Robot Vision

ICP and 3D Scan Registration

Dr. Chen Feng

cfeng@nyu.edu

ROB-UY 3203, Spring 2024

Overview

- Iterative Closet Point (ICP)
  - 3D data representation
  - Point cloud registration
  - ICP
- Variants of ICP
  - Point-to-plane ICP
  - Colored ICP
- Application of ICP
  - Kinect Fusion

References

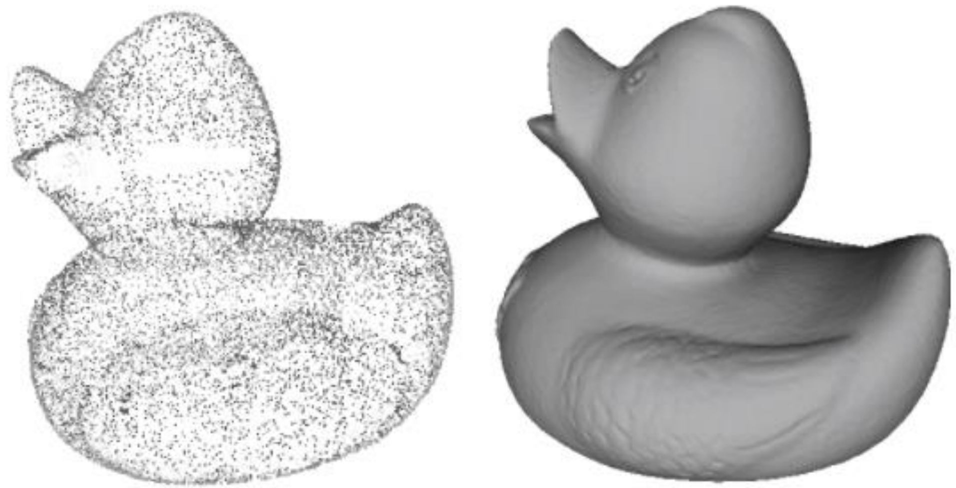
- Corke 2017:
  - Section 14.5
- Forsyth & Ponce 2011
  - Section 14.3
- Szeliski 2022:
  - Section 13.2.1

- Umeyama, S. (1991). Least-squares estimation of transformation parameters between two point patterns. IEEE Transactions on Pattern Analysis & Machine Intelligence, (4), 376-380.
- Low, K.L., 2004. Linear least-squares optimization for point-to-plane icp surface registration. *University of North Carolina Chapel Hill*, *4*(10).
- Park, J., Zhou, Q. Y., & Koltun, V., (2017). Colored point cloud registration revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 143-152).

3D Data Representations

Point Cloud/Mesh

- ✓ Raw format/Efficiency
- • Explicit representation
- × Unorganized/Unordered



https://elmoatazbill.users.greyc.fr/point_cloud/reconstruction.png

Voxel

- • Implicit representation
- × Resolution/Scalability
- × Discretization artifact

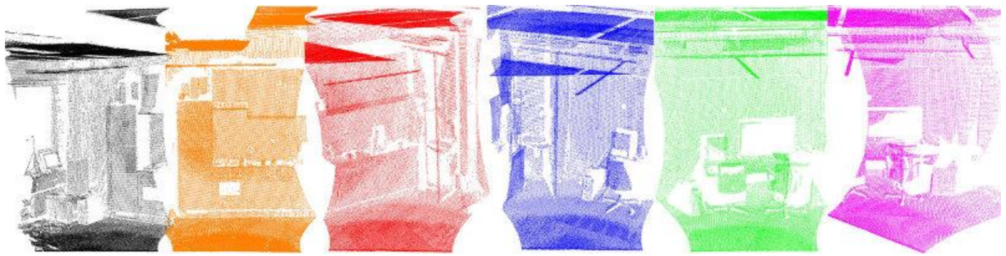
https://www.planetminecraft.com/project/giant-snowman-1638162/

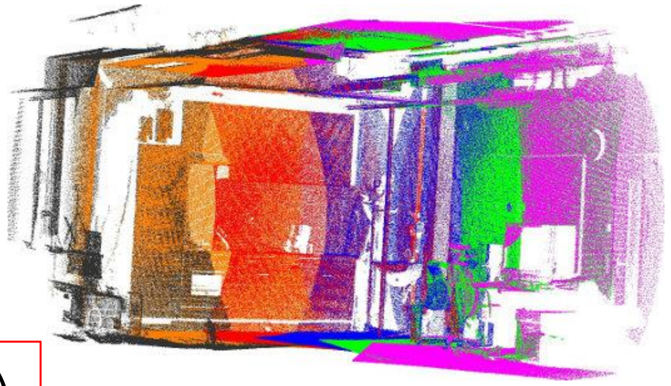
Primitives

- ✓ Compact
- ✓ Ready for interaction
- • Complex shapes?

http://pointclouds.org/gsoc/

Point Cloud Mapping / Scan Registration

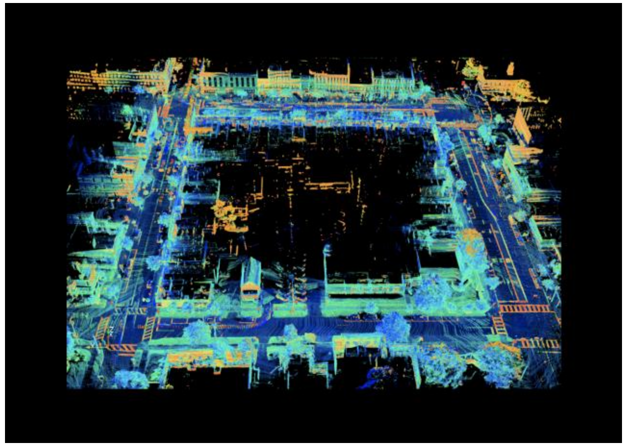




$$\underset{\{R_i, t_i\}}{\operatorname{argmin}} \sum_{i,j} \sum_{k} \rho\left(\min_{s} \left\| \begin{bmatrix} R_j & t_j \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_s^j \\ 1 \end{bmatrix} - \begin{bmatrix} R_i & t_i \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_k^i \\ 1 \end{bmatrix} \right\|^2 \right)$$

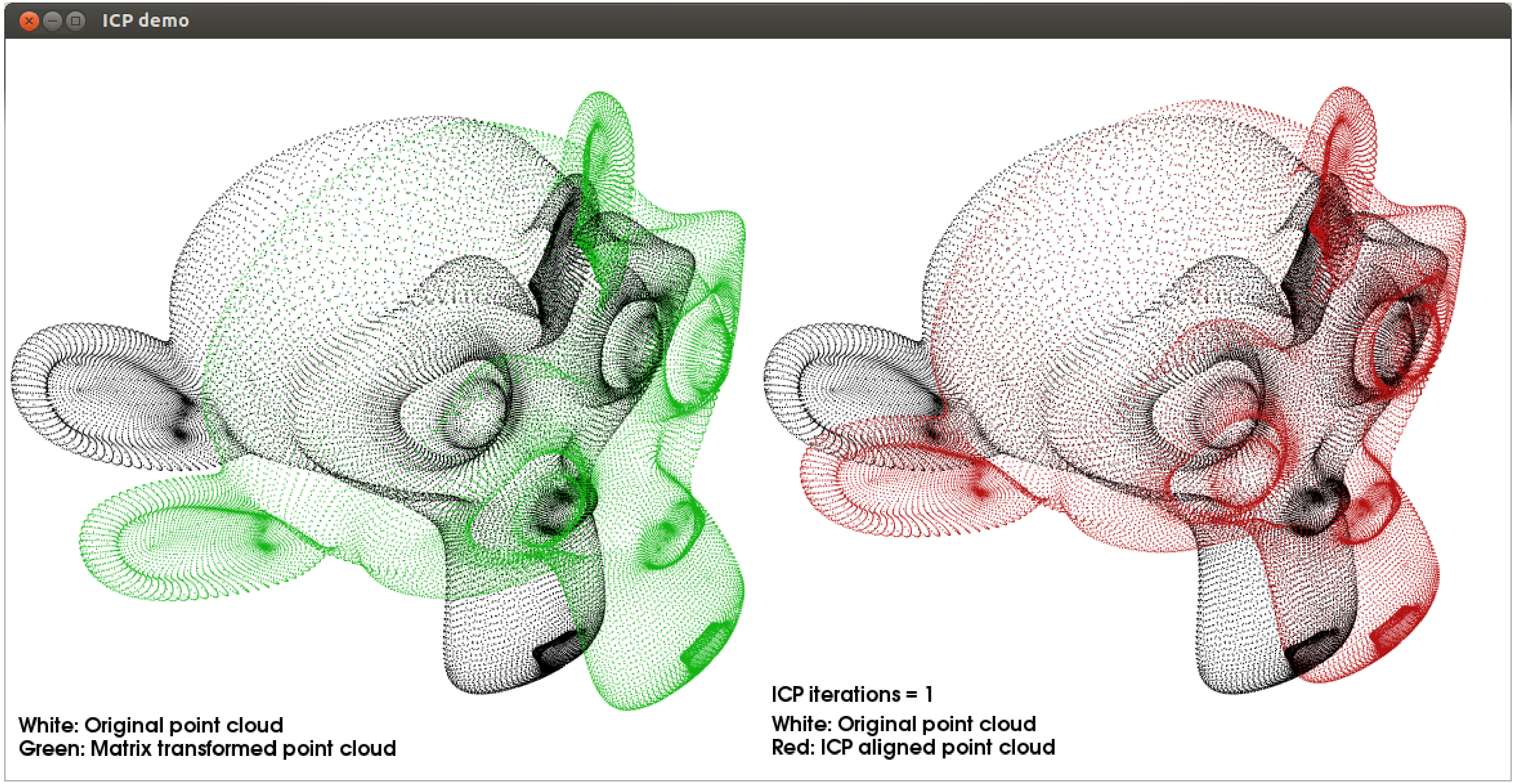
$s.t. \quad R_i \in SO(3)$

$t_i, P_k^i \in \mathbb{R}^3, \; \rho(\cdot)$ is a robust loss function (e.g., Huber loss)



