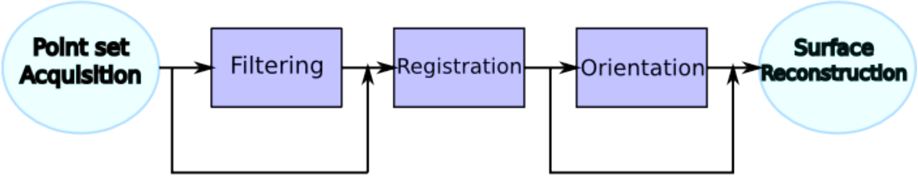
Processing Point Set Surfaces

Julie Digne

LIRIS - Équipe GeoMod - CNRS

12/09/2018

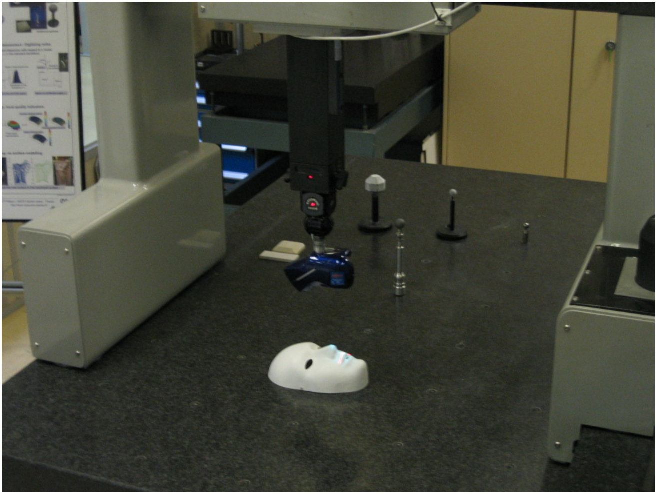
Introduction



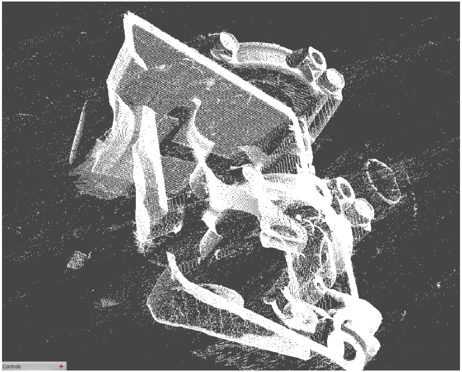
- Each of these blocks is a challenge!
- Sampling of the existing methods

Thanks to Pierre Alliez and Misha Kazhdan for providing some of the slides.

Introduction: Acquisition of point clouds

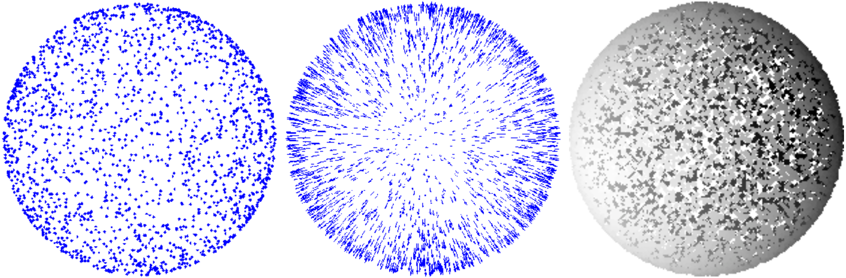


3d surfaces typical challenges: Cleaning the physical measure

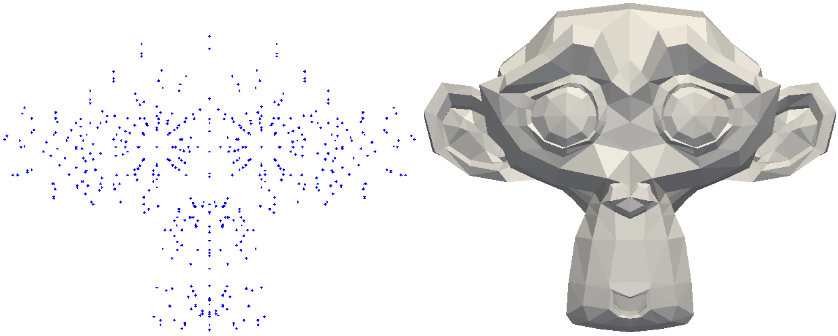


3d surfaces typical challenges: Registering and merging scans

3d surfaces typical challenges: Orienting the point set

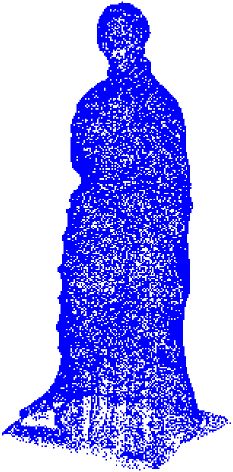


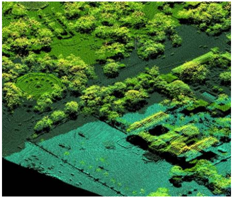
3d surfaces typical challenges: Building a mesh from a set of points

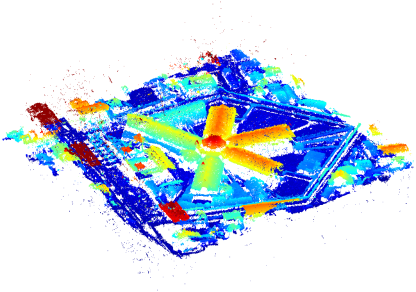


Shape courtesy of *blender*

Results of the acquisition process







Outline

1. Geometry Processing basics
2. Surface reconstruction: Methods from Computational Geometry
3. Surface Reconstruction: Potential Field Methods

Riemannian surface definition

Riemann Surface

A Riemann surface S is a separated (Hausdorff) topological space endowed with an atlas: For every point $x \in S$ there is a neighborhood $V(x)$ containing x homeomorphic to the unit disk of the complex plane. These homeomorphisms are called charts. The transition maps between two overlapping charts are required to be holomorphic.

Riemannian surface definition

Riemann Surface

A Riemann surface S is a separated (Hausdorff) topological space endowed with an atlas: For every point $x \in S$ there is a neighborhood $V(x)$ containing x homeomorphic to the unit disk of the complex plane. These homeomorphisms are called charts. The transition maps between two overlapping charts are required to be holomorphic.

- At each point of the surface one can find an intrinsic parameterization $T(u, v)$.

Riemannian surface definition

Riemann Surface

A Riemann surface S is a separated (Hausdorff) topological space endowed with an atlas: For every point $x \in S$ there is a neighborhood $V(x)$ containing x homeomorphic to the unit disk of the complex plane. These homeomorphisms are called charts. The transition maps between two overlapping charts are required to be holomorphic.

- At each point of the surface one can find an intrinsic parameterization $T(u, v)$.
- We restrict this small introduction to surfaces of dimension 2 embedded in $\mathbb{R}^3$.

Let $\mathcal{S}$ be a smooth surface embedded in $\mathbb{R}^3$, parameterized over a bounded domain $\Omega \subset \mathbb{R}^2$ with parameterization:

$$\mathbf{x}(u, v) = \begin{pmatrix} x(u, v) \\ y(u, v) \\ z(u, v) \end{pmatrix}$$

- Define $\mathbf{x}_u(u_0, v_0) = \frac{\partial \mathbf{x}}{\partial u}(u_0, v_0)$
- $\mathbf{x}_v(u_0, v_0) = \frac{\partial \mathbf{x}}{\partial v}(u_0, v_0)$ is tangent to the curve on the surface defined by $s \rightarrow \mathbf{x}(u_0, v_0 + s)$.
- $\mathbf{x}_u(u_0, v_0)$ and $\mathbf{x}_v(u_0, v_0)$ are two vectors tangent to the surface $\mathcal{S}$.
- If the parameterization is *regular*, ($\|x_u \times x_v\| \neq 0$), these vectors span the tangent plane to the surface at $\mathbf{x}(u_0, x_0)$.

Normal computation

- If the parameterization is *regular*, the normal to the surface is computed as:

$$\mathbf{n} = \frac{x_u \times x_v}{\|x_u \times x_v\|}$$

- *Directional derivatives* Given a direction w in the tangent plane, the directional derivative of $\mathcal{S}$ in direction w is the tangent to the curve $C_w(t) = x(u_0, v_0 + tw)$

First Fundamental Form

Definition (First Fundamental Form)

The **First Fundamental Form** is defined as $I = J \cdot J^T$ (2×2 matrix). or equivalently:

$$I = \begin{pmatrix} \mathbf{x}_u^T \mathbf{x}_u & \mathbf{x}_u^T \mathbf{x}_v \\ \mathbf{x}_u^T \mathbf{x}_v & \mathbf{x}_v^T \mathbf{x}_v \end{pmatrix}$$

where J is the Jacobian matrix of $\mathcal{S}$: $J = \begin{pmatrix} \mathbf{x}_u & \mathbf{x}_v \end{pmatrix}$ (3×2 matrix).

Why is the first fundamental useful?

