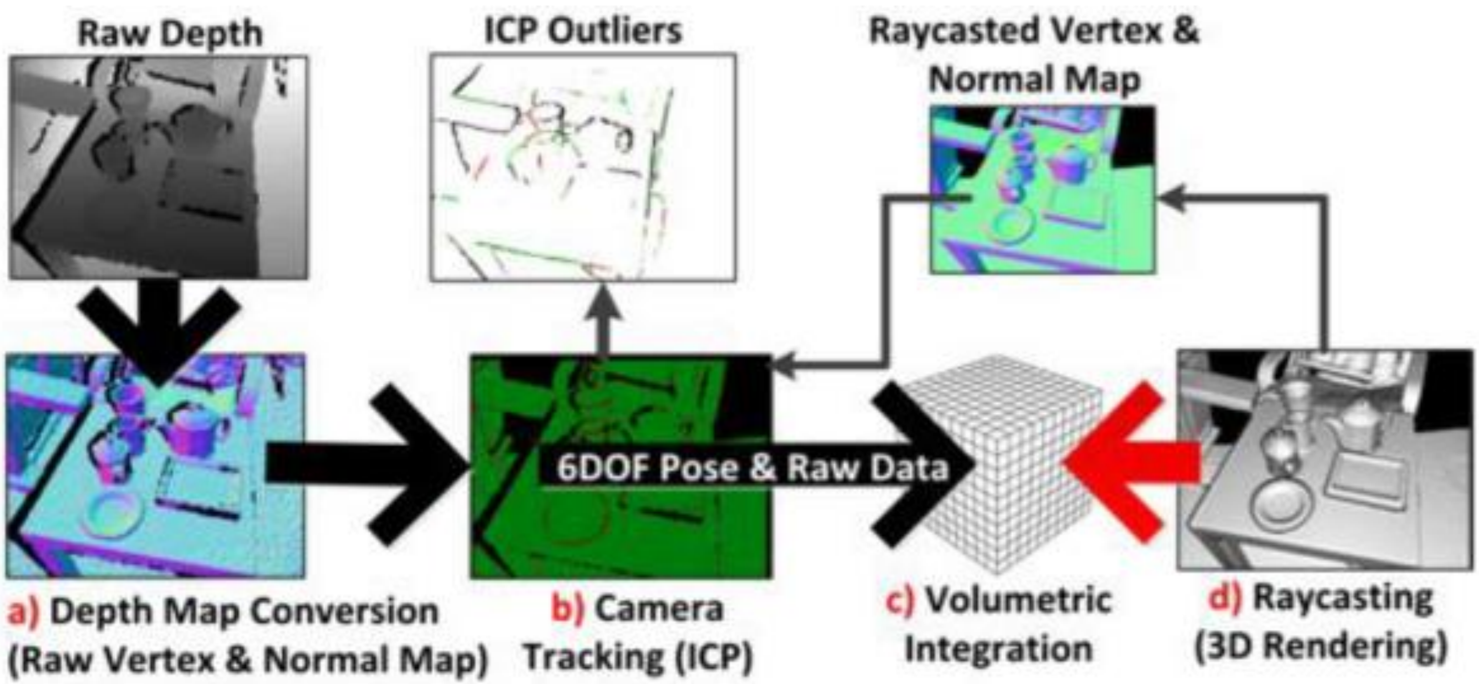
3D Scan Processing

contents

OVERVIEW AND MOTIVATION

Computer Graphics and Visualization

Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera



http://research.microsoft.com/en-us/projects/surfacerecon

Kinect Fusion

SIGGRAPH Talks 2011

KinectFusion:

Real-Time Dynamic 3D Surface Reconstruction and Interaction

Shahram Izadi 1, Richard Newcombe 2, David Kim 1,3, Otmar Hilliges 1,
David Molyneaux 1,4, Pushmeet Kohli 1, Jamie Shotton 1,
Steve Hodges 1, Dustin Freeman 5, Andrew Davison 2, Andrew Fitzgibbon 1

1 Microsoft Research Cambridge 2 Imperial College London
3 Newcastle University 4 Lancaster University
5 University of Toronto

More Recent Fusion Approach

BundleFusion: Real-time Globally Consistent 3D Reconstruction using Online Surface Re-integration

Angela Dai[1] Matthias Nießner[1] Michael Zollhöfer[2] Shahram Izadi[3] Christian Theobalt[2]

[1]Stanford University [2]Max Planck Institute for Informatics [3]Microsoft Research

ACM Transactions on Graphics 2017

http://graphics.stanford.edu/projects/bundlefusion

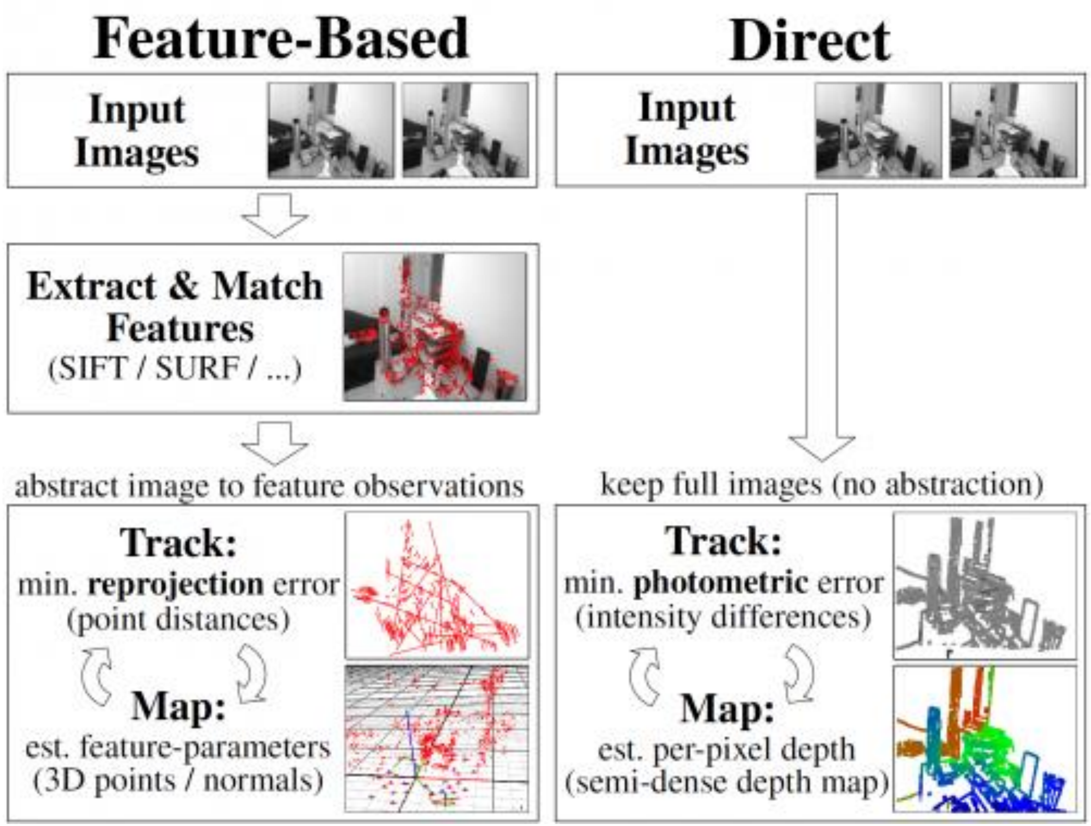
BundleFusion: Real-time Globally Consistent 3D Reconstruction using Online Surface Re-integration

Angela Dai[1] *Matthias Nießner*[1]
Michael Zollhöfer[2] *Shahram Izadi*[3]
Christian Theobalt[2]

[1]Stanford University
[2]Max Planck Institute for Informatics
[3]Microsoft Research

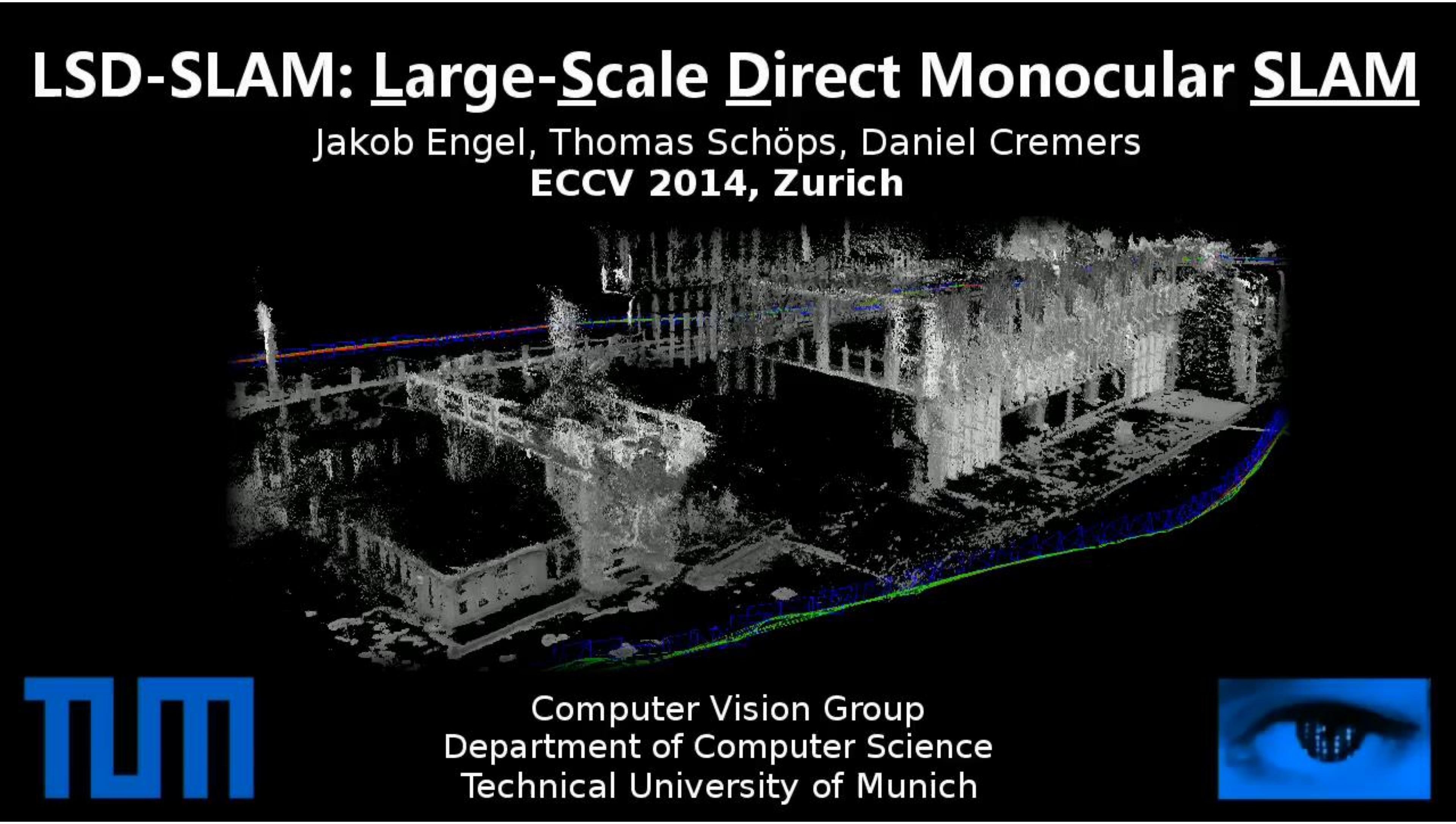
(*contains audio*)

Large-Scale Direct Monocular SLAM

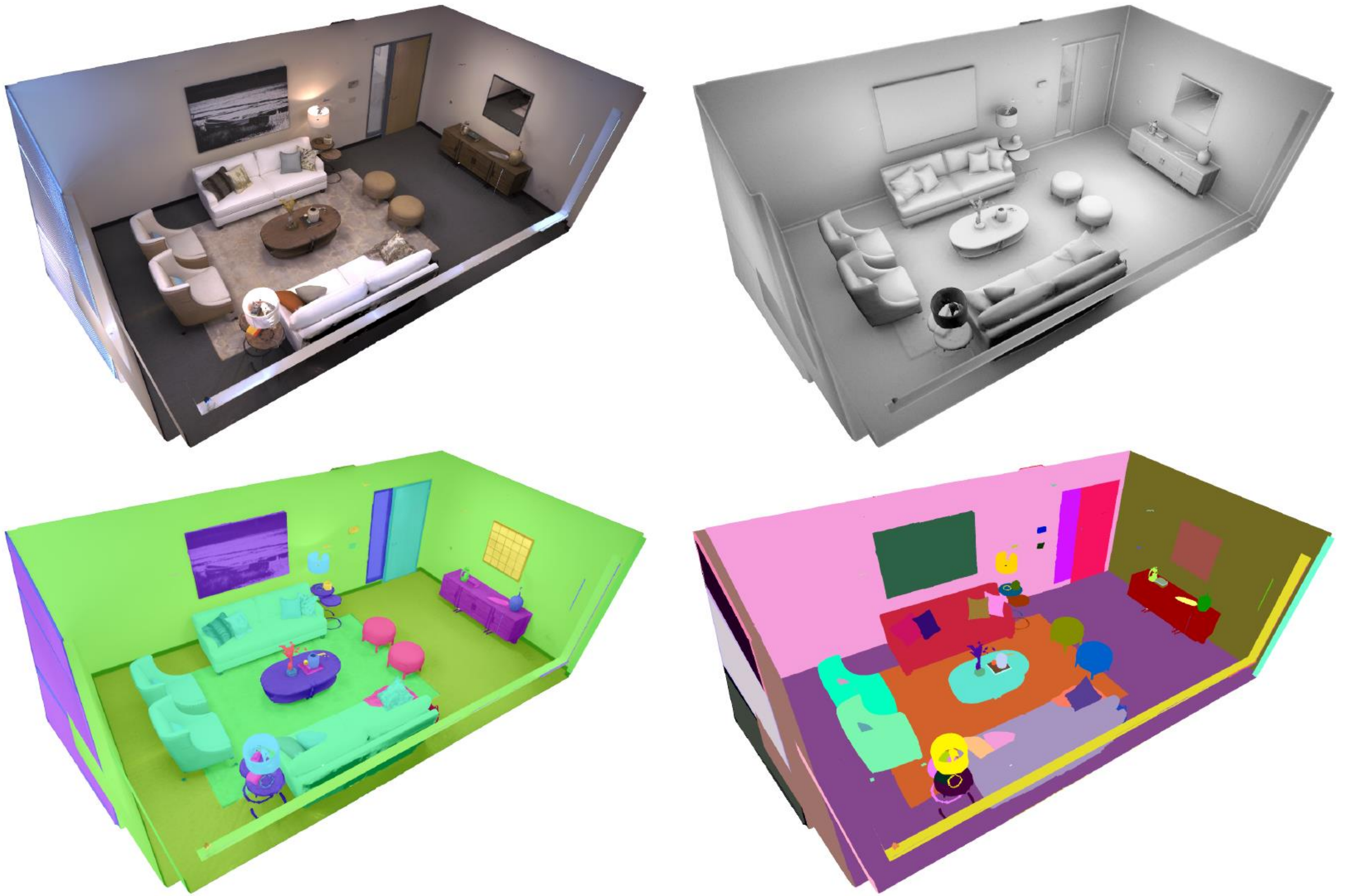


- http://vision.in.tum.de/research/vslam/lsdslam

LSD-SLAM


LSD-SLAM: Large-Scale Direct Monocular SLAM
Jakob Engel, Thomas Schöps, Daniel Cremers
ECCV 2014, Zurich
Computer Vision Group
Department of Computer Science
Technical University of Munich

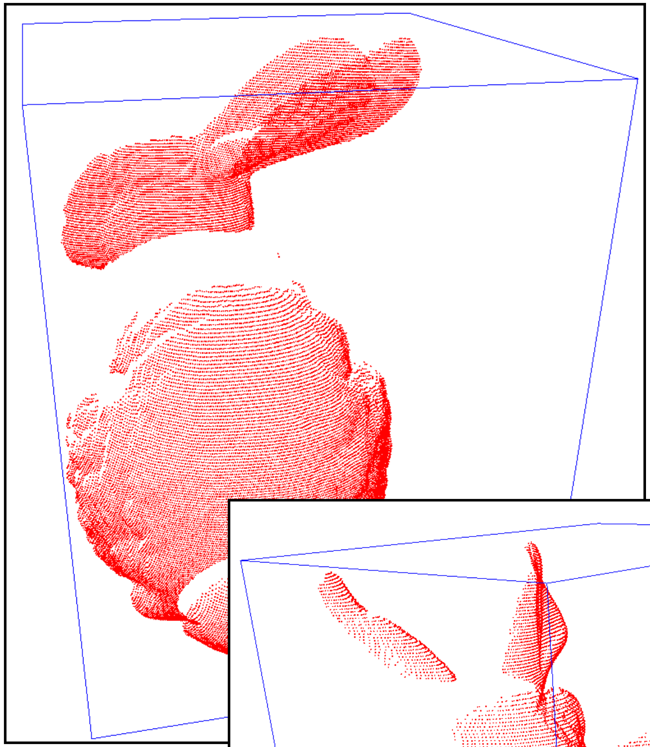
Replica Dataset (2019)

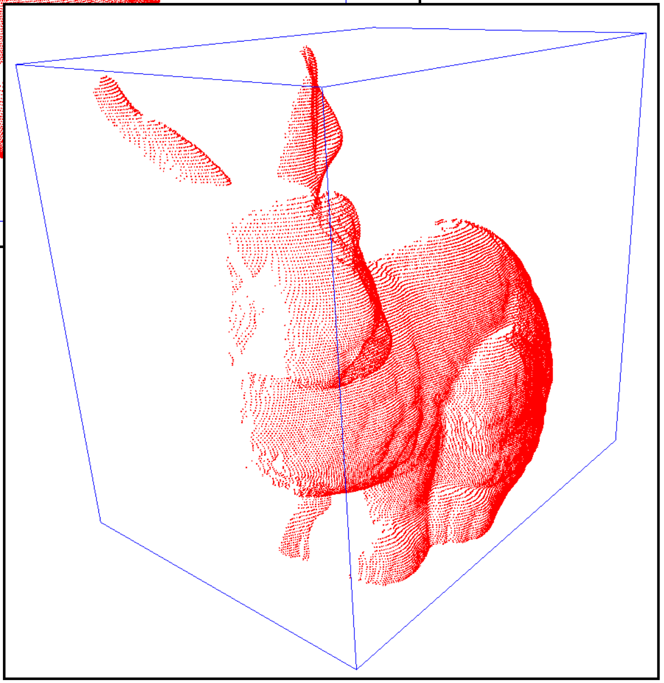


https://github.com/facebookresearch/Replica-Dataset

3D Scan Processing

- **local features:** compute
  - classification (outlier, boundary, sharp edge, corner, smooth)
  - tangent space or surface normal
  - curvatures and higher moments
  - histogram descriptors
- **matching:** find correspondences
  - distance based
  - projection based
  - feature based
- **registration:** two interpretations:
  - bring 3D scans in same coordinate system
  - estimate pose of camera (camera localization)
- **fusion:** merge partial scans
  - point filtering
  - signed distance fields
- **reconstruction:** estimate globally consistent surface

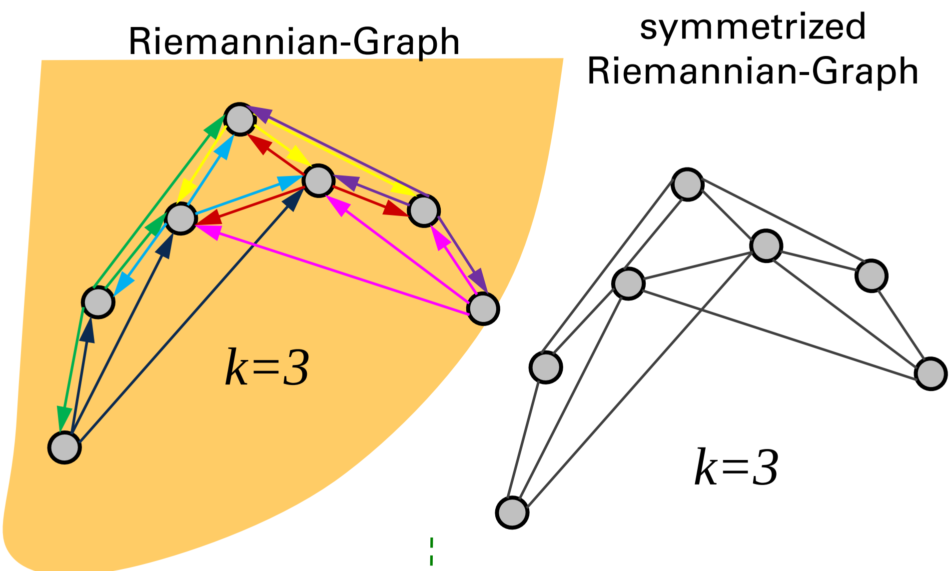


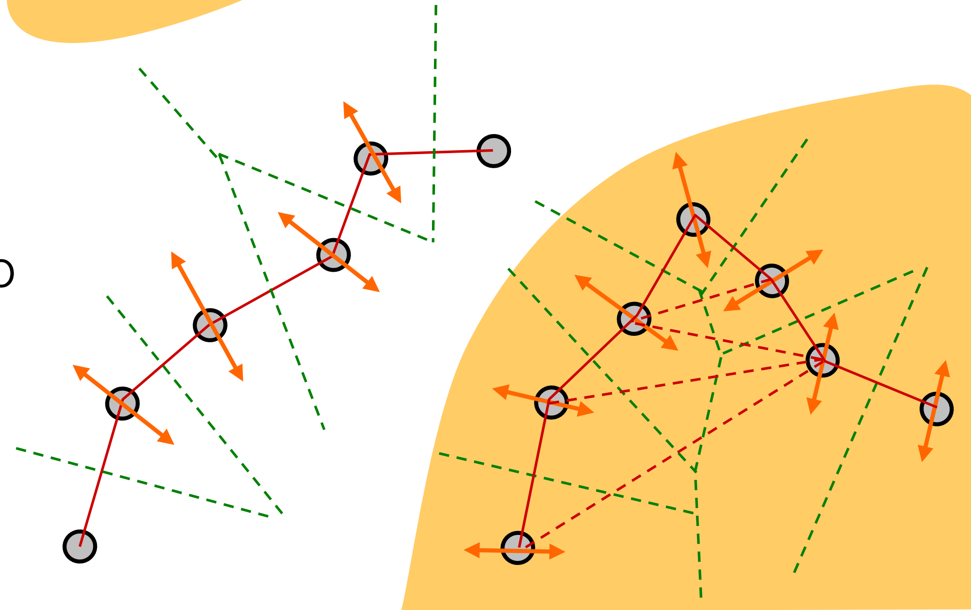


NEIGHBORGRAPHS

Point Neighborhood Definitions

- Riemannian-Graph: includes for each point outgoing edges to the k nearest neighbors (typically $k \in [6,20]$)
- symmetrized Riemannian-Graph eliminate directedness of edges
- Connect to all neighbors within a sphere of fixed radius estimated from sampling density
- filtered edges of Delaunay-Tetrahedralization: estimate per point normal direction from largest extent of Voronoi-cell and eliminate close to parallel edges
- Robust and fast implementations of Delaunay-Tetrahedralization in CGAL or qhull.

