# ELEC 3210
# Introduction to Mobile Robotics
# Lecture 5

**(Machine Learning and Infomation Processing for Robotics)**


Huan YIN

Research Assistant Professor, Dept. of ECE

eehyin@ust.hk





# L4 - Sensors

- Sensors
  - Interoceptive: IMU
  - Exteroceptive: GNSS, Camera, LiDAR, Radar, RGBD
  - Pros and Cons of each sensor

Conclusion

- There is no perfect sensor
- Multi-Sensor fusion is the trend for robotics

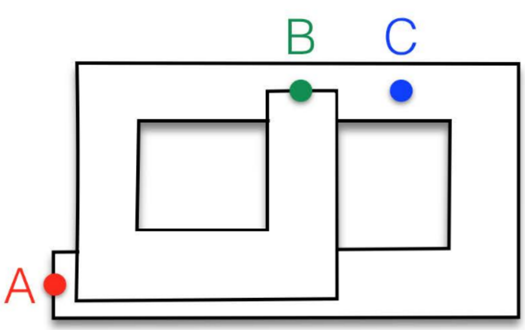
# 3D LiDAR Scanner

- LiDAR - Light Detection and Ranging
- Compared to 2D Laser Scanner
  - 3D Data
  - More expensive

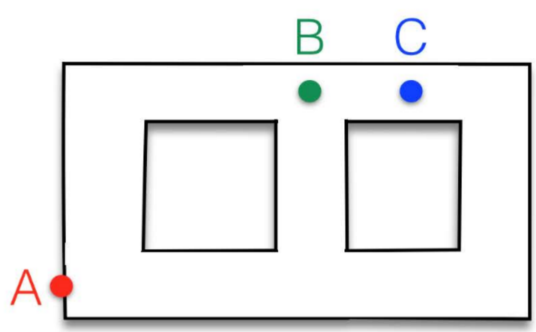

**3 Years Ago**

# L3 - Robot Localization

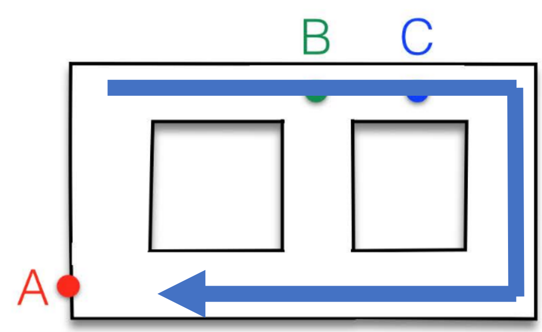
- **Odometry**
  - Wheel Odometry
  - Visual Odometry
  - LiDAR Odometry
  - etc