- If **a** is a vector of Ω, $\tilde{a}$ its corresponding tangent vector, then: $\|\mathbf{a}\|^2 = \tilde{a}^T J^T J \tilde{a} = \tilde{a}^T I \tilde{a}$
- Compute the length of a curve $C(t) = \mathbf{x}(u(t), v(t))$:

$$l_{[a,b]} = \int_{[a,b]} \left(u_t v_t\right) I \left(u_t v_t\right)^T$$

- The Surface Area $\mathcal{A} = \int \int_{\mathcal{A}} \sqrt{detI} dudv$

Second Fundamental Form

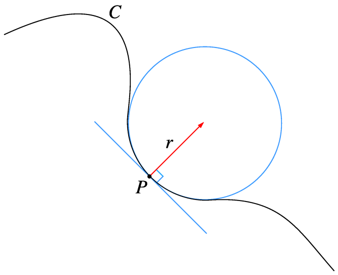
Definition (Second Fundamental Form)

The ***Second fundamental form*** characterizes the way a surface bends:

$$II = \begin{pmatrix} x_{uu}^T \cdot n & x_{uv}^T \cdot n \\ x_{uv}^T \cdot n & x_{vv}^T \cdot n \end{pmatrix}$$

It is a quadratic form on the tangent plane to the surface.

As a starter: curvature of a curve



Normal Curvature

Definition (Normal Curvature)

For each tangent vector $\mathbf{t}$ at a point p of the surface, the normal curvature is defined as:

$$\kappa_n(\mathbf{t}) = \frac{\mathbf{t}^T \cdot II \cdot \mathbf{t}}{\mathbf{t}^T \cdot I \cdot \mathbf{t}}.$$

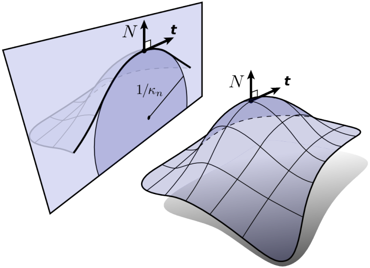


Image from Crane et al. 2013

- The normal curvature varies with $\mathbf{t}$.

Principal curvatures and directions

Definition (Principal curvature)

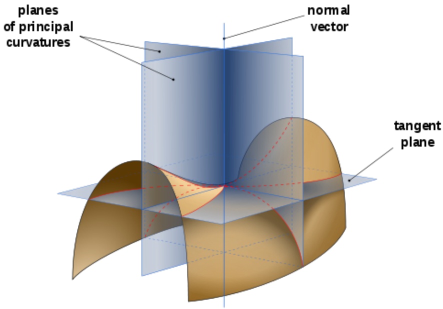
Let κ_1 be the minimum of $\kappa_n(\mathbf{t})$ (normal curvature at p) and κ_2 be the maximum of $\kappa_n(\mathbf{t})$. κ_1 and κ_2 are called the *principal curvatures* of the surface at p.

- If $\kappa_1 \neq \kappa_2$, the two associated tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ are called principal directions and they are orthogonal
- $\kappa_1, \kappa_2, \mathbf{t}_1, \mathbf{t}_2$ are the eigenvalues and eigenvectors of the *Shape Operator*:

$$S = I^{-1} \cdot II$$

Principal curvatures and directions

- If $\kappa_1 = \kappa_2$, the point is called an umbilic or umbilical point and the surface is locally spherical.
- $\kappa_n(\mathbf{t}) = \kappa_1 cos^2\phi + \kappa_2 sin^2\phi$ (Euler) (ϕ is the angle between $\mathbf{t}_1$ and $\mathbf{t}$
- $(\mathbf{t}_1, \mathbf{t}_2, \mathbf{n})$ is called the local intrinsic coordinate system.



Curvature Tensor

Definition (Curvature Tensor)

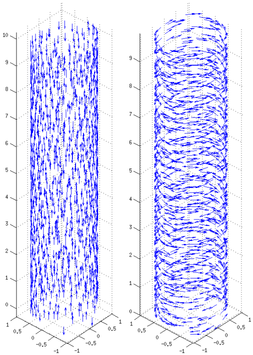
The ***Curvature Tensor*** is a symmetric 3×3 matrix C whose eigenvalues are $(\kappa_1, \kappa_2, 0)$ and corresponding eigenvectors $(\mathbf{t}_1, \mathbf{t}_2, \mathbf{n})$. More precisely:

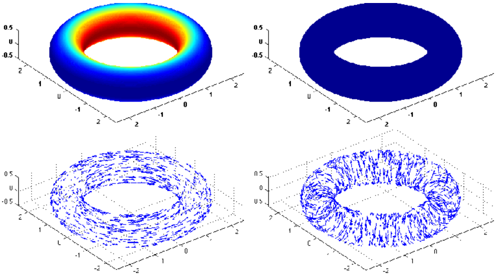
$$C = PDP^{-1}$$

where P is the matrix whose columns are $\mathbf{t}_1, \mathbf{t}_2, \mathbf{n}$ and D is a diagonal matrix with diagonal values $\kappa_1, \kappa_2, 0$.

- ***Mean curvature*** average of the normal curvature: $H = \frac{\kappa_1 + \kappa_2}{2}$
- ***Gaussian curvature*** product of the principal curvature $K = \kappa_1 \cdot \kappa_2$

Examples

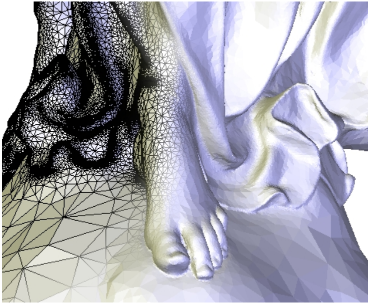




Representing manifold surfaces

Mesh Surface

Polygonal meshes are a piecewise linear approximation of the shape. It is a set of polygons linked together by edges.

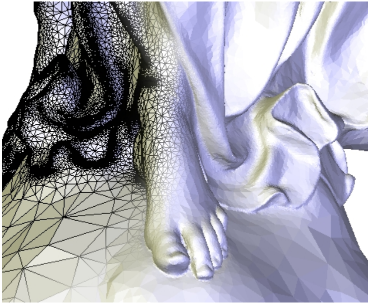


Representing manifold surfaces

Mesh Surface

Polygonal meshes are a piecewise linear approximation of the shape. It is a set of polygons linked together by edges.

- Triangular or quadrilateral meshes are used.



Triangular Meshes

Triangular Meshes

Each point on the surface can be expressed in terms of barycentric coordinates of the three vertices of the facet it belongs to.

Triangular Meshes

Triangular Meshes

Each point on the surface can be expressed in terms of barycentric coordinates of the three vertices of the facet it belongs to.

Euler Formula

Link between the number of triangles F, edges E and vertices V of a closed non-intersecting triangular mesh [Coxeter89] with genus g (number of handles in the surface).

$$V - E + F = 2(1 - g)$$

Triangular Meshes

Triangular Meshes

Each point on the surface can be expressed in terms of barycentric coordinates of the three vertices of the facet it belongs to.

Euler Formula

Link between the number of triangles F, edges E and vertices V of a closed non-intersecting triangular mesh [Coxeter89] with genus g (number of handles in the surface).

$$V - E + F = 2(1 - g)$$

- "Manifoldness": at each point, the surface is locally homeomorphic to a disk (or half disk if the point lies on the boundary).

Differential quantities estimation

Normal estimation

Compute the normal per triangle. For each vertex compute a (possibly weighted) average of the normals of incident triangles.

Differential quantities estimation

Normal estimation

Compute the normal per triangle. For each vertex compute a (possibly weighted) average of the normals of incident triangles.

Curvature tensor estimation

Normal Cycles: For each edge of the meshed surface, $\kappa_2 = 0$ and $\kappa_1 = \beta(e)$ is the dihedral angle between the normals of the two facets adjacent to edge e. Let: $\bar{e} = e/\|e\|$

$$C(v) = \frac{1}{A(v)} \sum_{e \in \mathcal{N}(v)} \beta(e) \|e \cap A(v)\| \bar{e} \cdot \bar{e}^T.$$

Morvan, Cohen-Steiner 2003

Our data: point clouds

Point Clouds

A set of 3D coordinates $(x_i, y_i, z_i)_{i=0 \cdots N-1}$ without any graph structure

Our data: point clouds

Point Clouds

A set of 3D coordinates $(x_i, y_i, z_i)_{i=0 \cdots N-1}$ without any graph structure

- We can still estimate differential quantities

Our data: point clouds

Point Clouds

A set of 3D coordinates $(x_i, y_i, z_i)_{i=0 \cdots N-1}$ without any graph structure

- We can still estimate differential quantities
- We need some notion of neighborhoods: K-nearest neighbors or fixed radius neighborhood

Differential Quantities Estimation

Moving Least Squares

Around each point p fit a regression surface, and estimate the *{Curvature, Gradient, Normal,...}* on this surface.

$$\sum_{q \in \mathcal{N}(p)} w(p,q) \| f(x_q, y_q) - z_q \|^2$$

- Deriving the first and second fundamental form from f is easy.

Differential Quantities Estimation

Moving Least Squares

Around each point p fit a regression surface, and estimate the $\{$*Curvature, Gradient, Normal,...*$\}$ on this surface.

$$\sum_{q \in \mathcal{N}(p)} w(p,q) \| f(x_q, y_q) - z_q \|^2$$

- Deriving the first and second fundamental form from f is easy.

Special Cases

- Normal direction: eigenvector corresponding to the least eigenvalue of the local covariance matrix [Hoppe92, Mitra2003]
- Mean curvature: proportional to the displacement induced by projecting a point to its local regression plane [Digne2011]

$$\mathcal{P}(p) - p = -\frac{1}{4} H(p) r^2 + O(r^2)$$

Adding a graph structure to a point cloud

Goal

Build a surface mesh (a set of triangles glued by edges) that represents the surface.

