Scan Matching

Pieter Abbeel
UC Berkeley EECS

Many slides adapted from Thrun, Burgard and Fox, Probabilistic Robotics

Scan Matching Overview

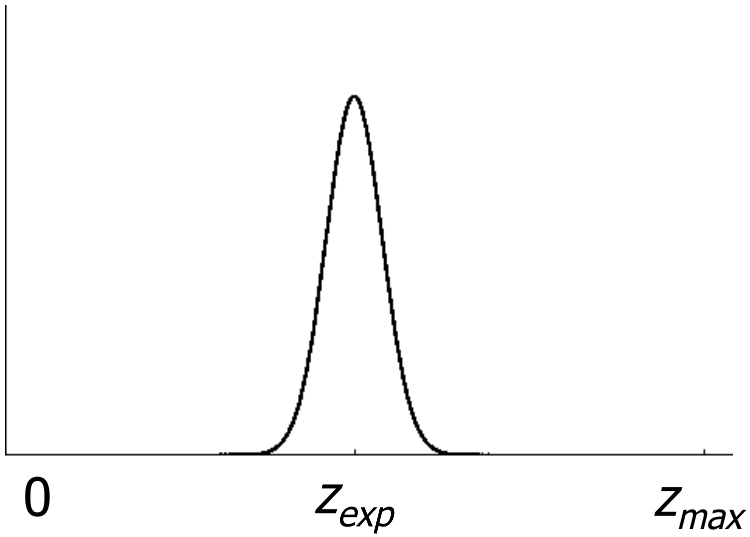
- Problem statement:
  - Given a scan and a map, or a scan and a scan, or a map and a map, find the rigid-body transformation (translation+rotation) that aligns them best
- Benefits:
  - Improved proposal distribution (e.g., gMapping)
  - Scan-matching objectives, even when not meaningful probabilities, can be used in graphSLAM / pose-graph SLAM
- Approaches:
  - Optimize over x: p(z | x, m), with:
    - 1. p(z | x, m) = beam sensor model --- sensor beam full readings <-> map
    - 2. p(z | x, m) = likelihood field model --- sensor beam endpoints <-> likelihood field
    - 3. $p(m_{local} \mid x, m)$ = map matching model --- local map <-> global map
  - Reduce both entities to a set of points, align the point clouds through the Iterative Closest Points (ICP)
    - 4. cloud of points <-> cloud of points --- sensor beam endpoints <-> sensor beam endpoints
- Other popular use (outside of SLAM): pose estimation and verification of presence for objects detected in point cloud data

Outline

- **1. Beam Sensor Model**
- 2. Likelihood Field Model
- 3. Map Matching
- 4. Iterated Closest Points (ICP)

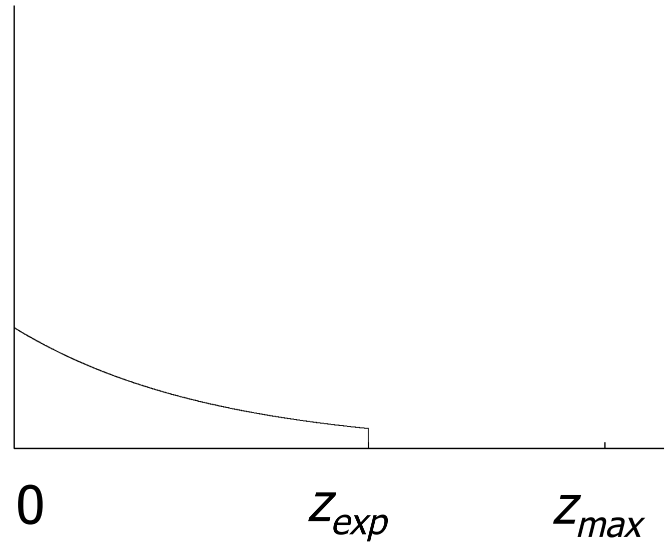
Beam-based Proximity Model

Measurement noise



$$P_{hit}(z \mid x, m) = \eta \frac{1}{\sqrt{2\pi b}} e^{-\frac{1}{2}\frac{(z - z_{\exp})^2}{b}}$$

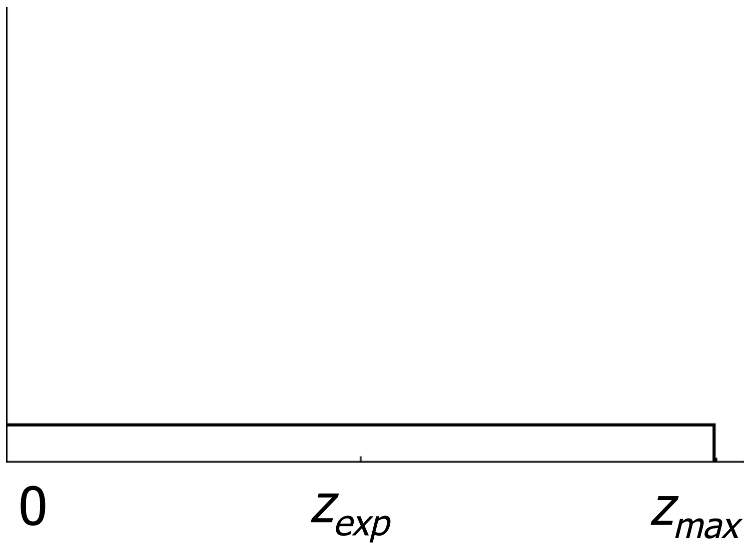
Unexpected obstacles



$$P_{\text{unexp}}(z \mid x, m) = \begin{cases} \eta\, \lambda\, e^{-\lambda z} & z < z_{\exp} \\ 0 & otherwise \end{cases}$$

