Point Cloud Surface Representations

Mark Pauly

2003

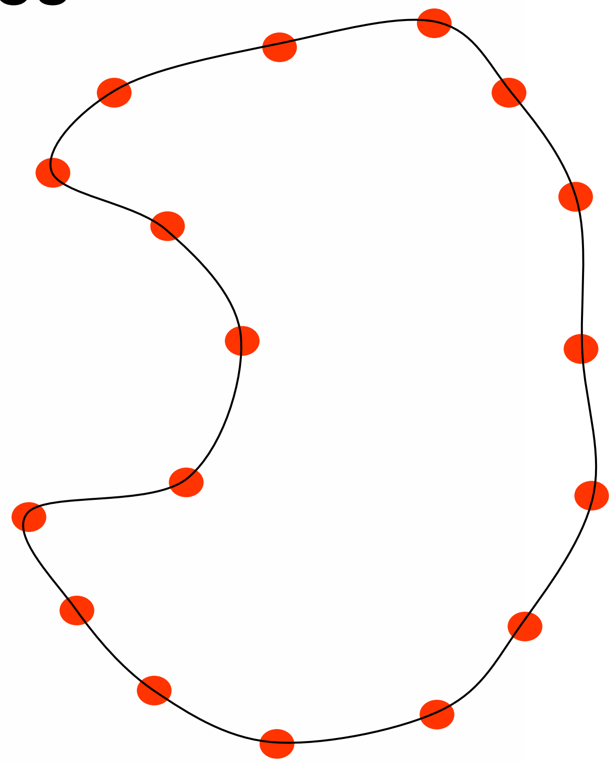
see also EG2003 course on Point-based Computer Graphics available at:
http://graphics.stanford.edu/~mapauly/Pdfs/PointBasedComputerGraphics_EG03.pdf

Papers

- Hoppe, DeRose, Duchamp, McDonald, Stuetzle: **Surface Reconstruction from Unorganized Points**, SIGGRAPH 92
- Carr, Beatson, Cherrie, Mitchell, Fright, McCallum, Evans: **Reconstruction and Representation of 3D Objects with Radial Basis Functions**, SIGGRAPH 01
- Kalaiah, Varshney: **Statistical Point Geometry**, Symposium on Geometry Processing, 2003

Introduction

- Many applications need a definition of surface based on point samples
  - Reduction
  - Up-sampling
  - Interrogation (e.g. ray tracing)
- Desirable surface properties
  - Manifold
  - Smooth
  - Local (efficient computation)

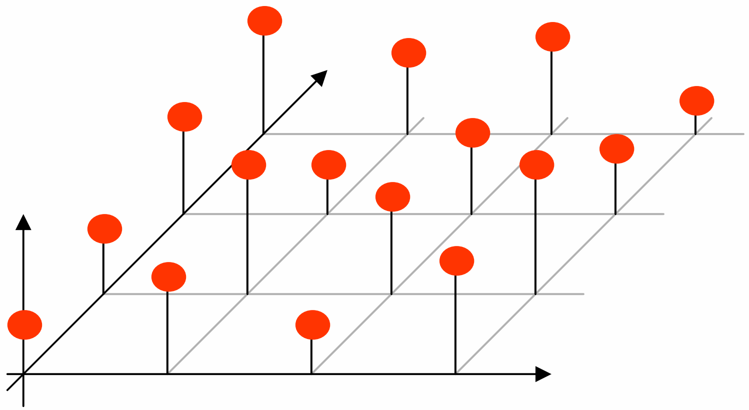


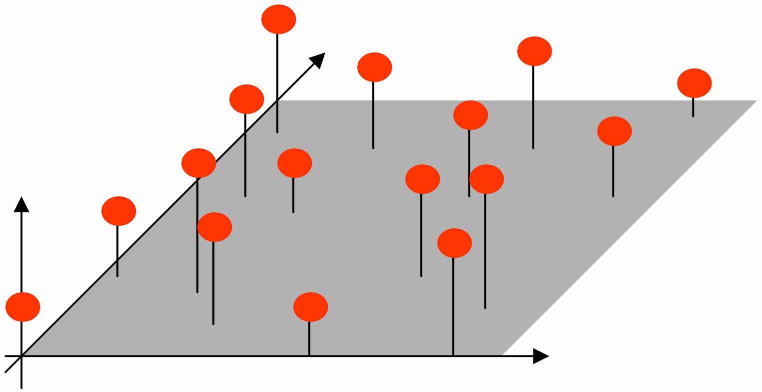
Introduction

- Terms
  - Regular/Irregular, Approximation/Interpolation, Global/Local
- Standard interpolation/approximation techniques
  - Triangulation, Least Squares (LS), Radial Basis Functions (RBF)
- Problems
  - Sharp edges, feature size/noise
- Functional -> Manifold

Terms: Regular/Irregular

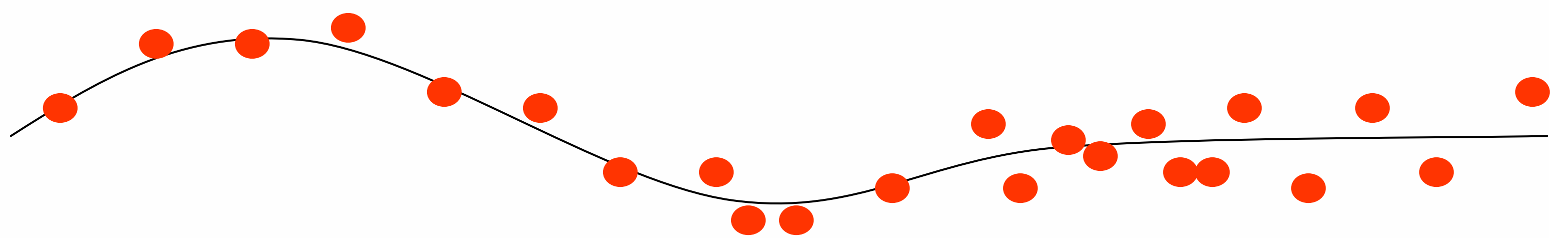
- Regular (on a grid) or irregular (scattered)
- Neighborhood is unclear for irregular data



