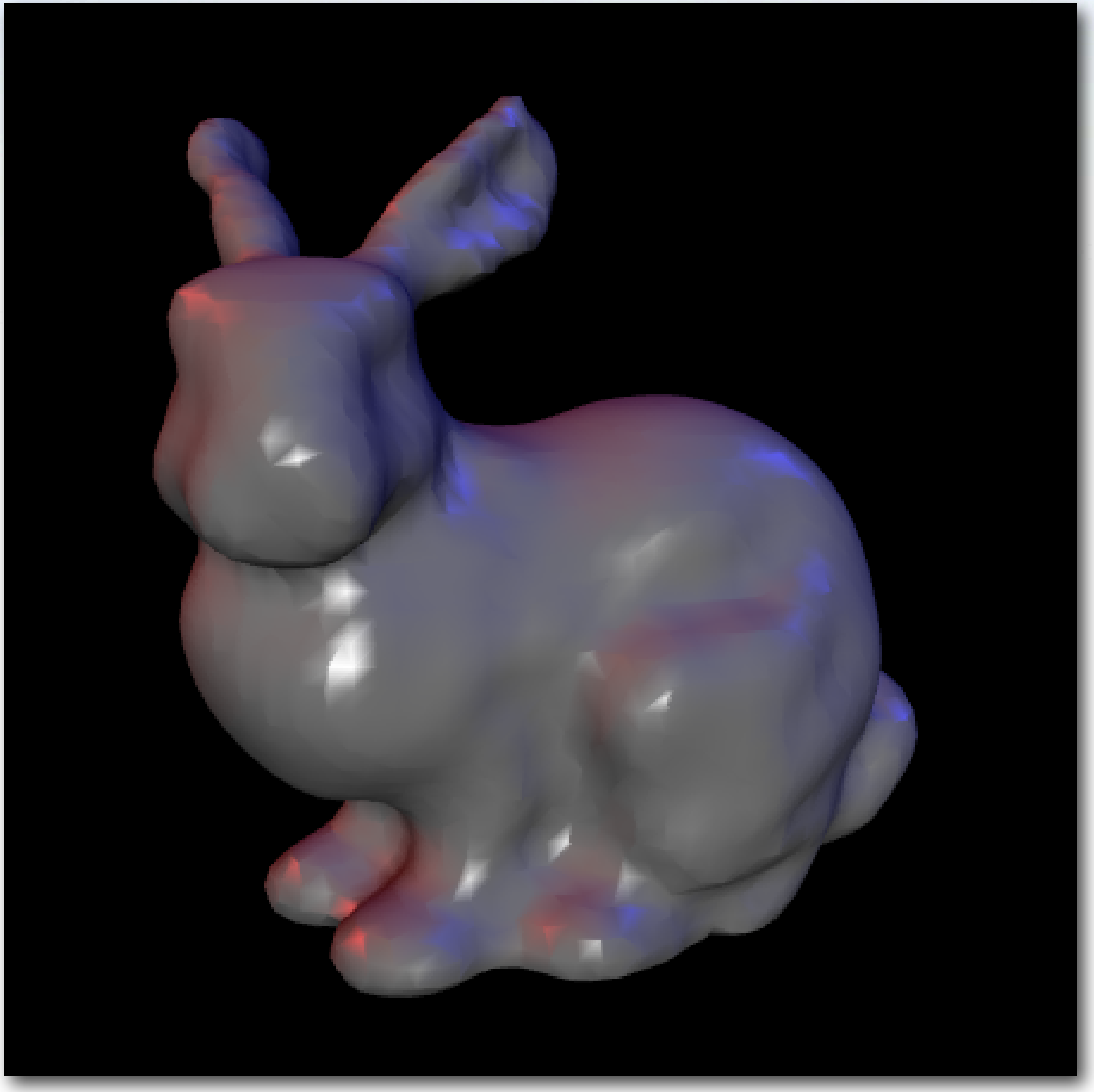
$$d(\mathbf{x}_i) = 0$$

$$\wr$$

$$\begin{bmatrix} \phi(\|\mathbf{x}_1 - \mathbf{x}_1\|) & \dots & \phi(\|\mathbf{x}_1 - (\mathbf{x}_n + \epsilon \mathbf{n}_n)\|) \\ \vdots & \ddots & \vdots \\ \phi(\|(\mathbf{x}_n + \epsilon \mathbf{n}_n) - \mathbf{x}_1\|) & \dots & \phi(\|(\mathbf{x}_n + \epsilon \mathbf{n}_n) - (\mathbf{x}_n + \epsilon \mathbf{n}_n)\|) \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_{2n} \end{bmatrix} = \begin{bmatrix} d_1 \\ \vdots \\ d_{2n} \end{bmatrix}$$

distance d is fully defined

After a few seconds of computation you should obtain this



One more exercise...

Exercise 2.3 (optional): For the passionate

- Implement the *Wendland* RBF kernel
- Create more point data sets for implicit surface fitting

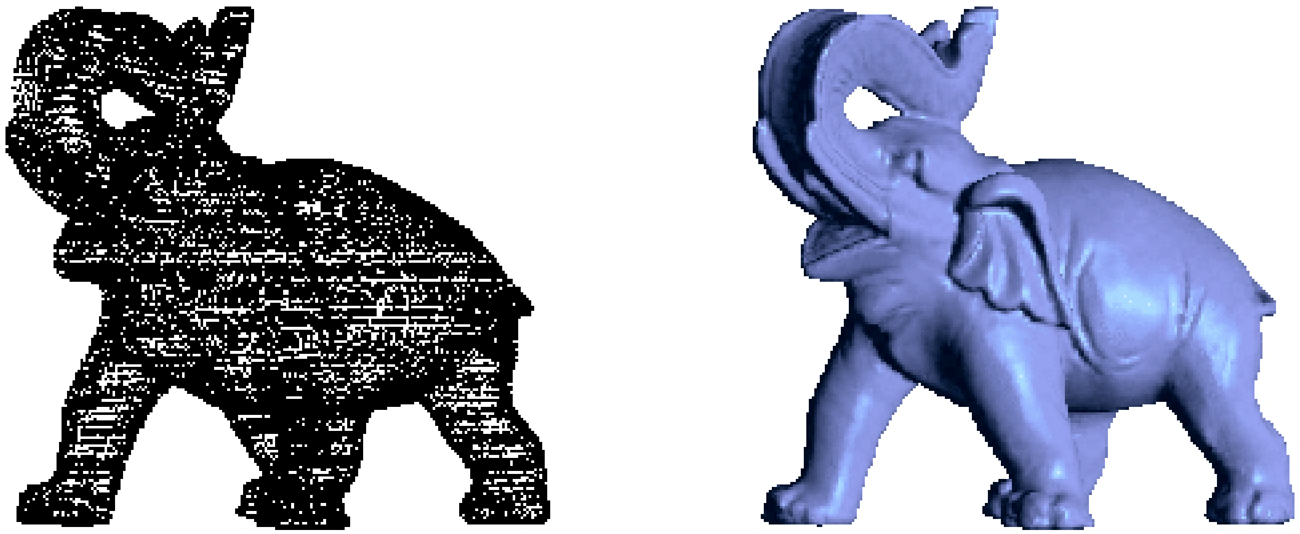


Image courtesy: Universität Göttingen

Further readings

- Surface reconstruction from unorganized points. [Hoppe et al. ‘92]
- A volumetric method for building complex models from range images [Curless and Levoy ‘96]
- Reconstruction and representation of 3D objects with radial basis functions [Carr et al. ‘01]

?

hao@inf.ethz.ch