Robotics

Estimation and Learning
with Dan Lee

Week 4.
Localization

4.1 Odometry Modeling
4.2 Sensor Registration
4.3 Particle Filter

Penn Engineering

Week 4.

Localization

4.1 Odometry Modeling

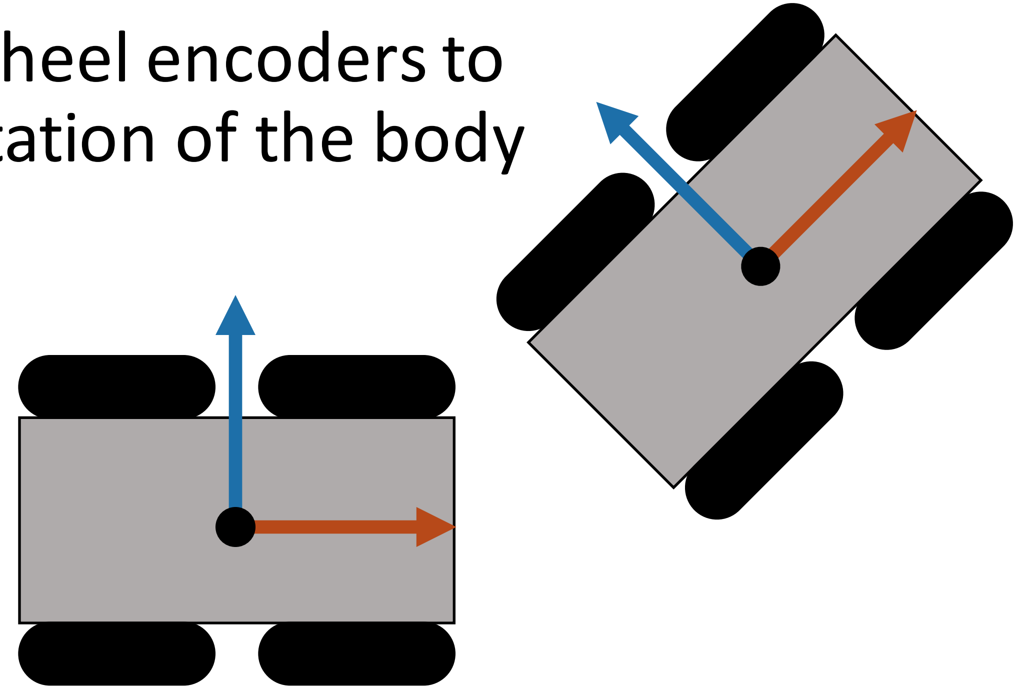
Localization

- Encoder and local sources of information can be very precise ~200,000 pulse/m [1]
- Laser range finding: 3-5 cm obstacle detection [2,3]
- RGB(-D) Vision: 1-10cm
- Gyroscope (turning)
  - Accelerometer integration provides poor accuracy
- Higher precision, but *local* information

[1] http://www.clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/
[2] http://www.hokuyo-aut.jp/02sensor/07scanner/utm_30lx_ew.html
[3] http://velodynelidar.com/vlp-16-lite.html
[4] Robust Real-Time Visual Odometry for Dense RGB-D Mapping

Simple Approach

- Integrate odometry information
- Form a model of the vehicle
  - Skid steer, in this case
- Map ticks of the wheel encoders to translation and rotation of the body



Tracking Angular Movement

- Encoder ticks (e) are observed at the inner and outer radii

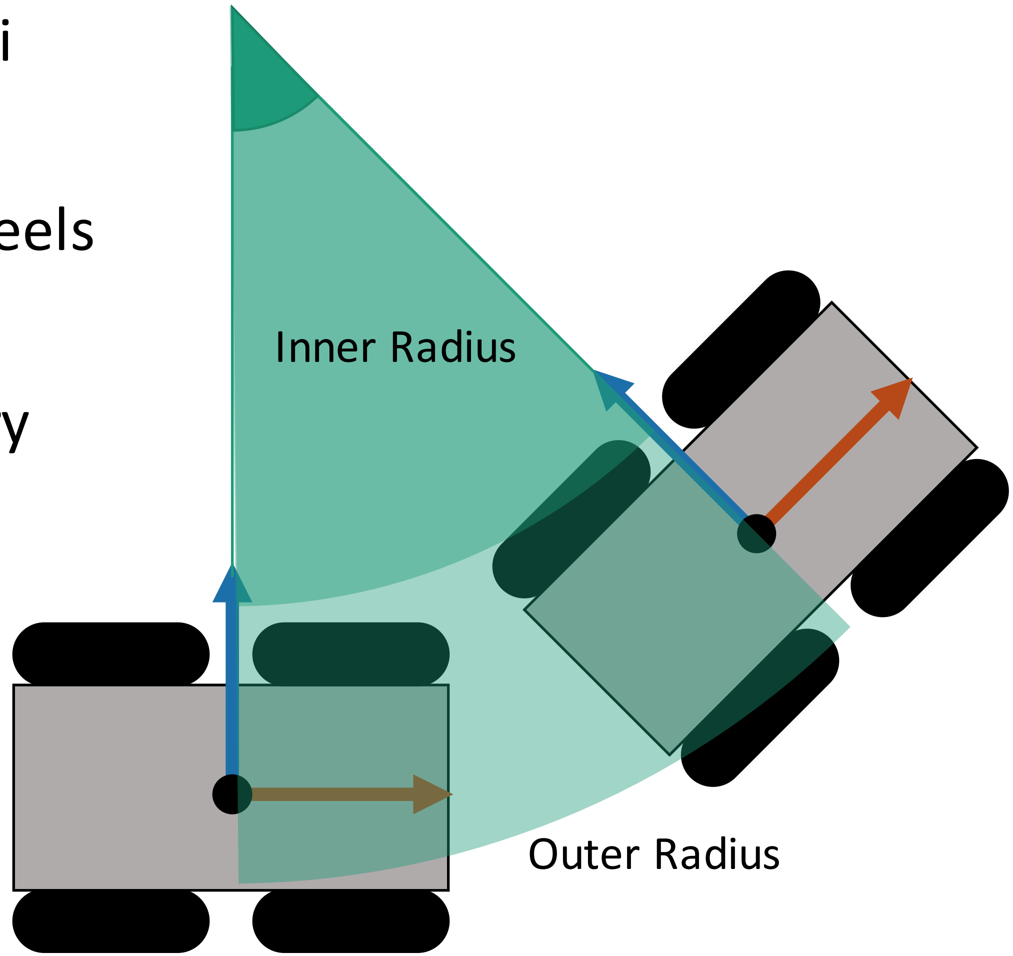
$$e_i = \theta r_i \quad e_o = \theta r_o$$

- Known width between wheels

$$\mathrm{w} = r_o - r_i$$

- Calculate angular odometry

$$\boxed{\theta = \frac{e_o - e_i}{w}}$$

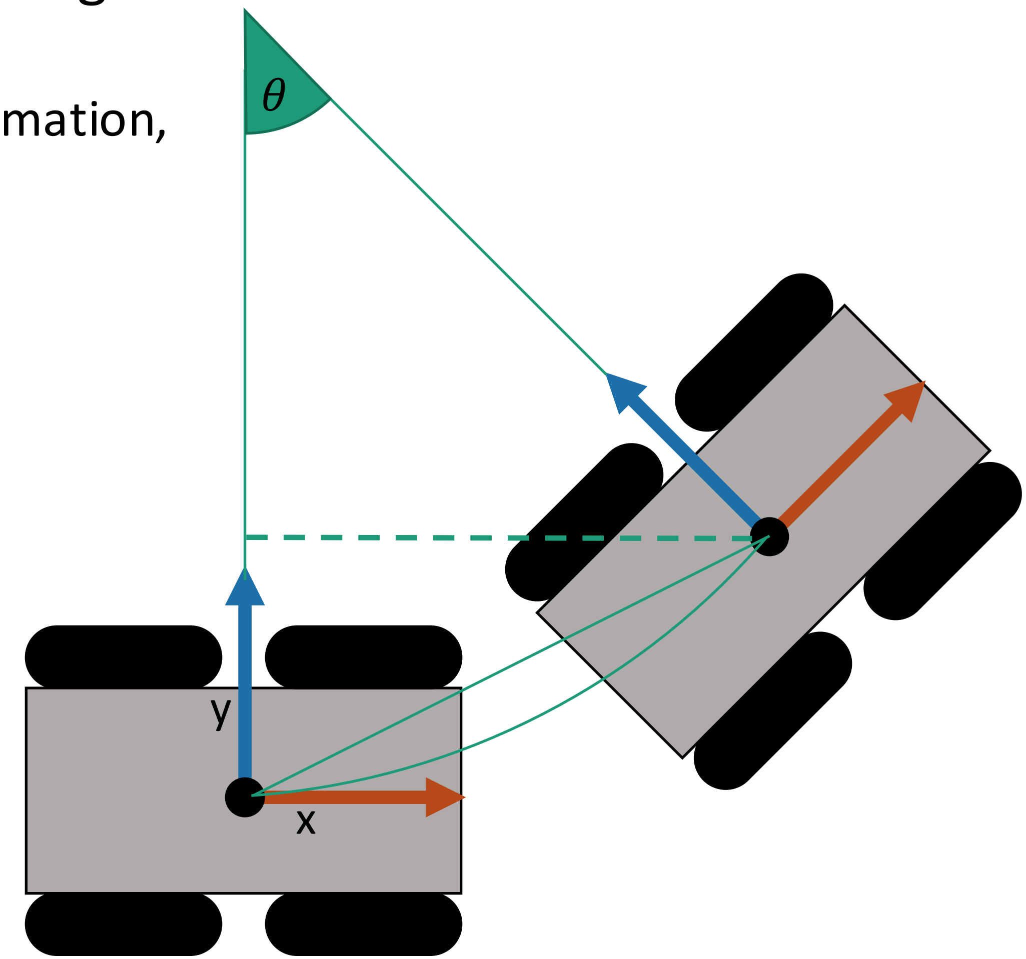


Tracking Translational Motion

- Translation requires knowledge of the angular movement
  - Use circular sector approximation, valid for small movements
- *Quiz: Spinning in Place*