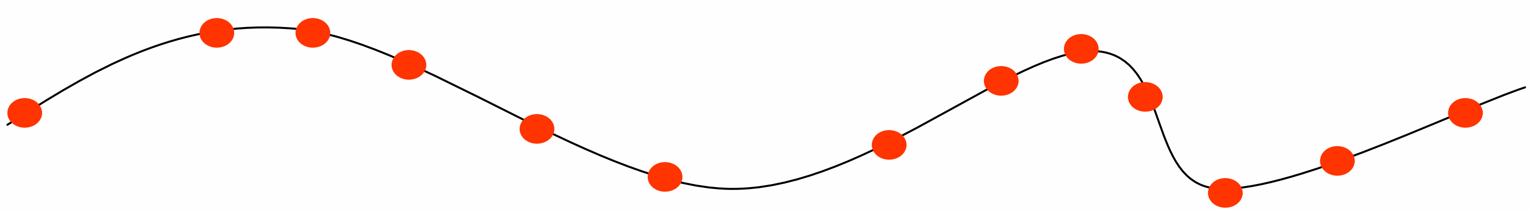


Terms: Approximation/Interpolation

- Noisy data -> Approximation

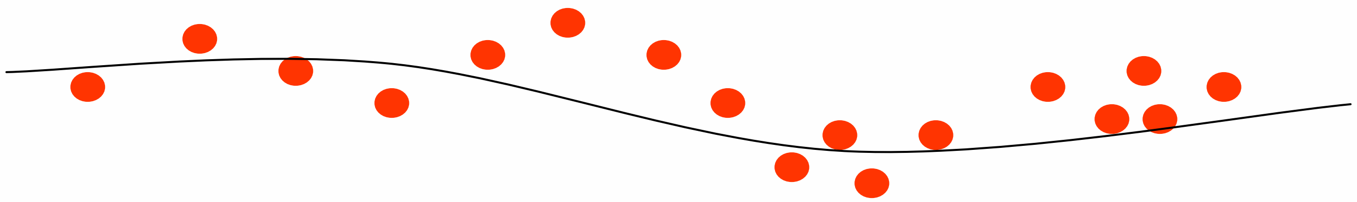


- Perfect data -> Interpolation

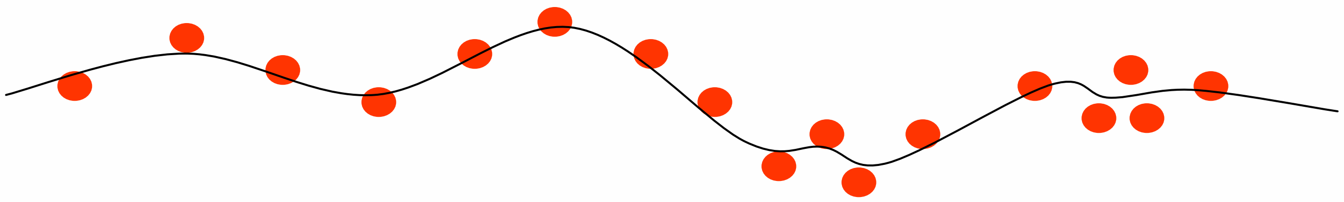


Terms: Global/Local

- Global approximation



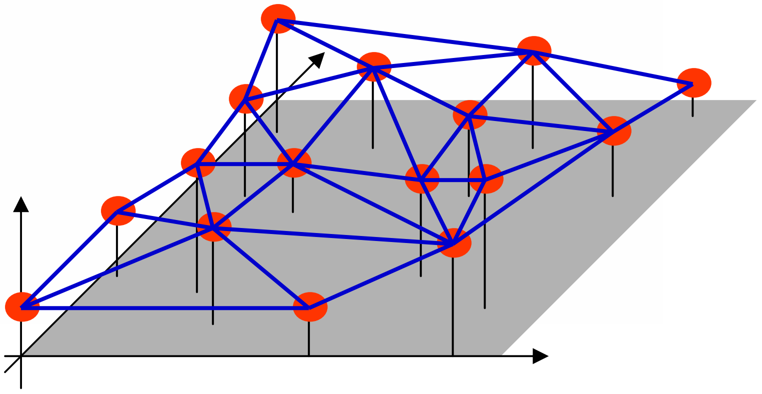
- Local approximation



- Locality comes at the expense of smoothness

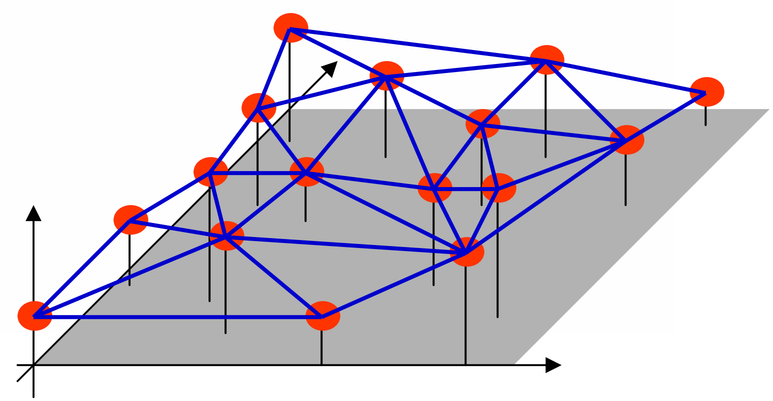
Triangulation

- Exploit the topology in a triangulation (e.g. Delaunay) of the data
- Interpolate the data points on the triangles
  - Piecewise linear → C^0



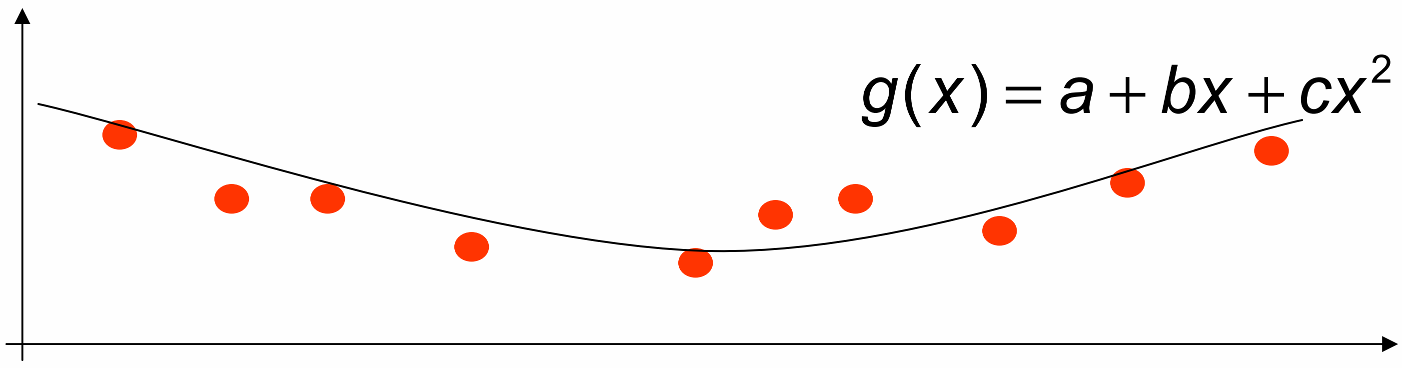
Triangulation: Piecewise linear

- Barycentric interpolation on simplices (triangles)
  - given $d+1$ points x_i with values f_i and a point x inside the simplex defined by x_i
  - Compute α_i from
    $x = \Sigma_i\ \alpha_i \cdot x_i$ and $\Sigma_i\ \alpha_i = 1$
  - Then
    $f = \Sigma_i\ \alpha_i \cdot f_i$



Least Squares

- Fits a primitive to the data
- Minimizes squared distances between the p_i's and primitive g



$$\min_g \sum_i \left(p_{i_y} - g\left(p_{i_x}\right)\right)^2$$

Least Squares - Example

- Primitive is a polynomial

$$g(x) = \left(1, x, x^2, \ldots\right) \cdot \mathbf{c}^T$$

- $$\min \sum_i \left(p_{i_y} - \left(1, p_{i_x}, p_{i_x}^2, \ldots\right)\mathbf{c}^T\right)^2 \Rightarrow$$

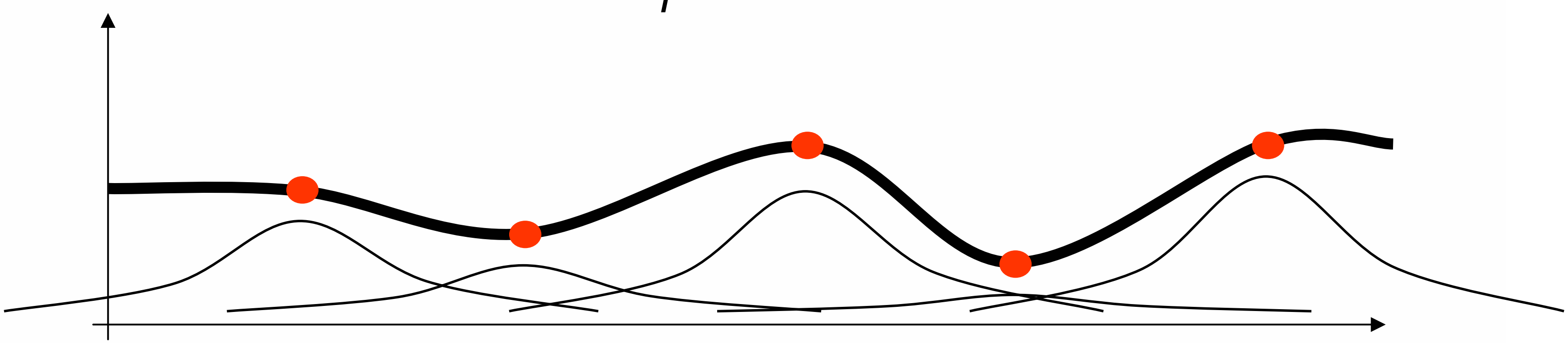
$$0 = \sum_i 2 p_{i_x}^j \left(p_{i_y} - \left(1, p_{i_x}, p_{i_x}^2, \ldots\right)\mathbf{c}^T\right)$$

- Linear system of equations that can be solved using normal equations
- Leads to a system of *dim(**c**)* equations.