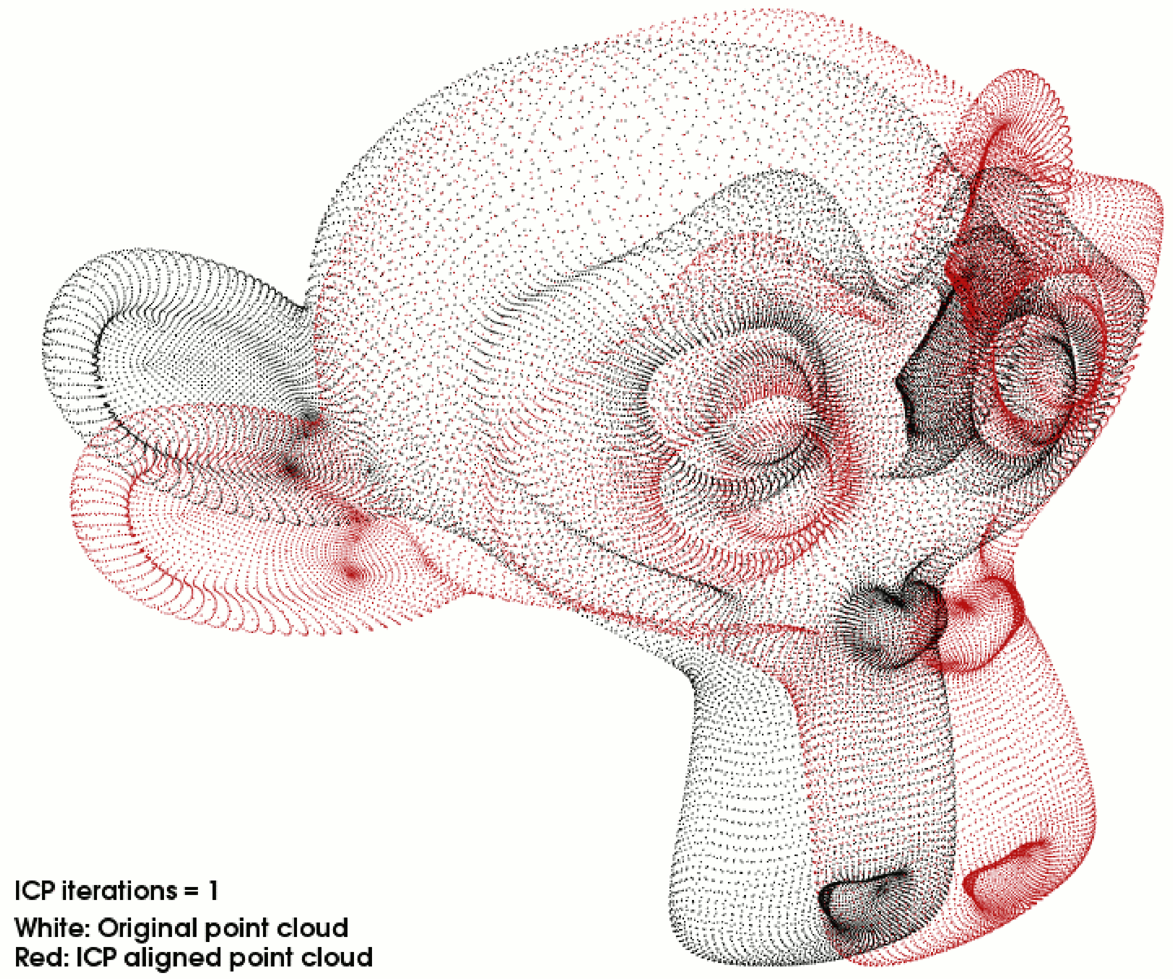
Images from http://pointclouds.org/documentation/tutorials/registration_api.php

Iterative Closest Point (ICP)



http://pointclouds.org/documentation/tutorials/interactive_icp.php#interactive-icp

ICP Process



http://pointclouds.org/documentation/tutorials/interactive_icp.php#interactive-icp

Core Problem Definition

- Given two sets of **corresponding** points
  - Source: $S = \{S_1, S_2, \cdots, S_n\}$
  - Destination $\mathrm{D} = \{D_1, D_2, \cdots, D_n\}$
- Find the translation and rotation that optimize the cost function
  - $C(R, t) = \sum_i \|D_i - RS_i - t\|^2$
- A fancy name of the problem: procrustes analysis
  - We've seen a simpler version of this
  - Hand-eye calibration
  - Vanishing-point-based camera orientation estimation

Solve Translation First

- Idea: assume the optimal rotation has been found
- Set partial derivative w.r.t. translation to zero, solve the optimal translation

- Result
  - $t^* = (\sum_i D_i - R \sum_i S_i)/n$

Solve Rotation

- Idea: substitute the optimal translation back into the cost function
- $C(R,t) = \sum_i \left\| D_i - RS_i - (\sum_i D_i - R\sum_i S_i)/n \right\|^2$

- Define new sets of corresponding points
  - Source: $s = \{s_i\} = \{S_i - \sum_i S_i\,/n\}$
  - Destination $\mathrm{d} = \{d_i\} = \{D_i - \sum_i D_i\,/n\}$
  - A common term of this process: **demean, i.e. move data center to origin**
- $C(R,t) = \sum_i \|d_i - Rs_i\|^2$

- Solve this by the Orthogonal Procrustes Problem
  - Polar decomposition on the 3x3 covariance matrix $\sum_i d_i s_i^T$ by SVD

- Basically: **Center** point clouds, then **rotate** to align further. Repeat as necessary.

Wait, but I Do Not Know Correspondences!

- Approximate correspondences
  - Simply pick the closest point
- Then iterate until converge

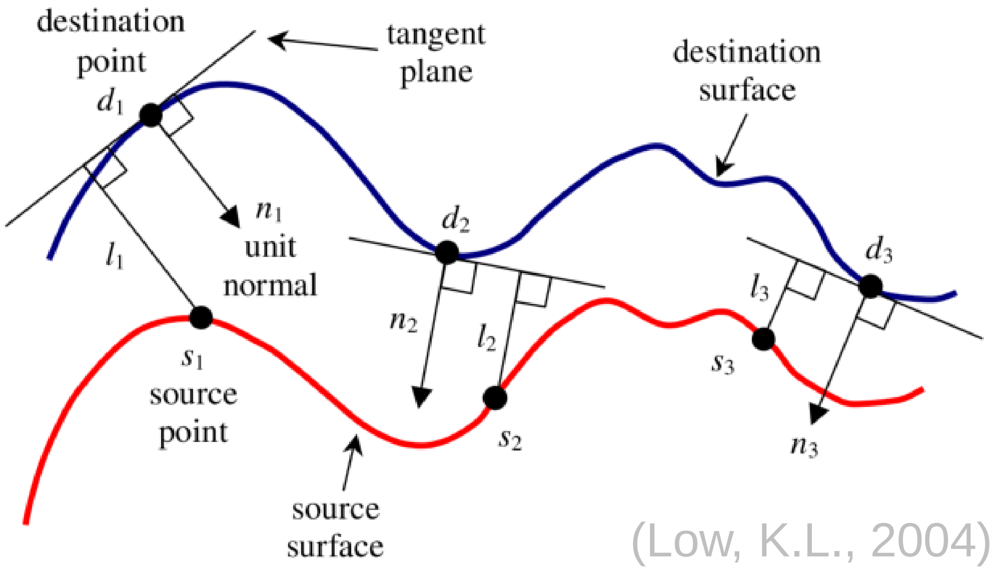
- ICP step-by-step
  1. **Find Correspondences**:
     Find correspondences of **S** in **D** by searching closest points (Brute-Force, KdTree)

  2. **Compute Center and Rotation**:
     Solve the current optimal translation and rotation by Procrustes analysis

  3. **Align via Centering and Rotating**:
     If cost **decreased**, apply the current solution back to **S**
     If cost **stopped** changing or smaller than a threshold, stop.
     Otherwise **repeat** the above steps.

ICP variants - Change Cost Function

- Point-to-Point: $C(R,t) = \sum_i \|D_i - RS_i - t\|^2$
- Point-to-Plane: $C(R,t) = \sum_i \left\|N_i^T(D_i - RS_i - t)\right\|^2$, N_i being $D_i's$ unit normal
  - Often significantly better convergence rate than point-to-point ICP
  - Linearization under small rotation assumption



(Low, K.L., 2004)