- SLAM
  - Simultaneous localization and mapping

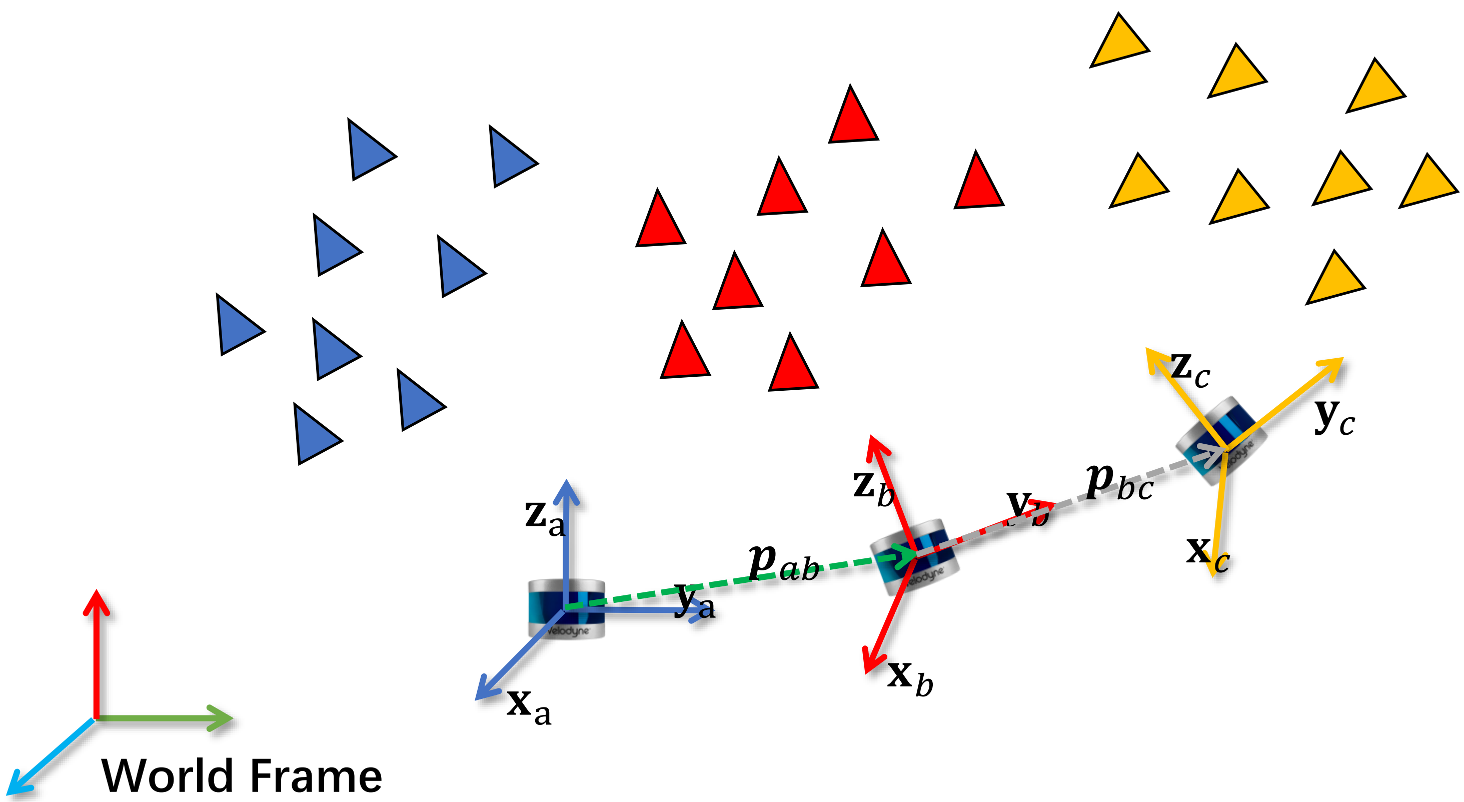



- Map-based Localization
  - Localize on a given map

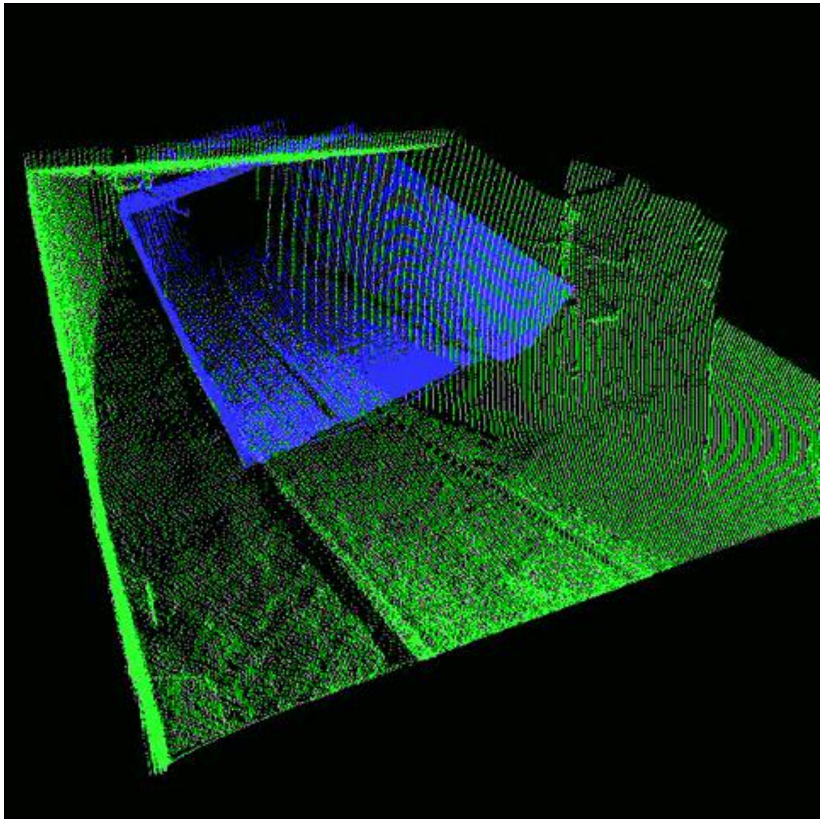


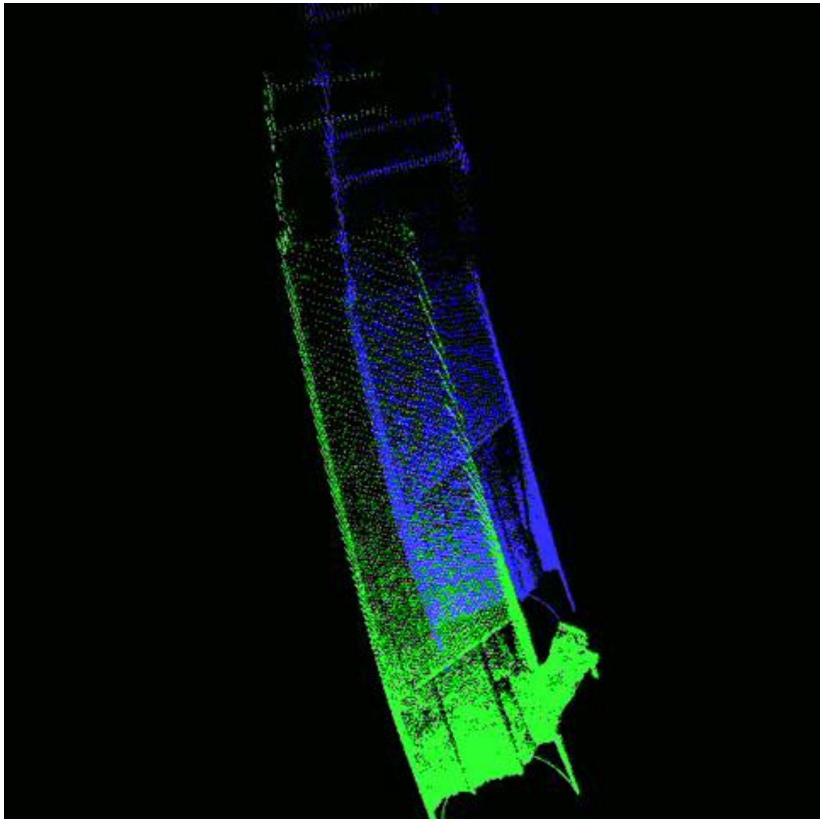
# LiDAR odometry by ICP

- Iterative Closest Points (ICP)
- A reduced LiDAR SLAM system without loop closure
  - simple but useful
  - only point cloud registration/alignment



# Iterative Closest Point



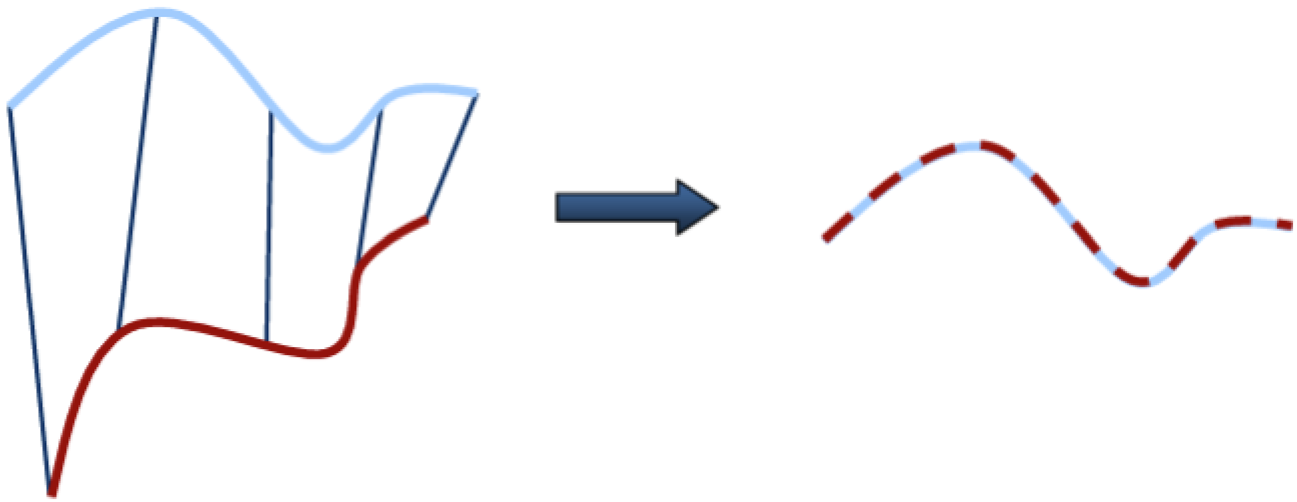


# Laser Mapping


Courtesy: Shaojie Shen

# ICP - Alignment of 3D Points

- Goal:
  - find the parameter of transformation that best aligns two point sets
- Two main steps:
  - Find the correspondences
  - Estimate the transformation



Courtesy: Cyrill Stachniss

# Correpondence

- Student: "What are the three most important problems in computer vision?"
- Takeo Kanade: "Correspondence, correspondence, correspondence!"

**Prof. Takeo Kanade**

# Known correspondences

# Notations

- Given two point clouds
  - Source: $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_J\}$
  - Target: $Y = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_I\}$
  - with known correspondences: $\mathcal{C} = \{(i, j)\}$
- Estimate translation and rotation that minimize the sum of the squared errors:

$$\sum_{(i,j)\in C} \|\boldsymbol{y}_i - R\boldsymbol{x}_j - \boldsymbol{t}\|^2 \to \min$$

Courtesy: Cyrill Stachniss

# Notations

- Reorder point clouds given the correspondences with index
- Point Clouds: $\{\boldsymbol{x}_n\}\,\{\boldsymbol{y}_n\}$
- Find the rigid body transformation

$$\overline{\boldsymbol{x}}_n = R\boldsymbol{x}_n + \boldsymbol{t} \quad n = 1, \ldots, |\mathcal{C}| =: N$$

- The transformed point cloud $\{\overline{\boldsymbol{x}}_n\}$ will be as close as possible to the target point cloud $\{\boldsymbol{y}_n\}$

Courtesy: Cyrill Stachniss

# Notations

- Reorder point clouds given the correspondences with index
- Point Clouds: $\{\boldsymbol{x}_n\}\,\{\boldsymbol{y}_n\}$
- Find the rigid body transformation

$$\overline{\boldsymbol{x}}_n = R\boldsymbol{x}_n + \boldsymbol{t} \quad n = 1, \ldots, |\mathcal{C}| =: N$$

- The transformed point cloud $\{\overline{\boldsymbol{x}}_n\}$ will be as close as possible to the target point cloud $\{\boldsymbol{y}_n\}$

- Non-rigid?

$$\overline{\boldsymbol{x}}_n = \lambda R\boldsymbol{x}_n + \boldsymbol{t}$$

Courtesy: Cyrill Stachniss

# Formal Problem Definition

- Given corresponding points

$$\boldsymbol{y}_n, \boldsymbol{x}_n \quad n = 1, \ldots, N$$

- and optional weights:

$$p_n \quad n = 1, \ldots, N$$

- Find the transformation of the rigid body transformation:

$$\overline{\boldsymbol{x}}_n = R\boldsymbol{x}_n + \boldsymbol{t} \quad n = 1, \ldots, N$$

- so that the squared error is minimized:

$$\sum \|\boldsymbol{y}_n - \overline{\boldsymbol{x}}_n\|^2 p_n \to \min$$

Courtesy: Cyrill Stachniss

# Direct Optimal Solution

- There exists a direct and optimal solution
  - Direct = no initial guess needed
  - Optimal = no better solution exists

- Informally speaking:
  - Computes a **shift** involving the **center of masses** of both point clouds
  - Performs a rotational alignment using **singular value decomposition (SVD)**

Courtesy: Cyrill Stachniss

# Computing the Rotation Matrix

$$\boldsymbol{y}_0 = \frac{\sum \boldsymbol{y}_n p_n}{\sum p_n} \qquad \boldsymbol{x}_0 = \frac{\sum \boldsymbol{x}_n p_n}{\sum p_n}$$

$$\boldsymbol{H} = \sum \left(\boldsymbol{y}_n - \boldsymbol{y}_0\right)\left(\boldsymbol{x}_n - \boldsymbol{x}_0\right)^\top p_n$$

$$\operatorname{svd}(H) = UDV^\top$$

$$R = VU^\top$$

Courtesy: Cyrill Stachniss

# Singular Value Decomposition

- The SVD is a matrix factorization of a $m \times n$ matrix into

$$A = U\Sigma V^T$$

where U is a $m \times m$ **orthogonal** matrix, VT is a $n \times n$ **orthogonal** matrix and $\boldsymbol{\Sigma}$ is a $m \times n$ **diagonal** matrix.