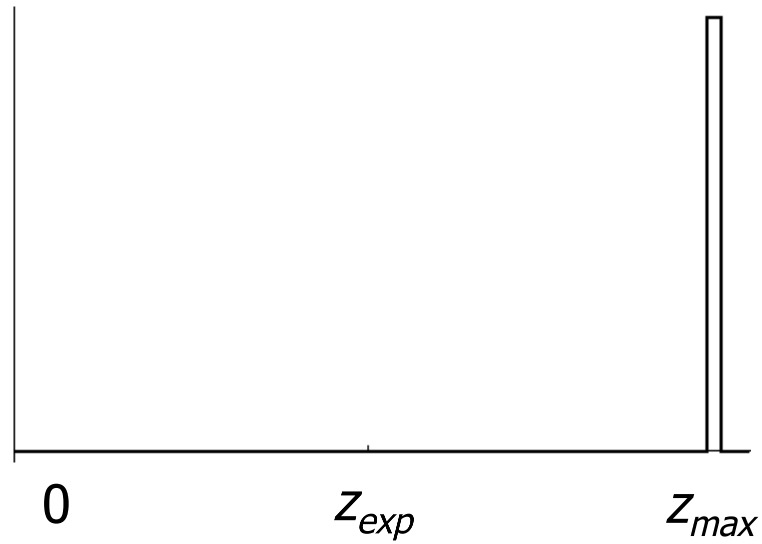
Beam-based Proximity Model

Random measurement



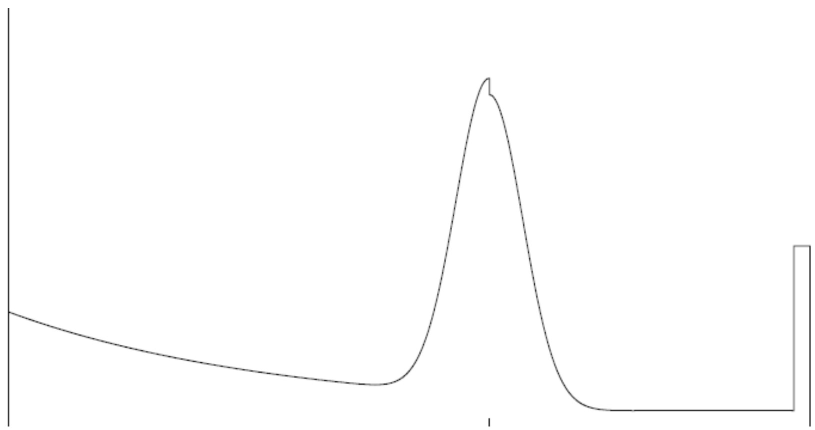
$$P_{rand}(z \mid x, m) = \eta \frac{1}{z_{\max}}$$

Max range



$$P_{\max}(z \mid x, m) = \eta \frac{1}{z_{small}}$$

Resulting Mixture Density

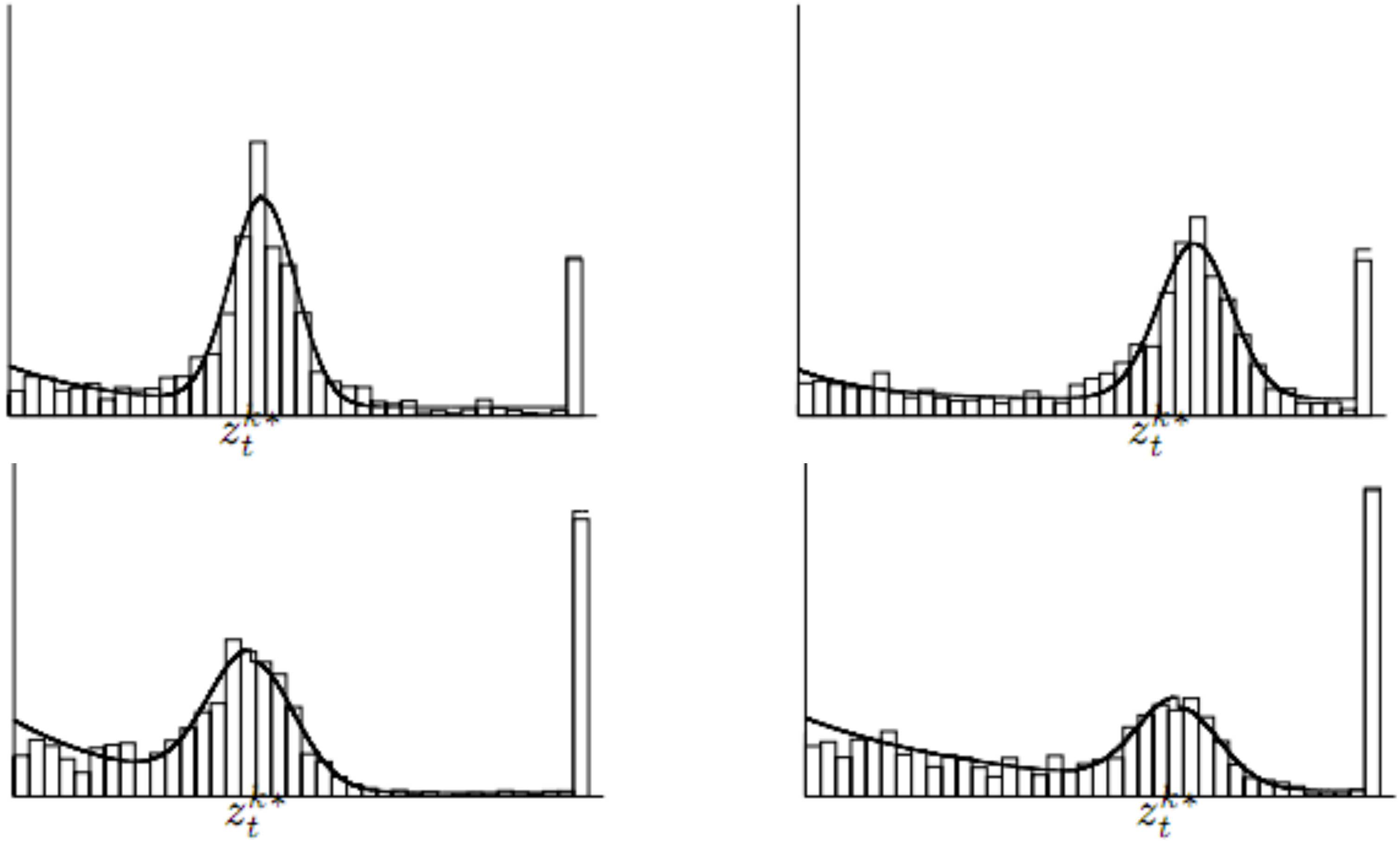


$$P(z \mid x, m) = \begin{pmatrix} \alpha_{\text{hit}} \\ \alpha_{\text{unexp}} \\ \alpha_{\max} \\ \alpha_{\text{rand}} \end{pmatrix}^T \cdot \begin{pmatrix} P_{\text{hit}}(z \mid x, m) \\ P_{\text{unexp}}(z \mid x, m) \\ P_{\max}(z \mid x, m) \\ P_{\text{rand}}(z \mid x, m) \end{pmatrix}$$

How can we determine the model parameters?

Approximation Results

Laser



300cm 400cm

Sonar

Summary Beam Sensor Model

- Assumes independence between beams.
  - Justification?
  - Overconfident!
- Models physical causes for measurements.
  - Mixture of densities for these causes.
  - Assumes independence between causes. Problem?
- Implementation
  - Learn parameters based on real data.
  - Different models should be learned for different angles at which the sensor beam hits the obstacle.
  - Determine expected distances by ray-tracing.
  - Expected distances can be pre-processed.