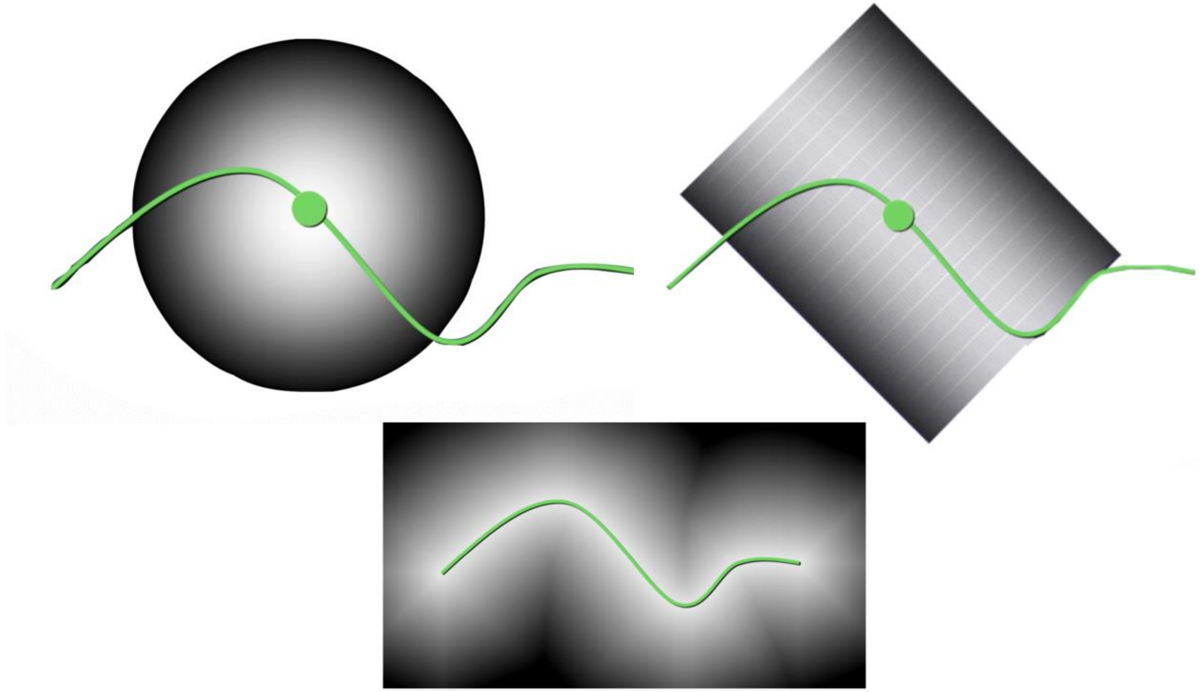
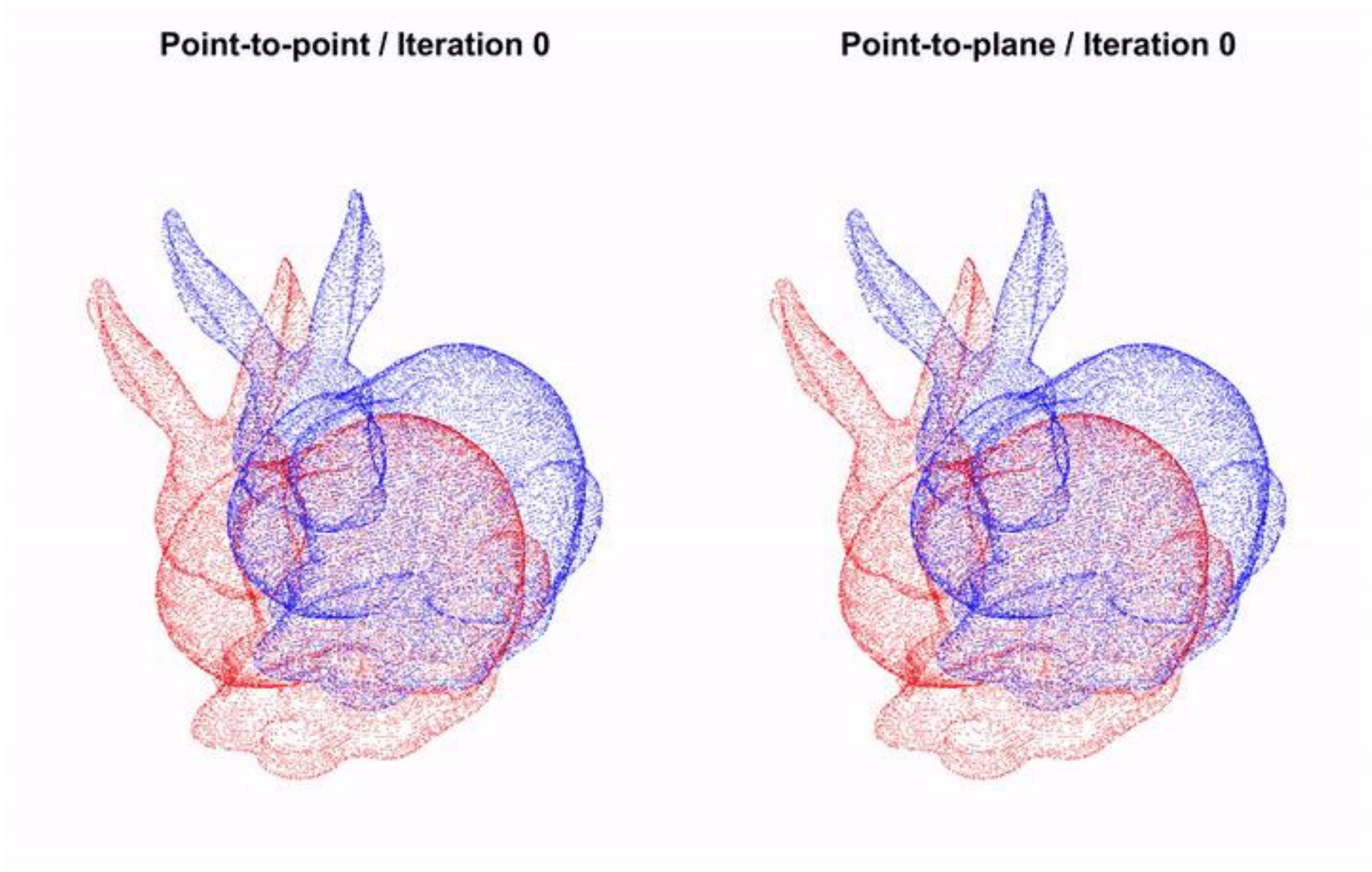


Image from Hao Li.

Yang Chen, and Gerard Medioni. *Object Modeling by Registration of Multiple Range Images*. International Journal of Image and Vision Computing, 10(3), pp. 145–155, 1992.

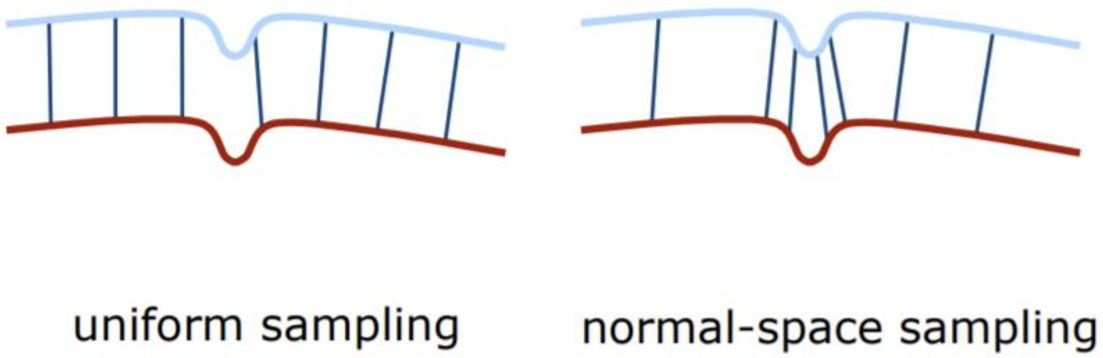
ICP variants - Point-to-Point vs Point-to-Plane ICP

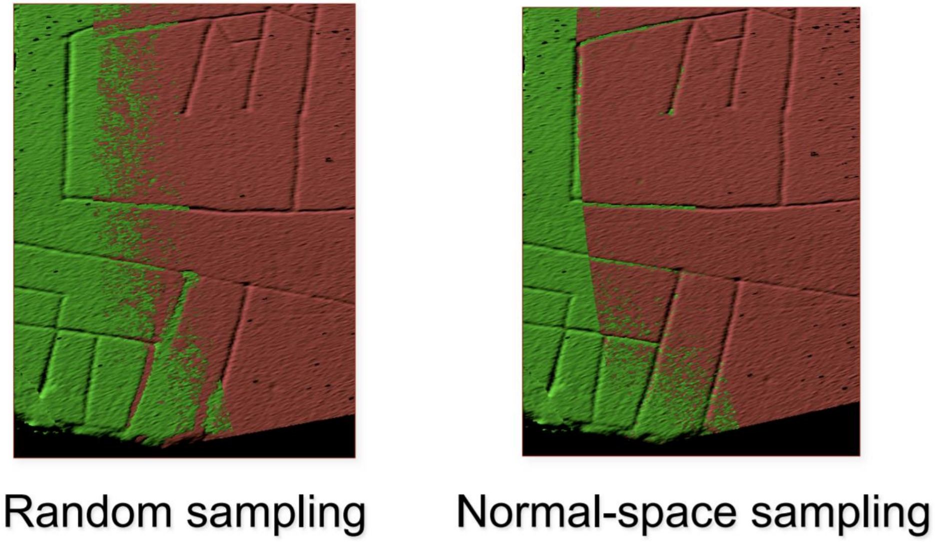


From Philip Glira - https://www.youtube.com/watch?v=LcghboLgTiA

ICP variants – Sampling Source Points

- Normal-space sampling is better for mostly smooth areas with sparse features





(Rusinkiewicz et al., 2001)

ICP variants – Sampling Source Points

- Feature-based Sampling
  - Find 3D keypoints
  - Often with higher efficiency and higher accuracy
  - May require RANSAC to remove outliers

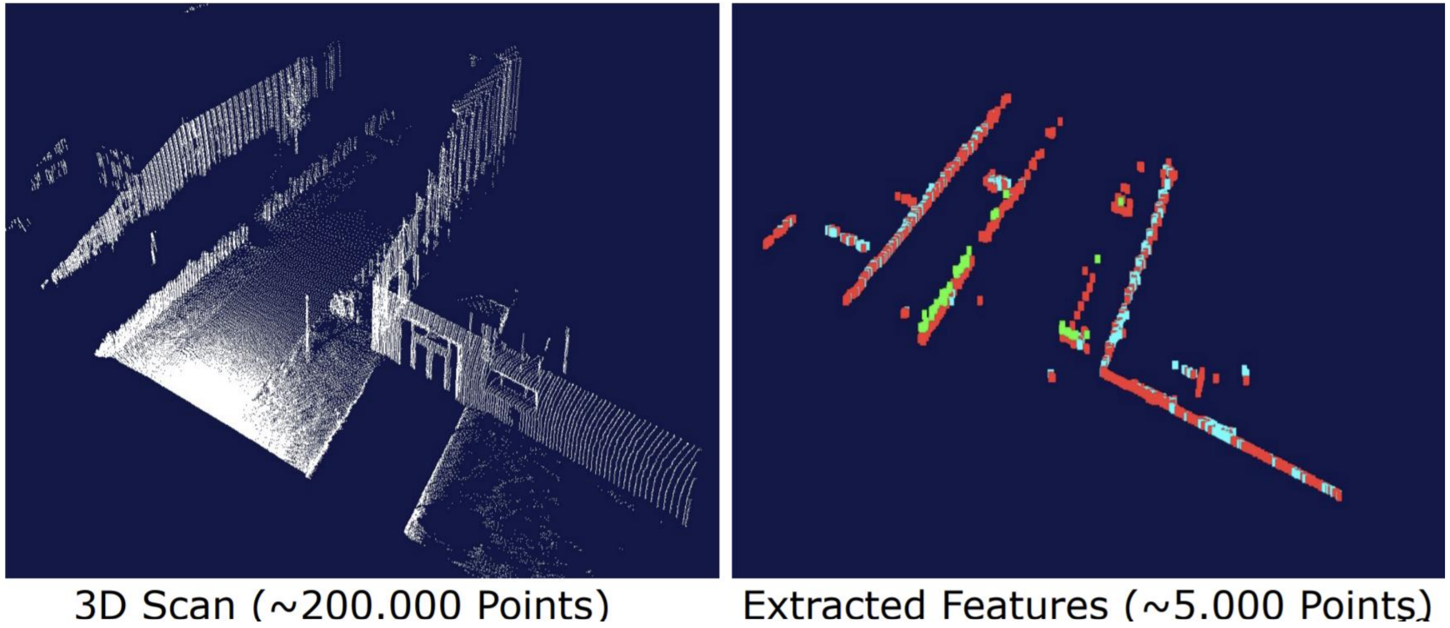


Image from Burgard, et al.

