ECE276A: Sensing & Estimation in Robotics
Lecture 11: Simultaneous Localization and Mapping using a Particle Filter

Lecturer:

Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistants:

Siwei Guo: s9guo@eng.ucsd.edu

Anwesan Pal: a2pal@eng.ucsd.edu

UC San Diego

JACOBS SCHOOL OF ENGINEERING

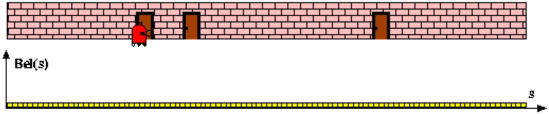
Electrical and Computer Engineering

Markov Localization

- **Robot Localization Problem**: Given a map m, a sequence of control inputs $u_{0:t-1}$, and a sequence of measurements $z_{0:t}$, infer the state (e.g., pose) of the robot x_t
- **Approach**: use a Bayes filter with a multi-modal distribution in order to cature multiple hypotheses about the robot state, e.g.:
  - Gaussian mixture filter
  - Particle filter
  - Histogram filter
- **Pruning**: need to keep the number of hypotheses/components under control
- **Important considerations**: What are the motion and observation models and how is the map represented?

Histogram Filter Localization (1-D)

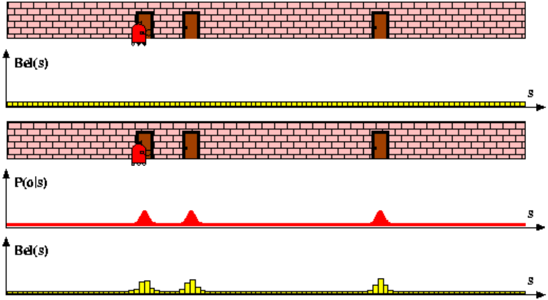
Prior:



Histogram Filter Localization (1-D)

Prior:

Update:

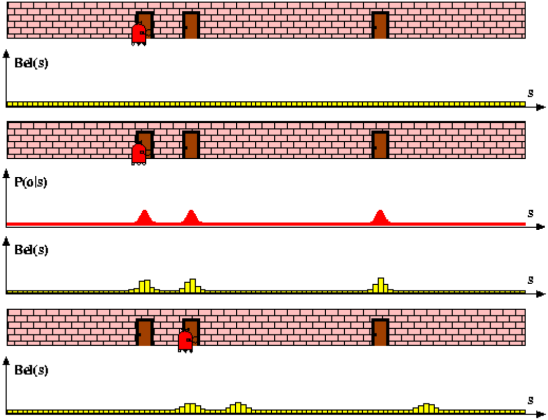


Histogram Filter Localization (1-D)

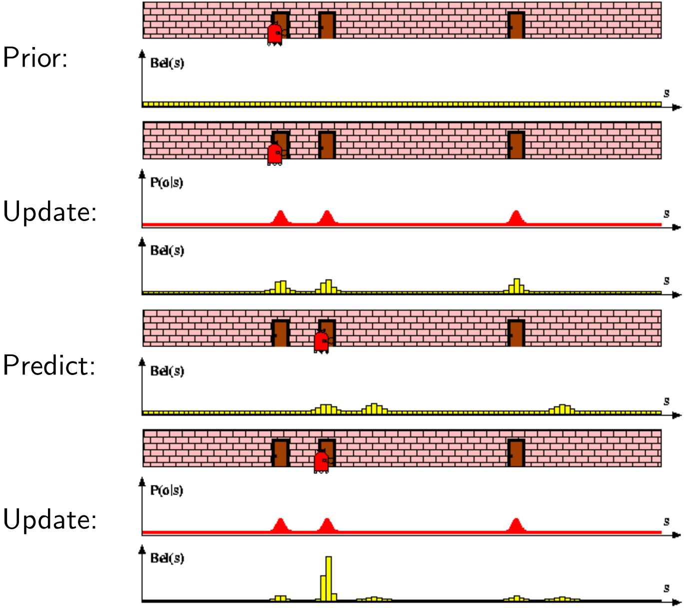
Prior:

Update:

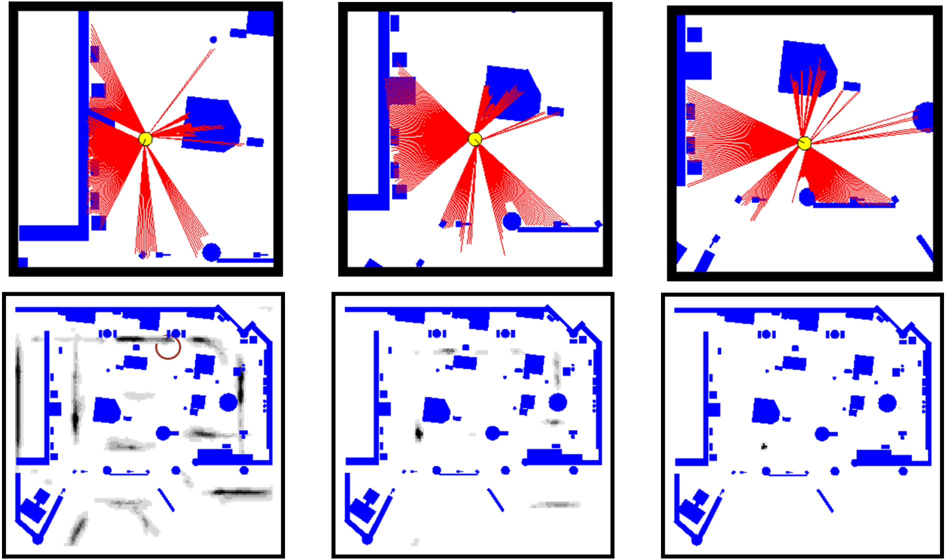
Predict:



Histogram Filter Localization (1-D)

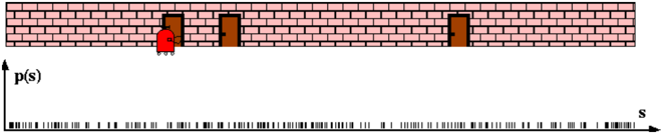


Histogram Filter Localization (2-D)



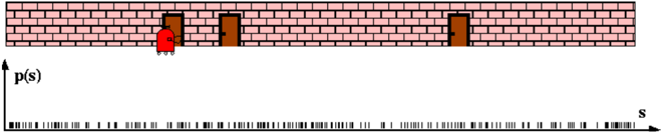
Particle Filter Localization (1-D)