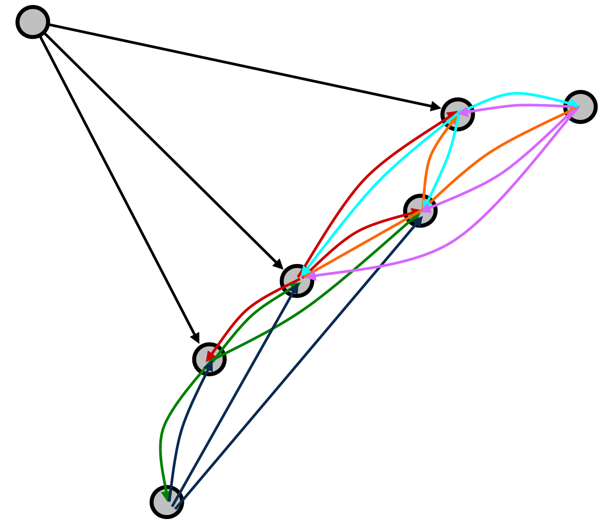




Outlier detection & Density estimation

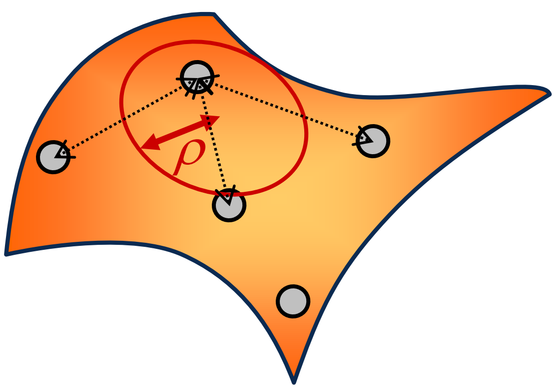
Outlier detection

- In not symmetrized Riemannian-Graph most edges of an outlier point are outgoing and only few ingoing
- Detection by thresholding of fraction of bidirectional edges over unidirectional outgoing edges.



Estimation of sampling density

- The sampling density ρ is defined as the minimum radius of a circle in tangential space, in which at least one surface sample is found in scan. Sampling density typically varies over surface.
- ρ can be estimated from average distance to 3rd up to sixth nearest neighbor points.



ESTIMATION OF LOCAL QUANTITIES

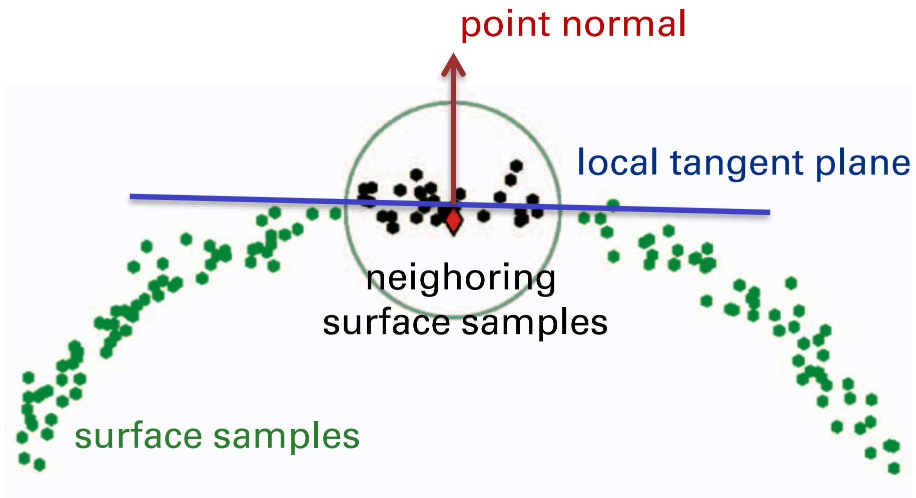
Surface Denoising and Surface Normal Estimation

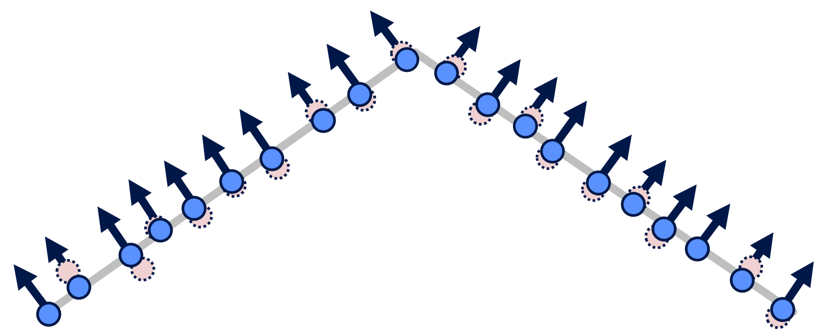
PLANE FITTING

Demo

Normal Estimation in 3D Scans

- Input: set of 3D points sampled from surface
- Output: set of denoised 3D points with normal
- Approach:
  - for each point collect neighborhood
  - define distance-based weight function
  - estimate local tangent plane from weighted least squares problem
  - orthogonally project input point onto local tangent plane
  - assign tangent plane normal to output point





integrated normal estimation and denoising

Weighted Plane Fitting Problem

Computer Graphics and Visualization

- for each point $\boldsymbol{x}$ of point cloud collect m points $\boldsymbol{x}_j$ with knn-query sorted by distance, typically $10 < m < 50$.
- estimate reference radius: $h = \|\boldsymbol{x}_{10} - \boldsymbol{x}\|$
- assign pre-weights: $\omega'_j = \exp\left(-\frac{\|\boldsymbol{x}_j - \boldsymbol{x}\|^2}{h^2}\right)$ and normalize: $\omega_j = \frac{\omega'_j}{\Omega'}$, $\Omega' = \sum_{j=1}^{m} \omega_j{}'$.
- residuals: $r_j = r_j(\boldsymbol{x}) = n_x x_j + n_y y_j + n_z z_j + d$
- parameters: $\tilde{\boldsymbol{p}} = (\hat{\boldsymbol{n}}, d)$ with $\|\hat{\boldsymbol{n}}\| = 1$
- As $\boldsymbol{b} = \boldsymbol{0}$ the normalization constraint is essential to avoid the trivial solution $\tilde{\boldsymbol{p}}^* = \boldsymbol{A}_W^+ \boldsymbol{b} = \boldsymbol{0}$. The constrained LLS is:

$$\underset{\tilde{\boldsymbol{p}}=(\hat{\boldsymbol{n}},d)\,|\,\|\hat{\boldsymbol{n}}\|=1}{\text{minarg}} \quad f(\tilde{\boldsymbol{p}}), f(\tilde{\boldsymbol{p}}) = \left\|\sqrt{\boldsymbol{W}}\boldsymbol{A}\tilde{\boldsymbol{p}}\right\|_2^2 = \tilde{\boldsymbol{p}}^T \underbrace{\boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A}}\tilde{\boldsymbol{p}} = \tilde{\boldsymbol{p}}^T \boldsymbol{M}_W \tilde{\boldsymbol{p}}$$

- For brevity we define a weighted cov. matrix $\boldsymbol{M}_W$

Reduced Plane Fitting Problem

- Theorem: if $\tilde{\boldsymbol{p}}^* = (\hat{\boldsymbol{n}}^*, d^*)$ is the solution to the plane fitting problem, then the weighted center of mass
$$\overline{\boldsymbol{x}} = \sum_{j=1}^{m} \omega_j \boldsymbol{x}_j$$
is on $\tilde{\boldsymbol{p}}^*$, i.e. ${\hat{\boldsymbol{n}}^*}^T \overline{\boldsymbol{x}} + d^* = 0$. [Proof by setting $\partial_d f(\tilde{\boldsymbol{p}}) = 0$]
- We can enforce $d = 0$ by considering a coordinate system with $\overline{\boldsymbol{x}}$ in the origin, a reduced parameter vector $\boldsymbol{p}'$ and reduced weighted cov. matrix:
$$\boldsymbol{p}' = \hat{\boldsymbol{n}}, \qquad \boldsymbol{x}_j' = \boldsymbol{x}_j - \overline{\boldsymbol{x}}, \qquad \boldsymbol{M}_W' := {\boldsymbol{A}'}^T \boldsymbol{W} \boldsymbol{A}' = \sum_{j=1}^{m} \omega_j \boldsymbol{x}_j' {\boldsymbol{x}_j'}^T$$
- The plane fitting problem reduces to
$$\hat{\boldsymbol{n}}^* = \underset{\hat{\boldsymbol{n}} \mid \|\hat{\boldsymbol{n}}\|=1}{\operatorname{minarg}} \hat{\boldsymbol{n}}^T \boldsymbol{M}_W' \hat{\boldsymbol{n}}$$

Reduced Plane Fitting Problem

- Eigenvalue decomposition of the symmetric matrix $\boldsymbol{M}'_W$
  $$\boldsymbol{M}'_W = \boldsymbol{V\Lambda V}^T = \lambda_1 \boldsymbol{v}_1 \boldsymbol{v}_1^T + \lambda_2 \boldsymbol{v}_2 \boldsymbol{v}_2^T + \lambda_3 \boldsymbol{v}_3 \boldsymbol{v}_3^T, \lambda_1 \le \lambda_2 \le \lambda_3$$
- plugged into the objective function yields
  $$\widehat{\boldsymbol{n}}^T \boldsymbol{M}'_W \widehat{\boldsymbol{n}} = \lambda_1 (\boldsymbol{v}_1^T \widehat{\boldsymbol{n}})^2 + \lambda_2 (\boldsymbol{v}_2^T \widehat{\boldsymbol{n}})^2 + \lambda_3 (\boldsymbol{v}_3^T \widehat{\boldsymbol{n}})^2$$
- From the increasing ordering of the λ_i it follows that the optimal normal is given by
  $$\boxed{\widehat{\boldsymbol{n}}^* = \pm \boldsymbol{v}_1}$$
- Note that the sign of the normal direction (plane orientation) is not unique. A globally consistent orientation is typically achieved by
  - knowledge of an exterior point (scanner location)
  - a region growing normal orientation algorithm

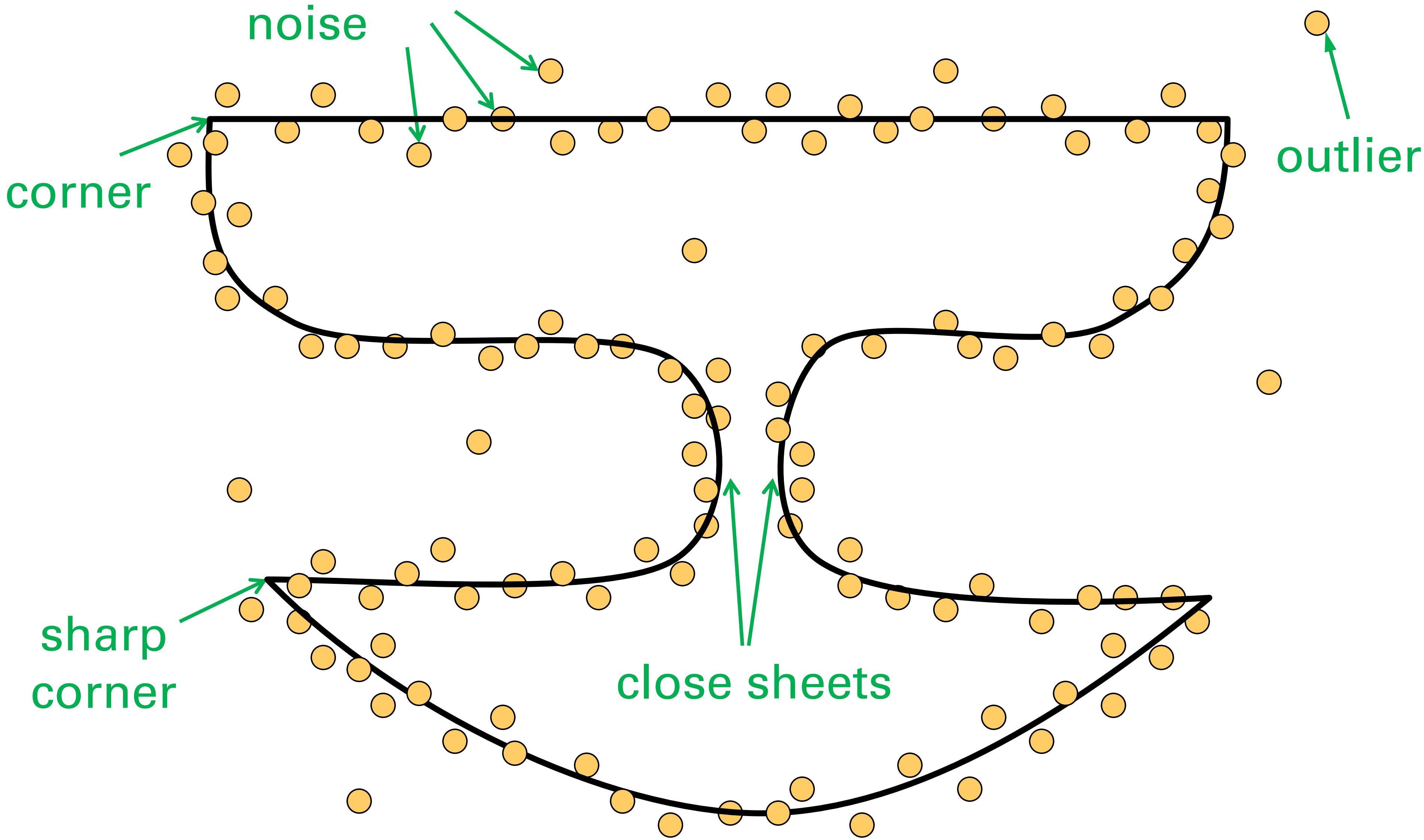
Summary of Plane Fitting Problem

- Input: set of m weighted points $\boldsymbol{x}_j, \omega'_j$.

1. normalize weights: $\omega_j = \frac{\omega'_j}{\Omega'}$, $\Omega' = \sum_{j=1}^{m} \omega'_j$
2. compute weighted center of mass $\bar{\boldsymbol{x}} = \sum_{j=1}^{m} \omega_j \boldsymbol{x}_j$
3. transform points: $\boldsymbol{x}'_j = \boldsymbol{x}_j - \bar{\boldsymbol{x}}$
4. compute weighted cov. matrix: $\boldsymbol{M}'_W = \sum_{j=1}^{m} \omega_j \boldsymbol{x}'_j {\boldsymbol{x}'_j}^T$
5. compute Eigenvector $\boldsymbol{v}_1$ of smallest Eigenvalue of $\boldsymbol{M}'_W$

- Output: return plane through $\bar{\boldsymbol{x}}$ orthogonal to $\hat{\boldsymbol{n}}^* = \pm \boldsymbol{v}_1$.

Difficulties in surface reconstruction

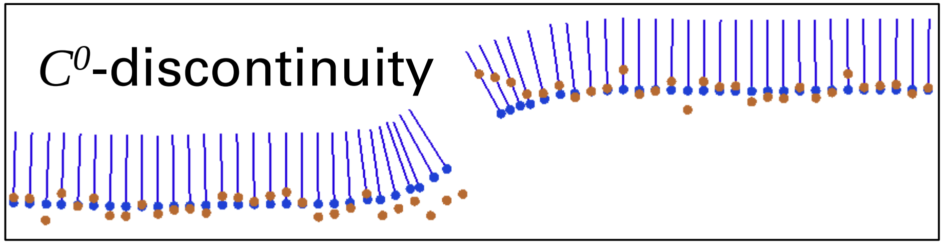


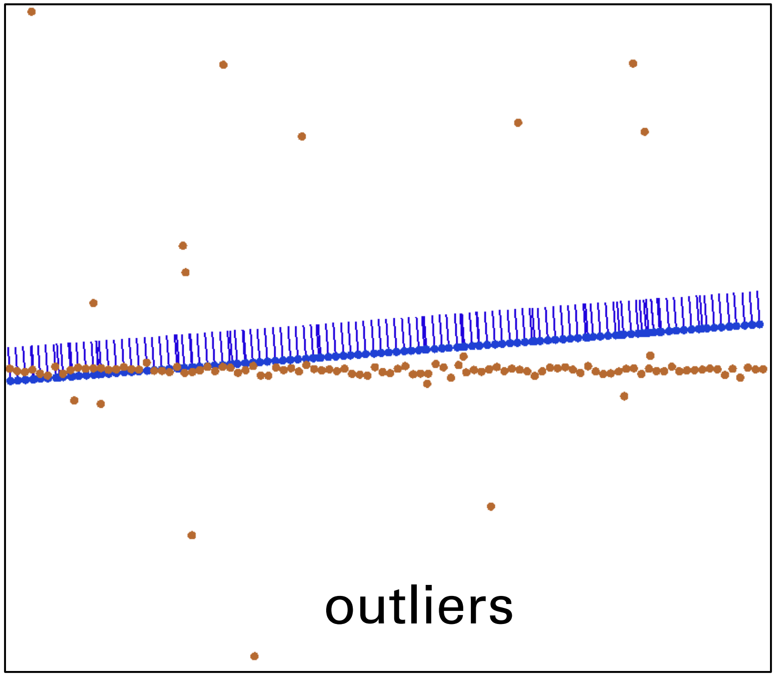
Problems of weighted Tanget Space Fit

- outliers and C^0-discontinuities significantly influence fit

➔ use robust norm $\rho(r_j)$

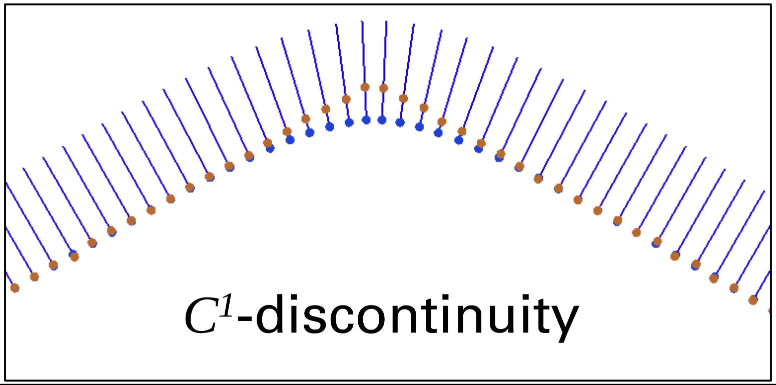
$$f(\boldsymbol{n}') = \sum_j \rho(r_j)$$





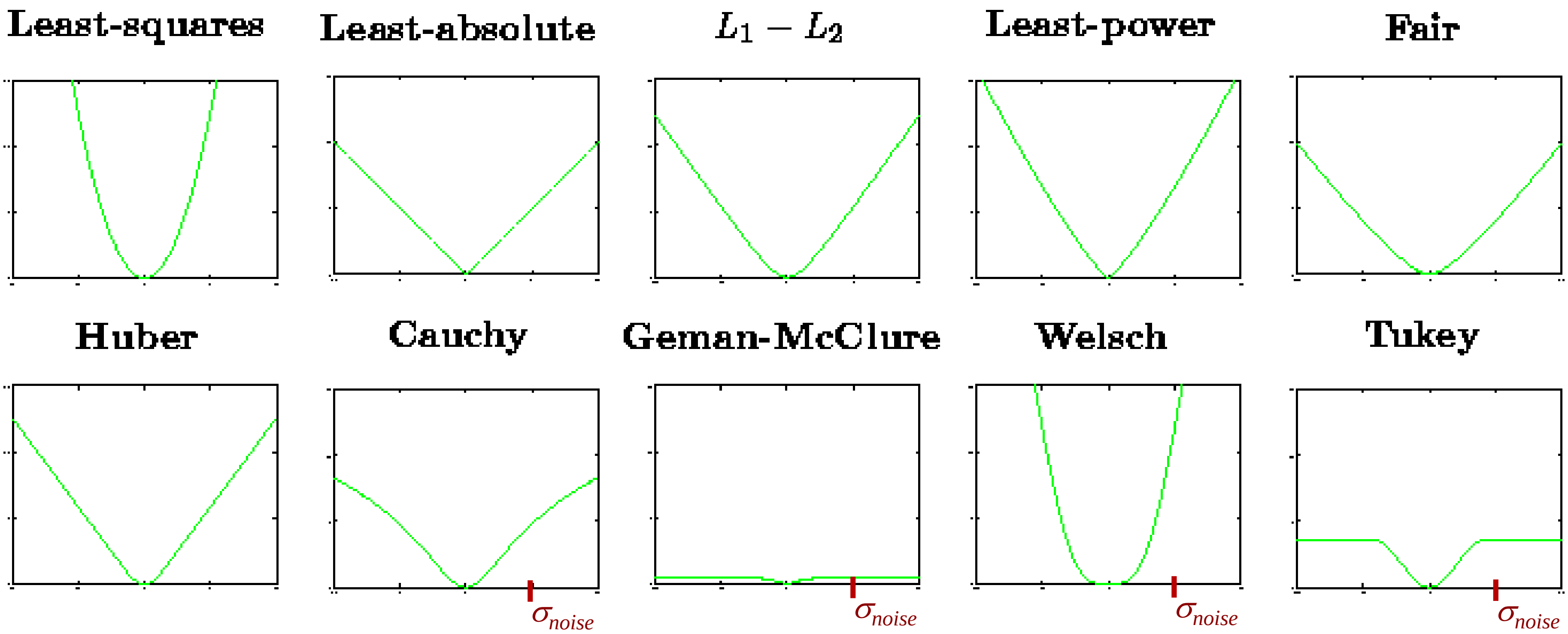
- C^1-discontinuities are smoothed out

➔ bilateral weights



Selection of robust norms

- On infinite plane with Gaussian noise, the L_2-norm $\rho(r) = r^2$ is optimal
- Otherwise (always) a large number of norms can be chosen from, which partially need to be scaled by noise scale σ_{noise}:



Iterated Re-weightes Least Squares

Computer Graphics and Visualization

- IRLS suitable for convex norms like the p-norm $\rho(r) = |r|^p$
- **Idea:** choose weights not for localization but to emulate robust norm with weighted least squares fit
- given robust norm $\rho(r)$ introduce influence function

$$\psi(r) := \frac{\partial\rho}{\partial r}(r)$$

- compare weighted least squares with robust norm:

$$f_2(\theta) = \frac{1}{2}\sum_j \omega_j r_j(\theta)^2 \iff f_\rho(\theta) = \sum_j \rho\big(r_j(\theta)\big)$$

- condition for optimum with respect to parameter vector θ

$$0 = \frac{\partial f_2}{\partial\theta} = \sum_j \omega_j r_j \frac{\partial r_j}{\partial\theta} \iff 0 = \frac{\partial f_\rho}{\partial\theta} = \sum_j \psi(r_j)\frac{\partial r_j}{\partial\theta}$$

- choose weights to identify both conditions: $\omega_j = \frac{\psi(r_j)}{r_j}$
- we finally need a strategy to ensure the last identity

Iterated Re-weightes Least Squares

IRLS solution strategy

- iteratively re-compute weights
- add superscript to weights that marks iteration number l

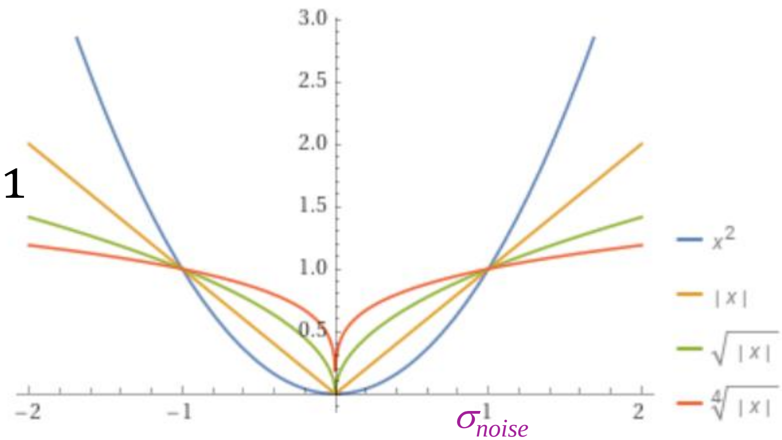
1. start with regular least squares fit: $\omega_j^{l=0} \equiv 1$
2. perform weighted least squares fit with ω_j^l yielding parameters θ^l and residua $r_j^l = r_j(\theta^l)$
3. starting with second iteration check f_ρ or θ for termination
4. re-compute weights $\omega_j^{l+1} = \frac{\psi(r_j^l)}{r_j^l}$ and `goto` 2.