- For a square matrix ($m$=$n$):

$$\boldsymbol{A} = \begin{pmatrix} \vdots & \dots & \vdots \\ \boldsymbol{u}_1 & \dots & \boldsymbol{u}_n \\ \vdots & \dots & \vdots \end{pmatrix} \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix} \begin{pmatrix} \dots & \mathbf{v}_1^T & \dots \\ \vdots & \vdots & \vdots \\ \dots & \mathbf{v}_n^T & \dots \end{pmatrix}$$

$$\boldsymbol{A} = \begin{pmatrix} \vdots & \dots & \vdots \\ \boldsymbol{u}_1 & \dots & \boldsymbol{u}_n \\ \vdots & \dots & \vdots \end{pmatrix} \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix} \begin{pmatrix} \vdots & \dots & \vdots \\ \boldsymbol{v}_1 & \dots & \boldsymbol{v}_n \\ \vdots & \dots & \vdots \end{pmatrix}^T$$

$$\sigma_1 \geq \sigma_2 \geq \sigma_3 \dots$$

Courtesy: UIUC

# Why shift and rotate?

- Symbols change slightly (latex on powerpoint ☹)
- We solve a minimization problem for N >= 3 point correspondences:

$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i}^{N} \|\boldsymbol{y}_i - (\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t})\|^2$$

- After differentiating with respect to $\boldsymbol{t}$, we observe that the translation is the difference between the centroids:

$$\mathbf{t} = \frac{1}{N}\sum_{i}^{N} \boldsymbol{y}_i - \boldsymbol{R}\frac{1}{N}\sum_{i}^{N} \boldsymbol{x}_i = \boldsymbol{y}_0 - \boldsymbol{R}\boldsymbol{x}_0$$

Courtesy: Shaojie Shen

# Why SVD?

- The objective function as

$$\min_{R} \|\boldsymbol{Y} - \boldsymbol{RX}\|_F^2$$

where

$$\boldsymbol{Y} = [\boldsymbol{y}_1 - \boldsymbol{y}_{\mathbf{0}}, \dots, \boldsymbol{y}_n - \boldsymbol{y}_{\mathbf{0}}]$$

and

$$\boldsymbol{X} = [\boldsymbol{x}_1 - \boldsymbol{x}_{\mathbf{0}}, \dots, \boldsymbol{x}_n - \boldsymbol{x}_{\mathbf{0}}]$$

- Some useful mathematics
  - Frobenius norm $\|\boldsymbol{A}\|_F = \sqrt{\sum\sum|a_{ij}|^2} \rightarrow \|\boldsymbol{A}\|_F = \sqrt{tr(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})}$
  - $\mathrm{t}r(\boldsymbol{AB}) = tr(\boldsymbol{BA})$
  - $\mathrm{t}r(\boldsymbol{A}) = tr(\boldsymbol{A}^T)$
  - $\mathrm{t}r(\boldsymbol{A} + \boldsymbol{B}) = tr(\boldsymbol{A}) + tr(\boldsymbol{B})$

Courtesy: Shaojie Shen

# Why SVD

- We rewrite the Frobenius norm using the trace of the matrix
$$\|\boldsymbol{Y}-\boldsymbol{R}\boldsymbol{X}\|_F^2 = tr(\boldsymbol{Y}^T\boldsymbol{Y}) + tr(\boldsymbol{X}^T\boldsymbol{X}) - tr(\boldsymbol{Y}^T\boldsymbol{R}\boldsymbol{X}) - tr\left(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{R}^{\boldsymbol{T}}\boldsymbol{Y}\right)$$

- And observe that only the two last terms depend on the unknown $\boldsymbol{R}$ yielding a maximization problem.

- Even without using the properties of the trace we can see that both last terms are equal to
$$\sum_i^N \boldsymbol{R}(\boldsymbol{x}_i - \boldsymbol{x}_{\boldsymbol{0}})(\boldsymbol{y}_i - \boldsymbol{y}_{\boldsymbol{0}})^T = tr(\boldsymbol{R}\boldsymbol{X}\boldsymbol{Y}^T)$$

- The 3D-3D pose problem reduced to
$$\max_{\boldsymbol{R}} \; tr(\boldsymbol{R}\boldsymbol{X}\boldsymbol{Y}^T)$$

Courtesy: Shaojie Shen

# Why SVD?

- If the SVD of $\boldsymbol{XY}^T$ is $\boldsymbol{USV}^{\boldsymbol{T}}$ and let $\boldsymbol{Z} = \boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{RU}$

$$tr\left(\boldsymbol{RXY}^{\boldsymbol{T}}\right) = tr\left(\boldsymbol{RUSV}^{\boldsymbol{T}}\right) = tr(\boldsymbol{ZS}) = \sum_{1}^{3} z_{ii}\sigma_i \leq \sum_{1}^{3} \sigma_i$$

- The upper bound is obtained by setting

$$\boldsymbol{R} = \boldsymbol{VU}^{\boldsymbol{T}}$$

Courtesy: Shaojie Shen

# Computing the Translation Vector

$$\boldsymbol{y}_0 = \frac{\sum \boldsymbol{y}_n p_n}{\sum p_n} \qquad R = VU^\top \qquad \boldsymbol{x}_0 = \frac{\sum \boldsymbol{x}_n p_n}{\sum p_n}$$

$$\boldsymbol{t} = \boldsymbol{y}_0 - R\boldsymbol{x}_0$$

Courtesy: Cyrill Stachniss

# SVD-based alignment (1)

- Compute means of the point clouds

$$\boldsymbol{x}_0 = \frac{\sum \boldsymbol{x}_n p_n}{\sum p_n}$$

$$\boldsymbol{y}_0 = \frac{\sum \boldsymbol{y}_n p_n}{\sum p_n}$$

- Compute mean-reduced coordinates

$$\boldsymbol{b}_n = (\boldsymbol{x}_n - \boldsymbol{x}_0)$$

$$\boldsymbol{a}_n = (\boldsymbol{y}_n - \boldsymbol{y}_0)$$

- Compute cross covariance matrix

$$H = \sum \boldsymbol{a}_n \boldsymbol{b}_n^\top p_n$$

Courtesy: Cyrill Stachniss

# SVD-based alignment (2)

- Compute SVD

$$\mathrm{svd}(H) = UDV^\top$$

- Rotation matrix is given by

$$R = VU^\top$$

- Translation vector is given by

$$\boldsymbol{t} = \boldsymbol{y}_0 - R\boldsymbol{x}_0$$

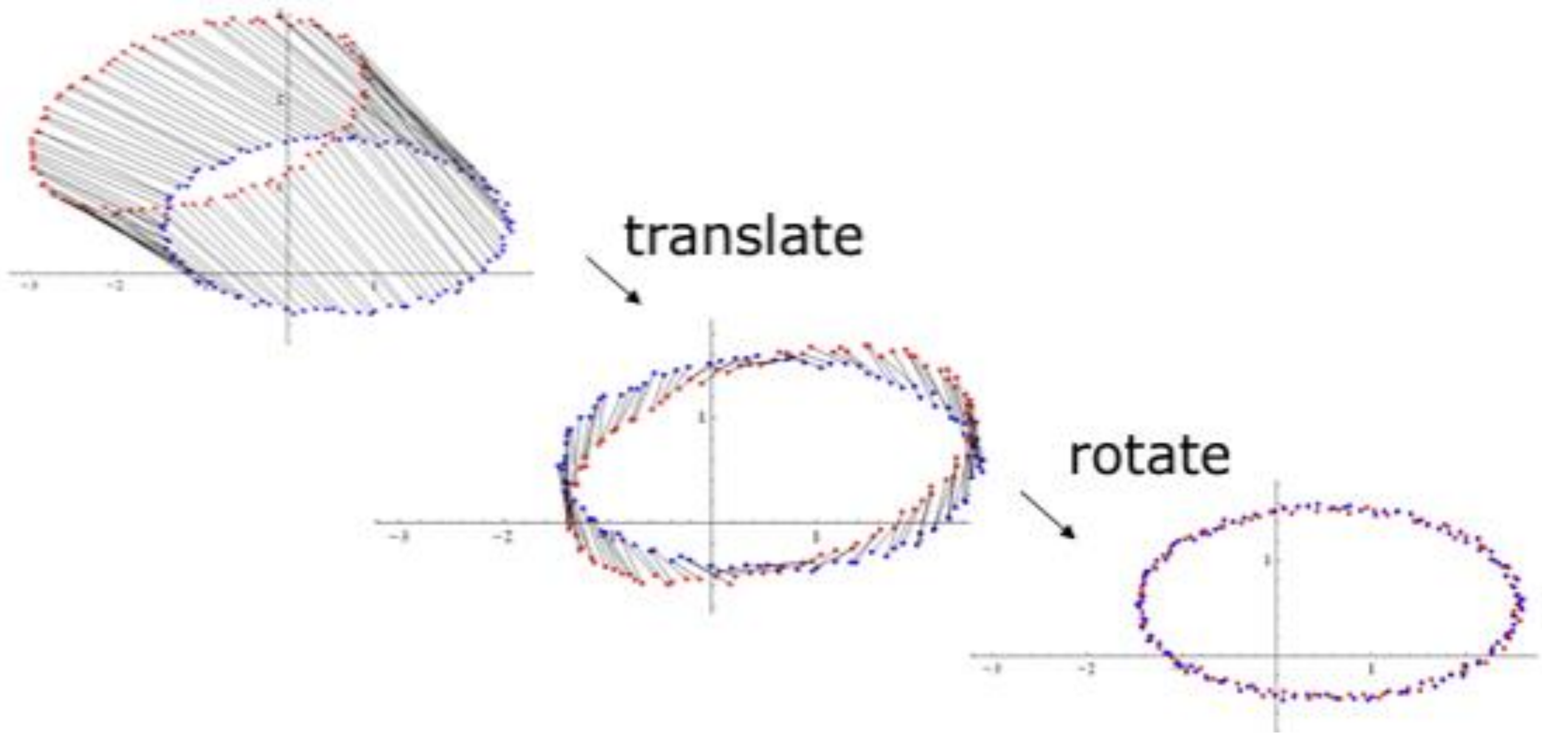
- Translate and Rotate points

$$\overline{\boldsymbol{x}}_n = R\boldsymbol{x}_n + \boldsymbol{t} \quad n = 1, \ldots, N$$