Prior:

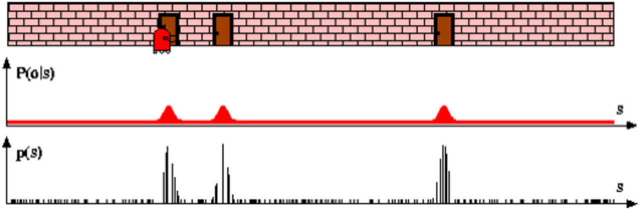


Particle Filter Localization (1-D)

Prior:

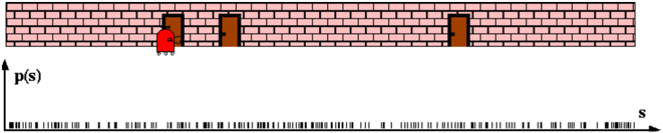


Update:

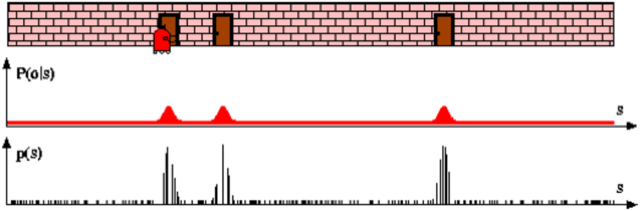


Particle Filter Localization (1-D)

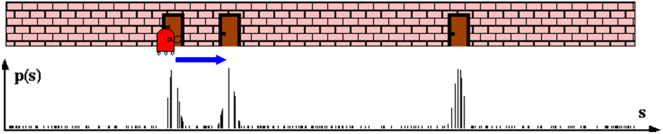
Prior:



Update:



Predict:



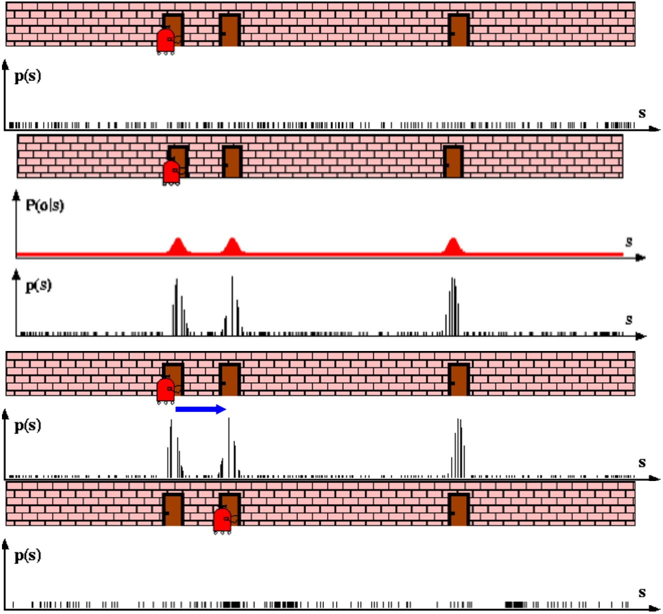
Particle Filter Localization (1-D)

Prior:

Update:

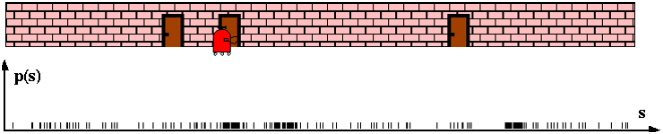
Predict:

Resample:



Particle Filter Localization (1-D)

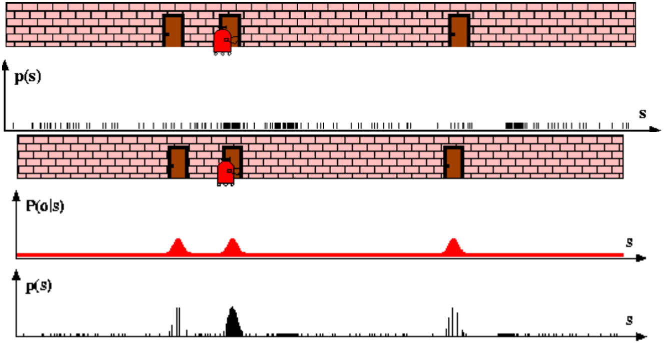
Prior:



Particle Filter Localization (1-D)

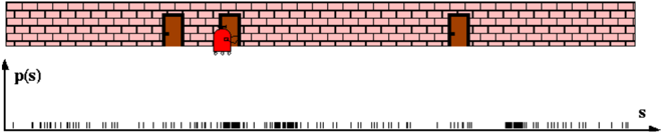
Prior:

Update:

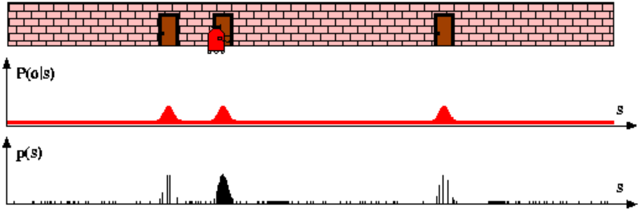


Particle Filter Localization (1-D)

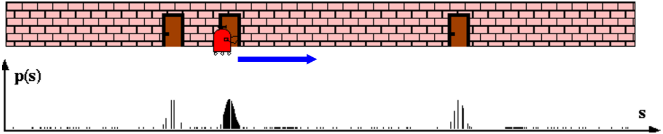
Prior:



Update:



Predict:



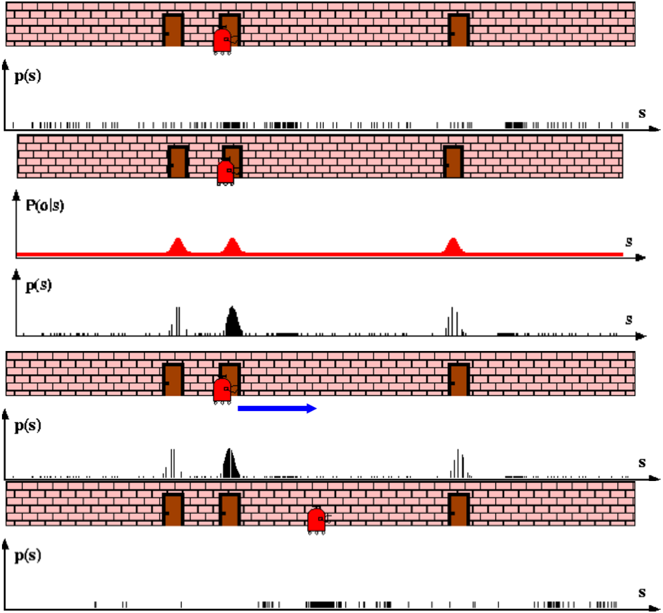
Particle Filter Localization (1-D)

