Introduction to

Simultaneous Localization and Mapping

Spring Semester 2019

CS284

Course Convener:

Prof Laurent Kneip

lkneip@ shanghaitech.edu.cn

Disclaimer

- Some of the lecturing material is naturally taken from publically available online material from other groups. Sources include:
  - Autonomous Systems Lab ETH Zurich
  - Research School of Engineering, ANU
  - Informatik Department, Uni Freiburg
  - MIT
  - …
- By using the present material you confirm that
  - You use it only internally and for the purpose of education
  - You are aware that the lecture material may originate from other sources, even if explicit reference is occasionally missing

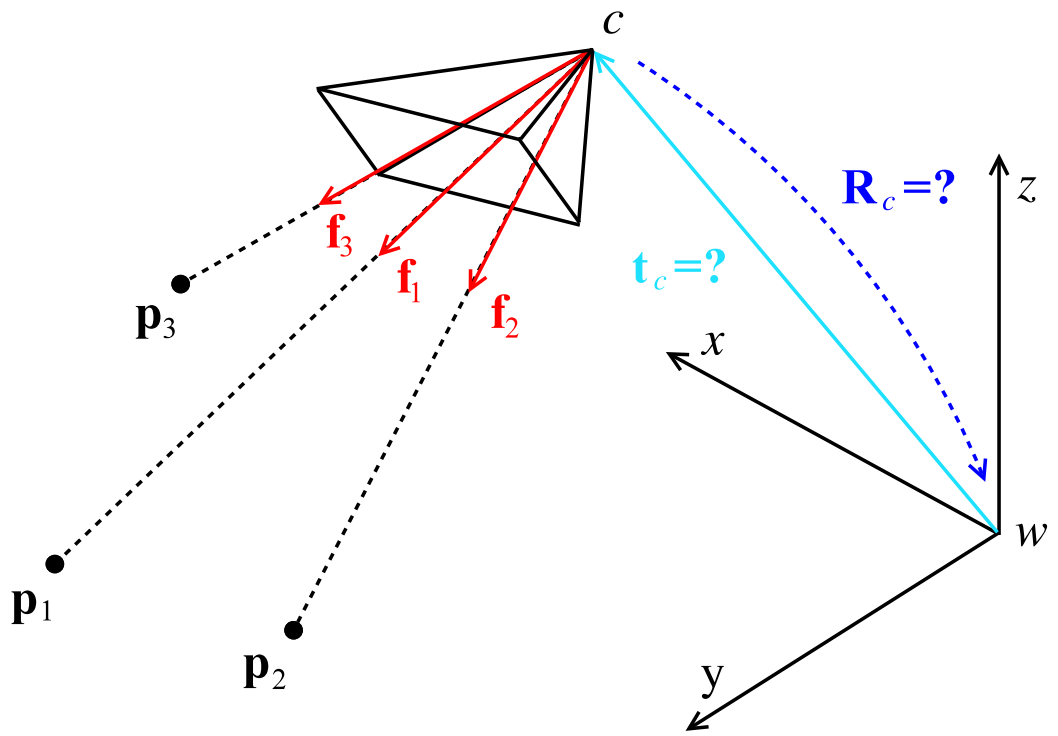
Price question

- Example: Absolute Pose
  - Objective:

$$\text{argmax}_{\mathbf{t}_c,\mathbf{R}_c} \sum_i \mathbf{f}_i^{\mathrm{t}} \cdot (\mathbf{R}_c^T \cdot (\mathbf{p}_i - \mathbf{t}_c))$$

  - What's a better objective?

$$\text{argmax}_{\mathbf{t}_c,\mathbf{R}_c} \sum_i \mathbf{f}_i^{\mathrm{t}} \cdot \frac{\mathbf{R}_c^T \cdot (\mathbf{p}_i - \mathbf{t}_c)}{\|\mathbf{R}_c^T \cdot (\mathbf{p}_i - \mathbf{t}_c)\|}$$

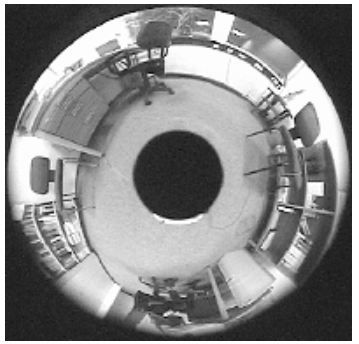


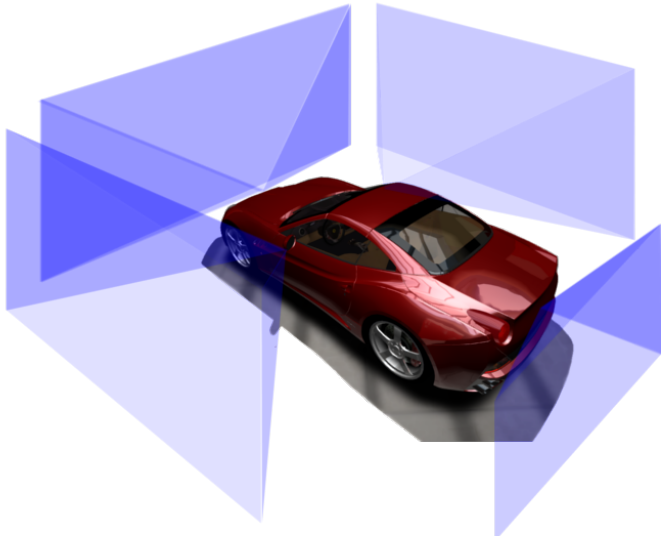
Where are we at?

- Today: Front-end choices, SLAM as a graphical optimization problem

Week	Tuesday course	Thursday course
18th February	Introduction (what is SLAM, curriculum, course structure (homeworks, projects), course webpage	Homogeneous coordinates, Transformations, etc.
25th February	Filtering methods (GF, PF)	Filtering methods (GF,PF)
4th March	Pose-only SLAM, Graph SLAM	Introduction to the projects
11th March	Camera as a sensor, camera calibration	Introduction into Visual SLAM, feature extraction, tracking, and matching
18th March	Feature extraction, tracking, and matching	Case study: MonoSLAM (why filter?)
25th March	Bootstrapping: The Relative Pose Problem	Full visual odometry pipeline, Non-linear optimization, Bundle adjustment
1st April	Non-linear optimization, Bundle adjustment	Place recognition, loop closure
8th April	Place recognition, loop closure, the absolute pose problem	Case study: ORB SLAM
15th April	Dense Tracking and Mapping (DTAM), Photometric vs geometric errors, semi-dense opt.	RGBD SLAM, Iterative closest points (Laser-point cloud registration), TSDF
22nd April	SLAM with Stereo and Multi-Perspective cameras	Midterm exam
29th April	Visual-Inertial SLAM	Project discussions/buffer
6th May	Dynamic and Multi-body SLAM	Project discussions/buffer
13th May	Semantic SLAM	Project Presentations

SLAM: different sensors

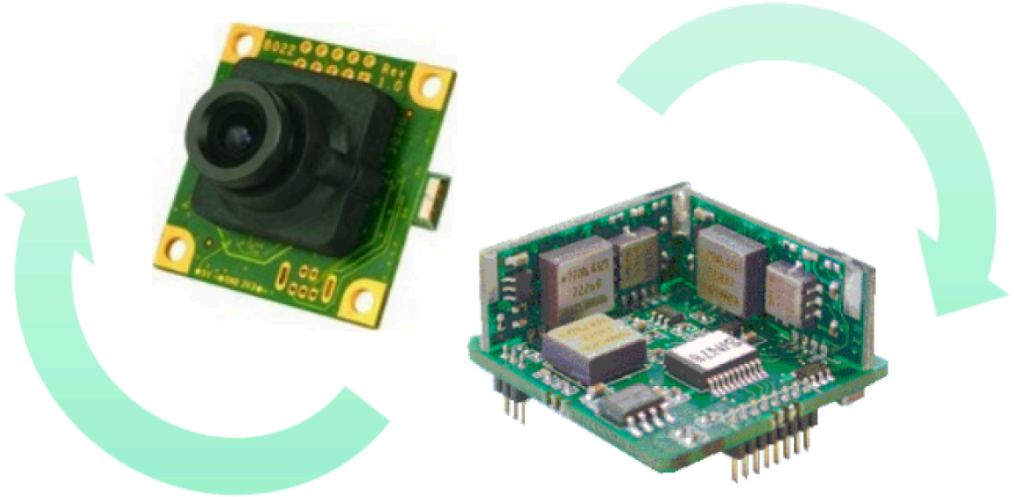


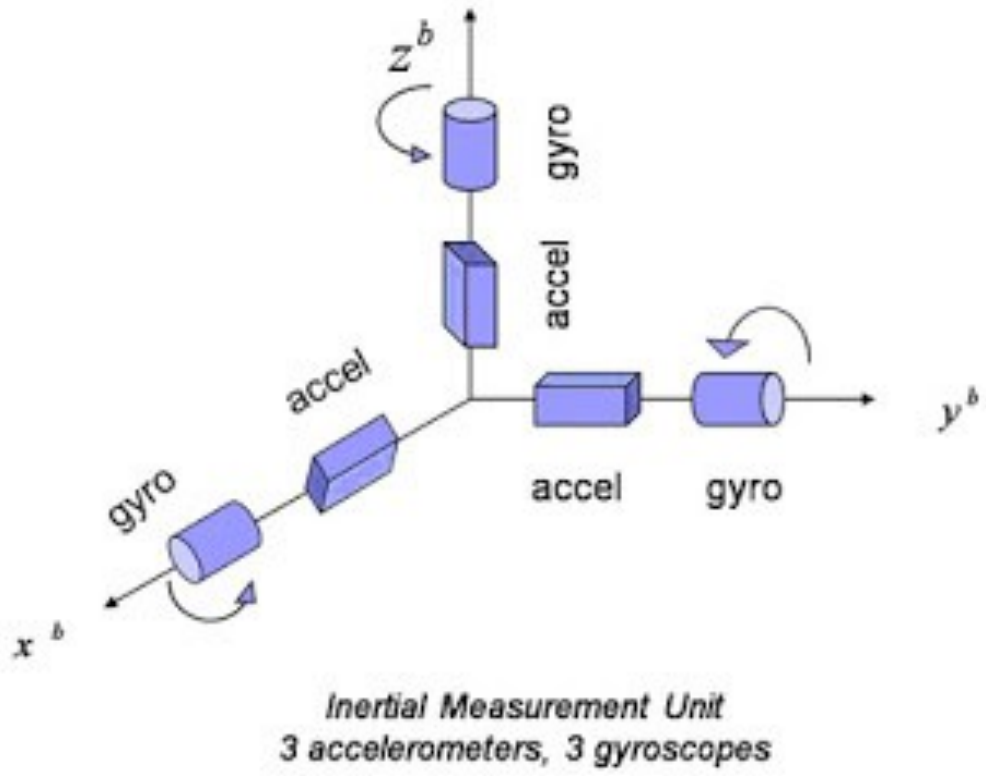


SLAM: different sensors

SLAM: different sensors

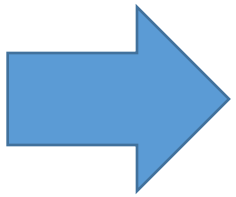
- Inertial sensors

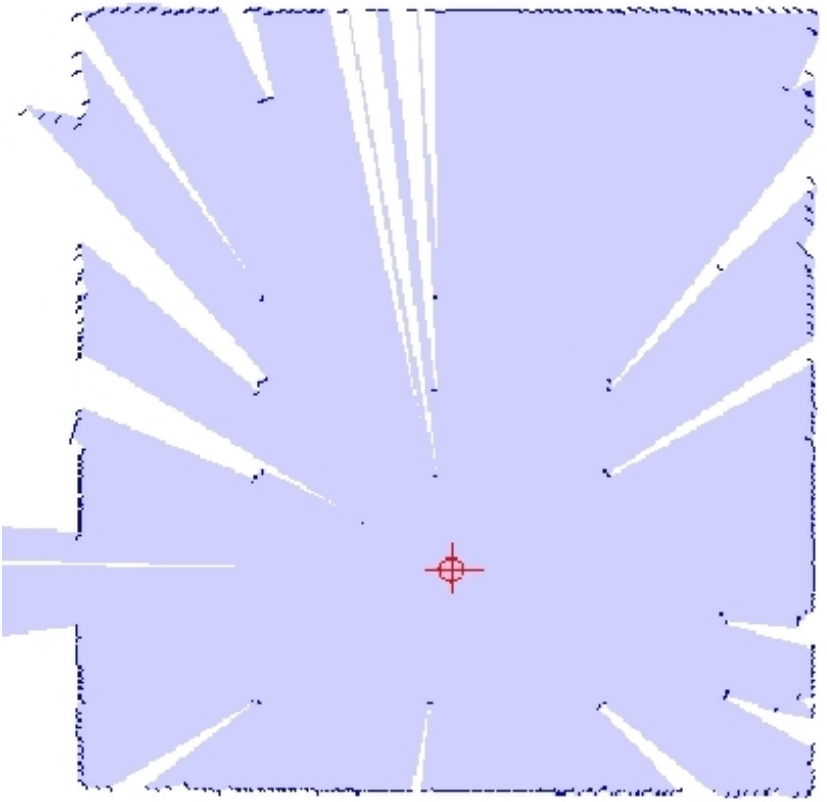




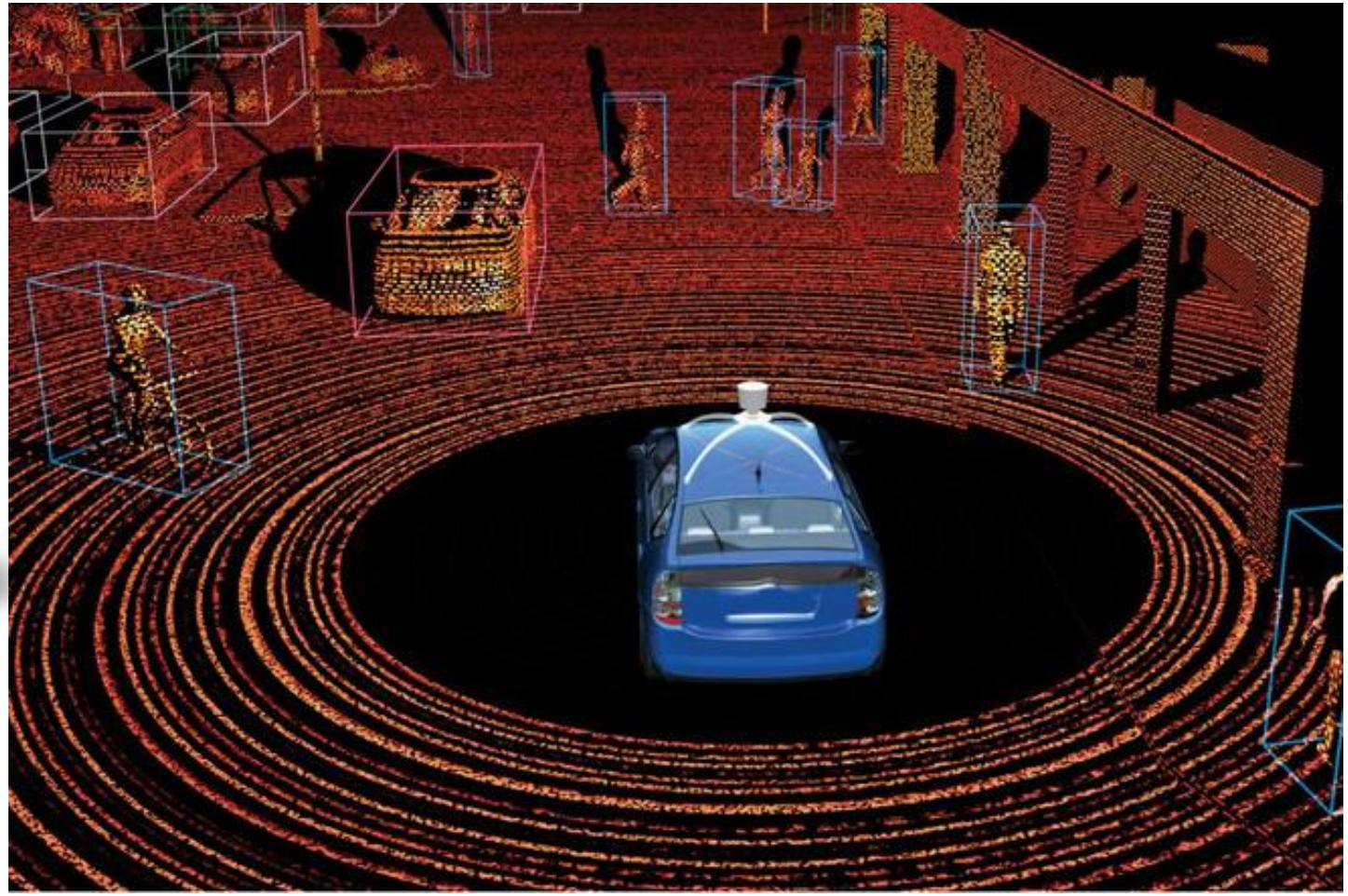
Inertial Measurement Unit
3 accelerometers, 3 gyroscopes

SLAM: different sensors

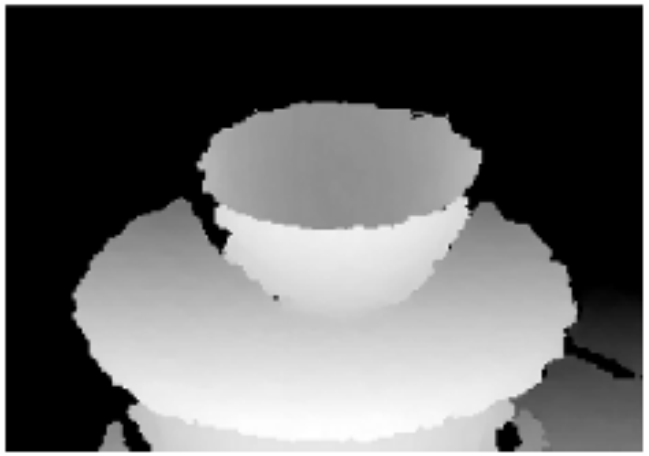




SLAM: different sensors



SLAM: different sensors



KINECT
for XBOX 360.

Sensor classes

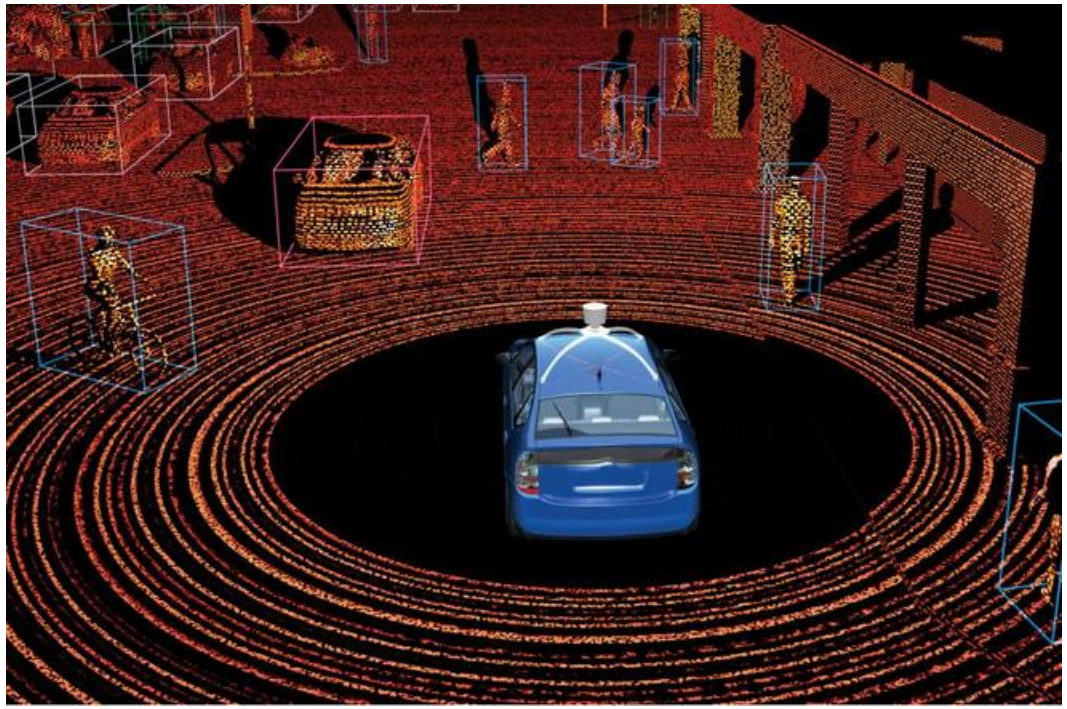
- Two main types of exteroceptive sensors used in robotics:
  - **Photometric:**
    energy reflection (regular images), energy emission
  - **Geometric:**
    position, surface, volume, etc.

Sensor classes

- Photometric (visual) sensors
  - Measures photometric effects
  - Allows us to extract features
    - 2D-2D, 2D-3D, or 3D-3D correspondences
  - Allows us to map those features to 3D points (explicitly)
    - Represented either as nodes in a graph, or in a state vector
    - Full graph representations depend on it

Sensor classes

- Geometric sensors (Laser scanners)
  - Typically: no feature/point correspondences!
    - No appearance information (or at least rarely used)
    - Only few, unstable local invariant keypoints (geometric!)



Sensor classes

	Photometric sensor	**Geometric sensors**
Pros	• Low cost/weight/energy sensors; • Rich textures; • Primary problem: pose estimation	• Direct depth measurements; • Robust to environment changes; • Primary problem: map representation

Sensor classes

	Photometric sensor	Geometric sensors
Pros	• Low cost/weight/energy sensors; • Rich textures; • Primary problem: pose estimation	• Direct depth measurements; • Robust to environment changes; • Primary problem: map representation
Cons	• No depth; scale unobservability • Dependent on external light source;	• Heavy/expensive sensors; • Systematic noise;

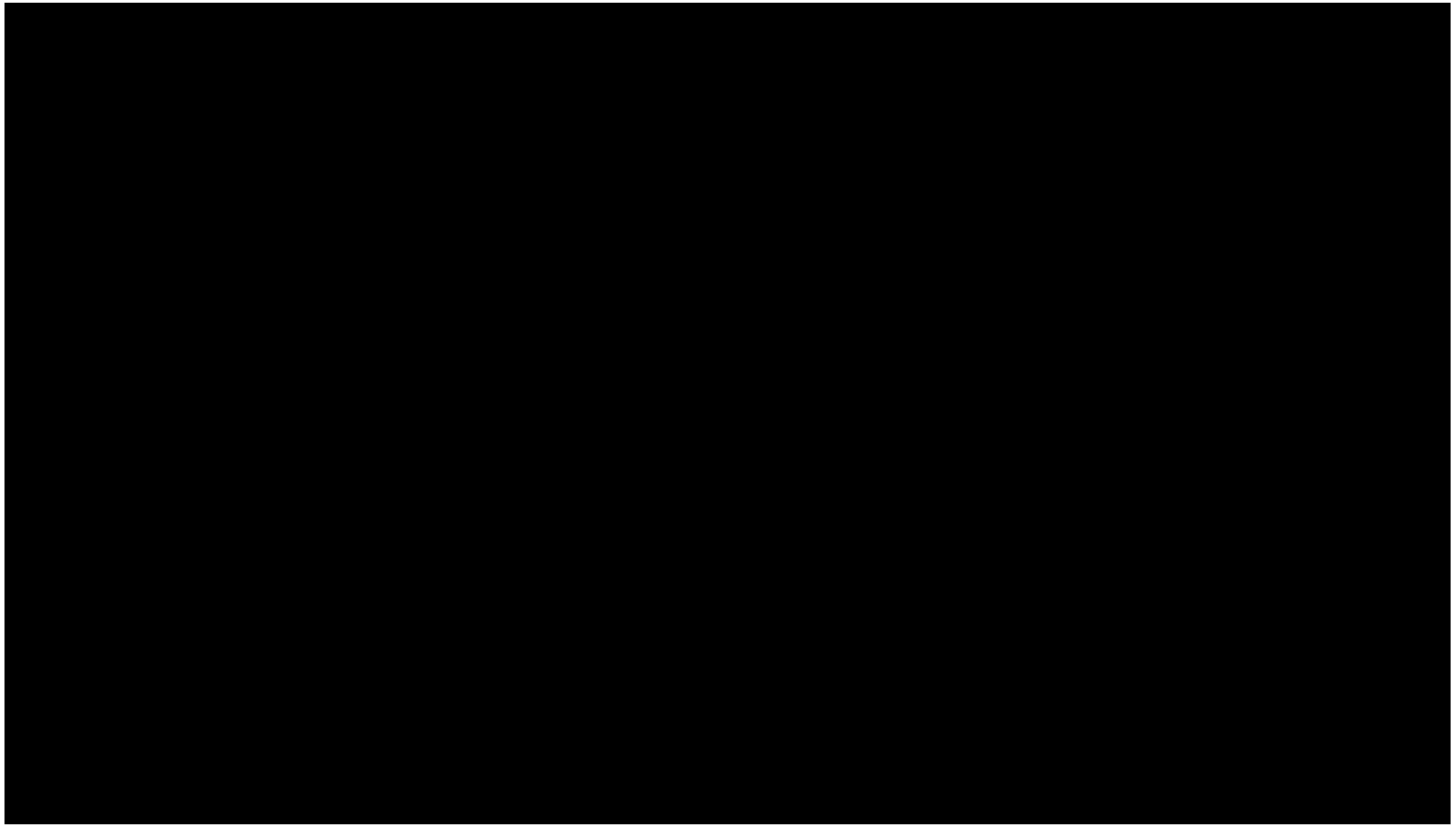
Some sensors make things easy

- Laser scanners
  - 2D or 3D
  - directly sense depth!
- Cheap alternative
  - RGB-D camera
- Let’s start with those examples!