Courtesy: Cyrill Stachniss

# SVD-Based Alignment Summary

- Alignment through translation and rotation

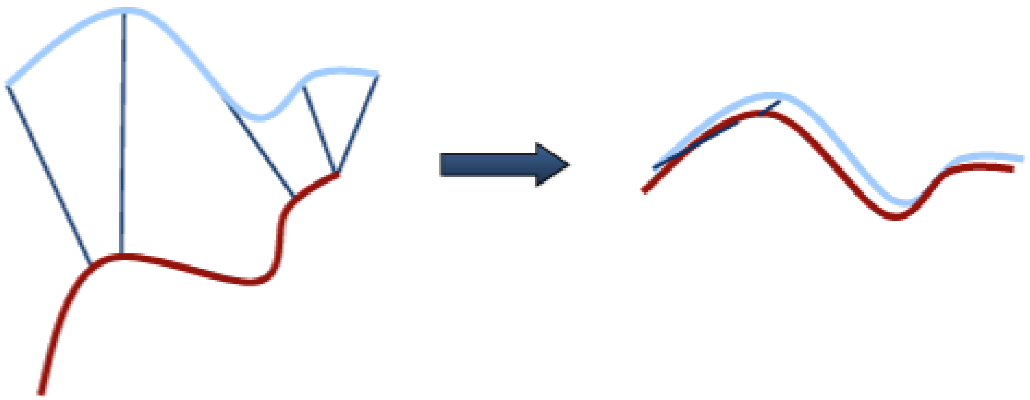



Courtesy: Cyrill Stachniss, Ju

# Iterative Closest Point

# Correspondences

- If the correct correspondences are not known, it is generally impossible to determine the optimal relative rotation and translation in one step.



Courtesy: Wolfram Burgard

# Iterative Closest Point

- Idea: **iterative** to find the alignment
- Besl, Paul J., and Neil D. McKay. 1992.

Method for registration of 3-D shapes

PJ **Besl**, ND McKay - Sensor fusion IV: control paradigms and …, **1992** - spiedigitallibrary.org

This paper describes a general purpose, representation independent method for the accurate and computationally efficient registration of 3-D shapes including free-form curves and surfaces. The method handles the full six-degrees of freedom and is based on the iterative closest point (ICP) algorithm, which requires only a procedure to find the closest point on a geometric entity to a given point. The ICP algorithm always converges monotonically to the nearest local minimum of a mean-square distance metric, and …



Courtesy: Wolfram Burgard

# Basic ICP algorithm

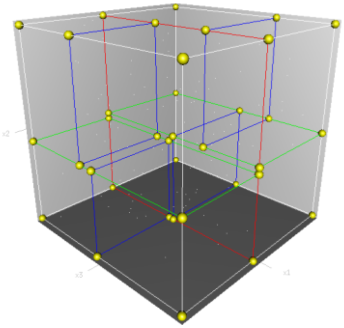
- Start with some initial guess of rotation and translation
- For each point in pointcloud 1, find its **nearest neighbor** in pointcloud 2 based on the current estimated rotation and translation
- Refine the rotation and translation based on the latest data association
- Iterate from step 2 until converge

Courtesy: Shaojie Shen

# Basic ICP algorithm

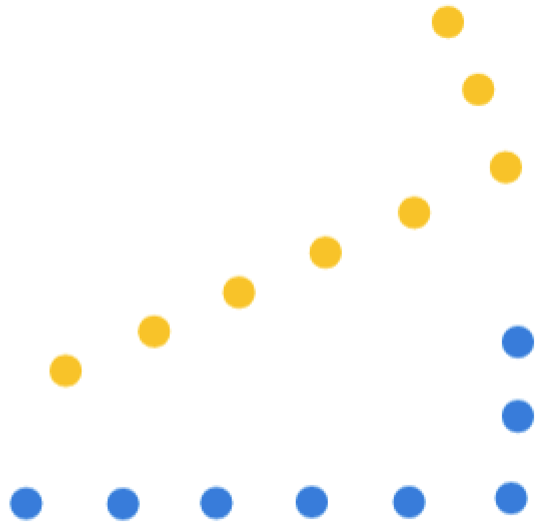
- Start with **some initial guess** of rotation and translation
- For each point in pointcloud 1, find its **nearest neighbor** in pointcloud 2 based on the current estimated rotation and translation
- Refine the rotation and translation based on the latest data association
- Iterate from step 2 until converge

- Nearest Neighbor Search
  - Need to speed up the search of nearest neighbors
  - Naive implementation: $O(N)$
  - K-d Tree: $O(logN)$



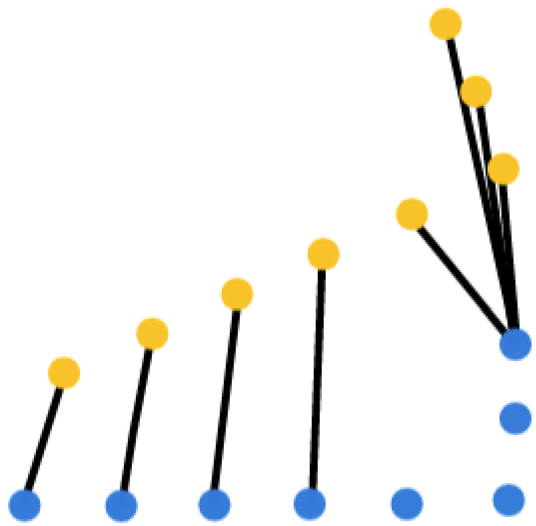
Courtesy: Shaojie Shen

# Basic ICP algorithm



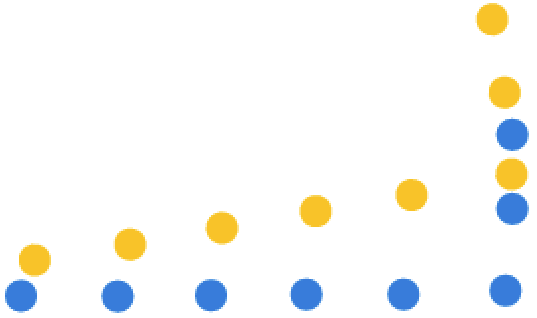
Courtesy:

# Basic ICP algorithm



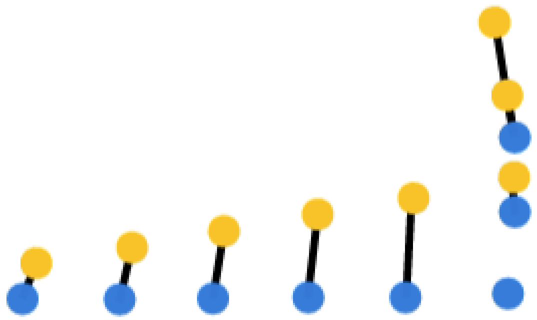
Courtesy:

# Basic ICP algorithm



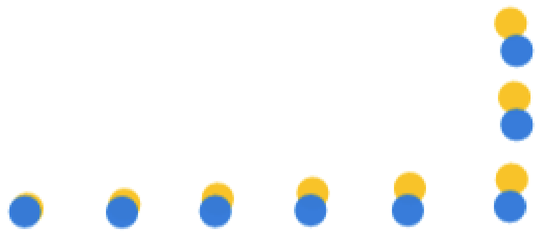
Courtesy:

# Basic ICP algorithm



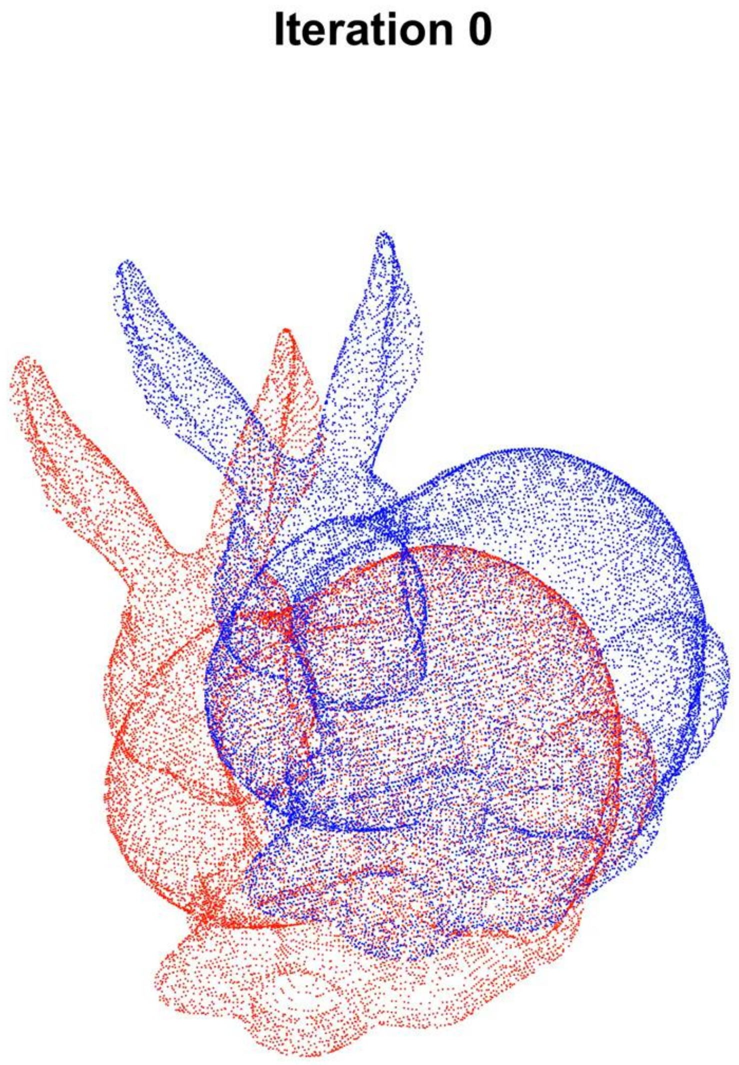
Courtesy:

# Basic ICP algorithm



Courtesy:

# Iterative Closest Point

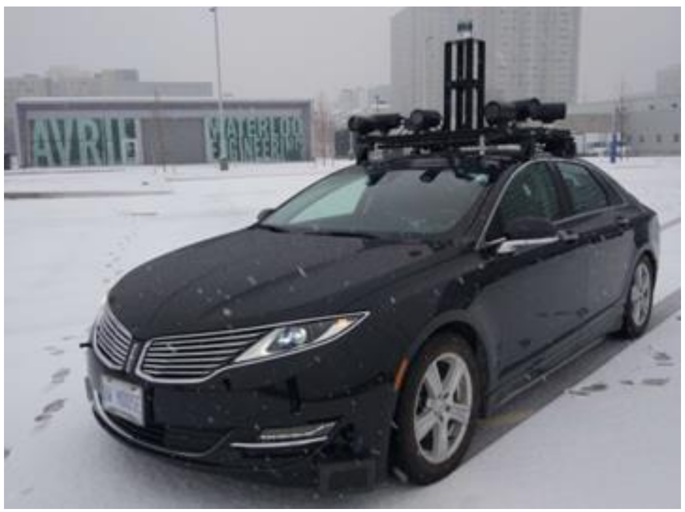


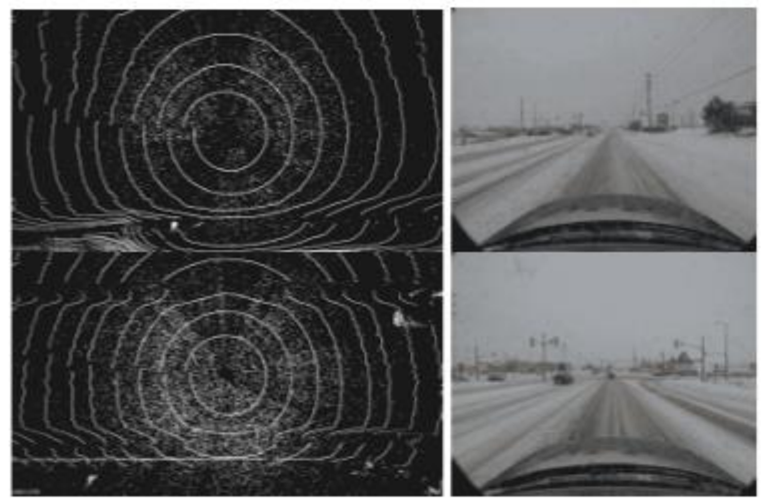
# Weakness of ICP

- Start with some **initial guess** of rotation and translation
- For each point in pointcloud 1, find its nearest neighbor in pointcloud 2 based on the current estimated rotation and translation
- Refine the rotation and translation based on the latest data association
- Iterate from step 2 until converge

# Real World

- Dense, Noises, Occluded etc.
- Such as self-driving in the snow





Courtesy: Canadian Adverse Driving Dataset

# ICP Variants

# Performance of Variants

- Speed
- Stability (local minima)
- Tolerance w.r.t. noise and outliers
- Basin of convergence (maximum initial misalignment)

Courtesy: Wolfram Burgard

# ICP Variants

- Select point cloud subsets (Samping/ Filter)
- Wighting the correspondences
- Data Associations
- Reject certain (outlier) point pairs

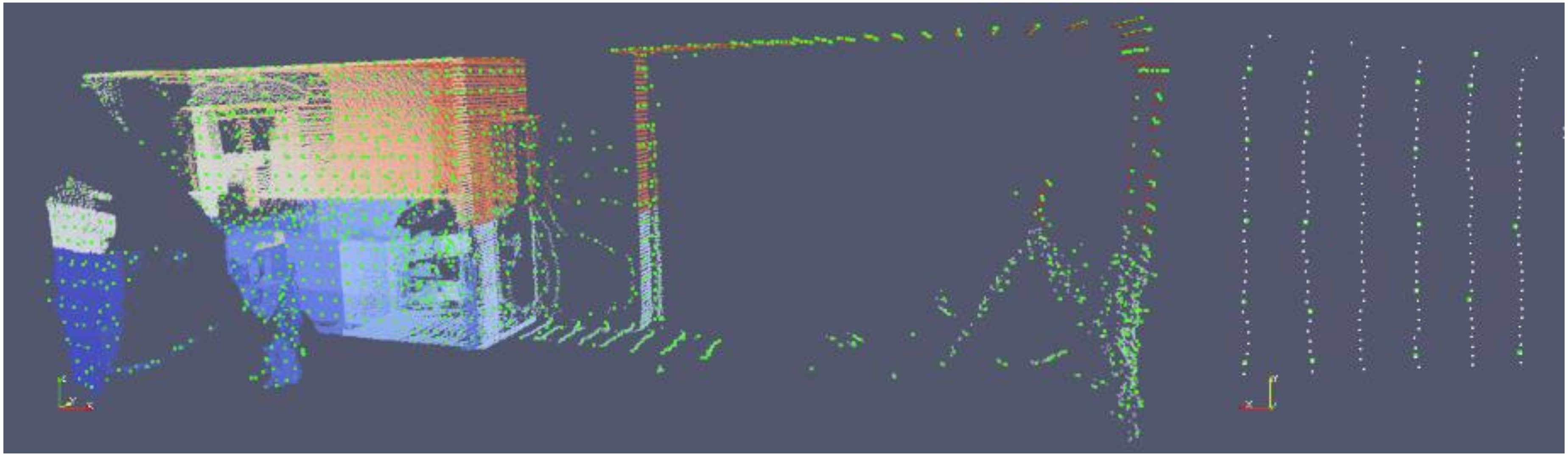
Courtesy: Wolfram Burgard

# Sampling

- Uniform sub-sampling
- Random sampling
- Feature-based sampling
- etc.

# Uniform Sampling

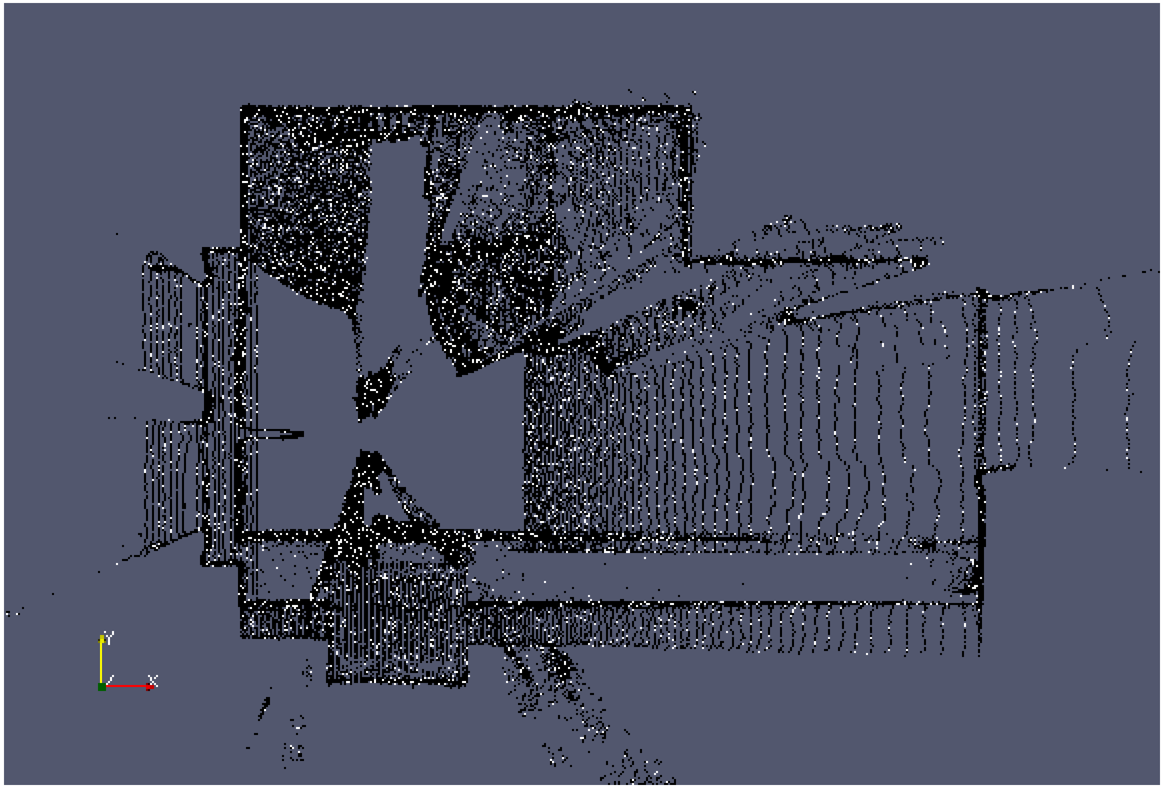
- Uniform sampling by Octree Grid
  - maxSizeByNode: 0.2 meter
  - green points are reserved after sampling