Drawbacks Beam Sensor Model

- Lack of smoothness
  - $P(z \mid x_t, m)$ is not smooth in x_t
  - Problematic consequences:
    - For sampling based methods: nearby points have very different likelihoods, which could result in requiring large numbers of samples to hit some "reasonably likely" states
    - Hill-climbing methods that try to find the locally most likely x_t have limited abilities per many local optima
- Computationally expensive
  - Need to ray-cast for every sensor reading
  - Could pre-compute over discrete set of states (and then interpolate), but table is large per covering a 3-D space and in SLAM the map (and hence table) change over time

Outline

- 1. Beam Sensor Model
- **2. Likelihood Field Model**
- 3. Map Matching
- 4. Iterated Closest Points (ICP)

Likelihood Field Model
aka Beam Endpoint Model aka Scan-based Model

- Overcomes lack-of-smoothness and computational limitations of Sensor Beam Model
- Ad-hoc algorithm: not considering a conditional probability relative to any meaningful generative model of the physics of sensors
- Works well in practice.
- Idea: Instead of following along the beam (which is expensive!) just check the end-point. The likelihood $p(z \mid x_t, m)$ is given by:

$$\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{d^2}{2\sigma^2}\right)$$

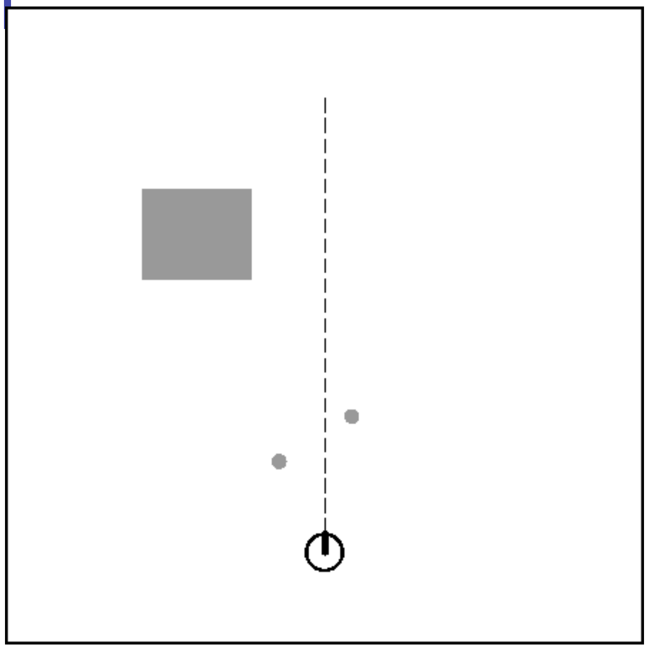
with d = distance from end-point to nearest obstacle.

Algorithm: likelihood_field_range_finder_model(z_t, x_t, m)

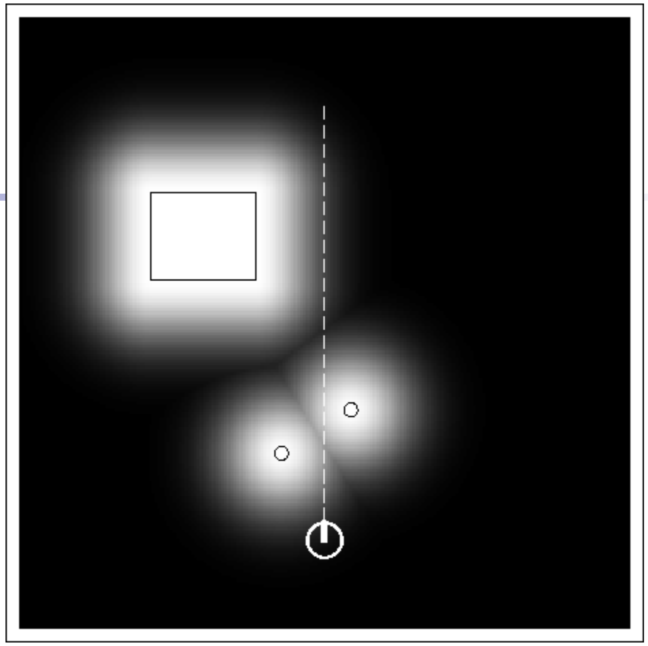
1. $q = 1$
2. for all k do
3. $\quad$ if $z_t^k \neq z_{\max}$
4. $\quad\quad x_{z_t^k} = x + x_{k,\text{sens}} \cos\theta - y_{k,\text{sens}} \sin\theta + z_t^k \cos(\theta + \theta_{k,\text{sens}})$
5. $\quad\quad y_{z_t^k} = y + y_{k,\text{sens}} \cos\theta - x_{k,\text{sens}} \sin\theta + z_t^k \sin(\theta + \theta_{k,\text{sens}})$
6. $\quad\quad d = \min_{x',y'} \{(x_{z_t^k} - x')^2 + (y_{z_t^k} - y')^2 | (x', y') \text{is ocupied in } m\}$
7. $\quad\quad q = q \cdot (p_{\text{hit}} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{d^2}{2\sigma^2}) + p_{\text{random}} \frac{1}{z_{\max}})$
8. return q

In practice: pre-compute “likelihood field” over (2-D) grid.