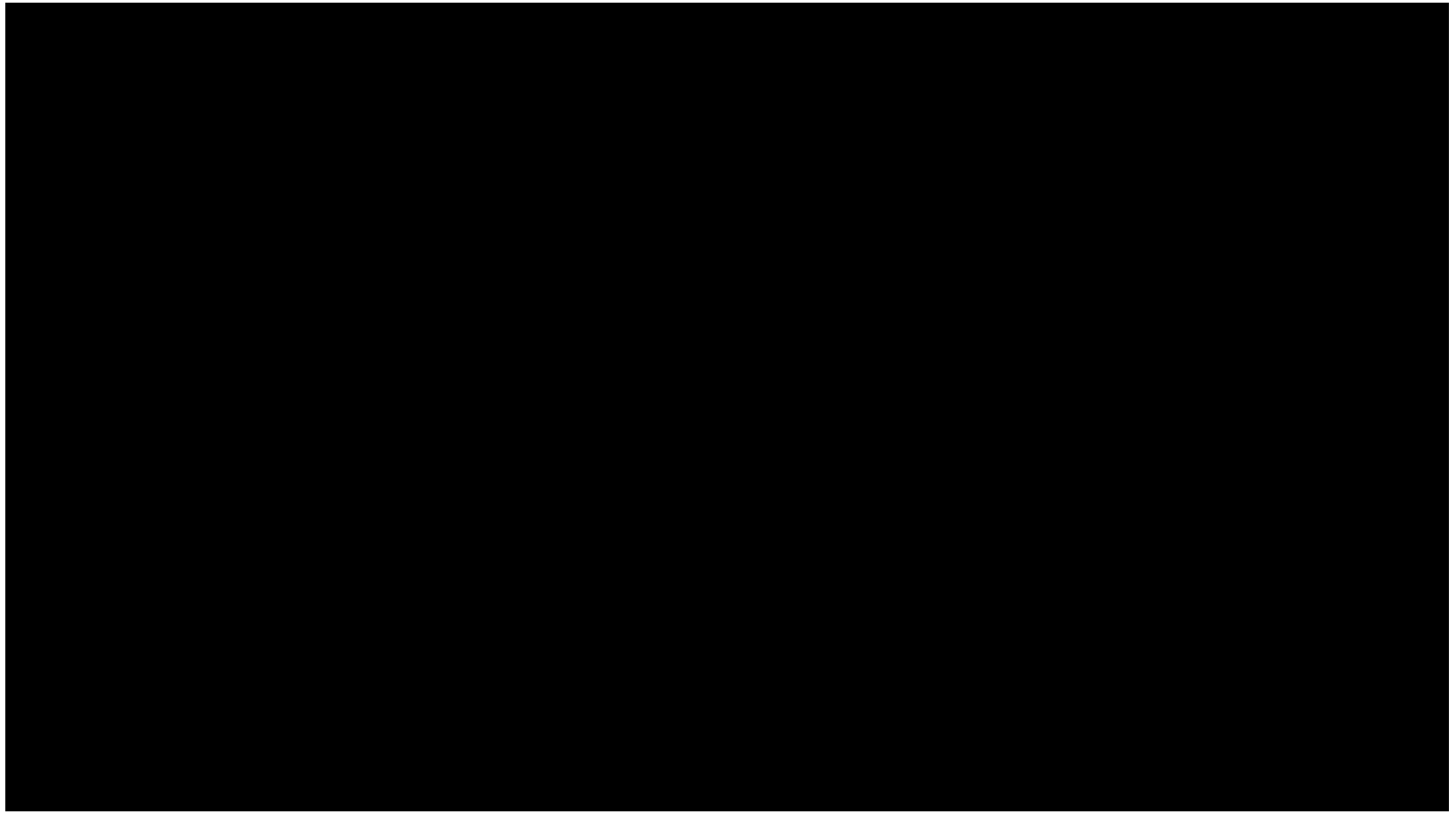
Example

- Robot providing odometry
- Equipped with
  - Planar Laser scanner
  - Pitching planar Laser scanner
  - Velodyne
  - …

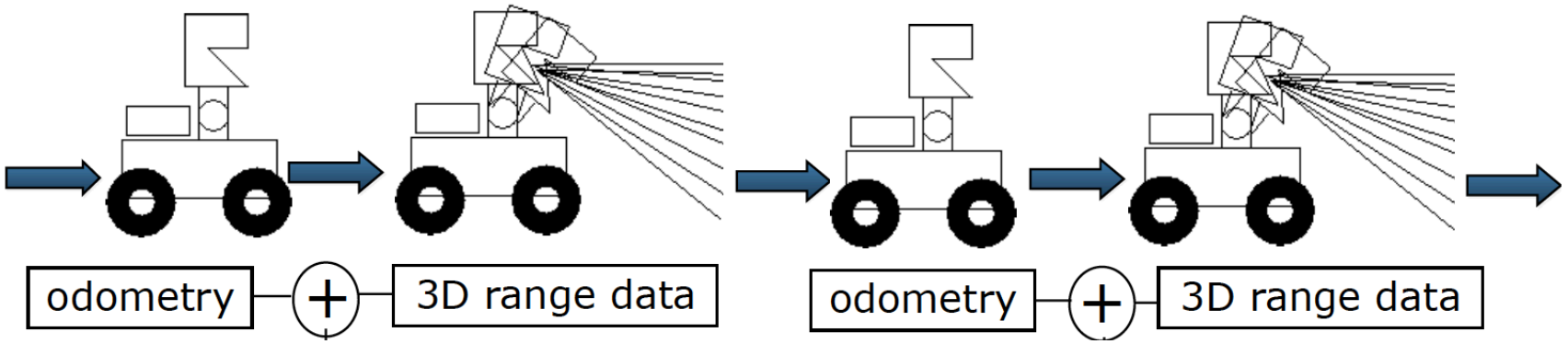
Example



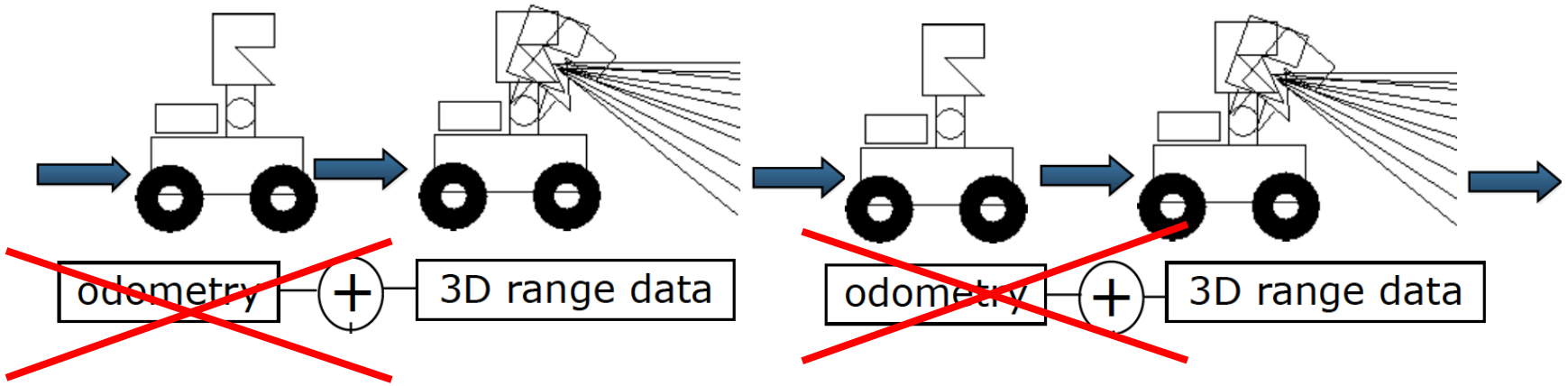
Also possible in full 3D (or 6D)



Example

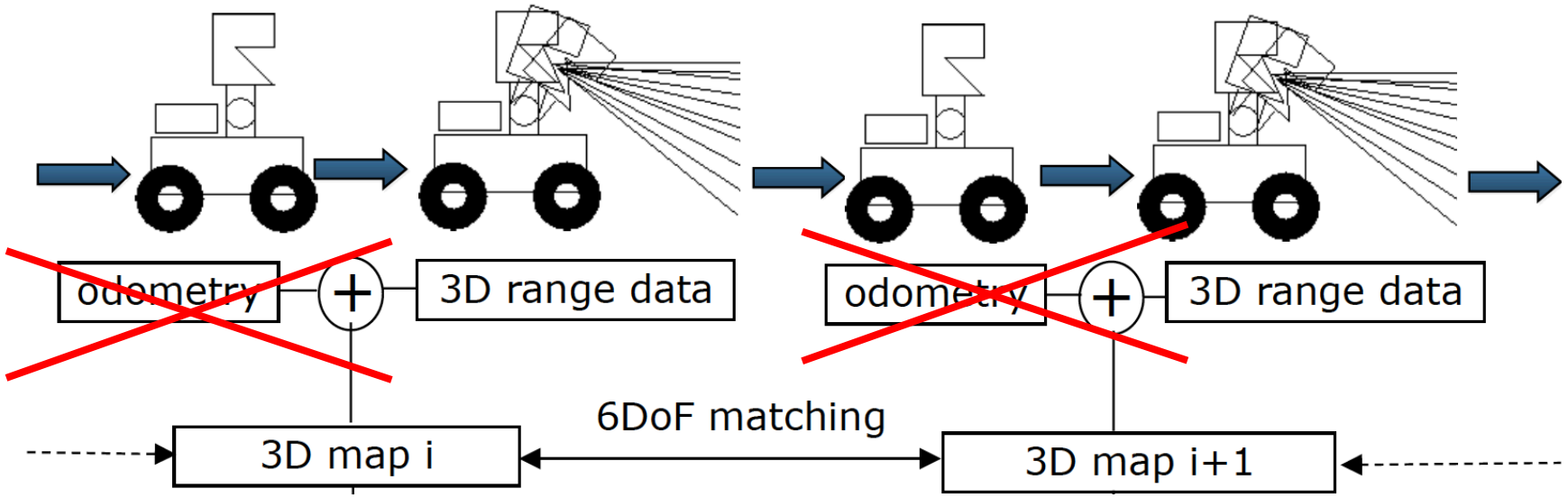


Example

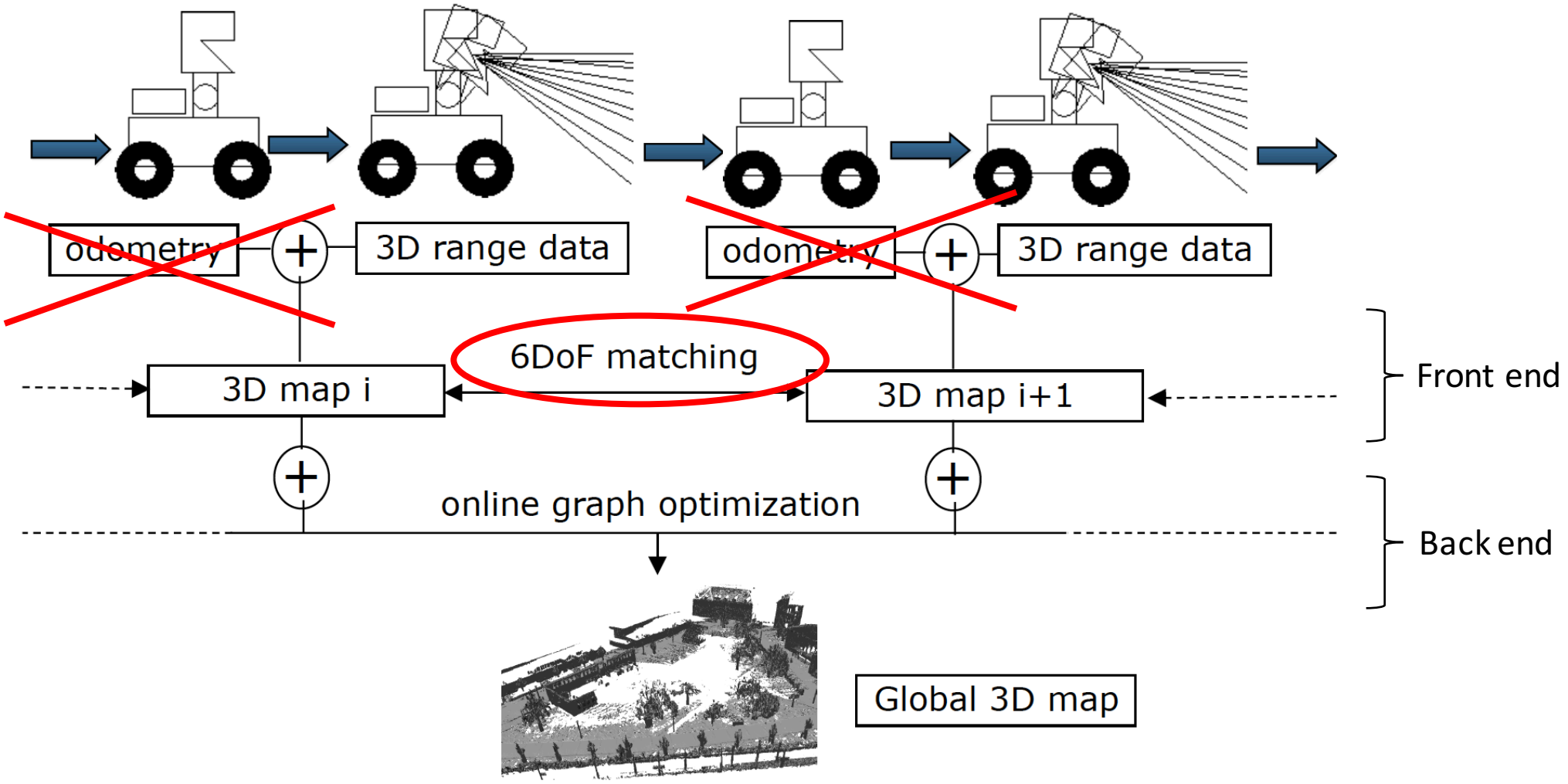


- Odometry is inaccurate
  - Cheap sensors
  - Wheel slippage
  - Difficult terrain

Example

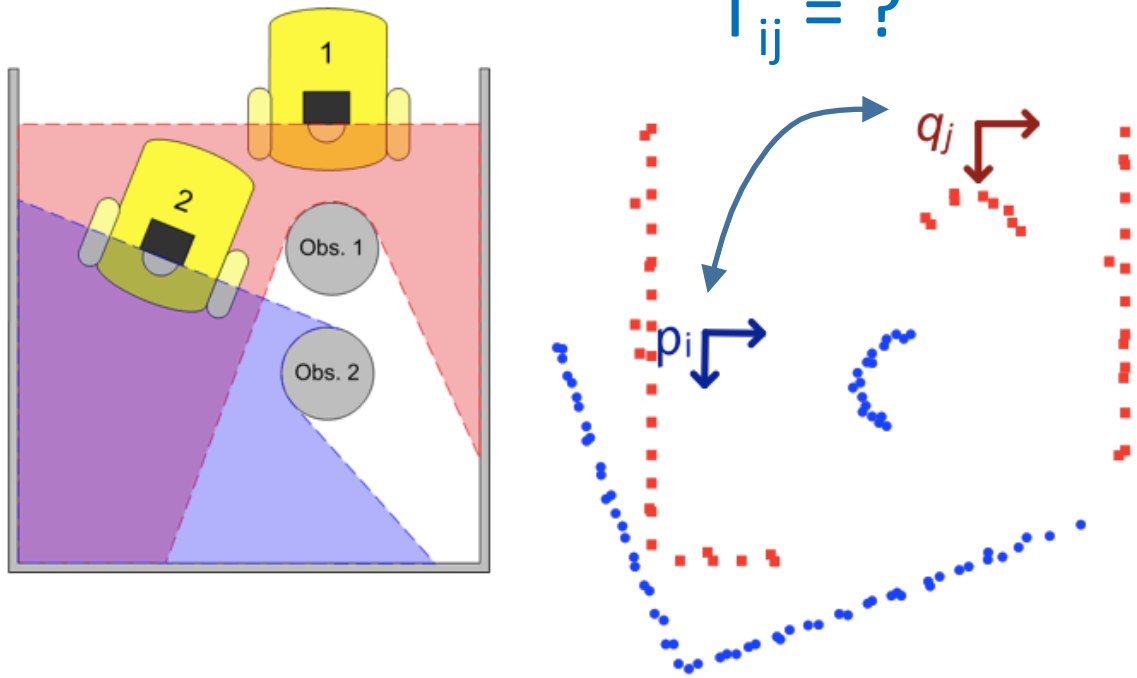

odometry
3D range data
odometry
3D range data
6DoF matching
3D map i
3D map i+1

Example


odometry
3D range data
odometry
3D range data
6DoF matching
3D map i
3D map i+1
Front end
online graph optimization
Back end
Global 3D map

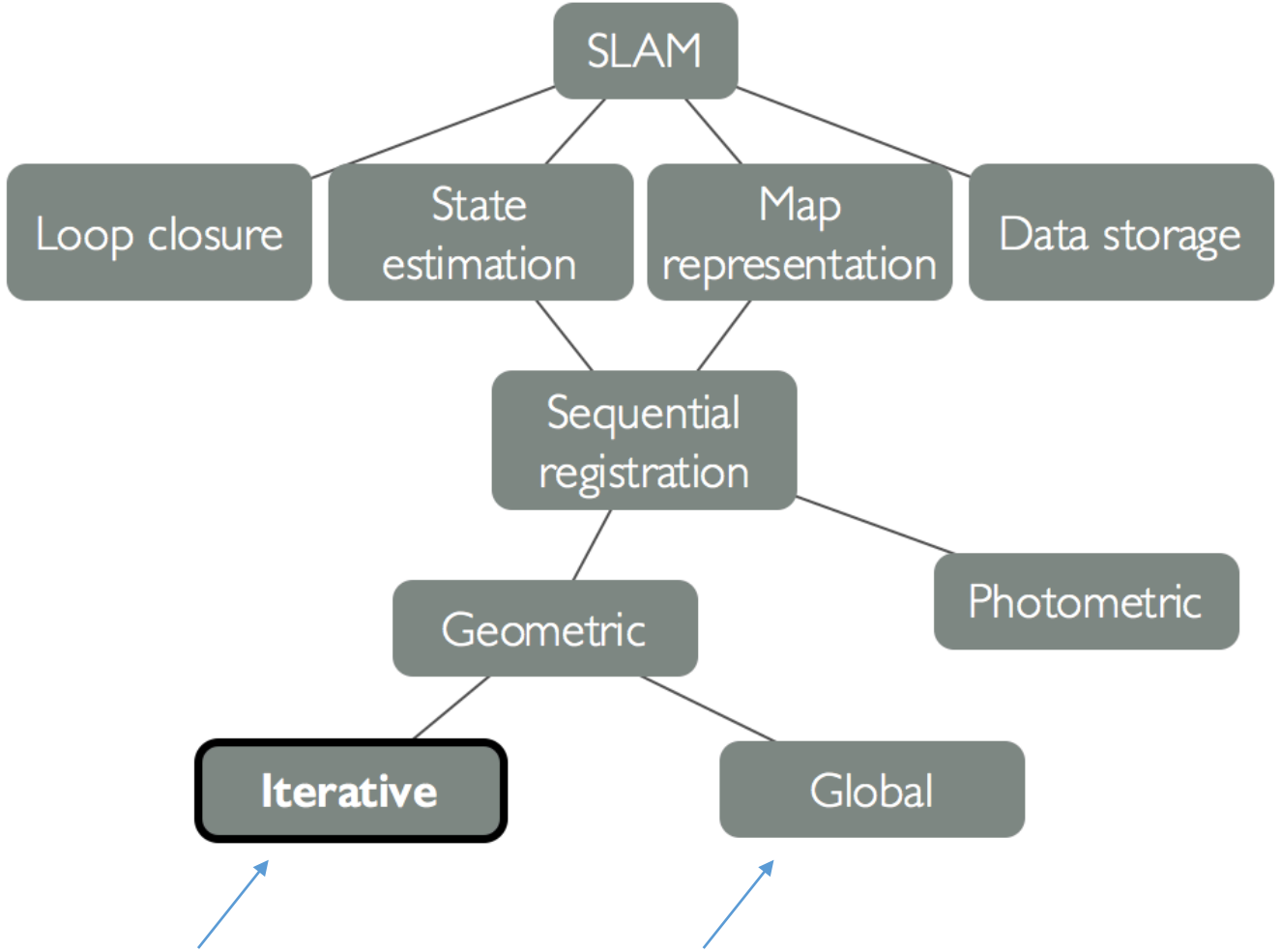
SLAM with Lasers

- How to do 6 DoF scan registration for relative localization?

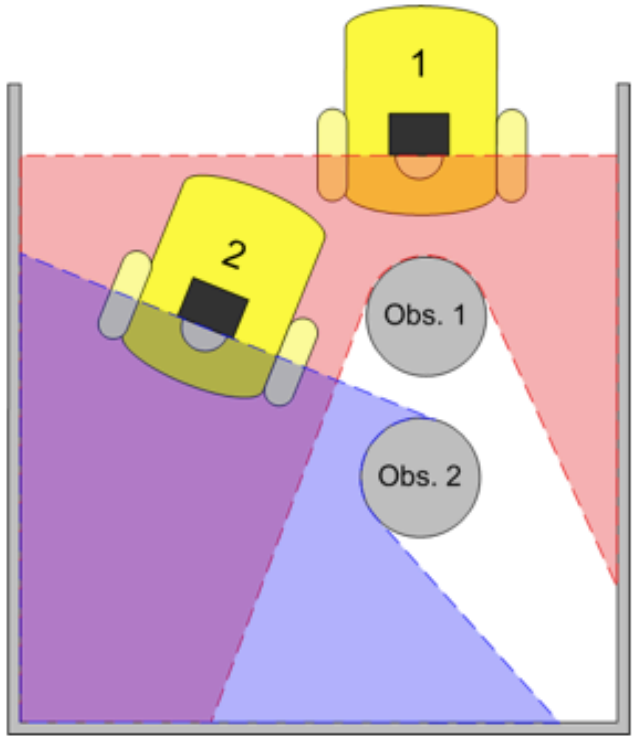


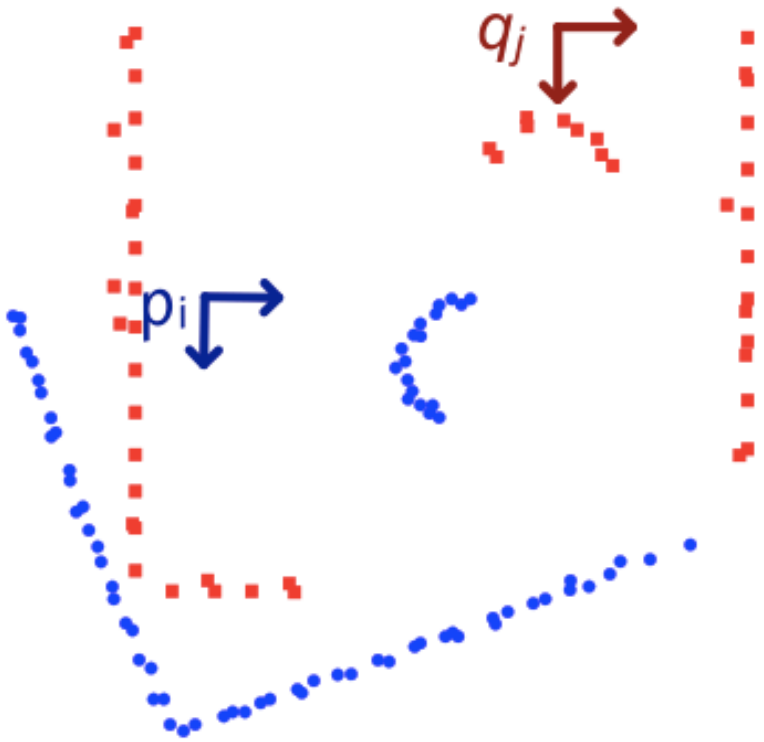
Iterative Closest Points (ICP)

Classification of registration methods



ICP: Toy example in 2D

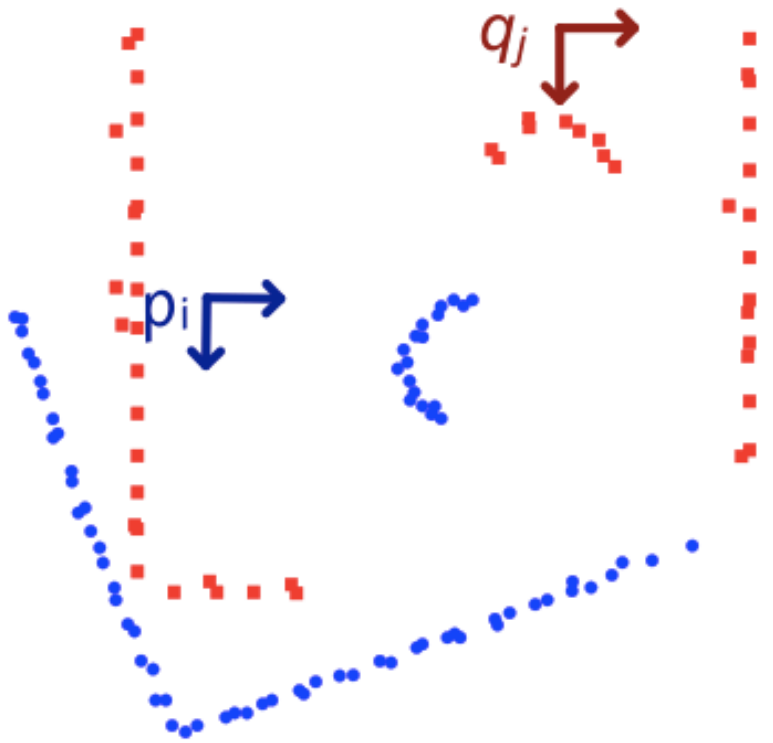




ICP: Toy example in 2D

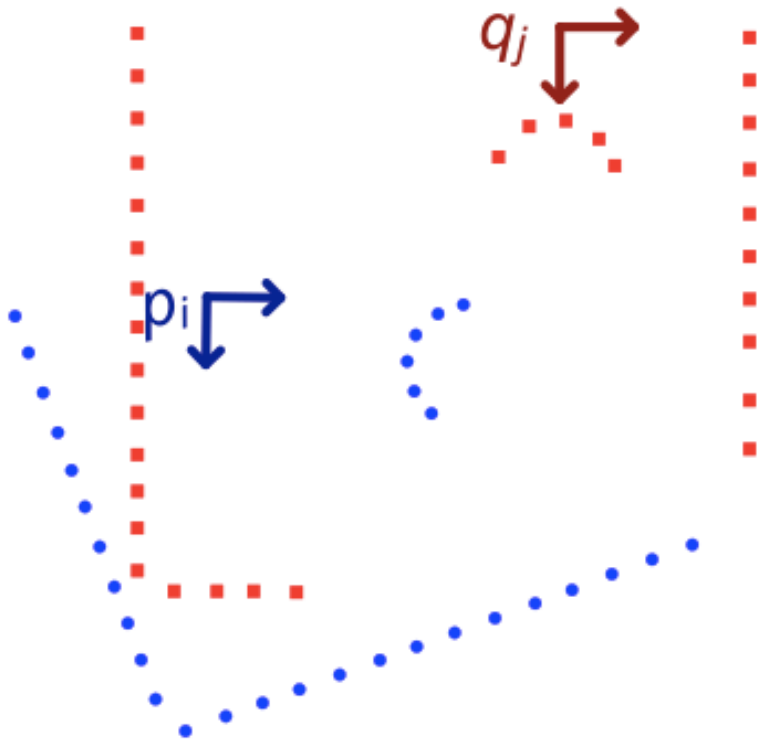
1. Preprocessing
2. Matching
3. Weighting
4. Rejection
5. Error
6. Minimization

(Steps defined in Rusinkiewicz 01)



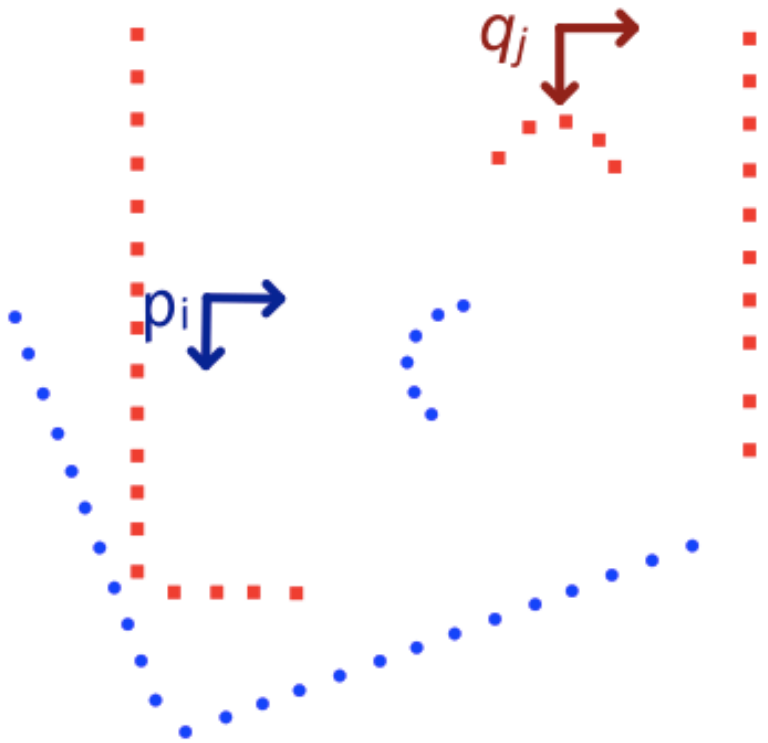
ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. Weighting
4. Rejection
5. Error
6. Minimization