- Interpolating/Approximating?
- Closed surface reconstruction? Boundary preserving surface reconstruction?
- Smooth/piecewise smooth surface?
- Detail preservation/representation sparsity?

Different reconstruction methods depending on the application

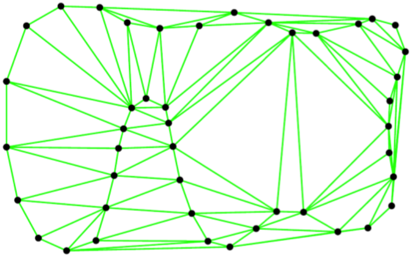
Outline

Methods coming from computational geometry

- Convex Hulls...
- Crust, Eigencrust, powercrust
- Delaunay filtering
- α-shapes
- Ball Pivoting Algorithm

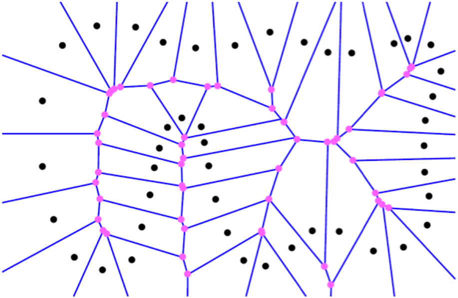
Delaunay triangulation

- A Delaunay Triangulation of S is the set of all triangles with vertices in S whose circumscribing circle contains no other points in S^*.
- *Compactness Property:* this is a triangulation that maximizes the minimum angle



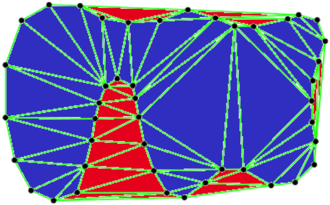
Computational Geometry

- The Voronoi Diagram of S is a partition of space into regions $V(p)$ $(p \in S)$ such that all points in $V(p)$ are closer to p than any other point in S.
- For a vertex, we can draw an empty circle that just touches the three points in S around the vertex.
- Each Voronoi vertex is in one-to-one correspondence with a Delaunay triangle



From Delaunay to a surface mesh

- Given a set of points, we can construct the Delaunay triangulation
- Label each triangle/tetrahedron as inside/outside
- Reconstruction = set of edges/facets that lie between inside and outside triangles/tetrahedra
- Different ways of assigning the labels [Boissonat 84], tight cocoone [Dey Goswami 2003], Powercrust [Amenta et al. 2001] Eigencrust [Kolluri et al. 2004]



The Crust Algorithm [Amenta et al. 1998]

- If we consider the Delaunay Triangulation of a point set, the shape boundary can be described as a subset of the Delaunay edges.
- How do we determine which edges to keep?
- Two types of edges:
  - Those connecting adjacent points on the boundary
  - Those traversing the shape
- Discard those that traverse the shape

The Crust Algorithm [Amenta et al. 1998]

In 2D:

- Given a point set S compute its Voronoi diagram and Voronoi vertices V
- Compute the Delaunay triangulation of $S \cup V$
- Keep only edges that connects points in S (eq. to keeping all edges for which there is a circle that contains the edge but no Voronoi vertices)

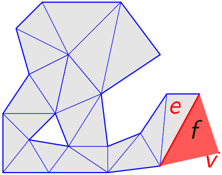
In 3D: Not all Voronoi Vertices are added to the set. Only the poles (furthest points of the Voronoi cell) are considered.

Ball Pivoting Algorithm [Bernardini et al. 99]

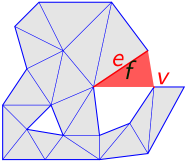
- BPA computes a triangle mesh *interpolating* a given point cloud
- Three points form a triangle if a ball of a user-specified radius ρ touches them without containing any other point
- Start with a seed triangle
- The ball pivots around an edge until it touches another point, forming another triangle
- Expand the triangulation over all edges then start with a new seed

Different types of expansion

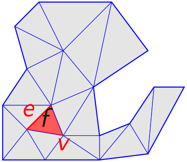
- Advancing front triangulation
- Front is a set of edges



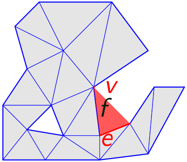
(f) Expansion case



(g) Gluing case

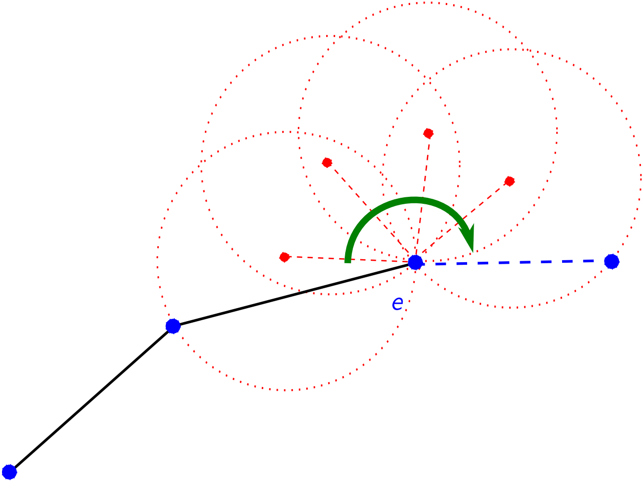


(h) Hole filling case

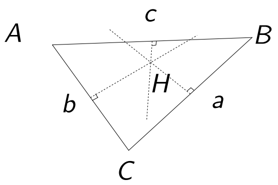


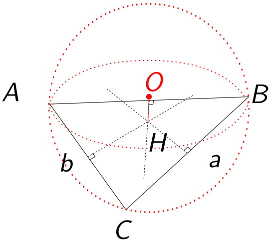
(i) Ear filling case

Rotating the sphere



Finding the R-circumsphere





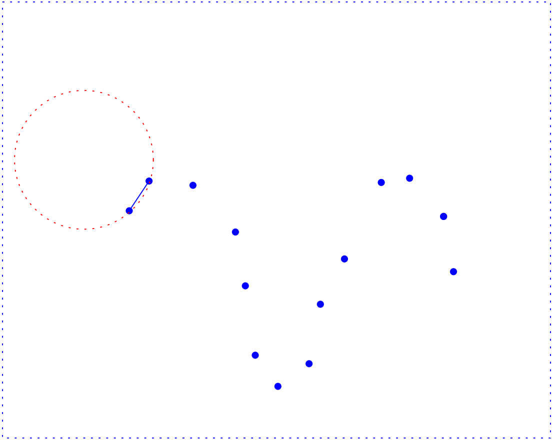
- Such a sphere exists only if $R_b^2 - R^2 \geq 0$.
- Let us denote by $\mathbf{n}$ the normal to the triangle plane, oriented such that is has a nonnegative scalar product with the vertices normals. Provided $R_b^2 - R^2 \geq 0$ (hence the sphere existence), the center O of the sphere can be found as:

$$O = H + \sqrt{R_b^2 - R^2} \cdot \mathbf{n}.$$

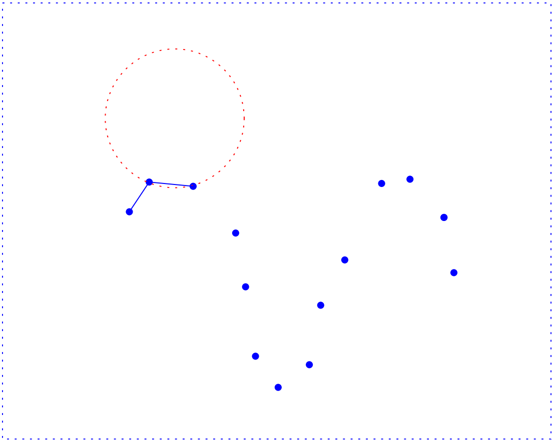
Properties and Guarantees of the resulting mesh

- The surface is guaranteed to be self-intersection free (no triangle will intersect each other except at an edge or vertex, and at most two triangles can be adjacent to an edge).
- Normal coherence on a facet.
- For each triangle there exists an empty ball incident to the three vertices with empty interior