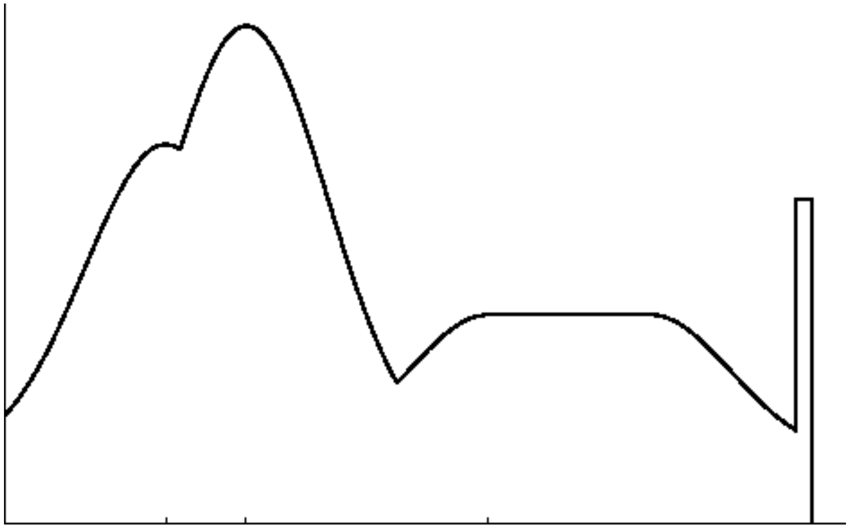
Example



Map *m*



Likelihood field

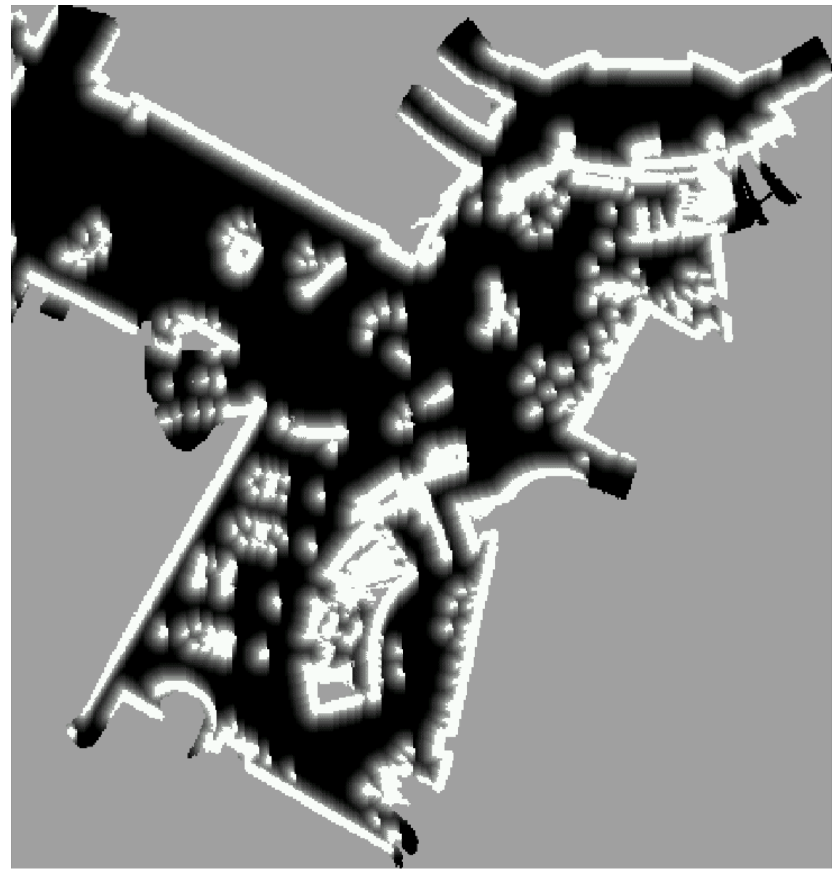


P(z|x,m)

Note: "p(z|x,m)" is not really a density, as it does not normalize to one when integrating over all z

San Jose Tech Museum

Occupancy grid map



Likelihood field

Drawbacks of Likelihood Field Model

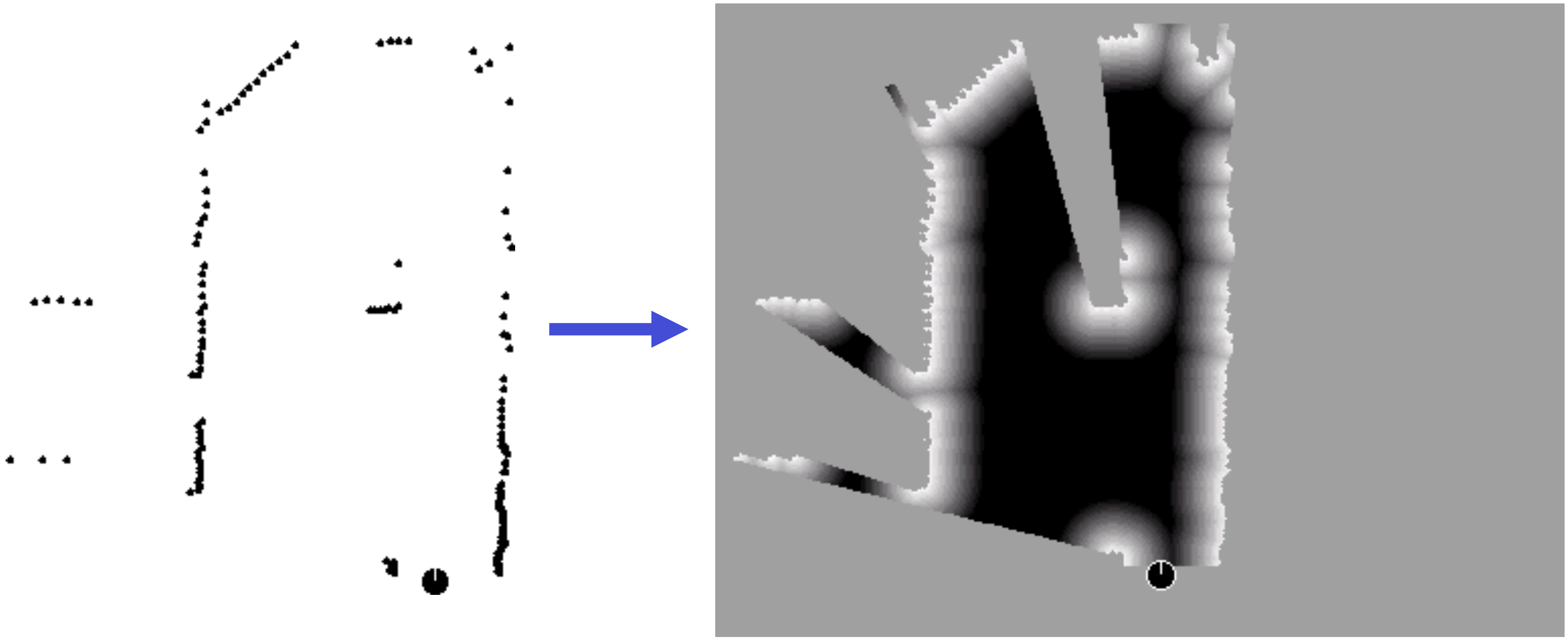
- No explicit modeling of people and other dynamics that might cause short readings
- No modeling of the beam --- treats sensor as if it can see through walls
- Cannot handle unexplored areas
  - Fix: when endpoint in unexplored area, have $p(z_t \mid x_t, m) = 1 / z_{max}$

Scan Matching

- As usual, maximize over x_t the likelihood $p(z_t \mid x_t, m)$
- The objective $p(z_t \mid x_t, m)$ now corresponds to the likelihood field based score

Scan Matching

- Can also match two scans: for first scan extract likelihood field (treating each beam endpoint as occupied space) and use it to match the next scan. [can also symmetrize this]



Properties of Likelihood Field based Scan Matching

- Highly efficient, uses 2D tables only.
- Smooth w.r.t. to small changes in robot position.
- Allows gradient descent, scan matching.
- Ignores physical properties of beams.

Outline

- 1. Beam Sensor Model
- 2. Likelihood Field Model
- **3. Map Matching**
- 4. Iterated Closest Points (ICP)

Map Matching

- Generate small, local maps from sensor data and match local maps against global model.

- Correlation score:

$$\rho_{m,m_{\text{local}},x_t} = \frac{\sum_{x,y}(m_{x,y} - \bar{m}) \cdot (m_{x,y,\text{local}}(x_t) - \bar{m})}{\sqrt{\sum_{x,y}(m_{x,y} - \bar{m})^2}\sqrt{\sum_{x,y}(m_{x,y,\text{local}}(x_t) - \bar{m})^2}}$$

with $\bar{m} = \frac{1}{2N}\sum_{x,y}(m_{x,y} + m_{x,y,\text{local}})$

- Likelihood interpretation:

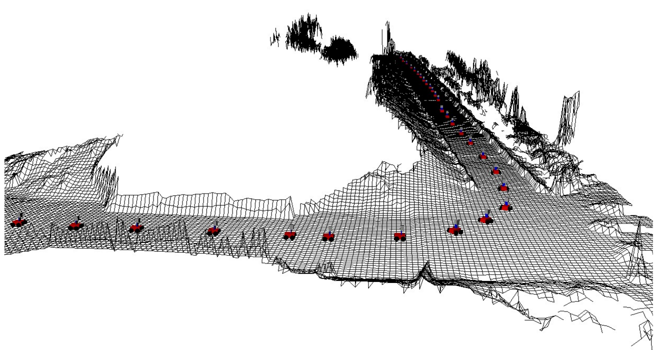
$$p(m_{\text{local}}|x_t, m) = \max\{\rho_{m,m_{\text{local}},x_t}, 0\}$$