- typically a small number of iterations are sufficient
- **homework**: combine IRLS with localization weighting

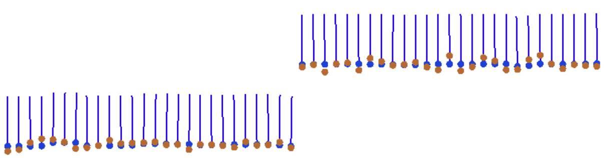
IRLS discussion

- A good family of robust norms is Minkowski norm for $0 < \nu < 1$:

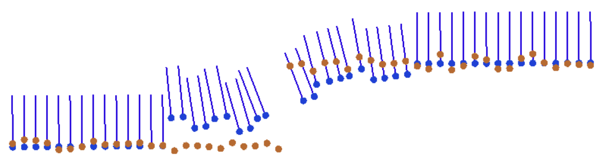
$$\rho(r) = \frac{1}{\nu}\left|\frac{r}{\sigma_{noise}}\right|^{\nu} \text{and } \psi(r) = \left|\frac{r}{\sigma_{noise}}\right|^{\nu-1}$$



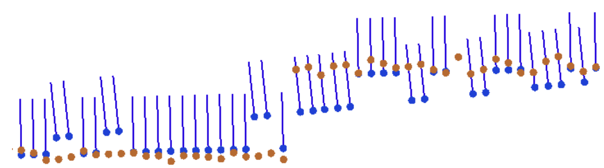
- choice of localization and noise scales is important:



σ_{loc} and σ_{noise} good



for σ_{loc} too large



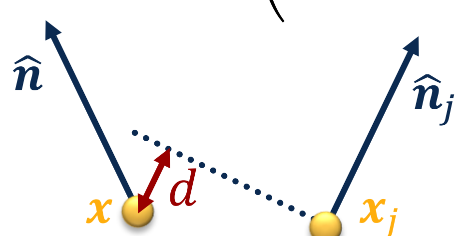
for σ_{noise} too large

- for depth sensors the noise scale depends on the viewing angle and is not easy to estimate from the data

bilateral weighting

- To support sharp creases and corners, a second weight π_j is multiplied to the localization weight: $\omega_j^{bil} = \omega_j \cdot \pi_j$
- π_j decreases with increasingly different tangent spaces
- Two choices have been proposed:
  1. normal distance
  $$\pi_j^{\hat{\boldsymbol{n}}} = \exp\left(-\frac{\left\|\hat{\boldsymbol{n}}_j - \hat{\boldsymbol{n}}\right\|^2}{\sigma_{\hat{\boldsymbol{n}}}^2}\right)$$
  2. plane distance

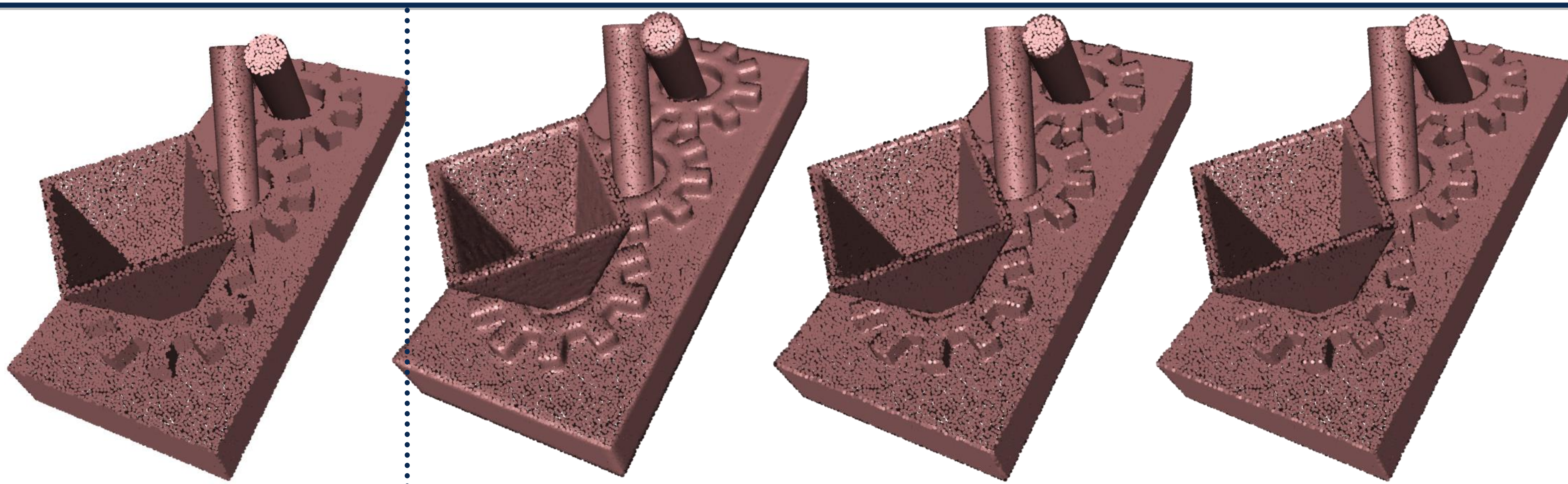
[Fleishman03]
  $$\pi_j^{d} = \exp\left(-\frac{\left\|\overbrace{\hat{\boldsymbol{n}}_j^T \boldsymbol{x} - c_j}^{d}\right\|^2}{\sigma_d^2}\right)$$



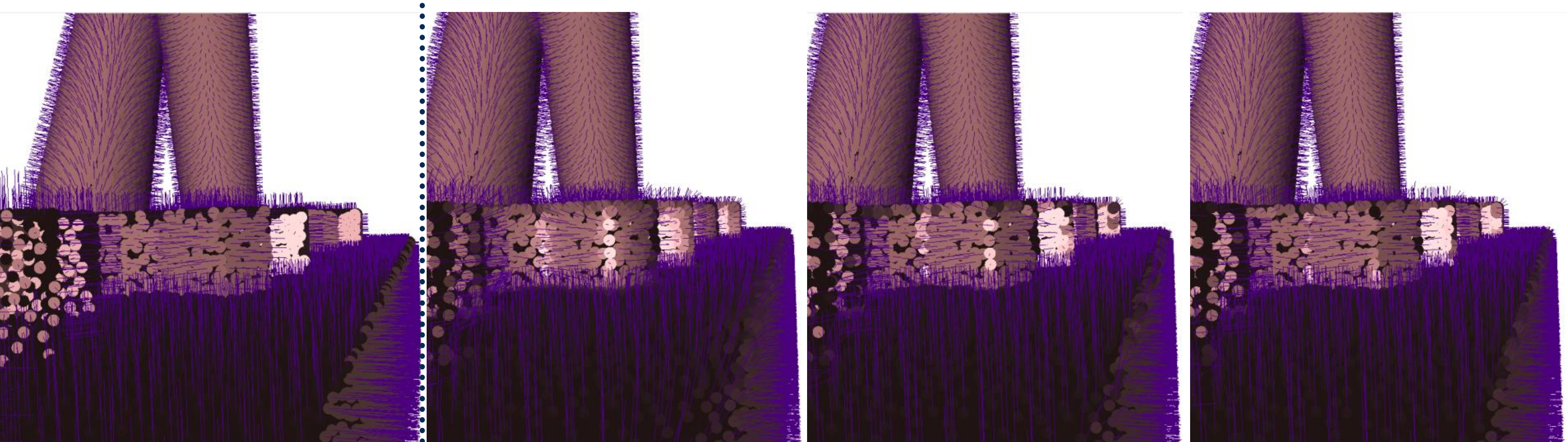
- The bilateral weights depend on the to be estimated normals and couple the local optimzation problems into a global, non-linear optimization problem
- Typically, the following simple solution strategy is used:

1. compute neighbor graph
2. initialize all bilateral weights to 1
3. fit tangent plane $(\hat{\boldsymbol{n}}_i, c_i)$ at each point $\boldsymbol{x}_i$
4. iterate till convergence
   - compute bilateral weights
   - fit tangential planes with new weights

example convergence

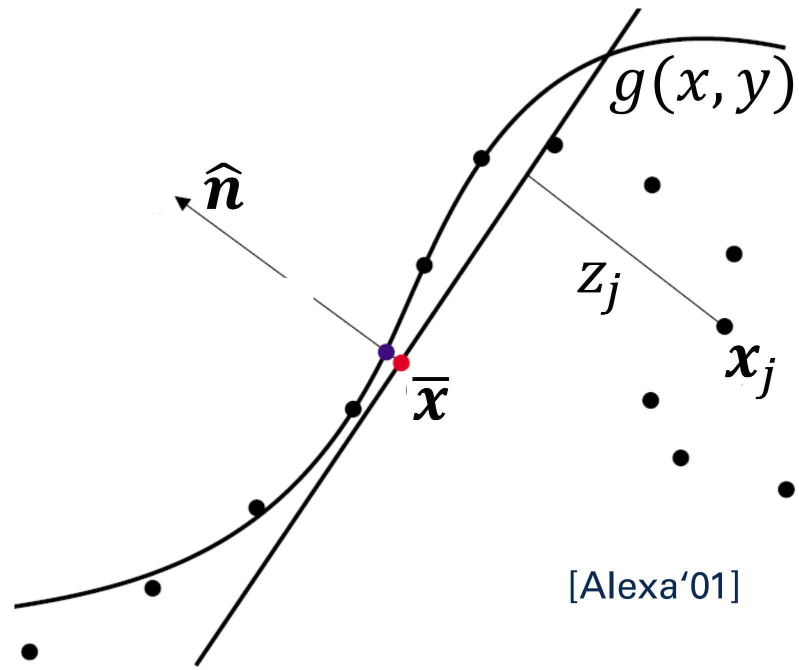


ground truth | iteration 0 | iteration 1 | iteration 2



Curvatures and Higher Order

- After fitting a tangent space through $\bar{\boldsymbol{x}}$ orthogonal to $\hat{\boldsymbol{n}}$, one can define a local 2D coordinate system in the tangent space (x, y)
- The neighboring points can be transformed to local coordinates with z along the normal direction
- We can fit a height field $g(x, y)$ in a polynomial description to the neighboring points and use the tailor series of $g(x, y)$ around the 2D origin to compute the curvature properties



[Alexa'01]

Curvatures and Higher Order

Computer Graphics and Visualization

Local Polynomial Fit

- monoms are the simplest basis and can be built for increasing degrees:
  degree 0: $1,$
  degree 1: $x, y,$
  degree 2: $x^2, xy, y^2,$
  degree 3: x^3, x^2y, xy^2, y^3
- $g(x, y)$ is a linear combination of the basis functions, which can be transformed to vector notation:

$$g(x,y) = a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 xy + a_5 y^2 + \cdots$$
$$= \sum_i a_i \phi_i(x,y) = \langle \vec{\boldsymbol{a}}, \vec{\phi}(x,y) \rangle$$

- This is a standard least squares problem with:
$$r_j = g(x_j, y_j) - f_j$$
- and weights Gaussian
$$\omega_j \propto \exp\left(-\left\|\boldsymbol{x}_j - \overline{\boldsymbol{x}}\right\|^2 / \sigma_{loc}^2\right)$$
- The solution can be computed via the normal equations
$$\boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A} \vec{\boldsymbol{a}} = \boldsymbol{A}^T \boldsymbol{W} \vec{\boldsymbol{f}}$$
- with $\boldsymbol{A}_{ji} = \phi_i(x_j, y_j)$ and $\vec{\boldsymbol{f}}_j = f_j$.
- Solve with weighted pseudo inverse or with SVD.
- estimate curvature from curvature of $g(x = 0, y = 0)$.
- compare CG1 script on surface analysis,

with $\underline{\boldsymbol{s}}(x,y) = \begin{pmatrix} x \\ y \\ g(x,y) \end{pmatrix}$

Spin Images [Johnson'97]

- describe larger neighborhood of point with a histogram over angle α and height β
- for this compute tangent space and local coordinate frame
- per neighbor point measure angle α to x-axis and local height β over tangent plane
- compute histogram on grid vertices (not on cells) by adding weights of bilinear interpolation (resulting in extrapolation)

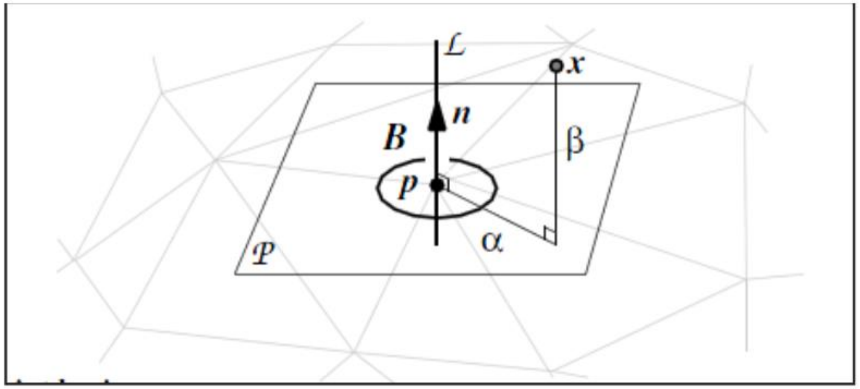


Figure 1: The cylindrical coordinate system and its (p, n) 2-D basis (Taken from [2]).

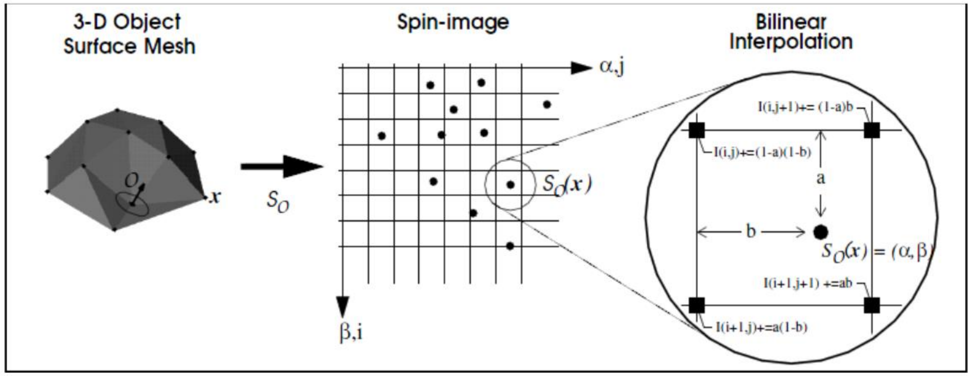
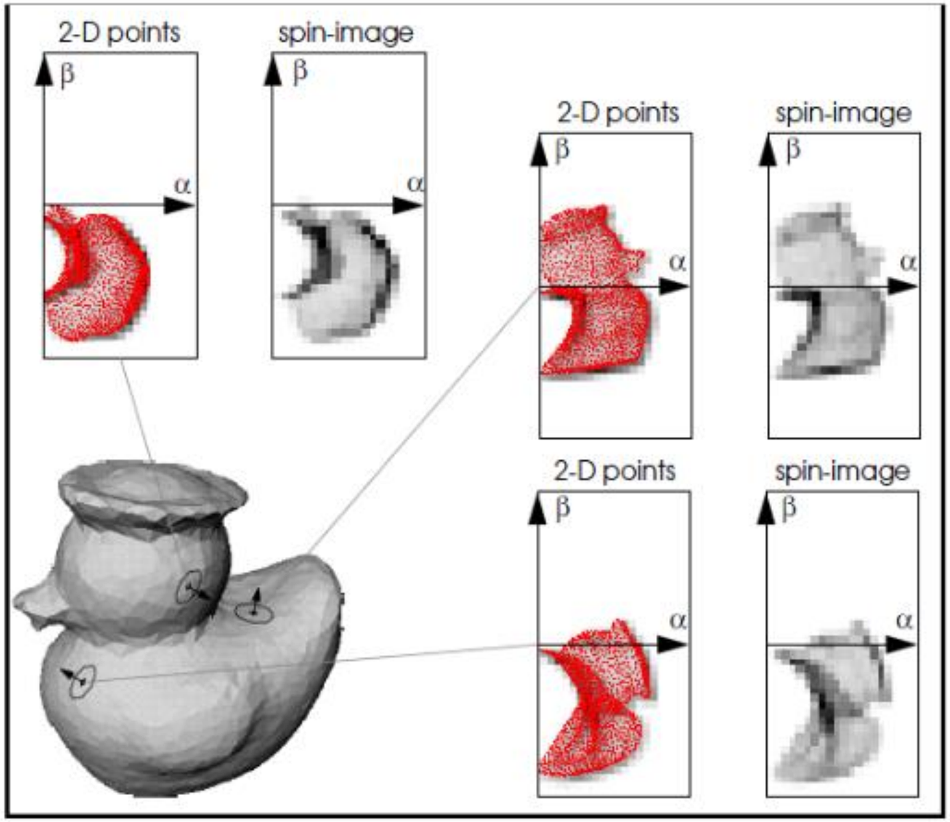


Figure 2: Creation of the 2-D array representation of a spin image (Taken from [2]).

Spin Images

- Spin images can well distinguish different local surface types
- They are used frequently in shape matching approaches
- choice of x-axis is a degree of freedom that cyclically translates spin image in α-direction
- to make matching approaches robust against choice of x-axis, identify all α-translations of splin image



Literature

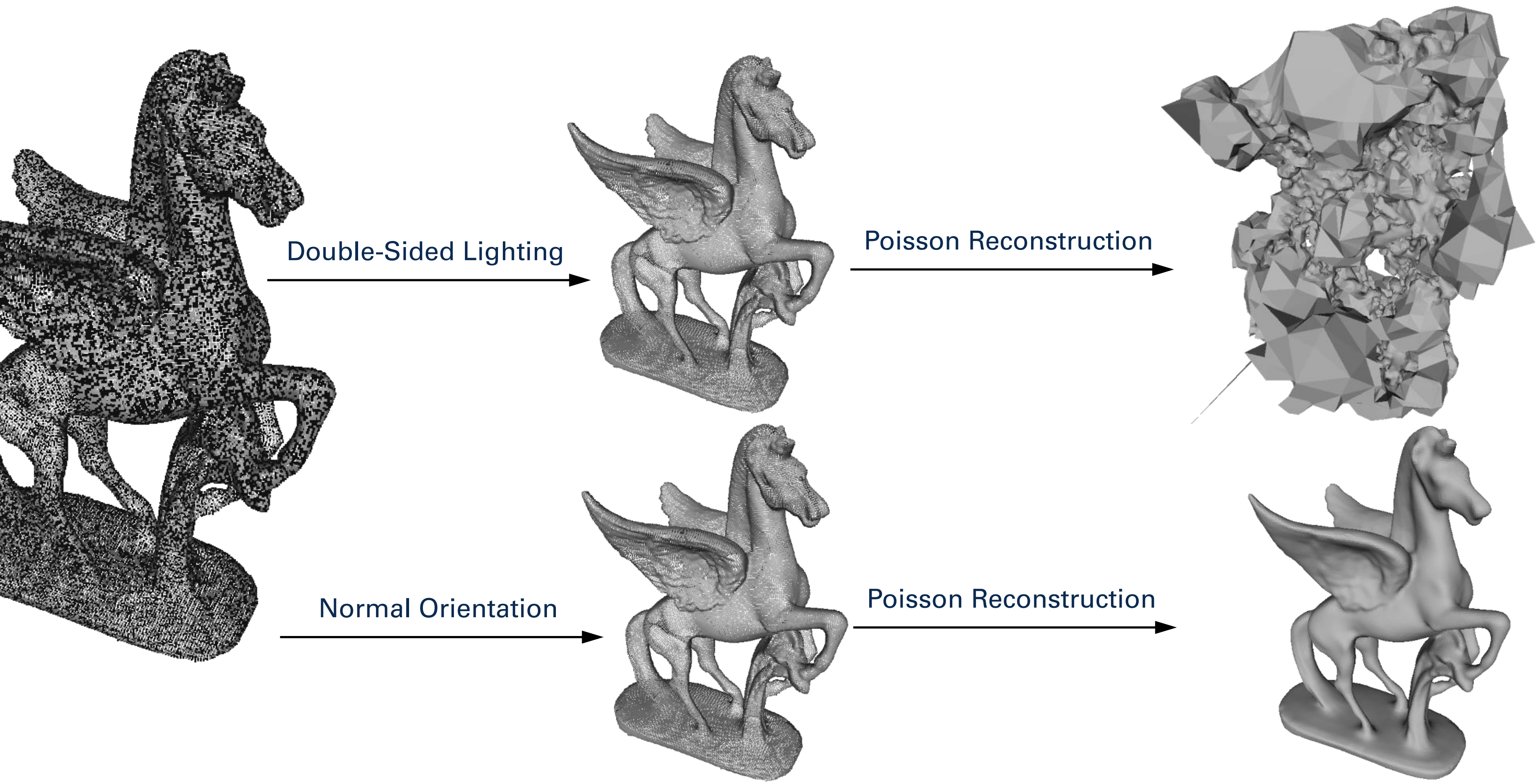
- [Johnson'97] … A. E. Johnson. *Spin-Images: A Representation for 3-D Surface Matching*. PhD thesis, Carnegie Mellon University, 1997.
- [Alexa'01] … M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C. T. Silva. *Point set surfaces*. IEEE Visualization'01, acm.
- [Fleishman03] … S. Fleishman, I. Drori, D. Cohen-Or. *Bilateral mesh denoising*. SIGGRAPH'03, doi.
- [Jones03] … T. R. Jones, F. Durand, M. Desbrun. *Non-iterative, feature-preserving mesh smoothing*. SIGGRAPH'03, doi
- Y. Lipman, D. Cohen-Or, D. Levin. 2006. *Error bounds and optimal neighborhoods for MLS approximation*. SGP '06, acm