Radial Basis Functions

- Represent interpolant as
  - Sum of radial functions r
  - Centered at the data points p_i

$$f(x) = \sum_i w_i r(\|p_i - x\|)$$



Radial Basis Functions

- Solve $p_{j_y} = \sum_i w_i r\left(\left\| p_{i_x} - p_{j_x} \right\|\right)$

  to compute weights w_i
- Linear system of equations

$$\begin{pmatrix} r(0) & r\left(\left\|p_{0_x} - p_{1_x}\right\|\right) & r\left(\left\|p_{0_x} - p_{2_x}\right\|\right) & \cdots \\ r\left(\left\|p_{1_x} - p_{0_x}\right\|\right) & r(0) & r\left(\left\|p_{1_x} - p_{2_x}\right\|\right) & \\ r\left(\left\|p_{2_x} - p_{0_x}\right\|\right) & r\left(\left\|p_{2_x} - p_{1_x}\right\|\right) & r(0) & \\ \vdots & & & \ddots \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} p_{0_y} \\ p_{1_y} \\ p_{2_y} \\ \vdots \end{pmatrix}$$

Radial Basis Functions

- Solvability depends on radial function
- Several choices assure solvability
  - $r(d) = d^2 \log d$ (thin plate spline)
  - $r(d) = e^{-d^2/h^2}$ (Gaussian)
    - h is a data parameter
    - h reflects the feature size or anticipated spacing among points

Interpolation

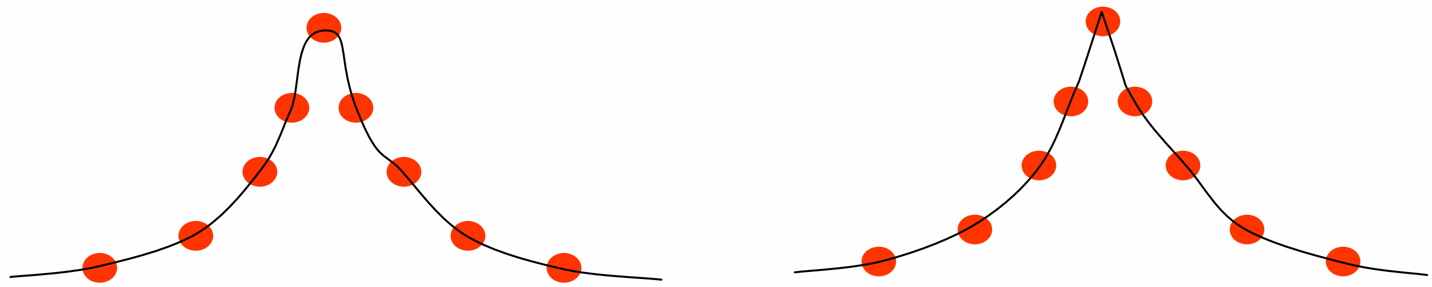
- Monomial, Lagrange, RBF share the same principle:
  - Choose basis of a function space
  - Find weight vector for base elements by solving linear system defined by data points
  - Compute values as linear combinations
- Properties
  - One costly preprocessing step
  - Simple evaluation of function in any point

Interpolation

- Problems
  - Many points lead to large linear systems
  - Evaluation requires global solutions
- Solutions
  - RBF with compact support
    - Matrix is sparse
    - Still: solution depends on every data point, though drop-off is exponential with distance
  - Local approximation approaches

Typical Problems

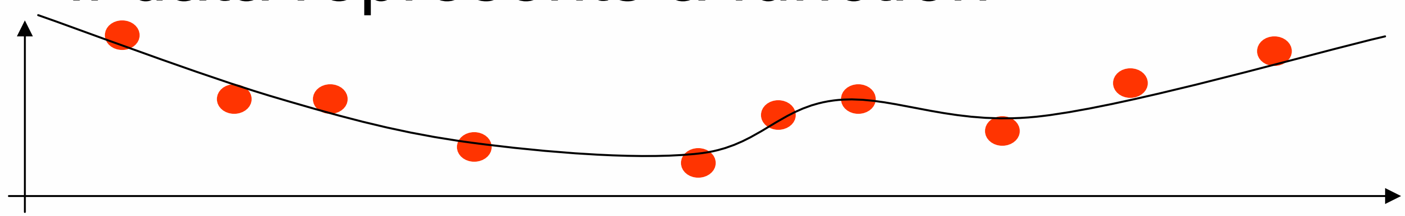
- Sharp corners/edges



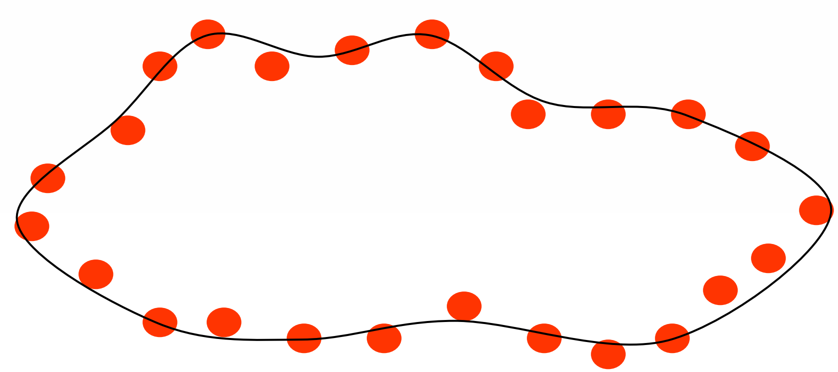
- Noise vs. feature size

Functional -> Manifold

- Standard techniques are applicable if data represents a function

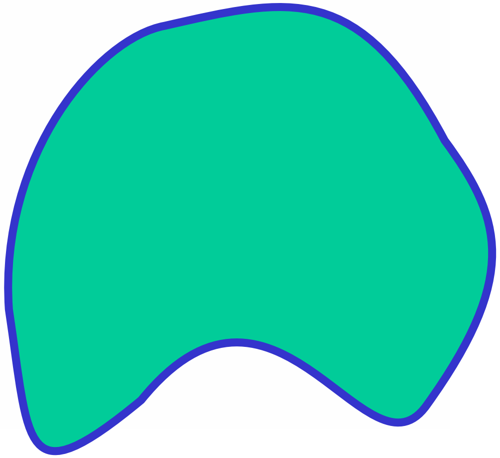


- Manifolds are more general
  - No parameter domain
  - No knowledge about neighbors