ICP variants - Change Data Association

- Beyond closest point search

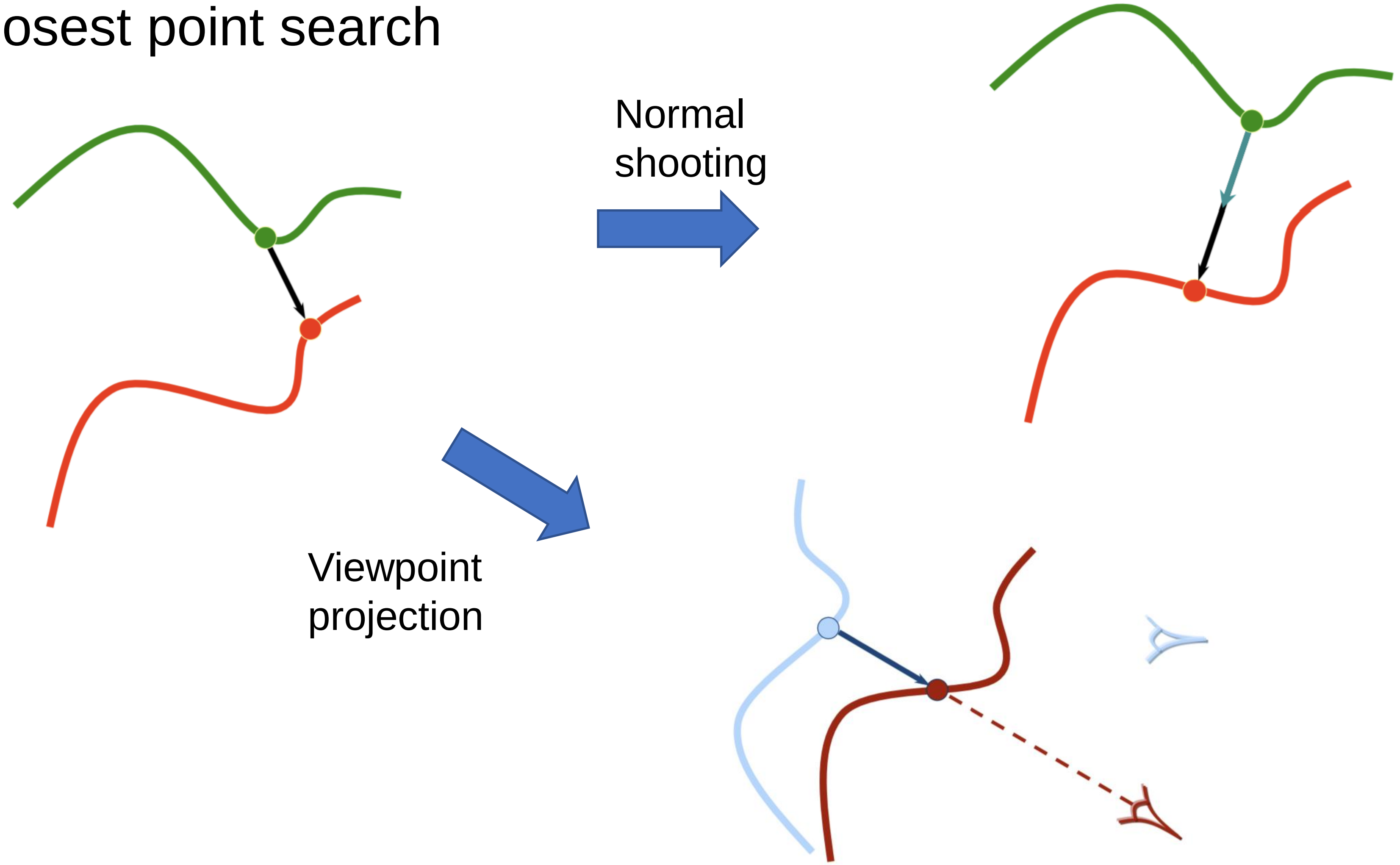


Image from Burgard, et al.

ICP for Degenerated Cases

- Solution: use color-based ICP

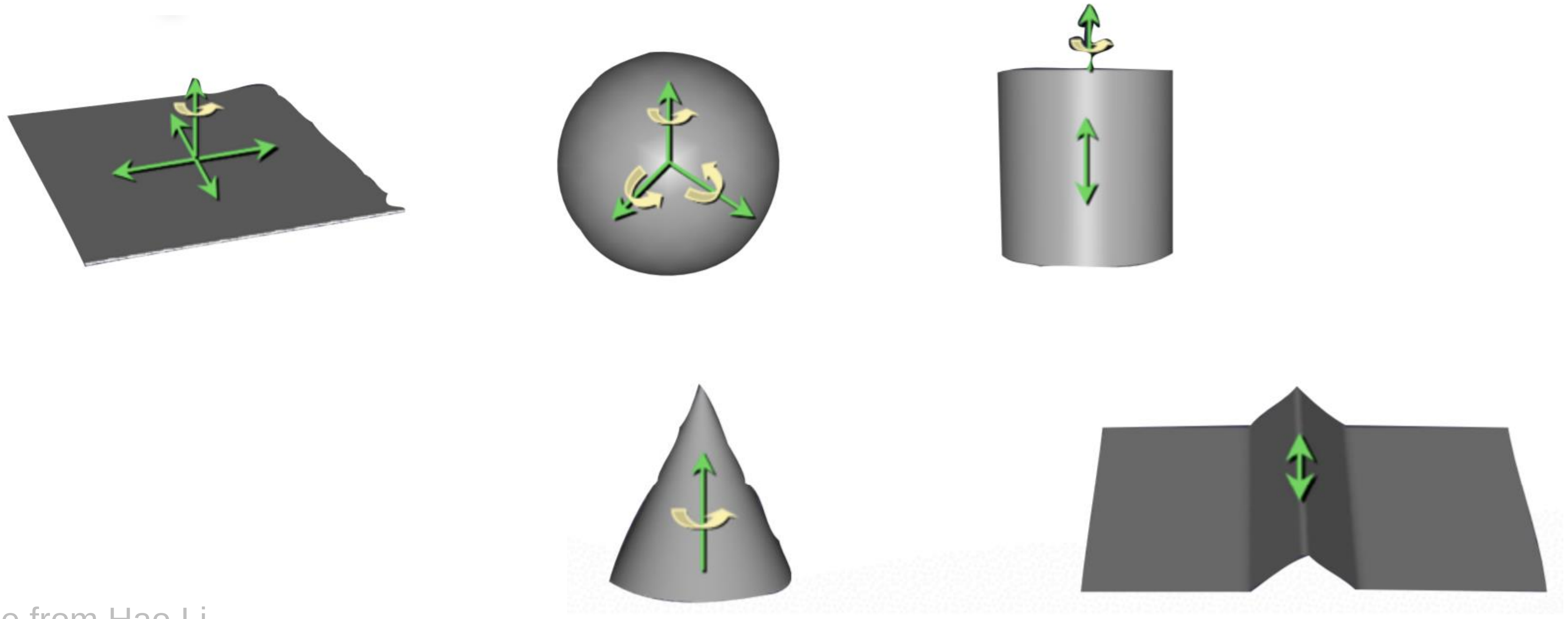


Image from Hao Li.

Global Registration – Avoid the problem

- In some cases, have 1 (possibly low-resolution) scan that covers most surface
- Align all other scans to this “anchor” [Turk 94]
- Disadvantage: not always practical to obtain anchor scan

http://www.hao-li.com/cs599-ss2015/slides/Lecture05.1.pdf

Global Registration – The greedy solution

- Align each new scan to all previous scans [Masuda '96]
- Disadvantages:
  - Order dependent
  - Doesn't spread out error

http://www.hao-li.com/cs599-ss2015/slides/Lecture05.1.pdf

Global Registration – The Brute-Force Solution

- While not converged:
  - For each scan:
    - For each point:
      - For every other scan
        - Find closest point
  - Minimize error w.r.t. transforms of **all** scans
- Disadvantage:
  - Solve (6*n*)x(6*n*) matrix equation, where *n* is number of scans

http://www.hao-li.com/cs599-ss2015/slides/Lecture05.1.pdf

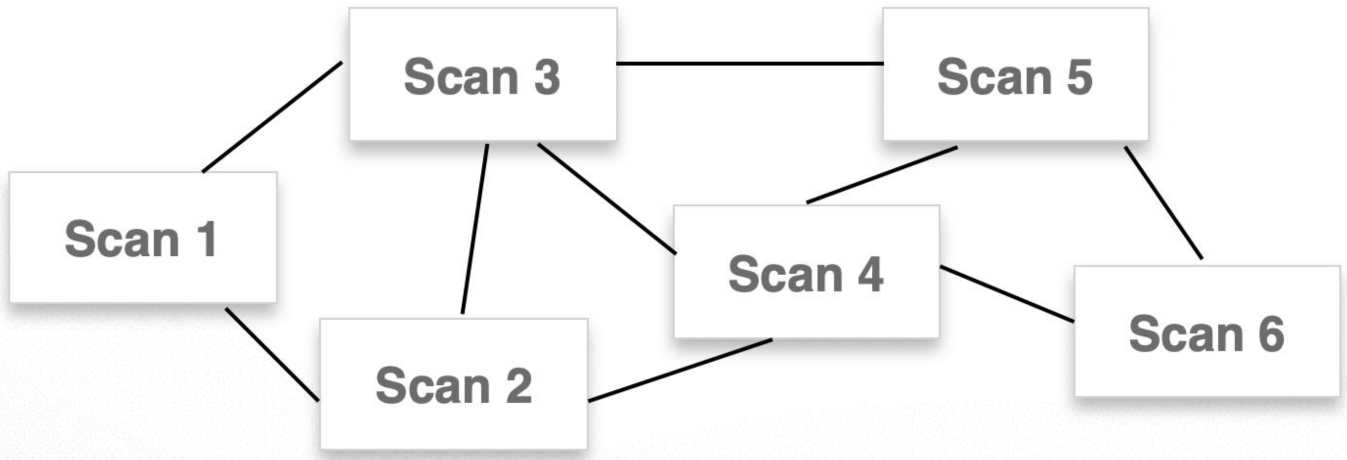
Global Registration – The Brute-Force Solution