Courtesy: libpointmatcher

# Random Sampling

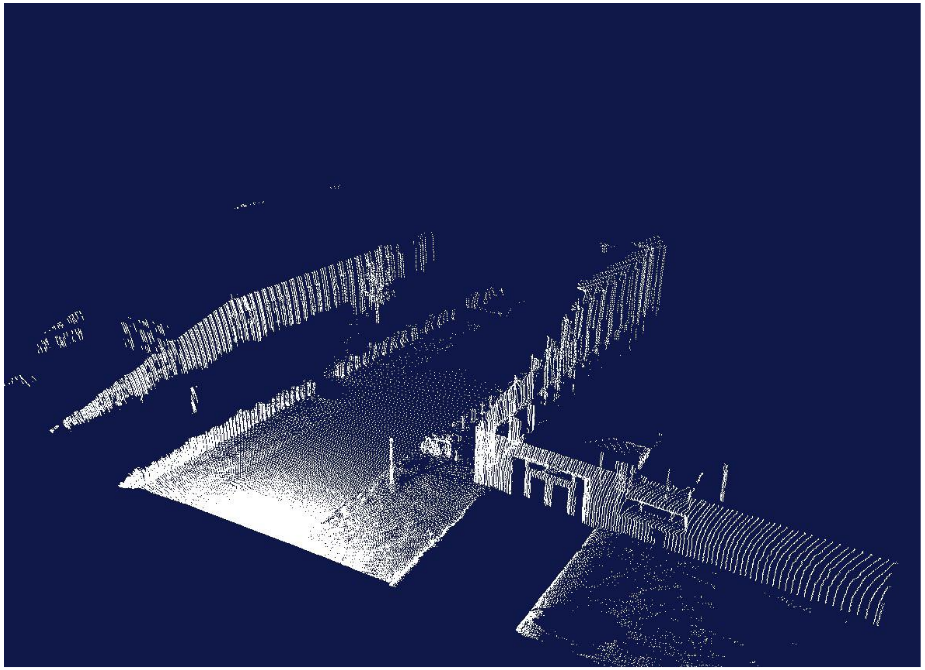
- After applying the random sampling filter
  - with a probability of 0.1.
  - white points are reserved after sampling

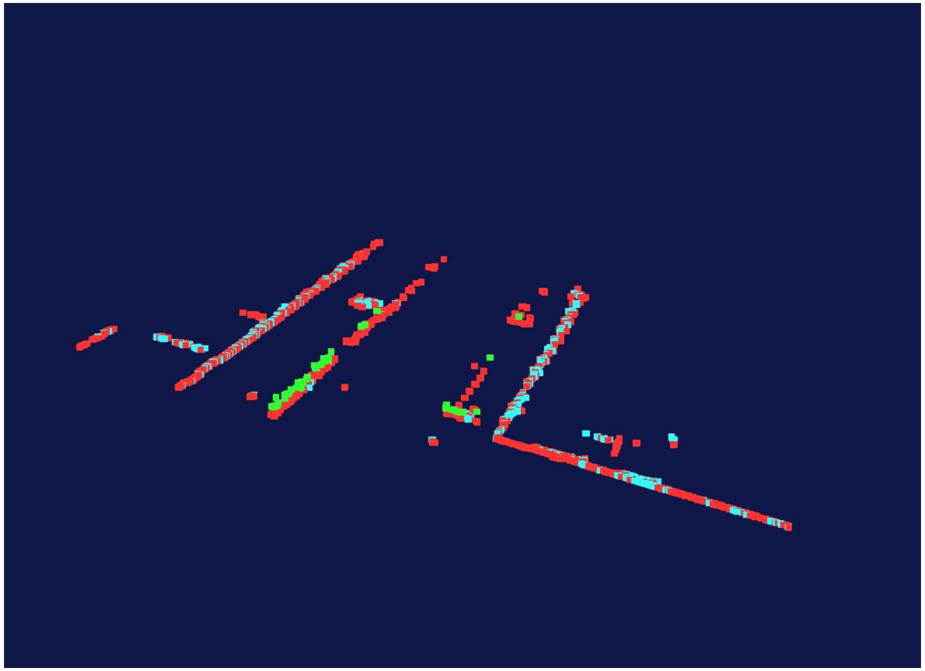



Courtesy: libpointmatcher

# Feature-based Sampling

- Try to find “important” points
  - Handcrafted or learning -based
  - From ~2000,000 to ~5,000





Courtesy: Wolfram Burgard

# ICP Variants

- Select point cloud subsets (Samping/ Filter) ☑
- Wighting the correspondences
- Data Associations
- Reject certain (outlier) point pairs

Courtesy: Wolfram Burgard

# Re-Weighting

- **Weight the corresponding pairs**
- Noise: Weighting based on sensor uncertainty
- Outlier: Assign **lower weights** for points with **higher point-point distances**
- Determine transformation that minimizes the weighted error function

Courtesy: Wolfram Burgard

# ICP Variants

- Select point cloud subsets (Samping/ Filter) ☑
- Wighting the correspondences ☑
- Data Associations
- Reject certain (outlier) point pairs

Courtesy: Wolfram Burgard

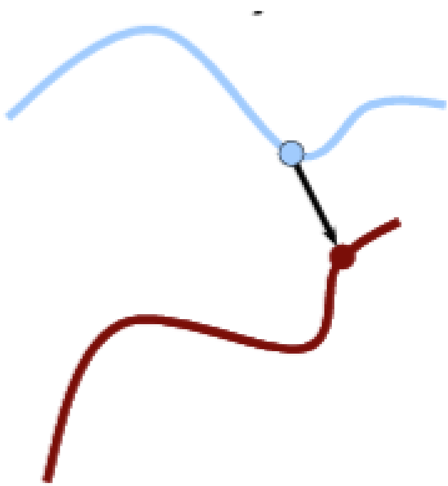
# Data Association

- Has greatest effect on convergence and speed
- Matching methods:
  - **Closest point**
  - **Point-to-plane**
  - Normal shooting
  - Closest compatible point
  - Projection-based approaches
  - etc.

Courtesy: Wolfram Burgard

# Closest Point

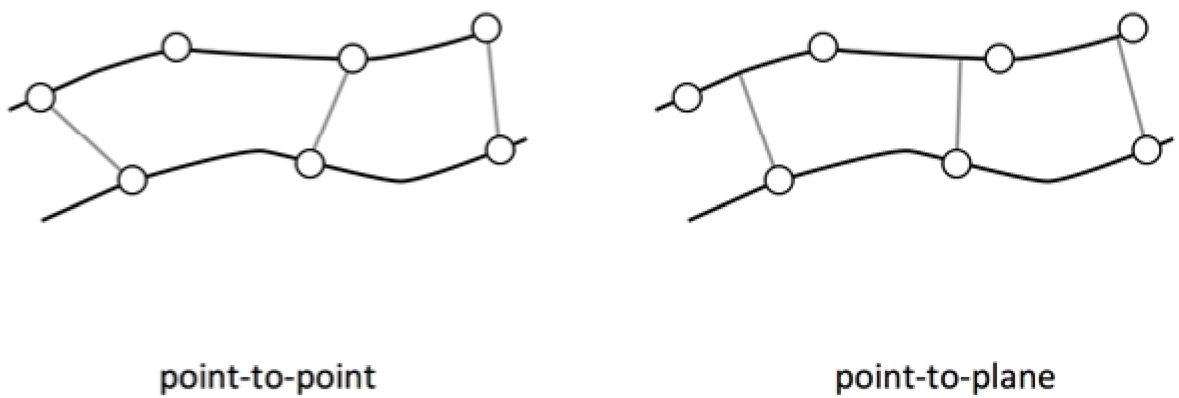
- Find closest point in other the point set (using kd-trees)
- Generally stable, but slow convergence and requires preprocessing



Courtesy: Wolfram Burgard

# Point-to-Plane

- Minimize the sum of the squared distances between a point and the **tangent plane** at its correspondence point
- Each iteration generally slower than the point-to-point version, however, often significantly better convergence rates




Courtesy: Wolfram Burgard

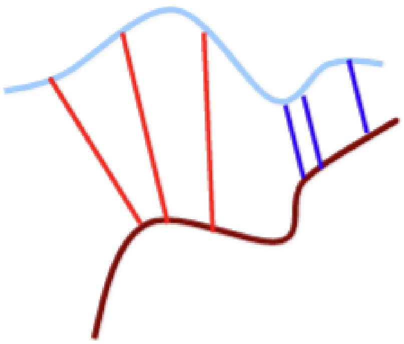
# ICP Variants

- Select point cloud subsets (Samping/ Filter) ☑
- Wighting the correspondences ☑
- Data Associations ☑
- Reject certain (outlier) point pairs