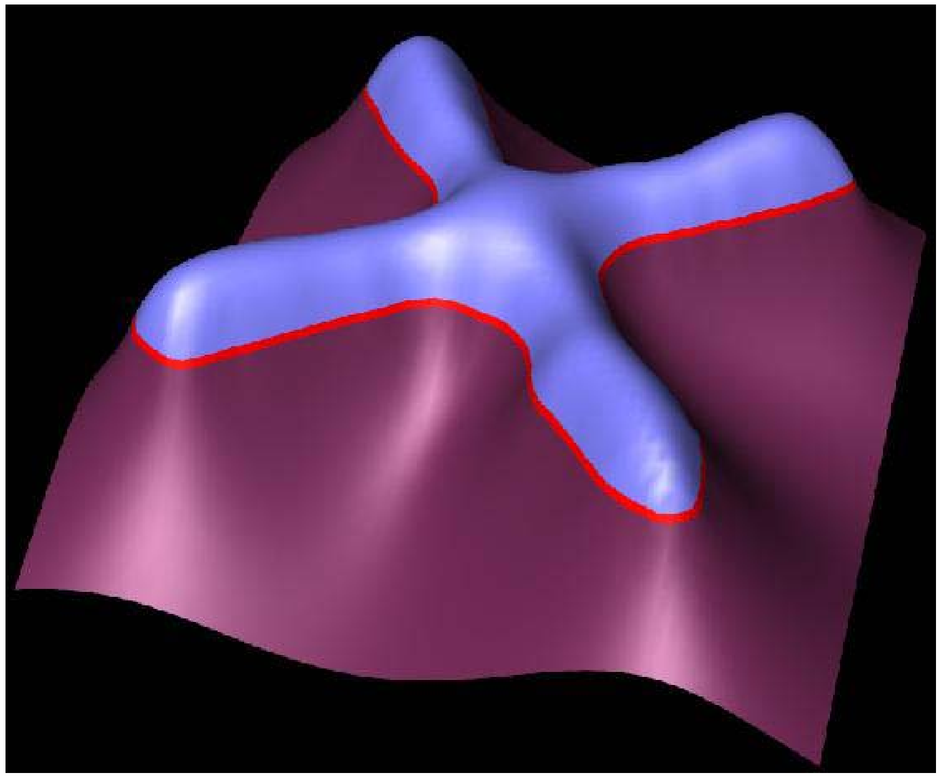


Implicits

- Each orientable n-manifold can be embedded in n+1 – space
- Idea: Represent n-manifold as zero-set of a scalar function in n+1 – space
  - Inside: $f(\mathbf{x}) < 0$
  - On the manifold: $f(\mathbf{x}) = 0$
  - Outside: $f(\mathbf{x}) > 0$



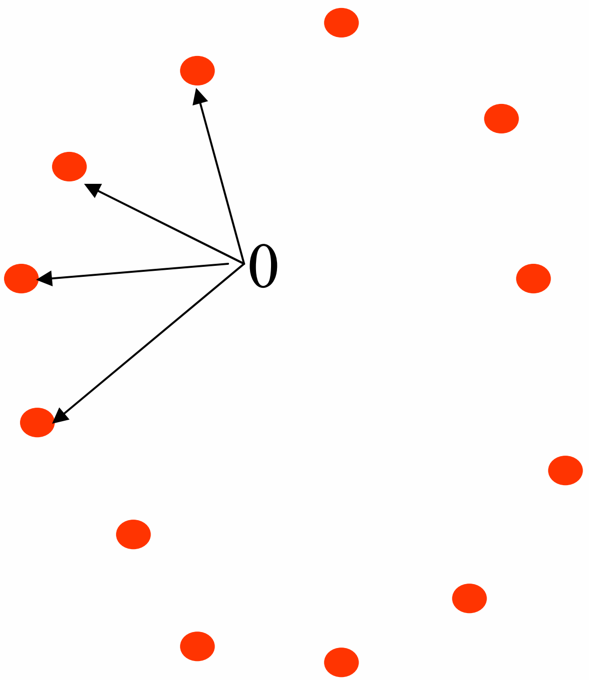
Implicits - Illustration



- Image courtesy Greg Turk

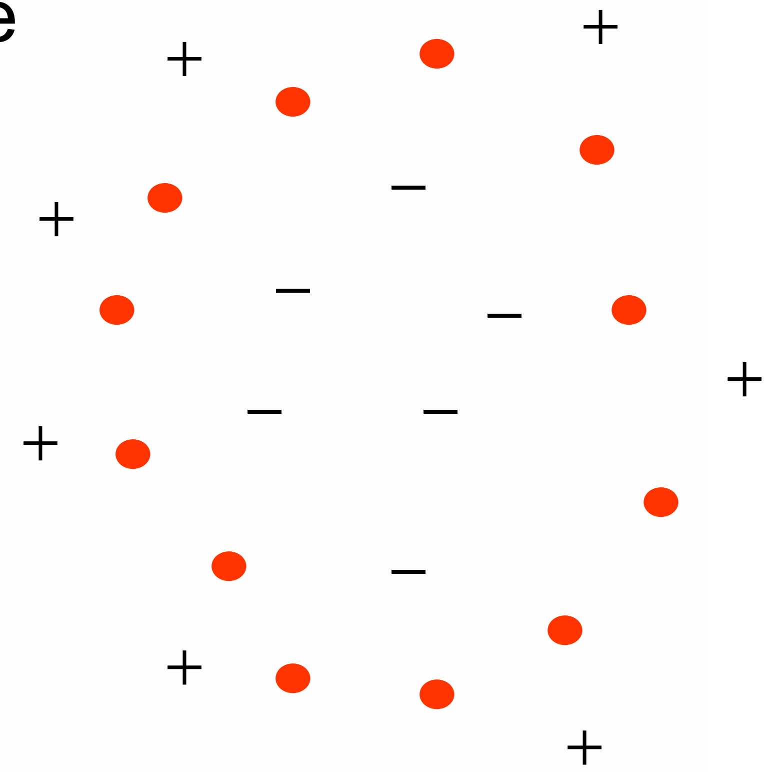
Implicits from point samples

- Function should be zero in data points
    - $f(\mathbf{p}_i) = 0$
- Use standard approximation techniques to find f
- Trivial solution: $f = 0$
- Additional constraints are needed



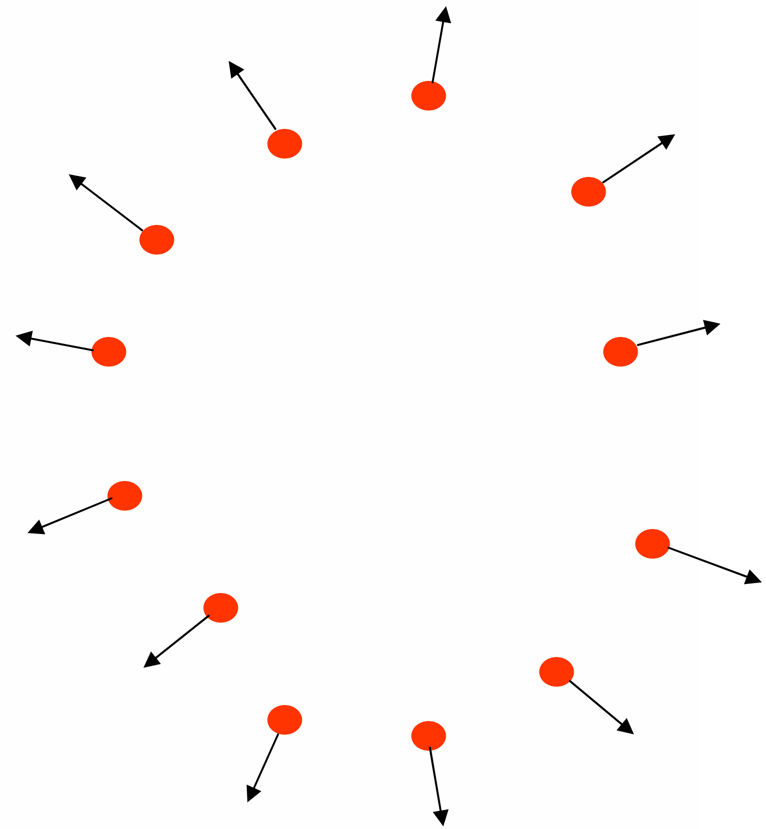
Implicits from point samples

- Constraints define inside and outside
- Simple approach (Turk, O'Brien)
  - Sprinkle additional information manually
  - Make additional information soft constraints



Implicits from point samples

- Use normal information
- Normals could be computed from scan
- Or, normals have to be estimated

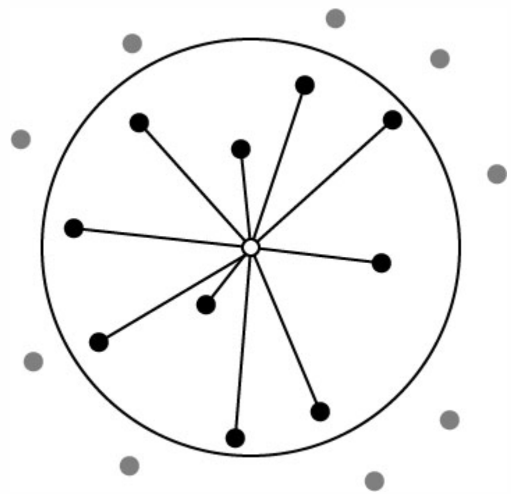


Detour: Local Surface Analysis

- Estimate local surface properties from local neighborhoods:
  - No explicit connectivity between samples (as with triangle meshes)
  - Replace geodesic proximity with spatial proximity (requires sufficiently high sampling density!)
  - Compute neighborhood according to Euclidean distance