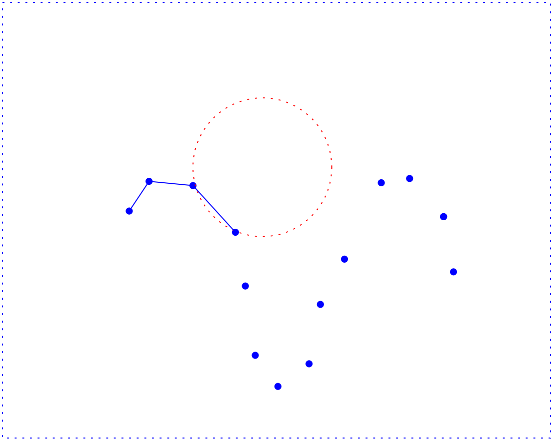
Detailed area



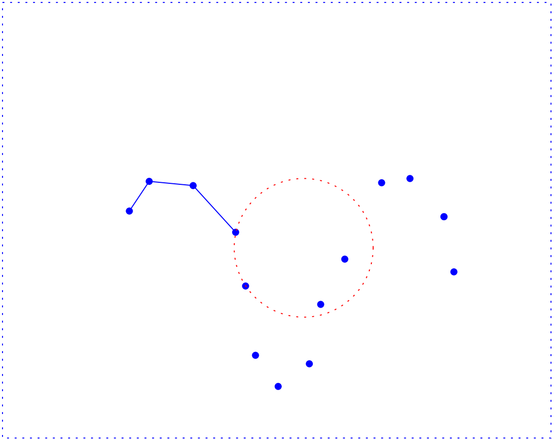
Detailed area



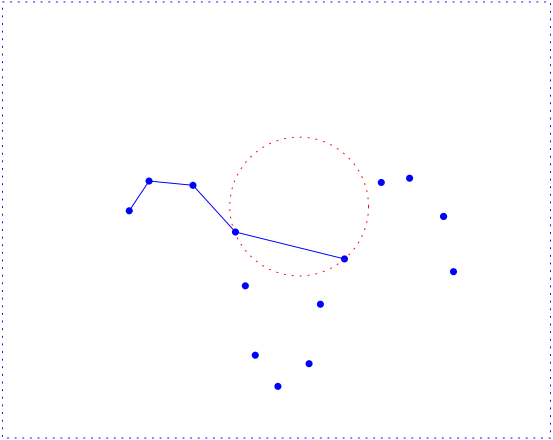
Detailed area



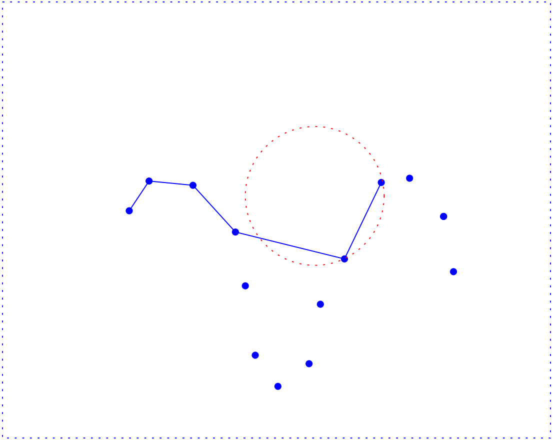
Detailed area



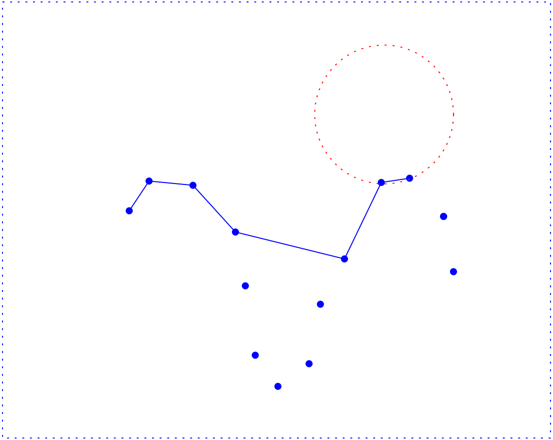
Detailed area



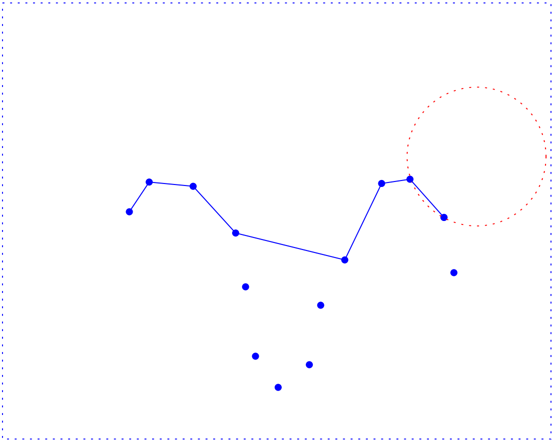
Detailed area



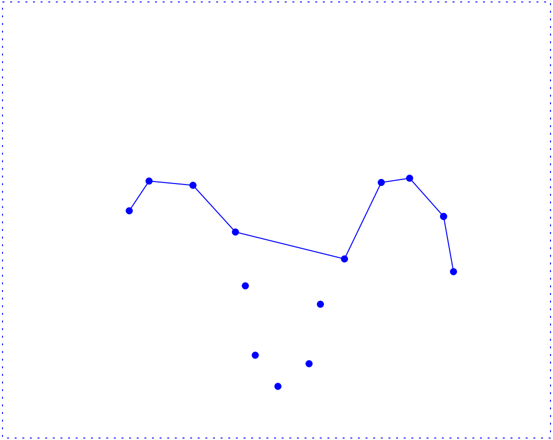
Detailed area



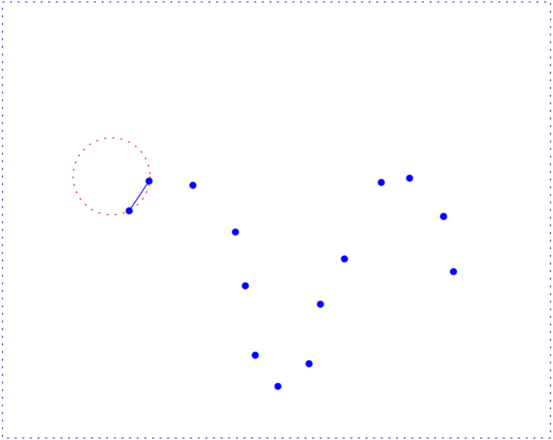
Detailed area



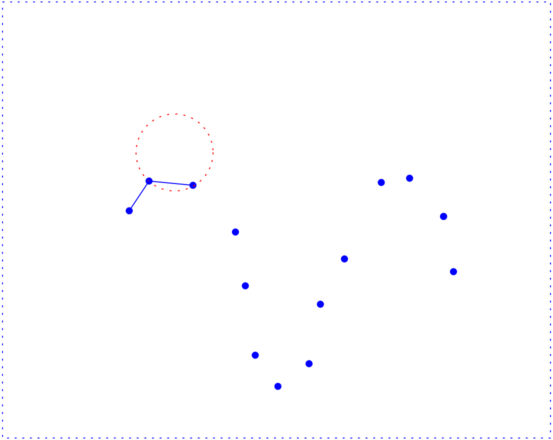
Detailed area



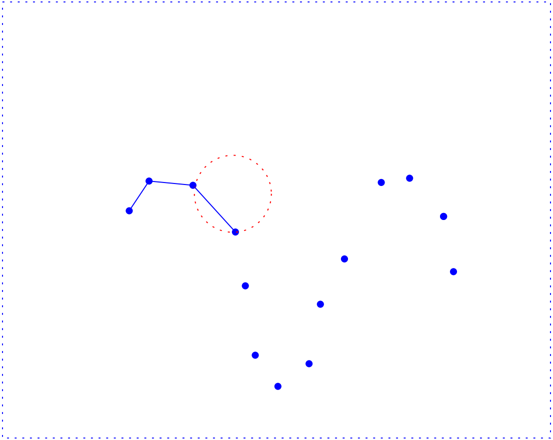
Smaller ball radius



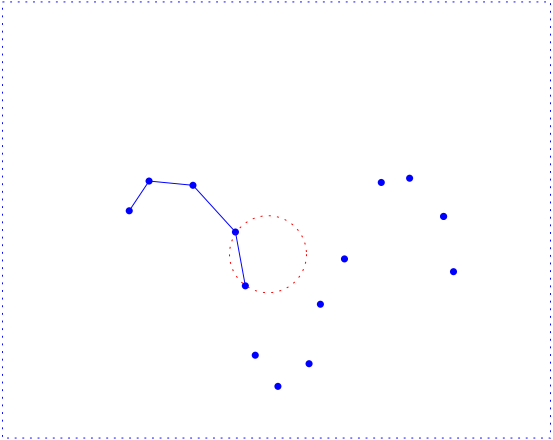
Smaller ball radius



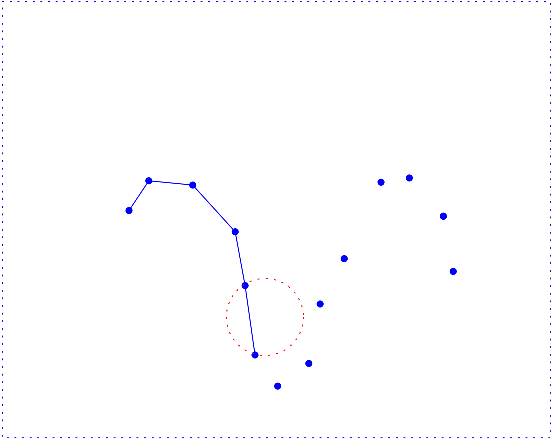
Smaller ball radius



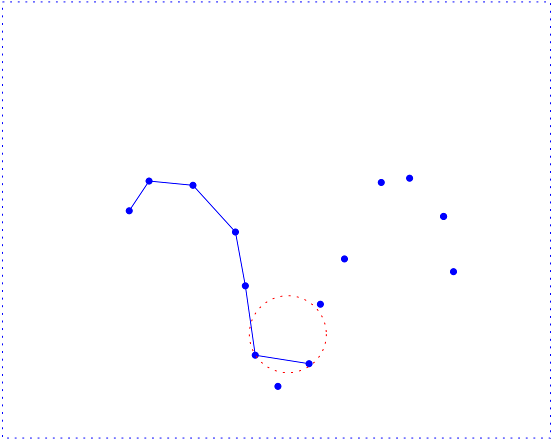
Smaller ball radius



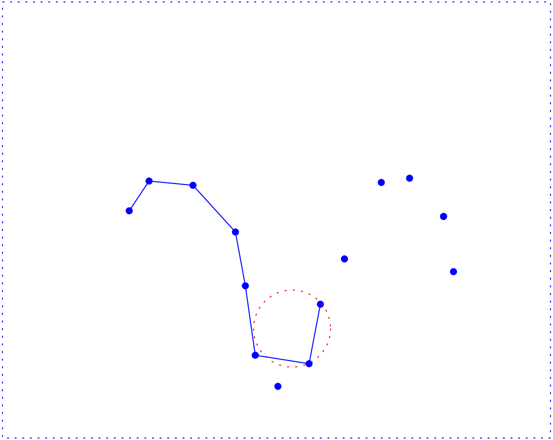
Smaller ball radius



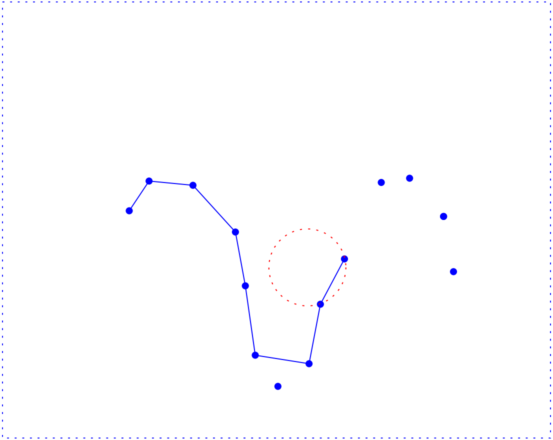
Smaller ball radius



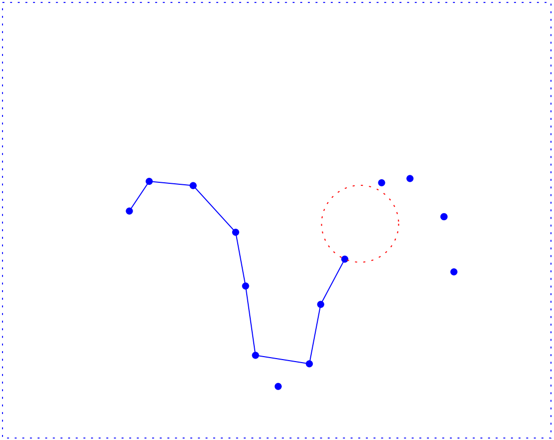
Smaller ball radius



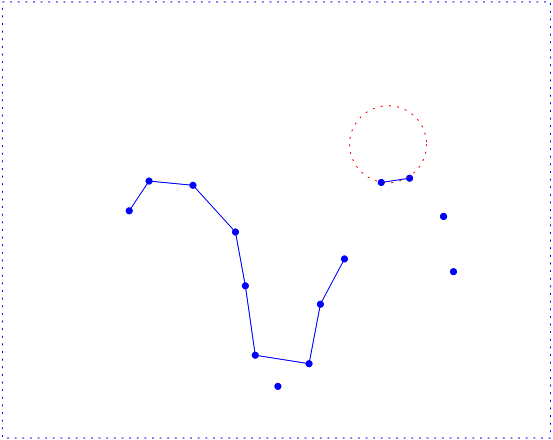
Smaller ball radius



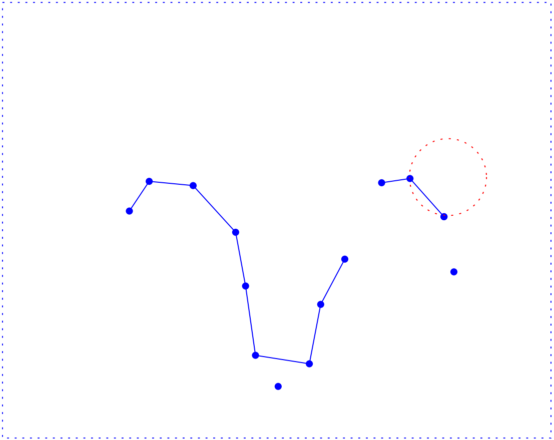
Smaller ball radius



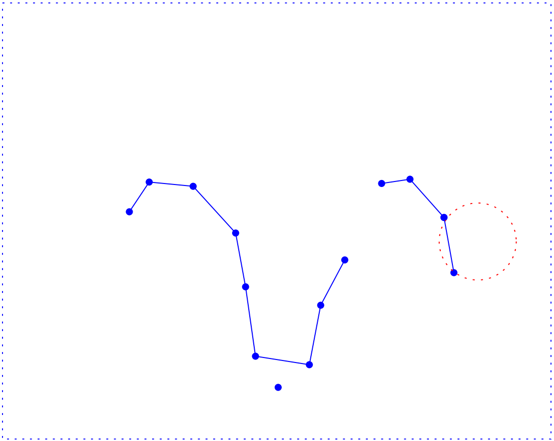
Smaller ball radius

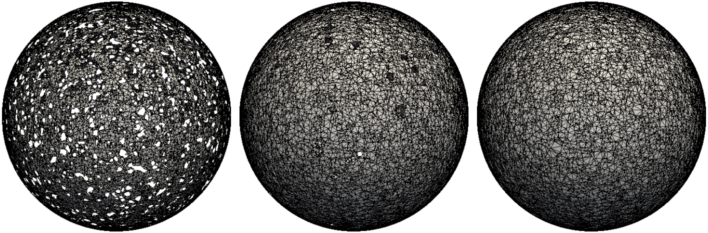


Smaller ball radius



Smaller ball radius



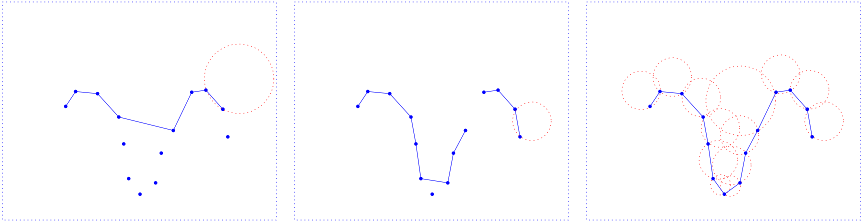


(j) $r = 0.02$ (k) $r = 0.03$ (l) $r = 0.05$