Prior:

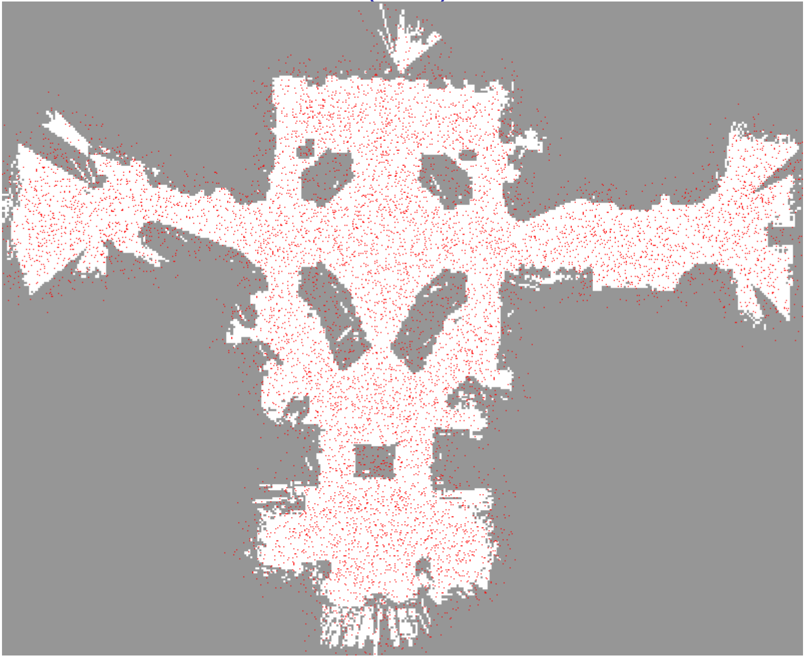
Update:

Predict:

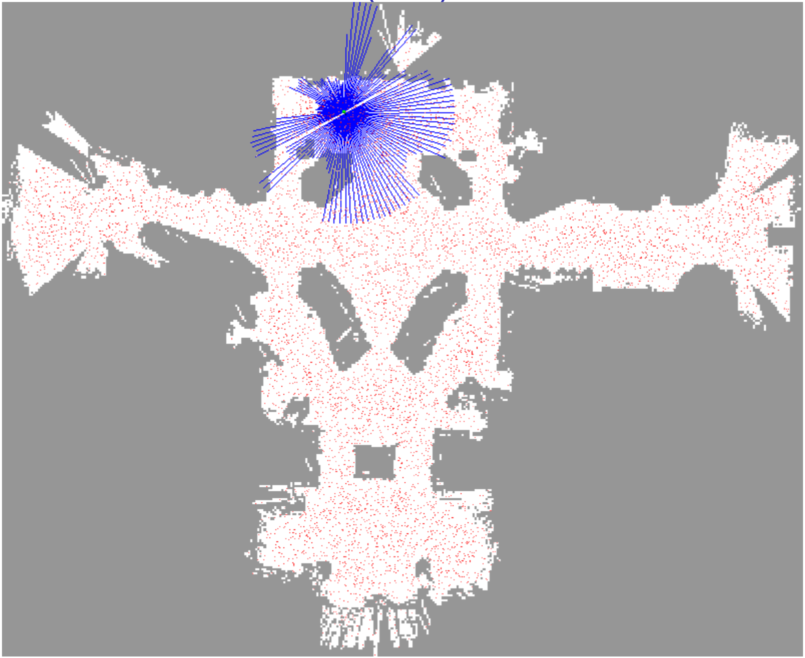
Resample:



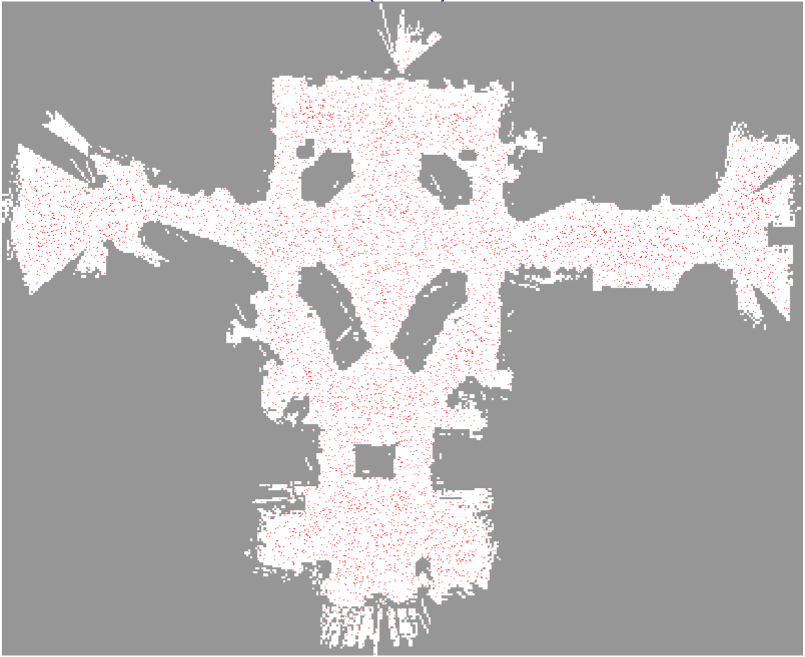
Particle Filter Localization (2-D)