$$\boxed{y = \frac{e_o + e_i}{2}\cos\theta}$$

$$\boxed{x = \frac{e_o + e_i}{2}\sin\theta}$$



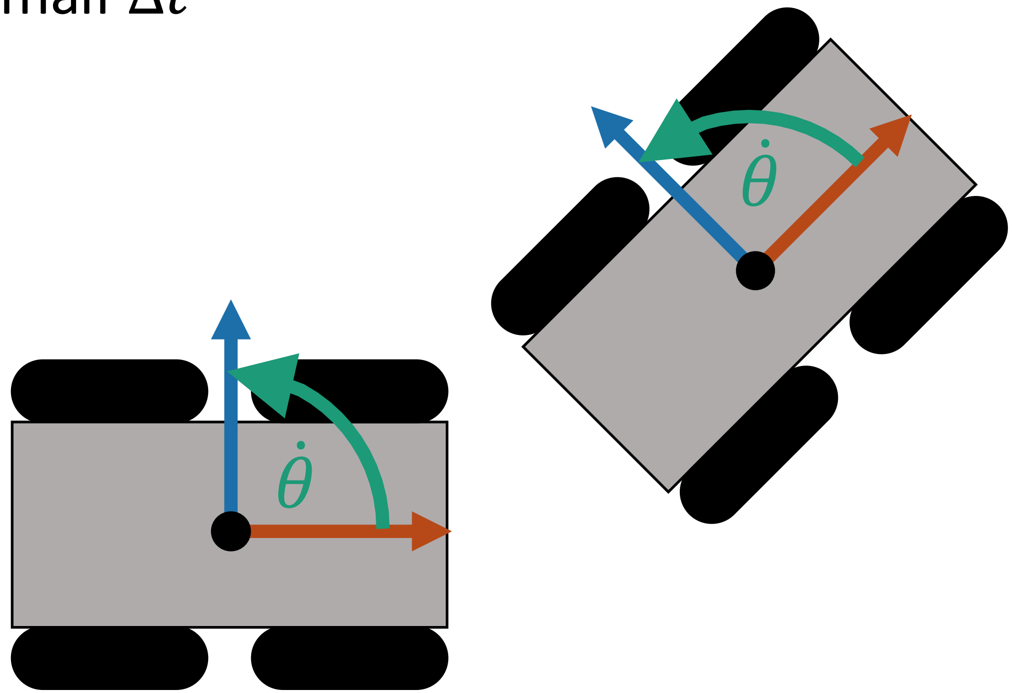
Aiding Pure Encoder Odometry

- Encoder measurements can be noisy
- Angular estimate feeds into translation, propagating error
- *Solution:* Utilize more precise gyroscope for angular change
- Gyroscope is accurate for small Δt

$$\boxed{\theta = \dot{\theta}\Delta t}$$

$$\boxed{y = \frac{e_o + e_i}{2}\cos\theta}$$

$$\boxed{x = \frac{e_o + e_i}{2}\sin\theta}$$



Simple Approach Characteristics

- Local frame of reference of the robot starting point
- *Issue:* Encoders suffer from slippage, missing counts
- *Issue:* Gyroscope integration suffers from drift
- Utilizing maps of the world can correct localization errors

Robotics

Estimation and Learning
with Dan Lee

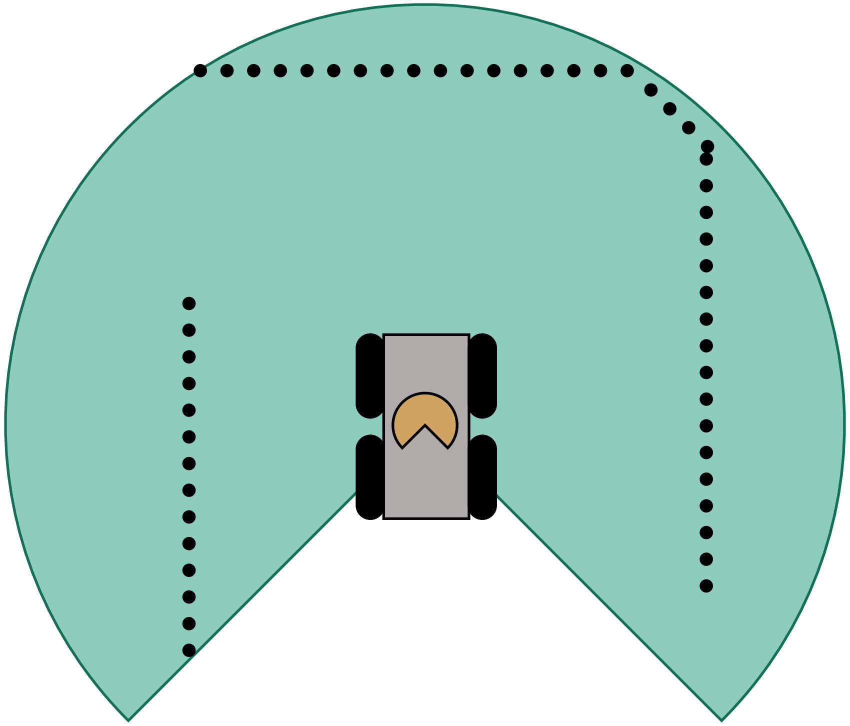
Week 4.
Localization

4.2 Map Registration

Penn Engineering

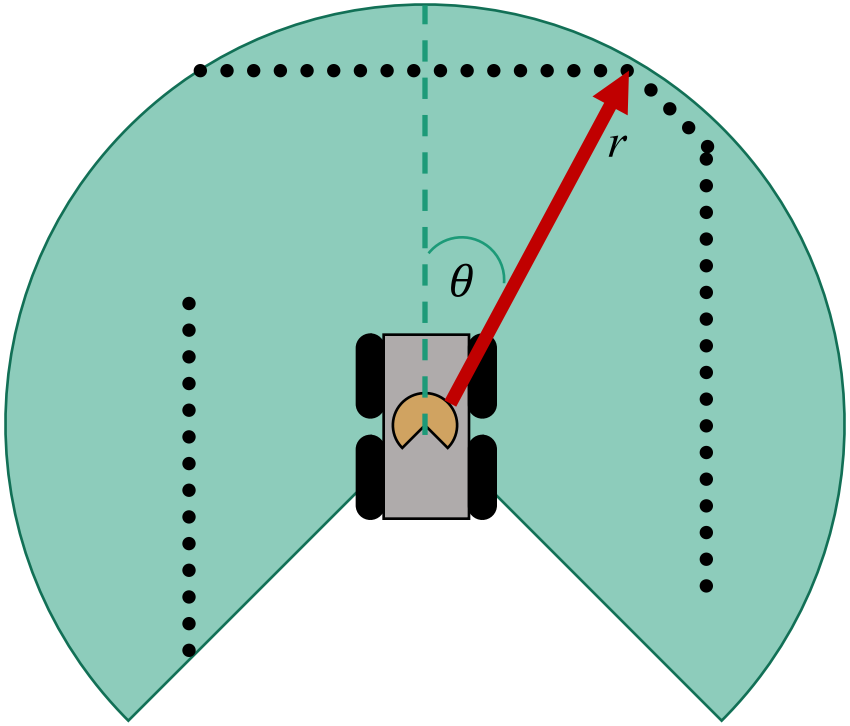
LIDAR Depth Sensor

- Depth measurements made in polar coordinates
- Continuous readings, r, at discrete angles, θ



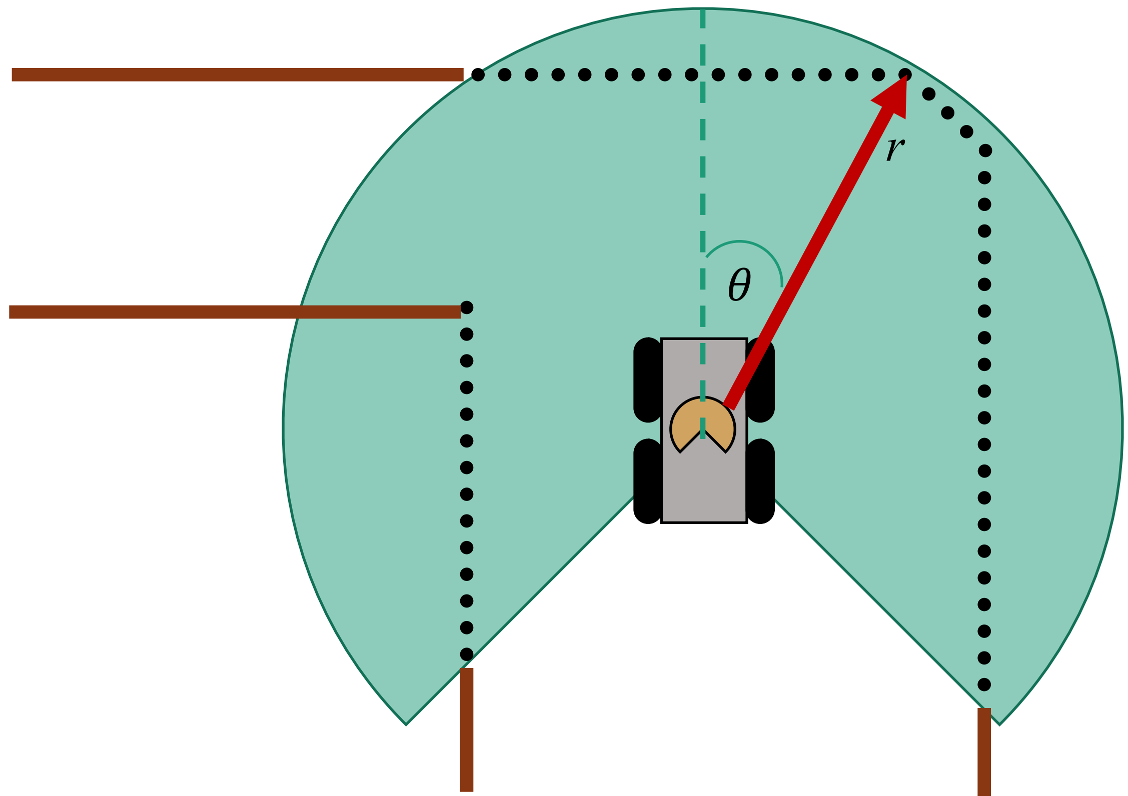
LIDAR Depth Sensor

- Depth measurements made in polar coordinates
- Continuous readings, r, at discrete angles, θ