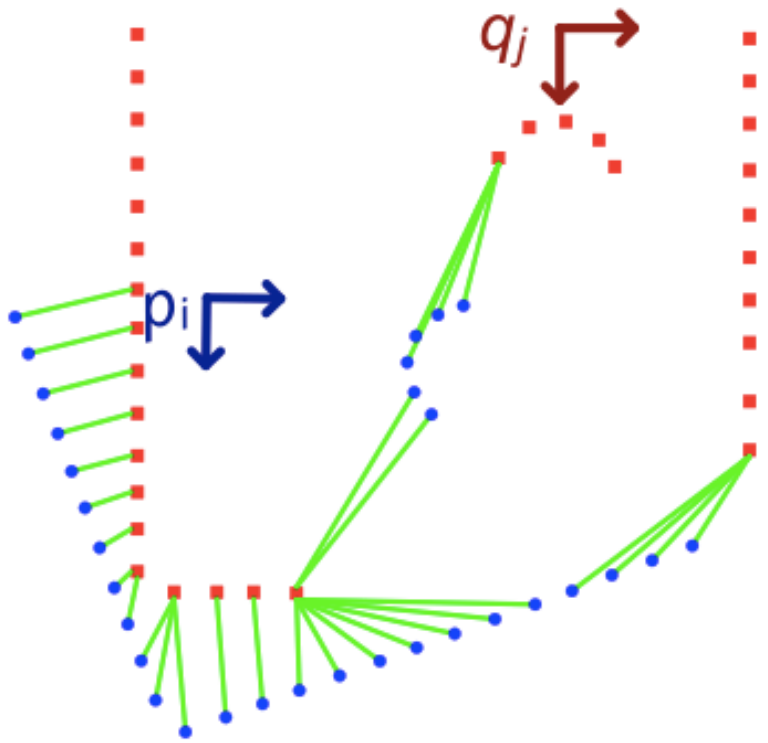
ICP: Toy example in 2D

1. Preprocessing
2. **Matching**
3. Weighting
4. Rejection
5. Error
6. Minimization



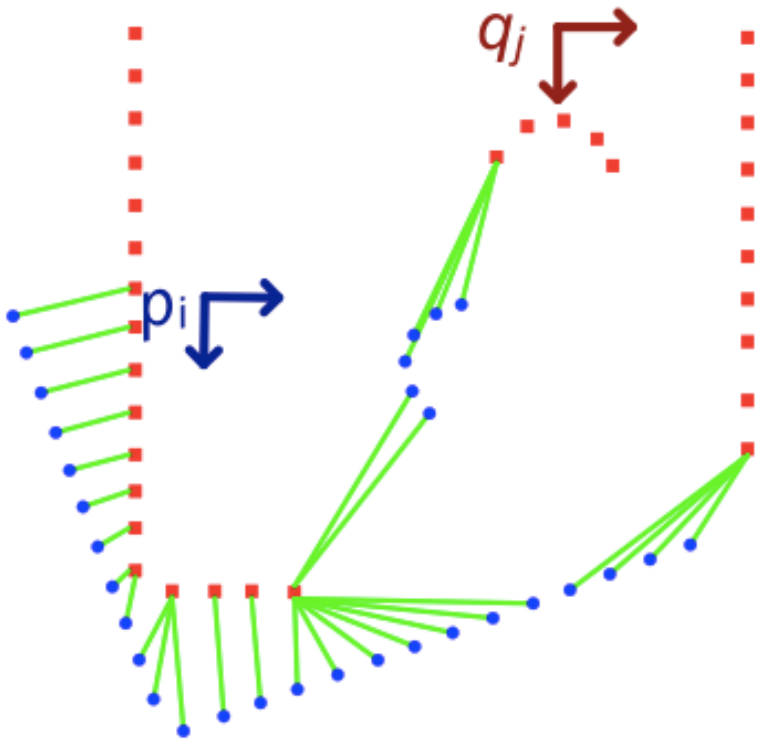
ICP: Toy example in 2D

1. Preprocessing
2. **Matching**
3. Weighting
4. Rejection
5. Error
6. Minimization



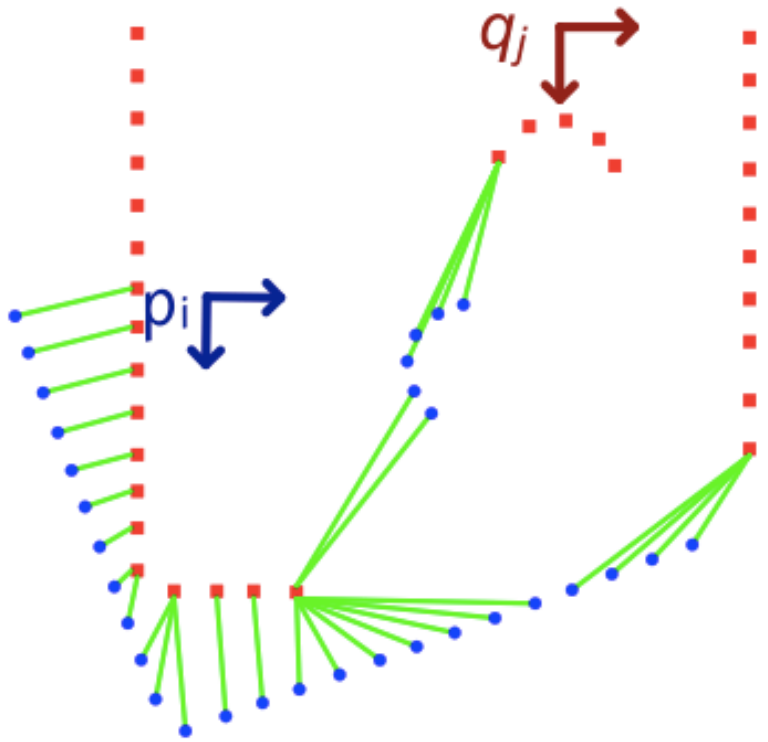
ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. **Weighting**
4. Rejection
5. Error
6. Minimization



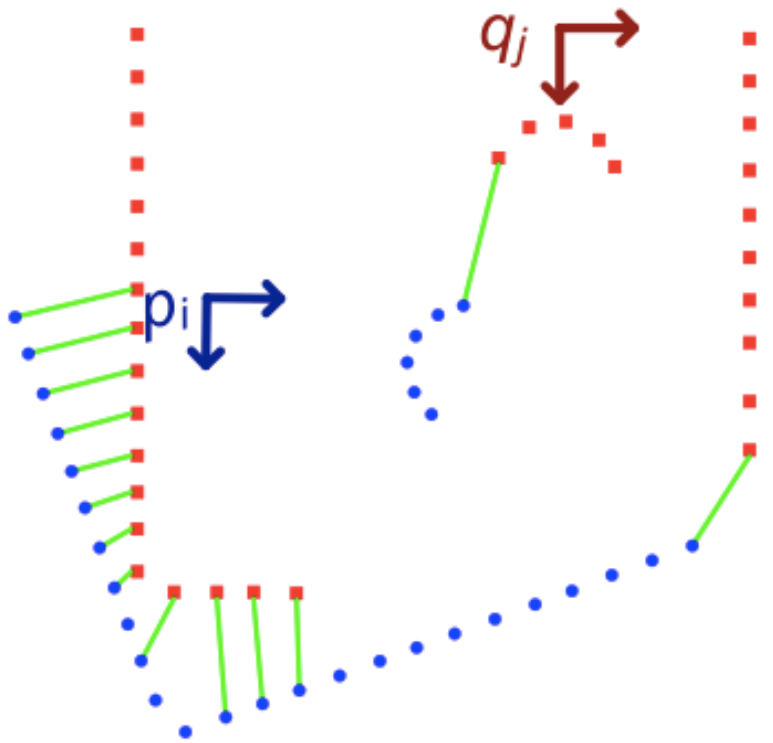
ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. Weighting
4. **Rejection**
5. Error
6. Minimization



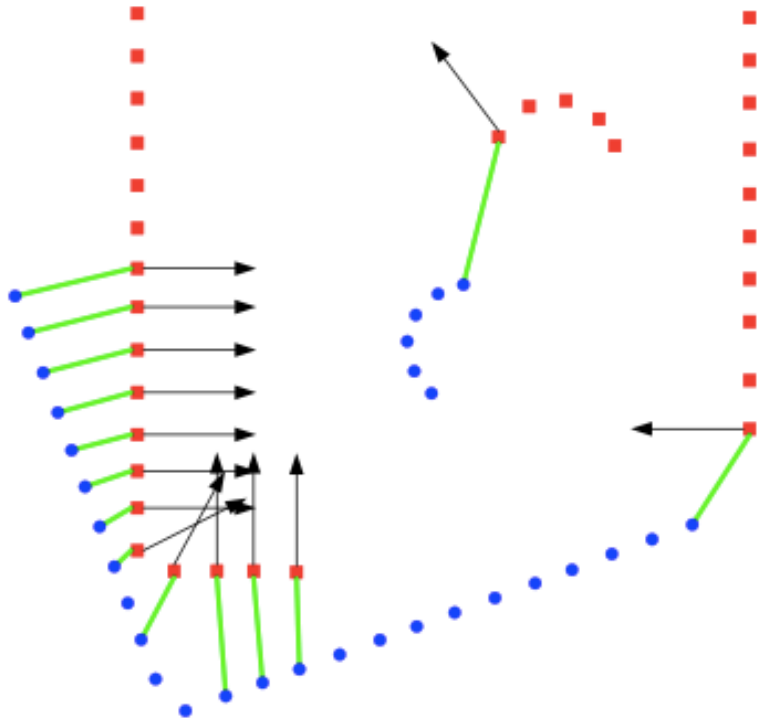
ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. Weighting
4. **Rejection**
5. Error
6. Minimization



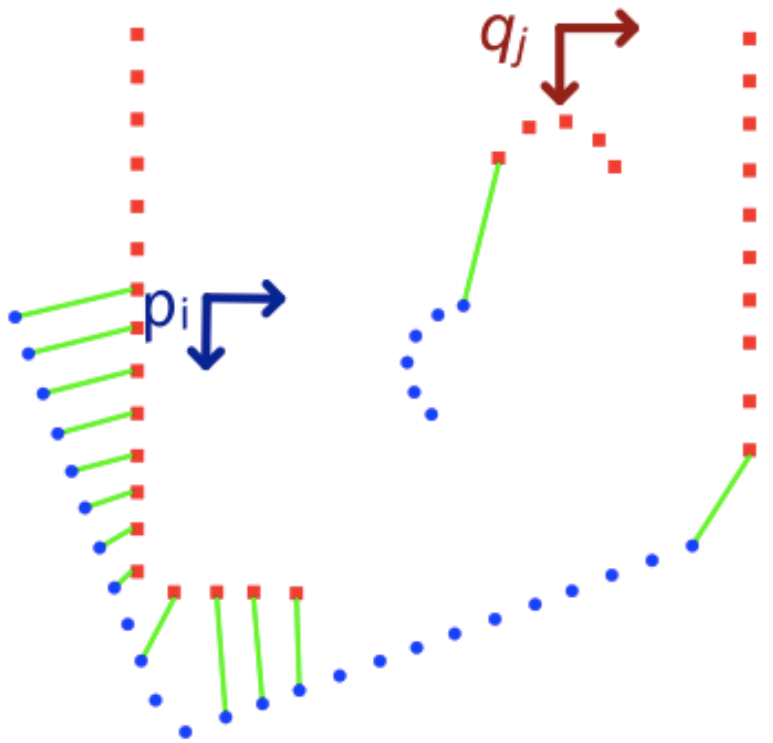
ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. Weighting
4. Rejection
5. **Error**
6. Minimization



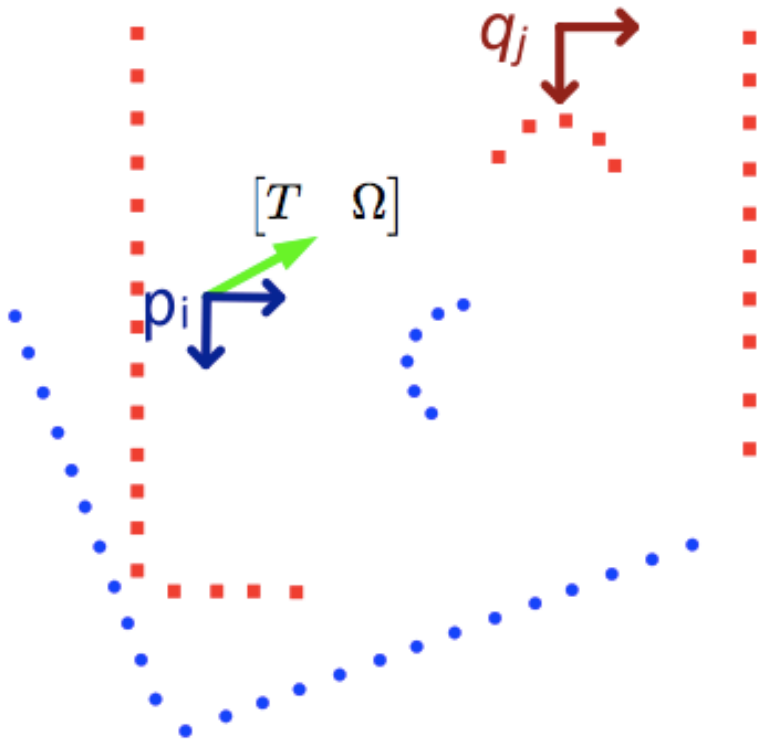
ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. Weighting
4. Rejection
5. Error
6. **Minimization**



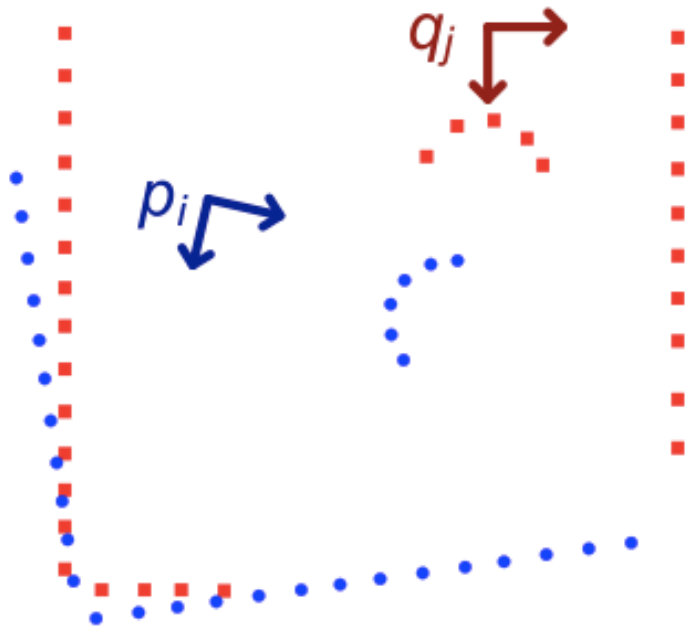
ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. Weighting
4. Rejection
5. Error
6. **Minimization**



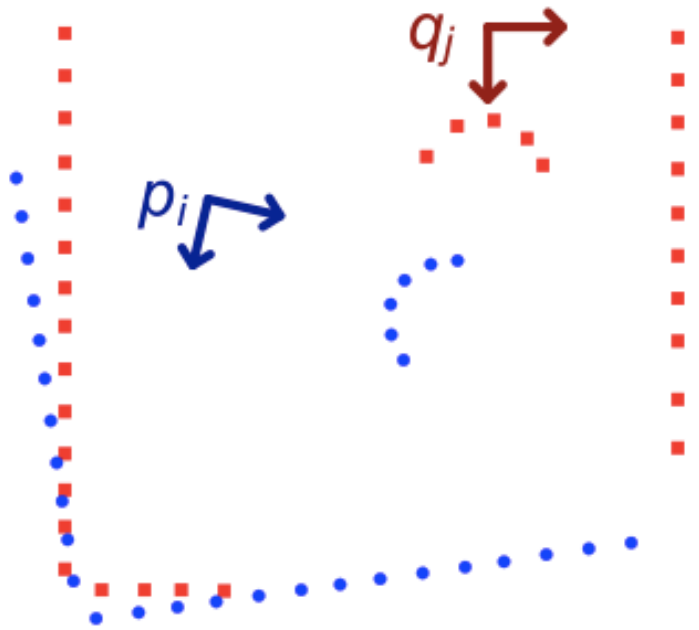
ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. Weighting
4. Rejection
5. Error
6. **Minimization**



ICP: Toy example in 2D

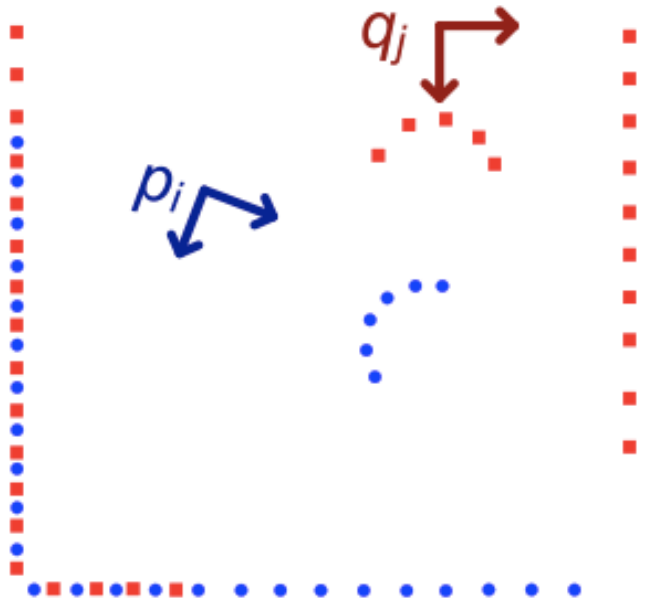
1. Preprocessing
2. Matching
3. Weighting
4. Rejection
5. Error
6. **Minimization**