Figure: Radius too small: areas with lower density are not triangulated. Large radius : higher computation times + detail loss.

Figure: Reconstructing the Stanford Bunny point cloud, with a single radius (0.0003), two radii (0.0003; 0.0005) and three radii (0.0003; 0.0005; 0.002).

Radius	Time(s)	vertices	facets	boundary edges
0.0003	$10s$	318032	391898	272832
0.0003; 0.0005	$21s$	356252	698963	22727
0.0003; 0.0005; 0.002	$29s$	361443	713892	7897

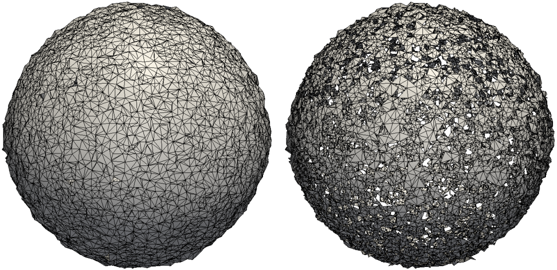


(a) Detail loss

(b) Hole creation

(c) A possible correction: using multiple radii

Figure: Detail loss and hole creation due to a too large radius (left) and a too small one (middle). A possible solution is to use multiple radii (right).



(a) $r = 0.05$ (b) $r = 0.02; 0.03; 0.05$

Figure: Applying the ball pivoting to a noisy sphere: $r = 0.05$ (left) and $r = 0.02; 0.03; 0.05$ (right). A single radius does not allow to interpolate the input data and applying multiple radii is not a solution in addition to being difficult to tune.

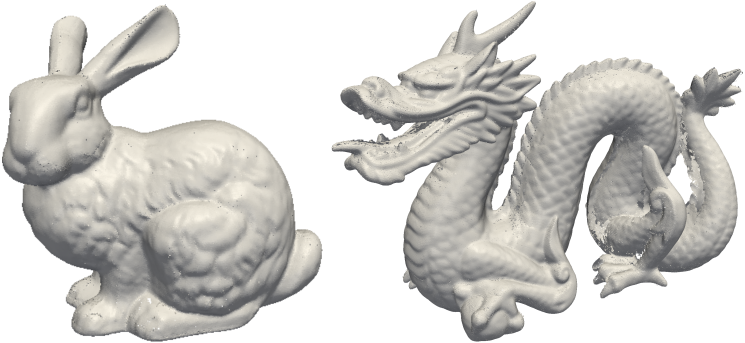


Figure: Bunny and Dragon reconstruction

Problems and solutions

- The larger the ball radius the slower the computation
- The larger the ball radius the more details will be lost
- The smaller the ball radius the more dependent on the sampling
- Varying ball radius $\leftarrow$ slow down the process
- Use of a *scale space*: a multiscale representation of the point cloud.