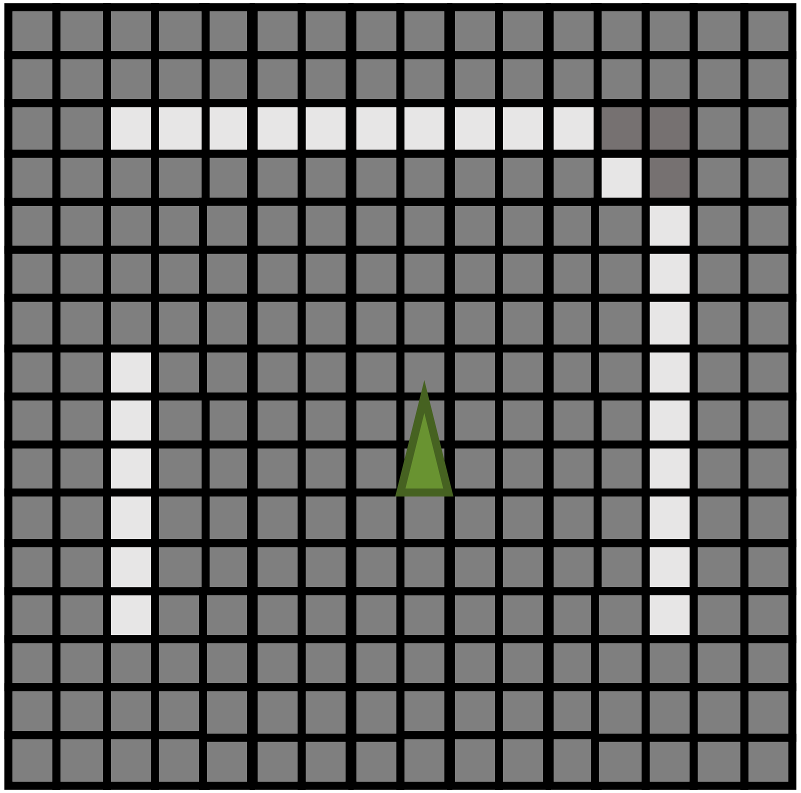
LIDAR Depth Sensor

- Depth measurements made in polar coordinates
- Continuous readings, r, at discrete angles, θ



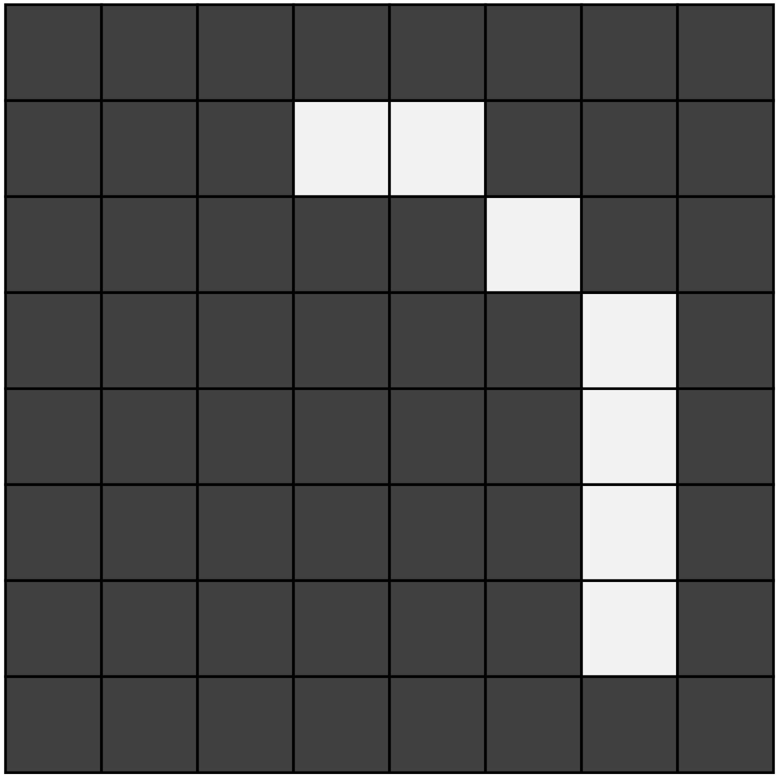
Map Representation

- Discrete Grid representing 2D space *(see Week 3)*
- White cells represent the presence of an obstacle