Neighborhood

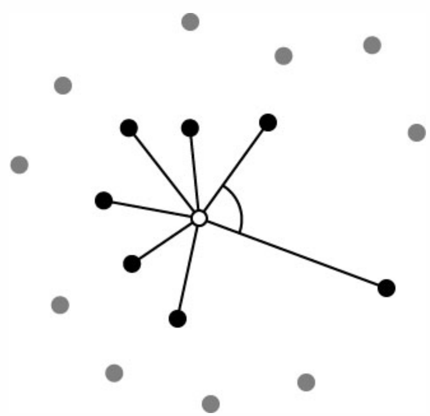
- K-nearest neighbors



  - Can be quickly computed using spatial data-structures (e.g. kd-tree, octree, bsp-tree)
  - Requires isotropic point distribution

Neighborhood

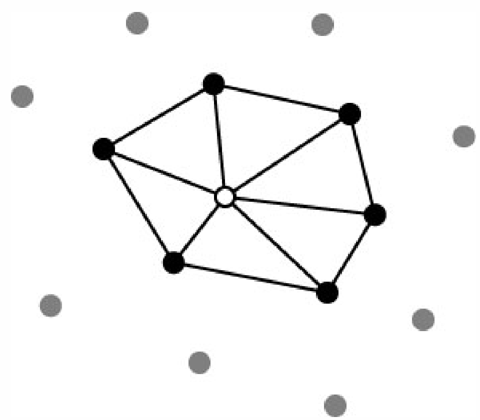
- Improvement: Angle criterion (Linsen)



  - Project points onto tangent plane
  - Sort neighbors according to angle
  - Include more points if angle between subsequent points is above some threshold

Neighborhood

- Local Delaunay triangulation (Floater)



  - Project points into tangent plane
  - Compute local Voronoi diagram

Covariance Analysis

- Covariance matrix of local neighborhood N:

$$\mathbf{C} = \begin{bmatrix} \mathbf{p}_{i_1} - \overline{\mathbf{p}} \\ \cdots \\ \mathbf{p}_{i_n} - \overline{\mathbf{p}} \end{bmatrix}^T \cdot \begin{bmatrix} \mathbf{p}_{i_1} - \overline{\mathbf{p}} \\ \cdots \\ \mathbf{p}_{i_n} - \overline{\mathbf{p}} \end{bmatrix}, \qquad i_j \in N$$

- with centroid $\overline{\mathbf{p}} = \frac{1}{|N|} \sum_{i \in N} \mathbf{p}_i$

Covariance Analysis

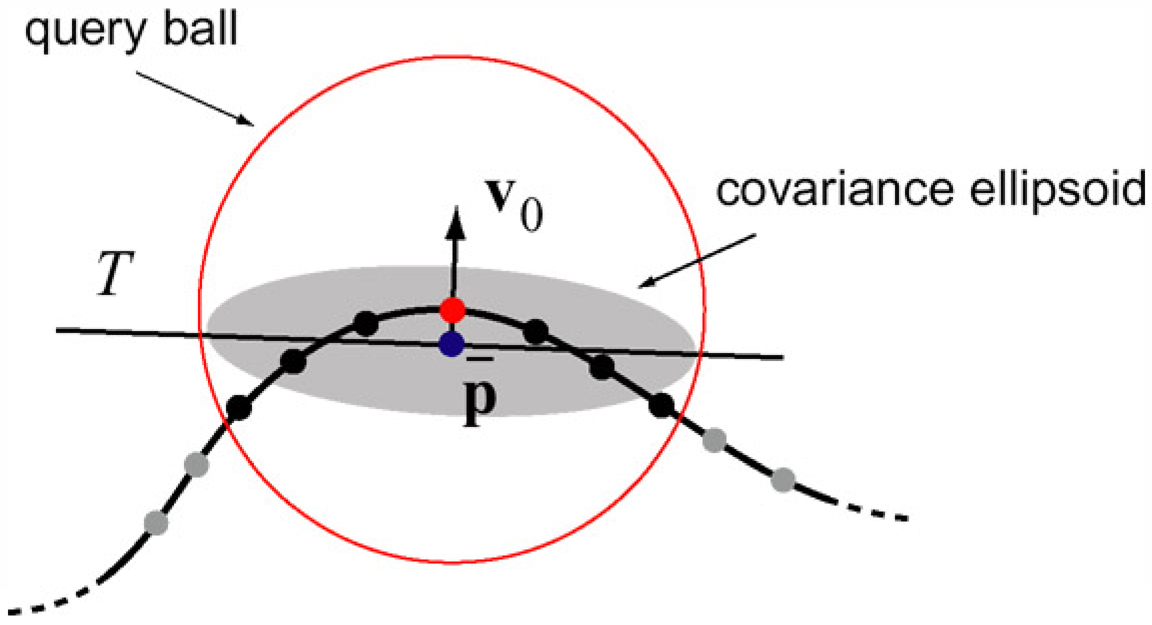
- Consider the eigenproblem:

$$\mathbf{C} \cdot \mathbf{v}_l = \lambda_l \cdot \mathbf{v}_l, \qquad l \in \{0,1,2\}$$

- C is a 3x3, positive semi-definite matrix
  - ⇨ All eigenvalues are real-valued
  - ⇨ The eigenvector with smallest eigenvalue defines the least-squares plane through the points in the neighborhood, i.e. approximates the surface normal

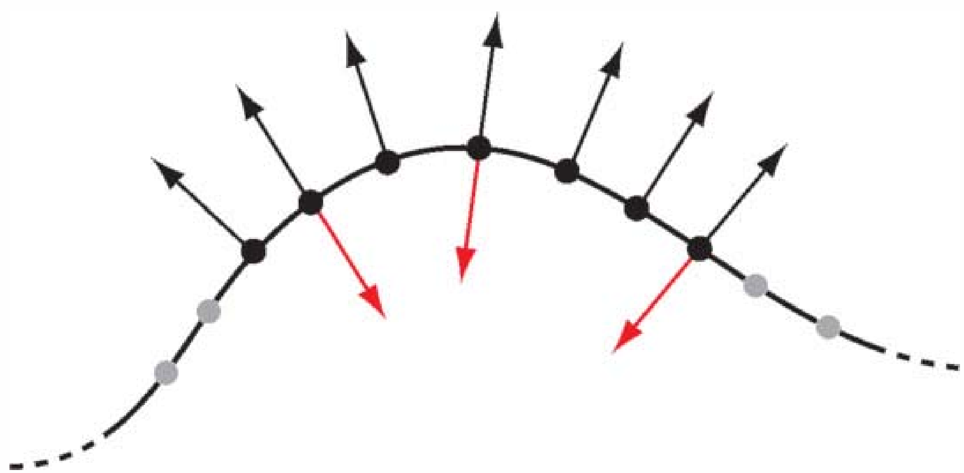
Covariance Analysis

- Covariance ellipsoid spanned by the eigenvectors scaled with corresponding eigenvalue