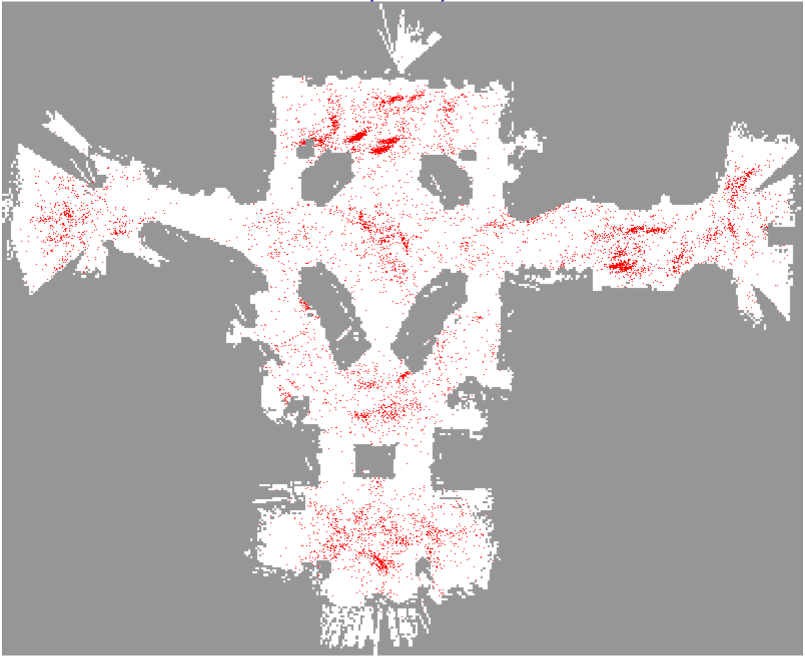
Particle Filter Localization (2-D)



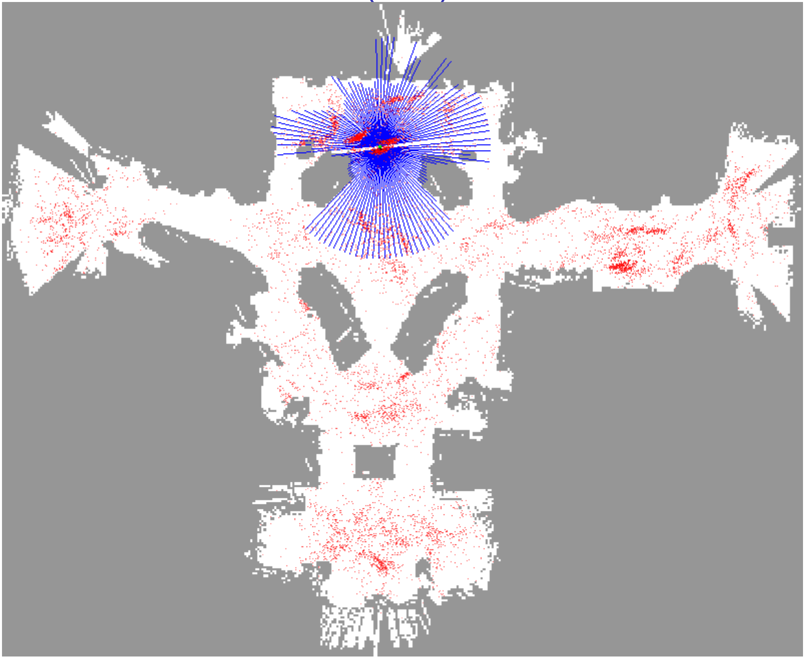
Particle Filter Localization (2-D)



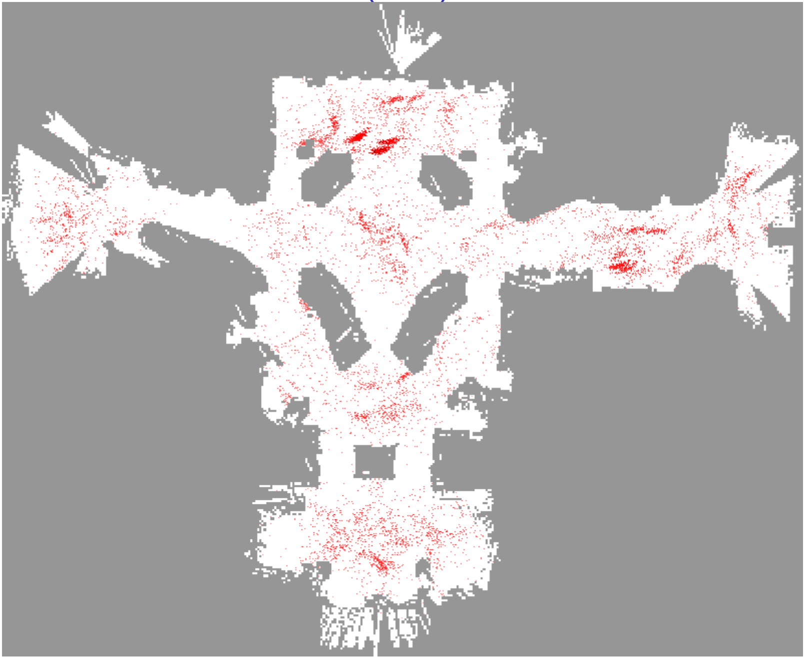
Particle Filter Localization (2-D)



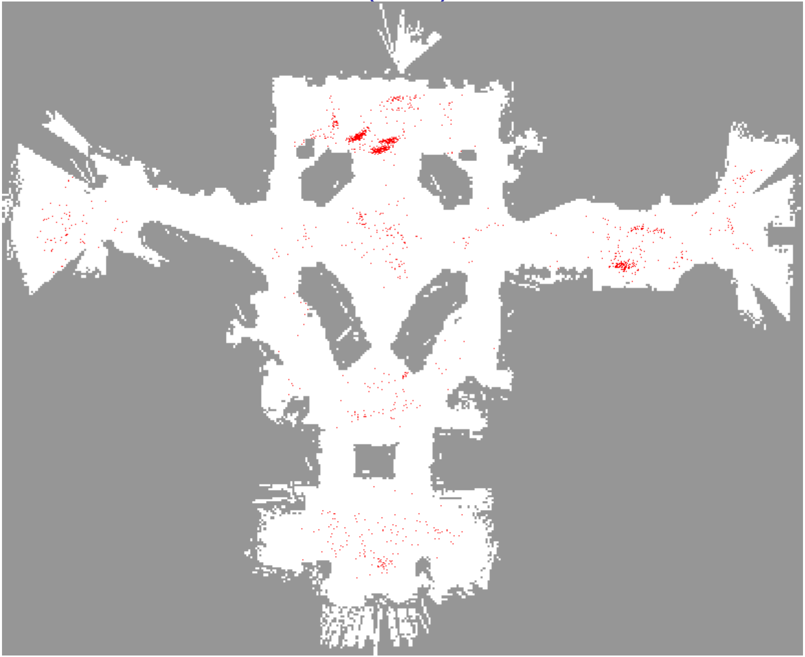
Particle Filter Localization (2-D)



Particle Filter Localization (2-D)