Courtesy: Wolfram Burgard

# Reject point pairs

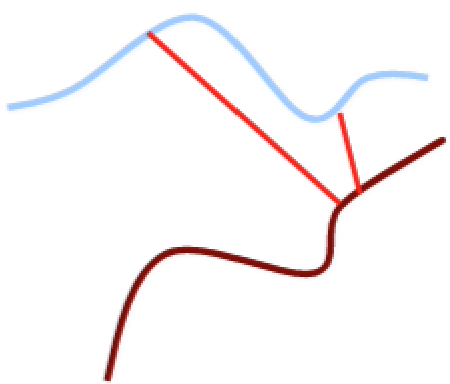
- Point-to-point distance larger than a given threshold
  - also works for point-to-plane



Courtesy: Wolfram Burgard

# Reject point pairs

- Point-to-point distance larger than a given threshold
  - also works for point-to-plane
- Rejection of pairs that are not consistent with their neighboring pairs



Courtesy: Wolfram Burgard

# Reject point pairs

- Point-to-point distance larger than a given threshold
  - also works for point-to-plane
- Rejection of pairs that are not consistent with their neighboring pairs
- Trimmed ICP: Sort correspondences w.r.t. their error, ignore the worst t%
  - t is related to overlap and outlier ratio
  - Knowledge about the overlap has to be estimated

Courtesy: Wolfram Burgard

# ICP Variants

- Select point cloud subsets (Samping/ Filter) ☑
- Wighting the correspondences ☑
- Data Associations ☑
- Reject certain (outlier) point pairs ☑

Courtesy: Wolfram Burgard

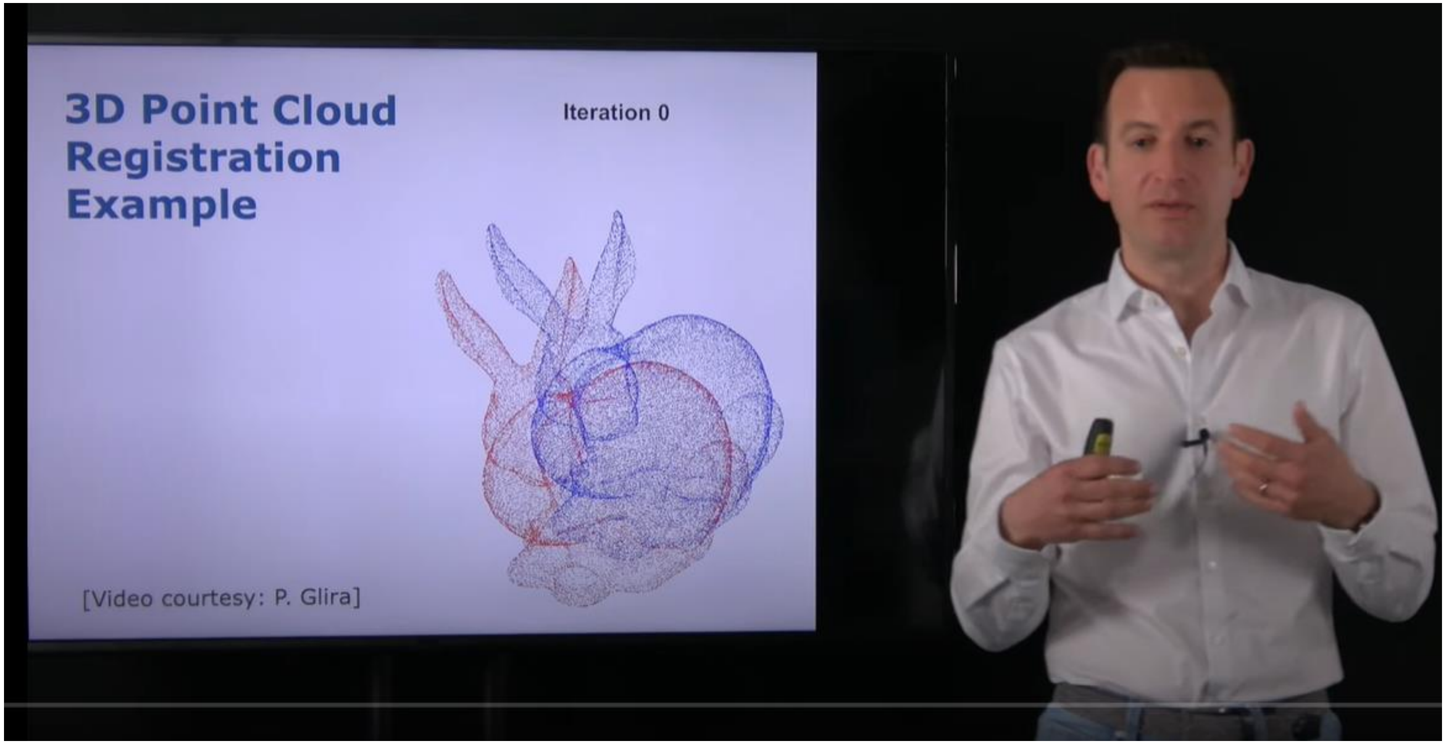
# ICP Algorithm

- Potentially subsample point clouds
- Determine corresponding points
- Potentially weight or reject pairs
- Compute rotation R, translation t (SVD)
- Apply R and t to all points of the set to be registered
- Compute the error E(R,t)
- While error decreased and error > threshold
  - Repeat to determine correspondences etc.
- Output final alignment

Courtesy: Wolfram Burgard

# Resources

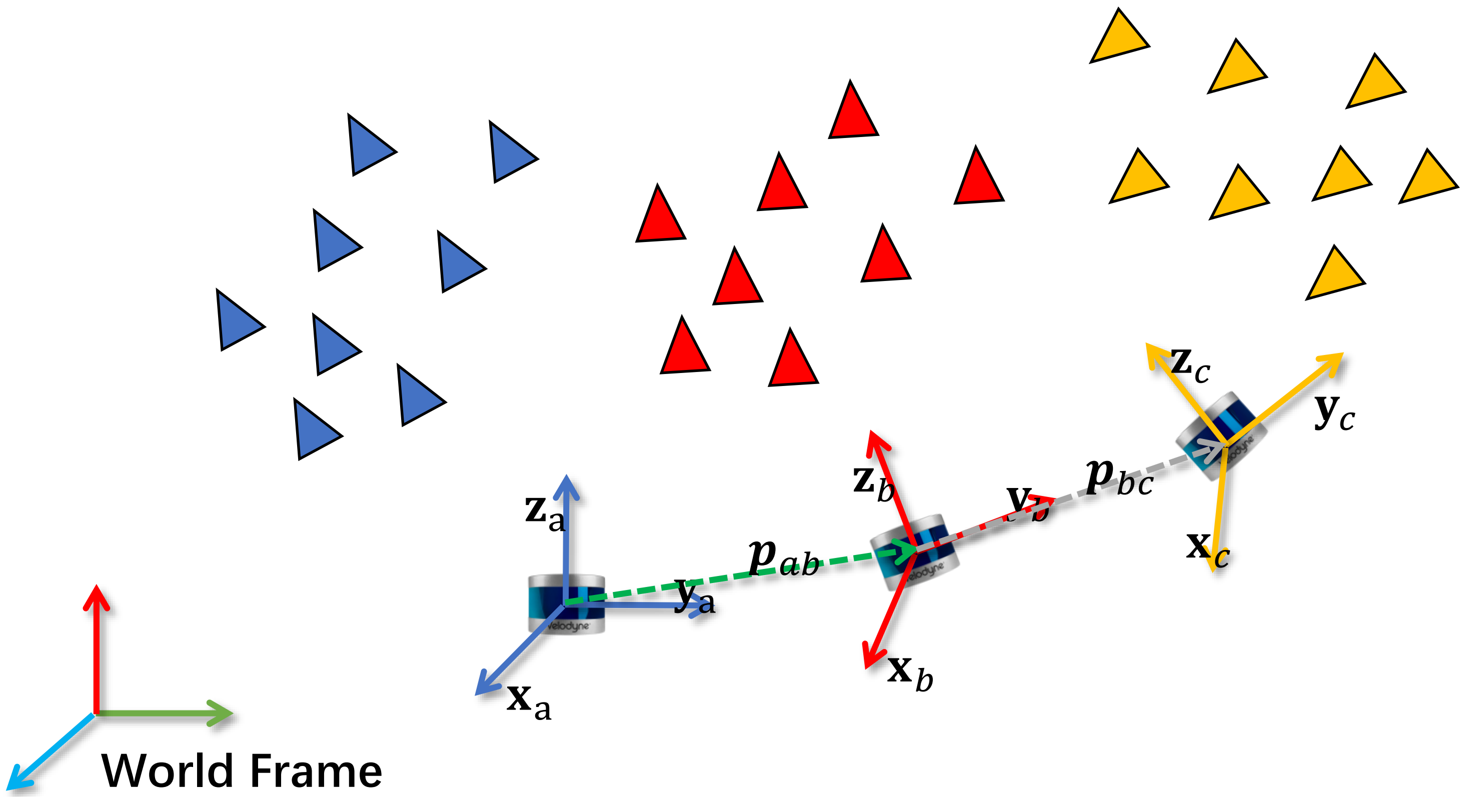
- ICP & Point Cloud Registration
  - Part 1 - Known Data Association & SVD
  - Part 2 - Unknown Data Association
  - Part 3 - Non-linear Least Squares