ORIENTATION OF NORMALS

Motivation Normal Orientation Problem

Computer Graphics and Visualization

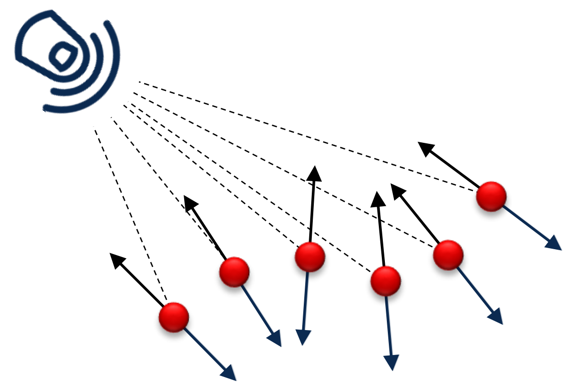
- consistent normal orientation is precondition of several surface reconstruction techniques

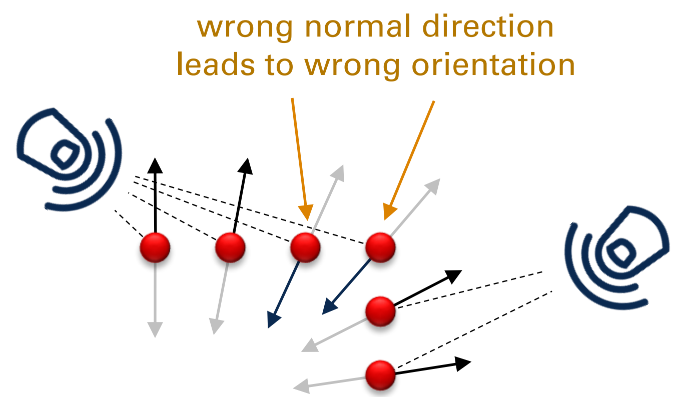


Motivation Normal Orientation Problem

Computer Graphics and Visualization

- tangent plane fitting does not give sign of normal
- sign defines outside direction which needs to be globally consistent
- position of 3D scanner defines outside direction, but
  - information is often lost
  - difference between computed normal directions to original surface normals can lead to wrong orientation estimate





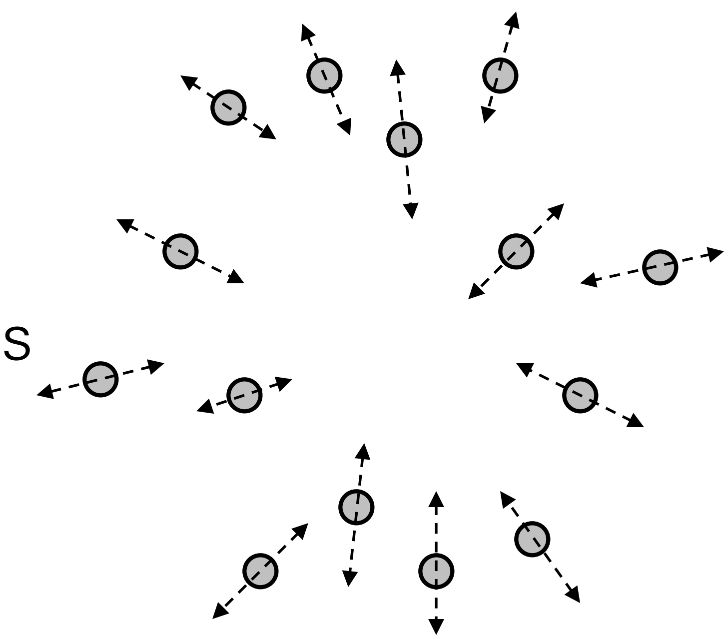
Overview on normal orientation

normal orientation problem

- input: points $\boldsymbol{x}_i$ with normals $\hat{\boldsymbol{n}}_i$
- output: one sign s_i per normal where
  - $s_i = +1$ … keep normal
  - $s_i = -1$ … negate normal

solution strategy

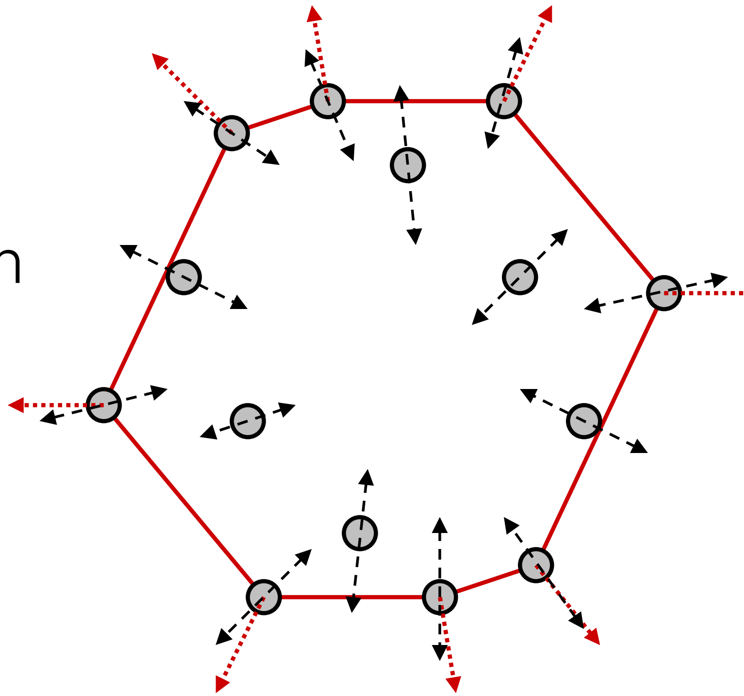
- estimate one or several normals from additional knowledge
- build neighbor graph and propagate normal orientation along graph edges



normal orientation solution strategy

Computer Graphics and Visualization

- initialization: estimate one or several normals per connected component from additional knowledge (i.e. outside direction of convex hull)
- build neighbor graph (i.e. Riemann graph with $k = 16$)
- weight each graph edge by unreliability measure
- define flip criterion and propagate normal orientations from initialization over graph edges by one of the two strategies:
  1. propagate orientation along minimal spanning tree, or
  2. setup global unreliability minimization problem and solve with approximate solver [Schertler16]



flip criteria and unreliability measures

- [Hoppe92] locally assumes planar surface and directly compares normals of points incident to edge:

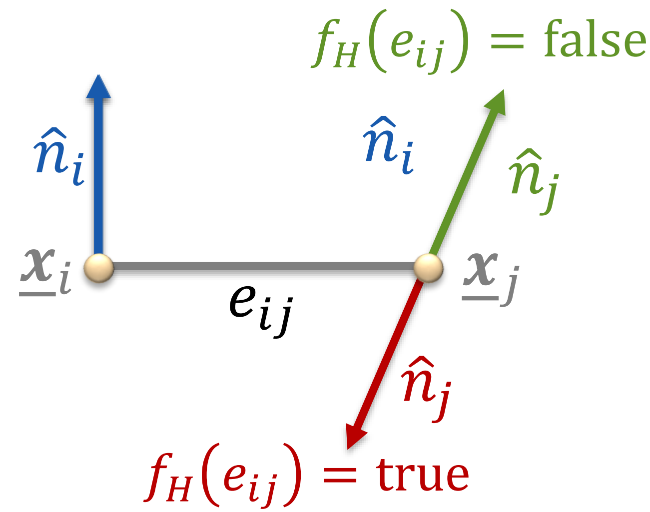
$$f_H(e_{ij}) \coloneqq \langle \hat{n}_i, \hat{n}_j \rangle < 0$$

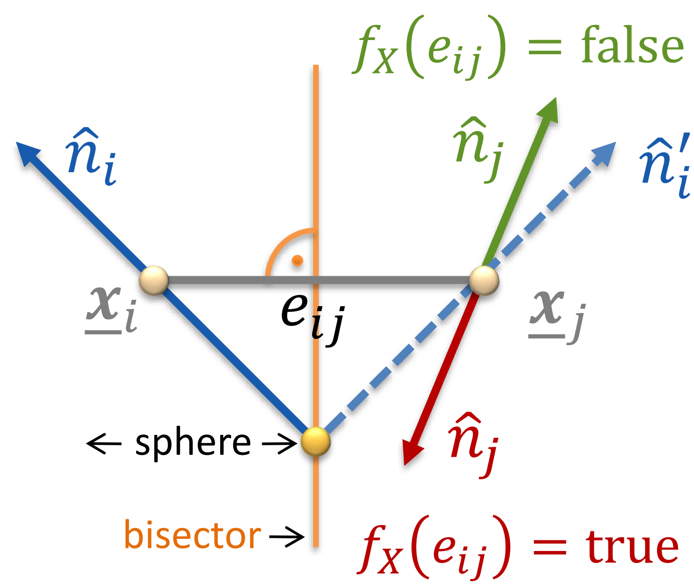
- Hoppe measures reliability from the absolute value of cosine of angle between the normals

$$u_H(e_{ij}) \coloneqq 1 - |\langle \hat{n}_i, \hat{n}_j \rangle|$$

- [Xie03] assume constant curvature and transport normal by reflection at edge bisector. Flip criterion and unreliability measure are defined as in Hoppe's approach:

$$f_X(e_{ij}) \coloneqq \langle \hat{n}_i', \hat{n}_j \rangle < 0$$
$$u_X(e_{ij}) \coloneqq 1 - |\langle \hat{n}_i', \hat{n}_j \rangle|$$

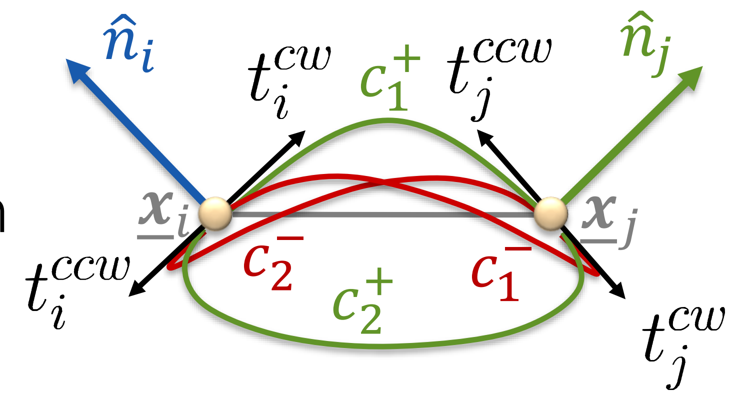


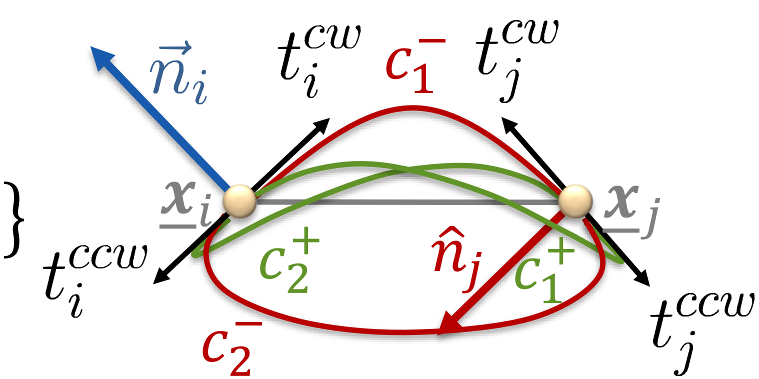


flip criteria and unreliability measures

- [König09] proposes to define flip criterion and unreliability measure from curve complexity
$$C(c_k) = \int |\kappa_k(s)| ds$$
- 2D coordinate system is built from edge $\hat{e}$ and $\hat{e} \times (\hat{n}_i \times \hat{n}_j)$.
- for both orientations $s_j = \pm 1$ two Hermit curves c_1^+, c_2^+ and c_1^-, c_2^- are defined to connect points
- flip criterion and unreliability are defined from smaller curve complexity $C^\pm = \min\{C(c_1^\pm), C(c_2^\pm)\}$
$$f_K(e_{ij}) := C^- < C^+$$
$$u_K(e_{ij}) := \frac{\min\{C^-, C^+\}}{\max\{C^-, C^+\}}$$

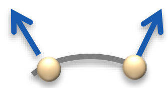
$f_K(e_{ij}) = \text{false}$

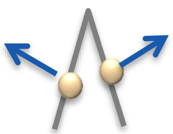


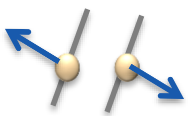


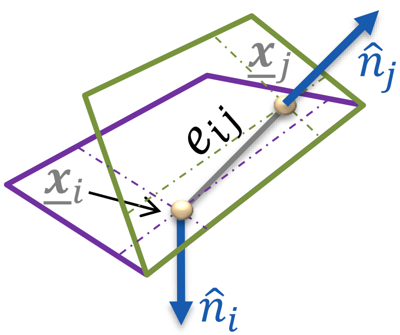
$f_K(e_{ij}) = \text{true}$

Comparison

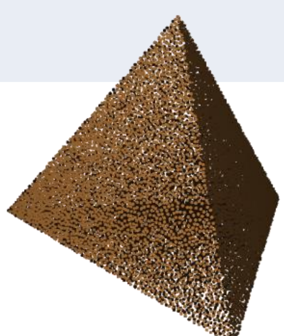








Case	Hoppe	Xie	König
Smooth surface / low curvature	++	++	++
High curvature / acute angles	-	++	++
nearby sheets	-	+	+
high normal twist along edge	-	-	+



Minimal Spanning Tree (MST)

- given initial orientations and a connected graph of edges with unreliability as cost
- MST can be computed with Kruskal's algorithm in $O(m \log m)$ for m edges and minimizes unreliability.
- The idea is to add edges with increasing unreliability starting from least unreliable one and to avoid cycles by tracing sets of connected graph vertices with union find data structure (compare CG1)

```
vector MST_Kruskal(n,E,C)
  vector MST;
  union_find U(n);
  sort (E,C) increasingly
  iterate e=(vi,vj) in E {
     if (U.find(vi) !=
         U.find(vj)) {
        MST.push_back(e);
        U.union(vi, vj);
     }
  }
  return MST;
```

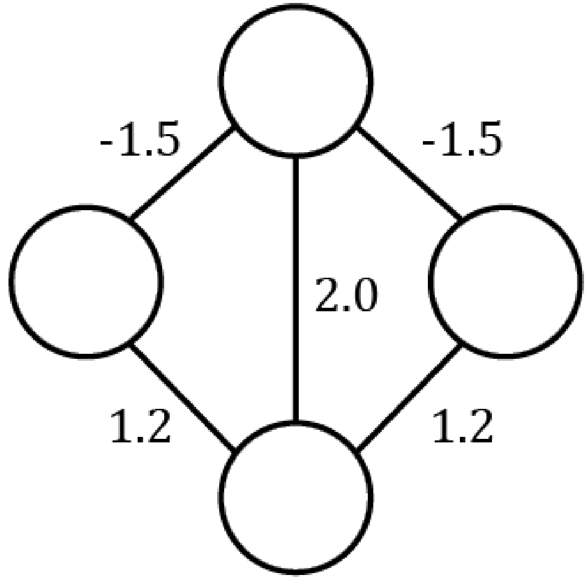
avoids cycles

Pseudo code of Kruskal's algorithm using the union find data structure

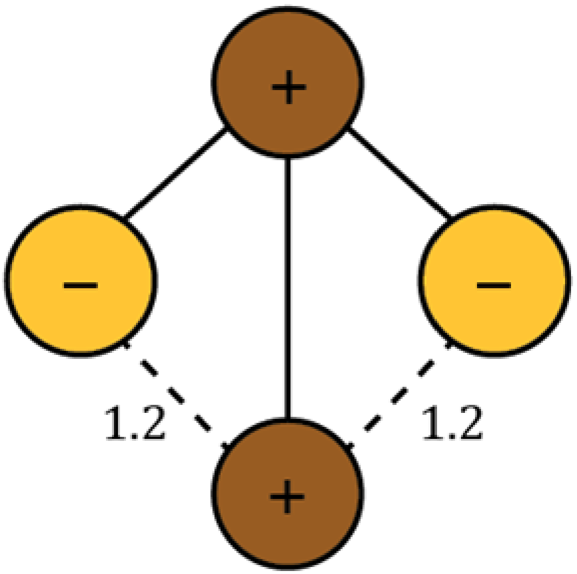
(see also www.youtube.com/watch?v=3fU0w9XZjAA)

Motivation of Global Optimization

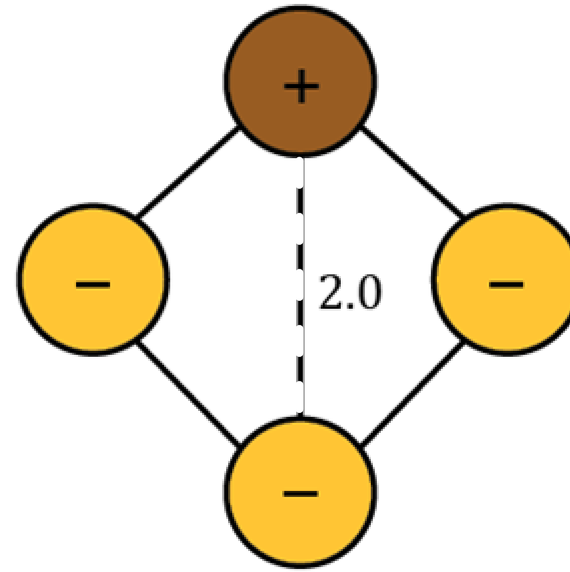
alternative view:
maximize reliability
$r(e_{ij}) := |\langle \hat{n}_i, \hat{n}_j \rangle| = 1 - u(e_{ij})$



example of distance-weighted output of reliability measure
(sign encodes flip criterion)



Maximum Spanning Tree greedily minimizes reliability of sum of contradicted flip criteria solution, E=2.4



Optimal Solution of the minimization of sum of contradicted broken flip criteria: E=2.0

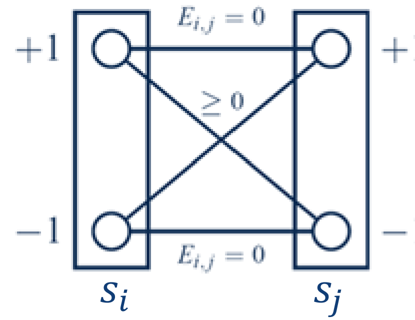
Global Orientation Problem

- use sign $s_i \in \{-1,+1\}$ as point label
- per edge in neighbor graph define energy for label assignments that defaults to zero.
- for assignments that contradict flip criterion, assign positive term with a distance decreasing weight ω_{ij}:

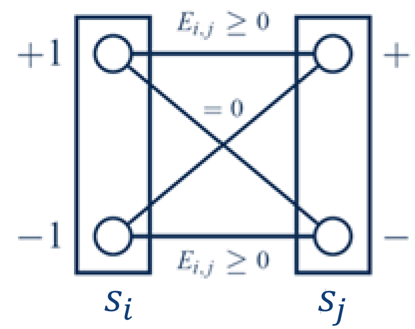
$$E_{i,j} = r_*(e_{ij})\underbrace{\left(1 - \frac{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2}{l_{\max}^2}\right)}_{\omega_{ij}\in[0,1]}$$

  with the maximum edge length $l_{\max}$ in the graph
- finally optimize labels

$$L^* = \underset{S=\{-1,+1\}^{|\mathcal{P}|}}{\arg\min} \sum_{(i,j)\in\mathcal{E}} E_{i,j}(s_i, s_j)$$



$f_*(e_{ij})$ =false



(b) $\phi(i,j) < 0$

$f_*(e_{ij})$ =true

Global Orientation Problem

- problem is NP-hard ➔ only semi-approximate solvers like QPBO (Quadratic Pseudo-Boolean Optimization) feasible

Xie, MST   Xie, MST + QPBO-I   Hoppe, MST   Hoppe, MST + QPBO-I

Literature

- [Hoppe92] … H. Hoppe, T. DeRose, T. Duchamp, J. MCDonald, W. Stuetzle. *Surface reconstruction from unorganized points*. SIGGRAPH 1992.
- [Xie03] … H. Xie, J. Wang, J. Hua, H. Qin, A. Kaufman. *Piecewise C1 continuous surface reconstruction of noisy point clouds via local implicit quadric regression*. IEEE Visualization 2003,
- [König09] … S. König, S. Gumhold. *Consistent Propagation of Normal Orientations in Point Clouds*. VMV 2009
- [Schertler16] … N. Schertler, B. Savchynskyy, S. Gumhold. *Towards Globally Optimal Normal Orientations for Large Point Clouds*. CGF, doi:10.1111/cgf.12795

