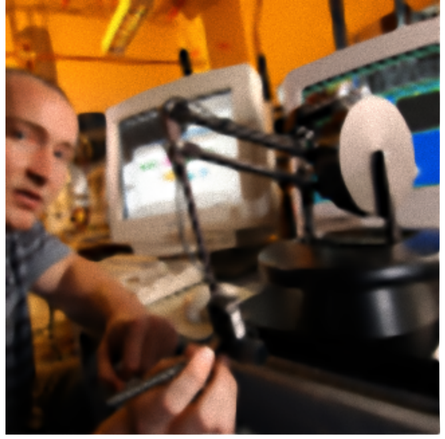
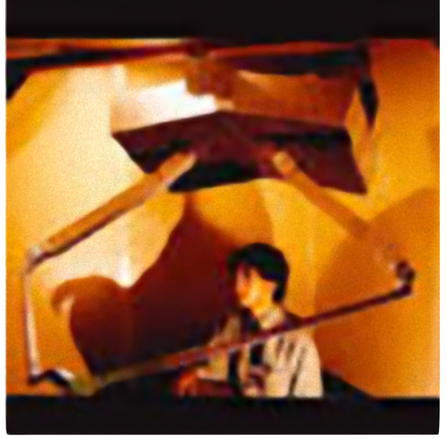
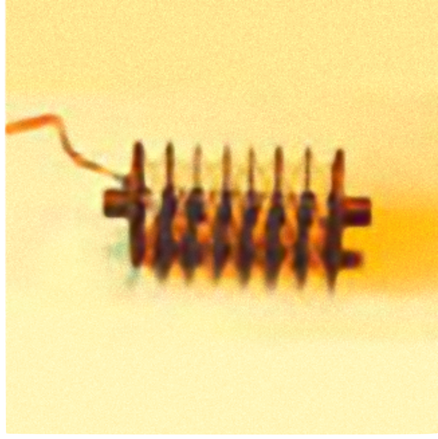
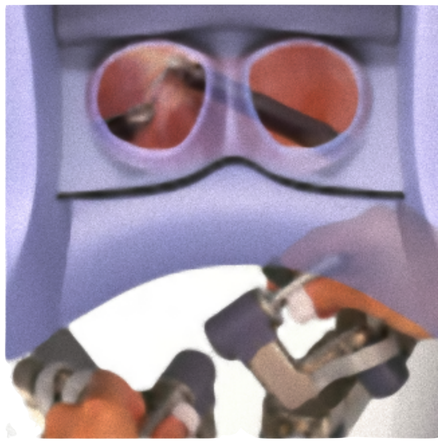