Summary: Advantages/Drawbacks of the ball pivoting

Drawbacks

- Size of the ball?
- No suppression of redundant points
- No hole closure

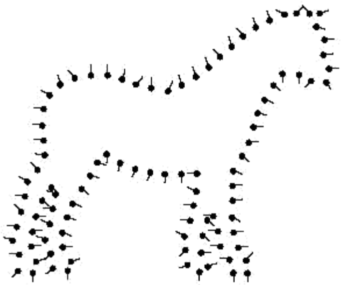
Advantages

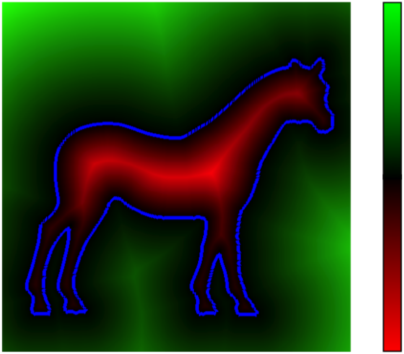
- Control on the size of the triangles created
- Radius of the ball determines what is a hole
- Surface boundary preservation

Modification through the use of a *scale space* for better detail preservation [Digne et al. 2011].

Outline

Implicit surface reconstruction - Level set methods





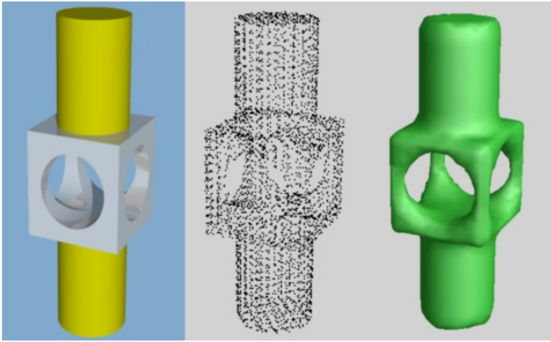
- See the surface as an isolevel of a given function
- Extract the surface by some contouring algorithm: Marching cubes [Lorensen Cline 87], Particle Systems [Levet et al. 06]

Surface reconstruction from unorganized points [Hoppe et al. 92]

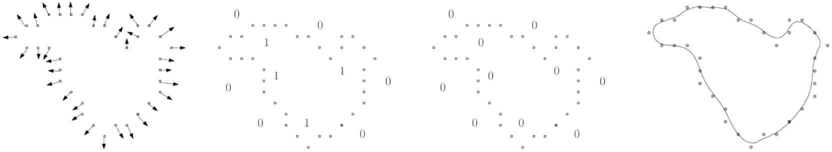
- Input: a set of 3D points
- *Idea:* for points on the surface the signed distance transform has a gradient equal to the normal
$$F(p) = \pm \min_{q \in \mathcal{S}} \|p - q\|$$
- 0 is a regular value for F and thus the isolevel extraction will give a manifold
- Compute an associated tangent plane (o_i, n_i) for each point p_i of the point set
- Orientation of the tangent planes as explained before.

Surface reconstruction from unorganized points [Hoppe et al. 92]

- Once the points are oriented
- For each point p, find the closest centroid o_i
- Estimated signed distance function: $\hat{f}(p) = n_i \cdot (p - o_i)$



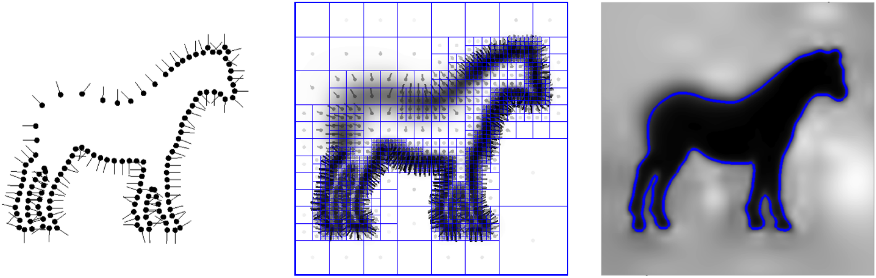
Poisson Surface Reconstruction [Kazhdan et al. 2006]



- Input: a set of oriented samples
- Reconstructs the indicator function of the surface and then extracts the boundary.
- Trick: Normals sample the function's gradients

Poisson Surface Reconstruction [Kazhdan et al. 2006]

1. Transform samples into a vector field
2. Fit a scalar-field to the gradients
3. Extract the isosurface

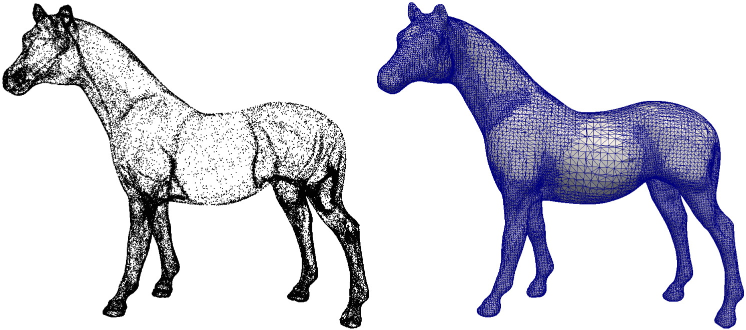


Poisson Surface Reconstruction [Kazhdan et al. 2006]

- To fit a scalar field χ to gradients $\vec{V}$, solve:

$$\min_{\chi} \|\nabla\chi - \vec{V}\|$$

$$\nabla \cdot (\nabla\chi) - \nabla \cdot \vec{V} = 0 \Leftrightarrow \Delta\chi = \nabla \cdot \vec{V}$$



- Gradient Function of an indicator function = unbounded values on the surface boundaries
- We use a smoothed indicator function

Lemma

The gradient of the smoothed indicator function is equal to the smoothed normal surface field.

$$\nabla \cdot (\chi \star \tilde{F})(q_0) = \int_{\partial M} \tilde{F}(q_0 - p) \cdot \vec{N}_{\partial M}(p) dp$$

Chicken and Egg problem: to compute the gradient one must be able to compute an integral over the surface!!

- Approximate the integral by a discrete summation
- Surface partition in patches $\mathcal{P}(s)$:

$$\nabla \cdot (\chi \star \tilde{F})(q_0) = \sum_s \int_{\mathcal{P}(s)} \tilde{F}(q_0 - p) \cdot \vec{N}_{\partial M}(p) dp$$

- Approximation on each patch:

$$\nabla \cdot (\chi \star \tilde{F})(q_0) = \sum_s |\mathcal{P}(s)| \tilde{F}(q_0 - s) \cdot \vec{N}(s)$$

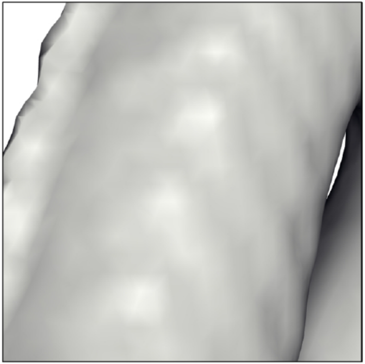
- Let us define $V(q_0) = \sum_s |\mathcal{P}(s)| \tilde{F}(q_0 - s) \cdot \vec{N}(s)$

Problem Discretization

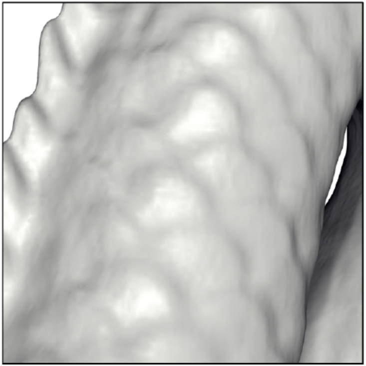
- Build an adaptive octree $\mathcal{O}$
- Associate a function F_o to each node o of $\mathcal{O}$ so that: $F_o(q) = F(\frac{q-o.c}{o.w})\frac{1}{o.w^3}$ ($o.c$ and $o.w$ are the center and width of node o).$\Rightarrow$ multiresolution structure
- The base function F is the *nth* convolution of a box filter with itself
- $$\vec{V}(q) = \sum_{s \in S} \sum_{o \in \mathcal{N}(s)} \alpha_{o,s} F_o(q) s.\vec{N}$$
- Look for χ such that its projection on $span(F_o)$ is closest to ∇V :
- Minimize $\sum_{o \in \mathcal{O}} \langle \Delta\chi - \nabla \cdot V, F_o \rangle^2$
- Extracted isovalue: mean value of χ at the sample positions

Varying octree depth

Varying octree depth



Varying octree depth



Resilience to bad normals

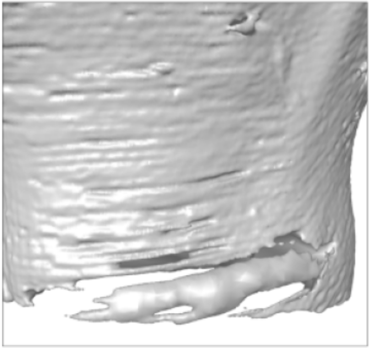
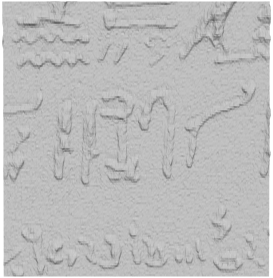


Image from Mullen et al. Signing the unsigned, 2010

detail preservation

Poisson

BPA



Scale Space + BPA

Advantages and drawbacks of the Implicit surface reconstruction methods

Drawbacks

- Only semi-sharp, loss of details
- Final mesh not interpolating the initial pointset
- Marching cubes introduces artefacts
- Watertight surface, very bad with open boundaries

Advantages

- Noise robustness
- Watertight surface, hole closure
- Most standard way of reconstructing a surface