FEATURE EXTRACTION

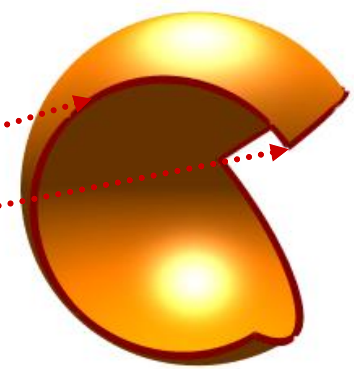
Classification of Surface Points

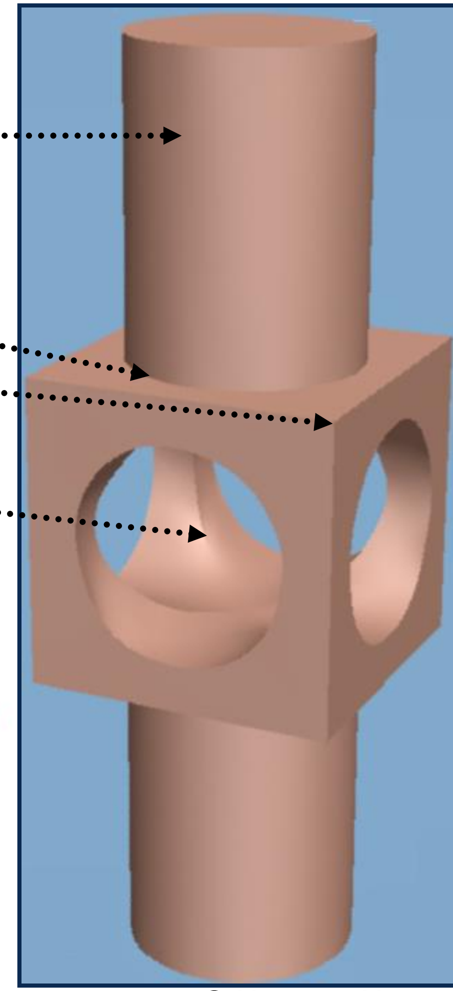
Mannifold

- smooth surface point
- sharp crease point
- sharp corner
- „<u>darts</u> “: transition from a sharp edge into a planar area

Mannifold with Boundary

- smooth border curve
- border corner





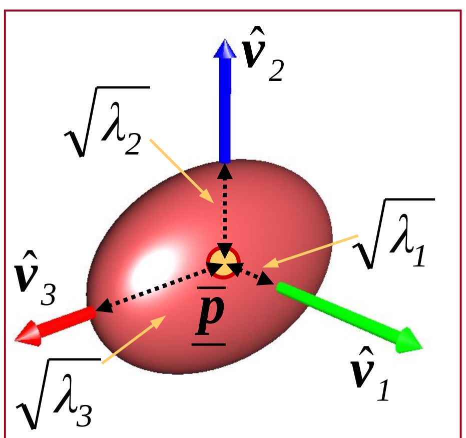
© Hughes Hoppe

corners, darts and border corner are hard to detect locally

Classification with covariance matrix

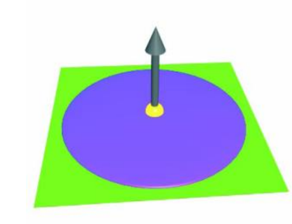
Local Point Density Analysis

- given m neighbor points $\underline{\boldsymbol{p}}_j$ with weights ω_j we compute
  - barycenter: $\underline{\overline{\boldsymbol{p}}} = \sum_{j=1}^{m} \omega_j \underline{\boldsymbol{p}}_j$ and
  - covariance matrix: $\boldsymbol{C} = \sum_{j=1}^{m} \omega_j \vec{\boldsymbol{p}}'_j \vec{\boldsymbol{p}}'^{T}_j$ with $\vec{\boldsymbol{p}}'_j = \underline{\boldsymbol{p}}_j - \underline{\overline{\boldsymbol{p}}}$
- ellipsoid representing local point density from Eigen decomposition $\boldsymbol{C} = \lambda_1 \hat{\boldsymbol{v}}_1 \hat{\boldsymbol{v}}_1^T + \lambda_2 \hat{\boldsymbol{v}}_2 \hat{\boldsymbol{v}}_2^T + \lambda_3 \hat{\boldsymbol{v}}_3 \hat{\boldsymbol{v}}_3^T$ with $\lambda_1 \leq \lambda_2 \leq \lambda_3$
- $\sqrt{\lambda_i}$ are radii of *represen-tative ellipsoid*:

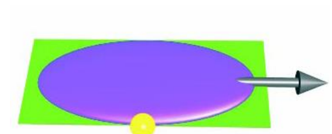


Feature Detection

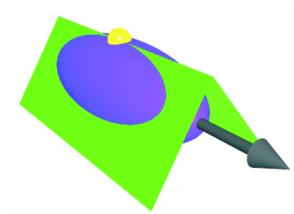
- representative ellipsoid is suitable for characterization of neighborhood of point $\boldsymbol{p}_k$:
- smooth surface:
  - $0 \approx \lambda_1 << \lambda_2 \approx \lambda_3$
  - $\underline{\overline{\boldsymbol{p}}} \approx \underline{\boldsymbol{p}}_k$



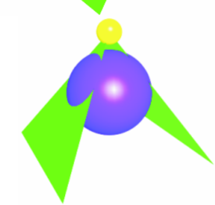
- smooth border:
  - $0 \approx \lambda_1 << \lambda_2 \approx ½\, \lambda_3$
  - $\left\| \underline{\overline{\boldsymbol{p}}} - \underline{\boldsymbol{p}}_k \right\| \approx \sqrt{\lambda_2}$



- sharp crease
  - $0 << \lambda_1 \approx \lambda_2 < \lambda_3$
  - $\left\| \underline{\overline{\boldsymbol{p}}} - \underline{\boldsymbol{p}}_k \right\| \approx \sqrt{\lambda_2}$



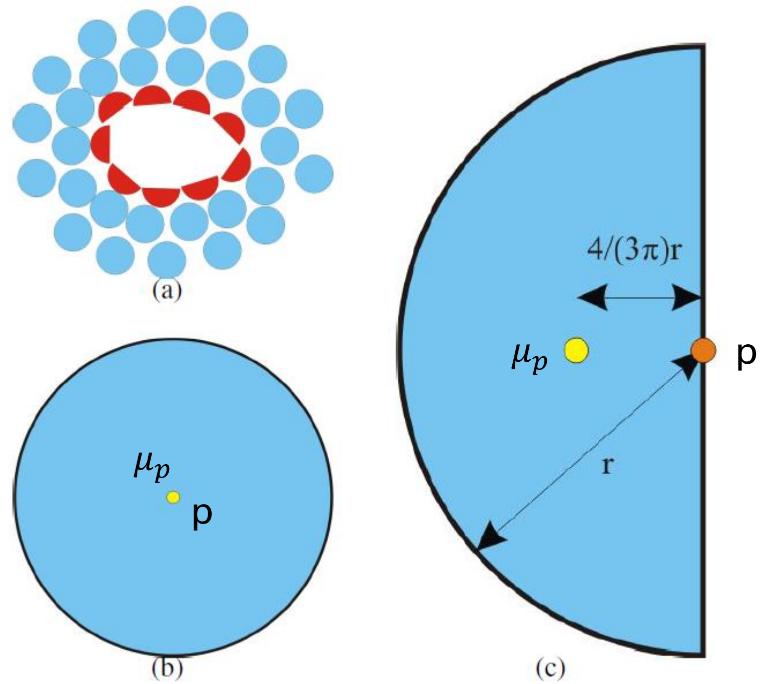
- corner
  - $0 << \lambda_1 \approx \lambda_2 \approx \lambda_3$



Alternative Border Detection 1

Half-Disk Criterion

- The local neighborhood of points located on the surface boundary is homeomorphic to a half-disk as opposed to the full disk of an interior point
- For an interior point, the average μ_p of the neighborhood points will coincide with the interior point itself
- for a boundary point, it will deviate in direction of the interior of the surface.



Boundary Probability:

$$\Pi_\mu(\mathbf{p}) = \min\left(\frac{\|\mathbf{p} - (\mu_\mathbf{p})\|}{\frac{4}{3\pi} r_\mathbf{p}}, 1\right)$$

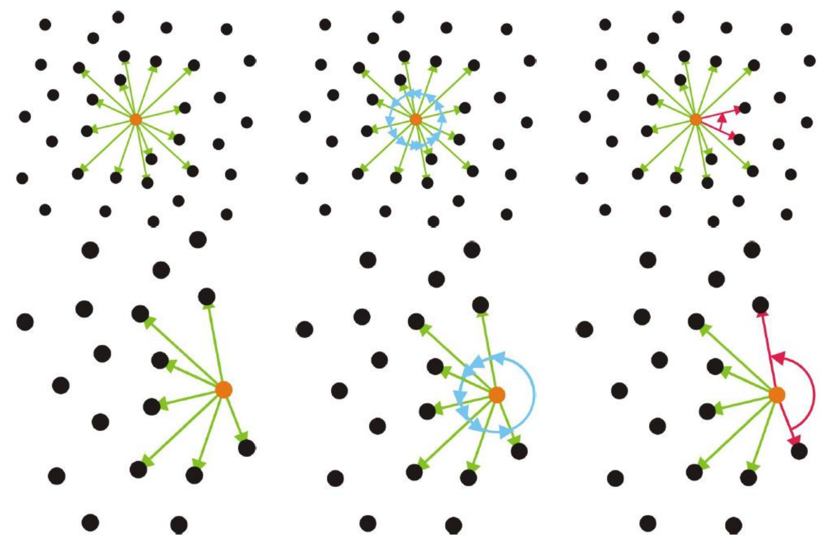
average distance to neighboring points

Alternative Border Detection 2

Computer Graphics and Visualization

Angle Criterion

- Project neighboring points onto tangential plane
- Sort projected points according to their angle around the center point p
- find largest gap g
- g will be significantly larger for a boundary point than for an interior point



Boundary probability:

$$\Pi_\angle(\mathbf{p}) = \min\left(\frac{g - \frac{2\pi}{|N_\mathbf{p}|}}{\pi - \frac{2\pi}{|N_\mathbf{p}|}}, 1\right).$$

number of neighbor points

Feature Line Extraction [Gum01]

compute neighbor graph → compute represent. ellipsoids → minimal spanning pattern → prune short arms → fit spline and reconstruct corners

- MST weights are defined for creases and border separately (formula see paper)

```
iterate over all edges e in neighbor graph
    if weight(e).penalty < τ then
        queue.insert(e, weight(e))

while not queue.empty() do
e = (p,q) := queue.extractMinimum()
    if disjointSet.findAndUpdate(p,q) or
       pattern.pathIsLonger(p,q,ρ) then
         pattern.addEdge(p,q)
```

Literature

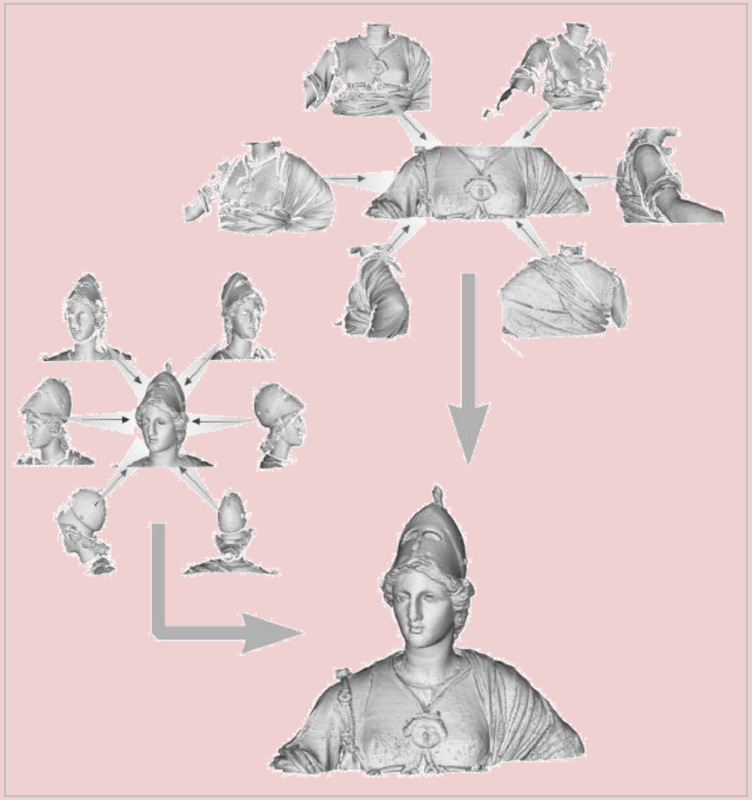
Computer Graphics and Visualization

- [Gum01] … Gumhold, Stefan, Xinlong Wang, and Rob S. MacLeod. "Feature Extraction From Point Clouds." IMR. 2001
- Pauly, Mark, Richard Keiser, and Markus Gross. "Multi-scale Feature Extraction on Point-Sampled Surfaces." *Computer graphics forum*. Vol. 22. No. 3. Blackwell Publishing, Inc, 2003.
- Daniels, Joel II, et al. "Robust smooth feature extraction from point clouds." *Shape Modeling and Applications, 2007. SMI'07. IEEE International Conference on*. IEEE, 2007.

REGISTRATION

3D Dataset Registration (CG1)

- Registration is the process of bringing two data sets into a joint coordinate system based on feature correspondences.
  - common features are points, lines and planes
  - common correspondences are point-to-point, point-to-plane or line-to-line
  - one distinguishes between rigid and non-rigid registration
- For registration of 3D scans we consider rigid registration with point-to-point or point-to-plane correspondences.
- Given two scans A & B we want to find a rigid transformation T s.t. $A = T(B)$
- Standard approach: ICP algorithm



for acquisition of a 3D Model several 3D-Scans from different view points need to be transformed into a common coordinate system and then fused

Iterative Closest Points (ICP) Algorithm

- Input: two point clouds and coarse initial alignment
- ICP alternates between generation of correspondences based on closeness according to some distance function and the alignment according to correspondences.
- algorithm is iterative and assumes a coarse initial alignment of the 3D scans, as well as an overlap of the scans
- Pseudo-Code:
  1. find coarse initial alignment T_0 (you can use markers, geometry features or do it manually)
  2. find correspondences: subsample A and B to S_A and S_B and find $\forall a \in S_A$ closest point $b_a \in B$ and define (a, b_a) as correspondence (similarly $\forall b \in S_B$). Filter correspondences, for example only symmetric ones where a is closest point to b_a.
  3. compute T_{i+1} such that squared distance of all corresponding point pairs with respect to T_i is minimized (Kabsch Algorithm)

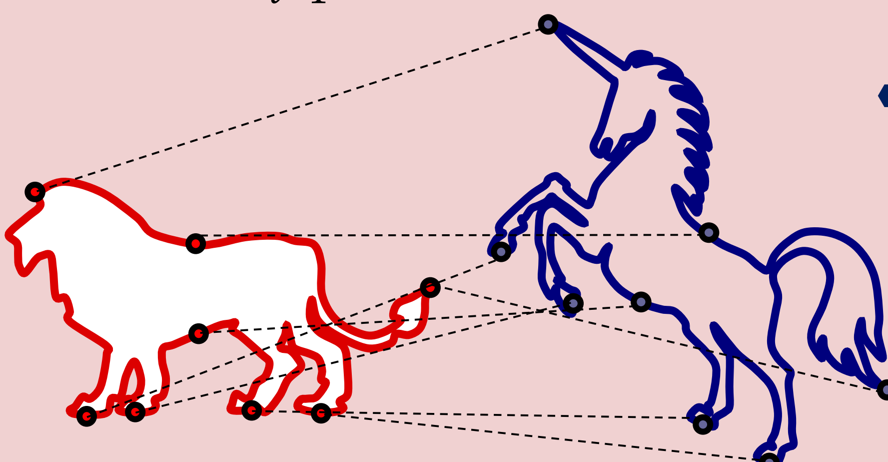
until convergence: $T_{i+1} \approx T_i$

Kabsch Algorithm

Computer Graphics and Visualization

- Input: n correspondences $\{(\boldsymbol{p}_1, \boldsymbol{q}_1), \dots, (\boldsymbol{p}_n, \boldsymbol{q}_n)\}$ on two different shapes A and B.
- Goal: find rigid transformation $(\boldsymbol{R}, \vec{\boldsymbol{t}})$ (rotation matrix $\boldsymbol{R}$ and translation vector $\vec{\boldsymbol{t}}$) that minimizes the squared distances between all point-to-point correspondences: $(\boldsymbol{R}^*, \vec{\boldsymbol{t}}^*)$

$$= \underset{\boldsymbol{R}, \vec{\boldsymbol{t}} | \boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{1}}{\text{minarg}} \sum_{i=1}^{n} \left\| \boldsymbol{p}_i - \left(\boldsymbol{R}\boldsymbol{q}_i + \vec{\boldsymbol{t}}\right) \right\|^2$$



- Translate the input points to the centroids:

$$\boldsymbol{p}'_i = \boldsymbol{p}_i - \overline{\boldsymbol{p}}, \qquad \boldsymbol{q}'_i = \boldsymbol{q}_i - \overline{\boldsymbol{q}}$$

- Compute the "covariance matrix"

$$\boldsymbol{H} = \sum_{i=1}^{n} \boldsymbol{q}'_i \boldsymbol{p}'^T_i$$

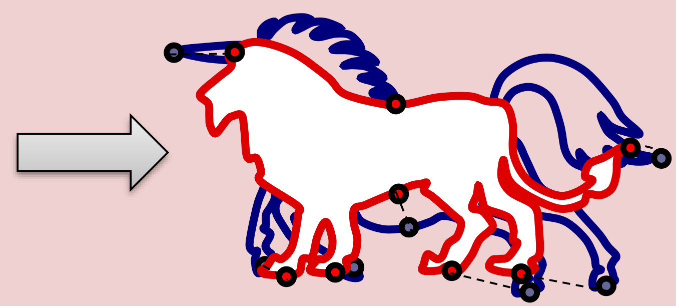
- Compute the SVD of $\boldsymbol{H}$:

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$$

- The optimal rotation is

$$\boldsymbol{R}^* = \boldsymbol{V} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\boldsymbol{V}\boldsymbol{U}^T) \end{pmatrix} \boldsymbol{U}^T$$

- translation vector is: $\vec{\boldsymbol{t}}^* = \overline{\boldsymbol{p}} - \boldsymbol{R}^*\overline{\boldsymbol{q}}$



ICP Variants [Rusinkiewicz01]

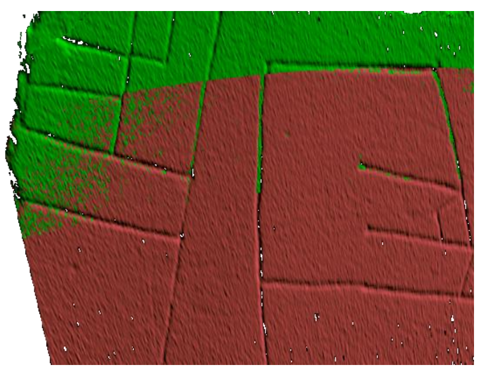
- Decompose ICP into six steps:

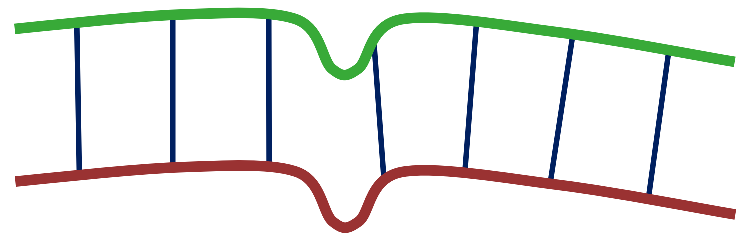
1. **Selection** of some set of points in one or both meshes.
2. **Matching** these points to samples in the other mesh.
3. **Weighting** the corresponding pairs appropriately.
4. **Rejecting** certain pairs based on looking at each pair individually or considering the entire set of pairs.
5. Assigning an **error metric** based on the point pairs.
6. **Minimizing** the error metric.

- Possible criteria for comparison
  - Speed [Rusinkiewicz01]
  - Stability
  - Tolerance with respect to noise / outliers
  - Maximum initial misalignment

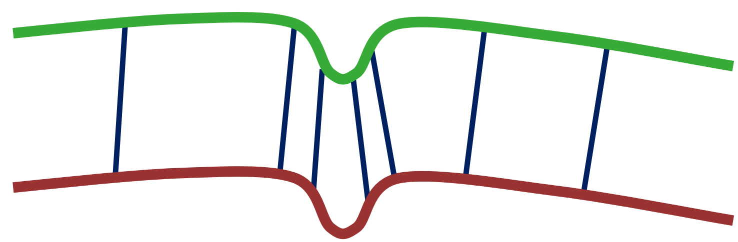
ICP Variants – Selection

- Use all points
- Uniform subsampling
- Random sampling
- Normal-space sampling
  - Ensure that samples have normals distributed as uniformly as possible
  - Demands for additional data structure
  - Is more efficient if surface has narrow features





Uniform Sampling

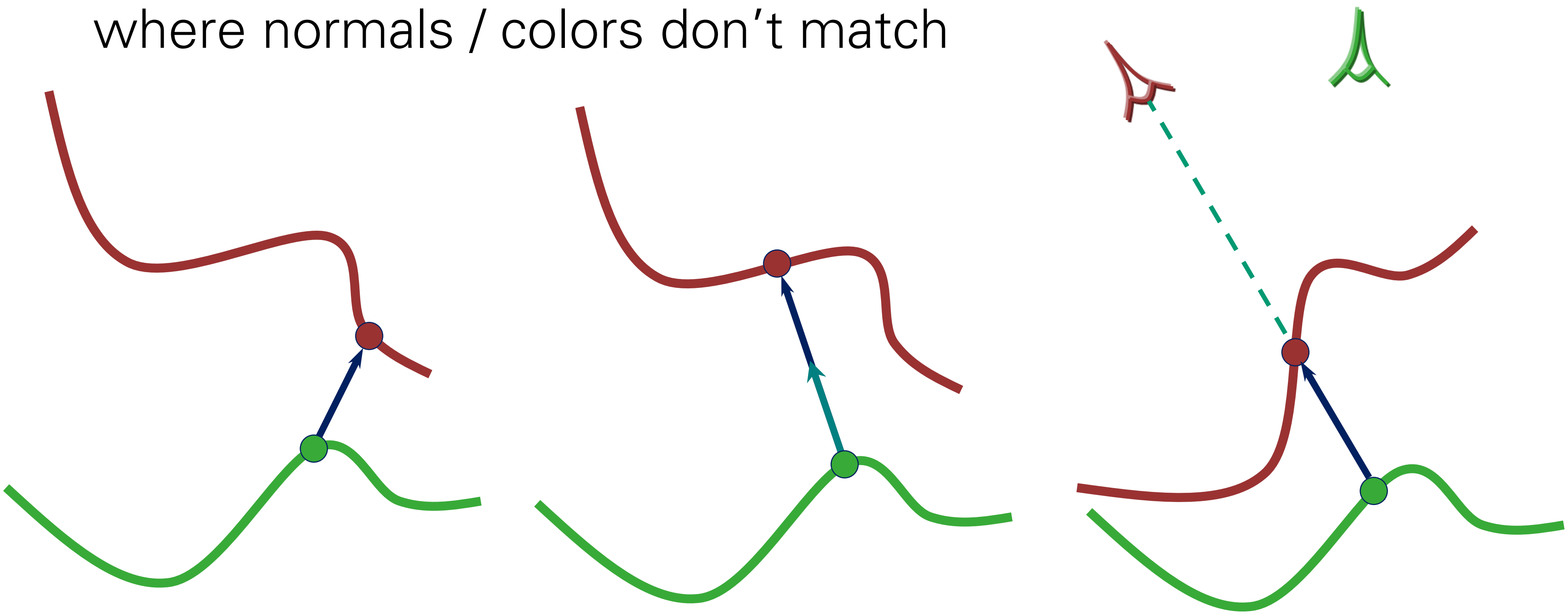


Normal-Space Sampling

ICP Variants – Matching

Computer Graphics and Visualization

- great effect on convergence and speed
- all approaches can discard matches where normals / colors don't match



closest-point matching generally stable, but slow and requires preprocessing

normal shooting slightly better than closest point for smooth meshes, worse for noisy or complex meshes

projection much faster than closest point (can be implemented through rendering), a bit less stable such that point to plane distance necessary