- While not converged:
  - For each scan:
    - For each point:
      - For every other scan
        - Find closest point
  - Minimize error w.r.t. transforms of **all** scans
- Disadvantage:
  - Solve $(6n)\times(6n)$ matrix equation, where n is number of scans

http://www.hao-li.com/cs599-ss2015/slides/Lecture05.1.pdf

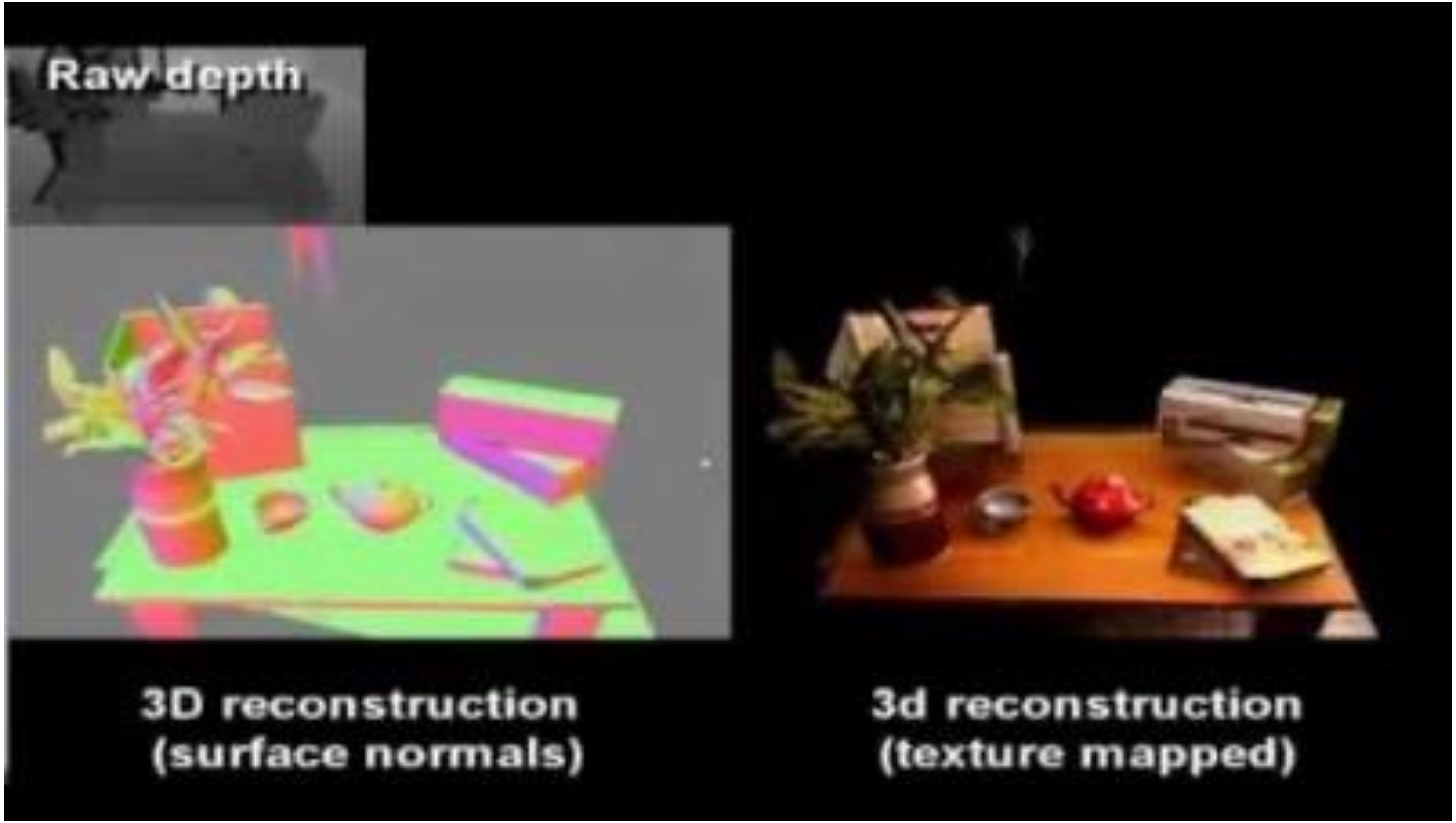
Global Registration – Graph Methods

- Many global registration algorithms create a graph of **pairwise alignments** between scans



http://www.hao-li.com/cs599-ss2015/slides/Lecture05.1.pdf

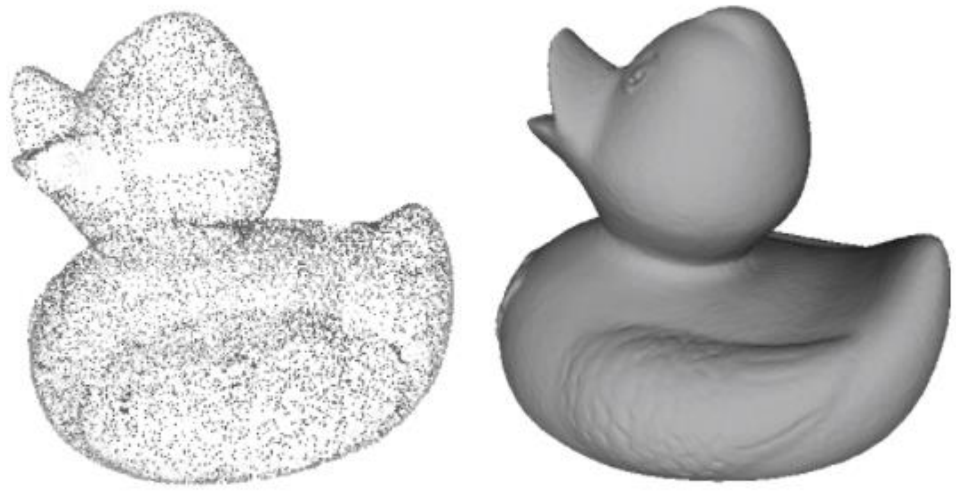
Kinect Fusion



3D Data Representations

Point Cloud/Mesh

- ✓ Raw format/Efficiency
- • Explicit representation
- × Unorganized/Unordered



https://elmoatazbill.users.greyc.fr/point_cloud/reconstruction.png

Voxel

- • Implicit representation
- × Resolution/Scalability
- × Discretization artifact

https://www.planetminecraft.com/project/giant-snowman-1638162/

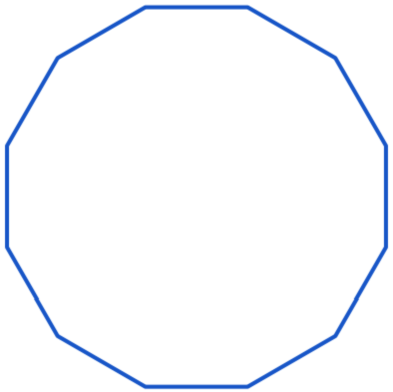
Primitives

- ✓ Compact
- ✓ Ready for interaction
- • Complex shapes?

http://pointclouds.org/gsoc/

Explicit Surface Representation

- Geometry/Shape is stored explicitly as a list of points, triangles, or other geometric fragments
  - Point clouds, mesh, …



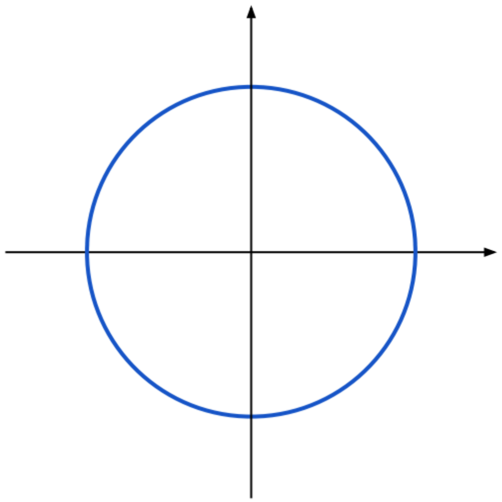
Vertices: [(x0, y0, z0), (x1, y1, z1), …, (xn, yn, zn)]

Indices: [(i0, i1), (i2, i3), …, (in-1, in)]

Slides modified from Tanner Schmidt

Implicit Surface Representation

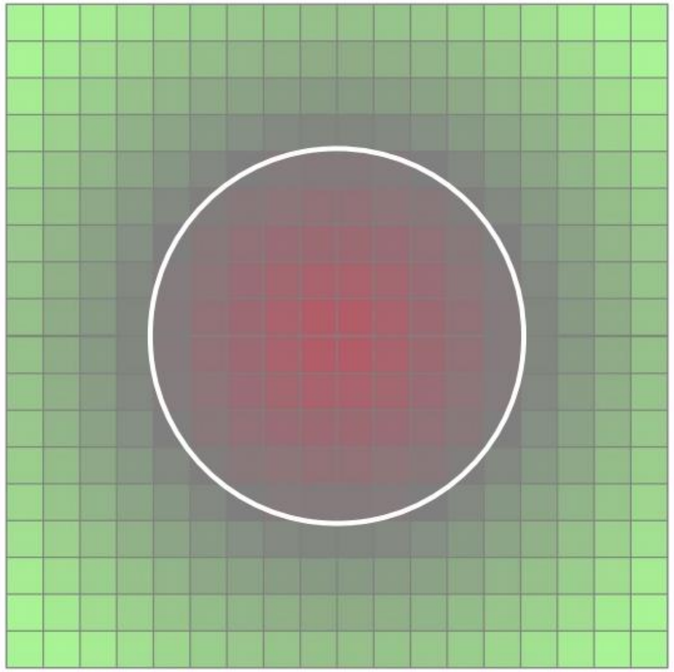
- Geometry/Shape is not stored explicitly but rather defined as a level set of a function defined over the space in which the geometry is embedded
  - Some are **parametric**



$$f(x, y) = x^2 + y^2 - r^2$$

Implicit Surface Representation

- Geometry/Shape is not stored explicitly but rather defined as a level set of a function defined over the space in which the geometry is embedded
  - Some are **non-parametric: Signed Distance Function/Field (SDF)**



Example

SDF of a circle centered at origin, with radius of 1 from before, we'll have the following SDF and sample values