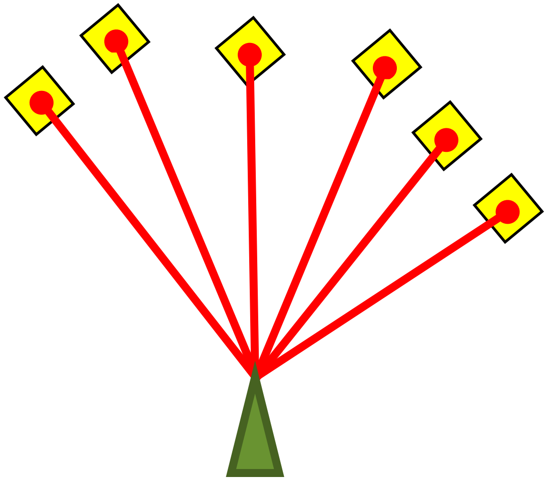


Map Measurements

Map

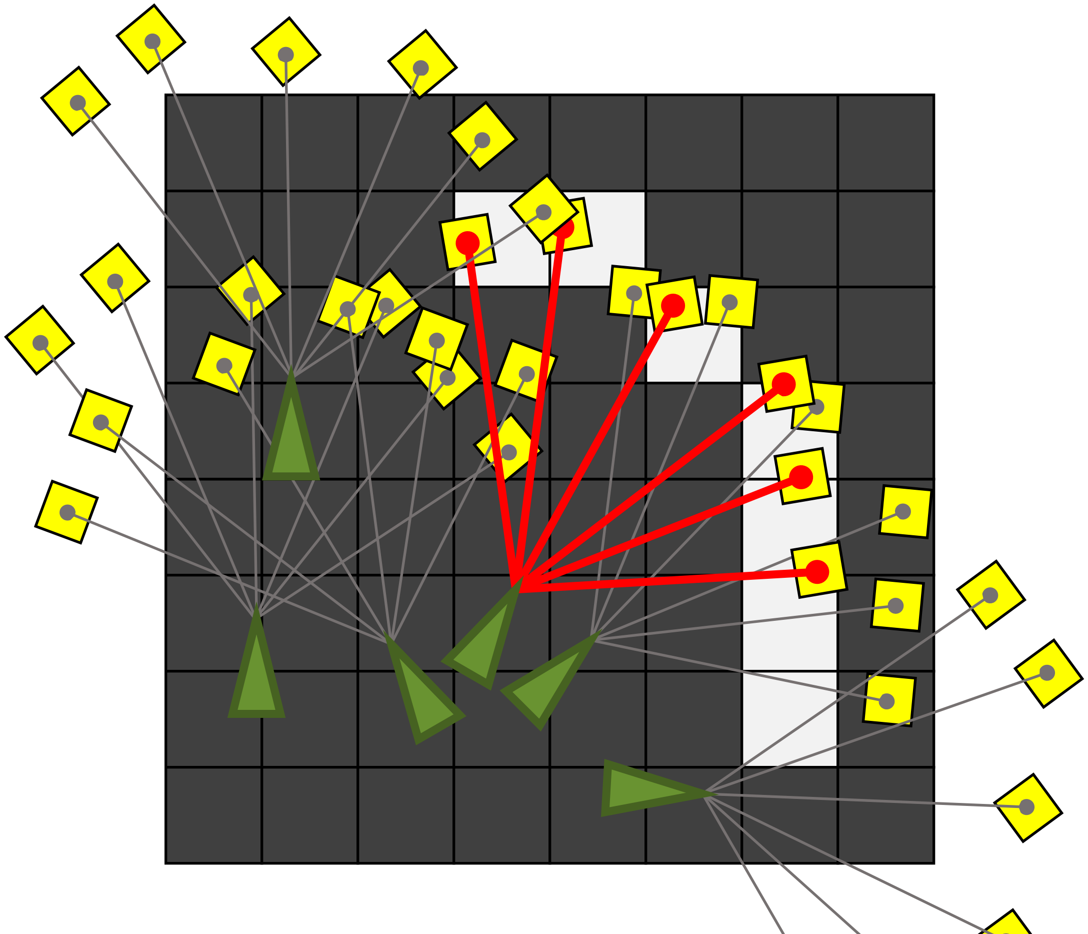


Measurements



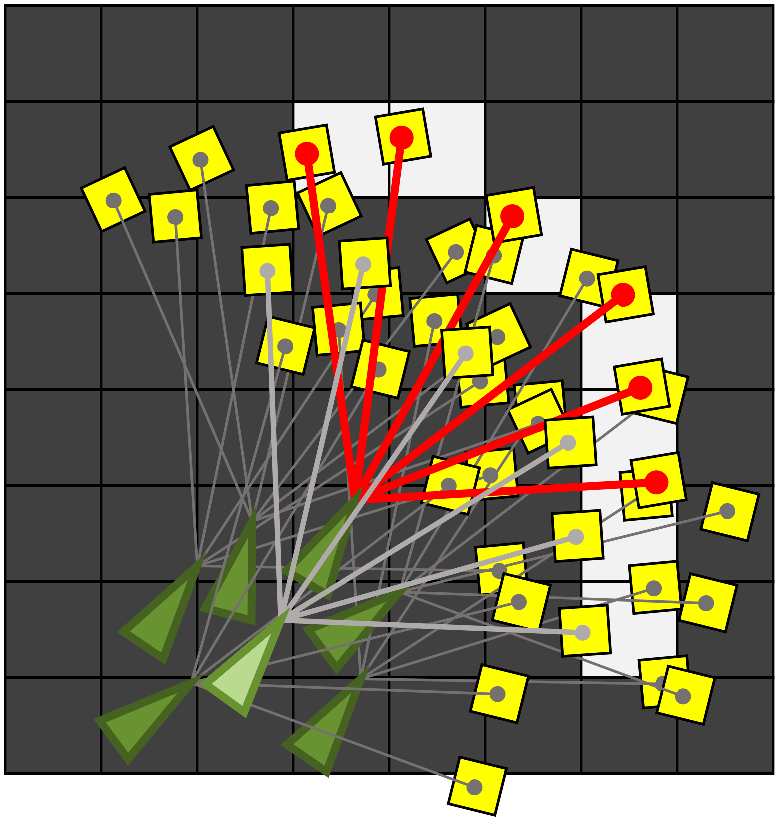
Map Registration

- Find robot pose that best explains the measurements

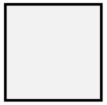


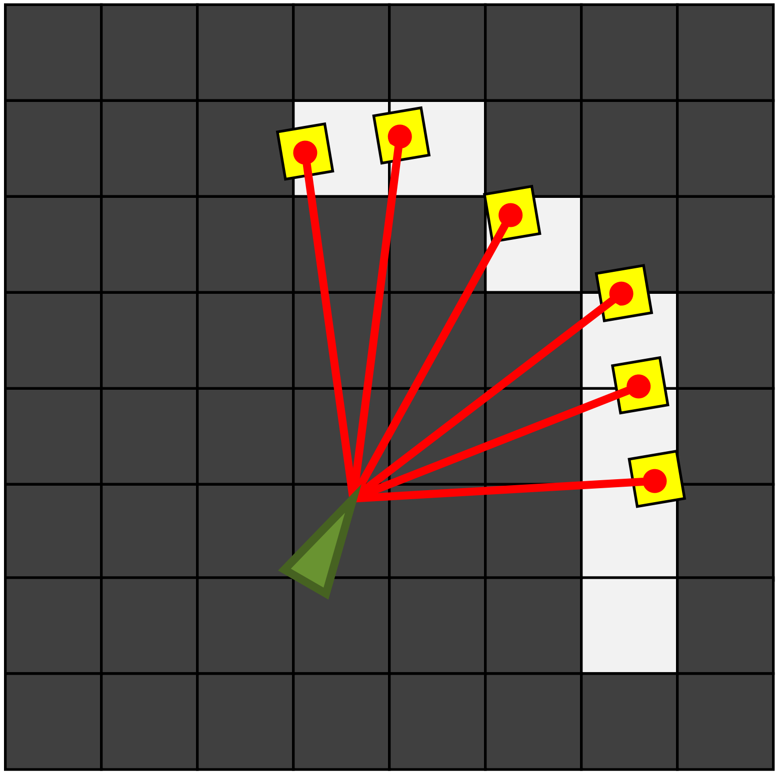
Map Registration

- Find robot pose that best explains the measurements



Map Registration

- Correlate laser obstacles with map obstacles 
- Correlate laser free space with map free space 

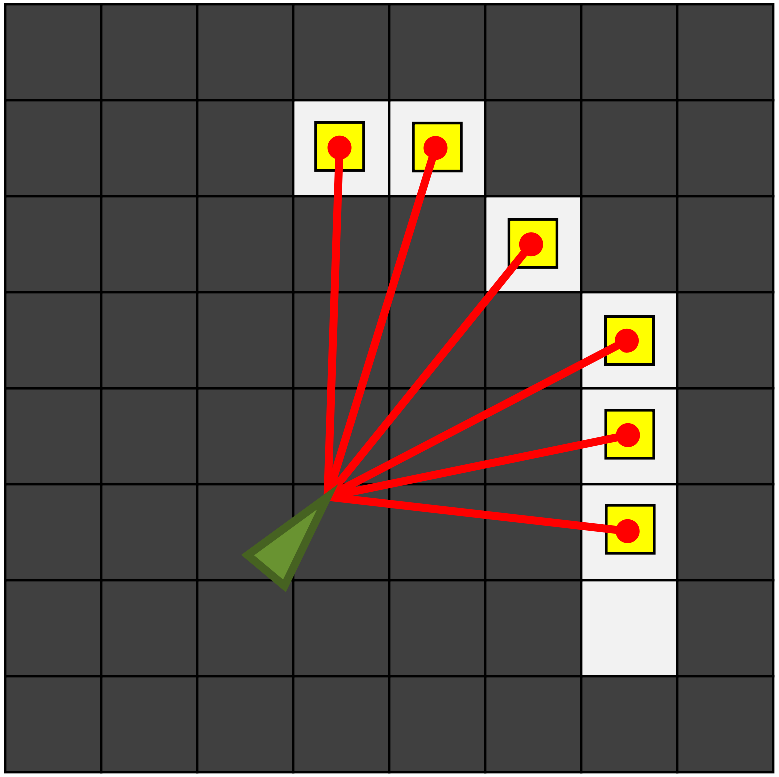


$$\sum_r \delta\left(p_x + r\cos(p_\theta + r_\theta)\,, p_y + r\sin(p_\theta + r_\theta)\right) \cdot m(x, y)$$

Map Registration

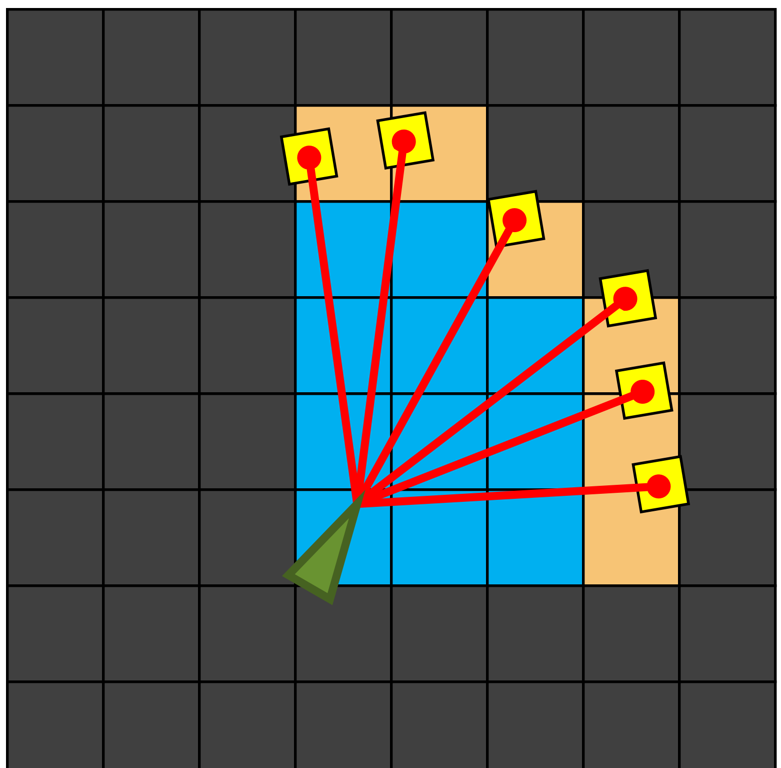
Find the best pose

$$\max_{p} \sum_{r} \delta\big(p_x + r\cos(p_\theta + r_\theta)\,, p_y + r\sin(p_\theta + r_\theta)\big) \cdot m(x, y)$$



Map Registration

- Laser scans penetrate free space
- Laser scans return distance to obstacle



Map Registration

- Adding pose uncertainty

Robotics

Estimation and Learning
with Dan Lee

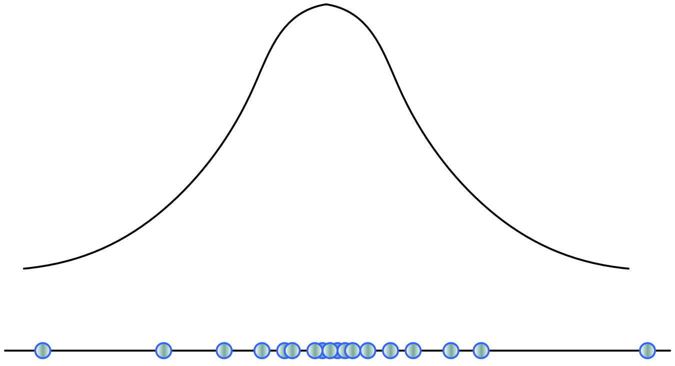
Week 4.
Localization

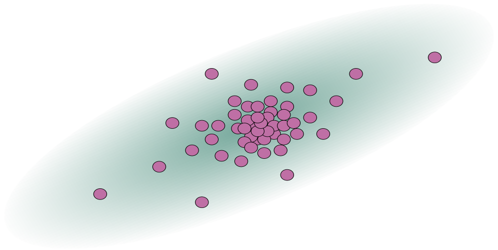
4.3 Particle Filter

Penn Engineering

Particle Filter

- Samples approximate a probability distribution
- Fast and efficient non-parametric model
- Ability to represent multimodal distributions
  - Mixtures of Gaussians, multi-hypothesis Kalman Filter