ICP Variants – Error Metric

- Point to point distance

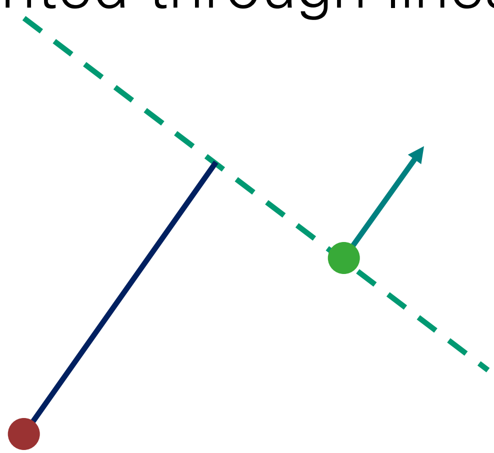
residual vector

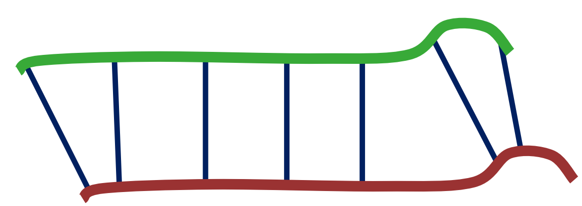
$$\left(\boldsymbol{R}^*, \vec{\boldsymbol{t}}^*\right) = \underset{\boldsymbol{R},\vec{\boldsymbol{t}}|\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{1}}{\text{minarg}} \sum_{i=1}^{n} \left\| \overbrace{\left(\boldsymbol{R}\boldsymbol{q}_i + \vec{\boldsymbol{t}}\right) - \boldsymbol{p}_i}^{\vec{\boldsymbol{r}}_i(\boldsymbol{R},\vec{\boldsymbol{t}})} \right\|^2$$

- Given normals $\widehat{\boldsymbol{n}}_i$ at points $\boldsymbol{p}_i$ one can minimize point to plane distance yielding faster convergence

$$\left(\boldsymbol{R}^*, \vec{\boldsymbol{t}}^*\right) = \underset{\boldsymbol{R},\vec{\boldsymbol{t}}|\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{1}}{\text{minarg}} \sum_{i=1}^{n} \langle \widehat{\boldsymbol{n}}_i, \vec{\boldsymbol{r}}_i \rangle^2$$

Implemented through linearization of $\boldsymbol{R}$

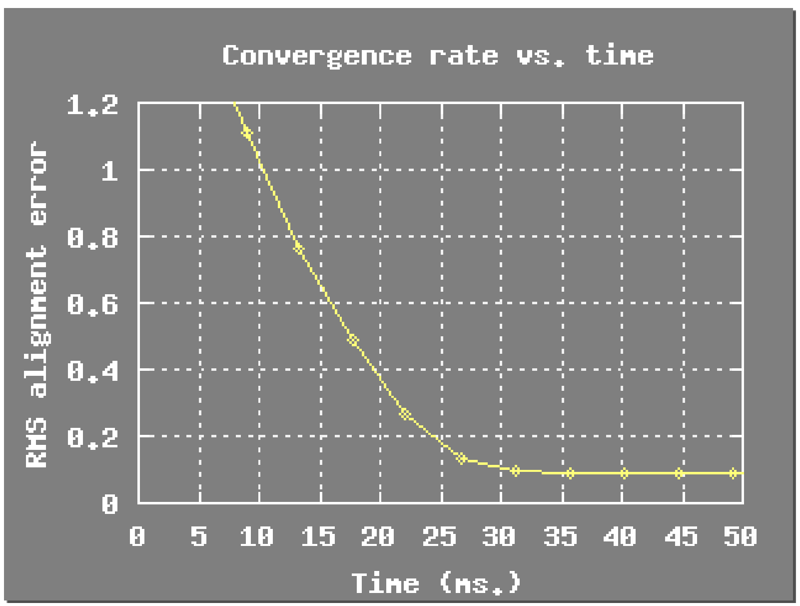




High-Speed ICP Algorithm

[Rusinkiewicz01]

- ICP algorithm with projection-based correspondences, point-to-plane matching can align meshes in a few tens of ms. (cf. over 1 sec. with closest-point)



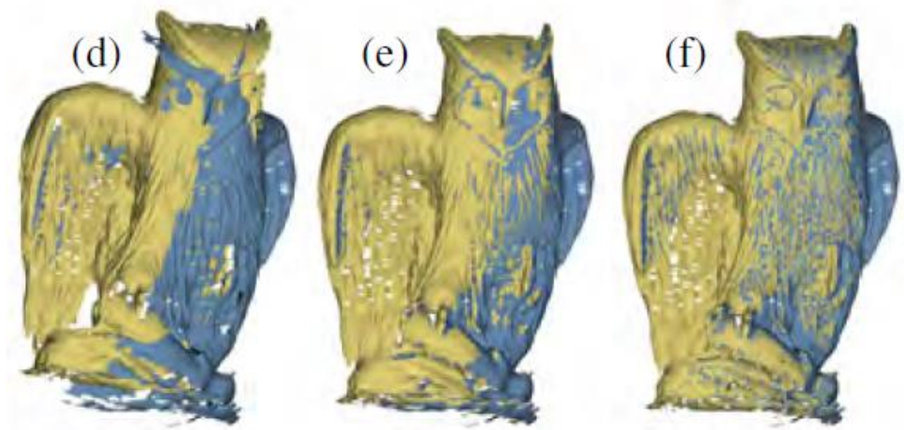
Sparse ICP [Bouaziz13/14]

Computer Graphics and Visualization

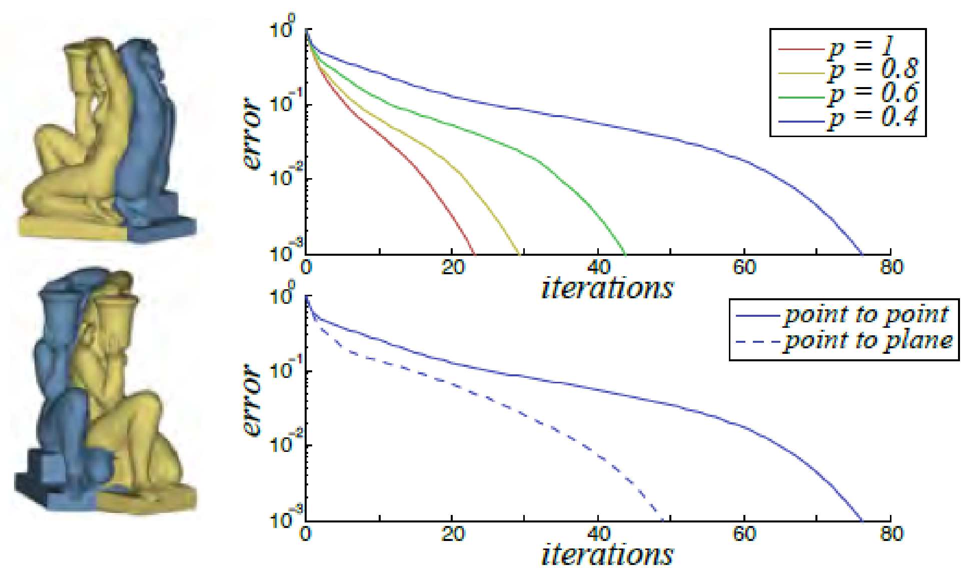
- Robust norm makes vector of residuals sparse and approach less sensitive to outliers and noise
- propose to use Minkowski norm with exponent $\nu = 0.4$ (in their paper denoted p).
- iteratively re-weighted least squares approach is too unstable for small residuals
- They derive an *Alternating Direction Method of Multipliers (ADMM)* using Lagrange multipliers $\vec{\boldsymbol{\lambda}}_i$ and an additional vector $\vec{\boldsymbol{z}}_i$ per correspondence
  - Step 1: $\underset{\boldsymbol{z}_i}{\text{minarg}} \sum_i \|\vec{\boldsymbol{z}}_i\|_2^\nu + \frac{\mu}{2}\left\|\vec{\boldsymbol{r}}_i - \vec{\boldsymbol{z}}_i + \vec{\boldsymbol{\lambda}}_i/\mu\right\|_2^2$
  - Step 2: $\underset{\boldsymbol{R},\vec{\boldsymbol{t}}|\boldsymbol{R}\boldsymbol{R}^T=\mathbf{1}}{\text{minarg}} \sum_i \left\|\vec{\boldsymbol{r}}_i - \vec{\boldsymbol{z}}_i + \vec{\boldsymbol{\lambda}}_i/\mu\right\|_2^2$
  - Step 3: $\vec{\boldsymbol{\lambda}}_i \leftarrow \vec{\boldsymbol{\lambda}}_i + \mu(\vec{\boldsymbol{r}}_i - \vec{\boldsymbol{z}}_i)$
- μ is a penalty to automatically eliminate outliers

Sparse ICP [Bouaziz13/14]

- when decreasing the norm exponent ν, the method
  - becomes more robust, but
  - slows down
- using point to plane distance increases convergence speed significantly
- purely header based source code available at http://lgg.epfl.ch/sparseicp (comes with a compatible version of Eigen)



$p = 2,\ \delta_{th} = 20\%$, $\varepsilon = 7.5e^{-2}$ | $p = 1$, $\varepsilon = 1.6e^{-2}$ | $p = 0.4$, $\varepsilon = 4.8e^{-4}$



Sparse ICP [Bouaziz13/14]

Sparse ICP

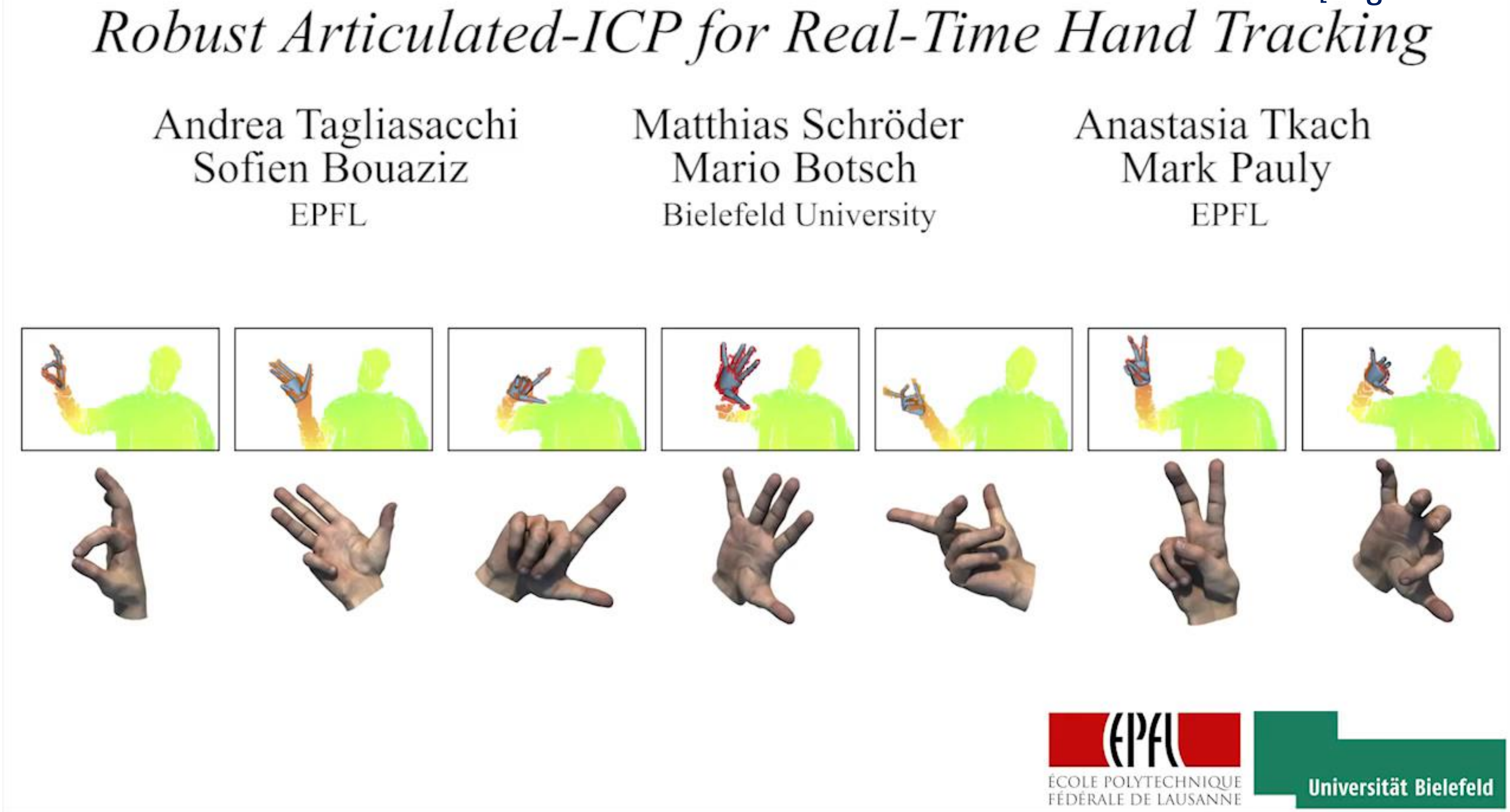
Sofien Bouaziz   Andrea Tagliasacchi   Mark Pauly

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Articulated-ICP for Hand Tracking

- current research generalizes ICP to flexible and articulated models

[Tagliasacchi15]



Literature

- [Chen92] … Y. Chen and G. Medioni. *Object modelling by registration of multiple range images*. Image Vision Computing. doi
- [Rusinkiewicz01] … S. Rusinkiewicz, M. Levoy, *Efficient Variants of the ICP Algorithm*, Third International Conference on 3D Digital Imaging and Modeling'01, doi
- [Bouaziz13] … S. Bouaziz, A. Tagliasacchi, M. Pauly. *Sparse iterative closest point*. Computer graphics forum Vol. **32**. No. 5. Blackwell Publishing Ltd, 2013.
- [Bouaziz14] … S. Bouaziz, A. Tagliasacchi, M. Pauly, Dynamic 2D/3D Registration, Eurographics'14 Tutorial (www)
- [Tagliasacchi15] … A. Tagliasacchi, M. Schroder, A. Tkach, S. Bouaziz, M. Botsch, M. Pauly. *Robust Articulated-ICP for Real-Time Hand Tracking*. SGP'15. (www)

SURFACE RECONSTRUCTION

priors

- surface smoothness
- visibility
- volumetric smoothness
- geometric primitives
- data-driven
- interactive user input

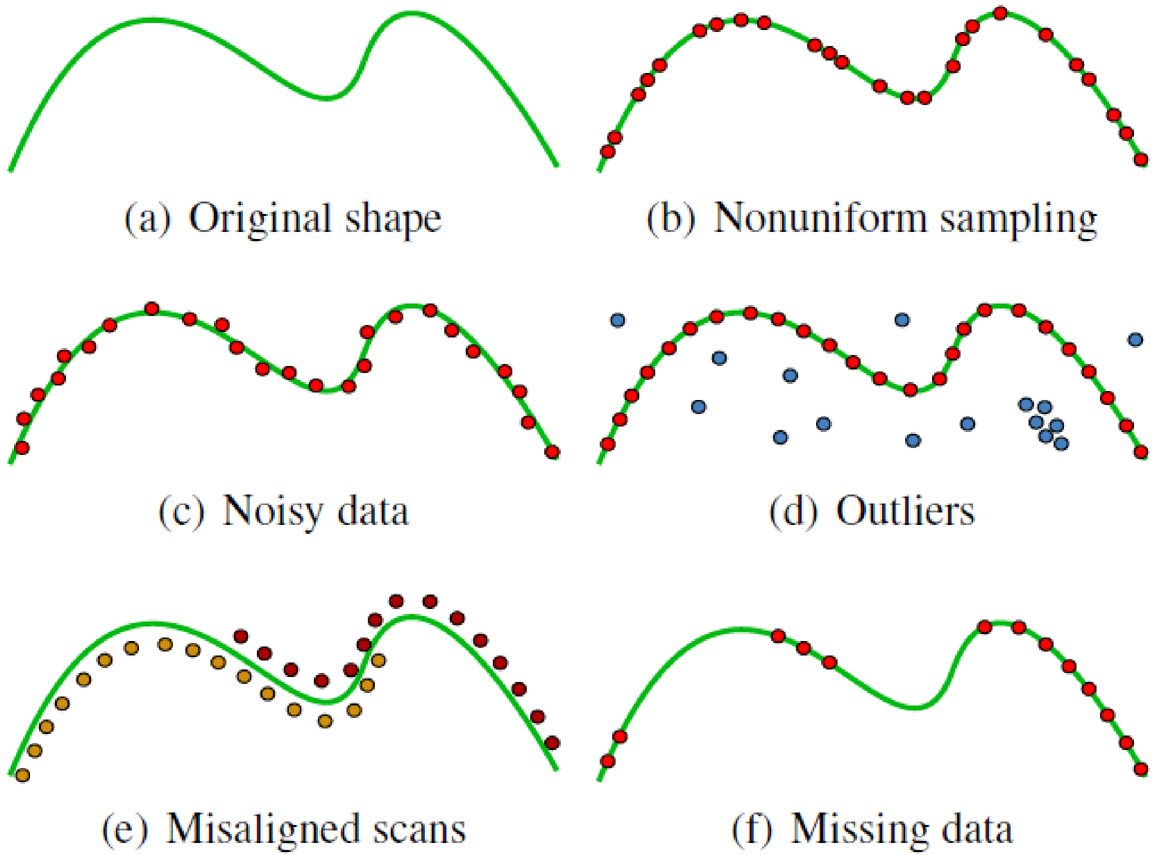


Figure 2: *Different forms of point cloud artifacts, shown here in the case of a curve in 2D.*

Overview of surface reconstruction

Input

points		
	tangent spaces	
		oriented normals

approaches

• zippering • ball pivoting • Voronoi diagram based direct tessellation • dilation based methods • moving least squares projections	• local Voronoi diagram methods • moving least squares projections	• implicit function fitting • indicator function fitting

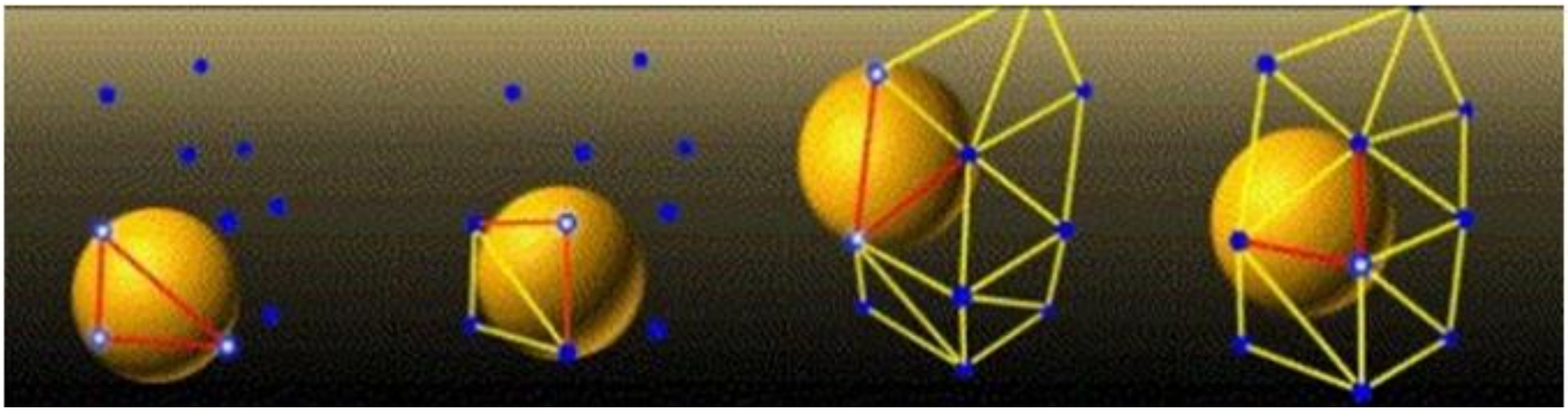
Output triangle mesh, implicit surface, point set, volumetric segmentation

Ball pivoting

[Bernardini99]

- start with seed triangle
- grow region over boundary edges that are organized in queue by rolling ball of user specified radius over edge until it hits a third point

➔ all balls touch three points and do not contain further point



- Simple and fast implementation with low memory demands and suitable for out-of-core

Ball pivoting

- But not adaptive as we need to fix a sphere radius!

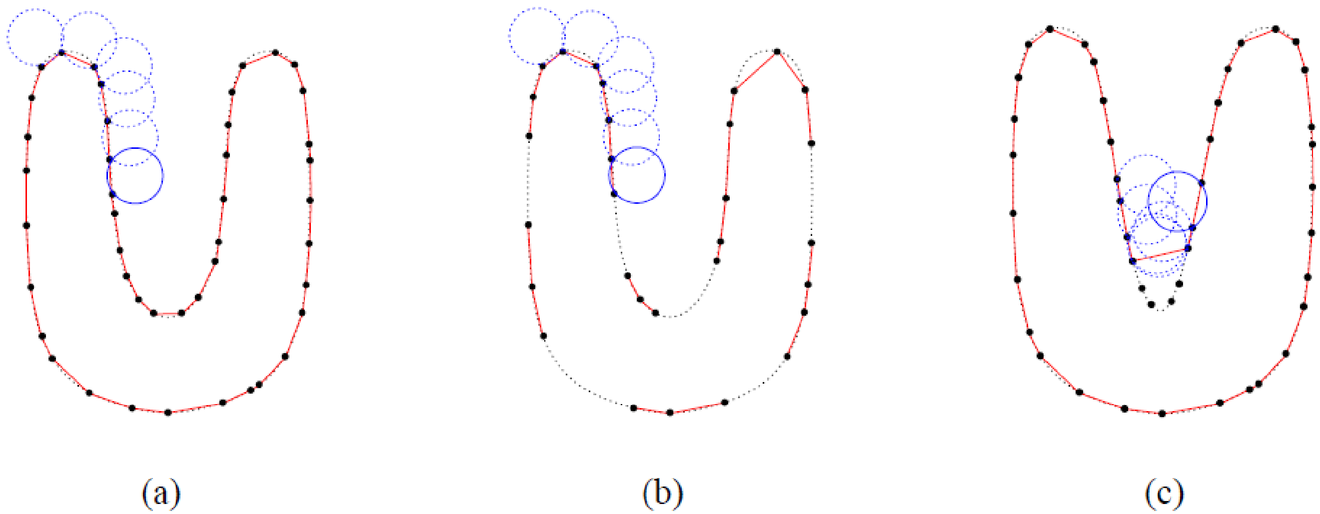


Fig. 3. The Ball Pivoting Algorithm in 2D. (a) A circle of radius ρ pivots from sample point to sample point, connecting them with edges. (b) When the sampling density is too low, some of the edges will not be created, leaving holes. (c) When the curvature of the manifold is larger than $1/\rho$, some of the sample points will not be reached by the pivoting ball, and features will be missed.