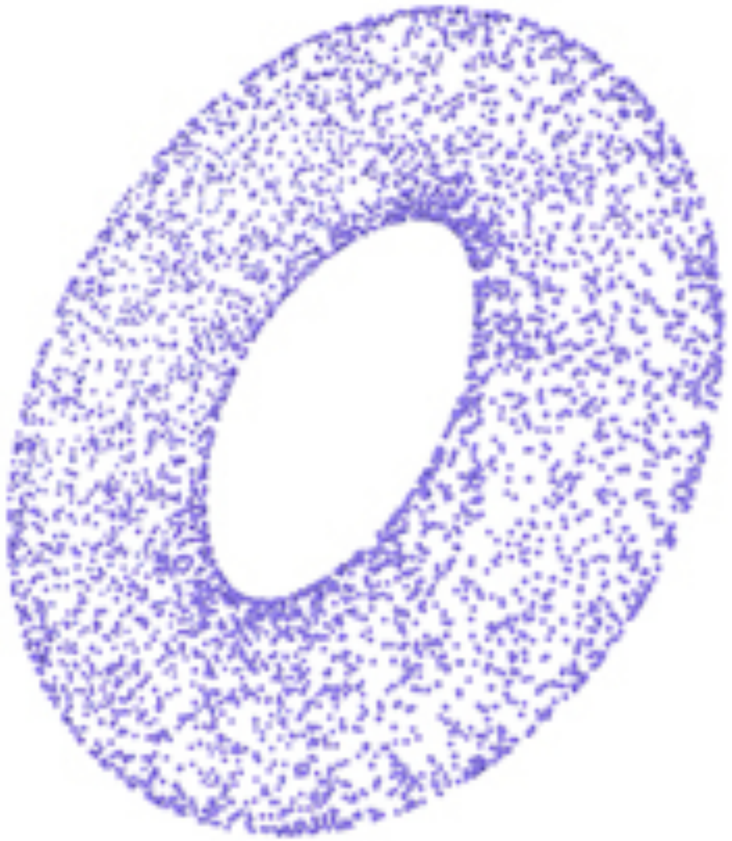


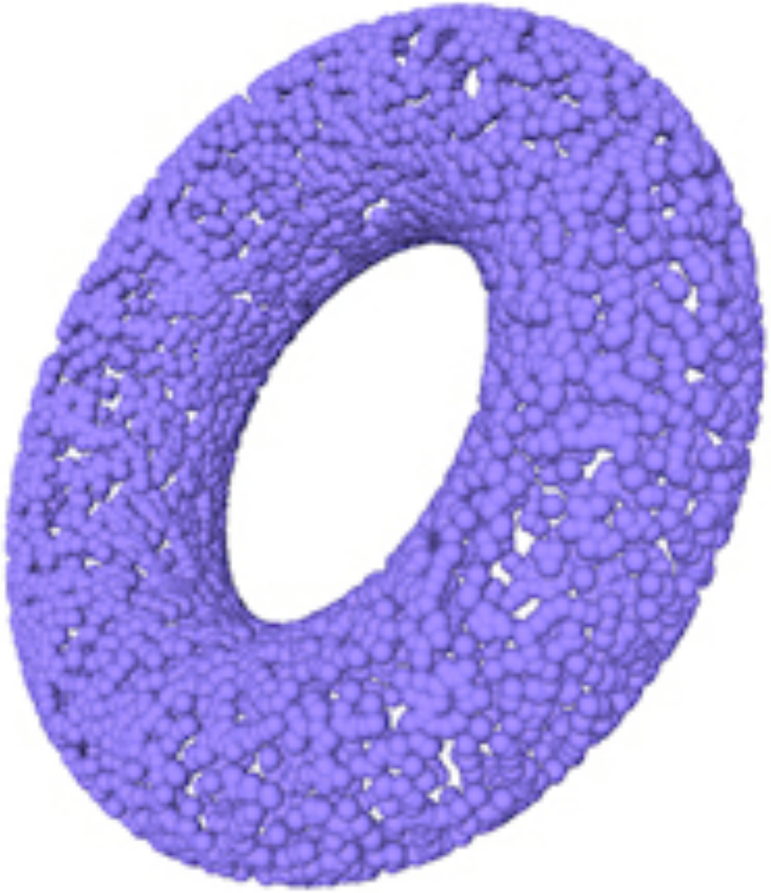



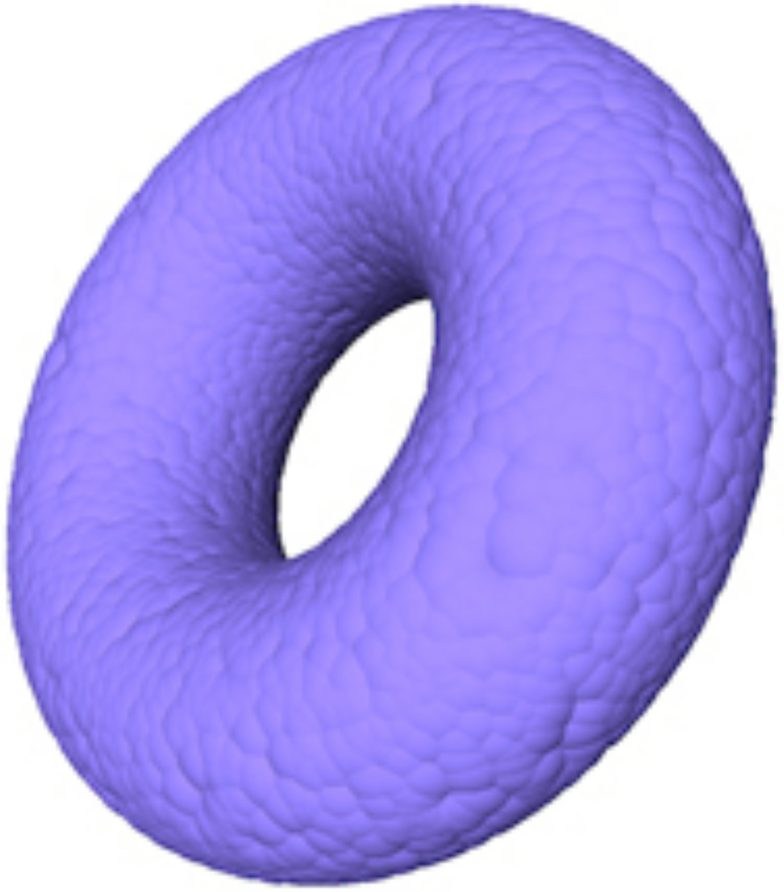
Point Set Surfaces

CPSC 599.86 / 601.86

Sonny Chan
University of Calgary





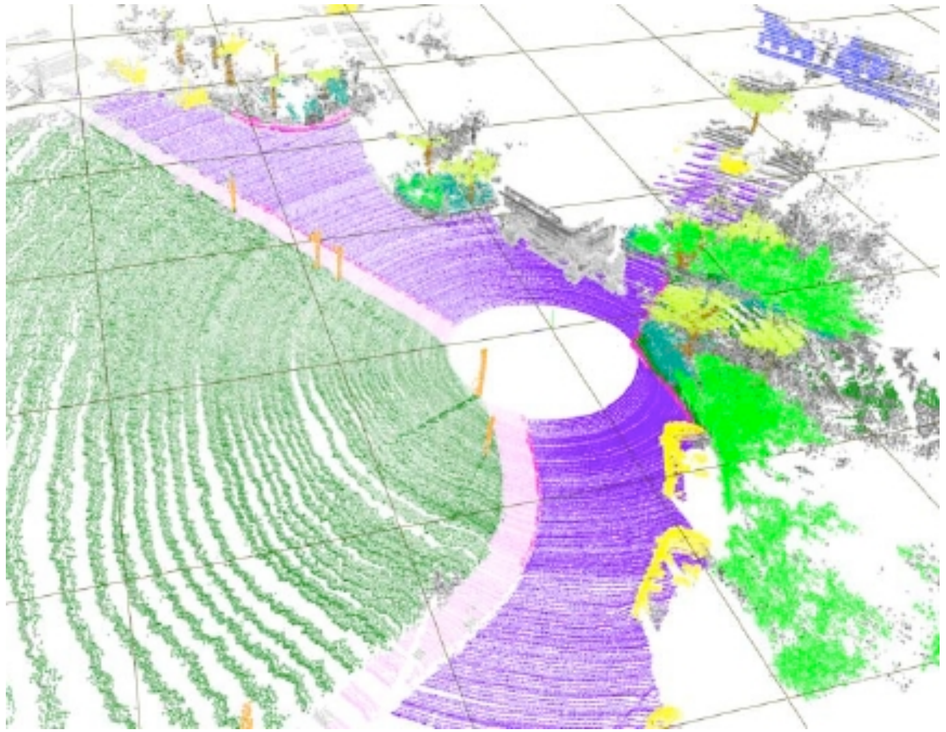


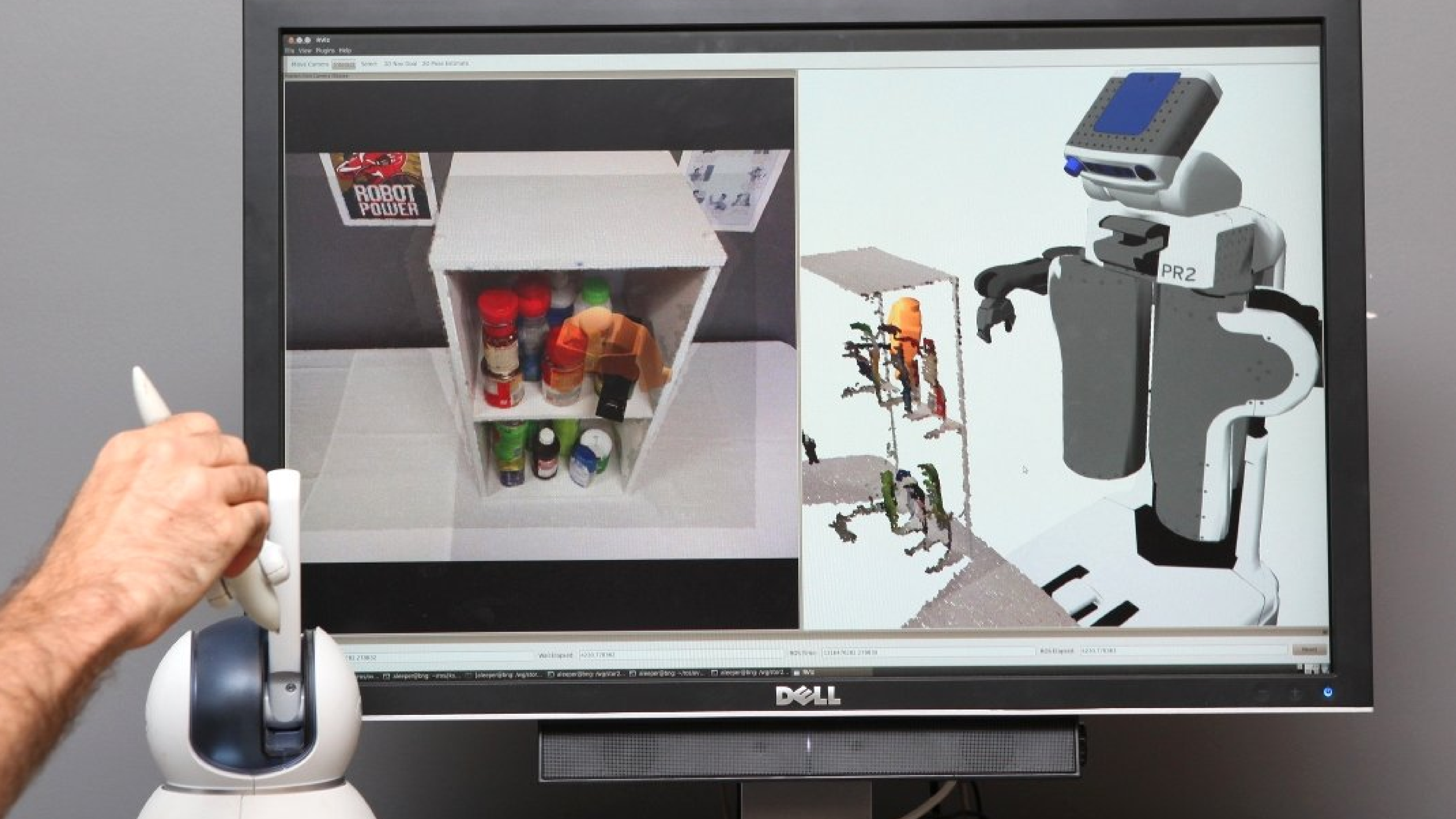
Unstructured Point Sets

as a surface representation

Data Sources

- Why might we want to touch these surfaces? (aside from assignment #2)