Normal Estimation

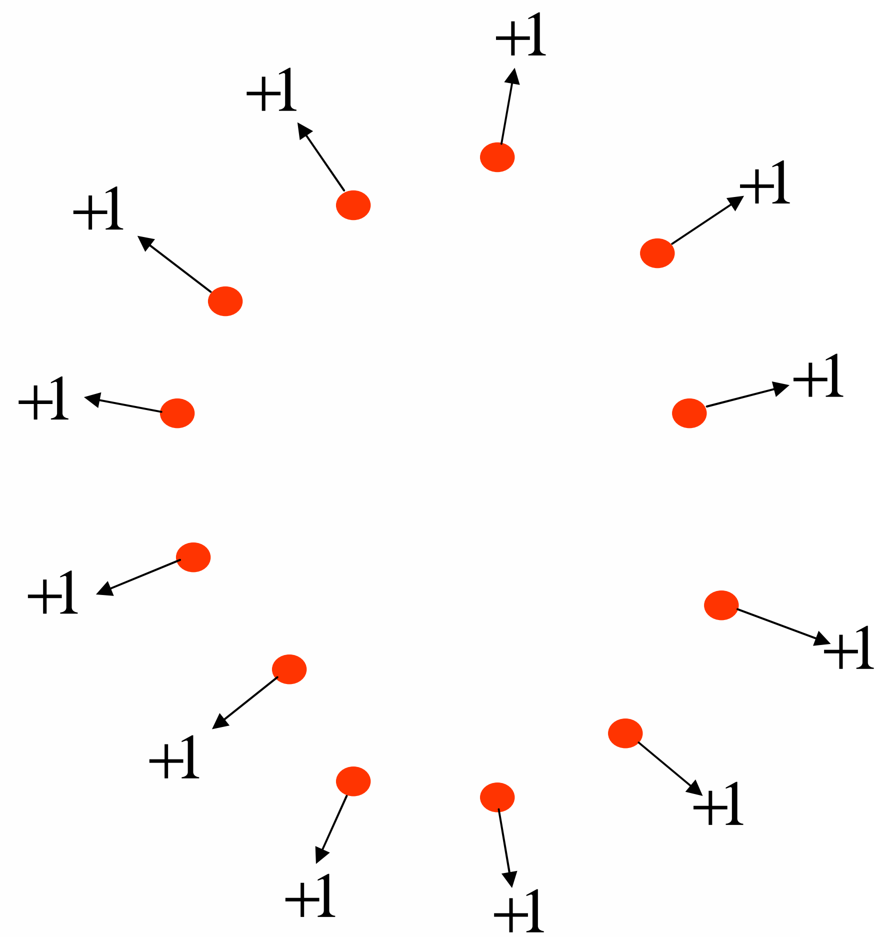
- Estimate normal direction by least squares fit
- Compute consistent orientation by incremental propagation



Implicits from point samples

- Compute non-zero anchors in the distance field
- Use normal information directly as constraints

$$f(\mathbf{p}_i + \mathbf{n}_i) = 1$$



Implicits from point samples

- need to constrain distance to avoid self-intersections

$$f\left(\mathbf{p}_i + d_i \mathbf{n}_i\right) = 1$$

Computing Implicits

- Given N points and normals p_i, n_i and constraints

$$f(\mathbf{p}_i) = 0, f(\mathbf{c}_i) = d_i$$

- Let $\mathbf{p}_{i+N} = \mathbf{c}_i$
- An RBF approximation

$$f(\mathbf{x}) = \sum_i w_i r(\|\mathbf{p}_i - \mathbf{x}\|)$$

- leads to a system of linear equations

Computing Implicits

- Practical problems: N > 10000
- Matrix solution becomes difficult
- Different solutions
  - Sparse matrices allow iterative solution
  - Fast multi-pole methods
  - Smaller number of RBFs

Computing Implicits

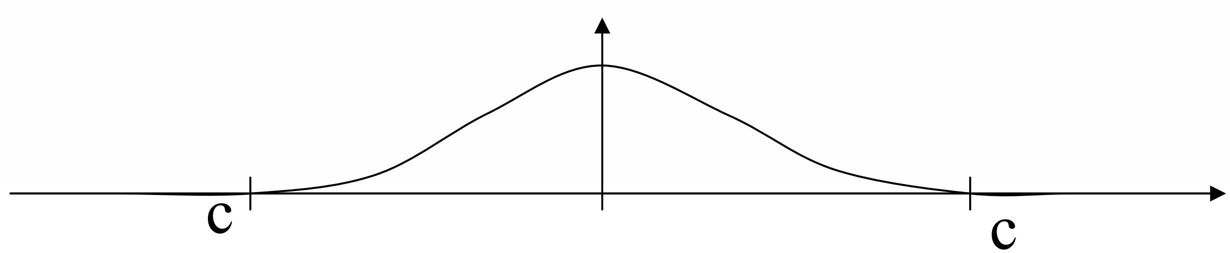
- Sparse matrices

$$\begin{pmatrix} r(0) & r(\|p_0 - p_1\|) & r(\|p_0 - p_2\|) & \cdots \\ r(\|p_1 - p_0\|) & r(0) & r(\|p_1 - p_2\|) & \\ r(\|p - p_0\|) & r(\|p_2 - p_1\|) & r(0) & \\ \vdots & & & \ddots \end{pmatrix}$$

  - Needed:

$$d > c - \quad r(d) = 0, r'(c) = 0$$

  - Compactly supported RBFs



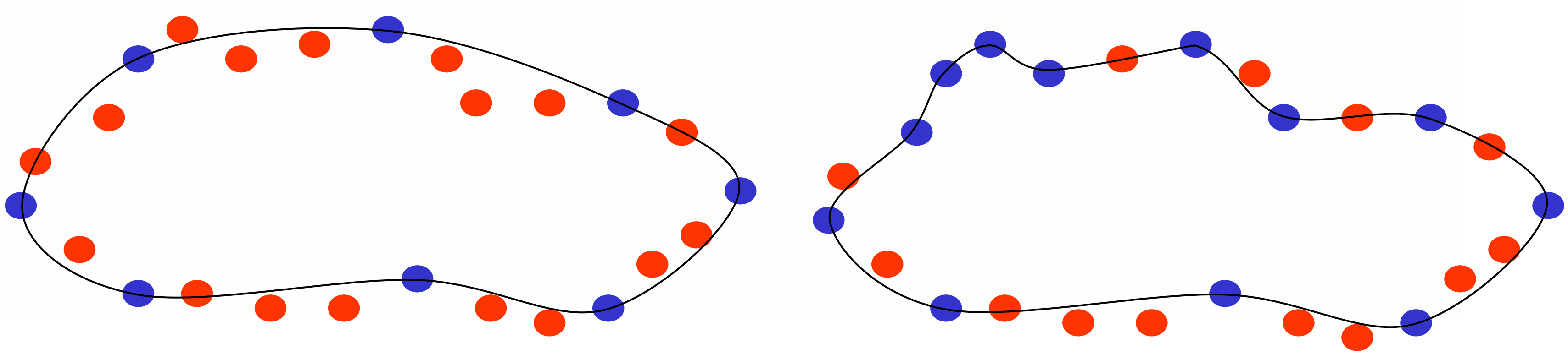
Computing Implicits

- Fast multi-pole methods
  - approximate solution using far- and near-field expansion
  - hierarchical clustering of nodes
  - introduces fitting error and evaluation error

	Direct Methods	Fast Methods
Storage	*O(N^2)*	*O(N)*
Solve system	*O(N^3)*	*O(NlogN)*
Evaluation	*O(N)*	*O(1) + O(NlogN)* setup