ICP: Toy example in 2D

1. Preprocessing
2. Matching
3. Weighting
4. Rejection
5. Error
6. Minimization

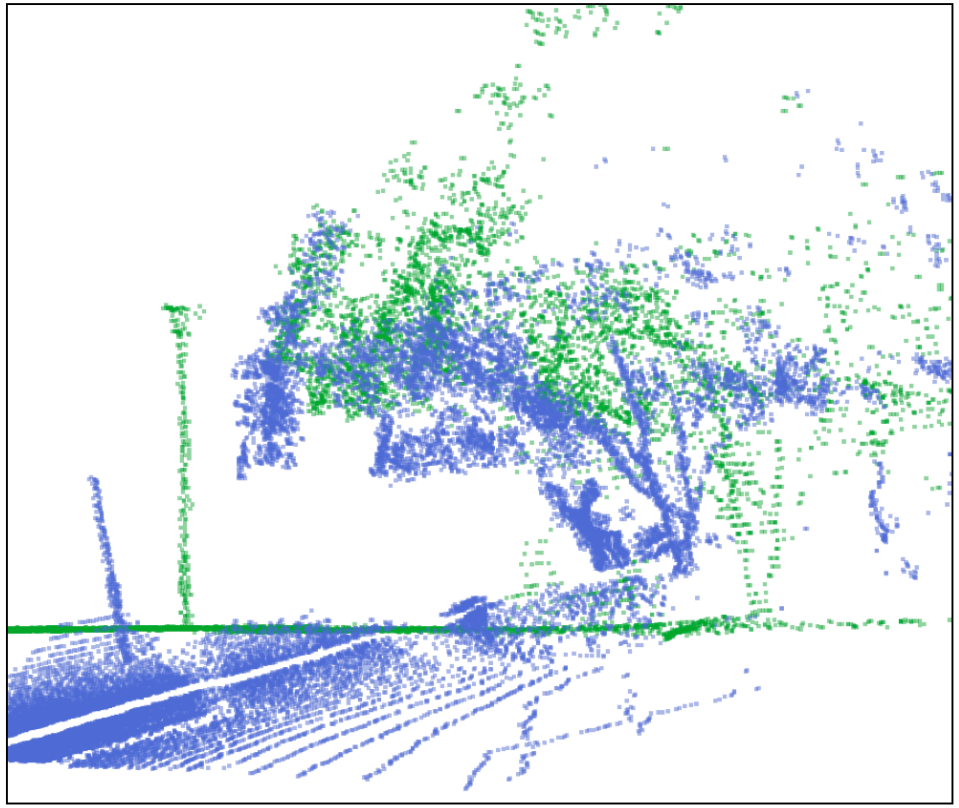
End conditions



ICP: Example in 3D

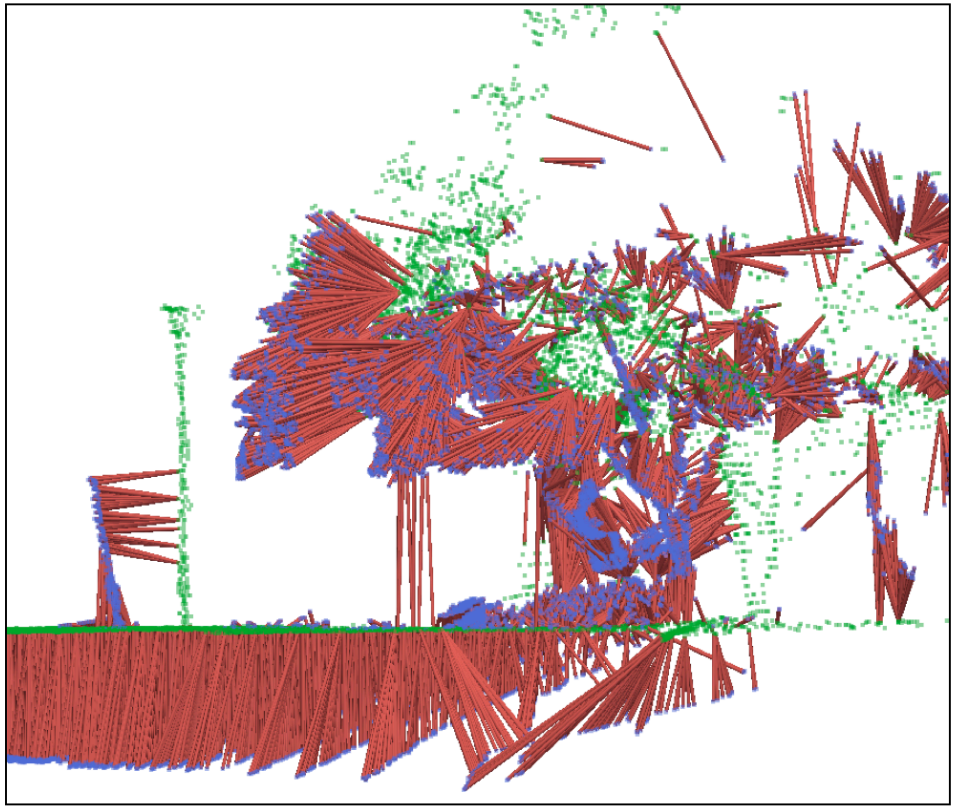
Reading

Reference



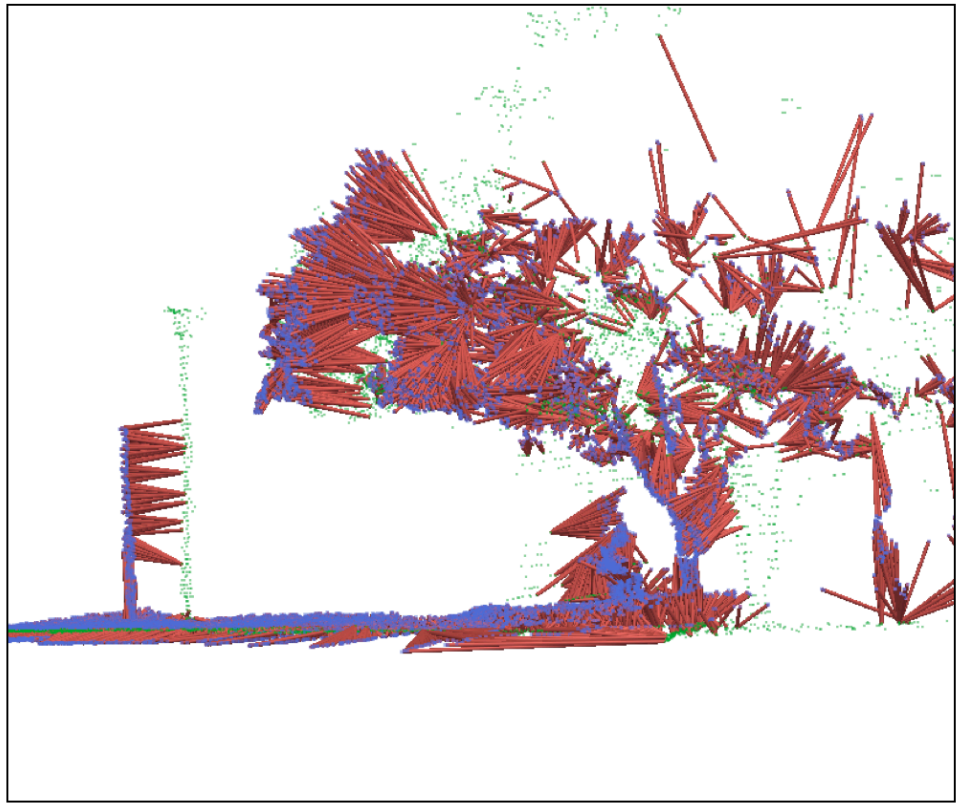
ICP: Example in 3D

Reading Error Reference



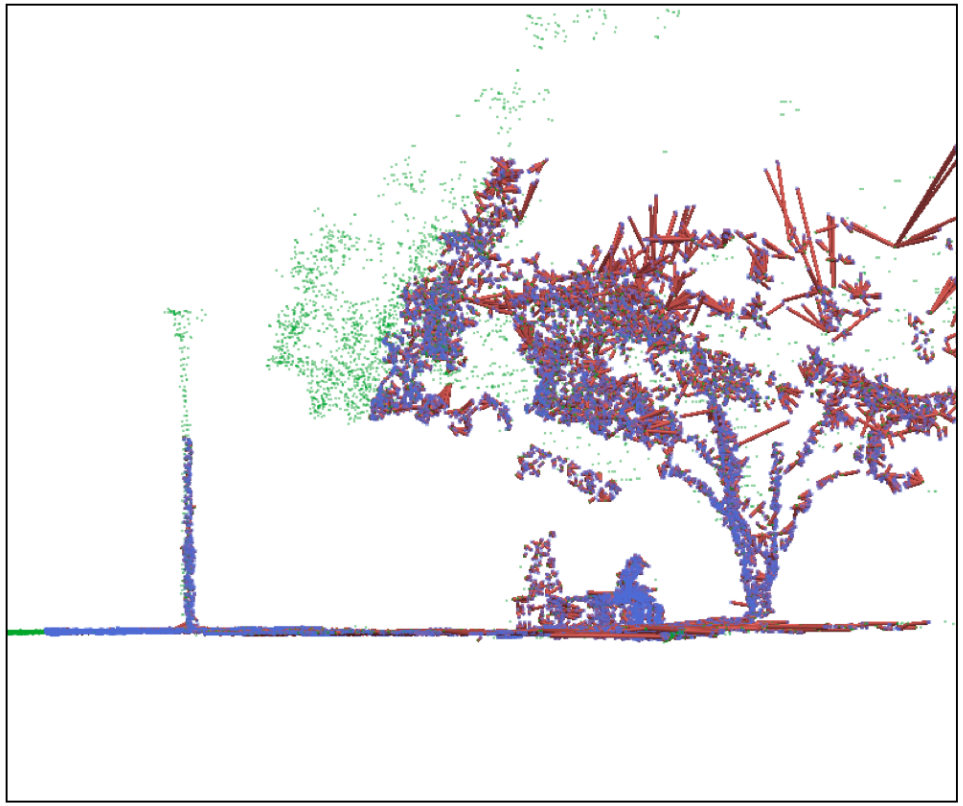
ICP: Example in 3D

Reading Error Reference



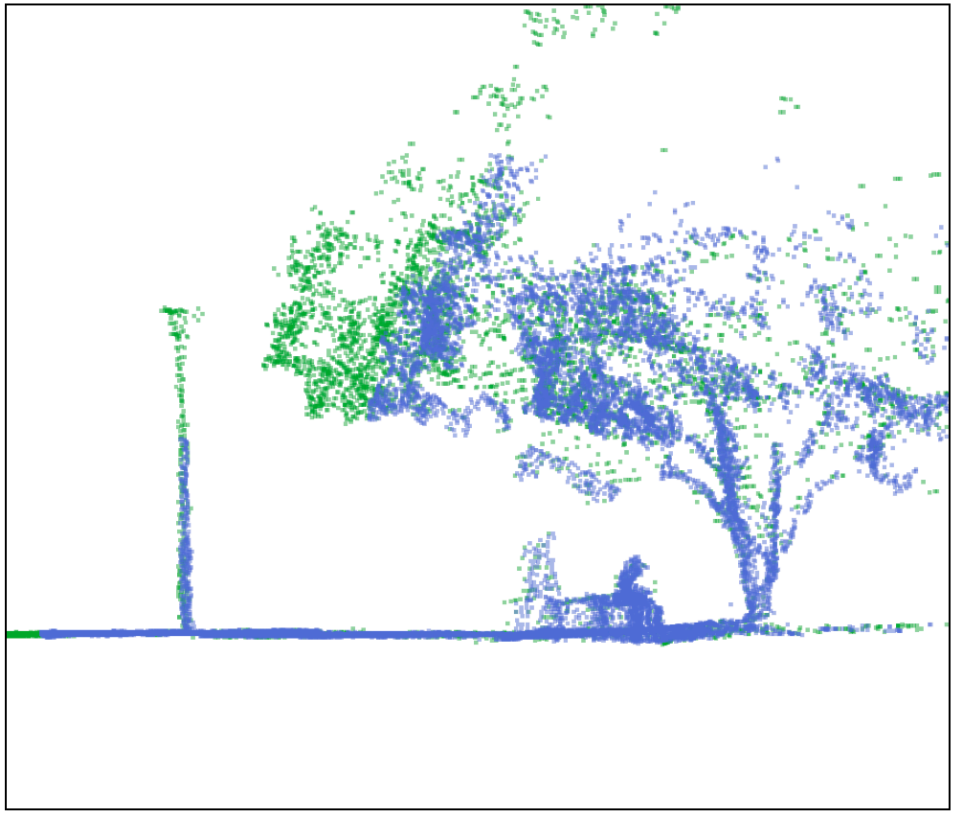
ICP: Example in 3D

Reading Error Reference

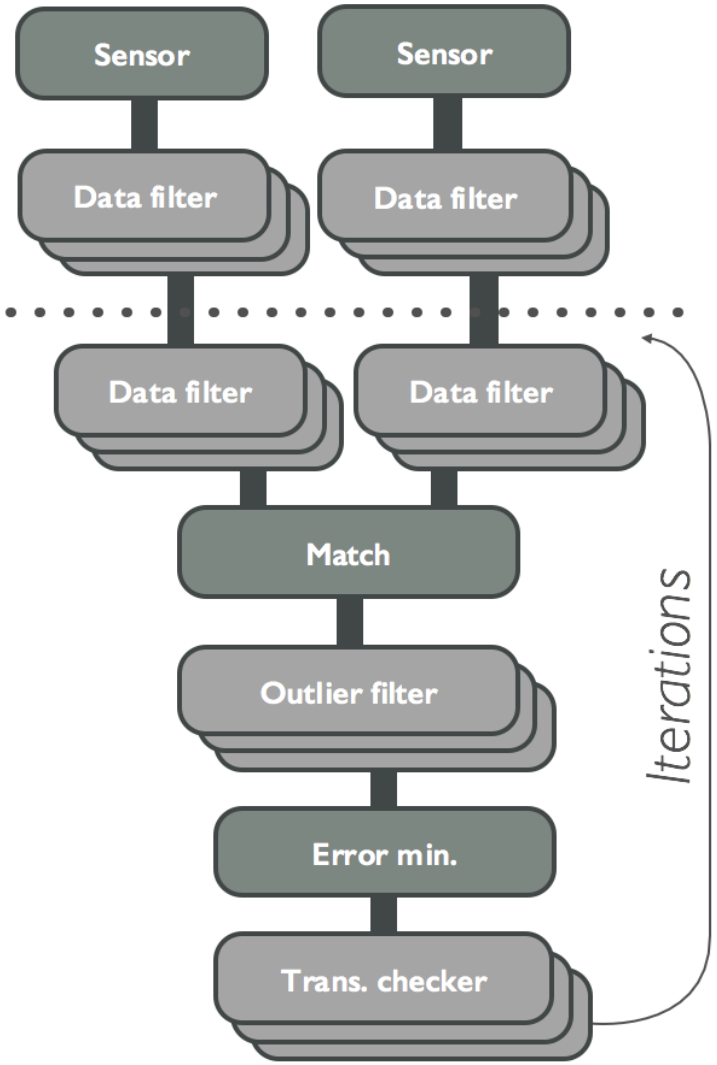


ICP: Example in 3D

Reading Error Reference

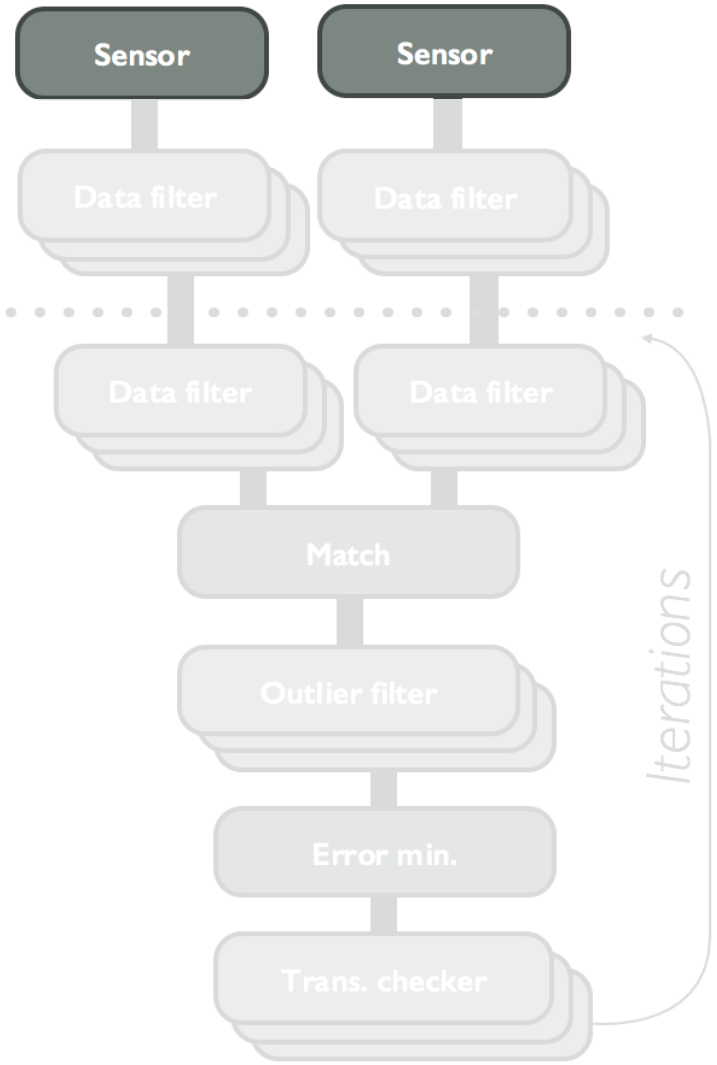


Overview



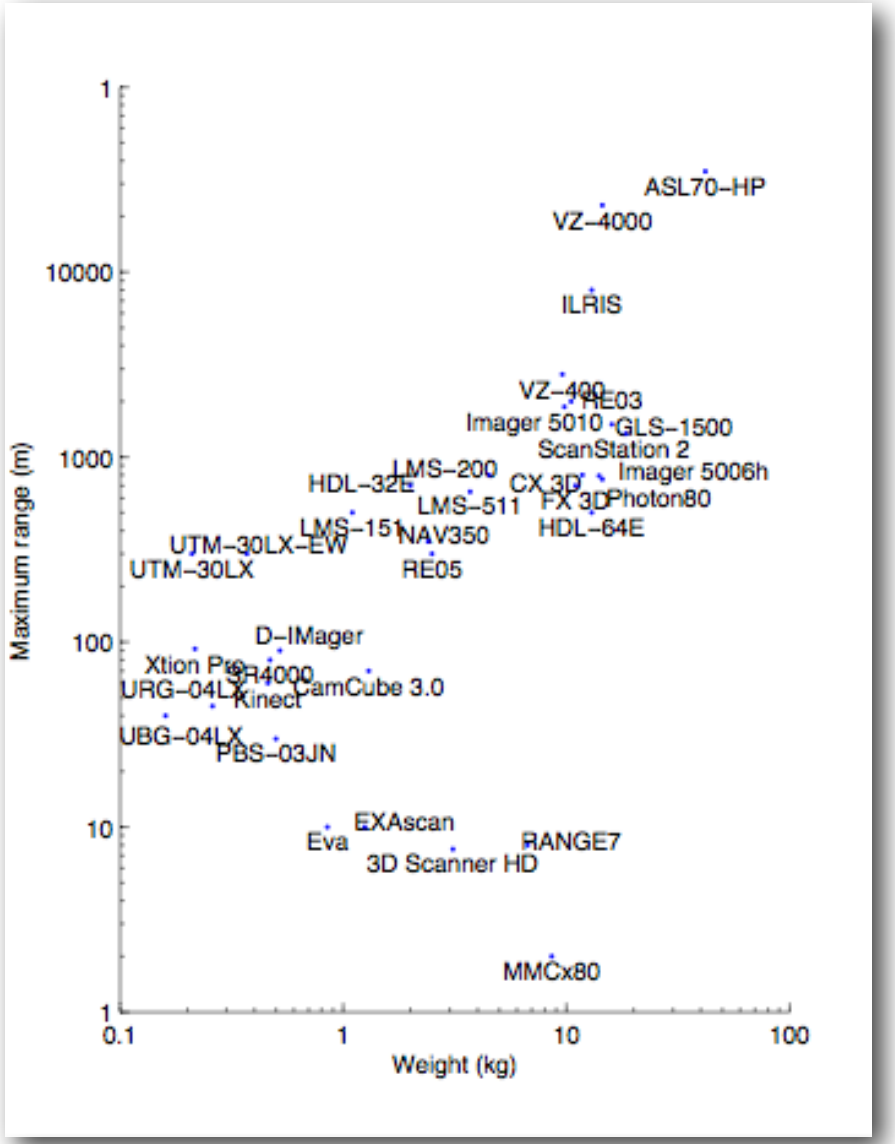
- Sensors
- Data filters
- Matching function
- Robust regression
- Minimization
- Convergence detection

Sensors

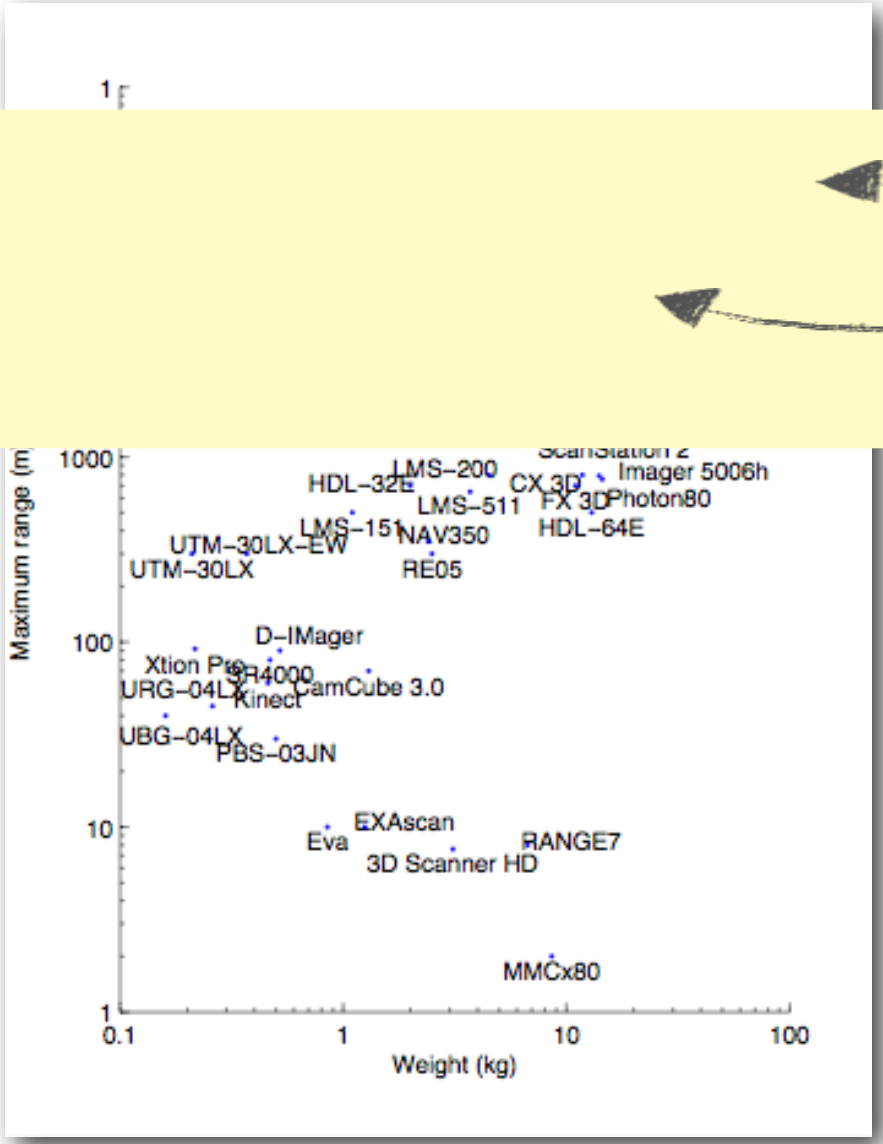


- Characteristics
  - Power consumption
  - Field of view
  - Accuracy/repeatability
  - Min/max distance
  - Frequency
  - Weight
  - Volume
  - Type of detection
  - Wave length
  - Etc.