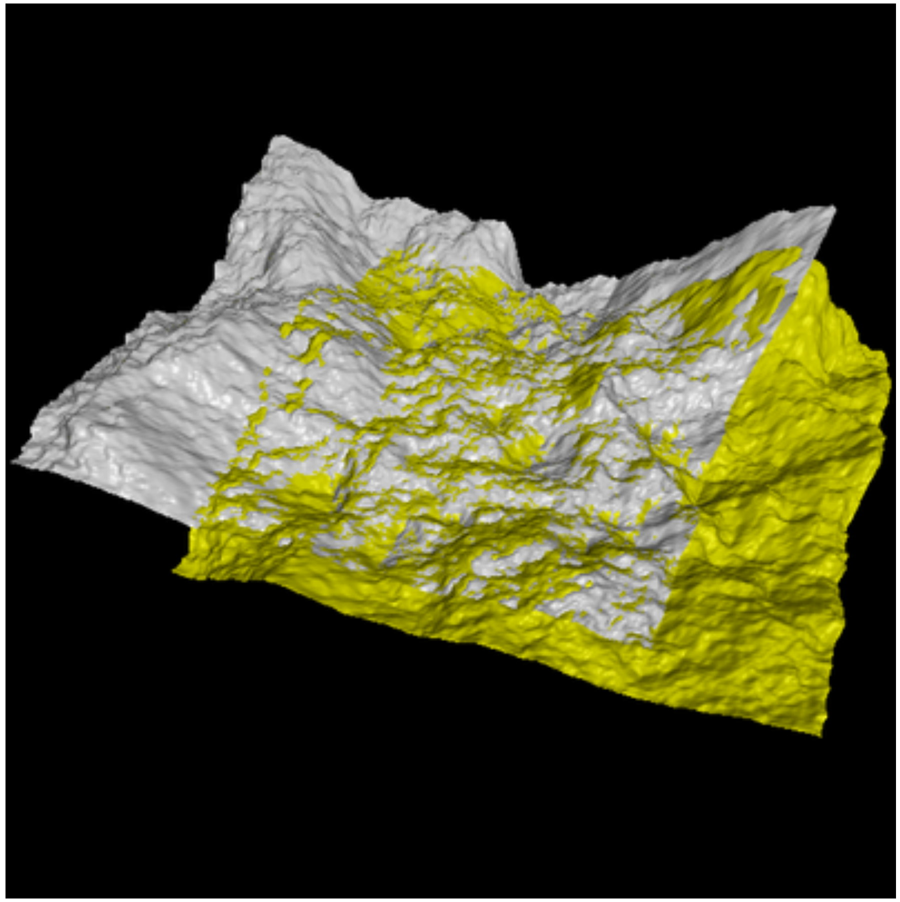
- To obtain smoothness: convolve the map m with a Gaussian, and run map matching on the smoothed map

Outline

- 1. Beam Sensor Model
- 2. Likelihood Field Model
- 3. Map Matching
- **4. Iterated Closest Points (ICP)**

Motivation





Known Correspondences

- Given: two corresponding point sets:

$$X = \{x_1, ..., x_n\}$$

$$P = \{p_1, ..., p_n\}$$

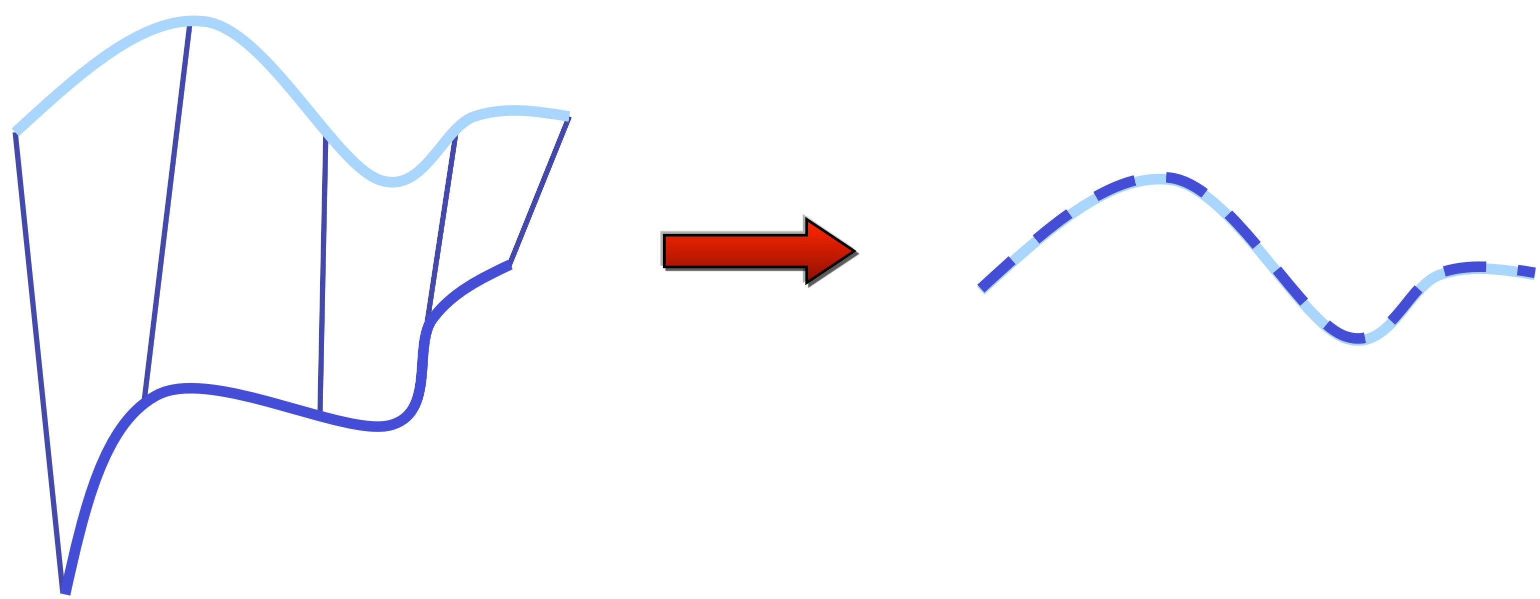
- Wanted: translation t and rotation R that minimizes the sum of the squared error:

$$E(R, t) = \frac{1}{N_p} \sum_{i=1}^{N_p} ||x_i - Rp_i - t||^2$$

Where x_i and p_i are corresponding points.

Key Idea

- If the correct correspondences are known, the correct relative rotation/translation can be calculated in closed form.



Center of Mass

$$\mu_x = \frac{1}{N_x}\sum_{i=1}^{N_x} x_i \quad \text{and} \quad \mu_p = \frac{1}{N_p}\sum_{i=1}^{N_p} p_i$$

are the centers of mass of the two point sets.

Idea:

- Subtract the corresponding center of mass from every point in the two point sets before calculating the transformation.
- The resulting point sets are:

$$X' = \{x_i - \mu_x\} = \{x'_i\} \quad \text{and}$$
$$P' = \{p_i - \mu_p\} = \{p'_i\}$$

SVD

Let $W = \sum_{i=1}^{N_p} x_i' p_i'^T$

denote the singular value decomposition (SVD) of W by:

$$W = U \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} V^T$$

where $U, V \in \mathbb{R}^{3\times 3}$ are unitary, and

$\sigma_1 \geq \sigma_2 \geq \sigma_3$ are the singular values of W.