ROBOT
POWER
PR2
DELL

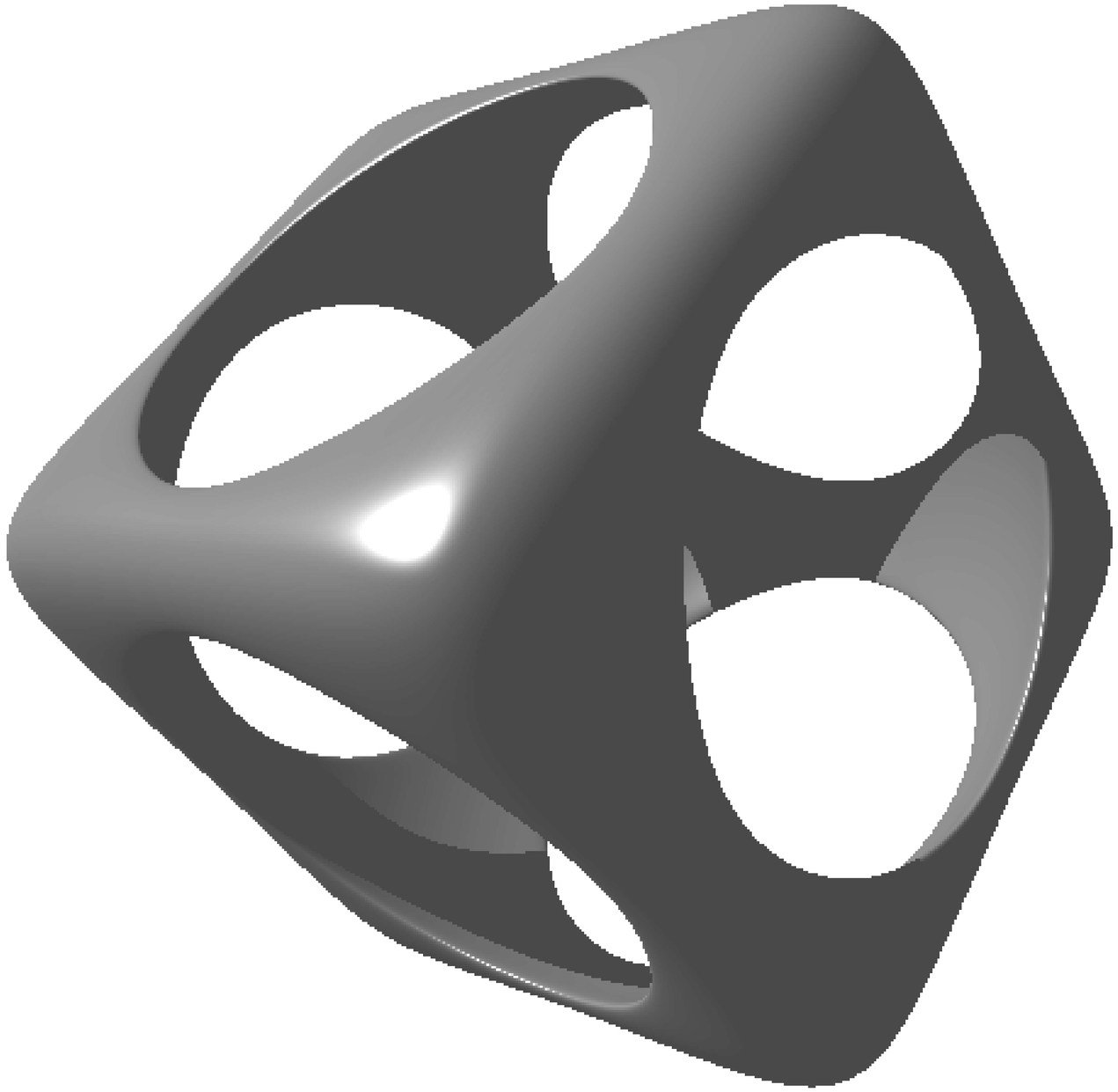
Point Set Surfaces

- Approximates a *smooth* surface from irregularly sampled points
- Create a local estimate of the surface at every point in space
- Can haptically render using implicit surface algorithm!
  - Test for intersection with the surface approximation

[from M. Alexa *et al.*, *IEEE Trans. Viz. & Comput. Graphics* 9(1), 2003.]

Implicit Surface Requirement

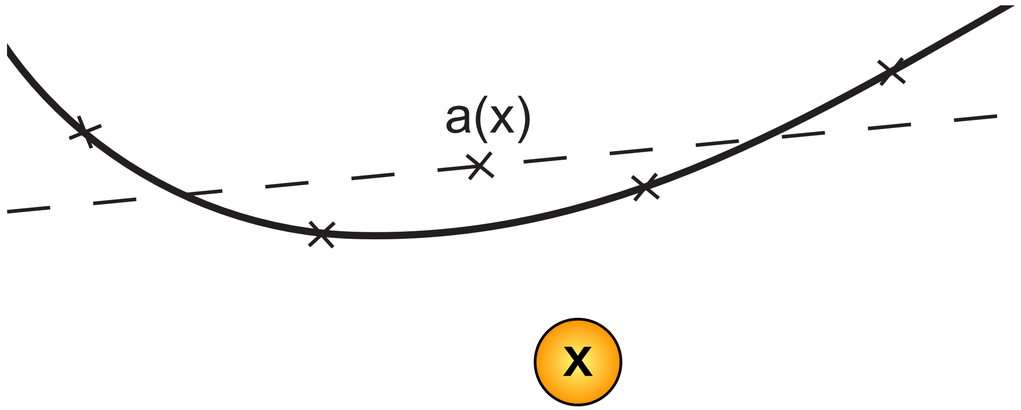
- Recall that the implicit surface algorithm needs two things:
  - A scalar surface function, $S(x, y, z)$, that tells us whether we're inside or outside the object
  - The gradient of the surface function, $\nabla S(x, y, z)$, that tells us the direction of the surface normal
- How do we get these?



Estimating Local Surface Position

- Weighted average of nearby points
- If we are at position **x**, estimate a point on the surface at

$$\mathbf{a}(\mathbf{x}) = \frac{\sum_i^n \theta_i(\|\mathbf{x} - \mathbf{p}_i\|)\mathbf{p}_i}{\sum_i^n \theta_i(\|\mathbf{x} - \mathbf{p}_i\|)}$$



  - based on object points $\mathbf{p}_i$, where θ is a weighting function of distance

Estimating Local Surface Normal

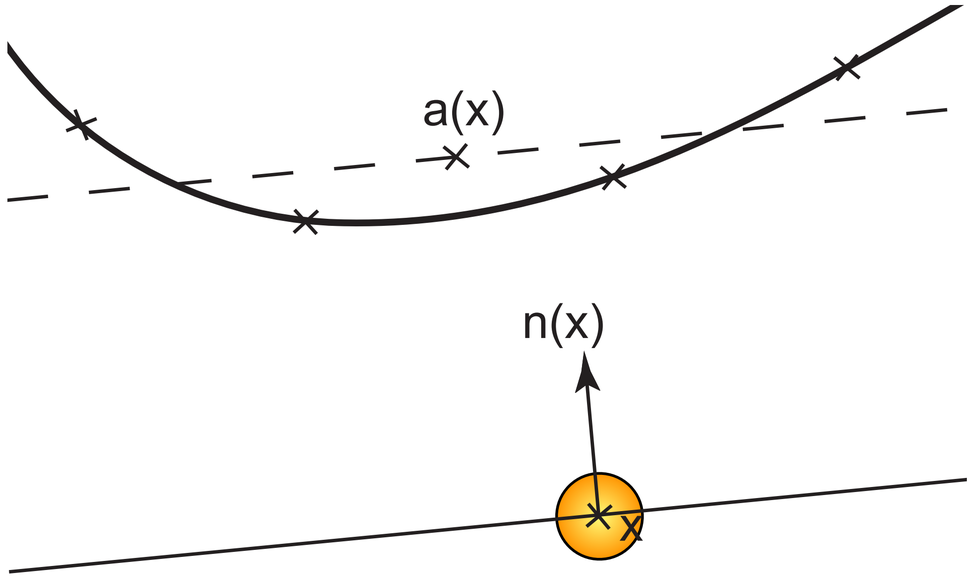
- Direction of smallest weighted covariance of nearby points