$$SDF = \sqrt{x^2 + y^x} - 1$$

$$f(1,0) = 0$$

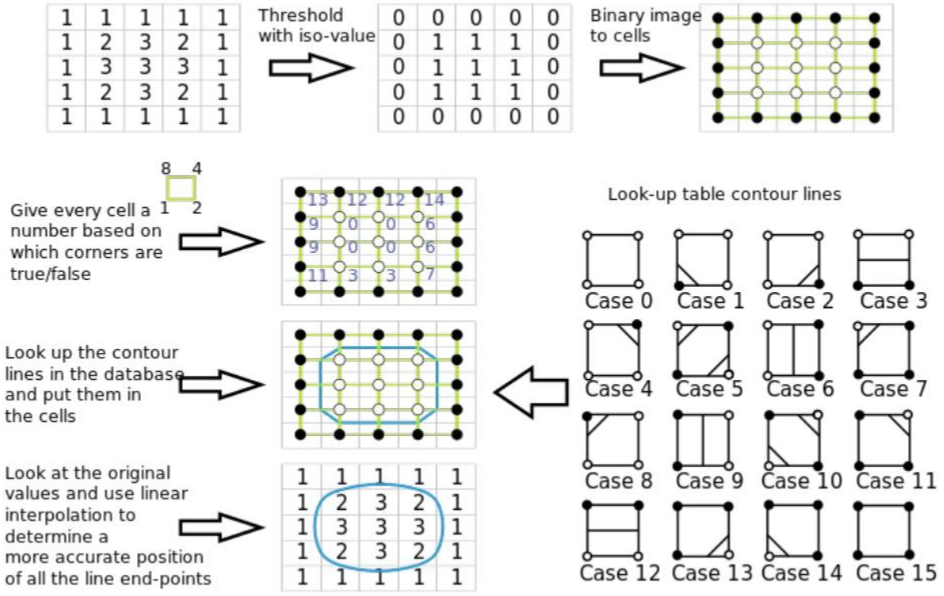
$$f(0,2) = 3$$

$$f(0,0) = -1$$

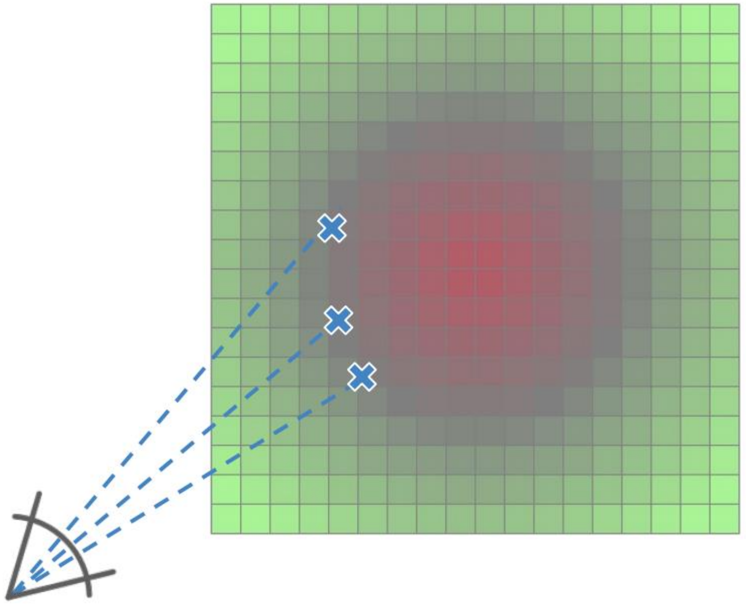
$$f(0.5,0) = -0.5$$

Implicit-to-Explicit Conversions

Marching Squares (2D)



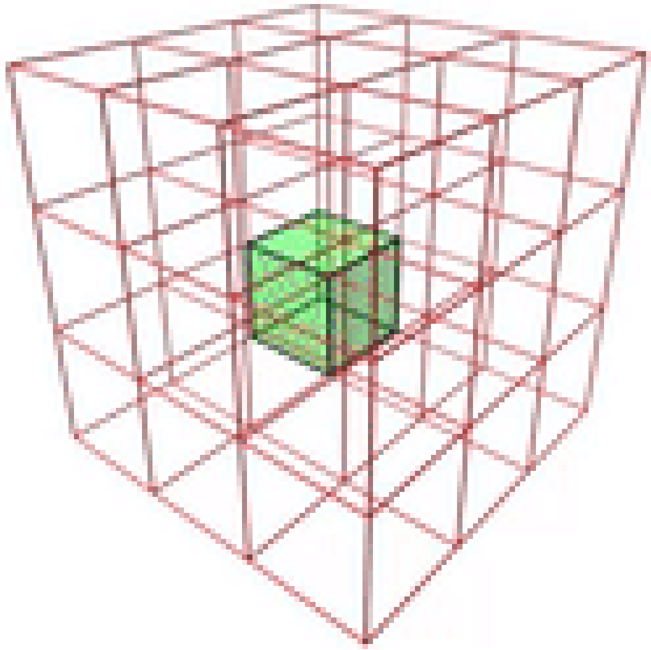
Raycasting for partial conversion

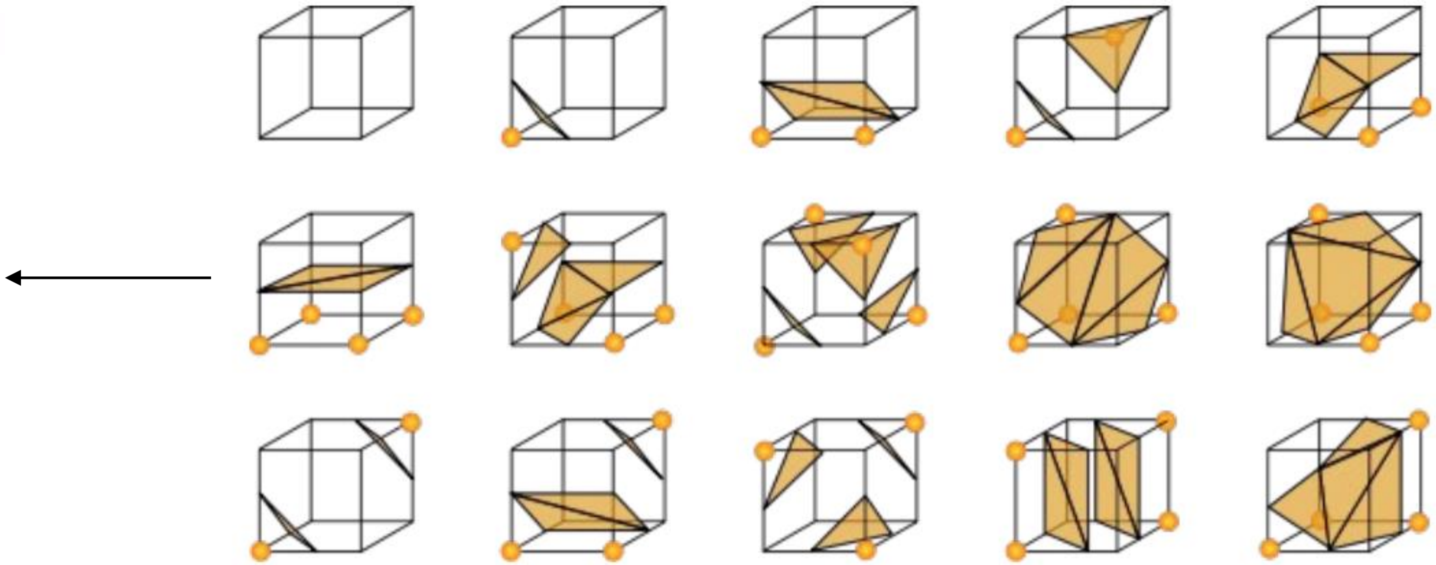


Slides modified from Tanner Schmidt

Implicit-to-Explicit Conversions

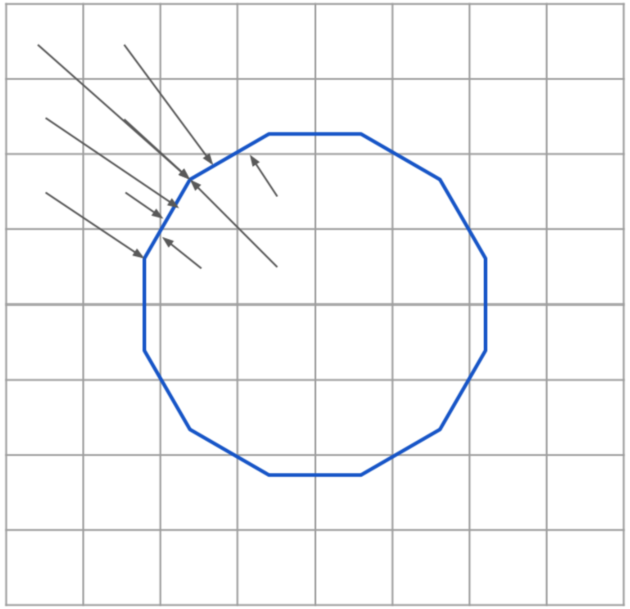
Surface Reconstruction via Marching Cubes (3D)



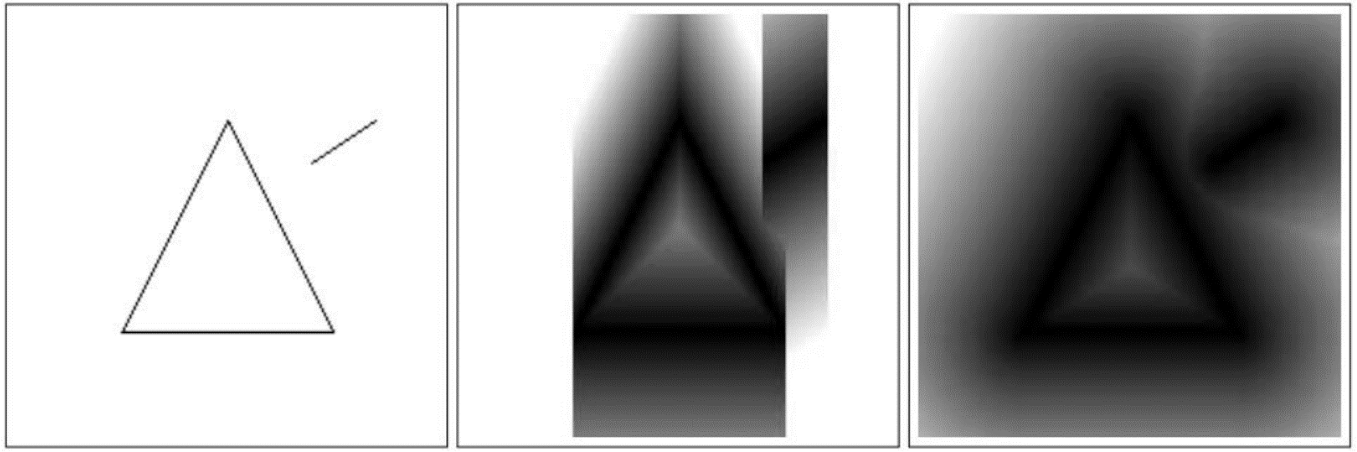


Explicit-to-Implicit Conversions

By finding closest points to the surface



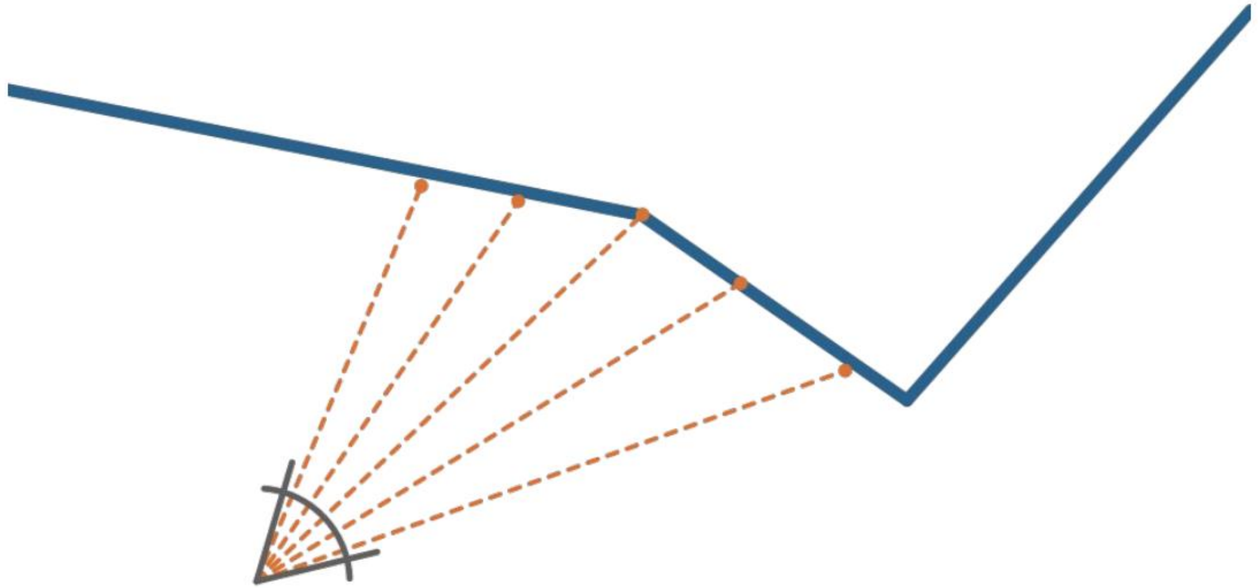
Distance Transform (2D)