Courtesy: Cyrill Stachniss

# LiDAR odometry by ICP

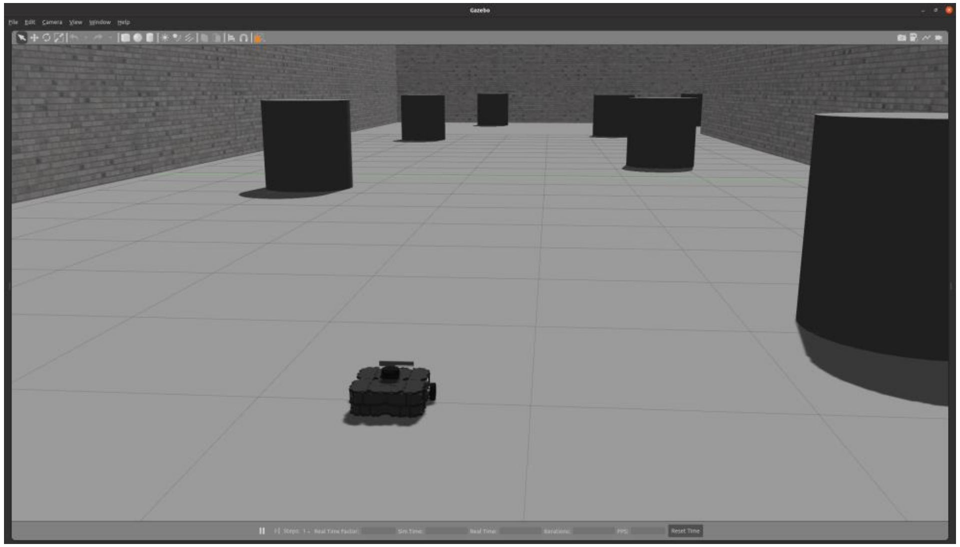
- Iterative Closest Points (ICP)
- A reduced LiDAR SLAM system without loop closure
  - simple but useful
  - only point cloud registration/alignment



# Project 1

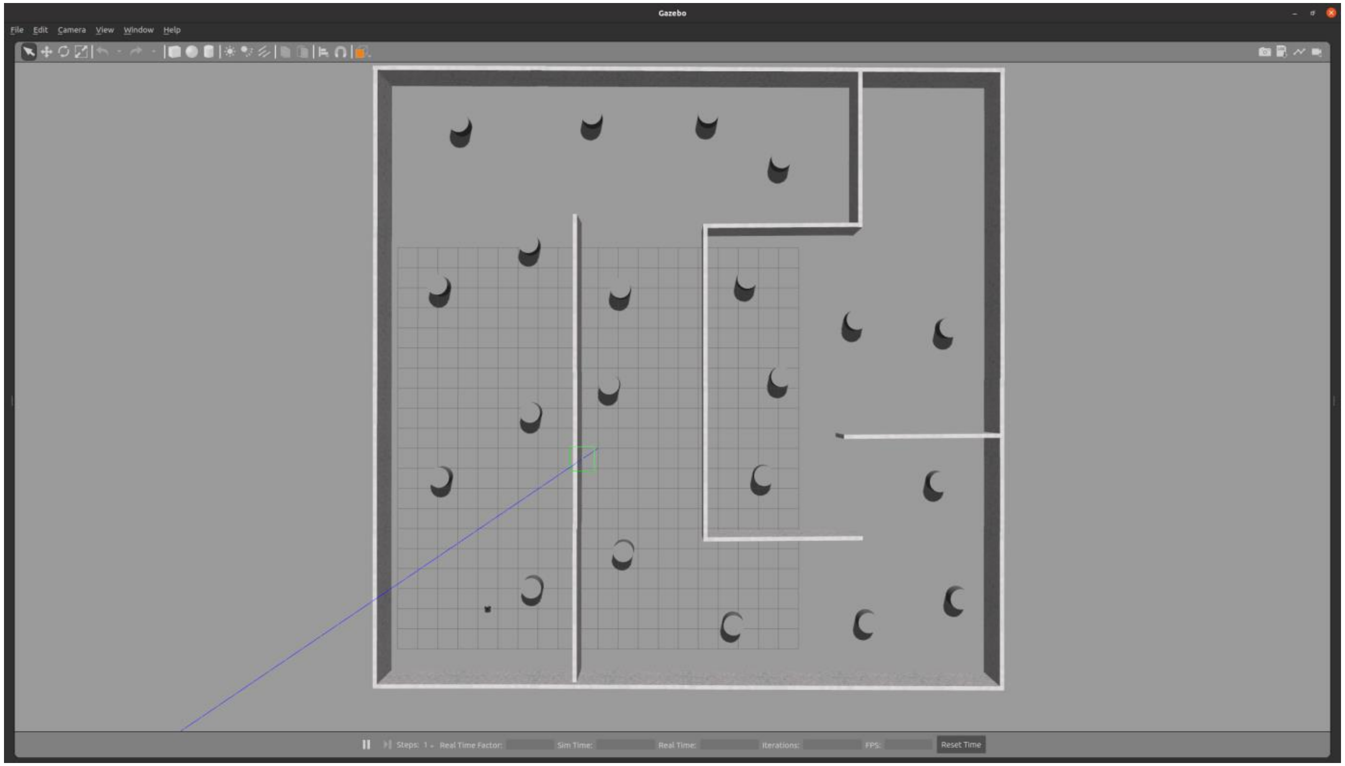
# Virtual Lab

- No lab time, On your PC
- A mobile robot with a LiDAR Scanner

# Virtual Lab

- on Gazebo, ROS
- Provide Rosbag for projects



# P1 -ICP Mapping

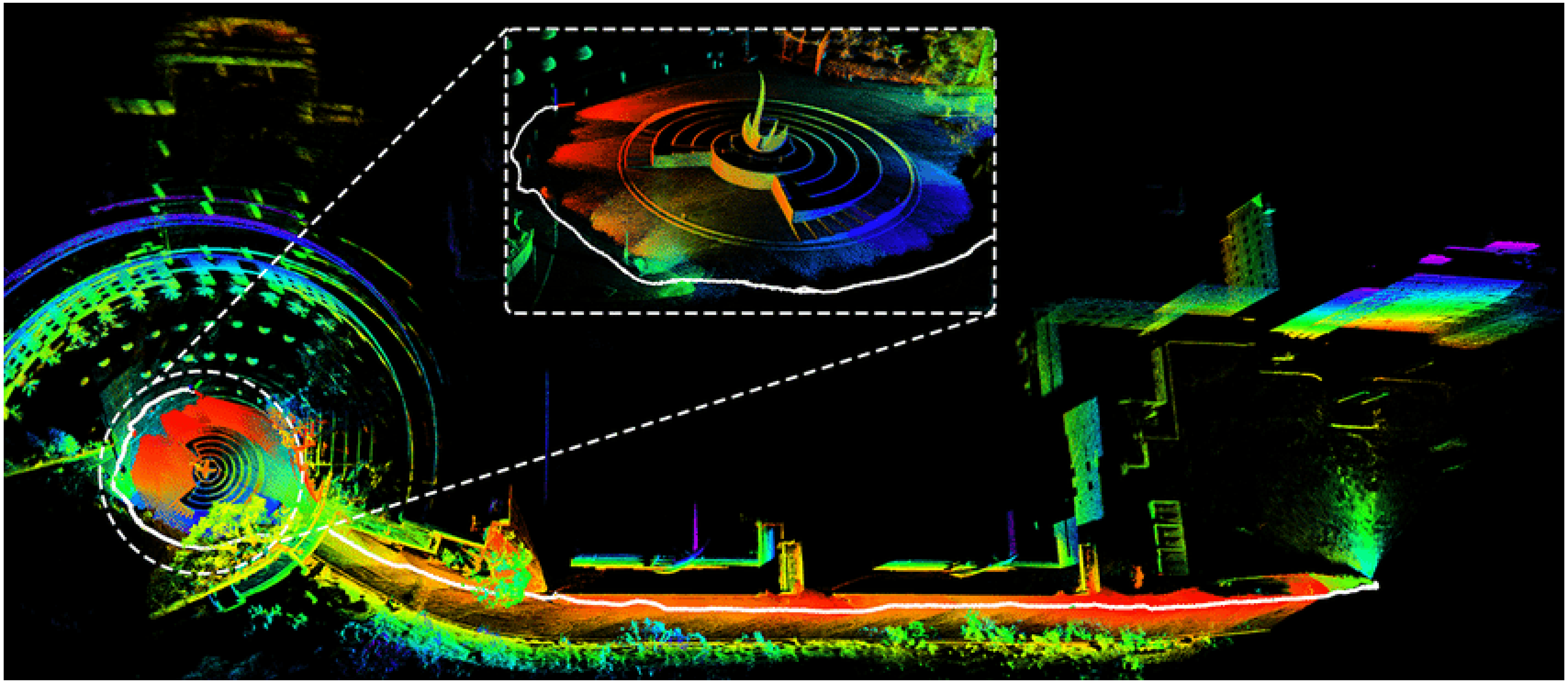
- LiDAR Odometry and Mapping by Iterative Closest Point (ICP)

# Summary

- 3D pose estimation with known correspondences
  - Rotation and translation
  - SVD
- Unknown correspondences
  - Iterative closest search
- ICP and its variants

# Problem

- What if the map is too large?
- Other map representations beyond point clouds?



**The large scale mapping of the HKUST campus**

Courtesy: Jiarong Lin and Fu Zhang

# Next Lecture

- Map Representations
- Robot Operating System (ROS)