- If the weighted covariance is expressed as

$$\sigma_{\mathbf{n}}^2(\mathbf{x}) = \frac{\sum_i^n \theta_i(\|\mathbf{x}-\mathbf{p}_i\|)\,(\mathbf{n}\cdot(\mathbf{x}-\mathbf{p}_i))^2}{\sum_i^n \theta_i(\|\mathbf{x}-\mathbf{p}_i\|)}$$

- Then the surface normal direction is

$$\min_{\mathbf{n}} \sigma_{\mathbf{n}}^2(\mathbf{x})$$



Computing the Local Surface Normal

- So how do we find the normal that minimizes covariance?
- First, define the weighted covariance matrix:

$$\mathbf{W}(\mathbf{x}) = [w_{jk}] \in \mathbb{R}^{3\times 3}, \quad \text{where}$$

$$w_{jk} = \sum_i \left(\mathbf{e}_j \cdot (\mathbf{x} - \mathbf{p}_i)\right)\left(\mathbf{e}_k \cdot (\mathbf{x} - \mathbf{p}_i)\right)\theta(||\mathbf{p}_i - \mathbf{x}||)$$

  - with the standard basis vectors

$$\mathbf{e}_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}, \quad \mathbf{e}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}, \quad \text{and} \quad \mathbf{e}_3 = \begin{bmatrix}0\\0\\1\end{bmatrix}$$

Computing the Local Surface Normal

- Then find the eigenvectors and eigenvalues of the covariance matrix:

$$\mathbf{W}(\mathbf{x})\mathbf{v} = \lambda\mathbf{v}$$

- If the three eigenvectors $\mathbf{v}_0$, $\mathbf{v}_1$, $\mathbf{v}_2$ correspond to the eigenvalues $\lambda_0 \leq \lambda_1 \leq \lambda_2$, we select as the normal

$$\mathbf{n} = \mathbf{v}_0$$

- The Eigen library within CHAI3D has functions to do the heavy lifting for you…

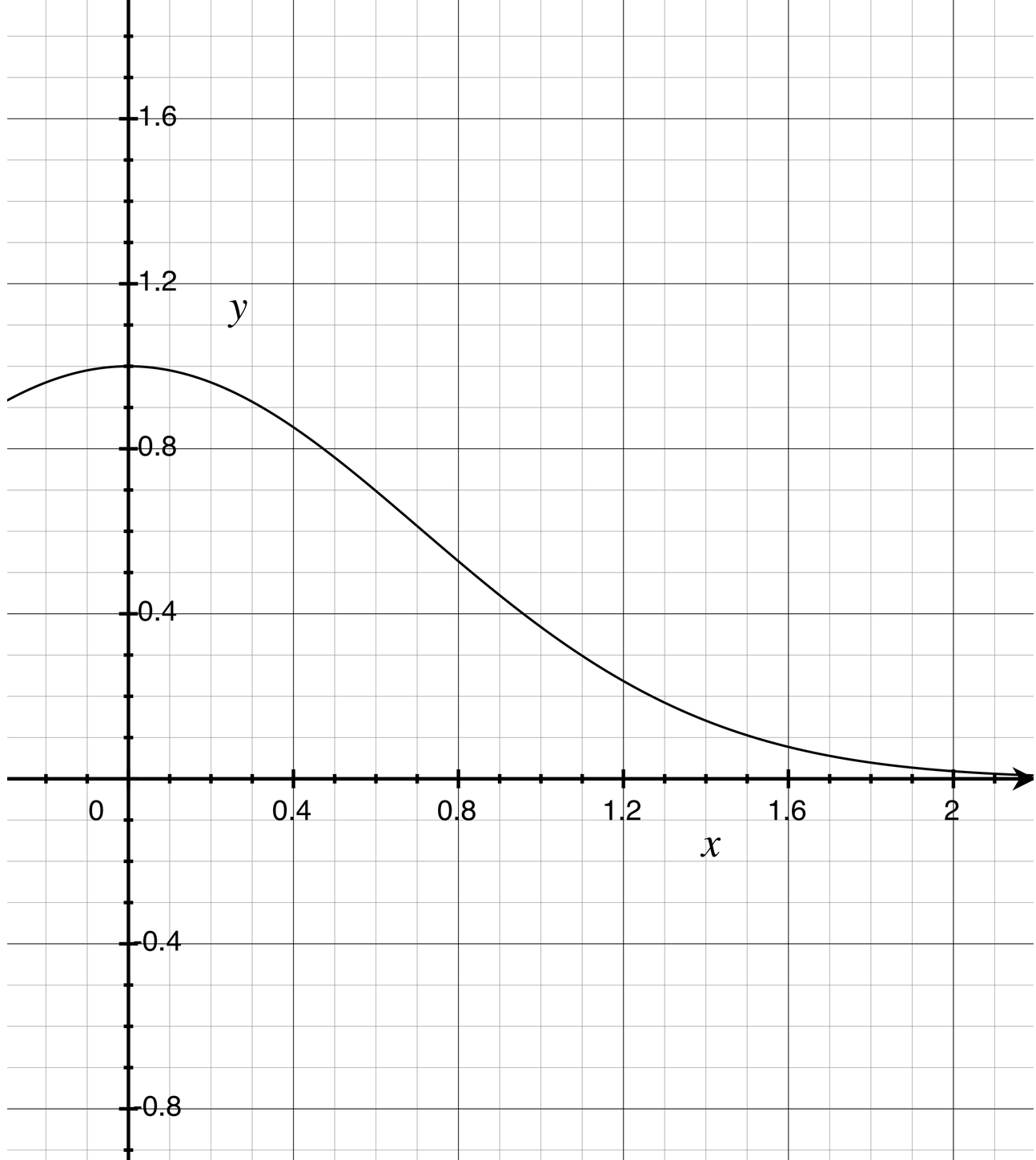
Choice of Weight Function

- Adamson & Alexa chose to use a Gaussian function:

$$\theta(r) = \exp\left(\frac{-r^2}{h^2}\right)$$

  - where *h* is an influence parameter

- What are the challenges with using a Gaussian?