Slides modified from Tanner Schmidt

Projective SDF and Truncated SDF (TSDF)

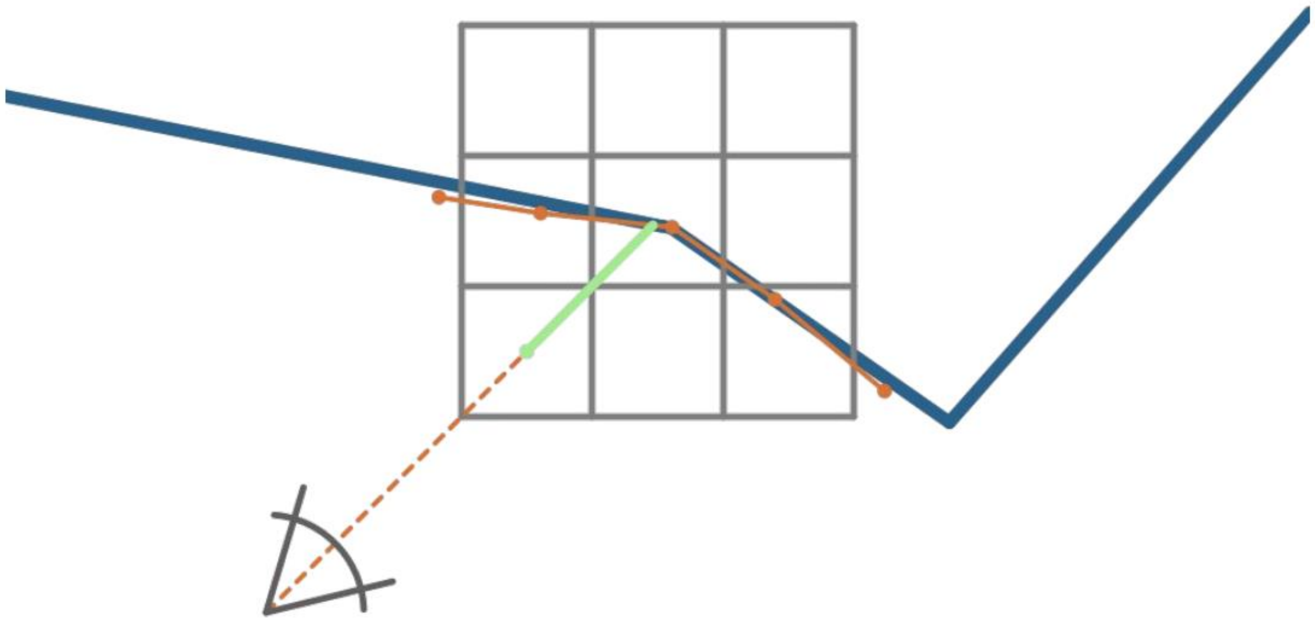
- Computing SDF requires a closed surface
- **What if I have only partial observation of the surface?**



Slides modified from Tanner Schmidt

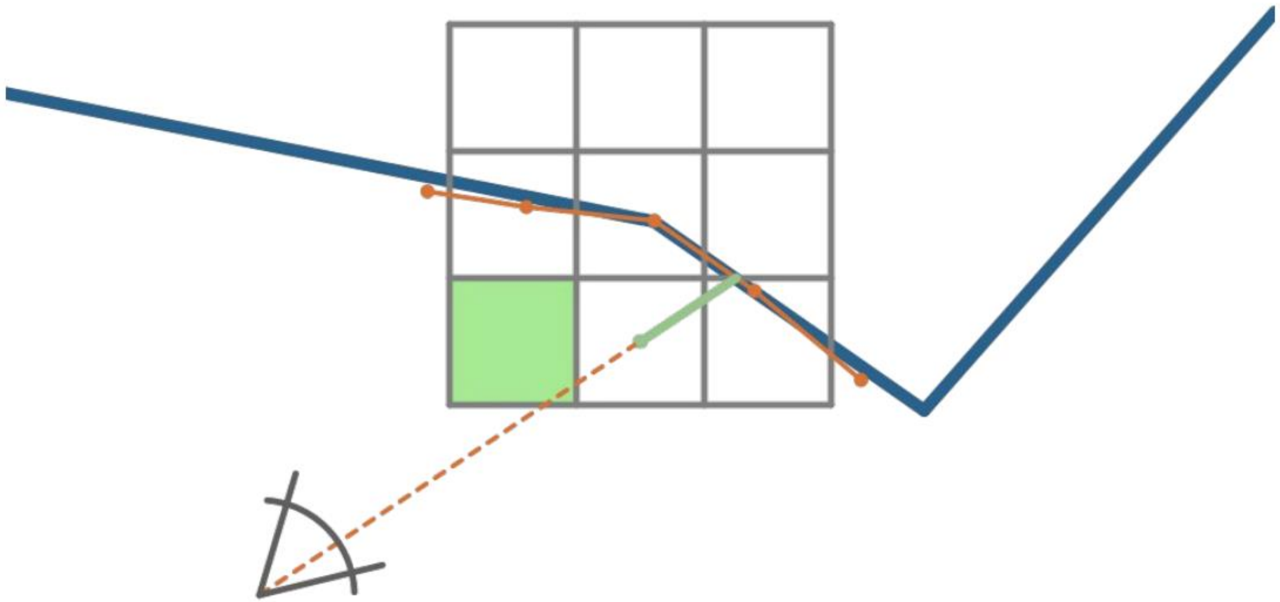
Projective SDF and Truncated SDF (TSDF)

- What if I have only partial observation of the surface?
- We can define projective SDF:



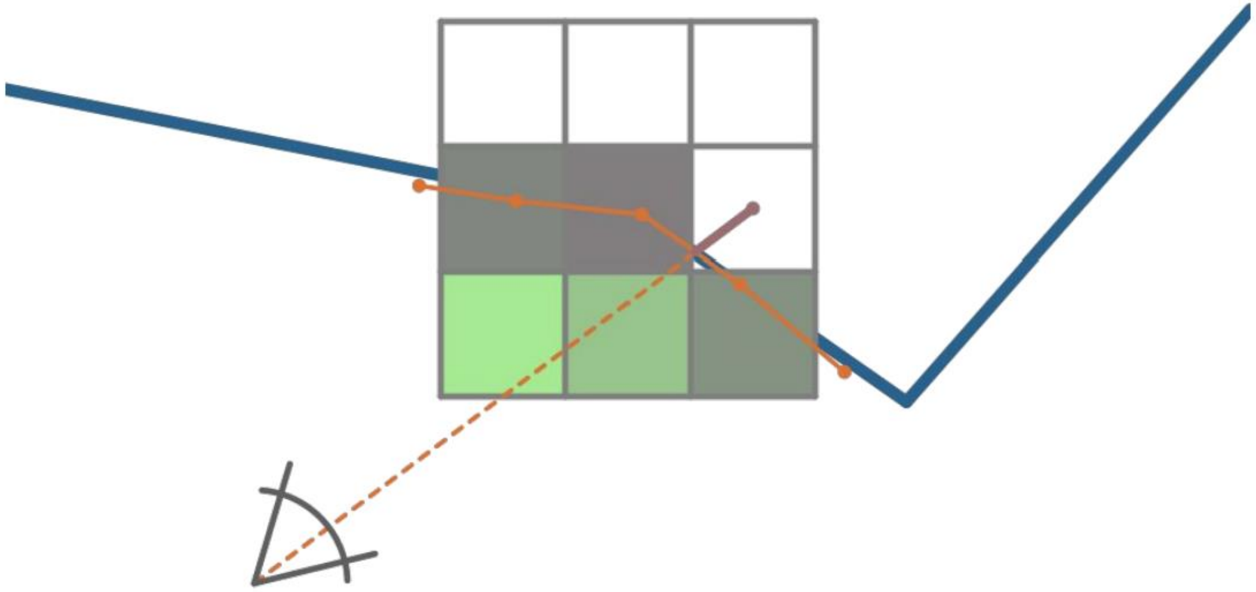
Projective SDF and Truncated SDF (TSDF)

- What if I have only partial observation of the surface?
- We can define projective SDF:



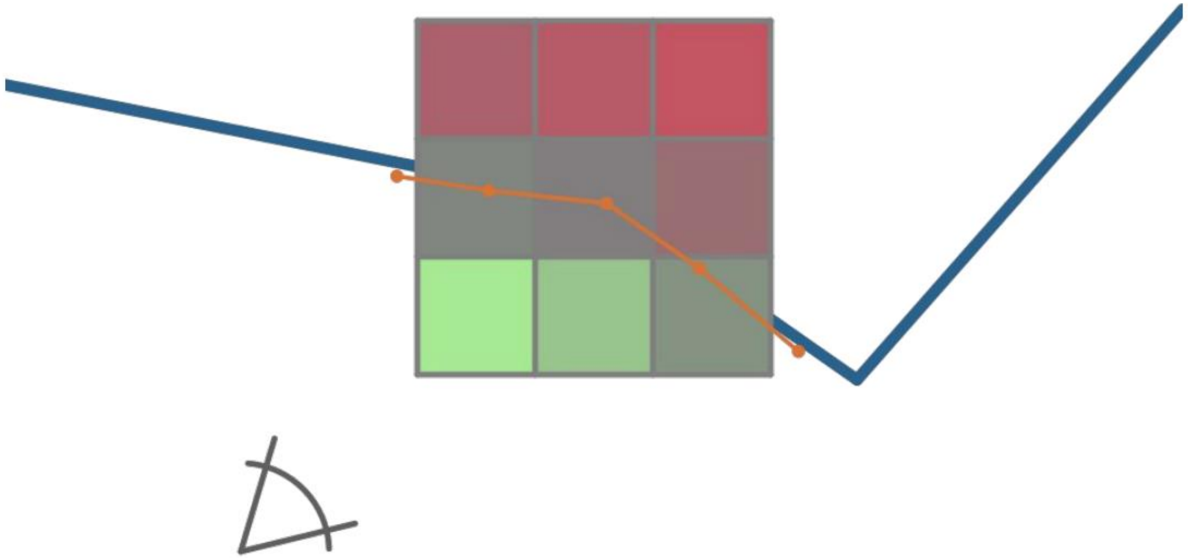
Projective SDF and Truncated SDF (TSDF)