SVD

Theorem (without proof):

If rank(W) = 3, the optimal solution of E(R,t) is unique and is given by:

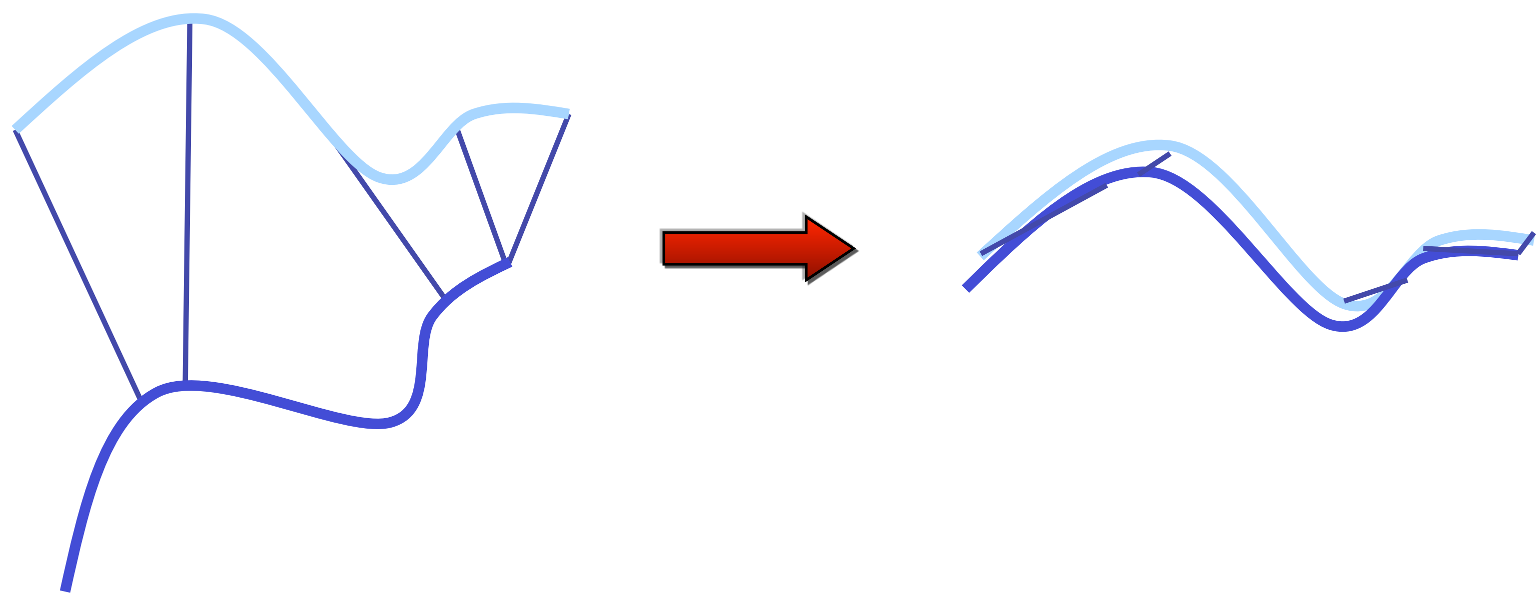
$$R = UV^T$$
$$t = \mu_x - R\mu_p$$

The minimal value of error function at (R,t) is:

$$E(R,t) = \sum_{i=1}^{N_p} (||x_i'||^2 + ||y_i'||^2) - 2(\sigma_1 + \sigma_2 + \sigma_3)$$

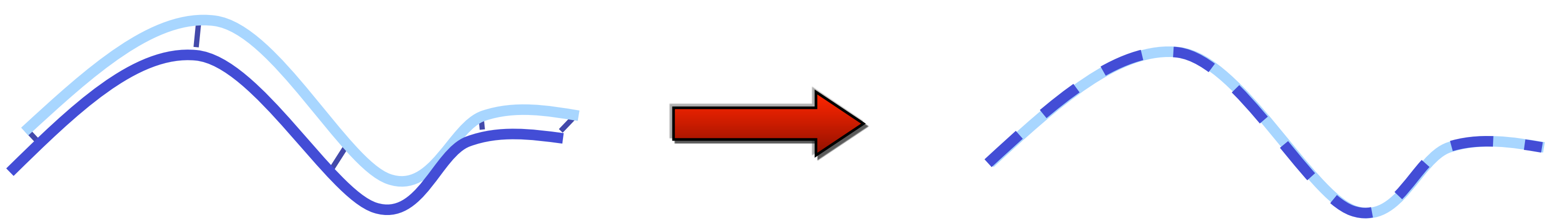
Unknown Data Association

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



ICP-Algorithm

- Idea: iterate to find alignment
- Iterated Closest Points (ICP) [Besl & McKay 92]
- Converges if starting positions are "close enough"



ICP-Variants

- Variants on the following stages of ICP have been proposed:

1. Point subsets (from one or both point sets)
2. Weighting the correspondences
3. Data association
4. Rejecting certain (outlier) point pairs

Performance of Variants

- Various aspects of performance:
  - Speed
  - Stability (local minima)
  - Tolerance wrt. noise and/or outliers
  - Basin of convergence
    (maximum initial misalignment)
- Here: properties of these variants

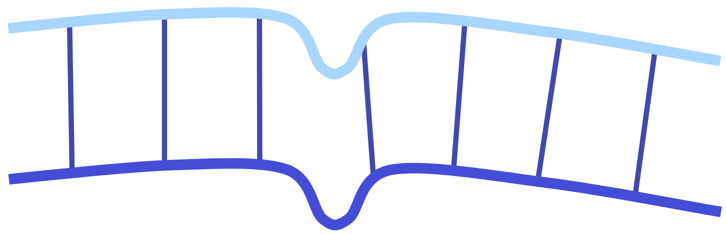
ICP Variants

1. Point subsets (from one or both point sets)
2. Weighting the correspondences
3. Data association
4. Rejecting certain (outlier) point pairs

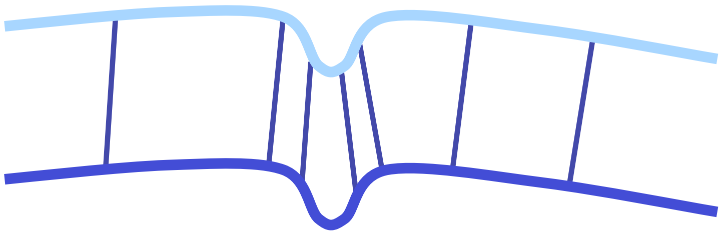
Selecting Source Points

- Use all points
- Uniform sub-sampling
- Random sampling
- Feature based Sampling
- Normal-space sampling
  - Ensure that samples have normals distributed as uniformly as possible