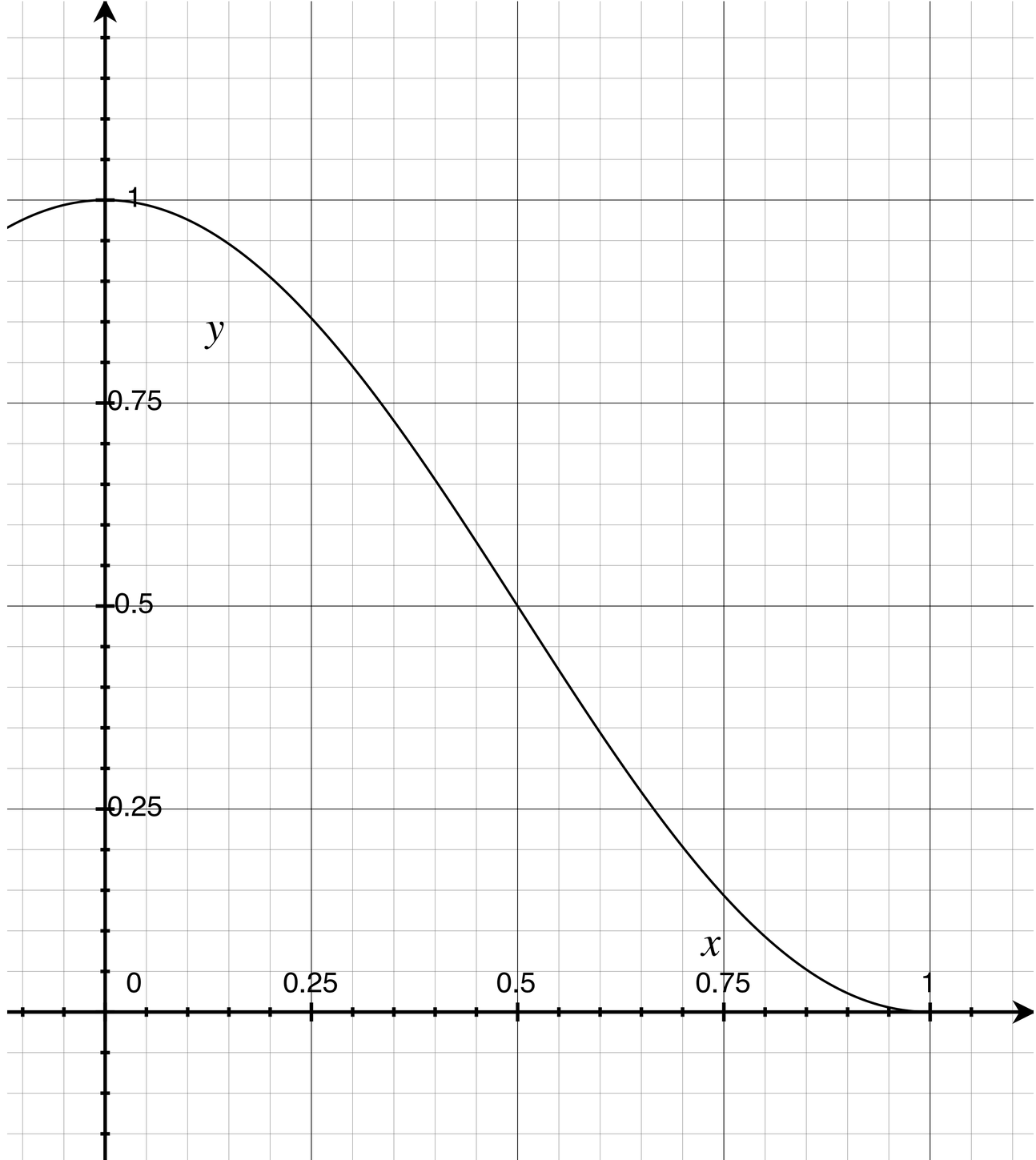


Choice of Weight Function

- We want to limit the domain of the function to $r \in [0, R]$, where R is radius of influence

- If we take the function to be cubic in r^2, then we get

$$\theta(r) = 1 - \frac{22}{9}\left(\frac{r}{R}\right)^2 + \frac{17}{9}\left(\frac{r}{R}\right)^4 - \frac{4}{9}\left(\frac{r}{R}\right)^6$$



[from G. Wyvill *et al.*, *The Visual Computer*, 1986.]

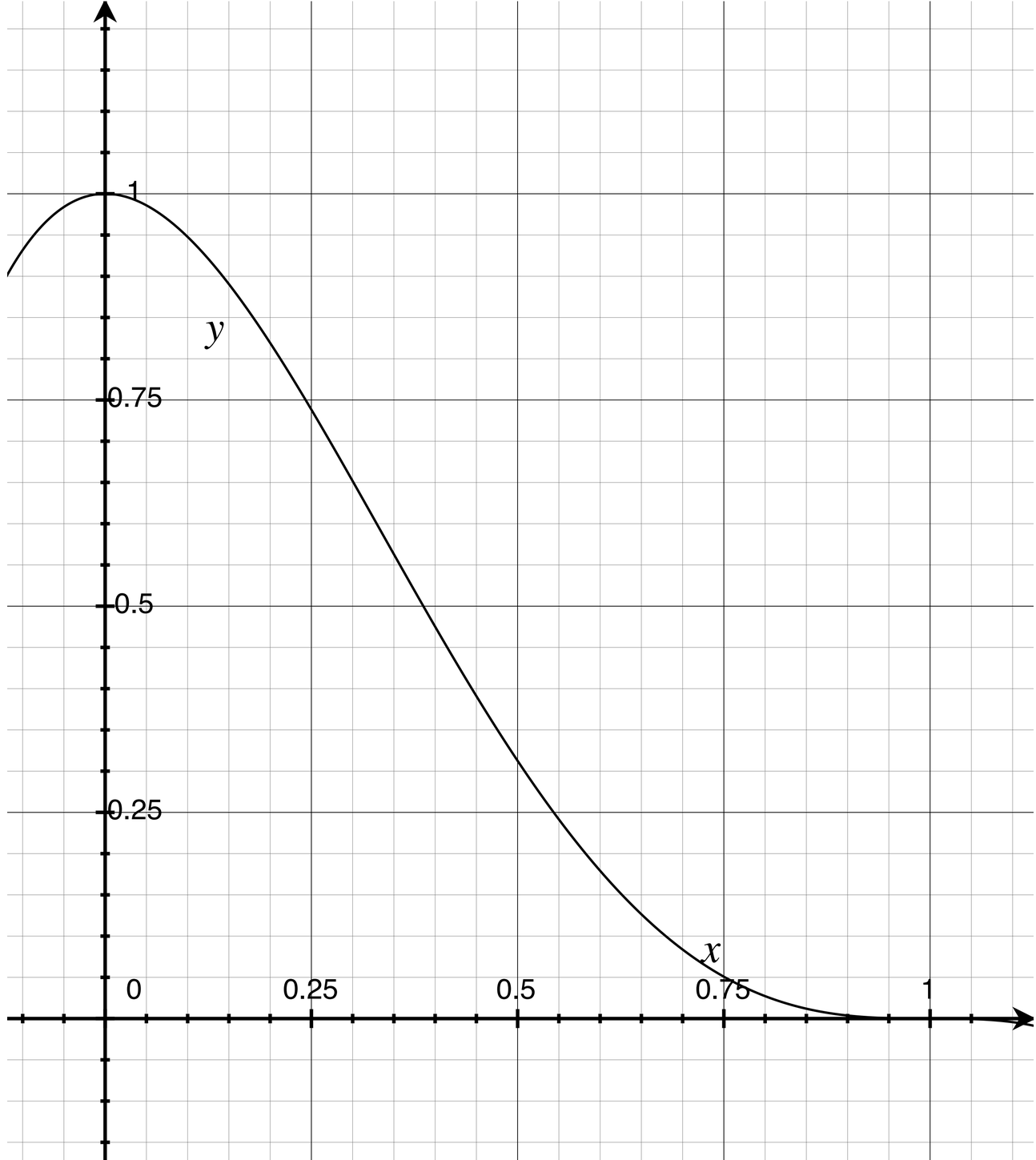
Choice of Weight Function

- Wendland's compactly supported radial basis function:

$$\theta(r) = (1 - r)^3(3r + 1)$$

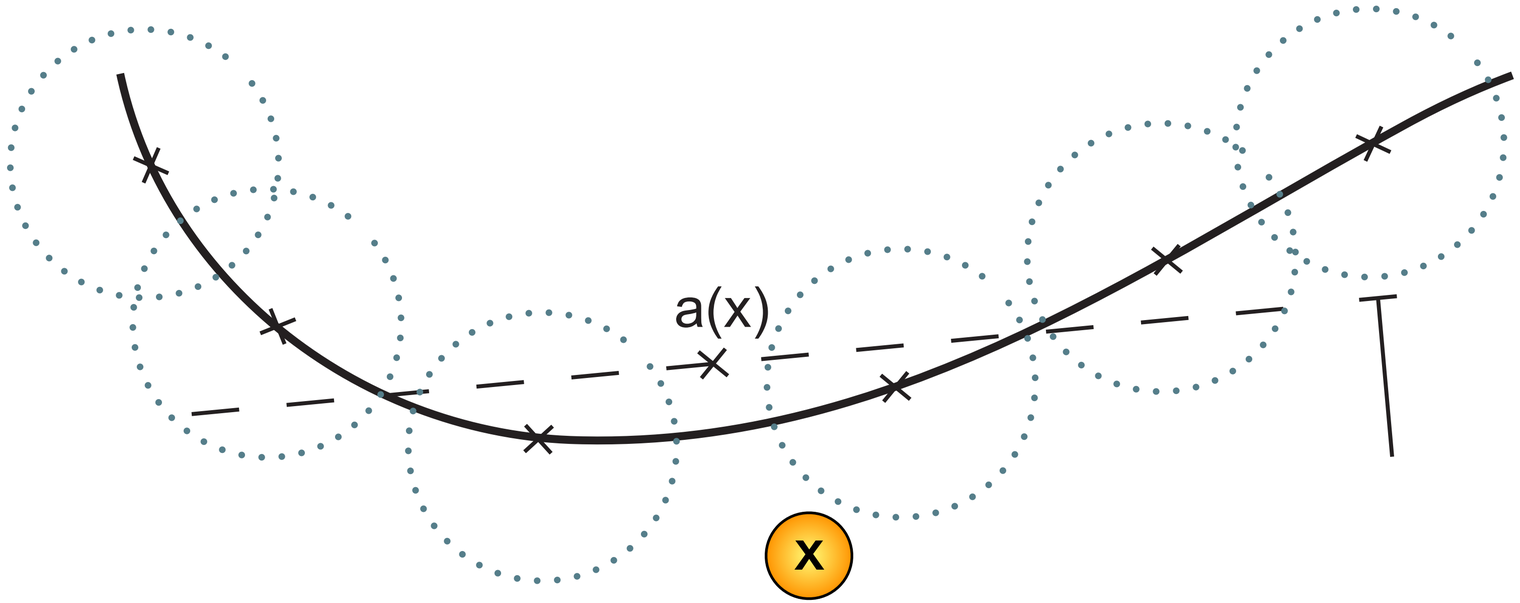
  - for domain $r \in [0, 1]$

- This one is closest in shape to the Gaussian function

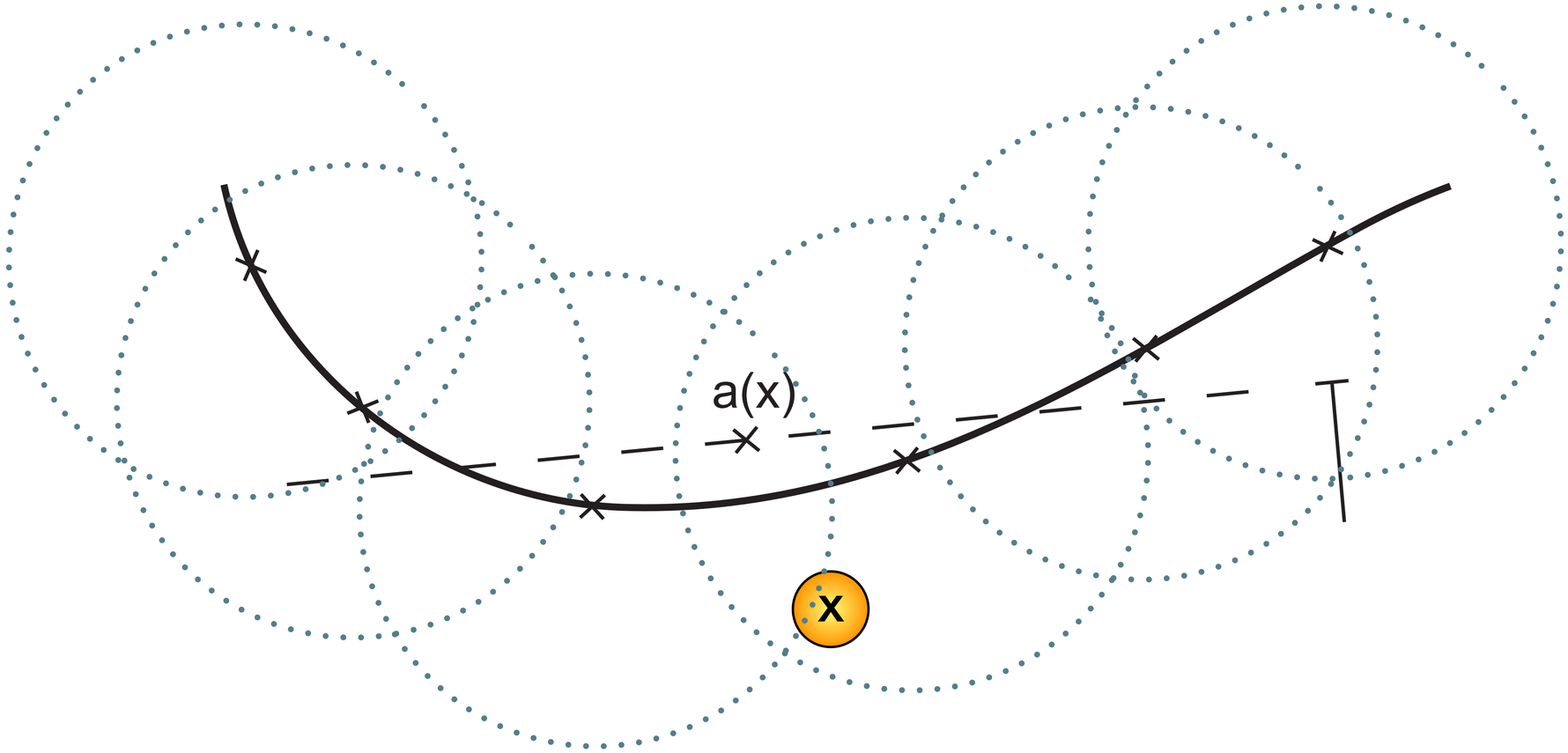


[from H. Wendland, *Adv. Comp. Mathematics*, 1995.]