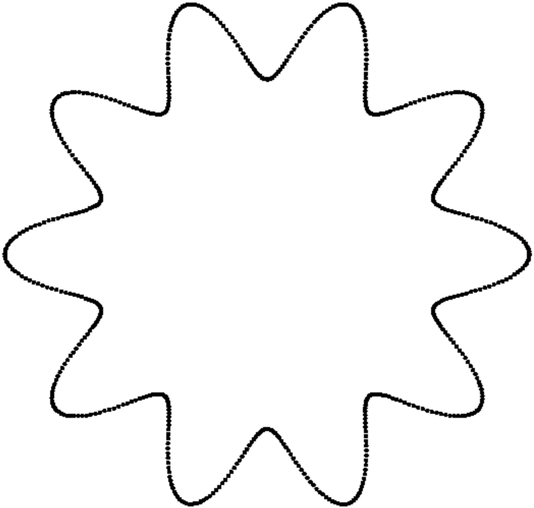
From the signed distance function to the mesh

- At each point in $\mathbb{R}^3$, the signed distance function to the surface can be estimated
- Extract the 0 levelset of this function: points where this function is 0

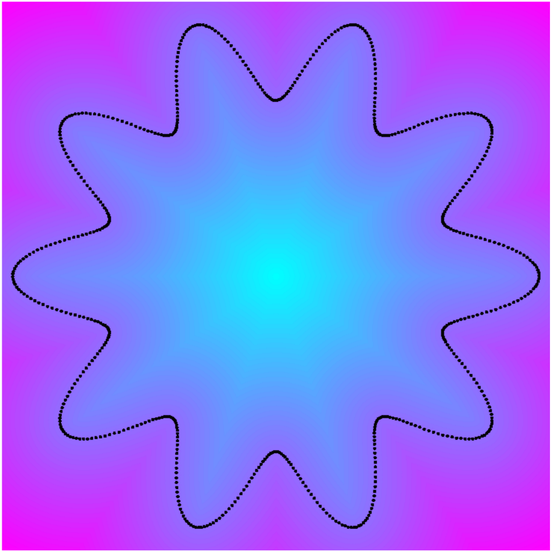
Approximation

Evaluate the function at the vertices of a grid and deduce the local geometry of the surface in each grid cube.