- What if I have only partial observation of the surface?
- We can define projective SDF:



Projective SDF and Truncated SDF (TSDF)

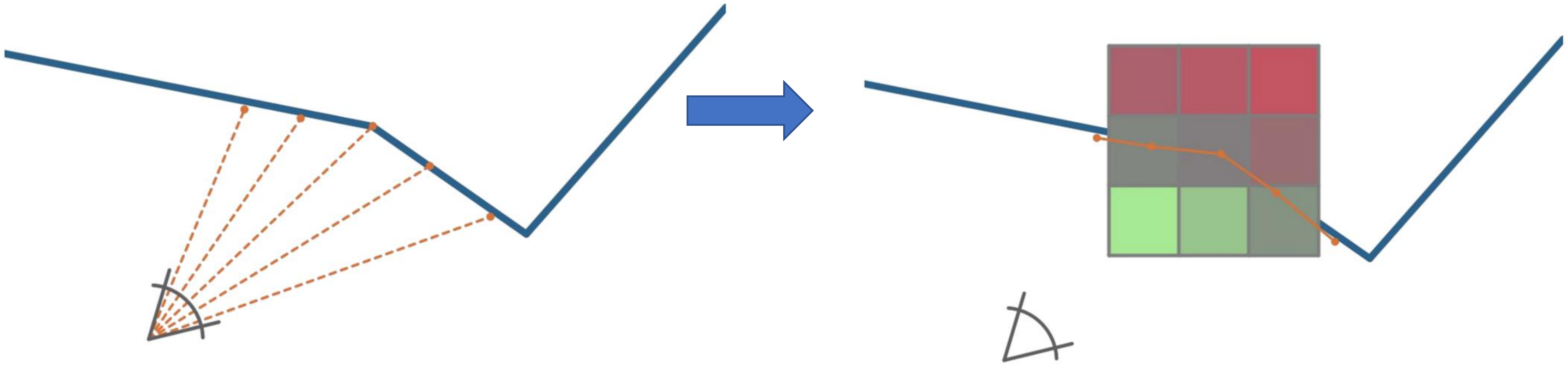
- What if I have only partial observation of the surface?
- Since we only compute the distance near both sides of the surface, we call this projective SDF as a **Truncated SDF**, or **TSDF**.



Slides modified from Tanner Schmidt

Projective SDF and Truncated SDF (TSDF)

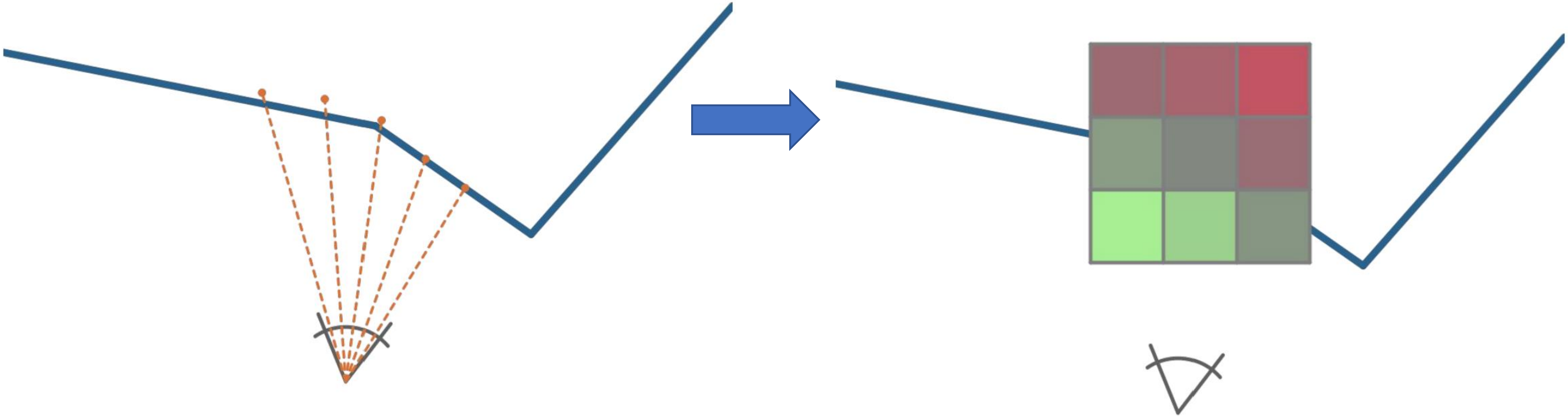
- TSDF is view dependent



Slides modified from Tanner Schmidt

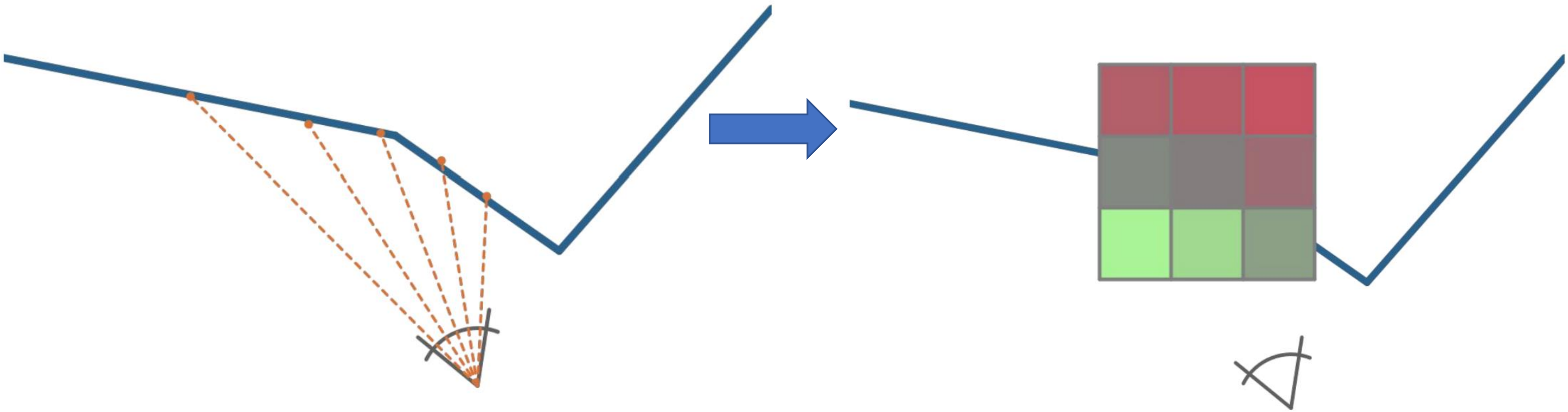
Projective SDF and Truncated SDF (TSDF)

- TSDF is view dependent



Projective SDF and Truncated SDF (TSDF)

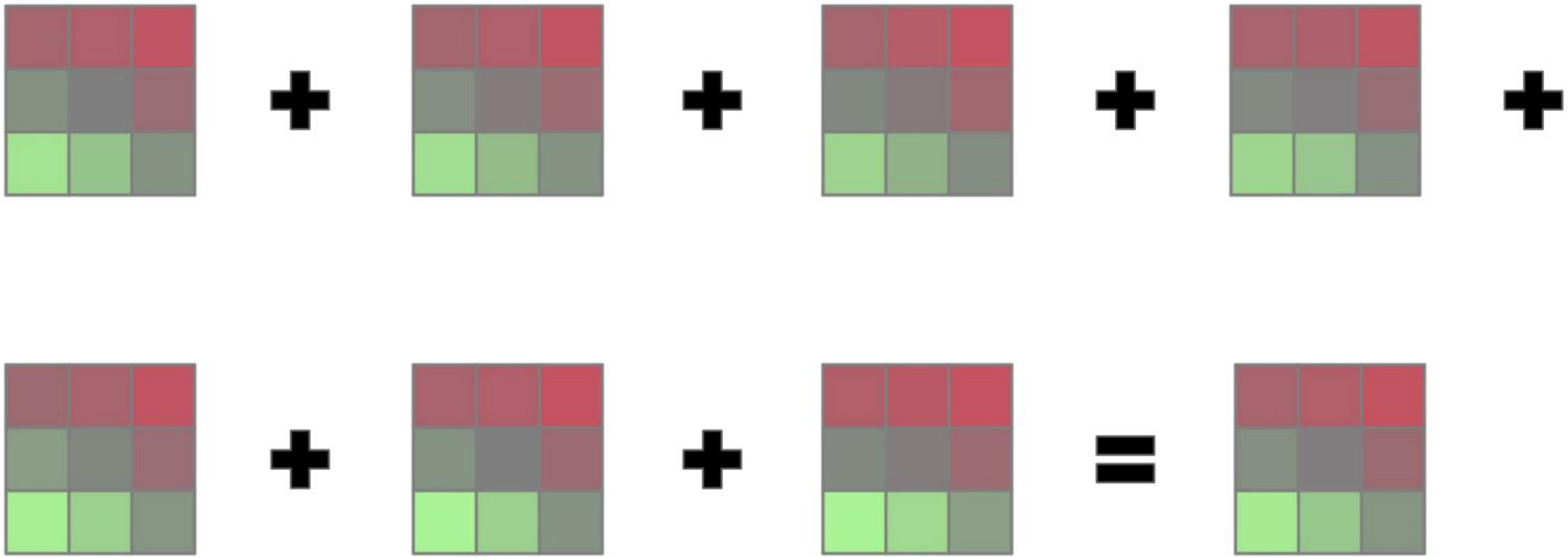
- TSDF is view dependent



Slides modified from Tanner Schmidt

Projective SDF and Truncated SDF (TSDF)

- Fusing multiple TSDF gives a good approximation of the true SDF



Slides modified from Tanner Schmidt

References

- Hartley & Zisserman 2003:
  - Section 4.7
- Corke 2011:
  - Section 14.1, 14.2.3
- Forsyth & Ponce 2011:
  - Chapter 5, Section 10.4
- Szeliski 2022:
  - Section 7.1, 7.2, 8.1.4