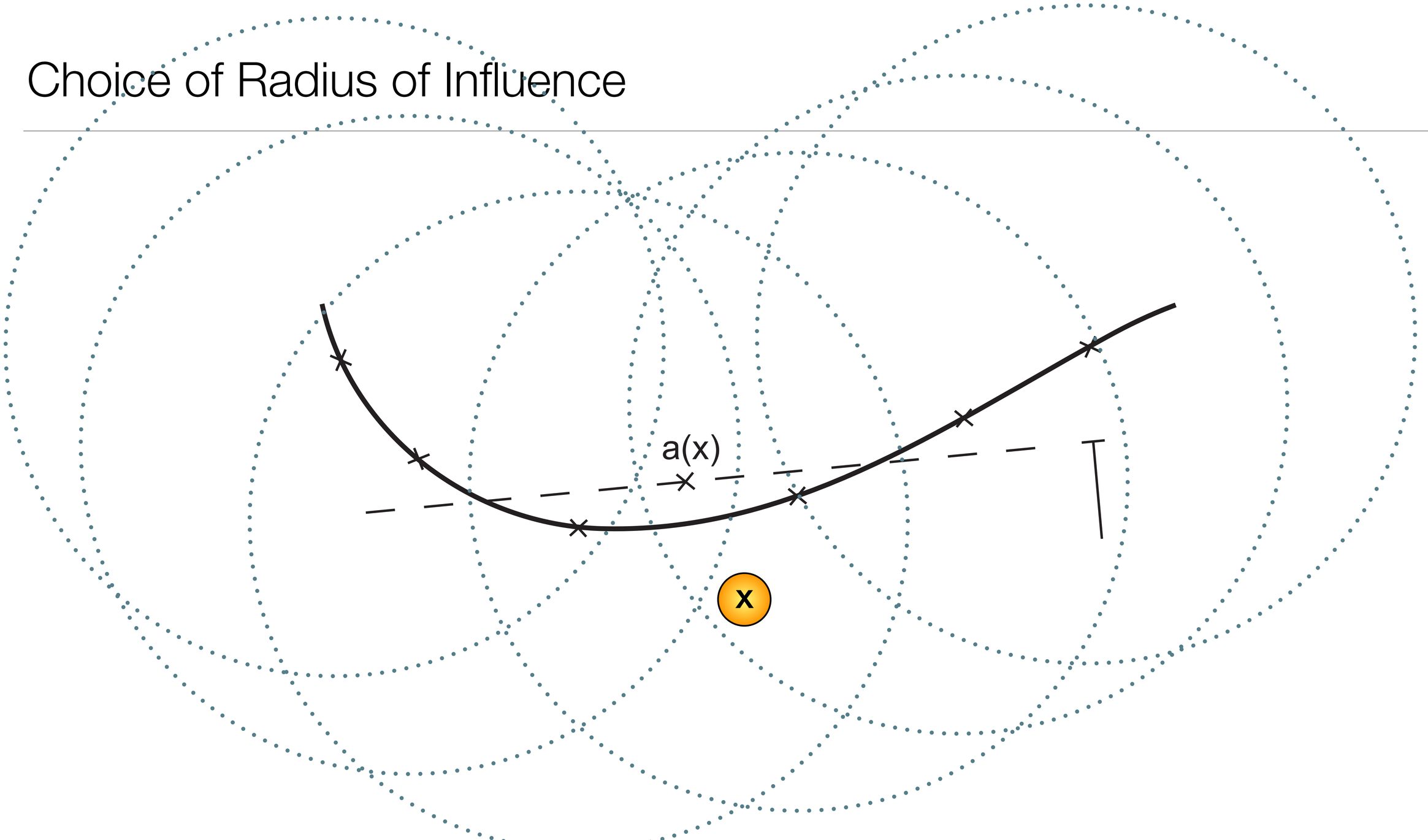
Choice of Radius of Influence



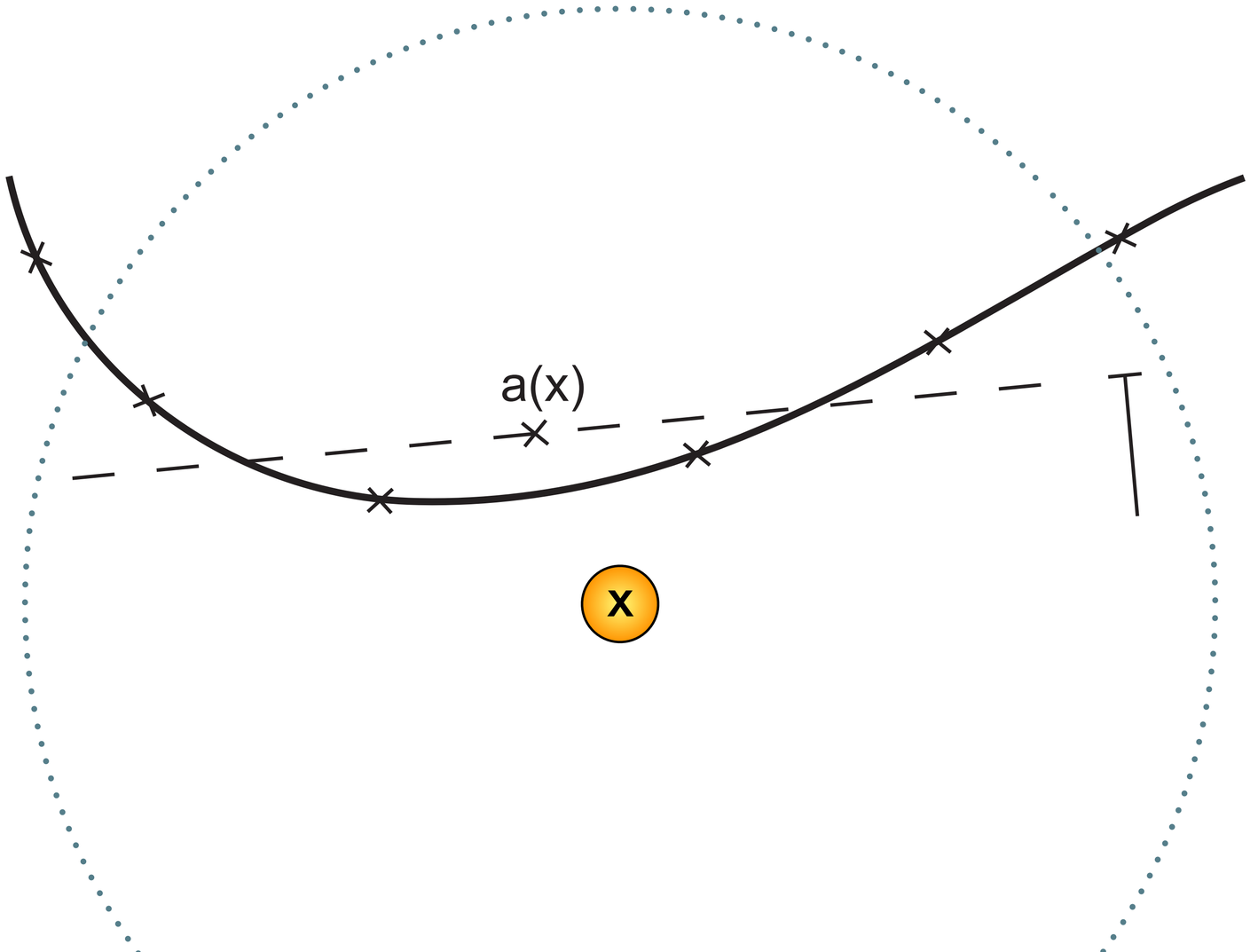
Choice of Radius of Influence



Choice of Radius of Influence



Choice of Radius of Influence



Choice of Radius of Influence

Cyberware Rabbit, 67038 points

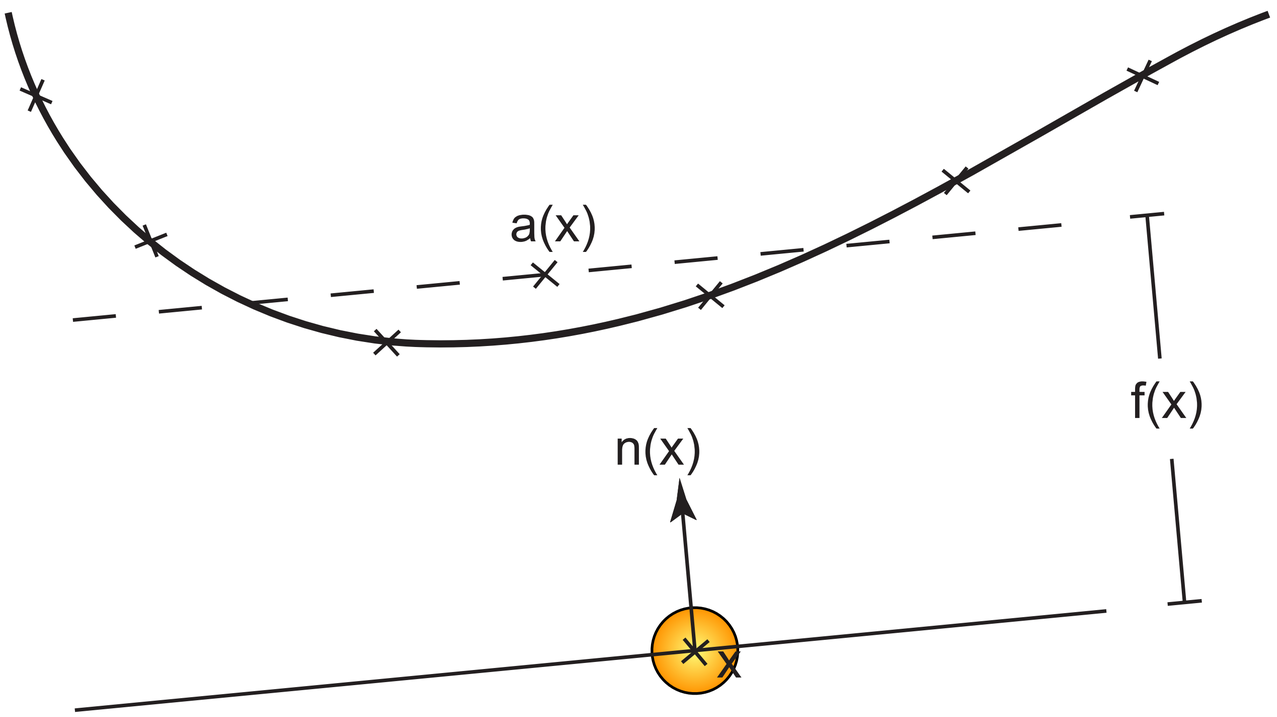
[from A. Adamson & M. Alexa, *Proc. Eurographics Symp. on Geometry Processing*, 2003.]