- Can not handle very noisy data sets

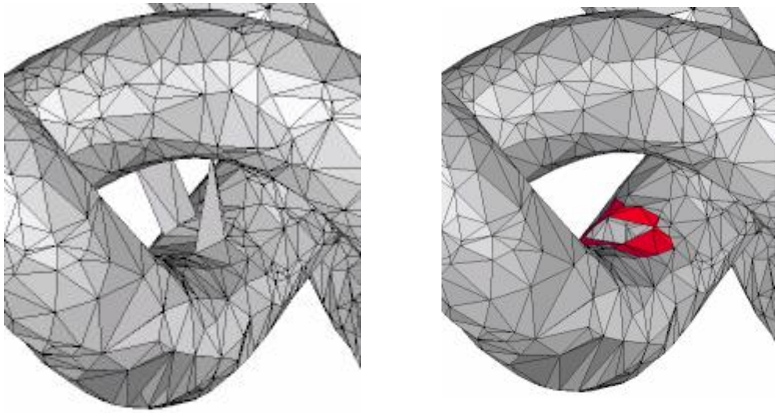
Voronoi based approaches

Crust

- compute Delaunay tetrahedralization
- for each sample compute two poles (most distant Voronoi vertices)
- add poles to point set and compute joint Delaunay tetrahedralization
- output all triangles that connect original samples

Variants

- power crust
- cocone
- umbrella filter

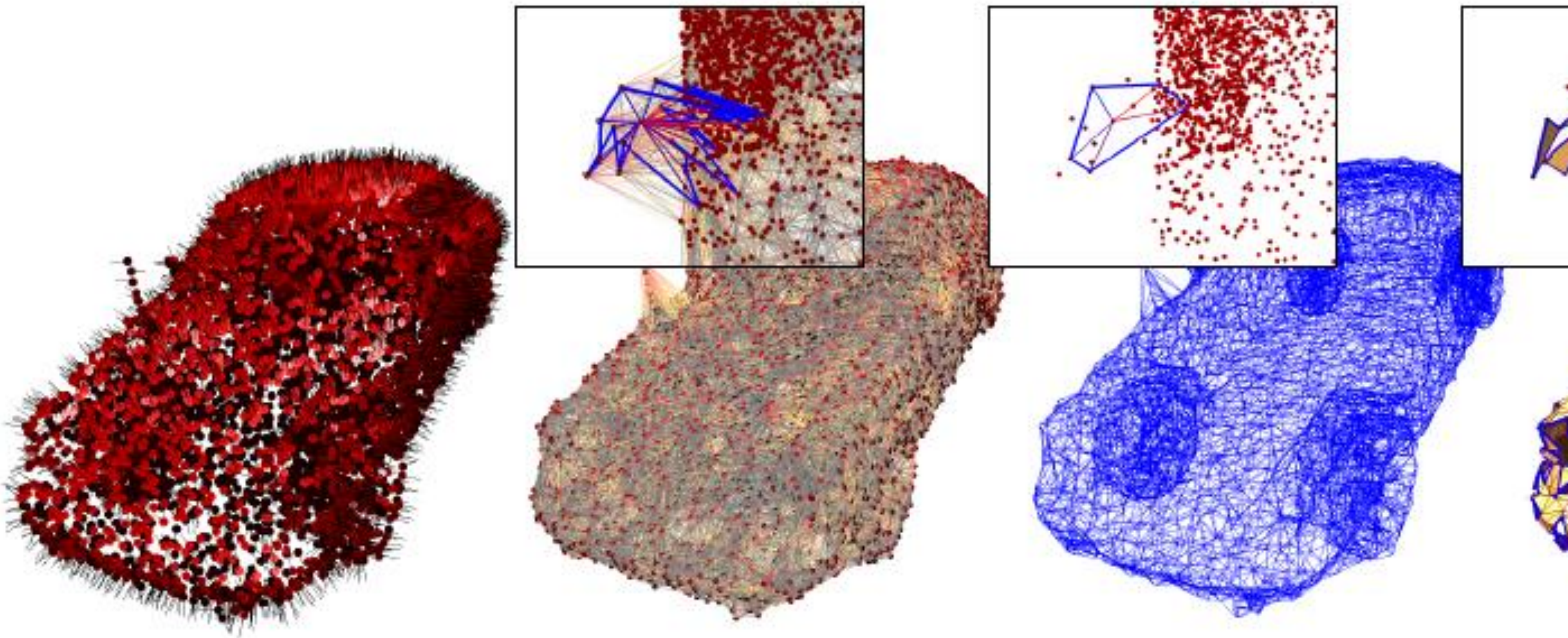


Umbrella filter for topological consistency

[König13]

Local Voronoi based triangulation

- Input: points with tangent spaces (unoriented normals)
- project point neighborhood into 2D tangent space
- compute local 2D Delaunay triangulation

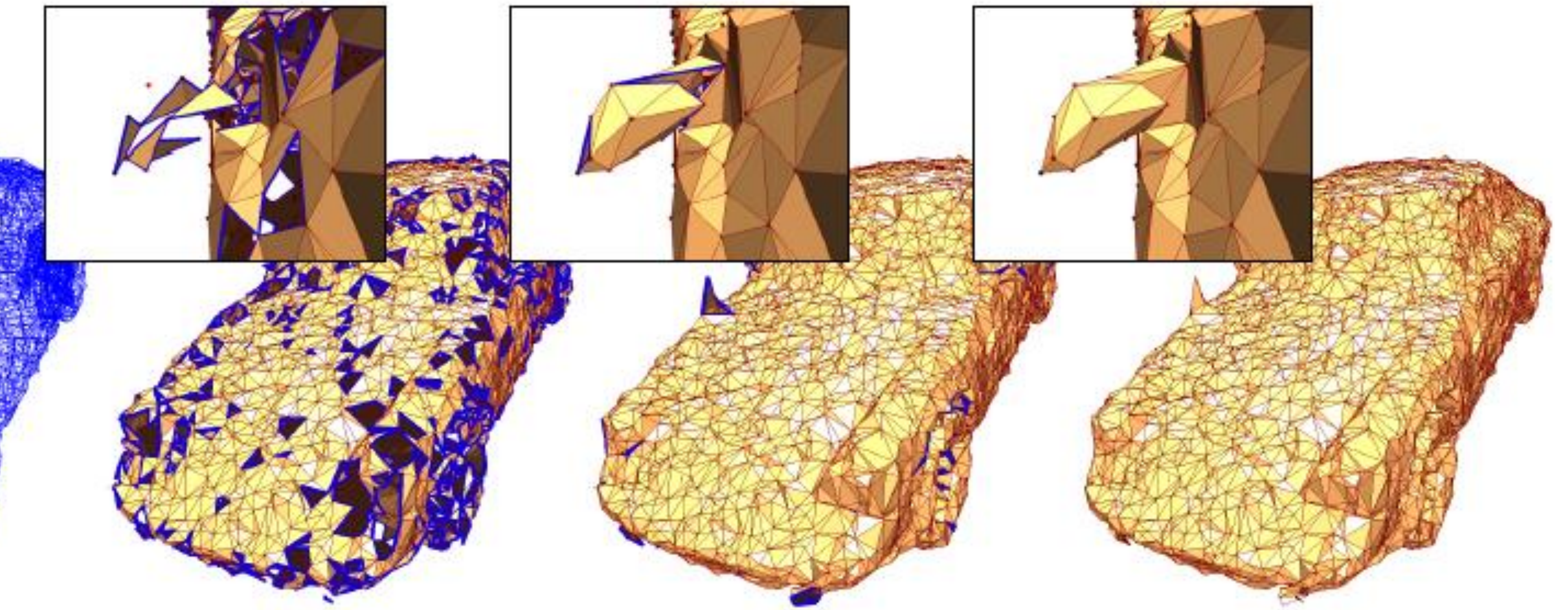


a) input point cloud b) knn-graph: k=30 c) tangentially Delaunay filtered

Local Voronoi based triangulation

[König13]

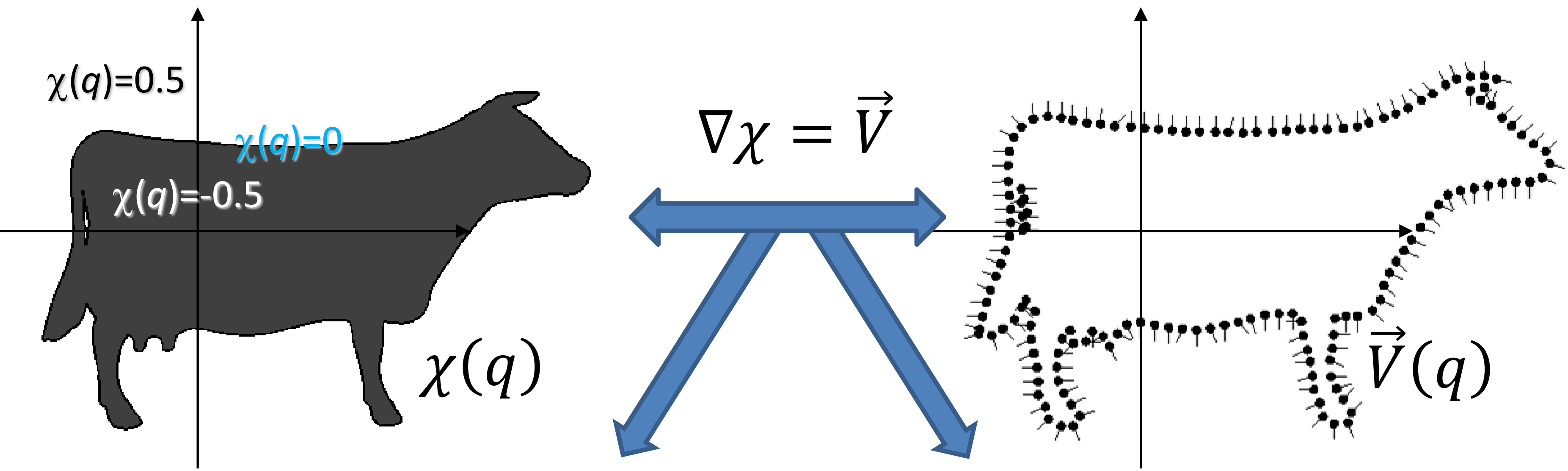
- find triangles that are part of all local Delaunay triangulations
- use priority queue to add nearly consistent triangles
- close remaining small holes with tessellation strategy



filtered d) consistent triangles e) region grown f) holes closed

Indicator Function Based Reconstruction

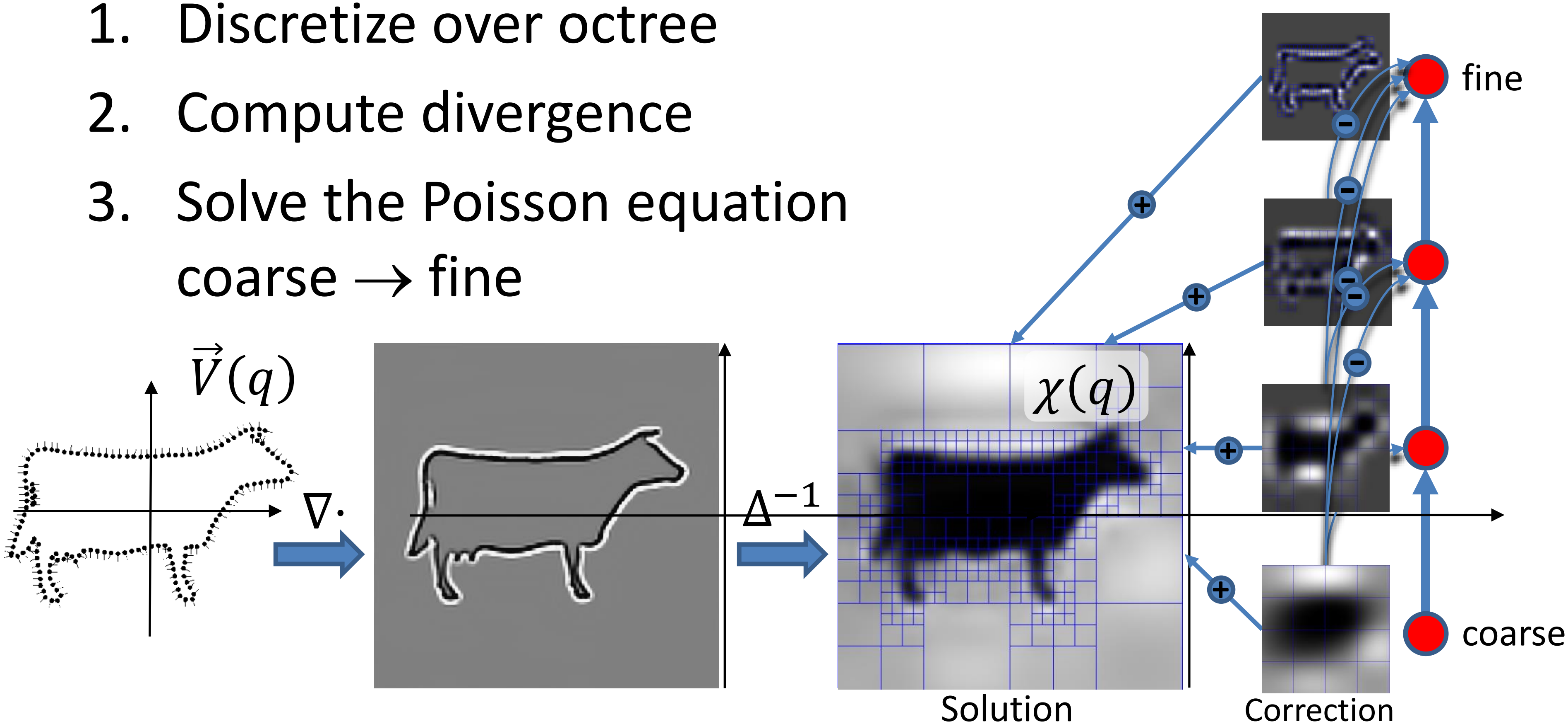
- M. Kazhdan, M. Bolitho, H. Hoppe, *Poisson surface reconstruction*, Symposium on Geometry Processing 2006, 61-70.
- M. Kazhdan, H. Hoppe. Screened Poisson surface reconstruction, ACM Trans. Graphics, 32(3), 2013.



$$\Delta\chi = \nabla\vec{V} \qquad \chi = \underset{F}{\mathrm{minarg}}\left\|\nabla F - \vec{V}\right\|^2$$

Indicator Function Based Reconstruction

1. Discretize over octree
2. Compute divergence
3. Solve the Poisson equation coarse $\rightarrow$ fine

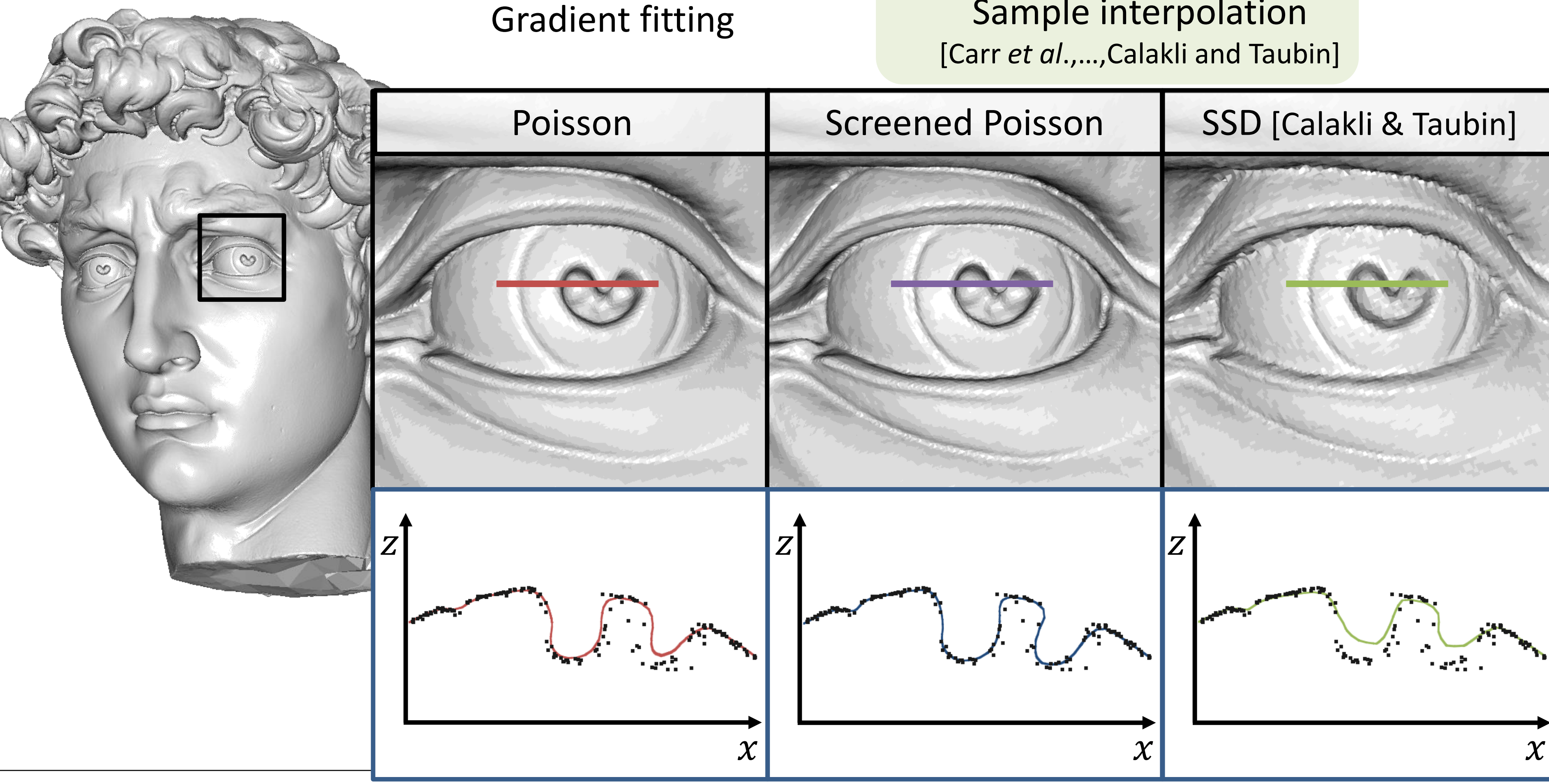


Indicator Function Based Reconstruction

Computer Graphics and Visualization

$$E(\chi) = \underbrace{\int \left\| \nabla\chi(q) - \vec{V}(q) \right\|^2 dq}_{\text{Gradient fitting}} + \underbrace{\lambda \sum_{p \in P} \| \chi(p) - 0 \|^2}_{\text{Sample interpolation}}$$

[Carr *et al*.,…,Calakli and Taubin]



Indicator Function Basedeconstruction

- Screended Poisson surface reconstruction toolchain available at https://github.com/mkazhdan/PoissonRecon

color values from the input samples can be obtained by calling:

```
% PoissonRecon --in eagle.points.ply --out eagle.pr.color.ply --depth 10 --colors
```

using the --colors flag to indicate that color extrapolation should be used.

A reconstruction of the eagle that does not close up the holes can be obtained by first calling:

```
% SSDRecon --in eagle.points.ply --out eagle.ssd.color.ply --depth 10 --colors --density
```

using the --density flag to indicate that density estimates should be output with the vertices of the mesh, and then calling:

```
% SurfaceTrimmer --in eagle.ssd.color.ply --out eagle.ssd.color.trimmed.ply --trim 7
```

to remove all subsets of the surface where the sampling density corresponds to a depth smaller than 7.

This reconstruction can be chunked into cubes of size 4×4×4 by calling:

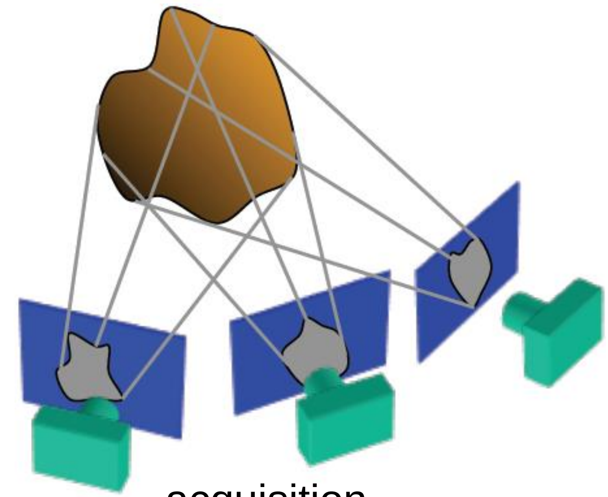
```
% ChunkPly --in eagle.ssd.color.trimmed.ply --out eagle.ssd.color.trimmed.chnks --width 4
```

which partitions the reconstruction into 11 pieces.