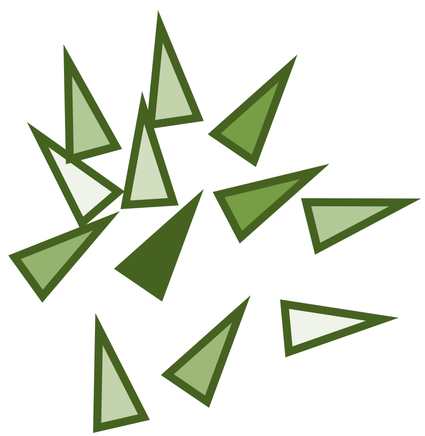


Particle Filter

- Dirac Delta function
  - Sigma is going to zero, Gaussian distribution
- Particle Filter : Limit of Gaussian mixtures when $\sigma \to 0$ (variances shrink to zero)

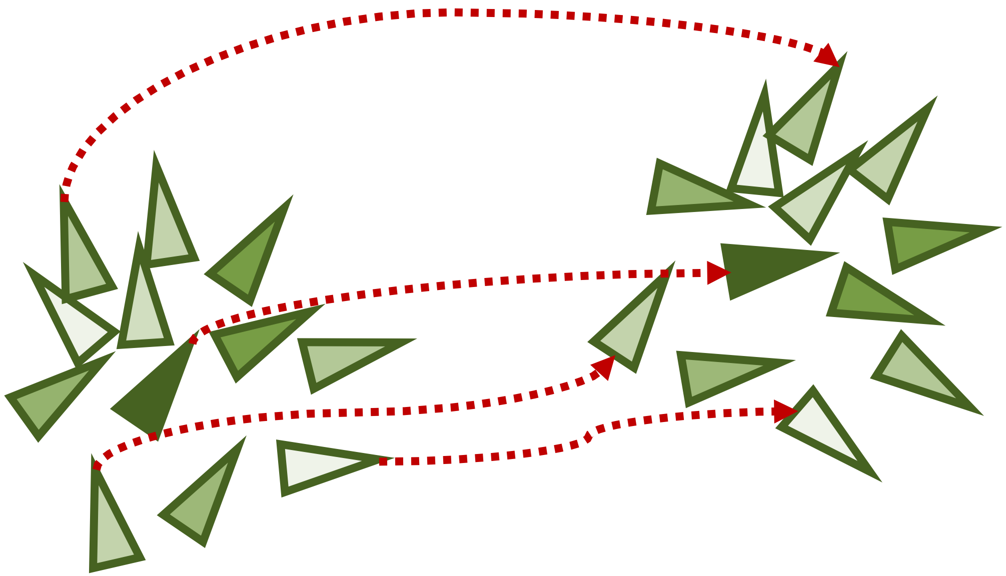
Initial Population

- Initial group of particles represents the underlying distribution of the belief state
- Particle is comprised of (pose, weight)
- Here, darker colors represents a higher *weight*
  - Represents probability, such that *weight = prob(pose)*



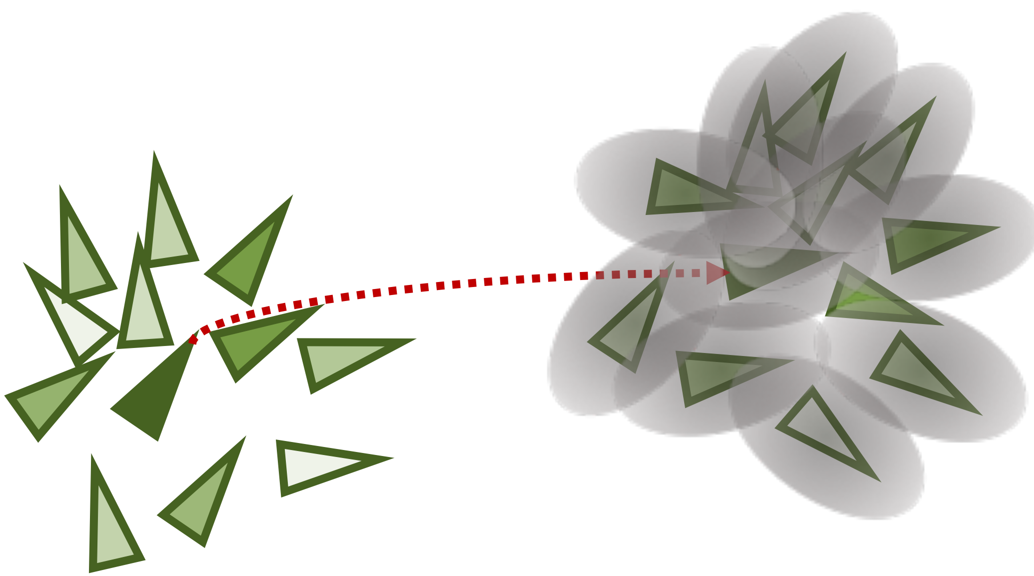
Odometry Update

- Move the particles based on odometry information
- Each particle represents a possible pose, so individually must be moved via its local frame



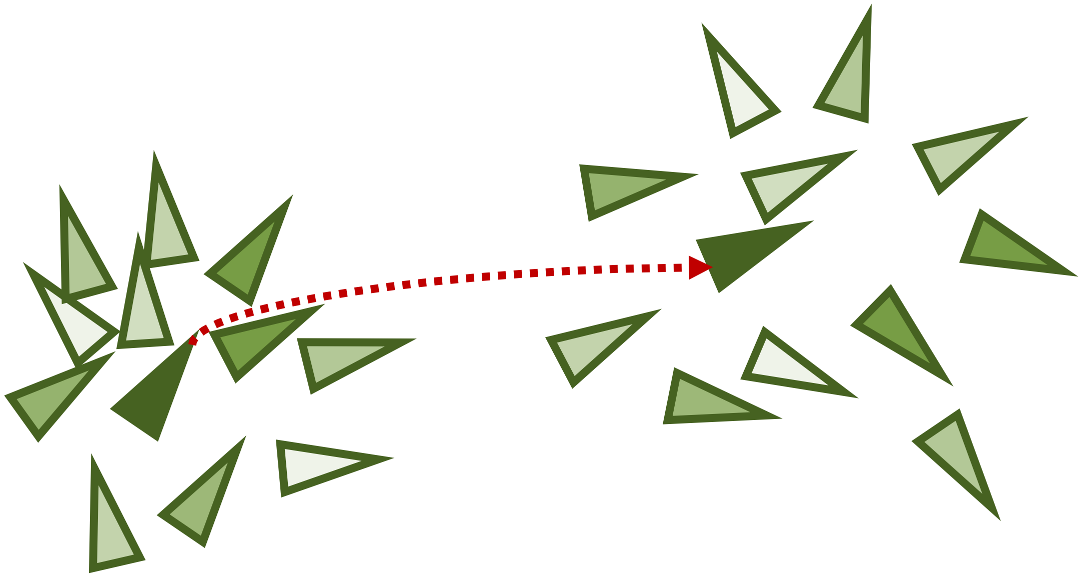
Odometry Update

- Include odometry noise model
- Sampled for each particle from the odometry noise distribution $p_i' = p_i + \mathcal{N}(0, \Sigma)$