Point Set Implicit Surface

- We can now define the surface by the implicit function

$$S(x) = \mathbf{n}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{a}(\mathbf{x})) = 0$$

- This surface approximates the original shape if it was *well-sampled* with points
  - *i.e.* If normals are well-defined within a neighbourhood of the surface



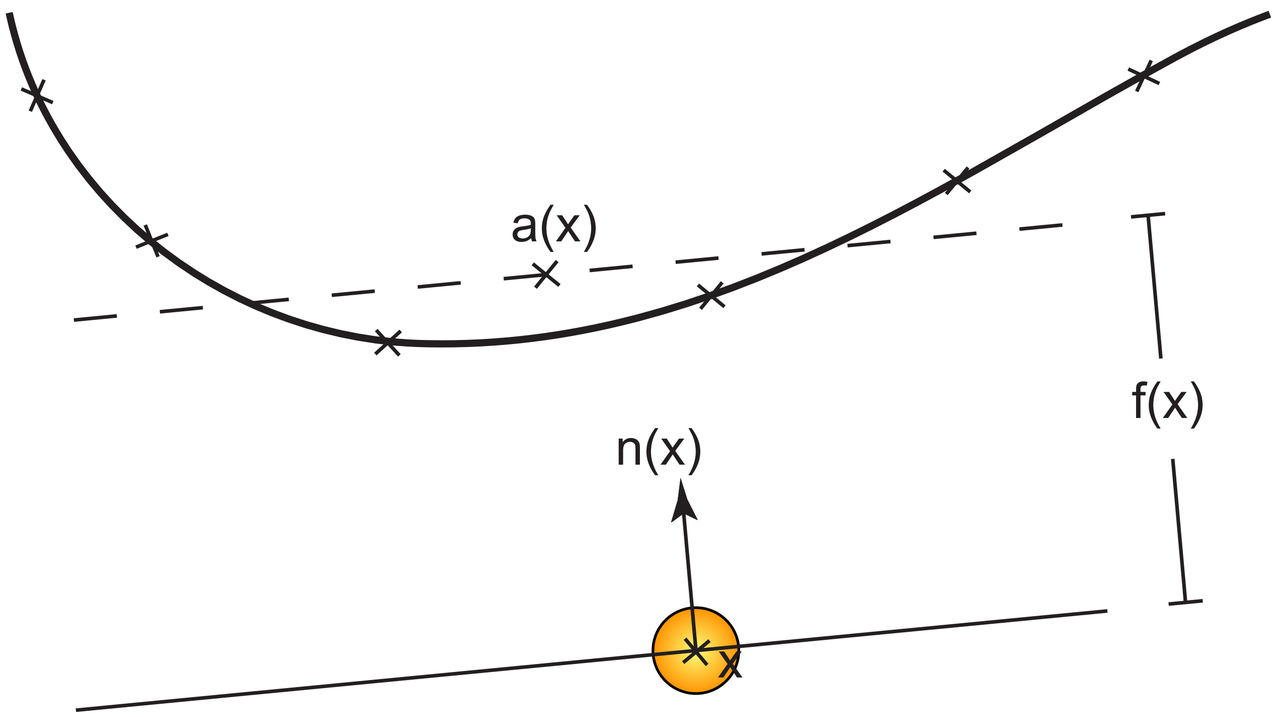
The Gradient?

- Given the surface function

$$S(x) = \mathbf{n}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{a}(\mathbf{x})) = 0$$

- What is the gradient?

$$\nabla S(x) = ?$$



The Gradient?

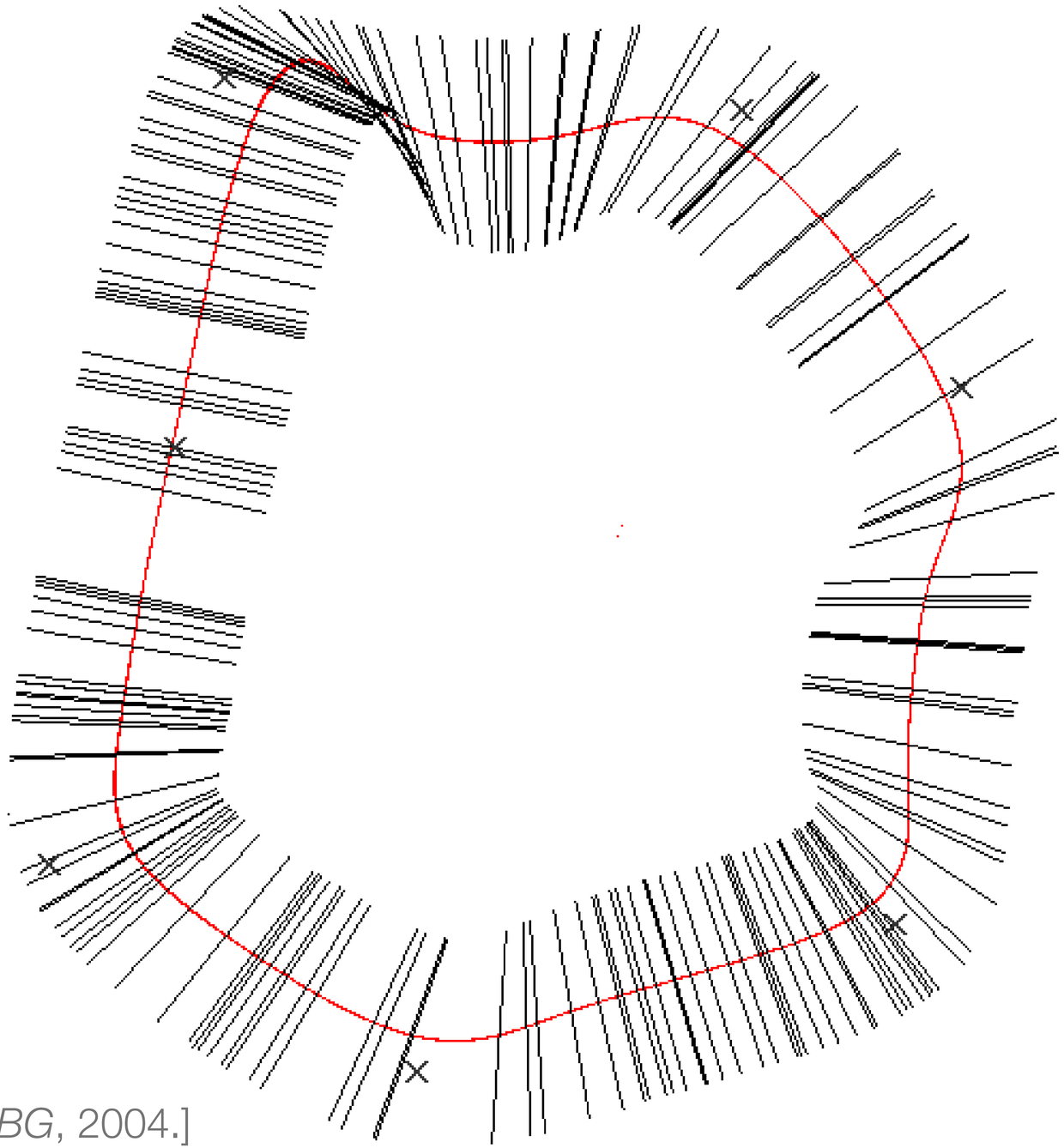
- Given the surface function

$$S(x) = \mathbf{n}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{a}(\mathbf{x})) = 0$$

- What is the gradient?

$$\nabla S(x) = ?$$
$$\nabla S(x) \neq \mathbf{n}(\mathbf{x})$$

  - but it's close enough!



[from M. Alexa & A. Adamson, *Proc. Eurographics Symp. PBG*, 2004.]

Summary

- Point set surfaces are a representation of geometry that is growing in popularity
- They can be haptically rendered by specifying an approximating implicit surface
  - Adamson & Alexa
- If you want to learn more…