Available sensors

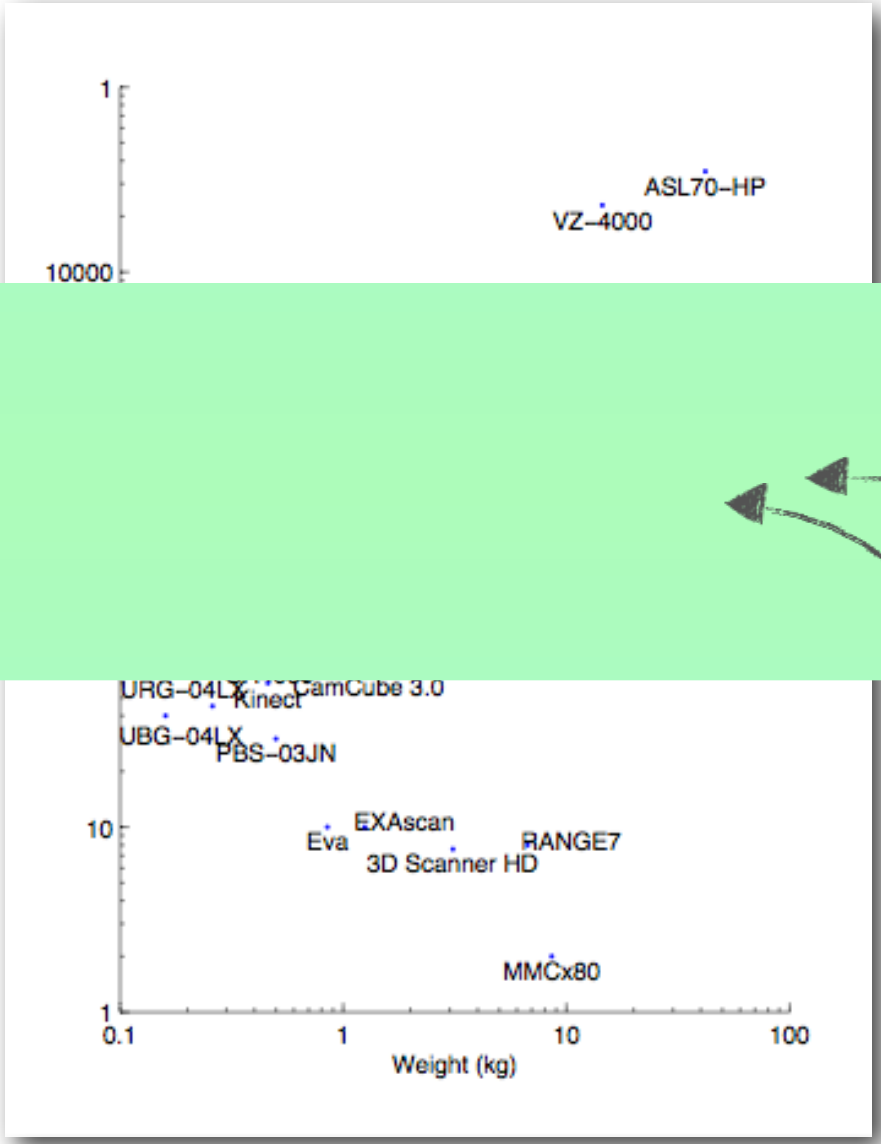


Available sensors

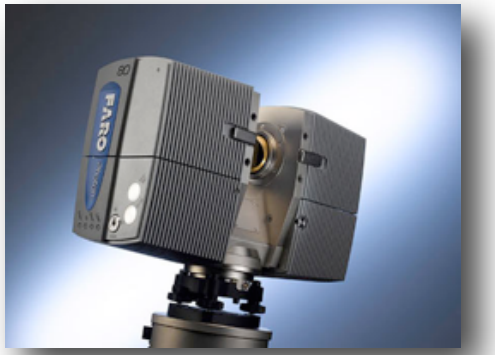


Airborne Survey

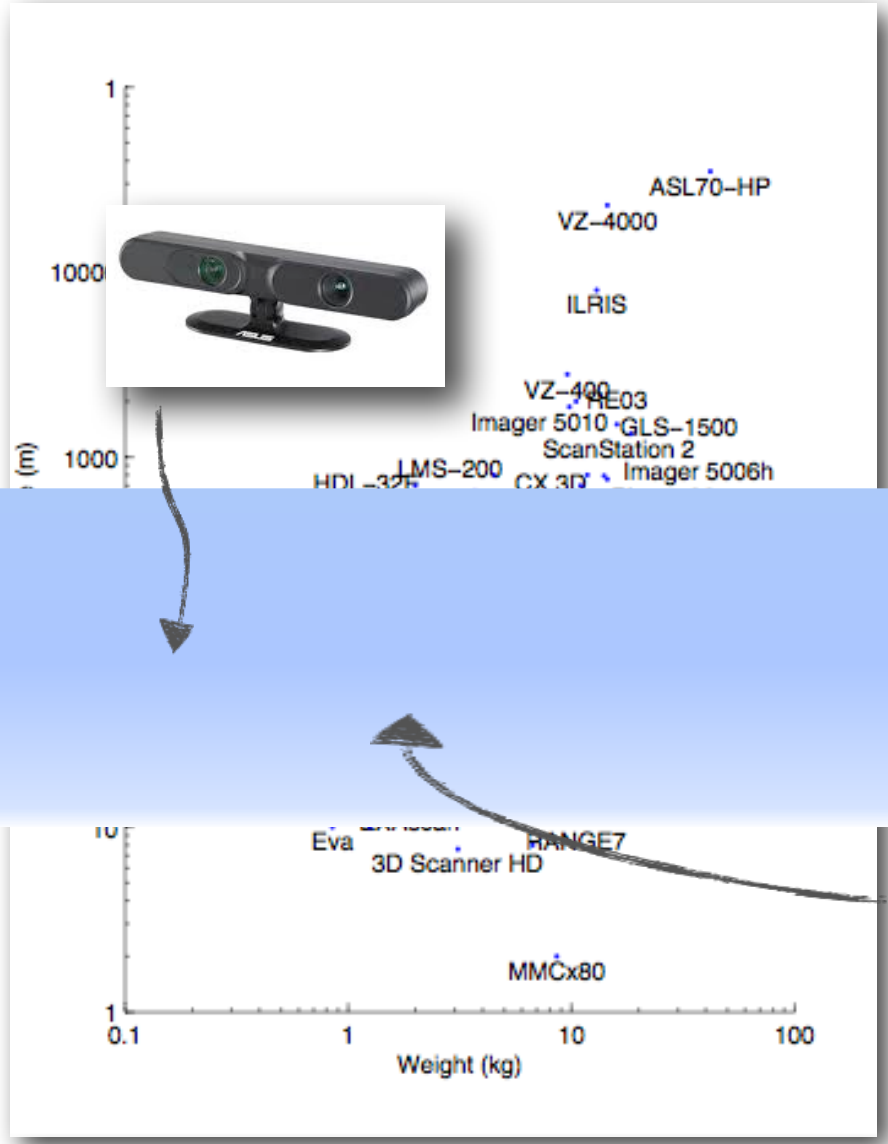
Available sensors



Terrestrial Survey



Available sensors

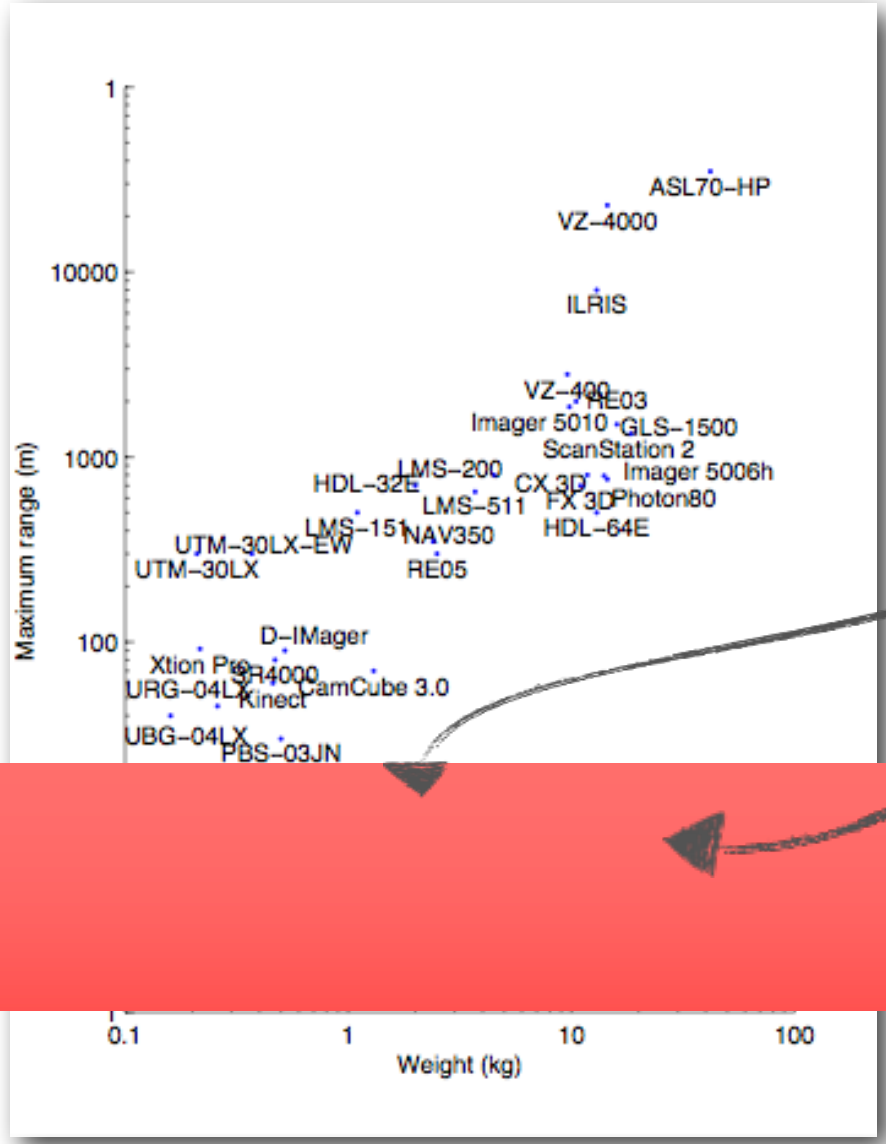


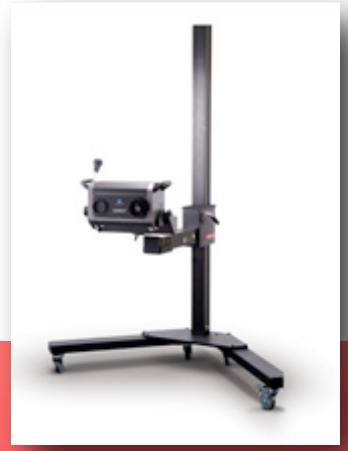
Safety

Robotics

Entertainment

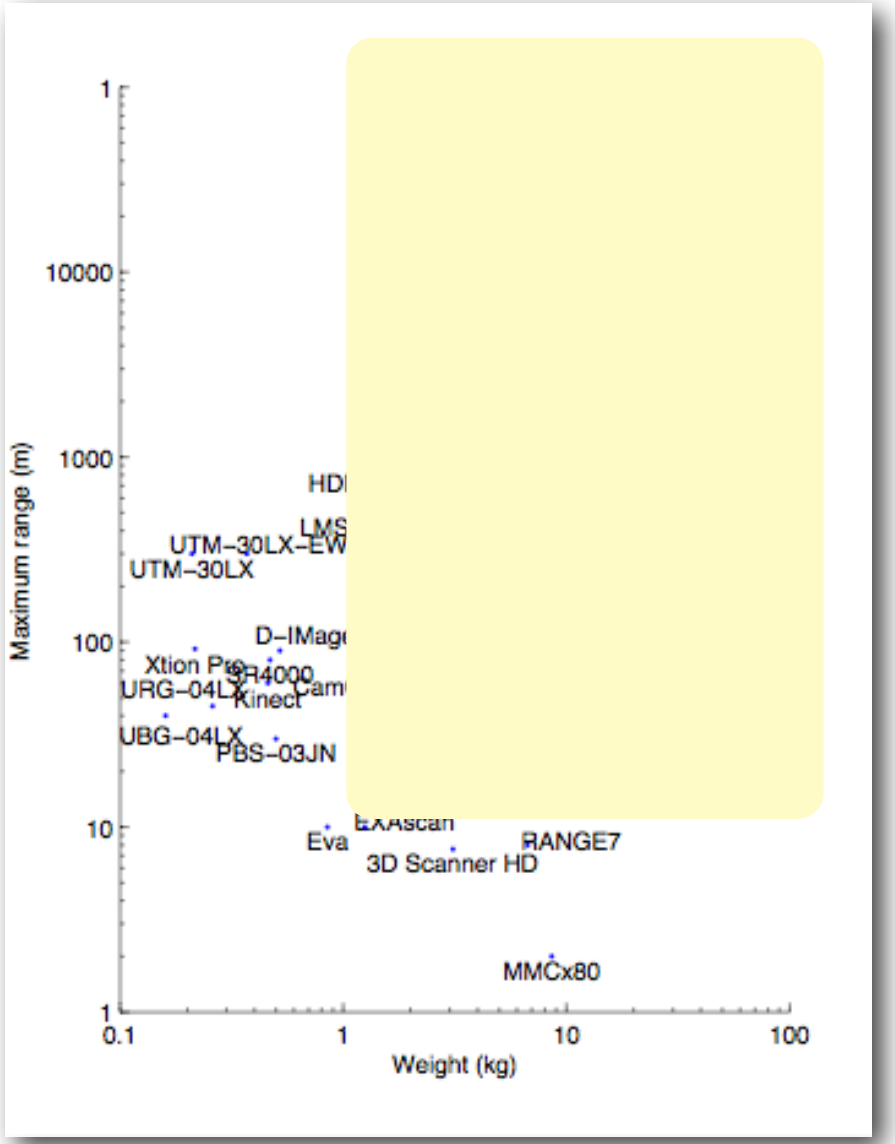
Available sensors



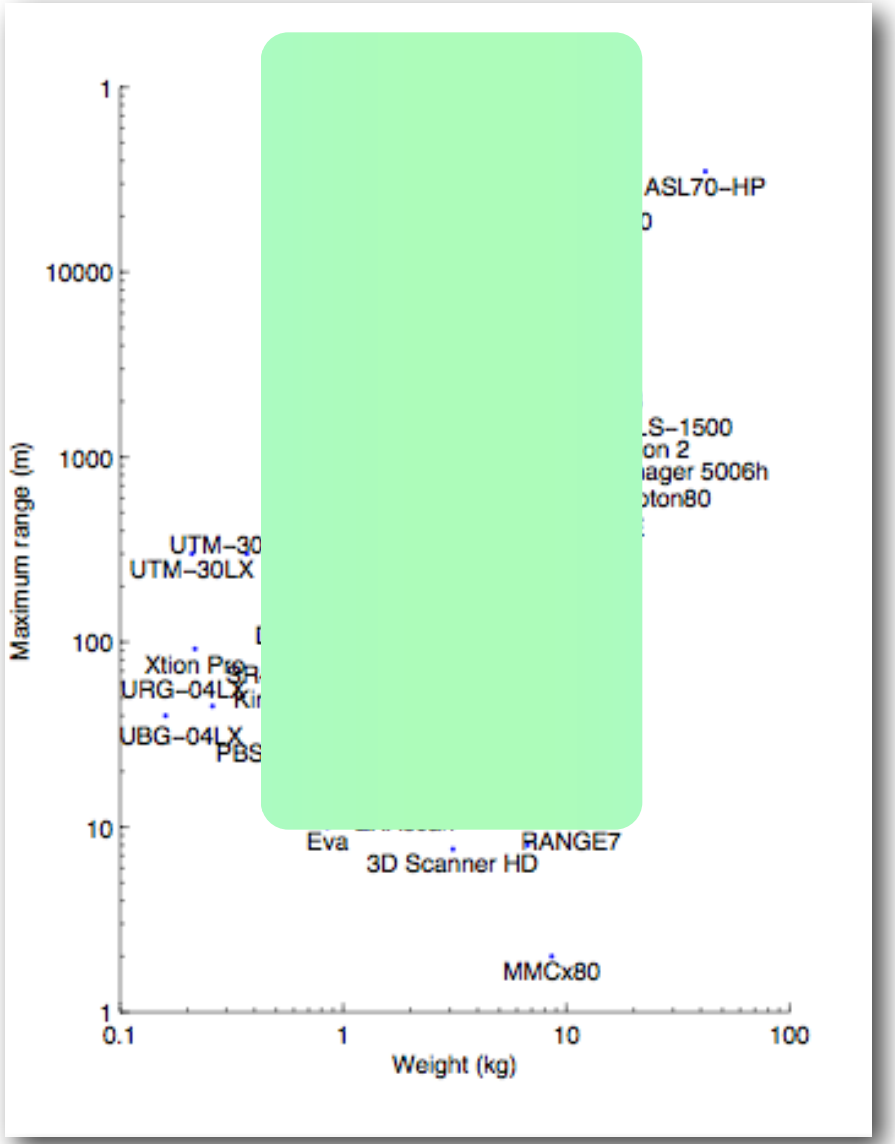


Industrial Inspection

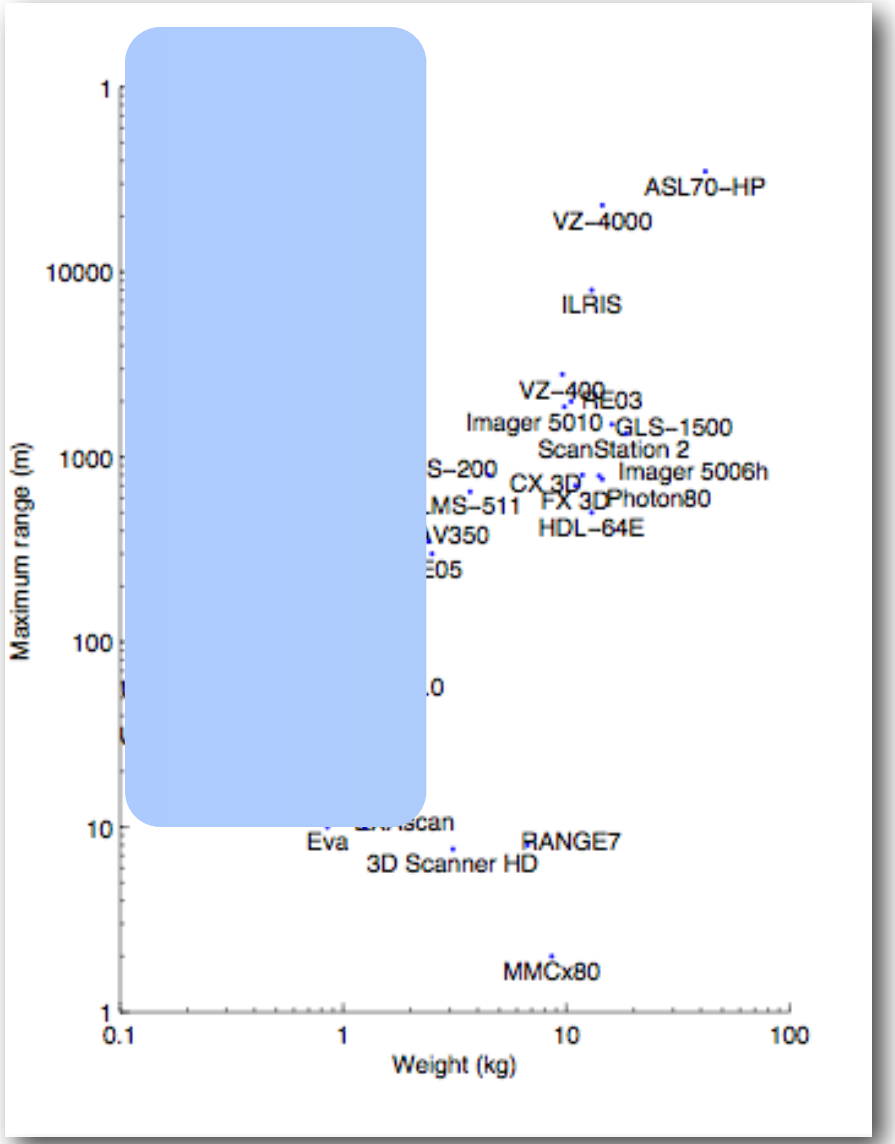
Available sensors

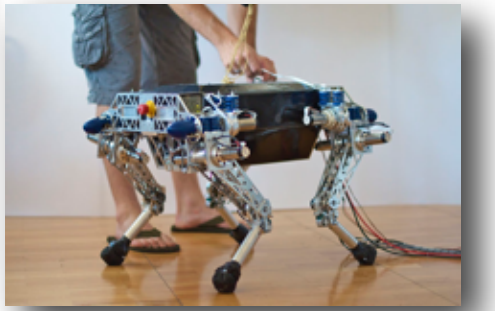


Available sensors



Available sensors

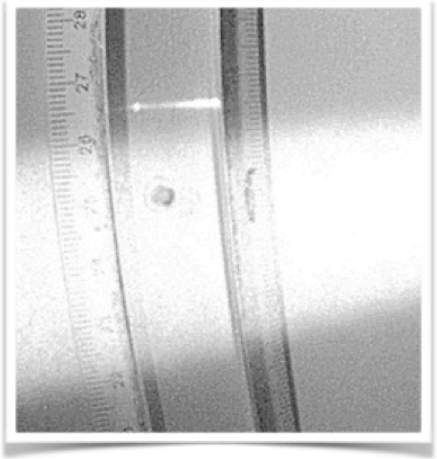




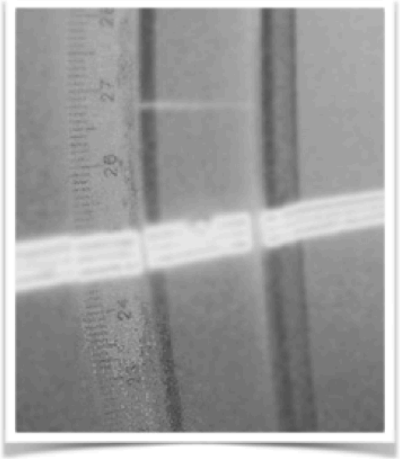
Sensors: quality criteria

- Uncertainty
  - Laser beam width
  - Depth measurement
  - Types of surfaces
- Impact on registration
  - Mixed pixels
  - Point cloud density
  - Sensor position

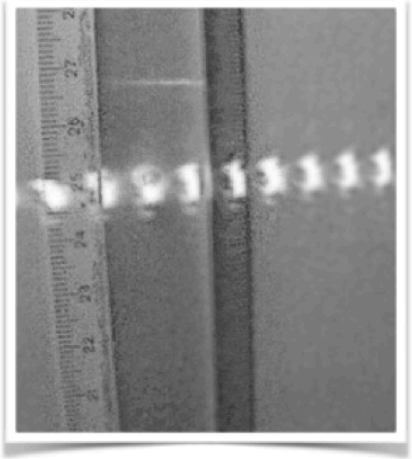
Sensor uncertainty



LMS-151: **0.83 deg**

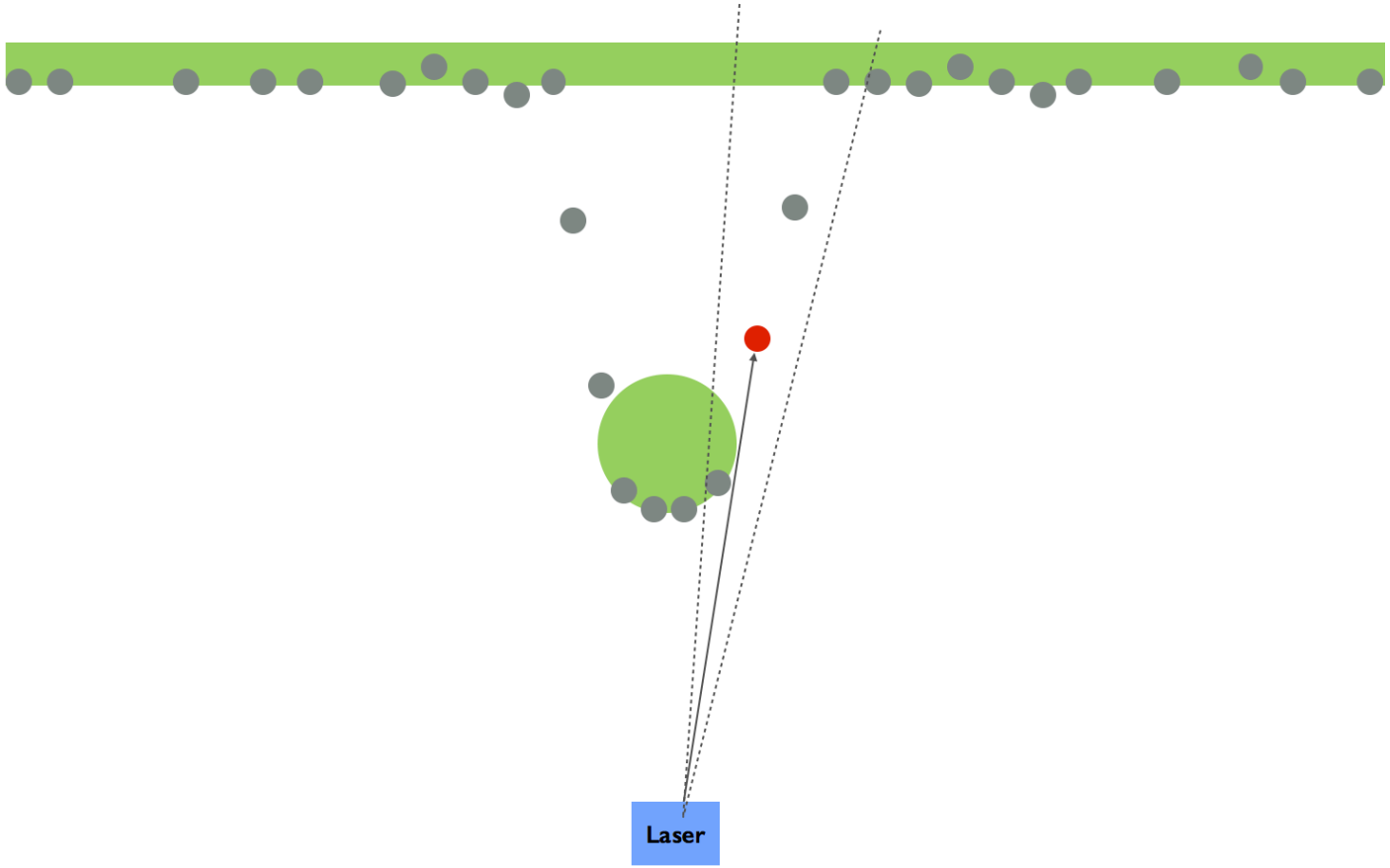


UTM-30LX: **0.13 deg**



URG-04LX: **0.14 deg**

Shadow points / mixed pixels



Shadow points / mixed pixels

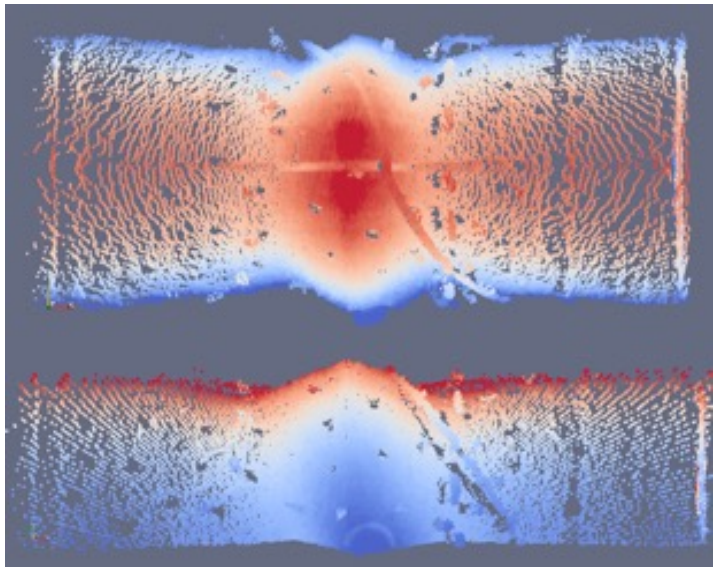
Top view of a staircase

Sensor beam

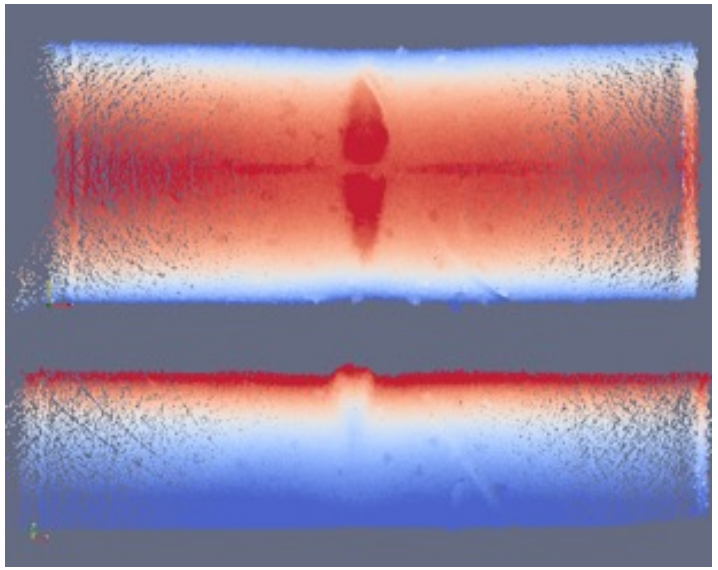
- Impact of large beam radius
  - More mixed pixels (shadow points)
  - More safety (with overlap between beams)

Types of surface

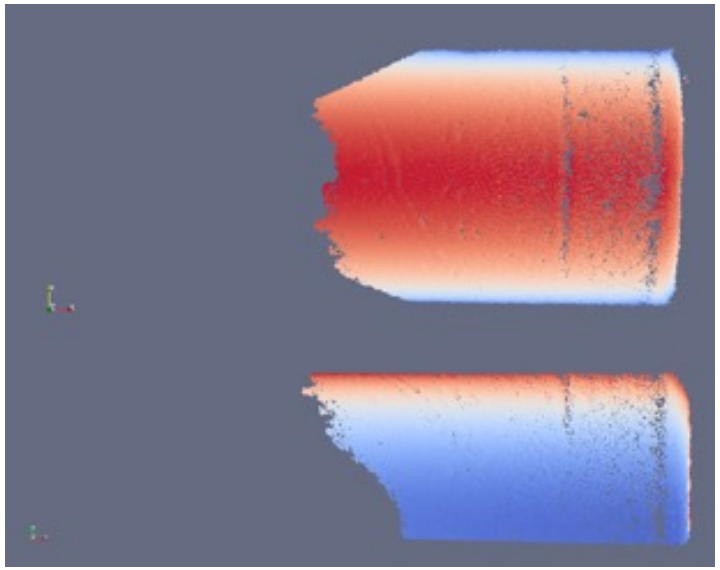
URG-04LX



UTM-30LX

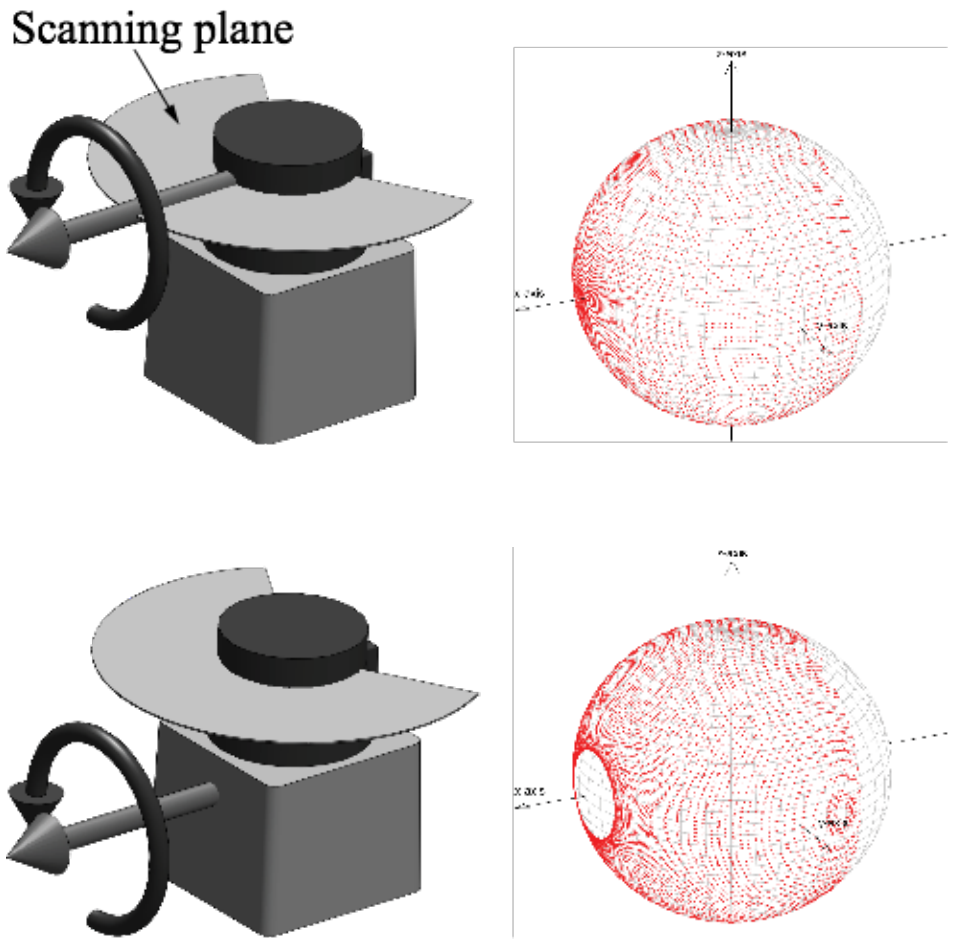


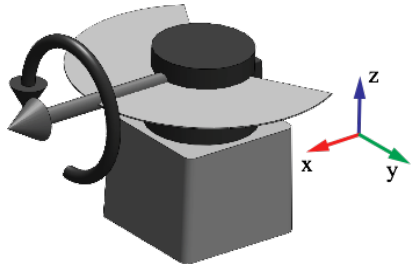
Kinect

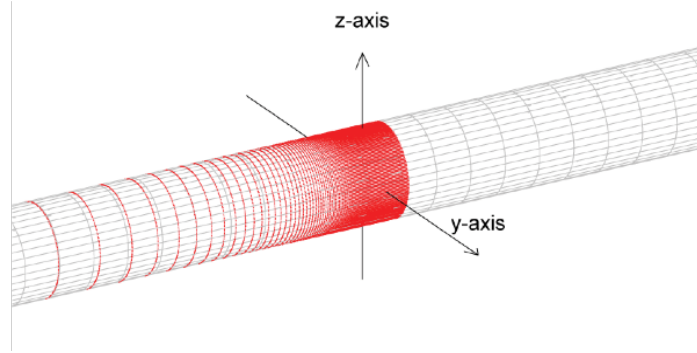


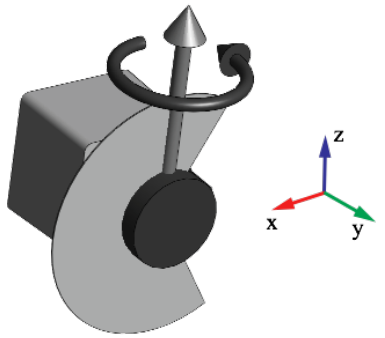
Aluminum tube

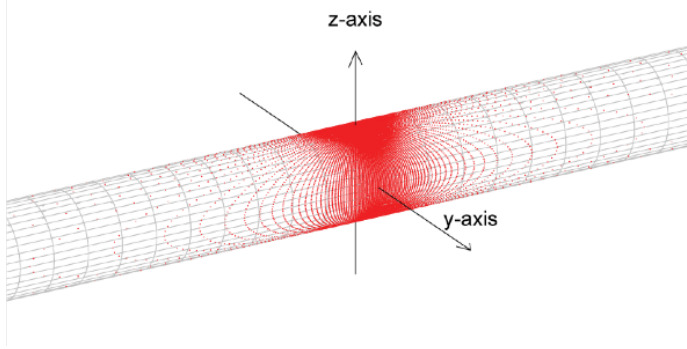
Point cloud density

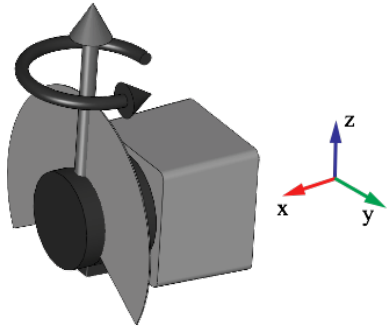


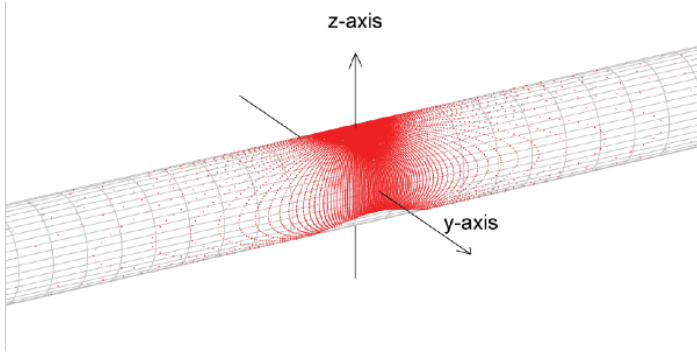




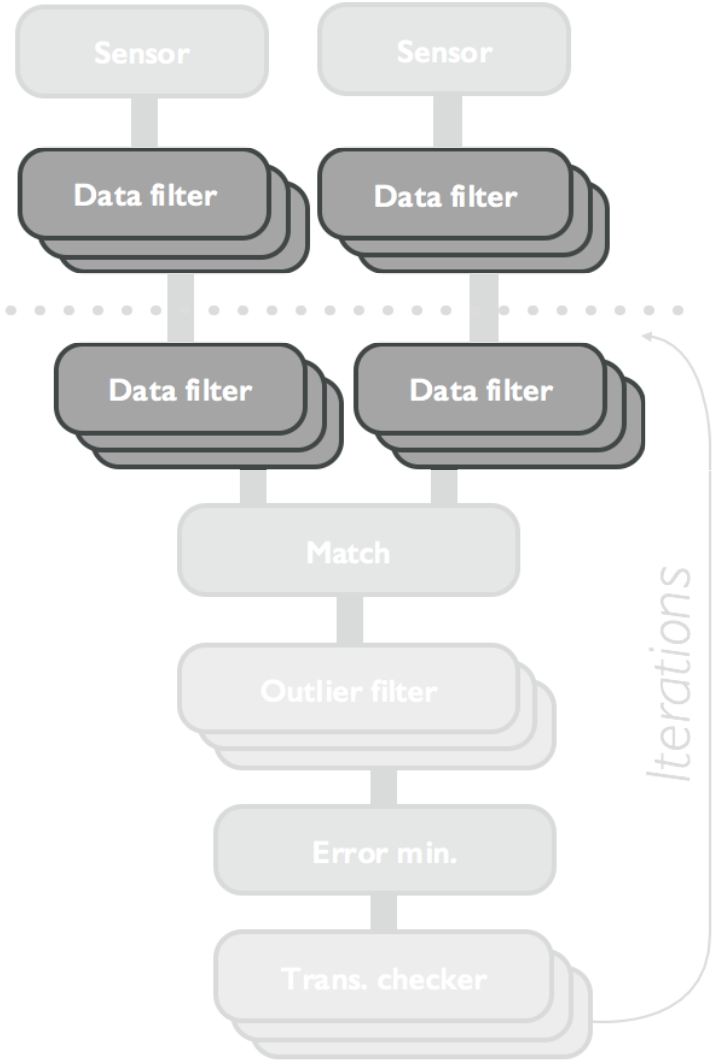






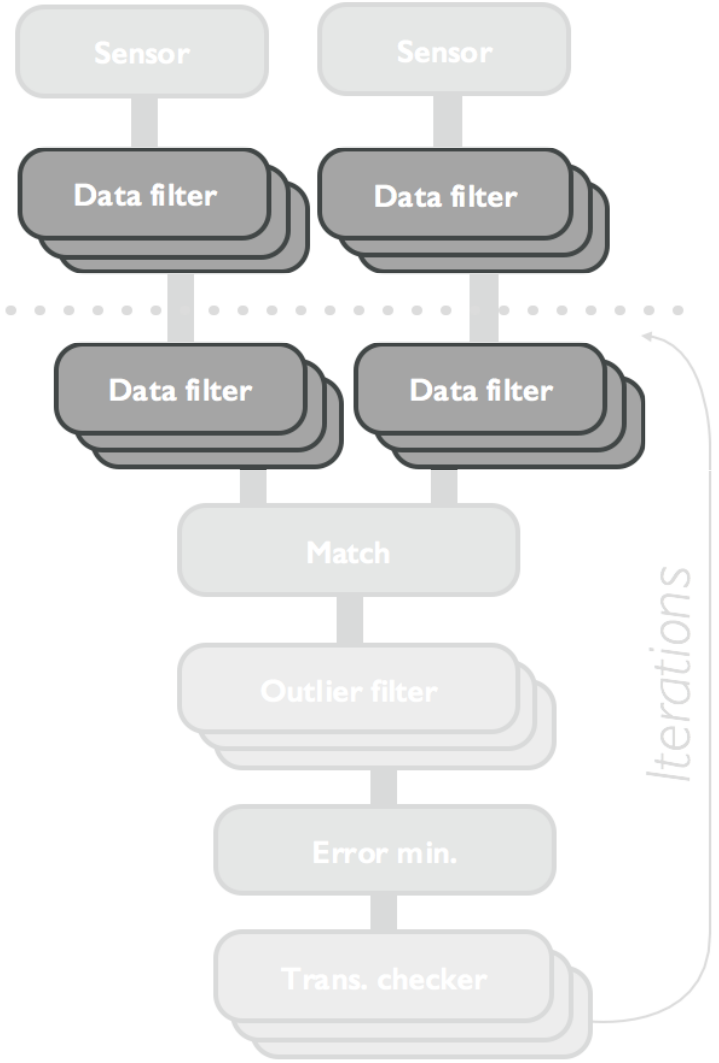


Data filters



- Main goals
  - Filter noise
  - Reduce redundant information
  - Augment discrepancy of points