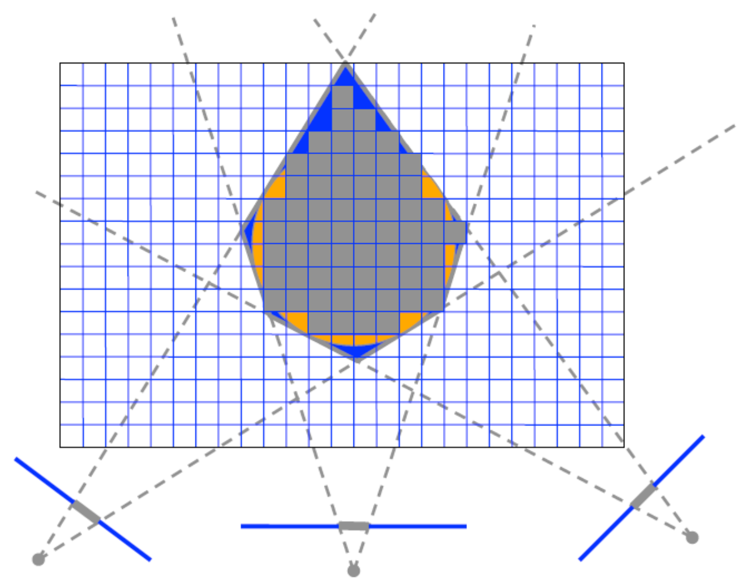
PHOTOGRAMMETRIC SURFACE RECONSTRUCTION

Silhouette Carving

- Acquire object from a large number of registered views (for example with turn table)
- Extract binary Object masks (border corresponds to silhouette of object)
- From silhouette the visual hull can be reconstruct as follows:
  - Define volume grid
  - Project each voxel to all mask images and check whether it falls inside all silhouettes
- The resulting set of voxels (grey in figure) approximate the visual hull of the object



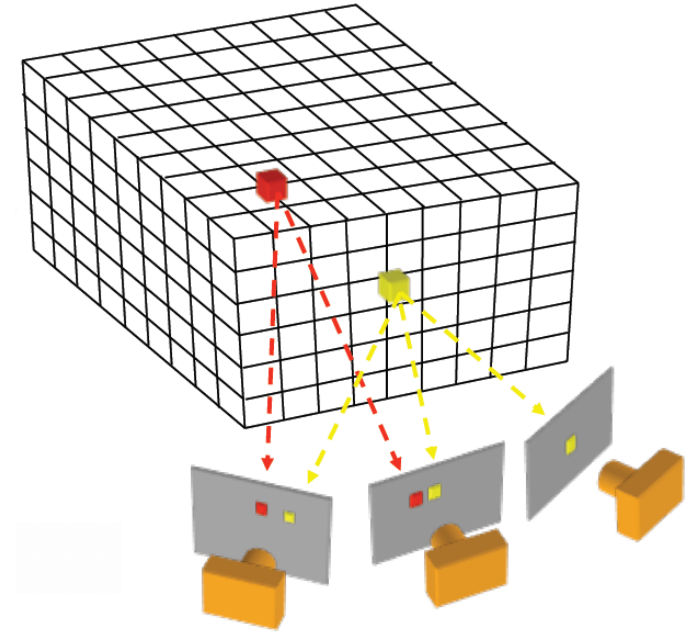
acquisition



reconstructed voxels

Space Carving

- idea: exploit the colors of the image pixels and remove further voxels that are not photoconsistent as follows:
  - define volume
  - optionally perform silhouette carving as initialization
  - check for each voxel on the surface, whether it is photo-consistent: project voxel into all images, compute variance and threshold variance
  - discard inconsistent voxels until surface is consistent



$$\Phi(v) = f\left(\frac{1}{K}\sum_{j=1}^{K}\left(c_j - c_{mean}\right)^2\right)$$

variance of average colour c_j over all K visible images [Seitz&Kutulakos]

K. N. Kutulakos and S. M. Seitz, *A Theory of Shape by Space Carving, ICCV 1999.*

Results Space Carving

Input Image (1 of 45)

Reconstruction

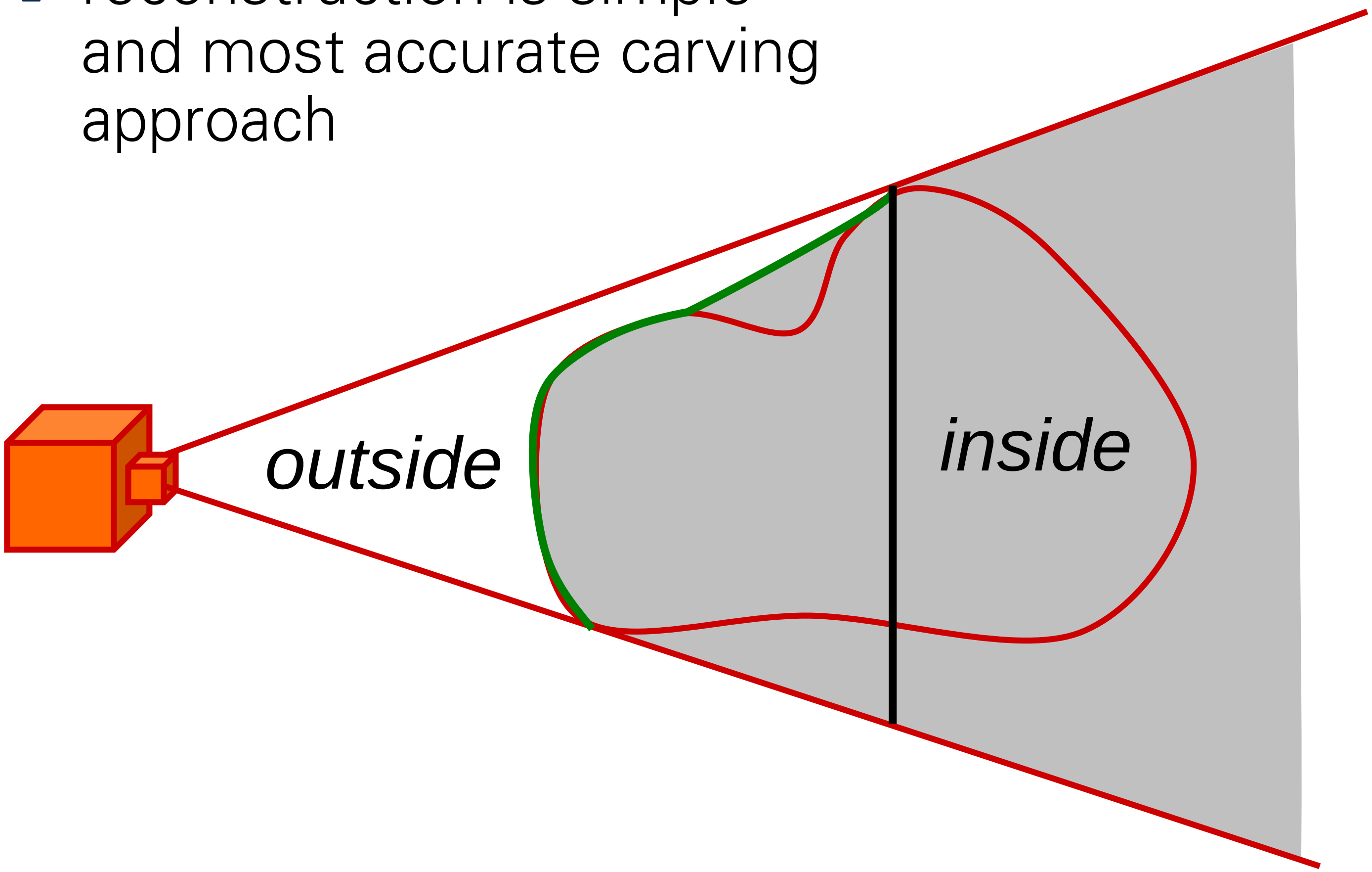
Reconstruction

Reconstruction

Source: S. Seitz

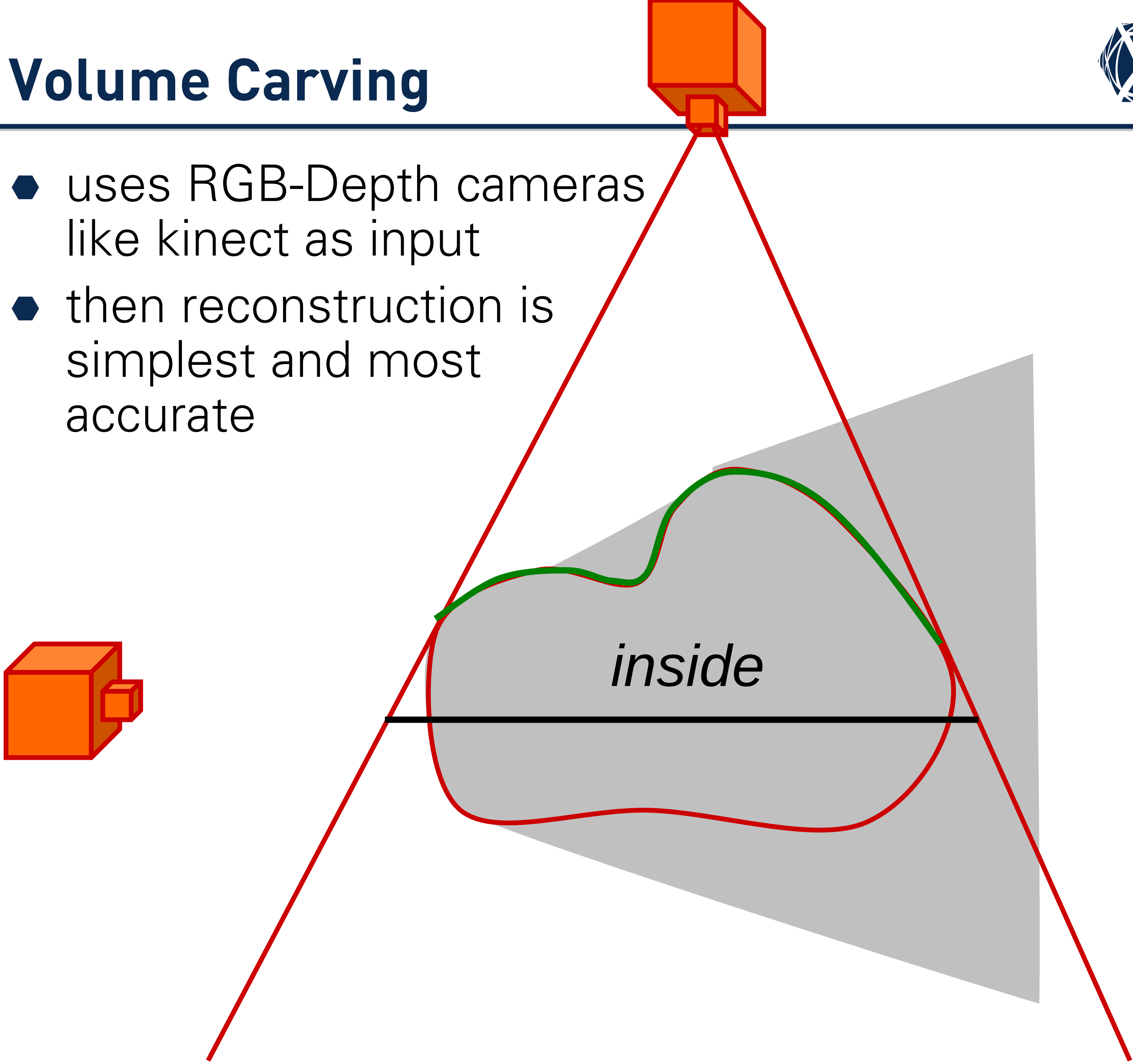
Volume Carving

- uses RGB-Depth cameras like kinect as input
- reconstruction is simple and most accurate carving approach



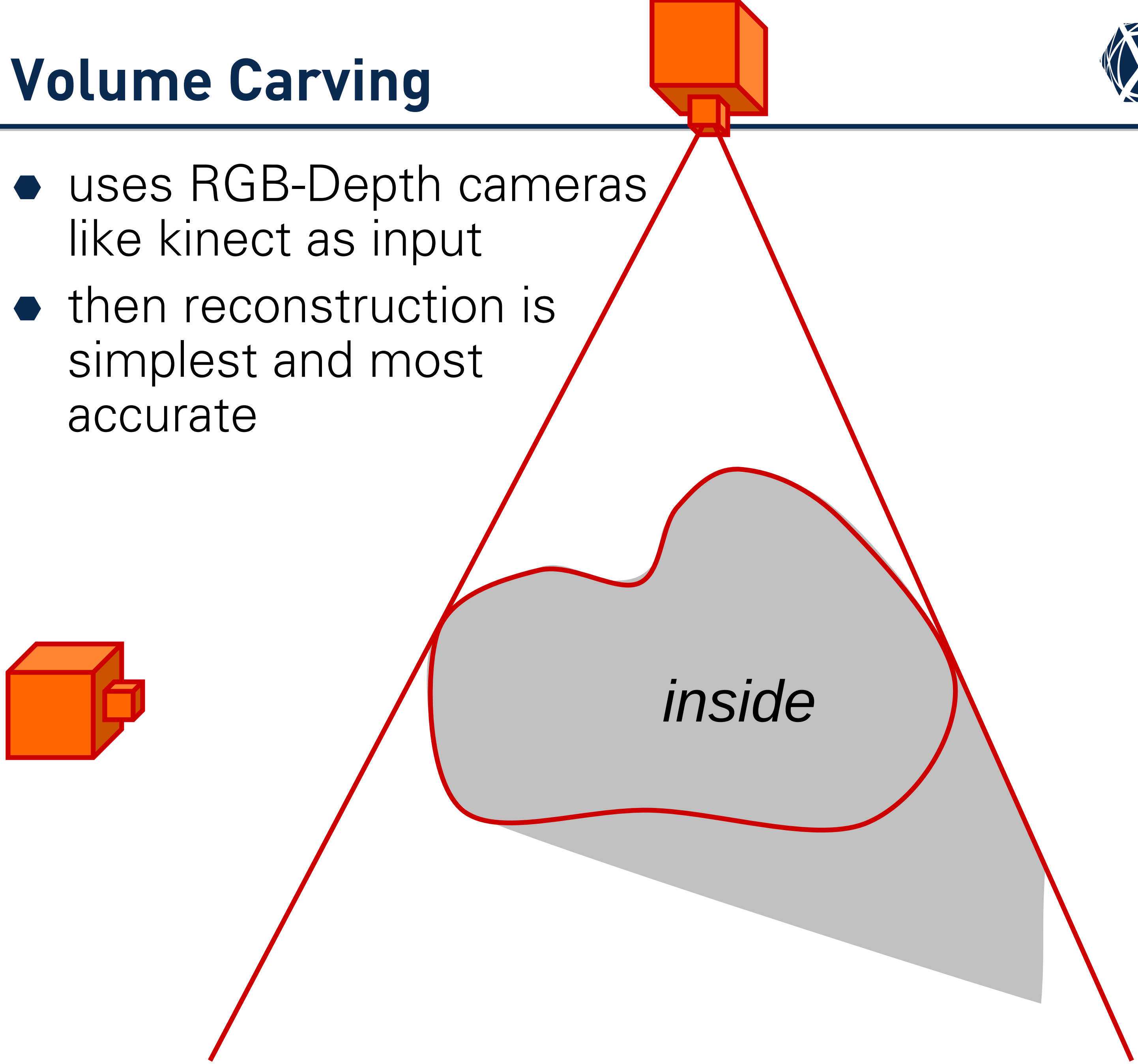
Volume Carving

- uses RGB-Depth cameras like kinect as input
- then reconstruction is simplest and most accurate



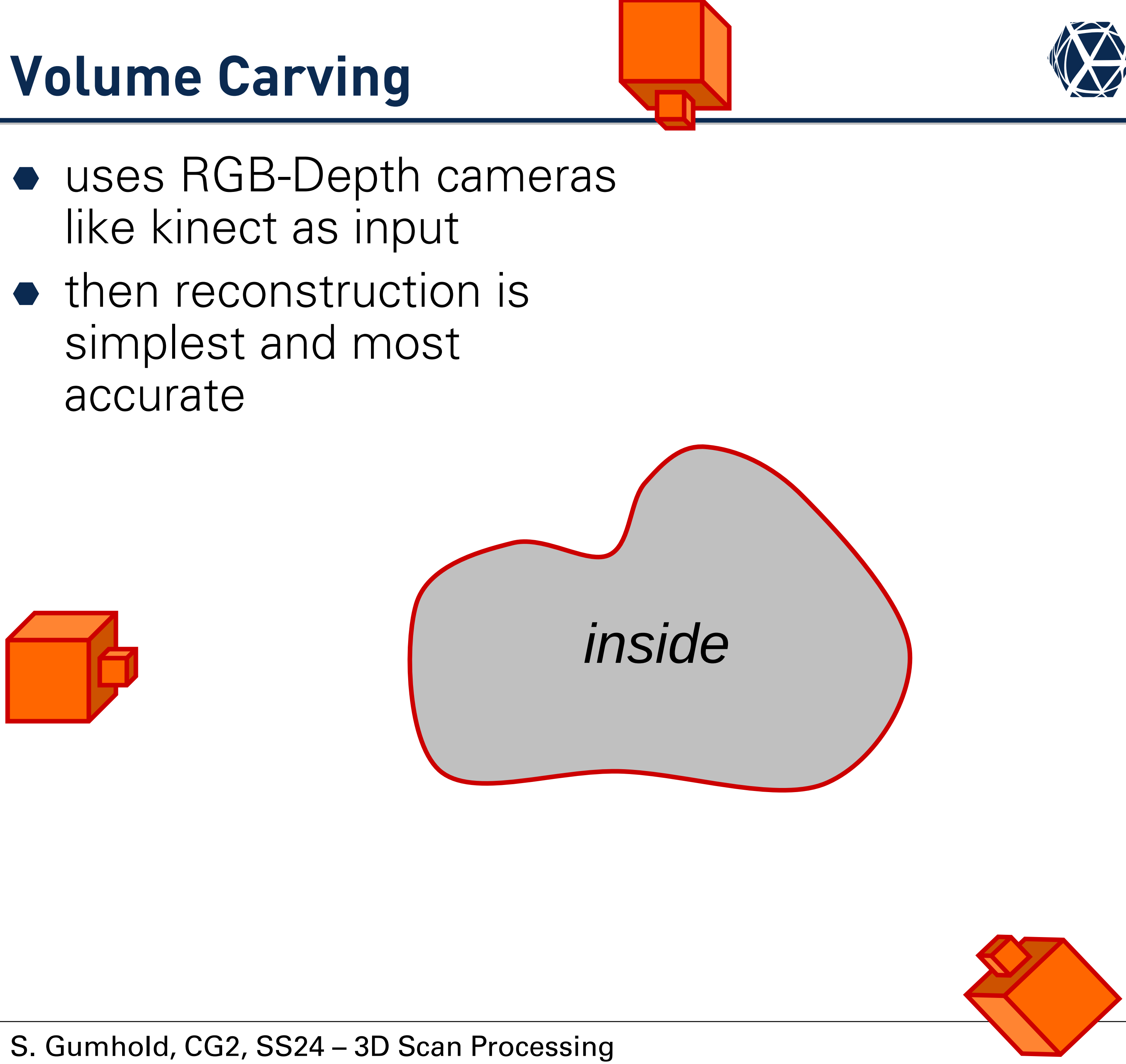
Volume Carving

- uses RGB-Depth cameras like kinect as input
- then reconstruction is simplest and most accurate

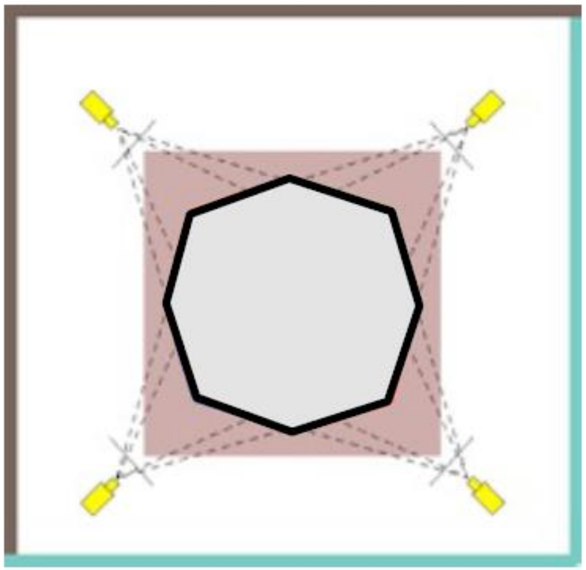


Volume Carving

- uses RGB-Depth cameras like kinect as input
- then reconstruction is simplest and most accurate



Comparison of Reconstructions



visual hull from silhouettes (outer bound of possible scenes)

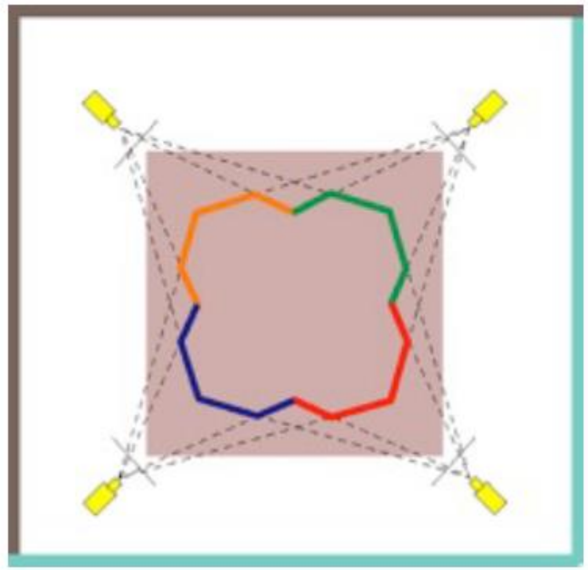
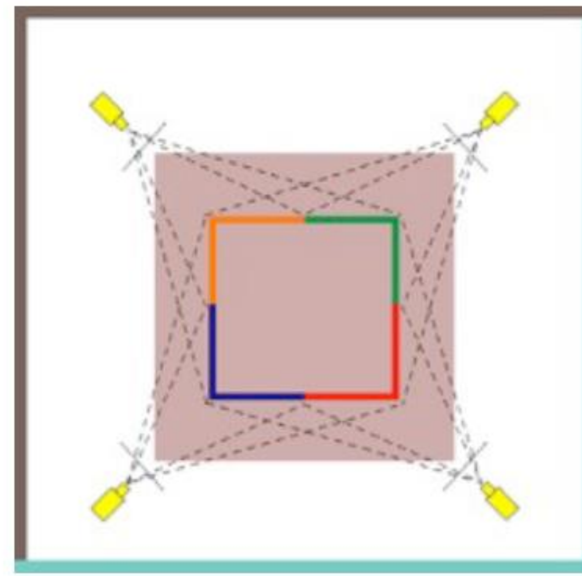


photo consistency hull (better outer bound of scenes)



real object can be reconstructed with volume carving from RGB-Depth

References

- [Bernardini99] … F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin. *The Ball-Pivoting Algorithm for Surface Reconstruction*. TVCG'99. doi
- K. N. Kutulakos and S. M. Seitz, *A Theory of Shape by Space Carving, ICCV 2000.* www
- [Kazhdan06] … M. Kazhdan, M. Bolitho, H. Hoppe, *Poisson surface reconstruction*, Symposium on Geometry Processing 2006, 61-70. www
- [König13] … S. König, S. Gumhold. *Robust Surface Reconstruction from Point Clouds*. TechRep TU Dresden. Qucosa
- [Berger14] … M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, J. A. Levine, A. Sharf, C. Silva. *State of the Art in Surface Reconstruction from Point Clouds*, Eurographics'14 STAR. www