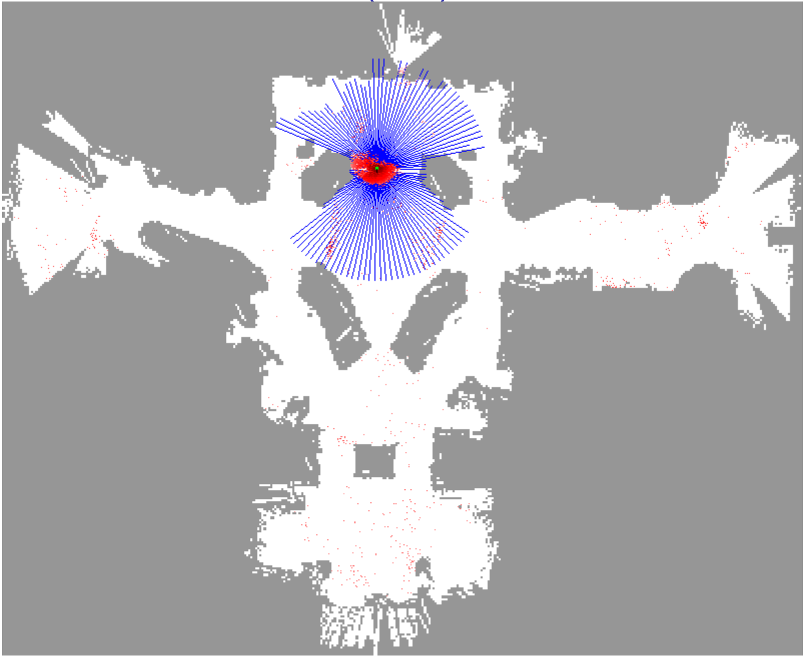


Particle Filter Localization (2-D)



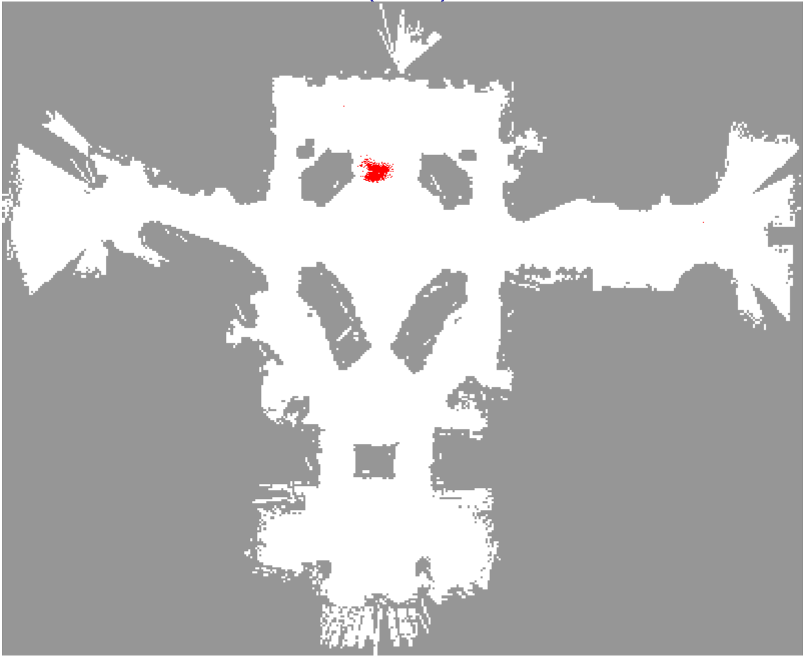
Particle Filter Localization (2-D)

Particle Filter Localization (2-D)

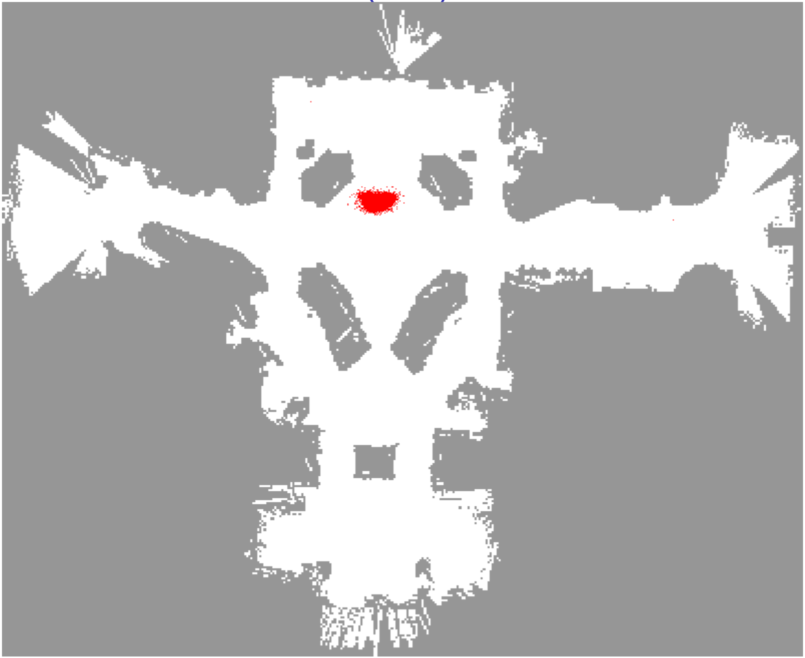


Particle Filter Localization (2-D)

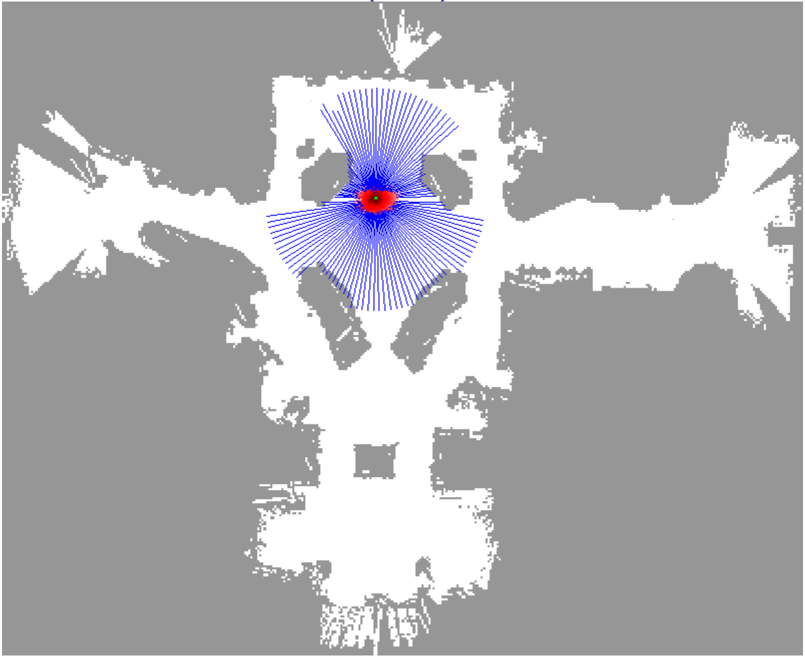
Particle Filter Localization (2-D)