Data filters



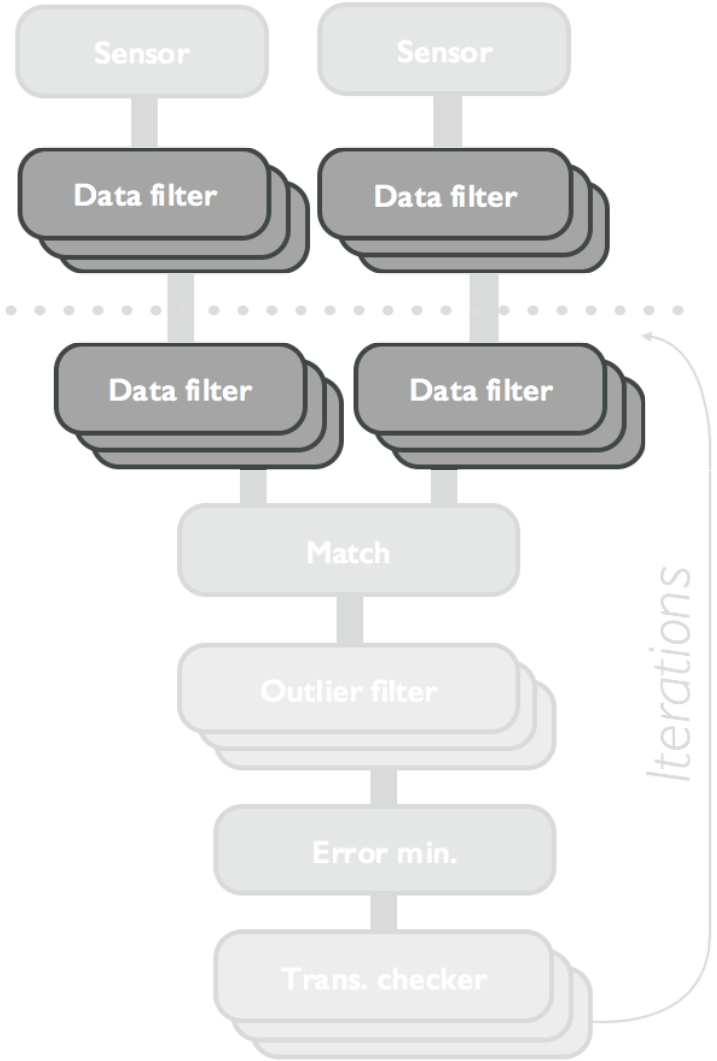
Number of points **N**

Dimension **K**

$$\begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_n \\ y_1 & y_2 & y_3 & \cdots & y_n \\ z_1 & z_2 & z_3 & \cdots & z_n \end{bmatrix}$$

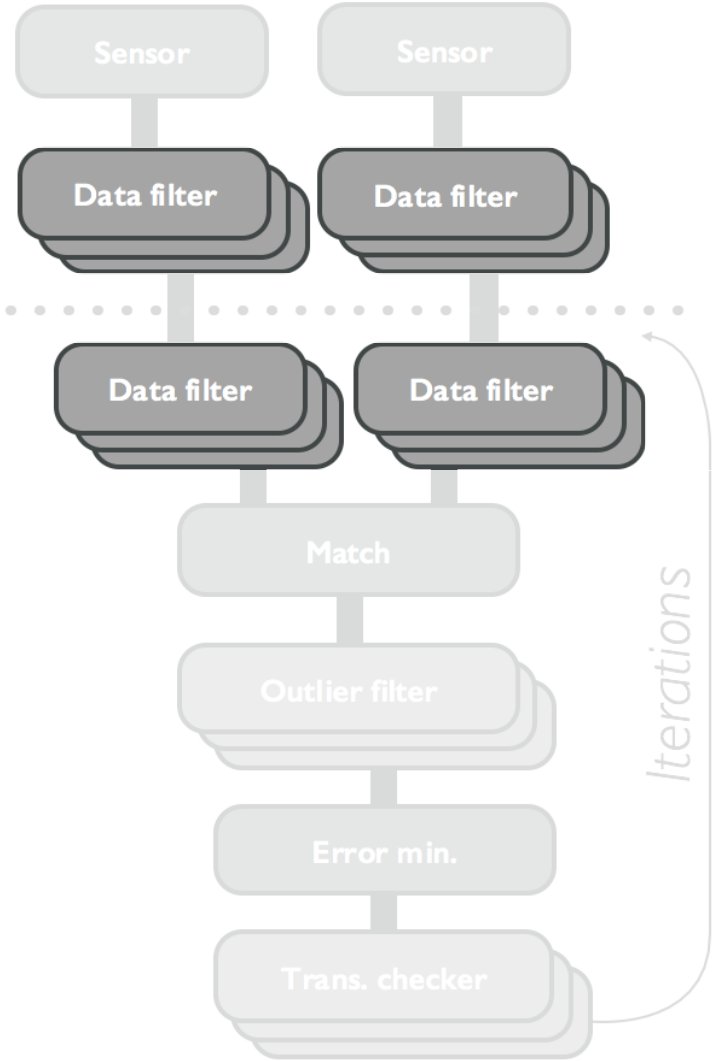
$$\begin{bmatrix} r_1 & r_2 & r_3 & \cdots & r_n \\ g_1 & g_2 & g_3 & \cdots & g_n \\ b_1 & b_2 & b_3 & \cdots & b_n \end{bmatrix}$$

Data filters



- Main goals:
  - Filter noise (reduce **N**)
  - Reduce redundant information (reduce **N**)
  - Augment discrepancy of points (augment **K**)

Data filters



- Examples:
  - Point reduction
  - Sensor noise weight
  - Surface extraction

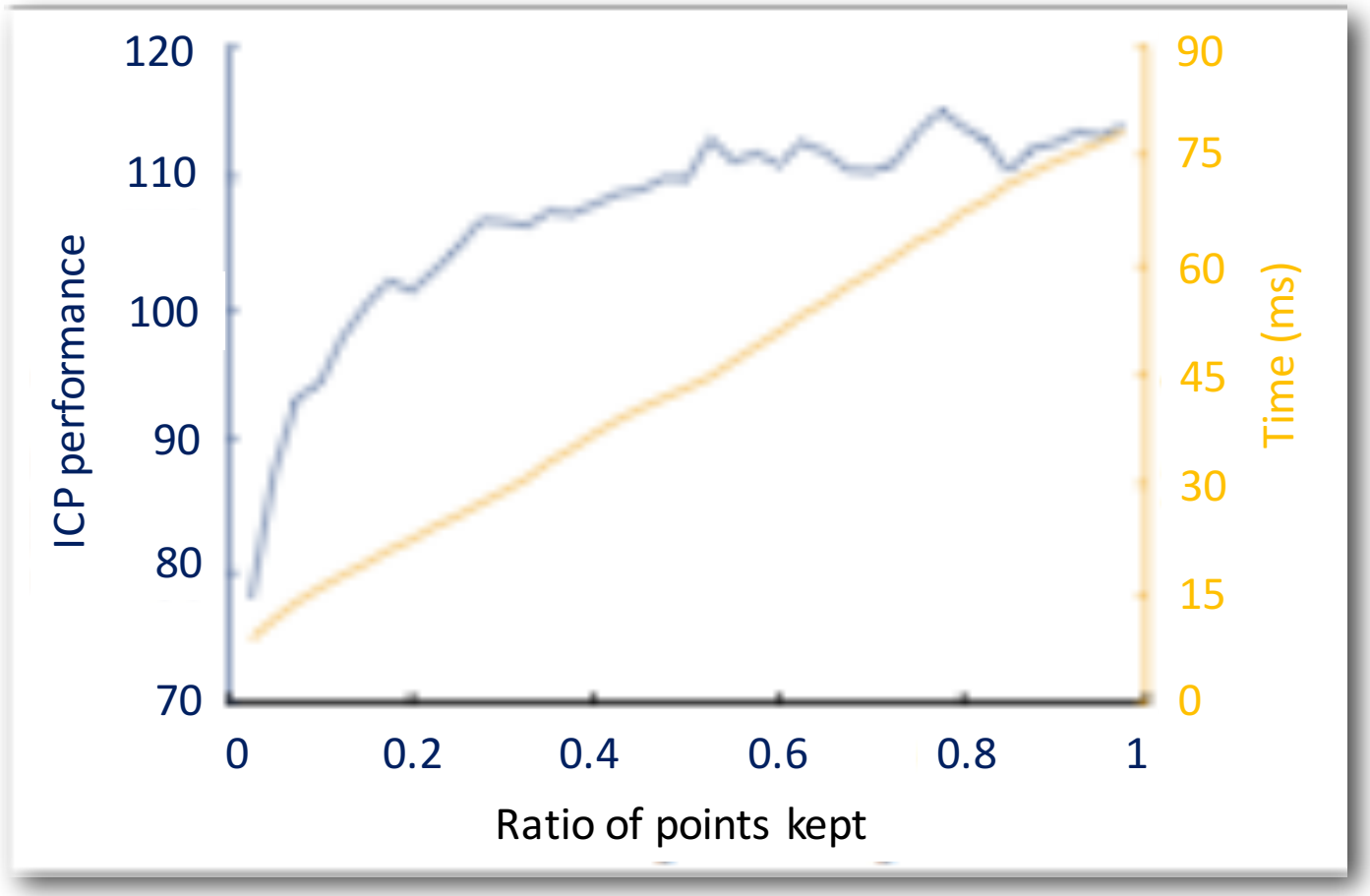
Point reduction

- Fixed-step subsampling
  - Very fast
  - May generate artifact
- Random sub-sampling
  - Fast
  - More control on reduction factor
- Density
  - Slow (needs a kD-tree)
  - Reduce minimization local minima
- Geometric sampling [Gelfand et al. 2003]
  - Very slow
  - Reduce more minimization local minima

Point reduction

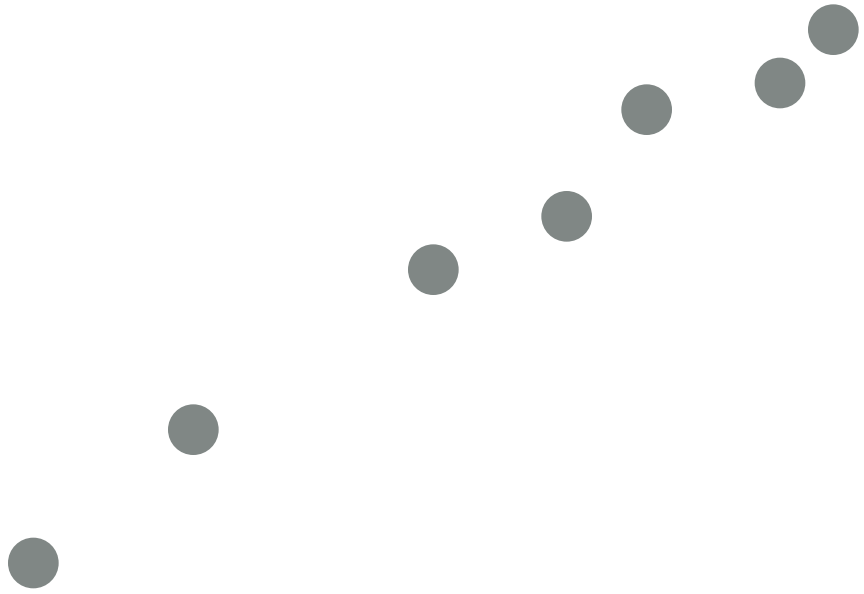
- Random subsampling

N = 40,230



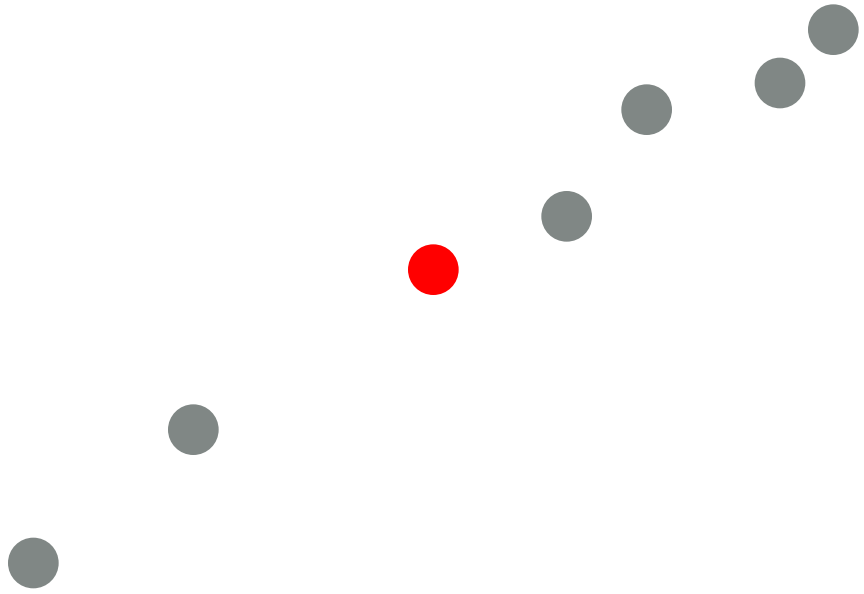
Surface extraction

- For each point

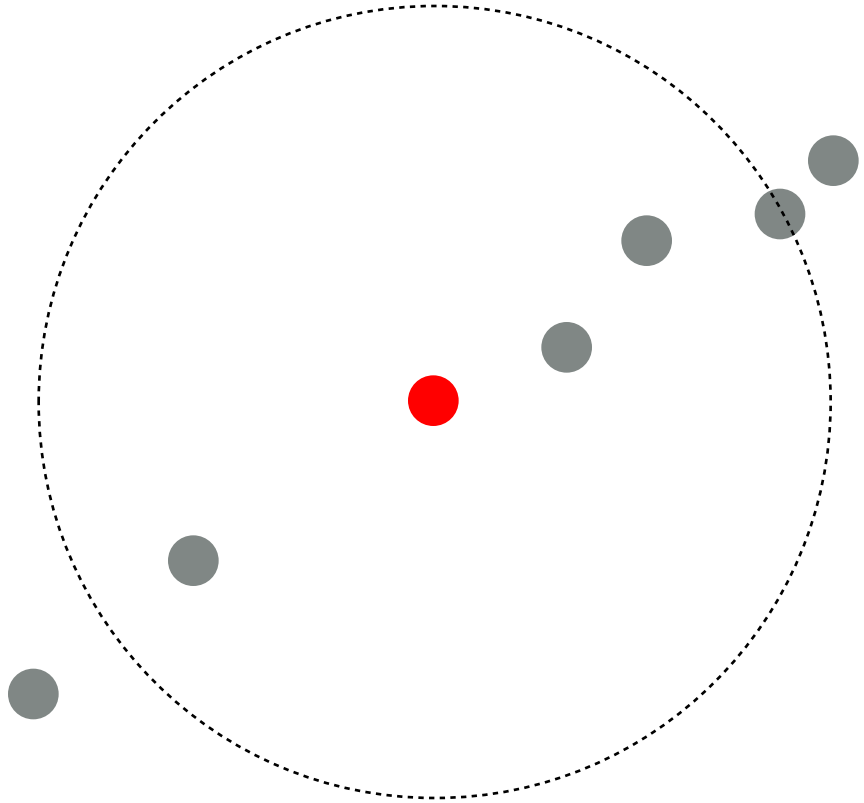


Surface extraction

- For each point

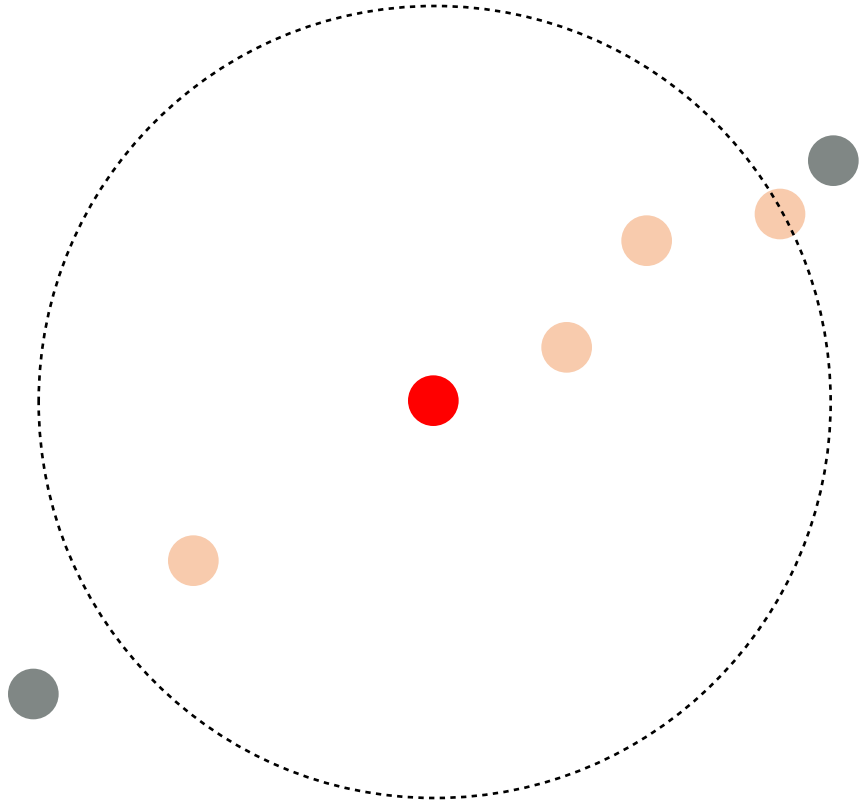


Surface extraction



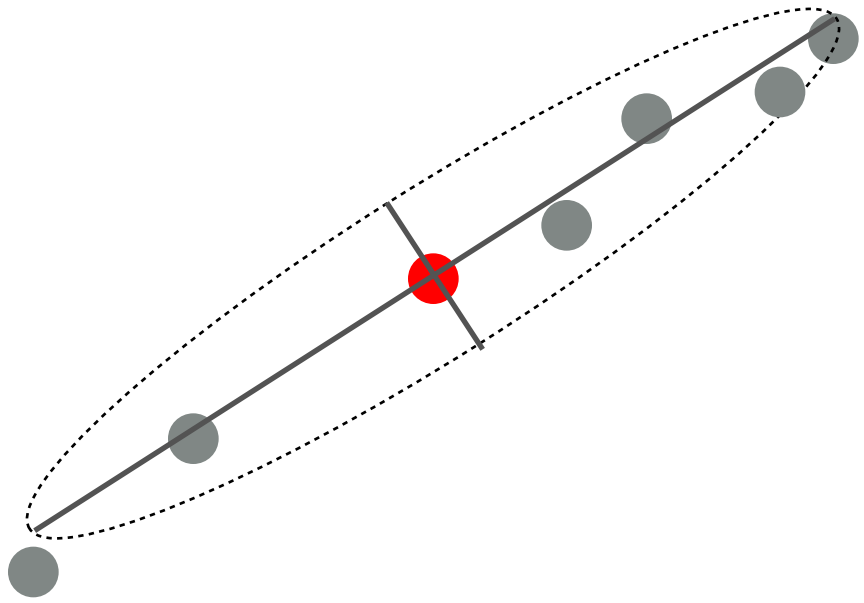
- For each point
  - Define neighborhood

Surface extraction



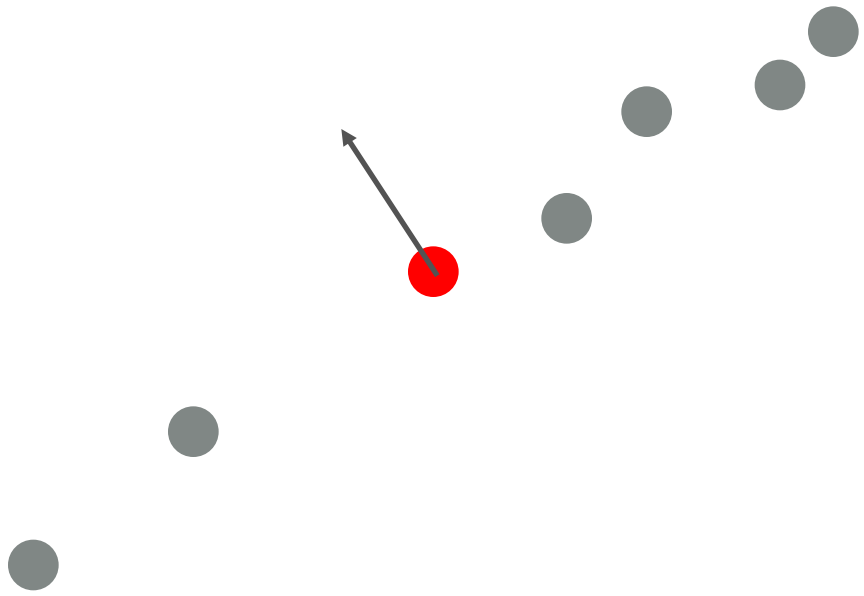
- For each point
  - Define neighborhood
  - Subtract “center”

Surface extraction



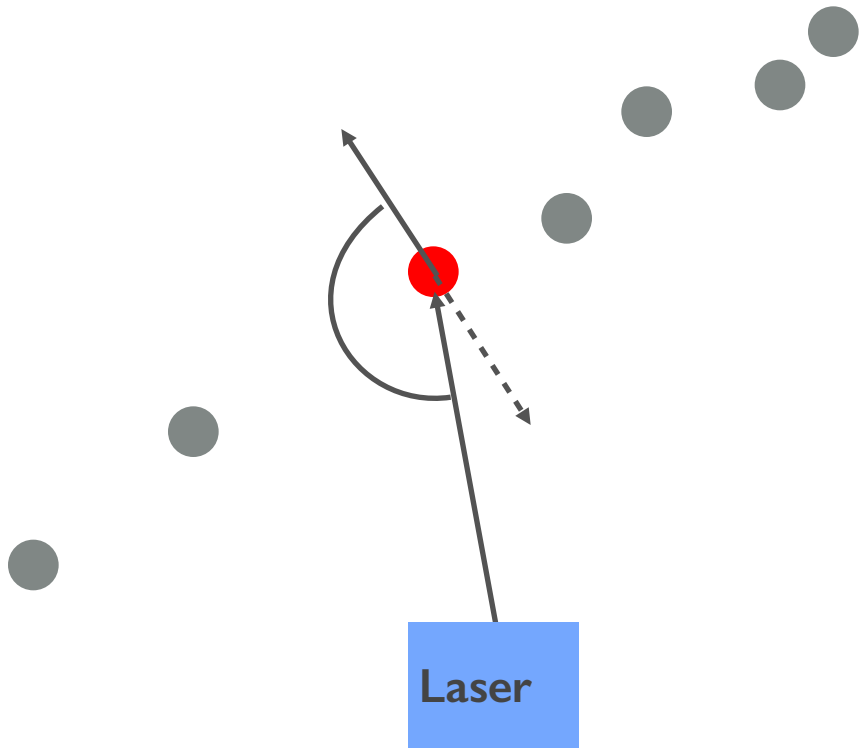
- For each point
  - Define neighborhood
  - Subtract “center”
  - Eigenvalue decomposition

Surface extraction



- For each point
  - Define neighborhood
  - Subtract “center”
  - Eigenvalue decomposition
  - Select the eigenvector associated with the smallest Eigenvalue

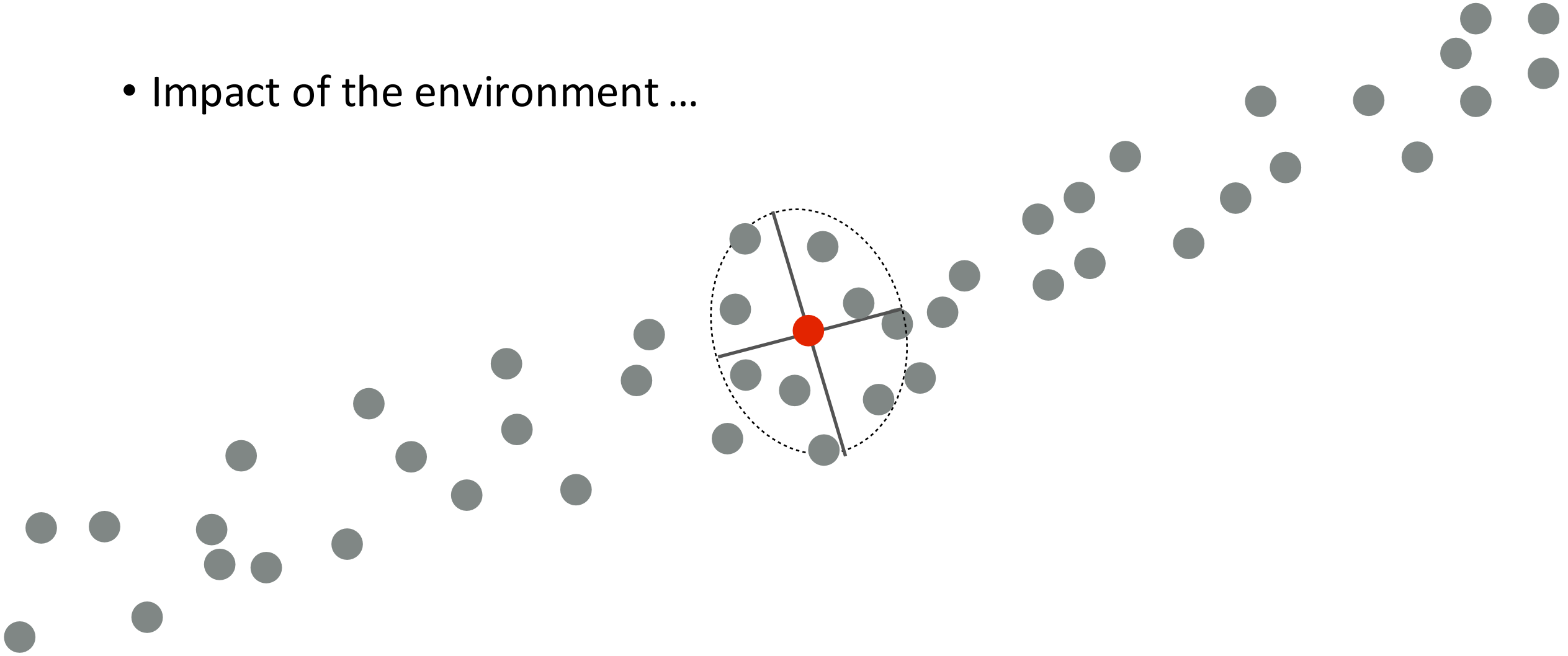
Surface extraction



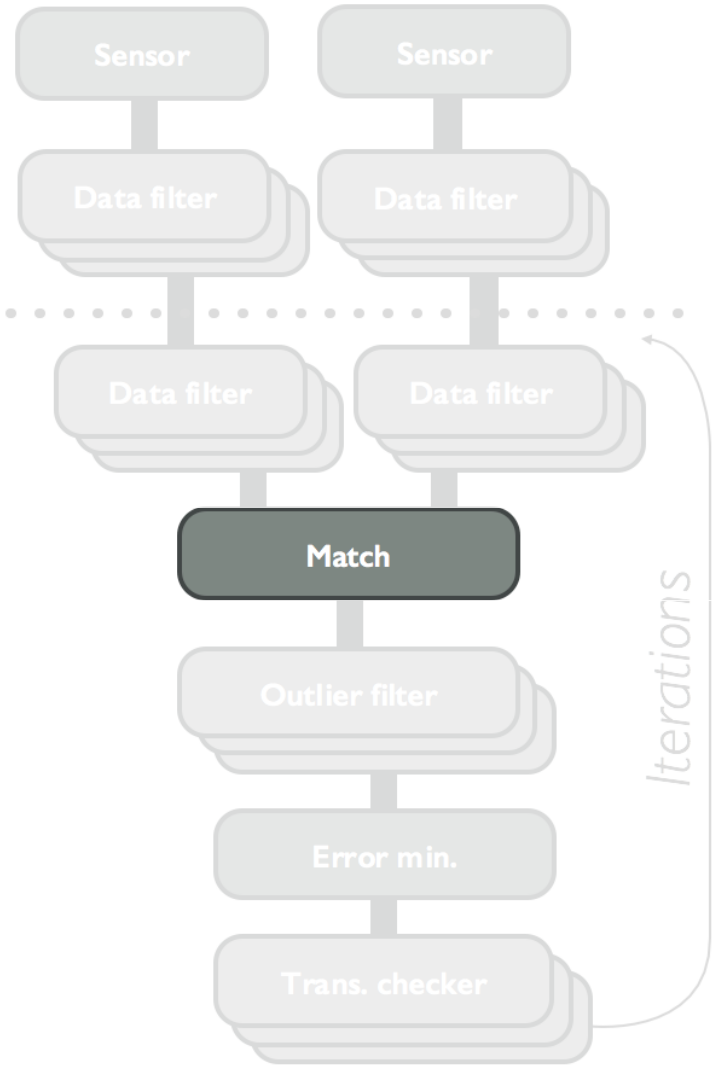
- For each point
  - Define neighborhood
  - Subtract “center”
  - Eigenvalue decomposition
  - Select the eigenvector associated with the smallest Eigenvalue
- Bonus:
  - Reorient the vector