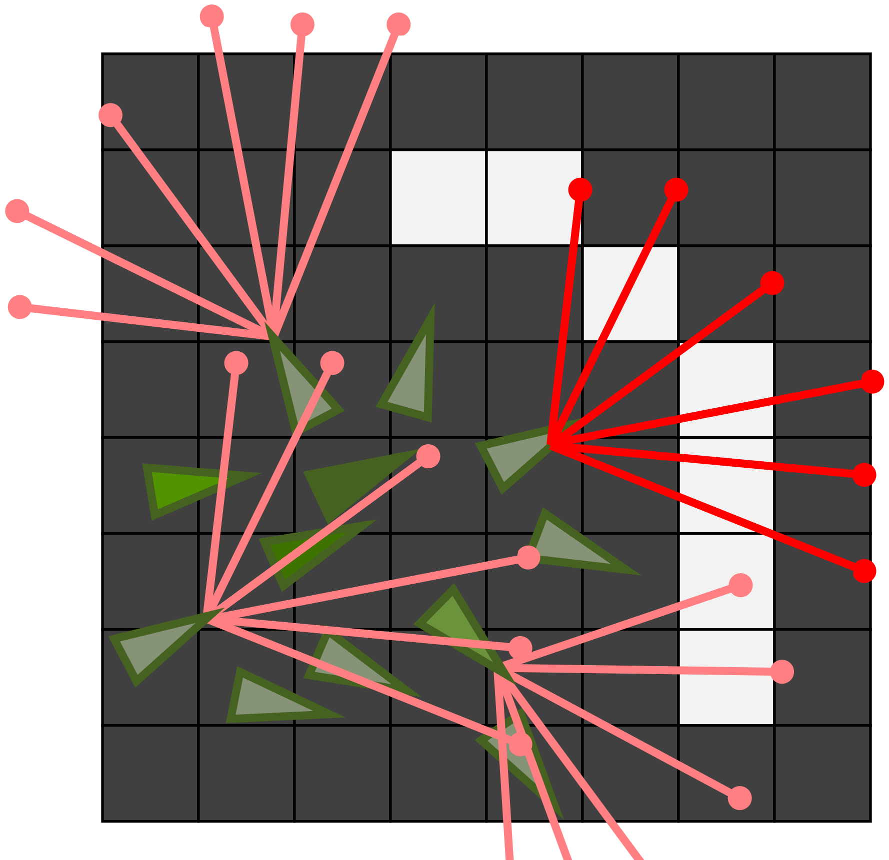
Odometry Update

- Dispersion of particles represents the added uncertainty from moving



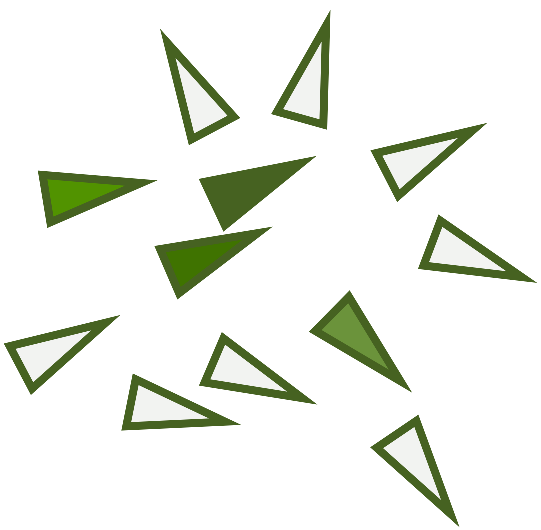
Correlation Update

- The weights of the particles can be updated based on LIDAR correlation data, $w_i' = w_i \cdot corr(p_i)$



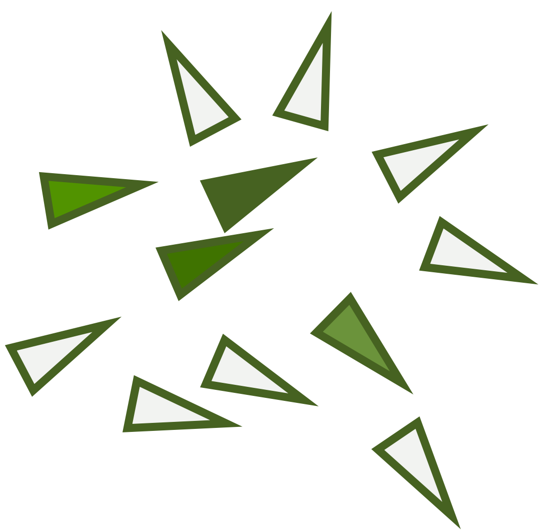
Correlation Update

- The new set of particles capture the distribution after odometry and sensor measurement
- However, this may not be the optimal set to represent the distribution



Particle Resampling

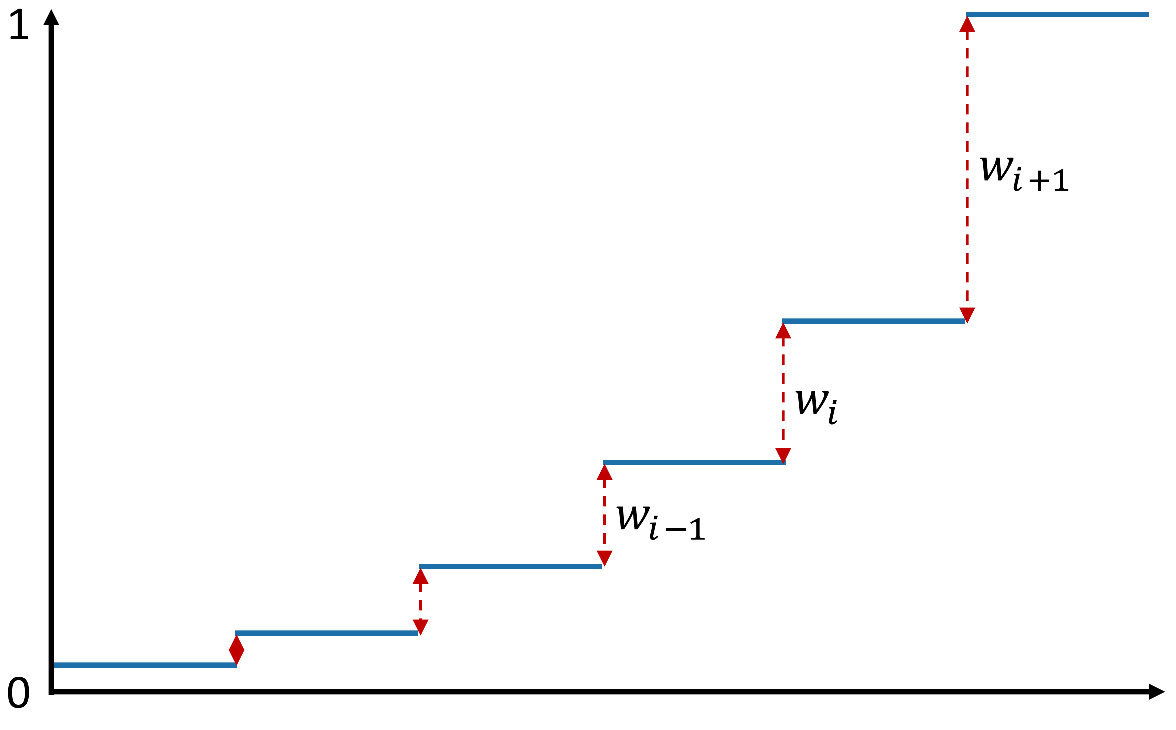
- Check a resampling criterion – the number of effective particles
- If the number of effective particles is too low, then *resample* to increase the effective number



$$n_{effective} = \frac{\left(\sum_i w_i\right)^2}{\sum_i w_i^2}$$

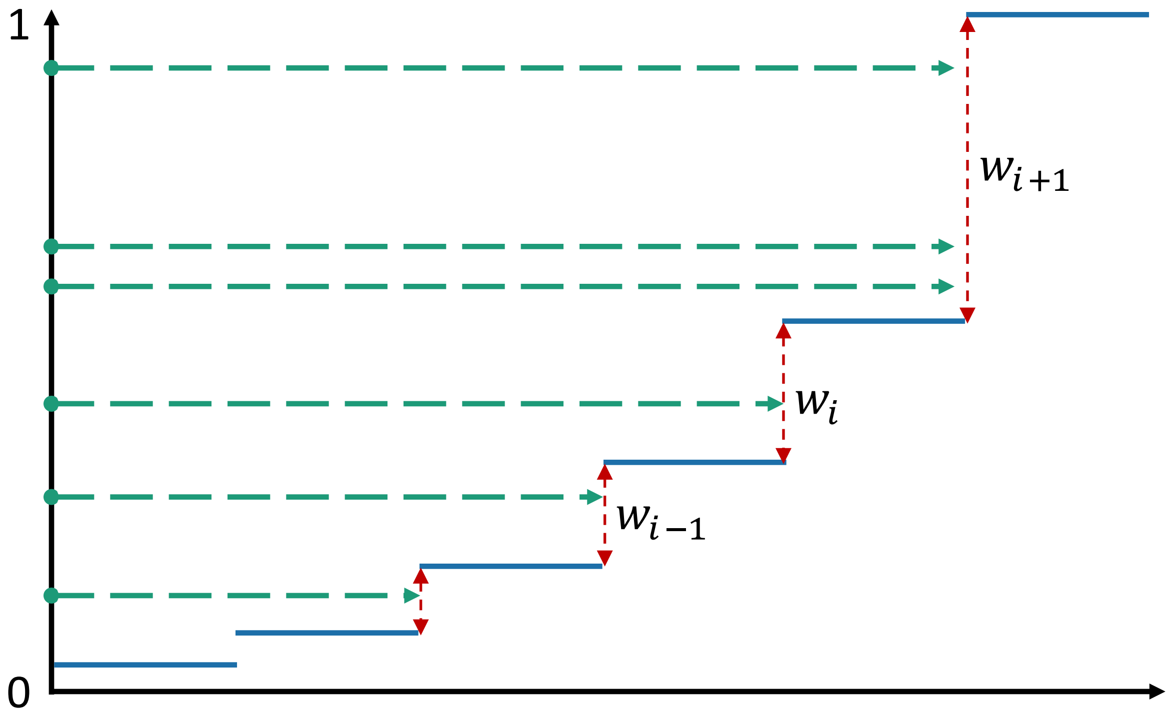
Particle Resampling

- Use the cumulative probability to aid in resampling
- Sum of normalized weights is 1



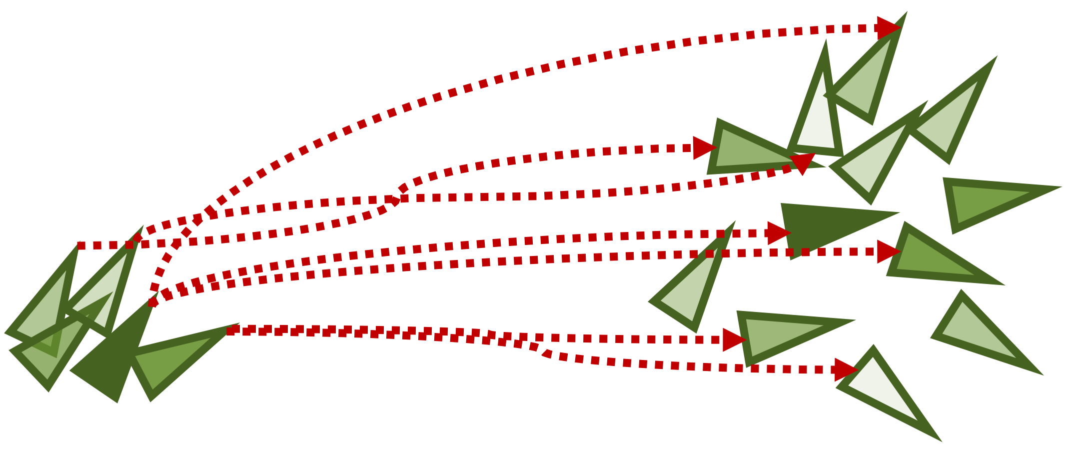
Particle Resampling

- Sample number uniformly between 0 and 1 of the cumulative range, and find which w_i includes that number



Particle Resampling

- The particles with the indices found in the resampling approach become the new set of particles to be fed into the next odometry update
- Particles may be duplicated, but the odometry noise will differentiate these particles.