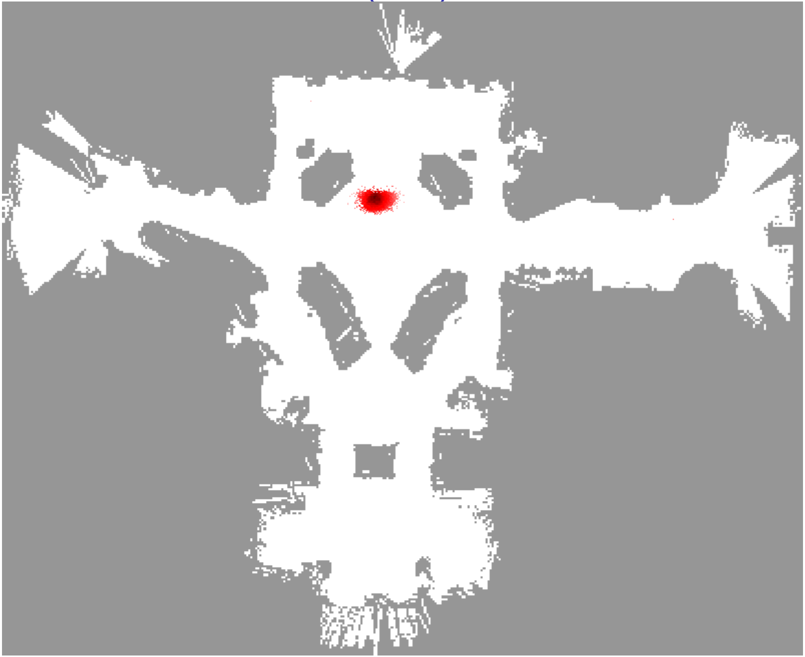
Particle Filter Localization (2-D)



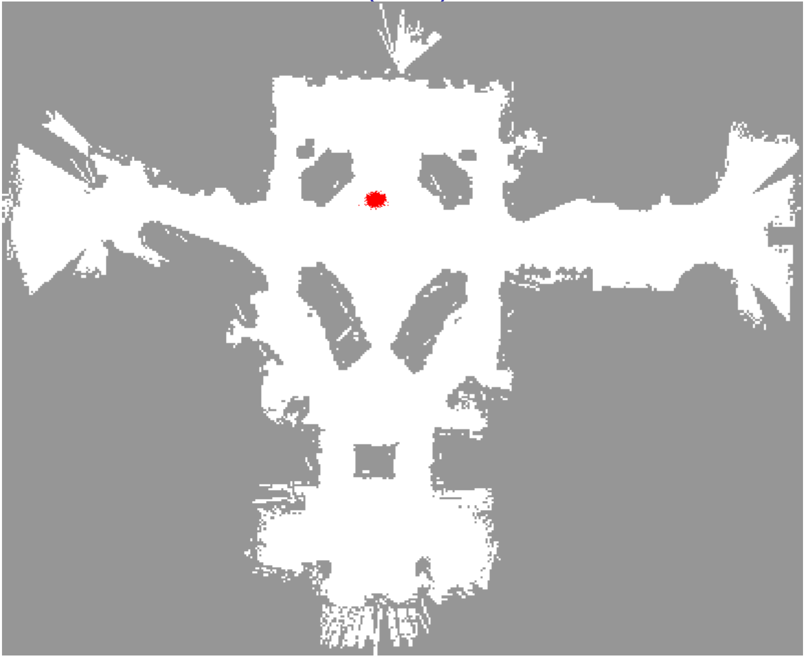
Particle Filter Localization (2-D)



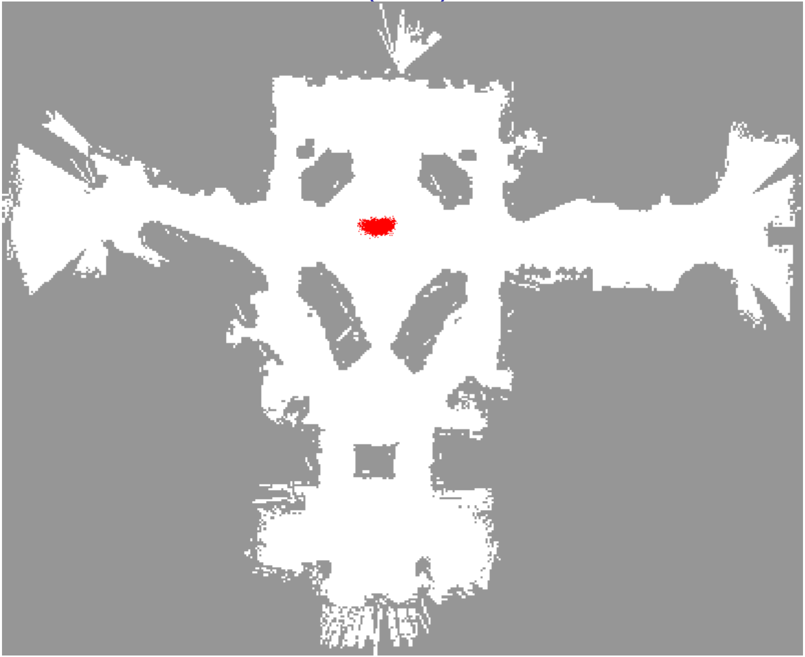
Particle Filter Localization (2-D)