Surface extraction

- Impact of the environment …



Matching



- Main goal
  - Augment robustness to misalignment

Matching

- Match direction:
  - From reading to reference
  - From reference to reading
  - Both direction

Matching

- Match direction:
  - From reading to reference
  - From reference to reading
  - Both direction
- Number of matches:
  - One-to-one (typical)
  - One-to-many
    (slower but more robust)

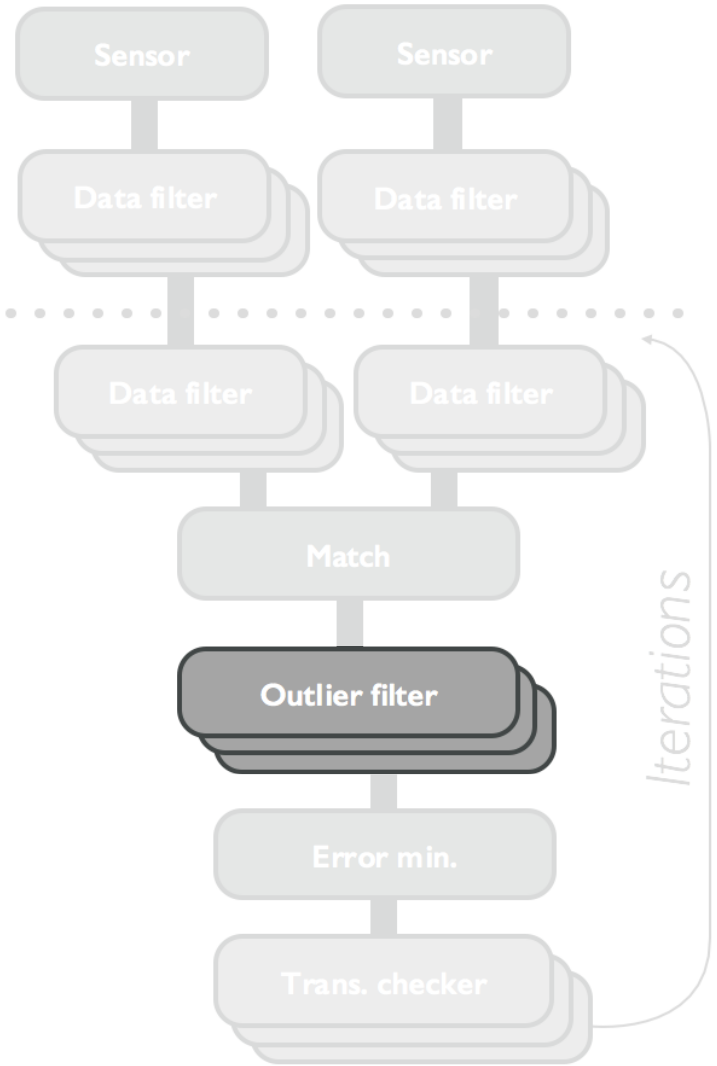
Matching

- Match direction:
  - From reading to reference
  - From reference to reading
  - Both direction
- Number of matches:
  - One-to-one (typical)
  - One-to-many
    (slower but more robust)

- Metric
  - Euclidean distance on points
  - Euclidean distance on points and descriptors
  - Euclidean distance on descriptors
  - Mahalanobis distance
  - Probablistic distance (entropy)
  - Others

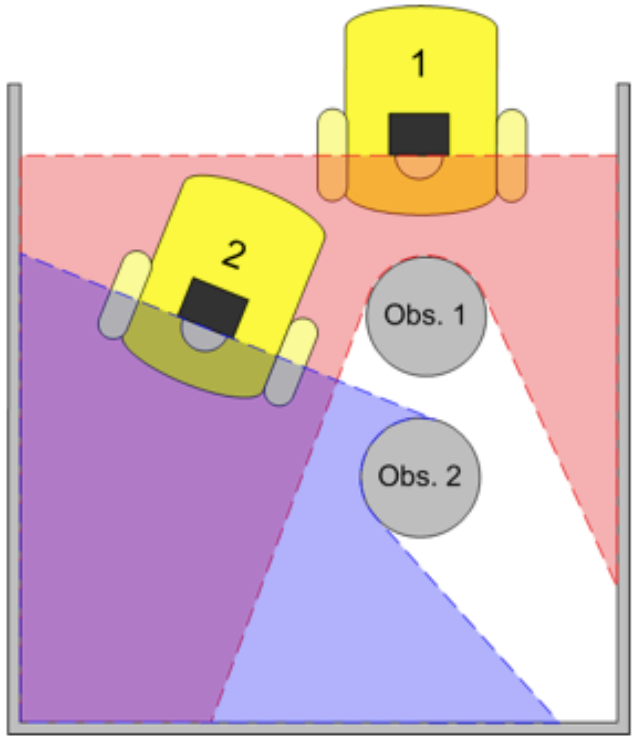
kD-tree

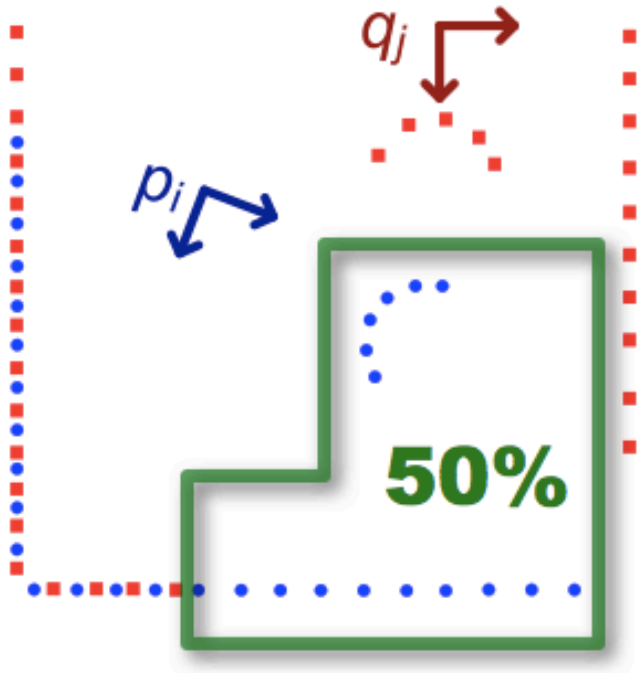
Robust regression



- Main goal
  - Augment robustness to overlap
  - Augment robustness to noise
  - Augment robustness to dynamic elements

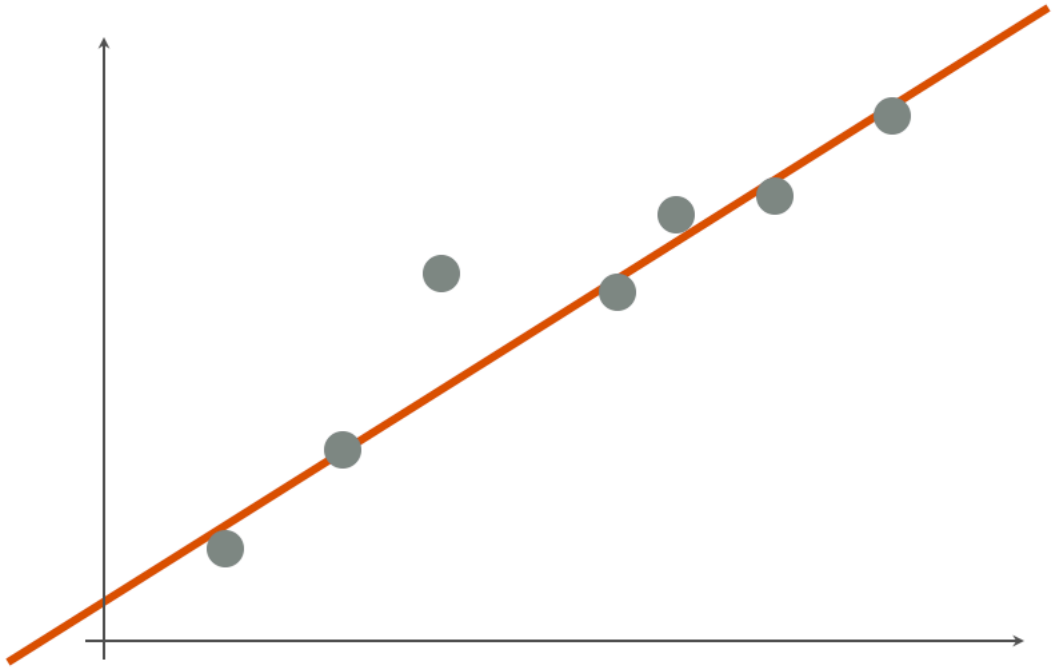
Robustness in case of partial overlaps





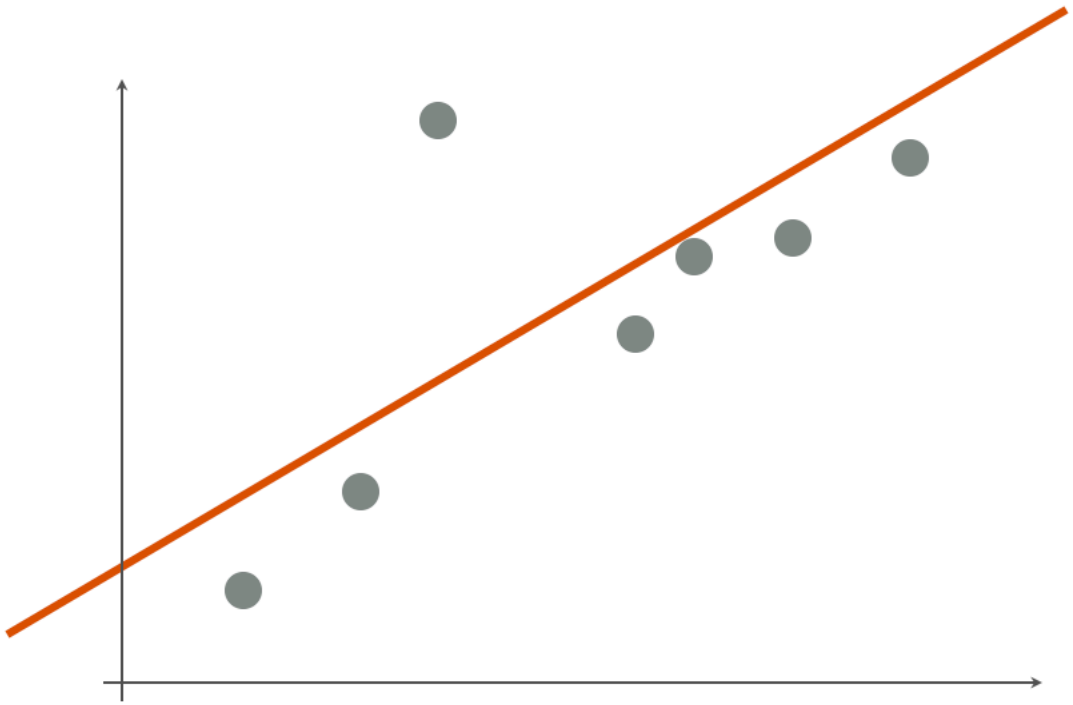
Robust regression

Why mean least-squared is sensible to outlier?



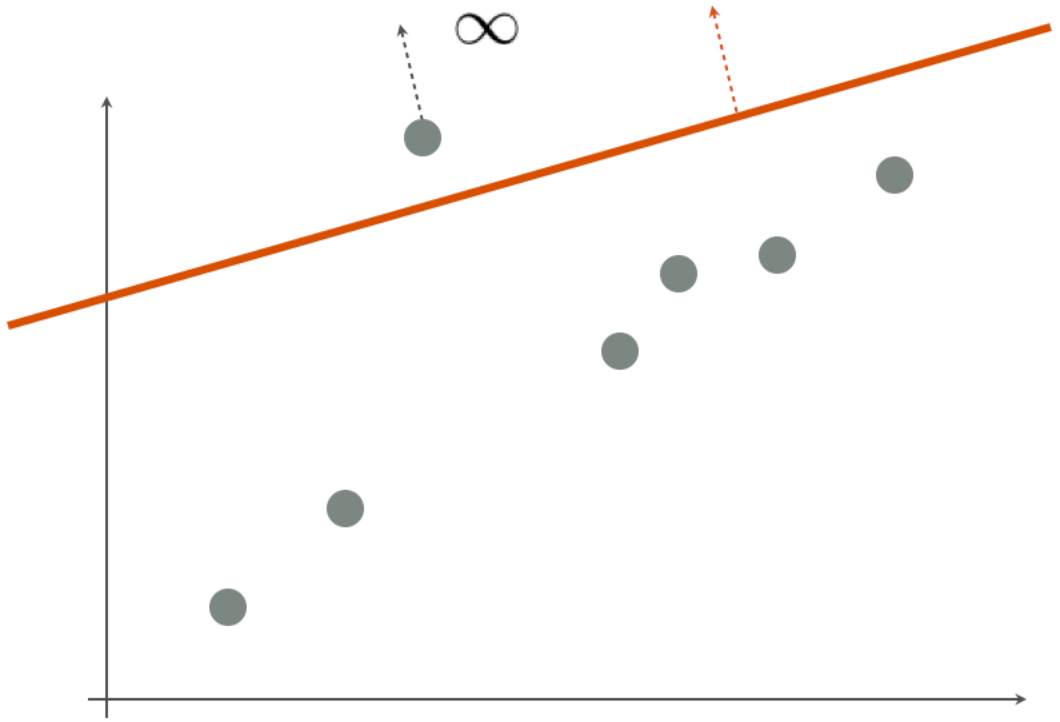
Robust regression

Why mean least-squared is sensible to outlier?



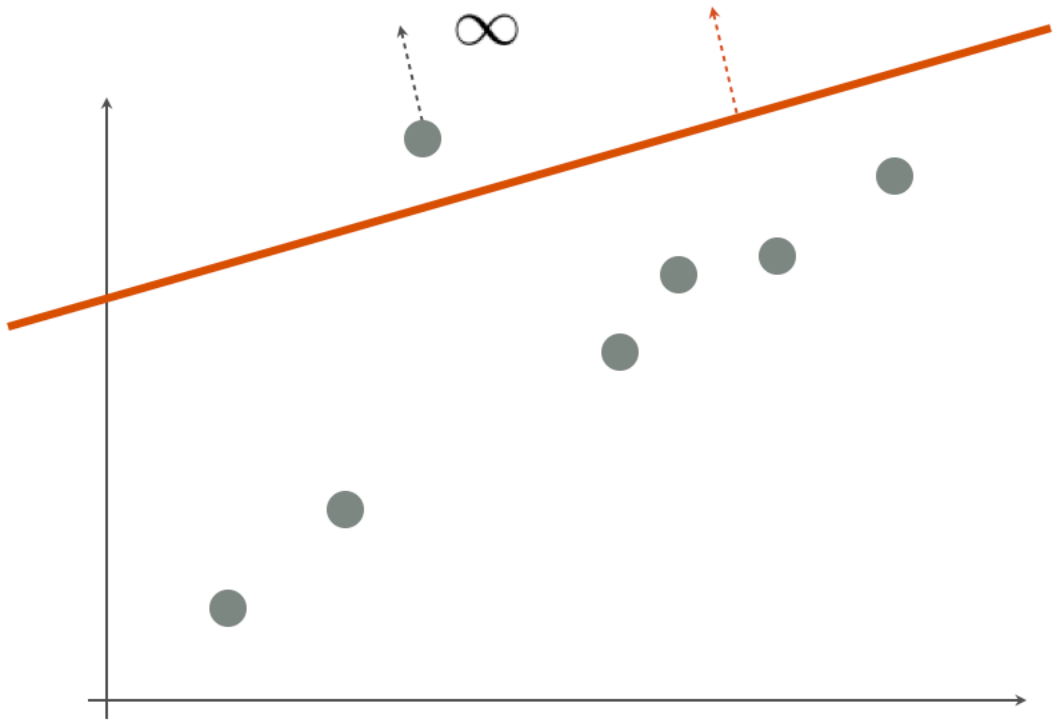
Robust regression

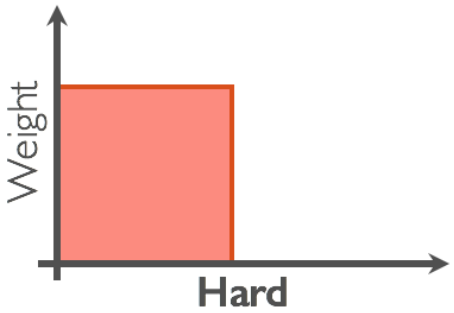
Why mean least-squared is sensible to outlier?

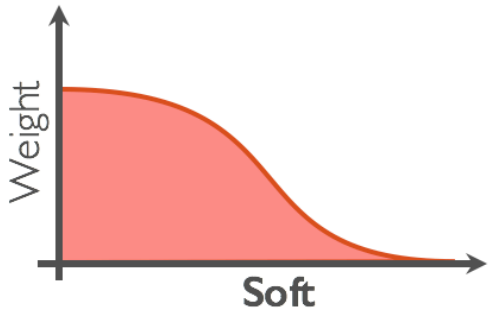


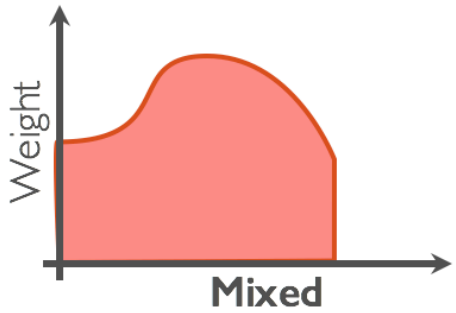
Robust regression

Why mean least-squared is sensible to outlier?









Rejection

- **Fix**

$$d_i < d_{max}$$

- Zhang
- Mean
- Median
- Trimmed

Rejection

- Fix
- **Zhang** *(Zhang 94)*

$$d_i < \begin{cases} \mu + 3\sigma, & \text{if } 0 \leq \mu < \eta \\ \mu + 2\sigma, & \text{if } \eta < \mu \leq 3\eta \\ \mu + \sigma, & \text{if } 3\eta < \mu \leq 6\eta \\ median, & \text{otherwise} \end{cases}$$

- Mean
- Median
- Trimmed

Rejection

- Fix
- Zhang
- **Mean** *(Druon 06)*

$$d_i < \mu + \sigma$$

- Median
- Trimmed

Rejection

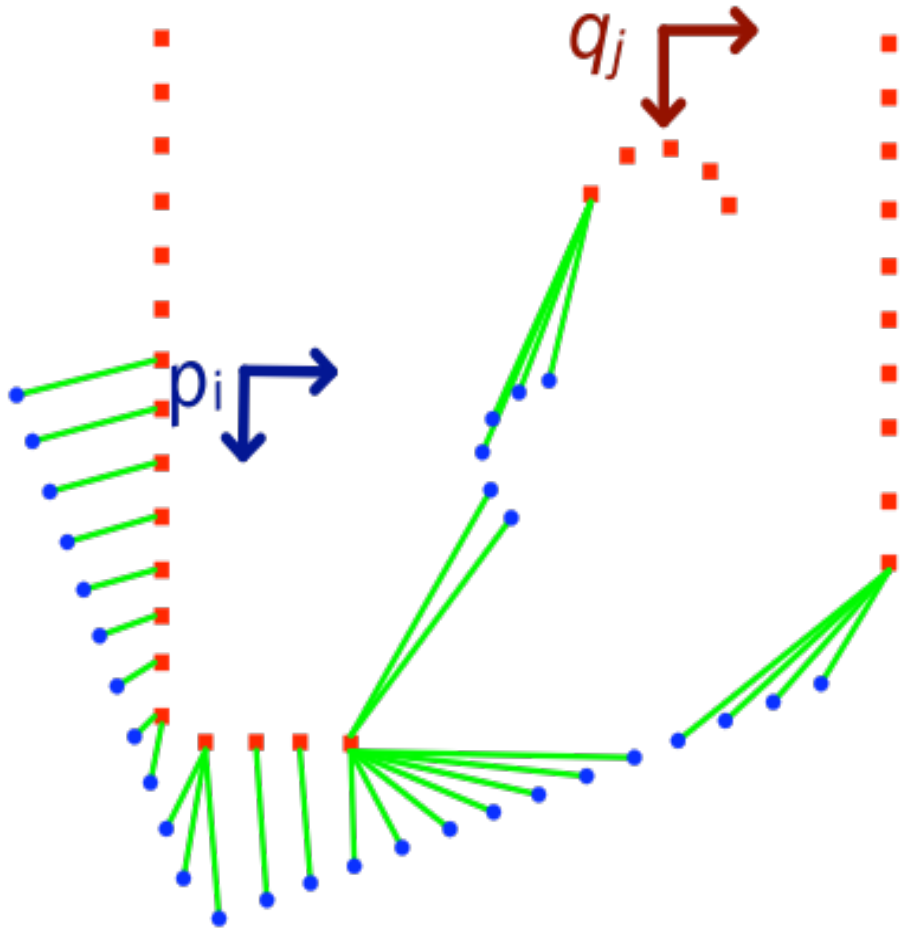
- Fix
- Zhang
- Mean
- **Median** *(Diebel 04)*

$$d_i < 3 \times median$$

- Trimmed

Rejection

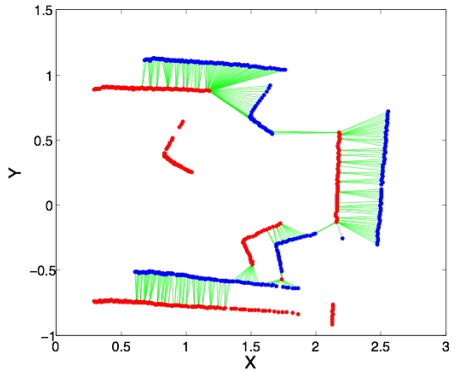
- Fix
- Zhang *(Zhang 94)*
- Mean *(Druon 06)*
- Median *(Diebel 04)*
- Trimmed *(Chetverikov 02)*
  - *remove "x percent" of worst matches*

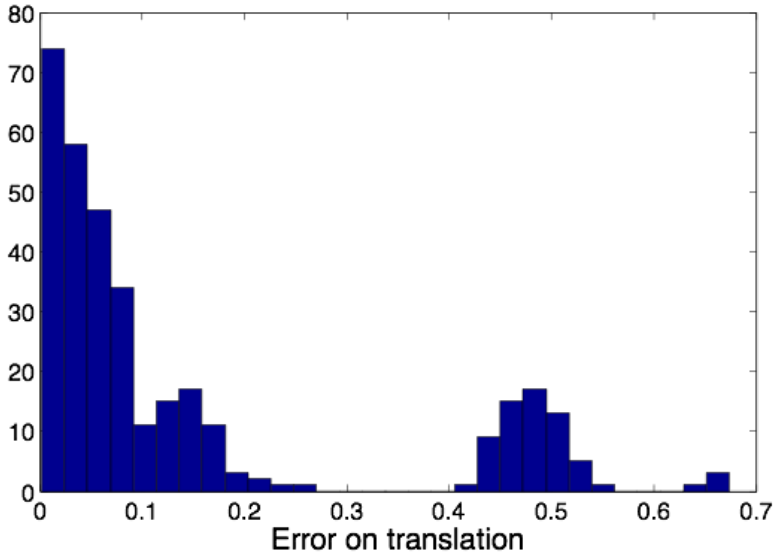


Rejection

- Fix
- Zhang *(Zhang 94)*
- Mean *(Druon 06)*
- Median *(Diebel 04)*
- Trimmed *(Chetverikov 02)*

Assume unimodal distribution

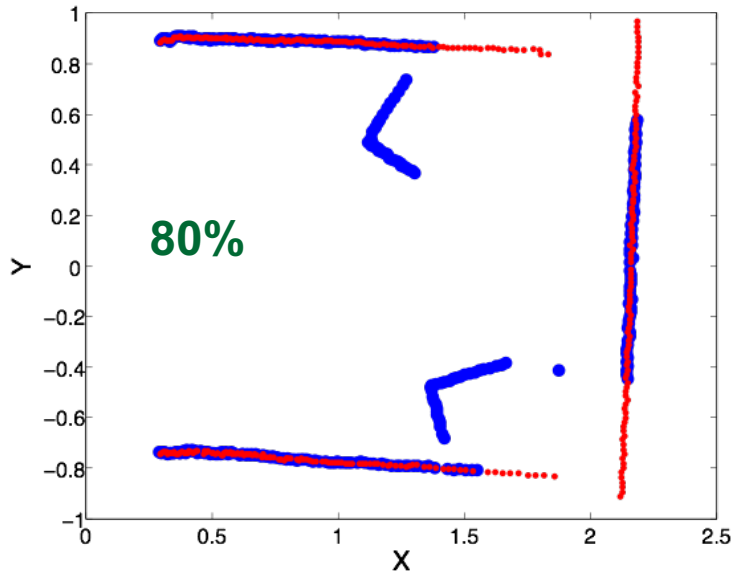


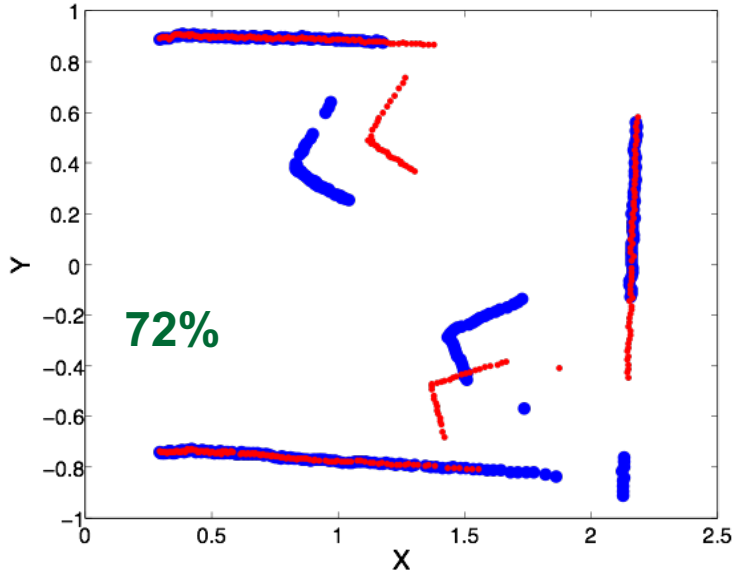


Rejection

- Fix
- Zhang *(Zhang 94)*
- Mean *(Druon 06)*
- Median *(Diebel 04)*
- Trimmed *(Chetverikov 02)*

Assume few variation in overlapping ratio





Rejection

- Fix
- Zhang *(Zhang 94)*
- Mean *(Druon 06)*
- Median *(Diebel 04)*
- Trimmed *(Chetverikov 02)*