Robotics

Estimation and Learning
with Dan Lee

Week 4.
Localization

4.4 Iterative Closest Point (ICP) Algorithm

Penn Engineering

Review: EM Algorithm

$$\arg\max_{\theta} F(X|\theta)$$

Initialize $\hat{\theta}$

E-step

Find a lower bound $G\left(\theta|\hat{\theta}\right)$

M-step

$\hat{\theta}_{new} = \arg\max_{\theta} G(\theta|\hat{\theta})$

Stop if converged

General EM

$$\arg\max_{\boldsymbol{\mu},\Sigma} \sum_{i=1}^{N} \ln\left\{\frac{1}{K}\sum_{k=1}^{K} g_k(\mathbf{x}_i|\boldsymbol{\mu}_k,\Sigma_k)\right\}$$

Initialize $\boldsymbol{\mu}$ and Σ

E-step

Update z_k^i

M-step

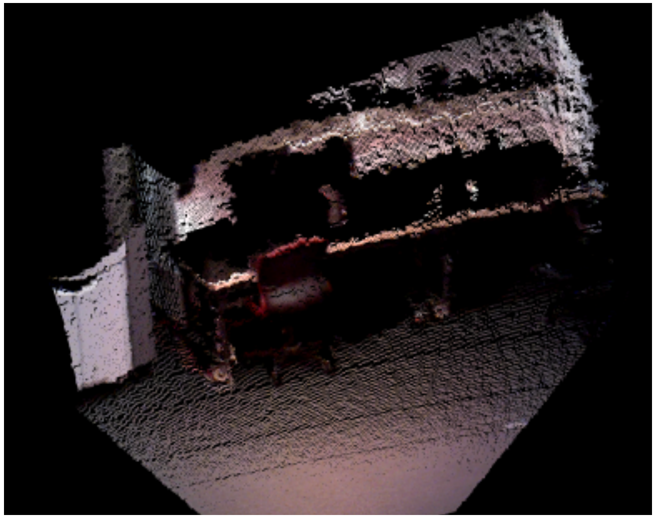
Update $\boldsymbol{\mu}$ and Σ

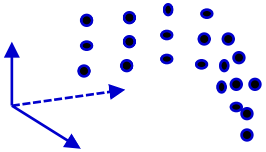
Stop if converged

EM for GMM

Review: 3D Map Representation

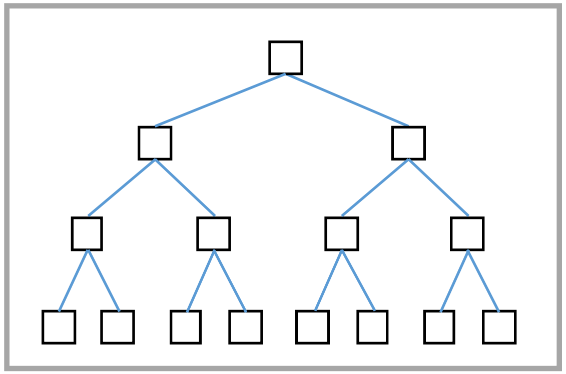
3D point cloud measurement





Map visualized in 3D

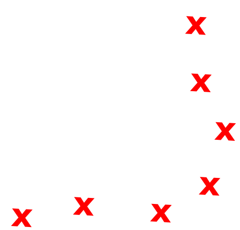
Implementation Example



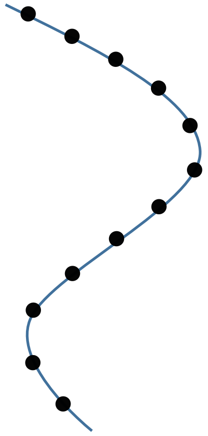
ICP Algorithm

- Problem: Register two point sets X and Y.

Measurement (X)



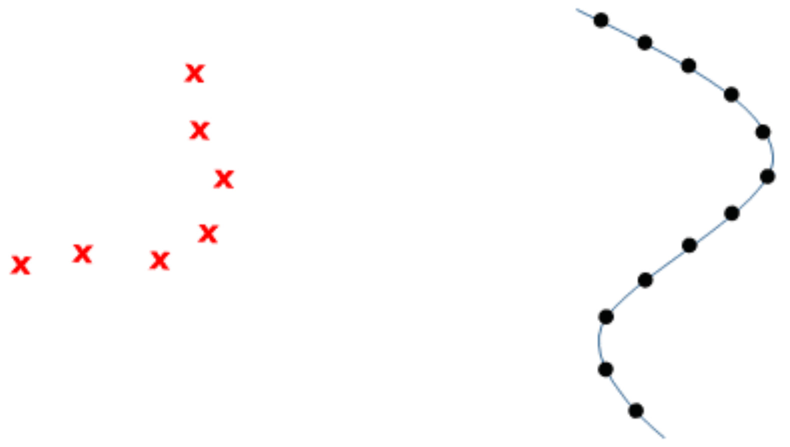
Model (Y)



ICP Algorithm

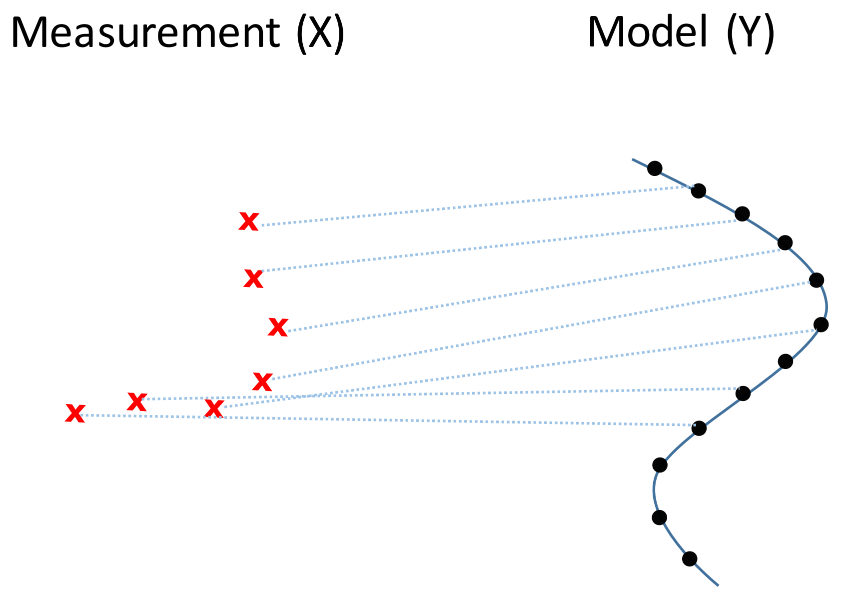
- Problem 1: Rotation and translation?

Measurement (X) Model (Y)



ICP Algorithm

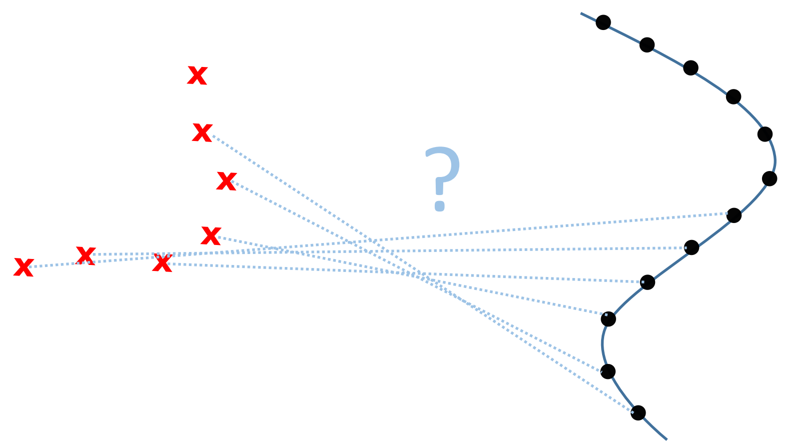
- Problem 2: Correspondences?