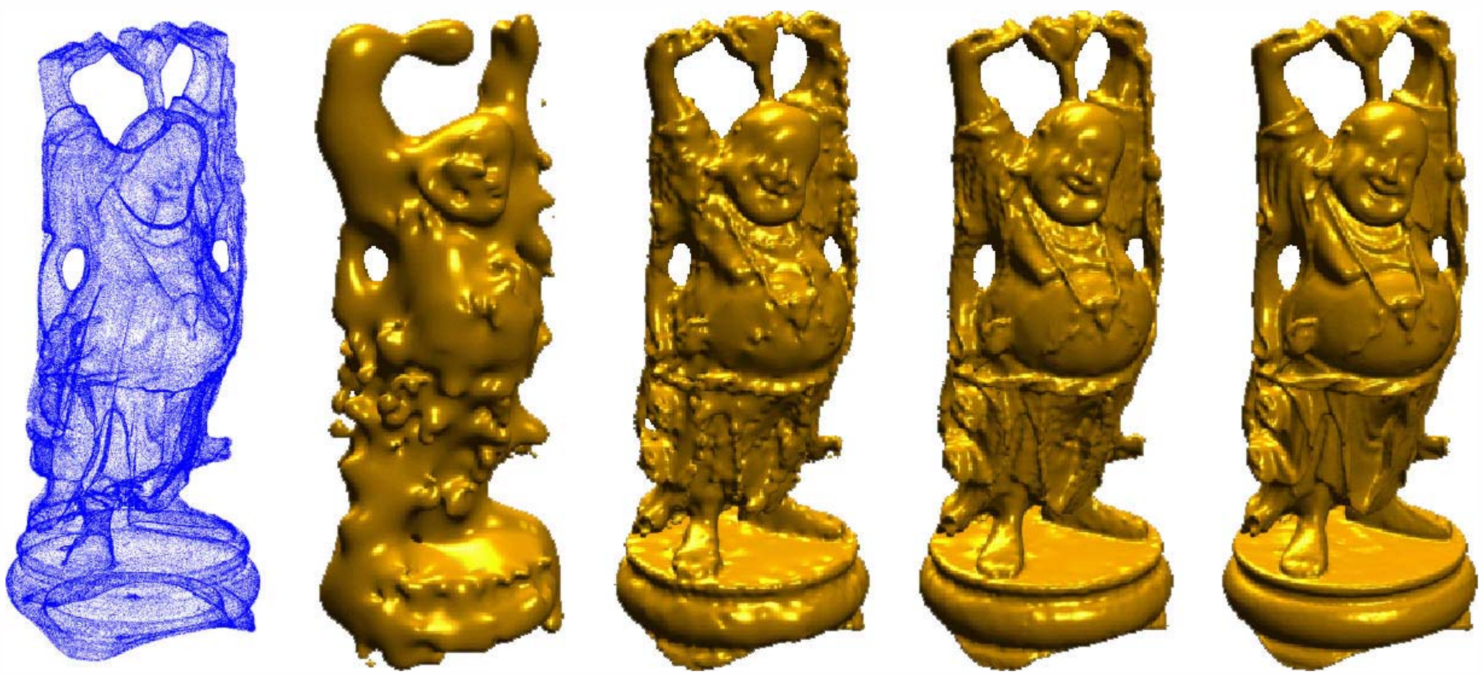
Computing Implicits

- RBF center reduction exploits the redundancy in many point sampled models
- Greedy approach (Carr et al.)
  - Start with random small subset
  - Add RBFs where approximation quality is not sufficient



Computing Implicits

- RBF center reduction: Example



Implicits - Conclusions

- Scalar field is underconstrained
  - Constraints only define where the field is zero, not where it is non-zero
  - Additional constraints are needed
- Signed fields restrict surfaces to be unbounded
  - All implicit surfaces define solids

Paper

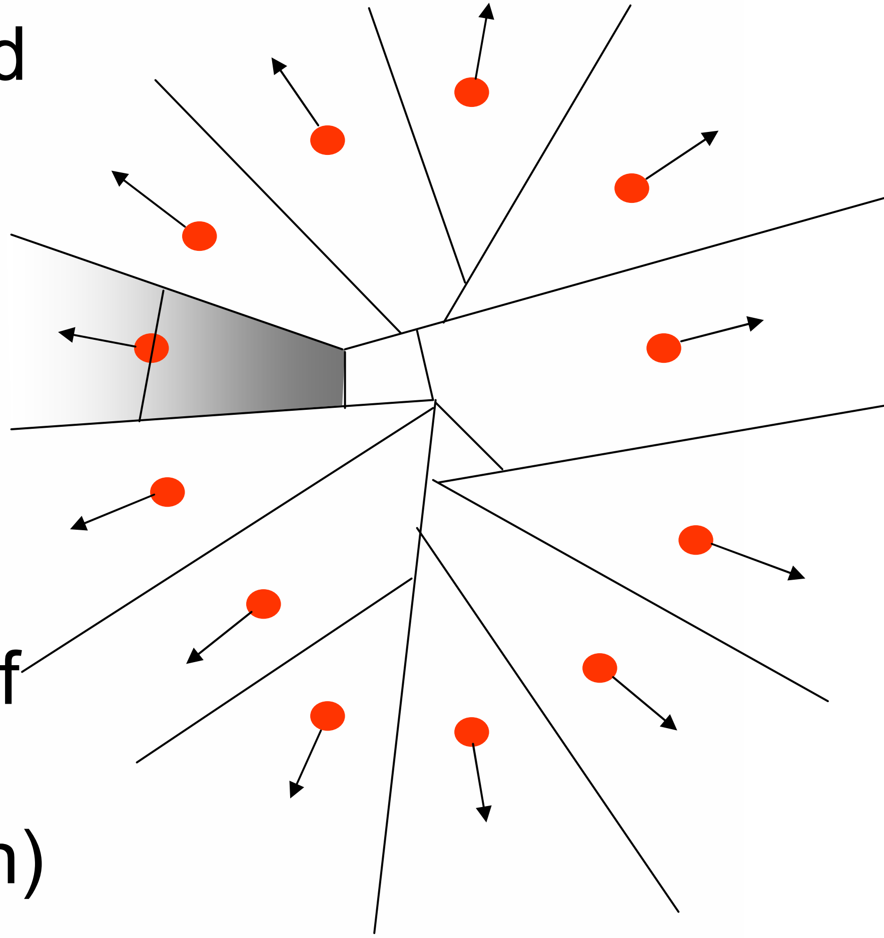
- Hoppe, DeRose, Duchamp, McDonald, Stuetzle: **Surface Reconstruction from Unorganized Points**, SIGGRAPH 92

Summary

- Goal:
  - Reconstruct polygonal surface from unorganized set of point samples
- Approach:
  - Approximate signed distance function
  - Use contouring method (marching cubes) to extract triangle mesh

More Details

- Use linear distance field per point
  - Direction is defined by normal
  - Normal estimated using covariance analysis
- In every point in space use the distance field of the closest point (Voronoi decomposition)



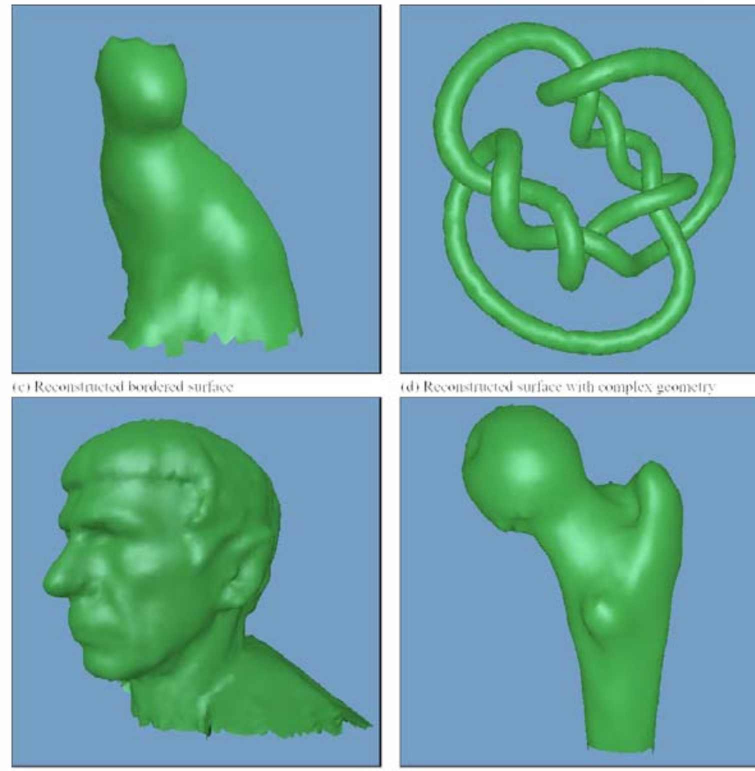
More Details

- $X=\{x_0,..,.x_n\}$ sample of an unknown surface S
- δ-noisy: $x_i = y_i + e_i$, y_i on S, $|e_i| < \delta$
- ρ-dense: Any sphere with radius ρ and center on S contains at least one sampe x_i