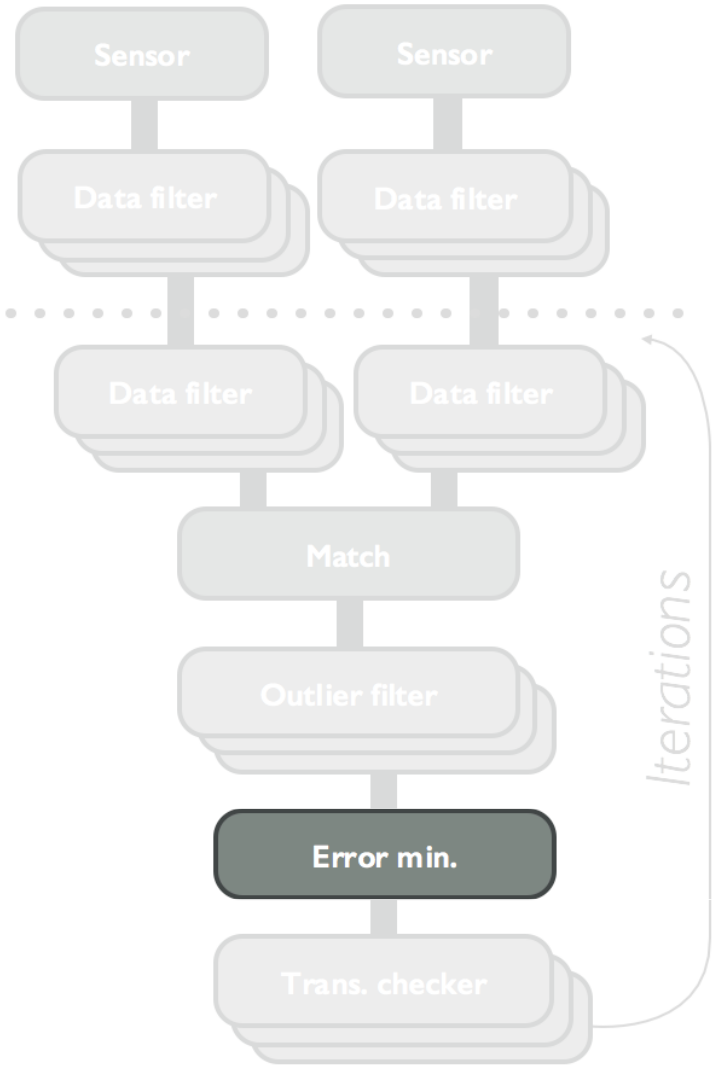
Require sorting

Rejection

- Fix
- Zhang *(Zhang 94)*
- Mean *(Druon 06)*
- Median *(Diebel 04)*
- Trimmed *(Chetverikov 02)*

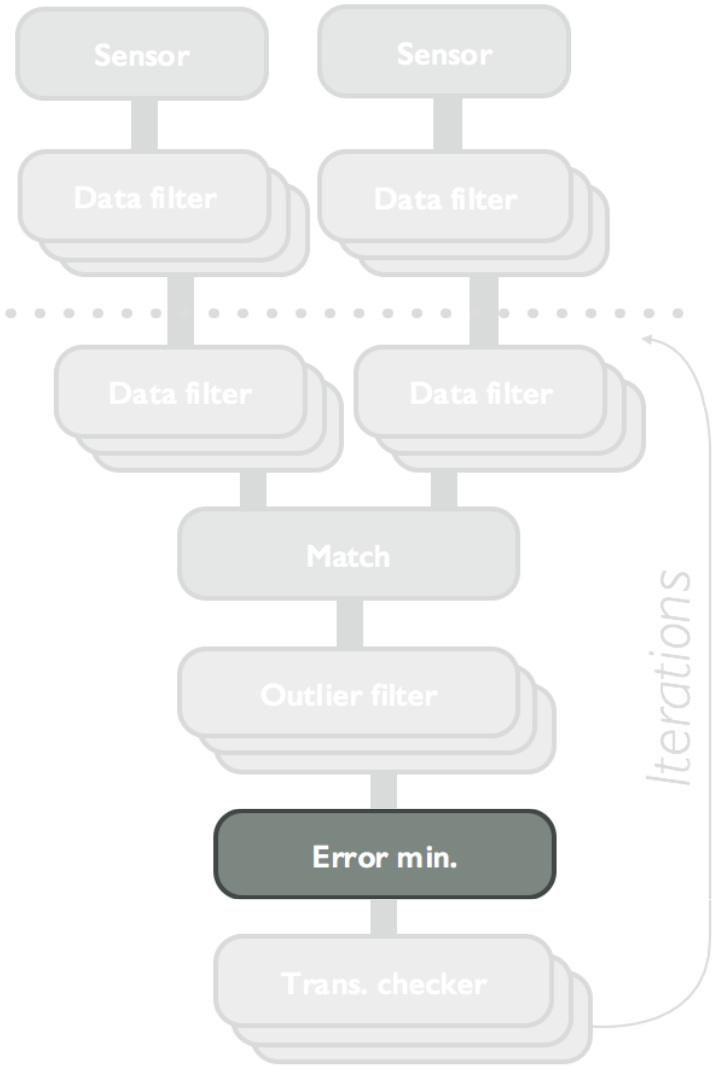
- Probabilistic *(Feldmar 96) (Philips 07)*
- RANSAC *(Arun 87)*

Minimization



- Main goal
  - Rapid convergence
  - Well constrained

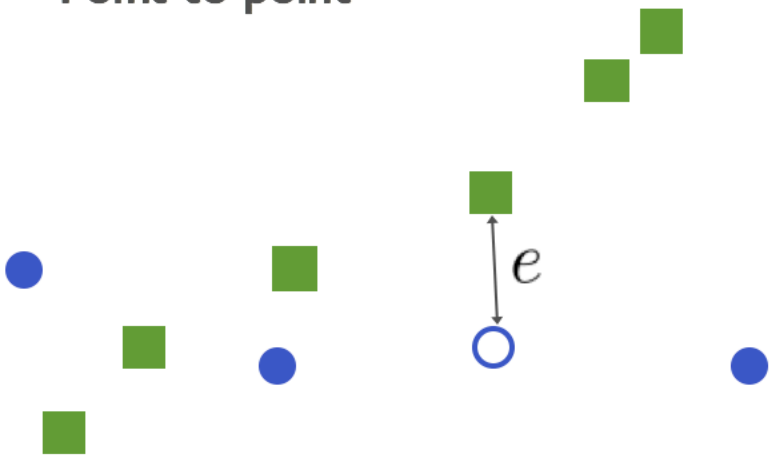
Minimization



- Error metrics
  - Point-to-point
  - Point-to-plane
  - and many more …

Minimization

Point-to-point



$$\sum_{i=0}^{N} \left\| \hat{T} * p_i - q_i \right\|^2$$

Besl and McKay [1992]

Walker et al. [1991] Horn [1987] Arun et al. [1987]

Procrustes alignment

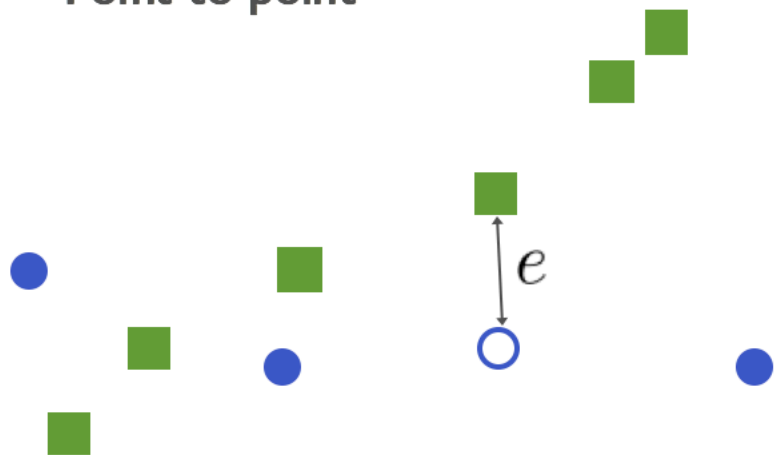
- Arun's method
  - Let $\mathbf{p}_i$ be the points in scan 1 and $\mathbf{q}_i$ be the points in scan 2
  - Compute the centers of both point clouds:

$$\bar{\mathbf{p}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{p}_i \qquad \bar{\mathbf{q}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{q}_i$$

  - Compute the matrix $\mathbf{Q} = \sum_{i=1}^{n} (\mathbf{q}_i - \bar{\mathbf{q}})(\mathbf{p}_i - \bar{\mathbf{p}})^t$
  - Rotation $\mathbf{R} = \mathbf{V}\mathbf{U}^t$ where $SVD(\mathbf{Q}) = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$
    Translation $\mathbf{t} = \bar{\mathbf{p}}\text{-}\mathbf{R}\bar{\mathbf{q}}$

Minimization

Point-to-point

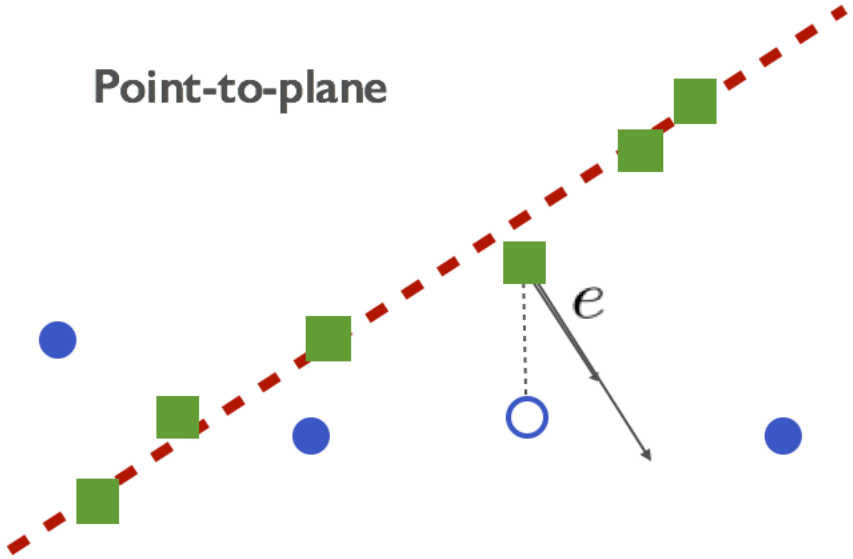


$$\sum_{i=0}^{N} \left\| \hat{T} * p_i - q_i \right\|^2$$

Besl and McKay [1992]

Walker et al. [1991] Horn [1987] Arun et al. [1987]

Point-to-plane

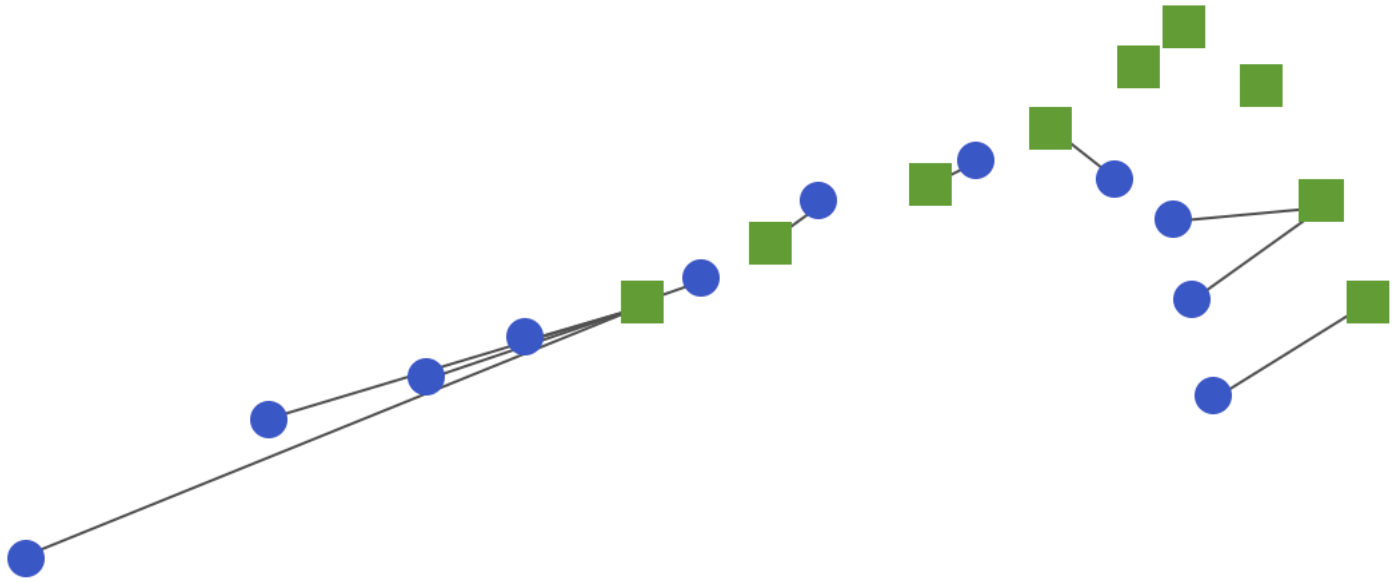


$$\sum_{i=0}^{N} \left\| (\hat{T} * p_i - q_i) \cdot n_i \right\|^2$$

Chen and Medioni [1992]

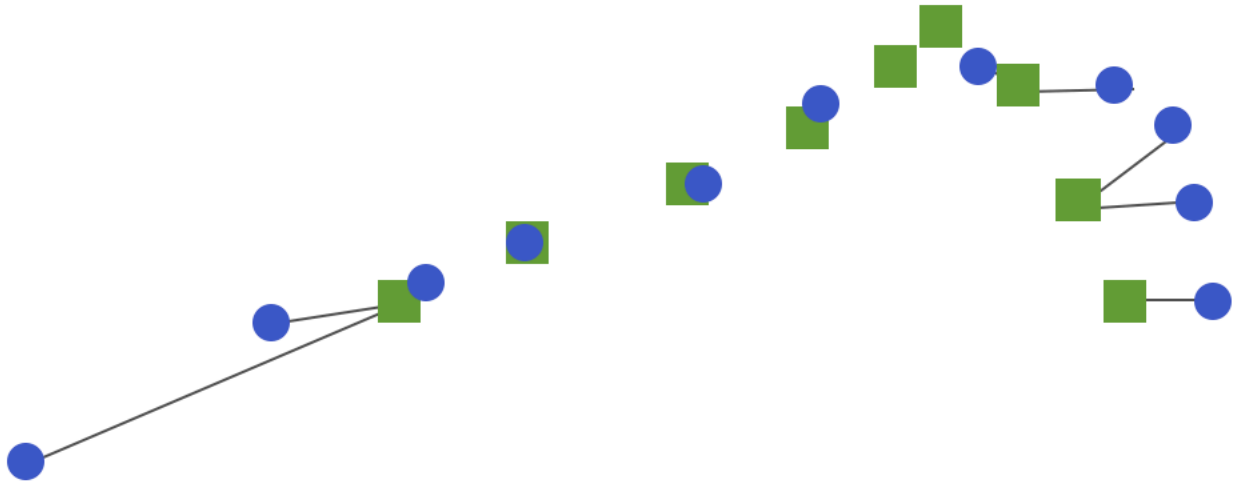
Minimization

- Sliding situation



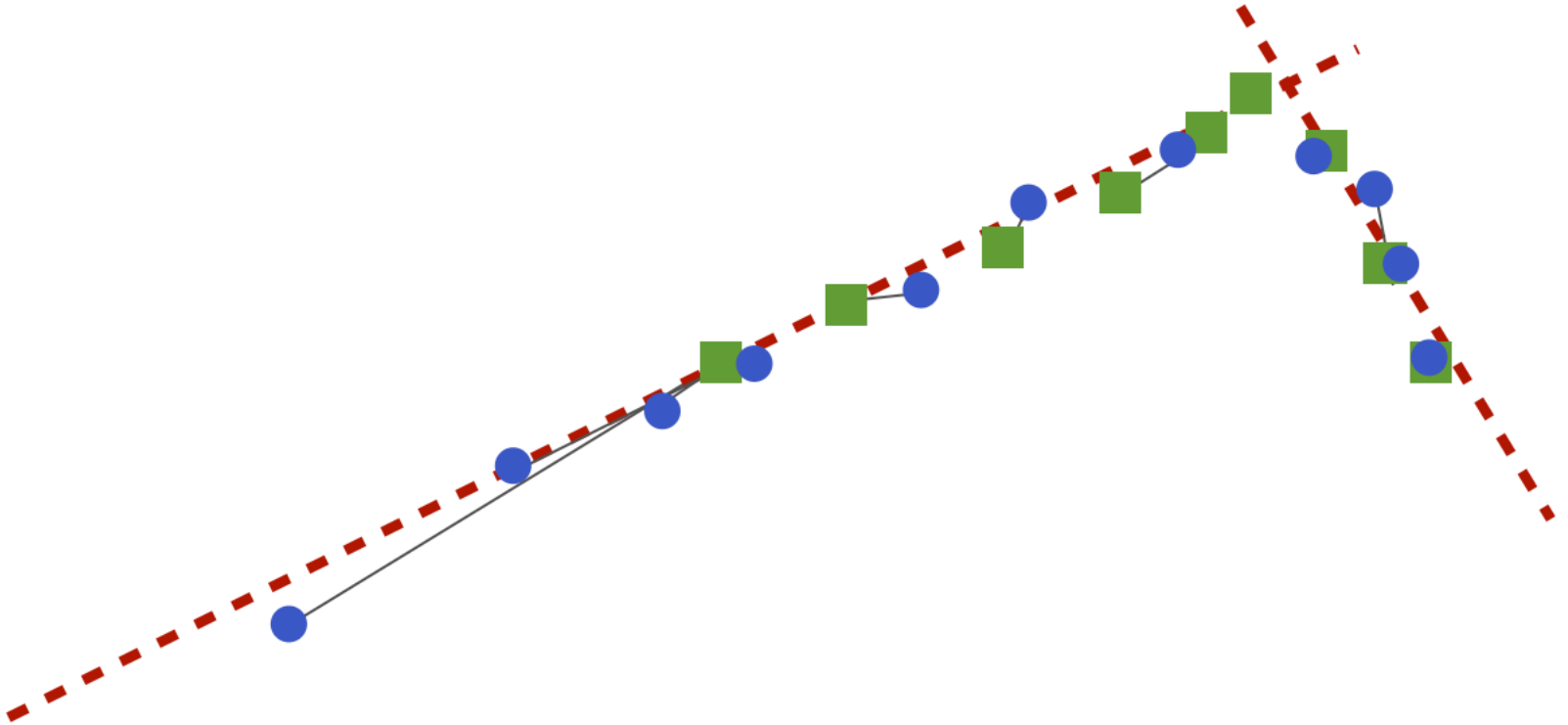
Minimization

- Point-to-point distance

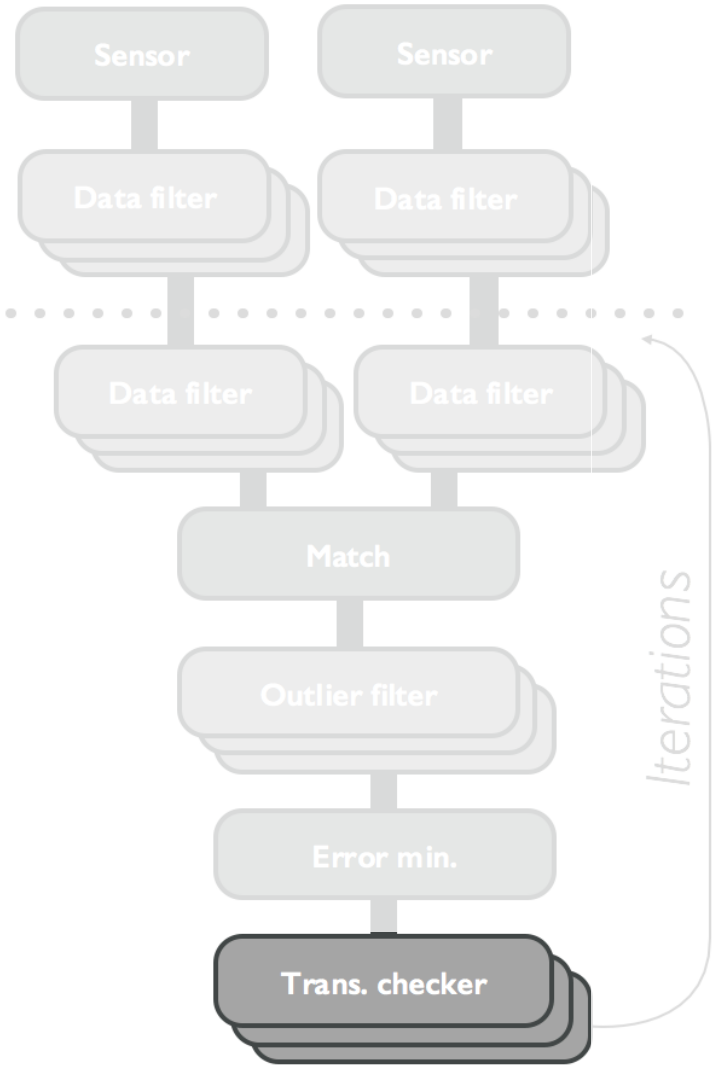


Minimization

- Point-to-plane distance



Convergence detection

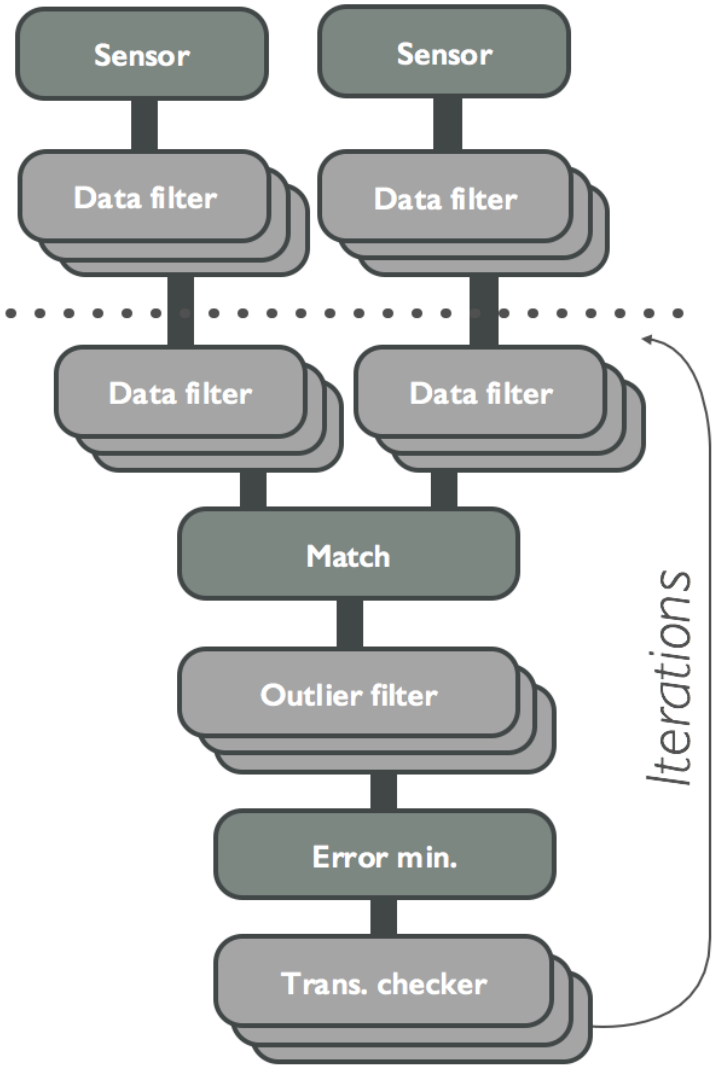


- Main goal:
  - Iteration early out

Convergence detection

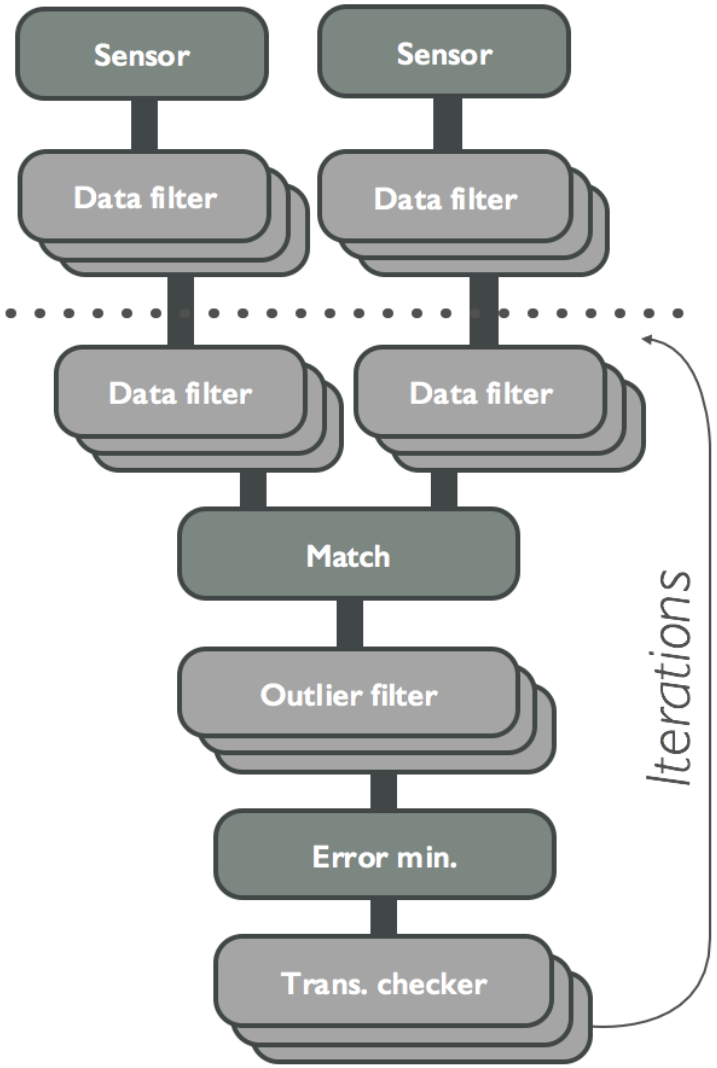
- Maximum number of iterations
  - Blind
- Absolute motion
  - Limit based on expected sensor noise
  - Detect divergence
  - Deals with rotation and translation separately
- Motion difference between iterations
  - Detect convergence up to wanted noise
  - Deals with rotation and translation separately

Summary



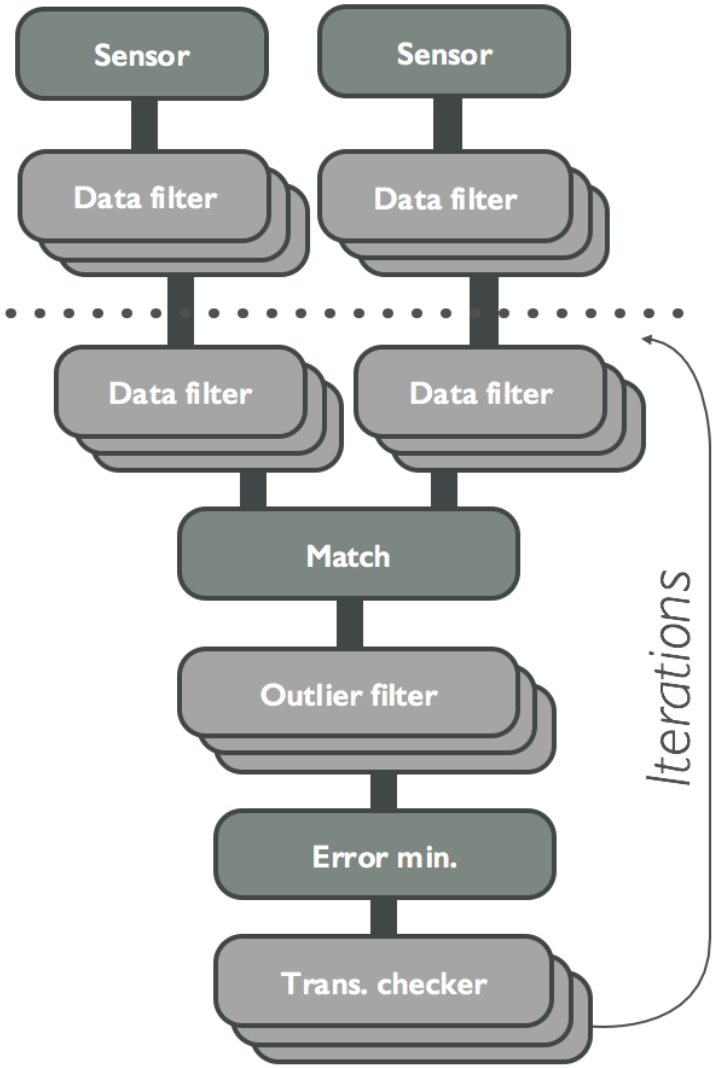
- Accuracy vs. Speed!

Summary



- Augment Accuracy
  - Keep more points
  - Augment the number of matches
  - Adjust your expected overlap
  - Verify that normal vectors are well extracted
  - Add more iterations

Summary



- Augment Speed
  - Reduce number of points as soon as possible, compute only the minimum
  - Approximative search (kD-tree)
  - Point-to-plane takes less iterations
  - Relax targeted values