Normal-Space Sampling



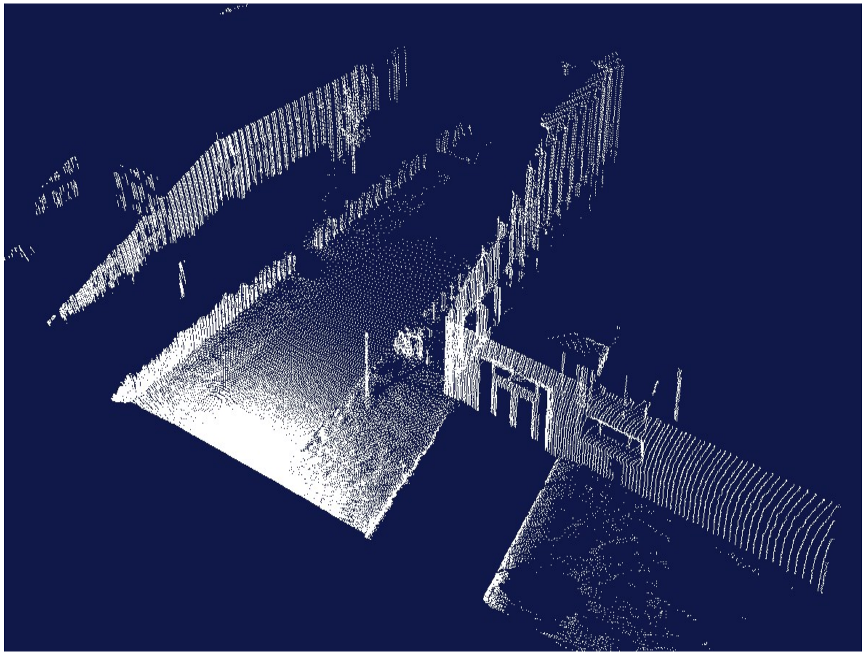
uniform sampling



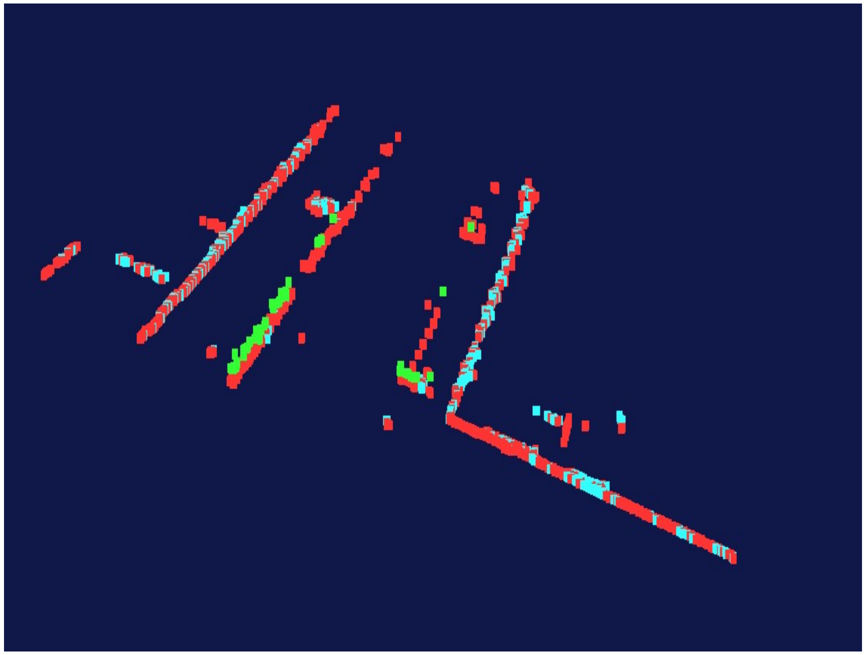
normal-space sampling

Feature-Based Sampling

- try to find "important" points
- decrease the number of correspondences
- higher efficiency and higher accuracy
- requires preprocessing

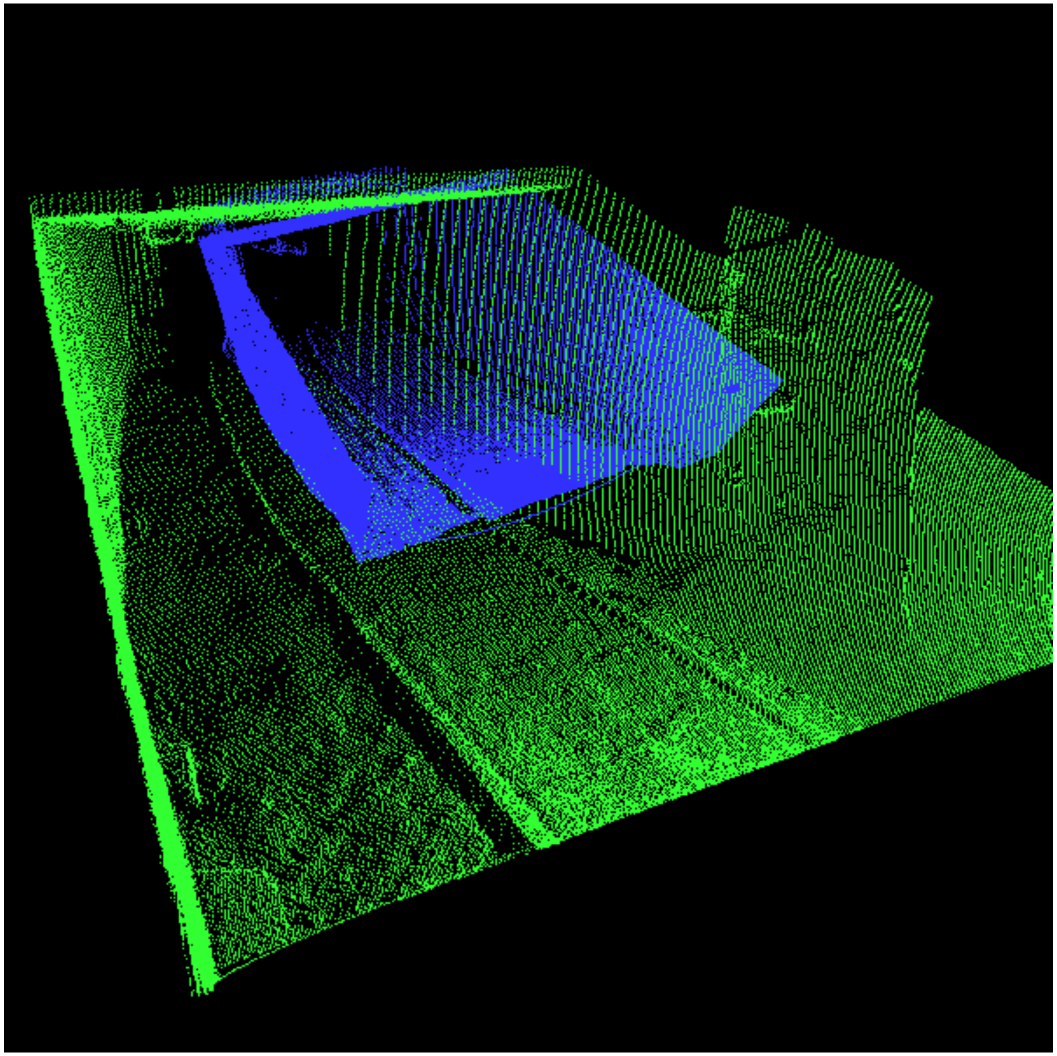


3D Scan (~200.000 Points)



Extracted Features (~5.000 Points)

Application



[Nuechter et al., 04]

ICP Variants

1. Point subsets (from one or both point sets)
2. Weighting the correspondences
3. Data association
4. Rejecting certain (outlier) point pairs

Selection vs. Weighting

- Could achieve same effect with weighting
- Hard to guarantee that enough samples of important features except at high sampling rates
- Weighting strategies turned out to be dependent on the data.
- Preprocessing / run-time cost tradeoff (how to find the correct weights?)