  ⇨ justification for using k-nearest neighbors
- Algorithm complexity:
  - k-nearest neighbors: *O(k*logN)*
  - normal orientation: *O(NlogN)*
  - contouring: *O(m), m* = #visited cubes

Results

+ shapes of arbitrary topology
+ simple and efficient computation
- crude approximation of signed distance field
- no topological guarantees



(c) Reconstructed bordered surface

(d) Reconstructed surface with complex geometry

(e) Reconstruction from cylindrical range data

(f) Reconstruction from contour data

Paper

- Carr, Beatson, Cherrie, Mitchell, Fright, McCallum, Evans: **Reconstruction and Representation of 3D Objects with Radial Basis Functions**, SIGGRAPH 01

Summary

- Goal:
  - Reconstruct implicit surface from unorganized point set
- Approach:
  - RBF implicit representation
  - Fast computation of matrix solution using multi-pole method and RBF center reduction
  - RBF approximation of noisy data

More Details

- RBF interpolation
  - $s(x_i)=f_i,\ i=1,\dots,N$
  - additional constraints using normal information
  - "smoothest" interpolant: $s^* = \arg\min_{s \in S} \|s\|$ according to rotation-invariant semi-norm ||.||
  - for noisy surface look for least-squares approximation

$$\min_s \rho \|s\|^2 + \frac{1}{N} \sum_{i=1}^{N} (s(x_i) - f_i)^2$$

More Details

- RBF center reduction
  1. Choose subset of nodes and fit RBF *s(x)*
  2. Evaluate residual $e_i = f_i - s(x_i)$ for all x_i
  3. If $\{max \{|e_i|\} <$ *fitting accuracy*, stop
  4. else append new centers where e_i is large
  5. recompute *s(x)* and goto 2

Results

+ Reconstruction from large point sets
+ Irregular sampling distributions
+ Smooth extrapolation for hole filling

Results

- Smoothing operation does not preserve features
- Still relatively slow: Fitting time in order of hours, surface time in order of minutes

Paper

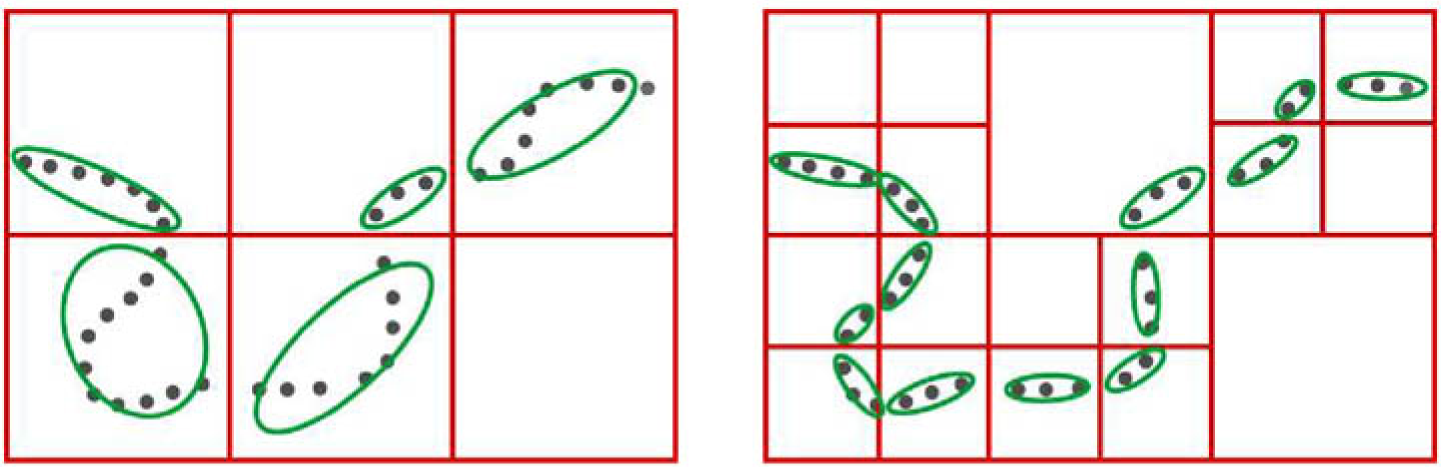
- Kalaiah, Varshney: **Statistical Point Geometry**, Symposium on Geometry Processing, 2003

Summary

- Goal:
  - Efficiently represent point clouds using statistical methods
- Approach:
  - Octree subdivision
  - PCA on positions, normals, and color
  - k-means clustering and quantization

More Details

- Subdivide point cloud into clusters using octree hierarchy

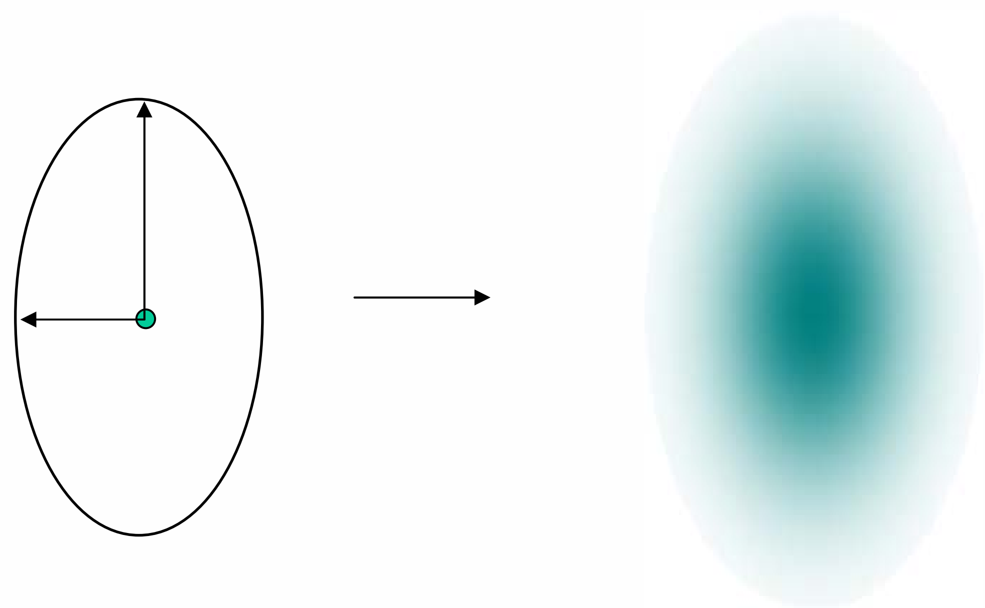


More Details

- Apply principal component analysis (PCA) on each cluster (covariance analysis)
- Treat positions, normals, colors separately
- Represent each cluster by mean + covariance ellipsoid
- Collection of ellipsoids provides statistical representation of original point cloud

More Details

- Application: Randomized Rendering
  - sample PCA ellipsoids using trivariate Gaussian



PCA ellipsoid

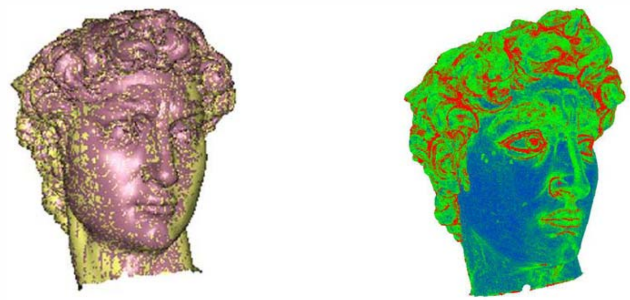
Gaussian random distribution

More Details

- Randomized rendering

More Details

- Compression:



Octree Level	Compression (in %)	Max. error (in %)	Avg. error (in %)
Level 9	213.79	0.4161	0.0751
Level 8	829.23	0.546	0.1466
Level 7	3392.11	1.186	0.280
Level 6	13316.08	1.952	0.641
Level 5	45433.10	3.768	1.504
Level 4	104477.13	8.086	3.458

Results

+ Statistical approach well suited for large models
+ Can handle (some) noise
- Decoupling of position and normals leads to inferior rendering quality (no coherence)
- Compression (probably) not competitive
- Hard to apply interrogation or other operators using this representation