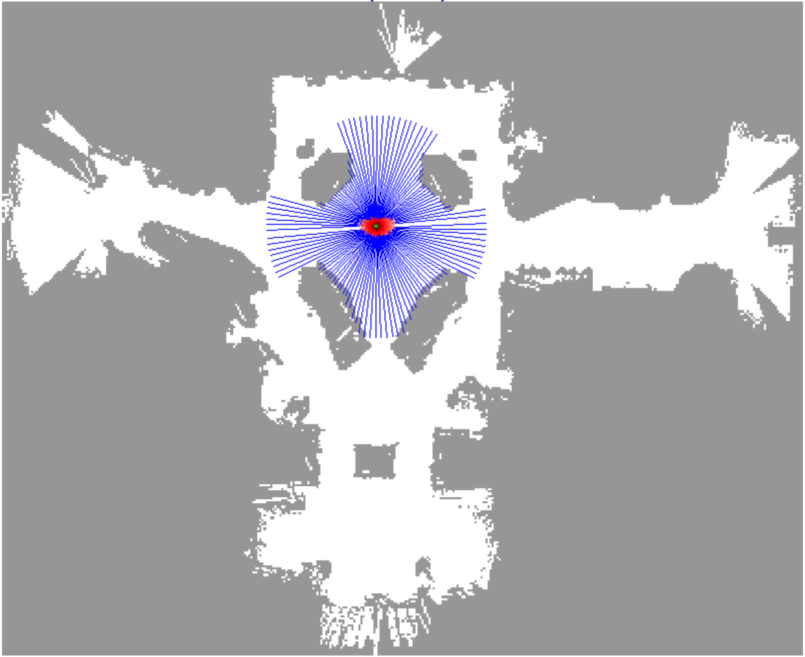
Particle Filter Localization (2-D)



Particle Filter Localization (2-D)

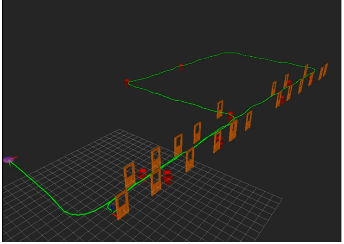


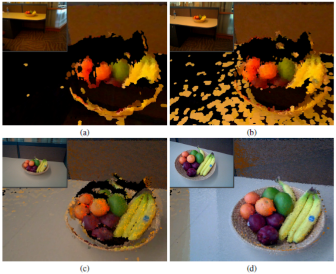
Particle Filter Localization (2-D)

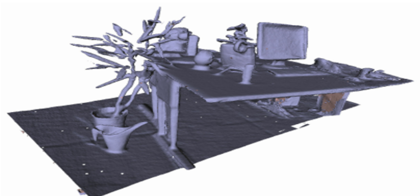


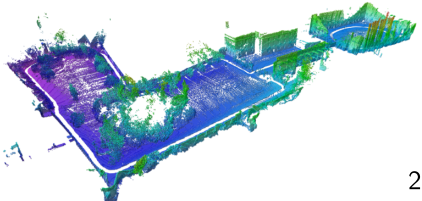
Map Representations

- **Landmark-based**: a collection of objects, each having a position, orientation, and object class
- **Polygonal mesh**: a collection of points and connectivity information among them, forming polygons
- **Surfels**: a collection of oriented discs containing photometric information
- **Occupancy grid**: a discretization of space into cells with a binary occupancy model

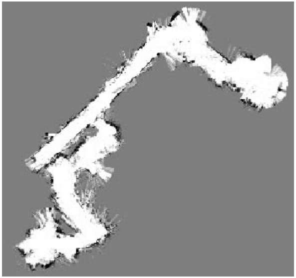








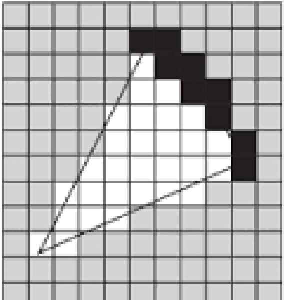
Occupancy Grid



- An **occupancy grid** is a collection $m \in \{0,1\}^d$ of independent Bernoulli random variables m_i for $i = 1, \ldots, d$
- Given occupancy measurements $z_{0:t}$, the distribution of m_i is:

$$m_i \mid z_{0:t} = \begin{cases} \text{Occupied (1)} & \text{with prob. } \gamma_{i,t} := p(m_i = 1 \mid z_{0:t}) \\ \text{Free (0)} & \text{with prob. } 1 - \gamma_{i.t} \end{cases}$$

- How do we update the map distribution over time?



- Estimate the occupancy probability $\gamma_{i,t}$ using the measurements $z_{0:t}$ and the robot state to transform them to the world frame
- Keep track of the cell log-odds:
  $\lambda_{i,t+1} = \lambda_{i,t} + \Delta\lambda_{i,t} \quad \leftarrow$ Measurement "trust"
- Usually constrain $\lambda_{MIN} \leq \lambda_{i,t} \leq \lambda_{MAX}$
- May put a decay on $\lambda_{i,t}$ to handle changing maps

Bayes Rule using Log-Odds

- The **odds ratio** of a binary random variable m_i updated over time via Bayes rule and measurements z is:

$$\left.\begin{aligned} p(m_i = 1 \mid z) &= \frac{1}{\eta} p_h(z \mid m_i = 1) p(m_i = 1) \\ p(m_i = 0 \mid z) &= \frac{1}{\eta} p_h(z \mid m_i = 0) p(m_i = 0) \end{aligned}\right\} \Rightarrow \underbrace{\frac{p(m_i = 1 \mid z)}{p(m_i = 0 \mid z)}}_{o(m_i \mid z)} = \underbrace{\frac{p_h(z \mid m_i = 1)}{p_h(z \mid m_i = 0)}}_{g(z)} \underbrace{\frac{p(m_i = 1)}{p(m_i = 0)}}_{o(m_i)}$$

- A simple observation model, specifying how much we trust occupancy measurements (e.g., from a Laser scanner), can be used. Example:

$$g(1) = \frac{p_h(z \mid m_i = 1)}{p_h(z \mid m_i = 0)} = \frac{80\%}{20\%} = 4 \qquad g(0) = \frac{1}{4}$$

- Estimating the pdf of m_i conditioned on $z_{0:t}$ is equivalent to accumulating the log-odds ratio:

$$\lambda(m_i \mid z_{0:t}) := \log o(m_i \mid z_{0:t}) = \log(g(z_t \mid m_i) o(m_i \mid z_{0:t-1})) = \lambda(m_i) + \sum_{s=0}^{t} \log g(z_s \mid m_i)$$

- Recover pdf from log-odds: $p(m_i = 1 \mid z_{0:t}) = 1 - \frac{1}{1 + \exp(\lambda(m_i \mid z_{0:t}))}$