ICP Variants

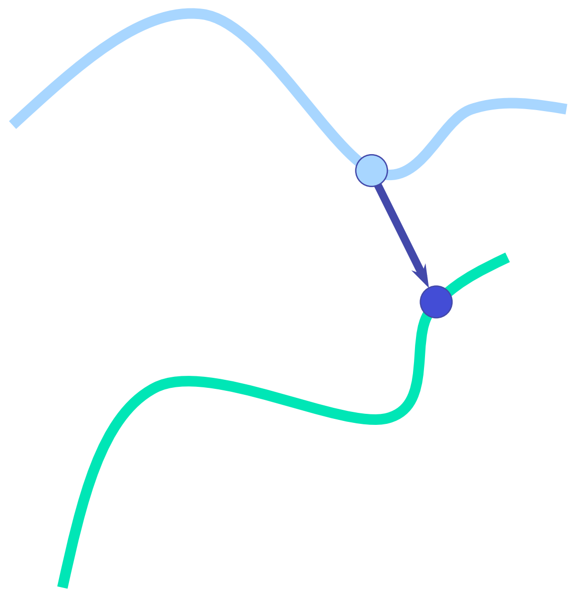
1. Point subsets (from one or both point sets)
2. Weighting the correspondences
3. Data association
4. Rejecting certain (outlier) point pairs

Data Association

- has greatest effect on convergence and speed
- Closest point
- Normal shooting
- Closest compatible point
- Projection
- Using kd-trees or oc-trees

Closest-Point Matching

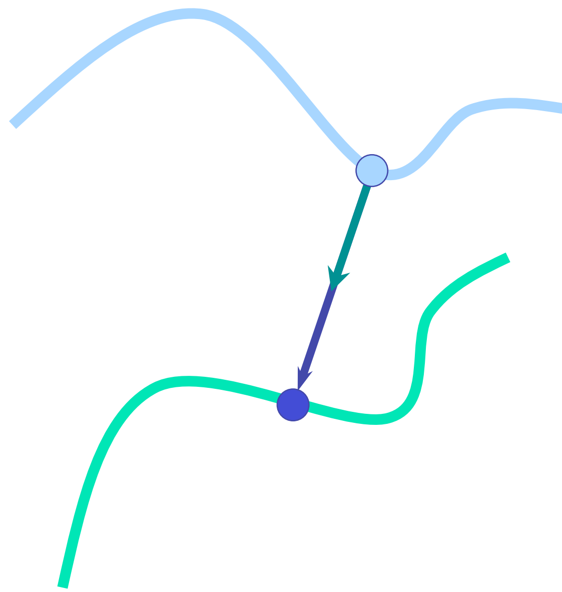
- Find closest point in other the point set



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

Normal Shooting

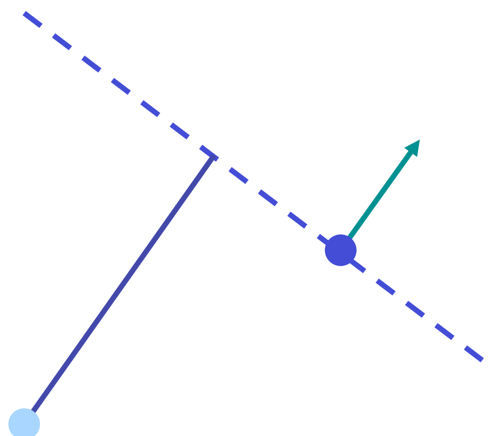
- Project along normal, intersect other point set

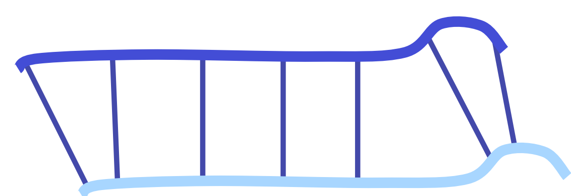


Slightly better than closest point for smooth structures, worse for noisy or complex structures

Point-to-Plane Error Metric

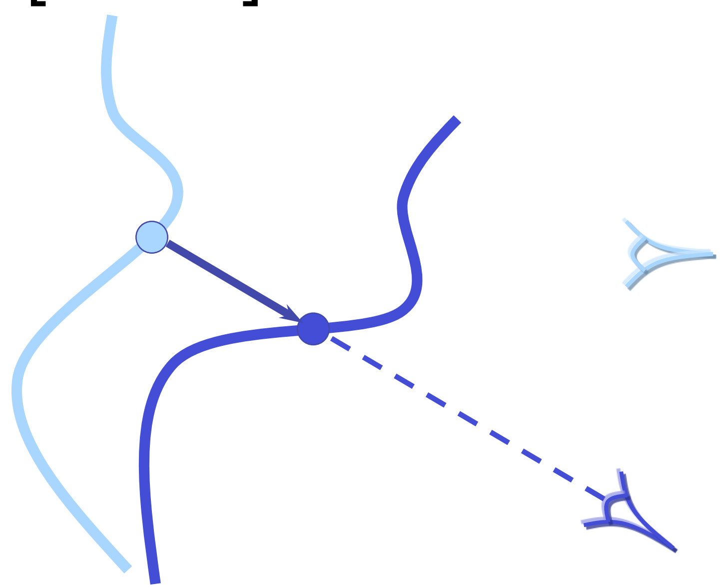
- Using point-to-plane distance instead of point-to-point lets flat regions slide along each other [Chen & Medioni 91]





Projection

- Finding the closest point is the most expensive stage of the ICP algorithm
- Idea: simplified nearest neighbor search
- For range images, one can project the points according to the view-point [Blais 95]



Projection-Based Matching

- Slightly worse alignments per iteration
- Each iteration is one to two orders of magnitude faster than closest-point
- Requires point-to-plane error metric

Closest Compatible Point

- Improves the previous two variants by considering the compatibility of the points
- Compatibility can be based on normals, colors, etc.
- In the limit, degenerates to feature matching

ICP Variants

1. Point subsets (from one or both point sets)
2. Weighting the correspondences
3. Nearest neighbor search
4. Rejecting certain (outlier) point pairs

Rejecting (outlier) point pairs

- sorting all correspondences with respect to there error and deleting the worst t%, Trimmed ICP (TrICP) [Chetverikov et al. 2002]
- t is to Estimate with respect to the Overlap

➡ Problem: Knowledge about the overlap is necessary or has to be estimated

ICP-Summary

- ICP is a powerful algorithm for calculating the displacement between scans.
- The major problem is to determine the correct data associations.
- Given the correct data associations, the transformation can be computed efficiently using SVD.