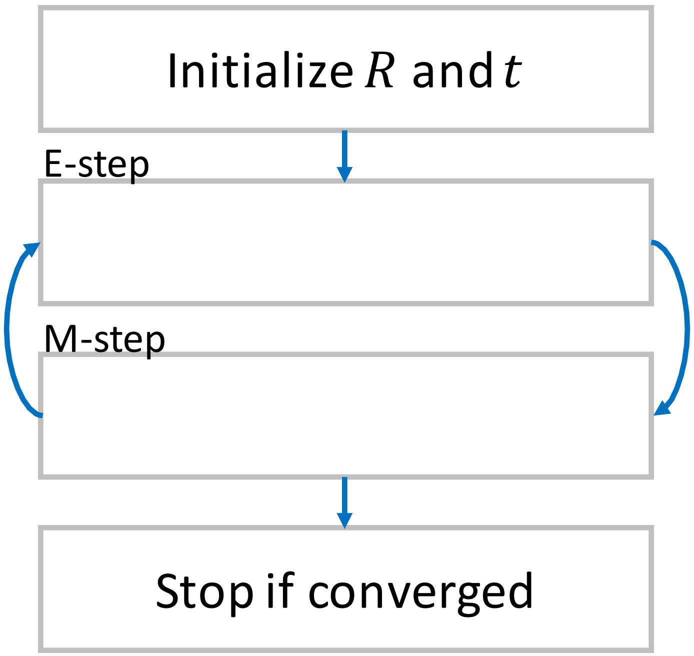
ICP Algorithm

- Problem 2: Correspondence

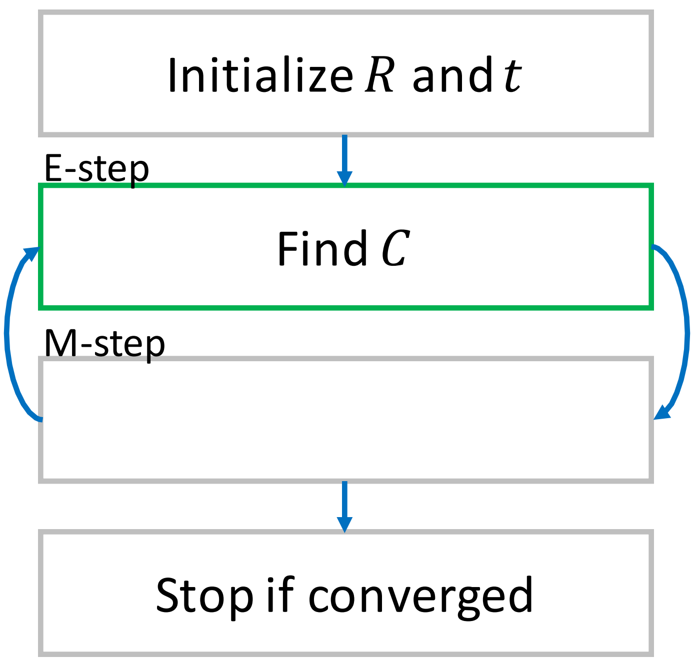
Measurement (X) Model (Y)



ICP Algorithm

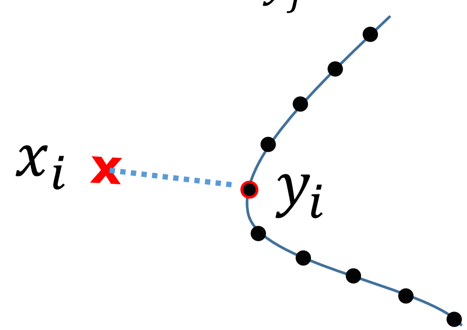


ICP Algorithm

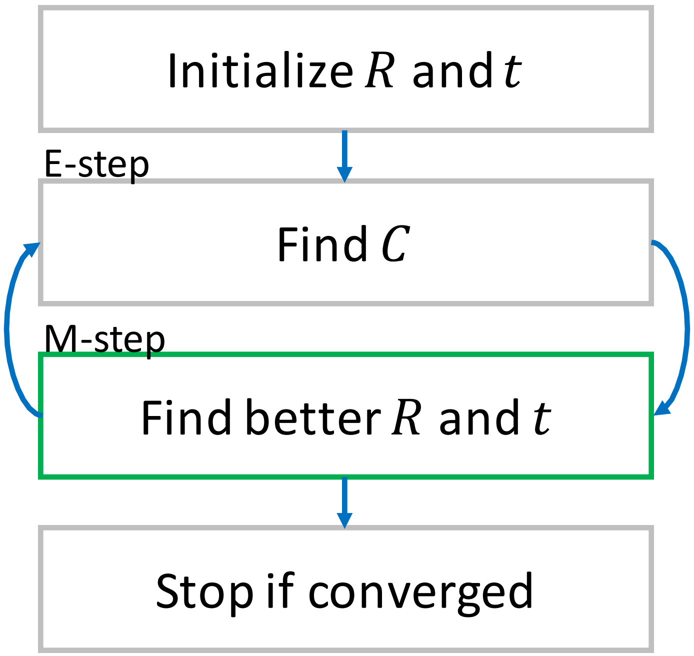


Closest point!

$$y_i = \arg\min_{y_j \in Y} \| x_i - y_j \|$$



ICP Algorithm



$$R, t = \arg\min \sum_{x_i, y_i \in C} \|d(x_i, y_i)\|^2$$

[SOLUTION] K. Arun, T. Huang, and S. Blostein, "Least-squares fitting of two 3D point set", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5), pp. 698–700, 1987.

ICP Algorithm

Initialize R and t

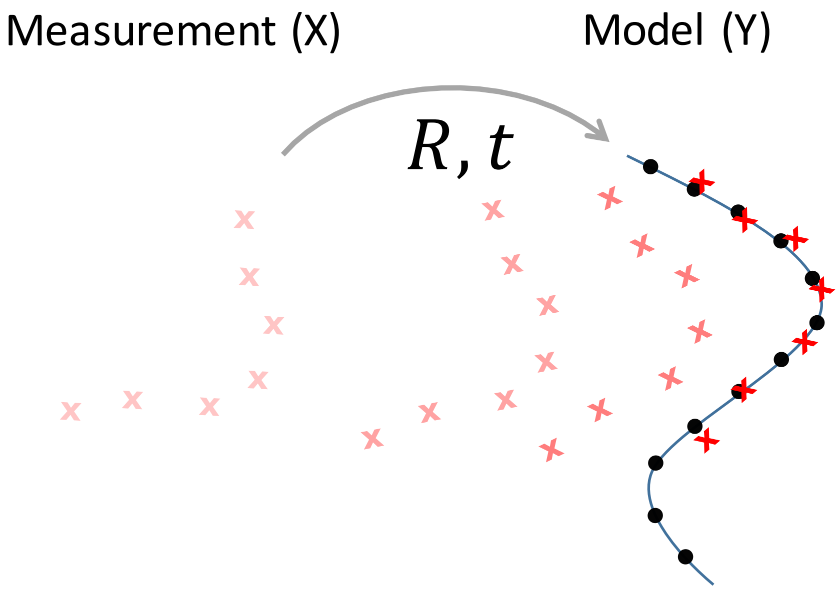
E-step

Update C

M-step

Update R and t

Stop if converged



ICP: Example

Initialize R and t

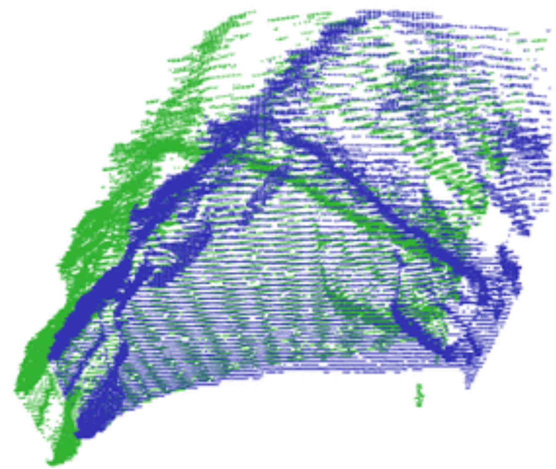
E-step

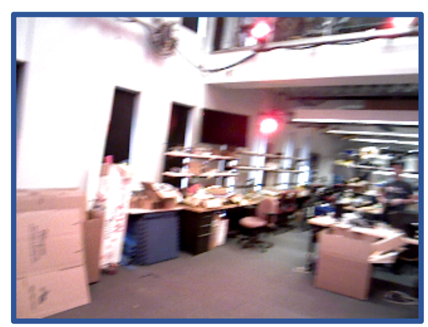
Update C

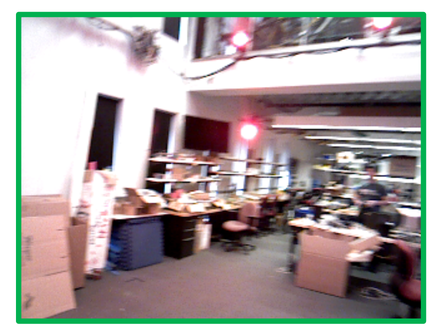
M-step

Update R and t

Stop if converged

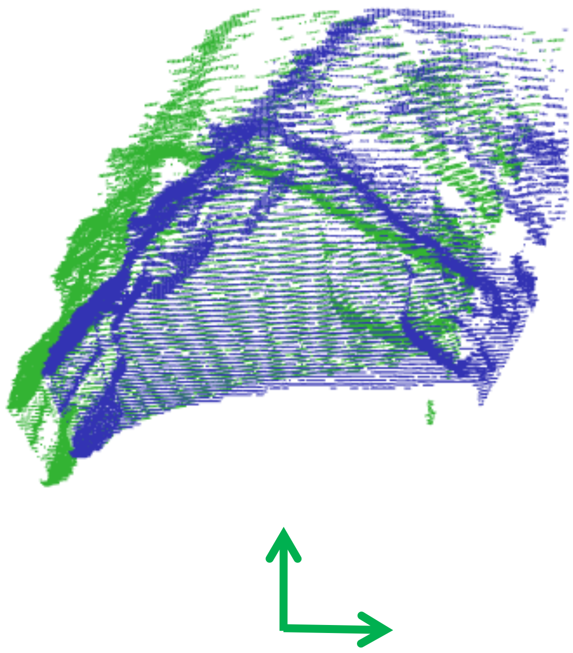






ICP: Motion Increment

Raw measurements are
in the local coordinate frame.



Registration gives the motion
increment of the body w.r.t the model