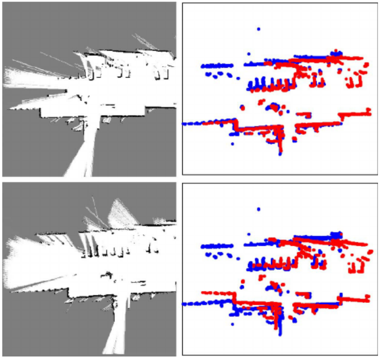
Scan Matching Observation Model

- An observation model for a laser scan z obtained from sensor pose x in an occupancy map m can be obtained by modeling the correlation between z and m as follows:
  1. Transform the scan z to the world frame using x and find all points (or only hit points) y in the grid that correspond to the scan
  2. Let the observation model be proportional to the similarty $\mathbf{corr}(y, m)$ between the transformed scan y and the grid m
- The correlation is large if y and m agree:

$$\mathbf{corr}(y, m) := \sum_i \mathbb{1}\{m_i = y_i\}$$

- The weights can be converted to probabilities via the **softmax** function:

$$p_h(z \mid x, m) = \frac{e^{\mathbf{corr}(y,m)}}{\sum_v e^{\mathbf{corr}(v,m)}} \propto e^{\mathbf{corr}(y,m)}$$



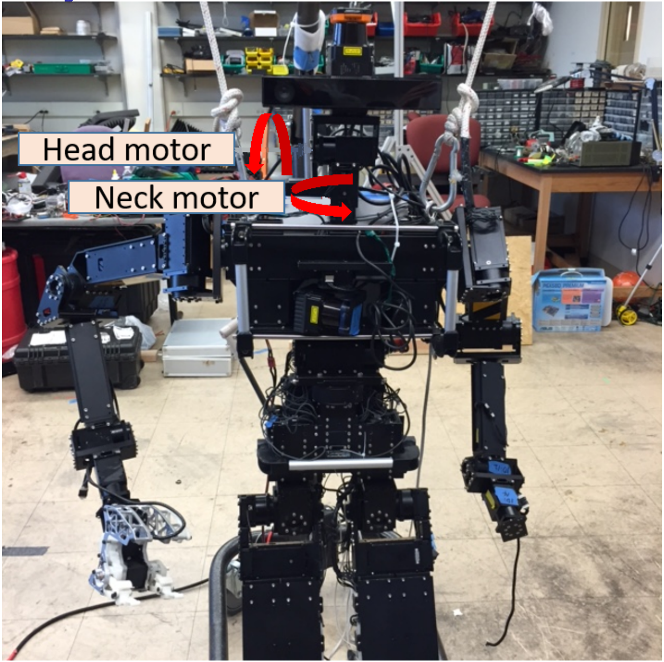
Simultaneous Localization & Mapping (SLAM)

- Chicken-and-egg problem:
  - Given the pose – it is easy to build a map (accumulate log-odds!)
  - Given the map – it is easy to localize (particle filter + scan matching)
- **EM**: suppose x_t is a hidden variable and m are the parameters. Given an inital map $m^{(0)}$, e.g., obtained from the first scan, iterate:
  - E: Estimate the distribution of x_t given $m^{(i)}$
  - M: Update $m^{(i+1)}$ by maximizing (over m) the log-likelihood of the measurements conditioned on x_t and m
- **Filtering**: maintain a joint pdf over the robot state x_t and map m via KF, EKF, UKF: $p(x_t, m \mid z_{0:t}, u_{0:t-1})$
- **Smoothing**: maintain a pdf over the robot trajectory $x_{0:t}$ and map m:
  - Occupancy grid: **Fast SLAM** exploits that the occupancy grid cells are independent conditioned on the robot trajectory:

$$p(x_{0:t}, m \mid z_{0:t}, u_{0:t-1}) = p(x_{0:t} \mid z_{0:t}, u_{0:t-1}) \prod_i p(m_i \mid z_{0:t}, x_{0:t})$$

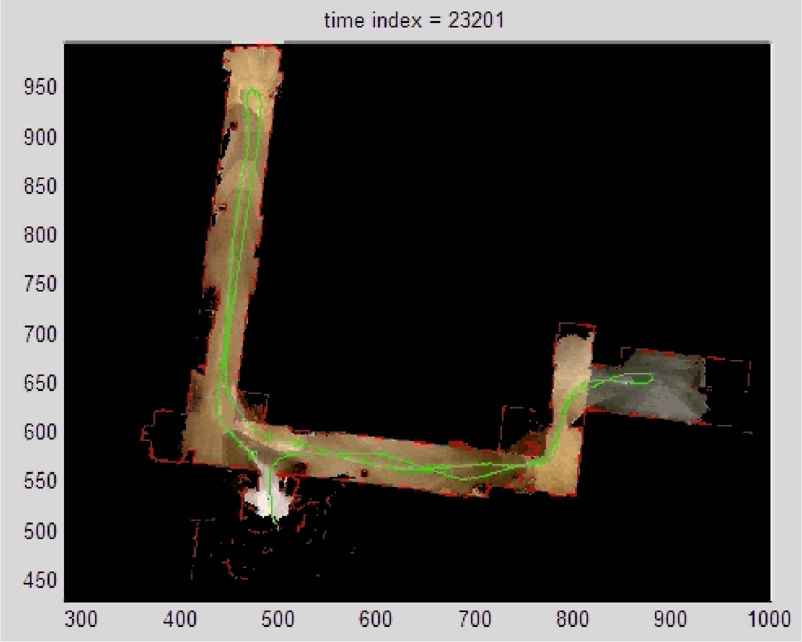
  - Landmark-based: **Rao-Blackwellized Particle Filter** uses particles for $x_{0:t}$ and Gaussian distributions for the landmark poses
  - Landmark-based: **Kalman smoothing** and **Factor graphs** are the state-of-the-art

Project 3: Humanoid THOR



- RGBD camera
- 2D Lidar
- IMU
- Odometry
- Transforms

Project 3: Localization and Textured Map



Project 3: Overview

- Initial particle set $\mu_{0|0}^{(k)} = (0,0,0)^T \in SE(2)$ with weights $\alpha_{0|0}^{(k)} = \frac{1}{N}$
- Use first laser scan to initialize the map:
  1. use head angle to remove the ground plane from the scan
  2. convert the scan to Cartesian coordinates
  3. convert the scan to cells and update the map log-odds (via **bresenham2D** or **cv2.drawContours**)
- Use an **odometry motion model** to predict motion for each particle
- Use the laser scan from each particle to compute map correlation (via **getMapCorrelation**) and update the particle weights
- Choose the best particle, project the laser scan, and update the map log-odds (in general, each particle should maintain its own map)
- **Textured map**: use RGBD images from the best particle pose to assign colors to the occupancy grid cells