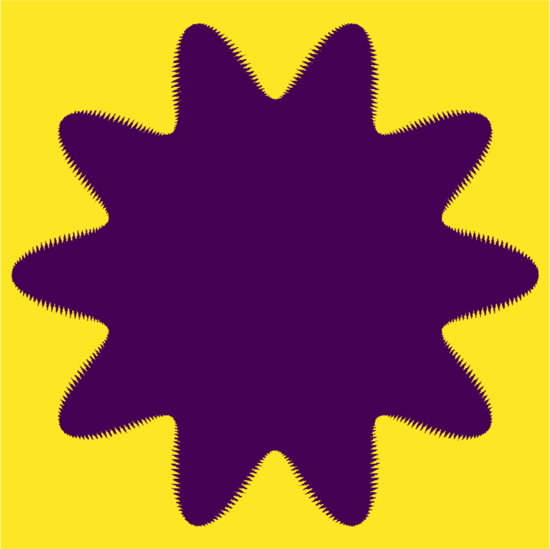
Example in 2D



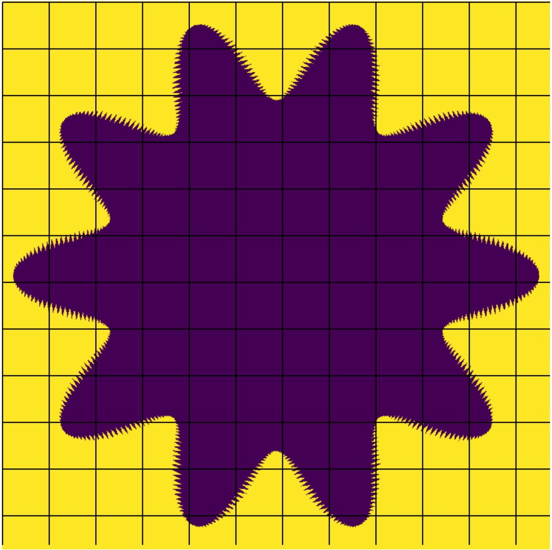
Example in 2D



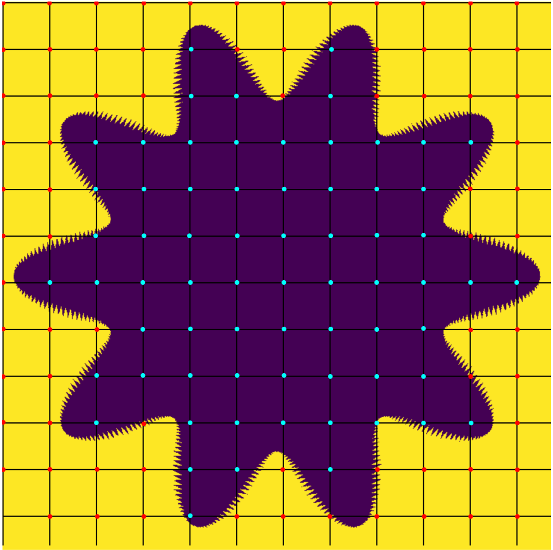
Example in 2D



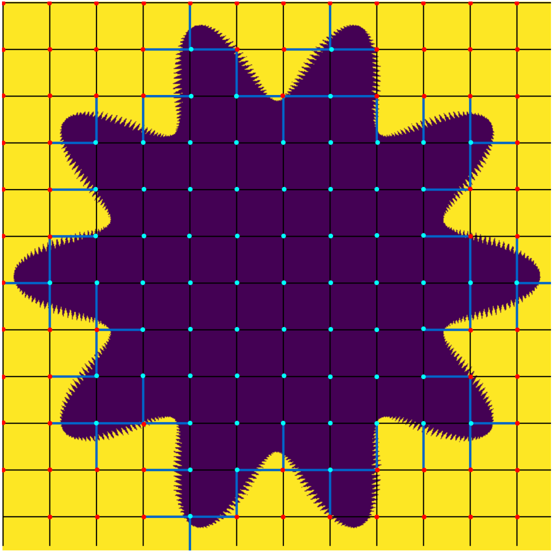
Example in 2D



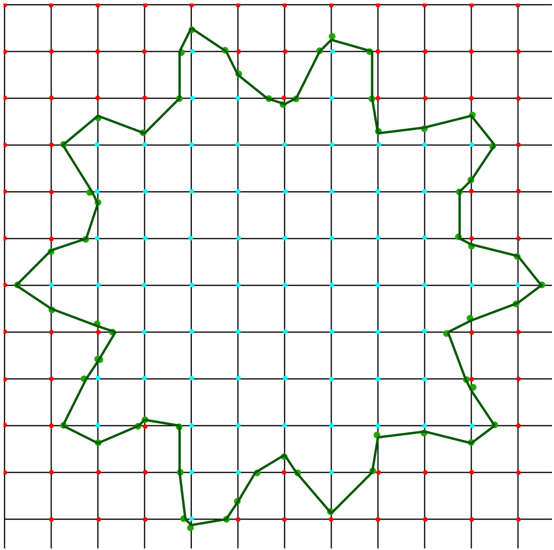
Example in 2D



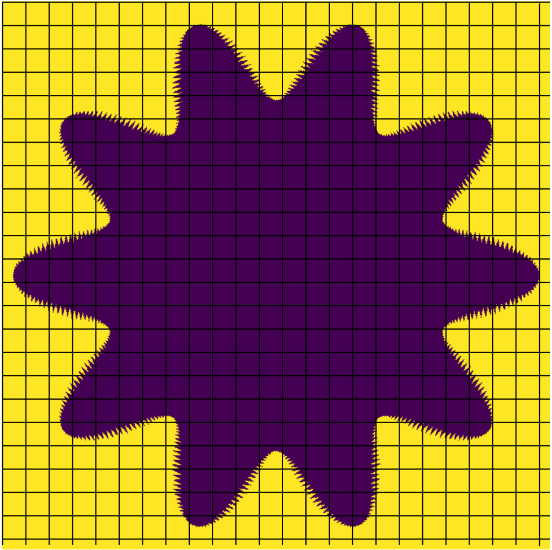
Example in 2D



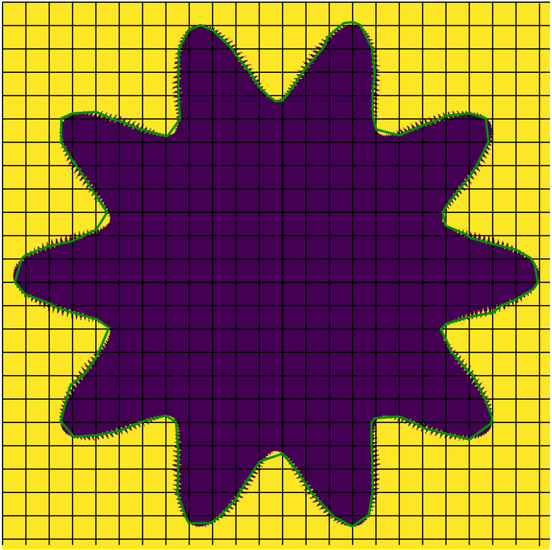
Example in 2D



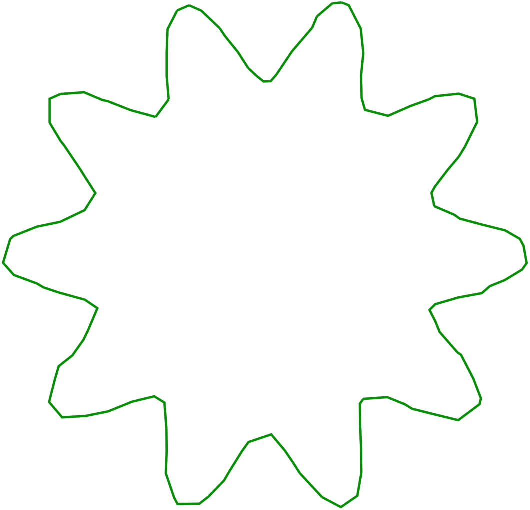
Example in 2D



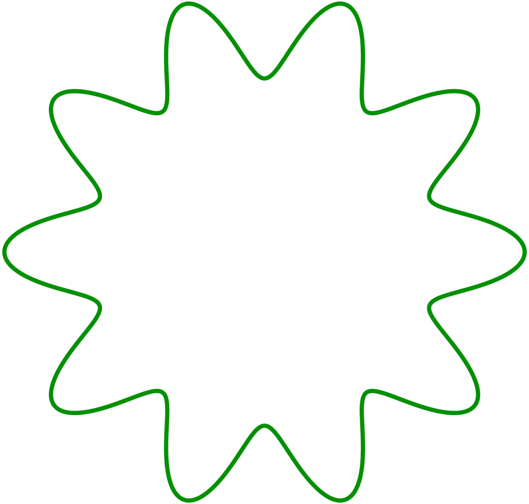
Example in 2D



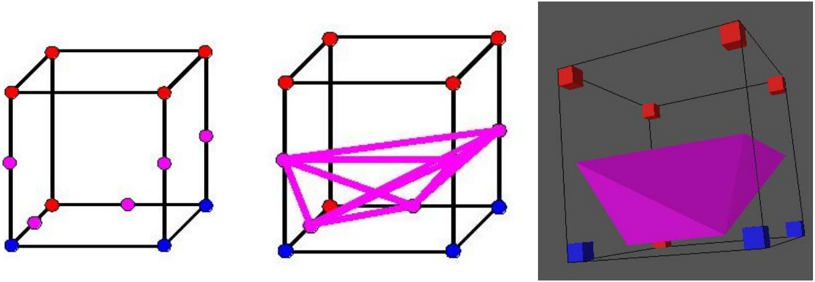
Example in 2D



Example in 2D



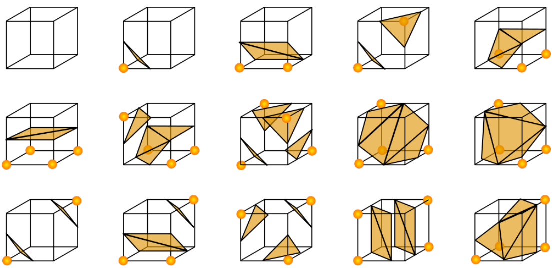
From Marching Squares to Marching Cubes



Drawing lines between intersection points is ambiguous and does not give a surface patch.

Images by Ben Anderson

Look-up tables



- There are $2^8 = 256$ possible cases for cube corner values.
- By symmetry + rotation arguments it reduces to 15 cases.
- It is thus possible to build a look-up table giving the grid cell triangulation based on the corner values case.

And then? Laplace-Beltrami discretization on a mesh

- Mesh triangles are not regular in general
  - Triangle edges DO NOT have constant length
  - Triangle angles ARE NOT constant
- Yet we need to account for the function variations on the surface

Mesh Laplacian

There exist many different Laplacians. We follow the terminology of [Zhang et al. 2007] and [Vallet and Levy 2008]

Combinatorial Laplacian

Definition

Given a triangular manifold mesh with N vertices $(v_i)_{i=1\cdots N}$, let E be the set of edges. The uniform Laplacian, *umbrella operator* is defined as a matrix L such that:

$$L_{i,j} = \begin{cases} 1, & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

Combinatorial Laplacian

Definition

Given a triangular manifold mesh with N vertices $(v_i)_{i=1 \cdots N}$, let E be the set of edges. The uniform Laplacian, *umbrella operator* is defined as a matrix L such that:

$$L_{i,j} = \begin{cases} 1, & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

- Directly derived from the graph Laplacian.

Combinatorial Laplacians

- **Tutte Laplacian**

$$L_{ij} = \begin{cases} \frac{1}{d_i} \text{ if } (i,j) \in E \\ 0 \text{ otherwise} \end{cases}$$

Combinatorial Laplacians

- **Tutte Laplacian**

$$L_{ij} = \begin{cases} \frac{1}{d_i} \text{ if } (i,j) \in E \\ 0 \text{ otherwise} \end{cases}$$

- **Normalized Graph Laplacian**

$$L_{ij} = \begin{cases} \frac{1}{\sqrt{d_i d_j}} \text{ if } (i,j) \in E \\ 0 \text{ otherwise} \end{cases}$$

Combinatorial Laplacians

- **Tutte Laplacian**

$$L_{ij} = \begin{cases} \frac{1}{d_i} \text{ if } (i,j) \in E \\ 0 \text{ otherwise} \end{cases}$$

- **Normalized Graph Laplacian**

$$L_{ij} = \begin{cases} \frac{1}{\sqrt{d_i d_j}} \text{ if } (i,j) \in E \\ 0 \text{ otherwise} \end{cases}$$

- Other Discretizations: Mean Value Coordinates, Wachspress coordinates....

Combinatorial Laplacian

Combinatorial Laplacian

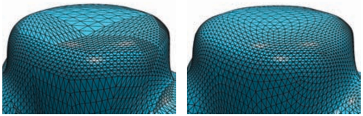
A combinatorial Laplacian depends solely on the connectivity of the mesh.

Combinatorial Laplacian

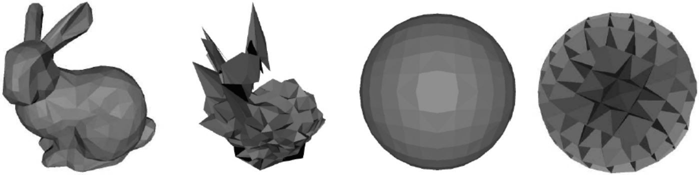
Combinatorial Laplacian

A combinatorial Laplacian depends solely on the connectivity of the mesh.

- The Laplacian is computed independently of its geometrical embedding



Processing with Combinatorial Laplacians



Original bunny. GL compression. Original sphere. GL compression.

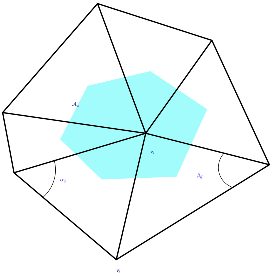
Compression using Normalized Graph Laplacian

Image from Zhang et al. 2004

Geometric Laplacian

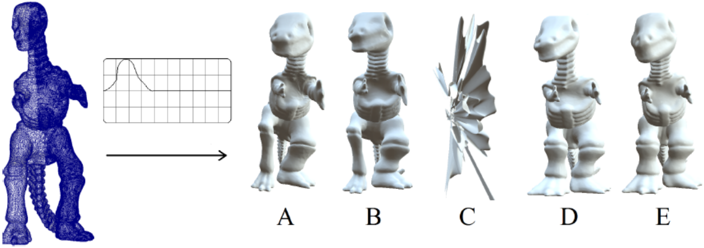
Pinkall Polthier 93

$$(Lf)_i = \sum_{j \in N_i} \frac{1}{2}(cot\alpha_{ij} + cot\beta_{ij})(f_i - f_j)$$



- There is no perfect Laplacian discretization on triangle meshes [Wardetsky et al. 2007]

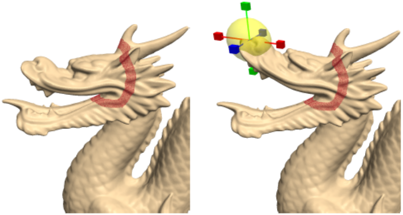
Laplacian Comparisons



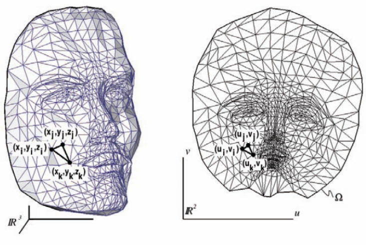
Combinatorial Laplacian, unweighted cotan, weighted cotan, two versions of the symmetrized weighted cotan

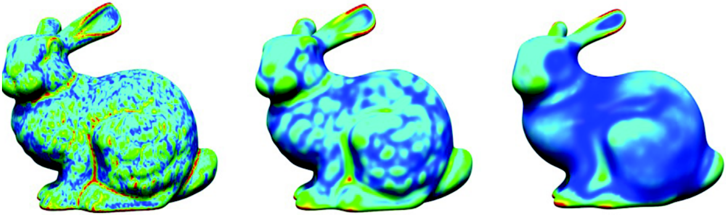
Image from [Vallet and Lévy 2008]

Applications of the Laplace-Betrami Operator



(a) (b)





0 Iterations 10 Iterations 100 Iterations

Conclusion

- Point Set = raw output of many measurement devices
- Graph structure not always necessary for early processing
- Topics not addressed: denoising, entire shape matching, normal orientation, rendering...